Securing Network Communications
With a practical example of a Firewall written in C++

A Thesis
presented
to the
Institute of Technology, Sligo

In partial fulfilment
of the requirements for the Degree
of Master of Computing

By
Alan Barry
June 2006
TABLE OF CONTENTS

Acknowledgements iv
Introduction v
List of Tables viii
List of Diagrams viii
List of Figures x

Part I. Literature Review and Research

CHAPTER 1. Need for Security

10 Introduction 2
11 Modern Computer Problems 2
12 Security 3
13 Security Policy 6
14 Risks 7
  14.1 Worms 7
  14.2 Trojans and Spyware 8
  14.3 Denial of Services 10
15 Benefits 11
16 Implementation 13
17 Rule Sets 14
18 Intrusions 15
19 Information Systems 17
110 Conclusions 17

CHAPTER 2. Firewall Technologies

20 Introduction 20
21 Proxy Systems 21
22 Packet Filters 23
23 Network Address Translation Systems (NATS) 25
24 Bastion Hosts 29
25 Virtual Private Networks 32
26 Application Level Firewalls 34
CHAPTER 3. Developing a Security Strategy

3.0 Steps involved in securing Computer Communications 37
3.1 Senior Management Buy-In 38
3.2 Detailed Analysis of Current Work Practices 40
3.3 Adoption of “Best Practices” 41
3.4 Adequate Resourcing 43
3.5 Disaster Recovery Plan 45

Part II. A Practical Implementation of a Firewall in C++

CHAPTER 4. Project Aims and Design

4.0 Introduction 49
4.1 Aims 49
4.2 Design 51

CHAPTER 5. Implementation and Code

5.0 Introduction 55
5.1 GetFileData 55
5.2 StartFirewall 56
5.2.1 SocketClient 57
5.2.2 SocketCall 59
5.2.3 BindToSocket 60
5.2.4 Socket 63
Acknowledgements

I would like to thank the individuals for whom I have worked for in the Limerick Institute of Technology while I was pursuing the MSc in Computing. They have my gratitude for allowing me the time to attend the various modules they include -

Michael Guenn Head of the Department of Information Technology,
Terry Twomey Registrar,
Michael O’Connell Secretary/ Financial Controller and
Niall Corcoran IT Manager

I would like to specially mention the late Bernard Burke who was responsible for the running of the Training of Trainers programmes who was always so helpful in all the dealings I had had with him and his staff

I would like to thank Terry Young Head of School, Business and Humanities in the Institute of Technology, Sligo and also the exams office staff in IT Sligo for their professionalism

I would also like to thank my work colleagues who put up with my mood swings as I travelled the road of learning C++ I am grateful for their patience

Des O’Carroll for helping me with the world of C++ and for the generous giving of his time

I must sincerely thank Dr Seamus O’Shea of the University of Limerick for agreeing to be my supervisor. I am indebted to his wisdom on how I needed to approach this project and am grateful for his encouragement and advice

Finally I thank my family, my parents who for some reason believed I had the ability to pursue further education and who constantly pushed me to go further. My children Aoife, Saoirse and Cathal who forgave me all the long absences due to attending modules throughout the country. Also for their understanding when I needed to study and sit at the computer learning how to program in C++

My wife Eilis who bore the brunt of the workload as I concentrated on my studies for the constant love and support I thank you most of all
Introduction

The statistics that are quoted about user interactions with the Internet are hard to comprehend, a technology that didn't exist in any real commercial way less than 20 years ago is now probably the singularly most dominant resource available today. The growth of the Internet has taken place entirely in my lifetime and its evolvement from its early educational existence to its current presence is incredible. When I started on this Masters in Computing I jumped at the chance of being able to do the module on Computer Networks followed a year later by the module on Telecommunications. I had been for years using the Internet without a strong understanding of the principles behind its workings and saw these modules as a way to further improve my knowledge of this area.

After the first module I knew that my project on this Programme would have to, in some way, take into account network communications and so since 2000 I have been steadily learning more and more about network programming using C++. I originally planned to write an application that would allow a program to get down to the physical layer and I was successful in using a resource called Packet32.dll to achieve this.

However as Microsoft updated their operating systems from Windows NT to Windows 2000 and then onto Windows XP the ability to get down to the physical layer became extremely difficult. In my research to build a project with C++ network programming, I came across a paper titled Beej's Guide to Network Programming using Internet Sockets (http://beej.us/guide/bgnet/). The paper described how one can...
This Project shows how to design and build a simple Firewall that can protect a service/application being run by a company. It is based on using the Internet sockets in C++ and shows how to manipulate data sent over the network. The Firewall publishes a port number for an internal server's application. It will handle the communications between any outside clients trying to use this application, basically it will operate as the go-between between the client and server. No other personal computer, not on the applications server's network, can access the server directly without going through the firewall.

I originally started writing this project using Turbo C++ and have migrated along with Microsoft to Visual Studio 6.0 and then onto Visual Studio .NET. I have to admit that all attempts to put a visual front end on the application have proven too difficult and so as with most Microsoft Windows services I decided to opt for an INI file to hold the initialisation data. My learning curve has shown me the power of these developing environments but I do admit to missing Turbo C++.

The first part of this report stresses the need to Secure Network Communications. It shows my research and understanding into the area of network security. The second part is the documentation for a practical implementation of a Firewall in C++. It demonstrates the key principle of a firewall in its ability to act as safe go-between two networks. In most cases this is the outside Internet and the inside private network.
This implementation is designed to handle TCP network protocols and the common ports they use like HTTP (80), FTP (21) and TELNET (23). It shows how one can write with relative ease using Internet sockets reliable and stable network applications.

I will explain the code used in the application and give samples of the data that was sent to the Firewall, I will also show how the Firewall responded to this data by showing the results of simple tests.
List of Tables

Table 1  Virus/Worms payload dates set for 10 days of September 6

List of Diagrams

1 4  Typical Firewall 7
1 7  Limerick Institute of Technology's Cisco PIX rule set 15
2 1  Proxy Server 21
2 2  Detailed Analysis of network packets 24
2 3  Private IP Addresses passing through a single device 26
2 4  Bastion Host 29
2 5  VPN network between Branch office and Head Office over Internet 33
3 3  typical "best practices" flow 42
3 5  Disaster Recovery Plan Stages 46
4 1  Network Reference Model 49
4 2 1  Firewall as Choke Point 52
4 2 2  Top Down Design Chart 53
6 1  Test Scenario 68
6 2 1  Client IP Settings 69
6 2 2  Output from Route Print command 70
6 2 3  Clients Proxy settings show empty 71
6 3 1  Firewalls IP settings 72
6 3 2  Output from Route Print command on Firewall 73
6 3 3  Firewalls Proxy settings 73
List of Diagrams (Continued)

6.4.1 Client attempting access before firewall is running .................. 74
6.4.2 Successful connection to 172.16.150.102 ................................. 75
6.4.3 Successful connection to page 2 on 172.16.150.102 ................... 76
6.5.1 FTP Configuration setting on client ....................................... 77
6.5.2 Client unable to FTP to server .............................................. 78
6.5.3 Client successfully connecting to FTP to server ....................... 78
6.6.1 Client unable to telnet to server ......................................... 80
6.6.2 Client able to telnet to server ............................................. 80
## List of Figures

<table>
<thead>
<tr>
<th>PAGE</th>
<th>Figure Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>Socket Structure</td>
</tr>
<tr>
<td>55</td>
<td>Sample firewall</td>
</tr>
<tr>
<td>57</td>
<td>SocketClient Class</td>
</tr>
<tr>
<td>58</td>
<td>C++ Class for Storing Time Value</td>
</tr>
<tr>
<td>59</td>
<td>SocketCall Class method Data_EXISTS</td>
</tr>
<tr>
<td>59</td>
<td>BindToSocket Class</td>
</tr>
<tr>
<td>61</td>
<td>BindToSocket Class member function Accept</td>
</tr>
<tr>
<td>63</td>
<td>Socket Class member function Start</td>
</tr>
<tr>
<td>63</td>
<td>Socket Class member function Close</td>
</tr>
<tr>
<td>63</td>
<td>Socket Class destructor</td>
</tr>
<tr>
<td>64</td>
<td>Socket Class member function ReceiveData</td>
</tr>
<tr>
<td>65</td>
<td>Socket Class member function SendData</td>
</tr>
</tbody>
</table>
PART 1

The Need for Secure Network Communications

Literature Review and Research
Chapter 1

Need for Security
1.0 Introduction

Today's businesses rely heavily on information to compete in the global economy and the tool that allows greater and faster access to information in the workplace is unquestionably the Personal Computer. Information Technology has empowered the modern worker to achieve levels of performance which, until recently, would have been unattainable. Information Technology is constantly evolving but the key characteristic is the ability to deliver crucial information at crucial times. This asset has allowed companies to invest heavily in their computer infrastructure mainly with the guiding principle of getting the company a presence on the ever-growing Internet to communicate to the maximum number of potential customers.

The Internet is, in essence, a network of networks, every day thousands of new users are connecting to this international resource. It offers, from both a Commercial and Educational point of view, a vast resource for users that never before would have been thought possible. The small to medium size enterprises now have the means of accessing markets, that up to now, were only available to the large multinationals. The growing level of users on the Internet and the increase of e-commerce shows the importance of network communications in our world.

1.1 Modern Computer Problems

There is however a negative side, the Internet has proven that it too is susceptible to misuse. The majority of these problems with using networks today can be labelled into two main categories, that of nuisance problems on the one hand and the more serious types of cyber crime on the other. Each of these problems has led to the serious problem of Securing Network Communications. As the basis for this thesis we
Chapter 1

Need for Security

will look at the reasons for and the technologies available for protecting one’s network communications against these modern computer problems. Security is a common trend in all aspects of life today, from protecting your family, house, car etc., it is therefore no surprise that we must also protect our networks.

The first question one must ask when securing something is what exactly are you being asked to do? The example of the house is a good analogy, in securing your house you are not really concerned about someone stealing the property itself but mainly the goods within it. As with most businesses today, it is the information they have which is probably their greatest asset, thus the need to protect it. The firewall offers major advantages to companies in this context. The ability to channel all traffic to your network through one central point allows the network administrator to verify that the traffic entering his network is as expected and not unwanted. It would be the same as putting a moat around your house with only one bridge across and sentries constantly manning a checkpoint on the bridge.

1.2 Security

Security must be viewed in two major ways. Securing from external attack as well as monitoring attacks from within. The attack from outside is, the majority of times, an attack of chance, vulnerability is looked for, found and then taken advantage of. The attack from within the organisation can be more critical as often the location of sensitive data is known by the employee and can be copied or destroyed at ease. The firewall allows for protection on both counts, one can predict the type of traffic that is allowed into one’s organisation and also log the traffic that passes outwards. The level
of access defined for internal users can be stringently adhered to ensuring that the employee is only sending out information that he/she is expected to send out

In extreme circumstances internal attacks from disgruntled employees can take the form of damage to physical resources like routers, switches and servers. It is necessary to ensure these resources are properly secured and closely monitored. Security measures must also ensure that access to such critical equipment is restricted. This can take the form of locating these resources in secure rooms and also preventing unwanted access by restricting the ability to logon to such equipment with logon names and passwords.

