TAMPEREEN TEKNILLINEN YLIOPISTO

ANTTI NIEMELÄ

TRAFFIC ANALYSIS FOR INTRUSION DETECTION IN TELECOMMUNICATIONS NETWORKS

Master of Science Thesis

Examiners: Professor Jarmo Harju and senior researcher Marko Helenius

Examiners and topic approved in the Computing and Electrical Engineering faculty council meeting on 8 December 2010

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
NIEMELÄ, ANTTI: Traffic analysis for intrusion detection in telecommunications networks
Master of Science Thesis, 67 pages, 9 Appendix pages
03 2011
Major: Communication networks and protocols
Examiners: Professor Jarmo Harju and senior researcher Marko Helenius
Keywords: Anomaly detection, intrusion detection system, feature extraction, network security.

Threats from the Internet have become more and more sophisticated and are able to bypass the basic security solutions such as firewalls and antivirus scanners. Additional protection is therefore needed to enhance the overall security of the network. One possible solution to improve the security is to add an intrusion detection system (IDS) as an additional layer in the security solutions.

In order for the anomaly detection based IDS to decide what is normal and what is abnormal in the data monitored, it has to have a point of comparison. In the context of networks, this point of comparison is known as a model of normal network traffic. Once the model is created, it is then used as a basis when traffic is monitored.

Feature extraction plays an important role when creating a model of the network traffic. The features should represent the traffic flows as good as possible. The challenge is on finding out the most suitable features for the anomaly detection based IDS, for it to efficiently detect intrusion from the data monitored.

Through analysis of different attacks, it is possible to find out what their effect is to the traffic flows. Common attacks; denial of service, probing and attacks against the services of the network are taken as a basis for the evaluation. After analysing the attacks it was seen that attacks of similar type also have similar effect to the network traffic and thus, subsets of features were formed for each attack type.

The results, however, show that it is clear that more investigation on the differences between operating systems and attacks against them need to be done in order to find out more suitable sets of features. The results also showed that attacks of similar type have different level of effect to the network traffic. Although there were huge differences in the results, they were still more or less according to the expectations. Nevertheless, the results can be thought of as an encouragement, that it is possible to use smaller feature groups to detect specific attack categories with less processing requirements.

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Signaalinkäsittelyn ja tietoliikennetekniikan koulutusohjelma
NIEMELÄ, ANTTI: Tietoliikenneverkon liikenneanalyysi tunkeutumisen havaitsemisjärjestelmälle
Diplomityö, 67 sivua, 9 liitesivua
03 2011
Pääaine: Tietoliikenneverkot ja protokollat
Tarkastajat: Professori Jarmo Harju ja vanh. tutkija Marko Helenius
Avainsanat: Väärinkäytöksen havaitseminen, poikkeavan käytöksen havaitseminen, tunkeutumisen havaitsemisjärjestelmä, verkon tietoturva

Perinteisesti tietoturvaa ovat ylläpitäneet palomuurit ja virustentorjuntaohjelmat. Niiden kyky tunnistaa sekä ehkäistä tietoturvaa rikkovien tahojen toiminta on kuitenkin saavuttanut äärirajansa. Valitettavasti hyökkäysmenetelmät ovat kehittyneet nopeaa vauhtia yhä älykkäämmiksi ja kykenevät läpäisemään nämä perinteiset turvamenetelmät. Yksi mahdollinen ratkaisu ongelmaan on lisätä verkon turvakerroksiin älykkäämpiä menetelmiä, kuten tunkeutumisen havaitsemisjärjestelmiä (engl. intrusion detection system, IDS).

Jotta poikkeavan käyttäytymisen havaitsemiseen perustuva IDS toimisi tehokkaasti teleoperaattoriverkoissa, tulisi tutkia minkälaista tietoa verkkoliikenteestä olisi kerättävä, jotta tuloksekas tunkeutumisyritysten havainnointi olisi mahdollista. Perimmäisenä ajatuksena on etsiä sopivimmat piirteet verkkoliikenteestä, joiden perusteella voidaan luoda mahdollisimman kuvaava malli verkon normaalista toiminnasta ja malliin vertaamalla havaita poikkeamat verkkoliikenteestä.

Palvelunesto- ja verkon urkintahyökkäykset sekä hyökkäykset verkon palveluita kohtaan edustavat tyypillisimpiä uhkia Internetistä. Näitä hyökkäystyyppejä analysoimalla havaittiin kuinka samantyyppisillä hyökkäyksillä on samanlainen vaikutus verkkoliikenteeseen. Näiden hyökkäystyyppien perusteella luotiin piirrejoukot, jotka otettiin vertailun kohteeksi.

Tulosten perusteella on selvää, että tutkimustyötä käyttöjärjestelmien eroista ja niihin kohdistuvista hyökkäyksistä tulee vielä jatkaa, jotta voidaan löytää sopivimmat piirrejoukot. Tulokset osoittavat myös, että samantyyppisten hyökkäysten aiheuttamien vaikutusten välillä on suuria eroja. Vaikka erot eri piirrejoukkojen välillä olivat suuria, saavutettiin niillä kuitenkin lähes odotusten mukaisia tuloksia. Näiden tulosten pohjalta voidaan sanoa, että on mahdollista käyttää pienempiä piirrejoukkoja eri hyökkäystyypeille ja siten suorittaa laskennallisesti kevyempää poikkeamien havainnointia.

FOREWORD

This Master's Thesis has been written as a partial fulfilment for the Master of Science Degree at Tampere University of Technology (TUT). The work was done for Nokia Siemens Networks (NSN) as a part of the Future Internet program of TIVIT (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT).

I wish to thank my supervisors Perttu Halonen and Kimmo Hätönen for providing the opportunity to work in NSN and write this thesis under their guidance.

I would like to express my gratitude to Professor Jarmo Harju and senior researcher Marko Helenius from TUT for the valuable feedback and effort they put in guiding me.

I would also like to thank Pekka Kumpulainen from TUT for his valuable advices. Furthermore, I wish to thank my colleagues in NSN for cheering me up when I needed it the most. Especially I would like to thank Alexander Zahariev for his valuable comments and advices.

This thesis would not have finished without the love and support from my girlfriend. Thank you Iiris for being there for me through this process.

On 20th of March 2011, in Tampere, Finland

Antti Niemelä
antti.j.niemela@gmail.com

CONTENTS

TERMS AND ABREVIATIONS

2G	2nd Generation Mobile Communications
3G	3rd Generation Mobile Communications
AAA	Authentication, Authorization and Accounting
AAFID	Autonomous Agents for Intrusion Detection
AD	Anomaly detection
AFRL	Air Force Research Laboratory
ARGUS	Audit Record Generation and Utilisation System
BN	Bayesian Network
BSC	Base Station Controller
BTS	Base Transceiver Station
CART	Classification and Regression Tree
CIDF	Common Intrusion Detection Framework
CIDS	Central Intrusion Detection System
CPU	Central Processing Unit
CSP	Communications Service Provider
CSV	Comma Separated Value
DARPA	Defense Advanced Research Projects Agency
DNS	Domain Name System
DoS	Denial of Service
DPI	Deep Packet Inspection
DR	Detection Rate
D-SCIDS	Distributed Soft Computing Intrusion Detection System
EMERALD	Event Monitoring Enabling Responses to Anomalous Live Disturbances
EMP	Electromagnetic Pulse
ePDG	Evolved Packet Data Gateway
Feature	Synonym to variable, descriptor and parameter used in IDS.
GPRS	General Packet Radio Service
GrIDS	Graph-based Intrusion Detection System
HIDS	Host based intrusion detection system
HLR	Home Location Register
HRPD	High Rate Packet Data
HSGW	HRPD Serving Gateway
HSS	Home Subscriber Server
ICMP	Internet Control Message Protocol
ID	Intrusion Detection
IDS	Intrusion Detection System
IMS	IP Multimedia Subsystem
IP	Internet Protocol

IP packet quintuple	IP packet destination and source address and port together with protocol identifier forms the IP packet quintuple
IPS	Intrusion Prevention System
LTE	Long Term Evolution
MME	Mobility Management Entity
NIDES	Next-Generation Intrusion Detection Expert System
NIDS	Network based Intrusion Detection System
P-GW	Packet Data Network Gateway
PCA	Principal Component Analysis
PCRF	Policy and Charging Rules Function
PSO	Particle Swarm Optimization
RAN	Radio Access Network
RNC	Radio Network Controller
RST	Rough Set Theory
R2U	Root to User
S-GW	Serving Gateway
SGSN	Serving GPRS Support Node
SNMP MIB	Simple Network Management Protocol Management Information Base
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TRCNetAD	Telecommunication Research Center Network Anomaly Detection
U2R	User to Root is an attack category in the Lincoln laboratory dataset
VoIP	Voice over IP
VLR	Visitor Location Register
VPN	Virtual Private Network
WLAN	Wireless Local Area Network
xDSL	Digital Subscriber Line, the letter x before the abbreviation means all the DSL technologies such as ADSL, ADSL2 etc.

1 INTRODUCTION

In the 21st century the development of telecommunications networks has taken giant leaps from circuit and packet switched networks towards all-IP based networks. This development has created a unified environment where communication of applications and services (data and voice) are being transferred on top of the IP-protocol.

At the same time the data transmission speeds in both uplink and downlink have increased significantly from the second generation (2G) of radio access networks to the third generation (3G) of radio access networks. Also the devices that subscribers of telecommunications networks are using have been developing and the boundary between computers and mobile phones has become unclear. With the modern mobile devices, also known as smart phones, the subscriber can do almost everything that can be done with basic personal computers. This means that the full content of the Internet is now also in the pockets of each smart phone owners.

Although the development of communication networks has been towards a better sustainability of technologies it has also raised new unwanted possibilities. Threats that were applicable only in the fixed networks are now feasible in the radio access networks. When taken into account that threats are becoming more and more sophisticated it also means that the security systems have to become more intelligent. The basic security measurements such as firewalls and antivirus scanners are in their limits to cope with the overgrowing number of intelligent attacks from the Internet. A solution to enhance the overall security of the networks is to increase the security layers with intrusion detection systems.

To understand what role intrusion detection has in telecommunications networks it can be thought through a simple example. Think of intrusion detection as a security guard that is guarding the front gate of a factory premises. The premises of the factory represent the network of a mobile operator and the fence surrounding the factory is the operator's firewall. Employees of the factory represent the traffic in the operator's network.

It is know that factories are well protected and they do not want to let people inside the premises that do not have the required clearances. The fence or firewall in this case, is in charge to keep all unwanted visitors outside the factory premises. Just like in a firewall, a fence has holes (gates) in it to let employees move in and out of the factory premises. These holes in the fence though leave the factory vulnerable to the unwanted visitors and this is why the factory has a security guard guarding the gate.

Depending on the role that the security guard is in, while he is monitoring the people going in and out of the factory premises, he either notifies the head of security when he detects a suspicious looking person walking through the gate. Or he steps in and prevents this person from entering the factory premises.

The basic functionality of an intrusion detection system is the first example of the security guard. IDS generate an alarm when it detects something suspicious and then the security personnel of the network operator further investigate the cause of the alarm.

In order for the security guard to do his job well, a set of rules and instructions are needed. In the context of IDS in telecommunication networks the rules and instructions are algorithms that IDS uses to analyse network traffic. The question is: "How should these rules and instructions be defined and, especially, what are the criteria to decide what features should be monitored?"

This thesis takes various approaches to answer to the question how the features should be selected from the network traffic so that the intrusion detection system can efficiently detect threats in the environment of telecommunications networks.

The rest of the thesis is organized in the following manner. Chapter 2 introduces the basics of intrusion detection systems and how it fits into the environment of telecommunications networks. In addition, a discussion on the prior art of research on the field of intrusion detection is given in this chapter.

Chapter 3 gives an overview of feature extraction methods for intrusion detection systems and what challenges the environment sets to the extraction. In addition the features used in the research field of network based intrusion detection systems are discussed in the end of this chapter.

Chapter 4 discusses the approaches to the feature extraction used in this thesis. The results of these approaches are summarised in the end of this chapter.

Chapter 5 describes the evaluation process of feature subsets and the data that is used as a basis in the evaluation. The results of the feature performance analysis are discussed in Chapter 6. Conclusions are presented in Chapter 7.

2 INTRUSION DETECTION IN TELECOMMUNICATIONS NETWORKS

Introduction to telecommunications networks and intrusion detection systems' role in it is discussed in this chapter. In addition a brief overview of features that were used in prior research of IDSes is presented.

2.1 Telecommunications Networks

Development in telecommunications networks has been going towards mobility with radio access networks. For example, in Finland, many operators have been pulling up their copper wires in rural areas and are replacing digital subscriber lines (xDSL) connections to 3G subscriptions. According to news articles reported in HS.fi [1] and Tietokone.fi [2] TeliaSonera announced its' plans on pulling up the copper wires.

In a way the development or some may say non-development of networks has been from local area networks (LAN) towards radio access networks (RAN) for its' easier and cheaper set-up in rural areas where the density of network infrastructure is not sufficient. Despite the fact that in Finland there has been a discussion [3] about developing a country wide optical fibre network, for the time being the only option for many is still to use RAN connections.

2.1.1 Infrastructure of Telecommunications Networks

From the subscriber's point of view it might look like the infrastructure of telecommunications networks consists only from a group of radio towers that are scattered all around the cities and rural areas. In reality the underlying infrastructure of the network is a far more complex thing than just the base stations and radio interfaces.

Telecommunications networks have a lot in common with enterprise networks. In enterprise networks there are hundreds of computers and users connected together with routers, switches and interconnected subnets. In telecommunications networks there are the same elements as in enterprise networks but in addition there are also multiple radio access networks (RAN) from GSM to LTE and a huge amount of fixed and mobile users. The infrastructure of telecommunications networks can be divided into three sub-networks; access network, core network and service network. This division is illustrated in Figure 2.1.

Access Networks

The part of the network that connects and gives access to subscribers to their service provider is called access network (see Figure 2.1). Access networks can be further divided into fixed line access networks (Ethernet, xDSL, and Cable) and into radio access networks (2G, 3G, LTE, CDMA and WLAN). Another term used in telecommunications networks is a subscriber network. The subscriber network is a combination of the access network together with subscriber's user equipments such as mobile phones, laptops etc. [4]

Radio access networks have been evolving towards all IP based networks but at the same time older radio techniques has to be supported. According to global GSM incremental market analysis [5] done by ZTE, in 2010 the GSM and 2G are still the most commonly used technique to use calling and data services globally. Over 80% of global mobile subscribers use only GSM accounts while 3G and CDMA share the rest 20%. This is why in radio access networks there are still different base stations; base transceiver station (BTS) for 2G, Node B (different name for BTS) for 3G and evolved Node B (eNode B) for LTE. Different radio techniques require different controllers; base station controller (BSC) for 2G and radio network controller (RNC) for 3G. In LTE and CDMA all the mobility management operations are handled by mobility management entity (MME) in the core network. [4, pp. 44-48]

Evolved Packet Core Network

The intermediate network that connects access networks to service networks is called evolved packet core network (see Figure 2.1). In addition to operating as an intermediate, core network is responsible for circuit-switching and packet-switching operations, subscriber charging, AAA services and subscriber's mobility management services. [4, pp. 44-48]

Because of the wide variety of access networks the core network has evolved into a complex environment. The core network has to support older radio access techniques where voice and data is separated between packet-switched and circuit-switched networks (2G, 3G) and at the same time it has to provide services for the newer radio access networks (LTE) where voice and data is not separated anymore. [4, pp. 44-48]

In a 2G network the packet-switching operations for data transmissions and circuit-switching operations for calls are provided by serving GPRS support node (SGSN). In evolved packet core, the 3G is using SGSN only for the circuit-switching operations. The data transmissions in 3G are handled by serving gateway (S-GW) together with a packet data network gateway (P-GW). In LTE voice and data is not separated anymore and therefore all the packet data operations are handled by S-GW together with P-GW. High rate packet data serving gateway (HSGW) is providing voice and data operations for the CDMA radio networks. [6, p. 156] Evolved packet data gateway (ePDG) is providing packet data operations for the WLAN. [6, pp. 24-29]

In addition to these previously mentioned elements in the core network there are also home and visitor location registers (HLR/VLR) for the subscriber mobility management, AAA for the subscriber authentication, authorization and accounting functions, charging for the subscriber billing services, policy and charging rules function (PCRF) for quality of service and charging related policies. [4, pp. 44-48]

Service Network

Service network provides services like connection to the Internet. Service network is also responsible for providing access to company intranets and operator specific services. In addition to these it also provides access to IP multimedia subsystem (IMS) for multimedia and voice applications such as VoIP. [4, pp. 44-48]

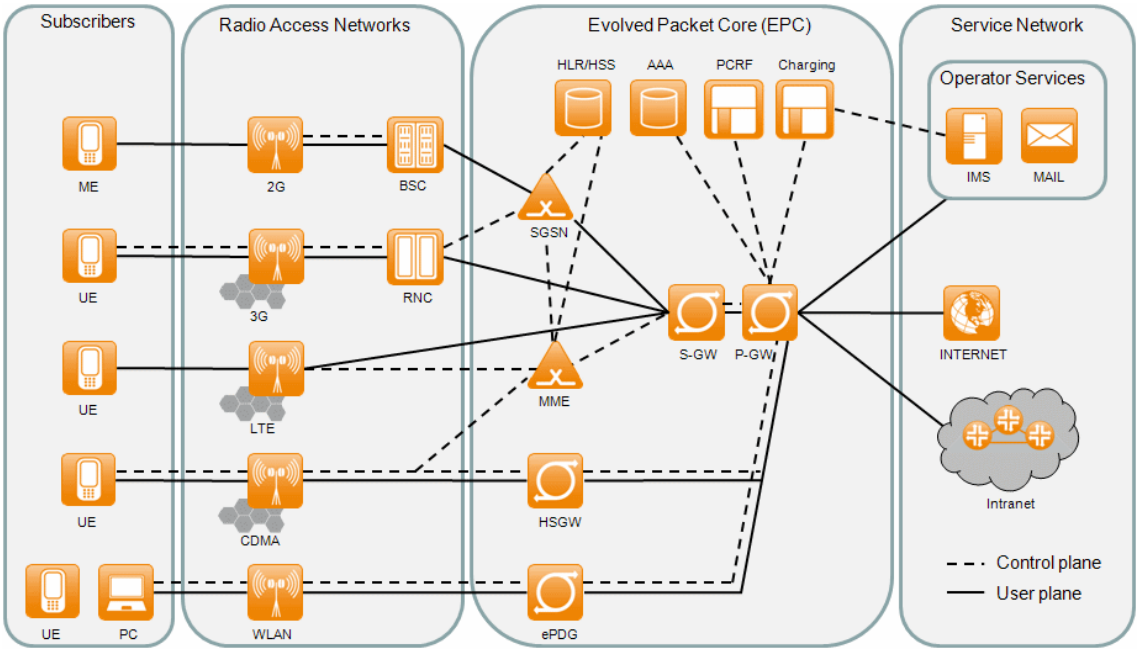

Figure 2.1 Telecommunications networks' infrastructure [6, p. 17; 156]

2.1.2 Threats against Telecommunications Networks

According to CERT [7] the attack sophistication has increased during the past 30 years while at the same time the intruder knowledge has been coming down. This development is illustrated in Figure 2.2. Reason for the increasing sophistication of attacks can be explained with the fact that the use of Internet has become more common and the security solutions protecting the Internet users have become more intelligent. Of course the computers and operation systems have also become more secure. In order to penetrate intelligent security measures the attacks have to be intelligent also. The downward trend in intruder knowledge can be explained by the wide availability of freely distributed applications that can be used to perform attacks. In most cases the user of this kind of an application does not even know what he or she is doing.

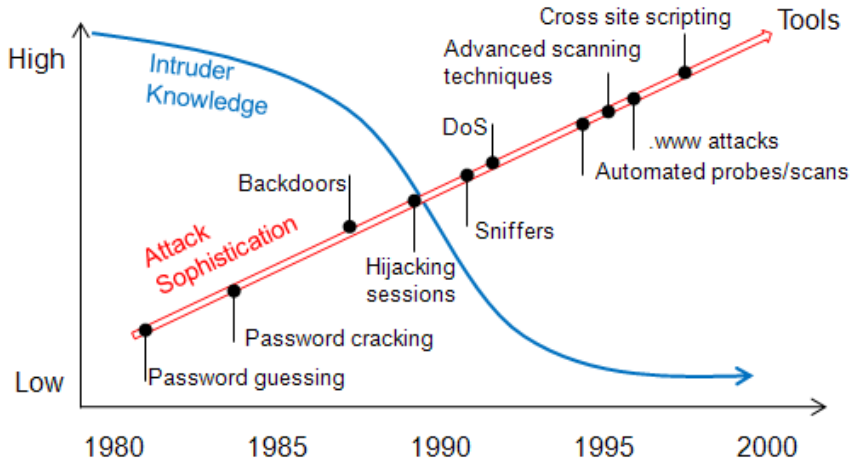

Figure 2.2 Attack sophistication vs. intruder knowledge according to Carnegie Mellon University [7]

Another security concern in modern communication networks is that the networks are vulnerable to threats despite the fact that they might not even be connected to the Internet. One example of this concern was confirmed in July 2009 when Internet worm Stuxnet was discovered. Stuxnet was targeting particular process control systems, especially industrial installations, such as uranium enrichment plants. What makes Stuxnet so efficient is its self-replicating functionality. Stuxnet can replicate itself into USB devices and network shares and then further spread into networks that are not directly or not at all connected to the Internet. [8]

As the development of telecommunications networks has been toward all-IP-based networks and services, it has also created new possibilities for malicious entities to perform illegal activities. In addition the attacks that were applicable only for fixed connections can now be used against mobile connections as well. Some of the typical types of threats are discussed in the following paragraphs.

Reconnaissance

Attacks whose goal is to map the network services, used and open ports, operating systems in use etc. are called reconnaissance attacks. Reconnaissance can be divided into two groups; to the ones that come from the outside of the network (external reconnaissance) and to the ones that come from within the network (internal reconnaissance).

