WAVELET BASED DETECTION OF NETWORK TRAFFIC ANOMALIES

by

Dağhan Hasan

B.S., Electronics and Telecommunication Engineering,

İstanbul Technical University, 2004

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering

Boğaziçi University

2007

ACKNOWLEDGEMENTS

I would like to thank to my thesis supervisor Prof. Dr. Emin Anarım for his guidance in this work, especially his technical advices and his patience. I'm indebted to him for his interest, guidance and tolerance.

Additionally, I would like to thank to Assist. Prof. Dr. Frédéric Kerem Harmancı for his kind interest, technical advices and his patience.

Additionally, I would like to thank to Assist. Prof. Dr. Fatih Alagöz for his kind interest, technical advices and support in this thesis.

This work is supported by the State Planning Organization under "The Next Generation Satellite Network and Applications" Project, No: DPT 2003-K120250.

ABSTRACT

WAVELET BASED DETECTION OF NETWORK TRAFFIC ANOMALIES

Computer Networks can be considered as an important component of today's human life. Since data and information of various organizations and companies are transferred through private and public networks such as global internet network, thus special attention to the security parameters of these networks has emerged and is even increasing progressively. In order to increase the security of these networks, tools such as firewalls and intrusion detection systems (IDS) are used. [1]

An intrusion detection system (IDS) generally detects unwanted manipulations to computer systems, mainly through Internet. These manipulations may take the form of attacks by hackers. [5]

In this thesis, signal processing techniques are applied to intrusion detection systems, and a framework for real time wavelet-based analysis of network traffic anomalies is developed and implemented. A metric, namely percentage deviation to evaluate the detection parameters is used. Using these detection parameters, which are the processed wavelet coefficients, a decision for an instance is made and therefore an alert is generated if there is some anomalous state.

The motivation for this work is to justify the assumptions that wavelets can be used to develop a real time network intrusion detection system.

Using the KDD Data Set anomalies are aimed to be detected in short time periods. We believe that this knowledge could indeed be useful in developing such an intrusion detection system, as the achieved anomaly detection ratio in this thesis work is very satisfactory.

ÖZET

DALGACIK TABANLI AĞ ANORMALLİKLERİNİN TESPİT EDİLMESİ

Bilgisayar ağları günümüzde, insan hayatının çok önemli bir parçasını oluşturmaktadır. Çeşitli organizasyon ve firmalara ait çeşitli veriler ve bilgiler, özel ağlar ve Internet gibi herkese açık ağlar üzerinden aktarılmaktadır. Bu sebepten ötürü bu ağların güvenlik parametrelerini belirlemeye ve sağlamaya özen gösterilmektedir ve gösterilen bu özen ve ilgi gitgide artmaktadır. Bu ağ sistemlerinin güvenliklerini sağlamak için Firewall ve IDS (Saldırı Tespit Sistemi) gibi ağ cihazları kullanılmaktadır.

Bir saldırı tespit sistemi genel olarak, bilgisayar sistemlerine çoğunlukla Internet üzerinden gelen istenmeyen işlemleri ve değişiklikleri tespit eder. Bu işlemler ve değişiklikler, kırıcıların yaptıkları saldırılar olabilir.

Bu tez çalışmasında, işaret işleme yöntemleri kullanılarak bir saldırı tespit sistemi geliştirilmiştir. Oluşturulan yapı aracılığıyla gerçek zamanlı olarak dalgacık tabanlı bir ağ trafiği anormalliklerini analiz sistemi geliştirilmiştir. Bu sistemde, anormallik tespit parametrelerini oluşturmak üzere "Percentage Deviation" isimli bir metrik kullanılmıştır. İşlenmiş dalgacık katsayıları kullanılarak, belli bir anda tespit edilen anormal durum karşısında bir alarm üretilmesi sağlanmıştır.

Bu çalışmanın yapılmasındaki en büyük motivasyon, dalgacıkların gerçek zamanlı bir ağ saldırı tespit sistemi tasarlanmasında kullanılabileceğini kanıtlamaktır.

KDD veri seti kullanılarak, kısa zaman aralıklarında bu set içinde yer alan ağ anormalliklerinin tespiti hedeflenmiştir. Bu tez çalışmasında elde edilen sonuçlar doğrultusunda, bu bilgilerin kullanılması ile geliştirilecek bir saldırı tespit sisteminin başarılı sonuçlar elde edeceği görüşündeyiz.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS……. iii
ABSTRACT……..…… iv
ÖZET……. v
LIST OF FIGURES……..... viii
LIST OF TABLES…….. x
LIST OF SYMBOLS / ABBREVIATIONS…….…….. xii
1. INTRODUCTION……...…… 1
1.1. Network Security…….. 1
1.1.1. Intrusion Detection System……..…….…….... 3
1.1.1.1. Types of Intrusion Detection Systems…..……..…….….. 3
1.2. Attack Definitions and Types……... 5
1.3. Choice of Variable…….…….. 7
1.4. Motivation...……... 12
1.5. Thesis Outline……. 13
2. ANOMALY DETECTION METHODS……...……. 14
2.1. Rule Based Models…… 14
2.2. Pattern Matching Models……..……. 14
2.3. Multiscale and Multidimensional Analysis……... 15
2.4. TrafficMatrices Models…… 17
2.5. Image Based Anomaly Detection……... 17
2.6. Change Point Detection…….. 18
2.7. PCA…… 19
2.8. PAYL Model…….. 19
2.9. Statistical Analysis Models…… 20
2.10. Comments on Related Works……... 20
3. METHODOLOGY……..……... 21
3.1. Auto Regressive (AR)……...…….……... 21
3.1.1. SNMP Protocol Overview…..…….....…….. 21
3.1.2. MIB…….……...…..……..…… 23
3.1.3 Least Squares Estimates……..……..…….…….. 25

3.1.4. Yule-Walker Method……...……..……... 31
3.1.5. SVD to Define A Operator Matrix…….…….……..……..... 32
3.1.6 Results……..…….…… 34
3.2. Wavelet…..……...…….. 36
3.2.1. Wavelet-Modulus Maxima Model……..…….. 36
3.2.2. DWT……..…… 40
3.2.3. Percentage Deviation…..……. 43
4. WAVELET MODEL……...…….. 44
4.1. Model Architecture…….…… 44
4.2. Performance Measurements……...……..…....…….…… 49
4.2.1. Data Set and Attack Instances……..……. 49
4.2.2. Intrusion Detection Capability……..……..… 50
4.3. Results…..…….……..…… 52
5. CONCLUSION…….. 75
APPENDIX A: PERFORMANCE TABLES…….…….... 79
APPENDIX B: MATLAB CODES…….…….……... 81
REFERENCES……..... 83

LIST OF FIGURES

Figure 3.1. Network for SNMP Data Set..……. 23

Figure 3.2. MIB Variables for SNMP Data Set…….. 24

Figure 3.3. Test and Learning Windows…….……..…. 25

Figure 3.4. Results for Yule-Walker Method with Operator Matrix A…….. 34

Figure 3.5. Results for Yule-Walker Method with SVD…… 35

Figure 3.6. Wavelet Model…….…….……... 36

Figure 3.7. Haar and Mexican Hat Wavelets...…….. 38

Figure 3.8. DTW Decomposition……..…….……. 40

Figure 4.1. Wavelet Model…….…….……..... 44

Figure 4.2. Window Shifts…...…….. 46

Figure 4.3. Features with huge peaks...……...…….……... 48

Figure 4.4. Alerts for Daubechies 6 Wavelets with a Threshold Value 1……... 53

Figure 4.5. Alerts for Daubechies 6 Wavelets with a Threshold Value 2……... 54

Figure 4.6. Alerts for Daubechies 6 Wavelets with a Threshold Value 3……... 55

Figure 4.7. Alerts for Daubechies 6 Wavelets with a Threshold Value 4……... 57

Figure 4.8. Alerts for Daubechies 6 Wavelets with a Threshold Value 10……..... 58

Figure 4.9. Alerts for Daubechies 2 Wavelets with a Threshold Value 1..……..…..... 59

Figure 4.10 Alerts for Daubechies 2 Wavelets with a Threshold Value 2..…….…..... 60

Figure 4.11. Alerts for Daubechies 2 Wavelets with a Threshold Value 3..…..….…..... 62

Figure 4.12. Alerts for Coiflet 2 Wavelets with a Threshold Value 2……...…….…..... 63

Figure 4.13. Alerts for Coiflet 2 Wavelets with a Threshold Value 3…....…….…..... 64

Figure 4.14. Alerts for Haar Wavelets with a Threshold Value 1……....…….…..... 66

Figure 4.15. Alerts for Haar Wavelets with a Threshold Value 2…..….....…….…..... 67

Figure 4.16. Alerts for Window Shifts of 2500 Connections with a Threshold Value 2…….....…….…..... 68

Figure 4.17. Alerts for Window Shifts of 2500 Connections with a Threshold Value 3…….....…….…..... 69

Figure 4.18. Alerts for Window Shifts of 3000 Connections with a Threshold Value 2…….....…….…..... 70

Figure 4.19. Alerts for Window Shifts of 3000 Connections with a Threshold Value 3…….....…….…..... 71

Figure 4.20. Alerts for Window Shifts of 500 Connections with a Threshold Value 2…….....…….…..... 72

Figure 4.21. Alerts for Window Shifts of 500 Connections with a Threshold Value 3…….....…….…..... 74

LIST OF TABLES

Table 1.1. KDD Features……... 10

Table 1.2. KDD Features Continued...……...……... 11

Table 4.1. Attack Instances……... 49

Table 4.2. Attack Instances Continued..……... 50

Table 4.3. Number of attacks instances, false positives and false negatives for a threshold of 1 with Daubechies 6.……...……... 54

Table 4.4. Number of attacks instances, false positives and false negatives for a threshold of 2 with Daubechies 6.……...…... 55

Table 4.5. Number of attacks instances, false positives and false negatives for a threshold of 3 with Daubechies 6.……...…... 56

Table 4.6. Number of attacks instances, false positives and false negatives for a threshold of 4 with Daubechies 6.……...……... 58

Table 4.7. Number of attacks instances, false positives and false negatives for a threshold of 10 with Daubechies 6...……...…... 59

Table 4.8. Number of attacks instances, false positives and false negatives for a threshold of 1 with Daubechies 2.……...……... 60

Table 4.9. Number of attacks instances, false positives and false negatives for a threshold of 2 with Daubechies 2..……...……... 61

Table 4.10. Number of attacks instances, false positives and false negatives for a threshold of 3 with Daubechies 2..……..…….... 63

Table 4.11. Number of attacks instances, false positives and false negatives for a threshold of 2 with Coiflet 2.……...…….... 64

Table 4.12. Number of attacks instances, false positives and false negatives for a threshold of 3 with Coiflet 2.……..…….... 65

Table 4.13. Number of attacks instances, false positives and false negatives for a threshold of 1 with Haar...……...…….... 66

Table 4.14. Number of attacks instances, false positives and false negatives for a threshold of 2 with Haar...……...…... 67

Table 4.15. Number of attacks instances, false positives and false negatives for a threshold of 2 with window shifts of 2500 connections...……...…... 68

Table 4.16. Number of attacks instances, false positives and false negatives for a threshold of 3 with window shifts of 2500 connections.…….……... 69

Table 4.17. Number of attacks instances, false positives and false negatives for a threshold of 2 with window shifts of 3000 connections...……...…….... 71

Table 4.18. Number of attacks instances, false positives and false negatives for a threshold of 3 with window shifts of 3000 connections…...……... 72

Table 4.19. Number of attacks instances, false positives and false negatives for a threshold of 2 with window shifts of 500 connections……...…….... 73

Table 4.20. Number of attacks instances, false positives and false negatives for a threshold of 3 with window shifts of 2500 connections. …...…….... 74

Table 5.1. Cid Values for different Algorithms……...……..... 77

LIST OF SYMBOLS / ABBREVIATIONS

A	Operator matrix
A_{ip}	IP variables operator matrix
$A(\lambda,t)$	Average value of the signal $x(t)$
a	Vector of AR coefficients
B	Base rate
C	Covariance matrix
C_{ID}	Intrusion detection capability
$[c_{ij}]$	Covariance matrix
$c_{j,k}$	Approximated coefficients
d	Size of unitary matrix in SVD
$d_{j,k}$	Detailed coefficients
$D(\lambda,t)$	Difference between average values
ε_i	Residual error
e	Size of unitary matrix in SVD
$e_j(t)$	Residual of DWT
$E\{x\}$	Expected value of x
$f(\varepsilon_1,.....\varepsilon_i)$	Joint probability density function
g_n	High pass filter in DWT
h	AR decision threshold
h_n	Low pass filter in DWT
H_0	Hypothesis implying a no-change
H_1	Hypothesis implying a change
I	Identity matrix
ℓ	Likelihood ratio
L	Likelihood function
$L^2(R)$	Weighted spaces
μ	Mean of the segment

n	Abnormality vector length
N_R	Learning window length
N_S	Test window length
N'_S	$N_S - p$, test window length minus AR order
m	MA model order
M	Constant, determine the amount of overdetermination of ARMA
p	AR order
$p(\varepsilon_{N_R} / \alpha_p)$	Joint likelihood of the residual time series
PD_{avg}	Average of percentage deviation
PD_x	Percentage deviation of elements
$P(A)$	Probability of alarm
$P(I)$	Probability of intrusion
$R(t)$	Learning window
$r_i(t)$	Elements of learning window
$\tilde{r}_i(t)$	Estimation of $r_i(t)$
$r(i)$	Elements of covariance matrix of Yule Walker
R_n	Covariance matrix of Yule Walker
R_φ	Covariance matrix in SVD
$S(t)$	Test window
S	*d*-by-*e* matrix in SVD
t	Time
th	Threshold of decision
U	*d*-by-*d* unitary matrix
V	*e*-by-*e* unitary matrix
$\underline{v}$	Decorrelated abnormality vector
V_m	Subspaces
v	Elements of V
Y	Time series vector
W	MxM positive definite weighting matrix
W_m	Orthogonal complement of spaces

$W(\lambda,t)$	Wavelet transform, *t*: time, λ : scale
Σ	*d*-by-*e* with nonnegative numbers
x	General variable
$x(t)$	Real valued function
X	Input vector of "corrcoef" matlab function
2^j	Scale number of wavelet transform
α	AR parameters or AR coefficients
α_R	AR coefficients learning window segment
α_S	AR coefficients test window segment
σ^2	Variance of segment
σ_R^2	Variance of learning window segment
σ_S^2	Variance of test window segment
σ_p^2	Variance of pooled window segment
η	Likelihood ratio
η_L	Log likelihood ratio
η_{IR}	Likelihood ratio of IPinreceives
η_{IDe}	Likelihood ratio of IPindelivers
$\eta_{OR.}$	Likelihood ratio of IPoutrequest
λ	Scale of wavelet transform
λ_d	Decision Variable of AR model
θ	Vector of AR coefficients in Yule-Walker
$\hat{\gamma}_k$	Variance estimate of the Modified Yule-Walker method
$\psi(t)$	Mother wavelet function or basis function
$\varphi(t)$	Abnormality vector
$\phi(x)$	Scaling filter
AC	Autocorrelation
AR	Autoregressive
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
ASN	Abstract Syntax Notation

CD	Compact Disc
CPM	Change-Point Monitoring
CPU	Central Processing Unit
CUSUM	Cumulative Sum
CWT	Continuous Wavelet Transform
DARPA	Defense Advanced Research Projects Agency
DFT	Discrete Fourier Transform
DNS	Domain Name Service
DDoS	Distributed Denial of Service
DoS	Denial of Service
DWT	Discrete Wavelet Transform
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
GLR	Generalized Likelihood Ratio
HIDS	Host based Intrusion Detection System
IDS	Intrusion Detection System
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
IRC	Internet Relay Chat
IT	Information Technology
KB	Kilobyte
KDD	Knowledge-Discovery in Databases
LLR	Log likelihood Ratio
LS	Least Squares
MA	Moving Average
MFLOP	Millions of Floating Operations per Second
MIB	Management Information Base
MODWT	Maximal Overlap Discrete Wavelet Transform
NIDS	Network based Intrusion Detection System
NPV	Negative Predictive Value
NT	New Technology

OD	Origin Destination
OS	Operating System
PAYL	Payload Based Anomaly Detector
PC	Personal Computer
PCA	Principal Components Analysis
PPV	Positive Predictive Value or Bayesian Detection Rate
SNMP	Simple Network Management Protocol
SVD	Singular Value Decomposition
SYN	Synchronize Packet
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
UDP	User Datagram Protocol

1. INTRODUCTION

Computer Networks can be considered as an important component of today's human life. Since data and information of various organizations and companies are transferred through private and public networks such as global internet network, thus special attention to the security parameters of these networks has emerged and is even increasing progressively. In order to increase the security of these networks tools such as firewalls and intrusion detection systems (IDS) are used. [1]

Intrusion detection systems have been an active area of research for quite some time, and a very well developed commercial technology. There are many IDS systems available that are primarily signature-based detectors. Although these are effective at detecting known intrusion attempts and exploits, they fail to recognize new attacks and carefully crafted variants of old exploits. A generation of systems has appeared which are based upon anomaly detection. Anomaly Detection systems model normal or expected behavior in a system, and detect deviations of interest that may indicate a security breach or an attempted attack. [2]

In this Introduction, there will be a detailed description of Network Security, the function of IDS in Network Security, how IDS actually work and what type of IDS are available.

1.1. Network Security

In order to have an understanding of the performance and behavior of complex networks, huge amounts of information need to be collected and processed. Using some basic knowledge of the network layout as well as the traffic characteristics in the network it is possible to detect network anomalies and performance bottlenecks. The detection of these events can then be used to trigger alarms to a network management system [3].

The anomalies and performance problems that can be experienced in computer networks can be caused by several reasons. As some of these reasons may be due hardware problems in network devices, it can also be due to intentional or unintentional network attacks through internet or sometimes even from the internal network. In this paper, mainly network attacks will be considered. Some of the attack types that a computer network may suffer can be listed as below: [3, 4]

- Password cracking and access violation.
- Viruses, Worms and Trojan horses.
- Spoofing (deliberately misleading by impersonating or masquerading the host identity by placing forged data in the cache of the named server i.e. DNS (Domain Name Service) spoofing).
- Scanning ports and services, including ICMP (Internet Control Message Protocol) scanning (Ping), UDP, TCP Stealth Scanning TCP that takes advantage of a partial TCP connection establishment protocol.).
- Unauthorized network connections,
- Usage of IT resources for private purposes, for example to access pornography sites,
- Taking advantage of system weaknesses to gain access to resources or privileges,
- Unauthorized alteration of resources (after gaining unauthorized access):
- Falsification of identity, for example to get system administrator rights,
- Unauthorized configuration changes to systems and network services (servers),
- DoS, DDoS attacks.

In order to keep a computer network secure from attacks, security devices such as Firewalls are not enough by themselves. So some additional hardware (and also software) is needed to protect the computer network from such attacks, i.e. IDS.

1.1.1. Intrusion Detection System

An intrusion detection system (IDS) generally detects unwanted manipulations to computer systems, mainly through Internet. The manipulations may take the form of attacks by hackers. [5]

An intrusion detection system is used to detect many types of malicious network traffic and computer usage that can not be detected by conventional firewalls. This includes network attacks against vulnerable services, data driven attacks on applications, host based attacks such as privilege escalation, unauthorized logins and access to sensitive files, and malware (viruses, Trojan horses and worms). [5]

An IDS is composed of several components: Sensors which generate security events, a Console to monitor events and alerts and control the sensors, and a central Engine that records events logged by the sensors in a database and uses a system of rules to generate alerts from security events received. There are several ways to categorize an IDS depending on the type and location of the sensors and the methodology used by the engine to generate alerts. In many simple IDS implementations all three components are combined in a single device or appliance. [5]

1.1.1.1. Types of Intrusion Detections Systems In a network based intrusion detection system (NIDS), the sensors are located at choke points in the network to be monitored, often in the demilitarized zone (DMZ) or at network borders. The sensor captures all network traffic and analyzes the content of individual packets for malicious traffic. In systems, PIDS and APIDS are used to monitor the transport and protocols illegal or inappropriate traffic or constricts of language (i.e. SQL). In a host based system, the sensor usually consists of a software agent, which monitors all activity of the host on which it is installed. Hybrids of these two systems also exist. [5]

A network intrusion detection system is an independent platform which identifies intrusions by examining network traffic and monitors multiple hosts. Network Intrusion Detection Systems gain access to network traffic by connecting to a hub, network switch configured for port mirroring, or network tap. An example of a NIDS is Snort.

A protocol based intrusion detection system consists of a system or agent that would typically sit at the front end of a server, monitoring and analyzing the communication protocol between a connected device (a user/PC or system). For a web server this would typically monitor the HTTPS protocol stream and understand the HTTP protocol relative to the web server/system it is trying to protect. Where HTTPS is in use then this system would need to reside in the "shim" or interface between where HTTPS is unencrypted and immediately prior to it entering the Web presentation layer.

An application protocol based intrusion detection system consists of a system or agent that would typically sit within a group of server, monitoring and analyzing the communication on application specific protocols. For example; in a web server with database this would monitor the SQL protocol specific to the middleware/business-login as it transacts with the database.

A host-based intrusion detection system consists of an agent on a host which identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability/acl databases) and other host activities and state.

A hybrid intrusion detection system combines one or more approaches. Host agent data is combined with network information to form a comprehensive view of the network. An example of a Hybrid IDS is Prelude. [5]

The main focus of this paper is Network based IDS using Anomaly Detection method.

1.2. Attack Definitions and Types

Attacks can be divided as passive and active attacks. Passive attacks usually cause access gain to penetrate a system without compromising IT (Information Technology) resources. Active attacks result in an unauthorized state change of IT resources. In terms of the relation intruder-victim, attacks are categorized as, internal and external attacks. Internal attacks come from own enterprise's employees or their business partners or customers. External attacks come from outside of the enterprise network, frequently via the Internet. Attacks are also identified by the source category, namely those performed from internal systems (local network), the Internet or from remote dial-in sources.