To protect an organisation from an attack the entire IT infrastructure must be diagnosed for weaknesses. To do this we must break the network communications down to its different levels. This entails the networked personal computer, the company network and its link to the outside Internet. To fully protect a company's network resources all three separate entities need protection, this is the strength of the firewall. Placed strategically at the point of contact with the outside Internet, the firewall can quickly guard the internal network infrastructure from attack. Microsoft, reacting to the latest in a long line of security weaknesses discovered in its Win32 operating systems, is now even listing the steps to protecting your internet connectivity on their Windows Update Site (http://windowsupdate.microsoft.com/)
Chapter 1

Need for Security

1. Install a firewall
2. Keep your Virus Guard up-to-date
3. Patch your operating system to the latest release

To have the largest software vendor in the world listing installing a firewall as the first step in securing your personal computers shows the importance that firewalls are now seen in by today's corporate giants. To have a properly installed firewall at the point of access into your organisation allows you the time required to ensure steps 2 and 3 above can be carried out. Microsoft can justly describe the solution to security weaknesses above, but this does not take from the resources required to protect networks of 10 or more Personal Computers not to mention networks of 2 to 3 thousand computers. The two most damaging viruses/worms of 2003 were NACHI and Blaster, both these separate events cost millions in downtime for companies throughout the world. NACHI was proven preventable by anyone who had a firewall with its Internet facing interface blocking ports (135/TCP and 707/TCP) while Blaster was preventable from delivering its payload by having the ports (Port 135/TCP and 69/UDP) blocked. Both these Viruses could also have been avoided if a site had TCP port 4444 at the point of contact with the Internet blocked. In August 2005 a worm called Zotob was specifically written to take advantage of a known security issue on the Microsoft Windows 2000 operating system causing thousands of machines to shutdown worldwide, its payload was delivered on ports 8594 and 8080. The question is to what advantage was it to a company to have these ports open to the Internet in the first place.
1.3 Security Policy

A Firewall is not a one-stop shop for protecting your network from attack, it is however a major step forward in trying to establish a good security policy in preventing unwanted attention from the hacker. There are many commercial firewalls available to purchase in the market place today, most of these products are now widely respected by network administrators. The only problem with using one of these bought in products is the resources required in ensuring it meets your own company needs. The basis of a firewall is to ensure that only known computer traffic travels from and to your network. This requires a detailed look at the network traffic that runs on your network and agreeing what traffic needs to travel outwards.
A good policy for protecting one’s network communications from outside attack does not just consist of installing a firewall, a complete solution requires that first of all, one analyses closely the type of traffic that makes up your network communications. The protocol and the port numbers that a network packet carries, identify the type of communication that is in operation. It is the filtering of these protocols and port numbers that one can safely defend against unwanted traffic.

The exercise of checking your network traffic is often the most crucial step. One can restrict the allowed traffic through a firewall to the bare minimum and slowly over time allow the firewall to adapt to your company’s requirements. This is a major benefit of the firewall that it can implement the changing security requirements of your company. At its lowest level, the firewall can be viewed as a packet filter, checking each packet as it flows to and from your network and ensuring it conforms to the criteria you have set. Your internal network should be safe from undesired outside traffic unless it is traffic that you are expecting, and it also can be checked for accuracy by scanning with Anti-Virus Software etc.

1.4 Risks

Most organisations that operate an internal network are unfortunately at risk today by having unwanted Worms, Viruses, Trojans, Spyware and other such cyber problems inflicted on them if they have a connection to the Internet. The benefit of controlling access to the Internet through a firewall allows the network administrator to limit his initial weak point to the firewall itself. At any given point in time one can filter out traffic which is known as harmful, one can also install a virus checker to scan for known viruses before it travels on into the internal network. The best defence is a
mixture of implementing best practices (firewall, packet filter) together with Anti
Virus software and Anti-Spyware software and to constantly ensure these tools are
kept as up-to-date as possible

Diagram 1.4  Typical Firewall

1.4.1 Worms
The first prolific worm emerged in 1988 and was called the Morris Worm. It infected
at that time what was considered to be around 10% of the Internet (6500 Hosts) and
caused just over $100 Million dollars in damage. The first really prolific virus to
cause worldwide damage was the Melissa Virus in 2000. In its first few days it
infected over 320,000 hosts and in some cases sent confidential documents from these
machines to third parties all over the world, the cost of this virus was hard to quantify
but is put by most leading researchers in the billion dollar category.

"Lloyds of London insurers estimate could run into tens of millions of pounds in
the UK alone" – BBC 4th May 2000

1.4.2 Trojans and Spyware
The emergence of Trojans and Spyware has also highlighted the ever-constant threat
to network communications, how many people who have purchased online from
Amazon or one of their affiliates knows that a Trojan called ALEXA was placed on
their computer (http://pages.alexa.com/help/privacy.html?p=corp_w_t_40_b1). What percentage of hosts on the Internet report back their users' web usage, purchasing habits, search engines used and favourite sites to third parties unknown to the actual user? The majority of these Trojans may be harmless and cause no damage to one's computer but shouldn't you be in charge of what is being sent from your computer? The answer of course is yes, the problem is that you are not and like most users you just don't realize what is being sent. This is why securing network communications is so critical. The ability to prevent this is available and it is the firewall that can remove this problem, the majority of Trojans are merely installed in the background while one is browsing a website. Allowing web traffic only on the tcp protocol and on the http port (port 80) while browsing websites helps to ensure that non-html code is not activated without the users' knowledge. These Trojans may be the greatest cause of concern in the future. How many Trojans are already in existence spying on what your computer is doing? Who are these invasive software products reporting to and what are they reporting?

Even the American Federal Bureau of Investigation is working on its own Spyware product called Magic Lantern, which will spy on anyone who uses the software "pretty good privacy" for encrypting communications. This would allow the FBI to decrypt all encrypted communications with ease, this is in direct conflict of what the creators of PGP intended the software to be used for. It also raises the question how many other agencies or individuals have written Trojans to monitor for other specific habits of the computer user. Who can you trust? How safe is your data stored on your computer or on any file share that users have access to? The ease with which a specific Trojan could be written to discover all Microsoft Word documents or any
documents with doc as an extension could be fatal for any company trying to develop a product before the competition. What would be the cost of giving a competitor all your company's documents before you are in a position to release a product never mind after you release the product?

The purpose for installing a firewall is clear, it is to allow you to control the flow of information in and out of your network, this can be as simple as packet filtering or even a more productive approach would be to control traffic by user identification. It also allows for the installation at one specific location utilities to scan all data for known viruses and Spyware entering and leaving your network. A personal computer however is not fully protected unless its operating system is also patched to reflect weaknesses already discovered in the underlying operating system. Operating Systems such as the Microsoft Windows, Mac OS, Linux and other Unix operating systems are constantly being updated with patches to secure vulnerabilities that have been found in them. It is always important to ensure that your systems have these patches installed to protect against known weaknesses.

1.4.3 Denial of Services

The Denial of Service attack is potentially the greatest cost to any company, having the ICT resources, that the company may have invested thousands in, unavailable to both users and customers can result both in lost confidence and downtime. It might be a nuisance attack but it can prevent customers accessing your web site and this can be seen as lost revenue, it can also overwhelm servers to the point where they often fall over and cause a business endless hassle trying to get its users back online. The firewall may also be susceptible to denial of service attacks but at least the network
administrator can program it regarding how to react to certain known scenarios. It also prevents the attacks that are often untraceable from attacking the internal machines on the network.

This protection of internal machines allows for the company's internal network to continue on unscathed. The company might be unable to send or receive emails and other such traffic coming from the Internet but users still have access to their servers and can logon. Production networks behind the firewall can continue on while repairs or corrective action on the firewall can be put in place. Some organisations would also have the funds to have a failover firewall available to allow the business to continue seamlessly to the end user. The single channel into your network has already shown its benefits and once it is down the private network still cannot be attacked from outside, the administrator may still have a problem but at least he knows what the problem is and what it affects. If the firewall had not been in place how many network devices would have been compromised and how much greater would the job of recovery have been?

1.5 Benefits

One of the major benefits of firewalls is their ability to authenticate the traffic coming from the internal users. Most firewalls maintain a list of authenticated users and the permissions which have been assigned to them. Authentication allows for auditing, one can review the practices of one's users to see if more rules need to be built on the firewall. The practices of your users can show where one's security may need tightening or in certain cases loosening, it can be as simple as seeing a trend of users tuning in to radio stations on the web, to users downloading large size files through
peak times. Authentication allows you to pinpoint the users whose practices may need closer scrutiny, what would be the effect of a company being implicated in a child paedophilia ring if an employee was using its web site to send and receive criminal content. Who is responsible?

The other great benefit of authentication is its ability to trace back a problem that may be detected further down the road. If a company notices that the level of file transfer from its site was extraordinarily high during a period it can look through the logs and find who is responsible. This can show who might have brought in data on a disk etc., and infected their machine with a virus to just having some service misconfigured blasting out streams of data to some innocent third party. It does however arm the administrator with the ability to diagnose problems once they have been detected.

The firewall comes in all sizes from the enterprise models which may have a number of computers running the same services and sharing the task of protecting access to and from the network, to the personal firewall which is designed to offer a degree of security to the home user. They all share the same aims but have different levels of complexity involved. Microsoft Windows XP is the first generation of operating system to have a personal firewall embedded in the operating system. This firewall allows the operating system to create a temporary holding centre for all network communications so it can be processed against rule sets before being allowed to travel to the higher application layers. The standard firewall is usually installed on a dedicated machine, usually of a high specification to allow the fast flow of information to and from the Internet. The ability to protect communications is the main goal but this must be done in a timely manner, the modern computer users are not usually understanding if they are made to wait on services.
1.6 Implementation

The modern implementation of the firewall is usually on a personal computer with two or more interface cards. One configured with a private Internet protocol address to allow the pc to communicate with the internal LAN and the second configured with a real Internet protocol address to allow the pc have a presence on the Internet. It is the firewall software that allows for the traffic from one network to pass over onto traffic for the other network and vice versa. This software must deal with all the various types of attacks from unwanted intrusions, theft and denial of services to the different types of attacker i.e., the amateur hacker to the determined cyber terrorist.

The risk of attack by worms, viruses, hackers, denial of service (DOS) is also a big factor in determining what type of investment one must put into building a firewall. The risks are considerably different from being a governmental department network to that of a home user's pc. The habits of the network user play a key role in accessing your risk of exposure to an outside threat. When one is administrating a small network the potential threats can be easily prioritised as most important tasks such as allowing emails and web surfing can be catered for and as new needs arise they can be researched and implemented. Large networks on the other hand can often have a mix of users and with this mix come greater demands. Most large organisations now try and place most of their front facing network technologies like their Web Site, Software Depositories and Email Servers behind the firewall.

This greatly increases the complexity of sourcing a firewall that can cater for all the needs of a company. The frequency of attacks that are all too common place today...
have led organisations to publish the location of the Internet servers on the Firewall and bury the servers behind it. In practice this is a great asset to have, traffic can be streamed from advertised network addresses safely to the servers providing the organisations services. Web servers can only be accessed on protocols that have been set by the administrator, email servers only get to see emails that are being carried on TCP protocol 25. This greatly reduces the chances of having your organisation's services being overwhelmed by a Denial of Services attack, as the traffic is not reaching the server in the first place.

1.7 Rule Sets

The firewall can be programmed with different rule sets to overcome such known problems. A simple rule is to allow the programming of network addresses that the firewall will not accept traffic from, this enables the firewall to cope with Denial of Services attacks by dropping packets that are purposely being fired at it. It also enables the administrator to prevent traffic from sites that may have been identified as spam sites or nuisance sites that are constantly sending unwanted traffic. This is becoming more and more relevant in today's Internet as spammers constantly clog up communications with unsolicited emails. In itself spamming is not a major problem but who is paying the bills for the bandwidth that Spam email is taking up? Another simple but effective rule is to check for port scanning on its outward facing interfaces, a process can be put in place such that when traffic from another host on the internet attempts connections to multiple ports, to mark the host as suspicious.
Diagram 1.7 Limerick Institute of Technology’s Cisco PIX rule set

1.8 Intrusions

Intrusions are commonplace on today’s networks. Networks are constantly being probed for known weaknesses, vendors of software are often at fault for releasing patches to known problems, this allows the hacker to search for these published problems on hosts throughout the Internet. When a company like Microsoft publishes a patch and states that is to overcome a known problem in some process that may lead to remote control being handed over to a third party they can open the door for wide scale misuse. “Cyber geeks” are somehow armed with the knowledge that a flaw exists and they pursue it till they learn the full reason behind the patch. Most companies do not have the technical ability to stay up-to-date with patches so they are open to the vulnerability until they get a chance to install the patch. This window of opportunity allows the hacker the ability of sometimes taking over one’s PC and gaining access to your information.
The ability of the hacker to do this is another worrying problem facing network communications to date. This ability to overcome one's defences ensures that the day of every PC being directly connected to the Internet is long gone. The day of having to implement a firewall is on us and the sooner this technology is implemented by all networks is probably when the Internet will become a useful medium for communication. Years after the emergence of e-commerce the rise of the web for commercial business is still being thwarted by the lack of confidence the potential customer has for using web based technologies. Horror stories are abundant and are usually based on some degree of truth. The technology for promoting confidence is the firewall, securing one's network enables the customer to trust the services offered by a host. The ability to ensure that any data being transmitted over the Internet is safe is of key importance and that every effort is being made to keep it secure, promotes user confidence.