In external reconnaissance the attacker tries to gain information about the operator's network infrastructure and to find security vulnerabilities that could be later used as a medium to get inside the network. In internal reconnaissance the attacker has access to the internal network infrastructure either legitimately or illegitimately. The attacker could use the same methods as in external reconnaissance to map the network infrastructure from inside the network. In addition the attacker could access network elements and computers with privileged rights and steal confidential information from databases and information banks that holds knowledge about the network infrastructure.

Denial of Service attacks

DoS attacks are trying to deny or limit the subscribers or operators use of services. This can be achieved by exhausting all the resources (CPU, memory and bandwidth) of the targeted subscriber's user equipment to prevent it from using his or her device. This can be achieved by publishing the targets phone number or IP address on public forums or in other media that could cause a huge amount of people trying to access the target at the same time.

According to Cisco the mobile data traffic will double every year to 2014. This will set a huge pressure on CPS's network operation and service quality as the amount of traffic and the number of mobile subscribers keeps on increasing at the same time. [9] The increasing amount of network traffic might as well cause similar situations as in DoS by overloading the network infrastructure.

Malicious Content

The amount of malicious web sites poses a significant threat to UEs when the UEs are getting more and more similar features as desktop computers. For example, the current Linux phones have desktop computer's performance and applications running on it. At the same time the mobile web browsers are supporting java, flash and other media players to display the webpage content as it is displayed with desktop computers. This also means that the same threats that might cause damage to desktop computers are also applicable with UEs.

According to McAfee lab's 2010 third quarter threats report [8] the amount of new malware Internet sites are constantly increasing. For example, in September 2010 the amount of new malware sites fluctuated from a few hundred to more than four thousand per day. The same figures are valid also with the amount of new phishing sites per day. [8]

Malware attacks

In this scenario the infrastructure of telecommunications networks is targeted with a sophisticated worm that has self replicating functionality to spread even further among network elements.

Stuxnet [8] is an example of this kind of worm that spreads through USB-devices and Internet shares towards a specific target. In stuxnet's case the worm is targeting specifically industrial controlling machines, especially in a certain country.

The attacker could modify Stuxnet in such way that instead of targeting process control systems it would attack against network management elements. In the worst case scenario this kind of a threat could lead into a critical failure in an operator or in every operator's networks. At worst this would mean that all communications would be denied for the subscribers locally or even globally.

With a worm like Stuxnet is could be possible to sabotage the entire communication network of a country. As can be seen from Figure 2.3 a targeted attack can be very precise but at the same time it can spread widely. Figure 2.3 is a representation from

McAfee Global Threat Intelligence Stuxnet map [8]. The circles illustrate the amount of Stuxnet infections in that area. The bigger the circle the more infections there are in that area. Although in the case of Stuxnet it is believed that its main target was in Iran but today India is suffering the most [8].

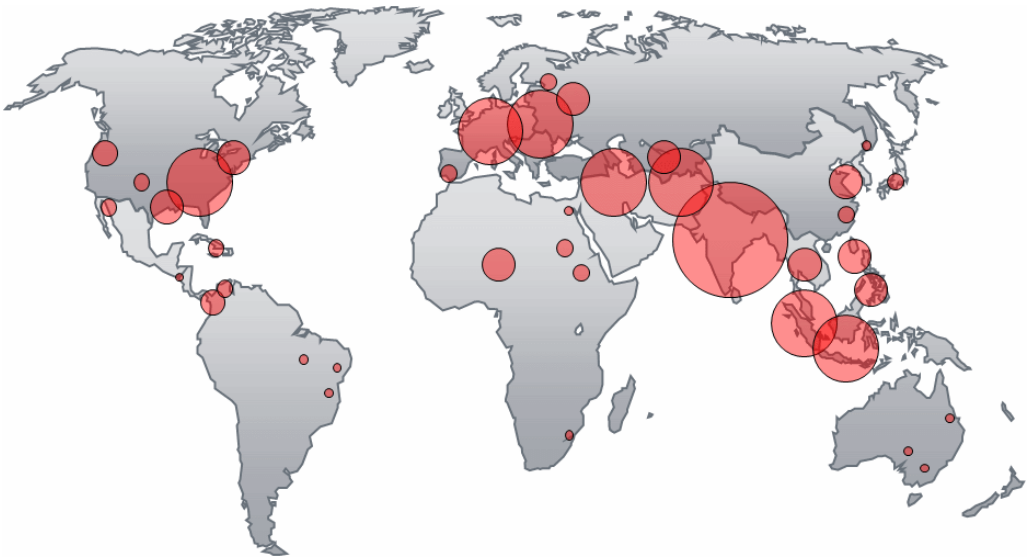

Figure 2.3 Stuxnet infections according to McAfee Global Threat Intelligence [8]

2.2 Intrusion Detection Systems

Intrusion detection system (IDS) can be software or hardware that monitors for intrusions and anomalies from the environment it is set to guard. In general the IDS is a security monitoring tool like a firewall that tries to detect and possibly prevent malicious activity.

Two main techniques for intrusion detection exist based on what they can detect. These two techniques are misuse detection and anomaly detection. Misuse detection and anomaly detection systems can be further divided into two groups based on the detection method; into behaviour based and into knowledge based IDS. Behaviour based IDS monitor behaviour deviations of the system in order to detect intrusions and anomalies. Knowledge based IDS monitors a system using patterns of known intrusions. [10]

Basic functionality of IDS is to act as a passive alerting system. This means that once intrusion is detected the IDS generates an alarm and provides all the relevant information (time, IP packets, etc.) that triggered the alarm. IDS that operates in active mode, reacts to detected intrusions by using countermeasures to prevent the access of the intrusive data accessing the system. Active IDSes are called intrusion prevention systems (IPS). For example, IPS can alter the firewall rules, change routing tables, limit network bandwidth or just disconnect the connection. IDSes can be further divided into two systems depending on where the IDS is placed. The IDS can be either a Network based IDS (NIDS) or Host based IDS (HIDS). Network intrusion detection system

monitors for intrusions in network traffic and host intrusion detection system monitors the behaviour of a local machine. [10]

2.2.1 Intrusions and Anomalies

In the same context of IDS, words intrusions and anomalies are commonly used. The term intrusion is a bit confusing as the system that tries to detect intrusions is also a general term for the system that tries to find anomalies.

From a security point of view, intrusion is a malicious activity against the confidentiality, integrity or availability of information. An anomaly is a deviation from what is thought of as normal. [10] The difference between an anomaly and intrusion is somewhat depending on the environment. For example, intrusion is always more or less a deviation from normal behaviour. But on the other hand an anomaly is not always an intrusion. For example, a failure on a network element might cause abnormal activity in the network but it is not an intrusion. In this document the word IDS is therefore used to describe a system that can be used to detect both, intrusive and anomalous behaviour.

2.2.2 Architecture of IDS

Intrusion detection systems are constructed from three components; sensors, analyser and user interface. Sensors are collecting data such as network traffic, log files and system trace files. Once the data is collected it is then forwarded to the analyser. Analysers or detection engines are responsible for determining if there was an intrusion among the data. After an intrusion is detected the analyser's output is either an alarm or action. The sensor and analyser can be a single system or they can be separated into individual components depending on how the IDS is constructed. For example, one analyser might get traffic data from multiple sensors or the sensor might be embedded into the analyser. The user interface provides the means for the administrator to monitor the output of an analyser and configure analyser and sensor operations. The general architecture of intrusion detection system is illustrated in Figure 2.4. If the IDS is an reactive type, the components can also conduct an action when intrusion is detected, to prevent any further damage to the system. These preventions are illustrated as action arrows in the Figure 2.4.

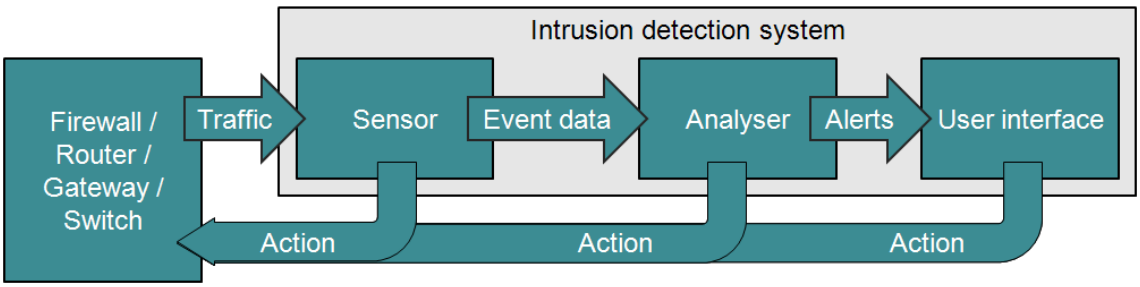

Figure 2.4 IDS architecture

2.2.3 IDS in Layered Defence In-Depth Strategy

Usually the IDS is working behind firewalls in order to detect intrusions that firewalls have missed. The IDS or IDS sensor may as well be placed before the firewall in order to collect illegal traffic data that would otherwise be rejected by the firewall. In some cases it is useful to collect such information in order to recognise and to know when the network is being targeted. In general IDS gives an extra protection layer to the defence in-depth strategy [11]. An example of the defence in-depth strategy is illustrated in Figure 2.5 where on the left are some of the possible threats from the Internet endangering the overall security of the telecommunications networks operation.

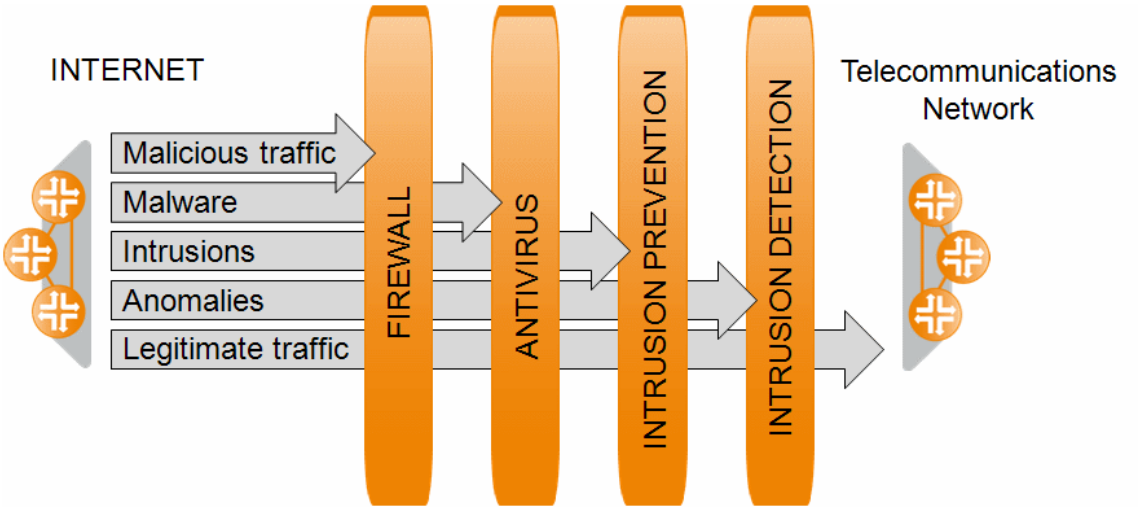

Figure 2.5 Layered defence in-depth strategy

The basic idea in layered defence in-depth strategy is to enhance the overall security of the protected system. This can be achieved by adding multiple security measurements and improve security awareness on all levels from people to operations. In each of the layers some parts of the traffic that might be malicious are detected and prevented from accessing the targeted network. [11]

In the context of telecommunications networks this means that the overall security can be enhanced by adding IDS and possibly IPS functionality into strategic places, like for example, in outer gateways. IDS's role in this defence in-depth strategy is to detect possible threats that have passed firewall rules and antivirus scanners.

2.2.4 False positives and False negatives

In order to evaluate the IDS's performance and detection accuracy there are four possible occurrences whose ratio is monitored. These occurrences are illustrated in Figure 2.6.

False positives are legal occurrences that are incorrectly marked as anomalous. True positives are occurrences that are correctly marked as anomalous. False negatives are anomalous occurrences that are missed by the detector and therefore are not marked as

anomalous. True negatives are occurrences that are correctly marked as legal activity. In order to find out whether the anomaly or intrusion is a false positive or false negative, it has to be investigated by a network operator. [10] These occurrences are illustrated in Figure 2.6. where the vertical axis presents how activity is detected by IDS and the horizontal axis show what the activity actually is.

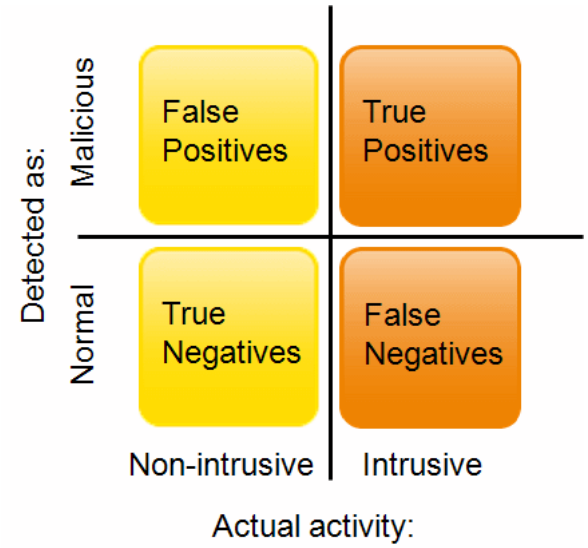

Figure 2.6 False positives, true positives, false negatives and true negatives

From the network operator's point of view the most risky situations are the false negatives. Possibly intrusive activity that passes through IDS as normal and legitimate activity might harm the whole network and therefore it would affect every subscriber using the network. False positives are not as harmful as false negatives because it will affect only one subscriber. Of course from the subscribers' point of view this would seem like bad service when the subscriber's access to the network is denied. In most cases though, user's network activity is not completely denied. [10]

2.2.5 Misuse Detection

Misuse detection can be thought to behave like a virus scanner. Virus scanners are looking for known patterns or signatures of viruses, in the same manner misuse detection is based on known intrusion patterns and signatures. These patterns can be, for example, certain character strings in IP packet contents. In short it can be said that misuse detection deals with known attacks. As such it can be used to analyse network traffic efficiently for known intrusions. [10]

The downside of misuse detection is that it can be avoided by changing the attack pattern slightly so that it will not match the pattern anymore [12]. It is also problematic to write the signatures so precisely that they match to all possible variations of intrusive activities and at the same time avoids matching to non-intrusive activities. Just like virus scanners, IDSes that are based on misuse detection needs to be updated regularly for the latest patterns and signatures. [10]

Most of the available IDSes use misuse detection because it has been studied the most and in a way it is easier to match activities based on known attack patterns rather than finding out whether the ongoing activity is malicious or not just by analysing the activity without previous knowledge of it. This is where misuse detection and anomaly detection differentiate the most. [10]

2.2.6 Anomaly Detection

As misuse detection was based on previously known patterns anomaly detection may detect also something that has not yet been discovered. It is also worth noting that while intrusion detection assumes all the matching activities as malicious, anomaly detection does not assume all anomalies necessarily malicious. [10] It depends on the environment and the rules and regulations whether the detected anomaly is malicious or not.

Network traffic anomaly detection is based on two presumptions. The first presumption is that network traffic has distinguishable characteristics in normal conditions. A model of these normal conditions can be created with parameters. The second presumption is that deviations from this normal model are rare and potentially might be a result of intrusive activity. These two presumptions are according to what is presented in the field literature. [13; 14; 10]

Anomaly Detection as a Process

As a process, anomaly detection can be divided into two phases. In the first phase a model of normal network traffic is created. This model can be derived or learned from training data using model generation algorithms or mathematical models. In the second phase traffic is monitored for deviations from the normal model. [10]

The model of normal network traffic is created by using features from the traffic. Feature in the context of anomaly detection means a value or symbol which describes the network traffic. These features should represent the traffic behaviour and characteristics but in the same time they should not contain any redundant information in order to be as lightweight as possible. In the field literature, the word, feature has numerous synonyms such as variable, parameter and descriptor.

In order to create a model of the normal network traffic, it needs to be clean from malicious activities and at the same time it needs all the variations of the environment it is monitoring. Generating such traffic data is difficult and ready data sets like Lincoln laboratory datasets (see Section 2.2.8) are rare. It is difficult to simulate normal traffic in a laboratory environment as the traffic never is evenly distributed between different network protocols. Also network element failures and performance fluctuates significantly in a normal network which is not easily simulated in a laboratory. [13]

Once the model of normal network traffic is created, traffic is then monitored for deviations from the model. Some analysis is needed to decide whether the deviation is intrusive or malicious. Normally this analysis is done by a network security guard. As the detected anomalies might be previously unknown it is difficult to know what is

actually causing the anomaly and whether it is intrusive or not. [13] This anomaly analysis process needs to be supported by as much information as possible, so that the security guard could work efficiently. In anomaly detection, there is a wide variety of approaches to choose from and some of them are discussed in the following paragraphs.

Statistical based Anomaly Detection

In a statistical based method anomalies are detected from statistics. Statistical based methods create models based on history. These models are then compared to the current situation and deviations between these models are considered as anomalies. Once a deviation is monitored its severity is then evaluated and graded. The more severe the anomaly is the higher the grade is. [15] For example, the average number of times a user has accessed the network daily is compared to the current amount. If the current number of access to the network exceed the average number by one or two it is not maybe considered as a severe anomaly. But in case the number is, for example, ten times or even hundred times higher, it might be a severe anomaly. This of course depends on how the grading rules are defined.

Rule-modelling based Anomaly Detection

In a rule-modelling based method rules are defined for the system and once these rules are broken, those instances are marked as anomalies. [10] Basically this is similar to how firewalls operate. Firewalls have predefined rules which are matched against network traffic. If the traffic is not in conflict with these rules it is then allowed to pass through. Everything that is against these rules is dropped. In anomaly detection this would mean that everything that is against the rules is thought of as an anomaly.

Threshold based Anomaly Detection

In a threshold based method, thresholds are defined for the data deviation monitoring. Once a threshold is crossed, that instance is marked as an anomaly. [10] In a way, threshold based anomaly detection is a combination of statistical based and rule-modelling based methods. Threshold itself is a rule that is created based on statistics. The network administrator knows, for example, how high the CPU usage is on a network element. Therefore he can set a threshold that creates a rule which says that CPU usage cannot be more than 80 percent. An alarm is triggered once this threshold is crossed.

Machine-learning based Anomaly Detection

In a machine-learning based method anomaly detection models are constructed based on past behaviour. The learning algorithm analyses, for example, previously recorded data sets containing network traffic and create a model of normal behaviour. After the learning period the detector monitors deviations from this created model. A machine-learning based detector can adapt to changes in the network traffic when, for example,

some application is distributed to all local machines in the network and this application generates previously unknown traffic to the network. [16]

Payload based Anomaly Detection

In a payload based method, anomaly detection models are created based on the application payload data to a specific host and port. In addition to this a standard deviation is calculated based on the payload length. Once the model is created, all the traffic coming in to a specific port is analysed and the payload length is matched against the model's average length. If the difference is too large, an alarm is triggered. [17]

Protocol based Anomaly Detection

A protocol based anomaly detection monitors protocols for deviations from the protocol standard specifications. The detector creates models based on TCP/IP protocol specification which is then matched against the network traffic. If the monitored traffic operates with a protocol that is in conflict with the specification, it is then marked as an anomaly. Most of the protocol based anomaly detectors are built as state machines. This is understandable as all connection oriented protocols have a state. The detector is therefore monitoring transitions from one state to another and if the anticipated transition is different from the transition that has occurred, an alarm is triggered. [18]

Graph based Anomaly Detection

A graph based anomaly detection creates activity graphs of hosts and the activity in a network. These activity graphs describe how the activity is spreading in a network. For example, if the activity graph becomes a huge tree-like graph the activity is then considered as anomalous or a worm spreading to be more precise. [19]

Signal Processing Techniques based Anomaly Detection

Methods that are based on signal processing techniques are also researched widely. For example Fontugne et al. [20] used image processing-based approach in their anomaly detection system. Their system is based on pattern recognition, where anomalous traffic flows are detected through behaviour-based signatures. The most common interest has been on using signal processing methods to enhance the overall efficiency and at the same time reduce the amount of false positives. [21]

Data Mining based Anomaly Detection

A data mining based anomaly detection tries to automatically discover consistent patterns of features from large stores of data that describe the behaviour of network traffic, user or programs. Classifiers are constructed based on these features which are used to classify the monitored features into anomalies and known intrusions. Data mining is an example of method that combines algorithms used in different methods like in machine-learning, statistical and signal processing based methods. [22]

All of these methods have their pros and cons depending on what is the monitoring target. A protocol based detection method is efficient on analysing network protocols but is not capable of detecting malicious payload. The same applies vice versa, payload based detection method can be efficient in detection malicious data in payloads but is not efficient in detecting intrusive use of protocols. In some cases a combination of different methods is more suitable. The environment and its features have to be evaluated in order to choose the most efficient setup to detect intrusions in that specific environment.

2.2.7 Prior Research on Intrusion Detection

Intrusion detection has been studied widely since Anderson introduced the concept of intrusion detection in 1980 [23]. However the initial push forward in the field of IDS research was received seven years later in year 1987, when Denning introduced an intrusion-detection model, also known as Denning's model [14].

Denning's Model

Denning presented an idea that malicious behaviour could be perceived from system use by comparing it against a model of a normal system use. Denning's model describes the operation of a host based IDS that is used to monitor usage of a local machine. Intrusions in her model are detected by first creating profiles of normal system usage and then the system's usage is monitored and compared against these pre-defined profiles. Denning's idea is that malicious usage of the system can be detected as a deviation on normal usage profile. [14]

Denning's model has been widely used as a basis for different intrusion detection systems and its influence can be seen on the prior research on intrusion detection where the focus has been mainly on host based IDS. Axelsson [24] published a survey on intrusion detection systems in year 2000 in which he listed 20 research projects from years 1988 to 1998. From the 20 studies on IDS there were 14 that were completely host based, three that operated both in host and in network and two that were completely network based. [24]

Gates et al. [13] challenges the use of Denning's model as an inclusive model for all types of IDSes (NIDS and HIDS). Their argument is that as Denning's model is designed to be a model for host based IDS. As such without modifications it might not be usable as a basis for network IDS. Second argument from them was that Denning's model was created in 1987 when detecting system behaviours on a local machine was more important than analysing the network traffic, the model itself might be too old to meet the requirements of modern environments.