Attacks related to unauthorized actions can be listed below: [4]

- Password cracking and access violation.
- Viruses, Worms and Trojan horses.
- Interceptions; most frequently associated with TCP/IP stealing and interceptions that often employ additional mechanisms to compromise operation of attacked systems (for example by flooding); man in the middle attacks).
- Spoofing (deliberately misleading by impersonating or masquerading the host identity by placing forged data in the cache of the named server i.e. DNS (Domain Name Service) spoofing).
- Scanning ports and services, including ICMP (Internet Control Message Protocol) scanning (Ping), UDP, TCP Stealth Scanning TCP that takes advantage of a partial TCP connection establishment protocol.).
- Remote OS (Operating System) Fingerprinting, for example by testing typical responses on specific packets, addresses of open ports, standard application responses (banner checks), IP stack parameters etc.,
- Network packet listening (a passive attack that is difficult to detect but sometimes possible),
- Stealing information, for example disclosure of proprietary information,

- Authority abuse; a kind of internal attack, for example, suspicious access of authorized users having odd attributes (at unexpected times, coming from unexpected addresses).
- Unauthorized network connections,
- Usage of IT resources for private purposes, for example to access pornography sites,
- Taking advantage of system weaknesses to gain access to resources or privileges,
- Unauthorized alteration of resources (after gaining unauthorized access):
- Falsification of identity, for example to get system administrator rights,
- Information altering and deletion,
- Unauthorized transmission and creation of data (sets), for example arranging a database of stolen credit card numbers on a government computer (e.g. the spectacular theft of several thousand numbers of credit cards in 1999),
- Unauthorized configuration changes to systems and network services (servers),

Denial of Service (DoS) Type Attacks:

- Flooding – A remote system is overwhelmed by a continuous flood of traffic designed to consume resources at the targeted server (CPU cycles and memory) and/or in the network (bandwidth and packet buffers). These attacks result in degraded service or a complete site shutdown.
- Ping flood (Smurf) – An attacker sends forged ICMP echo packets to broadcast addresses of vulnerable networks. All the systems on these networks reply to the victim with ICMP echo replies. This rapidly exhausts the bandwidth available to the target, effectively denying its services to legitimate users.

- SYN flood – Taking advantage of the flaw of TCP three-way handshaking behavior, an attacker makes connection requests aimed at the victim server with packets with unreachable source addresses. The server is not able to complete the connection requests and, as a result, the victim wastes all of its network resources. A relatively small flood of bogus packets will tie up memory, CPU (Central Processing Unit), and applications, resulting in shutting down a server.
- Logic or software attacks - A small number of malformed packets are designed to exploit known software bugs on the target system. These attacks are relatively easy to counter either through the installation of software patches that eliminate the vulnerabilities or by adding specialized firewall rules to filter out malformed packets before they reach the target system. These attacks can be used detected signature based detection.
- Distributed Denial of Service (DDoS); DoS attacks coming from multiple sources.
- Buffer Overflow, for example Ping of Death — sending a very large ICMP (exceeding 64 KB).
- Remote System Shutdown.
- Web Application attacks; attacks that take advantage of application bugs.

1.3. Choice of Variable

The Information Systems Technology Group (IST) of MIT Lincoln Laboratory, under Defense Advanced Research Projects Agency (DARPA ITO) and Air Force Research Laboratory (AFRL/SNHS) sponsorship, has collected and distributed the first standard corpora for evaluation of computer network intrusion detection systems. They have also coordinated, with the Air Force Research Laboratory, the first formal, repeatable, and statistically-significant evaluations of intrusion detection systems. Such evaluation efforts have been carried out in 1998 and 1999.

These evaluations measured probability of detection and probability of false-alarm for each system under test. These evaluations contributed significantly to the intrusion detection research field by providing direction for research efforts and an objective calibration of the technical state-of-the-art. They are of interest to all researchers working on the general problem of workstation and network intrusion detection. The evaluation was designed to be simple, to focus on core technology issues, and to encourage the widest possible participation by eliminating security and privacy concerns, and by providing data types that were used commonly by the majority of intrusion detection systems. [6]

The 1998 DARPA Intrusion Detection Evaluation Program was prepared and managed by MIT Lincoln Labs. The objective was to survey and evaluate research in intrusion detection. A standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment, was provided. The 1999 KDD intrusion detection contest uses a version of this dataset.

Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks. The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records.

A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes.

Attacks fall into four main categories:

- DOS: denial-of-service, e.g. syn flood;
- R2L: unauthorized access from a remote machine, e.g. guessing password;
- U2R: unauthorized access to local superuser (root) privileges, e.g., various ``buffer overflow'' attacks;
- probing: surveillance and other probing, e.g., port scanning.

It is important to note that the test data is not from the same probability distribution as the training data, and it includes specific attack types not in the training data. This makes the task more realistic. Some intrusion experts believe that most novel attacks are variants of known attacks and the "signature" of known attacks can be sufficient to catch novel variants. [7]

Connection records were also sorted by destination host, and features were constructed using a window of 100 connections to the same host instead of a time window. This yields a set of so-called host-based traffic features. A complete listing of the set of features defined for the connection records is given in the three tables below.

Table 1.1. KDD Features

feature name	*description*	*type*
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
service	network service on the destination, e.g., http, telnet, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	number of urgent packets	continuous
Basic features of individual TCP connections.		
hot	number of ``hot'' indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of ``compromised'' conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
num_root	number of ``root'' accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	1 if the login belongs to the ``hot'' list; 0 otherwise	discrete
is_guest_login	1 if the login is a ``guest''login; 0 otherwise	discrete
Content features within a connection suggested by domain knowledge.		

Table 1.2. KDD Features Continued

feature name	*description*	*type*
count	number of connections to the same host as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-host connections.	
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-service connections.	
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous

1.4. Motivation

Real time network monitoring for intrusions is offered by various host and network based intrusion detection systems. These systems largely use signature or pattern matching techniques at the core and thus are ineffective in detecting unknown anomalous activities.

In [3], Thottan suggest a method for the detection of network anomalies. These network anomalies do not consist of attacks but they are mainly network problems i.e. a router going down or a data link being congested and etc. This anomaly detection was done by applying an AR 1 model to some MIB variables that were derived from the network via SNMP. A likelihood ratio is created, using the variances of the residual of a learning window and a test window. This likelihood ratio is the basis of the anomaly detection. We claimed that applying such a signal processing method for the detection of network attacks would bring good results. But only using an AR 1 model would not be satisfactory so we have taken interest in other methods, too.

In this paper, signal processing techniques are applied to intrusion detection systems, and a framework for real time wavelet-based analysis of network traffic anomalies is developed and implemented. A metric, namely percentage deviation is used to evaluate the detection parameters. Using the processed wavelet coefficients, a decision for an instance is made, generating an alert if there is some anomalous state.

In [8], a wavelet analysis of flows exported from Cisco routers was carried out. A three month long signal containing different kinds of anomalies was analyzed. It was found that an increase in the local variance of a time-series signal that is generated from raw traffic strongly indicated an anomaly. Wavelet analysis on a raw traffic signal allows for observation on many different levels of traffic, by removing certain components of the signal at each level, and generating wavelet coefficients. This feature extraction at different levels is known as Multi-Resolution Analysis (MRA), and has gained popularity as the method of analysis for non-stationary signals.

In [9], Huang et al. proposes a wavelet based method, called WIND, to detect network failures and other problems. The energy function of a signal with exponentially

increasing arrival time of components shows that the values are roughly constant across all scales, which could refer to the white noise of a signal. However, the wavelet transform of a signal that is comprised of values taken at different sample points exhibits details hidden within the signal.

In [10], the authors use wavelets to detect congestion on shared links in networks. The technique suggests that two paths sharing a congested link have a high correlation between their one way delays. This correlation has been monitored, and wavelet denoising is used to remove noise due to queuing delay and mild congestion. Thresholding is applied to arrive at a binary decision regarding a shared link being congested. The authors have found the Daubechies 6 wavelet has the most correlation with the congestion implementation and hence it was used as the mother wavelet for wavelet denoising in this work. The superior characteristics of wavelets for denoising have been demonstrated.
The motivation for this work is to justify the assumptions that wavelets can be used to develop a real time network intrusion detection system.

Using KDD Data Set anomalies are aimed to be detected in short time periods. We believe that this knowledge could indeed be useful in developing such an intrusion detection system.

1.5. Thesis Outline

The rest of the thesis is organized in the following fashion. Section two makes introduction to Anomaly Detection Methods. In section three Wavelet Functions and Coefficients are described. In section four Wavelet-Percentage Deviation model has been proposed. In section five results of the model has been showed. Section six concludes the thesis.

2. ANOMALY DETECTION METHODS

In this section, the most commonly used network anomaly detection methods will be reviewed. The methods described are rule-based approaches, pattern matching, Wavelet Models, and statistical analysis.

2.1. Rule Based Models

Early work in the area of fault or anomaly detection was based on expert systems. In expert systems, an exhaustive database containing the rules of behavior of the faulty system is used to determine if a fault occurred. Rule-based systems are too slow for real-time applications and are dependent on prior knowledge about the fault conditions on the network [3]. Furthermore, the identification of relevant criteria for the different faults will, in turn, require a set of rules to be developed. In addition, using any functional approximation scheme, such as back propagation, causes an increase in computation time and complexity. The number of functions to be learned also increases with the number of faults studied [3].

Alhamaty et al concentrate on finding a solution to the intrusion detection main attacks of fragmentation information packets. Main idea is to check TCP packet integrity so as not to restrict the check attack special signature. This work focuses on the packet that whether packet rightly is fragmented or not, until by change attack signature [3].

2.2. Pattern Matching Models

The efficiency of this pattern matching approach depends on the accuracy of the traffic profile generated. Given a new network, it may be necessary to spend a considerable amount of time building traffic profiles. In the face of evolving network topologies and traffic conditions, this method may not scale gracefully [3]. The methods of data analysis and pattern recognition presented are the basis of a technology study for an automatic intrusion detection system that detects the attack in the reconnaissance stage [11].

2.3. Multiscale and Multidimensional Analysis

Wavelet-based scaling analysis tools are incredibly useful for describing and detecting certain kinds of properties of one-dimensional functions, measures, or random processes. They can compute summary statistics about scale-dependent properties, local scaling behavior, and even extremely localized information about the local regularity of network traffic. Because these methods can be implemented in an on-line fashion, that use them to monitor network links, either at one main link or at many access points. However, these tools do have some serious drawbacks when it comes to the next step in network measurements. A. C. Gilbert works only with information local to a network. Gilbert cannot address distributed network measurements at all [12].

An approach for real-time network monitoring in terms of numerical time-dependant functions of protocol parameters was suggested in [11]. *Gudkov et al* have applied complex systems theory for information flow analysis of networks, the information traffic is described as a trajectory in multi-dimensional parameter-time space with about 10-12 dimensions. The network traffic description is synthesized by applying methods of theoretical physics and complex systems theory, to provide a robust approach for network monitoring that detects known intrusions, and supports developing real systems for detection of unknown intrusions [11].

In [13] *Zhang et al* have showed that the ARIMA (Autoregressive Integrated Moving Average) methods, FFT (Fast Fourier Transform) and Wavelet anomography approaches have superb performance the number of false negatives is very low. This indicates that very few important traffic anomalies can pass undetected by these approaches. The PCA based approaches, however, identify about half of the anomalies [13].

In [14], *Huang et al* apply signal processing techniques in intrusion detection systems, and develop and implement a framework, called Waveman, for real time wavelet-based analysis of network traffic anomalies. Then, they use two metrics, namely percentage deviation and entropy, to evaluate the performance of various wavelet functions on detecting different types of anomalies like Denial of Service (DoS) attacks and portscans. Results show that Coiflet and Paul wavelets perform better than other wavelets in detecting most anomalies considered in this work [14].

Inspired by the methods that use the selfsimilarity property of data network traffic as normal behavior and any deviation from it as the anomalous behavior, In [15], *Rawat et al* have proposed a method for anomaly based network intrusion detection. Making use of the relations present among the wavelet coefficients of a self-similar function in a different way, method determines the possible presence of not only an anomaly, but also its location in the data. They provide the empirical results on KDD (Knowledge-Discovery in Databases) data. Hurts parameter was used to perform anomaly detection [15].

In [16], *Kim et al* suggest a technique for traffic anomaly detection based on analyzing correlation of destination IP addresses in outgoing traffic at an egress router. This address correlation data are transformed through discrete wavelet transform for effective detection of anomalies through statistical analysis. Results from trace-driven evaluation suggest that proposed approach could provide an effective means of detecting anomalies close to the network [16]. Based on statistical bounds on normal traffic patterns of the correlation signal of destination addresses, sudden changes can be used to detect anomalies in traffic behavior. A correlation calculation is using a simple data structure. These correlation data are processed through coefficient selective discrete wavelet transform for effective and high-confidence detection [17].

2.4. Traffic Matrices Models

The link loads and traffic matrices are simply related by a linear equation $b = Ax$ The vector b contains the link measurements, and A is the routing matrix. They wish to infer x, which contains the unknown traffic matrix elements written as a vector. Tomographic inference techniques seek to invert this relationship to find x. Two basic solution strategies to network anomography: (i) *early inverse*, and (ii) *late inverse*. Early inverse approaches may appear more intuitive. The early inverse approach tackles the problem in two steps. The first is the *network tomography* step, where OD (Origin Destination) flow data at each interval j are inferred from the link load measurements by solving the ill-posed linear inverse problem. Given the estimated OD flow data x_j at different time points j, in the second step, *anmaly detection* can then be applied to the x_j [13].

2.5. Image Based Anomaly Detection

In [18] *NetViewer* was introduced a network measurement approach that can simultaneously detect, identify and visualize attacks and anomalous traffic in real-time by passively monitoring packet headers. *Kim et al* propose to represent samples of network packet header data as frames or images. With such a formulation, a series of samples can be seen as a sequence of frames or video, revealing certain kinds of attacks to the human eye. This enables techniques from image processing and video compression to be applied to the packet header data to reveal interesting properties of traffic. They show that “scene change analysis” can reveal sudden changes in traffic behavior or anomalies. They also show that “motion prediction” techniques can be employed to understand the patterns of some of the attacks. They show that it may be feasible to represent multiple pieces of data as different colors of an image enabling a uniform treatment of multidimensional packet header data [18].

Their approach passively monitors packet headers of network traffic at regular intervals and analyzes the aggregate data for anomaly detection. This approach generates images of the packet header data for both visualization and for effective processing of the collected data. During network anomalies or attacks, the usage pattern of network may change and the peculiarities could become visible in the traffic images. When anomalies are detected, further analysis can characterize the anomalies by their nature into several categories and help in mitigating the attacks [18].

2.6. Change Point Detection

Wang et al present a simple mechanism, called Change-Point Monitoring (CPM), to detect denial of service (DoS) attacks. The core of CPM is based on the inherent network protocol behaviors and is an instance of the Sequential Change Point Detection. To make the detection mechanism insensitive to sites and traffic patterns, a nonparametric Cumulative Sum (CUSUM) method was applied, thus making the detection mechanism robust, more generally applicable, and its deployment much easier [19].

CPM compares the observed sequence with the profile that represents the user's normal behavior and detects any significant deviation from the normal behavior. The key difference of CPM from others is that CPM exploits the inherent network protocol behaviors, instead of traffic patterns, for detecting network anomalies. The objective of Change-Point Detection is to determine if the observed time series is statistically homogeneous and, if not, to find the point in time when the change happens [19].

2.7. PCA

Labib et al have proposed a multivariate statistical method called Principal Component Analysis is used to detect Denial-of-Service and Network Probe attacks using the 1998 DARPA (Defense Advanced Research Projects Agency) data set. Visualization of network activity and possible intrusions is achieved using Bi-plots, which are used as a graphical means for summarizing the statistics. The principal components are calculated for both attack and normal traffic, and the loading values of the various feature vector components are analyzed with respect to the principal components. The variance and standard deviation of the principal components are calculated and analyzed. A brief introduction to Principal Component Analysis and the merits of using it for detecting the selected intrusions are discussed [20].

2.8. PAYL Model

Bolzoni et al have proposed POSEIDON which is payload-based, and has a two-tier architecture: the first stage consists of a Self-Organizing Map, while the second one is a modified PAYL (Payload Based Anomaly Detector) system. Their architecture combines a SOM with a modified PAYL algorithm. POSEIDON, like most network intrusion detection systems, is packet-oriented. This architecture presents two main advantages: firstly, POSEIDON can identify and block an attack while it is taking place (intrusion prevention). Secondly, connection-based systems are computationally more expensive, in particular they require a huge amount of memory resources to keep all the segments to analyze. This makes connection-based system more suitable for off-line analysis [21].

Bolzoni et al have proposed APHRODITE which is an architecture designed to reduce false positives in network intrusion detection systems. APHRODITE works by detecting anomalies in the output traffic, and by correlating them with the alerts raised by the NIDS working on the input traffic. Benchmarks show a substantial reduction of false positives and that APHRODITE is effective also after a “quick setup”, i.e. in the realistic case in which it has not been “trained” and set up optimally. APHRODITE works as follows: when the NIDS raises an alert, the correlation engine checks whether the communication that raised this alert also causes an anomaly in the output (detected by the

OAD (Output Anomaly Detector)). If this is the case, the alert is considered a true positive and APHRODITE forwards it to the IT professionals, otherwise, it is discarded as a false positive [16]. They have tested APHRODITE together with both POSEIDON and Snort to on the traffic of weeks 4 and 5 of DARPA.

2.9. Statistical Analysis Models.

Using online learning and statistical approaches, it is possible to continuously track the behavior of the network. Statistical analysis has been used to detect both anomalies corresponding to network failures, as well as network intrusions [3].

Qingtao et al has presented a method of detecting network anomalies by analyzing the abrupt change of time series data obtained from Management Information Base (MIB) variables. The method applies the Auto-Regressive (AR) process to model the abrupt change of time series data, and performs sequential hypothesis test to detect the anomalies [23].

2.10. Comments on Related Works

Signature based models and pattern matching models can not detect newly designed attacks because of lack of information of new attacks types. They need continuous updates to catch the up to date attacks. Using case-based reasoning for describing fault scenarios also suffers from heavy dependence on past information. Furthermore, the identification of relevant criteria for the different faults will, in turn, require a set of rules to be developed. In addition, using any functional approximation scheme, such as back propagation, causes an increase in computation time and complexity. The number of functions to be learned also increases with the number of faults studied.

In this thesis, wavelet based analysis on network traffic features is performed, which are derived from the DARPA data set. Applying the percentage deviation method on the wavelet coefficients and setting a threshold to generate network anomaly alerts, network attacks are detected near real-time periods with good false positive and false negative rates.

3. METHODOLOGY

We suggest that, applying statistical analysis methods for the detection of Network anomalies, mainly Network attacks would bring good results. In [3], Thottan suggested that by using AR models to model the network traffic behavior, network anomalies such as router, switch problems or link congestions can be easily detected. In their work, by using an AR 1 model, Thottan was able to detect such anomalies.

As a starting point, we claimed that using such a model we would be able to detect, not only network problems but also network attacks. Therefore in this work an AR 3 model is applied to a network data, which includes two network attacks. The results of anomaly detection for this model are derived. As this data and only using AR 3 modeling was not very reliable, wavelet-methods are decided to be used for the anomaly detection.

There will be a small explanation of the AR model and its parameters, which are used as the starting point. And after this explanation the wavelet-method for the anomaly detection will be explained.

3.1. Auto Regressive (AR)

3.1.1. SNMP Protocol Overview

SNMP is a communication protocol that has gained widespread acceptance since 1993 as a method of managing TCP/IP networks. SNMP was developed by the IETF (Internet Engineering Task Force), and is applicable to any TCP/IP network, as well as other types of networks. The protocol has been in existence for some time, and has been written about extensively [24].

SNMP defines a client/server relationship. An SNMP agent is some software that resides on a network node and is responsible for communicating with managers regarding that node. The node is represented as a managed object having various fields or variables that are defined in the appropriate MIB. The MIB (Management Information Base) is a

method of describing managed objects by specifying the names, types, and order of the fields (or variables) that make up the object. The MIB can either be a standard one or can be what is known as an enterprise MIB.

The client (network manager) makes virtual connections to a server (SNMP agent), which executes on a remote network device, and serves information to the manager regarding the device's status. The database, controlled by the SNMP agent, is referred to as the SNMP Management Information Base (MIB), and is a standard set of statistical and control values. SNMP additionally allows the extension of these standard values with values specific to a particular agent through the use of private MIBs.

Directives, issued by the network manager client to an SNMP agent, consist of the identifiers of SNMP variables (referred to as MIB object identifiers or MIB variables) along with instructions to either get the value for the identifier, or set the identifier to a new value.

Through the use of private MIB variables, SNMP agents can be tailored for many specific devices, such as network bridges, gateways, and routers. The definitions of MIB variables supported by a particular agent are incorporated in descriptor files, written in Abstract Syntax Notation (ASN.1) format, made available to network management client programs so that they can become aware of MIB variables and their usage.

SNMP has several advantages. Its biggest strength is arguably its widespread popularity. SNMP agents are available for network devices ranging from computers, to bridges, to modems, to printers. The fact that SNMP exists with such support gives considerable credence to its reason for existence; SNMP has become interoperable.

The SNMP data, which is used to create our dataset for the AR detection, is taken from a real network, which can be shown like in the figure below. This data contains two attacks: First attack, which is directed to the target host and the second attack, which is directed to the switch.

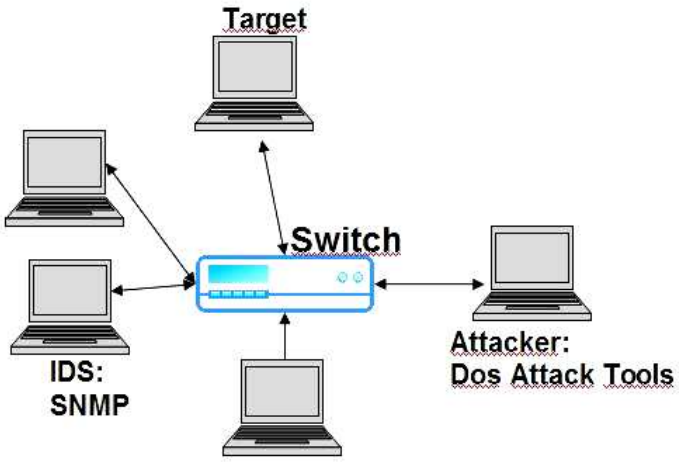

Figure 3.1. Network for the SNMP Data set

For the detection of the attacks in our dataset, three MIB variables are used. The details related to this data will be explained in the following chapter.

3.1.2. MIB

An SNMP agent uses the standard MIB variables as its input parameters.. Furthermore, MIB variables are supported by most network devices, thus making widespread application of the agent feasible [2].

