

Intrusion Detection and Management over the World Wide Web

Hai Ying Luan

A Dissertation Submitted to the School of Computing

Faculty of Engineering and Computing

Dublin City University

For The Degree of Master of Science

Supervisor: Dr Darragh O'Brien

January 2010

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Master of Science is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____

ID No.: _____

Date : _____

Acknowledgements

Firstly, I would like to thank my supervisor, Dr Darragh O'Brien, for many interesting discussions, patient correction, constructive criticism and encouragement.

I would like to give special thanks to my colleague, Darren Fitzpatrick, for his help and readiness to share his knowledge throughout the course of the project.

Sincere thanks are due to Declan McMullen for all his support and invaluable assistance with the proof reading of this thesis.

I wish to thank Enterprise Ireland for funding the development of the software described in this thesis under its "Proof-of-Concept" scheme.

Last but not least, I would like to thank my parents for their support and constant encouragement to keep going.

Abstract

As the Internet and society become ever more integrated so the number of Internet users continues to grow. Today there are 1.6 billion Internet users. They use its services to work from home, shop for gifts, socialise with friends, research the family holiday and manage their finances. Through generating both wealth and employment the Internet and our economies have also become interwoven. The growth of the Internet has attracted hackers and organised criminals. Users are targeted for financial gain through malware and social engineering attacks. Industry has responded to the growing threat by developing a range defences: antivirus software, firewalls and intrusion detection systems are all readily available. Yet the Internet security problem continues to grow and Internet crime continues to thrive. Warnings on the latest application vulnerabilities, phishing scams and malware epidemics are announced regularly and serve to heighten user anxiety. Not only are users targeted for attack but so too are businesses, corporations, public utilities and even states. Implementing network security remains an error prone task for the modern Internet user. In response this thesis explores whether intrusion detection and management can be effectively offered as a web service to users in order to better protect them and heighten their awareness of the Internet security threat.

List of Figures and Tables

Table 2.1	The TCP/IP Model Layers.....	14
Figure 2.1	TCP Three-way Handshake.....	19
Figure 2.2	Buffer Overflow	23
Figure 3.1	Firewalls Locations.....	32
Table 3.1	Iptables Rules	35
Table 3.2	Connection State Table.....	36
Figure 3.2	Threshold.....	45
Figure 3.3	NIDS Locations	48
Figure 3.4	AAFID Architecture	50
Figure 3.5	Components' Hierarchy in AAFID.....	51
Figure 4.1	System Overview.....	58
Figure 4.2	Proposed IDS over the Network.....	59
Figure 4.3	Packets Decision Tree.....	61
Figure 4.4	An Example of an Event in IDMEF.....	65
Figure 4.5	Heartbeat XML Messages.....	66
Figure 4.6	System Architecture.....	67
Figure 4.7	Server Functions.....	68
Figure 4.8	Main Page – Administrator View.....	70
Figure 4.9	Main Page – User View.....	70
Figure 4.10	Examine Events.....	71

Figure 4.11	A Server Rule Based on the Slammer Worm.....	72
Figure 4.12	WatchList Configurations.....	73
Figure 4.13	WatchList.....	73
Figure 4.14	Add Client.....	74
Figure 4.15	Create Client Configuration.....	75
Figure 4.16	Create Client Ruleset.....	76
Figure 4.17	Statistic.....	77
Figure 4.18	Database Tables of the System.....	79
Figure 4.19	Web 2.0.....	80
Figure 4.20	AJAX Application Working Principles.....	81
Figure 4.21	Implementation Techniques.....	82
Figure 4.22	Events with Main Components.....	83
Figure 4.23	Processing Flow of the System.....	83
Table 4.1	Requirements Met.....	85
Figure 5.1	Performance Evaluation with Page Speed.....	87, 88
Figure 5.2	Detection of Backdoor and Ping of Death.....	89
Figure 5.3	Scans Detection.....	90
Figure 5.4	NSE Scripts Detection.....	91
Table 5.1	Detected Threats.....	92
Figure 5.5	Suspicious Internal IP addresses.....	94
Figure 5.6	School of Computing Network Configuration.....	95
Figure 5.7	Foreign IP addresses and the Slammer worm	96

Table of Contents

Declaration	I
Acknowledgements	II
Abstract	III
List of Figures and Tables.....	IV
Table of Contents	VI
1 Introduction.....	1
1.1 The Internet Security Problem: Contributing Factors	2
1.2 Research Question	5
1.3 Thesis Structure	7
2 The Internet Security Threat	8
2.1 A Brief History of Networks, the Internet and WWW.....	8
2.1.1 From ARPANET to the Internet	9
2.1.2 The Internet and WWW Explosion.....	10
2.1.2.1 The Internet in the Home	11
2.1.2.2 The Internet in Business	12
2.2 Network Communication Layers and Protocols.....	12
2.2.1 Network Communication Layers	13
2.2.2 Network Protocols.....	15
2.2.3 Example.....	17
2.3 Internet Threats and Attacks.....	17

2.3.1	Attacking Internet Services and Resources.....	18
2.3.2	Attacking Applications.....	20
2.3.3	Social Engineering Attacks.....	25
2.3.4	Attacking Misconfigurations.....	26
2.4	Malware.....	28
2.5	Summary	30
3	The Defences	31
3.1	Firewalls	32
3.1.1	Network-based Firewall.....	33
3.1.2	Host-based Firewall	34
3.1.3	Packet Filtering	34
3.1.4	Proxy Server.....	36
3.1.5	Summary	37
3.2	Antivirus Software	37
3.2.1	Signature-based detectors.....	39
3.2.2	Behaviour-based detectors	39
3.2.3	Summary	39
3.3	Intrusion Detection	40
3.3.1	A Brief History.....	41
3.3.2	Primary Components.....	42
3.3.3	Signature Detection vs. Anomaly Detection.....	44
3.3.4	Host-based IDS vs. Network-based IDS.....	46

3.3.5	Hybrid IDS	49
3.3.6	IDS Today	52
3.3.7	Summary	53
3.4	Summary	54
4	Intrusion Detection and Management over the WWW	55
4.1	Meeting the Typical User's IDS Needs.....	55
4.2	Design and Implementation of a Hybrid WWW Intrusion Detection System ..	58
4.2.1	System Overview	58
4.2.2	Gathering Data: Agent Design and Implementation.....	60
4.2.2.1	Design	60
4.2.2.2	Implementation	64
4.2.3	Managing Clients and Events: Server Design and Implementation.....	67
4.2.3.1	Design	68
4.2.3.2	Implementation	78
4.3	System Use Cases.....	83
4.4	Summary	85
5	Results.....	86
5.1	Performance Testing.....	86
5.2	System Validation	89
5.3	Live Testing.....	93
5.4	Summary	97
6	Conclusions and Future Work	99

6.1	Chapter Summary	99
6.2	Conclusions	100
6.3	Future Work	100
	References	104

1 Introduction

As the Internet and society become ever more integrated so the number of Internet users continues to grow. Today there are 1.6 billion Internet users [1]. They use its services to work from home, shop for gifts, socialise with friends, research the family holiday and manage their finances. No longer a medium for the passive delivery of information the Internet and the World Wide Web (WWW) it supports have evolved into full application development frameworks and thanks to Web 2.0 technologies the distinction between traditional desktop PC applications and the latest Internet applications is blurred. Such applications along with an ever-expanding list of web browser plug-ins continue to enrich the WWW experience, bringing it to life and attracting new interest. Already embedded in our culture, the Internet and our economies have also become interwoven. The Internet generates both wealth and employment. In 2008 almost two thirds of all enterprises in the EU had a website [2]. In the same year e-commerce sales in the EU were worth 106 billion euros and this figure is projected to reach 323 billion euros by 2011 [3]. Given its growing cultural and economic importance there is every reason to believe the Internet explosion is set to continue.

The surge in Internet use has not gone unnoticed by hackers and organised criminals. Internet crime has expanded in parallel with the growing number of Internet users. Users are at risk, targeted for financial gain through malware and social engineering attacks. Industry has responded to the growing threat by developing a range of products for the home and business user: antivirus software, firewalls and intrusion detection systems are all available in commercial and free, open source form. Yet despite the widespread availability of such countermeasures the Internet security problem continues to grow and Internet crime continues to thrive. Between 2001 and 2006 Kaspersky Labs reported an 800% increase in the number of new malicious programs [4]. Online crime costs an estimated one trillion USD annually according to the World Economic Forum [5]. Warnings on the latest application vulnerabilities, phishing scams and malware epidemics are announced regularly and serve to heighten user anxiety. Not only are users at risk but so too are businesses, corporations, public utilities and even states [6].

The remainder of this chapter is structured as follows. In Section 1.1 we consider the factors that make the Internet security problem difficult to solve and we ask why Internet crime and hacker activity are expanding despite the widespread availability of

defences. In light of the evident difficulty faced by today's users in implementing and maintaining network and Internet security we present the research question this thesis seeks to address in Section 1.2. The structure of the remainder of the thesis is given in Section 1.3.

1.1 The Internet Security Problem: Contributing Factors

Despite the availability of security-enforcing/monitoring software and the security industry's understanding of the issues involved, implementing network security remains an error prone task for the modern Internet user. Below we consider some reasons why this is the case and why failing to adequately secure their machines is exposing users to a growing range of threats. These questions are explored further in Chapter 2 but briefly described here in order to set the context for our research question.

Evolving Internet User Profiles (Who is using the Internet?)

Internet use has exploded over the last 15 years. New media-rich Internet applications and technologies are attracting not only an increasing number of users but also new kinds of user. Social networking sites such as Bebo [7], MySpace [8] and Facebook [9] are marketed towards younger audiences. The average age of a Facebook user is 21 years and most users are 18-24 years of age. According to research conducted in 2008 by Ofcom, the UK's broadcasting and telecommunications regulator, 49% of children aged 8-17 years who use the Internet have a profile on a social networking site [10]. Not only are new younger users targeted by such websites but so too are older ones: Saga Zone [11] is a social network specifically aimed at the over 50s market.

This new brand of user lacks technical expertise and/or has only a limited awareness of the Internet security threat. Such users may not be equipped to be solely responsible for the secure configuration and maintenance of an Internet-connected PC. For them, a minimum requirement is that defences come pre-installed and pre-configured. However, even pre-installed defences such as antivirus software and firewalls may present challenges. While antivirus software requires minimal maintenance, securely configuring a firewall is not so straightforward. The default firewall setting may allow all incoming and outgoing connections in order to avoid early troubleshooting. Ensuring a secure Internet connection is thus not merely a question of running a firewall but running a *securely configured* firewall. In addition, some interaction between the users

and firewall is often required in order to enable/disable certain services and to differentiate between private and public networks etc. Confused and/or busy users may resort to a default "Allow Full Internet Access" policy when confronted with such questions. Many of today's users thus cannot be consistently relied upon to secure such software and as a result many users' machines will often be vulnerable to attack due to improper configuration and maintenance. Worse yet, defences may not even be installed. In 2005 Australian researchers found that only one in seven computers in Australia used a firewall and only about one in three used up-to-date antivirus software [12]. More complex configuration, such as that required to install and run an intrusion detection system such as Snort [13] is beyond the scope of average users. Although potentially of significant benefit to users in both raising their security awareness and in determining whether their machine has been successfully exploited, installing such software is too complex a task for the average user to consider.

There is clear evidence that in addition to failing to run defences many of today's users do not regularly update or apply patches to the software they use to interact with a hostile Internet: the browser. In 2008 there were 650 million Internet users browsing the web with known, exploitable vulnerabilities in their browsers [14]. Consequently, the browser serves as a popular attack vector. Even when the browser itself cannot be exploited, ill-equipped users may be incentivised into installing trojan software through social engineering. It is estimated that 59 million users in the US have spyware or other types of malware on their computers [15].

In summary, many of today's Internet users are unaware they are targeted and/or do not have the technical experience to ensure their safety in an increasingly hostile environment.

Proliferation of Wireless Networks and Devices (How are users connecting?)

Not only is it that more and more non-security-conscious users are connecting to the Internet but a further contributing factor to weak network security is from where these users are connecting: wireless devices and networks have surged in popularity over recent years. Wireless-network-ready devices are now the norm as users seek to surf the web without the hindrance of network cables. Devices such as PDAs (Portable Digital Assistants), smart phones, laptops and games consoles (e.g. Sony's Play Station Portable) are all wireless-network-ready. One factor driving the expansion of wireless

networks is convenience: because they offer a flexible and affordable approach to rapid network deployment, they are an attractive option for businesses such as cafés, airports, hotels etc. seeking to cash in on an ever growing set of Internet-hungry customers. However, the anonymity afforded to wireless network users compared with users of traditional wired networks makes the former attractive hunting grounds for eavesdroppers and intruders [16, 17]. Furthermore, their very convenience means such networks are routinely administered by non-security professionals and often run under insecure configurations. In effect we are witnessing the emergence of a new brand of Internet administrator as well as user. Users of public wireless networks may mistakenly assume that defences typically deployed at the network perimeter (e.g. a firewall) are in place when in fact they are absent, leaving client machines open to attack. Users of such networks are generally unaware of the implicit trust they place in its administrator and in their fellow users.

WEP (Wired Equivalent Privacy/Wireless Encryption Protocol) is often the default encryption algorithm on wireless routers. It can be broken in a matter of minutes using freely available utilities such as Aircrack [18][19]. Many of the new Internet generation's users and administrators are unaware of this fact. Once a WEP key is obtained sniffing network traffic including confidential personal details suitable for identity and financial theft is straightforward in the absence of further encryption [20]. Similarly straightforward is bandwidth hijacking for the download of illegal content. Many networks are even deployed with no encryption at all. Implementing a secure wireless network is more involved than simply adding a wireless router to a broadband connection. Securely configuring the wireless router should include hiding the network's SSID (Service Set Identifier), applying MAC (Media Access Control) and IP (Internet Protocol) address filters, creating a secure access password for the network and choosing an appropriate encryption standard. Many users are failing to take these steps. We look further at wireless security issues in Chapter 2.

Rising Hacker and Malware Sophistication (Aren't attacks difficult to mount?)

Hackers and the malware they write are becoming more sophisticated. It is also becoming easier to create malware as more and more free tools and malware development kits appear online. Software originally developed to aid security specialists in verifying network security is used by intruders to detect and attack vulnerabilities.

Classic hacker behaviour follows a three step process: Network reconnaissance, host/service enumeration and finally exploitation. For each of these steps the hacker has a selection of tools to choose from. The target is selected during the network reconnaissance stage. Here tools such as *nslookup* and *dig* [21] help the intruder gain a high level overview of the target by querying publicly available information. During the second stage accessible hosts are detected and scanned for services using a network scanner. *Nmap* [22] allows the identification of the target operating system and the versions of the various network services it exports. Once a vulnerability has been identified the final step for the hacker is its exploitation. Here a library of exploits such as that provided by the Metasploit project [23] does the complicated work. The attacker passes to Metasploit a vulnerability descriptor and specifies the level of control she wishes to gain over the host (e.g. a remote root shell). Metasploit generates the payload and will even inject it into the target host. Once control over a host is achieved the attacker may install malware in order to retain complete control should the original vulnerability be patched. Here too the attacker has a range of freely available malware and rootkits to choose from across the Internet [24]. The victim may be merged into a botnet controlled using a Distributed Denial of Service (DDoS) toolkit such as Trinoo, TFN2K or MStream [25]. Once the intruder has control of one host and gained a foothold in the target network, transitive trust relations may be exploited to take over other hosts.

Users and businesses are suffering as they fail to adequately defend themselves from malware. One survey [26] puts the annual cost of cleaning up malware in the US at 67 billion USD. The impact on users is more difficult to estimate but US consumers spent an estimated 4 billion USD over a two year period repairing damage due to malware [15].

1.2 Research Question

In summary, a growing number of users of insufficient security awareness are accessing an increasingly hostile Internet from an expanding number of potentially insecure networks [27]. Users are not taking sufficient precautions because they are not aware of the Internet threat and/or lack the technical ability to mitigate it. Our research question asks whether intrusion detection and management can be effectively offered as a web service to users in order to better protect them and heighten their awareness of the

Internet security threat (raising threat awareness has been identified by the OECD as a key factor in fighting malware [28]). We propose a defence-in-depth approach to protecting users and the addition of an extra security layer, an IDS (Intrusion Detection System) designed with the following goals. It should:

- Demonstrate to users that they are targets in order to raise their security awareness
- Provide feedback to users on the behaviour of their machine (as seen by others on the network) in order to detect possible malware infection
- Be remotely administered and configured to relieve the user of technical complexity
- Offload intrusion analysis to make it suitable for deployment on mobile devices of limited processing power
- Increase vigilance only in the face of a possible attack in order to limit excess network traffic
- Be web accessible so mobile users (business and recreational) are free to roam networks and the Internet and receive constant feedback
- Export a web interface to ensure users receive feedback through a medium the majority of them are comfortable with: their browser
- Employ open source tools to avoid duplication of effort
- Be simple to install compared with other IDSs

This thesis describes the motivation for, design, implementation and operation of a system that attempts to meet the above goals. The system is deployed on and results generated from a university campus network. The campus network makes an ideal test setting since it is exposed to the risks the proposed system seeks to mitigate: the network is used daily by a large number of users of varying technical expertise and wireless LANs are open and offer anonymity to would-be intruders.

1.3 Thesis Structure

The remainder of this thesis is structured as follows:

In Chapter 2 the history and development of computer networks is reviewed. The Internet and World Wide Web along with their users are given particular attention. With an understanding of the context in place we move on to look at Internet threats and a taxonomy of such threats is presented. Further evidence is presented to highlight the growing Internet security problem.

In Chapter 3 the range of currently available defences for dealing with network threats is described including firewalls, antivirus software and intrusion detection systems. Because it is the focus of the work described here, particular attention is paid to the area of intrusion detection and a history of work in the area is presented.

In Chapter 4 a distributed, WWW-enabled and remotely administered approach to intrusion detection and feedback is proposed. The design and implementation of a prototype system are described. Its goals (listed above) are to mitigate the threats described in Chapter 2 while avoiding the complexity of the systems outlined in Chapter 3.

In Chapter 5 results of an evaluation of the system described in Chapter 4 carried out on a university campus network are presented and analysed.

Conclusions and suggestions for future work are given in Chapter 6.

2 The Internet Security Threat

Having asserted in Chapter 1 that the Internet is an increasingly hostile operating environment, this chapter describes and analyses a representative sample of the Internet security threats faced by today's users. Before considering specific threats and the attacks that realise them, however, some background material on the evolution of networks, the Internet and the World Wide Web (WWW) is presented. An understanding of the Internet's infrastructure and its users is necessary in order that attacks presented later can be analysed in context. A successful attack often entails the subsequent installation of some malware and we examine how the latter has evolved with the Internet and describe some of the modern malware at the hacker's disposal.

The chapter is structured as follows. In Section 2.1 a brief history of computer networks, the Internet and the WWW is presented. The growing economic importance of the Internet and its general role in society today is discussed. This is followed in Section 2.2 by a more detailed description of Internet infrastructure and the network model behind the Internet: TCP/IP [29]. The model's constituent layers and some of the major protocols inside those layers are examined. With the background material covered an exploration of Internet security threats is presented Section 2.3. Where applicable, attacks are related to the network layers of Section 2.2 that they target. While Section 2.3 deals with how attackers exploit vulnerabilities and users, Section 2.4 looks at the malware attackers leave behind once they have gained control over a host. A summary and conclusions are presented in Section 2.5.

2.1 A Brief History of Networks, the Internet and WWW

A computer network can be defined as a distributed system consisting of a number of separated but interconnected computers, where those computers communicate with each other to exchange and share resources and information [30]. Computer networks now play an essential role in meeting the world's information gathering, processing and distribution requirements. To better understand how they work, we first briefly consider below the history of computer networks highlighting some major milestones along the way. This discussion leads us to the modern Internet and WWW and we examine their expanding cultural and economic roles.

2.1.1 From ARPANET to the Internet

Designed by John Mauchly and J. Presper Eckert of the University of Pennsylvania and funded by the US military, the first electronic computer, ENIAC (Electronic Numerical Integrator and Computer) [31] was unveiled on 14th February 1946. By the 1950's computer usage had expanded to the extent that the ability to flexibly share information and resources between them was becoming a requirement and in 1954 the US Air Force began funding the development of one of the first wide area networks at the Massachusetts Institute of Technology (MIT). The result was SAGE (Semi-Automatic Ground Environment) a wide area network that linked radar installations [32]. It was deployed in 1963 and not decommissioned until 1983. One major concern for the US Air Force was the survivability of the network (and communications in general) in the face of a Soviet nuclear attack. Thus over the course of the SAGE project Paul Baran of the RAND (Research AND Development) organisation was commissioned to research survivable communications networks. He developed the concept of packet switching networks and published the details in 1964 in "On Distributed Communications Networks" [33].

The Advanced Research Projects Agency (ARPA) was established in 1958 by the US Department of Defence to develop military science and to keep pace with the Soviet Union's technological development. A computer network was envisioned that would allow ARPA agencies and researchers to collaborate and exchange information. The network was to be called ARPANET. When it came time to develop ARPANET Larry Roberts was recruited from MIT in 1966 to act as its chief architect. He adopted Baran's packet switching approach. ARPANET the world's first wide area packet switched network went live in 1969 when the first host to host message was sent from the University of California Los Angeles to the Stanford Research Institute. By the end of 1969 ARPANET had four network nodes. Fifteen nodes were added during the early 1970s, including Harvard University and the National Aeronautics and Space Administration (NASA). The addition of University College London and Norway's Royal Radar Establishment made ARPANET a global network.

In 1983 the US military handed control of ARPANET to public authorities (after establishing their own MILNET network). The removal of military interest meant researchers could use ARPANET for broader public interests. The success of

ARPANET inspired the creation of other networks like CSNET, EUNET and NSFNET. By the 1980s there were ARPANET gateways to external networks across North America, Europe and in Australia. ARPANET was by now just one network in a network of networks and the Internet was born. A standard was required to enable different computers on different networks to communicate and TCP/IP [34, 35] was adopted.

In 1984 the number of Internet hosts was 1000, by 1987 the number had increased to 10,000, and in 1989 the number reached 100,000. The first dot com domain *symbolics.com*, was registered in 1985 [36]. In 1989 the first large-scale Internet worm [37] infected thousands of Internet hosts. ARPANET was retired in 1990 and universities still connected to it were moved to NSFNET.

Before 1990 much of the Internet served as a government-subsidized research and educational tool for universities. The primary user groups were academic, government and industrial researchers [38]. There were four main applications available for users. The File Transfer Protocol (FTP) [39] was developed to copy files between nodes on the Internet. Articles and databases were shared in this way enabling collaborative research between different universities. E-mail has been available since the early days of the ARPANET. Researchers used news applications to exchange topics and ideas in computers, science and politics over USENET [40]. Equipped with a username and password users could remotely access other machines using the *telnet* [41] program.

2.1.2 The Internet and WWW Explosion

Tim Berners-Lee invented the WWW [42] in 1989 and is today the director of the World Wide Web Consortium (W3C), which oversees its continued development. The WWW is an architectural framework of interlinked hypertext documents accessed via the Internet. The WWW and Internet are not equivalent. The WWW is one of many Internet services. WWW applications do not change any of the underlying facilities provided by the Internet but rather make them easier to use through browsers. A browser provides a friendly GUI that makes it straightforward for beginners to access text, images, videos and other multimedia available on the WWW. The development of the WWW and browser technologies enabled and triggered the explosive growth of the Internet. By the end of 2008, the total number of Internet hosts was approximately 565

million [43]. The current IP protocol, IPv4 is set to "run out" of free IP addresses and its successor IPv6 is being phased in as the Internet continues to expand.

2.1.2.1 The Internet in the Home

Once the Internet was transferred from military to civilian control and its operation privatised it became much more accessible to the general public. The WWW opened up possibilities to many people for information gathering, social interaction, entertainment and self-expression over the Internet. Thus the Internet was transformed from a research tool into a popular medium. The Internet was no longer an academic and military research tool but a public utility.

According a report by the United Nations [44], the number of Irish broadband subscribers was 10,600 in 2002, only 0.3% of Irish citizens. Over the following four years, the number increased nearly 50-fold, and was 517,300 by 2006, 12.3% of the population. The data from the latest report shows that by the end of 2008, the number of broadband subscribers in Ireland had reached 1.2 million [45].

As the Internet has become more important in everyday life so mobile Internet access has become a requirement. People are no longer satisfied with accessing the Internet from a fixed location, they expect to use the Internet whenever and wherever they require. Numerous mobile devices have been developed in recent years, laptops, PDAs (Personal Digital Assistant) and smart mobile phones, such as the iPhone come WiFi- and Internet-ready. Mobile broadband is now available for these devices. From the beginning of 2008, most Irish mobile service companies such as Vodafone, O2, and Meteor provide mobile broadband services. Data from the latest report at the end of 2008 [45], indicated the number of users who used mobile broadband from their mobile phones or laptops to access the Internet was up to 200,000 in Ireland. According to the same report between 2008 and 2009 there was a 41% increase in wireless access points in caf  s, airports, hotels etc. in Ireland.

Today there are 1.6 billion Internet users around the world of varying age and levels of computer knowledge and security awareness. Compared with user groups before the 1990s, current Internet users exhibit a broader age range, are less sophisticated (in the computer security sense), and are more demanding of rich Internet content.

2.1.2.2 The Internet in Business

The WWW has revolutionised the way business is conducted. Many companies use the Internet as their primary means of enabling business transactions and generating revenue. For some, revenue is solely generated through e-commerce. For example, to purchase products from Dell [46] or Amazon [47] a customer must go online. The development of TLS/SSL cryptographic protocols used to encrypt sensitive information over a network was key in making such business possible. Other companies sell services and not products over the Internet e.g. web hosting companies will provide, for a fee, a hosting service to companies who do not wish to purchase and maintain their own hardware, such as the WildWestDomains.com [48] in U.S. and hosting365.ie [49] in Ireland. Internet Service Providers (ISPs) sprang up in response to the business opportunities arising from the Internet access requirements of both business and home users. The business of advertising the products and services provided by companies has led to the emergence of new revenue generating models. One of the most obvious and successful examples is Google [50]. Google provides free services to Internet users and revenue is generated through advertising third party products and services to those users.

Estimates of total online retail sales for 2002 were 43 billion USD for the United States and 28 billion USD for the European Union [51]. At the end of the 2007, online retail sales were up to 175 billion USD in United States, and this number was expected to reach 204 billion USD at the end of the 2008 [52].

The Internet is also used to promote the reputation of companies and to provide support services to customers. Corporations and many small companies today have their own website. At the very least such websites provide an overview of the company and advertise its products. According to the 2007-2008 report of the United Nations [44], by the end of 2006 in Ireland, 94% of enterprises were using the Internet and 67% of enterprises had interactive websites.

2.2 Network Communication Layers and Protocols

As the Internet expanded, standardisation of the software and hardware that enabled it became an issue. Models of network architecture were developed and protocols were proposed to implement and comply with those models. In this section we concentrate on one particularly important model, the Internet Protocol Suite, more commonly referred

to as the TCP/IP model [29]. The TCP/IP model is fundamental to today's Internet and examining it will allow the threats covered in Section 2.3 to be analysed in context. It should be noted that although we focus here the TCP/IP model, others exist. One heavily referenced alternate model is the Open Systems Interconnection (OSI) reference model [53]. The OSI model was designed in a top-down fashion and over the course of its lengthy standardisation process the bottom-up TCP/IP model gained widespread industry acceptance. Protocols associated with the TCP/IP model are the more prevalent today.

2.2.1 Network Communication Layers

Models abstract network implementation into a number of layers. Layers capture the function their contents serve in the overall network. In the resulting service stack each layer provides support to the one above and a layer need not be concerned with how the services below it are implemented.

The TCP/IP model has four layers (OSI has seven). From the top down they are: the application, transport, Internet and link layers. To communicate over the Internet, a host must implement one protocol from each layer. The layers and their constituent protocols are described in more detail in Table 2.1.

The TCP/IP Model	
Layer	Functions and Protocols
Application	The application layer is the top layer in the TCP/IP model. It contains the higher-level protocols that provide services directly to applications and common system services. The protocols in this layer include Telnet, FTP, SMTP (Simple Mail Transfer Protocol), SNMP (Simple Network Management Protocol), POP (Post Office Protocol), TLS/SSL (Transport Layer Security/Secure Sockets Layer), HTTP (Hypertext Transfer Protocol), SOAP (Simple Object Access Protocol) and DNS (Domain Name System).
Transport	The transport layer is responsible for encapsulating and abstracting data/messages into packets suitable for transfer and delivery to hosts. It provides end-to-end communication services for applications. There are two primary protocols in this layer, TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).
Internet	The Internet layer is used to transport packets from the initiating host across network boundaries to the destination host specified by IP address. Packets can be injected into any network and travel independently of each other through different networks to their destination. Protocols in this layer include IP (Internet Protocol) and ICMP (Internet Control Message Protocol).
Link	The link layer is the lowest layer in the model. This layer is primarily concerned with the network interface and the physical and logical components used to interconnect nodes on the Internet. Widely used protocols that belong to this layer include ARP (Address Resolution Protocol) and MAC (Media Access Control).

Table 2.1 The TCP/IP Model Layers

2.2.2 Network Protocols

In addition to specifying an abstract layered network framework the TCP/IP model also specifies a set of protocols to meet the functions expected from each layer. In network communication, a protocol is an agreement between the communicating parties that dictates the process of communication, including the syntax, semantics and synchronisation of messages. Protocols define the standards and conventions that control connections and communications. Any violation of a protocol can make communication difficult or impossible. Numerous protocols have been proposed and implemented in response to different communication requirements. Below we give a brief summary of some of the protocols, from the bottom-up, in the TCP/IP suite.

Link Layer: ARP

ARP (Address Resolution Protocol) [54] is used by the Internet Protocol (IP) in mapping an IP network address to a hardware address such as the Media Access Control (MAC) address used in Ethernet. For example, a machine will cache the MAC address of its gateway since that is the machine it sends packets to before they reach the Internet. Although the ultimate destination address of the packet is remote (given by an IP address) the MAC address is that of the next machine. On its journey a packet's MAC address will be rewritten as the packet traverses the Internet. MAC addresses are thus for short-haul routing compared to long-haul IP routing.