Network Intrusion Detection System Researches

From 21st century onwards, while networks have been developing rapidly, network based IDS has received more attention. Change of focus in IDS research from HIDS towards NIDS can also be explained by the research value. HIDS has been studied

widely and new findings on that field are difficult to find. NIDS instead is a more interesting topic because network based intrusions are constantly increasing. This gives new opportunities for the research field to discover new methods that are able to detect previously unknown threats and publish a research paper from it.

Another reason for NIDS popularity as a field of research is that firewalls have not developed as fast as NIDSes have. Firewalls are able to give basic security but they are not able to cope with constantly evolving attacks. Currently the de facto standard in network intrusion detection is Snort [25] which could be easily and falsely described as a network firewall.

Snort is an IDS/IPS that combines signature, protocol and anomaly based intrusion detection methods to efficiently detect and prevent intrusions. Snort has been developed by Sourcefire that also regularly provides rule updates to Snort [26]. In addition to Snort, some network based IDS studies are discussed in the following paragraphs.

Autonomous Agents for Intrusion Detection (AAFID)

AAFID was a project within the centre for education and research in information assurance and security (CERIAS) in Purdue University. The project group consisted of students and faculty who were interested in developing a new type of intrusion detection system. Their approach is to use a distributed architecture of IDS agents to cover the operation of the whole network. [27]

Common Intrusion Detection Framework (CIDF)

CIDF is a project in which a common framework for protocol and application programming interfaces is developed. The project is currently coordinated by Schnackenberg and Tung. Their goal is to make it easier for intrusion detection research projects to share information and resources. [28] Based on the field literature it seems that the CIDF is not widely used.

Distributed soft computing intrusion detection system (D-SCIDS)

D-SCIDS consists of multiple distributed IDS sensors over a large network. IDSes communicate with each other directly or through a centralized server that also provides advanced network monitoring. In their research Abraham et al. [29] evaluated three fuzzy rule-based classifiers to detect intrusion in network and were then further compared with other machine learning techniques. [29]

Next-Generation Intrusion Detection Expert System (NIDES)

NIDES is a real-time IDS that monitors user activity on multiple target systems. NIDES is placed on a single host that analyses audit data collected from interconnected systems. Intrusion detection on NIDES is a hybrid of misuse detection and anomaly detection; a rule based signature analysis and a statistical profile-based anomaly detector. The notation expert in NIDES means a system that is intelligently processing intrusion

alarms to decide whether further investigation from a security guard is needed or not. Further development of NIDES evolved into SRI's project called EMERALD. [30]

Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD)
EMERALD is a tool for tracking intrusive activity through and across large networks. The EMERALD consists of multiple polymorphically distributed detectors that can be tuned independently. The detectors are EMERALD eXpert and EMERALD eBayes. EMERALD eXpert is a signature based intrusion detector and EMERALD eBayes is an adjustable anomaly detector. CIDF [28] is used as a basis for communicating information between detectors. [30]

Graph-based Intrusion Detection System (GrIDS)
Staniford-Chen et al. [19] have presented a graph-based IDS that collects activity data on computers and network traffic between them and then aggregates the information into activity graphs. These graphs reveal the causal structure of network activity and allow detection of large-scale attacks. Intrusions are detected by analysing the characteristics of the activity graphs.

Spitfire
Spitfire was developed to enhance the work of NIDS operators. It can be used as a replacement or as a supplement to the Cisco Net Ranger or ISS Real secure GUI. Spitfire can be used in real time operation or it can be used to analyse historical information. Spitfire provides a robust historical database of intrusion activity that can be used to detect trends and patterns. [31]

2.2.8 Lincoln Laboratory Dataset

Lincoln laboratory data sets are "*the first standard corpora for evaluation of computer network intrusion detection systems*" [32] and were created under the sponsorship of Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL). [32]

Lincoln laboratory collected two dataset in consecutive years in 1998 and 1999. The 1998 dataset contains seven weeks of training data and two weeks of testing data which contain network traffic and operating system logs. These datasets contain labelled anomalies and network attacks mixed with normal network traffic. Similarly, the 1999 data set contains five weeks of training data and testing data but in addition to 1998's data set, the 1999 contains also attack free training data. This attack free data can be used by IDS to create a model of normal network traffic. [32]

Lincoln laboratory datasets have been used many times by IDS researchers since they were published. For example, Lu et al. [33] converted network packet logs into network flow-based logs and used this converted dataset in their wavelet analysis based IDS. In addition, the Lincoln laboratory 1998 dataset is also converted into connection-based dataset which is also known as KDD cup 1999 [34]. The KDD cup dataset is

especially used in evaluation of IDSes that are based on the machine-learning method. For example, Abraham et al. [29] evaluated their distributed soft computing intrusion detection system (D-SCIDS) with the KDD cup dataset.

The datasets have proven their usefulness in evaluation of the IDS in the past but current IDSes and their evaluation should not rely only on these datasets. As time has passed, mobile operators have more and more new network protocols flowing through their networks and nowadays there are more applications that generate traffic into modern networks. For example, at the time when Lincoln laboratory created these datasets, there was no torrent traffic which is nowadays causing most of the network traffic [35].

Attacks in Lincoln Laboratory Datasets

Both of the datasets contain four categories of attacks; denial of service (DoS), user to root (U2R), remote to user (R2U) and probing attacks. In addition to these four categories, the 1999 dataset contains a group which is called data. Full list of attacks with descriptions are in Appendix 2.

Attacks that belong to the DoS category make the computing or memory resources in the targeted system too busy or full. In general, the attacks exhaust resources in such length that the use of the resources is completely denied to the legitimate users. [32]

U2R attacks exploit vulnerabilities in the targeted system to gain a root access. What is similar to all of the attacks belonging to this category is that the attacker begins with a normal user account that is obtained by other means such as social engineering or phishing attacks. After accessing the targeted system with a legitimate user account the attacker begins to exploit vulnerabilities in the system that would eventually lead into a situation where the attacker is given root access rights. Common exploit is to cause a buffer overflow in which the targeted system tries to read data into the buffer without checking whether the data fits into the buffer or not. As a result the system crashes into a state where the user accessing the system can change the user account into administrator. [32]

Attacks belonging to the R2U are remotely exploiting vulnerabilities in the targeted system in order to gain an unauthorised access. In general the attacker tries to gain a local access as a user in the targeted system. An example of an attack belonging to this category is a dictionary attack in which the attacker tries to repeatedly guess usernames and passwords in the targeted system. [32]

Port scanning and network mapping are a good example of probing attacks. Both attacks try to find out information of the targeted system or network. In general the attacker tries to find out possible medium, for example an open port, which he or she could exploit. [32]

Data category contains an attack called secret. It is an attack where a legitimate system user performs actions that he or she is able to do but which are not allowed according to the use policy. [32]

2.3 IDS in Telecommunications Networks

IDS's role in telecommunications networks is to enhance the overall security of the network together with existing security measures such as firewalls and antivirus scanners (see Section 2.2). IDS's place in the telecommunications networks depends on what it is supposed to monitor and protect. For example, IDSes could be monitoring intrusions from either inside or outside the core network.

2.3.1 IDS placement Challenges

The placement of IDS depends on the type of the IDS. Host based IDSes are typically placed on elements that provide important services to the network. Network based IDSes on the contrary are more difficult ones. NIDS placement has to be balanced between network coverage and allocated resources.

In access networks (see Figure 2.1) IDSes monitors for malicious payloads in transit through the network and intrusive subscribers whose actions could disturb the service which other subscribers are enjoying. In addition IDSes monitors for intrusions whose target is inside the core network.

In a core network (see Figure 2.1) IDS monitors for intrusions that try to gain an access to the core elements such as gateways, HLR/VLR and subscriber charging. Access to these elements could harm the overall operation of the network. Preferable IDS type in the core network would be host-based IDS on important network elements. The HIDS instead of NIDS is preferred as it is known which elements are the most important ones and which also requires protection. It should be taken into account that in addition to monitoring the system the HIDS is located; it is also monitoring the network traffic from and to the host.

If the used strategy is centralized IDS then its placement needs to be considered more carefully than in case of distributed IDS. In Figure 2.7 possible locations of IDSes in telecommunications networks are presented.

These locations in Figure 2.7 are based on Cisco's IDS sensor deployment considerations [36] in which the deployment is began by doing an analysis on network topology. The key factors are Internet access points, extranet access points, remote access and intranet separation. IDSes monitoring subscriber network provide security to all key factors. To enhance the overall security IDSes should be placed inside the core network, to monitor gateway towards Internet and extranets and in addition to these, host based IDSes should be considered in subscriber equipments.

In addition to Cisco's deployment considerations, a host-based IDS could be used in the user equipments such as mobile phones and computers. Miettinen et al. [37] proposed a unified IDS framework for mobile phones. The same framework could be used with other mobile devices as well.

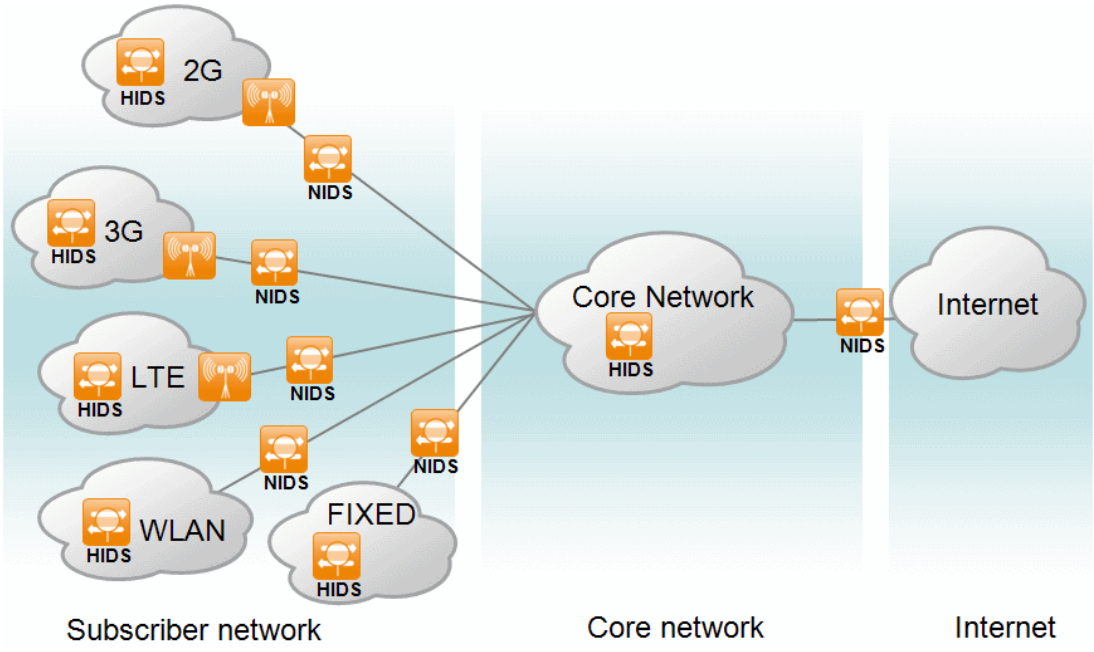

Figure 2.7 Possible locations of IDSes in telecommunications networks

2.3.2 Centralized Model

The most common setup for IDS is to use one dedicated IDS, also known as centralized IDS, which is monitoring all the incoming and outgoing links in the network. In Figure 2.8 this would mean that the central IDS would be in charge of all the intrusion and anomaly detection operations.

Centralized IDS provides easier operation and management functionality in comparison to a distributed model when the network size and the amount of traffic are small. Scalability can become a problem when the network size grows.

2.3.3 Distributed Model

Distributed IDS model in telecommunications networks is presented in Figure 2.8 in which the whole intrusion and anomaly detection workload is distributed among IDS agents, also known as IDS sensor, together with the central IDS. [29]

IDS agents could be used as sensors to pre-analyse network traffic and generate alarms from detected intrusions. These IDS agents could also pre-process the captured packets to enhance the efficiency of the Central IDS (CIDS). They could, for example, discard unnecessary information from the IP packet header fields. [29]

For example Handley et al. [38] normalize IP packet header fields to detect skilled attackers that try to evade detection by exploiting ambiguities in the traffic stream. In addition to header modification, IDS agents could also generate additional information such as adding grades to alarms that would categorize the alarms into three groups depending on the level of seriousness of the detected intrusion.

The Central IDS would be responsible for gathering information from IDS agents. CIDS would have a more advanced view of the network and its state and therefore it could detect coordinated network wide attacks. [29]

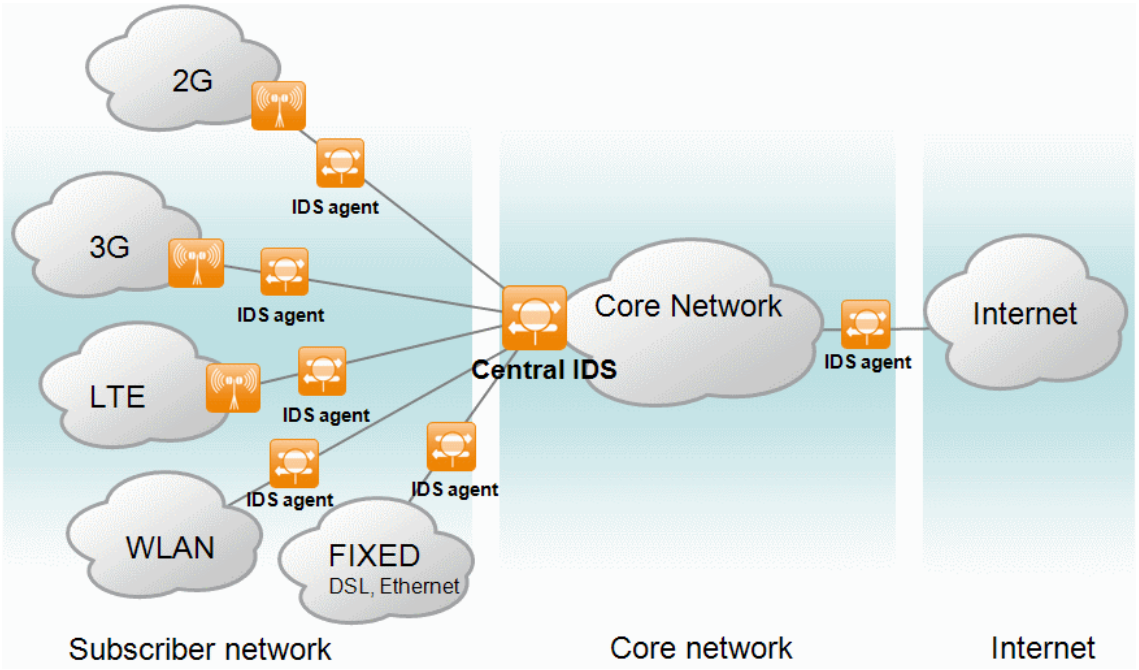

Figure 2.8 Distributed intrusion detection model in telecommunications networks

Distributed IDS model is able to scale up into large size networks, especially when the amount of monitored links is huge. Scalability is the most relevant feature in comparison to the centralized model.

3 FEATURES FOR IDS

Before anomaly detection based IDS can raise an alarm, it needs to have some kind of model on what is normal traffic and what is not. It also needs to have predefined methods in which traffic is filtered and possibly modified in order to cope with the huge amount of data going through the network daily. This is why it is necessary to choose among the data what is relevant for monitoring and what is not. Feature extraction plays an important role in the choosing of relevant features for the IDS.

The basic principle in feature extraction is that the fewer features there are to be monitored, the faster the IDS is. Vice versa the more features the IDS has to monitor the less accurate it is. Of course this is not an absolute truth as there are cases in which detection accuracy has increased after taking additional features into the monitoring.

These cases are discussed in more details in Chapter 3.3. The importance of feature analysis is significant when evaluating the performance and detection rate of an IDS. For example network traffic contains features that are redundant or their contribution to the detection process is little. By reducing the amount of features the IDS's computational speed is improved and the overall performance is increased. These principles are according to what is presented in the literature of the field. [39; 40; 41; 42]

3.1 Feature extraction

Intrusion detection systems can either have univariate approach or a multivariate approach to detect intrusions depending on the algorithm used. In the univariate approach a single variable of the system is analysed. This can be, for example, port number, CPU usage of a local machine etc. In multivariate approach a combination of several features and their inter-correlations are analysed. [10] In addition based on the method the way in which features are chosen for the IDS can be divided into two groups; into feature selection and feature reduction.

3.1.1 Feature Selection

In the feature selection method the features are either picked manually from the data monitored or by using a specific feature selection tool. The most suitable features are selected by handpicking from the feature spectrum based on the prior knowledge about the environment that the IDS is monitoring. For example features that can distinguish certain type of traffic from the traffic flows are picked for the network traffic model training.

The idea behind the feature selection tools is to reduce the amount of features into a feasible subset of features that do not correlate with each other. Examples of feature selection tools are Bayesian networks (BN) and classification and regression tree (CART). Bayesian network is a probabilistic graphical model that represents the probabilistic relationships between features. [43] CART is a technique that uses tree-building algorithms to construct a tree-like if-then prediction patterns that can be used to determine different classes from the dataset. [44]

Feature selection process is illustrated in Figure 3.1 On the left there are the features ($F_0 \ldots F_N$) that are available from the data monitored, which is, for example, from network traffic. On the right side is the output ($F_0 ... F_M$) of the selection tool. The number of features in the output varies based on the selection tool used and the inter-correlation of features in the input. Following the basic principles of feature analysis the number of features in the output (M in Figure 3.1) is in most of the cases less than the number of features in the input (N in Figure 3.1). However, it is possible that the output is equal to the input.

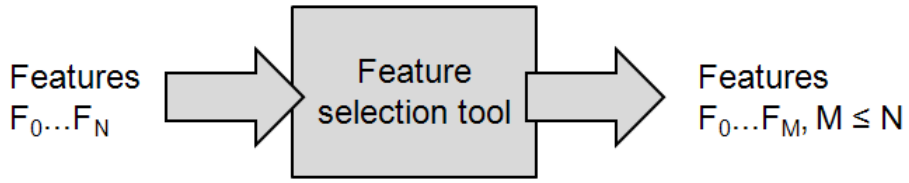

Figure 3.1 Feature selection

If the Lincoln laboratory dataset is taken as an example the feature selection tool will choose features from the network traffic header fields such as IP source address, source port number and other features described in Appendix 1.

3.1.2 Feature Reduction

In the feature reduction method a new set of features is extracted based on the features available from the data monitored such as network traffic data. The basic idea behind feature reduction method is to reduce the total number of features used in the network traffic model training. In general feature reduction means that during a certain period of time a number of different features are monitored and a new set of features are then calculated from this monitored data. For example the feature reduction tool could monitor number of packets to a specific destination, within a certain period of time. Then, once the monitoring period is over, a new feature (number of packets to that destination) is available for the IDS.

Another example of a feature reduction method is a principal component analysis (PCA). PCA is an algorithm that checks and converts the data set for all the correlated variables into a set of uncorrelated variables, also known as principal components. [45]

Feature reduction process is illustrated in Figure 3.2. On the left there are the features ($F_0 \ldots F_N$) that are available from the monitored data, for example, from the

network traffic. On the right is the output ($V_0 \ldots V_N$) of the reduction tool. The number of features in the output usually is less than in the input but it might as well be the same. The new features ($V_0 \ldots V_N$), can be calculated based on a single feature or a combination of multiple features ($F_0 \ldots F_N$).

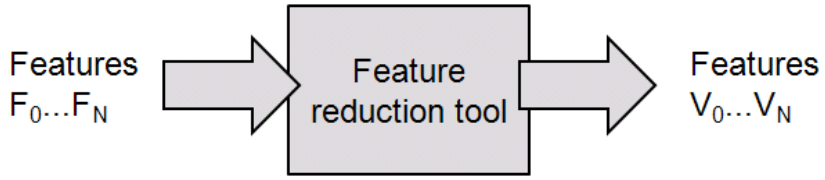

Figure 3.2 Feature reduction

KDD cup 1999 dataset can be thought as an example of feature reduction. The KDD cup consists of features (see Table 3.1) that are calculated from the network packet-based traffic in the Lincoln laboratory dataset to a flow-based traffic. These converted features are used for example in machine-learning based IDSes.

3.1.3 Challenges in Feature Extraction

The environment in which the feature extraction is done is a mobile operator's network with real people (subscribers) using it. This means that the network traffic contains user confidential information. For example in Finland user network traffic is protected by the data protection law [46]. Because of this, only a limited analysis for the network traffic can be done, meaning that a deep packet analysis cannot be done. In general, only the header fields of the packets can be checked but not the user data in the payload.

Scalability is an issue with IDSes. Because of the huge amount of data flowing through the mobile operator's network, it is not an easy task to find out the right information needed for an IDS. The problem is to find an answer to the question: "What features need to be taken into account when calculating or analysing whether the activity is malicious or not?"

In telecommunications networks link traffic can reach up to 150 Gbps traffic rates while current IDSes are capable of monitoring only some parts of the traffic. For example Sourcefire's IPS is capable to monitor network traffic speeds from 5Mbps up to 20 Gbps [26]. In order to cover the whole bandwidth, the traffic needs to be divided somehow and monitored by multiple IDSes. Then again the information provided by the IDSes needs to be correlated somehow which again adds another challenges to the whole intrusion and anomaly detection process.

Based on prior research on IDSes it is clear that either one of the techniques alone cannot detect everything but the combination of the both is the most promising approach. For example misuse detection can be used to filter known threats from the traffic to make it easier for the anomaly detection system to focus on the unknown.

Even though IDSes have been researched over 20 years, we still do not have an answer to the question of what features should be monitored. So far different kinds of

methods and algorithms have been developed for anomaly detection but the focus has been on making them more efficient. Almost all of them are lacking the same information; what features are important for IDS, especially in telecommunications networks? For some reason information on the used features is not easily found from IDS research publications. No matter what the reason is the result is the same; every researcher has to figure out by themselves which features should be used for the monitoring.

3.2 Audit Data Sources

IDS's operation is based on data analysis. In telecommunications networks there is a wide variety of different sources of data that produce information, or features to be more specific, that the IDS can analyse for intrusions and anomalies. In general there are two main sources of audit data that IDSes are using; network data and host-based security logs [24].

