Firewalls in Linux:
Principles and Implementation

Yordanos G. Beyene

February 2001

Contents

Abstract

This paper is meant to provide techniques in building and installing a standalone packet filtering firewall in Linux machines, mainly for small sites who don't give much service to Internet users. It deals with attenuating the effect of the most common types of attacks using ipchains. It guides how to design, implement, debug, and maintain Firewall. Techniques for continuously monitoring attacks is attempted. It also give a historical, architectural and technical overview of firewalls and security attacks.

Acknowledgement

I express my great thanks to my supervisor Bjorn Victor, for the guidance, suggestions and feedback he gave me. I also want to acknowledge the assistance provided by my colleagues and friends at Uppsala University. Special thanks to my friends Peter Lindh, Jila Ashier and Marit Linnebo for their indispensable efforts for helping and motivating me in every aspect they can.

1 Introduction

1.1 Firewall History

The Internet, which started as the Advanced Research Projects Agency Network (ARPANET), was a small, almost closed, community. However, something happened that changed the Internet forever. Peter Yee at the NASA Ames Research Centre sent a note out to the Internet mailing list that reported, "We are currently under attack from an Internet VIRUS! This report was the first documentation of what was to be later called The Morris Worm. This "worm" propagated its way from one computer to another, sliding through security holes in commonly used programs.

This incident brought the need for network security into the spotlight. Suddenly, people were thinking and talking about vulnerabilities and related attacks from the bad guys in cyberspace.

These concerns were well justified. More attacks were to follow: E.g. the password-capture attack during 1994, IP-Spoofing in 1995. Furthermore in 1996, denial-of-service attacks were in the security news; in this scenario, the attacker floods networked computers with information in an attempt to cause the computers to stop functioning or be too busy to service legitimate requests.

The big changes in the Internet community raised awareness of the need for good computer and network security. As these, and other, events were unfolding, the firewall was starting its rapid evolution.

A firewall is a necessary part of the overall security of a corporate network, though alone it is insufficient to provide adequate network security for businesses connected to the Internet. The rationale for installing a firewall is almost always to protect a private network against intrusion. The purpose of an Internet firewall is to provide a single point of defence with controlled and audited access to services, between an inside network and a less trustworthy outside network. It implements a security policy. The policy might be to prevent any access from outside while allowing traffic to pass from the inside to the outside. Alternatively it might be to permit access only from certain places, from certain users, or for certain activities.

The first security firewalls were used in the early 1990s. They were IP routers with filtering rules. The next security firewalls were more elaborate and more tunable. There were firewalls built on so called bastion hosts. Probably the first commercial firewall of this type, using filters and application gateways (proxies), was from Digital Equipment Corporation.

In 1993, the Trusted Information Systems (TIS) Firewall Toolkit (FWTK)

was released in source code form to the Internet community. It provided the basis for TIS' commercial firewall product, later named Gauntlet. In 1994, Check Point followed with the Firewall-1 product, introducing "user friendliness" to the world of Internet security.

Early firewall requirements were easy to support because they were limited to the Internet services available at that time. The typical organisation or business connecting to the Internet needed secure access to remote terminal services (Telnet), file transfer (File Transfer Protocol [FTP]), electronic mail (Simple Mail Transfer Protocol [SMTP]), and USENET News (the Network News Transfer Protocol-NNTP). Today, we add to this list of "requirements" access to the World Wide Web, commerce, live news broadcasts, weather information, stock quotes, music on demand, multimedia, database access, and file sharing with new features cropping up almost daily. Many of these become must-have services, and each has its own security concerns and weaknesses. With these changes, the frequency and sophistication of Internet attacks have increased [1].

1.2 Objective

Firewalls are a very effective way to protect your system from intentional hostile intrusion that could compromise confidentiality or result in data corruption or denial of service. And since Linux is growing in popularity so quickly, security issues are a critical component of the viability of Linux. People have to be confident that Linux not only works well, but has features and related products that allow for a relatively secure environment for data. To that extent built in security features are included in Linux and great improvements are being made.

The focus of this paper is in designing and implementing a standalone packet filtering firewall for Linux using ipchains. Pros and cons of Firewalls, methods of attack, security policies, debugging and finally monitoring system integrity and intrusion detection are addressed.