The Management Information Base variables (MIB II), which are used in this work are standardized for the Simple Network Management Protocol (SNMP) version (1) and they fall into different groups. The variables *ipIR* (In Receives) represents the total number of datagram received from all interfaces of the router, *ipIDe* (In Delivers) represents the number of datagram correctly delivered to the higher layers, as this node was their final destination, and *ipOR* (Out Requests) represents the number of datagram passed on from the higher layers of the node to be forwarded by the *ip* layer. The MIB variables chosen, although non-redundant, are not strictly independent and the relationships between them have been incorporated at the combination stage described in [25].

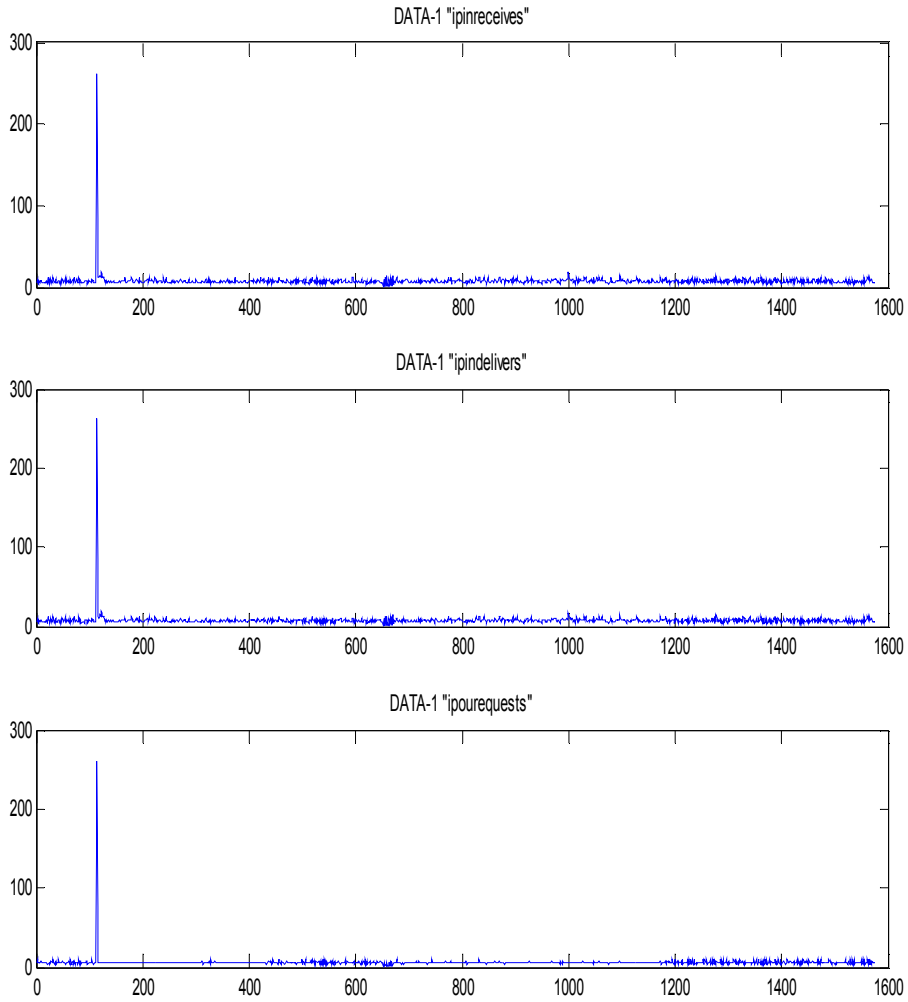

Figure 3.2. MIB Variables for SNMP Data Set

3.1.3. Least Squares AR(1) Estimate

In statistical analysis, a network anomaly is modeled as correlated abrupt changes in network data. An abrupt change is defined as any change in the parameters of a time series that occurs on the order of the sampling period of the measurement. Abrupt changes in time series data can be modeled using an autoregressive (AR) process [26]. *Totthan et al* have suggested a statistical analysis method to detect abrupt changes based on AR(1) LS. Using a change detection algorithm with appropriate set of MIB variables, alarms could be generated for various network anomaly conditions, such as router and switch failures [26]. It has been experimentally shown that changes in the statistics of traffic data can be used to detect faults [26]. The detection algorithm was implemented independently on each MIB variable.

The increments in the MIB counters were obtained every second and the data thus generated constituted a time series. This data shows a high degree of non-stationarity. Piecewise stationary Autoregressive models have been used to describe these non-stationary stochastic time series signals [26].

Thus the MIB data were divided into 10 time lags piecewise stationary windows. Within a time window of size N ($N=10$), the MIB data was linearly modeled using a first-order AR process. Piecewise stationary segments $R(t)$ and $S(t)$ are shown in Figure 3.3. $R(t)$ is the learning window and $S(t)$ is the test window. Non overlapping windows were used in order to obtain less correlated residuals. $N_R=N_S=10$.

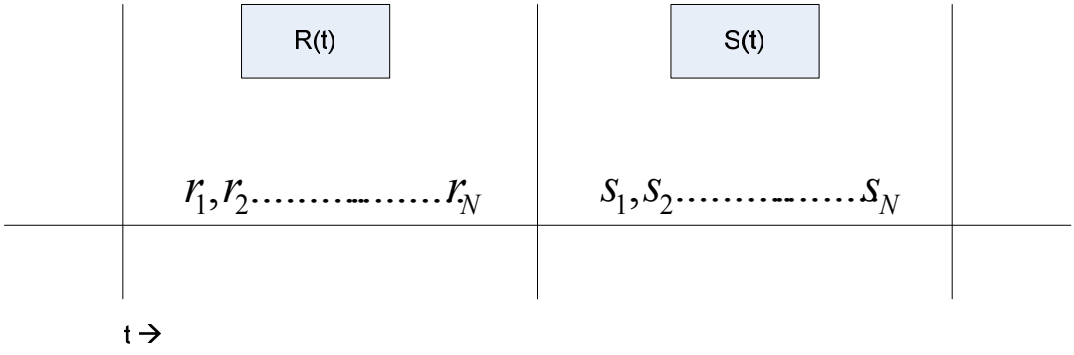

Figure 3.3. Test and Learning Windows

R(t) is defined as:

$$R(t) = \{r_1(t), r_2(t), ..., r_{N_R}(t)\} \tag{3.1}$$

Any $r_i(t)$ is stated as $\tilde{r}_i(t)$, where $\tilde{r}_i(t) = r_i(t) - \mu$ and μ is the mean of the *R(t)*. $\tilde{r}_i(t)$ can be estimated as an AR order *p* process (*p=1*) with a residual error ε_i;

$$\varepsilon_i(t) = \sum_{k=0}^{p} \alpha_k \tilde{r}(t-k) \tag{3.2}$$

Where $\alpha_R = \{\alpha_1, \alpha_2, ..., \alpha_p\}$ are the AR parameters, and $\varepsilon_i(t)$ is assumed to be the white noise. The joint probability density function of $\varepsilon_1(t), \varepsilon_2(t), ..., \varepsilon_i(t)$ are given by [27],

$$f(\varepsilon_1, ... \varepsilon_i) = (2\pi\sigma^2)^{-N/2} \exp\{-\frac{1}{2\sigma^2} \sum_{t=1}^{N} \varepsilon_t^2\} \tag{3.3}$$

Making use of the $x_{1-p}, x_{2-p}, ...x_0$ the likelihood function *L* of the $\{\varepsilon_i(t)\}$ becomes:

$$L = (2\pi\sigma^2)^{-N/2} \exp\{-\frac{1}{2\sigma^2} \sum_{t=1}^{N} (\sum_{i=0}^{p} \alpha_i \chi_{t-i})^2\} \tag{3.4}$$

Which can be rewritten as:

$$L = (2\pi\sigma^2)^{-N/2} \exp\{-\frac{1}{2\sigma^2} Na'Ca\} \tag{3.5}$$

Where *a* is the column vector given by, $a' = [1, \alpha_1, ..., \alpha_p]$ and C=[c_{ij}] is the $(p+1)x(p+1)$ matrix of covariance given by,

$$c_{ij} = \frac{1}{N} \sum_{t=1}^{N} \chi_{t-i} \chi_{t-J} \qquad i, j = 0,1,...., p \tag{3.6}$$

In order to obtain the maximum-likelihood estimates of σ^2 and α, L must be maximized with respect to σ^2 and α. *Peter et al* show that this leads to the estimates σ^2 and α where $\tilde{\sigma}$ (**covariance estimate**):

$$\hat{\sigma}^2 = a'Ca \tag{3.7}$$

Joint likelihood of the residual time series is obtained as:

$$p(\varepsilon_{p+1},.....,\varepsilon_{N_R} / \alpha_1,.....,\alpha_p) = \left(\frac{1}{\sqrt{2\pi\sigma_R^2}}\right)^{N'_R} \exp\left(\frac{-N'_R\hat{\sigma}_R^2}{2\sigma_R^2}\right) \tag{3.8}$$

Where σ_R^2 is the variance of residual in segment *R(t)* and $N'_R = N_R - p$ and $\hat{\sigma}_R$ is the covariance estimate of σ_R^2 [3]. Joint likelihood *L* of the residuals $R(t)$ and $S(t)$ becomes:

$$l = \left(\frac{1}{\sqrt{2\pi\sigma_R^2}}\right)^{N'_R}\left(\frac{1}{\sqrt{2\pi\sigma_S^2}}\right)^{N'_S} \exp\left(\frac{-N'_R\hat{\sigma}_R^2}{2\sigma_R^2}\right)\exp\left(\frac{-N'_S\hat{\sigma}_S^2}{2\sigma_S^2}\right) \tag{3.9}$$

And $N'_S = N_S - p$, σ_S^2 is the variance of the residual in the segment *S(t)*. Two hypotheses are H_0 implying that no change. H_1 implying a change. Under the hypothesis H_0; $\alpha_R = \alpha_S$ and $\sigma_R^2 = \sigma_S^2 = \sigma_p^2$ where σ_P^2 is the pooled variance:

$$l_p = \left(\frac{1}{\sqrt{2\pi\sigma_p^2}}\right)^{N'_R+N'_S} \exp\left(\frac{-(N'_R+N'_S)\hat{\sigma}_P^2}{2\sigma_P^2}\right) \tag{3.10}$$

Under the hypothesis H_1; $\alpha_R \neq \alpha_S$ and $\sigma_R^2 \neq \sigma_S^2$ and under hypothesis H_0 $\hat{\sigma}_R^2 = \sigma_R^2$ and $\hat{\sigma}_S^2 = \sigma_S^2$, thus the likelihood ratio is obtained as:

$$\ell = \sigma_P^{-(N'_R+N'_S)}\sigma_R^{N'_R}\sigma_S^{N'_S} * \exp\left(\frac{-\hat{\sigma}_P^2(N'_R+N'_S)}{2\sigma_P^2} + \frac{1}{2}\left[\frac{N'_R\hat{\sigma}_R^2}{\sigma_R^2} + \frac{N'_S\hat{\sigma}_S^2}{\sigma_S^2}\right]\right) \tag{3.11}$$

Furthermore on using the maximum likelihood estimates for the variance terms, the log likelihood ratio is derived as:

$$-\ln \ell = N'_R(\ln \hat{\sigma}_P - \ln \hat{\sigma}_R) + N'_S(\ln \hat{\sigma}_P - \ln \hat{\sigma}_S) \tag{3.12}$$

The log likelihood ratio $-\ln \ell$ is compared with an optimally chosen threshold "h" where the threshold was exceeded were considered to be change points. That is:

$$\begin{aligned} -\ln \ell > h &\Longrightarrow H_1 \quad \text{change} \\ -\ln \ell \le h &\Longrightarrow H_0 \quad \text{no change} \end{aligned} \tag{3.13}$$

Applying this method results a decision parameter for each of the MIB variables that are used in this work. Therefore a decision of an alert has to be done according to three different parameters. In order reduce the decision parameters to one parameter *Totthan et al* have suggested a combination matrix [4] which is also suggested in [1].

Thus hypothesis is defined again, H_0 is implying no change and H_1 is implying change. The expression for l is a sufficient statistic and is used to perform a binary hypothesis test. Under the hypothesis H_0, implying that no change is observed between the two Windows H_0 implying no change; likelihood l_o under hypothesis H_1, implying that a change is observed between the two windows we have, $l_1 = l$. In order to obtain a value for the likelihood ratio η that is bounded between [0 1], η is defined as follows:

$$\eta = \frac{l_0}{l_1 + l_0} \tag{3.14}$$

Furthermore, on using the maximum likelihood estimates for the variance terms in equations (3.9) and (3.10):

$$\eta = \frac{\hat{\sigma}_R^{-N'_R}\hat{\sigma}_T^{-N'_T}}{\hat{\sigma}_R^{-N'_R}\hat{\sigma}_T^{-N'_T} + \hat{\sigma}_p^{-(N'_R+N'_T)}} \tag{3.15}$$

Is derived and $N'_R = N_R - p \quad N'_T = N_T - p$, p is the order of AR model.

There is a new definition of Likelihood ratio in [23]. *Qingtao et al* have called Log likelihood Ratio (LLR) η_L as follows:

$$\eta_L = \log \frac{l_1}{l_0} \tag{3.16}$$

And after simplification, it becomes:

$$\eta_L = \log \frac{\hat{\sigma}_p^{-(N'_R+N'_T)}}{\hat{\sigma}_R^{-N'_R}\hat{\sigma}_T^{-N'_T}} \tag{3.17}$$

Input vector φ is constructed with components of likelihood ration η and $\varphi(t)$ is the Abnormality Vector which is defined as:

$$\varphi = [\eta_1 \ldots\ldots\ldots \eta n] \tag{3.18}$$

The operator matrix A was designed to obtain a scalar value $E(x)$, which is used as the measure of the anomaly. The abnormality vectors are formed into a new vector via the use of the operator matrix A, which is $E(x)$ and it is defined as the following:

$$\varphi A \varphi^T = E(x) \tag{3.19}$$

For example $\varphi(t)$ can be chosen like $\varphi_{ip} = \alpha_R[\eta_{IR}.\eta_{IDe}.\eta_{OR.}]$. η_{IR}, η_{IDe}, and $\eta_{OR.}$ represents the MIB variables Ipinreceives, Ipindelivers, and Ipoutrquest respectively. So operator A matrix is represented like;

$$A_{ip} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.20}$$

Elements of matrix A is composed of spatial correlation between IP variables. Such as, the coupling between ipIDe and ipOR is a_{23} and by symmetry $a_{21}=a_{12}$, $a_{31}=a_{13}$, and $a_{23}=a_{32}$.

The operator matrix A that is suggested in [4] as a static matrix:

$$A_{ip}\begin{bmatrix} 0{,}87 & 0{,}08 & 0{,}05 \\ 0{,}08 & 0{,}6 & 0{,}32 \\ 0{,}05 & 0{,}32 & 0{,}63 \end{bmatrix} \tag{3.21}$$

The equations that are mentioned above are based on [3, 25, 26]. The results will be derived according to this method. The above mentioned method is based on GLR(Generalized Likelihood Ration) test and the AR estimation part is based on Least Squares Estimation [27].

Using this model two different results will be shown in this work; first, Yule-Walker Estimate results will be used instead of LSE. Modified Yule-Walker estimation will not be mentioned in this work, as the results for this method does not vary with the normal Yule-Walker estimation. Second results with A operator matrix, which is derived by using SVD (Singular Value Decomposition) will be shown. These contributions are derived by me (Dağhan Hasan), Umut Güven and Derya Erhan. These contributions are a product of a collaborative study of us.

3.1.4. Yule Walker Method

Yule Walker Method will be used to estimate the AR parameters. Variance terms $\sigma_R^2, \sigma_S^2,$ and σ_P^2 will be estimated, which are the learning window variance, test window variance, and pooled window variance respectively. *Y* is the time series vector and it is representing one of the MIB variables. A time window of size *N (N=10)* and *p* is the AR model order. Let's define covariance $r(i)$:

$$r(i) = \frac{1}{N}\sum_{i=0}^{p}\{[y(1), y(2),...., y(N-i)] * [y(i+1),...y(N)]\} \tag{3.22}$$

After acquiring $r(i)$, the next step is estimating the AR parameters:

$$\begin{bmatrix} r(0) & r(-1) & \cdots & r(-p) \\ r(1) & r(0) & \cdots & \vdots \\ \vdots & \vdots & \ddots & r(-1) \\ r(p) & \cdots & \cdots & r(0) \end{bmatrix} * \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.23}$$

The above equations (3.22, 3.23) are called the Yule-Walker equations or Normal equations. Equation (3.23) can be written as:

$$\theta = [a_1, ..., a_p]^T \tag{3.24}$$

$$\begin{bmatrix} r(1) \\ \vdots \\ r(p) \end{bmatrix} + \begin{bmatrix} r(0) & \cdots & r(-p+1) \\ \vdots & \ddots & \vdots \\ r(p-1) & \cdots & r(0) \end{bmatrix} * \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.25}$$

$$r_p + R_p\theta = 0 \tag{3.26}$$

The solution is $\theta = -R_p^{-1} r_p$. Once θ is found, σ^2 can be found from the first row of Equation (3.23). The Yule-Walker method for AR spectral estimation is based directly on (3.23). Given data $y(t)$, first sample covariance $r(i)$ is obtained, the equation (3.23) is

obtained next and the equation (3.26) is solved. When θ is found, the variance (σ^2) of given vector can be found easily [28].

3.1.5. SVD to Define A Operator Matrix

SVD (Singular value decomposition) is used to define a dynamic *A* operator matrix. In general case SVD is defined as the following; S is an *d*-by-*e* matrix, which entries come from the field *K*, which is either the field of real numbers or the field of complex numbers. Then there exists a factorization of the form.

$$S = U \sum V^* \tag{3.27}$$

Where *U* is an *d*-by-*d* unitary matrix over *K*. The matrix Σ is *d*-by-*e* with nonnegative numbers on the diagonal and zeros off the diagonal. V^* denotes the conjugate transpose of *V*, an *e*-by-*e* unitary matrix over *K*. Such a factorization is called a singular-value decomposition of *S* [7]. The matrix *V* thus contains a set of orthonormal "input" or "analyzing" basis vector directions for *Si* The matrix *U* contains a set of orthonormal "output" basis vector directions for *S*. The matrix Σ contains the singular values, which can be thought of as scalar "gain controls" by which each corresponding input is multiplied to give a corresponding output.

In order to create an *A* operator matrix based on orthonormalilty of Singular Value Decomposition (SVD) method, we can define a R_φ covariance matrix:

$$R_\varphi = E\left[\varphi_k \varphi_k^H\right] \tag{3.28}$$

This can be written as by making singular value decomposition:

$$\hat{R}_K = U \sum U^H \tag{3.29}$$

By applying a whitening transform:

$$V = U^H \varphi \tag{3.30}$$

$$E[VV^H] = U^H R_\varphi U = \sum \tag{3.31}$$

$$x = \varphi^H A \varphi \tag{3.32}$$

$$V^H = \begin{bmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{bmatrix} V \tag{3.33}$$

The v values can be chosen like $v_1 = 1/\sigma_1^2$ and similarly for the other values with their corresponding variances. Then A operator matrix becomes:

$$A = U \begin{bmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{bmatrix} U^H \tag{3.34}$$

3.1.6. Results

Using the Yule-Walker Method for the estimation of the AR parameters gives the best result for the anomaly detection. As the main purpose of this part is to show the work done about the AR modeling, only the results related to the Yule-Walker method will be given. The results related to other estimation methods and the rest of the work, which is done about this field, will not be mentioned.

In the figures below, the x-axis stands for the time step and the y-axis stands for the $E(x)$ value. There are two remarkable peaks. The first one stands for the attack which is directed to the host and the second peak stands for the attack which is directed to the switch. The first figure shows the result for the operator matrix A, which is given in (3.21).

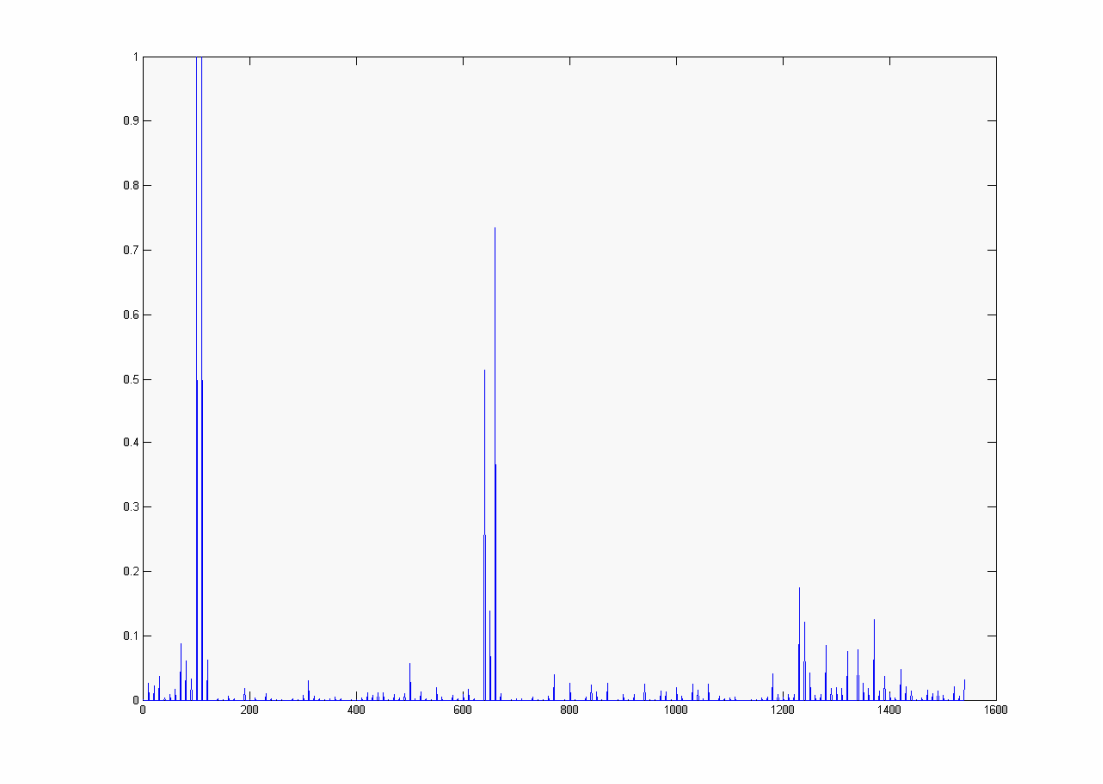

Figure 3.4. Results for Yule-Walker Method with Operater Matrix A

The second figure shows the result for the operator matrix A' which is derived by using SVD.