3.2.1 Network Data

Network data is collected using packet and flow capturers. Packet capturers can be either software or hardware based products. Wireshark and Tcpdump are the most known freely available software based packet capturers. In addition to these two, most of the manufacturers of network elements such as routers and switches are also providing packet and flow capturers as a product that can be attached to their equipment. For example, Cisco provides a product called NetFlow [47] that can capture network traffic flows. The flow capturers monitor packet data and create flow information based on the communication between two endpoints. How flow is understood varies based on the monitoring system. The parameter that defines when a flow ends and a new one begins is the idle time between the communications of two endpoints and this time changes within monitoring systems. In addition there are tools that convert the packet data into network flow data. An example of such a tool is Argus [48].

Argus

Argus is a combination of two elements; argus-server and a set of argus-clients. The server is responsible of reading and converting the network traffic from packet data into flow based data. The argus-server can be used to monitor network traffic in real time or it can read packet data files that are stored in tcpdump- or pcap-format. [48]

Argus-clients are small programs that can read and extract additional information from the data flow created by the Argus-server. For example, the client program, racluster, can find the top talkers (communicates the most) and listeners (with less activity) within the flow data. The most relevant client program is named "Read argus" (ra). Basically it reads the Argus-based data flow and displays the flow information on the screen or writes it into a file. [48]

3.2.2 Host-based Security Logs

Logs contain records of events that have occurred within an organisation's system or network. Logging systems were created for the management and network failure detection point of view. As such they were not designed to be used as a security feature. Nevertheless, nowadays logs are also used for security purposes.

SNMP MIB can be used as an IDS data source (see Section 3.3.3). With SNMP it is possible to query the status of a network element or the element might independently send updates of its own status. In general the management entity requests the status of a network element using SNMP. The network element then sends log entries that correspond to the received request with SNMP. [49]

Another example of logs usage in anomaly detection is described by Höglund [50]. Höglund used UNIX user account logs to identify network user behaviour patterns and to recognize when the user's behaviour changes significantly from the normal pattern. [50]

In telecommunication networks there are multiple architectural elements (see Figure 2.1) that produce information that could be used in intrusion and anomaly detection. Just to name a few there are AAA-server, databases, gateways, SGSN, MME, Charging and HLR/VLR. These elements generate security and management logs that can be used in intrusion and anomaly detection. For example the information gathered from the network elements could be used together with the alarm reports generated by IDS to find the root cause for the intrusion or anomaly.

As most of the network elements generate logs there are other sources of security logs as well. Kent et al. [51] define three categories of audit data sources that generate security logs; security software, operating systems and applications.

Security Software

Network- or host-based security software can be classified, for example as antimalware program, a firewall, a proxy server, an intrusion detection and prevention system and an authentication server. Security software's main purpose is to provide security information that can be used by other security solutions such as IDS. [51]

Firewalls and antimalware software generate logs on events when suspicious or malicious activity is detected. Proxy servers generate logs from network connections and web requests associated with the connections. In addition if the proxy server has user authentication functionality it will also log user credentials from the persons who are accessing web resources. Proxy servers with AAA functionality are especially useful in audit trailing. Intrusion detection and prevention systems produce logs as any other security software. [51] These logs can be used for example in a distributed IDS architecture (see Section 2.3.3) where the central IDS collect the logs from the IDS sensors.

Operating Systems and Applications

Computers, mobile devices, servers and networking devices such as routers and switches are operated by an operating system. Applications are operating on the top of the operating system and are therefore able to access some of the information that the operating systems are generating. While performing operational actions the operating systems and applications generate logs from system events and audit records. [51]

System events are generated usually from successfully or unsuccessfully completed actions, status of the system and services that are running. Audit records contain information about authentication operations such as successful or failed user authentication. In addition audit records are generated with information on what files the user is accessing and with which privileges. [51]

3.3 Features used in Prior Art

Features used in prior research on IDSes are roughly organized into five categories; features based on flow data, packet data, SNMP data, features collected from UEs and features used in ad-hoc network monitoring.

3.3.1 Flow-based Features

Lakhina et al. [52] analysed events that affected to the distribution of traffic features and marked these as anomalies. They monitored network-wide backbone traffic using the following IP packet header data:

- source IP address
- destination IP address
- source port number
- destination port number.

They grouped known anomalies into seven categories based on the type of the detected attack. These were DoS, Flash Crowd, port scan, network scan, outage events and worms to name few. The classification was done using multiway subspace method together with the k-means clustering algorithm. The multiway subspace method is able to isolate correlated changes on the four IP packet header features (source and destination IP address, source and destination port number) between traffic flows. [52]

The same features are also used by Fontugne et al. [20] in their image processing-based approach to detect anomalies. They compared their proposed anomaly detection method against a statistical-based method proposed by Dewaele et al. [53]. The comparison was done using a network traffic data collected from Trans-Pacific. Fontugne et al. [20] categorised the results in similar way than Lakhina et al. [52] did but instead of grouping the detected anomalies into seven groups, they grouped them into 15.

Gorton [54] used two detection methods to analyse a router log data; a single event and threshold analysis. The single event analysis raises a flag of intrusive activity when a single event is discovered. In the threshold analysis intrusive activity is flagged with respect to accumulated activities. In his analysis he collected syslog messages from Cisco routers and transformed the log data into a set of features that are:

- time from the syslog
- status that can be either permitted or deny
- protocol identifier
- type of service
- source IP address
- source port number
- destination IP address
- destination port number
- number of ICMP messages
- number of packets.

With single event analysis Gorton was able to detect spoofed connection attempts, connection attempts to known Trojan horses, connection attempts to known vulnerable ports, the Land DoS attack, TCP-broadcasting, the echo-chargen attack, ICMP and UDP echo request. With threshold analysis Gorton was able to detect SYN flooding, network mapping and port scans to name few. [54]

Knuuti [55] compared the usability and performance of three different IDSes in a large IP networks. The evaluated IDSes were Snort, Bro-IDS and TRCNetAD. Snort and Bro-IDS are capable of analysing traffic in real-time when TRCNetAD is a non-real-time anomaly detection based IDS. [55] Features that Knuuti used are [55]:

- IP address
- time stamp
- number of ICMP packets
- number of UDP flows
- number of TCP connections
- amount of received data
- amount of sent data
- number of received packets
- number of sent packets
- number of different port numbers used over 1024
- number of port numbers used over 1024
- number of different port numbers used below 1024
- number of port numbers used below 1024
- number of receiving sequences from different IP’s

- number of receiving sequences
- number of sending sequences to different IP's
- number of sending sequences.

Knuuti conducted two, one week long, traffic capturing periods to collect data for the IDSes. From the data collected he then generated time series that are 60 minutes long in order to create clusters and analyse the data with self-organising maps. Snort detected over 1.5 million intrusions during the one-week traffic capturing period. Snort was able to detect the following attacks:

- buffer overflow attacks
- Trojan
- denial of service
- VoIP attacks
- Heap overflow attack
- DNS spoofing attack
- spyware.

Bro-IDS detected approximately eight thousand intrusions which were address and port scan. TRCNetAD detected 150 thousand anomalies during the same time period. Knuuti also evaluated alarm similarities between the detectors and his conclusions were that TRCNetAD was able to detect some of the port and address scans that Bro-IDS discovered but there were no similarities between Snort's and TRCNetAD's findings. [55]

KDD CUP 1999 Studies

As mentioned in Section 2.2.8, the KDD cup is widely used in evaluating the IDSes' performance and detection rate. The same fundamental problem exists with these studies as described in Section 2.2.7. Most of the studies are only describing the achieved results of the IDS not how they managed to reach them. What creates even more confusion is that in some of the studies the researchers are implying that they are using all or just a specific amount of features from the 41 features in KDD cup. Comparison of these studies is therefore impossible based on the data available. However, because of KDD cup dataset's popularity, there are also studies available on the Internet which do provide detailed information about the features and the methods that they used. Such studies are presented in the field literature [39; 40; 41; 42; 56].

These studies evaluate optimal feature subsets of each of the five categories (see Section 2.2.8) in the Lincoln laboratory 1998 dataset. The features extracted for the KDD cup 1999 dataset are listed in Table 3.1. The features in Table 3.1 were converted into data flows from the packet data in 1998 Lincoln laboratory dataset using a Bro-IDS

[57, p. 146]. The Bro-IDS [58] is very much similar to Argus discussed in Section 3.2.1 with the difference that Bro is also an IDS.

Table 3.1 Features in KDD cup 1999 dataset [34]

Label	Feature	Label	Feature	Label	Feature
A	duration	O	su_attempted	AC	same_srv_rate
B	protocol_type	P	num_root	AD	diff_srv_rate
C	service	Q	num_file_creations	AE	srv_diff_host_rate
D	flag	R	num_shells	AF	dst_host_count
E	src_bytes	S	num_access_files	AG	dst_host_srv_count
F	dst_bytes	T	num_outbound_cmds	AH	dst_host_same_srv_rate
G	land	U	is_hot_login	AI	dst_host_diff_srv_rate
H	wrong_fragment	V	is_guest_login	AJ	dst_host_same_src_port_rate
I	urgent	W	count	AK	dst_host_srv_diff_host_rate
J	hot	X	srv_count	AL	dst_host_serror_rate
K	num_failed_logins	Y	serror_rate	AM	dst_host_srv_serror_rate
L	logged_in	Z	srv_serror_rate	AN	dst_host_rerror_rate
M	num_compromised	AA	rerror_rate	AO	dst_host_srv_rerror_rate
N	root_shell	AB	srv_rerror_rate		

Zainal et al. [39] evaluated in their study the detection rate of IDSes by using five optimal feature subsets extracted from the 41 features in the KDD cup dataset (see Table 3.1). For the extraction they used five different methods to calculate and select six most important features for each subset.

They extracted two of the optimal feature subsets by using particle swarm optimisation (PSO) and rough set theory (RST). The remaining three subsets were chosen according to the study by Sung et al. [56] in which they used support vector decision function ranking (SVDF), linear genetic programming (LGP) and multivariate regression splines (MARS) to select optimal feature subsets. [39, see 56] The extracted features are summarised in Table 3.2 (SVDF, MARS, LGP, Rough set and Rough-PSO).

PSO is a population-based search algorithm that organises particle swarms into an optimal regions based on the historical behaviour of each particle and its neighbours. RST is a feature selection tool to find data dependencies and to reduce the number of features in a dataset. [39] SVDF, LGP and MARS are used in a similar fashion to select optimal feature subsets to reduce the number of features in a dataset. [56]

Mukkamala et al. [40] used two feature ranking and selection methods to choose feature subsets for each attack type groups described in APPENDIX 2 Appendix 2. These feature selection methods were performance-based ranking method (PBRM) and support vector decision function ranking method (SVDFRM). The selected features are summarised in Table 3.2 (SVM, SVM (PBMR) and SVM (SVDFMR)).

In PBRM in every loop one feature is dropped from the feature set and the remaining feature set is used to train the IDS. Then this IDS's performance is evaluated and if the performance is improved the dropped feature is marked as non-important feature. In case the performance is lowered the dropped feature is marked as an

important feature and is taken back to the feature set. This iteration is continued until all the features in the feature set are tested and evaluated. [40]

In SVDFRM the contribution of each feature to the classification is ranked by the weight of contribution to the anomaly. If one feature affects significantly to the classification it is then given a higher weight value than to the feature whose influence to the classification is minor. The weight of each feature can be extracted from the support vector decision function. [40]

Chebrolu et al. [41] evaluated the performance of two feature selection algorithms Bayesian networks (BN) and classification and regression trees (CART) (Section 3.1.1) and their ensemble. The selected features are summarised in Table 3.2 (BN, CART and BN+CART). Their conclusions were that the detection rate changes significantly between the feature selection methods and therefore an IDS should be modularly designed. In general the modularity means that each module would use different feature subsets to detect a specific group of the attack categories.

According to studies presented in the literature of the field [39; 40; 41; 42; 56], it can be said that the detection rate for different attack types is higher by using different feature sets for each attack type category instead of using the same features for all the attack types. In addition it can be said that by using less features it is possible to reach higher detection rate than by using all of the available 41 features in KDD cup dataset. More detailed results are presented in Appendix 3.

Table 3.2 Features used in KDD CUP 99 studies

Method	No. features	Features
SVDF	6	B,D,E,W,X,AG
MARS	6	E,X,AA,AG,AH,AI
LGP	6	C,E,L,AA,AE,AI
Rough set	6	D,E,W,X,AI,AJ
Rough-PSO	6	B,D,X,AA,AH,AI
SVM	41	ALL
SVM (PBMR)	31	A,C,E,F,H,I,J,N,O,P,Q,R,S,T,U,V,W,X,Y,Z, AA,AB,AC,AF,AG,AI,AJ,AL,AM,AN,AO
SVM (SVDFMR)	23	A,B,C,D,E,F,J,L,Q,W,X,Y,Z,AA,AB,AC,AE, AG,AH,AJ,AL,AM
BN	41	ALL
BN	17	A,B,C,E,G,H,K,L,N,Q,V,W,X,Y,Z,AD,AF
BN	12	C,E,F,L,W,X,Y,AB,AE,AF,AG,AI
CART	41	ALL
CART	17	A,B,C,E,G,H,K,L,N,Q,V,W,X,Y,Z,AD,AF
CART	12	C,E,F,L,W,X,Y,AB,AE,AF,AG,AI
BN+CART	41	ALL
BN+CART	17	A,B,C,E,G,H,K,L,N,Q,V,W,X,Y,Z,AD,AF
BN+CART	12	C,E,F,L,W,X,Y,AB,AE,AF,AG,AI

3.3.2 Packet-based Features

Kabiri et al. [59] have conducted research on identifying effective features for intrusion detection. They have done related research for detecting probing attacks [60] and for detecting smurf attacks [61]. Results from these researches are used in [59] as well.

Kabiri et al. [59] used Lincoln laboratory dataset 1998 to select optimal features from the IP and TCP packet header fields. Appendix 1 lists all the 32 basic features that they extracted from network traffic header fields. They used principal component analysis (PCA) method to select optimal feature subsets from the 32 features for each of the five categories (see Section 2.2.8) in the Lincoln laboratory dataset. The suggested feature subsets are listed in Table 3.3.

In their work Kabiri et al. [59] investigated the information value for each category and their conclusion for future work stated that these features should be experimented in an intrusion detection system. In addition a comparison of accuracy and efficiency should be done using the feature subsets and by using all the 32 features.

Table 3.3 Features Kabiri et al. used [59]; [60]

No.	Feature	DoS	U2R	R2L	Probing	Normal
1	Protocol	x			x	
5	Coloring_rule_name	x	x		x	
10	IP_Total_Lenght				x	
12	MF_Flag_IP		x		x	
13	DF_Flag_IP		x		x	
16	Protocol_no	x				
19	Stream_index	x				
24	Urgent_flag		x	x		
25	Ack_flag				x	x
26	Psh_flag				x	
27	Rst_flag			x	x	x
28	Syn_flag	x	x		x	
29	Fin_flag				x	

Carrascal et al. [62] used self-organising maps together with learning vector quantization in their machine-learning based method to detect intrusions. They evaluated their anomaly detection efficiency by using Lincoln laboratory data sets as a testing data. Their system's detection rate was 72% and false positive rate 2%. In comparison they provided a list of other AD methods whose detection rate was better than their method's but with a higher false positive rate. Features that Carrascal et al. used were [62]:

- codification of TCP flags
- IP protocol number
- IP type of service

- TCP window
- packet size
- codification of <source port / source IP address, destination port / destination IP address>
- destination port
- source port
- source IP
- destination IP
- codification of TCP options.

Most of the features are self-explanatory but the coded features are not as clear. Carrascal et al. [62] combined features that have multiple parameters such as TCP flags and TCP options into single features. The authors do not explain in details how the codification is done so one can only guess what the exact features are in reality.

3.3.3 SNMP-based Features

Lee et al. [63] used Simple Network Management Protocols Management Information Base (SNMP MIB) [49] to detect intrusion. SNMP is a protocol used in TCP/IP-network management and the idea to use it as a security monitoring tool is intriguing. SNMP logs are generated in network devices in any case and by using the already available logs do not add new requirements to the network infrastructure. By using SNMP MIB, some of the challenges in network intrusion detection can be avoided. There are no privacy concerns as user confidential information is not needed for the analysis. Also the data rates are low compared to network traffic amounts. SNMP MIB does not require any new hardware as the SNMP is widely supported. [63]

In their work Lee et al. [63] used 12 features from SNMP MIB in intrusion detection. Traffic on interfaces is estimated by analysing the correlation between IP group objects and interface group objects of SNMP MIB. Features from SNMP MIB that Lee et al. used are described in Table 3.4. In conclusions they proposed that only IP group features could be used to enhance the analysis performance. [63]

Table 3.4 SNMP MIB features [63]

Feature	Description
ipInReceives	Total number of input datagrams received from interfaces, including those received by error [64]
ipOutRequest	Total number of IPv4 datagrams which local IPv4 user protocols (including ICMP) supplied to IPv4 in requests for transmission [64]
ipForwDatagrams	Number of input datagrams for which this entity was not their final IPv4 destination, as a result of which an attempt was made to find a route to forward them to that final destination [64]

ipOutDiscards	Number of output IPv4 datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded [64]
ipOutNoRoutes	Number of IPv4 datagrams discarded because no route could be found to transmit them to their destination [64]
ipFragOKs	Number of IPv4 datagrams that have been successfully fragmented at this entity [64]
ipFragFails	Number of IPv4 datagrams that have been discarded because they needed to be fragmented at this entity but could not be, e.g., because their Do not Fragment flag was set [64]
ipFragCreates	Number of IPv4 datagram fragments that have been generated as a result of fragmentation at this entity [64]
ifInUcastPkts	Number of packets, delivered by this sub-layer to a higher (sub-) layer, which were not addressed to a multicast or broadcast address at this sub-layer [64]
ifInNUcastPkts	Number of packets, delivered by this sub-layer to a higher (sub-) layer, which were addressed to a multicast or broadcast address at this sub-layer [64]
ifOutUcastPkts	Total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent [64]
ifOutNUcastPkts	Total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent [64]

3.3.4 Features used in User Equipment monitoring

A combination of host based intrusion detection (HIDS) and remote IDS server is a system proposed by Miettinen et al. [37] and later used by Schmidt et al. [65] where user equipment (UE) has a host-based IDS monitoring the UE's behaviour and forwarding these monitored features to a remote IDS server. Remote server analyses the features for anomalies such as worms and other malware. [37] Features that Schmidt et al. monitored are [65]:

- amount of available RAM
- number of created TCP/IP connections
- user idle time in seconds
- CPU usage in percent
- battery charge level
- Boolean user idle indicator that is true if the user is idle and false if not
- amount of available hard disk space

- amount of running threads
- mobile phone network cell ID
- number of installed applications
- amount of opened Bluetooth connections
- amount of sent SMS messages
- amount of sent MMS messages
- number of received MMS messages.

Transfer of monitored features is proposed to bypass the processing and memory limitations of UEs. UEs are capable of monitoring its own system behaviours but they are lacking processing capacity for the intrusion detection analysis. Remote IDS server is capable of analysing inputs from multiple UEs. [37]

3.3.5 Features used in Ad-Hoc Network monitoring

Huang et al. [66] describe a method for detecting routing anomalies in Ad-Hoc networks. They created a list of attribute sets for two different groups; non-traffic related and traffic related. [66]

Non-traffic related attributes are [66]:

- time stamp
- node movement velocity (scalar)
- route add count for routes newly added via route discovery
- route removal count for stale routes being removed
- route find count for routes in cache with no need to re-discovery
- route notice count for routes added via overhearing
- route repair count for broken routes currently under repair
- total route change rate within the period
- average length of active routes.

Traffic related attributes are [66]:

- packet type data, route (all), ROUTE REQUEST, ROUTE REPLY, ROUTE ERROR and HELLO messages
- flow direction received, sent, forwarded and dropped
- sampling periods 5, 60 and 900 seconds
- statistics, measures, count and standard deviation of inter-packet intervals.

The traffic related list of attributes creates a list of 132 different features. Formula that is used to calculate the amount of different features is (6 x4 - 2) x3 x2. The same features are used by Huang et al. [67].

Wang et al. [68] used similar attributes as Huang et al. [66] but with a slight difference. They divided packet type data into two separate types; data size and data number. In addition, they dropped the statistics, measures, count and standard deviation of inter-packet intervals from the monitored attributes in order to decrease the total amount of features which would be 150 in total. Without those attributes the total amount of monitored features is 75. In their calculations Wang et al. have further decreased the total amount of features from 75 to 25. [68]

4 FEATURE SELECTION

The environment where the IDS is located affects significantly to the features that can be used by IDS. In Section 3.3 a set of different feature lists were described. It is clear that different sources of data produce different kind of features. For example, if the network traffic capturer is a flow-based capturer it provides completely different information than a packet capturer does. In addition to network traffic, there are other sources of data (see Section 3.2) from which a subset of features is selected or extracted using feature analysis methods described in Section 3.1.

Three approaches are used in this thesis to select the relevant features from the network traffic. These are; analyse the feature list described in Section 3.3, analyse different attack methods and how they affect to the network traffic and evaluate what other information the field literature holds on this topic.

4.1 Feature Analysis

As mentioned in Section 3.1, there are two different trends on how to extract the features. They can be chosen by using an algorithm that calculates correlated features and reduces the redundancy in the dataset (feature reduction) or new features can be extracted from the already available ones (feature selection).

In this thesis the feature selection is done by analysing the attacks within the Lincoln laboratory 1999 dataset (see Section 2.2.8) and how each attack are affecting to the network traffic. Through the scenarios a subset of features that are the most relevant to the corresponding attack category are then chosen.

The method used to monitor network traffic is to use flow-based data. There are many advantages in using flow data instead of packet data. The major advantage comes from the reduced need of storage space for the data. Network flows requires a one tenth of the original packet-based data which is a huge difference. Another advantage is that the flow data does not contain payload data at all. So the user privacy is no longer a problem. Also the traffic volumes such as the number of packets and bytes between destinations are easily extractable from the flow data so extra calculation is not therefore needed. The disadvantage with this data is of course the loss of individual packet information such as the size of the packet, structure of the packets in order to detect malformed ones etc. However these can be monitored by other methods such as misuse detection based IDSes.