1.3 Additional Protection and Future Directions

While firewalls protect the integrity and data of the network that sits behind it from outside attacks, internetworks present other dangers, for example, eavesdropping. There are many ways to protect data, and all involve cryptography.

Privacy Enhanced Mail, for example, is an Internet standard that provides message privacy through encryption and integrity checking through digital signatures. Electronic mail can be sent in such a way that only the intended recipient can read it and any tampering with the message can be detected.

If across-the-board network service encryption is needed for services like FTP or Telnet, encryption devices are employed. These devices examine every network packet before it leaves the private network, and if a packet is destined for outside networks specified by a security administrator, the data portion of the packet is encrypted. Anyone capturing that data packet as it travels over an outside network connection is unable to read it. When the packet reaches its destination, an encryption device examines it to see if it is sent from its encryption partner. If so, the data are decrypted.

As organisations broaden the base of measures and countermeasures used to implement a comprehensive network and computer security policy, firewalls will need to communicate and interact with other devices. Intrusion detection devices running on or separate from the firewall must be able to reconfigure the firewall to meet a new perceived threat (just as dynamic filtering firewalls today "reconfigure" themselves to meet the needs of a user).

Firewalls will have to be able to communicate with network security control systems, reporting conditions and events, allowing the control system to reconfigure sensors and response systems. A firewall could signal an intrusion detection system to adjust its sensitivity, as the firewall is about to allow an authenticated connection from outside the security perimeter. A central monitoring station could watch all this, make changes, react to alarms and other notifications, and make sure that all antivirus software and other content screening devices were functioning. Some products have started down this path already. The Intrusion Detection System and firewall reconfiguration of network routers based on perceived threat is a reality today. The evolution continues and firewalls are changing rapidly to address the next Internet years [10].

As the use of Internet and internetworked computers continues to grow, the use of Internet firewalls will also grow in parallel. They not be the only security mechanism, but will cooperate with others on the network.

2 Why Firewalls?

2.1 A firewall is a Focus of Security Decision

All traffic in and out passes through this single, narrow choke point. It lets you concentrate your measures on this choke point. It is far more efficient than spreading security decisions and technologies around.

Although firewalls cost tens of thousands of dollars to implement, most sites find that concentrating the most effective hardware and software at the firewall is less expensive and more effective as opposed to when it is being distributed on many hosts. In particular, one-time password systems and other add-on authentication software could be located at the firewall as opposed to each system that needed to be accessed from the Internet. It is also the only way to collect manageable network usage statistics which is helpful for detecting whether it is implementing the desired policy and detecting any intrusion.

2.2 A firewall can enforce a security policy

A firewall provides the means for implementing and enforcing a network access policy. In effect, a firewall provides access control to users and services. A firewall can greatly improve network security and reduce risks to hosts protected on the subnet by filtering inherently insecure services. As a result, the subnet network environment is exposed to fewer risks, since only selected traffic will be able to pass through the firewall. For example, a firewall could prohibit certain vulnerable services such as NFS from entering or leaving a protected subnet. This provides the benefit of preventing the services from being exploited by outside attackers, but at the same time permits the use of these services with greatly reduced risk of exploitation. Services such as NIS or NFS that are particularly useful on a local area network basis can thus be enjoyed and used to reduce the host management burden.

Firewalls can also provide protection from routing-based attacks, such as source routing and attempts to redirect routing paths to compromised sites via ICMP redirects. A firewall could reject all source-routed packets and ICMP redirects and then inform administrators of the incidents.

This brings to the fore an access policy that firewalls are particularly adept at enforcing: do not provide access to hosts or services that do not require access. Put differently, why provide access to hosts and services that could be exploited by attackers when the access is not used or required? If, for example, a user requires little or no network access to her desktop workstation, then a firewall can enforce this policy.

2.3 A Firewall Provides Enhanced Privacy

Privacy is of great concern to certain sites, since what would normally be considered innocuous information might actually contain clues that would be useful to an attacker. Using a firewall, some sites wish to block services such as 'finger' and 'Domain Name Service'. finger displays information about users such as their last login time, whether they've read mail, and other items. But, 'finger' could leak information to attackers about how often a system is used, whether the system has active users connected, and whether the system could be attacked without drawing attention.

Firewalls can also be used to block DNS information about site systems, thus the names and IP addresses of site systems would not be available to Internet hosts. Some sites feel that by blocking this information, they are hiding information that would otherwise be useful to attackers.