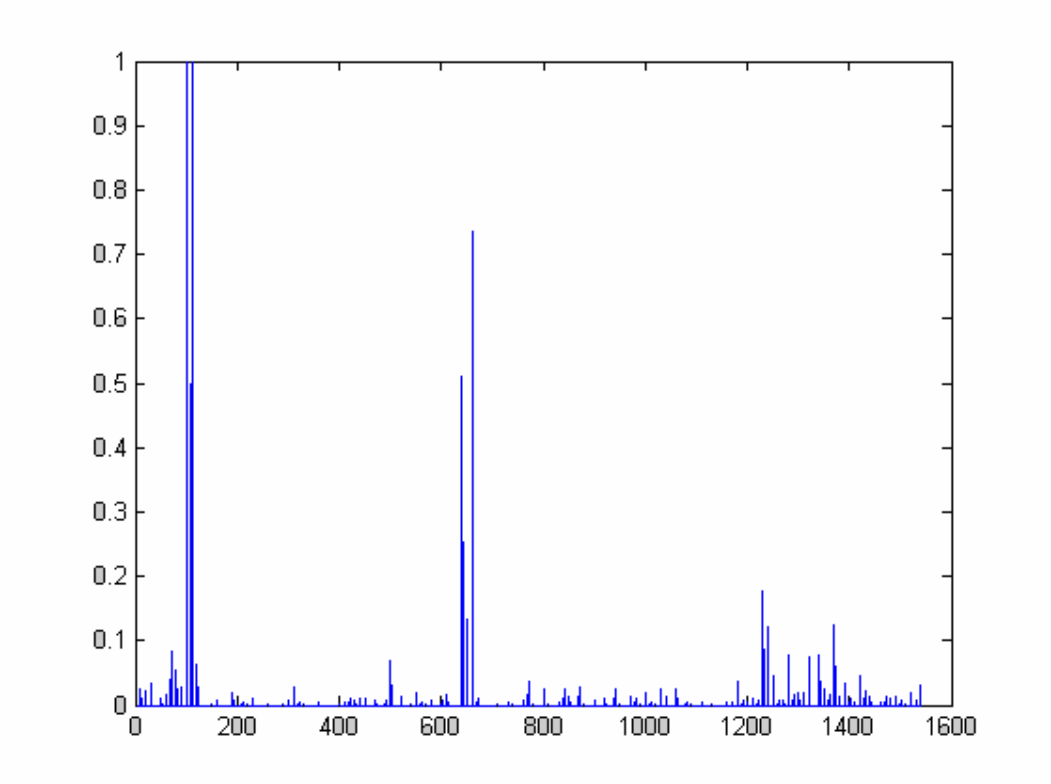

Figure 3.5. Results for Yule-Walker Method with SVD

The results in both cases do not differ much. The first attack at the time steps 110-130, which is quite remarkable in the MIB variables, is detected via this method. Also the second attack at the time steps 630-670, which can not be easily seen in the MIB variables, is detected via this method. If the threshold value for the detection of an anomaly is set to 0.5, the method results in a successful detection.

3.2. Wavelet

In the wavelet model being used in this thesis, Modulus Maxima function is used for abnormality analysis via Wavelet and the Percentage Deviation is applied for the detection of an anomalous state. The figure below shows the basic diagram of the model being used. The details of the functions used, namely Modulus Maxima and Percentage Deviation will be explained in the following chapters.

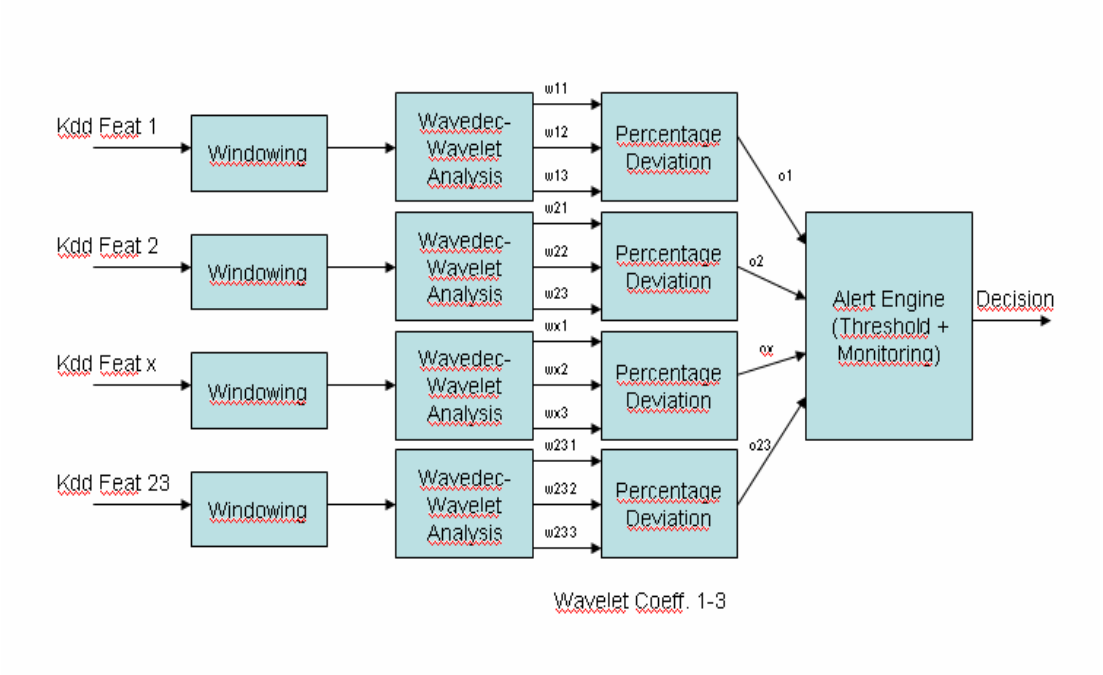

Figure 3.6. Wavelet Model

3.2.1. Wavelet-Modulus Maxima Model

For edge or singularity detection, local maxima of $\left|W^1(\lambda,t)\right|$ is of interest. When detecting the local maxima of $\left|W^1(\lambda,t)\right|$, the value of the wavelet transform at the corresponding location can be kept. [29]

Local maxima of the wavelet transform modulus will be defined next. Let $W(\lambda,t)$ be the wavelet transform of a function $x(t)$.

- Local extremum is defined as a point (λ_0, t_0) such that $(\partial W(\lambda_0, t))/(\partial(t))$ has zero crossing at $t = t_0$, when t varies.
- Modulus maximum is defined as a point (λ_0, t_0) such that $|W(\lambda_0, t)| < |W(\lambda_0, t_0)|$ when t belongs to either a right or the left neighborhood of t_0, and $|W(\lambda_0, t)| \le |W(\lambda_0, t_0)|$ when t belongs to the other side of the neighborhood of t_0.
- maxima line is defined as any connected curve in the scale pace (λ, t) along which all points are modulus maxima.

A modulus maximum (λ_0, t_0) of the wavelet transform is a strict local maximum of the modulus either on the right or the left side of the t_0 [29].

Wavelets are mathematical tools for analyzing time series or images. This work is focused on wavelet for the analysis of discrete time series. Wavelets are a relatively new way of analyzing time series in that the formal subject dates back to the 1980s, but in many aspects wavelets are a synthesis of older ideas with new elegant mathematical results and efficient computational algorithms [30].

Wavelet is a "small wave". A small wave grows and decays essentially in limited time periods. A wavelet can be notated as a real-valued function $\psi(\cdot)$, which is defined over the real axis $(-\infty, \infty)$ and satisfies two basic properties

.

1. The integral of $\psi(\cdot)$ is zero:

$$\int_{-\infty}^{\infty} \psi(u) du = 0 \tag{3.41}$$

2. The square of $\psi(\cdot)$ integrates to unity.

$$\int_{-\infty}^{\infty} \psi^2(u) du = 1 \tag{3.42}$$

Hence equations (3.41) and (3.42) lead to “small wave” or wavelet [30]. Two different wavelets are plotted in Figure 3.7.

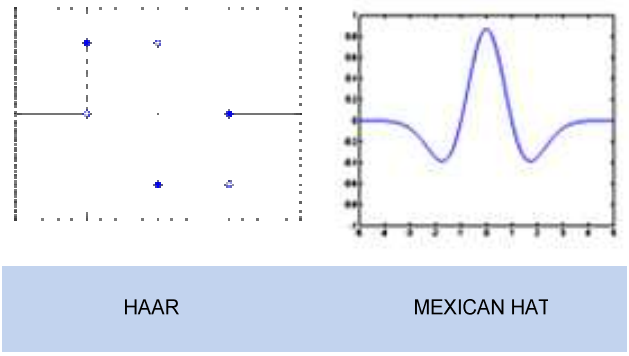

Figure 3.7. Haar and Maxican Hat wavelets

The Haar wavelet is one of the oldest wavelet. The first wavelet called the Haar wavelet and its function:

$$\psi^{(H)}(u) \equiv \begin{cases} -1/\sqrt{2}, & -1 < u \le 0; \\ 1/\sqrt{2}, & 0 < u \le 1 \\ 0, & otherwise \end{cases} \tag{3.43}$$

Wavelets show how weighted averages of certain functions vary from one averaging period to the next. This interpretation of wavelet analysis is a key concept. This can be explained as the following; For the real-valued function $x(\cdot)$, $\alpha(a,b)$ is defined as the average value of $x(\cdot)$:

$$\frac{1}{b-a}\int_a^b x(u)du \equiv \alpha(a,b), \tag{3.44}$$

The length of the interval and the center of interval can be considered as $\lambda \equiv b-a$ and $t=(a+b)/2$. Using λ and t, following equation can be defined as:

$$A(\lambda,t) \equiv \alpha(t-\frac{\lambda}{2}, t+\frac{\lambda}{2}) = \frac{1}{\lambda}\int_{t-\frac{\lambda}{2}}^{t+\frac{\lambda}{2}} x(u)du. \tag{3.45}$$

$A(\lambda,t)$ is defined as the average value of the signal $x(\cdot)$ over a scale of λ centered about time t. In order to measure the changes between the averages following equation can be defined as:

$$D(\lambda,t) \equiv A(\lambda,t+\frac{\lambda}{2}) - A(\lambda,t-\frac{\lambda}{2}) = \frac{1}{\lambda}\int_{t}^{t+\lambda} x(u)du - \frac{1}{\lambda}\int_{t-\lambda}^{t} x(u)d(u). \tag{3.46}$$

For example, A plot of $D(1,t)$ would show, how quickly the daily average temperature is changing from one day to the next. Similarly, by increasing the scale λ up to a year, a plot of $D(1,t)$ would show how much the yearly average temperature is changing from one year to next. Because that two integrals in Equation (3.46) involve adjacent non-overlapping intervals, it can be combined into a single interval over the entire real axis to obtain:

$$D(\lambda,t) = \int_{-\infty}^{+\infty} \widetilde{\psi}_{\lambda,t}(u)x(u)du, \tag{3.47}$$

Where,

$$\widetilde{\psi}_{\lambda,t}(u) \equiv \begin{cases} -1/\lambda, & t-\lambda < u \le t; \\ 1/\lambda, & t < u \le t+\lambda; \\ 0, & otherwise \end{cases} \tag{3.48}$$

If Equation (3.48) is compared to Haar wavelet, it can be noted that $\widetilde{\psi}_{1,0}(u) = \sqrt{2}\psi^{(H)}(u)$. the scheme of looking at differences on unit scale, integrating the product with the signal $x(\cdot)$;

$$\int_{-\infty}^{\infty} \psi^{(H)}(u)x(u)du \equiv W^{(H)}(1,0), \tag{3.49}$$

The Haar wavelet extracts information about how much difference there is between the two unit scale averages of $x(\cdot)$ bordering on time $t = 0$ [30].

3.2.2. DWT

DWT (Discrete Wavelet Transform) will can be thought as a judicious sub-sampling of $W(\lambda,t)$ in which just dyadic scales and then within a given dyadic scale 2^{j-1} are dealt with at times t that are separated by multiples of 2^j [30].

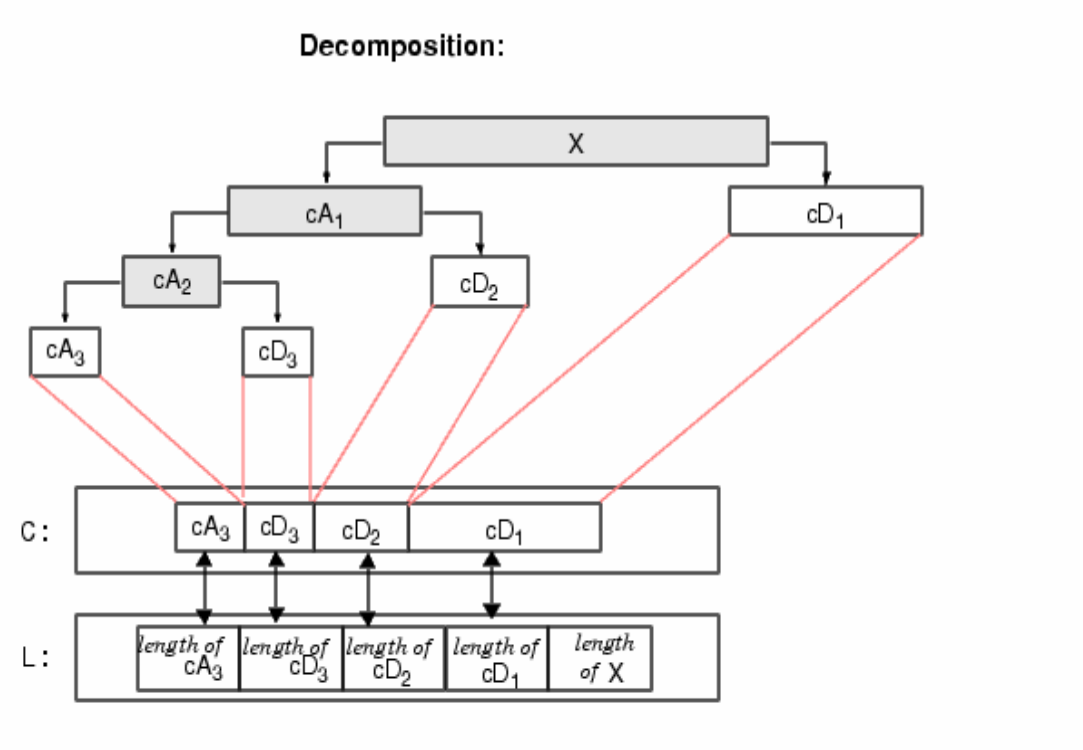

Figure 3.8. DTW Decomposition

MODWT (Maximal Overlap DWT) will be explained as the next step, MODWT can be thought of as a sub-sampling of the CWT (Continuous Wavelet Transform) at dyadic scales are dealt all times t [30].

The detailed coefficients, approximate coefficients, Wavelet filter, Scaling Filter, Low Pass filter, High Pass filter and relation between these phonemes will be explained briefly.

The space $L^2(R)$ is built up of the set of "rings" that are differences between two consecutive spaces. These difference spaces are denoted by W_m and are defined as orthogonal complement of spaces V_m with respect to V_{m-1}:

$$V_{m-1} = V_m \oplus W_m, \quad V_m \perp W_m \tag{3.50}$$

Let $\psi(x) = \psi_{0,0}(x)$ be a basis function of W_0. Since $\psi_{0,0}(x) \in W_0 \subset V_{-1}$ it can be written as:

$$\psi_{0,0}(x) = 2^{1/2} \sum_n g_n \phi_{-1,n}(x), \tag{3.51}$$

$\phi(x), g_n$ are scaling filter and high pass filter respectively. There are strong relation between $\psi(x)$, $\phi(x)$, g_n, and h_n.

$$\Phi(\omega) = \prod_{m=1}^{\infty} H(\frac{\omega}{2^m}) \tag{3.52}$$

Rewriting (3.51) in frequency domain:

$$\Psi(\omega) = G(\frac{\omega}{2})\Phi(\frac{\omega}{2}) \tag{3.53}$$

And replacing $\Phi(\omega)$ using the infinite product of (3.52) yields:

$$\Psi(\omega) = G(\frac{\omega}{2})\prod_{m=1}^{\infty} H(\frac{\omega}{2^m}) \tag{3.54}$$

Next, the relation between two sequences g_n and h_n is defined as:

$$g_n = (-1)^n h_{-n+2t+1} \tag{3.55}$$

The equivalent of (3.55) in frequency domain is;

$$G(\omega) = -H(-\omega + \pi)e^{+i\omega(2t+1)} \tag{3.56}$$

With wavelet functions, any function $x(\cdot)$ can be written in $L^2(R)$ as a sum of projections on W_j, $j \in R$:

$$x(t) = \sum_{j=-\infty}^{\infty} e_j(t) \tag{3.57}$$

Where,

$$e_j(t) = \sum_k \left\langle \psi_{j,k}(t), x(t) \right\rangle \psi_{j,k}(t) \tag{3.58}$$

If at a certain scale *m* is stopped, then the function $x(\cdot)$ can be written as the sum of a low resolution part $x_m(t) \in V_m$ and number of detailed parts $e_j(t) \in W_j$;

$$\begin{aligned} x(t) &= x_m(t) + \sum_{j=-\infty}^{m} e_j(t) \\ &= \sum_n \left\langle \phi_{m,n}(t), x(t) \right\rangle \phi_{m,n}(t) + \sum_{j=-\infty}^{m} \sum_k \left\langle \psi_{j,k}(t), x(t) \right\rangle \psi_{j,k}(t) \end{aligned} \tag{3.59}$$

$$x(t) = \sum_n c_{m,n} \phi_{m,n}(t) + \sum_{j=-\infty}^{m} \sum_k d_{j,k} \psi_{j,k}(t) \tag{3.60}$$

$x(n)$ is the discrete version of its continues time version and can be decomposed in two functions $x_1(n) \in V_1$ and $e_1(n) \in W_1$ containing the overall characteristics and details of $x_0(n)$, respectively,

$$x_0(n) = x_1(n) + e_1(n) = \sum_k c_{1,k} \phi_{1,k}(x) + \sum_k d_{1,k} \psi_{1,k}(x) \tag{3.61}$$

Now two new sequences $c_{1,k}$ and $d_{1,k}$ have been generated. $c_{1,k}$ is the approximate coefficients and $d_{1,k}$ is the detailed coefficients. It is possible to calculate the coefficients $c_{1,k}$ and $d_{1,k}$ iteratively from the previous scale $j-1$ without explicit use of the functions $\phi(x)$ and $\psi(x)$. For arbitrary j, it can be defined as:

$$c_{j,k} = 2^{1/2} \sum_{n} c_{j-1,n} h_{n+2k,} \tag{3.62}$$

$$d_{j.k} = 2^{1/2} \sum_{n} c_{j-1,n} g_{n+2k,} \tag{3.63}$$

Description of the decomposition process is completely discrete. The sequences h_n and g_n are called filters.

3.2.3. Percentage Deviation

Percentage deviation algorithm is based on median calculation of input vector,

$$PD_X = (x - medain(X)) * 100 \tag{3.64}$$

Median is in probability theory and statistics, a number dividing the higher half of a sample, a population, or a probability distribution from the lower half. The median of a finite list of numbers can be found by arranging all the observations from lowest value to highest value and picking the middle one. Then we will go one step further, average of Percentage Deviation;

$$PD_{avg} = \sum_{i=1}^{n} PD_i / n \tag{3.65}$$

This PD_{avg} gives us the decision threshold. If $PD_x > PD_{avg}$ it shows us there is an anomaly at point $x(i)$, else is normal condition. [9]

4. WAVELET MODEL

4.1. Model Architecture

The functions of the Wavelet model will be explained, which is applied to the KDD data set. The figure of the model being used, can be seen below:

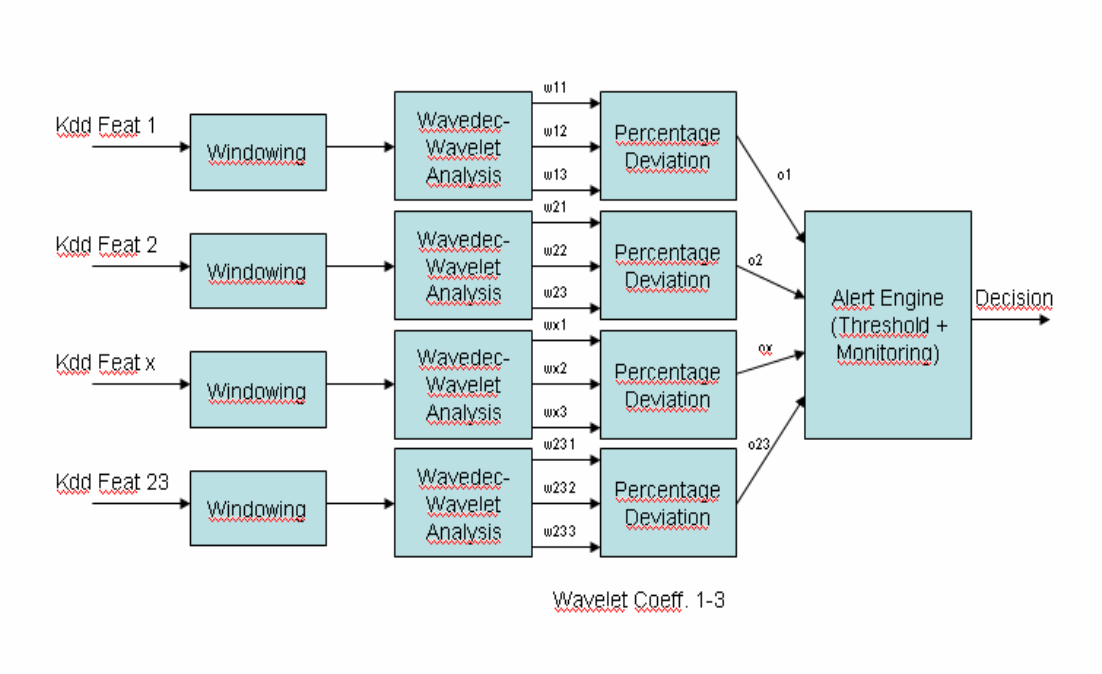

Figure 4.1. Wavelet Model

The inputs to this model are the features from the KDD data set. There are a total of 41 Features in the KDD data set. The descriptions of these features are given in the previous chapters. Not all of these 41 features are useful for the detection of an anomalous state. As the total number of connections in this data set is huge, there was a need to reduce the number of these Features.