4.1.1 Attack Scenarios

By analysing known attacks and their influence to the normal network traffic it is possible to define which features are affected and therefore should be monitored. The idea behind this approach is to define the characteristics of a specific attack group. This is done by analysing the attacks in Lincoln laboratory 1999 dataset (see Section 2.2.8). The attack categories in the dataset are:

- denial of service
- probing
- user to root
- remote to user
- data

Denial of Service

DoS attacks affect to the usability and reach ability of network services such as web, mail, voice and data. These attacks also affect to the reputation of a CSP. In most cases the attack itself is not detectable before the service or element in the network is overwhelmed by the amount of data it receives. Despite this fact there are some patterns that might be detectable.

According to Depren et al. [69] some of the DoS attacks are detectable by monitoring from the traffic flows the amount of data received by the destination in comparison to the amount of data sent by the source. In normal case the amount of sent data is around 40-50 bytes and as well the amount of received data is around 40-50 bytes. In a Dos case, the amount of bytes sent remains on the same level of 40-50 bytes but the amount of bytes received is zero.

The Lincoln laboratory dataset contains multiple DoS attacks that use different methods and techniques to crash the targeted host or service. In the following paragraphs some of the attacks and their influence to the network traffic are discussed. In addition to these attacks there are few others in the Lincoln data but their influence to the network traffic is either similar to the ones discussed below or the detection requires DPI which is not possible to do because of the user privacy constraints.

Using HTTP it is possible to cause a DoS state. This can be achieved by inserting multiple (more than 20) headers into a single HTTP-request message. In Lincoln laboratory dataset the attack Apache2 sends a HTTP-request that contains 10000 headers in a single message. [32]

The attacker sends a TCP SYN-message that has the same address as the source and destination. The land attack requires only a single packet sent to the destination. This attack is not anymore feasible as the new systems can cope with these messages. But in case of a mistake in the system code or reuse of an old one, this might still be feasible even today. This is why these packets should be monitored within the network traffic. [70]

ICMP-messages with a larger payload than 64kB might cause unpredictable reactions in the targeted systems. This attack, also known as ping of death, was applicable to older operating systems that could not cope with abnormal ICMP-messages. A malformed ICMP-message caused freeze, reboot or crash on the destination system. Modern operation systems are not affected anymore by this attack but it is still possible that some behave abnormally when oversized ICMP-message is received. Therefore these messages should be monitored. [71] In addition other protocols should be monitored as the same method used with ICMP can be used with other protocols as well. Targa3 is an example of tool that generates malformed IP packets [72].

In smurf attack the targeted host is flooded with multiple ICMP response messages from multiple sources. The attack requires three entities, the attacker, middleman and the destination. The attacker sends ICMP echo request packets to the middleman with the target host as the source address. The middleman then sends response messages to the targeted host. In order for this attack to cause a DoS scenario to the target, the attacker needs to send multiple messages to multiple middlemen. These middlemen would then send a large number of response messages to the target that it would not be able to cope with the number of received messages. This kind of distributed attack is also known as distributed DoS. Smurf attack can be detected when a large number of ICMP echo replies are sent to a single destination. [73]

SYN-flooding together with IP spoofing attack is an example of DoS. In SYN-flooding the attacker sends multiple SYN-messages to the targeted server with a spoofed IP source address. The server tries to respond to these SYN-messages with a SYN-ACK-message and waits for ACK-message from the source. Because the source address is spoofed, the server will never get an answer to the SYN-ACK-message. The server creates a transmission control block (TCB) state that is reserved for each connection and is released after the connection is closed (received an ACK-message). If the attacker keeps on sending SYN-messages, the TCB-table begins to fill and after a while the table is full of these half-open connections and any further coming connections are rejected. TCB is emptied within a certain period of time but this does not help if the attacker keeps on sending SYN-messages with a spoofed IP address. [74]

Detecting a SYN-flooding attack might be difficult as the messages itself look legitimate. Still there are some clues that might give a hint of the ongoing SYN-flooding attack. One way, of course, is to notice that the targeted host is not reachable. This is, of course, the outcome of the attack and is not therefore the best way to find out that something malicious has happened. Another way to find out that a possible SYN-flooding attack is ongoing is to check host's state tables. If there are too many connections in SYN_RECEIVED state it might be because of a SYN-flooding attack. [74]

Flash crowd is an attack that does not belong to the attacks within the Lincoln laboratory data but is a very common root cause for a DoS state in a service. Therefore it is discussed also in this context. Flash crowd is an attack that is based on massive

amount of people requesting a connection or service from a single destination. The amount of requests becomes too large for the destination to handle which will eventually lead into a DoS state. Flash crowd can develop intentionally or unintentionally. Intentionally caused flash crowd attack is done by leading people or computers to try to connect to a single service at the same time. Example of an unintentionally created flash crowd happens often when the national lottery with a huge winning prize is drawn and people are trying to see the results from the web pages. Flash crowd attacks can be detected from the network traffic amounts and especially from the amount of service request within a short period of time [75].

In general most of the DoS attacks require multiple packets sent to the targeted host which will cause a memory and processing overloads, reboots etc. abnormality that prevents users accessing the service. There are though examples of single packets causing a crash on the destination such as in case of ping of death attacks. The features that should be monitored for DoS attacks are:

- Amount of received and sent bytes
- Number of connection from multiple sources to a single destination
- Number of packets to a single destination
- Number of flows to a single destination
- ICMP packet size (not detectable in flow data)
- Malformed packets such as HTTP messages with multiple header fields (not detectable in flow data)

Probing

Network mapping and probing are examples of reconnaissance attacks where the attacker tries to map out IP addresses and operating systems that are in use. In addition, the attacker tries to find out what services the computers are providing. Network mapping means an action where the attacker tries to map out the infrastructure of the network. To do this, the attacker is therefore targeting all the computers within the network. However, probing is an attack that tries to find out information from a single computer.

By conducting network mapping and probing attacks the attacker tries to find out all the possible means and methods that it can use to perform other attacks such as denial of service or gaining an unauthorised access to the inner network. Although reconnaissance attacks are not as serious threat as DoS attacks are, t they are still worth monitoring, because they are an omen for more harmful activity.

To send a single ICMP Echo Request message is the most common way to find out if there is a computer having this IP-address. This kind of network mapping attack is easily detected by firewalls and therefore does not pose a significant threat to the network.

A more sophisticated method is to reconnaissance through a port that uses TCP-protocol and communications through this port is allowed to pass firewalls. By sending messages through this port it is possible to map every computer inside the network. For

example, if Telnet connections are allowed through port number 23, the attacker could send TCP SYN-messages to the computers through this port. This kind of an attack is easily detectable if the attacker sends multiple messages within a short period of time. If the attacker distributes the reconnaissance messages and randomises the time between the packets, it is then more difficult to find out that these individual messages are part of a reconnaissance attack. Especially if the attacker is only interested in finding a small group of computers inside the network instead of trying to find every possible computer, it would be even more difficult to detect the attack. [76]

By using a different TCP flag option (ACK) the attacker could make it look like he is responding to connection requests that would look like a legal action. In this case the attacker would send TCP messages with ACK flag through the same port (23) likewise previously to the computers within the network. To make the attack even more sophisticated the attacker could use a specific source port number such as 80 to mimic a web server response messages. [76] In order to detect these reconnaissance attacks the IDS or firewall would have to be keep track of the connection states. In a traffic flow between two entities this means that if there are a lot of ACK-flagged TCP messages coming in without any SYN-flagged TCP messages ever send, the states are incorrect and the messages are part of reconnaissance attack. The attacker could also use RESET flags to achieve the same goal as was with ACK and SYN flags. RESET messages pose a greater threat than the others as they are not always monitored by firewalls and other monitoring systems. In case the firewalls or IDSes are stateless they might allow these messages go through to the inner network.

By scanning all or just a specific group of ports from the targeted computer, the attacker tries to find out if there is an open port or service that it can exploit. In addition to finding out what ports are in use or open, the attacker tries to find out what versions of the services are used. This information is valuable for the attacker because it can then find out what are the known vulnerabilities with the specific version of the service. [77]

Port scanning itself can be detected easily if the scanning is a constant activity, meaning that the attacker frequently sends packets to multiple ports on a single host. If the attacker distributes the port scanning attack it is then more difficult to detect. For example, the attacker might send a single packet to the destination and wait for a long period of time before sending another packet. IDSes without the knowledge of the past are in trouble detecting this kind of single packet attacks. Another version of distributed port scan attack is to use multiple sources to perform the scan and then combine the results afterwards. Again if the attack is conducted within a short period of time it is more detectable than when the time between packets is longer. [77] Other means to do a reconnaissance are social engineering, phishing and passive eavesdropping of network traffic. However, these attacks are not detectable by monitoring the network traffic.

Network mapping and probing attacks are based on methods that either use single packets or multiple packets. With a stateful firewall or IDS it is possible to easily detect most of the attacks belonging to this group. The features that should be monitored for probing attacks are:

- State of the connections
- Number of ports accessed by a single source
- Number of ICMP packets from a single source
- TCP flag combinations

Attacks against Mail or Web Servers

This group of attacks is a combination of DoS and probing attacks which are targeting services in the monitored network. Attacks in this category are web and mail bombs. Both of the attacks are relying on the same technique to cause a crash or attenuation of service in the targeted host. In both cases the attacker sends multiple messages (mail or web request) to the target service. Once the amount of received messages becomes too large to handle the targeted service's quality of service will suffer and in worst case the service crashes. [32]

Password guessing can be categorised in this category as the attacker tries to gain an unauthorised access to the CSP's services. Like in network mapping, the attacker could try to guess the password and gain an access to the service by brute force that would require multiple service requests within a short period of time. The attacker might as well distribute the password guessing by sending single requests once in a while with randomised time difference. Distributed attacks are more difficult to detect than brute force attacks. With a host based IDS the targeted system can keep track on the number of wrongly guessed passwords from certain IP addresses. So even when the attacker has distributed the service requests the HIDS is able to detect them. From the network traffic the same can be detected by monitoring the service request amounts to a single destination. [75]

With mail services it is difficult to set up a specific threshold which would distinguish a legitimate amount of messages from abnormal amount. This requires monitoring of message amounts in order to find out what is a normal message amount. The features that should be monitored for attacks against the services are:

- Number of service request
- Number of packets to a single service
- Number of flows to a single service

User to Root

U2R attacks are detectable with misuse based IDS from the network traffic packets. Attacks belonging to this category have a distinguishable pattern or a string in the payload that can be looked for. [32] An anomaly detection based IDS that does not monitor packet payload therefore cannot detect attacks belonging to this category. Of course some attacks might be detectable but as most of them are affecting only to the payload, detection of these attacks with anomaly detection based IDS is therefore not expected.

Remote to User
Similarly to U2R attacks, the R2U attacks are detectable only from the payload data by looking for specific patterns. Some of the attacks are though also detectable from the network traffic by looking for malformed packets that are oversized, fragmented or using, for example, abnormal TCP flag options. [32]

Data
This group contains an attack known as "secret" in which the attacker tries to transfer data from a legal place to a place where it does not belong. In order to detect these actions the system needs to know which files are secret. This requires a host-based IDS which would monitor actions regarding the use of these files. [32]

4.1.2 Prior Art

Maselli et al. [78] have defined global features that can be used to profitably detect network anomalies, regardless of the network infrastructure or users. They have explored different approaches to the problem of choosing the most relevant features to monitor. Their investigation combined static and dynamic traffic knowledge. Static traffic knowledge contained analysis of network security violations, IP protocol dissection, and network traffic monitoring metrics. In addition they surveyed what features network system administrators monitor. Dynamic traffic knowledge contained analysis of how the system administrators define counters and corresponding thresholds for each protocol in order to model normal network traffic and to distinguish anomalous traffic from it. A summary of their conclusions is presented with the following lists:

Monitor traffic volumes according to TCP/IP protocols:

- Number of source packets per protocol.
- Number of destination packets per protocol.
- Number of source bytes per protocol.
- Number of destination byte per protocol.

Monitor TCP session history that contains knowledge of:

- source IP address
- source port number
- destination IP address
- destination port number
- duration of the connection
- TCP window size
- TTL statistics
- amount of re-transmitted data
- fragmented packets percentage.

Monitor traffic distribution:

- Amount of local traffic vs. amount of remote traffic.
- Amount of traffic per each connection/flow per application.
- Amount of used bandwidth.

Monitor packet distribution:

- Number of packets by packet size.
- Amount of IP vs. non-IP traffic.
- Unicast vs. multicast vs. broadcast.

Although the features described by Maselli et al. [78] are on a high level, they still give ground to the conclusions that are made based on the attack analysis and comparison of used features in prior art.

4.2 Feature Subsets

Based on the available information from different sources like Knuuti's thesis [55] and Gorton's research of alert correlation [54], research on detecting intrusions in network traffic share common features that are called IP packet quintuple flow identifiers; destination address, source address, destination port, source port and protocol identifier.

Also based on the different sources like [55] and [54] it seems that an efficient IDS can be done just by using the IP packet quintuple as a basis. With the IP packet quintuple it is possible to detect most of the known anomalies or at least group them into seven commonly known groups like Lakhina et al. did in [52].

It is possible to make IDS even more precise in detecting intrusions when the basis of features is broadened with environment or monitoring specific features. For example, a combination of statistics from the network elements (their status, CPU consumption) together with user statistics (their amount, activity, etc.) and network traffic flows could improve accuracy by less false positives and negatives.

After analysing the features from the attack scenarios of view it seemed that the features used by Knuuti [55] are very similar to the features that should be monitored for each attack category. Therefore the features used by Knuuti (see Section 3.3.1) were chosen as the basis from which the subsets of features would be selected. The features to be monitored are listed in Table 4.1.

Table 4.1 Selected feature subsets

Feature	All	Knuuti	Probe	DoS	Mail server
IP address	x	x	x	x	x
timestamp	x	x	x	x	x
number of receiving sequences	x	x			
number of receiving sequences from	x	x			

different IP's					
number of sending sequences	x	x		x	
number of sending sequences to different IP's	x	x			
amount of data received	x	x		x	
amount of data sent	x	x		x	
amount of packets received	x	x		x	x
amount of packets sent	x	x		x	x
number of different port numbers used over 1024	x	x	x		
number of port numbers used over 1024	x	x	x		
number of different port numbers used below or at 1024	x	x	x		
number of port numbers used below or at 1024	x	x	x		
number of UDP flows	x	x	x	x	x
number of TCP connections	x	x	x	x	x
number of ICMP packets	x	x	x	x	x
number of SMTP connections	x				x
number of FTP connections	x				x
number of HTTP connections	x				x
number of DNS connections	x				x
number of Telnet connections	x				x
number of SSH connections	x				x

The features described in Table 4.1 are a statistical representation of the network traffic activity with a given time window. This format is also known as time series. From the anomaly detection point of view the time series are useful as they are lighter from the processing requirements point of view and they require less space in the hard drives when comparing them against the packet data.

5 EVALUATION OF THE FEATURE SUBSETS

The feature subsets are evaluated independently in two phases by first creating a model of a normal network traffic using training data. In the second phase the created model of normal network traffic is analysed against testing data. Then all the anomalous indications are analysed to find out how well the feature subsets performed.

5.1 Anomaly Detection and Feature Subset evaluation

The process from feature selection to analysis of detected anomalies is illustrated in Figure 5.1. The process consists of three main themes which are the feature subset decision making (upper group in Figure 5.1), data processing (left group in Figure 5.1) and analysis of anomaly detection (right group in Figure 5.1). The criteria to choose the feature subsets were discussed in Chapter 4. In the following sections the data processing and anomaly analysis is discussed in detail.

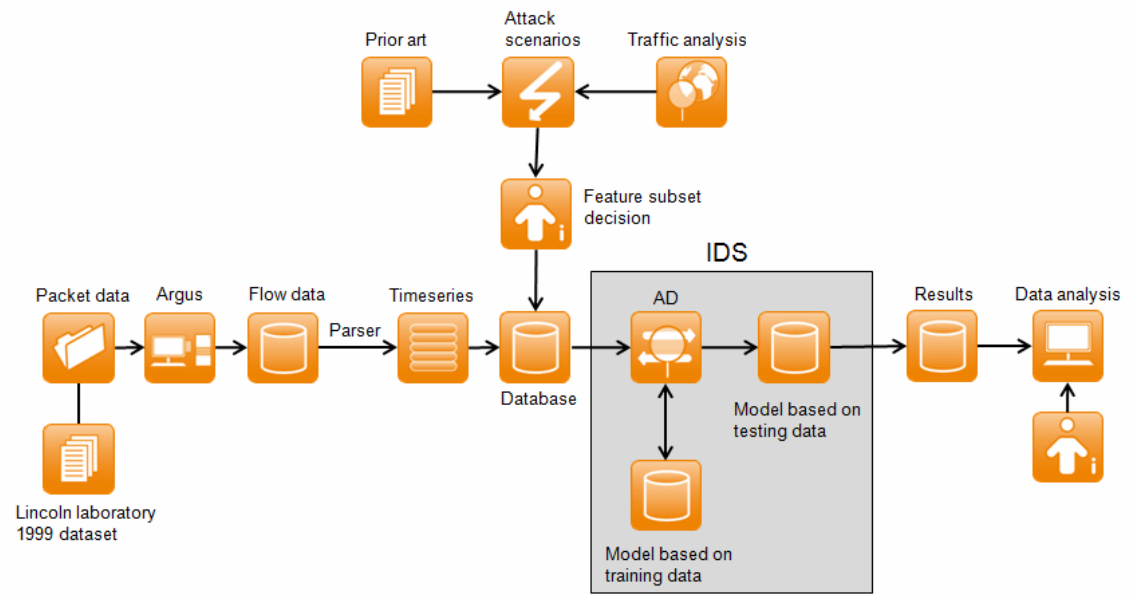

Figure 5.1 The feature subset evaluation process

5.1.1 Training and testing Data

The Lincoln laboratory 1999 dataset was chosen for the evaluation of selected feature subsets. The 1999 data in comparison to 1998 data contains attack free traffic which is crucial for creating a model of normal traffic and training the analyser with this model. In addition, the 1999 data contains some newer attack methods that are also targeting

the network services such as mail servers. Some of the attack methods used in the 1999 data are still used today so therefore it is a better option to evaluate the selected features.

The Lincoln dataset 1999 contains five weeks of network traffic data collected from the network [32]. The first three weeks in the dataset are training data and the last two are testing data. However, the second week of training data is not attack free and thus it is not used in the training phase. Using data that contains attacks could affect the model training in such way that the IDS recognise the attacks as normal traffic. In the testing phase the performance of each feature subset is evaluated against the first week (week four) of testing data. The second week of testing data (week five) contained issues such as restoring a computer from a back-up that also confused the time stamps in the testing data [32]. Week five was therefore excluded from the testing phase.

Three computers running different operating systems (Solaris, NT and Linux) were chosen from the dataset to get a wider scope in the feature subset analysis but also to reduce the amount of information that needs to be analysed in the anomaly detection phase. In addition, the number of different attacks from the attack categories was also kept in mind when choosing the computers. The selected computers and attacks against them are listed in the Table 5.1.

Table 5.1 Selected computers from Lincoln laboratory dataset and the number of attacks in each attack group. The numbers of attacks longer than 60 second in duration are presented within the brackets.

Name	IP address	Operation system	Total No. of attacks	No. of probe attacks	No. of DoS attacks	No. of attacks against the mail server
Pascal	172.16.112.50	Solaris	49 (14)	1 (0)	17 (1)	1 (1)
Hume	172.16.112.100	NT	39 (15)	9 (1)	4 (3)	0 (0)
Marx	172.16.114.50	Linux	23 (12)	3 (0)	6 (5)	4 (2)

5.1.2 Anomaly Detection Tool

Evaluating the efficiency of the features is done using an anomaly detection test bench for mobile network management (ADAI) by Kumpulainen and Hätönen [79]. The tool takes time series data as an input together with a configuration file that defines the format and variable names used in the time series. The time series are separated into two files; into day overviews and detailed files. The day files are a summary of the total number of occurrences in each day. The detailed files are time series information that represents the flow information within a specified time window.

Once the data is read, it is possible to choose a specific timeframe of interest from the preview window shown on the right in Figure 5.2. For example, it can be used to separate the training period from the testing period. More detailed description of the tool and its features are given in [79].

The anomaly detection is done in two phases. First the testing data needs to be chosen in order to create the model of the normal network traffic. This model is then used as a point of comparison in the second phase when the testing data is analysed.

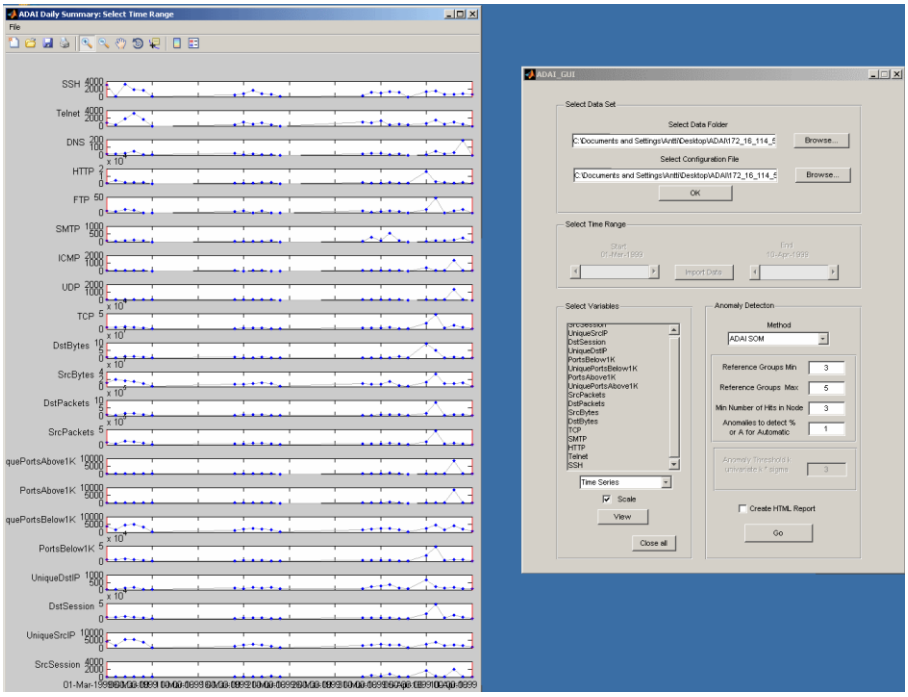

Figure 5.2 ADAI GUI

ADAI supports different anomaly detection algorithms from which a local anomaly detection method was chosen for the feature evaluation. [79]

5.1.3 Anomaly Detection Method

Local anomaly detection method is an improvement of a global AD method [50]. The method combines K-means clustering with Kohonen's [80] self-organising maps (SOM) to detect anomalies.