Internet Layer: IP

IP (Internet Protocol) [55] is a core protocol in the TCP/IP model. It is used for transmitting and receiving data across a packet-switched network. Each endpoint on the Internet has at least one IP address that uniquely identifies it among all endpoints. The task of this protocol is to deliver data from the source addressed host to the destination addressed host by routing packets through networks. IPv4 is the most widely used version today and provides a 32-bit IP address space. However as the number of Internet hosts increases IPv4 is approaching its end-of-life. IPv4 addresses may run out as soon as 2011 [56]. IPv6 is slowly displacing IPv4 and will provide a 128-bit IP address space.

Internet Layer: ICMP

ICMP (Internet Control Message Protocol) [57] is a supporting protocol in the TCP/IP model. The operating systems of networked computers use ICMP to send error messages and report problems such as undelivered packets. Typical problems include “destination unreachable”, “router unreachable” and “requested service does not exist”.

Transport Layer: TCP

TCP (Transmission Control Protocol) [58] is another core protocol in the TCP/IP model. TCP is a higher level protocol compared to IP. It is used by computer applications to send data over IP and guarantee delivery. Thus while IP specifies that a packet should be delivered from one host to another, TCP specifies additional streaming properties that guarantee both delivery and order of delivery. TCP is a stateful protocol in that it establishes a connection between hosts. It takes responsibility for dividing a message into individual packets, keeping track of each packet and reassembling packets into the original message at the destination. IP packets can be lost or delivered out of order, TCP solves these issues as it requests retransmission of lost packets or rearranges out-of-order packets.

Transport Layer: UDP

UDP (User Datagram Protocol) [59] is an alternative to TCP and is used to offer a limited delivery service that requires low transport overhead with non-guaranteed packet delivery. Message sending/receiving is direct, there is no prior establishment of a communications channel. Compared with TCP, UDP is a stateless, non-connected protocol.

Application Layer: DNS

DNS (Domain Name System) [60] is a service that translates domain names into IP addresses. A domain name is more meaningful and easier to remember than an IP address. It is in the highest layer of the TCP/IP protocol suite, and uses UDP as its communication protocol.

Application Layer: HTTP

HTTP (Hypertext Transfer Protocol) [61] is an application protocol running on top of the TCP/IP model. It defines a set of rules for transferring text, images, sound, video and other multimedia files over the WWW.

2.2.3 Example

As an example of this layered architecture in action consider a web browser seeking to send an HTTP request to a web server. The request, being constructed by a browser, makes HTTP an application layer protocol. The HTTP request is sent over TCP to ensure it arrives at its intended destination and its constituent parts arrive in the correct order. A TCP connection must be established in advance of sending the request. As the connection is across the Internet (although TCP does not care and is independent of how the source and destination are addressed) the connection must be established between two Internet hosts identified by IP address. IP addresses are tied to physical hardware using ARP (although again IP does not care how the IP address is associated with a specific piece of hardware).

2.3 Internet Threats and Attacks

Not only does the Internet make available useful and/or critical services to a connecting user but it also turns that user's machine into an addressable collection of resources and data. These resources are increasingly targeted by criminals. In addition to making their hardware accessible the Internet also makes the users themselves and, potentially, their personal data accessible. Given the number of Internet users is approaching 1.6 billion, potential for financial gain is attracting growing organised criminal activity. Internet users are thus under threat of attack. In this section we look at an illustrative subset of the attacks they may face.

A "threat" is defined by in the ISG (Internet Security Glossary) [62] as: "A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm". Thus a threat is something that *may* occur and exists even in the absence of an attack. The three pillars of security are confidentiality, integrity and availability [63]. An "attack" is defined as an "assault on a system" and a successful one will breach at least one of the three pillars of security.

Users put in place defences to fend off attacks and lower the threat level they face on the Internet. Given the vast array of attacks to which Internet applications and users may be subjected we can examine only a sample here. We broadly classify Internet attacks into four categories:

- Attacks against services
- Attacks against applications
- Attacks against users
- Attacks against configuration errors

Below we explain and give examples of each. We also reference some of the tools available to attackers to facilitate implementing attacks.

2.3.1 Attacking Internet Services and Resources

Attacks in this category are denial-of-service (DoS) attacks. In a DoS attack consumers are prevented from accessing a provider's services [64]. For a service provider this can result in serious financial loss. For a consumer it can, at the very least, cause inconvenience. DoS attacks fall into two categories: vulnerability exploiters and resource consumers.

Vulnerability Exploiters

The Ping of Death (PoD) [65] is a classic example of this kind of DoS attack from the 1990s. It is an attack that exploits a vulnerability in the reassembly process of IP packet fragments to crash machines. Simple ICMP ping utilities can be used to implement this kind of attack and gave the attack its name although the vulnerability is in the IP and not in the ICMP protocol. The TCP/IP protocol allows a single IP packet to have a maximum size of 65535 bytes. A normal ICMP ping packet is a few bytes in size. However, in the attack an oversized ICMP ping packet can be sent over IP if fragmented. When a target machine received and reassembled the packet whose specified size was larger than the maximum allowed, a buffer overflow could occur. Many machines froze, crashed, or rebooted on receiving such a packet. This attack was anonymous because the source IP address could be spoofed (no connection was established). Operating

systems have since been patched to solve the problem by adding checks in the IP packet reassembly process.

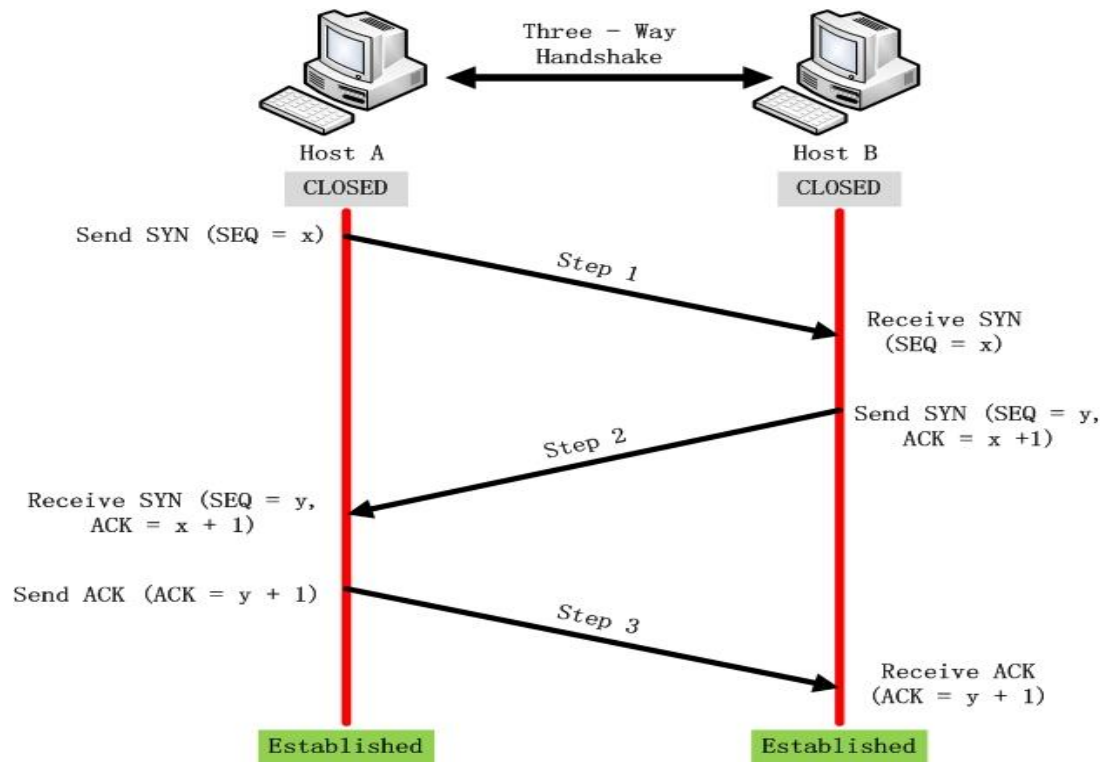


Figure 2.1 TCP Three-way Handshake (adapted from [66])

Resource Consumers

Another variety of DoS attack aims to exhaust so many of a provider's resources (bandwidth, CPU cycles, memory) that it can no longer maintain adequate service levels. Businesses that rely on the Internet for revenue streams are threatened by this kind of attack. So too are public utilities implemented on IP networks. Such attacks require little skill to implement and the necessary tools are freely available on the Internet [25]. Over recent years, many web servers have been targeted by these attacks including those operated by Yahoo, eBay and CNN [67] causing their websites to be unreachable for a period and incurring significant revenue losses. In recent years, a more destructive version of the DoS attack has emerged: the distributed DoS (DDoS) attack. In a DDoS attack the attacker bombards the victim from a botnet army that may contain thousands of nodes [68]. DDoS attacks are next to impossible to defend against given the difficulty faced by the victim in distinguishing between legitimate and attack traffic whose purpose is solely to consume resources. Botnet armies can be leased by the hour for the purpose of implementing such attacks. The 2007 E-Crime Watch Survey [69] reported

49% of 671 surveyed security executives and law enforcement officials had experienced DoS attacks in 2006. Even states are not immune, between April and May 2007 various Estonian web sites (e.g. banks, schools and government agencies) were crippled by DoS attacks. Early investigations suggested Russian hackers might be involved in the attacks. However, a 20-year-old Estonian student Dmitri Galushkevich admitted to participating in the attacks and was fined [70].

The SYN flood attack is a good example of one that targets resources. A flood of TCP connection requests is sent to the target system. In each request the SYN flag is set and the source IP is forged. The SYN packet is the first step of the three-way handshake that establishes a TCP connection (see Figure 2.1). The flood causes the victim's kernel socket table to become saturated with half-open connections as the operating system waits for an ACK response from a nonexistent IP address. The victim is soon unable to accept new legitimate connections. This is a low level attack against the TCP protocol of the Transport Layer in TCP/IP model. One technique that guards against SYN attacks is the cookie approach [71]. Under this approach, Host A initiates a TCP connection by sending a TCP SYN packet to the Host B (see Figure 2.1). Host B sends back the appropriate SYN + ACK response to Host A but discards the SYN queue entry from the TCP connection table. One of the values in Host B's response is a sequence number y generated by Host B. According to the TCP specification, the first sequence number sent by a host can be any 32 bit value. Host B carefully constructs the sequence number y according to certain rules, so that if a subsequent ACK response $y+1$ is received by Host B, Host B will be able to authenticate it to determine whether it is a legitimate reply. Basically, rather than the server allocate resources it embeds the equivalent information in the sequence number sent to the client which will subsequently be returned to it.

2.3.2 Attacking Applications

As stated in Chapter 1, classic hacker behaviour follows a three step process: Network reconnaissance, host/service enumeration and finally exploitation. For each of these steps the hacker has a selection of tools to choose from. The target is selected during the network reconnaissance stage. Here tools such as *nslookup* and *dig* [21] help the intruder gain a high level overview of the target by querying publicly available information. During the second stage accessible hosts are detected and scanned for

services using a network scanner. *Nmap* [22] allows the identification of the target operating system and the versions of the various network services it exports. Vulnerabilities due to users failing to apply patches in a timely manner [14] are common. For example some estimates put the number of machines infected by the Conficker worm at 15 million [72, 73] and this despite the patch for the underlying problem being released two months before the worm first appeared [74].

Once a vulnerability has been identified the next step for the hacker is its exploitation. Here a library of exploits such as that provided by the Metasploit project [23] does the hard work. The attacker passes to Metasploit a vulnerability descriptor and specifies the level of control she wishes to gain over the host. Metasploit generates the payload (an exercise that used to require some assembly programming skills) and will even inject it into the target host. Once the host is controlled it may become part of a botnet and from then on is controllable using a DDoS toolkit such as Trinoo [25]. Once the intruder has control of one host and gained a foothold in the target network she typically targets other hosts on the same network under the assumption they contain similar vulnerabilities.

An alternative to the classic attack pattern described above has also emerged in recent times. In a so-called “drive-by” attack a user is induced to visit a web site. The web site contains script that attempts to compromise the user's browser and use it as an attack vector for malware installation. The web site may be run by the attackers or by a legitimate but hacked third party. On visiting one such infecting-site researchers reported the installation on the visiting host of 50 malware binaries [75]. The latter study identified 450,000 URLs that attempted malware installation against vulnerable browsers. As mentioned earlier, 60% of users are not using the most secure version of their browser [14].

Whether the attacker targets a service directly or induces the user to visit a site it is vulnerabilities in the application that are exploited. Applications are becoming more complex as users demand more features. Complexity (often regarded as the enemy of secure code [76]) and tighter deadlines in a commercially competitive software industry mean bugs are introduced into software products. A bug may allow an application to carry out more actions than intended by design. When those actions implement an attack, the bug is an exploitable security vulnerability. If the application can be accessed across the Internet then the vulnerability is remotely exploitable. Such vulnerabilities

are the most serious particularly when they occur in common services such as web [77] and RPC [78] servers. As usual there is a wide variety of attacks that fall into this category. Here we look at two common and dangerous ones: buffer overflow attacks and SQL injection attacks. Both feature regularly in "Top 10" security flaws [79]. Their underlying cause is a failure to properly sanitise untrusted data.

Buffer Overflow Attacks

In a C/C++ program it may be possible, as a result of careless programming, to write to a container more data than it can hold. The lack of type safety and bounds checking in C/C++ means no exception is thrown or error is reported. Thus a bug in a C/C++ program may for example allow more user input to be written to an array than it can cater for. The extra data overflows the buffer and overwrites adjacent process state. When the adjacent state is control-flow-related an attacker may take over the process and cause it to execute arbitrary code. This is a buffer overflow attack. Buffer overflow attacks remain common vulnerabilities despite a widespread understanding of the issue. Typically they occur in C/C++ programs when unsafe functions are used to process user input and carry out no bounds checking on that input.

Figure 2.2 illustrates the attack in action where function A has just called function B. A stack-based four byte character buffer, here called *string*, is used to store data supplied by an untrusted source. Assuming an unsafe C/C++ string handling function e.g. `strcpy` is involved a buffer overflow attack is possible. By supplying an overly long input an attacker can cause the buffer to overflow and can rewrite adjacent state. Here that state includes a saved frame pointer “%ebp” and a return address. The return address is the location in memory of the instruction in A to be executed once the currently executing function B has completed. The frame pointer and return address are completely under the attacker's control. The return address may be set to point back into the buffer itself. The contents of the buffer are however also under the attacker's control. By placing in the buffer (here the buffer is small but in an attack it may be larger) machine executable opcodes and rewriting the return address to point into them the attacker can execute arbitrary code. (Unusually the code is executing on the stack and not in the text section of the process address space.) If the target application is running with elevated privileges, those privileges are inherited by the attack code. The opcodes or “payload” will typically launch a remote shell. Only 30-40 bytes are required to do so and give the attacker complete control of the host. While writing the exploit "shellcode"

may require some skill if done from scratch, the Metasploit Project [23, 80] provides a library of exploits to choose from for a variety of operating systems. The required level of assembly know-how is thus greatly reduced.

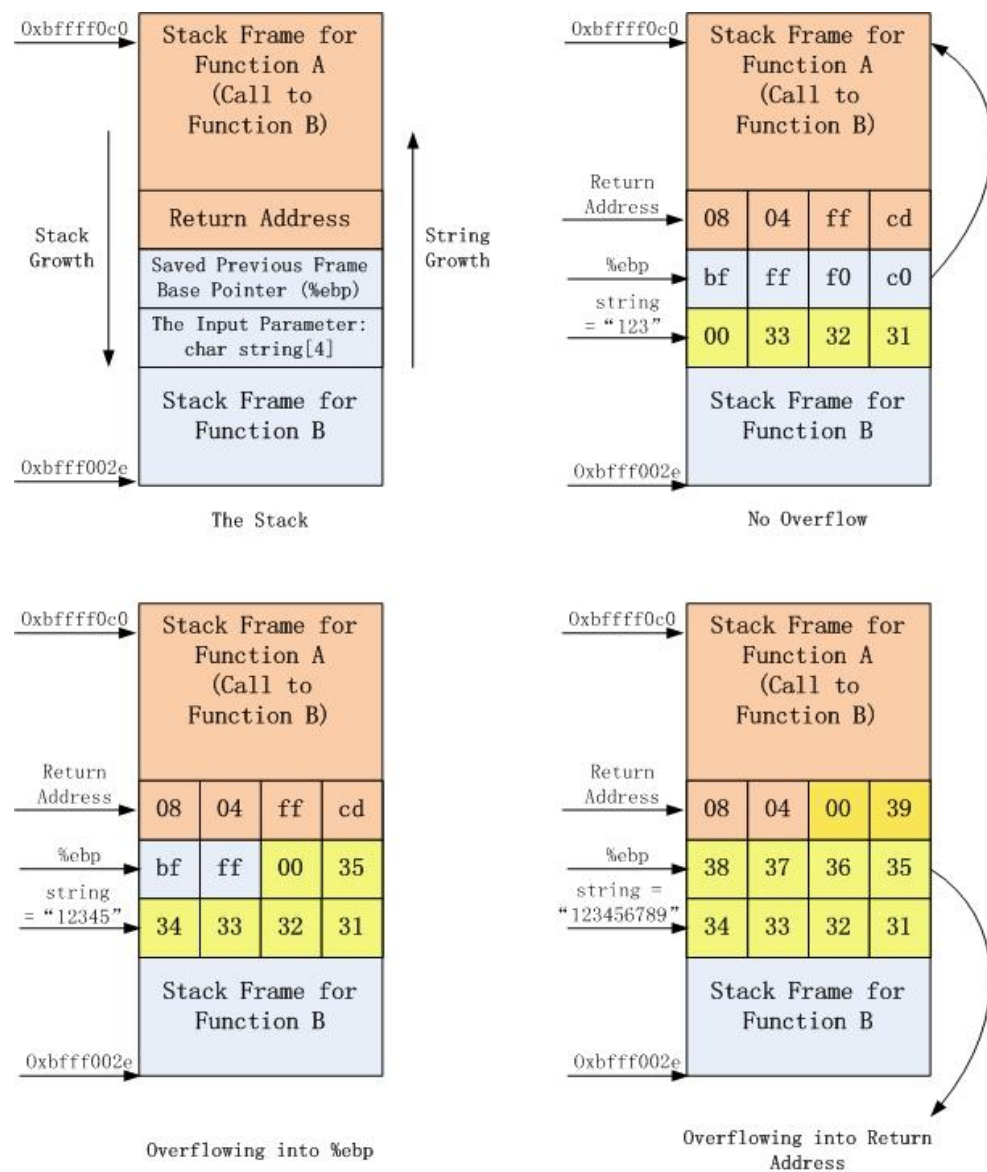


Figure 2.2 Buffer Overflow

The first Internet worm, the Morris Worm of 1988 [37], exploited a buffer overflow vulnerability in the UNIX finger daemon to infect thousands of hosts. Similar vulnerabilities arose in Microsoft's web server, Internet Information Services (IIS) in 1999 and again in 2001 [81]. Because they occurred in web servers they were particularly harmful and led to the spread of the Code Red Worm [81].

SQL Injection

SQL injection vulnerabilities again stem from a failure to adequately sanitise user input. Here the victim application is typically a database behind a web application [82]. Many web applications take user input through a HTTP form and use it in constructing an SQL query to be passed to a backend database. When user input is improperly combined with SQL statements the potential for an SQL injection attack arises. Consider the code below where *password* is a user supplied parameter:

```
statement = "select * from client where  
password = ' " + password + " ' ";
```

While the code looks innocuous and code like it is extremely common consider what happens when a user enters a password such as:

```
something' or 'x' = 'x
```

The resulting SQL query is:

```
statement = "select * from client  
where password = 'something' or 'x' = 'x' ";
```

This SQL statement selects and returns all rows in the table resulting in a breach of confidentiality. A successful SQL injection attack can cause disclosure of sensitive data, modification of data and even execution of arbitrary operations on the host. The problem is that control sequences/metacharacters present in user input are not properly escaped before being handed to the database. Such characters cause the database's SQL interpreter to "context switch" from a data processing mode to an instruction processing one. In the above example the metacharacters in question are the string delimiters ' and '. User data is treated not as data but as instructions in the same way as in a buffer overflow attack. The threat is removed by properly escaping user input through the use of prepared statements. Prepared statements hand the escaping problem over to the database. Since it knows its metacharacters it is best placed to escape them. For example a prepared statement could fix the above problem:

```
statement = "select * from client where password = ? ";
```

```
statement.setString (1, password);
```

```
resultSet = statement.executeQuery();
```

The “ ? ” is a placeholder, when the query is executed, a value will be supplied and replaces the “ ? ” in the query. Passing input to the statement as a parameter is sometimes referred to as a parameterised query.

2.3.3 Social Engineering Attacks

Social engineering techniques are used by attackers to target not technical vulnerabilities in applications but Internet users themselves. One approach in this category is phishing. Often initiated with an email request to check sensitive information such as an online bank balance the email purports to come from a trustworthy entity. It contains a link that if followed leads to a fake website. The fake website is constructed by attackers to look identical to the authentic one. Any usernames and passwords entered are submitted to the attacker. The FBI reported in 2008 that 265 million USD was lost to phishing and Internet fraud by US citizens [83].

Social engineering attacks can also be used to trick users into installing malware such as viruses, spyware, adware and even rootkits. The "ILOVEYOU" [84] and "Anna Kournikova" [85] viruses are cases in point. Delivered to victims as an email attachment, when opened a Visual Basic script was executed. The script emailed itself to the victim's contacts (extracted from their address book) and made changes to the host. Another common approach to inducing users into installing malware uses malicious websites that offer "free content" that can only be viewed with the supplied codec. The "codec" is in fact trojan software that once installed incorporates the host into a botnet army, installs a backdoor, spyware etc.

Application attacks and social engineering attacks may be combined. In the drive-by met earlier the victim is induced to visit some website, again perhaps by clicking on a link in an email. On following the link their browser is scanned for vulnerabilities by the malicious web server. Any weaknesses discovered are exploited and malware installed [75].

2.3.4 Attacking Misconfigurations

Lastly we consider attacks that exploit misconfigured networks and software. These attacks target common mistakes or omissions by users and network administrators. The means exist for secure system configuration but the user does not know how to apply them or is unaware of the risk of not doing so. The threat is obviously highest when novice or busy users are working with complex systems. Below we look at some common examples that illustrate the problem.

Networks

Domain Name Services (DNS) are regularly insecurely configured. A given domain will typically be managed by multiple name servers. Several are used for redundancy: if one goes down another name server can replace it. Thus name servers need to be kept synchronised and they do so by one requesting a zone transfer from the next. Zone transfers should only occur between internal name servers. It is often the case however that an attacker can connect to a corporate DNS server and successfully request from it a zone transfer. This results in detailed information on the organisation of the internal network (host names, IP addresses, software versions, hardware information) etc. being made public. Such details can aid an attacker in targeting weak points in the network. The problem is solved by simply configuring DNS to only transmit zones between designated primary and backup servers.

Wireless networks are also often vulnerable, being left open or configured to use WEP. WEP keys can be broken in a matter of minutes. In one sense using WEP is more dangerous than employing no encryption because it offers a false sense of security. Once a WEP key is broken, bandwidth is open to hijacking and network traffic can be sniffed. Modern wireless routers all offer the more secure and only slightly more difficult to configure WPA encryption option. Relatively few users employ it however.

The TK Maxx credit card theft in 2007 [20] clearly illustrates the wireless network threat. In the largest network security data breach in history 45.7 million debit and credit card records were stolen. The techniques used by the intruders to hack the TK Maxx wireless network were simple. War-driving [86] detected the vulnerable wireless network and was followed by key-breaking and network sniffing [87]. As a result of the

weak encryption, initial hacking and decrypting of the stolen data was a relatively simple task.

Research carried out by Magee [88] reveals the scale of the problem in Dublin, Ireland. On a war-driving exercise conducted over a return journey from the Dublin City University (DCU) campus to the city centre, 3143 wireless access points were revealed. (The areas covered included the Irish Financial Services Centre.) 25% of access points were openly accessible and applied no encryption whatsoever while another 60% relied on the easily broken WEP standard. Many access points also broadcast the network's SSID (Service Set Identifier) and gave it a meaningful name such as "Boardroom" and "Manager's Office".

An Internet facing **firewall** for a large network may have to deal with extremely large volumes of data and in the interests of maximising throughput certain features may be disabled by default. One such example relates to packet fragmentation. Rather than having the firewall hold packet fragments temporarily and reassemble them before making an allow/deny decision it may blindly forward fragmented packets to their destination, leaving the host to make the decision. This firewall evasion technique is a favourite of hackers. *Nmap* allows the option of conducting scans using fragmented packets.

Software

Personal **host-based firewalls** require configuration. A default ruleset may allow all inbound and outbound traffic in order to get a user up and running rapidly. A README file advising rule modifications may never be read or if it is read may not be understood. (Linux distributions may install a reasonably configured firewall but by default the Linux firewall, the *netfilter/iptables* suite, allows all inbound and outbound connections.)

As software grows increasingly complex even experienced network administrators can make errors in deploying it securely. The *sendmail* program that implements an SMTP service for email routing and delivery on Unix systems is an example. Given its complexity, it may be deployed with debug support enabled. Debug support can be exploited to execute instructions on the host server. The Morris worm, in addition to exploiting a buffer overflow, also sought out debug sendmail installations and exploited them to propagate [89]. Mistakes in web service configuration are another source of

attack [90, 91]. Configuration errors in database applications such as unchanged default database administrator passwords are also common.

In summary, misconfigurations often stem from default settings chosen for user friendliness and convenience rather than security. Appropriately secure configuration takes understanding and requires regular revisiting.

2.4 Malware

An attack is often only the first step for a criminal. The real intent is the subsequent installation of malware which allows their presence to remain undetected and permanent (even after the original vulnerability is patched). Malware can be generally defined as "a piece of software inserted into an information system to cause harm to that system or other systems, or to subvert them for use other than that intended by their owners" [28]. Below we distinguish between how malware propagates and its capabilities.

Propagation

Up until the 1990s the malware world was dominated by viruses and trojans. A virus attaches itself to a host program and generally requires the activation of the host program or some user intervention in order to propagate. A trojan is malware disguised as some useful application the user is tricked into installing. The user intervention required for the spread of viruses and trojans is often incentivised through social engineering techniques (as mentioned in section 2.3.3). The propagation vectors (i.e. the means by which the virus/trojan spreads) include email (ILOVEYOU and Anna Kournikova), P2P networks, USB devices, shared network drives etc. An interesting recent take on propagation is the drive-by attack mentioned above where rather than bringing the malware to the user, the user is brought to the malware and a vulnerable browser is exploited to install a payload.

While a virus requires a host and activation of that host in order to spread a worm does not and combines self-replication and self-propagation abilities. With the rise of the Internet, worms can spread rapidly by exploiting vulnerabilities in common network services, infecting thousands of machines in a matter of hours. Examples of well known worms include Code Red [81], Slammer [92] and more recently Conficker [74].

Capabilities

Once an attacker gains a foothold on a host she typically embeds software in it such that long after the original vulnerability is patched she can maintain a presence on the machine. The host is then under remote and, depending on the privilege level at which the malware executes, total control by the attacker. Data can be encrypted and the owner held to ransom. Spyware such as keyloggers may be installed in order to steal confidential details such as passwords for online banks and other information suitable for identity theft. Industrial and international espionage can be implemented in the same way (in 2006 Michael Haephrati and his wife Ruth were extradited from Britain to Israel for creating a trojan used for espionage [93]). The owner may not be targeted directly but rather their machine may be incorporated into a botnet and become an unwitting player in a subsequent DDoS attack. Nowadays malware is typically general purpose in that it can be put to a variety of uses. Interestingly the attacker no longer contacts the malware directly in order to give it instructions but rather the malware contacts some central point e.g. a web site where its instructions have been posted. This helps ensure attacker anonymity.

As malware has evolved the high profile if "harmless" mass-spreading worms and viruses of previous years are being replaced by stealthier varieties more suitable for supporting criminal activity. The trend has been towards "rootkits". A rootkit is a sophisticated hacking tool that can provide permanent administrator-level access to a host. Once a rootkit has been installed, it provides the attacker with administrator level access to the system and thus provides complete control over it. What differentiates a rootkit from traditional malware is the level at which it operates: kernel level. A rootkit is thus part of the operating system and can interfere with its operation. This low level of operation presents significant challenges to host-based malware scanners: asking a rootkit-infected operating system whether it is infected with malware (through a userland malware detector) is pointless since all answers returned by the operating system are modifiable by the rootkit. Thus a rootkit can hide its existence and go undetected by users. The rootkit at kernel-level can subvert the monitoring and reporting mechanisms of a system, evade host-based security tools and rewrite logged system behaviour. Rootkit software is easy to obtain and can be downloaded freely [24]. Given their stealth and the difficulty faced by host-based software in detecting their presence it makes sense that third party feedback from its peers on how a host is

behaving on the network is valuable. The system described in Chapter 4 makes such information available.

The Malware Cost

The annual malware cost to US businesses was put at 67 billion USD by one report [26]. US consumers paid almost 4 billion USD over two years for malware removal [15]. ISPs pay when dealing with infected hosts on their networks: there may be degraded service for legitimate ISP customers if the infected hosts are generating large volumes of network traffic. The ISP may become blacklisted and lose reputation if its hosts have been reported for generating spam [94]. Companies suffer financially when forced to over-provision in terms of bandwidth and processing power in order to deal with the DDoS threat. Critical infrastructures running on IP networks are also at risk: in 2003 the Slammer worm infected a nuclear power plant's safety monitor in the US (a contractor had connected to their home network from a host in the plant leading to infection) [28]. Even the CERN particle accelerator has been hacked [95].

2.5 Summary

In order to understand the Internet security threat it is necessary to have an understanding of its underlying technology and users. This chapter gave a brief history and overview of computer network development in Section 2.1. The explosion of the Internet and WWW were also covered. Network communication layers and protocols were examined in Section 2.2. In Section 2.3 we surveyed specific Internet security threats and the attacks that realise those threats. Attacks were analysed as belonging to different categories and associated with corresponding network communication layers in order to be seen in context. In general attacks have been moving up the network stack over time, no longer targeting protocols but users and applications. Lastly in Section 2.4 we looked at the malware attackers install on victim machines and its capabilities.

In response to this range of threats a wide array of defences has been proposed. In Chapter 3 we examine those defences. Given intrusion detection is the focus of this thesis the majority of the chapter is devoted to exploring the history of that field. The latest intrusion detection approaches and example applications will also be presented.