2.4 A firewall can log Internet activities

If all access to and from the Internet passes through a firewall, the firewall can log accesses and provide valuable statistics about network usage. A firewall, with appropriate alarms that sound when suspicious activity occurs can also provide details on whether the firewall and network are being probed or attacked.

It is important to collect network usage statistics and evidence of probing for a number of reasons. Of primary importance is knowing whether the firewall is withstanding probes and attacks, and determining whether the controls on the firewall are adequate. Network usage statistics are also important as input into network requirements studies and risk analysis activities.

3 Methods of Attack

For better understanding of the implementation of the Linux Filtering firewalls, description of the most common methods of attack is given.

3.1 IP Spoofing Attacks

A spoofing attack involves forging one's source address. It is the act of using one machine to impersonate another. A malicious cracker may gain entry by "spoofing" the source IP address of packets sent to the firewall. The firewall may let them through if the address used is a trusted host identity. To avoid such attacks responsible management of information is essential. Moreover, this type of attack is usually combined with other types of attack to hide the identity of crackers, and makes detection and prevention hard.

Unfortunately, with the current IP protocol technology, it is impossible to eliminate IP-spoofed packets. However, it is possible to reduce the number of IP-spoofed packets entering and exiting a private networks. One method to reduce such attacks is to install a filtering router that rejects incoming packets to external interface having an internal source address or uses a reserved private network numbers or other invalid addresses. In addition, outgoing packets having a source address different from your internal network has to be blocked to prevent a source IP spoofing attack from originating from your site. These filters will not stop all spoofed attacks , since outside attackers can spoof packets from any outside network, and internal attackers can still send attacks spoofing internal addresses.

An effective measure against IP spoofing is the use of a Virtual Private Network (VPN) protocol such as IPSec. This methodology involves encryption of the data in the packet as well as the source address. The VPN software or firmware decrypts the packet and the source address and performs a checksum. If either the data or the source address have been tampered with, the packet will be dropped. Without access to the encryption keys, a potential intruder would be unable to penetrate the firewall [8].

Here are some tips for minimising spoofed attacks incoming to external interface.

- Block Broadcast addresses: The addresses to block here are network 0.0.0.0 and network 255.255.255.255
- Your local network(s)
- Reserved private network numbers. Refer [11].

3.2 Denial of Service Attacks

A Denial of Service attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. Attempts to "flood" a network, thereby preventing legitimate network traffic and attempts to prevent a particular individual from accessing a service are some examples of this type of attack. Illegitimate use of resources may also result in denial of service. For instance, an intruder may use your anonymous ftp area as a place to store illegal copies of commercial software, consuming disk space and generating network traffic. Thus Denial of service attacks can essentially disable your computer or your network and cause many organisations to suffer big financial loss. Denial of Service attacks can be classified as three basic types. They are explained briefly below [2].

3.2.1 Consumption of Scarce Resources

Computers and networks need certain things to operate: network bandwidth, memory and disk space, CPU time, data structures, access to other computers and networks are some of these resources. I try to give a brief description of such problems below.

- Network Connectivity Denial of Service attacks are most frequently executed against network connectivity. The goal is to prevent hosts or networks from communicating on the network. Typical example of this type of attack is the "TCP SYN flood" attack.

  In this type of attack, the attacker begins the process of establishing a connection to the victim machine by sending a SYN message. The server (victim machine) then acknowledges the SYN message by sending SYN-ACK message back to the client (attacking machine) but has not yet received the ACK message. This means the victim machine has reserved one of a limited number of data structures required to complete the impending connection. Normally there is a timeout on the order of minutes associated with a pending connection, so the half open connections will eventually expire and the victim server system will recover. However, the attacking system can simply continue sending IP spoofed packets requesting new connections faster than the victim system can expire the pending connections. The result is that the victim of such attack will have difficulty in accepting any new incoming connections as the intruder is consuming kernel data structures involved in establishing a network connection.

This attack can be easily accomplished with IP Spoofing which makes detection/prevention difficult. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system.

- Bandwidth Consumption: An intruder may also be able to consume all the available bandwidth on your network by generating a large number of packets directed to your network. Typically, these packets are ICMP ECHO packets, but in principle they may be anything.