Some of the features in the KDD data set do not carry any change at all, i.e. the feature has a single scalar value at all time steps. These features are not useful for the method, as they do not carry any information. There are also features which resemble each other, i.e. they carry nearly the same information. These features are also put aside and are not given as inputs to the model.

There are a total of 23 Features, i.e. 23 different input vectors, which are processed at the same time by the model. These 23 different features, which are taken from the KDD data set are as the following:

- Feature-1=> duration: Length (number of seconds) of the connection.
- Feature-2=> dst_bytes: Number of data bytes from destination to source.
- Feature-3=> wrong_fragment: Number of ``wrong'' fragments.
- Feature-4=> hot: Number of ``hot'' indicators.
- Feature-5=> num_failed_logins: Number of failed login attempts.
- Feature-6=> logged_in: 1 if successfully logged in; 0 otherwise.
- Feature-7=> num_compromised: Number of “compromised” conditions.
- Feature-8=> root_shell: 1 if root shell is obtained; 0 otherwise.
- Feature-9=> num_file_creations: number of file creation operations.
- Feature-10=> num_shells: Number of shell prompts.
- Feature-11=> num_access_files: Number of operations on access control files.
- Feature-12=> is_guest_login: 1 if the login is a “guest” login; 0 otherwise.
- Feature-13=> count: Number of connections to the same host as the current connection in the past two seconds.
- Feature-14=> serror_rate: % of connections that have “SYN” errors.
- Feature-15=> srv_serror_rate: % of connections that have “SYN” errors.
- Feature-16=> rerror_rate: % of connections that have “REJ” errors.
- Feature-17=> srv_rerror_rate: % of connections that have “REJ” errors.
- Feature-18=> same_srv_rate: % of connections to same services.
- Feature-19=> diff_srv_rate: % of connections to different services.
- Feature-20=> srv_diff_host_rate: % of connections to different hosts.
- Feature-21=> dst_host_count: Number of connections to the same host.
- Feature-22=> dst_host_same_srv_rate: % of connections to the same host.
- Feature-23=> dst_host_diff_srv_rate: % of connections from different hosts.

The first processing, that is applied to these Input features is the “Windowing” function. The total data set, which is used in this work, contains a total of 32000 connections. It is not desired to process all these connections as one window. The main reason for this is to prevent the domination of peaks in the whole window. As some peaks in the features may cause, other valuable connection information to be lost. One other reason is to improve the computational efficiency. Therefore a total window size of 3000 connections has been chosen. Each window follows another window after 1250 connections. By doing so, alerts that are given in different windows but at the same connection, would show that there is a high probability of an anomalous state. This condition can be presented with the figure below:

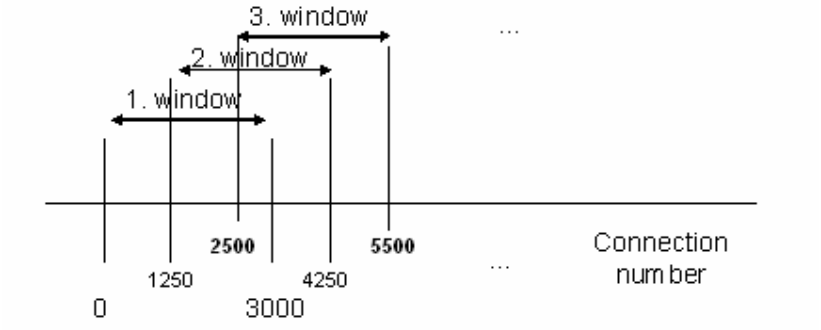

Figure 4.2. Window Shifts

The output for the “Windowing” functions for all inputs (23 Features) results as 23 vectors, which contain the information related to 3000 connections. There will be a total of 32 windows, which will be processed by the model.

The next function in the model is the “Wavedec-Wavelet Analysis” function. The function in Matlab, which is used for wavelet analysis, is defined as “Wavedec”. This Matlab function is stands for the “Wavedec-Wavelet Analysis” function in the model. This Matlab function is described as the following: [*C, L*] = wavedec(*X*, *N*, 'wname') returns the wavelet decomposition of the signal *X* at level *N*, using 'wname'. *N* must be a strictly positive integer. Pre-defined Mother-Wavelets can be used in Matlab. The output decomposition structure contains the wavelet decomposition vector *C* and the bookkeeping vector *L* [12].

Only the low pass filter output of Wavelet analysis has been used, i.e. only the approximate coefficients $c_{1,k}$ of the Wavelet analysis has been used. In this work 3rd level of wavelet decomposition has been used. So the outputs of the “Wavedec-Wavelet Analysis” functions will be three different wavelet coefficients for all our input features, i.e. 23 features. Therefore the input for the following function, the “Percentage Deviation”, consists of 23x3=69 vectors.

The range of the Input values for each feature varies. The wavelet coefficients are the outputs of the “Wavedec” function. In the figure below, the approximate level-1 coefficients of three different features are given. In the first figure, there is a peak with a value of 1,2. In the second figure, there is a peak with a value of 50000. The peaks in the last figure take values in the range of 2000-3000. There is other valuable information hidden in this data, which can diminish due to these peaks. In order to make better use of these information, a normalization function is applied to the output of the “Percentage Deviation” function.

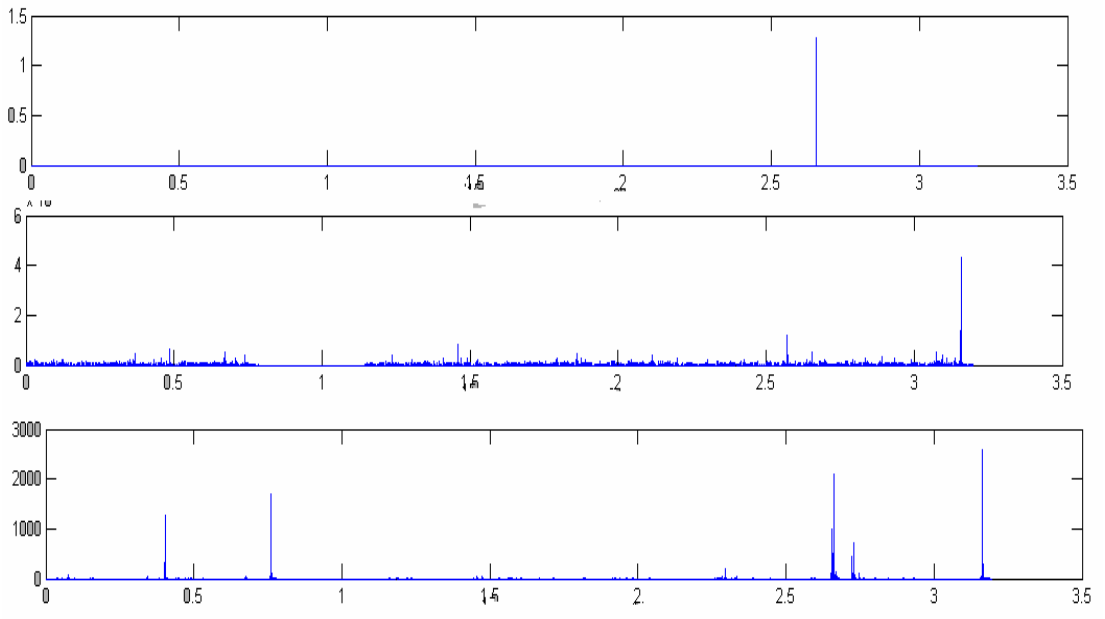

Figure 4.3. Features with huge peaks.

As explained, in the previous chapter, "Percentage Deviation" algorithm is based on median calculation of input vector: $PD_X = (x - medain(X)) * 100$. Then as the next step, the average of Percentage Deviation: $PD_{avg} = \sum_{i=1}^{n} PD_i / n$ is taken. As described above, a normalization operation is done to the PD_{avg} value. This normalization function normalizes all values in the vector in terms of the maximum value in the vector. As the next operation, all values in the function are added with some constant value and normalized to 1 again. Therefore, there is no longer a huge gap between the peak and the mean value of the vector. If the vector values at point $x(i)$ are greater than the normalized PD_{avg} value, it shows that there is an anomaly at point $x(i)$, else it is a normal condition.

The output of the "Percentage Deviation" function becomes 23X3=69 vectors with binary values, i.e. a "1" indicating an anomaly and a "0" indicating a normal state. All these outputs are given to the "Alert Engine" of our model. A threshold value is set in the "Alert Engine". If the number of alerts for the same connection is bigger than this threshold value, an alert of an anomalous state is given. This is the final decision given by the whole model, which indicated an anomaly or normal state.

The idea behind using wavelet analysis is to eliminate the high variations from input vector while not affecting the input vector's originality. It gives good detection ratio. A detailed explanation of the results will be shown with multiple examples. In the next chapter the performance comparison for different threshold values, which is set at the "Alert Engine" will be shown.

4.2. Performance Measurements

4.2.1. Data Set and Attack Instances

The data set which is given as an input to the "Wavelet-model" consists of 32000 different connections. Among these connections, there are a total of 23 attack instances, which include 11 different attacks. The attack instances and the related attack names can be found below:

Table 4.1. Attack Instances

Attack Instances:	**Attack Names:**
745-746	buffer overflow
4050	loadmodule
4114	perl
7602-7603	neptune
7794-11488	smurf
15700	guess password
15785-15804	pod
19287-19385	teardrop
22751-22802	guess password
22815-22831	portsweep
26527-26533	ipsweep
26584-26590	ipsweep
26596-26602	ipsweep
26612-26618	ipsweep
26633-26639	ipsweep
26652-26659	ipsweep
26672-26678	Ipsweep
26687-26693	Ipsweep

Table 4.2. Attack Instances Continued

26712-26717	ipsweep
26728-26734	ipsweep
26749-26756	ipsweep
26767-26774	ipsweep
31690	land

4.2.2. Intrusion Detection Capability

It has always been a problem to measure the performance of IDS systems. The common parameters that have been used are the False Positive and False Negative values. But as there is a trade-off between these 2 values, it is not easy to decide how to tune the IDS and where to stop. A new metric called Intrusion Detection Capability, C_{ID} is proposed. C_{ID} is simply the ratio of the mutual information between IDS input and output, and the entropy of the input. C_{ID} has the desired property that:

- It considers all the important aspects of detection capability naturally, i.e., true positive rate, false positive rate, positive predictive value, negative predictive value, and base rate.
- It objectively provides an intrinsic measure of intrusion detection capability.
- It is sensitive to IDS operation parameters.

In order to fine tune an IDS system, the C_{ID} value is desired to be maximized. The maximum C_{ID} value that is achieved would be the point where the IDS system is able to function to its fullest.

The performance parameters, that are used for the evaluation of C_{ID} are as the following: B or $P(I)$ is the base rate of data, FP (False Positive, $P(A/I')$), TP (True Positive, $P(A/I)$), FN (False Negative, $P(A'/I)$), TN (True Negative, $P(A'/I')$), PPV (Positive Predictive Value or Bayesian Detection Rate, $P(I/A)$), NPV (Negative Predictive Value, $P(I'/A')$), and C_{ID} (Intrusion Detection Capability).

False positive rate (FP) is the probability that the IDS outputs an alarm when there is no intrusion. False negative rate (FN) is the probability that the IDS output no alarm when there is an intrusion.

There are also two other parameters that are used for the performance measurement of IDS systems. They are the positive predictive value (PPV) and negative predictive value (NPV). PPV is the probability of when there is an intrusion, the IDS will give an alert, NPV is the probability of when there is no intrusion, the IDS will not give any alert.

Base rate can be entered as a piece of prior information about the IDS operational environment in the Bayesian equations.

C_{ID} is simply the ration of the mutual information between IDS input and output, and the entropy of the input[32]. It can be calculated via the equations below:

$$PPV = P(I/A) = \frac{B(1-FN)}{B(1-FN)+(1-B)FP} \tag{4.1}$$

$$NPV = P(I'/A') = \frac{(1-B)(1-FP)}{(1-B)(1-FP)+B.FP} \tag{4.2}$$

$$C_{ID} = \frac{I(X;Y)}{H(X)} = \frac{(H(X)-H(X/Y))}{H(X)} \tag{4.3}$$

Where the values for $H(X)$ and $H(X/Y)$ can be calculated with the following equations:

$$H(X) = -B\log B - (1-B)\log(1-B) \tag{4.4}$$

$$H(X/Y) = -B(1-FN)\log\frac{B(1-FN)}{B(1-FN)+(1-B)FP} - B.FN.\log\frac{B.FN}{B.FN+(1-B)(1-FP)} \tag{4.5}$$

$$-(1-B)(1-FP).\log\frac{(1-B)(1-FP)}{(1-B)(1-FP)+B.FN} - (1-B).FP.\log\frac{(1-B)FP}{(1-B).FP+B(1-FN)}$$

4.3. Results

The results in this work will be measured according to this method. As it was explained in the previous chapter, the alerts are given according to a Threshold in the alert engine. For example, for a given threshold value of 2, if there is 2 or more "1"s at the same instance among the total 69 outputs of the Percentage Deviation module.

There is always a trade off between the threshold value and the model performance. Increasing the threshold value decreases the amount of false positives but also at the same time it increases the amount of false negatives.

The results will be shown below according to different Threshold values for different wavelets and different window shifts. Each Figure consists 32000 connection steps. All of the real attacks are shown as red bars in the figures. The blue bars indicate the alerts given by the wavelet model.

For Daubechies 6 Wavelets, for a threshold value of 1:

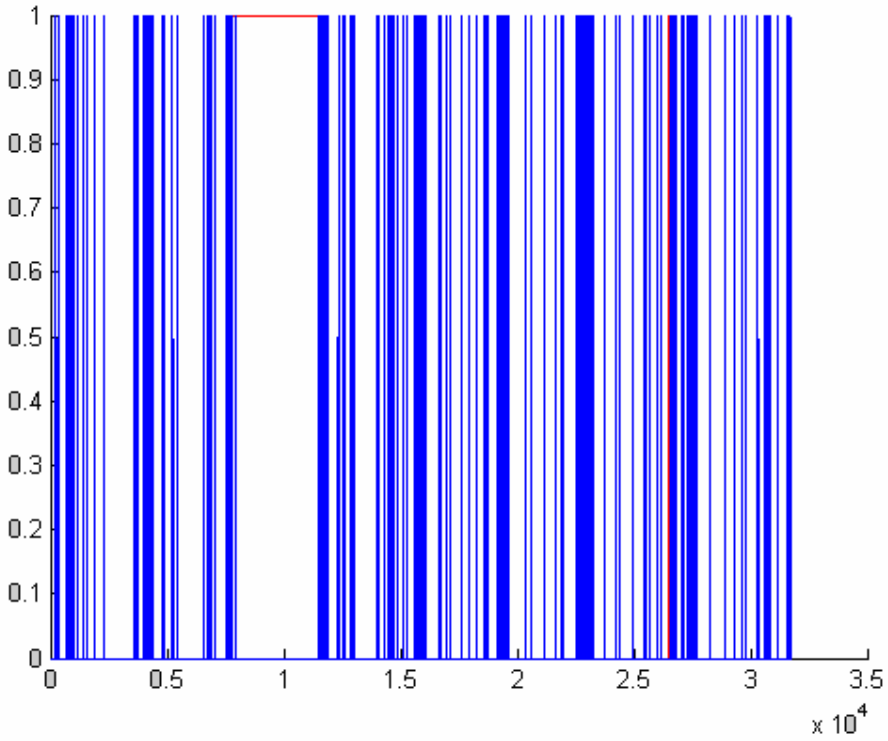

Figure 4.4. Alerts for a threshold value of 1

Putting a threshold value of 1 at the alert module results a large amount of false positives, but there is no false negatives at all. This means that all of the attacks in the analyzed data are detected successfully by our wavelet model. But there are also alerts that were given at 387 instances, indicating that there are attacks at the related instances, while there is actually none at all. If the network administrators concern is not to miss any attacks, using this wavelet model with a threshold value of 1 would satisfy his concerns.

Considering that there are a total of 32000 connections in this whole data, having 387 false positives is still not a very problematic situation. The network administrator would have to check 387 false alerts unnecessarily and this would bring some additional work to the administrator. But in order to be able to detect all attacks or anomalies, this additional work can be done. The C_{ID} value is about 0.9 which is a good detection value.

Table 4.3. Number of attacks instances, false positives and false negatives for a threshold of 1.

	Attack Instances	False Positives		False Negatives	
#	4139	387		0	
B	FP	FN	PPV	NPV	Cid
0.129	0.0121	0	0.9245	1	0.9029

For Daubechies 6 Wavelets, for a threshold value of 2:

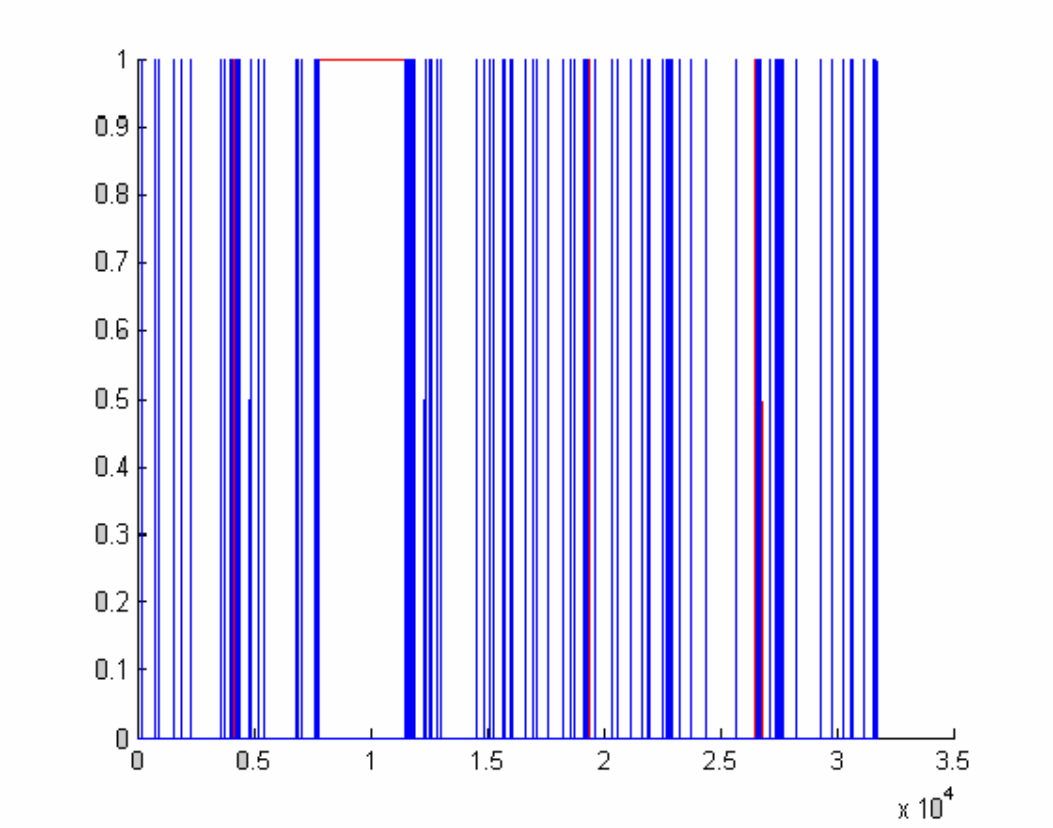

Figure 4.5. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a large amount of false positives and a low amount of false negatives. There are alerts that were given at 96 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 22 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 96 false positives and 22 false negatives is still not a very problematic situation. Although there are some attacks that were not detected by the model, the C_{ID} value is increased to 0.9643, which is a very good value.

Table 4.4. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	False Positives		False Negatives	
#	4139	96		22	
B	FP	FN	PPV	NPV	Cid
0.129	0.0030	0.000687	0.9801	0.9999	0.9643

For Daubechies 6 Wavelets, for a threshold value of 3:

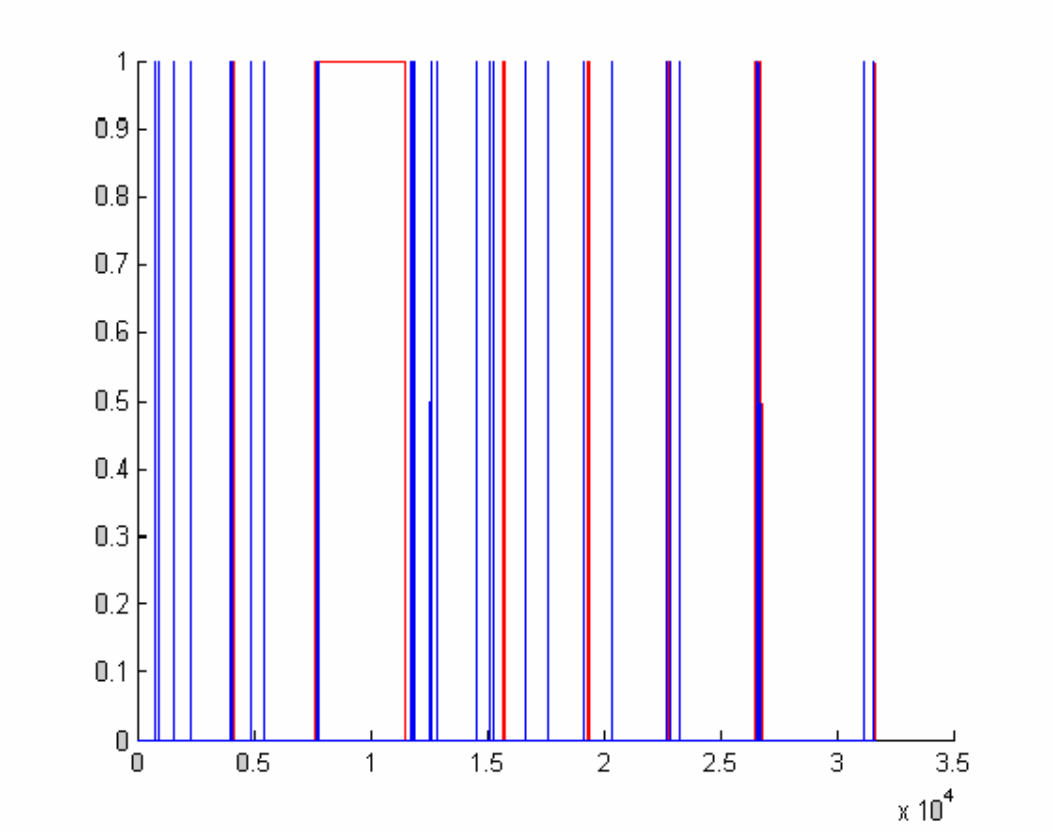

Figure 4.6. Alerts for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives and a large amount of false negatives. There are alerts that were given at 15 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 176 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 15 false positives and 176 false negatives is still not a very problematic situation. The C_{ID} value for this threshold value is the maximum value that could be achieved for Daubechies 6 Wavelets.