3 The Defences

An overview of the typical threats against which computer networks, applications and users must defend themselves was presented in the previous chapter. This chapter will describe the typical forms those defences take: firewalls, antivirus software and intrusion detection systems. All defences are designed to prevent, detect, identify or remove threats. Each form of defence has different functionality and features to protect resources from certain kinds of threat. A firewall protects computers or computer networks from unwanted or malicious connection attempts and network traffic. Antivirus software focuses on protecting the computer system and applications from infiltration across the network by malware. Another defence is intrusion detection applications, which are used to detect, prevent and react to intrusions or threats against computers or networks. In summary, a firewall aims to prevent intrusion, antivirus software tries to detect infection during or after the fact and an intrusion detection system attempts to detect and respond to attacks in real time.

Since the focus of this thesis is intrusion detection, this chapter focuses on the development of intrusion detection systems and technology. In addition to describing the different approaches to intrusion detection, a short history of the area is presented and some major events in the area are mentioned. In line with the goals outlined in Chapter 1 our focus is on open source and free systems but some commercial intrusion detection systems will also be referenced.

The chapter is structured as follows: In Section 3.1 firewalls are discussed. This is followed in Section 3.2 by a description of antivirus software. Section 3.3 provides a more detailed description of intrusion detection technology. A brief history of intrusion detection is first presented. The primary components of intrusion detection applications are presented next. With reference to the latter components intrusion detection applications can be broken down into different categories. These different approaches are then compared and their advantages and disadvantages discussed. A description of several modern intrusion detection systems is also presented. The chapter concludes with a summary in Section 3.4.

3.1 Firewalls

A firewall is a form of defence for protecting computers and computer networks. A formal definition from the Internet Security Glossary [62] states that a firewall is: “An internetwork gateway that restricts data communication traffic to and from one of the connected networks (the one said to be “inside” the firewall) and thus protects that network’s system resources against threats from the other network (the one that is said to be “outside” the firewall).” This definition is network-oriented. Today a firewall in general is a set of computer applications designed to protect a host or network from network-based security threats such as unauthorized access attempts, unwanted connections etc. The rapid growth of the Internet and the mobility of its users has meant a firewall is more important than ever in protecting users. Today a firewall can be implemented in hardware, software, or as a combination of both. A firewall enforces that part of a business’s security policy that is designed to protect hosts and computer networks such as corporate LANs. Home user PCs must also be protected by a firewall when connected to the Internet. Thus there are two firewall types: network-based and host-based. Both kinds of firewall are presented in next two sections.

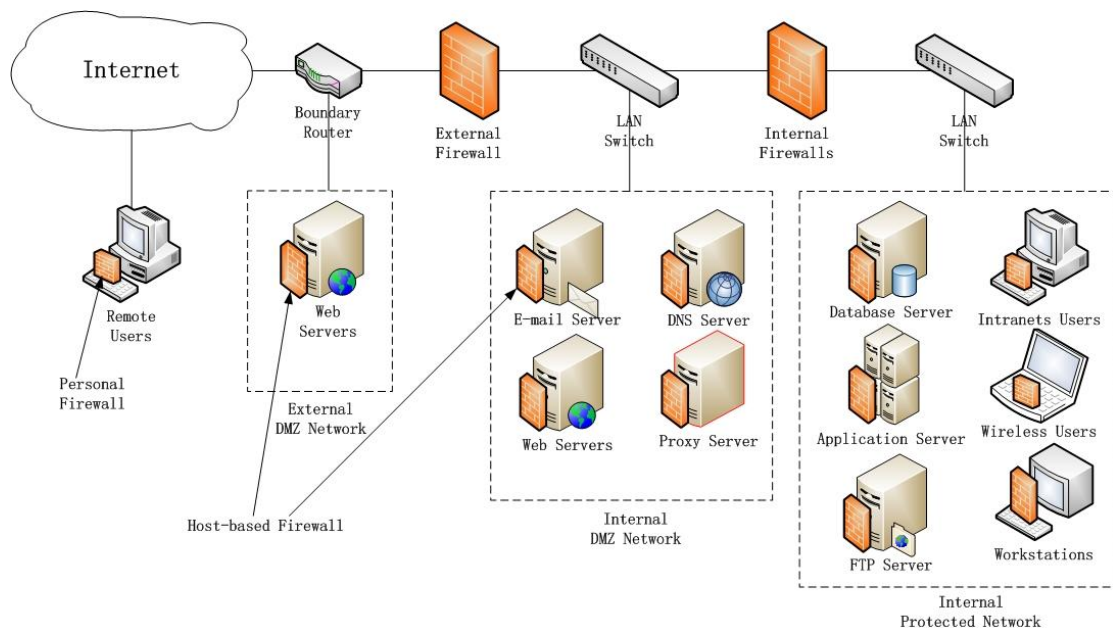


Figure 3.1 Firewalls Locations

3.1.1 Network-based Firewall

A network-based firewall is the computer or set of computers that inspects all traffic passing between untrusted networks such as the Internet and trusted networks such as internal networks. This type of firewall can be installed at a single location e.g. on a computer connected to routers and switches. Multiple firewalls can also be installed in a corporate network to protect subnets with different security requirements.

Network-based firewalls examine both incoming and outgoing packets to determine whether to forward them toward their destination or to drop them. To install, configure and maintain network-based firewalls is one of a security administrator's most important tasks. The principal role of a firewall is to provide a protective barrier between external, potentially untrusted sources of traffic and internal networks or servers. Figure 3.1 (adapted from [64]) shows the typical sites where network firewalls are located. An external network-based firewall is placed at the edge of a local or corporate network, just inside the boundary router. One or more internal network-based firewalls can also be deployed and are used to protect different enterprise LAN networks and servers. The area between external and internal firewalls is called the Demilitarized Zone (DMZ). One or more networks or devices, which require external connectivity but need some protection, are located in the DMZ region.

In Figure 3.1, two DMZs are established: external DMZ and internal DMZ. Web servers that need less protection because less critical information is located on them can be placed in an external DMZ. However host-based firewalls are required to protect these web servers. The external firewall provides protection for the internal DMZ allowing access to remotely accessible services and also provides a basic level of protection for the remainder of the network. Internal firewalls have stricter filtering capabilities compared to the external firewall, in order to protect inside servers from external attacks. Internal firewalls provide strength-in-depth and have the ability to defend against potential attacks from within the internal DMZ to the remainder of the network and from the remainder of the network to the DMZ area. Internal firewalls also have the ability to protect internal networks from inside attacks between internal networks. These firewalls are necessary to again provide defence-in-depth and to resist insider attacks since there is little point in securing the perimeter if anonymous attacks are allowed from an internal wireless network against other internal networks.

3.1.2 Host-based Firewall

A host-based firewall is an application used to secure an individual host. Many operating systems provide this kind of application. Host-based firewalls can be used to protect individual servers on corporate intranets such as a database server. They can be specifically configured for the server environment, such as to only accept communication on port 1433 for SQL Server or block all ports except 80 for a web server whereas a network-based firewall has to consider the entire protected network. User workstations can themselves run host-based firewall applications to restrict access to the services that may be on by default.

The personal firewall is a kind of host-based firewall. It controls the network traffic between a personal computer and the Internet or local network. Host-based firewalls or personal firewalls are essential given the mobility of modern users. Using a laptop from a workplace WLAN is presumably safer than using one in a public place such as an Internet café. In the former setting we place trust in the corporate firewalls, network administrators and in the other users of the network. In the latter setting we cannot afford to do so. Host-based firewalls or personal firewalls can give users a level of protection in a public place setting.

3.1.3 Packet Filtering

Packet filtering is a function of firewalls that examines each packet entering or leaving the protected hosts or networks and accepts, drops or rejects it based on pre-defined rules. Filtering rules can be relatively difficult to configure for non-technical, average users. Where packets match rules, they will be accepted (forwarded as normal), dropped (discarded silently) or rejected (discarded and responded to with an error). If there is no match to any rule then a default action will be taken.

Packets are typically filtered based on the header information within each packet such as the TCP and IP header contents. The header information which the user can draw upon in order to define rules includes the source IP address, the destination IP address, the source port, the destination port, the protocol, the source network etc. An example of a packet filter rule set from the Linux firewall *iptables* is shown in Table 3.1. In the first line, all input packets directed towards the HTTPS web application (port 443) on the IP address 136.206.18.95 are accepted. The next four lines are used to reject all SQL

SERVER database access (port 1433), HTTP access (port 80), SSH access (port 22) and FTP access (port 20, 21) on the same machine. The sixth line drops all packets from a suspicious IP address. The seventh line is use to hide the web server from ICMP requests by dropping all ICMP packets. The last line is configured as default rule: reject all input packets if no previous rule matched. In iptables the order of the rules is important, once a rule matches a packet, the action indicated in the rule will be implemented.

```
iptables -A INPUT -d 136.206.18.95 -p tcp --dport 443 -j ACCEPT
iptables -A INPUT -d 136.206.18.95 -p tcp --dport 1433 -j REJECT
iptables -A INPUT -d 136.206.18.95 -p tcp --dport 80 -j REJECT
iptables -A INPUT -d 136.206.18.95 -p tcp --dport 22 -j REJECT
iptables -A INPUT -d 136.206.18.95 -p tcp --dport 20:21 -j REJECT
iptables -A INPUT -s 61.164.117.35 -j DROP
iptables -A INPUT -d 136.206.18.95 -p icmp -j DROP
...
iptables -A INPUT -j REJECT
```

Table 3.1 iptables rules

Stateless Packet Filtering

Table 3.1 shows a typical stateless packet filtering rule set, which does not take into account the “context” of packets such as whether they belong to an established connection. This type of packet filtering takes no account of whether a packet is part of an existing stream (it stores no information on connection “state”). Instead, it filters each packet based only on information contained in the packet itself.

Stateful Packet Filtering

Stateful packet filtering, however, makes filtering decisions not only on the rule sets but also on the state of the corresponding connection. An active table of all in progress sessions is maintained during stateful packet filtering, called a connection state table [96]. Table 3.2 shows a simple connection state table. Each entry is a tracked TCP connection in states of new, established or closed. Once a connection-establishing packet is received with the SYN flag set, an entry is created with a state called new. The

received ACK will be authenticated with data in the state table, if the authentication is successful, the state of the entry will be updated. The same process will be gone through when the connection is ended. All incoming packets are compared to this table to assist in access control decisions. Compared to stateless packet filtering, stateful packet filtering is more powerful and has more information available when making an access decision. For example, with stateful packet filtering, the ACK scan can be effectively avoided. Stateful packet filtering can differentiate between valid and faked ACK packets by checking for the corresponding entry in the connection state table and comparing sequence numbers. If there is no such entry or sequence numbers are not correct, all invalid ACK packets will be dropped.

Source Address	Source Port	Destination Address	Destination Port	Flags	Sequence Number	...	Connection State
192.168.1.66	1056	137.59.26.57	80	SYN	New
192.168.1.7	3645	14.36.96.74	80	SYN&ACK	Established
192.168.1.155	4895	83.18.72.92	25	FIN	Closed
217.36.59.66	7836	192.168.1.95	80	SYN&ACK	Established
25.79.24.11	4514	192.168.1.95	80	SYN&ACK	Established

Table 3.2 Connection State Table (adapted from [96])

3.1.4 Proxy Server

A firewall acting as a proxy server intercepts all packets entering and leaving the protected network. For the packets leaving the protected network, the proxy replaces the source IP address with its own IP address and forwards them, in order to keep protected hosts and networks behind it anonymous for security. It can hide the true IP addresses of hosts in protected networks. Outside machines such as remote web servers can only communicate with the proxy server instead of every host on the internal network. A proxy server acts on behalf of the protected host. The typical location of the proxy server is shown in Figure 3.1. A proxy server can effectively hide internal IP address information in that it transparently replaces the origin address of traffic coming through it before passing the traffic to the Internet.

3.1.5 Summary

Firewalls are essential for restricting access to specific services for protected hosts or networks. They are also useful for auditing as it is possible to log all through traffic locally or remotely.

Firewalls also have their limitations [97]. They must be securely configured. They cannot protect any service or host to which access has been authorised. If an accessible application has flaws, the firewall cannot prevent an attack. The firewall also cannot stop attacks that bypass the firewall, such as separate dial-out or dial-in services in a corporation's internal LAN. Attackers could also break the weak WEP [18] encryption of a WLAN and access internal servers through the trusted WLAN instead of across firewalls. Another example that bypasses the firewall is where portable devices such as laptops and PDAs may become infected outside the network and once plugged back into the protected network are erroneously trusted. Malware may not be the cause of the internal attack, an employee may take part in an insider attack. Also firewalls provide no protection against application attacks such as browser exploits and social engineering attacks.

Many personal firewalls do not provide clear feedback for users and the fact that they are under attack is lost and so too is an important opportunity to raise security awareness. Furthermore, firewall logs may show information on the behaviour of other machines on the network. That behaviour, which may indicate the presence of malware, is not conveyed automatically to the owners of those machines.

A survey in 2007 showed that only one in seven Australian online users have firewalls [12]. Configuration of firewalls is increasingly complex as more applications wish to act as servers or make outbound Internet connections and an improperly configured firewall allows attackers to break into the network [98]. Also numerous and repetitive firewall-related security questions [99, 100] may result in users disabling security features altogether.

3.2 Antivirus Software

According to the Internet Security Glossary a virus can be formally defined as “A hidden, self-replicating section of computer software, usually malicious logic, that

propagates by infecting –i.e., inserting a copy of itself into and becoming part of – another program. A virus cannot run by itself; it requires that its host program be run to make the virus active.” [62]. With the Internet explosion and the increasing use of the Internet by millions of users, today the term “virus” means more than the latter definition constructed almost ten years ago. Today an “average” computer and Internet user commonly uses the term virus to refer to a variety of malware including adware and spyware, which do not have the required reproductive ability (but do often exhibit similarly malicious intentions). In general malware, including viruses have evolved to become the rootkits covered in the previous chapter.

Viruses and worms are often confused too. A virus requires that a host program be executed in order that the viral code be activated while a worm requires no host. It can propagate from one computer to the next over a network or the Internet by exploiting security flaws in common services (e.g. a web server). In 2006 65% of respondents to the CSI/FBI Computer Crime and Security Survey [101], had experienced viruses in 2006 and 16 million USD was lost to virus contamination.

As the term virus now refers to a broad range of malware so “antivirus” software refers to software that detects more than the presence of viruses. The meaning of “antivirus” has also expanded to meet this more general definition. In general antivirus software is a computer application that is designed to search a computer system for any known or potential computer viruses. As Internet use grew and attracted millions of business and personal users, antivirus applications became a growing industry. Antivirus software now can detect, prevent and remove adware, spyware and other forms of malware such as backdoors left behind by worms or trojans. There are many successful antivirus software products such as Norton AntiVirus from Symantec Corporation [102] and McAfee’s VirusScan [103].

Antivirus software is one of the earliest forms of computer defence. The first antivirus application “Reaper” [104] appeared soon after the first computer virus “Creeper”. It was developed by Bob Thomas and enhanced by Ray Tomlinson in 1971 [105]. Since then, antivirus applications have been used to protect computer system. Various strategies have been used by antivirus applications to search for and identify viruses. Methods are usually signature- or behaviour-based.

3.2.1 Signature-based detectors

Most antivirus applications use the signature-based approach because of its simplicity. They define and create a group of signatures capturing the specific attributes of known computer viruses. Signatures take the form of a database that includes viral opcodes, messages which viruses generate and file types they install. Generic signatures are used to look for unknown or potentially malicious opcodes (since viruses are usually polymorphic, detectors often look for a general decryption routine). Today antivirus software has become the first choice in personal defence and is installed immediately after the operating system on personal computers. It is up to the user to regularly update virus signatures.

3.2.2 Behaviour-based detectors

Another common approach used by antivirus applications is behaviour-based detection. This approach enables antivirus applications to monitor the behaviour of programs on a local computer to look for suspicious actions that violate general policies. For example, a program attempting to write data to into an executable program could be deemed suspicious behaviour. This approach has the potential to protect against unknown viruses showing up on a local system. However, the behaviour of non-malicious applications can change over time, and this approach can generate large numbers of false positives. ThreatFire [106] uses a behaviour-based approach.

3.2.3 Summary

Antivirus software is well-suited to detecting, preventing and removing known malicious software, such as viruses and worms. It can be scheduled to examine an entire system on a regular basis.

Antivirus software has its drawbacks. Firstly, virus signatures can obviously only be created and updated after the virus has been released and captured and analysed [107]. Users can be infected before updating to the latest signature set. Secondly, when antivirus software scans for viruses, it may significantly diminish the performance of the system and be disabled because of it. Viruses are now polymorphic and what signature detectors look for is the decryption routine. However code obfuscation techniques can make even the decryption routine difficult to identify.

3.3 Intrusion Detection

Intrusion detection technology is commonly considered the third form of computer and network defence after antivirus software and firewall applications. Intrusion detection is defined in the Internet Security Glossary [62] as “A security service that monitors and analyzes system events for the purpose of finding, and providing real-time or near real-time warning of attempts to access system resources in an unauthorized manner.” The unauthorised manipulation or access can compromise the confidentiality, integrity or availability of resources. The process of detection takes place by gathering and analyzing information from various sources within a computer or across a network for signs of intrusion. Evidence of intrusion can be extracted from system log files, system or user activities and network traffic.

Many network and security administrators implement intrusion detection by manually inspecting the logs of antivirus software and firewalls. They constantly check system and security log files and examine LAN traffic for actions of suspicious intent, known malware and unauthorized applications. By contrast an intrusion detection system (IDS) is a software application that automatically performs intrusion detection tasks such as inspecting system changes and sniffing network packets. A typical IDS is developed with the ability to detect a probable intrusion, log relevant information to files or databases, alert the administrator and deploy countermeasures to reduce the impact of an attack. Reactive measures can be implemented by the IDS alone or with the aid of other security tools such as firewalls.

An IDS can be host-based if it protects a single host by monitoring the configuration files of the system, security logs, user activities and so on. An IDS can also be network-based if it monitors network traffic. Some IDSs employ a combination of the two approaches in order to protect a networked host. IDSs can also be compared in terms of their detection mechanisms. The intrusion can be identified either by matching captured data against patterns or signatures of known attacks or comparing activities against a “normal” threshold. More details will be presented in the following sections.

3.3.1 A Brief History

The concept of IDS has been explored in the face of an increasing number of attacks on computers, major networks and applications. Below a timeline of major events over the history of intrusion detection is presented.

1980 – 1989

The notion of intrusion detection was introduced in Anderson's paper "Computer Security Threat Monitoring and Surveillance" in 1980 [108]. In this paper, the concepts of threat and attack were defined and an approach to auditing data was proposed to recognize them. This paper is regarded as providing the foundation for later intrusion detection application design and development.

The first IDS was developed between 1983 and 1986 by Denning and Neumann [109]. Called the Intrusion Detection Expert System (IDES), it used statistical anomaly detection. In 1983, a U.S. government project was launched by the Stanford Research Institute (SRI) International [110] led by Denning, the senior staff scientist of SRI. This new effort targeted intrusion detection development. Audit trails of government computer users were analysed and profiles created. The first functional IDS prototype, IDES was developed by SRI in 1984. Denning was one of the main developers of this system and the details of this system were included in her paper "An Intrusion Detection Model" [109] published in 1987. IDES was a real-time intrusion detection expert system, which had the ability to detect system penetrations, break-ins and abuses. The detection was implemented by looking for behavioural deviations of system usage based on profiles which represented the normal behaviour of the system in terms of statistical models.

In 1988, another IDS was developed by the Haystack project at the Lawrence Livermore National Laboratory (LLNL) at the University of California Davis for the U.S. Air Force, called the Distributed Intrusion Detection System (DIDS) [111]. DIDS extended existing IDS to monitor hosts across a small network. This system detected intrusions by analyzing audit data against defined attack patterns. The audit data was produced based on the raw events collected from each host, such as file accesses, system calls, process executions and logins. A commercial IDS, Stalker, was released in 1989 by

Haystack Labs, a company formed by the developers from the Haystack project. Stalker was based on DIDS.

1990 – 1999

The first network intrusion detection application, Network Security Monitor (NSM) [112], was developed by Todd Heberlein, who was a student at the University of California Davis and also participated in the development of DIDS. NSM analysed network traffic as the source of audit data. The idea of hybrid intrusion detection was subsequently introduced by Heberlein having worked on both NSM and the Haystack project [112].

In 1994, the first commercially available network intrusion detection device, NetRanger Sensor, was produced by the Wheel Group. The NetRanger Sensor was a network device that filtered suspicious activities based on a signature library. Cisco Systems bought Wheel Group in 1998. The NetRanger Sensor and software were subsequently integrated in the Cisco Router.

Snort, an open source libpcap [113] -based network IDS was developed in 1998 [13].

2000 – Present

IDSs have steadily grown more important in computer and network security. The 2007 E-Crime Watch Survey [69] showed 81% of 671 surveyed security executives and law enforcement officials were using network-based IDS and 71% of them were using host-based IDS.

3.3.2 Primary Components

Although numerous varieties exist, every IDS is composed of three functional components. These components are: data sensors, analysis engine and management console [64]. An IDS can be categorised in terms of the location of data sensors and the methodology used by the analysis engine to detect intrusions. Details of these three components are as follows.

Data Sensors

Data sensors are responsible for data collection and forwarding data to the analysis engine. The source of the input for the data sensors could be any part of an operating

system or application, such as user and system activity, log files, system call traces etc.. Another important source of information is network packets. The collected data may contain evidence of attacks and potential intrusions. The collected data is transferred to the analysis engine for further examining.

The locations of data sensors vary according to the information they collect. Based on this location, the IDS can be classified into host-based IDS, network-based IDS and hybrid IDS. With a host-based IDS, the data sensors are installed on the host and concentrate on collecting data on the host activities and states, such as system calls, application logs, file-system modifications etc. With a network-based IDS, the data sensors are installed at a strategic point such as a computer connecting to a hub, network switch or router in order to monitor the entire traffic of the network. A hybrid IDS combines the two approaches. It distributes data sensors to network hosts in order to collect both host behaviour and network traffic. More details on these approaches are presented below.

Analysis Engine

The analysis engine receives input from data sensors and is responsible for analyzing the input and determining whether an intrusion has or is taking place. The analysis engine also records possible malicious events in a database. The output of the analysis engine could be an alert to the identified threat delivered to the network administrator. The output may also recommend responses to the detected intrusions.

The analysis engine can use different mechanisms to detect intrusions. Based on the mechanisms, the IDS can be separated into signature detection IDS and anomaly detection IDS. Signature detection identifies intrusions based on a group of predetermined attack signatures. Anomaly detection identifies intrusions based on a threshold derived from statistical evaluations of normal computer host activities and/or normal network traffic. Both detection techniques are presented below

Management Console

The management console provides the User Interface (UI) to end users. The end user uses this console to configure the IDS, control the sensors and review the output of the analysis engine.

3.3.3 Signature Detection vs. Anomaly Detection

Two detection techniques are currently used by all IDSs: signature detection and anomaly detection. Neither are perfect and some IDSs use both approaches to enhance the accuracy of detection. Both detection techniques are described in this section along with their advantages and disadvantages.

Signature Detection

Signature detection is the most popular detection technique used by IDSs. The approach is also known as misuse, rule, pattern and policy violation detection. This technique attempts to detect intrusive activity by matching observed events or behaviour to a set of rules or attack patterns [114][115]. Thus, at the core of this technique is a set of signatures of known harmful activities such as failed login attempts, specific command or program execution profiles, attempted file accesses and network traffic. Those signatures may include a list of all known unacceptable actions of system users or packet contents associated with known attacks.

Advantages: Once a signature database and rule set are created, recognizing a threat can be simply implemented by pattern matching [116]. Since all signatures are created to match known malicious activities and threats, detection accuracy is assured and the number of false-positives is kept low. As a result countermeasures can be applied with confidence. Given these advantages, signature detection has been widely used in research, commercial and open source applications such as Snort [117]. Vital to their success is the establishment of a new signature immediately on the identification of a new threat.

Disadvantages: This technique, like the same antivirus approach, suffers from an inability to recognize previously unobserved attacks because of the absence of the new attack's signature [118]. In order to have the best detection ability, the database has to be updated and some users must be victims in order for a signature to be created. Along with the increased size of the signature database and the rule set, detection performance can be negatively impacted by matching against large amounts of data. In an effort to alleviate this problem, only the most common signatures may be placed on the list to keep performance efficient. In order to not decrease detection reliability, generic

signatures or wildcards can be created to detect more than one event, but at the cost of an increase in the number of false positives.

Anomaly Detection

The first anomaly detection IDS was proposed in 1985 by Dr. Dorothy E. Denning [109]. Intrusive activities are defined as anomalous activities [119]. The aim of anomaly detection is to detect anomalous activities which deviate from established accepted thresholds. Thresholds can be established based on statistical evaluations of a collection of data over a period of time.

Generally, training data is collected from a host or a network and observable events that are typically collected and measured and for example login times of a single user during an hour, a count of login failures during a minute, total time consumed by a program, network traffic. It must be ensured that no unacceptable activities occur during the sampling period (may be problematic if realistic data is to be gathered). Thresholds can be quantified as a number, a percentage or a number of standard deviations. With such systems, often anomalous and normal behaviour cannot be clearly differentiated. An appropriate threshold can reduce the number of false positives (where normal activities are classified as anomalous activities), and false negatives (where anomalous activities are classified as normal activities). See Figure 3.2.

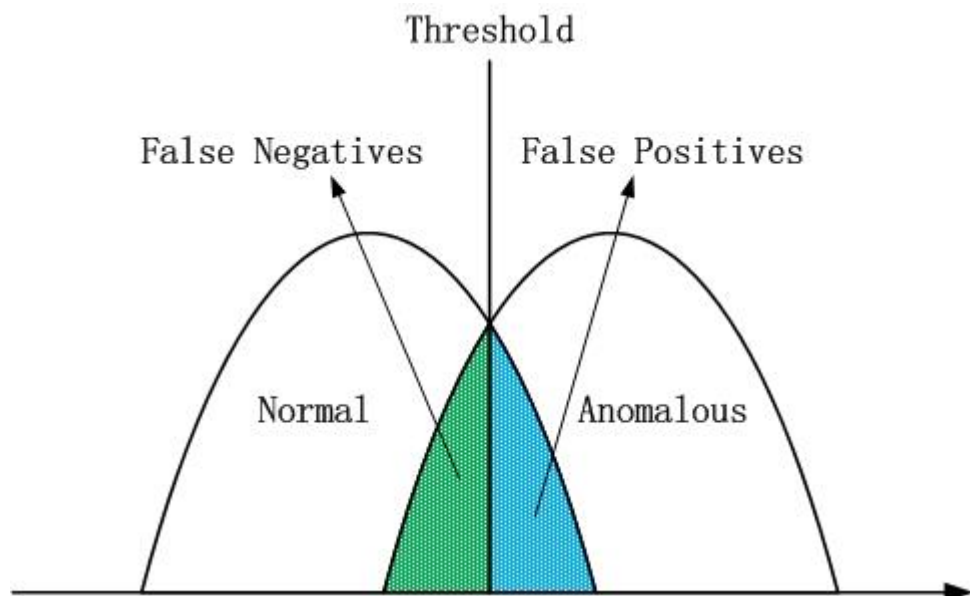


Figure 3.2 Threshold

Advantages and Disadvantages: Anomaly detection can be considered as firstly the learning of normal behaviours of systems or networks and the subsequent detection of anomalies that violate those normal behaviours. Such systems are capable of identifying a wide range of malicious events, such as an application consuming all available CPU cycles of a host, all available bandwidth on a network, or a sudden increase in the number of FTP sessions.

Unfortunately anomaly detection is not suited to adapting to benign changes in normal activities. A change of schedule in lunch breaks can trigger an alert. This problem can seriously increase the number of false positives. The repeated updating of system and network profiles helps solve the problem, but is expensive and user and network activity are changing all the time [118]. Another problem with regularly updating profiles is poisoning: by gradually augmenting attack traffic an intruder may be able to effectively retrain the system to accept attack and reject normal traffic.

3.3.4 Host-based IDS vs. Network-based IDS

Data sensors are another primary element of an IDS. Data sensor location affects monitoring scope and the range of protection offered by an IDS. Depending on the placement of data sensors, IDSs are classified into host-based IDS and network-based IDS.

Host-based IDS

A host-based intrusion detection system (HIDS) is installed on a single host and monitors its resources for any malicious activity or events [120]. The host can be a database server, an administrative workstation or any “important” network computer. Most work as a specialized security layer and require modifying the underlying operating system’s kernel. Integrated into the kernel, HIDS data sensors are the first to inspect system data minimising the risk of its modification by malware attempting to hide its tracks. Data sensors have access to resources usage patterns, system security logs, application logs, system memory, registry and file system modifications, and network traffic. The analysis engine and management console of a HIDS are commonly combined with data sensors and installed on the same host. An example of an open source HIDS is OSSEC [121]. Another example is Tripwire [122] which is a security

tool operating at the system level. It monitors the integrity of system files and reports suspicious file changes to the user.

Advantages: HIDSs have the ability to obtain high quality data from low level local activities because they operate close to the operating system kernel. Such activities include file accesses, changes to file permissions and attempts to access privileged services. Information can be collected quickly by data sensors. And if an attack is detected, a response can be rapidly initiated [123]. HIDSs are well suited to detecting attacks from inside the host rather than from the network. The installation of a HIDS is simple as there are no additional hardware requirements compared to other types of IDS, such as the dedicated servers required for network-based IDS.

Disadvantages: Since HIDSs are deployed at single hosts and work alone, they normally perform the analysis alone, there is no collaboration with other hosts, they are not able to detect attacks aimed at a number of hosts. A single installation benefits only a single host and there may be much duplication of effort across multiple installations. A HIDS often can only detect an attack once a suspicious log entry has been made, and in many cases, this means the system has already been compromised. Within a compromised system, a HIDS may be susceptible to illegal tampering [124].

Network-based IDS

A network-based intrusion detection system (NIDS) captures and analyzes network traffic at selected points on a network to protect the entire network from various threats [112]. Traffic is examined packet-by-packet in order to identify malicious or intrusive activities or events such as port scans, DoS attacks etc. NIDSs have the whole network as their protection scope and are commonly installed on dedicated machines within the network [125]. In order to access overall network traffic and detect threats in real-time or close to real-time, NIDS data sensors usually are located at the boundaries of network segments (Figure 3.3). Network devices such as routers, hubs, switches or even a network card in promiscuous mode, can work as data sensors which collect network traffic for the NIDS. A typical NIDS analyzes the network traffic locally rather than remotely due to the huge amount of network traffic to be processed. Captured traffic is analyzed by the analysis engine which inspects headers and packet data. Contrasting with HIDSs, NIDSs are designed to protect more than one host within networks while HIDSs monitor users and software activities on a single host [64][115]. There are many

network-based IDSs, including Snort [117] and Internet Security Systems' [126] RealSecure Sensor.