K-means clustering is an algorithm that classifies data set to a certain K number of clusters. Each cluster has a centroid and the data is classified by the distance from a centroid. Each data point is classified to the cluster with the closest centroid [81]. Self-organising map is a neural network tool for mapping high-dimensional data into one- or two- dimensional map that can be visualised [80].

Kumpulainen and Hätönen [82] improved the anomaly detection method by using local thresholds instead of global thresholds. As a result of this improvement the amount of false positives were reduced. The idea and comparison of global and local thresholds is illustrated in Figure 5.3.

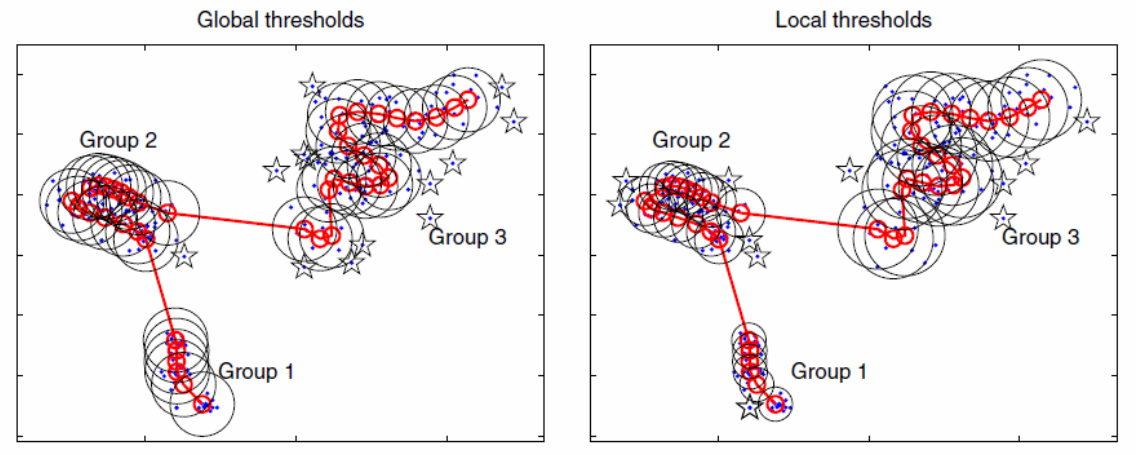

Figure 5.3 Anomaly detection using global and local thresholds [83].

In general the local anomaly detection method creates a map of the data, groups the neurons and calculates the local variances within the neuron groups. All the data points that are far from the neuron groups are marked as anomalies. In Figure 5.3 these data points are marked by stars. [83]

ADAI with the local anomaly detection method calculates anomalous events from the time series and gives a list of them as an output. The tool supports exporting of anomalies into a file for further evaluation. In addition, the tool can plot time series figure together with the detected anomalies. The tool can also plot figures that show the distribution of anomalies according to the day of the week, time of the day; how the anomalies are grouped and how scattered the data is within the groups. These figures give additional information when analysing the anomalies. [79] However, only the anomaly exporting functionality is used in this thesis to analyse the features.

The output is a list of all the events that are detected as anomalous. All the events contain the timestamp, level of anomaly which gives estimation on the seriousness of the detected anomaly and in addition the events contain top three features that are contributing most to the anomaly. [79] These anomaly lists are analysed against the information on the attacks (starting time and duration) given by Lincoln laboratory [32].

The used version of the tool (0.81) uses a size of SOM that is hard coded in to the program. This has some disadvantages when creating a model of the normal network traffic of two weeks. Originally the tool was designed to be used with a specific amount of data which was far less than the amount of training data. As a result the processing requirements became too high to handle when using a time window size of 5 seconds. Therefore a 60 second time window size was used instead to overcome this limitation.

5.2 Preparing the Data

As the anomaly detection tool requires time series data as an input the packet data needs to be converted. First the packet data is converted into flow-based traffic data from which the time series can be extracted. The data conversion process is described in the following sections.

5.2.1 Pre-processing the Data

Each week in the Lincoln dataset is divided into five day files which are from Monday to Friday. Each day is a tcpdump capture and therefore they are in tcpdump format. These capture files contain non-IP-based traffic such as link layer messages and others that do not have an IP address. In the evaluation of feature subsets the focus is on IP-based traffic and therefore all non-IP based traffic needs to be filtered out before further processing the data. The filtering is done by using tcpdump's own filtering options. In general the capture files are read using tcpdump with the following command:

```
tcpdump -r  file.tcpdump  ip -w ip_only.tcpdump
```

5.2.2 Packet Data into Flow Data

After the filtering the capture files contain only IP-based traffic and it can be further processed into flow data. This is done using Argus-server. Argus takes the capture files as an input and converts the packet data into bi-directional flow data. This is done using the following command:

```
argus  -r ip_only.tcpdump -w flow.argus
```

The output file of the Argus-server is in argus-format that contains all the flow information collected from the packet data. In order to process the argus-based data it needs to be read using Argus-client, Ra (read Argus), that comes with the Argus installation. The Ra-function works in similar way as did the Argus-server. It takes as an input the argus-based data and either prints the output on the screen or into a specified file.

The features required in the anomaly detection affects the flow features that need to be read from the flow data. The general idea is to choose flow features that contain valuable information on the network traffic behaviour. These flow features are used to create a model of the normal network traffic and therefore they should represent the network traffic as well as possible. In this case the output is saved into a comma separated value (CSV) file using the following command:

```
ra -u -nr flow.argus -c ";" -s stime proto saddr sport spkts sbytes daddr dport dpkts dbytes > ./file.csv;
```

The output of the Ra in this case is in csv-format that contains the following features:

- starting time of the flow in unix time format
- protocol (TCP, UDP or ICMP)
- source IP-address
- source port number
- number of packets sent by the source

- amount of bytes sent by the source
- destination IP-address
- destination port number
- number of packets received by the destination

All of the above mentioned operations need to be done for every capture file. In the end there are five csv-formatted day files for each week that contain the flow information. To make things easier the csv-files can be concatenated using the following command:

```
cat day1.csv day2.csv day3.csv day4.csv day5.csv > week.csv
```

The whole process is automated with a shell-script presented in Appendix 4.

5.2.3 Feature extraction

In order to extract the features defined in Section 4.2 the csv-formatted flow data need to be parsed. The parser presented in Appendix 5, checks the flow data using a pre-defined time window and creates time series. An event in the time series represents the flow information within the time set by the time window.

The parser is a modified version of Knuuti's parser [55, pp. 63-65]. Knuuti's parser has been an excellent basis for the flow data parser. When Knuuti's parser was created to choose a specific IP-address range, the parser in Appendix 5 is taking into account all the IPs in the flow data. In addition to the features collected by Knuuti the parser used in this thesis collects information on the used services (SMTP, FTP, SSH, Telnet, DNS and HTTP) as well.

As a result the parser creates time series of the 23 features described in Section 4.2. These features are used as a basis in the analysis. The subsets of features are selected from this list according to the categories discussed in Section 4.2 that are used in the training and testing phases of the anomaly detection.

6 RESULTS

The feature subsets are evaluated against each other and thus the evaluation is not done against the prior art. One reason for this is because the prior art uses the dataset from 1998 and in this thesis the data from 1999 is used instead. Therefore the results are not comparable with other studies such as KDD CUP 99 discussed in Section 3.3.1.

Expectation was that by using the feature subsets it is possible to detect attacks from the specific attack categories within the data. In addition, the performance is expected to be better with the feature subsets in comparison to the use of all the features. It was also expected that the chosen time window size (60 seconds) will affect negatively to the detection of attacks whose duration is less than the window size. For example, the duration of most of the probing attacks is one to three seconds. It is therefore expected that most of these short period attacks might not be detectable. The results in the following sections are illustrated based on the data in Appendix 6.

6.1 Detected Attacks

Results of the IDS' performance are discussed in the following sections. Taking into account the expectations the results are divided in the following manner. In Sections 6.1.1 and 6.1.2 are the detection rate results of attacks from all of the five attack categories in the testing data. In Sections 6.1.3 and 6.1.4 are the detection rate results of all attacks from the selected attack categories. The selected attack categories are probing attacks, denial of service attacks and attacks against the mail server.

In Sections 6.1.5, 6.1.6 and 6.1.7 are the detection rate results of attacks from each selected attack categories using each feature subset.

6.1.1 Detection Rates of Attacks

Results on the detection rate of all the attacks against each computer with the feature subsets are presented in Figure 6.1. The overall detection rates are between 10 and 30 percent. These results are more or less what were anticipated as most of the attacks are short in duration and the attacks from U2R, R2L and Data categories are greater in number when comparing them against probing and DoS attack amounts. The number of attacks for Solaris is (49), for NT (39) and for Linux (23) (see Table 5.1 in Section 5.1.1).

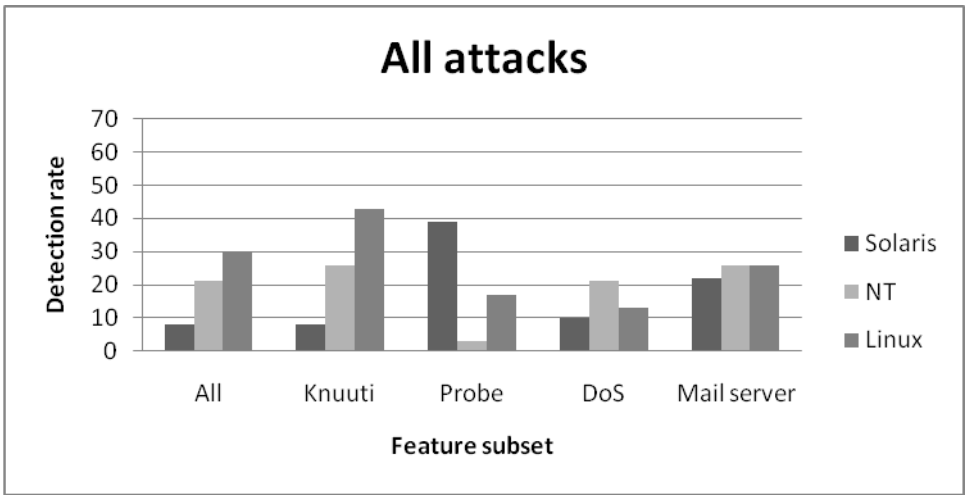

Figure 6.1 Detection rates of all attacks

Some of the interesting findings from the results in Figure 6.1 are that when comparing operating systems and the detection rates with the feature subsets it is noticeable how the attacks and their effect to the network traffic is significantly different. For example, attacks against the Solaris computer are best detected using the probe feature subset but NT probe subset's performance is the worst. It seems that the detection rates with the NT and Linux are opposite of what was achieved with the Solaris. The attacks against NT and Linux are best detected using the Knuuti feature subset.

6.1.2 Detection Rates of Attacks longer than 60 Seconds in Duration

The detection rates of all attacks which were 60 seconds or longer in duration are illustrated in Figure 6.2. The overall detection rate is far better when comparing them with the detection rates of all the attacks in Figure 6.1. The number of attacks for Solaris is (14), for NT (15) and for Linux (12) (see Table 5.1 in Section 5.1.1).

On average the results are between 30 to 50 percent. It was expected that when taking into account only the attacks which duration is longer than 60 seconds, the detection rate would also be better. Most of the attacks that are longer than 60 seconds are from the DoS attack category but there are also attacks from all of the other categories as well.

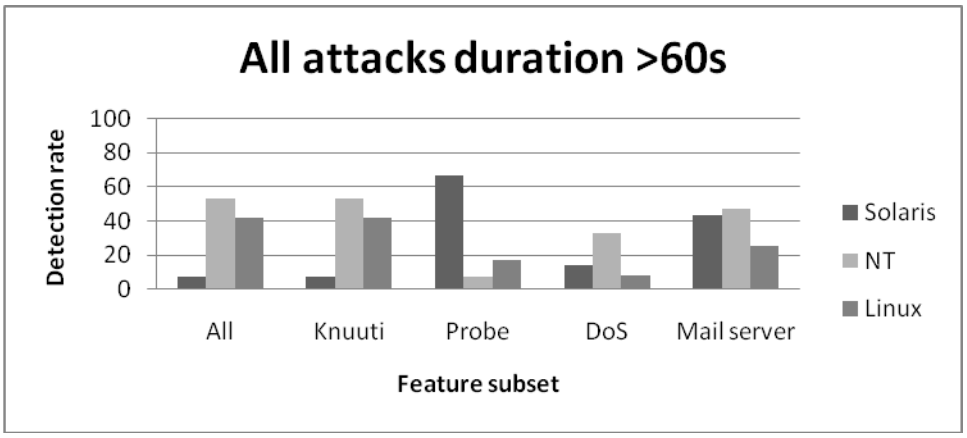

Figure 6.2 Detection rates of attacks longer than 60 seconds in duration

From the Figure 6.2 it can be seen that the best performed feature subsets are the same as was in Figure 6.1. However the difference between the feature subsets for NT and Linux is not huge when detecting attacks that are longer than 60 seconds in duration. When looking the result for the NT computer it seems that all and Knuuti feature subsets are equally good in detecting attacks with a detection rate of 53%. The same is valid with the attacks against the Linux computer. The all and Knuuti feature subsets achieved both a detection rate of 42%

6.1.3 Detection Rates of Selected Attacks

When taken into account the fact that attacks from the U2R, R2L and Data categories are not detectable using the defined features the results are somewhat different in comparison to the results presented in Figures 6.1 and 6.2. The detection rates of attacks from the DoS and Probe categories are summarised in Figure 6.3. The detection rates of attacks from the selected categories longer than 60 seconds in duration are summarised in Figure 6.4. The number of attacks for Solaris is (18), for NT (13) and for Linux (9) (see Table 5.1 in Section 5.1.1).

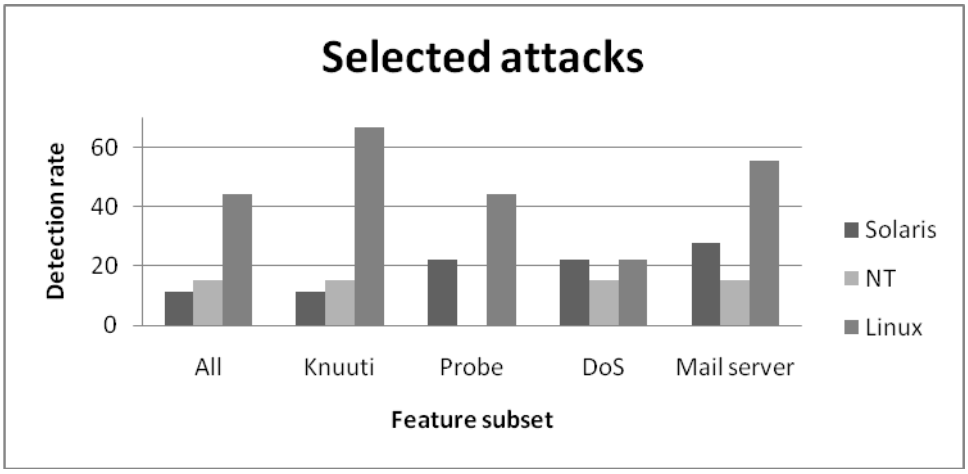

Figure 6.3 Detection rates of selected attacks

From the Figure 6.3 it can be seen that the detection rate of attacks against the Solaris computer is better with every feature subset except with the probe features. It seems that most of the attacks detected with the probe subset are from the three unlikely detectable attack categories. The same can be said with the NT computer as the detection rate is lower when only the selected attack categories are taken into account.

However the detection rates of attacks against the Linux computer are approximately 10% higher than when taking into account all the five attack categories. The biggest improvement was achieved with the Knuuti feature subset whose detection rate rose almost 25% from the results in Figure 6.1.

6.1.4 Detection Rates of Selected Attacks longer than 60 Seconds in Duration

Results shown in Figure 6.4 are containing only attacks from the selected attack categories and in addition the ones that are longer than 60 seconds in duration. These results should be according to the expectations. At first glance the results are better than shown in Figure 6.1. However, the number of selected attacks for each computer is far less than when taking into account all the attacks. The number of attacks for Solaris is (1), for NT (4) and for Linux (5) (see Table 5.1 in Section 5.1.1).

When looking the results, it seems that the smaller feature subsets are able to detect the only one selected attack for Solaris computer. It might be that, the effect of this attack gets mixed up in the mass of other features and is not therefore detectable when using multiple features.

Half of the attacks against the NT computer were detectable with each feature subset except with the probe features. This result is interesting as one of the four attacks is from the probe attack category. Therefore, this feature subset did not perform well.

When looking the results for the Linux computer, it seems that the DoS features are not performing well. The all, Knuuti and mail server features on the other hand are detecting 60% of the selected attacks. The most interesting results are achieved with the probe features. This result is interesting as there were no probe attacks among the selected attacks and therefore the ones that were detected, were from other categories.

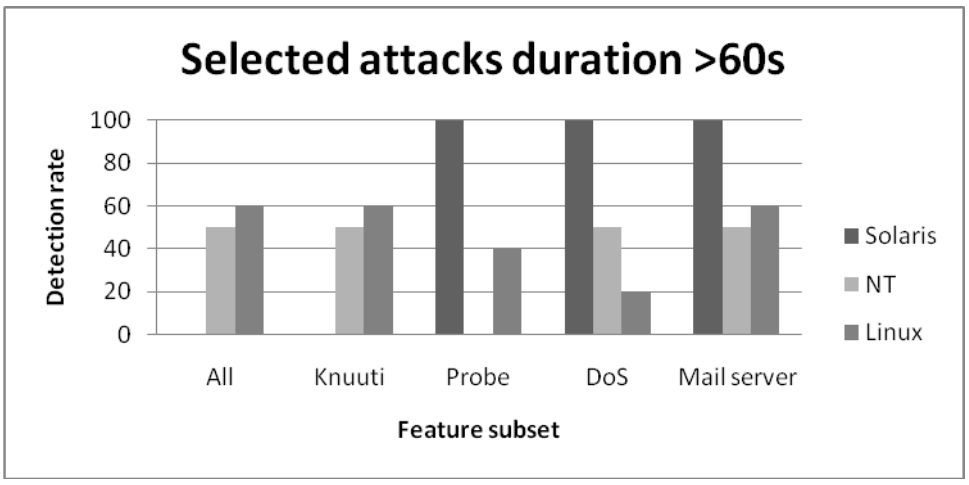

Figure 6.4 Detection rates of selected attacks longer than 60 seconds in duration

6.1.5 Probe Attacks

The results of detecting attacks from the Probe category are illustrated in Figure 6.5. The detection rate of probe attacks against the Solaris computer are 100% with the probe, DoS and mail server feature subsets. In total there was only a single probe attack against the Solaris computer. However the probing lasted only one second which still was detected against the expectation with the three mentioned feature subsets.

The probing attacks against the NT computer were almost completely passing the detection. The attacks were not detected with the probe feature subset which was again against the expectations. The other subsets were able to detect one of the probing attacks which was the only one that lasted longer than 60 seconds.

With the Linux computer the results were against the expectations as the Knuuti feature subset outperformed the probe subset. This shows again the difference between operating systems and the attacks against them and how they affect to the features.

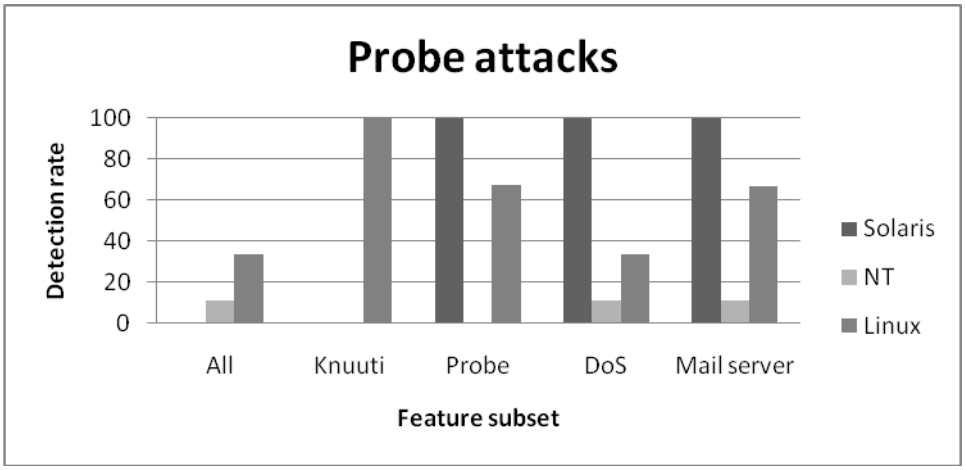

Figure 6.5 Detection rates of Probe attacks

6.1.6 DoS Attacks

The results of detecting attacks from the DoS category are illustrated in Figure 6.6. DoS attacks against the Solaris computer are best detected using the DoS feature subset which was an expected result. However, the detection rate with the probe and mail server subsets is equally good.

The detection rate of DoS attacks against the NT computer is equal between the feature subsets. It is interesting that for some reason the DoS feature subset is not better than the other subsets. One reason for this result is that the DoS attacks against the NT that were detected cause significant changes to most of the network traffic features which then are also detectable with the other feature subsets.

The detection rate of DoS attacks against the Linux computer is completely against the expected results. The DoS feature subset performs the worst in comparison to the other subsets. It seems that the DoS attacks against Linux computers are causing changes into totally different features than the attacks against the NT and Solaris.