Table 4.5. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	**False Positives**		**False Negatives**	
#	**4139**	**15**		**176**	
B	**FP**	**FN**	**PPV**	**NPV**	**Cid**
0.129	**0.000468**	**0.0055**	**0.9968**	**0.9992**	**0.9779**

For Daubechies 6 Wavelets, for a threshold value of 4:

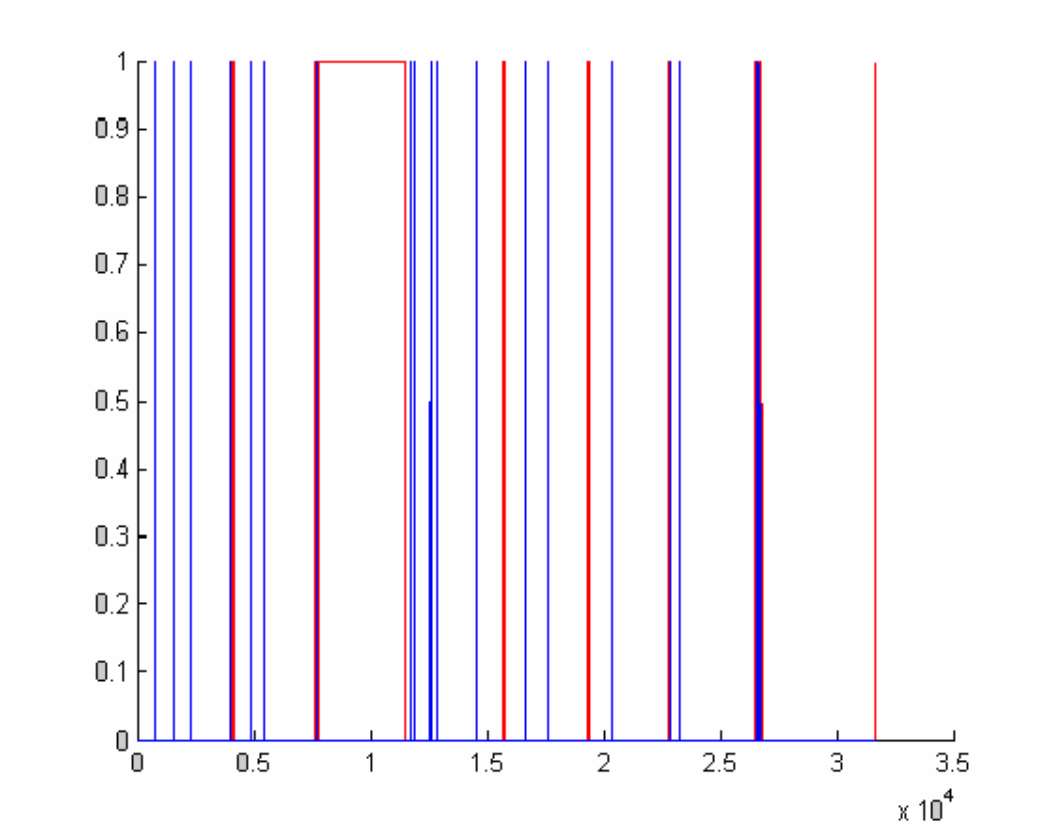

Figure 4.7. Alerts for a threshold value of 4

Putting a threshold value of 4 at the alert module results a low amount of false positives and a very large amount of false negatives. There are alerts that were given at 8 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 4012 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 15 false positives is not a big issue but having 4012 false negatives is almost the same as missing all of the attacks in the data. The C_{ID} value for this threshold value is 0.7858, which is quite low when compared to the other threshold values.

Table 4.6. Number of attacks instances, false positives and false negatives for a threshold of 4.

.	Attack Instances	False Positives		False Negatives	
#	4139	8		4012	
B	FP	FN	PPV	NPV	Cid
0.129	0.00025	0.1254	0.9981	0.9818	0.7858

For Daubechies 6 Wavelets, for a threshold value of 10:

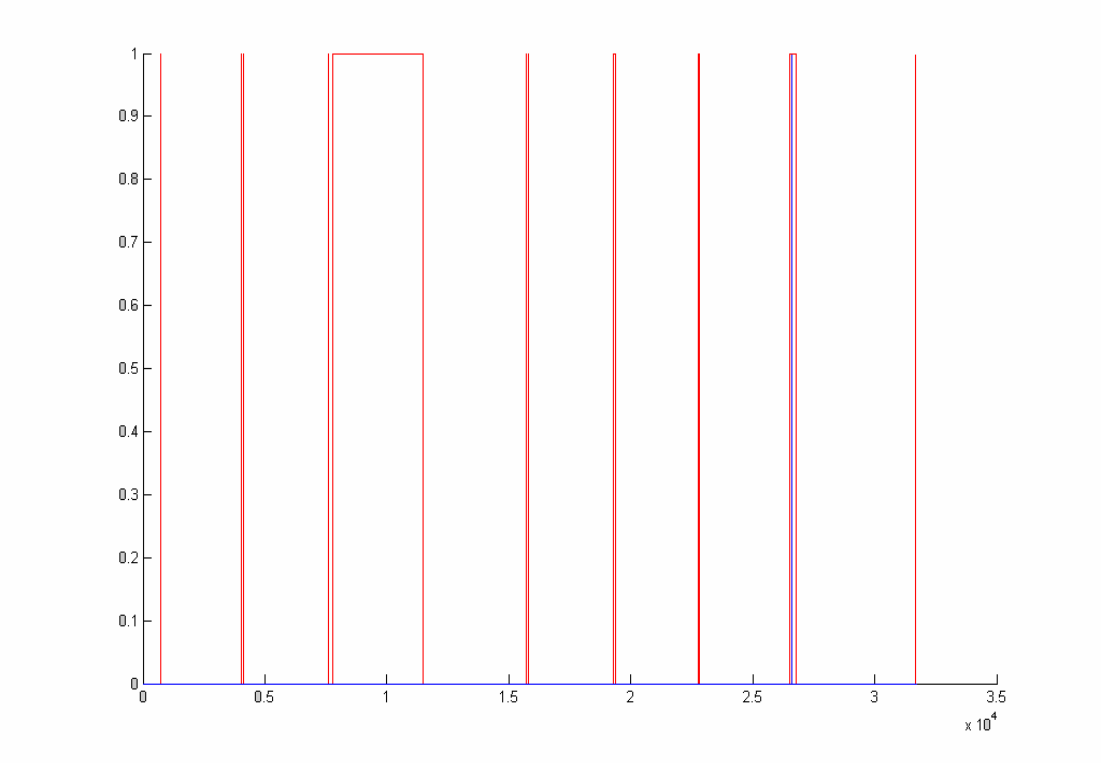

Figure 4.8. Alerts for a threshold value of 10

Putting a threshold value of 10 at the alert module results zero false positives and a very large amount of false negatives. There are 4014 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 4014 false negatives is almost the same as missing all of the attacks in the data. The C_{ID} value for this threshold value is 0.7897, which is quite low when compared to the other threshold values.

Table 4.7. Number of attacks instances, false positives and false negatives for a threshold of 10.

	Attack Instances	False Positives		False Negatives	
#	4139	0		4014	
B	FP	FN	PPV	NPV	Cid
0.129	0	0.1254	1	0.9818	0.7897

For Daubechies 2 Wavelets, for a threshold value of 1:

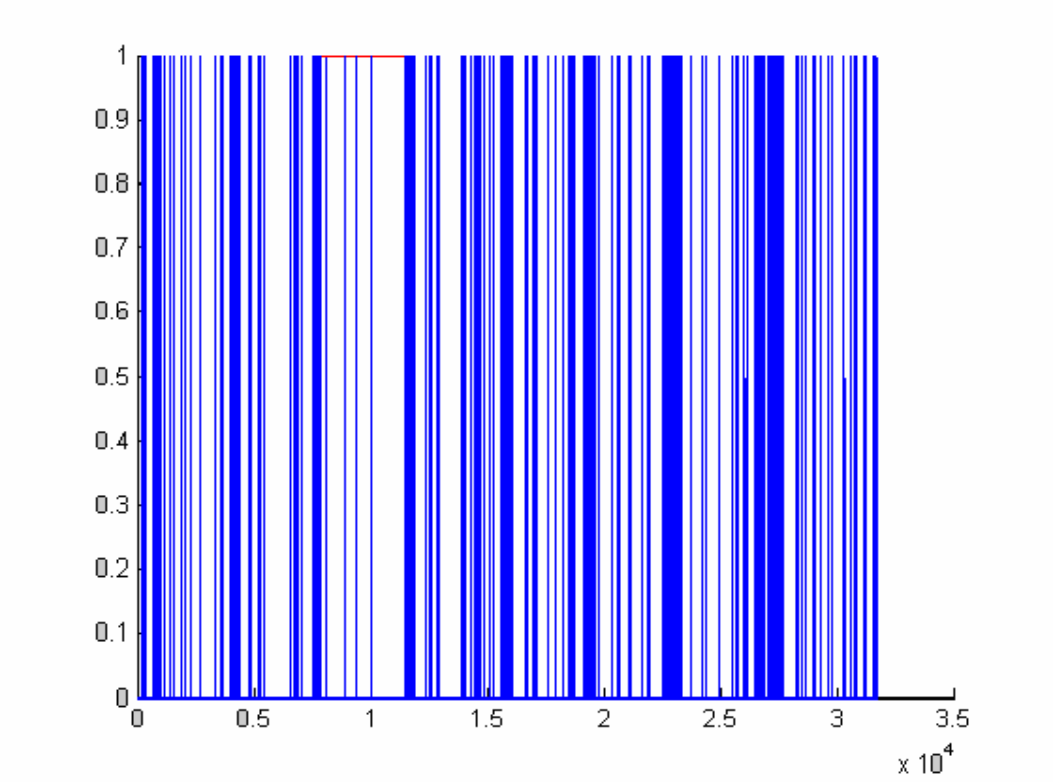

Figure 4.9. Alerts for a threshold value of 1

Putting a threshold value of 1 at the alert module results a large amount of false positives, which is lower when compared to Daubechies 6 results. There is only 1 false negative. There are alerts that were given at 340 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there is 1 attack instance which was not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 340 false positives and 1 false negative is not a big issue. The C_{ID} value for this threshold value is higher by 0.01 when compared to Daubechies 6 results.

Table 4.8. Number of attacks instances, false positives and false negatives for a threshold of 1.

	Attack Instances	False Positives		False Negatives	
#	4139	340		1	
B	FP	FN	PPV	NPV	Cid
0.129	0.0106	0.000031	0.9331	1	0.9115

For Daubechies 2 Wavelets, for a threshold value of 2:

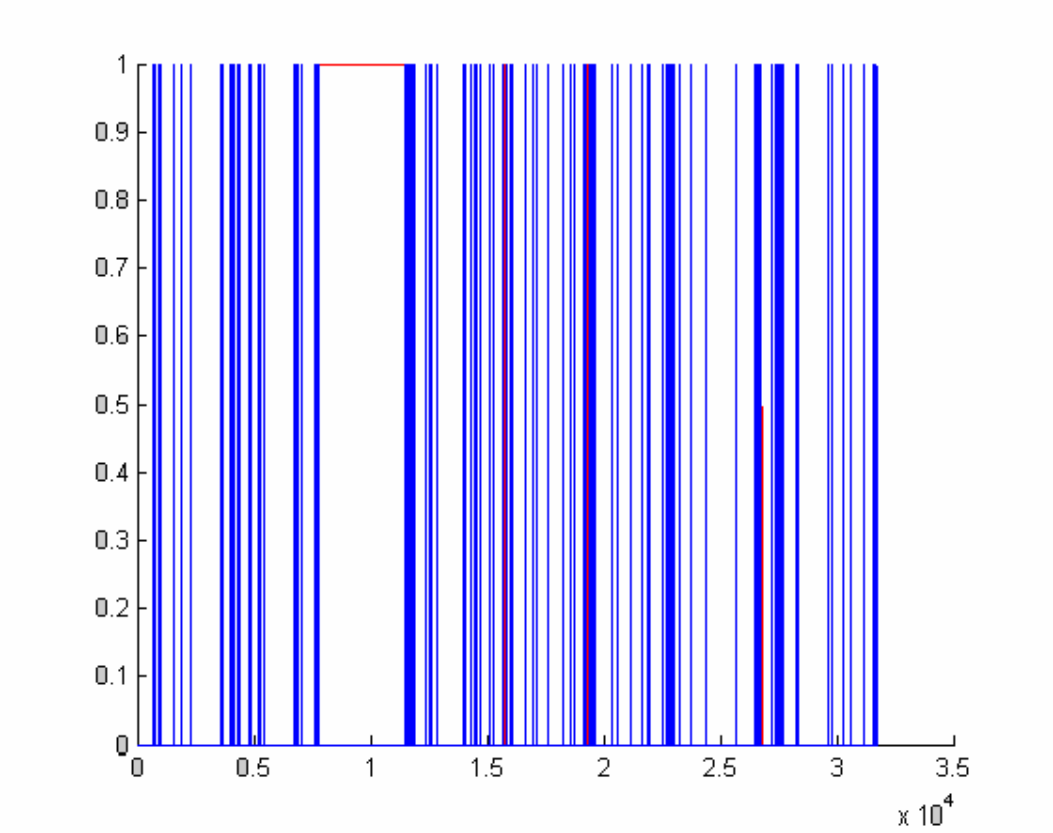

Figure 4.10. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a large amount of false positives, which is a bit larger when compared to Daubechies 6 results. There is only 2 false negatives, which is quite low when compared to Daubechies 6 results. There are alerts that were given at 97 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there is 2 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 97 false positives and 2 false negative is not a big issue. The C_{ID} value for this threshold value is only a bit higher when compared to Daubechies 6 results.

Table 4.9. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	**False Positives**		**False Negatives**	
#	**4139**	**97**		**2**	
B	**FP**	**FN**	**PPV**	**NPV**	**Cid**
0.129	**0.003**	**0.000062**	**0.9799**	**1**	**0.9661**

For Daubechies 2 Wavelets, for a threshold value of 3:

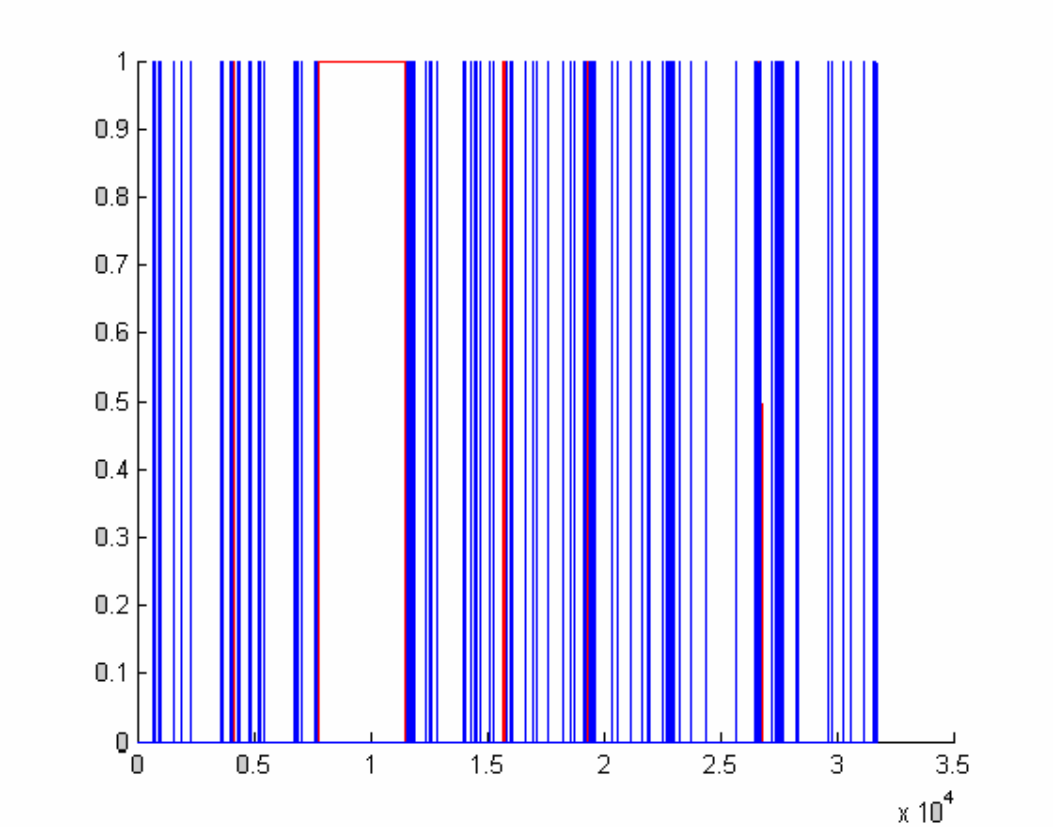

Figure 4.11. Alerts for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives, which is a bit larger when compared to Daubechies 6 results. There are 3858 false negatives, which is quite larger when compared to Daubechies 6 results. There are alerts that were given at 43 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 3858 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 43 false positives and 3858 false negative is almost the same as not being able to detect any attacks, therefore the C_{ID} value for this threshold value is 0.7794, which is much lower when compared to Daubechies 6 results.

Table 4.10. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	False Positives		False Negatives	
#	4139	43		3858	
B	FP	FN	PPV	NPV	Cid
0.129	0.0013	0.1206	0.9898	0.9824	0.7794

For Coiflet 2 Wavelets, for a threshold value of 2:

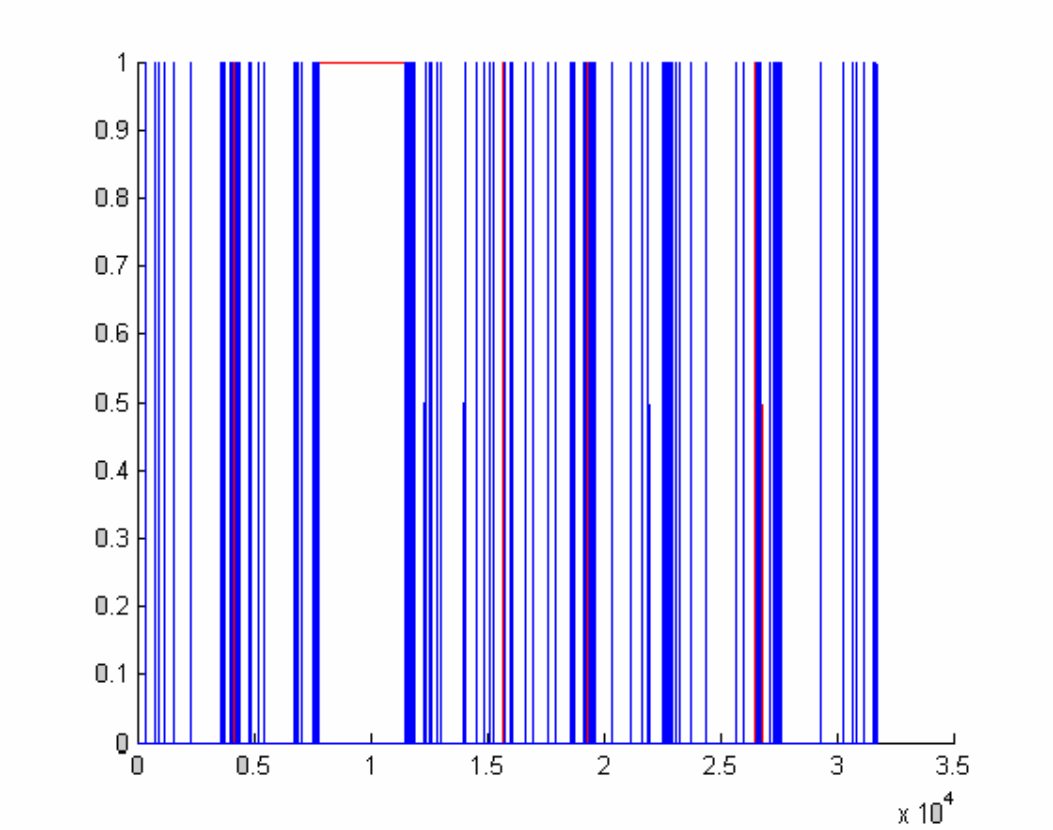

Figure 4.12. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a low amount of false positives and a low amount of false negatives. There are alerts that were given at 88 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 23 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 88 false positives and 23 false negative is not a big issue, therefore the C_{ID} value for this threshold value is 0.9664, which is a good detection value

Table 4.11. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	False Positives		False Negatives	
#	4139	88		23	
B	FP	FN	PPV	NPV	Cid
0.129	0.0027	0.00071	0.9818	0.9999	0.9664

For Coiflet 2 Wavelets, for a threshold value of 3:

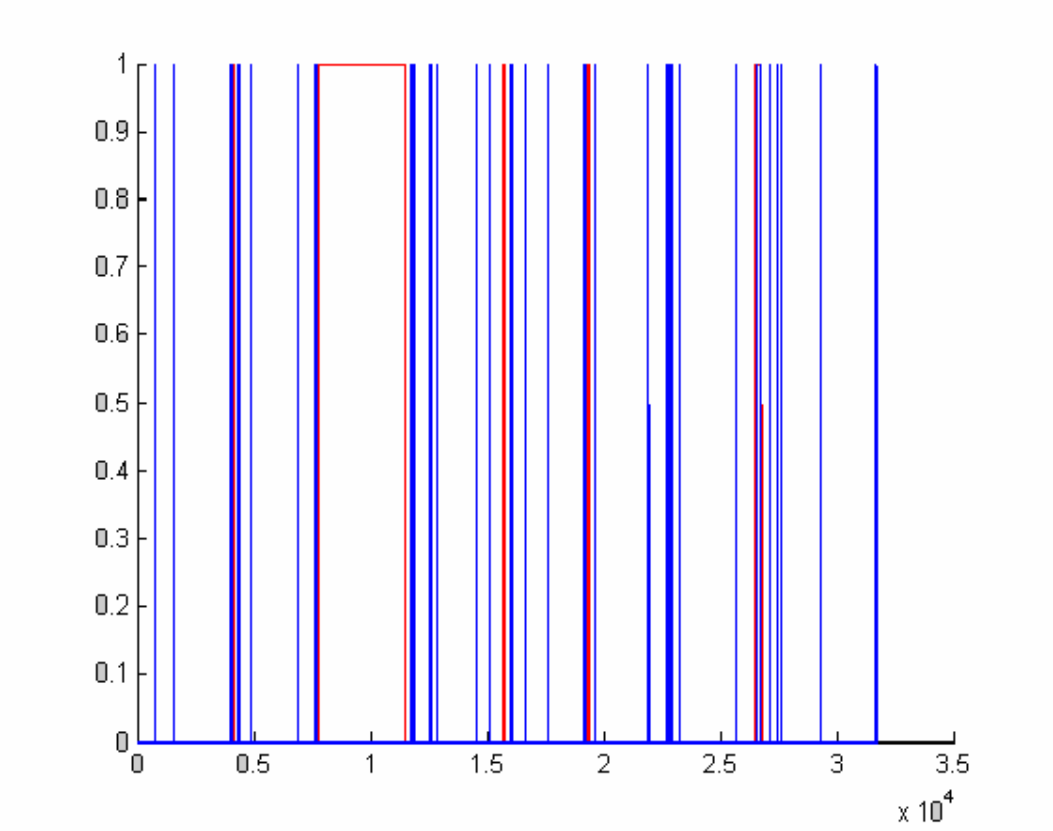

Figure 4.13. Alerts for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives, but there are 3816 false negatives. There are alerts that were given at 22 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 3816 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 22 false positives and 3816 false negatives is almost the same as not being able to detect any attacks, therefore the C_{ID} value for this threshold value is 0.7884, which is much lower when compared to Daubechies 6 results.