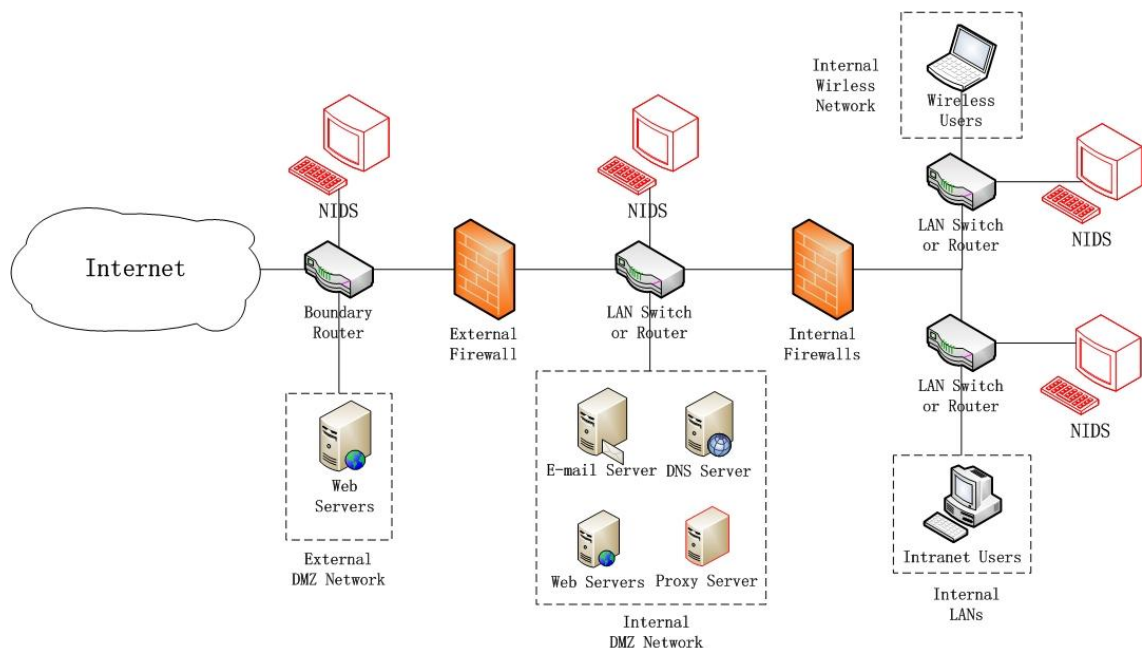


Figure 3.3 NIDS Locations

There are several locations that data sensors can be installed. Firstly, data sensors can be placed between the external firewall and the Internet to sniff all packets which target the protected network. The second location is just inside the external firewall to catch the traffic that manages to get through the firewall. Examining the traffic in this stage may also catch problems with the external firewall policy. The third location is the boundary of each internal network in order to monitor segments of the network such as separated LANs in order to detect insider attacks.

Advantages: NIDSs protect not only networks but also multiple hosts within networks [127]. Because NIDSs examine the overall network traffic, they are able to detect intrusions occurring at different hosts. NIDSs do not need to install software on every host in the network to collect data, so most NIDSs are operating system independent. NIDSs continuously inspect network traffic, record and report suspicious events to administrators for appropriate response. NIDSs are able to detect and respond to attacks before they have been successfully completed because they can be intercepted before reaching their target.

Disadvantages: When NIDSs work with heavy network traffic, they may drop packets due to performance problems. NIDSs are not able to inspect encrypted traffic [128].

Although the content of encrypted packets cannot be inspected, traffic analysis may still reveal useful information such as the volume of traffic, the source and destination ports, the source and destination IP addresses and protocols used.

3.3.5 Hybrid IDS

Both primary types of IDS have been presented in previous sections, their strengths and weaknesses have been explained. Combining both approaches into a hybrid system is an obvious idea. Many hybrid IDSs have appeared in the intrusion detection area, such as the distributed IDS [129], the autonomous agents IDS [130, 131], the mobile agent-based distributed IDS [114] and multi-sensor IDS [132].

Generally a hybrid IDS will have many host agents installed on a selection of hosts or on every host in the network. Each agent acts as a data sensor to collect and transfer audit data from its host to a central console. The audit data includes local system activities and/or network traffic and the central console will further analyze the information sent from host agents and generate alerts. The central console also provides the user interface to end users and is able to configure and control the entire system and every host agent. With a decentralized architecture, host agents coordinate their activities and exchange information with each other.

Advantages and Disadvantages: The hybrid IDS distributes the heavy network traffic processing load to agents on network hosts. Distributed intrusions can be easily detected by a hybrid IDS. However hybrid IDSs also have weaknesses [133]. A host agent on a single host may need to inspect both system activities and its own network traffic and send it to a central console or other host agents. Processing such heterogeneous data could seriously reduce the performance of both the IDS and the operating systems that have agents installed. The information sent to the central console or other host agents could increase the overall network traffic, slow network speed and consume network bandwidth. With a centralized hybrid IDS architecture, the central console can fail in attempting to deal with the mass of data sent from all host agents.

Given that the IDS that will be proposed in Chapter 4 is closest in design to that of an agent-based hybrid IDS, the general architecture of such systems is described below.

Autonomous Agents For Intrusion Detection (AAFID)

The idea of using autonomous agents for intrusion detection (AAFID) was first mentioned by Crosbie and Spafford in 1994 [134]. The AAFID architecture has since been explored in numerous studies [114][130][131]. It combines features of both host-based and network-based IDS, and both signature and anomaly detection can be used. In the approach, data collection and analysis is implemented by a set of distributed low-level agents. An agent is defined in [135] as: "...a software entity which functions continuously and autonomously in a particular environment...able to carry out activities in a flexible and intelligent manner that is responsive to changes in the environment...". Figure 3.4 shows the AAFID architecture. It has four primary components: user interface, monitors, transceivers and agents. Agents are a fundamental element in AAFID and are used to gather raw data while transceivers and monitors co-ordinate data collection and analysis.

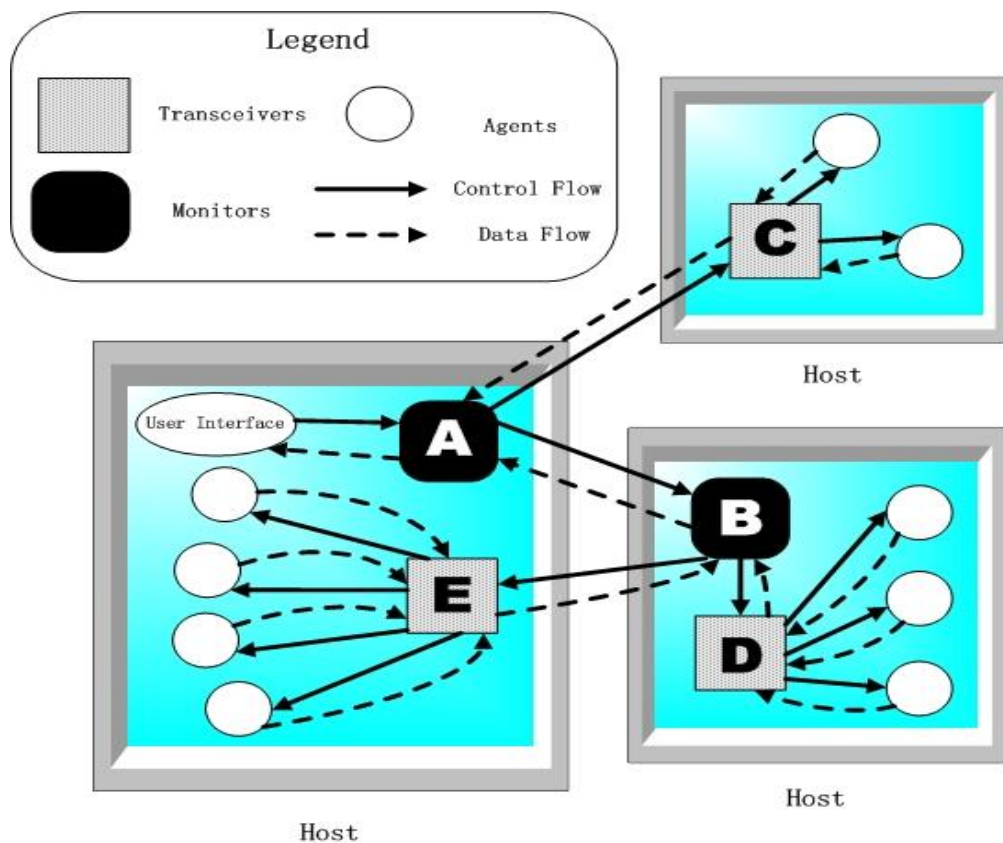


Figure 3.4 AAFID Architecture (adapted from [130])

Agents: One or more agents are installed on the monitored host and run independently, there is no communication between agent. Different agents are responsible for monitoring different host behaviours. One may inspect network events such as *telnet*

connections for example while another watches for registry changes. Once suspicious events are detected, agents submit reports to their transceiver (a transceiver must be installed on each host running an agent).

Transceivers: Transceivers control agents and receive data from them. A transceiver may carry out some processing on the data received (e.g. removing duplicate data, performing compression) before passing the data on to a monitor. Transceivers in turn are controlled by monitors.

Monitors: The highest-level components in the AAFID architecture are monitors. They manage transceivers and process data submitted by them in order to detect intrusions. A monitor can evaluate overall network health as it receives reports from a distributed set of transceivers and can detect attacks that a single transceiver might miss. For redundancy a transceiver may report to more than one monitor. Monitors may be managed by other monitors. The agent-transceiver-monitor hierarchy is illustrated in Figure 3.5.

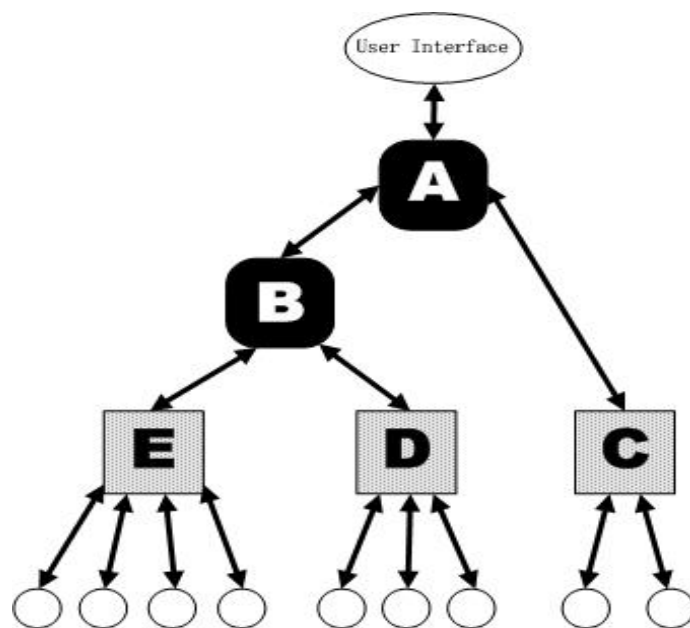


Figure 3.5 Components' hierarchy in AAFID (adapted from [130])

User Interface: The highest-level monitor has a user interface. Through this interface, users can request information from and control IDS components as well as check overall network health.

The AAFID approach has some of the advantages of both host-based and network-based IDSs. However data analysis is complicated by the fact that it comes from

different sources. In order to detect attacks, a large number of agents may be needed on each host and they may consume significant system resources.

3.3.6 IDS Today

In this section, a brief overview of a selection of popular open source and proprietary IDSs is presented.

Open Source

Snort: [13, 117] Snort is an open source network intrusion prevention and detection system originally developed by Marty Roesch in 1998. It is perhaps the most popular open source IDS in use today and is supported by an open source community who provide add-ons and detection signatures. A signature detection technique is used by Snort. It has its own rule-driven language to implement rules that detect intrusive activity. This language combines signatures, network protocols and anomalous network activities. Snort can be used as a network or host-based IDS (by being installed on a single host and inspecting its local network traffic). Snort was originally developed for UNIX, but it has been successfully ported to the MS Windows operating system and has its own management console. How Snort is deployed in practice will be considered in Chapter 4.

OSSEC (Open Source Security): OSSEC [121] is a free, open source host-based IDS. Its analysis engine can check file integrity, monitor the Windows registry, detect rootkits and perform log analysis. It includes two important components: a manager and agent. The central manager receives data from agents. Agents of OSSEC run on multiple platforms such as Linux, MacOS and Windows.

Prelude: Prelude [136] open source IDS was created in 1998 by Yoann Vandorrselaere. The Prelude IDS is a hybrid IDS that inspects both host system activities and network traffic. It standardises detected security events into an international standard format called the "Intrusion Detection Message Exchange Format" (IDMEF) [137]. There are four important parts in the Prelude IDS: sensors, management server, database, and web GUI.

Proprietary

Internet Security Systems (ISS): The Internet Security Systems [126] are a selection of network-based information security products developed by IBM. IBM ISS products include Proventia Network Intrusion Prevention System and RealSecure Server Sensor.

Cisco IDS: Cisco IDS [138] is a network-based system that combines several Cisco hardware and software products. It helps large and small corporations protect their sensitive data and internal network.

3.3.7 Summary

Intrusion detection systems aim to proactively monitor and protect hosts and networks from malicious activity. Whether a NIDS is implemented across the entire network or a HIDS is installed on a specific host, the IDS will attempt to identify suspicious or malicious events which may have bypassed the firewall or originate from internal networks. A correctly configured IDS can also take pre-defined actions to respond to the ongoing threats.

Intrusion detection applications can improve overall security, but have weaknesses. False positives are a serious problem for IDS. Non-malicious, innocent network anomalies arise and cause false alarms. If false alarms are fired with high enough frequency, real attacks can be missed and ignored. A high false positive rate also has negative effects beyond the obscuring of real attacks. Manually investigating false positives imposes a time-consuming administrative burden on the user. This problem can be reduced by fine-tuning IDSs after initial installation and keeping threat signatures up to date. However maintaining IDSs can require much effort and expertise, such as continued configuration, alert follow-up and rule and signature updates. NIDSs sometimes drop packets and miss events when they are handling large amounts of traffic. In general IDSs are complex pieces of software with numerous dependencies. This complexity can make them difficult to install, configure and use. Even though there are weaknesses in IDSs, they have a role to play in network security to protect hosts and networks and the problems of installation, configuration and feedback interpretation need solving. In the next chapter a system is proposed to reduce these problems.

3.4 Summary

Different approaches to defending against threats to computers and networks were the subject of this chapter, specifically, firewalls, antivirus software and intrusion detection systems. In particular, we centered our attentions on intrusion detection techniques. A brief history of intrusion detection was presented, highlighting the development of several types of intrusion detection techniques and systems. Different techniques including signature and anomaly detectors were described and compared. Different types of IDS including network-based, host-based and hybrid IDS were presented. Advantages and disadvantages of each type of system were compared.

In Chapter 4, and in response to the IDS issues identified above, a hybrid web-based intrusion detection application will be proposed. Details of the proposed application will be presented including its requirements, component technologies, design and implementation.

4 Intrusion Detection and Management over the WWW

In Chapters 1 and 2 it was established both that Internet user profiles have changed and that Internet threat levels are rising. In Chapter 3 the available network defences for Internet users were explored with a particular emphasis on IDS. Clearly, for users operating in an increasingly hostile Internet environment IDS could have an important role to play in their protection and in raising their threat awareness levels. However IDS has yet to be widely adopted by today's typical user. In this chapter we explore why this has been the case and in response draw up requirements for a new IDS aimed at today's user. The main part of this chapter describes the design and implementation of the latter IDS.

This chapter is structured as follows. In section 4.1 requirements for the proposed IDS are presented using the popular IDS, Snort, as a case study to highlight why IDS have not been widely adopted by typical users. With the requirements defined we specify in Section 4.2 the design and implementation of an IDS aimed at addressing those requirements. The section begins with an overview of the system followed by a more detailed description of its components. In Section 4.3 two use cases are provided to illustrate how the proposed system would be used in practice and the associated security benefits. The chapter concludes with a summary in Section 4.4.

4.1 Meeting the Typical User's IDS Needs

The popular IDS Snort, described in previous chapters, can be configured as a host-based IDS to protect a single host or deployed as a network-based IDS to monitor an entire network. Below, in order to introduce the requirements of the system proposed in this chapter we consider the problems arising when Snort is used in either of these scenarios.

Example: Snort as a host-based IDS

The Snort IDS can be installed on a single host to protect it against network threats. However the installation, configuration and maintenance of a Snort system are not straightforward. In particular the following tasks need to be carried out:

1. The PCAP packet capture library must be installed (libpcap [113] for UNIX and winpcap [139] for Windows) before Snort can be deployed.
2. A database server such as MySQL, Microsoft SQL Server or Oracle must be installed as required by Snort.
3. The database must be configured for use with Snort: the Snort user must be created, privileges assigned and tables created. This database will contain detected attacks.
4. Snort itself can now be downloaded and installed.
5. Snort must be appropriately configured through modifying the *snort.conf* file. Examples include configuring network settings, rule settings and output settings. The network settings are used to define the network interface that Snort is working on. Using rule settings the user can enable or disable rules which are used to detect the threats. The output settings allow Snort's information to be presented to users in different ways such as in a text file or in a database [77].
6. Installing a graphical user interface to the Snort output requires the downloading and installation of a web server such as Apache [117] or IIS [140], PHP [117], and the Analysis Console for Intrusion Detection (ACID) [117].
7. For effective protection it is necessary to regularly update's Snort rules and signature database.

Carrying out the above steps in addition to filtering and interpreting Snort's output are tasks outside the abilities of today's typical Internet user. Furthermore such a simple installation while allowing the detection of inbound attacks does not allow users to observe how their machine is behaving on the network and how it is viewed by other machines sharing the network. Such a view would help reveal the infection of a host with malware that initiated network traffic instead of the owner.

Example: Snort as a network-based IDS

The Snort IDS can also be used as a network IDS to protect a network. By connecting Snort to a router or a switch an entire network or subnet can be monitored. If an experienced network administrator implements the Snort installation, configuration and maintenance, the user is relieved of the technical challenges and is effectively protected.

However, under such a configuration Snort can only protect computers within the network. Given the mobility of modern users many machines will lose this defence when users work or browse the Internet from home or go on a business trip. Worse yet, users' machines while not being protected may be infected by malware before subsequently rejoining a network where only the perimeter is monitored. Also, deployed as a network IDS, Snort provides no straightforward feedback to computer network users. The fact that a machine may have been scanned and subsequently further investigated by an attacker as a potential target is not communicated its owner.

Requirements

In light of the above shortcomings we give below a list of requirements for the IDS to be described in subsequent sections. The new IDS should:

- Demonstrate to users that they are targets in order to raise their security awareness (raising threat awareness has been identified by the OECD as a key factor in fighting malware [28])
- Be web accessible so mobile users (business and recreational) are free to roam networks and receive constant feedback
- Export a web interface to ensure users receive feedback through a medium the majority of them are comfortable with: their browser
- Provide feedback to users on the behaviour of their machine (as seen by others on the network) in order to detect possible malware infection
- Be remotely administered and configured to relieve the user of technical complexity
- Offload intrusion analysis to make it suitable for deployment on devices of limited processing power
- Only increase processing in the face of a possible attack in order to limit network traffic
- Employ open source tools to avoid duplication of effort
- Be simple to install compared with other IDSs

The design and implementation of a prototype system aimed at meeting these requirements is described below.

4.2 Design and Implementation of a Hybrid WWW Intrusion Detection System

An overview of the system is provided in the section below. This is followed by a more detailed description of its design and implementation.

4.2.1 System Overview

Figure 4.1 gives an overview of the system. The approach draws on features of both host-based and network-based IDS with distributed agents reporting to a central server. The system works as follows:

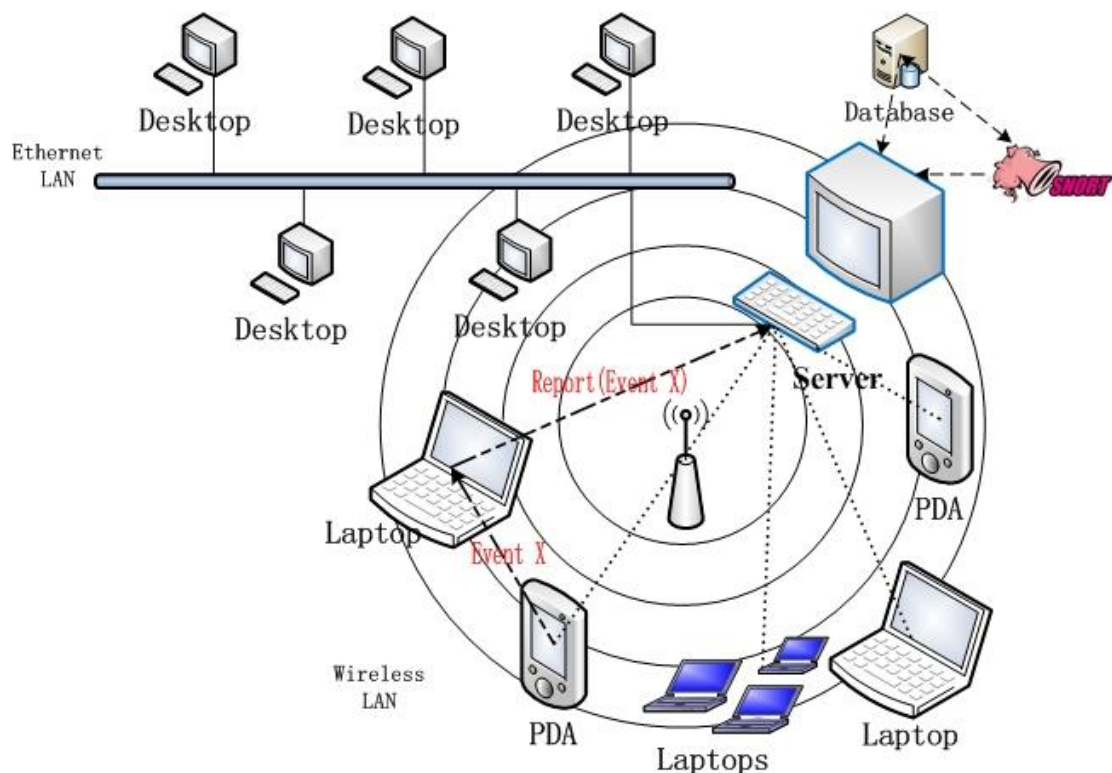


Figure 4.1 System overview (A laptop catches a suspicious event X from a PDA and reports it to the server)

- Agents are downloaded from the server and installed across a number of client machines. Those machines may be on a shared network or different networks (see Figure 4.1). In the prototype system described here, the security of the agent

is assumed. Any real implementation would, however, have to deal with various issues including client-side tampering of configuration files, secure authentication of the server, secure communication with the server, denial-of-service attacks and coding vulnerabilities (e.g. buffer overflow). Addressing these issues is covered under future work in the final chapter of the thesis.

- Each agent monitors its inbound network traffic and reports traffic deemed worthy of analysis to the central server. Only inbound traffic is ever passed to the server for analysis. Such an approach aims to reduce the performance cost of running an agent (suitability of the agent for resource-limited devices is a requirement). It is also assumed that a client infected with malware will attempt to infect other clients whose agents will submit that traffic to the server. Only considering inbound traffic thus avoids duplicate submissions to the server. (The process behind deciding whether traffic is worthy of analysis is covered in a later section.)
- Data for analysis is submitted over the WWW to the central server.
- Analysis is carried out at a central server with the aid of a local Snort installation.
- All security events are managed at the server.
- All client configuration is managed at the server.

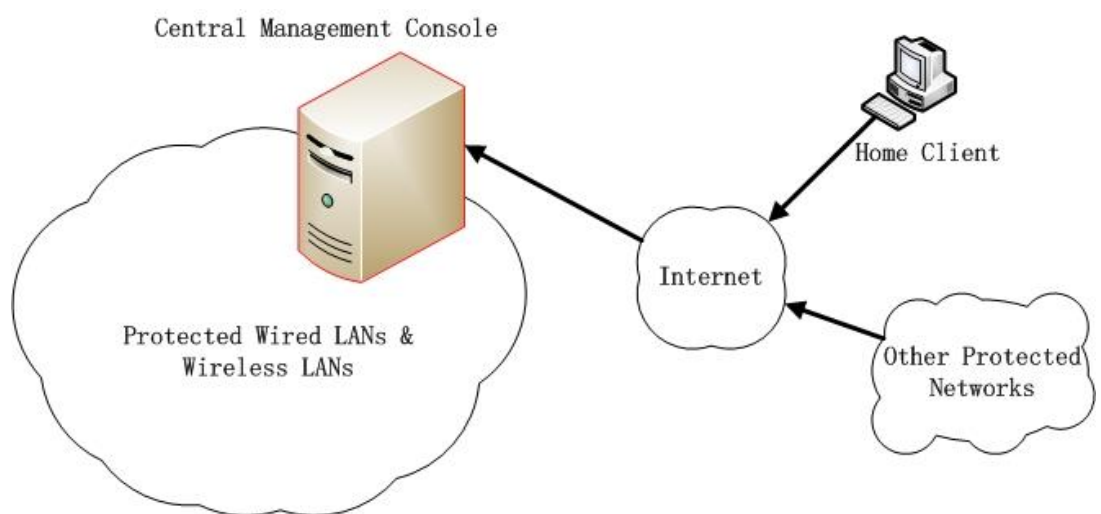


Figure 4.2 Proposed IDS over the Network

- Users (owners of client machines) can log into the server over the WWW to receive feedback on their machine. Feedback shows whether their machine has been targeted and whether their machine has been reported by other agents.
- Agents submit only minimal information until faced with a perceived threat when the amount of information submitted increases such that a more detailed analysis can be carried out by the server.
- All agents are remotely administered at the server. Network administrators use a web interface to monitor network health, supervise agents and take security-related action if appropriate. The central server also utilises the Snort IDS to provide an informed diagnosis to administrators where possible.

4.2.2 Gathering Data: Agent Design and Implementation

In this section, the design and implementation of the agent is presented in detail.

4.2.2.1 Design

The agent¹ is designed to minimise memory and CPU usage to make it suitable for running on a variety of devices from PDAs to laptops, which can take part in network communications. It offers no GUI itself but rather users connect to a web application to receive appropriate feedback. Below we describe the agent's main features.

Reporting

The main function of the agent application is to monitor its host's network traffic and report potentially security-relevant traffic to the server. The decision process involved in determining whether traffic is to be reported is depicted in Figure 4.3. Only inbound TCP, UDP and ICMP traffic is considered for reporting. These protocols account for more than 95% of Internet traffic [141]. In order to understand how the agent works we step through the decision tree below.

1. The IDS described here was developed in collaboration with a colleague [27]. Although the author did not implement the agent, he was involved in its design. It is described here for completeness. The author designed and implemented the server and all server-side processing.

1. When a packet arrives it is first compared to client rules that implement a whitelist e.g. an agent may be installed on a machine that runs a web server in which case traffic to port 80 from some set of IP addresses may be whitelisted. Any traffic that matches a whitelist entry is ignored by the agent.

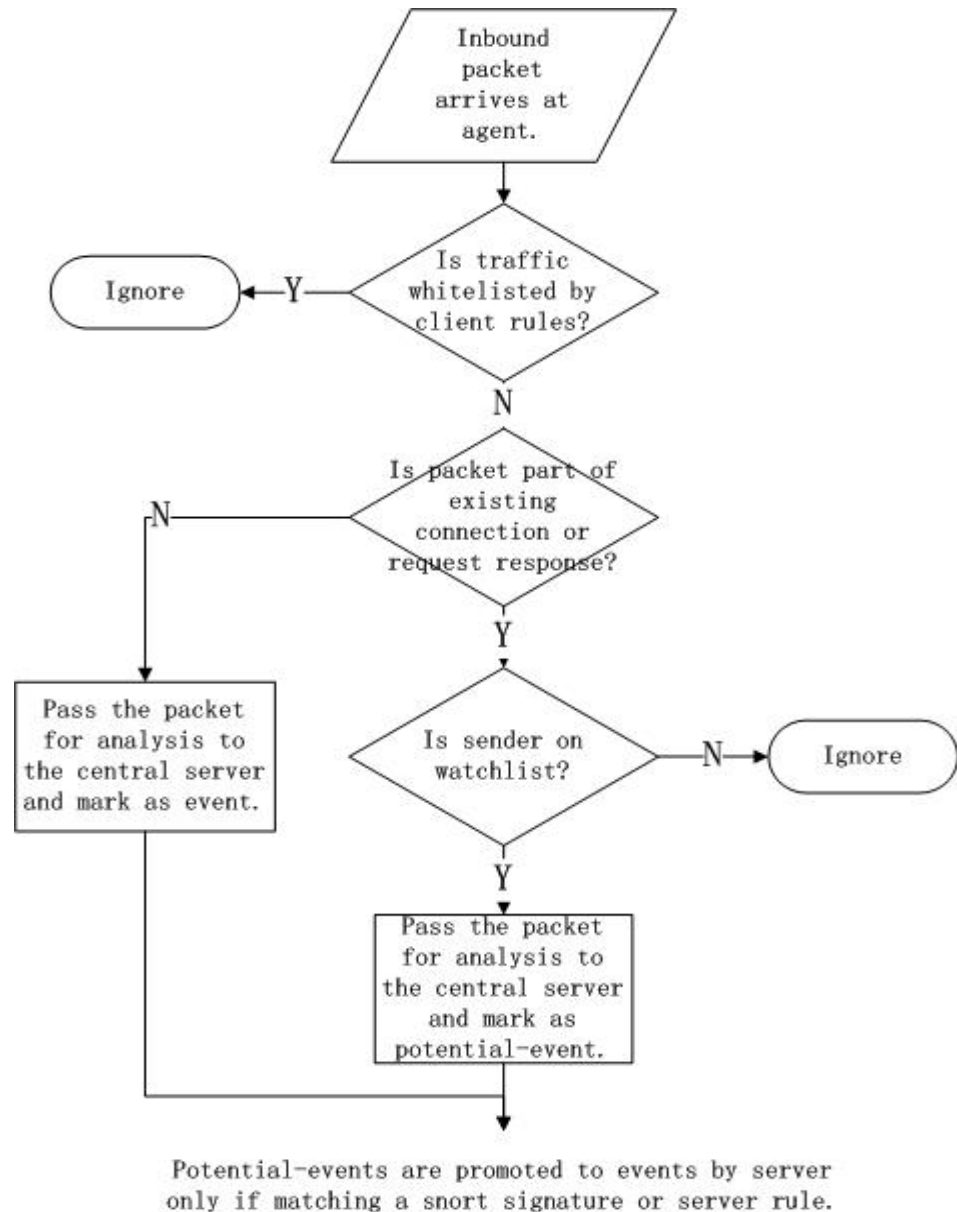


Figure 4.3 Packets Decision Tree

2. Next the packet is compared to a list of established connections and a list of expected responses for connectionless protocols and if not part of one of those connections it is deemed “unsolicited traffic” and is passed to the server for analysis. It is marked as an event for the server to indicate that it is security-relevant (all events are displayed and browsable at the server). To implement

this logic the client must implement stateful packet inspection. Thus the establishment of incoming and outgoing connections must be tracked so that subsequent traffic can be related to those connections. Note traffic that is part of an established connection may be passed to the server if the sender is on a watchlist (see below).