The growing use of computers in education is the testing ground for trying to ensure that good policies are put in place to protect network services. Universities worldwide are common targets for the hacker, the university usually offers large bandwidth, multiple hosts and wide diversity of technologies. In itself these resources can be seen as target-rich environments, attracting attention from all sorts of unwanted third parties. Colleges usually have web sites advertising college courses as well as email servers, intranets, ftp servers, name servers and many more services. Most major colleges will be probed nightly for weaknesses. While writing this chapter the Limerick Institute of Technology was probed by sixteen different hosts on one particular night alone.
1.9 Information Systems

The growing number of Information Systems and Production Systems that now run on networks show the real threat from breaches in security. Some modern companies can be fully controlled from a computer. Shannon Turbine Technologies allows for the majority of its production line to be controlled remotely by computer. After the Piper Alpha disaster in the North Sea in 1988, many of the drilling platforms were changed to allow remote control of the platforms positioning by computer from a shore based network. This was seen as a response to allowing the owners move the platform that may have been abandoned due to fire in the oil field. What if a hacker compromised this technology, what devastation would he/she wreak? The power outages in New York and Canada in the summer of 2004 were investigated to see if computer fraud was involved. The investigation was not based on evidence gathered, but rather the possibility that it may have been due to the large dependence on computers and networks.

1.10 Conclusion

There is a need to ensure our data is safe from harm. There are many technologies available but the one technology that seems to come to the forefront of security policies time and time again is the firewall. It is not an easy fix but arguably one of the most important steps in implementing a good security policy. It promotes the location of a network behind a safe bastion, it gives the administrator a central choke point to concentrate his efforts on how to protect his network, this buys the time required to allow the patching of all other weaknesses throughout the network.
allows for a coherent policy to be formulated and implemented which can grow as the network and the needs of the network change.

Firewall technology is based on implementing a layer of control between your network and the Internet, it is designed to allow your users to carry out the normal communications that you feel your users need in their day-to-day operations. Most firewalls now have a proxy facility that can also speed up access to sites that are used most frequently. At the same time the firewall protects your network and the data that may reside on it from unwanted attention. The cost of an implementation can be designed to reflect the risk and level of security required, firewalls are scaleable to ensure they grow as the demands on them grow. These benefits make it certain that any organisation, which needs to maintain a connection to the Internet for any type of communications purposes, needs to invest in a firewall.

The days of being able to neglect security on your network are long gone as there will only be one outcome, and the speed at which this lesson will be learned when one goes online on the Internet is now reduced to hours if not minutes. There is now no excuse for not taking the task of securing network communications seriously.
Chapter 2

Firewall Technologies
Chapter 2

2.0 Introduction

There are many different firewall technologies available to protect the computer networks today, the main technologies used are Proxy systems, Packet Filters, Network Address Translation Systems (NATS), Bastion hosts and Virtual Private Networks. The main commercial security products available can implement any one of these technologies to protect customer networks, the market leaders in security software usually tend to implement a mixture of these technologies to give the best total solution to the needs of the modern network. The common practice of the network administrator today is to build a Local Area Network using the Internet Protocol (IP) on its communications layer with reserved private IP addresses and placing the firewall as the choke point or Network Address Translator (NAT) to enable the network clients talk to the outside Internet.

This design gives the network administrator the best defence in ensuring his internal network will not be attacked from external sources. No packet from the private IP range can talk directly to a computer with a real IP address and vice versa, no computer with a real IP address can talk directly with a computer who has a private IP address. The communications can only happen through the aid of a Network Address Translation service. As discussed earlier it is at this choke point that the firewall plays a key role in ensuring only safe traffic passes to/from one's network. All the above technologies build on this scenario to ensure that the communications to/from external sources are checked against a rule-set of what is allowed on your network. Based on the rule set network traffic is either allowed through the firewall or the packets are dropped.
Each technology is based on protecting the computers on the internal network from direct access from outside sources. The importance of the technologies above is how efficient they are at evaluating the traffic that comes to and from the internal network for unwanted and dangerous packets. The ability to do this in a correct and speedy manner enables the internal network a level of protection that prevents constant downtime in repairing systems from malicious damage. Unfortunately, this protection must be present for any company to survive the constant attacks being perpetrated against business networks today.

2.1 Proxy Systems

In its simplest form, a Proxy System is a system that provides Internet access for a host by passing on its requests from the network to the Internet server and receives back any responses generated from the Internet server and passes them on to the network client that originally initiated the request. The ability to only be able to talk to the proxy system rather than the Internet Server allows the proxy server to decide which requests to pass on and which to disregard. The user of the network client does not in fact see any difference using this method than talking directly with the Internet Server.
Proxy Systems became popular when the numbers of machines being attached to the Internet exploded in the late 80's. As there were only a limited number of real IP addresses available in Internet Protocol Version 4, Internet authorities began reducing the number of IP addresses they would give to new applications for connectivity onto the Internet. Proxying proved that all any company wishing to connect their network to the Internet needed was a single real IP address. Since the proxy is the only machine that communicates directly with the Internet, the use of address translation will allow as many hosts on the clients' network to talk to the Internet as it requires.

A benefit of this technology is also the ability, if required, to proxy just for a particular protocol or for a number of different protocols. Most of the modern operating systems allow the user to program their Internet client software to use a proxy for the different protocols. An example of this is normally where one can program HTTP to go to one particular proxy on port 80 and FTP to go out on another different proxy on port 21 etc. This ability allows different Internet usages to be logged by different servers as well as load balancing the traffic that is being generated out of a site. Adding to this the ability to cache the different proxy servers allows companies to provide access to Internet resources at a faster rate than putting their personal computer's directly on the Internet.

The cache proxy for HTTP sites can create a local copy of the services on offer from an Internet server. It can then by merely checking the date and time stamp on the actual site against that of a version held in local cache on the proxy server. The cached version can then be returned to the client PC at a vastly improved rate. This also reduces the level of traffic passing over the Internet and reduces the workload of
the proxy server in connecting to outside servers. This has the added advantage of reducing the level of traffic and contention for Internet bound traffic and traffic coming from the Internet. It also can provide services when the Internet connection is unavailable for some reason while still informing the user that the web link is from a cached archive, etc.

2.2 Packet Filters

Packet Filtering is one of the most powerful tools available to the system administrator today. It can decide what traffic is permissible out of and into one's network, this allows for greater security to be implemented when building a secure network. The Internet is built on the ability to route packets (information) from one network to the other, all communications are contained in these packets in a very structured and documented way. The ability to check this information at a choke point enables a system to react differently to varying kinds of information by merely examining the type of packet that is being sent to the choke point for passing on outwards to the internet.

Packet filtering allows for information to be checked in two very important ways:

- Information travelling out of the network
- Information travelling in to the network

One might only want a system to allow particular traffic to gain access to the Internet; a choke point might be programmed to allow only HTTP traffic the ability to access outside hosts. This would ensure that employees are only using the Internet for browsing websites. This could be further restricted by ensuring only listed external IP addresses are unblocked and only predefined and acceptable websites are contactable.
All communications out onto the Internet are two-way normally, a request for service, the response from a server, ready to receive acknowledgement and then the flow of information. This enables the packet filter to check that any outside traffic travelling to the network has a valid external address. This proves that it was originally contacted by the internal client and is not some external host trying to gain unwanted access. Packets that are detected as unwanted, or not expected, can be dropped at the packet filtering system ensuring no breach of the network. External addresses can also be checked against existing listings of servers that are deemed unsuitable or should not be allowed to communicate with internal clients.

Diagram 2.2  
Detailed Analysis of network packets

This form of packet filtering is often used to combat the ever-increasing problem of spam mail. Some sites publish their mail server behind a packet filtering system maintaining a list of sites that are noted for producing unsolicited Spam mail. The packet filtering system can detect communications from blocked or blacklisted sites.
and drop these packets without having to take any further action. This enables the mail server behind to work away as normal and reduces the level of traffic it would need to process. Most sites that operated Packet Filtering on the choke point of their networks were protected from the summer onslaught of Viruses and Worms released during the summer of 2003. Looking for well-documented traffic on particular TCP ports easily identified Nachi or Blaster, Welch's and earlier worms such as Slammer. This traffic was then dropped outside the network leaving the administrator the time required to ensure that all internal computers were patched to prevent infection.

Screening traffic into one's network is by far the greatest tool at protecting your IT infrastructure from attack, the main problem associated with screening however is one can restrict all traffic and allow only the protocols which users prove are required. This entails having a strong knowledge of TCP/IP and also understanding users who can afford to wait while the site traffic is being profiled. Although the most common solution, is usually the opposite, everything is permitted and slowly as attacks and weaknesses are found the offending protocol is closed from abuse. There is also the problem where the standard sets of protocols are constantly being tested for weaknesses. When a weakness is found sites that falsely believed that they were protected are most at risk as they believe that they have protected their site from attack.

2.3 Network Address Translation Systems (NATS)

Network Address Translation Systems are the means by which modern secure networks communicate with hosts on the Internet. A typical large company today with a hundred or more networked devices would probably only have a handful of real IP
addresses. NATS allows a considerable number of hosts to share the same IP address when communicating out onto the Internet. Typically most companies will use a Dynamic Host Configuration Protocol server (DHCP server) to assign private IP addresses to hosts on the internal network. This in turn will be programmed to direct all packets needing to be routed to the Internet to a central routing service. This service can, by use of NATS, replace the source address of the TCP/IP packet with one from a handful of addresses programmed as part of the routing service.

![Diagram 2.3](image)

**Diagram 2.3** Private IP Addresses passing through a single device

This procedure ensures that all hosts contacted on the Internet are returning their communications to the one address. This address will have a NAT system running that knows from the destination address what was originally sent out and what internal client should have the packet returned to it. The NAT system modifies the returning packets by placing the internal client's IP address (source address) into the packet and forwarding them on. This shows how there is a reduced need for real IP addresses, as only the NAT system in theory requires an interface with a real IP address.

Modern implementations of NATS usually use a set of external addresses for the different types of traffic that are generated internally by the network. HTTP requests might go out on one IP address, SMTP requests go out on another different IP address and so on, this enables different packet filtering to be added to the traffic coming back on different interfaces. One can also specify that particular traffic may communicate...
with the NAT Server on an non standard port (i.e. Ports above 1024). The NAT server can in turn replace this unusual port number with the standard or common port number when communicating outwards. This will ensure that any outside host compromising the NAT Server will still not know what port normal communications with client machines occurs on.

There are numerous ways NAT Servers can operate. The standard configuration is where all traffic generated on the internal private network that requires passing on to the Internet has the client address section of their packets modified to show the external address of the NAT Servers interface only. Another variation is when the NAT Server maintains a list of multiple valid IP addresses for different types of communications as discussed earlier. This allows for load balancing but more importantly the ability to filter only prescribed communications on certain network interfaces.

Some organisations also run with a version of NATS that always maps the internal address onto a pre-programmed external address. This ensures that any PC on the internal private network always identifies itself uniquely to the outside host. Rather than having every host on the internal LAN appear as the same IP address, now machines can appear as a uniquely identifiable host at the server end. This process enables organisations to subscribe specific machines on the internal LAN on to web services such as commercially available databases. It also allows the administrator to identify the traffic that is being generated by internal hosts and the servers they are contacting outside of the network.
Chapter 2  
Firewall Technologies

The advantages of using a NATS system are two-fold: it can reduce the number of real IP addresses required and the number of physical connections to link a LAN onto the Internet and by doing so the costs of maintaining such links. Although not specifically designed as a security system, it offers protection to the systems behind the NAT Server while they communicate freely with the Internet by camouflaging their real location or address.

The disadvantages of NATS are also important to note. After installing a NAT Server one has now created a system or link to the Internet which has a single identifiable point of failure. The server is now mission critical to any services your company offers on the Internet as well as ensuring that communication lines stay open. It is another configurable resource that depends on expert technical knowledge to implement and often to support. NAT Servers also can be prone to poor performance as the internal network grows with the number of connections that need to be maintained at the choke point to ensure reliable communications.