Table 4.12. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	**False Positives**		**False Negatives**	
#	**4139**	**22**		**3816**	
B	**FP**	**FN**	**PPV**	**NPV**	**Cid**
0.129	**0.00068**	**0.1192**	**0.9948**	**0.9826**	**0.7884**

For Haar Wavelets, for a threshold value of 1:

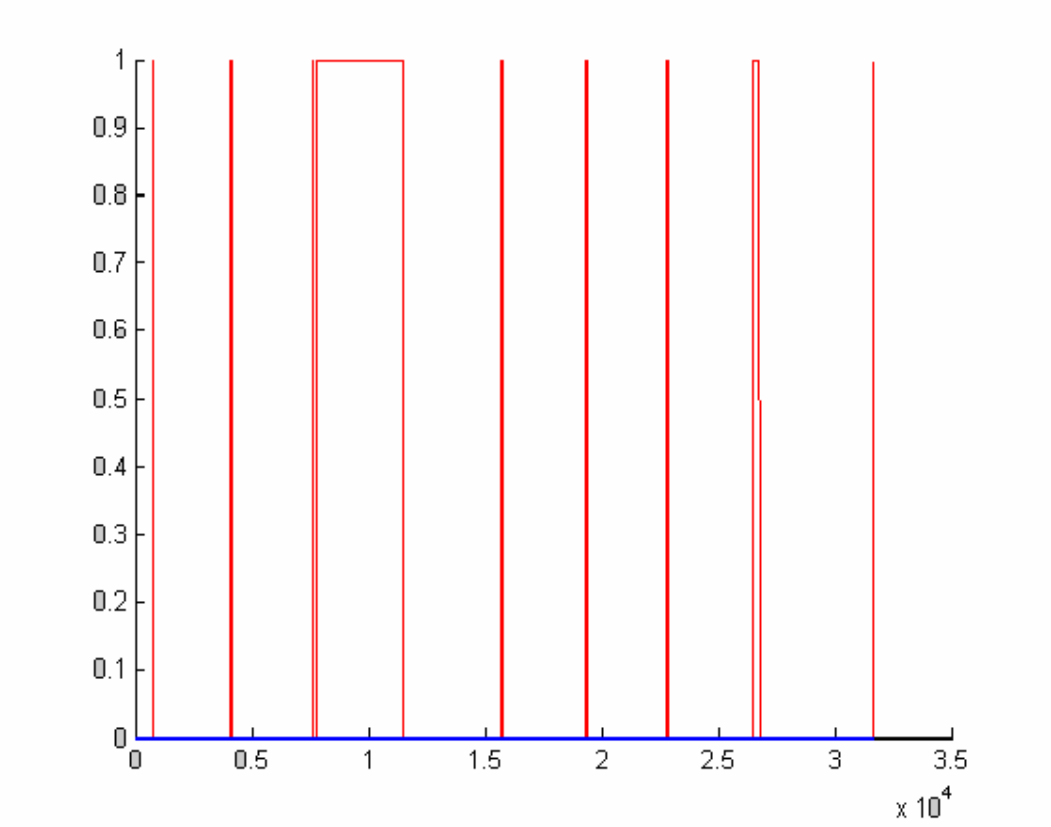

Figure 4.14. Alerts for a threshold value of 1

Putting a threshold value of 1 at the alert module results zero false positives and 4014 false negatives. This means that even if we put the lowest threshold for Haar Wavelets, the model is unable to detect any attacks, therefore the C_{ID} value for this Wavelet and threshold is 0.7897, which is a low detection value.

Table 4.13. Number of attacks instances, false positives and false negatives for a threshold of 1.

	Attack Instances	False Positives		False Negatives	
#	4139	0		4014	
B	FP	FN	PPV	NPV	Cid
0.129	0	0.1254	1	0.9818	0.7897

For Haar Wavelets, for a threshold value of 2:

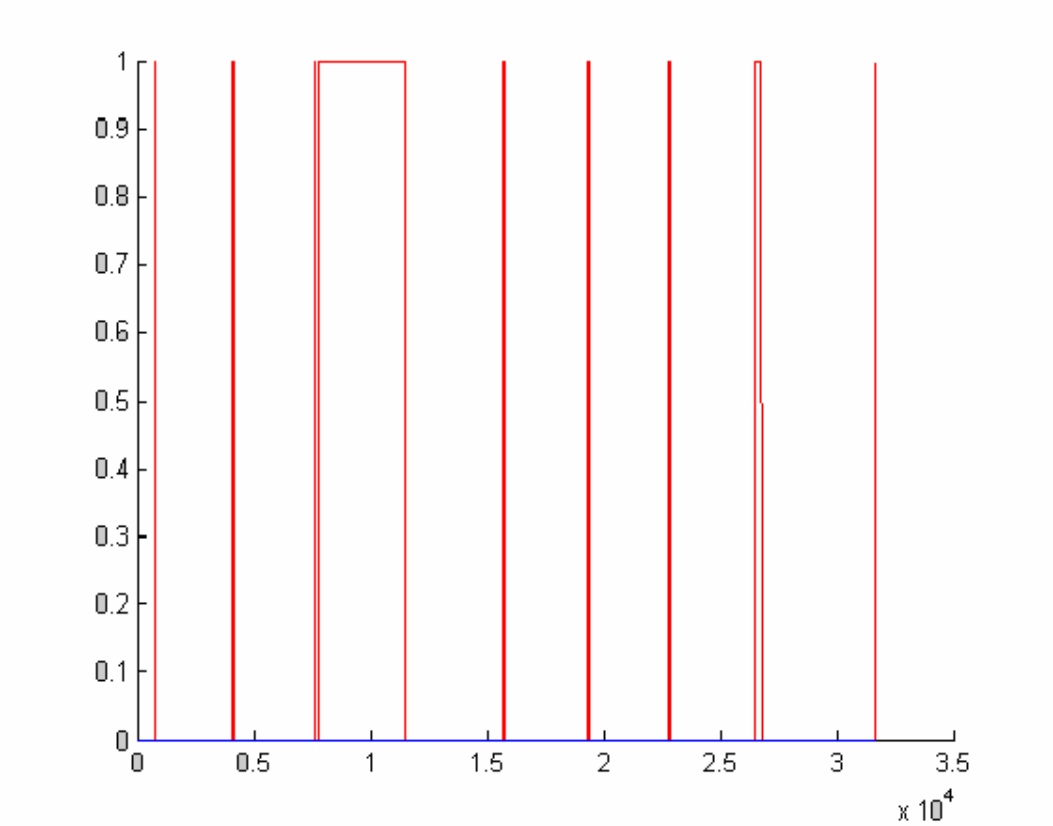

Figure 4.15. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module does not improve the results, actually it has only the potential to make it worse. The detection value does not change as it can be seen below:

Table 4.14. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	False Positives		False Negatives	
#	4139	0		4014	
B	FP	FN	PPV	NPV	Cid
0.129	0	0.1254	1	0.9818	0.7897

For Window shifts of 2500 connections, for a threshold value of 2:

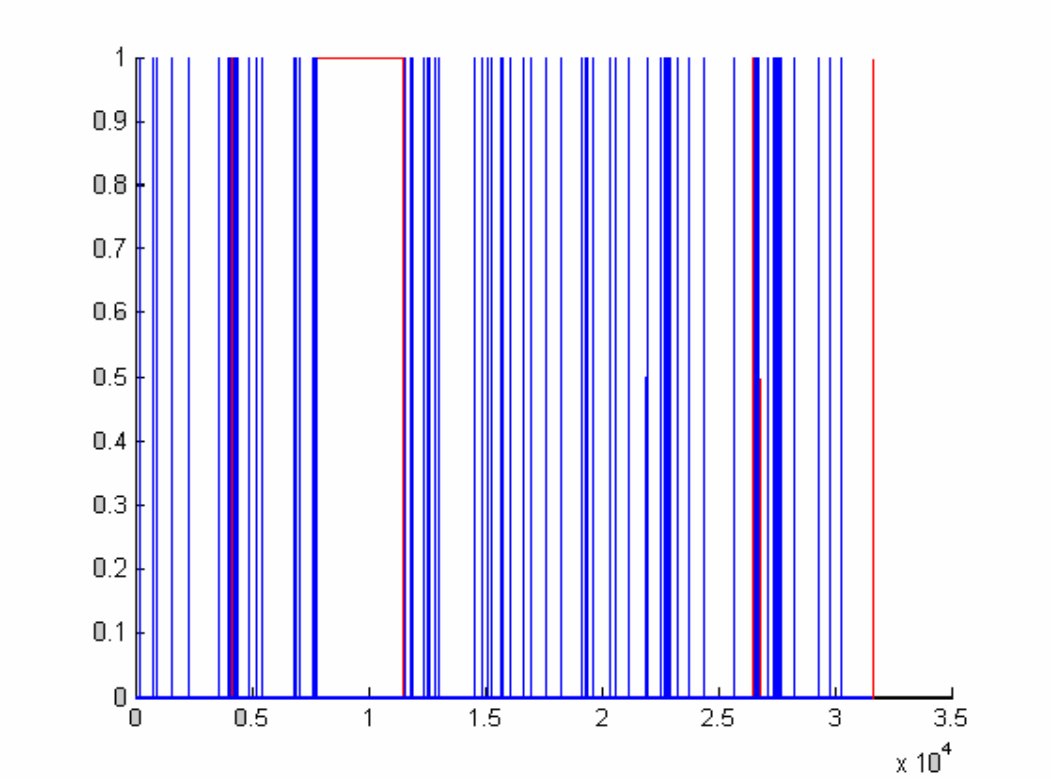

Figure 4.16. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a low amount of false positives and a low amount of false negatives. There are alerts that were given at 89 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 22 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 89 false positives and 22 false negative is not a big issue, therefore the C_{ID} value for this threshold value is 0.9662, which is a good detection value.

Table 4.15. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	False Positives		False Negatives	
#	4139	89		22	
B	FP	FN	PPV	NPV	Cid
0.129	0.0028	0.000687	0.9816	0.9999	0.9662

For Window shifts of 2500 connections, for a threshold value of 3:

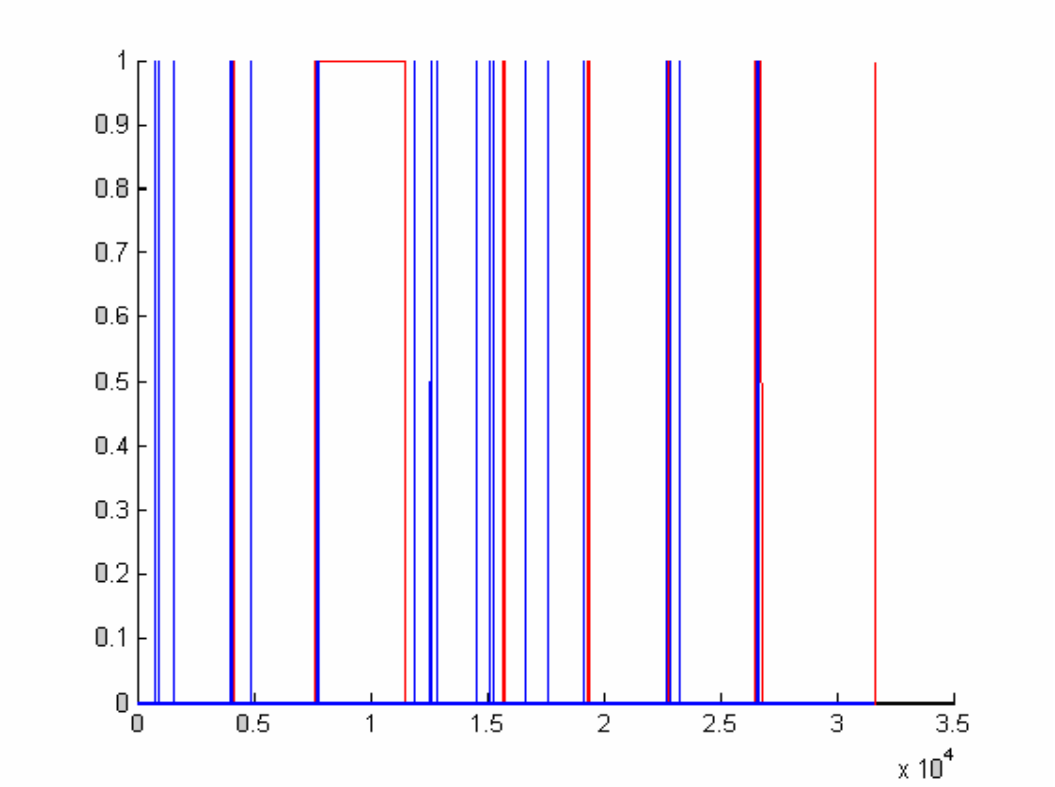

Figure 4.17. Alerts for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives and a large amount of false negatives. There are alerts that were given at 15 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 317 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 15 false positives and 317 false negatives is still not a very problematic situation, therefore the C_{ID} value for this threshold value is 0.9679, which is a good detection value.

Table 4.16. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	**False Positives**		**False Negatives**	
#	**4139**	**15**		**317**	
B	**FP**	**FN**	**PPV**	**NPV**	**Cid**
0.129	**0.00046**	**0.0099**	**0.9968**	**0.9985**	**0.9679**

For Window shifts of 3000 connections, for a threshold value of 2:

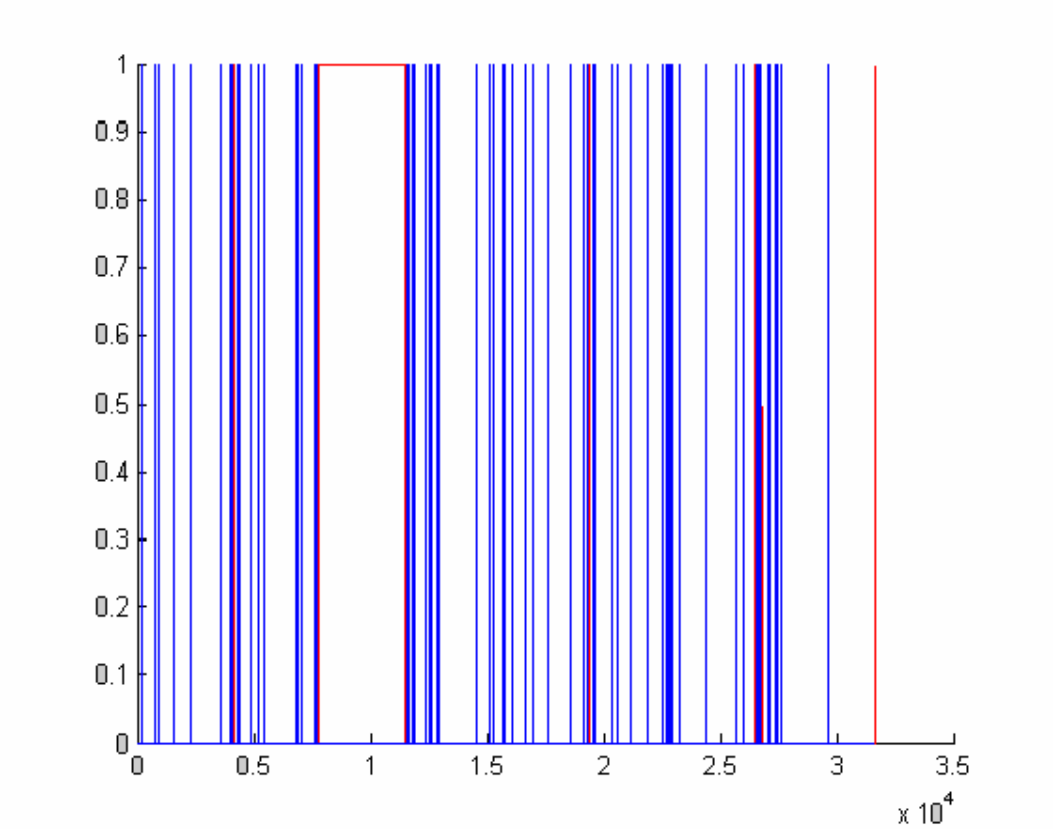

Figure 4.18. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a low amount of false positives and a low amount of false negatives. There are alerts that were given at 89 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 22 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 89 false positives and 22 false negative is not a big issue, therefore the C_{ID} value for this threshold value is 0.9662, which is a good detection value. As it can be seen there is no difference between the detection parameter values of window shifts of 2500 connections and 3000 connections, as there is an improvement in the C_{ID} value when compared to window shifts of 1250 connections.

Table 4.17. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	False Positives		False Negatives	
#	4139	89		22	
B	FP	FN	PPV	NPV	Cid
0.129	0.0028	0.000687	0.9816	0.9999	0.9662

For Window shifts of 3000 connections, for a threshold value of 3:

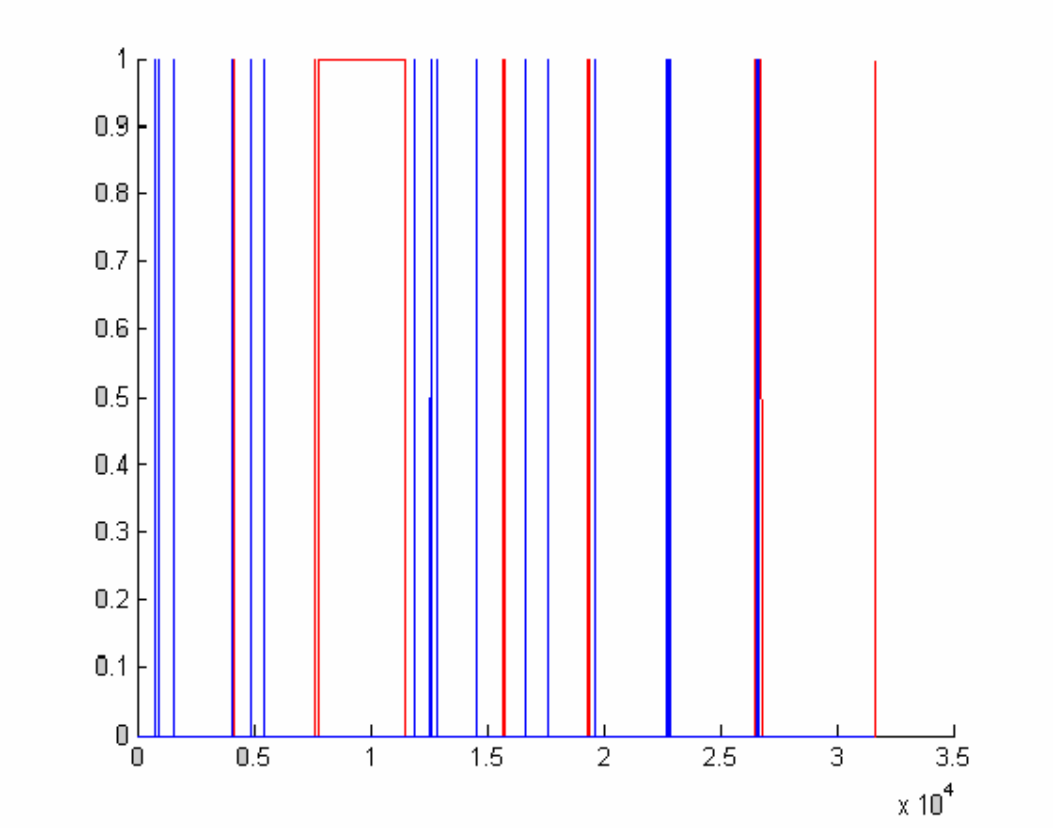

Figure 4.19. Alerts for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives, but there are 3697 false negatives. There are alerts that were given at 15 instances, indicating that there are attacks at the related instances, while there is actually none at all.

Considering that there are a total of 32000 connections in this whole data, having 15 false positives and 3697 false negatives is almost the same as not being able to detect any attacks, therefore the C_{ID} value for this threshold value is 0.7851, which is much lower when compared to Daubechies 6 results with window shifts of 1250 and 2500 connections.