3. At this stage we know the packet is part of an existing connection or a response request. However, a machine may have been behaving suspiciously in the past and have been placed on a watchlist. The next question asks if the machine from which the packet was received is currently on the watchlist and thus being monitored. If so the packet is forwarded for analysis whether it belongs to an established connection or not. Note: it is not marked as an event but rather as a potential event. It becomes an event if at the server it is deemed after analysis to be security-relevant. This approach helps minimise the amount of data presented at the server.
4. If the packet is part of an established connection and not from a machine on the watchlist it is ignored.

Thus, and in summary, an agent reports two things: events and potential events. It should be stressed that potential events become events only if analysis at the server reveals them to be security relevant. This avoids complicating the server interface with too much data.

Client Rules (whitelisting)

Client rules implement a whitelist and are remotely configured at the server and drawn down by the agent. By default all agents are furnished with a blank whitelist. At the server side the whitelist can be further configured to filter out irrelevant data. Blacklisting in the face of an attack is currently not implemented and is presented as a possible extension under future work in the concluding chapter.

Connection List

In order that only unsolicited or unexpected traffic be reported as an event to the server the client must track the establishment of all existing connections (TCP) and outbound information requests (ICMP, UDP). Should incoming traffic be part of an existing

connection or be in response to a request it is processed differently. After a timeout any non-active connections are removed from the connection list.

Watchlist

The watchlist is a group of IP addresses managed by the server and drawn down by each agent. Machines on the watchlist are those that have passed some suspicion threshold in order to be placed there. For example, a machine having been reported ten times in a one minute period might be worthy of detailed monitoring. At the server such a machine would be placed on the watchlist and the updated watchlist delivered to relevant clients. Once on a watchlist all inbound traffic to *any* agent from that IP address is forwarded to the server for analysis. More details on watchlist management are presented in a subsequent section. An IP-based watchlist has its limitations and these are dealt with under future work in the concluding chapter.

Communication

In addition to network traffic various other information between agent and server are exchanged as follows:

Heartbeat

In order for the server to keep track of which clients are online and actively participating, a heartbeat message is periodically sent from each agent to the server. This heartbeat consists of a unique agent identifier. At the server side a list of currently active agents and corresponding client information is always on display.

Updates

The agent is constantly listening for updates from the server. These may be updates to its client rules, watchlist or configuration (e.g. heartbeat period). When a client is first installed it has an empty ruleset. Upon the first heartbeat being received from this client the server will update it with a ruleset, watchlist and configuration. As stated earlier, secure authentication of the server by the client is, in this prototype system, simply assumed. A secure implementation would require the client authenticate the server using, for example, public key certificates before accepting any updates from it.

4.2.2.2 Implementation

In this section we provide some relevant details related to the implementation of the agent described above.

Implementation language and target operating system

The agent is an MS Windows application and is written in C#. C# was chosen as it has WinPCAP library bindings to simplify traffic analysis (see below) and being a managed language is more secure than C. The MS Windows operating system was chosen as it is the most popular desktop operating system in the world [142].

Identifying agents

Due to the Dynamic Host Configuration Protocol (DHCP), there may be no fixed IP address associated with a client and IP addresses thus cannot be used to track client machines and agents. As a result a unique identifying code is required for each agent. It is assigned by the central server when the agent is installed. The identifying code appears in every communication between each agent and the server in order that a server can relate submitted data to agents.

Catching events: WinPCAP Library

The agent must be capable of catching network traffic before it reaches applications in order to decide whether to report it to the server according to the client ruleset. The WinPCAP packet capture library [139] is ideal for this purpose. WinPCAP is an open source port of the Unix PCAP library [113] to the MS Windows operating system. Using the library, hooks can be inserted in the network stack such that packets received on the monitored network adapter are copied to a user space buffer where the PCAP application can inspect them. Filters can be created to enable the copying only of packets of interest while ignoring others. In the agent described here filters are applied such that only ICMP, UDP and TCP packets are caught. The agent catches both inbound and outbound traffic. Although outbound traffic is never forwarded to the server it must be caught in order that connections with and requests to other machines can be tracked and stateful packet filtering implemented.

Communicating events: Intrusion Detection Message Exchange Format (IDMEF)

In the proposed system, in order to ensure interoperability and facilitate integration with other IDS and security applications, and to adhere to Internet and network standards, two network protocols are used for the exchange of events. These protocols, published by the Network Working Group, are the Intrusion Detection Exchange Protocol (IDXP) [143] and the Intrusion Detection Message Exchange Format (IDMEF) [137]. They are a set of specifications that allow the transfer of intrusion detection information between the detection device (the agent) and a management station (the server). IDMEF defines the data format and structure of messages and IDXP provides exchange procedures for transferring messages in IDMEF. Both protocols employ XML (eXtensible Markup Language) to standardise the format of data (see Figure 4.4).

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
- <idmef:Alert>
  <idmef:Analyzer auth="ekw2t" name="URANUS" analyzerid="00121794A517" />
  <idmef:CreateTime>2008-05-21T15:26:01.123Z</idmef:CreateTime>
  <idmef:DetectTime>2008-05-21T15:26:01.383Z</idmef:DetectTime>
  <idmef:AnalyzerTime>2008-05-21T15:26:02.000Z</idmef:AnalyzerTime>
- <idmef:Source>
- <idmef:Node>
  - <idmef:Address category="ipv4-addr">
    <idmef:address>136.206.218.187</idmef:address>
  </idmef:Address>
  - <idmef:Address category="mac">
    <idmef:address>00121794A517</idmef:address>
  </idmef:Address>
  - <idmef:Service>
    <idmef:Port>12</idmef:Port>
  </idmef:Service>
</idmef:Node>
</idmef:Source>
- <idmef:Target interface="1">
- <idmef:Node>
  - <idmef:Address category="ipv4-addr">
    <idmef:address>136.206.218.119</idmef:address>
  </idmef:Address>
  - <idmef:Address category="mac">
    <idmef:address>0015DFE6CB32</idmef:address>
  </idmef:Address>
  - <idmef:Service>
    <idmef:Port>45</idmef:Port>
  </idmef:Service>
</idmef:Node>
</idmef:Target>
<idmef:PacketData protocol="TCP" TTL="125" direction="in" payloadLen="512" RS2="false" RS1="false" URG="false" ACK="true" PSH="false" RST="false" SYN="true"
  FIN="false" seqNo="12412414145" ackNo="86575675446" DF="false" TOS="1" sessionId="858564633" winSize="645645634" bytesInPacket="45234"
  tcpOptions="21" arpSenderAdd="" arpSenderMac="" arpTargetAdd="" arpTargetMac="" arpOp="" />
  <idmef:type event="true" />
</idmef:Alert>
</idmef:IDMEF-Message>
```

Figure 4.4 An Example of an Event in IDMEF

All other data exchanges in the proposed system, specifically those not defined by these protocols (heartbeat, client rules, and client configuration), are also formatted and transmitted using XML. The flexibility of XML boosts interoperability and allows for simple integration with databases on the server (see Figure 4.5).

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
- <idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
- <idmef:Heartbeat>
  <idmef:Analyzer auth="4fe3z" />
- <idmef:node>
  - <idmef:Address category="ipv4-addr">
    <idmef:address>136.206.218.45</idmef:address>
  </idmef:Address>
  - <idmef:Address category="mac">
    <idmef:address>001cbf329084</idmef:address>
  </idmef:Address>
  </idmef:node>
</idmef:Heartbeat>
</idmef:IDMEF-Message>

```



Figure 4.5 Heartbeat XML Message

Avoiding firewalls: WWW and HTTP

A requirement is that the system works with mobile users who are free to roam networks and the Internet and receive constant feedback. Although users are mobile they are typically connected to the WWW and Internet and it makes sense that communication between agents and server occur over the WWW. Thus a user behind a firewall that only allows web traffic in and out will still be able to communicate with the server and no new ports in the firewall need be opened. Thus the server is implemented as a web application and deployed on a web server and agents communicate with it over Hypertext Transfer Protocol (HTTP) [144].

The HTTP protocol is a request/response application-level protocol between a client and server. Typically a HTTP client such as a web browser initiates a request to a web server where a deployed web application is listening on port 80 for the request message from the HTTP client. Both request and response message bodies contain the required content such as a text message or, as is the case, here an XML formatted message. When the agent needs to communicate with the server, it initiates a HTTP request and waits for the response.

The agent must also be ready to accept messages from the server in the form of updates. A HTTP listener is set up in the agent to listen for the HTTP requests from the web server. When the web server needs to communicate to agents, it initiates a request to an agent's listener and waits for the response. The required information is exchanged within the request/response message. (Note this approach causes problems when ports are blocked by a client firewall and when DHCP means client IP addresses may change

on the fly. Resolving these issues through having the agent poll the server for updates is covered in the final chapter of the thesis under future work.)

Because the exchanged information may contain sensitive data, the exchange ought to occur over HTTPS (Hypertext Transfer Protocol Secure) [145]. HTTPS is a combination of the HTTP protocol and a cryptographic protocol, it encrypts HTTP messages prior to transmission and decrypts messages upon arrival. The incorporation of HTTPS is considered again under future work. Most web servers including IIS support HTTPS.

4.2.3 Managing Clients and Events: Server Design and Implementation

In this section, the design and implementation of the server is presented. An overview of the server is depicted in figure 4.6.

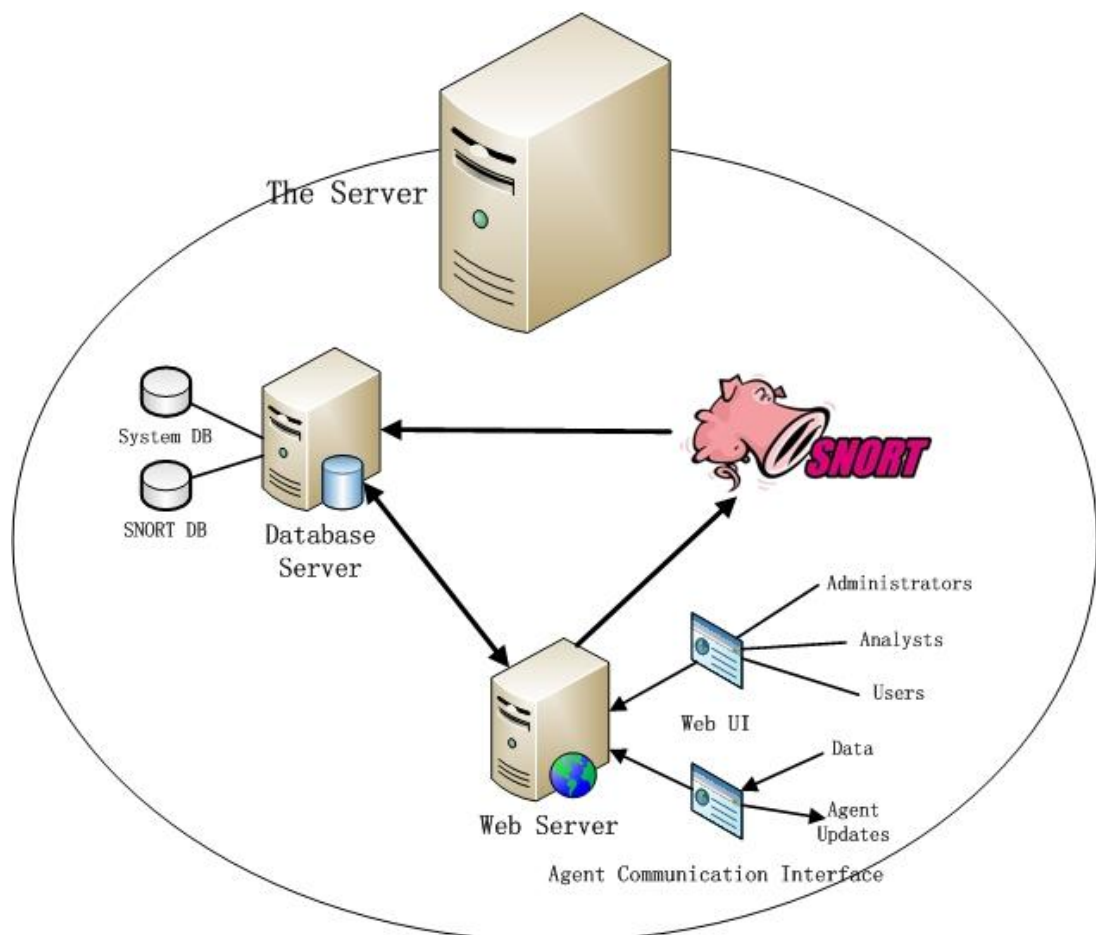


Figure 4.6 System Architecture

4.2.3.1 Design

An overview of the server is presented in Figure 4.6. The server combines several components including a web server, a database server, and a server-side Snort IDS installation. One important role of the server is to analyse data submitted by agents and present it to both administrators and end users. One interface, the agent communication interface, receives data from distributed agents and uses Snort to analyse that data before inserting it in a database. A second web user interface allows administrators and users to view events. This second interface also allows administrators to manage clients, configure the overall system and respond to events.

Server Functions

The server is designed around the functions it is required to support and an overview of the server's main functions is presented in Figure 4.7. In the following sections each of these major functions is described. Where applicable each function will be related to the requirement it is designed to meet.

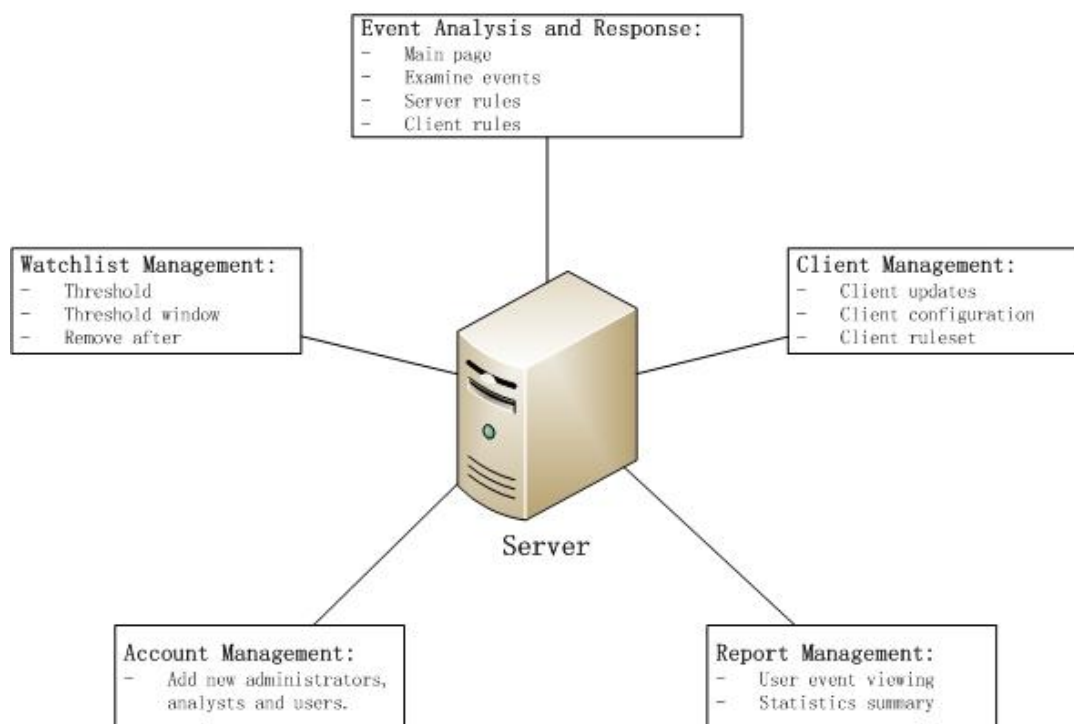


Figure 4.7 Server Functions

- **Event analysis and response**

Analysis:

Data submitted from clients is inserted in a database and analysed by Snort. Data which is not marked as an event and does not match a Snort rule or server rule is discarded. Events remain in the database and are displayed for viewing. Submitted events are cross-referenced with currently online agents and if an online agent is the subject of a report that fact is noted and the event report is associated with that agent. In this way, users running an agent can view how their machine is behaving irrespective of its changing IP address. A GEOIP database [146] is consulted and each IP address is associated with a location where possible. When an event is displayed along with it is the corresponding date and time, source IP, source name (and geographic location), destination IP, destination name, network protocol and Snort or server rule mark up. If the event matches no Snort rule or server rule then a default message “Unsolicited Traffic” is displayed.

Main page administrator view:

Given the latest events are of interest to administrators and users alike, the server main page makes that information immediately available. The main page is the web page presented immediately after the logging in to the server. A screenshot is presented in Figure 4.8. This page is the primary page of the web UI and displays various information including the function navigation menu, system state summary information, the active client table and the latest events table. For administrators all events irrespective of origin or destination are displayed.

DCU Intrusion Detection System Management Console									
Examine Events System Management Client Management Report Management Account Management Log Out ee, Administrator Current Users: 1 Events Today: 31 Total Events: 30515 Active Clients: 7	Latest Events								Active Clients
	Date	Time	From	Name	To	Name	Protocol	Detection	
	08-11-2009	16:17:28	95.24.183.74 : 1289	95-24-183-74.broadband.corbina.ru	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32.slammer propagation.]	1125-56, 136.206.18.95
	08-11-2009	16:05:26	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	1125-55, 136.206.18.94
	08-11-2009	14:35:57	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 1537	1125-56	tcp	unsolicited traffic!	1125-54, 136.206.18.93
	08-11-2009	14:26:35	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2049	1125-56	tcp	unsolicited traffic!	1125-52, 136.206.18.91
	08-11-2009	14:20:16	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	1125-51, 136.206.18.90
	08-11-2009	14:17:22	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 1569	1125-56	tcp	unsolicited traffic!	1125-50, 136.206.18.89
	08-11-2009	14:15:56	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	1125-49, 136.206.18.88
	08-11-2009	13:12:29	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	
	08-11-2009	12:42:38	93.67.203.11 : 1074	net-93-67-203-11.cust.dsl.vodafone.it	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32.slammer propagation.]	
	08-11-2009	12:37:59	61.143.134.85 : 1053	china (cn)	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32.slammer propagation.]	

Figure 4.8 Main Page – Administrator View

Different server functions can be accessed from the function navigation menu. The system state summary information shows the user name associated with the currently logged in user, their user level, the number of users that are currently using the system, a count of today's events, the number of total events in the database and the number of active clients.

DCU Intrusion Detection System Management Console									
Examine Events System Management Client Management Report Management Account Management Log Out L125-56, User	Latest Events								Active Clients
	Date	Time	From	Name	To	Name	Protocol	Detection	
	08-11-2009	16:17:28	95.24.183.74 : 1289	95-24-183-74.broadband.corbina.ru	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32.slammer propagation.]	
	08-11-2009	16:05:26	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	
	08-11-2009	14:35:57	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 1537	1125-56	tcp	unsolicited traffic!	
	08-11-2009	14:26:35	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2049	1125-56	tcp	unsolicited traffic!	
	08-11-2009	14:20:16	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic!	
	08-11-2009	14:17:22	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 1569	1125-56	tcp	unsolicited traffic!	

Figure 4.9 Main Page – User View

Main page user view:

Users (owners of client machines) can also log into the system to view events reported by their machine and reports of their machine. In this way they use a web interface to check how their machine is behaving on the network and check whether it has been

subjected to any attacks. An example screenshot of this perspective is given in Figure 4.9.

Examine Events:

The server provides a web page for administrative users to examine in detail events received from clients. An example is provided in Figure 4.10. From each event a new client rule or server rule may be generated. A client rule might be to whitelist traffic in order that it is not reported again. Such rules can be pushed out to a single client or to all clients. A server rule might allow further action to be taken by the server on receipt of future reports that match this one. For example, a server rule might say that when an event has a particularly high Snort severity level associated with it that all clients should cease communication with that client effectively isolating it. We return to this issue under future work.

Examine Events

Buttons: Clean Up, Ignore All Events, Back

Client: All Clients Date: Today

● NewEvents ● New Events with Signatures

Buttons: Select All, Unselect All, Refresh, Last Page, Next Page, Ignore Selected Events, Go

Events from 1 to 20 of 20

Operations	Date	Time	From	Name	To	Name	Protocol	Detection	Details
<input type="checkbox"/>	08-11-2009	16:17:28	95.24.183.74 : 1289	95-24-183-74.broadband.corbina.ru	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32.slammer propagation.]	<input checked="" type="checkbox"/> details
<div>Client MAC: 0021703AB49A Source IP: 95.24.183.74 Source Name: Source Port: 1289 Source MAC: 00d0c0524bfc Destination IP: 136.206.18.95</div> <div>Destination Name: L125-56 Destination Port: 1434 Destination MAC: 0021703ab49a Protocol: UDP Receive: 16:17:28.750 08-11-2009 Send: 16:16:22.433 08-11-2009</div> <div>Detect: 16:16:22.433 08-11-2009 Create: 16:16:25.577 08-11-2009 TTL: 37 Direction: in Payload Length: 418 Flag: 0 - 0 - 0 - 0 - 0 - 0 - 0</div> <div>Sequence No: 0 Acknowledgement No: 0 DF: False TOS: 0 Session Id: 0 Network Interface: 1</div> <div>Window Size: 0 Packet Bytes: 376 TCP Option: - ARP Sender IP: - ARP Sender MAC: - ARP Target IP: -</div> <div>ARP Target MAC: - ARP Option: 0 Alert Message: - Sig_name: WORM - W32 Slammer Propagation Sig_class_name: bad-unknown Sig_priority: 2</div>									
<input type="checkbox"/>	08-11-2009	16:05:28	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic	<input type="checkbox"/> details
<input type="checkbox"/>	08-11-2009	14:35:57	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 1537	1125-56	tcp	unsolicited traffic	<input type="checkbox"/> details
<input type="checkbox"/>	08-11-2009	14:26:35	208.43.230.45 : 80	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2049	1125-56	tcp	unsolicited traffic	<input type="checkbox"/> details
<input type="checkbox"/>	08-11-2009	14:20:16	218.61.18.245 : 80	china (cn)	136.206.18.95 : 7686	1125-56	tcp	unsolicited traffic	<input type="checkbox"/> details

Figure 4.10 Examine Events

When an event is selected as the basis for the creation of a rule all rule fields are filled in based on the contents of the event (Figure 4.11). The administrator can then make the rule less specific by disabling certain fields. When events are displayed they are marked as matching particular server rules with an alert message displayed in addition to any other message provided by Snort.

Add Server Rule

Add Back Back To Main Page

-----Server Rule Information-----

Action Type Default		Range All Clients		Process Type Auto		Alert Message nmap XMAS SCAN		Remark 2009-10-11	
Source IP		Source Port		Source MAC		Destination IP		Destination Port	
Destination MAC		Protocol TCP		TTL 43		Direction IN		Payload Length	
Host Name		Sequence No.		Ack No.		DF UNSET		TOS	
Session Id		Network Interface		Window Size 1024		Packet Bytes		TCP Option 21	
RS2 0	RS1 0	URG 1	ACK 0	PSH 1	RST 0	SYN 0	FIN 1		

Figure 4.11 A Server Rule Based on Nmap Xmas scan

An example server rule is shown in Figure 4.11. This rule will be fired when a TCP packet header has the URG, PSH and FIN flags set indicating the reporting client has been the victim of an Xmas scan. (This particular scan would also be detected by Snort and is presented only as an example of a server rule. Server rules do not match against packet contents, only headers, and their application in the current system is limited.) An alert message might recommend for example that the client contact their network administrator as soon as possible. When a user logs in this message will be made available to them.

- **Watchlist Management**

The server supports a watchlist function that is designed to limit the amount of data passed to it for analysis. Only IP addresses that have passed some administrator defined “suspicion threshold” are placed on the watchlist. The suspicion threshold is measured in terms of events. Once a particular IP address has exceeded the threshold all client watchlists are updated and thereafter all inbound traffic between that IP and any agent is passed to the server. Not all data submitted to the server for analysis is displayed. Only the data marked as events and the data that is designated an event by Snort. Immediately the watchlist is updated it is distributed to all clients.

The watchlist-related parameters can be updated through the update watchlist configuration function (Figure 4.12). The watchlist threshold refers to the number of events attributed to a specific IP address that must be exceeded in order for it to be moved onto the watchlist. The threshold window is the period of time within which the threshold must be exceeded. This is necessary so that all machines do not ultimately end

up on the watchlist. Once on the watchlist a machine remains there until a defined period elapses during which it goes unreported. This is necessary to ensure that a machine does not remain on the watchlist indefinitely.

Update WatchList Configuration

Threshold: times *

Threshold Window: hours *

Remove After: hours *

Auto Manage WatchList: ☒

* : Mandatory Fields

Figure 4.12 Watchlist Configurations

For example the settings in Figure 4.12 correspond to a threshold of 60 suspicious events initiated by the same IP address over a period of 24 hours. If the server has not received any events initiated by this IP over the last 48 hours, this IP will be removed from the watchlist.

The watchlist can also be managed manually. Here administrators can view watchlist entries and add and remove IP addresses to/from it (Figure 4.13).

Update WatchList

Threshold Window	Remove After	Threshold	Auto Manage
24 hour/s	48 hour/s	60 times	on

☒ ALL IP ADDRESSES
 ☒ IPs ON WATCHLIST
 ☐ IPs NOT ON WATCHLIST

Watch List from 1 to 10 of 35

IP	MAC	Times	DateTime	ClientMAC	State	Operation
136.206.11.243	00d0c0524bfc	11404	01-11-2009 03:35:08.470am	-	on	<input type="checkbox"/> remove
136.206.11.61	00d0c0524bfc	7804	04-11-2009 12:24:12.280pm	-	on	<input type="checkbox"/> remove
136.206.11.208	00d0c0524bfc	2690	09-11-2009 08:20:17.017am	-	on	<input type="checkbox"/> remove
136.206.11.209	00d0c0524bfc	2011	09-11-2009 08:20:16.937am	-	on	<input type="checkbox"/> remove
136.206.11.240	00d0c0524bfc	356	08-11-2009 10:10:24.377am	-	on	<input type="checkbox"/> remove

Figure 4.13 Watchlist

The limitations of a watchlist based solely on IP addresses are understood. For example, an infected machine, by simply acquiring a new IP address from a DHCP server, may escape a watchlist. Also an innocent machine may acquire an IP address previously

associated with a misbehaving machine. Its traffic will be passed for further analysis to the server even though the machine is clean of malware.

- **Client Management**

Every client downloaded has an identifying code associated with it and the server allows the management of that client. Client management functions include add client, remove client, edit client configuration, and edit client ruleset.

Adding a client:

When a client is downloaded it is assumed that it has an identifying code inside that will be submitted with all communications with the server. To enable that client an administrator must add it to the system. This is done through the add client function. Here details of the client are filled in by the administrator. When a client is added it has a rule set and configuration must be assigned to it (see Figure 4.14). A similar window is displayed under the edit client function except with previously assigned settings already applied.

Add Client

* : Mandatory Fields

Clear OK Back

001DD9E60B39 Disabled Client

----- Client Details -----

MAC Address: 001DD9E60B39 Host Name: L125-50
Category: Desktop Group Name: Lab machine
Client RuleSet: Lab machines Client Configuration: Lab machines

----- Client RuleSet Contents -----

IP	MAC Address	Port	Protocol	Action
138.206.1.17	-	80	tcp	allow
138.206.1.20	-	80	tcp	allow

----- Client Configuration Contents -----

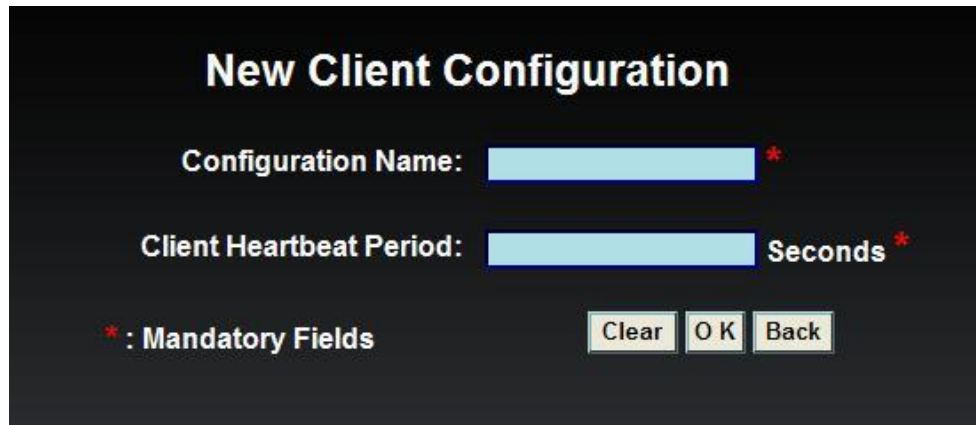
Name	Heartbeat Period(Seconds)
lab machines	20

Figure 4.14 Add Client

Configuring a client:

Client configuration management is used to manage the configuration of client agents. The configuration currently only includes configuration name and heartbeat period (Figure 4.15). However, in the future different configurations can be created depending on different client uses. A group of clients which have the same use can be set to use the

same client configuration. Once a client configuration is created it is available for assigning to clients under the add and edit client functions.

A screenshot of a web form titled "New Client Configuration". The form has a dark background with white text. It contains two input fields: "Configuration Name:" and "Client Heartbeat Period:". Both fields are followed by a red asterisk, indicating they are mandatory. The "Client Heartbeat Period:" field is followed by the word "Seconds" and another red asterisk. At the bottom left, there is a legend: "* : Mandatory Fields". At the bottom right, there are three buttons: "Clear", "O K", and "Back".