- Consumption of Other Resources: In addition to network bandwidth, intruders may be able to consume other resources that your systems need in order to operate. For example, in many systems, a limited number of data structures are available to hold process information (process identifiers, process table entries, process slots, etc.). An intruder may be able to consume these data structures by writing a simple program or script that does nothing but repeatedly create copies of itself. Many modern operating systems have quota facilities to protect against this problem, but not all do. Further, even if the process table is not filled, the CPU may be consumed by a large number of processes and the associated time spent switching between processes.

  An intruder may also attempt to consume disk space by generating excessive numbers of mail messages, intentionally generating errors that must be logged or placing files in anonymous ftp areas or network shares.

  Also, many sites have schemes in place to "lockout" an account after a certain number of failed login attempts. A typical set up locks out an account after 3 or 5 failed login attempts. An intruder may be able to use this scheme to prevent legitimate users from logging in. In some cases, even the privileged accounts, such as root or administrator, may be subject to this type of attack. Be sure you have a method to gain access to the systems under emergency circumstances. Consult your operating system vendor or your operating systems manual for details on lockout facilities and emergency entry procedures.

  An intruder may be able to cause your systems to crash or become unstable by sending unexpected data over the network.

3.2.2 Destruction or Alteration of Configuration Information

An improperly configured computer may not perform well or may not operate at all. An intruder may be able to alter or destroy configuration information that prevents you from using your computer or network.

For example, if an intruder can change the routing information in your routers, your network may be disabled.

3.2.3 Physical Destruction or Alteration of Network Components

The primary concern with this type of attack is physical security. You should guard against unauthorised access to computers, routers, network wiring closets, network backbone segments, power and cooling stations, and any other critical components of your network.

Denial-of-service attacks can result in significant loss of time and money for many organisations. The following options are worthy considering to minimise such attacks:

- Implement router filters to lessen your exposure to certain denial-of-service attacks. You can guard against TCP SYN flooding.
- Disable any unused or unneeded network services. This can limit the ability of an intruder to take advantage of those services to execute a denial-of-service attack.
- Enable quota systems for all accounts on your operating system if these services are available. In addition, if your operating system supports partitions or volumes (i.e., separately mounted file systems with independent attributes) consider partitioning your file system so as to separate critical functions from other activity.
- Observe your system performance and establish baselines for ordinary activity. Use the baseline to gauge unusual levels of disk activity, CPU usage, or network traffic.
- Routinely examine your system performance and configuration information to detect unexpected changes. Establish and maintain regular backup schedules and policies, particularly for important configuration information. You also need to examine your physical security. Consider servers, routers, unattended terminals, network access points and other components of your system.

Many organisations can suffer financial loss as a result of a denial-of-service attack and may wish to pursue criminal or civil charges against the intruder.

3.3 Spoofed/Forged Email

Email spoofing occurs when a user receives email that appears to have originated from one source when it actually was sent from another source. Email spoofing is often an attempt to trick the user into making a damaging statement or releasing sensitive information (such as passwords).

One such example of spoofed email that could affect the security of your site is: email claiming to be from a system administrator requesting users to change their passwords to a specified string and threatening to suspend their account if they do not do this.

It is easy to spoof email because SMTP (Simple Mail Transfer Protocol) lacks authentication. If a site has configured the mail server to allow connections to the SMTP port, anyone can connect to the SMTP port of a site and issue commands that will send email that appears to be from the address of the individual's choice. In addition to connecting to the SMTP port of a site, a user can send spoofed email via other protocols (for instance, by modifying their web browser interface).

Here are some tips for preventing Spoofed Email.

- Use cryptographic signatures (e.g., PGP "Pretty Good Privacy") to exchange authenticated email messages. Authenticated email provides a mechanism for ensuring that messages are from whom they appear to be, as well as ensuring that the message has not been altered in transit.
- Configure your mail delivery daemon to prevent someone from directly connecting to your SMTP port to send spoofed email to other sites.
- Ensure that your mail delivery daemon allows logging and is configured to provide sufficient logging to assist you in tracking the origin of spoofed email.
- Configuring your firewall so that SMTP connections from outside your firewall must go through a central mail hub. A single point of entry will provide you with centralised logging, which may assist in detecting the origin of mail spoofing attempts to your site.
- Educate your users to avoid disclosing sensitive information (such as passwords) and report any such activities to the appropriate system administrator(s) as soon as possible [2].

3.4 Email Bombing and Spamming

Email bombing is characterised by abusers repeatedly sending an identical email message to a particular address.