Another downside of NATS is that the process of modifying internal traffic as it passes on outwards to the Internet negates the use of certain security technologies such as Digital Signatures on emails, etc. The email will always show that it has been tampered with when it arrives at the recipient end, as the NAT Server has replaced the source address on each packet with the IP address of the NAT Server. Technologies such as Digital Signatures are the new breed of tools being adopted to ensure reliability on the Internet and the possibility that the NAT System can undermine this service is a serious problem for those trying to ensure reliability and promote confidence with their customers.
2.4 Bastion Hosts

As the name describes, Bastion Hosts are computers on the perimeter of a company's network. They are the link between the internal Local Area Network (LAN) and the Internet. Normally, a Bastion Host would have security services like proxying and Network Address Translation Services (NATS) embedded on top of its operating system to allow connectivity with the Internet. The Bastion Host is the choke point between the Internal LAN and the Internet, because of this, one must be careful to configure the Bastion Host, as it is the most vulnerable from outside attack.

Because the Bastion Host has a direct connection with the Internet, one must pay particular attention to how securely it is configured from outside probes. The Bastion Host is the key component for linking LAN's and their services to the Internet, and due to this, one must ensure that it can withstand the hacking and other cyberspace attacks it will undoubtedly face.

Diagram 2.4 Bastion Host

The secret to developing a stable and well-configured Bastion Host is to know from the start that there is a likelihood that it may be compromised. In today’s Internet environment, it will almost certainly have its defenses overwhelmed in one shape or another.
Chapter 2

Firewall Technologies

another One should take every precaution in ensuring known weaknesses and common security flaws are hardened but it is just as important to ensure that when the Bastion Host is compromised the Internal Network is still safe. The easiest way of preventing your security measures from being overcome is that only necessary services are installed when building the servers operating system. Also one must ensure that only the ports for protocols and services such as the proxy service are left open.

Most organisations that depend on their Internet connectivity would install multiple Bastion Hosts for each of the services it would provide to its customers and employees. A Bastion Host may exist for supplying connections to the internal email Server, this may only have ports 25 and 110 open as all functions required by a standard email server can be carried out with just these ports. Port 25 would allow emails to be sent and received, while port 110 would allow users to POP (retrieve) their emails. Similarly another Bastion Host may only have port 80 open to allow outside access to the internal Web Server. Port 80 would allow all external users to request and retrieve web pages using the HyperText Transfer Protocol from the web server.

The ability to restrict the open ports allowed on each Bastion Server vastly reduces the number of potential weaknesses a machine may have. Most operating systems come with a wide range of services and applications to make the PC the powerful tool that it is today. These services and applications require multiple communication ports to be open. A Bastion host has need only for the ports relating to services it provides to be left open so during the initial load of the operating system it is advisable to
install only the minimum of options. One can always add on extra services if they are required but it is often far more complex to remove services cleanly.

A common practise in securing today's networks communications is cloning the Bastion Host when you have configured it to a level that you are happy with and all known weaknesses and potential exploits have been closed. This allows the network Administrator to rebuild his connection point to the Internet in a relatively short time span. Once a compromised system has been detected one can investigate the reason its defences where overcome, you can then rebuild your Bastion Host by using the cloned copy as discussed earlier. Now one can close or patch up the reason for the compromise on the newly built system and then re-clone the machine. Taking a cloned image again at this point ensures that one is constantly refining the protection on offer against outside attack. There is a learning curve but at least the Bastion Host is reducing the amount of potential damage by sacrificing only itself.

The downside of losing connection to the Bastion Host is obvious, but limiting the services it offers reduces not only the potential for compromise but also the effect it will have on your users when it is not available. Adding other technologies on top of the Bastion Host such as proxymg and NATS ensures one is making it harder and harder for external forces to affect your network performance and reliability. Only the most concerted of efforts will bring down a network of this design, at least now you have prevented the malicious attack of chance and will learn from the more serious attacks.
The advantages of having Bastion Hosts are easily seen and one must also recognise
that Bastion Hosts are cheap to implement, normally there is no other specialised
software needed, instead it is merely learning how to configure the operating system
on the servers at the periphery of you network. Most common operating systems are
used as Bastion hosts and there are even documented steps widely available on the
Internet.

2.5 Virtual Private Networks

A common problem before the emergence of the Internet was the huge cost in
connecting various networks together. Mostly the need for such a set-up was with the
large multinationals that may have had plants located all over the world. Spending
large amounts of money traditionally with the national telecommunication carrier to
lease lines connecting one or more sites was seen as a tremendous drain on a
company’s resources. The Internet on the other hand was designed exactly for this
reason - the joining of Local area Networks together to form one big Wide Area
Network (WAN).

The only problem with the Internet was the lack of security offered to a company in
trying to link its different networks together. The Virtual Private Network (VPN) was
seen as the solution to this problem. It allowed networks to be linked together over a
public network by using an extra-secured layer of encryption and authentication. The
Internet Engineering Task Force was tasked with the creation of standards for
building this secured link. They developed the IP Security Protocol Suite (IPSec)
which enables users to take advantage of cheap public networks to pass information.
between two different networks whilst still retaining the security that the information travelling over this link can not be seen by third parties

The scope a VPN allows to a company to securely and cheaply link its various networks is a great asset in ensuring the continuing growth of the Internet. There are many different types of VPN from the Hardware based system which strictly enforce encryption at the router level, to the software versions that enable organisations to create a VPN with clients or customers while not forcing them to purchase expensive hardware. A common implementation of the VPN is the firewall-VPN, which empowers the firewall with the ability to encrypt data being routed to a specific network. This allows the firewall to have the ability to have multiple VPN’s configured as well as standard access to the Internet. The hosts on the network can send information as normal on the internal private network but, when it hits the firewall, the firewall software can judge from its destination that the packets must be securely encrypted to prevent outside hosts on the Internet being able to interfere with the packet. Even if the packets are captured by a foreign host they are usually encoded with 128-bit encryption to prevent anyone from being able to read the information.

Diagram 2.5

VPN network between Branch office and Head Office over Internet
The downside of using a firewall-VPN is that once again you are loading the Firewall with extra duties. Each extra set of services that are placed on the firewall adds to potential performance problems in protecting your network. The added burden of encrypting packets can cause a slowdown with the communications between the firewall and its internal clients, this can often lead to reliability problems. The administrator is also concerned with the overhead of keeping the various VPN clients and their software versions up-to-date.

### 2.6 Application Level Firewalls

The application level firewalls allow the organisation greater scope in programming what information is permissible through its IT infrastructure. They are primarily being used as part of a firewall solution. It is common now to see a combination of firewall technologies being implemented, one at the network level such as the packet filtering, and then another at the application level.

While the network level firewall protects the organisation at a low level, the application level firewall allows the network administrator to scan emails or web pages for references or keywords such as 'porn', 'al-Qaida', 'paedophile' etc. It also allows the administrator to prevent downloads of data from web sites that may be listed as unsuitable.

Modern application level firewalls such as SNORT, Black Ice Defender and ZoneAlarm allow the administrator to maintain a database of keywords and sites that should be blocked. Commercial databases are also available for the administrator to subscribe to, these databases store information that the application level firewall can...
use to minimize the level of spam and unsolicited emails that pass through an email
server. It can also inform servers such as the proxy server on what sites are unsuitable
for users to access. This can be a great tool in ensuring that employees are not wasting
internet resources on unsuitable sites.

Another great advantage of the application level firewall is the ability to report on the
activity of users. Most of these types of firewalls come with excellent reporting
systems that allow the administrator to profile the internet usage of employees in the
organisation. This can easily identify workers who are operating outside the internet
usage policy of a company.
Chapter 3

Developing a Security Strategy

Securing Network Communications
3.0 Steps involved in securing Computer Communications

As already discussed there is no quick fix solution for securing your network communications, in actual fact, there is no absolute correct way of approaching such a task. Each network and the services it operates are usually very unique, most system administrators have their own methods of installing and securing computer systems and their implementation is totally dependent on what tasks the end users require of their network.

The network itself cannot be simply broken down into the physical infrastructure and the protocols that run over it when we need to investigate how to secure communications. We must also take account of the end user and the tasks they are trying to perform. Security is never a simple black and white decision. It cannot be broken into what should be allowed and what should be prevented no matter how desirable this would be. It is usually a best effort followed by constant tuning to get the best fit solution. The job of securing a computer network needs to be broken down into distinct and crucial tasks. One must always start with the adoption of a policy statement on what the security strategy is trying to achieve and how it will go about achieving these aims. The main points necessary for building a successful security strategy are:

1) Senior Management Buy-In
2) Detailed Analysis of Current Work Practices
3) Adoption of “Best Practices”
4) Adequate Resourcing
5) Disaster Recovery Plan
3.1 Senior Management Buy-In

All too often it is the Chief Information Officer or the Information Technology Manager who is given the responsibility for drawing up and implementing a computer security strategy within an organisation. It usually is a low priority in the main functions or running of the company. Most companies often adopt a standard strategy from the Internet with few or no modifications. As a service department, the IT section of most large corporations carry little or no weight when it comes to forming company policy. It is only when the CEO or Director signoff on the policy that it invariably becomes important.

The increasing reliance on MIS systems in companies is slowly starting to change this attitude. The increasing reliance on the company's computer network and its services places a greater need for compliance to a corporate or company-wide policy on how communications can be made secure. The first major step is having the heads of each section identify the needs of their departments with the computer network. They must take responsibility for the current practices that operate with their own departments. They must identify where the current weaknesses are and highlight what is crucial for them to do their daily business. They need to inform the IT department with the results of their audit in order to allow skilled IT personnel scrutinise what is happening and what needs to change.

Usually this internal audit identifies the majority of services that are critical and overlooks the normally mundane applications such as email and Web browsing that are taken for granted. MIS systems and Production systems such as SAP and JD
Developing a Security Strategy

Edwards are seen as key to a department’s success. Production or Operations Department will place little or no emphasis on Emails etc,

It is when these answers to the internal audit come back to the IT department a detailed breakdown or profile of the communications running over the network can be drawn up. Potential weaknesses can be pinpointed where the company might be compromised from a flaw in its security. The firewall as always is the greatest tool at ensuring the highest level of security can be enforced between the companies' private internal network and the outside Internet.

Senior management can be shown, after the installation of a firewall a breakdown of the traffic travelling to and from their company. This process is usually the one that awakens management’s interest and focuses their minds on the reliance their company may now have on computers and the communications between them. Certain services like E-Mail and WWW browsing, which are all too often taken for granted, are now highlighted as major resources required for the company to perform its day-to-day tasks. Management can now see for the first time how control over their communications can be organised and structured to meet their needs, it can also show how the firewall can be used to prevent outside groups from gaining access to critical data within the company. The benefits of a firewall can also be seen in preventing outside attacks on the services that the company has come to depend on, while ensuring that the services and tasks that are identified for each employee are being utilised for their intended purposes and not for non-work related items. This is another key component in winning over management as it ensures that the good name or
Chapter 3

Developing a Security Strategy

reputation of the company can be further protected by ensuring tight control over what company employees may be doing online with the Internet.

3.2 Detailed Analysis of Current Work Practices

The analysis of current work practices is a key step to ensuring that a good security policy can be drafted. The main aim of any such undertaking is to ensure that threats and risks to a company’s communications infrastructure can be identified. Once identified, these risks can then be managed to ensure the computer systems will react in a predictable manner. Thankfully there are many resources available today to the IT department to highlight potential problems with a company’s computing resources, the hardware, software and the network that the data is transmitted through.

When one identifies the different elements that make up the IT infrastructure, then we are able to ensure that protection is provided at every step in the system chain and not just at specific levels. The firewall may be the greatest defence in providing cover at a single location to the most computers but it still depends on a certain level of compliance to security standards by the devices it tries to protect. The security policy is the agreement as to the level of self-protection the computers and employees behind the firewall will need to maintain. A detailed list of the devices connected to a company’s network needs to be kept, a listing of the operating systems being used and the patches (both for the operating system and security) needs to be maintained. The key ingredient however is identifying the tasks that are being performed by the user and the services he uses to perform these tasks.
Developing a Security Strategy

Chapter 3

It is unrealistic to think that any organisation can have the unlimited resources required to audit the practices of every employee that uses a computer. It is however the aim of an audit on computer usage to begin the process of "employee involvement" and "self-learning" that will facilitate the adoption of a computer policy. It will eventually depend on the employee on how he will implement any such policy for protecting himself and the company by adhering to good computer practices. An important exercise during the audit is to get each end-user to complete a survey questionnaire which will identify the level of knowledge that employee’s have when it comes to handling the different levels of potential threats.