New Client Configuration

Configuration Name: *

Client Heartbeat Period: Seconds *

* : Mandatory Fields

Clear O K Back

Figure 4.15 Create Client Configuration

Configuring a client ruleset:

Client ruleset management includes a set of functions to create and edit client rulesets (Figure 4.16). The client ruleset is a set of rules located on the agent, which are used to filter the traffic and reduce the communication between agents and the server. Rules act as a whitelist effectively defining what traffic is allowed and will never be reported to the server.

A client rule consists of an IP address, MAC address, port number and protocol and action (Figure 4.16). The MAC address and IP address specify the connecting host. The port number and protocol related to the service running on the client. By default the action is set to allow. (Deny is not implemented but we return to this issue under future work.) A ruleset can be customised for a single client or created for a group of clients and used by them all. For example a group of laboratory machines would presumably share the same rules while a member of staff's laptop would have its own specific configuration.

Editing a ruleset allows the addition and deletion of rules to/from the ruleset. As stated earlier an event report can be used as the basis for creating a new rule. If a rule is added to a ruleset, all clients currently using that ruleset will be updated with the new rule. A ruleset example "Lab machine" is given in Figure 4.14.

Create Client Ruleset

RuleSet Name:

* : Mandatory Fields

Clear O K Back

IP: MAC: Port: Protocol: Action:

Add This Rule

IP	MAC Address	Port	Protocol	Action
136.206.1.17	-	80	tcp	allow

Figure 4.16 Create Client Ruleset

Queue clients' settings:

After modifications to client settings have been made, the agent communication interface will check whether these clients are reachable or not, if they are then updates are sent, if they are not then the update details are queued. Once the server receives the next heartbeat from those clients, the corresponding update will be sent.

- **Account Management**

Account management is a set of functions designed to manage different user accounts on the system. In order to use the system an account is required. There are three types of account: administrators, analysts and client users.

Client users with accounts supply a username and password to the login page and are presented with a view of all events received and generated by their machine. Their view is a passive one in that they cannot alter any settings and are the least privileged users.

Analysts have the same rights as client users. In addition they can view all events generated by each agent. Analysts can also add and configure clients and create server rules.

Administrators have the same rights as analysts. In addition they can create new accounts, as well as modify (e.g. reset password) or remove existing ones. They can also modify system settings. In short administrators have access to all system functions.

- **Supplementary Functions**

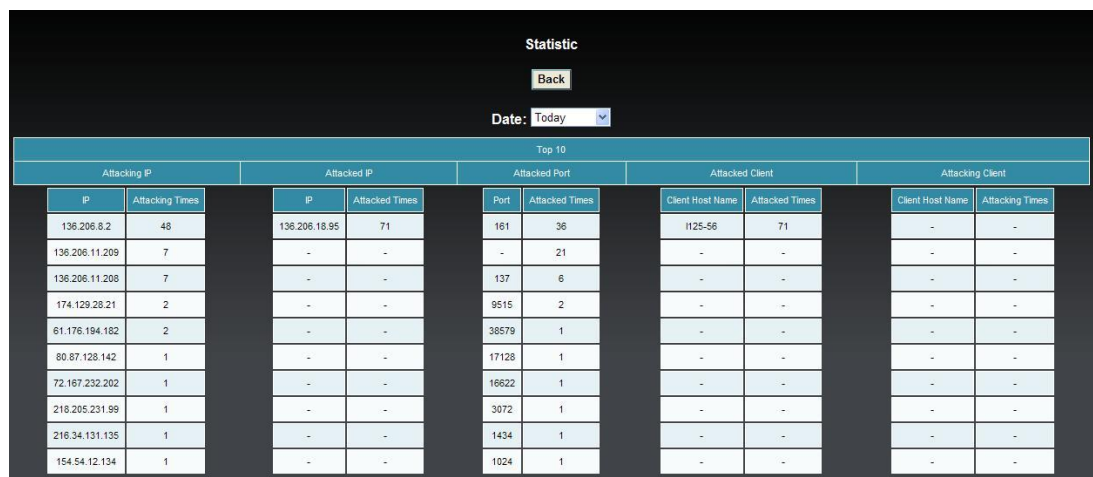
Supplementary functions allow report generation and audit trail review.

Report Generation:

Users can use this function to review summary statistics for the system as a whole. The following can be viewed (Figure 4.17):

- Most attacking IP address
- Most attacked IP address
- Most attacked port
- Most attacked client
- Most attacking client

Note that not all machines on the network will run an agent and so we distinguish between IP addresses and clients.



Statistic

Back

Date: Today

Top 10

Attacking IP		Attacked IP		Attacked Port		Attacked Client		Attacking Client	
IP	Attacking Times	IP	Attacked Times	Port	Attacked Times	Client Host Name	Attacked Times	Client Host Name	Attacking Times
136.206.8.2	48	136.206.18.95	71	161	36	1125-56	71	-	-
136.206.11.209	7	-	-	-	21	-	-	-	-
136.206.11.208	7	-	-	137	6	-	-	-	-
174.129.28.21	2	-	-	9515	2	-	-	-	-
61.176.194.182	2	-	-	38579	1	-	-	-	-
80.87.128.142	1	-	-	17128	1	-	-	-	-
72.167.232.202	1	-	-	16622	1	-	-	-	-
218.205.231.99	1	-	-	3072	1	-	-	-	-
216.34.131.135	1	-	-	1434	1	-	-	-	-
154.54.12.134	1	-	-	1024	1	-	-	-	-

Figure 4.17 Statistics

Audit trail review:

All actions carried out on the server are logged. Errors are also logged. Logged actions include logging in, logging out, adding, modifying, deleting, disabling and enabling every element of the system. Along with the action the ID of the user who carried out the action is also logged along with when it occurred and if it succeeded or failed and the error message if it failed. If problems arise, these logs may help in tracing their source.

4.2.3.2 Implementation

In this section we provide some of the more relevant details related to the implementation of the server described above.

Implementation language and target operating system

The server is a web application running on Microsoft's IIS server. It is implemented in ASP.NET and C#. Its interfaces use CSS and AJAX.

Implementing the logic: ASP.NET, C# and IIS

Active Server Pages (ASP) from Microsoft is a popular web development technology supported by Microsoft's ASP.NET framework. ASP is a free technology that is used to create dynamic web applications from small personal websites through to large enterprise-class sites. ASP supports several languages for developing extended application logic including C#, Visual Basic and C++. Here C# was used to implement the server-side logic. Microsoft also provides a web server for the ASP.NET framework, Internet Information Services (IIS).

In addition, ASP.NET also supports AJAX technology. After the term AJAX was coined in early 2005 [147], Microsoft announced "ASP.NET 2.0 AJAX". The latter is an AJAX-oriented .NET library that runs on .NET 2.0. ASP.NET AJAX can be used not only to perform ASP operations, but also supports many AJAX features at the client side.

Implementing the database: SQL Server

The database server is SQL SERVER 2005 from Microsoft and is located on the same machine as the web server. There are two databases on this server, one is the database of the proposed system, and another is the database required for Snort. The design of the database of the proposed system is shown in Figure 4.18.

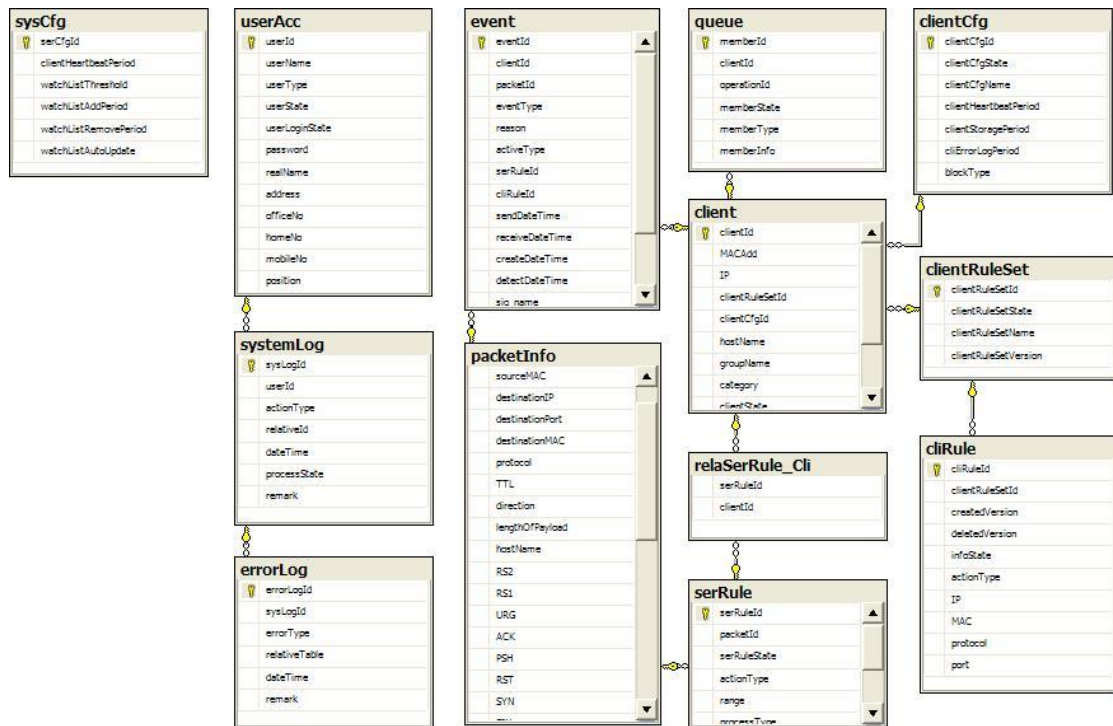


Figure 4.18 Database Tables of the System

Implementing the analysis: Snort

Snort is used by the server to help analyse the data submitted by agents. The server will employ Snort to check the original packets then import Snort feedback. Snort feedback is incorporated into the information made available to users in the server main window.

Implementing the interface: Web 2.0, AJAX and CSS

As the interface between end users and the system, the central management console plays an important role in the proposed IDS approach. It is required to display system information, latest events and active clients in real-time so that the state of the network is known and suspicious events are tracked. Client users use the console to check the behaviour of their hosts on the network. The information on display is constantly changing and so a dynamic web application rather than one based on static web content is required for the management console. Web 2.0, the second generation of web development technology provides the means to handle this content and bring the console closer to the interactive web applications users now expect.

With the advent of Web 2.0 technologies, web applications have evolved from static pages to dynamic, interactive web-based applications. The distinction between desktop

and Web applications is increasingly blurred. Dynamic Web 2.0 applications have been used in many popular websites such as Gmail, Facebook and Apple.



Figure 4.19 Web 2.0 [148]

Web 2.0 refers to web-oriented applications and services that use the Internet as a platform [149]. It is a collection of technologies (see Figure 4.19). One of its main aims is to provide a rich, responsive web application user interface. The Web 2.0 technology used in the system described here to provide an interactive interface is AJAX (Asynchronous JavaScript and XML).

AJAX applications use asynchronous interaction which is different from the traditional web application approach. With AJAX, client-side scripting is used to exchange data with a web server. Web applications can retrieve data from the server asynchronously, in the background, without interfering with the display and behaviour of the existing page. AJAX enables web pages to be updated dynamically without causing a full page refresh to occur. Web pages are more responsive exchanging small amounts of data with the server so that the entire web page does not have to be reloaded each time changes are requested.

The AJAX application's working principle is different to the classic web application (see Figure 4.20). Normally in a web application, users' actions such as clicking buttons on a form invoke HTTP requests sent back to the web server. The server processes the request, performs some calculations and database operations, and then returns back to the client a whole new page. The server and browser go through the process even if there is no change or only a small change. This behaviour sometimes causes an undesirable user experience such as losing interaction with the page or waiting a long time for a page to be updated. An AJAX-enabled application eliminates the intermittent nature of interaction through the introduction of a middle layer placed between the client and server. This middle layer is commonly called the AJAX engine and is composed of JavaScript code. The engine talks to the server on behalf of the client. It sends the HTTP request to the server and updates the contents of the page.

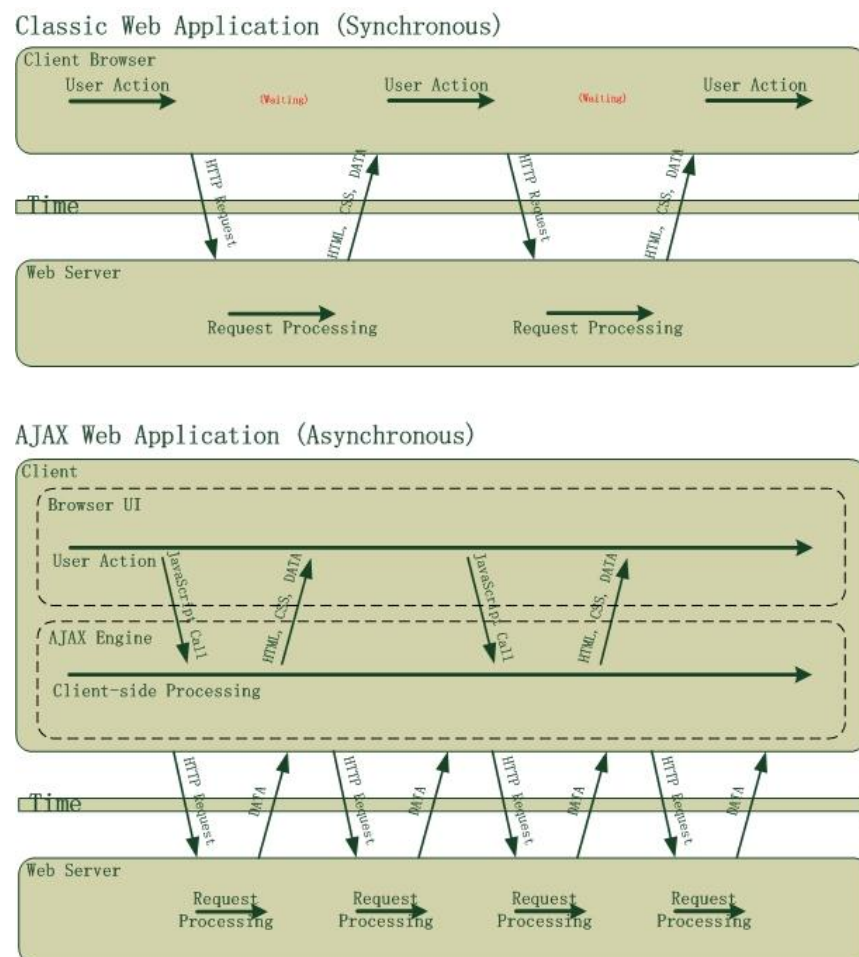


Figure 4.20 AJAX Application Working Principles (adapted from [147])

AJAX is used in the system described here. On the primary page of the central management console, the latest suspicious events sent from agents have to be displayed

in real time. The page needs to be updated when new events arrive. It is unnecessary to reload the entire page when new events come in each time, and the page only needs to be updated if there are new events. AJAX allows the page to communicate with the server asynchronously. Only if there are new events is the page updated. The update only reloads the currently viewable table. All other parts of the page are not updated and if there are no new events, then nothing is updated until the new events occur.

Cascading Style Sheets (CSS)

CSS [150] were introduced by the W3C (the World Wide Web Consortium) as a mechanism for controlling the appearance of HTML documents. CSS is a style sheet language used to describe the presentation of a document written in a markup language such as HTML, XHTML or XML. CSS is used to enable the separation of page contents on the management console from page presentation, such as colours, fonts and layout.

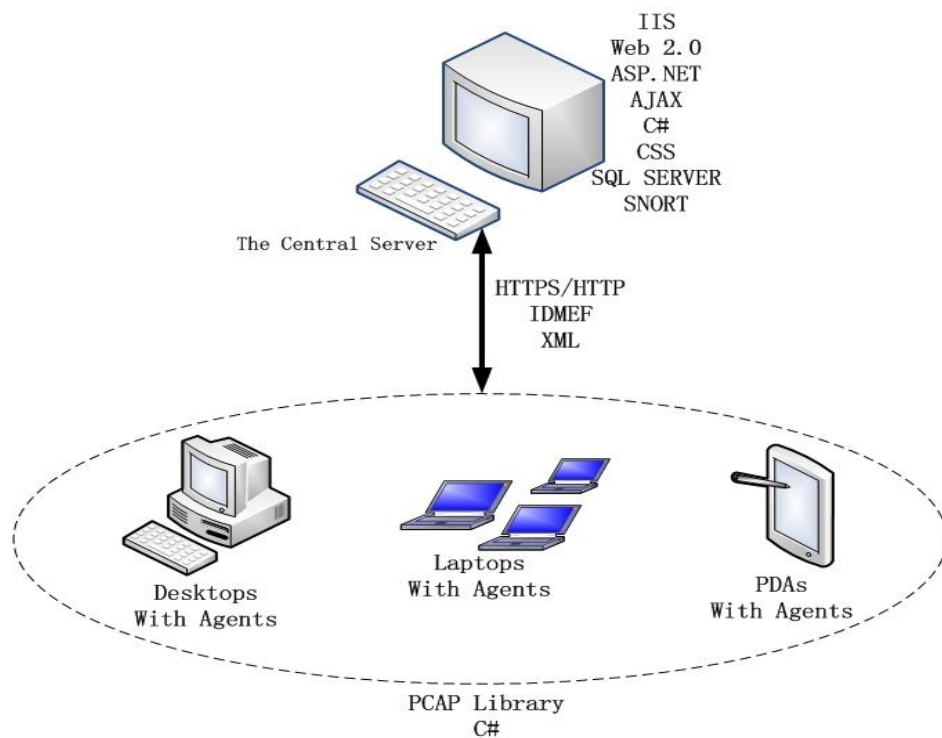


Figure 4.21 Implementation Techniques

An overview of the technologies used in the system is presented in Figures 4.21, 4.22 and 4.23. Figure 4.21 shows the technologies employed and where they fit in the implementation. Figure 4.22 shows the processing pipeline involved in moving from

raw network data observed by an agent to an alert presented in the server interface. Lastly, Figure 4.23 gives an overview of the entire system in action.

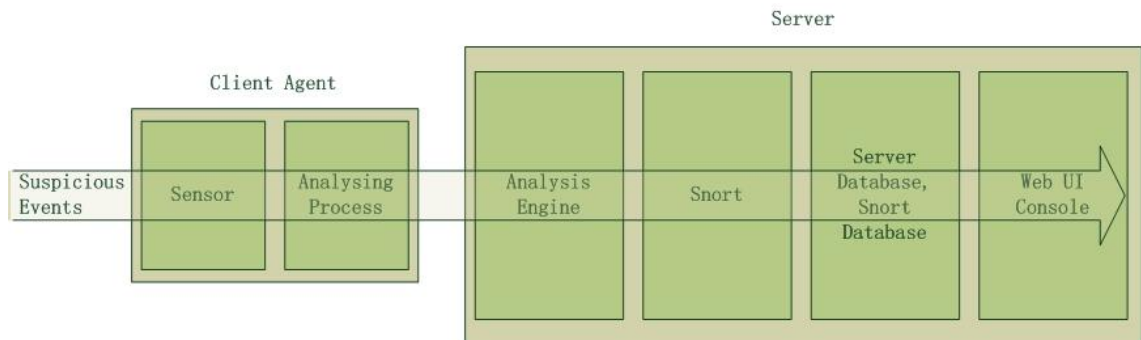


Figure 4.22 Events with Main Components

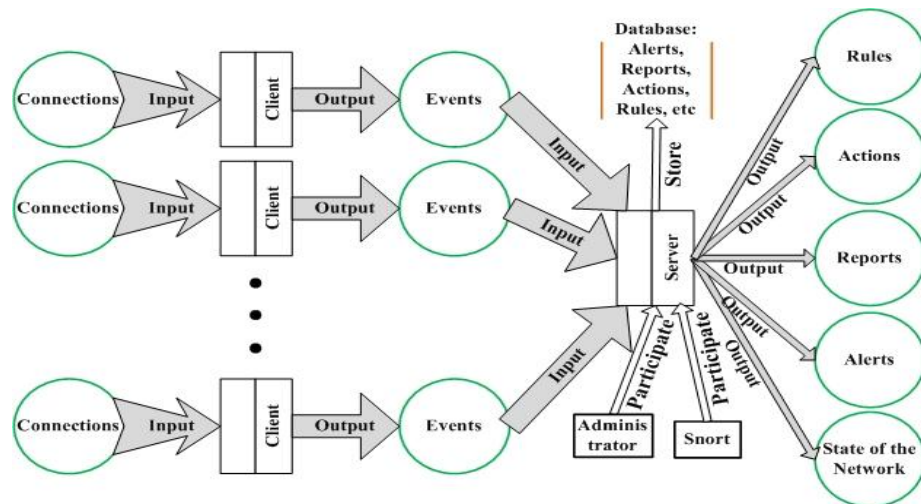


Figure 4.23 Processing Flow of the System

4.3 System Use Cases

With the description of the entire system complete we present below two use cases to illustrate the potential applications of the system. The two use cases are from the perspectives of two main user groups: network users and network administrators.

Use Case 1: The Small Business Network

As reported earlier in this thesis more and more wireless networks are being used by businesses. In some instances it can be assumed there is little network security expertise to hand. Consider a GP surgery where a number of doctors may provide services to

patients. A wireless network may be used to connect various machines together and to the Internet. In order to gain some form of protection such a group could sign up to be monitored by a security-provider. By simply installing the proposed agents on their machines they can receive feedback on the behaviour of each machine as seen by others and also whether their machines have been attacked. The security-provider receives regular updates from each machine and an expert can assess the security of the network. They can also make recommendations for improving network security in the presence of detected weaknesses. The surgery has a higher level of assurance that patient details are protected given that their security has been handed off to a knowledgeable and trusted third party. In addition the doctors themselves conveniently receive feedback on the security of their machines and networks through their browser, an interface with which they are familiar.

Use Case 2: The Corporate Network

Consider the case of a company with an internal network and in-house network security expertise. An administrator needs to monitor and enforce the security of the network. To that end firewalls, antivirus software and a proxy may be deployed. However with machines being taken often offsite for business meetings etc. a machine may become infected with malware and on rejoining the internal network bypass the deployed security measures and go on to infect other machines. Typically, once a machine is offsite it is no longer under the protection of the home network administrator and attacks against it go unobserved. Its user, being a business person, may be unaware of the security risks when operating in hotels, airports or Internet cafés.

By installing the agent on mobile machines the network administrator is able to track machines and gauge the risks to which they are exposed even while offsite. Client rulesets can be adjusted depending on their operating environment and feedback can be provided to a company's employees over the web. Problems on the home network can also be detected and network administrators have a new network health metric in the number of events reported by agents. Insider-attacks can be observed. Lastly, users knowing that events associated with their machines are logged are disinclined to launch attacks against co-workers.

4.4 Summary

Details of a hybrid IDS for use over the WWW were presented in this chapter. Starting from a list of requirements the design and implementation of a system aimed at meeting those requirements was presented. Those requirements are summarised in Table 4.1 along with a brief indicator of how they are met.

Requirements	Met by
Demonstrate to users they are targets	User interface shows attacks against a user's machine
Provide feedback to users on the behaviour of their machine	User interface show attacks launched by that user's machine
Be remotely administered and configured	Client management is handled by the server administrator
Offload intrusion analysis	Analysis is carried out at the server
Only increase traffic in face of attack	Only machines on watchlist are closely monitored
Be web accessible to mobile users	The application is implemented as a web application: events are submitted and feedback provided over the web
Export a web interface	An interface implemented with AJAX techniques provides feedback to users through their browser so no new applications need be learned
Employ open source tools	Snort is used for analysis
Be simple to install	Since analysis is done remotely, the client is simple to install

Table 4.1 Requirements Met

In the following chapter the system will be tested. First server performance will be considered. Then the system's ability to detect a range of attacks will be tested. Lastly, the results observed having deployed the system on the network of the School of Computing in DCU will be reported.

5 Results

A hybrid IDS prototype designed for deployment on the WWW was presented in the previous chapter. In this chapter we report the results of testing the proposed system in both laboratory and real world settings.

The chapter is structured as follows. In Section 5.1 we report on performance testing carried out on the system. The performance testing was carried out using Google's open source Page Speed application [151]. An overview of Page Speed and the results it produced will be presented. In Section 5.2 we present system validation results. A number of tools and security testing utilities were used to implement attacks against a given client in order to verify that the server (and its Snort installation) correctly identified and reported them to administrators and end users. In order to test the ability of the system to operate effectively "in the wild" we deployed it on the DCU School of Computing wireless and laboratory LANs. Results of this testing are reported in Section 5.3. The chapter concludes with a summary in Section 5.4.

5.1 Performance Testing

Users and administrators interact with the IDS over the WWW. To ensure end user acceptance it is important that the web application that implements the IDS be as responsive as possible. Since event analysis is carried out by Snort, and boosting its performance is beyond the scope of this project, here we concentrate on measuring the performance of the IDS web user interface as it appears in the end user's browser. A performance measuring utility produced by Google, Page Speed [151], was used to evaluate the server's web interface. Its aim is to detect and remove web page download problems so they load and display faster thus improving the user's experience of the application.

Page Speed comes as a Firefox add-on. It uses Firebug [152] to analyse the performance of the web pages delivered by a web application. It looks at a wide range of potentially performance-inhibiting issues and offers performing improving suggestions in the areas of cache optimisation, minimising round trip times, minimising request size and minimising payload size. For example, in the Figure 5.1, the suggestions "leverage browser caching" and "leverage proxy caching" are in red, which means they could be

improved. Many resources included in web pages are changed infrequently and take time to download over the network, such as JavaScript files, image files, CSS files and so on. The “optimize caching” suggests that the HTTP traffic received from the server is not maximising browser and proxy cache performance.

Page Speed measures the performance of each downloaded page and gives a corresponding score. It also provides suggestions on how the performance of the web application might be improved and draws attention to issues that may need particular attention.

Below is an example of Page Speed in action. In Figure 5.1 it is being run against our application’s examine events page. For each page a set of scores against different best practices is produced. These rules and suggestions are based on a set of commonly accepted best practices that Google and other websites implement. A green tick signifies compliance, a red circle means work required and an amber triangle indicates neutral. An overall performance summary is also provided (amber triangle in Figure 5.1).

Examine Events

Clean Up Ignore All Events Back

Client: All Clients Date: Today

● New Events ● New Events with Signatures

Select All Unselect All Refresh Last Page Next Page Ignore Selected Events Go Events from 1 to 20 of 20

Operations	Date	Time	From	Name	To	Name	Protocol	Detection	Details
	09-11-2009	08:14:10	136.206.48.107 : 35521	Ireland (ie)	136.206.18.95 : 678	I125-56	tcp	[anort: scan nmap xmas]	
	09-11-2009	08:13:30	136.206.48.107 : 51176	Ireland (ie)	136.206.18.95 : 789	I125-56	tcp	[anort: scan fin]	
	09-11-2009	06:59:05	201.242.181.99 : 1261	Venezuela (ve) , guarico	136.206.18.95 : 30607	I125-56	udp	[anort: bad traffic non-standard ip protocol]	
	09-11-2009	05:07:21	174.37.227.212 : 80	174.37.227.212-static.reverse.softlayer.com united states (us) , dallas, tx	136.206.18.95 : 1024	I125-56	tcp	unsolicited traffic!	
	09-11-2009	02:20:53	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2081	I125-56	tcp	unsolicited traffic!	

Console HTML CSS Script DOM Net Page Speed Page Speed Activity

Analyze Performance Show Resources Export Results Help

Overall performance summary:

- Leverage browser caching
- Leverage proxy caching
- Combine external CSS
- Remove unused CSS
- Enable gzip compression
- Combine external JavaScript
- Minify JavaScript
- Optimize images
- Specify image dimensions

(A) Before

Examine Events

Clean Up Ignore All Events Back

Client: All Clients Date: Today

NewEvents New Events with Signatures

Select All Unselect All Refresh Last Page Next Page Ignore Selected Events Go Events from 1 to 20 of 20

Operations	Date	Time	From	Name	To	Name	Protocol	Detection	Details
<input type="checkbox"/>	09-11-2009	08:14:10	136.206.48.107 : 35521	ireland (ie)	136.206.18.95 : 678	1125-56	tcp	[snort: scan nmap xmas]	<input type="checkbox"/> details
<input type="checkbox"/>	09-11-2009	08:13:30	136.206.48.107 : 51176	ireland (ie)	136.206.18.95 : 789	1125-56	tcp	[snort: scan fin]	<input type="checkbox"/> details
<input type="checkbox"/>	09-11-2009	06:59:05	201.242.181.99 : 1261	venezuela (ve) , guarico	136.206.18.95 : 30607	1125-56	udp	[snort: bad traffic non-standard ip protocol]	<input type="checkbox"/> details
<input type="checkbox"/>	09-11-2009	05:07:21	174.37.227.212 : 80	174.37.227.212-static.reverse.softlayer.com united states (us) , dallas, tx	136.206.18.95 : 1024	1125-56	tcp	unsolicited traffic!	<input type="checkbox"/> details
<input type="checkbox"/>	09-11-2009	02:20:53	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2081	1125-56	tcp	unsolicited traffic!	<input type="checkbox"/> details
<input type="checkbox"/>	09-11-2009	01:55:16	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 17441	1125-56	tcp	unsolicited traffic!	<input type="checkbox"/> details

Console HTML CSS Script DOM Net Page Speed Page Speed Activity

Analyze Performance Show Resources Export Results Help

Overall performance summary:

- ☐ Leverage browser caching

The following resources are missing a cache expiration. Resources that do not specify an expiration may not be cached by browsers. Specify an expiration at least one month in the future for resources that should be cached, and an expiration in the past for resources that should not be cached:

- http://136.206.48.162/NIDMSWUI/NIDMS.css
- http://136.206.48.162/NIDMSWUI/event.css
- /NIDMSWUI/images/ajax-loader.gif
- http://136.206.48.162/NIDMSWUI/images/bg.gif

- ☒ Leverage proxy caching
- ☒ Enable gzip compression
- ☒ Combine external JavaScript
- ☒ Minify JavaScript
- ☒ Optimize images
- ☒ Specify image dimensions
- ☒ Remove unused CSS
- ☒ Minimize cookie size
- ☒ Minimize DNS lookups

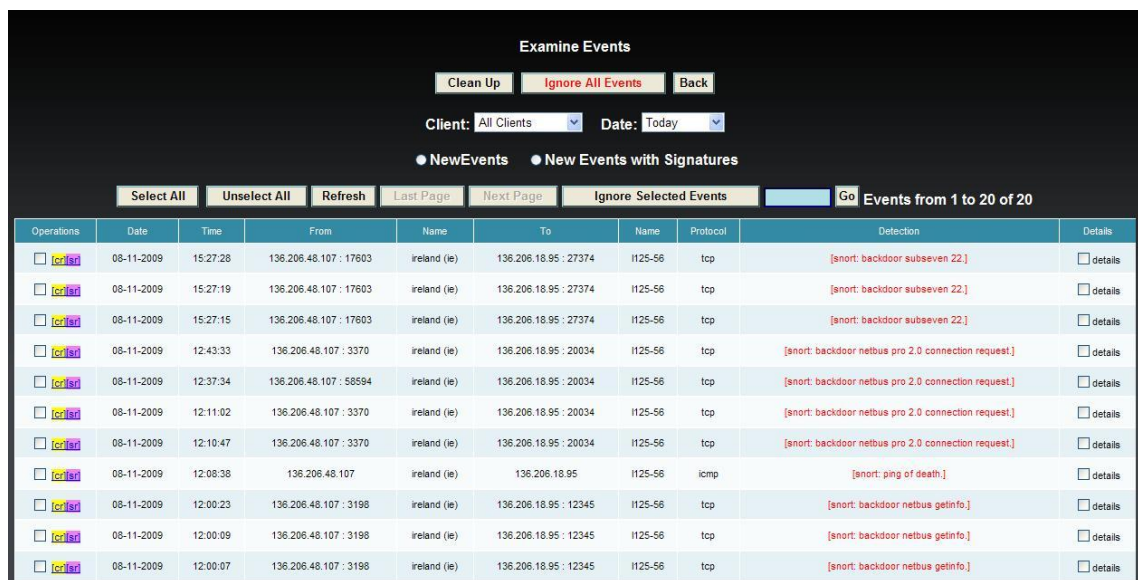
(B) After

Figure 5.1 Performance Evaluation with Page Speed

Based on the feedback provided by Page Speed several modifications were made to the web application. One particular improvement involved implementing the suggestion to “Remove unused CSS”. Before a browser can begin to render a web page, all style sheets associated with the page must be loaded. The CSS engine on the browser will evaluate every rule contained in the file to determine whether the rule applies to elements of the current page. Commonly, a single, large CSS file could be reused by many web pages in a web application, even if many of the rules defined in it do not apply to the current page. The browser has to download all rules including unused rules and parse them. As a result it is possible that delivered with pages is a large amount of CSS that is not required. The result is a delay in web page display times. Page Speed gives several recommendations on how to solve the problem e.g. removal of unused CSS rules and the splitting of a large CSS file into a number of files containing rules to be applied to specific pages. The latter approach was applied to our web application to improve its performance.