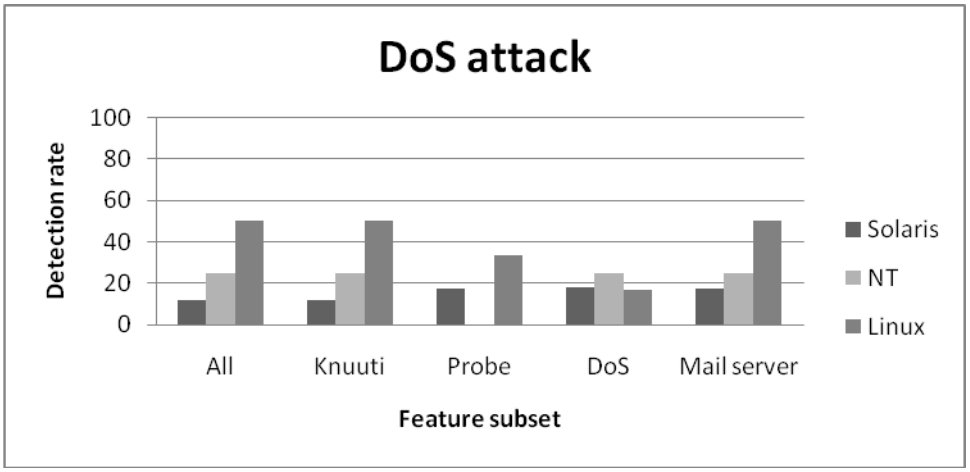

Figure 6.6 Detection rates of DoS attacks

6.1.7 Attacks against the mail server

The results of detecting attacks against the mail server are illustrated in Figure 6.7. It should be noted that there were no attacks against the NT and therefore it is not shown in the Figure 6.7.

The overall results are more or less according to the expectations as the smaller feature subsets were able to detect the attacks better than when using all of the features or the Knuuti features. When taking into account that when using fewer features the processing requirements are also smaller than when using a larger set of information. From this perspective the results were very good. The interesting thing though in these results is that the probe and Dos feature subsets were as good as the mail server features with the attacks against the Solaris computer. With the Linux computer all of the subsets except the DoS subset performed equally well.

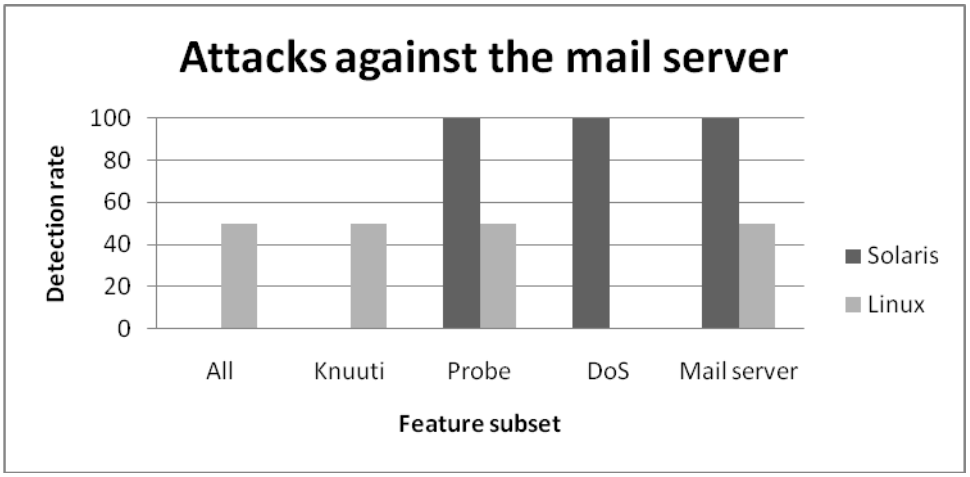

Figure 6.7 Detection rates of attacks against the mail server

6.2 True Positives and False Positives

The rate of detected anomalies that corresponds to an actual attack (true positives) is illustrated in Figure 6.8. It should be taken into account that even normal network traffic contains changes that can be detected as an anomalous behaviour.

From Figure 6.8 it can be seen that the network traffic to the Linux computer the local anomaly detection method detected more true positives than false positives with the probe and mail server feature subsets. The probe feature subset detected from the network traffic to the NT computer more false positives than true positives. With the Linux computer the DoS feature subset has similar results.

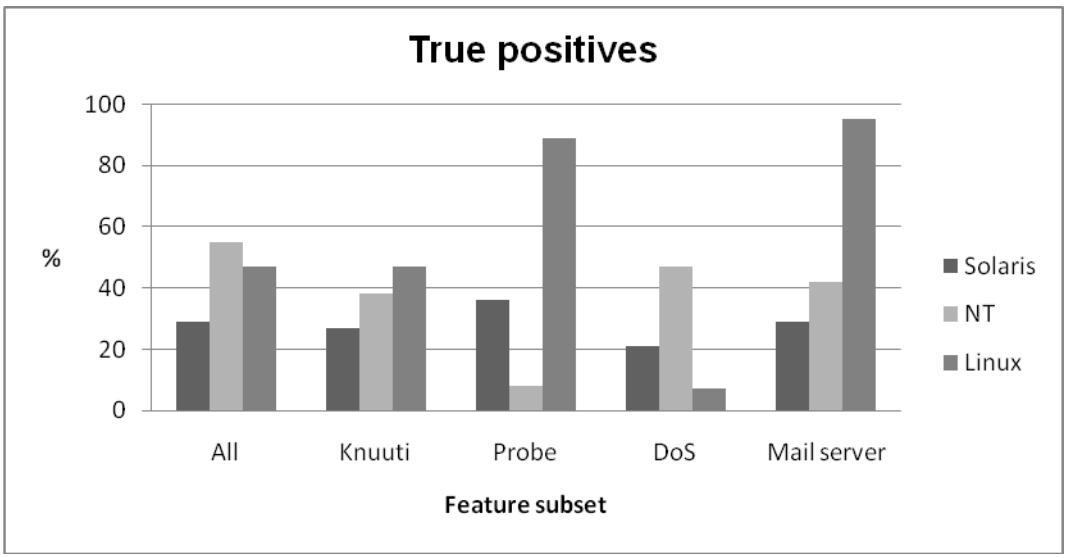

Figure 6.8 Rates of true positives with feature subsets

In Figure 6.9, 6.10 and 6.11 is shown the number of false positives in comparison to the number of true positives detected from the network traffics, to each computer with the feature subsets. With the NT and Linux computers the results are according to the expectations that when using more features it will also cause more false positives.

The Solaris however, gave the opposite results. With smaller feature subsets, the number of false positives is far greater, than when using the all features or Knuuti features.

As a conclusion it is clear that more investigation of differences between operating systems and attacks against them need to be done in order to find out more suitable set of features. Although there were huge differences in the results, they were still more or less according to the expectations. The results can be thought of as an encouragement, that it is possible to use smaller feature groups to detect specific attack categories with less processing requirements.

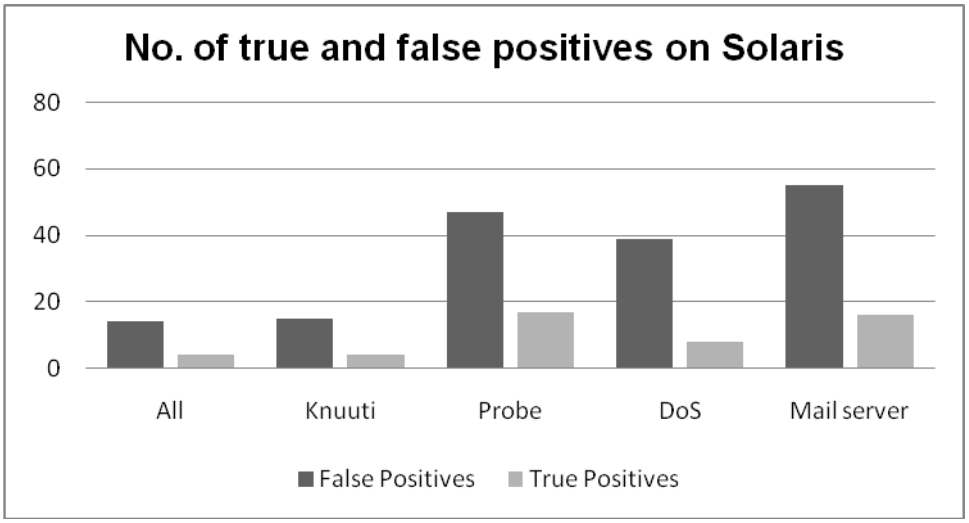

Figure 6.9 Comparison of the number of false and true positives that were detected from the network traffic to the Solaris computer

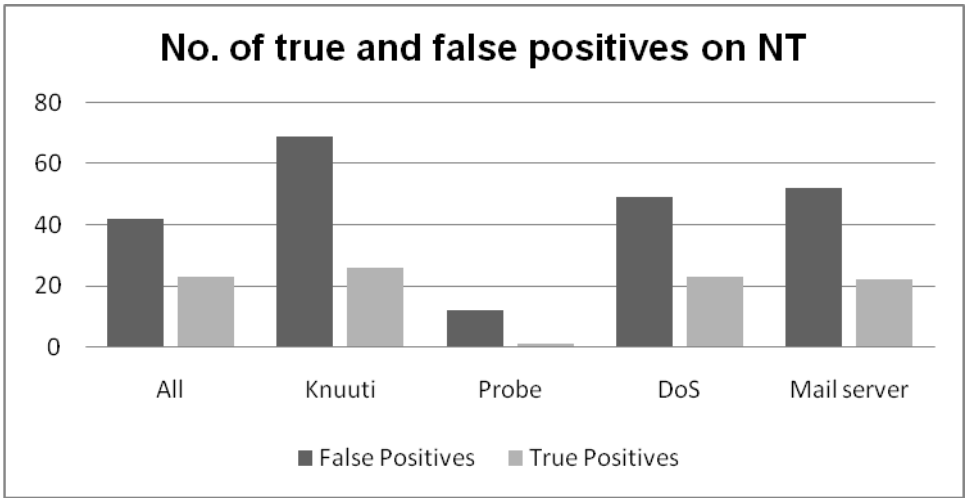

Figure 6.10 Comparison of the number of false and true positives that were detected from the network traffic to the NT computer

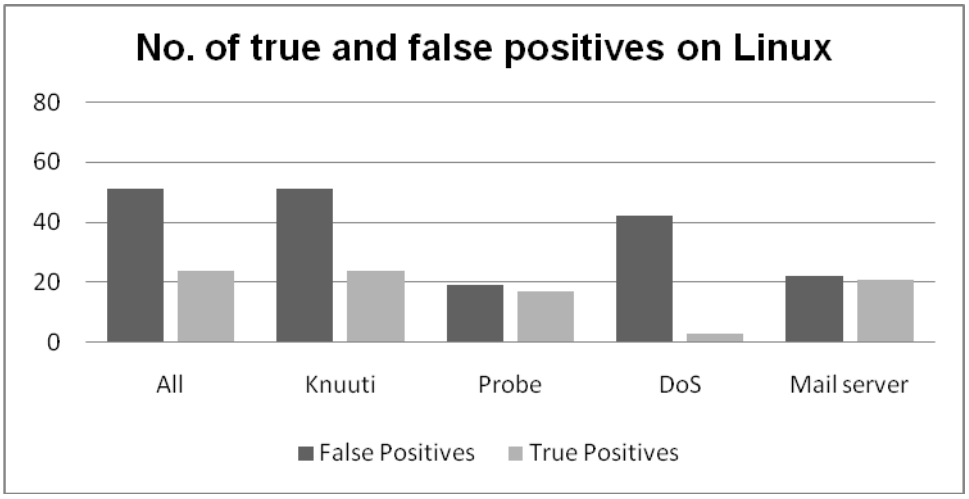

Figure 6.11 Comparison of the number of false and true positives that were detected from the network traffic to the Linux computer

7 CONCLUSIONS

The scope of this thesis was to find suitable subsets of features for the selected attack categories within the Lincoln laboratory dataset. The feature subsets were formed using prior knowledge from other IDS researches and in addition, the attacks and their effect to the network traffic was analysed to decide which features should be used in the anomaly detection.

The results (see Chapter 6) show that it is possible to use smaller subsets of features to find intrusion in the data monitored. Taking into account all the factors which affect to the results, the outcome was good, with 40-60% detection rate with most of the feature subsets (see Figure 6.4). Also the number of false positives was reduced with the smaller feature subsets with the Linux and NT computer. Although the results with the Solaris computer were completely reversed, the results can still be taken as a good sign that it is possible to ease the workload of the network administrator by detecting less false positives.

However, as was already discussed in Chapter 6, more investigation is still needed to achieve better results in anomaly detection. It is clear that in some cases the results are completely against expectations. To find out why, more research on this area is required. Also testing of the proposed feature subsets should be done using a smaller time window. If, for example, the time window would be five seconds, it should be theoretically possible to detect also the shorter attacks. Probing attacks are a good example of such short duration attacks.

Furthermore, the anomaly detection tool was used in default settings and with only one anomaly detection method. As the scope of this thesis was to evaluate the performance of various feature subsets, it was therefore decided, that the method is not relevant from the feature evaluation point of view and thus only one method was used. It seems though, that the method also plays a significant role in the detection performance. For example, the local anomaly detection method allows the user to define the number of clusters and the thresholds to be used in the detection phase. By testing different number of clusters for each attack categories, it might have been possible to achieve better results.

Analysis of modern attacks is also required as the attacks are becoming more sophisticated, but in the same time more difficult to find out. An excellent example of this is the Stuxnet virus discussed in Section 2.1.2. Another criterion in finding modern attacks is to use network traffic from the live networks. Especially when the IDS is supposed to work in telecommunications networks the data should be also collected from such network.

REFERENCES

[1] Lassila, A. Sonera korvaa lopetettavat lankaverkot 3g:llä ja Digitan @450-verkolla, HS.fi. [WWW]. [Cited 2011-02-27]. Available at: http://www.hs.fi/talous/artikkeli/Sonera+korvaa+lopetettavat+lankaverkot+3gll%C3%A4+ja+Digitan+450-verkolla/1135234793887 (in finnish)

[2] Lehto, T. Tietokone.fi. [WWW]. [Cited 2011-02-27]. Available at: http://www.tietokone.fi/uutiset/2008/sonera_sulkee_19_000_adsl_liittymaa (in finnish)

[3] L 22.12.2009/1186, Laki laajakaistarakentamisen tuesta haja-asutusalueilla. (Law for supporting broadband development in rural areas). (in finnish)

[4] 3GPP TS 23.401 V10.2.1, 3rd Generation Partnership Project; Technical Specification Group Services and Systems Aspect; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (Release 10). 3GPP, 2011. Technical Specification.

[5] ZTE, Global GSM Incremental Market Analysis. [WWW]. [Cited 2010-11-03]. Available at: http://wwwen.zte.com.cn/endata/magazine/ztetechnologies/2010/no4

[6] 3GPP TS 23.402 V9.4.0, 3rd Generation Partnership Project; Technical Specification Groups Services and System Aspects; Architecture enhancements for non-3GPP accesses (Release 9). 3GPP, 2010. Technical Specification.

[7] CERT Coordination Center, Vulnerability Discovery: Bridging the Gap Between Analysis and Engineering. [PDF]. [Cited 2010-11-16]. Available at: http://www.cert.org/archive/pdf/CERTCC_Vulnerability_Discovery.pdf

[8] McAfee Labs, McAfee Threats Report: Third Quarter 2010, [WWW]. [Cited 2010-11-19]. Available at: http://www.mcafee.com/us/threat_center/white_paper.html

[9] Cisco-1, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2009-2014, [WWW]. [Cited 2010-12-15]. Available at: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html

[10] Sundaram, A. An introduction to intrusion detection, Crossroads, Volume.2, Issue 4, pp. 3-7, April 1996

[11] NSA, National Security Agency. Defence in Depth. [PDF]. [Cited 2010-11-16]. Available at: http://www.nsa.gov/ia/_files/support/defenseindepth.pdf

[12] Fogla, P., Lee, W. Evading network anomaly detection systems: formal reasoning and practical techniques, Proceedings of the 13th ACM conference on Computer and communications security, pp. 59-68, Alexandria, Virginia, USA, 2006

[13] Gates, C., Taylor, C., Challenging the anomaly detection paradigm: a provocative discussion. In Proc. of ACM Workshop on New Security Paradigms 2006, Schloss Dagstuhl, Germany, September 2006.

[14] Denning, D. E., An intrusion-detection model, IEEE Transactions on Software Engineering, Volume 13, Issue 2, pp. 222-232, February 1987

[15] Javitz, H.S., Valdes, A. The SRI IDES Statistical Anomaly Detector, In Proceedings of the IEEE Symposium on Security and Privacy, pp. 316-326, May 1991

[16] Chan, P., Mahoney, M., Arshad, M. A Machine Learning Approach to Anomaly Detection, Department of Computer Sciences, Florida Institute of Technology, Melbourne, 2003

[17] Wang, K., Stolfo, S. J. Anomalous Payload-based Intrusion Detection, Computer Science Department, Columbia University, New York, 2004

[18] Das, K. Protocol Anomaly Detection for Network-based Intrusion Detection, SANS Institute, GSEC Practical Assignment Version 1.2f, 2001

[19] Staniford-Chen, S. et al. GrIDS-A graph based intrusion detection system for large networks, Department of Computre Science, University of California, Davis, 1996

[20] Fontugne, R., Hirotsu, T., Fukuda, K. An image processing approach to traffic anomaly detection, Proceedings of the 4th Asian Conference on Internet Engineering, pp. 17-26, November 2008, Pratunam, Bangkok, Thailand

[21] Thottan, M., Ji, C. Anomaly Detection in IP Networks. IEEE Trans. Signal Processing (Special issue of Signal Processing in Networking), pp. 2191–2204, August 2003

[22] Lee, W., Stolfo, S. J. Data Mining Approaches for Intrusion Detection, Proceedings of the 7th USENIX Security Symposium, pp. 26-29, San Antonio, Texas, January 1998

[23] Anderson, J.P. Computer Security Threat Monitoring and Surveillance. Technical report, Fort Washington, Pennsylvania, April 1980

[24] Axelsson, S. Intrusion detection systems: a survey and taxonomy. Technical report, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, March 2000

[25] Snort, Snort homepage, [WWW]. [Cited 2010-11-02]. Available at: http://www.snort.org/

[26] Sourcefire IPS, Sourcefire homepages, [WWW]. [Cited 2010-10-28]. Available at: http://www.sourcefire.com/content/next-generation-intrusion-prevention-system-ngips

[27] CERIAS, The center for education and research in information assurance and security. Autonomous Agents for Intrusion Detection (AAFID). [WWW]. [Cited 2010-10-25]. Available at: http://www.cerias.purdue.edu/about/history/coast/projects/aafid.php

[28] CIDF, Common intrusion detection framework project page. [WWW]. [Cited 2010-10-25]. Available at: http://gost.isi.edu/cidf/

[29] Abraham, A., Jain, R., Thomas, J., Han, S.Y. D-SCIDS: Distributed soft computing intrusion detection system, In Journal of Network and Computer Applications, Volume 30, Issue 1, pp. 81-98, January 2007

[30] SRI International, SRI International project page, [WWW]. [Cited 2010-10-27]. Available at: http://www.csl.sri.com/projects/

[31] Spitfire, Open channel foundation project page, [WWW]. [Cited 2010-10-27]. Available at: http://www.openchannelsoftware.com/projects/Spitfire/

[32] Massachusetts Institute of Technology (MIT), Lincoln laboratory, Cyber systems and technology. [WWW]. [Cited 2010-11-11]. Available at: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html

[33] Lu, W., Ghorbani, A.A. Network anomaly detection based on wavelet analysis, EURASIP Journal on Advances in Signal Processing, pp.1-16, January 2009

[34] KDD cup 1999, KDD cup 1999 data distribution page, [WWW]. [Cited 2010-12-07]. Available at: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[35] Schulze, H., Mochalski, K. Ipoque internet study 2008-2009. [PDF]. [Cited 2010-11-11]. Available at: http://www.ipoque.com/userfiles/file/ipoque-Internet-Study-08-09.pdf

[36] Cisco-2, Cisco IDS Sensor Deployment Considerations. [WWW]. [Cited 2010-11-17]. Available at: http://www.ciscopress.com/articles/article.asp?p=25327

[37] Miettinen M., Halonen P., Hätönen K. Host-based intrusion detection for advanced mobile devices, AINA ’06: proceedings of the 20th international conference on advanced information networking and applications, Volume 2 (AINA’06). IEEE Computer Society, Washington, DC, pp. 72–76, 2006

[38] Handley, C. M., Paxon, V. Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics. In Proceedings of the 10th USENIX Security Symposium, Washington, DC, August 2001

[39] Zainal, A., Maarof, M.A., Shamsuddin, S.M. Features Selection Using Rough-PSO in Anomaly Intrusion Detection, Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia

[40] Mukkamala, S., Sung, AH. Feature selection for intrusion detection using neural networks and support vector machines. J Transport Res Board Natl Acad, Transport Res Record No 1822 2003; 33-9.

[41] Chebrolu, S., Abraham, A., Thomas, JP. Feature Deduction and Ensemble Design of Intrusion Detection Systmes, Journal of Computers and Security. Volume 24, Issue 4, pp. 295-307, 2005

[42] Al-Sharafat, W.S., Naoum, R. Significant of features selection for detecting network intrusions, Internet Technology and Secured Transactions, 2009. ICITST 2009, Volume, pp.1-6, 9-12 Nov. 2009

[43] Ben-Gal I. Bayesian Networks, in Ruggeri F., Faltin F. & Kenett R. Encyclopedia of Statistics in Quality & Reliability, Wiley & Sons, 2007

[44] Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J. Classification and regression trees, Monterey, California, 1984

[45] Jolliffe, I.T. Principal Component Analysis, second edition, Springer-Verlag, New York, 2002

[46] L 16.6.2004/516 Sähköisen viestinnän tietosuojalaki. (Data protection law). (in finnish)

[47] Cisco-3, Cisco NetFlow. [WWW]. [Cited 2010-11-17]. Available at: http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html.

[48] QoSient, Argus - Auditing Network Activity. [WWW]. [Cited 2010-11-17]. Available at: http://www.qosient.com/argus/.

[49] Case, J. et al., A Simple Network Management Protocol. RFC 1098. Network Working Group, IETF, 1989..