The audit is also the key tool to identify what needs to be re-enforced by a security strategy. It pinpoints the need to highlight all the aspects of a good security strategy by identifying the different components required. Access control policy, email policy, internet policy and communications policy are but some of the separate policies required giving a complete and overall picture. The implementation of the security strategy will only succeed when all the components required are looked at and a detailed action plan on how they will be protected is agreed upon.

3.3 Adoption of "Best Practices"

The adoption of "best practices" when protecting one’s computer communications is now becoming easier as the Internet community is being constantly driven to improve the security mechanism’s on offer. Security is time and time again being identified as the one factor that is preventing the Internet becoming the commercial market place that most modern companies are aiming for. Institutions like the National Institute of Standards and Technology’s (NIST) operate a Computer Security Resource Centre.
(CSRC) who have listed as one of their aims - "Raising awareness of IT risks, vulnerabilities and protection requirements, particularly for new and emerging technologies," The American National Security Agency also freely distributes such guidelines as the C2 Security Standard to enable users harden their personal computers to a high standard. With these highly respected institutions making security advisories freely available there is now a general trend to co-operatively build a standard set of security guidelines to help in the building of formidable defences in protecting one's networked computers.

Microsoft, Linux and all other major software vendors now offer online services to allow users to patch their operating systems against known security weaknesses that have been discovered or reported to the vendors. Microsoft's update centre now receives millions of online requests per day from computers trying to download critical updates, the majority of these requests are now from computers that are scheduled to check for updates at regular intervals thus ensuring that they are kept up to date. Services like these are playing a major role in combating and closing down the opportunities that are available to hackers.

Diagram 3.3  
Typical "best practices" flow
Most organisations see the adoption of best practices as a means to prevent abuse of email and Internet use within a company, this is mainly to protect the good name of the organisation against any civil or criminal law suits which would be publicised by the media as another scandal using electronic communications. “Best Practices” in the IT industry cover more than just this one area; they are designed to enable an organisation to respond in a timely and appropriate fashion to security events which are happening all too often. Companies with a major investment or reliance upon their networks cannot merely be seen to be reacting to events but must put in place precautions to prevent potential security breaches that have already been discovered or documented widely on the Internet. They must identify by use of best practices the areas of their electronic communications that are vulnerable to unwanted third parties as well as using these best practices to ensure that necessary steps are being taken to ensure such weaknesses are addressed as well as planning how to cater for potential other weaknesses.

3.4 Adequate Resourcing

Budgeting for an IT department today is a very complex task, all too often major pressure is placed to ensure that costs are kept at a minimum. The IT department must thread the fine line at being able to improve on current IT performance as well as being able to maintain a high level of quality or standards. Most IT professionals always know of a system that does the job better or a new tool which would be of benefit but cannot quantify on how much the new resource would actually cost. All too often it is the purchase price of a new system be it a computer, software system, utility or service that is seen as the end cost to the organisation, this is not the case.
Security is now an ongoing cost and needs to be budgeted for by the IT Department and the problem is how do we put a price on maintaining security.

It is relatively easy to put a price on the cost of a security system like a firewall. Such market leading applications like CheckPoint, Cisco Firewall and NetScreen all have web sites advertising their product details and the cost per site/license for installation. They do not however detail the cost of having to continually keep such a purchase up to date. Security is not a straightforward problem as already discussed with a simple answer. Securing network communications is an ongoing process that is constantly evolving, if you don’t continually maintain or evolve your security services then you are automatically falling behind and leaving your network vulnerable. A published weakness in an operating system with an advertised update is only advantageous to a company when they install the update. Up to when the update is installed the weakness is a serious security gap in your network’s defences.

Knowledge is the main resource that must be applied to this problem and it is this knowledge or expertise that costs money. More and more companies are now advertising their services as security consultants. It is quite normal now for a lot of large companies to employ a consulting security firm to check out the reliability of security precautions on their sites. These security firms usually have expertise in the latest methods used by hackers to gain access and unleash havoc on ones network. Organisations must ensure that employing independent security consultants is part of their security strategy as it is these consultants who can audit what protection is on offer to your network. They can continually assess you network resources as the
common hacker would and by monitoring from the outside can identify what services are being advertised to the Internet and their weaknesses

The recommendations from these consultants can be a major contribution to developing a security policy that is unique to your organisation. The policy can take into account the type of weaknesses that were discovered and how they make the internal network vulnerable. The process is a continual one and constant tests of the hardening exercise must be undertaken to show that recommendations are being examined and learned from, it also shows that the security strategy is an evolving strategy and not just a response to a particular incident at a particular time.

3.5 Disaster Recovery Plan

Probably the most important sign of awareness to developing a security strategy is the knowledge that at sometime ones network communications may be compromised and may need to be restored. The ability to plan for such a disaster is one of the greatest tools available to ensure that your network communications can be made fault tolerant and withstand even a serious breach of your security systems. The firewall as described is our single greatest tool for implementing secure network communications, however the reliance on a firewall places great emphasis that this choke point must stay operational for our network communications to continue.

Modern implementations of firewalls now take advantage of technologies such as Cluster Servers, redundancy technologies, Bastion Hosts or Enterprise-Wide solutions, all of which offer protection against a single point of failure that would halt all network communications to and from the Internet. The creation of a Disaster...
Recovery Plan however ensures that a policy is in place that details the steps needed to allow the resumption of normal services after a compromise of the firewall. All too often the firewall is blamed as the reason that the Internet is unavailable to the users of a network, the benefits of having the firewall in the first place are not usually high on the priority list of the end user.
Chapter 3
Developing a Security Strategy

The risk analysis of your computer systems is the key tool as to how the Disaster Recovery Plan is drafted and what steps need to be put in place to ensure your network communications can recover from an unwanted security breach. Security Assessment of every aspect of one's network communications environment including operating procedures, contingency planning, database security, data communications, access control and personal computers usage must be analysed. The important factor in putting a good Disaster Recovery Plan in place when specifically planning for a recovery from a security breach, is to learn from the security breach itself and adopt a solution as to how the recovery will cater for this breach in the future.
PART 2

The Need for Secure Network Communications

A Practical Implementation of a Firewall in C++
Chapter 4

Project Aims and Design
Chapter 4

Project Design

4.0 Introduction

This project shows how to build a firewall using C++ and Internet Sockets. As the scope of the project is very large we will concentrate on implementing a Bastion Host or choke point between the external Internet and the internal application server. The building blocks will be laid with this piece of code to enable future functionality to allow greater security for a network.

4.1 Aims

The aim of the project is to show an understanding of C++ programming and being able to implement a system from design to testing. Originally my interests lay at getting to the physical layer of the network i.e., the lowest level or the TCP/IP Reference Model and getting to the network packets themselves. This however is now extremely difficult with Windows XP as Microsoft have tightened up the security and ability to go to such a level. The only documented example of how such access is capable from Microsoft was in the Microsoft Development Kit with their PassThru NDIS Intermediate driver code and this only allowed specific functionality and not packet manipulation.

TCP/IP Reference Model

<table>
<thead>
<tr>
<th>TCP/IP Ref Model</th>
<th>TCP/IP Protocols</th>
<th>OSI Ref Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>FTP</td>
<td>Application</td>
</tr>
<tr>
<td>Transport</td>
<td>Telnet</td>
<td>Presentation</td>
</tr>
<tr>
<td>Internetwork</td>
<td>HTTP</td>
<td>Session</td>
</tr>
<tr>
<td>Host to Network</td>
<td>TCP</td>
<td>Transport</td>
</tr>
<tr>
<td></td>
<td>UDP</td>
<td>Network</td>
</tr>
<tr>
<td></td>
<td>IP</td>
<td>Data link</td>
</tr>
<tr>
<td></td>
<td>Ethernet</td>
<td>Physical</td>
</tr>
<tr>
<td></td>
<td>Packet Radio</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Point-to-Point</td>
<td></td>
</tr>
</tbody>
</table>

Diagram 4.1 Network Reference Model
From researching the area the most common approach to network programming in C++ is using Internet Sockets, commonly referred to as Winsock Programming. Winsock programming is basically a set of routines that provide a programming interface between an application and the TCP/IP protocols. The main building principle of Winsock programming is the socket. Based on the Berkeley Software Distribution (BSD) Sockets, the socket allows for Client/Server communication. The socket is the point of contact between a Server and a Client.

In its simplest form the Server creates a socket using a structure called sockaddr_in. This enables the programmer to describe the type of connection, the IP address of the server, and the port of the server to connect to. Once this socket is created the server can listen on this port and accept/receive data from a peer and send data back to the peer.

```
struct sockaddr_in {
    short int    sin_family,
    unsigned short in sin_port,
    struct in_addr sin_addr,
},
```

*Figure 4.1 Socket Structure*

This firewall project is based on the communication between Server and Client using the above structure, you will see throughout the project examples of where sin_family will be set to AF_INET, sin_port set to the port being advertised by the firewall and also being protected by the firewall and sin_addr will have the IP address of the firewall. Sin_family set to AF_INET allows for both connection oriented and connectionless type of communication i.e. TCP and UDP.
This firewall is designed to act as a service on a computer running the Microsoft Windows XP operating system, once initiated it will read the initialisation parameters for the firewall from a text file it will then position itself to protect a named service and will continue to handle data delivered to its outside interface by passing them internally to the internal server service it is protecting.

4.2 Design

To achieve the aims of this project the project design must take into account the need for the firewall software to -

a) read initialisation parameters from a file
b) monitor a particular port on a named interface
c) accept and queue communications to this port
d) pass on these communications to the internal server
e) accept responses from the internal server
f) pass on internal communications to external peer
g) await further communications

The functionality described above will allow the application to act as a Bastion Host and showcase the major routines required for programming network communications. These routines can be identified in wmsock programming as socket, bind, connect, listen, accept, recv and send, all of which will be seen later in the code.

C++ has also the ability to set up distinct locations in memory to handle a running process. As the firewall will be running as a service I have used the ability of C++ to create threads to package the firewall in one memory location. This will offset the need to program the firewall to handle an ever-growing memory requirement at code level. As a service the application will be continually listening on one port and handle the potential for multiple connections, the amount of memory required could be immense. It is a strong advantage to the C++ programmer to be able to utilise threads.
throughout his/her program. The threat of an ever-increasing memory demand problem can easily be handed over to the operating system by encapsulating the firewall into one process and using the `_beginthreadex` function to handle potential memory handling problems.

The application will be the only route from which external clients can make contact with the internal server. All data must first be directed at the outside interface of the firewall and the correct port. The application will then queue the request received at the outside interface and redirect them to the internal server in order they arrived. It then waits for the response from the internal server and sends it on in turn to the external client.

**Diagram 4.2.1**  
Firewall as Choke point

The application on the internal server in this project will be any typical TCP application such as HTTP or FTP. The firewall will listen on the designated port and forward the request onto the proper port of the internal server such as port 80 for HTTP and 21 for FTP.
Diagram 4.2.2

Top Down Design chart

The file being used to store the initialisation data for the firewall is in the form of a flat file structure. The code is written to ignore any line with a comment, these comment lines are identified with the # character. It reads the remaining lines of the file to fill in the key security configuration data required by the firewall. These are the port the external address of the firewall is to listen on, the IP address of the internal server and the actual port number of the internal application. All the settings are stored in ASCII format.
Chapter 5

Implementation and Code
5.0 Introduction

I have implemented the code for this project using two main functions GetFileData and StartFirewall. These two main functions handle the initialisation of the Firewall with GetFileData and the handling of socket data with StartFirewall.

5.1 GetFileData

The function GetFileData uses 3 parameters which pass the initialisation values from this function back to the main body of the code. The three internal variables ConfigObject1, ConfigObject2 and ConfigObject3 are used to traverse over keywords in the file identifying the initialisation parameters. A simple identification of the file to be used can be made by ifstream inClientFile("firewall ini", ios::in), this allows for sequential access to the file. The ios::in declares the file is open and in read only mode. Ifstream by default creates an object called inClientFile which allows the programmer to access data in the file. The file that we are using is called “firewall ini”.