Email spamming is a variant of bombing; it refers to sending email to hundreds or thousands of users. Email spamming can be made worse if recipients reply to the email, causing all the original addressees to receive the reply. It may also occur innocently, as a result of sending a message to mailing lists and not realizing that the list explodes to thousands of users, or as a result of an incorrectly set-up auto-responder message.

Email bombing/spamming may be combined with email spoofing making it more difficult to determine who the email is actually coming from.

If your email system looks slow or email doesn't appear to be sent or received, the reason may be that your mailer is trying to process a large number of messages. When large amounts of email are directed to or through a single site, the site may suffer a denial of service through loss of network connectivity, system crashes, or failure of a service because of:

- overloading network connections
- using all available system resources
- filling the disk as a result of multiple postings and resulting syslog entries

Unfortunately Email spamming is almost impossible to prevent because a user with a valid email address can "spam" any other valid email address, newsgroup, or bulletin-board service. Here are some ways which minimise the impact of this type of attack.

- Develop in-house tools to help you recognise and respond to the email bombing/spamming and so minimise the impact of such activity.
- SMTP-based attack can also be minimised by making sure that your firewall can allow SMTP connections only to your central email hubs incase there are more email servers
- Ensure you are up to date with the most current version of email delivery daemon (sendmail, for example) and increase logging capabilities as necessary to detect or alert you to such activity.

3.5 Exploitation of known Weaknesses in Programs

Operating systems that are relatively new to IP networking tend to be more problematic, as more mature operating systems have had time to find and eliminate their bugs. An attacker can often make the target equipment continuously reboot, crash, lose the ability to talk to the network, or replace files on the machine. Here, running as few operating system services as possible can help. And, of course, choosing a stable operating system will help here as well. When selecting an OS, don't be fooled into believing that "the pricier, the better". Free operating systems are often much more robust than their commercial counterparts. With Open Sources, it is possible to repair the weaknesses in the software [3].

Various Internet services such as web servers, mail servers, and other contain bugs that allow remote Internet users to do things ranging from gaining control of the machine to making that application crash and just about everything in between. The exposure to this risk can be reduced by disabling vulnerable services, running only necessary services, and using products that have been around a while.

3.6 Source Routed Traffic

Normally, the route a packet takes from its source to its destination is determined by the routers between the source and destination. The packet itself only says where it wants to go, and nothing about how it expects to get there. However there is an option for the sender of a packet to include information in the packet that tells the route the packet should take to get to its destination. This is called source routing. For a firewall, source routing is dangerous, since an attacker can generate traffic claiming to be from a system inside the firewall. In general, such traffic wouldn't route to the firewall properly, but with the source routing option, all the routers between the attacker's machine and the target will return traffic along the reverse path of the source route.

Source routing is mostly used in debugging network problems or routing traffic over specific links for congestion control for specialised situations. Most commercial routers incorporate the ability to block source routing otherwise firewall must be built with source routing blocked at some point [3].

3.7 Eavesdropping

This is the simplest type of attack. A host is configured to to "listen" to and capture data not belonging to it. Carefully written eavesdropping programs can

take usernames and passwords from user login network connections. Broadcast networks like Ethernet are especially vulnerable to this type of attack. IP firewalling is not helpful in avoiding eavesdropping. To protect against this type of threat enforce the use of data encryption [5].

IPSEC (IP SECurity), a set of standards developed by the Internet Engineering Task Force (IETF), is a good way for preventing this type of attack. It enables host-to-host authentication (which will let hosts know that they're talking to the hosts they think they are) and encryption (which will prevent attackers).

4 Drawbacks with Firewalls

4.1 Firewalls are not enough to Secure a Network

Firewall is an integral part of any security program, but it is not a security program in and of itself. Security involves data integrity (has it been modified?), data confidentiality (has anyone seen it?) and authentication (are they really who they say they are?). But Firewalls only address the issues of data integrity, confidentiality and authentication of data that is behind the firewall. Any data that transits outside the firewall is subject to factors out of the control of the firewall [8].

It is therefore necessary to have a well planned and strictly implemented security program that includes but is not limited to firewall protection.

4.2 Large Potential for Back Doors

A firewall can effectively control the traffic that passes through it but it don't protect against back doors into the site. For example, if unrestricted modem access is still permitted into a site protected by a firewall, attackers could effectively jump around the firewall through such a modem.