5.2 System Validation

As a network intrusion detection system, the attack detection ability of the prototype system is of vital importance and the effectiveness of the IDS hinges on this ability. Given the system is built on the Snort IDS we are not testing the ability of Snort to detect attacks as this is taken for granted but rather validating that the combination of Snort with our web-based data submission approach functions correctly and successfully identifies network attacks and anomalous traffic. The Snort rule database contains more than 3,100 default signatures and rules separated in more than 40 rules files. Therefore it is not possible or feasible to test against each one. Instead we test against a subset and in this section report the results.














Operations	Date	Time	From	Name	To	Name	Protocol	Detection	Details
<input type="checkbox"/> 	08-11-2009	15:27:28	136.206.48.107 : 17603	ireland (ie)	136.206.18.95 : 27374	1125-56	tcp	[snort: backdoor subseven 22.]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	15:27:19	136.206.48.107 : 17603	ireland (ie)	136.206.18.95 : 27374	1125-56	tcp	[snort: backdoor subseven 22.]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	15:27:15	136.206.48.107 : 17603	ireland (ie)	136.206.18.95 : 27374	1125-56	tcp	[snort: backdoor subseven 22.]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:43:33	136.206.48.107 : 3370	ireland (ie)	136.206.18.95 : 20034	1125-56	tcp	[snort: backdoor netbus pro 2.0 connection request]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:37:34	136.206.48.107 : 58594	ireland (ie)	136.206.18.95 : 20034	1125-56	tcp	[snort: backdoor netbus pro 2.0 connection request]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:11:02	136.206.48.107 : 3370	ireland (ie)	136.206.18.95 : 20034	1125-56	tcp	[snort: backdoor netbus pro 2.0 connection request]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:10:47	136.206.48.107 : 3370	ireland (ie)	136.206.18.95 : 20034	1125-56	tcp	[snort: backdoor netbus pro 2.0 connection request]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:08:38	136.206.48.107	ireland (ie)	136.206.18.95	1125-56	icmp	[snort: ping of death]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:00:23	136.206.48.107 : 3198	ireland (ie)	136.206.18.95 : 12345	1125-56	tcp	[snort: backdoor netbus getinfo]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:00:09	136.206.48.107 : 3198	ireland (ie)	136.206.18.95 : 12345	1125-56	tcp	[snort: backdoor netbus getinfo]	<input type="checkbox"/> details
<input type="checkbox"/> 	08-11-2009	12:00:07	136.206.48.107 : 3198	ireland (ie)	136.206.18.95 : 12345	1125-56	tcp	[snort: backdoor netbus getinfo]	<input type="checkbox"/> details

Figure 5.2 Detection of Backdoor and Ping of Death

The agent was installed on one test machine (the victim) and the IP address of another test machine (the attacker) added to its watchlist via the server. As a result the victim would forward all inbound communication from the attacker to the IDS for analysis by Snort. Below we report the results of a number of tests carried out using this method.

Ping of Death

The Ping of Death attack was described in Chapter 2. It is a type of basic DoS attack. This attack was launched from Ping Master [153] from the attacker against the victim (Figure 5.2).

Backdoors

A backdoor [154] is a common malware threat faced by today's Internet users. Once installed on a victim machine it provides a hidden means of bypassing normal authentication to obtain remote access. A typical backdoor consists of two parts – client and server. The server part is installed through exploiting some vulnerability or through social engineering and remains active listening on a specific port for the client to connect. The server attempts to remain undetected. The client part is executed by an attacker and searches the network to locate corresponding servers. Once connected to the server an attacker often has complete control over its host.

Two backdoor programs SubSeven (Sub7) [155] and NetBus [156] were installed on the victim and corresponding clients were installed on the attacker. The attacker then connected to the victim through each backdoor and issued several commands to it. At the server, all traffic from the malicious computer to the testing client was fed through Snort and correctly identified and tagged as backdoor communications. Identified threat signatures even indicate the version of the backdoor, such as “backdoor subseven 2.2”, “backdoor netbus pro 2.0 connection request”, “backdoor netbus getinfo” and so on (Figure 5.2).

Probing with Nmap

Nmap [22] is defined on its website as “A free utility for network exploration, administration, and security auditing”. It can scan a network to discover hosts and accessible services running on those hosts. Service and operating system versions may also be ascertained. *Nmap* functions can be used by hackers to locate targets through network scanning. During testing we installed *Nmap* on the attacker and used it to scan the victim in order to test our IDS's ability to detect network scans. In particular both Xmas and FIN scans were launched against the victim.

DCU Intrusion Detection System Management Console									
Latest Events									Active Clients
Examine Events	Date	Time	From	Name	To	Name	Protocol	Detection	1125-56, 136.206.18.95
System Management	09-11-2009	08:14:10	136.206.48.107:35521	ireland (ie)	136.206.18.95:678	1125-56	tcp	[snort: scan nmap xmas.]	
Client Management	09-11-2009	08:13:30	136.206.48.107:51176	ireland (ie)	136.206.18.95:789	1125-56	tcp	[snort: scan fin.]	

Figure 5.3 Scans Detection

In addition to supporting scanning *Nmap* also allows the testing for specific vulnerabilities prior to the implementation of an attack. The *Nmap* Scripting Engine (NSE) allows users to write custom *Nmap* scripts in order to extend it to take in vulnerability detection. The custom scripts for NSE are written in the Lua programming language [157]. Scripts are passed to *Nmap* and executed against the specified target. A growing set of scripts come with an *Nmap* installation of *Nmap* and some of the suitable ones were used to test our IDS.

The screenshot shows the 'Manual Events Examine' interface. At the top, there are buttons for 'Clean Up', 'Ignore All Events', and 'Back'. Below these are filters for 'Client: All Clients' and 'Date: This Month'. There are also radio buttons for 'NewEvents' and 'New Events Matched Rules'. A summary bar shows 'Select All', 'Unselect All', 'Refresh', 'Last Page', 'Next Page', 'Ignore Selected Events', '23', and 'Go Events from 441 to 460 of 521'. The main table lists events with columns: Operations, From, Name, To, Name, Protocol, Time, Date, Alert, Signature, and Details. The events are all 'unsolicited connection' alerts from 136.206.48.107 to 136.206.18.95 on 04-08-2009, with various signatures like 'scan socks proxy attempt', 'snmp request tcp', and 'netbios smb-ds winreg create tree attempt'.

Operations	From	Name	To	Name	Protocol	Time	Date	Alert	Signature	Details
<input type="checkbox"/>	136.206.48.107 : 58228	-	136.206.18.95 : 1080	1125-56	tcp	18:13:35	04-08-2009	unsolicited connection	scan socks proxy attempt	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 44318	-	136.206.18.95 : 161	1125-56	tcp	18:12:46	04-08-2009	unsolicited connection	snmp request tcp	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8740	-	136.206.18.95 : 445	1125-56	tcp	18:06:41	04-08-2009	unsolicited connection	netbios smb-ds winreg create tree attempt	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8596	-	136.206.18.95 : 445	1125-56	tcp	17:49:01	04-08-2009	unsolicited connection	netbios smb-ds admin\$ share access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8596	-	136.206.18.95 : 445	1125-56	tcp	17:49:00	04-08-2009	unsolicited connection	netbios smb-ds ipc\$ share access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8595	-	136.206.18.95 : 445	1125-56	tcp	17:48:55	04-08-2009	unsolicited connection	netbios smb-ds ipc\$ share access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8567	-	136.206.18.95 : 445	1125-56	tcp	17:47:43	04-08-2009	unsolicited connection	netbios smb-ds winreg create tree attempt	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8567	-	136.206.18.95 : 445	1125-56	tcp	17:47:42	04-08-2009	unsolicited connection	netbios smb-ds ipc\$ share access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8493	-	136.206.18.95 : 445	1125-56	tcp	17:31:58	04-08-2009	unsolicited connection	netbios smb-ds ipc\$ share access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8492	-	136.206.18.95 : 445	1125-56	tcp	17:31:54	04-08-2009	unsolicited connection	netbios smb-ds winreg create tree attempt	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 8200	-	136.206.18.95 : 80	1125-56	tcp	17:04:23	04-08-2009	unsolicited connection	web-misc robots.txt access	<input type="checkbox"/> details
<input type="checkbox"/>	136.206.48.107 : 7928	-	136.206.18.95 : 80	1125-56	tcp	16:37:09	04-08-2009	unsolicited connection	web-misc trace attempt	<input type="checkbox"/> details

Figure 5.4 NSE Scripts Detection

Many of the vulnerability-detecting scripts provided by *Nmap* target specific services and servers running on the target machine. For example, scripts are provided to detect exploitable vulnerabilities in web servers, FTP servers, database servers, *telnet* services etc. Therefore, in order to implement testing it is necessary to have some of these services available on the victim machine. As a result, it was necessary to install a web server IIS (Internet Information Server) on the victim in order to run the corresponding scripts.

Number	Signature Name of Detected Threats (Snort)	Attack Tool
1	SCAN nmap XMAS	NMAP Scan
2	SCAN FIN	NMAP Scan
3	BAD-TRAFFIC udp port 0 traffic	NMAP Scan
4	BAD-TRAFFIC tcp port 0 traffic	NMAP Scan
5	BAD TRAFFIC Non-Standard IP protocol	NMAP Scan
6	ICMP Large ICMP Packet	Ping Master
7	BACKDOOR NetBus Pro 2.0 connection request	NetBus
8	BACKDOOR NetBus Pro 2.0 connection established	NetBus
9	BACKDOOR netbus getinfo	NetBus
10	BACKDOOR subseven 22	SubSeven
11	INFO TELNET Bad Login	Telnet Command
12	NETBIOS SMB-DS ADMIN\$ share access	NMAP Scripts
13	NETBIOS SMB-DS IPC\$ share access	NMAP Scripts
14	NETBIOS SMB-DS winreg create tree attempt	NMAP Scripts
15	WEB-IIS .cnf access	NMAP Scripts
16	WEB-MISC Admin_files access	NMAP Scripts
17	WEB-MISC backup access	NMAP Scripts
18	WEB-MISC Oracle Java Process Manager access	NMAP Scripts
19	WEB-MISC TRACE attempt	NMAP Scripts
20	WEB-MISC robots.txt access	NMAP Scripts
21	SNMP request tcp	NMAP Scripts
22	SCAN SOCKS Proxy attempt	NMAP Scripts

Table 5.1 Detected Threats

During testing the *http-iis-webdav-vuln.nse* and *smb-check-vulns.nse* scripts were executed against the victim. The *smb-check-vulns.nse* checks for Windows RPC vulnerabilities, an infection by the Conficker worm [74], SMB (Server Message Block) traffic exploit and a denial of service vulnerability under Windows 2000. The *http-iis-webdav-vuln.nse* checks for specific vulnerabilities and common configurations errors in IIS, such as IIS allowing arbitrary users to search and access password-protected folders.

Results

Given in Figures 5.3 and 5.4 are some snapshots of events detected after test attacks had been implemented. As can be seen in Table 5.1 all attacks were successfully identified.

There are several factors that can affect detection results, such as the servers or services installed on the testing client. More servers and services running on the testing host may make the host easier to be targeted, so that more *Nmap* scripts can be used to test the IDS.

5.3 Live Testing

DCU operates a class B campus-wide network (136.206.0.0-136.206.255.255) that is centrally administered by the Information Systems and Services (ISS) department. In addition to providing a traditional LAN, a wireless network is also accessible through a number of access points across the University. All internal machines and servers (apart from a number of gateways) are protected by firewalls from the external network. The DCU network is part of the national HEAnet [158] network. Access to the wireless LAN is free and open. In addition to the University network a number of Schools within the University administer their own networks and have dedicated network administrators. One such school is the School of Computing which runs its own laboratory, staff and wireless networks. The wireless network is free and open and used by both students and staff. For the purposes of testing we deployed the agent on two machines, one on the School of Computing's laboratory network and another on its wireless network. Given the busy nature of both of these networks they were deemed a suitable deployment setting for testing the ability of our IDS to detect network problems and/or attacks. Both agents were left to run over a period of several weeks.

Results

- Absence of internal filtering

Analysing the events submitted by each agent showed some interesting results. First it was apparent that traffic from subnets and networks across DCU was reaching our agents. Further investigation revealed that there is little or no filtering of traffic across DCU's various subnets. Once a machine is on any DCU subnet it can send traffic to any other host across the network. Tests were conducted by sending traffic from both

wireless networks (School of Computing and DCU) to the laboratory machine. All traffic got through indicating no filtering of traffic across network boundaries even when that traffic comes from an untrusted source such as a wireless network.

- Heavy traffic on ports 137 - 139

Analysing the reported events showed regular traffic to both agents on ports 137, 138 and 139. These three ports are used by NetBIOS (Network Basic Input/Output System) services, port 137 is for NetBIOS-ns (name service), port 138 is for NetBIOS-dgm (datagram service), and port 139 is for NetBIOS-ssn (session service). The NetBIOS services are used to allow communication between applications on different hosts within a LAN. They also provide much information about the status of hosts in the network. The information could be used to map a network and attack hosts.

- Suspicious internal IP addresses

During testing, numerous unsolicited connection attempts were reported from IP addresses internal to the campus network. Over a period one particular machine on the wireless network attempted repeated connections to port 445 on the laboratory machine (see Figure 5.5). Port 445 is listed as the top attacked port in the top 10 attacked ports on the network security website Sectegritty [159]. Running on port 445 is a service called Service Message Block (SMB) used for file sharing on Windows. Once a file or folder is shared, this port will be automatically opened.

DCU Intrusion Detection System Management Console									
Latest Events									Active Clients
Examine Events	Date	Time	From	Name	To	Name	Protocol	Detection	1125-56, 136.206.18.95
System Management	09-11-2009	11:57:01	136.206.11.220 : 51910	ireland (ie) , dublin	136.206.18.95 : 111	1125-56	udp	[snort: bad traffic non-standard ip protocol]	
Client Management	09-11-2009	11:57:01	136.206.11.220 : 51911	ireland (ie) , dublin	136.206.18.95 : 111	1125-56	udp	[snort: bad traffic non-standard ip protocol]	
Report Management	09-11-2009	11:33:07	136.206.218.26 : 3411	ireland (ie) , dublin	136.206.18.95 : 445	1125-56	tcp	unsolicited traffic!	
Account Management	09-11-2009	11:33:07	136.206.218.26 : 3411	ireland (ie) , dublin	136.206.18.95 : 445	1125-56	tcp	unsolicited traffic!	

Figure 5.5 Suspicious Internal IP addresses

Other logged events revealed evidence of misconfigured software on the network. For example one machine, used to host personal staff web sites to be served over the web, at regular intervals sent UDP packets to port 111 on the lab machine (and to every lab machine). The port 111 is used by Sun's RPC (Remote Procedure Call) Portmapper

[160]. The server flagged the UDP traffic as a “bad traffic non-standard ip protocol”. Soon after being reported to network administrators in the School of Computing the traffic ceased (see figure 5.5). Its origin remains unclear.

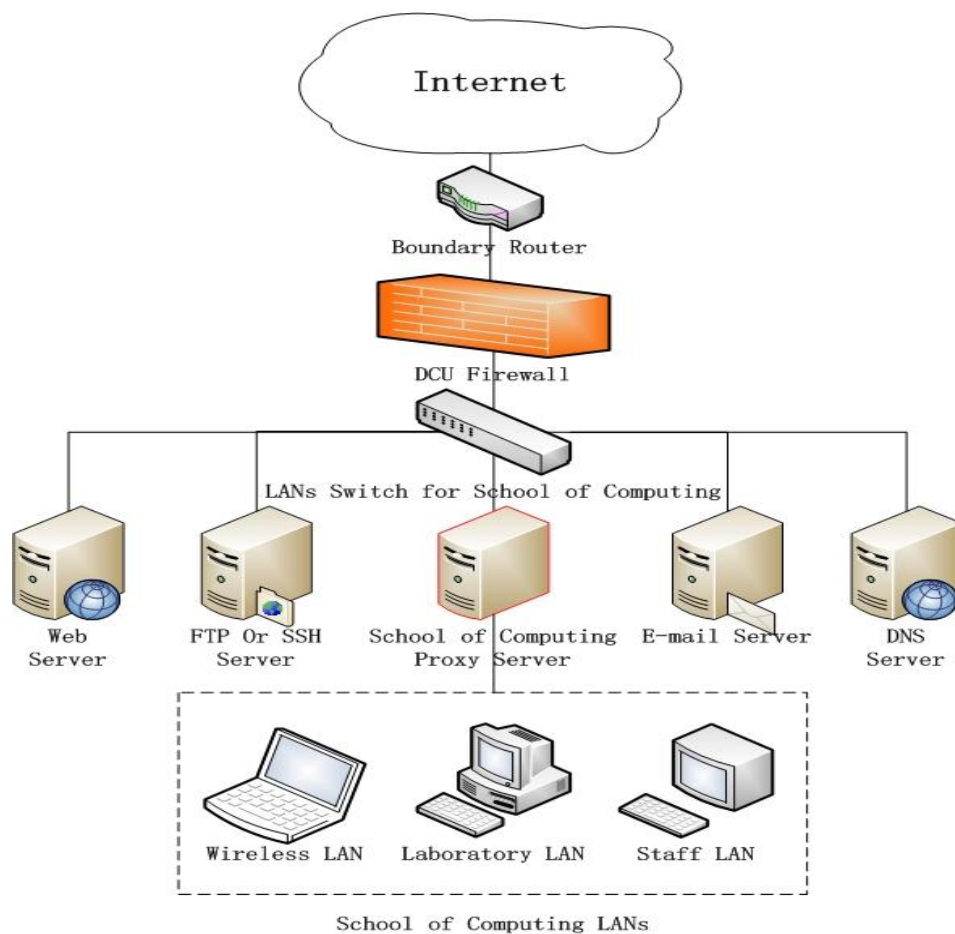


Figure 5.6 School of Computing Network Configuration

- Suspicious foreign IP address

The networks on which both agents were deployed sit behind the DCU firewall. Access to the Internet is available only through the School of Computing’s WWW proxy server. The only internal IP addresses that ought to be externally reachable belong to a number of web servers run by staff members. E-mail and DNS services are maintained by network administrators and a number of additional services (SSH, FTP etc.) are consolidated on an application gateway that is also maintained by network administrators (Figure 5.6). Thus, internal machines can access the web only through the proxy thereby hiding their IP addresses and restricting Internet traffic from laboratory machines. Should an internal user wish to connect to an external FTP server they must do so from an application gateway. Should an external user wish to FTP data

to their DCU account they can connect only to the application gateway. In summary internal IP addresses should be directly accessible only on a small number of machines and only on a restricted number of ports. Obviously, foreign IP addresses should not show up on DCU-internal network except on application gateways. Users expect a degree of protection from the Internet to be provided by the DCU firewalls.

However, not long after being deployed on internal networks foreign IP addresses began showing up in the event submissions from the two clients. Furthermore, even though the two agents were on separate subnets the same foreign IP addresses would often show up in submissions from both agents. A first assumption was that some malware on an internal network was spoofing an external IP address and sending the spoofed packet to the agent. This should not be allowed since a correctly configured router should drop externally-addressed packets arriving on an internal interface. A test was conducted and a spoofed packet (with a DCU-external IP address) was sent between two internal networks. The packet was successfully delivered to its destination. This indicates a misconfigured internal router and/or firewall. However, the number and frequency of packets on the network from foreign IP addresses and the times at which they were logged (early hours of the morning) indicated that perhaps the traffic was not all internally generated. Further testing was carried from external networks to direct packets at the internal machines hosting the agents. The IP addresses of the external host machines did indeed reach the agent. While not all packets were successfully delivered the fact that a proportion was delivered reveals a significant security flaw. This information has been passed on to network administrators and the cause is under investigation. (Note that the problem is not restricted to the School of Computing, similar foreign IP addresses showed up on the campus-wide wireless network).

DCU Intrusion Detection System Management Console									
Examine Events	Latest Events								Active Clients
	Date	Time	From	Name	To	Name	Protocol	Detection	1125-56, 136.206.18.95
System Management	09-11-2009	05:07:21	174.37.227.212 : 80	static.reverse.softlayer.com united states (us) , dallas, tx	136.206.18.95 : 1024	1125-56	tcp	unsolicited traffic!	
Client Management	09-11-2009	02:20:53	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 2081	1125-56	tcp	unsolicited traffic!	
Report Management	09-11-2009	01:55:16	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 17441	1125-56	tcp	unsolicited traffic!	
Account Management	09-11-2009	01:48:02	208.43.230.45 : 443	208.43.230.45-static.reverse.softlayer.com united states (us)	136.206.18.95 : 20001	1125-56	tcp	unsolicited traffic!	
Log Out	09-11-2009	23:17:01	61.153.247.162 : 4426	china (cn)	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32 slammer propagation]	
ee, Administrator	08-11-2009	22:27:20	218.204.137.156 : 1116	china (cn)	136.206.18.95 : 1434	1125-56	udp	[snort: worm - w32 slammer propagation]	
Current Users: 1									
Events Today: 4									

Figure 5.7 Foreign IP addresses and the Slammer worm

A sample of the foreign IP addresses (originating in China) was checked against both the Sectegritty and Dshield web sites. Four were listed in the Sectegritty Top 100 Attacking Hosts list [159] and each of these four addresses was also listed by DShield [161].

Not only were foreign packets hitting internal machines but the traffic reaching those machines was (and is) actively attacking them. UDP packets originating from external networks and submitted to port 1434 when analysed by Snort were shown to be attempted propagations by the Slammer worm [92]. Given this exploit proceeds without the establishment of a connection it is especially dangerous since should it hit a vulnerable internal machine infection is guaranteed. During the test period, more than one hundred Slammer worm attack attempts were detected by the system. Further analysis on the origin of these attacks showed 60% were from China and the other 40% from the US, Korea and Canada. The Slammer worm targets vulnerabilities in Microsoft SQL Server products. Once a host is infected by a Slammer worm, it will launch a denial-of-service attack against some Internet hosts and slow down network speed considerably.

Many suspicious source addresses had thousands of DShield reports associated with them and some were attempting to connect to port 1024 (Figure 5.7), a port that is often used by backdoor applications including NetSpy [162] and Mydoom [163]. Other popular destination ports included 10000 which hosts Webmin, a service with known vulnerabilities [164]. The server logs also revealed some foreign packets aimed at port 161, a port associated with SNMP (Simple Network Management Protocol) services. This event in itself seems suspicious, but more so given the documented insecurities in SNMP [165].

5.4 Summary

In this chapter the performance of the server in delivering web pages was measured using Google's Page Speed. Based on its recommendations, modifications were made to the server code to speed up page load times. In order to verify the system was capable of detecting and reporting attacks some malware and tools were used to attack an agent. In each case the server-side Snort installation detected, identified and reported the attack. Lastly, the system was deployed on two DCU-internal networks. There it successfully revealed several issues including the accessibility of internal networks to machines

operating from internal but untrustworthy locations (wireless networks). Reports submitted indicate the regular probing of internal hosts by other internal machines and also the presence of misconfigured software on at least one machine. Lastly and most seriously the agent revealed that internal machines are not adequately protected by firewalls and are being actively targeted for malware infection by external machines. It is our conjecture that had the system described here been deployed and used by DCU staff, made aware their machines were being targeted whilst on the DCU network the problem would have been fixed sooner. Furthermore, it would have demonstrated to users that they are targets, that assumed defences may not be protecting them and that a defence-in-depth approach is required. It is worth noting that DCU uses a third party to help maintain network security. In the next chapter we summarise the work presented in this thesis and suggest further work aimed at improving the described prototype system.

6 Conclusions and Future Work

This thesis sought to answer “Whether intrusion detection and management can be effectively offered as a web service to users in order to better protect them and heighten their awareness of the Internet security threat”. In this chapter the approach adopted to answering the latter research question is reviewed in Section 6.1 and each chapter’s contribution is summarised. In Section 6.2 conclusions and the thesis’s contribution to the area of IDS are presented. The chapter concludes in Section 6.3 with suggestions for future work.

6.1 Chapter Summary

Described in this thesis was the motivation for and implementation of a web-enabled intrusion detection system. It was organised as follows:

Chapter 1: This chapter provided the problem statement and research question. The poor state of network security was highlighted and our research question was stated. Factors that contribute to weak network security (increased mobility, lack of security awareness and a rising threat level) were presented. The security of the networked computers of non-technical users was the primary consideration of this research. A review of the requirements of individual users, small business networks and system administrators was presented.

Chapter 2: If there were no threat, no defence would be required. The aim of this chapter was to provide evidence that the threat is real and examples of those threats. In order to understand the threats to which users are subjected some network history is required. An explanation of network communication was presented including network communication layers and network protocols. Internet threats and attacks were investigated. Threats and attacks were separated into different categories, were analysed and examples were given.

Chapter 3: The threat having been established in Chapter 2, in this chapter the available defences were reviewed. Defences including firewalls, antivirus software and intrusion detection systems were covered. Because the focus of this research is on IDS, an emphasis was placed on those defences and how they are implemented.

Chapter 4: A set of requirements were presented for a new IDS to respond to the threats described in Chapter 2 and in the face of the shortcomings of the defences described in Chapter 3. The design and implementation of an IDS aimed at meeting these requirements was described in detail. A hybrid, distributed agent-based intrusion detection approach was proposed. Example use cases were presented.

Chapter 5: This chapter presented the results of testing of the system described in Chapter 4. Performance testing was carried out and some modifications made to the system based on the results. Various utilities were used to implement attacks against a test installation and attack detection and reporting was verified. Lastly the system was deployed on two networks in the School of Computing at DCU. The usefulness of the system was demonstrated in its ability to detect both misconfigured machines and network security problems.

6.2 Conclusions

The research presented in this thesis has demonstrated the feasibility and value of providing web-enabled IDS feedback to users faced with rising Internet threat levels. The primary contribution of this research is the development of a prototype implementation that has already proven its usefulness in detecting both real world attacks and network security weaknesses. It was not the aim of the work to develop the means of detecting new attacks but rather to explore the means of bringing expertise to users and raising their threat awareness levels. The prototype reported here meets both of those key requirements.

6.3 Future Work

The proposed IDS, as implemented, is a proof-of-concept and is not without limitations. Some of those shortcomings are described below along with proposed solutions:

Agent

- **Spoofing of events** is a problem for most IDSs and the one described here is no exception. If a reported event is attributed to a client, the server should be able to validate whether that event was genuinely initiated by that client, in order to avoid making incorrect decisions. Because the source IP address is easy to spoof,

some malicious computer could simulate a client by using a spoofed IP address to implement attacks. Spoofing makes it difficult to implement countermeasures.

- Currently the client user must seek feedback by logging into the application. Having feedback provided through a **browser plugin** would allow for more interaction with the server through pop up messages presenting reports as they come in.
- **Dynamic rulesets** could be applied to clients connecting from different network environments. For example, when a client is in its home network, which is presumably secure and protected by firewalls, a less strict configuration and ruleset could be applied. When this client is in a non-secure network such as on an airport or a café wireless network, the agent could ask the server to give it a different configuration and ruleset, relatively stricter than the configuration and ruleset used in the home network.
- There is a design flaw in the current prototype. The agent opens a port for listening for configuration updates from the server. This requires unwanted client-side firewall modification. This flaw could be avoided by the agent periodically polling for updates from the server rather than maintaining a listener and opening a port in the client firewall.
- **Over-reporting** is an issue. When a machine attempts to connect to a filtered port it will receive no response. The protocol may be such that it will automatically retry connecting. The extra protocol-initiated attempt should not be reported as an additional event as is currently the case.
- It is also possible for the agent to report thousands of similar events. This should not be the case as it could lead to a denial-of-service through bandwidth consumption and may also aid concealment of real attacks in server-side event logs. More sophisticated client-side processing might buffer events, analyse them for similarity and submit one plus a count of duplicates.
- Agent security is an issue. Currently the agent configuration files stored on the client are stored in plaintext and are open to manipulation and to local tampering. These configuration files should be encrypted for security. Also, the agent itself and its configuration files can be deleted. Such actions should be made

privileged operations and associated with a privileged account. The agent itself may contain exploitable vulnerabilities due to insecure code. A code review and static analysis should be carried out to catch such coding errors. Also, the agent has not been tested for denial-of-service vulnerabilities. It should be stress tested in order to reveal performance bottlenecks. Public key certificates and HTTPS should in future be applied to allow an agent to securely authenticate a server and to ensure all agent-server communication is encrypted.