The next part of the GetFileData function is to ensure that the file exists. If no file is found the “File Damaged or cannot be read” message is displayed at the console. The next statement

```cpp
while (inClientFile.peek() == '#') {
    inClientFile.ignore(80, '\n'),
}
```

ensures that the pointer for reading the file skips any commentary in the file and is placed onto a line which contains initialisation data.
#firewall ini File
#This file stores configuration Data for Alan’s Firewall
#
#The first piece of configuration Data is the actual address of the server
#that is being published behind the firewall
#
#The second configuration setting is the number of the socket that your server
#is been published on the external firewall port number
#
#The third is the actual port that the internal server will act on
Server 192 168 1 102
port 1285
redirect 80

Figure 5.1 Sample firewall ini

The pointer is now positioned at the 11th line of the file firewall ini i.e. Server
192 168 1 102 The next lines of code in the GetFileData read the key data in
variables -

inClientFile >> ConfigObject1 >> aname,

The object inClientFile reads the first piece of text into ConfigObject1 and then reads
the IP address of the server in the variable aname. It repeats this for the data that is
loaded into the variables aport and rport. The application now has the parameters
required for configuring the Firewall

5.2 StartFirewall

The StartFirewall function is the main function in this project, it holds all the classes
and procedures used throughout this application for passing data between external
hosts and the private network. When initiated, it creates a new thread on the operating
system

_startthreadex(0, 0, StartFirewall, (void*) s, 0, &ReturnValue),

Here we are letting the operating system take care of the maintenance of this function
and merely pass a pointer to a socket s to the function StartFirewall. A pointer to
any data type, socket in this case, can be assigned to a pointer of type void * It enables us to pass data to the thread in the form of a structure \textit{i.e.} socket.

The global function \texttt{StartFirewall} is defined using the following code:

\begin{verbatim}
unsigned __stdcall StartFirewall (void* a)
\end{verbatim}

\texttt{__stdcall} is used to clean the stack used by the operating system before \texttt{StartFirewall} is activated. The procedures used inside the \texttt{StartFirewall} and their memory requirements are all now managed by the operating system while the firewall is running. This is a significant advantage to the C++ programmer as it removes the task of handling numerous potential memory issues such as dwindling resources and contention, and hands them all over to the operating system.

### 5.2.1 SocketClient

We will now discuss the main classes used throughout the program. \texttt{SocketClient} is a class used in the program, we create an instance of \texttt{SocketClient} called \texttt{client}. This class is used to ensure we can talk back to the client that is requesting communications with the internal application server. After creating this instance we use the initialisation details read in the firewall ini (see Page 55) for the internal application server and the port it is set to listen on. In this case the protected Server has an IP address of 192.168.1.102 and the port it is listening on is the standard HTML port of port 80. The external client will have to know the external IP address of the firewall and the port it is listening on. It should not for safety reasons ever know the actual IP address and port number of the internal server.
SocketClient SocketClient(const string& host, int port) {  
    hostent *he,
    if ((he = gethostbyname(host c_str())) == 0) {
        throw "Cannot get Host Name",
    }
    sockaddr_in addr,
    addr.sin_family = AF_INET,
    addr.sin_port = htons(port),
    addr.sin_addr = *((in_addr *)he->h_addr),
    memset(&addr.sin_zero, 0, 8),
    if (connect(SocketforData, (sockaddr *) &addr, sizeof(sockaddr)))
    {
        throw "Error connecting with socket ",
        strerror(WSAGetLastError()),
    }
}

Figure 5.2.1 SocketClient Class

hostent *he sets up a structure he of type hostent, hostent is a combination of fields that contain information about the host you are identifying. In our example we are using the function gethostbyname to return the details of host to the he structure. In essence we are identifying that 192 168 1 102 exists. The next 4 lines allow us to specify the details of the internal server. The sockaddr_in addr line enables us to specify a structure addr that will enable the firewall to communicate with the protected server. addr.sin_family = AF_INET informs us that we will be using the INET objects as our address family is IPv4. addr.sin_port is set equal to the port the internal server is listening on. htons(port), htons sets the value of port to a two byte representation (network byte order).

addr.sin_addr = *((in_addr *)he->h_addr) points the required IP address to the address we loaded into the he structure earlier i.e. 192 168 1 102. We now have all the required information to create a connection to the internal server. It is important in Internet socket programming to ensure no data exists in the socket before we use it.
hence the `memset(&addr sin_zero), 0, 8)` as this loads zeros into the
`addr sin_zero` element in the sockaddr_in structure

### 5.2.2 SocketCall

SocketCall is the next class used in the project, it basically allows us to associate and handle the data being passed by the functions as if they were being read from a file.

As C++ began as a programming language on the UNIX operating systems a lot of the programming concepts are built around how UNIX viewed most devices as files. The FD_ZERO and FD_SET allow us to process bit fields stored as arrays of integers.

FD_ZERO allows us to initialize a file descriptor `&fdSocketforData` to the null set.

FD_SET allows us to include a particular file descriptor into the FDSET. This example -

```cpp
FD_SET(const_cast<Socket*>(si)->SocketforData, &fdSocketforData)
```

shows how the socket pointer `si` is used for member selection by the object `SocketforData` as `<socket*>(si)` is a pointer to a socket. We use `const_cast` to allow us to modify its value as it was originally passed to the class as a type `Socket`.

```cpp
const * const si
```

The ability to do this enables us to manipulate the data later in the program.

```cpp
TIMEVAL StopWatch,
StopWatch tv_sec = 10,
StopWatch tv_usec = 500000,
TIMEVAL *pStopWatch,
pStopWatch = 0;
```

**Figure 5.2.2.1** C++ Class for storing time value

The above code allows us to set up a structure call `StopWatch` of type `TIMEVAL` where we can specify time values. In Figure 5.2.2 we set seconds to 10 and microseconds to 5000000 (half a second as 1,000,000 equals a second).

```cpp
select (0, &fdSocketforData, (fd_set*) 0, (fd_set*) 0, pStopWatch)
```
The above function checks if sockets descriptors are ready to read and write, the `select` function gives us a way of simultaneously checking if there is data to be received or if we can now send data without blocking our lines of communication. `pStopWatch` sets the timeout value for the `select` function. It basically is set to return when an event occurs or after 10.5 seconds.

```cpp
bool SocketCall::Data_Exists(Socket const* const s) {
    if (FD_ISSET(s->SocketforData, &fdSocketforData)) return true,
    return false,
}
```

**Figure 5.2.2.2** SocketCall Class method `Data_Exists`

The code above ensures that the program knows if there is still a connection established with sockets to the server by returning a TRUE or FALSE value.

### 5.2.3 BindToSocket

```cpp
BindToSocket BindToSocket(int port, int connections) {
    sockaddr_in my_addr,
    my_addr.sin_family = AF_INET,
    my_addr.sin_port = htons(port),
    my_addr.sin_addr s_addr = inet_addr("193 1 88 47"),
    memset(&(my_addr.sin_zero), '0', 8),
    SocketforData = socket(AF_INET, SOCK_STREAM, 0),
    cout << "Firewalls IP address is " << inet_ntoa(my_addr.sin_addr) << endl,
    if (SocketforData == INVALID_SOCKET) {
        throw "INVALID_SOCKET",
    }

    // socket is bound to my internet address //
    if (bind(SocketforData, (sockaddr *)&my_addr, sizeof(sockaddr_in))
    == SOCKET_ERROR) {
        closesocket(SocketforData),
        throw "SOCKET_ERROR",
    }
    listen(SocketforData, connections),
}
```

**Figure 5.2.3.1** BindToSocket Class
BindToSocket is the function responsible for setting the firewall to listen for incoming connections. Once again, it uses the `sockaddr_in` structure to set up the value required to create a socket. This time, it takes the value from `FireWallPort` in the main body of the program when it is called to populate the port to listen on. In this section of code, I hard-coded the IP address of the external adapter by using the code referenced by `my_addr`. I could have used the system parameter `INADDR_ANY` instead of the actual IP address but if your firewall has two interfaces then both could end up listening for connections on the prescribed port.

`SocketforData` is set to a socket structure to allow TCP connections. If `SOCK_DGRAM` was used instead of `SOCK_STREAM`, you would be able to cater for UDP traffic. I have left a simple output message to show that it is the outside IP address that has now a socket available for connections. The remaining lines of code just check that the socket was created successfully and that it now exists. The code

```
bind(SocketforData, (sockaddr *)&my_addr, sizeof(sockaddr_in))
```

is the actual piece of code that associates the socket with the IP address and port number. The `bind` function always returns a zero on successful association or a -1 on an error.

The final function of the `BindToSocket` class is to tell the socket to listen for incoming connections. The external address of our firewall is now listening for connections, the code

```
listen(SocketforData, connections)
```

sets the server into a waiting state that will handle a number of pending connections identified by the value of the variable `connections`. In our example, we originally set...
connections in the main body of the code to 5. Hence we now have a socket set on
the external IP address of our firewall that will handle up to 5 pending connections
before dropping any new connections.

Socket* BindToSocket Accept() {
    struct sockaddr_in client_addr,
    socklen_t len,
    SOCKET new_sock = accept(SocketforData, 0, 0),
    len = sizeof(client_addr),
    getpeername(new_sock, (struct sockaddr*)&client_addr, &len),
    cout << "Peer IP Address " << inet_ntoa(client_addr.sin_addr) «
        endl,

    if (new_sock == INVALID_SOCKET) {
        int rc = WSAGetLastError(),
        if(rc==WSAWOULDBLOCK) {
            return 0, // nonblocked call
        }
        else {
            throw "Invalid Socket",
        }
    }
    Socket* r = new Socket(new_sock),
    return r,
}

Figure 5.2.3.2    BindToSocket Class member function Accept

Now that our firewall is awaiting connections we must deal with what happens when
they arrive. Here the key line of code is

    SOCKET new_sock = accept(SocketforData, 0, 0),

It creates a new socket descriptor called new_sock that handles all communications
with the connected client. getpeername is a built-in socket function that returns the
network information of the client at the remote end of the communication.
In socket programming a peer is seen as the remote end of the communication, in our example it is the client that has connected to the firewall’s external IP address.

`getpeername` returns the now familiar structure `sockaddr_in`

```c
getpeername(new_sock, (struct sockaddr*)&client_addr, &len)
```

Places the peer's information into `client_addr` which is identified by a pointer to `sockaddr` This in turn has key host information such as the IP address of the peer.

```c
inet_ntoa(client_addr sin_addr)
```

to ascertain the IP address of the peer we simply use the code above. This will allow us to identify the addresses of all external clients.

As with most firewalls we may choose to store a list of external clients that we do not wish to communicate with. Comparing this list with the IP addresses retrieved above will allow us to take appropriate action such as dropping the communication. Ending the communication can be as easy as activating a `closesocket` function. We may also choose to write back access denied information to this unwanted client informing the client that his connections will not be accepted.