4.3 Little Protection from malicious Insiders

Firewalls generally do not provide protection from insider threats. While a firewall may be designed to prevent outsiders from obtaining sensitive data, the firewall does not prevent an insider from copying the data onto a tape and taking it out of the facility. Thus, it is faulty to assume that the existence of a firewall provides protection from insider attacks or attacks in general that do not need to use the firewall. It is perhaps unwise to invest significant resources in a firewall if other avenues for stealing data or attacking systems are neglected.

4.4 Computer Virus

Firewalls can not protect against users downloading virus-infected personal computer programs from Internet archives or transferring such programs in attachments to e-mail. There are too many ways of encoding binary files for transfer over networks, and too many different architectures and viruses to try to search for them all. In general, a firewall cannot protect against a data-driven attack– attacks in which something is mailed or copied to an internal host where it is then executed.This form of attack has occurred in the past against various versions of sendmail, ghostscript, and others. The most practical way to address

the virus problem is through host-based virus protection software, and user education concerning the dangers of Viruses and precautions to take against them [3].

4.5 Firewalls might slow transfer rate of data

Firewalls represent a potential bottleneck, since all connections must pass through the firewall and, in some cases, be examined by the firewall. However, this is generally not a problem today, as firewalls can pass data at T1 (1.5 Megabits/second) rates and most Internet sites are at connection rates less than or equal to T1.

5 Types of Firewalls

Conceptually, two types of Internet firewalls exist.

5.1 Packet Filtering Firewalls

A Packet Filtering firewalls is normally implemented within the operating system and screens packets based at the IP network layer. It protects the system by making decisions after filtering packets based on information in the IP packet header[8]. The router filters IP packets based on some or all of the following fields

- The address the data is (supposedly) coming from
- The address the data is going to
- The session and application ports being used to transfer the data
- state tracking and/or protocol checking which include rules like:
  - Let incoming TCP packets through only if they are responses to outgoing TCP packets you have seen
  - Disconnect any FTP connection where the remote username is "anonymous"/ Do not allow HTTP transfer to these sites.

Below are some advantages of packet Filtering Firewalls.

- Because very little data is analysed and logged, filtering firewalls take less CPU and create less latency in your network.
- Filtering firewalls are more transparent to the user. The user does not have to setup rules in their applications to use the Internet. With most proxy servers this is not true.

Here are some weakness of this type of firewall

- Filtering firewalls do not provide for password controls. Users can not identify themselves. The only identity a user has is the IP number assigned to their workstation. This can be a problem if you are going to use Dynamic IP assignments. This is because rules are based on IP numbers: you will have to adjust the rules as new IP numbers are assigned.
- Packet filtering rules are complex to specify and usually no testing facility exists for verifying the correctness of the rules. For example, it is relatively straightforward to specify a rule to block all inbound connections to port

23 (the TELNET server). If exceptions are made, i.e., if certain site systems need to accept TELNET connections directly, then a rule for each system must be added.

- Another problem is that a number of RPC (Remote Procedure Call) services are very difficult to filter effectively because the associated servers listen at ports that are assigned randomly at system startup. Since the router cannot be told which ports the services reside at, it isn't possible to block completely these services unless one blocks all UDP packets (RPC services mostly use UDP). But blocking all UDP would block potentially necessary services such as DNS. Thus, blocking RPC results in a dilemma.

The implementation on this paper is based on this type of firewall.

5.2 Application Gateways/Proxy-Based Firewalls

Application gateways or proxy-based firewalls operate at the application level and can examine information at the application data level. It is usually implemented as separate applications for each service being proxied. Each proxy application appears to be the server to the client program, and appears to be the client to the real server. Specially configured client programs, connect to the proxy server instead of a remote server. The proxy establishes the connection to the remote server on the clients application's behalf, after substituting the client's source address with its own. Proxy applications can ensure data integrity-that is, the data appropriate to the service is being exchanged, filter against viruses, and enforce high-level access control policies [8].

Application gateways have a number of general advantages over the default mode of permitting application traffic directly to internal hosts. Below are some:

- information hiding, in which the names of internal systems need not necessarily be made known to outside systems, since the application gateway may be the only host whose name must be made known to outside systems,

- robust authentication and logging, in which the application traffic can be pre-authenticated before it reaches internal hosts and can be logged more effectively. It is also possible to use a one-time password.