Watchlist

- The IP-based watchlist in the prototype has its drawbacks. Receiving a new IP address over DHCP is enough to escape from the current watchlist. Also, a DHCP address on being freed may be reassigned to a trustworthy machine not requiring monitoring. In the future the watchlist could be changed to client-based or even both.

Threat Response

- If it could be established that a client machine was infected with malware (although a solution to the spoofing problem must be found before this can be done) then black listing of an infected client could be implemented. This might involve denying Internet access to a particular agent until the owner has had the machine sanitised.

Communication

- In order to encrypt communications between agents and the server, HTTPS should be used in future. Also, data at the server should be securely stored.

Server

- It is straightforward to launch a scan against an agent and have it submit thousands of events to the server. Browsing these events server side is currently cumbersome. It should be possible to remove duplication in the presented events.
- While the interfaces to both administrators and users provided by the server are functional, no usability testing has been carried out on those interfaces. Usability

testing is a key requirement for end user acceptance and usability testing should be carried out and feedback incorporated.

References

- [1] World Internet Usage Statistics News and World Population Stats [Online].
<http://www.internetworldstats.com/stats.htm> [accessed: 23/04/2009]
- [2] Eurostat Press Office. (2008, Dec 9). Almost two thirds of enterprises in the EU27 had a website in 2008. Eurostat Press Office, [Online]. Available:
http://epp.eurostat.ec.europa.eu/cache/ITY_PUBLIC/4-09122008-AP/EN/4-09122008-AP-EN.PDF
- [3] eMarketer. European E-Commerce on Pace to Reach €323 Billion in 2011 - eMarketer [Online]. <http://www.emarketer.com/Article.aspx?R=1005231> [accessed: 09/06/2009]
- [4] Kaspersky Labs. Malware Evolution 2006: Executive Summary [Online].
http://www.kaspersky.com/malware_evolution_2006_summary [accessed: 09/06/2009]
- [5] Mitchell Baker, Dave DeWalt, Tom Ilube, Andr   Kudelski, Jonathan Zittrain. World Economic Forum. Is the Internet at Risk? - World Economic Forum [Online].
http://www.weforum.org/en/knowledge/KN_SESS_SUMM_27219?url=/en/knowledge/KN_SESS_SUMM_27219 [accessed: 25/05/2009]
- [6] Massive DDoS attacks target Estonia; Russia accused - Ars Technica [Online].
<http://arstechnica.com/security/news/2007/05/massive-ddos-attacks-target-estonia-russia-accused.ars> [accessed: 25/05/2009]
- [7] www.bebo.com [Online]. <http://www.bebo.com/> [accessed: 09/06/2009]
- [8] MySpace [Online]. <http://www.myspace.com/> [accessed: 09/06/2009]
- [9] Welcome to Facebook! | Facebook [Online]. <http://www.facebook.com/> [accessed: 09/06/2009]
- [10] Ofcom.org.uk, "Social networking - A quantitative and qualitative research report into attitudes, behaviours and use," Ofcom.org.uk, Ofcom.org.uk, Apr 02, 2008.

- [11] Saga Zone | Social Networking | Over 50's [Online]. <http://www.sagazone.co.uk/> [accessed: 09/06/2009]
- [12] OECD. (2007, Jun 15). Summary record of the APEC-OECD malware workshop. Organisation for Economic Co-operation and Development, Philippines. [online]. Available: <http://www.oecd.org/dataoecd/37/60/38738890.pdf>
- [13] Snort - the de facto standard for intrusion detection/prevention [Online]. <http://www.snort.org/> [accessed: 27/03/2009]
- [14] S. Frei, T. Duebendorfer, G. Ollmann and M. May. (2008, June). Understanding the web browser threat. TIK, ETH Zurich,
- [15] B. Brendler. (2007, Apr). Spyware/Malware impact on consumers. APECTEL 35 Conference, APEC-OECD Malware Workshop.
- [16] Bittau Andrea. (2004 August). WiFi exposed. *Crossroads 11(1)*, pp. 3-3. Available: <http://doi.acm.org/10.1145/1031859.1031862>
- [17] N. Diksha and A. Shubham. (2006, Nov). Backdoor intrusion in wireless networks-problems and solutions. *Communication Technology, 2006. ICCT '06. International Conference on* pp. 1-4.
- [18] Bittau Andrea, Handley Mark and Lackey Joshua. The final nail in WEP's coffin. Presented at SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy. Available: <http://dx.doi.org/10.1109/SP.2006.40>
- [19] Aircrack [Online]. <http://aircrack-ng.org/doku.php> [accessed: 25/05/2009]
- [20] BBC NEWS | Technology | Card details stolen in web hack [Online]. <http://news.bbc.co.uk/1/hi/technology/7446871.stm> [accessed: 24/04/2009]
- [21] Anuskiewicz Neil. (2001 May). Take command: An introduction to DNS and DNS tools. *Linux J.* pp. 9.
- [22] Nmap - Free Security Scanner For Network Exploration & Security Audits.[Online]. <http://nmap.org/> [accessed: 23/04/2009]

- [23] The Metasploit Project [Online]. <http://www.metasploit.com/> [accessed: 23/04/2009]
- [24] rootkit.com [Online]. <http://www.rootkit.com/> [accessed: 12/06/2009]
- [25] DDoS Resources [Online]. <http://anml.iu.edu/ddos/tools.html> [accessed: 23/04/2009]
- [26] GAO. (2009, Jun 22). U.S. GAO - cybercrime: Public and private entities face challenges in addressing cyber threats. United States Government Accountability Office, [online]. Available: <http://www.gao.gov/new.items/d07705.pdf>
- [27] Darren Fitzpatrick, Hai Ying Luan, Darragh O'Brien, "Agent-based network intrusion detection," in *The 8th IT&T Conference Proceeding*, 2008,
- [28] Organisation for Economic Co-operation and Development (OECD), "Malicious software (malware) - A security threat to the internet economy - ministerial background report DSTI/ICCP/REG(2007)5/FINAL," OECD Ministerial Meeting, Jun, 2008.
- [29] V. Cerf and R. Kahn. (1974 May). A protocol for packet network intercommunication. *Communications, IEEE Transactions on* 22(5), pp. 637-648.
- [30] A. S. Tanenbaum., (2003), (Mar). *Computer Networks*, (4th ed.) Available: http://isbnadb.com/d/book/computer_networks_a10
- [31] H. H. Goldstine and A. Goldstine. (1996 Dec). The electronic numerical integrator and computer (ENIAC). *Annals of the History of Computing, IEEE* 18(1), pp. 10-16.
- [32] Living Internet. Semi-Automatic Ground Environment (SAGE) [Online]. http://www.livinginternet.com/i/ii_sage.htm [accessed: 20/07/2009]
- [33] P. Baran. (1964 Mar). On distributed communications networks. *Communications Systems, IEEE Transactions on* 12(1), pp. 1-9.
- [34] B. Leiner, et al. (1985 Mar). The DARPA internet protocol suite. *Communications Magazine, IEEE* 23(3), pp. 29-34.

- [35] Clark D. The design philosophy of the DARPA internet protocols. Presented at SIGCOMM '88: Symposium Proceedings on Communications Architectures and Protocols. Available: <http://doi.acm.org/10.1145/52324.52336>
- [36] Wendy Boswell. History of the Internet - Learn More About the History of the Internet [Online]. <http://websearch.about.com/od/whatistheinternet/a/historyinternet.htm> [accessed: 07/03/2009]
- [37] H. Orman. (2003 Dec). The morris worm: A fifteen-year perspective. *Security & Privacy, IEEE* 1(5), pp. 35-43.
- [38] W. C. Chan. (1996, Dec). Globalization of internet access. *Industrial Technology, 1996. (ICIT '96), Proceedings of the IEEE International Conference on* pp. 485-488.
- [39] Stewart G.W. (1991 Oct). FTP-file transfer program. *SIGNUM Newsl.* 26(4), pp. 2-3. Available: <http://doi.acm.org/10.1145/122645.122646>
- [40] Leavell Glenn. (1995 Jun). Usenet news for electronic information sharing. *SIGUCCS Newsl.* 25(1-2), pp. 30-36. Available: <http://doi.acm.org/10.1145/1098867.1098874>
- [41] Davidson J., Hathaway W., Postel J., Mimno N., Thomas R. and Walden D. The arpanet telnet protocol: Its purpose, principles, implementation, and impact on host operating system design. Presented at SIGCOMM '77: Proceedings of the Fifth Symposium on Data Communications. Available: <http://doi.acm.org/10.1145/800103.803338>
- [42] R. Cailliau., (2000), *How the Web was Born: The Story of the World Wide Web*, Available: http://isbnadb.com/d/book/how_the_web_was_born
- [43] The Central Intelligence Agency of US. CIA - Internet hosts [Online]. <https://www.cia.gov/library/publications/the-world-factbook/rankorder/2184rank.html> [accessed: 23/04/2009]
- [44] United Nations Conference on Trade and Development., (2007), (Dec). *Information Economy Report 2007-2008*,

- [45] The Commission for Communications Regulation (ComReg). (2009, Mar 19). ComReg quarterly report shows an increase in broadband take-up. Available: <http://www.comreg.ie/fileupload/publications/PR190309.pdf>
- [46] Dell [Online]. <http://www.dell.com/> [accessed: 23/04/2009]
- [47] Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs & more [Online]. <http://www.amazon.com/> [accessed: 23/04/2009]
- [48] Wild West Domains, Resell Domains, Start your online business [Online]. <http://www.wildwestdomains.com/> [accessed: 30/04/2009]
- [49] Hosting365 [Online]. <http://www.hosting365.ie/> [accessed: 30/04/2009]
- [50] Google [Online]. <http://www.google.com/> [accessed: 23/04/2009]
- [51] United Nations Conference on Trade and Development, "E-commerce and development report 2003," United Nations Publications, New York and Geneva, Dec, 2003.
- [52] Online retail sales in the U.S. to hit \$204B in '08 [Online]. <http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9075759> [accessed: 23/04/2009]
- [53] Day John. (1995 Oct). The (un)revised OSI reference model. *SIGCOMM Comput. Commun. Rev.* 25(5), pp. 39-55. Available: <http://doi.acm.org/10.1145/216701.216704>
- [54] RFC 826 - Ethernet Address Resolution Protocol [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=826> [accessed: 27/04/2009]
- [55] RFC 791 - Internet Protocol [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=791> [accessed: 27/04/2009]
- [56] C. Popoviciu and P. Dini. (2006, Dec). IPv6 as a practical solution to network management challenges. *Computing in the Global Information Technology, 2006. ICCGI '06. International Multi-Conference on* pp. 5-5.

- [57] RFC 792 - Internet Control Message Protocol [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=792> [accessed: 27/04/2009]
- [58] RFC 793 - Transmission Control Protocol [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=793> [accessed: 27/04/2009]
- [59] RFC 768 - User Datagram Protocol [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=768> [accessed: 27/04/2009]
- [60] RFC 1591 - Domain Name System Structure and Delegation [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=1591> [accessed: 27/04/2009]
- [61] RFC 1945 - Hypertext Transfer Protocol -- HTTP/1.0 [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=1945> [accessed: 27/04/2009]
- [62] The Internet Society. RFC 2828 - Internet Security Glossary [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=2828#top> [accessed: 02/05/2009]
- [63] M. Bishop., (2003), (Jan). *Computer Security : Art and Science*,
- [64] W. Stallings and L. Brown. (2007), *Computer Security: Principles and Practice*, Available: http://isbndb.com/d/book/computer_security_a35
- [65] CERT. Denial-of-Service Attack via ping [Online]. <http://www.cert.org/advisories/CA-1996-26.html> [accessed: 05/05/2009]
- [66] Kathleen Kelly. TCP/IP Protocol Suite [Online]. <http://kkkelly.com/Wk7Cisco.htm> [accessed: 01/07/2009]
- [67] Peng Tao, Leckie Christopher and Ramamohanarao Kotagiri. (2007 Dec). Survey of network-based defense mechanisms countering the DoS and DDoS problems. *ACM Comput.Surv.* 39(1), pp. 3. Available: <http://doi.acm.org/10.1145/1216370.1216373>
- [68] Gregg Keizer. Dutch Botnet Bigger Than Expected -- Huge Botnet, cybercrime -- InformationWeek [Online]. <http://www.informationweek.com/news/security/government/showArticle.jhtml?articleID=172303265> [accessed: 12/06/2009]

[69] Computer Emergency Response Team (CERT), "2007 E-Crime watch survey," CERT® Coordination Center, Sep, 2007.

[70] Jeremy Kirk. Student fined for attack against Estonian Web site | Security Central - InfoWorld [Online]. <http://www.infoworld.com/d/security-central/student-fined-attack-against-estonian-web-site-794> [accessed: 01/07/2009]

[71] C. Smith and A. Matrawy. (2008, Jun). Comparison of operating system implementations of SYN flood defenses (cookies). *Communications, 2008 24th Biennial Symposium on* pp. 243-246.

[72] www.pcworld.com. Conficker Worm: Not Finished Yet - PC World [Online]. http://www.pcworld.com/article/162477/conficker_worm_not_finished_yet.html [accessed: 09/06/2009]

[73] Conficker Work Group - Main - HomePage [Online]. <http://www.confickerworkinggroup.org/wiki/> [accessed: 25/05/2009]

[74] Microsoft. Microsoft. Conficker - Computer Worms | Microsoft Security [Online]. <http://www.microsoft.com/protect/computer/viruses/worms/conficker.mspx> [accessed: 25/05/2009]

[75] N. Provos, D. McNamee, P. Mavrommatis, K. Wang and N. Modadugu. The ghost in the browser analysis of web-based malware. Presented at HotBots'07: Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets.

[76] Y. Shin and L. Williams. Is complexity really the enemy of software security? Presented at QoP '08: Proceedings of the 4th ACM Workshop on Quality of Protection. [Online]. Available: <http://doi.acm.org/10.1145/1456362.1456372>

[77] Apache HTTP Server Vulnerabilities [Online]. http://httpd.apache.org/security/vulnerabilities_13.html [accessed: 12/06/2009]

[78] MS RPC Server Vulnerabilities [Online]. http://www.hsc.fr/ressources/articles/win_net_srv/msrpc_vulnerabilities.html [accessed: 12/06/2009]

- [79] Top 10 2007 - OWASP [Online]. http://www.owasp.org/index.php/Top_10_2007 [accessed: 09/05/2009]
- [80] Refai Mustapha. Exploiting a buffer overflow using metasploit framework. Presented at PST '06: Proceedings of the 2006 International Conference on Privacy, Security and Trust. Available: <http://doi.acm.org/10.1145/1501434.1501521>
- [81] Berghel Hal. (2001 Dec). The code red worm. *Commun ACM* 44(12), pp. 15-19. Available: <http://doi.acm.org/10.1145/501327.501328>
- [82] K. Wei, M. Muthuprasanna and Suraj Kothari. (2006, May). Preventing SQL injection attacks in stored procedures. *Software Engineering Conference, 2006. Australian* pp. 8 pp.
- [83] Internet Crime Complaint Center (IC3) - FBI 2008 Annual Reports [Online]. <http://www.ic3.gov/media/annualreports.aspx> [accessed: 27/03/2009]
- [84] D. Ian Hopper. Destructive 'ILOVEYOU' computer virus strikes worldwide - CNN.com [Online]. <http://edition.cnn.com/2000/TECH/computing/05/04/iloveyou.01/> [accessed: 01/07/2009]
- [85] BBC News. Anna Kournikova computer virus hits hard [Online]. <http://news.bbc.co.uk/2/hi/science/nature/1167453.stm> [accessed: 01/07/2009]
- [86] Berghel Hal. (2004 Sep). Wireless infidelity I: War driving. *Commun ACM* 47(9), pp. 21-26. Available: <http://doi.acm.org/10.1145/1015864.1015879>
- [87] N. T. Anh and R. Shorey. (2005, Jan). Network sniffing tools for WLANs: Merits and limitations. *Personal Wireless Communications, 2005. ICPWC 2005. 2005 IEEE International Conference on* pp. 389-393.
- [88] Ronan Magee, "802.IDS - Wireless Intrusion Detection," *DCU*, Aug. 2007.
- [89] Spafford Eugene H. (1989 Jan). The internet worm program: An analysis. *SIGCOMM Comput. Commun. Rev.* 19(1), pp. 17-57. Available: <http://doi.acm.org/10.1145/66093.66095>

- [90] D. Buchmann, D. Jungo and U. Ultes-Nitsche. (2007, Oct). Improving network reliability by avoiding misconfiguration. *Design and Reliable Communication Networks, 2007. DRCN 2007. 6th International Workshop on* pp. 1-8.
- [91] Oppenheimer David. (2002, Dec). Why do internet services fail, and what can be done about it? University of California at Berkeley, Berkeley, CA, USA.
- [92] D. Moore, et al. (2003 Aug). Inside the slammer worm. *Security & Privacy, IEEE I(4)*, pp. 33-39.
- [93] Antony Savvas. Israeli Trojan espionage writers extradited for trial | 1 Feb 2006 | ComputerWeekly.com [Online].
<http://www.computerweekly.com/Articles/2006/02/01/213977/israeli-trojan-espionage-writers-extradited-for-trial.htm> [accessed: 12/06/2009]
- [94] DNSBL Information - Spam Database Lookup [Online]. <http://www.dnsbl.info/> [accessed: 01/07/2009]
- [95] BBC NEWS. 'Big bang' experiment is hacked [Online].
<http://news.bbc.co.uk/2/hi/technology/7616622.stm> [accessed: 01/07/2009]
- [96] John Wack, Ken Cutler, Jamie Pole. (2002, Jan). Guidelines on firewalls and firewall policy. National Institute of Standards and Technology, U.S. Department of Commerce. [online]. Available: <http://csrc.nist.gov/publications/nistpubs/800-41/sp800-41.pdf>
- [97] M. Rhodes-Ousley, R. Bragg and K. Strassberg. (2004), *Network Security: The Complete Reference*, Available: http://isbnadb.com/d/book/network_security_a10
- [98] A. Wool. (2004 Jun). A quantitative study of firewall configuration errors. *Computer* 37(6), pp. 62-67.
- [99] Jason. Vista: Security So Annoying You'll Turn It Off | PC Tips [Online].
<http://www.pctipsbox.com/vista-security-so-annoying-youll-turn-it-off/> [accessed: 07/07/2009]

- [100] Ken Fisher. Vista's UAC security prompt was designed to annoy you - Ars Technica [Online]. <http://arstechnica.com/security/news/2008/04/vistas-uac-security-prompt-was-designed-to-annoy-you.ars> [accessed: 07/07/2009]
- [101] Lawrence A. Gordon, Martin P. Loeb, William Lucyshyn, Robert Richardson, "2006 CSI/FBI computer crime and security survey," Computer Security Institute, Computer Security Institute, Dec, 2006.
- [102] Symantec - AntiVirus, Anti-Spyware, Endpoint Security, Backup, Storage Solutions [Online]. <http://www.symantec.com> [accessed: 13/05/2009]
- [103] McAfee-Antivirus Software and Intrusion Prevention Solutions [Online]. <http://www.mcafee.com/> [accessed: 13/05/2009]
- [104] D. Russell, D. Russell and G. T. Gangemi. (1991), (Dec). *Computer Security Basics*, Available: http://isbndb.com/d/book/computer_security_basics_a01
- [105] J. R. Thomas Chen, "The Evolution of Viruses and Worms," *Statistical Methods in Computer Security*, Dec. 2004.
- [106] The PC Tools Advanced Research Team. ThreatFire AntiVirus - Behavioral Virus and Spyware Protection [Online]. <http://www.threatfire.com/> [accessed: 06/07/2009]
- [107] J. Sanok Daniel J. An analysis of how antivirus methodologies are utilized in protecting computers from malicious code. Presented at InfoSecCD '05: Proceedings of the 2nd Annual Conference on Information Security Curriculum Development. Available: <http://doi.acm.org/10.1145/1107622.1107655>
- [108] James P. Anderson. (1980, Apr). *Computer security threat monitoring and surveillance*. James P. Anderson Co., Fort Washington, PA.
- [109] D. E. Denning. (1987 Feb). An intrusion-detection model. *IEEE Trans. Software Eng.* 13pp. 222-232.
- [110] SRI. SRI Internation - Intrusion Detection [Online]. <http://www.csl.sri.com/programs/intrusion/> [accessed: 18/05/2009]

- [111] C. Endorf, E. Schultz and J. Mellander. (2004), (Jun). *Intrusion Detection & Prevention*, Available: http://isbnadb.com/d/book/intrusion_detection_prevention
- [112] L. T. Heberlein, et al. (1990, May). A network security monitor. *Research in Security and Privacy, 1990. Proceedings. , 1990 IEEE Computer Society Symposium on* pp. 296-304.
- [113] Open Source. TCPDUMP/LIBPCAP public repository [Online]. <http://www.tcpdump.org/> [accessed: 22/06/2009]
- [114] N. Patil, et al. (2008, Jul). Analysis of distributed intrusion detection systems using mobile agents. *Emerging Trends in Engineering and Technology, 2008. ICETET '08. First International Conference on* pp. 1255-1260.
- [115] Zhengbing Hu, Zhitang Li and Junqi Wu. A novel network intrusion detection system (NIDS) based on signatures search of data mining. Presented at E-Forensics '08: Proceedings of the 1st International Conference on Forensic Applications and Techniques in Telecommunications, Information, and Multimedia and Workshop.
- [116] K. Ilgun, R. A. Kemmerer and P. A. Porras. (1995 Mar). State transition analysis: A rule-based intrusion detection approach. *IEEE Trans.Softw.Eng.* 21(3), pp. 181-199. Available: <http://dx.doi.org/10.1109/32.372146>
- [117] R. U. Rehman., (2003), *Intrusion Detection Systems with Snort: Advanced IDS Techniques using Snort, Apache, MySQL, PHP, and ACID*, Available: http://isbnadb.com/d/book/intrusion_detection_systems_with_snort
- [118] Sundaram Aurobindo. (1996 Dec). An introduction to intrusion detection. *Crossroads* 2(4), pp. 3-7. Available: <http://doi.acm.org/10.1145/332159.332161>
- [119] D. H. Summerville, N. Nwanze and V. A. Skormin. (2004, Jun). Anomalous packet identification for network intrusion detection. *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC* pp. 60-67.
- [120] R. Moskovitch, et al. (2007, May). Host based intrusion detection using machine learning. *Intelligence and Security Informatics, 2007 IEEE* pp. 107-114.

- [121] OSSEC - Open Source Host-based Intrusion Detection System [Online]. <http://www.ossec.net/> [accessed: 21/05/2009]
- [122] Configuration Control - Tripwire [Online]. <http://www.tripwire.com/> [accessed: 16/01/2010]
- [123] J. M. Kizza., (2005), *Computer Network Security*, Available: http://isbndb.com/d/book/computer_network_security_a01
- [124] D. Wagner and P. Soto. Mimicry attacks on host-based intrusion detection systems. Presented at CCS '02: Proceedings of the 9th ACM Conference on Computer and Communications Security. Available: <http://doi.acm.org/10.1145/586110.586145>
- [125] B. R. Raghunath and S. N. Mahadeo. (2008, Jul). Network intrusion detection system (NIDS). *Emerging Trends in Engineering and Technology, 2008. ICETET '08. First International Conference on* pp. 1272-1277.
- [126] ISS - Information security products [Online]. <http://www-935.ibm.com/services/us/index.wss/offerfamily/iss/a1029097> [accessed: 22/05/2009]
- [127] R. Salim and G. S. V. R. Krishna Rao. (2006, Feb). Design and development of network intrusion detection system detection scheme on network processing unit. *Advanced Communication Technology, 2006. ICACT 2006. the 8th International Conference 2pp.* 1023-1025.
- [128] M. Garuba, Chunmei Liu and D. Fraites. (2008, Apr). Intrusion techniques: Comparative study of network intrusion detection systems. *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on* pp. 592-598.
- [129] S. R. Snapp, et al. (1991, Mar). A system for distributed intrusion detection. *Compcon Spring '91. Digest of Papers* pp. 170-176.
- [130] J. S. Balasubramanian, et al. (1998, Dec). An architecture for intrusion detection using autonomous agents. *Computer Security Applications Conference, 1998, Proceedings. , 14th Annual* pp. 13-24.

- [131] Eugene H. Spafford and Diego Zamboni, "Intrusion detection using autonomous agents," *Computer Networks*, vol. 34, pp. 547-570, Oct. 2000.
- [132] Bass Tim. (2000 Dec). Intrusion detection systems and multisensor data fusion. *Commun ACM* 43(4), pp. 99-105. Available: <http://doi.acm.org/10.1145/332051.332079>
- [133] P. Porras. (1993, Jul). STAT -- A state transition analysis tool for intrusion detection. University of California at Santa Barbara, Santa Barbara, CA, USA.
- [134] G. S. Mark Crosbie. Defending a computer system using autonomous agents. Presented at Proceedings of the 18th National Information Systems Security Conference.
- [135] J. M. Bradshaw. (1997, Dec). "An introduction to software agents," in *Software Agents* Anonymous Available: http://isbndb.com/d/book/software_agents
- [136] PreludeIDS: Universal SIM [Online]. <http://www.prelude-ids.com/en/solutions/universal-sim/index.html> [accessed: 23/05/2009]
- [137] Network Working Group. RFC 4765 - The Intrusion Detection Message Exchange Format (IDMEF) [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=4765> [accessed: 22/06/2009]
- [138] Cisco. Cisco - Cisco Intrusion Detection [Online]. <http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/index.shtml> [accessed: 23/05/2009]
- [139] Open Source. WinPcap, The Packet Capture and Network Monitoring Library for Windows [Online]. <http://www.winpcap.org/> [accessed: 22/06/2009]
- [140] Microsoft Corporation. The Official Microsoft IIS Site [Online]. <http://www.iis.net/> [accessed: 05/10/2009]
- [141] Henning Schulzrinne. Internet Technical Resources: Traffic [Online]. <http://www.cs.columbia.edu/~hgs/internet/traffic.html> [accessed: 16/01/2010]
- [142] Net Applications. Study claims Windows usage market [Online]. <http://www.tgdaily.com/trendwatch-features/38232-study-claims-windows-usage-market-share-could-fall-below-90-soon> [accessed: 16/01/2010]

- [143] Network Working Group. RFC 4767 - The Intrusion Detection Exchange Protocol (IDXP) [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=4767> [accessed: 22/06/2009]
- [144] Network Working Group. RFC 2616 - Hypertext Transfer Protocol - HTTP [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=2616> [accessed: 26/06/2009]
- [145] Network Working Group. RFC 2818 - HTTPS [Online]. <http://www.rfc-archive.org/getrfc.php?rfc=2818> [accessed: 26/06/2009]
- [146] IP Address Lookup - IP Trace - Geographic IP Database [Online]. <http://www.hostip.info/> [accessed: 08/11/2009]
- [147] Jesse James Garrett. Adaptive Path. Ajax: A New Approach to Web Applications [Online]. <http://www.adaptivepath.com/ideas/essays/archives/000385.php> Available: <http://adaptivepath.com/> [accessed: 22/06/2009]
- [148] Web 2.0 Tutorial [Online]. <http://web2tutorial.wikispaces.com/> [accessed: 08/11/2009]
- [149] Tim O'Reilly. What Is Web 2.0 - O'Reilly Media [Online]. <http://oreilly.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html> [accessed: 19/06/2009]
- [150] M. Keller and M. Nussbaumer. Cascading style sheets: A novel approach towards productive styling with today's standards. Presented at WWW '09: Proceedings of the 18th International Conference on World Wide Web. Available: <http://doi.acm.org/10.1145/1526709.1526907>
- [151] Google. Page Speed [Online]. <http://code.google.com/speed/page-speed/> [accessed: 07/08/2009]
- [152] Jamey Boje. Firebug - Web Development Evolved [Online]. <http://getfirebug.com/> [accessed: 23/10/2009]
- [153] CNET Download.com. Ping Master [Online]. http://download.cnet.com/Ping-Master/3000-2085_4-10437994.html [accessed: 23/10/2009]

- [154] F-Secure Corporation. Backdoor Virus Description [Online]. <http://www.f-secure.com/v-descs/backdoor.shtml> [accessed: 08/08/2009]
- [155] [www.sub7crew.Com](http://www.sub7crew.com). Official SubSeven Website [Online]. <http://hackpr.net/~sub7/main.shtml> [accessed: 08/08/2009]
- [156] www.tcp-ip-info.de. NetBus and NetBus 2.0 [Online]. http://www.tcp-ip-info.de/trojaner_und_viren/netbus_eng.htm [accessed: 08/08/2009]
- [157] The Programming Language Lua [Online]. <http://www.lua.org/> [accessed: 10/08/2009]
- [158] HEAnet, Ireland's National Education & Research Network [Online]. <http://www.heanet.ie/> [accessed: 16/01/2010]
- [159] Sectegritty Corportion. SecTegritty - Top 100 Attacking Hosts and Attacked Ports [Online]. <http://www.sectegritty.com/alerts/isc.shtml> [accessed: 11/08/2009]
- [160] Audit My PC .com. UDP 111 - Port Protocol Information and Warning![Online]. <http://www.auditmypc.com/port/udp-port-111.asp> [accessed: 23/10/2009]
- [161] DShield; Cooperative Network Security Community - Internet Security [Online]. <http://www.dshield.org/indexd.html> [accessed: 11/08/2009]
- [162] Symantec. Spyware.NetSpy | Symantec [Online]. http://www.symantec.com/security_response/writeup.jsp?docid=2004-080510-5653-99 [accessed: 11/08/2009]
- [163] Symantec. Mydoom | Symantec [Online]. http://www.symantec.com/security_response/writeup.jsp?docid=2005-021013-2446-99 [accessed: 11/08/2009]
- [164] DShield. Port 10000 (tcp/udp) | DShield; Cooperative Network Security Community - Internet Security [Online]. <http://www.dshield.org/port.html?port=10000> [accessed: 23/10/2009]
- [165] G. Jiang. (2002 Apr). Multiple vulnerabilities in SNMP. *Computer* 35(4), pp. 2-4.