### 5.2.4 Socket

The final main class used throughout the project is the `socket` class itself. It contains the most member functions of any of the classes as we can manipulate a socket in many ways. It is primarily responsible for starting, sending, receiving and ending the transmission of data over the network.
void Socket Start() {
    if (!nofSockets) {
        WSADATA info,
        if (WSAStartup(MAKEWORD(2, 0), &info)) {
            cout << stderr << "WSA Startup Failed" << endl,
            exit(1),
        }
        ++nofSockets,
    }
}

Figure 5.2.4.1  Socket Class member function Start

Figure 5.2.4.1 describes how we initialise the program to use winsock (windows internet sockets) WSADATA notifies the computer that we will be using the winsock library the definitions of this library are brought in to the program using
#include <wsSock2.h>  WSAStartup defines what parameters the winsock library will support in this case the highest level version 2 0

void Socket Close() {
    closesocket(s_)
}

Figure 5.2.4.2  Socket Class member function Close

The function above simply closes any socket that is passed to it, it is called by the deconstructor of the class socket in the code below. It simply checks to see that the counter has reached empty and closes the socket

Socket ~Socket() {
    if (!--(*refCounter_)) {
        Close(),
        delete refCounter_,
    }
}

Figure 5.2.4.3  Socket Class destructor

Other member functions include SendBytes and ReceiveBytes which can be seen used in the main StartFirewall function for sending and receiving the data from the client to the firewall and from the firewall to the internal application server. Using
the method `Data_Exists` as described earlier the program can check that all the communications between Client and Server have been catered for

If communications are still pending on the external interface of the firewall then the `StartFirewall` function can process the data by calling the `ReceiveBytes` function

```c
string Socket ReceiveData() {
    string ReturnValue, tempHold,
    char buf[100], //setting buffer to 100
    while (1) {
        unsigned long argp = 0,
        if (ioctlsocket(SocketforData, FIONREAD, &argp) '!= 0)
            break,
        if (argp == 0) break,
        if (argp > 100) argp = 100,
        int HeldData = recv (SocketforData, buf, argp, 0),
        if (HeldData <= 0) break,
        tempHold assign (buf, HeldData),
        ReturnValue = ReturnValue + tempHold,
    }
    returnReturnValue,
}
```

**Figure 5.2.4.4** Socket Class member function ReceiveData

It uses the `ioctlsocket` function to control the Input / Output mode of the socket and to see if there is pending data on the socket `SocketforData`. The amount of data that is returned is equal to the amount that can be read in call to the `recv` function `buf` stores the number of bytes that are read at any one read of the socket we make sure only 100 bytes or less are read at any one time

The `ReturnValue` will be a string with all the data sent to the firewall combined back together. The firewall can now act as the go-between to the internal server as it now has the information request from the client stored as a string called `bytes` in the `StartFirewall` function. Now that we have the full request from the client we can now prepare to send it on to the internal server. This is now done using the `SendData`
function. This is easier done as the control of the sending of the data is entirely down
to end server, in our case the internal application server

```c
void Socket_SendData(const string& s) {
    send(SocketforData, s.c_str(), s.length(), 0);
}
```

**Figure 5.2.4.5** Socket Class member function SendData

Basically we accept the string of data we have put together from our ReceiveData
function and use the class member function SendData to initiate sending data out over
our socket. It tells the internal server the size of data it needs to send with the
s.length() argument and continue to sends the actual data until all has been
successfully received by the internal server.

We can reverse the exchange of data communications to use ReceiveData to accept
the responses from the internal server and SendData to send them onto the external
peer. I have placed these lines of code inside a while loop

```c
while (1)
```

that ensure the program continues until some outside action happens such as halting
the program. The firewall reuses these main classes continually to ensure that it
handles all the communications required in a stable manner.
Chapter 6

Testing and Results
6.0 Introduction

The firewall was built to prove that access could be granted and controlled from an external computer into a private network. To prove that our code can handle these communications I have set up a number of tests. These tests are designed to show how three of the main TCP protocols can be handled by our firewall. The test environment described below will show how we can prove that intercommunication between networks is catered for by the firewall.

6.1 Test Environment

In order to test the firewall properly I have set up two networks, one a private network of 192.168.1.0/24, and the second a private network of 172.16.0.0/16. A gateway exists between both these networks, and any device wishing to inter-connect between the two networks must use this gateway. I have designed the networks so only the firewall knows the name of the gateway between both. No other device is aware of the interconnection between the networks.

Diagram 6.1 Test Scenario
The host IP address will be different in all 3 of the tests as this will be the IP address of a Web Server, FTP Server and also a TELNET server. The 3 test scenarios will prove how the TCP protocols HTTP on port 80, FTP on port 21 and Telnet on port 23 will be protected by the Firewall.

6.2 Client

The client I am using for the test scenarios is a laptop computer. It has the Microsoft Windows XP operating system installed and no other software. The network card has been configured to use the IP address of 192.168.1.44 with subnet of 255.255.255.0. It has purposely been programmed not to have a gateway address in order to ensure no other device may be passing on its communications.

![Internet Protocol (TCP/IP) Properties]

**Diagram 6.2.1** Clients IP settings
From the settings shown in diagram 6.2.1 this computer should only be able to talk on a peer-to-peer basis with any other computer on the 192.168.1.0/24 network. A look at the routes for this computer confirms this.

---

**Interface List**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1</td>
<td>MS TCP Loopback interface</td>
</tr>
<tr>
<td>0x2</td>
<td>00 04 76 49 2c ce 3Com 10/100 Mini PCI Ethernet Adapter - Packet SchedulerMiniport</td>
</tr>
</tbody>
</table>

---

**Active Routes**

<table>
<thead>
<tr>
<th>Network Destination</th>
<th>Netmask</th>
<th>Gateway</th>
<th>Interface</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.0</td>
<td>255.0.0.0</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>1</td>
</tr>
<tr>
<td>192.168.1.44</td>
<td>255.255.255.0</td>
<td>192.168.1.44</td>
<td>192.168.1.44</td>
<td>20</td>
</tr>
<tr>
<td>192.168.1.255</td>
<td>255.255.255.255</td>
<td>192.168.1.44</td>
<td>192.168.1.44</td>
<td>20</td>
</tr>
<tr>
<td>224.0.0.0</td>
<td>240.0.0.0</td>
<td>192.168.1.44</td>
<td>192.168.1.44</td>
<td>20</td>
</tr>
<tr>
<td>255.255.255.255</td>
<td>255.255.255.255</td>
<td>192.168.1.44</td>
<td>192.168.1.44</td>
<td>1</td>
</tr>
</tbody>
</table>

**Persistent Routes**

None

---

**Diagram 6.2.2**

Output from Route Print command on Client

The table above shows clearly how the client has only 2 routes available, both of which are its own interface i.e. 192.168.1.44 or the loopback address of 127.0.0.1. It has no available route to any other network. Another means by which a computer may communicate with the internet is by having a proxy server configured. Diagram 6.2.3 shows that no such proxy server is configured on our client.

This client is in a state where it can only talk to other computers on the physical network it is attached to, access to any other network is not available.
6.3 Firewall

The firewall computer is a standard Pentium computer. The only software loaded on this machine is the Microsoft Windows operating system and Microsoft Visual Studio .NET. Microsoft Studio is loaded so that the DLL’s required by the firewall application are present. The firewall was installed on the 192.168.1.0/24 network with an IP address of 192.168.1.47. This IP address was hard coded into the application for these tests. It is on this interface of 192.168.1.47 that our application will listen for communications from clients.
I have also programmed the firewall computer to have a route to the 172.16.0.0/16 network by sending data onto 192.168.1.7. This can be seen in Diagram 6.3.2, which shows that the Firewall unlike the client has 3 gateways. The three gateways are itself, in the form of 192.168.1.47, and the loopback address of 127.0.0.1. It also has a gateway with an IP address of 192.168.1.7. The IP address of 192.168.1.7 is programmed as a persistent route; so the firewall will always have a means by which to send on data to the 172.16.0.0/16 network.
Chapter 6  Testing and Results

Interface List
0x1 ........................... MS TCP Loopback interface
0x10003 ...00 30 05 3b 50 83 ...... Intel(R) PRO/100 VM Network Connection

Active Routes:

<table>
<thead>
<tr>
<th>Network Destination</th>
<th>Netmask</th>
<th>Gateway</th>
<th>Interface</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.0</td>
<td>255.0.0.0</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>1</td>
</tr>
<tr>
<td>172.16.0.0</td>
<td>255.255.0.0</td>
<td>192.168.1.7</td>
<td>192.168.1.47</td>
<td>1</td>
</tr>
<tr>
<td>192.168.1.0</td>
<td>255.255.255.0</td>
<td>192.168.1.47</td>
<td>192.168.1.47</td>
<td>20</td>
</tr>
<tr>
<td>192.168.1.47</td>
<td>255.255.255.255</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>20</td>
</tr>
<tr>
<td>192.168.1.1255</td>
<td>255.255.255.255</td>
<td>192.168.1.47</td>
<td>192.168.1.47</td>
<td>20</td>
</tr>
<tr>
<td>224.0.0.0</td>
<td>240.0.0.0</td>
<td>192.168.1.47</td>
<td>192.168.1.47</td>
<td>20</td>
</tr>
<tr>
<td>255.255.255.255</td>
<td>255.255.255.255</td>
<td>192.168.1.47</td>
<td>192.168.1.47</td>
<td>1</td>
</tr>
</tbody>
</table>

Persistent Routes:

<table>
<thead>
<tr>
<th>Network Address</th>
<th>Netmask</th>
<th>Gateway Address</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.16.0.0</td>
<td>255.255.0.0</td>
<td>192.168.1.7</td>
<td>1</td>
</tr>
</tbody>
</table>

Diagram 6.3.2  Output from Route Print command on Firewall

Once again we ensure that no proxy settings are installed, as it is now down to our own firewall code to send on the information to the external application server.

Diagram 6.3.3  Firewalls Proxy settings

Page 73  Securing Network Communications
We are now ready to begin our 3 tests; the testing will take a similar form for all of the TCP protocols. We will describe the 3 servers including their IP address and the service they are hosting. We will also attempt to access the servers before the firewall is configured from the client and show the results we achieved. Finally we will attempt to gain access to the server by attempting connection to the firewall’s IP address and advertised port number.

6.4 Testing the HyperText Transfer Protocol

For the purposes of testing HTTP traffic on port 80, I have created a Web Server using Microsoft Windows XP and Microsoft Internet Information Service Version 6.0. I have placed two simple html files on this server to show that access has been achieved. These html files will clearly show if a host is able to make connection to the Web Server.

![Diagram 6.4.1](client-attempting-access-before-firewall-is-running.png)

**Diagram 6.4.1** Client attempting access before firewall is running
Diagram 6.4.1 shows the output a personal computer that does not have a direct connection to the Web Server will get. This was taken from our client test machine before the firewall was executed.

The firewall.ini has the following 3 configuration lines for our firewall application.

Server 172.16.150.102
port 1285
redirect 80

This shows that any connection on port 1285 will be redirected to port 80 on IP address 172.16.150.102. For the purposes of our test we will use http://192.168.1.47:1285 to attempt to gain access to the internal server from Internet Explorer on our client.

Diagram 6.4.2 Successful connection to 172.16.150.102
The test in diagram 6.4.3 shows that we are successful in gaining access to the website hosted by 172.16.150.12 from the client (192.168.1.44). This is only because we are using the interface of our firewall (192.168.1.47) and the opened port of 1285. Once we clicked the link for page 2 on the main page the browser automatically used the web address of http://192.168.1.47:1285/page2.htm to access the second html page.

From the test above we can confirm that our Firewall is successfully handling the HTTP protocol.
6.5 Testing the File Transfer Protocol

The aim of this test is to show we can redirect FTP traffic through our firewall and onto the internal FTP Server. For the purposes of this test I have installed the Microsoft FTP server with Microsoft IIS 6.0 on another computer with an IP address of 172.16.150.250. In order to show FTP is working on the client I have installed a free piece of software called Smart FTP. Once again we shall use the same client as in the test carried out on the HTTP protocol; but we will point the firewall at the new server application with the settings shown below.

Server 172.16.150.250
port 18588
redirect 21

We now need to configure our client to use the firewalls IP address of 192.168.1.47 and the port of 18588.

![Diagram 6.5.1 FTP Configuration setting on client](image)

An attempt to use the FTP software without the firewall running culminated in the output shown in diagram 6.5.2.
Chapter 6  
Testing and Results

The diagram 6.5.2 shows in the log section that “no connection could be made” this is what would be expected without having the firewall publishing the internal server application.

Diagram 6.5.2  
client unable to FTP to server

Diagram 6.5.3  
client successfully connecting to FTP to server
Diagram 6.5.3 shows how using the SmartFTP software that the client has installed has made a connection through the firewall to the internal server. The internal server identifies itself as running the Microsoft FTP Service, the first few lines of the log show how the connection has been made through 192.168.1.47 on port 18588 and how the user apex has been able to logon.

This test now confirms that the firewall code is capable of handling FTP traffic successfully.

6.6 Testing the TELNET Protocol

The aim of this test is to show that the firewall will support a telnet connection through its IP address and pass the data onto the internal telnet server. In order to make this test possible I have installed Microsoft Exchange on a computer with an IP address of 172.16.150.12. Microsoft Exchange will support the telnet protocol on its POP3 (Post Office Protocol version 3) port 110. Once again we need to update the firewall ini configuration parameters, these should now show:

Server 172.16.150.12
port 26001
redirect 110

The port setting of 26001 is again chosen at random, but will be used by the firewall application to redirect traffic to port 110 on the IP address of 172.16.150.12. For the purposes of this test we can use the windows command "telnet exe" to make our connection to the IP address 192.168.1.47. Diagram 6.6.1 shows how the client is unable to make a telnet connection when the firewall is not active.
The command `telnet 192.168.1.47 26001` instructs the telnet program to attempt its connection to 192.168.1.47 on port 26001. The telnet program should return output identifying itself to any client trying to make a connection.