[50] Höglund, A. An anomaly detection system for computer networks, Master's thesis, Helsinki University of Technology, 1997

[51] Kent, K., Souppaya, M. Guide to Computer Security Log Management, Recommendations of the National Institute of Standards and Technology (NIST), September 2006

[52] Lakhina, A., Crovella, M., Diot, C. Mining anomalies using traffic feature distributions, Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 22-26, Philadelphia, Pennsylvania, USA, August 2005

[53] Dewaeke, G. et al. Extracting hidden anomalies using sketch and non gaussian multiresolution statistical detection procedures, LSAD '07, 2007. pp. 145-152

[54] Gorton, D. Extending Intrusion Detection with Alert Correlation and Intrusion Tolerance, Thesis for the Degree of Licentiate of Engineering, Chalmers University of Technology, Göteborg, Sweden 2003

[55] Knuuti, O. Intrusion detection system comparison in large IP-networks, Master's thesis, Tampere University of Technology, 2009

[56] Sung, AH., Mukkala, S. The Feature Selection and Intrusion Detection Problems, Proceedings of Advances in Computer Science – ASIAN 2004: Higher-Level Decision Making, 9th Asian Computing Science Conference, Volume 3321, pp. 468-482, 2004

[57] Kabiri, P., Zargar, G. R. Category-Based Selection of Effective Parameters for Intrusion Detection, International Journal of Computer Science and Network Security (IJCSNS), Volume 9, No. 9, pp. 181-188, 2009

[58] Lin, Y., Fang, B.-X., Guo, L., Chen, Y. TCM-KNN Algorithm for Supervised Network Intrusion Detection, Intelligence and Security Informatics, In proceedings of Pacific Asia Workshop (PAISI 2007), LNCS 4430, pp. 141-151, Chengdu, China, April 2007

[59] Lawrence Berkeley National Laboratory, Bro Intrusion Detection System. [WWW]. [Cited 2011-02-27]. Available at: http://bro-ids.org/

[60] Zargar, G.R., Kabiri, P. Identification of effective network features for probing attack detection, Networked Digital Technologies, 2009. NDT '09. First International Conference on Networked Digital Technologies (NDT 2009), VSB-Technical University of Ostrava, Czech Republic, pp. 405-410, 2009

[61] Zargar, G. R., Kabiri, P. Identification of Effective Network Features to Detect Smurf Attacks, Proceedings of 2009 Student Conference on Research and Development (SCOReD 2009), pp. 49-52, UPM Serdang, Malaysia, 2009

[62] Carrascal, A., Couchet, J., Ferreira, E., Manrique, D. Anomaly Detection using

prior knowledge: application to TCP/IP traffic, In Artificial Intelligence in Theory and Practice, pp. 139-148, 2006

[63] Lee, D. C. et al. Fast Traffic Anomalies Detection Using SNMP MIB Correlation Analysis. In proceedings of International Conference on Advanced Communication Technology (ICACT), 2009

[64] Cisco-4, Cisco SNMP object navigator. [WWW]. [Cited 2010-10-19]. Available at: http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en

[65] Schmidt, A-D. et al. Monitoring smart phones for anomaly detection, Mobile Networks and Applications, Volume 14, Issue.1, pp. 92-106, February 2009

[66] Huang, Y.-A. et al. Cross-Feature Analysis for Detecting Ad-Hoc Routing Anomalies. Providence RI, In proceedings of The 23rd International Conference on Distributed Computing Systems (ICDCS), 2003

[67] Huang, Y., Lee, W. A Cooperative Intrusion Detection System for Ad Hoc Networks, In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03), Fairfax VA, October 2003

[68] Wang, X., Lin, T.-L., Wong, J. Feature Selection in Intrusion Detection System over Mobile Ad hoc Network, Technical Report, Computer Science Department, Iowa State University, 2005

[69] Depren, O. et al. An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. 4, Elsevier, Expert Systems with Applications, Vol. 29, pp. 713-722, November 2005

[70] CERT Advisory CA-97.28. Teardrop Land. CERT, December 1997

[71] CERT Advisory CA-96.26. Ping of Death. CERT, December 1996

[72] Targa3, Targa3 source code. [WWW]. [Cited 2010-10-19]. Available at: http://mixter.void.ru/targa3.c.

[73] CERT Advisory CA-98.01. Smurf. CERT, January 1998

[74] CERT Advisory CA-96.21. TCP SYN flooding and IP spoofing attack. CERT, November 1996

[75] Jung, J., Krishnamurthy, B., Rabinovich, M. Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites. Honolulu, AT&T Labs-Research, 2002

[76] Etutorials.org, TCP Port Scanning. [WWW]. [Cited 2011-01-03]. Available at: http://etutorials.org/Networking/network+security+assessment/Chapter+4.+IP+Network+Scanning/4.2+TCP+Port+Scanning/

[77] CERT Advisory CA-95.06. Security Administrator Tool for Analyzing Networks (SATAN). CERT, April, 1995

[78] Maselli, G., Deri, L., Suin, S. Design and Implementation of an Anomaly Detection System: an Empirical Approach. In proceedings of Terena Networking Conference, 2003

[79] Kumpulainen, P., Hätönen, K. Anomaly Detection Algorithm Test Bench for Mobile Network Management, In proceedings of MathWorks Matlab User Conference Nordic, Stockholm, November, 2008

[80] Kohonen, T. The Self-Organizing Map, Proc. IEEE, Volume 78, No. 9, pp. 1464-1480, 1990

[81] MacQueen, J. B. Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297, Berkeley, University of California Press,1967

[82] Kumpulainen, P., Hätönen, K. Local Anomaly Detection for Network System Log Monitoring, Proceedings of the 10th International Conference on Engineering Applications of Neural Networks, pp. 34-44, 2007

[83] Kumpulainen, P., Hätönen, K. Local anomaly detection for mobile network monitoring, Information Sciences, Elsevier. Volume 178, Issue (No.) 20, pp. 3840-3859, 15 October 2008

APPENDIX 1 NETWORK TRAFFIC HEADER FIELDS

No.	Feature	Description
1	Protocol	Type of Protocol
2	Frame_lenght	Length of Frame
3	Capture_lenght	Length of Capture
4	Frame_IS_marked	Frame IS Marked
5	Coloring_rule_name	Coloring Rule name
6	Ethernet_type	Type of Ethernet Protocol
7	Ver_IP	IP Version
8	Header_lenght_IP	IP Header length
9	Differentiated_S	Differentiated Service
10	IP_Total_Lenght	IP total length
11	Identification_IP	Identification IP
12	MF_Flag_IP	More Fragment flag
13	DF_Flag_IP	Don't Fragment flag
14	Fragmentation_offset_IP	Fragmentation offset IP
15	Time_to_live_IP	Time to live IP
16	Protocol_no	Protocol number
17	Src_port	Source Port
18	Dst_port	Destination port
19	Stream_index	Stream Index number
20	Sequence_number	Sequence number
21	Ack_number	Acknowledgment number
22	Cwr_flag	Cwr Flag (status flag of the connection)
23	Ecn_echo_flag	Ecn Echo flag (status flag of the connection)
24	Urgent_flag	Urgent flag (status flag of the connection)
25	Ack_flag	Acknowledgment flag(status flag of the connection)
26	Psh_flag	push flag (status flag of the connection)
27	Rst_flag	Reset flag (status flag of the connection)
28	Syn_flag	Syn flag (status flag of the connection)
29	Fin_flag	Finish flag (status flag of the connection)
30	ICMP_Type	specifies the format of the ICMP message such as: (8=echo request and 0=echo reply)
31	ICMP_code	Further qualifies the ICMP message
32	ICMP_data	ICMP data

APPENDIX 2 ATTACKS IN LINCOLN DATA 1999

Category of attacks	**Types of attacks** [32]
Probe	ipsweep, nmap, portsweep, satan
Denial of Service (DoS)	back, land, Neptune, pod, smurf, teardrop
User to root (U2R)	buffer_overflow, loadmodule, perl, rootkit
Remote to Local (R2L)	ftp_write, guess_passwd, impat, multihop, phf, spy, warezclient, warezmaster

Type of attack	**Description** [32]
back	Denial of service attack against apache webserver where a client requests a URL containing many backslashes.
dict	Guess passwords for a valid user using simple variants of the account name over a telnet connection.
eject	Buffer overflow using eject program on Solaris. Leads to a user->root transition if successful.
ffb	Buffer overflow using the ffbconfig UNIX system command leads to root shell
format	Buffer overflow using the fdformat UNIX system command leads to root shell
ftp-write	Remote FTP user creates .rhost file in world writable anonymous FTP directory and obtains local login.
guest	Try to guess password via telnet for guest account.
imap	Remote buffer overflow using imap port leads to root shell
ipsweep	Surveillance sweep performing either a port sweep or ping on multiple host addresses.
land	Denial of service where a remote host is sent a UDP packet with the same source and destination
loadmodule	Non-stealthy loadmodule attack which resets IFS for a normal user and creates a root shell
multihop	Multi-day scenario in which a user first breaks into one machine
neptune	Syn flood denial of service on one or more ports.
nmap	Network mapping using the nmap tool. Mode of exploring network will vary—options include SYN
perlmagic	Perl attack which sets the user id to root in a perl script and creates a root shell
phf	Exploitable CGI script which allows a client to execute arbitrary commands on a machine with a misconfigured web server.
pod	Denial of service ping of death
portsweep	Surveillance sweep through many ports to determine which services are supported on a single host.
rootkit	Multi-day scenario where a user installs one or more components of a rootkit
satan	Network probing tool which looks for well-known weaknesses. Operates at three different levels. Level 0 is light
smurf	Denial of service icmp echo reply flood.
spy	Multi-day scenario in which a user breaks into a machine with the purpose of finding important information where the user tries to avoid detection. Uses several different exploit methods to gain access.
syslog	Denial of service for the syslog service connects to port 514 with unresolvable source ip.
teardrop	Denial of service where mis-fragmented UDP packets cause some systems to reboot.
warez	User logs into anonymous FTP site and creates a hidden directory.
warezclient	Users downloading illegal software which was previously posted via anonymous FTP by the warezmaster.
warezmaster	Anonymous FTP upload of Warez (usually illegal copies of copywrited software) onto FTP server

APPENDIX 3 COMPARISON OF KDD CUP 99 STUDIES

Method	No. features	Normal	Probe	DoS	U2R	R2L	DR
SVDF	6	-	-	-	-	-	88,72
MARS	6	-	-	-	-	-	92,80
LGP	6	-	-	-	-	-	87,71
Rough set	6	-	-	-	-	-	89,25
Rough-PSO	6	-	-	-	-	-	93,41
SVM	41	99,55	99,70	99,25	99,87	99,78	99,63
SVM (PBMR)	31	99,51	99,67	99,22	99,87	99,78	99,61
SVM (SVDFMR)	23	99,55	99,71	99,20	99,87	99,78	99,62
BN	41	99,57	99,43	99,69	64,00	99,11	92,36
BN	19	99,57	96,71	99,02	56,00	97,87	89,83
BN	17	99,64	98,57	98,16	60,00	98,93	91,06
BN	12	98,78	99,57	98,95	48,00	98,93	88,85
CART	41	99,64	97,85	99,47	48,00	90,58	87,11
CART	19	95,50	96,85	94,31	84,00	97,69	93,67
CART	17	99,64	100,00	99,97	72,00	96,62	93,65
CART	12	100,00	97,71	85,34	64,00	95,56	88,52
BN+CART	41	99,71	99,85	99,93	72,00	99,47	94,19
BN+CART	17	99,64	100,00	100,00	72,00	99,29	94,19
BN+CART	12	100,00	99,86	99,98	80,00	99,47	95,86

APPENDIX 4 TCPDUMP2SOM.SH

```
#!/bin/bash

echo "------------------------------------------------------"
echo "Filtering tcpdump files from all non-IP-based traffic"
echo "------------------------------------------------------"

if [ -a ip_*.tcpdump ]; then
        echo "------------------------------------------------------"
        echo "Tcpdump files already filtered"
        echo "------------------------------------------------------"
else
        for i in *.tcpdump; do
        tcpdump -r $i ip -w ip_$i;
        done
fi

echo "------------------------------------------------------"
echo "Converting tcpdump files: tcpdump > Argus data > csv"
echo "------------------------------------------------------"

if [ -a *.csv ]; then
        echo "------------------------------------------------------"
        echo "Files already converted"
        echo "------------------------------------------------------"
else
        COUNT=1;
        for i in ip*.tcpdump; do
        echo "Converting file $i"
        argus -w - -r $i | ra -u -nr - -c ";" -s stime proto saddr sport spkts sbytes daddr dport dpkts dbytes >
./$COUNT.csv;
        echo "File ./$COUNT.csv created"
        let COUNT=COUNT+1;
        done
fi

echo "------------------------------------------------------"
echo "Creating timeseries"
echo "------------------------------------------------------"

if [ -a week.csv ]; then
        echo "------------------------------------------------------"
        echo "Timeseries already created"
        echo "------------------------------------------------------"
else
        cat 1.csv 2.csv 3.csv 4.csv 5.csv > week.csv
        echo `./parser.py > ./SOM.csv`
fi
echo "DONE"
```

APPENDIX 5 PARSER.PY

```
#!/usr/bin/env python
# Parses argus data for SOM
# Original parser (c)Olli Knuuti & Mika Tuomi, v. 5.3.2009
# Modified by Antti Niemela, 2011 Nokia Siemens Networks
# Modified parser (c)Nokia Siemens Networks, 2011
# Input format CSV-file:
# time;protocol;source-ip;source-port;sent-packets;sent-bytes;dest-ip;dest-port;received-packets;received-bytes
# 920898003.071811;udp;192.168.1.1;520;1;66;224.0.0.9;520;0;0
# ra function:
# ra -u -nr - -c ";" -s stime proto saddr sport spkts sbytes daddr dport dpkts dbytes
from pprint import pprint
import operator
import gzip
import time
from glob import glob
def read_ra_sorted( filename ) :
    lines = open( filename ).readlines()
    lines.sort()
    for line in lines :
        line = line.strip().split(';')
                # Checking if the flow data contains correct number of features in each line
                # By default the number of features is 10
        if len( line ) ==  10 :
            #print line # For debuggin, will print all the lines used in the timeseries
            yield line

def dump_ip_list( ip_list, starttime ) :
     format =
     '%(ip)s;%(src_sessions)i;%(unique_src_ip_count)i;%(dst_sessions)i;%(unique_dst_ip_count)i;%(port_below_1k)i;%(
     unique_port_below_1k_count)i;%(port_above_1k)i;%(unique_port_above_1k_count)i;%(sent_packets)i;%(received_
     packets)i;%(sent_bytes)i;%(received_bytes)i;%(tcp)i;%(udp)i;%(icmp)i;%(smtp)i;%(ftp)i;%(http)i;%(dns)i;%(telnet)i;%
     (ssh)i;%(time)s;'
    strtime = time.strftime('%Y%m%d;%H:%M:%S', time.gmtime(starttime))
    for ip in ip_list :
        ip_list[ip]['ip'] = ip
        ip_list[ip]['time'] = strtime
        ip_list[ip]['unique_dst_ip_count'] = len(ip_list[ip]['unique_dst_ip'])
        ip_list[ip]['unique_src_ip_count'] = len(ip_list[ip]['unique_src_ip'])
        ip_list[ip]['unique_port_below_1k_count'] = len(ip_list[ip]['unique_port_below_1k'])
        ip_list[ip]['unique_port_above_1k_count'] = len(ip_list[ip]['unique_port_above_1k'])
        print format % ip_list[ip]
        #pprint(ip_list[ip])

def main() :
    # Define the timewindow for the timeseries, default value is 5 seconds.
    timewindow = 5
    # An Ip-filter can be set here. By default all IPs are analysed.
    ipfilter = ''
    # Open flow data files that are ending with .csv in the specified folder.
    files = glob('./*.csv')
    #print files # Debugging, to check which files are used in timeseries creation
    if len(files) == 0 :
        return

    ip_list = {}
    starttime = None
```

```
for file in files :
    for event in read_ra_sorted( file ) :
            stime,protocol,src_ip,src_port,src_packets,src_bytes,dst_ip,dst_port,dst_packets,dst_bytes = event
            ftime = int(float(stime) / timewindow) * timewindow
        # Creating the "service" variable for the counters
        service = ''
        # Checking wether the used service protocol is (SMTP, FTP, HTTP..)
        # Check if the service protocol is FTP = port:21
        if int(src_port) == 21 or int(dst_port) == 21 :
            service = 'ftp'
        # Check if the service protocol is  SSH = port:22
        if int(src_port) == 22 or int(dst_port) == 22 :
            service = 'ssh'
        # Check if the service protocol is  Telnet = port:23
        if int(src_port) == 23 or int(dst_port) == 23 :
            service = 'telnet'
        # Check if the service protocol is  SMTP = port:25
        if int(src_port) == 25 or int(dst_port) == 25 :
            service = 'smtp'
        # Check if the service protocol is  DNS = port:53
        if int(src_port) == 53 or int(dst_port) == 53 :
            service = 'dns'
        # Check if the service protocol is  HTTP = port:80
        if int(src_port) == 80 or int(dst_port) == 80 :
            service = 'http'

        if ftime != starttime :
            if (starttime != None) :
                        #print '-' * 50        # debug print between time windows
                        dump_ip_list( ip_list, starttime )
                        ip_list = {}
            starttime = ftime

        for ip,port,packets,bytes in ((src_ip,src_port,src_packets,src_bytes), (dst_ip,dst_port,dst_packets,dst_bytes)) :
            # Checking if IP-filter is used. All the IPs are analysed if the default value of ipfilter is used.
        if ip.startswith(ipfilter) :
        # If IP not examined before within the timeframe, create DB for it.
              if ip not in ip_list :
                 ip_list[ip] = {
                                    'unique_dst_ip' : {},
                                     'unique_src_ip' : {},
                                     'unique_port_below_1k' : {},
                                     'unique_port_above_1k' : {},
                                     'port_below_1k' : 0,
                                     'port_above_1k' : 0,
                                     'sent_packets' : 0,
                                     'sent_bytes' : 0,
                                     'received_packets' : 0,
                                     'received_bytes' : 0,
                                     'src_sessions' : 0,
                                     'dst_sessions' : 0,
                                     'tcp' : 0,
                                     'udp' : 0,
                                     'icmp' : 0,
                                     'smtp' : 0,
                                     'ftp' : 0,
                                     'http' : 0,
                                     'dns' : 0,
                                     'telnet' : 0,
                                     'ssh' : 0
                             }
```

```
                # Increase used transportation protocol counter by 1
                if protocol in ip_list[ip] :
                    ip_list[ip][protocol]   += 1

                            # Increase used service counter by 1
                            if service in ip_list[ip] :
                                    ip_list[ip][service] +=1

                # Just to convert string ports to zero
                try :
                    port = int(port)
                except :
                    port = 0

                            # Check if the used port is equal or below 1024
                if int(port) <= 1024 :
                    ip_list[ip]['port_below_1k']    += 1
                    ip_list[ip]['unique_port_below_1k'][port] = ip_list[ip]['unique_port_below_1k'].get(port, 0) + 1

                # Check if the used port is above 1024
                if int(port) > 1024 :
                    ip_list[ip]['port_above_1k']    += 1
                    ip_list[ip]['unique_port_above_1k'][port] = ip_list[ip]['unique_port_above_1k'].get(port, 0) + 1

                            # If the IP under examination is the source address in the flow:
                            # Increase the counters
                if ip is src_ip :
                    ip_list[ip]['src_sessions']     += 1
                    ip_list[ip]['sent_packets']     += int(src_packets)
                    ip_list[ip]['sent_bytes']       += int(src_bytes)
                    ip_list[ip]['received_packets'] += int(dst_packets)
                    ip_list[ip]['received_bytes']   += int(dst_bytes)
                    ip_list[ip]['unique_dst_ip'][dst_ip] = ip_list[ip]['unique_dst_ip'].get(dst_ip, 0) + 1

                            # If the IP under examination is the destination address in the flow:
                            # Increase the counters
                if ip is dst_ip :
                    ip_list[ip]['dst_sessions']     += 1
                    ip_list[ip]['received_packets'] += int(src_packets)
                    ip_list[ip]['received_bytes']   += int(src_bytes)
                    ip_list[ip]['sent_packets']     += int(dst_packets)
                    ip_list[ip]['sent_bytes']       += int(dst_bytes)
                    ip_list[ip]['unique_src_ip'][src_ip] = ip_list[ip]['unique_src_ip'].get(src_ip, 0) + 1

    dump_ip_list( ip_list, starttime )

if __name__ == '__main__': main()
```

APPENDIX 6 FEATURE SUBSET TABLES

Detection rate of all attacks

Solaris, 172.16.112.50					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	8	8	39	10	22
DR Attacks >60s	7	7	67	14	43
DR DoS All	12	12	18	18	18
DR DoS >60s	0	0	100	100	100
DR Probe All	0	0	100	100	100
DR Probe >60s	0	0	0	0	0
DR Mail server All	0	0	100	100	100
DR Mail server >60s	0	0	100	100	100

NT, 172.16.112.100					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	21	26	3	21	26
DR Attacks >60s	53	53	7	33	47
DR DoS All	25	25	0	25	25
DR DoS >60s	33	33	0	33	33
DR Probe All	11	0	0	11	11
DR Probe >60s	100	0	0	100	100
DR Mail server All	0	0	0	0	0
DR Mail server >60s	0	0	0	0	0

Linux, 172.16.114.50					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	30	43	17	13	26
DR Attacks >60s	42	42	17	8	25
DR DoS All	50	50	33	17	50
DR DoS >60s	60	60	40	25	60
DR Probe All	33	100	67	33	67
DR Probe >60s	0	0	0	0	0
DR Mail server All	50	50	50	0	50
DR Mail server >60s	100	100	100	0	100

Detection rate of selected attacks

Solaris, 172.16.112.50					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	11	11	22	22	28
DR Attacks >60s	0	0	100	100	100

NT, 172.16.112.100					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	15	15	0	15	15
DR Attacks >60s	50	50	0	50	50

Linux, 172.16.114.50					
Metric	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
DR Attacks All	44	67	44	22	56
DR Attacks >60s	60	60	40	20	60

Number of false and true positives

Solaris	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
False Positives	14	15	47	39	55
True Positives	4	4	17	8	16

NT	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
False Positives	42	69	12	49	52
True Positives	23	26	1	23	22

Linux	**All**	**Knuuti**	**Probe**	**DoS**	**Mail server**
False Positives	51	51	19	42	22
True Positives	24	24	17	3	21