Table 4.18. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	False Positives		False Negatives	
#	4139	15		3697	
B	FP	FN	PPV	NPV	Cid
0.129	0.000468	0.1237	0.9964	0.982	0.7851

For Window shifts of 500 connections, for a threshold value of 2:

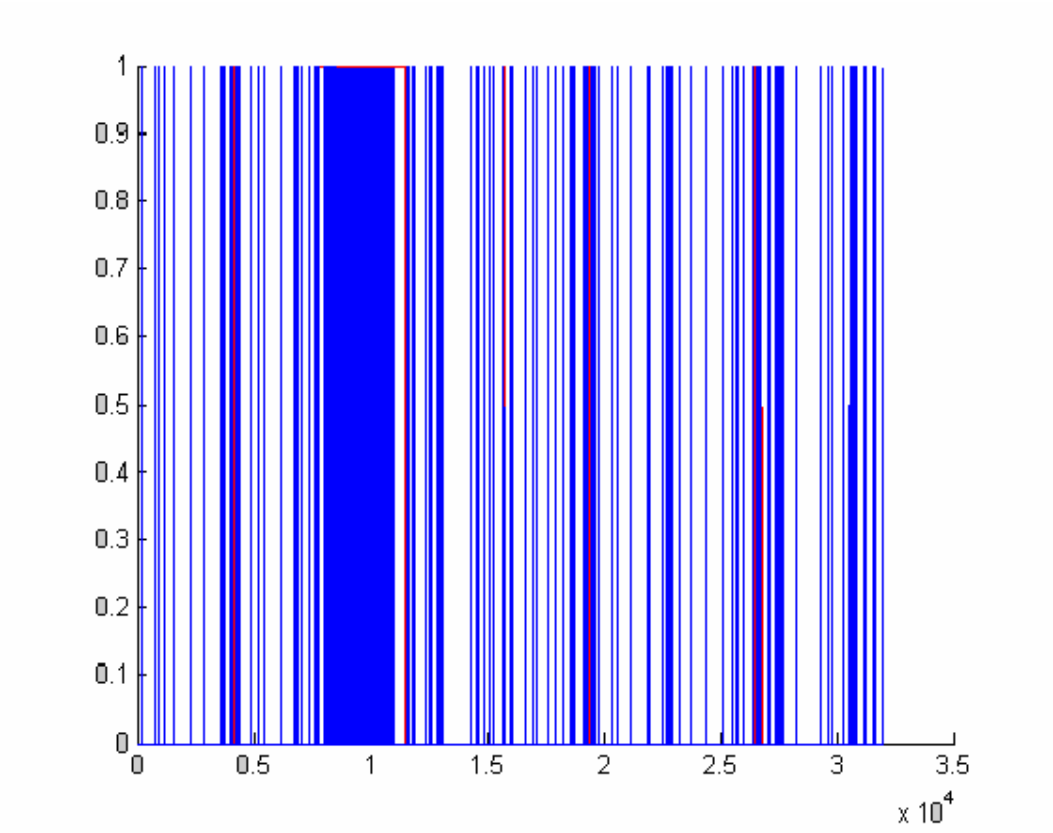

Figure 4.20. Alerts for a threshold value of 2

Putting a threshold value of 2 at the alert module results a low amount of false positives, and a low amount of false negatives. There are alerts that were given at 99 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 22 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 99 false positives and 22 false negatives is not a big issue, therefore the C_{ID} value for this threshold value is 0.9635, which is a good detection value.

Table 4.19. Number of attacks instances, false positives and false negatives for a threshold of 2.

	Attack Instances	**False Positives**		**False Negatives**	
#	**4139**	**99**		**22**	
B	**FP**	**FN**	**PPV**	**NPV**	**Cid**
0.129	**0.0031**	**0.0006875**	**0.9795**	**0.9999**	**0.9635**

For Window shifts of 500 connections, for a threshold value of 3:

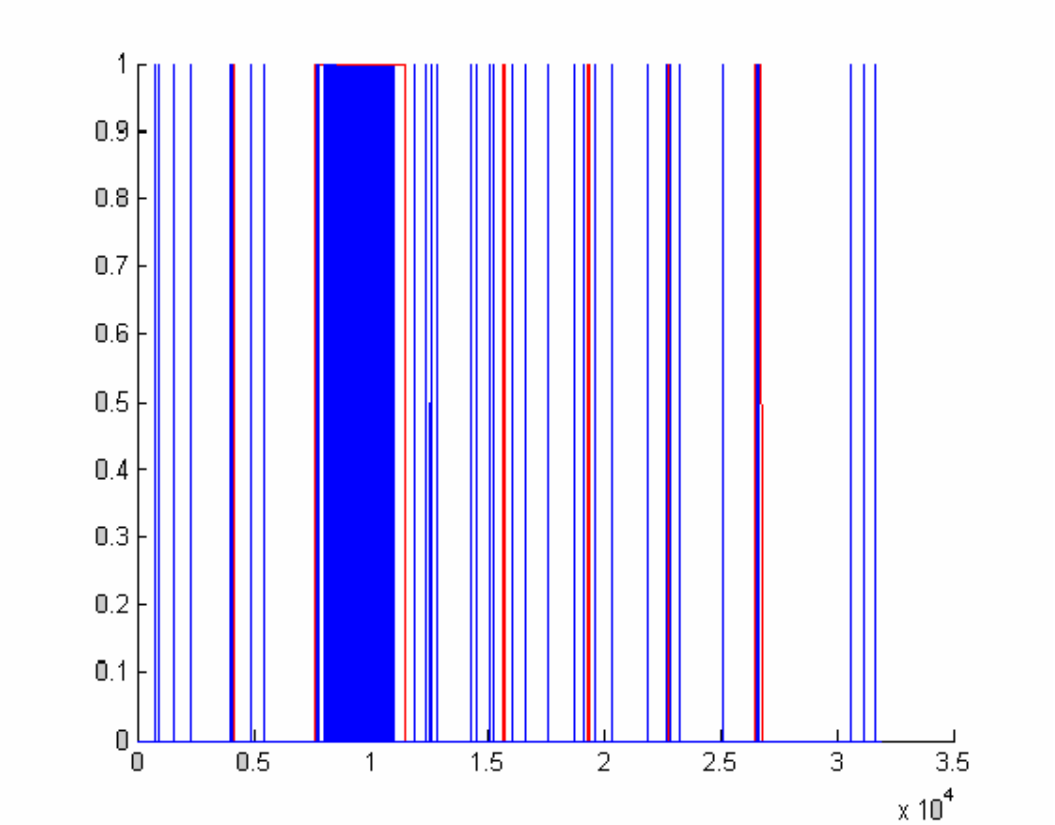

Figure 4.21. Alert table for a threshold value of 3

Putting a threshold value of 3 at the alert module results a low amount of false positives and a low amount of false negatives. There are alerts that were given at 18 instances, indicating that there are attacks at the related instances, while there is actually none at all. And there are 176 attack instances which were not detected by the model.

Considering that there are a total of 32000 connections in this whole data, having 18 false positives and 176 false negatives is not a big issue, therefore the C_{ID} value for this threshold value is 0.9767, which is a good detection value.

Table 4.20. Number of attacks instances, false positives and false negatives for a threshold of 3.

	Attack Instances	False Positives		False Negatives	
#	4139	18		176	
B	FP	FN	PPV	NPV	Cid
0.129	0.000562	0.0055	0.9962	0.9992	0.9767

5. CONCLUSION

We have proposed that we can apply signal processing methods to detect network anomalies, such as network attacks. In order to prove this thesis, we first decided to apply an AR modeling suggested by Marina Thottan for the detection of network attacks. Therefore we created a network data set in our lab, which included two network attacks. For the estimation of the AR coefficients and variance, various methods have been tested. Even though it was not fully mentioned, estimation methods like Least square estimates, yule-walker and modified yule-walker methods were tested. We have seen that yule-walker estimation gives the best results.

In order to test the dependency of the A operator matrix, a new matrix with Singular Value Decomposition (SVD) method was defined. This method increased the detection by a small bit, but it did not have a big effect in the results.

The data set, which had been used for the AR modeling was not very reliable, as it did not include many attacks. There was actually not much data flowing in the network, as there were only a couple nodes in the network. Therefore we needed a more reliable data set in order to create a better model for intrusion detection. Also in order to measure the detection capability, a more reliable data set was needed. The KDD data set was chosen as the new data set.

The KDD data set includes 41 different features, which were derived from a raw tcpdump data of nine weeks of network traffic of U.S. Air Force LAN. Some of these 41 features do not change at all, therefore they do not carry any information and thus they are not used for the detection of attacks. There are also some features that show similar characteristics with other features and therefore they are not used neither. These features are not used in order to improve the computational efficiency, as 41 features for each connection requires a lot of processing.

Applying the AR model suggested by Marina Thottan to the KDD data set does not work very well. The portion of the KDD data set that is used in this work consists of 32000

connections steps for each of the 23 features, as the data set that had been created in the lab consists of only 1600 timesteps for 3 different MIB variables. Therefore the computational power that is required to process that kind of a big data set is much higher. Even if we use only 3 features for the detection, it still requires a lot of time to process the results, which we do not desire. Therefore we suggested a new model for the detection of the network anomalies, namely the wavelet model. Even when all of the 23 features were used for the detection of the attacks, the wavelet model worked fine and did not show any computational problems.

Different wavelets such as Daubechies, Coiflet and Haar wavelets were tested for their performances. It was seen that Haar wavelets give the worst results with a C_{ID} value of 0.7887. Even if the threshold value is set to the minimum value of 1, there is a very poor detection rate. Daubechies wavelets gave very good results for a threshold value of 2 and 3. Coiflet wavelets showed a similar behaviour for a threshold of 2. It can be said that the Daubechies wavelets gives the best results, because the intrusion detection capability values for different threshold values are quite high when compared to other wavelets. For a threshold value of 3, the Daubechies6 wavelets result a Intrusion detection capability value of 0.9779, which is the highest C_{ID} value that has been achieved.

Also the intrusion detection capabilities for different window shifts have been measured when Daubechies 6 wavelets were used. The above mentioned results were derived for window shifts of 1250 connections. As the size of the window shift increases, the computation time decreases and vice versa as the size of the window shift decreases, the computation time increases. For window shifts of 500 connections, for threshold values of 2 and 3, the C_{ID} value varies between 0.96 and 0.97. For window shifts of 2500 connections, for threshold values of 2 and 3, the C_{ID} value varies around 0.96. For window shifts of 3000 connections, for a threshold value of 2, the C_{ID} value is around 0.96, which is a satisfactory value. But for a threshold value of 3, the C_{ID} value decreases to 0.78. In order to improve the efficiency of the model smaller window shifts can be used, but as there is a trade-off between the computation time and detection capability, the window shift size should be chosen as a feasible size. The window shift size of 1250 connections gives the best results in terms of detection capability and computation time.

There are 3 different machine learning algorithms in [33], which are used as a detection tree. The related C_{ID} values for these algorithms, after some fine tuning operations are like below:

Table 5.1. Cid Values for different Algorithms.

IDS	C_{ID}
Feature set 1 with C4.5	0.4258
Feature set 1 with Naive Bayes	0.3756
Feature set 1 with SVM	0.5642

When compared to the detection capability results in this thesis, these C_{ID} values above are very low. In this thesis, for Daubechies 6 wavelets when a threshold value of 2 or 3 was used, almost all of the attacks in the data set could be detected and therefore the C_{ID} value for these thresholds are quite big.

The performance results that were achieved for the wavelet model are quite promising, but it still requires some improvements. In this thesis, all Features were treated equally, so i.e. for the detection of an anomaly all features have the same weight. Also each wavelet coefficient for different features has the same weights, too. But actually each feature and each different wavelet coefficient should have a different impact at the detection of an anomaly, so an adaptive way of weighting the features and the wavelet coefficients could be designed.

All of the KDD features were used at the detection of an anomaly, but actually some features have a bigger impact at the detection of DoS type anomalies, some features have a bigger impact at the detection of unauthorized access anomalies, some features have a bigger impact at the detection of surveillance anomalies and etc.. Therefore by partitioning the features for different anomaly types, we can improve the computation efficiency and this results the need of an adaptive method for the decision of features for different anomaly types.

In this thesis, alerts for anomalies are given in the range of 15 connections prior to the beginning of the attack instance and 15 connections after the beginning of the attack.

So there is a range of 30 connections where an attack can be indicated via an alert. This range can be reduced with the above mentioned methods and therefore the IDS can become more precise.

While the results that are found are quite encouraging, as mentioned above there is still future work to be done in order to improve the detection capability and in order to make the IDS more precise. It is seen that the Wavelet-model achieves the design objectives of an Intrusion Detection System and therefore can be used as a reliable and robust, near real-time network based Intrusion Detection.

APPENDIX A: PERFORMANCE TABLES

The Appendix A has been given as a separate CD which includes;

- Performance Table.xls (Microsoft Office Excel is needed)
    - This Excel file includes 18 different sheets. Each sheet include a performance table. The name of the sheets are as the following:
        - Daubechies6 Wavelet-Threshold=1
        - Daubechies6 Wavelet-Threshold=2
        - Daubechies6 Wavelet-Threshold=3
        - Daubechies6 Wavelet-Threshold=4
        - Daubechies6 Wavelet-Threshold=10
        - Daubechies2 Wavelet-Threshold=1
        - Daubechies2 Wavelet-Threshold=2
        - Daubechies2 Wavelet-Threshold=3
        - Coiflet2 Wavelet-Threshold=2
        - Coiflet2 Wavelet-Threshold=3
        - Haar Wavelet-Threshold=1
        - Haar Wavelet-Threshold=2
        - WindowShift2500-Threshold=2
        - WindowShift2500-Threshold=3
        - WindowShift3000-Threshold=2
        - WindowShift3000-Threshold=3
        - WindowShift500-Threshold=2
        - WindowShift500-Threshold=3

- Jpeg files of each figure are included in the CD. The file names are as the following:f
    - Daubechies6-Threshold1
    - Daubechies6-Threshold2
    - Daubechies6-Threshold3
    - Daubechies6-Threshold4

- Daubechies6-Threshold10
- Daubechies2-Threshold1
- Daubechies2-Threshold2
- Daubechies2-Threshold3
- Coiflet2-Threshold2
- Coiflet2-Threshold3
- Haar-Threshold1
- Haar-Threshold2
- WindowShift-2500-Threshold2
- WindowShift-2500-Threshold3
- WindowShift-3000-Threshold2
- WindowShift-3000-Threshold3
- WindowShift-500-Threshold2
- WindowShift-500-Threshold3

APPENDIX B : MATLAB CODES

In this part the simulation codes of AR model and Wavelet model are explained. The implementation is evaluated in MATLAB 6P5 environment. MATLAB is chosen for its capability of providing efficient computation easiness on vectors and matrices. Below the functions and their subfunctions can be found:

Main codes:

- e_ar.m: AR LS estimation main code. It uses 2 subfunctions and requires snmp_data file as input.
- e_yule.m: AR Yule-Walker estimation main code. It uses 2 subfunctions and requires snmp_data file as input.
- e_yulesvd.m: AR Yule-Walker estimation with operator matrix A being created via SVD main code. It uses 2 subfunctions and requires snmp_data file as input.
- wavekdd-db6.m: Wavelet model with Daubechies 6 Wavelets. Uses 1 subfunction.
- wavekdd-db2.m: Wavelet model with Daubechies 2 Wavelets. Uses 1 subfunction.
- wavekdd-coif2.m: Wavelet model with Coiflet 2 Wavelets. Uses 1 subfunction.
- wavekdd-haar.m: Wavelet model with Haar Wavelets. Uses 1 subfunction.
- wavekdd-shift.m: Wavelet model for the usage of window shift testing. Uses 1 subfunction.

Sub-Functions:

1. modulusmaxima.m: Modulus Maxima calculation
2. lnar.m: A code call for Least-Squares
3. lsar.m Least Squares by R. Moses
4. yule_ar.m : A code call for Yule Walker
5. yulewalker.m: Yule Walker by R. Moses

Necessary Matlab functions are listed below:

- wavedec.m
- wrcoef.m
- svd.m

The Data set that were used are:

  - snmp_data.mat: Includes the MIB variables for the first data set.
  - kdd23.mat: Includes the features of KDD data set.

The Appendix B has been given as a CD includes;

- Appendix B.doc (you need Microsoft Office Word or Wordpad)
  Includes matlab codes definition and how to run them.

- Matlab Codes and Data Sets in each relevant folder (You need Matlab R14, Wavelet Toolbox or wavedec.m and wrcoef.m Matlab functions). Includes all Matlab Functions and data used for analyses.

  All the included files and data were explained in Appendix_B_.doc

REFERENCES

1. M. Alhamaty, A. Yazdian, F. Al-qadasi "Intrusion Detection System Based On The Integrity of TCP Packet", *2006 World Enformatika Society – Transactions on Engineering, Computing and Technology* V11 February 2006.

2. K. Wang, S. J. Stolfo, "Anomalous Payload-based Network Intrusion Detection" *Computer Science Department, Columbia University,* 2005.

3. Chuanyi Ji Marina Thottan. "Anomaly detection in ip Networks", *IEEE Transactions on Signal Processing*, 51(8):2191-2204, 2003.

4. Joseph Lo Ph.D "Denial of Service or "Nuke" Attacks",. **http://www.irchelp.org/irchelp/security/** ,Mar 12, 2005

5. Intrusion Detection System in "Wikipedia" **http://en.wikipedia.org/wiki/Intrusion-detection_system**

6. MIT Lincoln Laboratory – DARPA Intrusion Detection Evaluation Data Sets **http://www.ll.mit.edu/IST/ideval/data/data_index.html**

7. KDD-CUP-99 Tast description **http://kdd.ics.uci.edu/databases/kddcup99/task.html**

8. P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," *Proceedings of ACM SIGCOMM InternetMeasurement Workshop*, Marseille, France, 2002.

9. P. Huang, A. Feldmann, and W. Willinger, "A non-intrusive, waveletbased approach to detecting network performance problems," *Proc. Of Internet Measurement Workshop*, Nov. 2001.

10. M. Kim, S. Lam, T. Kim, Y. Shin, and E. J. Powers, "Wavelet-based Approach to Detect Shared Congestion," *Proceedings of ACM SIGCOMM 2004*, Portland, Oregon, August 2004.

11. Vladimir Gudkov and Joseph E. Johnson, "Multidimensional Network Monitoring for Intrusion Detection", **http://arxiv.org/PS_cache/cs/pdf/0206/0206020.pdf**

12. A. C. Gilbert, "Multiscale Analysis and Data Networks", Applied and Computational Harmonic Analysis 10, 185–202 (2001)

13. Y. Zhang, Z. Ge, M. Roughan, and A. Greenberg. "Network anomography", In Proceedings of the Internet Measurement Conference (IMC '05), Berkeley, CA, USA, October 2005.

14. Chin-Tser Huang, Sachin Thareja, Yong-June Shin, "Wavelet-based Real Time Detection of Network Traffic Anomalies" **http://www.cse.sc.edu/~huangct/wens06.pdf**

15. Sanjay Rawat, and Challa S. Sastry, "Network Intrusion Detection Using Wavelet Analysis", **https://www.springerlink.com/content/1ac9arcw4y1bkh3k/resource-secured/?target=fulltext.pdf**

16. Seong Soo Kim, A. L. Narasimha Reddy, and Marina Vannucci, "Detecting Traffic Anomalies through aggregate analysis of packet header data", **http://www.ee.tamu.edu/~reddy/papers/skim_net04.pdf**

17. Seong Soo Kim, A. L. Narasimha Reddy, and Marina Vannucci, "Detecting Traffic Anomalies using Discrete Wavelet Transform", **http://www.ece.tamu.edu/~reddy/papers/icoin_skim03.pdf**

18. Seong Soo Kim and A. L. Narasimha Reddy, "Image-based Anomaly Detection Technique: Algorithm, Implementation and Effectiveness", IEEE Journal on Selected Areas in Communications - HIGH-SPEED NETWORK SECURITY, Oct. 2006Volume: 24,

19. Haining Wang, Danlu Zhang, Member, and Kang G. Shin, "Change-Point Monitoring for the Detection of DoS Attacks", IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 1, NO. 4, OCTOBER-DECEMBER 2004

20. Khaled Labib and V. Rao Vemuri, "*Detecting And Visualizing Denial-of-Service And Network Probe Attacks Using PrincipalComponent Analysis*", **http://citeseer.ist.psu.edu/cache/papers/cs2/20/http:zSzzSzailab.das.ucdavis.eduzSz~klabibzSzdocszSzSAR04_Labib_Vemuri_CR_Final.pdf/labib04detecting.pdf**

21. Damiano Bolzoni, Sandro Etalle, Pieter Hartel, "POSEIDON: a 2-tier Anomaly-based Network Intrusion Detection System", **http://arxiv.org/PS_cache/cs/pdf/0511/0511043.pdf**

22. Damiano Bolzoni, Sandro Etalle, "APHRODITE: an Anomaly-based Architecture for False Positive Reduction", **http://arxiv.org/PS_cache/cs/pdf/0604/0604026.pdf**

23. Qingtao Wu; Zhiqing Shao, " Network Anomaly Detection Using Time Series Analysis ", *in* Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services (ICAS/ICNS 2005) 0-7695-2450-8/05 $20.00 © 2005 IEEE

24. SNMP in Wikipedia, **http://en.wikipedia.org/wiki/SNMP**

25. M. Thottan and C. Ji, "Fault Prediction at the Network Layer Using Intelligent Agents" in **www.bell-labs.com/user/marinat/pubs/im99.pdf**

26. M. Thottan and C. Ji, "Adaptive thresholding for proactive network problem detection", *in Proc. IEEE Int. Workshop Syst. Manage., Newport, RI, Apr. 1998*.

27. PETER . De SOUZA, "Statistical Test and Distance Measures for LPC Coefficients" on *IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. Assp-25 No:6, December 1977*

28. Stoica, P., and R. Moses. *Introduction to Spectral Analysis*. Upper Saddle River, NJ: Prentice Hall, 1997 pp. 89-90, 103-106,

29. Stephane Mallat and Wen Liang Hwang. "Singularity Detection and Processing with Wavelets", IEEE Transactions on Information Theory, Vol.38, No.2, March 1992.

30. Donald B. Percival ," Wavelet Methods for Time Series Analysis ", Cambrige University Press 2000. pp. 1, 2, 5-9, 13, 17,

31. "Wavelet Gallerie", **http://nt.eit.uni-kl.de/wavelet/gallery2.html**

32. Guofei gu, Prahlad Fogla, David Dagon, Wenke Lee, *An Information-Theoretic Measure of Intrusion Detection Capability*, **http://smartech.gatech.edu/dspace/bitstream/1853/9432/1/GIT-CC-05-10.pdf**