**Diagram 6.6.1**  
Client unable to telnet to server

**Diagram 6.6.2**  
Client able to telnet to server
In diagram 6.6.2 we see that the internal server is identifying itself to the client by giving a successful +OK statement. We can now interact with the mail server by typing known pop commands, i.e., user alan.barry@ht.ie. We see this command returns another successful +OK statement waiting for more commands to follow.

From these tests we have now shown how 3 of the TCP protocols are catered for successfully by the firewall application.
Conclusion

The aim of this thesis has been to show an understanding of firewalls and the key role they play in securing network communications. The research that went into producing this work was designed to bring my knowledge of the security aspect of the Internet up-to-date as well as learning how to program using C++. This research has shown me how security is now key to ensuring the Internet will play a bigger part in people's lives. The potential for the Internet both for business and enjoyment is immense. The only downside is the constant struggle against misuse. The benefit in having a secure network of information available will speed up the rate at which our world will evolve.

Users will only gain confidence in using the Internet if it is stable and offers a consistent service. Governments now understand that the Internet must be protected; it is too valuable a resource to have a small number of individuals ruin it for all. Economies are now being linked to Internet usage, the Dot-com bubble burst of 2000 wiped billions of stock exchanges throughout the world. This may have been down to a slow down in world economies but showed the level of growth of Internet companies at the time and the level of potential for investment in them.

It has shown me that it is time we educated ourselves about the Internet. We need to invest into the resources required to secure it and make our communications safe. The everyday user is the key to protecting the Internet. Patching operating systems, using
virus scanners and implementing a firewall are key to securing your own interactions with the Internet. The firewall will become more and more important in protecting our IT resources. We have seen already how the firewall has evolved from a technology that was only installed by large corporations with IT departments and technical staff to now been shipped as part of your operating system. All the major computing security vendors now offer hassle free personal firewalls and it is time we used them.

The end user must take a more active role and educate himself/herself into what are safe practices. The production of this firewall in C++ shows how one can gain an understanding as to what is required at a very high level and with C++ a way to implement very quickly and efficiently. Of course we all don’t have to go and write our own firewall but we should be fully versed in the technology and benefits it will bring.
<table>
<thead>
<tr>
<th>Glossary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>B</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>C</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>D</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>E</strong></td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td><strong>F</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>H</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>I</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>J</strong></td>
</tr>
<tr>
<td><strong>L</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>M</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>NACHI</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Network Address Translation (NAT)</td>
</tr>
<tr>
<td>Packet</td>
</tr>
<tr>
<td>Packet Filter</td>
</tr>
<tr>
<td>Patching</td>
</tr>
<tr>
<td>Peer</td>
</tr>
<tr>
<td>Pointer</td>
</tr>
<tr>
<td>Policy</td>
</tr>
<tr>
<td>Protocol</td>
</tr>
<tr>
<td>Proxy Server</td>
</tr>
<tr>
<td>Reference Model</td>
</tr>
<tr>
<td>Rule Set</td>
</tr>
<tr>
<td>SAP</td>
</tr>
<tr>
<td>Sockets</td>
</tr>
<tr>
<td>Software</td>
</tr>
<tr>
<td>Spyware</td>
</tr>
<tr>
<td><strong>T</strong></td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>U</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>V</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>W</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Bibliography

Elizabeth D Zwicky, Simon Cooper & Brent Chapman, Building Internet Firewalls 2nd Edition, O Reilly, 1-56592-871-7


Bruce Middleton, Cyber Crime Investigator's Field Guide, Auerbach Publications, 0-8493-1192-6

Debra S Herrmann, Security Engineering and Information Assurance, Auerbach Publications, 0-8493-1163-2


Bob Quinn, Windows Sockets Network Programming, Addison Wesley, 0-201-63372-8

Andrew S Tanenbaum, Computer Networks, Pearson, 0-133-94248-1


Brian Hahn, C++ A Practical Introduction, NCC Blackwell, 1-85554-325-7

Stephen R Davis, C++ for Dummies, Hungry Minds Inc, 0-764-56852-3

Web Resources

EBRARY http://site.ebrary.com/lb/
Technical Index http://www.tionestop.com/
Windows Sockets http://www.sockets.com/winsock.htm
Appendix I

Firewall cpp

#include "stdafx.h"
#include "SocketDef.h"
#include <iostream>
#include <process.h>
#include <sstream>
#include <fstream>
#include <iomanip>
#include <cstdlib>

using std::string,
using std::cout,
using std::cin,
using std::ios,
using std::cerr,
using std::endl,
using std::left,
using std::right,
using std::fixed,
using std::showpoint,
using std::ifstream,
using std::setw,
using std::setprecision,

int FireWallPort,
string PublishedServer,
int PublishedPort,

unsigned __stdcall StartFirewall (void* a) {
    Socket* SerVer = (Socket*) a,
    SocketClient client(PublishedServer, PublishedPort),

    while (1) { 
        SocketCall Client_Server(SerVer, &client),
        bool connection_exist = true,
        if (Client_Server Data_Exists(SerVer)) { 
            string bytes = SerVer->ReceiveData(),
            client SendData(bytes),
            if (bytes empty()) connection_exist=false,
            if (Client_Server Data_Exists(&client)) { 
                string bytes = client ReceiveData(),
                SerVer->SendData(bytes),
                if (bytes empty()) connection_exist=false,
            }
            if (! connection_exist) { 
                break,
            }
        }
        delete SerVer,
    }
    return 0,
}
void GetFileData(char aname[], char aport[], char rport[])
{
    char ConfigObject1[7],
    char ConfigObject2[7],
    char ConfigObject3[9],
    ifstream mClientFile("firewall ini", ios in ),
    if (mClientFile) {
        cerr << " File Damaged or cannot be read" << endl,
            exit( 1 ),
    }
    while (mClientFile peek() == '#') {
        mClientFile ignore(80, '\n'),
    }
    mClientFile >> ConfigObject1 >> aname,
    mClientFile >> ConfigObject2 >> aport,
    mClientFile >> ConfigObject3 >> rport,
}

int main() {
    std stringstream s,
    char name[15],
    char FireWallPortal[4],
    char redirectproxy[4],
    GetFileData(name, FireWallPortal, redirectproxy),
    sscanf(FireWallPortal,"%d",&FireWallPort),
    PublishedServer = name,
    sscanf(redirectproxy,"%d",&PublishedPort),
    BindToSocket in(FireWallPort,5),
    while (1) {
        Socket* s=in Accept(),
        unsigned ReturnValue,
        _beginthreadex(0, 0, StartFirewall,(void*) s,0,&ReturnValue),
    }
    return 0,
}
appendix ii

socket.cpp

#include "SocketDef.h"
#include <iostream>
#include "ws2tcpip.h"

using namespace std;
// ensure all <iostream> contents are defined as part of std
using std::cout;
using std::endl;
using std::string;
// from now on the cout and endl being used are those of std:: namespace

int Socket::SocketsUsed= 0;

void Socket::Start() {
if (!SocketsUsed) {
  WSADATA wsaData;
  if (WSAStartup(MAKEWORD(2, 0), &wsaData)) {
    cout << stderr << "WSA Startup Failed to initialize" << endl;
    exit(1);
  }
  ++SocketsUsed;
}
}

Socket::Socket() : SocketforData(0) {
  Start();
  SocketforData = socket(AF_INET,SOCK_STREAM,0);
  // It is possible to use UDP sockets by
  // SOCK_DGRAM instead of SOCK_STREAM
  // above here in the socket setup
  if (SocketforData == INVALID_SOCKET) {
    throw "Socket Invalid";
  }
  SocketCount = new int(1);
}

Socket::Socket(SOCKET s) : SocketforData(s) {
  Start();
  SocketCount = new int(1);
};
Socket::Socket() {
    if (!--(*SocketCount)) {
        Close();
        delete SocketCount;
    }
    --SocketsUsed;
    if (!SocketsUsed) {
        End();
    }
}

Socket::Socket(const Socket& socketUsed) {
    SocketCount=socketUsed.SocketCount;
    (*SocketCount)++;
    SocketforData = socketUsed.SocketforData;
    SocketsUsed++;
}

Socket& Socket::operator=(Sockets socketUsed) {
    (*socketUsed.SocketCount) ++;
    SocketCount=socketUsed.SocketCount;
    SocketforData = socketUsed.SocketforData;
    SocketsUsed++;
    return *this;
}

void Socket::Close() {
    closesocket(SocketforData);
}

string Socket::ReceiveData() {
    string ReturnValue, tempHold;
    char buf[100]; //setting buffer to 100
    while (1) {
        unsigned long argp = 0;
        if (ioctlsocket(SocketforData, FIONREAD, &argp) != 0)
            break;
        if (argp == 0) break;
        if (argp > 100) argp = 100;

        int HeldData = recv(SocketforData, buf, argp, 0);
        if (HeldData <= 0) break;
        tempHold.assign(buf, HeldData);
        ReturnValue = ReturnValue + tempHold;
    }
    return ReturnValue;
}

void Socket::SendData(const string& s) {
    send(SocketforData, s.c_str(), s.length(), 0);
}
BindToSocket::BindToSocket(int port, int connections) {
    sockaddr_in my_addr;
    my_addr.sin_family = AF_INET;
    my_addr.sin_port = htons(port);
    my_addr.sin_addr.s_addr = inet_addr("193.1.88.47");
    memset(&(my_addr.sin_zero), '\0', 8);
    SocketforData = socket(AF_INET, SOCK_STREAM, 0);
    cout << "Firewalls IP address is: " << inet_ntoa(my_addr.sin_addr) << endl;

    if (SocketforData == INVALID_SOCKET) {
        throw "INVALID_SOCKET";
    }

    /* bind the socket to my internet address */
    if (bind(SocketforData, (sockaddr *)&my_addr, sizeof(sockaddr_in)) == SOCKET_ERROR) {
        closesocket(SocketforData);
        throw "INVALID_SOCKET";
    }
    listen(SocketforData, connections);
}

Socket* BindToSocket::Accept() {
    cout << "BindToSocket::Accept()" << endl;
    struct sockaddr_in client_addr;
    socklen_t len;
    SOCKET new_sock = accept(SocketforData, 0, 0);
    len = sizeof(client_addr);
    getpeername(new_sock, (struct sockaddr*)&client_addr, &len);
    cout << "Peer IP Address : " << inet_ntoa(client_addr.sin_addr) << endl;

    if (new_sock == INVALID_SOCKET) {
        int rc = WSAGetLastError();
        if(rc==WSAWOULDBLOCK) {
            return 0; // nonblocked call
        } else {
            throw "Invalid Socket";
        }
    }
    Socket* r = new Socket(new_sock);
    return r;
}
SocketClient SocketClient(const string& host, int port) Socket() {
    hostent *he,
    if ((he = gethostbyname(host.c_str())) == 0) {
        throw "Cannot get Host Name",
    }
    sockaddr_in addr,
    addr.sin_family = AF_INET,
    addr.sin_port = htons(port),
    addr.sin_addr = *((in_addr *)he->h_addr),
    memset(&addr.sin_zero, 0, 8),
    if (connect(SocketforData, (sockaddr *)&addr, sizeof(sockaddr))) {
        throw "Error connecting with socket ", strerror(WSAGetLastError()),
    }
}

SocketCall SocketCall(Socket const * const s1, Socket const * const s2) {
    FD_ZERO(&fdSocketforData),
    FD_SET(const_cast<Socket*>(s1)->SocketforData,&fdSocketforData),
    FD_SET(const_cast<Socket*>(s2)->SocketforData,&fdSocketforData),
    TIMEVAL StopWatch,
    StopWatch.tv_sec = 10, //Seconds
    StopWatch.tv_usec = 500000, //microseconds - standard timeslice = 100
    milliseconds = 100 X 100 >=1
    TIMEVAL *pStopWatch,
    pStopWatch = 0,
    if (select (0, &fdSocketforData, (fd_set*) 0, (fd_set*) 0, pStopWatch) == SOCKET_ERROR)
        throw "Socket Error",
    bool SocketCall Data_Exists(Socket const* const s) {
    if (FD_ISSET(s->SocketforData,&fdSocketforData)) return true,
    return false,
}

void Socket End() {
    cout << "Closing down Network Communications " << endl,
    WSACleanup(),
}

95