- less-complex filtering rules, in which the rules at the packet filtering router will be less complex than they would if the router needed to filter application traffic and direct it to a number of specific systems. For instance some FTP application gateways include the capability to block all puts to the

anonymous FTP server; this would ensure that nothing can be uploaded to the server and would provide a higher degree of assurance than relying only on file permissions at the anonymous FTP server to be set correctly.

6 Firewall Architecture

There are various ways to put firewall components together. In this part of my thesis I will describe the most common types of firewall architectures. However, there is a good deal of flexibility in how you can configure and combine firewall components to best suit your hardware, your budget, and your security policy.

6.1 Screening Router

It is possible to use a packet filtering system by itself as a firewall to protect an entire network. It is the most common and easiest to employ for small, uncomplicated sites. It is a low cost system, since you only need to configure packet filtering rules in your router to block or filter protocols and addresses. The site systems usually have direct access to the Internet while all or most access to site systems from the Internet is blocked (Fig 1). However, the router could allow selective access to systems and services, depending on the policy. Usually, inherently-dangerous services such as NIS, NFS, and X Windows are blocked [10].

Below are some disadvantages of a packet filtering firewall in comparison to other architectures

- No way to be sure that what's coming in on a given port is the actuall protocol you wanted to allow.
- If complex filtering rules are required, the filtering rules may become unmanageable. Besides packet filtering rules are often difficult to test thoroughly, which may leave a site open to untested vulnerabilities,
- It gives you no depth of defence. If the router is compromised, you have no further security. Hence each host directly accessible from the Internet will require its own copy of advanced authentication measures.

6.2 Dual-Homed Gateway Firewall

A dual-homed host architecture is built around the dual-homed host computer, a computer that has at least two network interfaces, and with the host's IP forwarding capability disabled. see Figure 2. Unlike the packet filtering firewall, the dual-homed gateway is a complete block to IP traffic between the Internet and protected site. Services and access is provided by proxy servers on the gateway. It is a simple firewall, yet very secure [10].

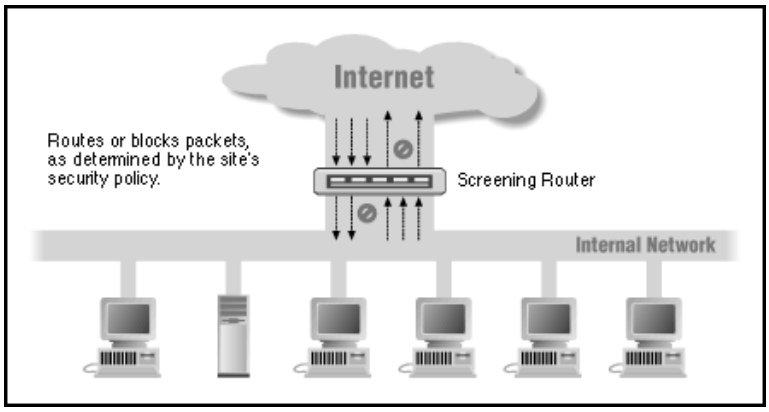

Figure 1: Screening router. From [10].

This type of firewall implements the "deny all services design policy unless they are specifically permitted" since no services pass except those for which proxies exist. Dual-homed hosts can provide a very high level of control. The names and IP addresses of site systems would be hidden from Internet systems, because the firewall would not pass DNS information. In addition because it uses a host system, the firewall can house software to require users to use authentication tokens or other advanced authentication measures. The firewall can also log access and log attempts or probes to the system that might indicate intruder activity.

A simple setup for a dual-homed gateway would be to provide proxy services for TELNET and FTP, and centralised e-mail service in which the firewall would accept all site mail and then forward it to site systems.

Although the dual-homed gateway is a better alternative to packet filtering router firewalls, it has its own weakness:

- a dual-homed host isn't a high performance device. A dual-homed host has more work to do for each connection than a packet filter does, and correspondingly needs more resources.

- Since a dual-homed host is a single point of failure, it's important to make certain that its host security is absolutely impeccable. An attacker who can compromise the dual-homed host has full access to your site and can cut you off from the Internet. This makes dual-homed hosts inappropriate if being able to reach the Internet is critical to your business.

- Proxying is much less problematic but may not be available for all services you're interested in.

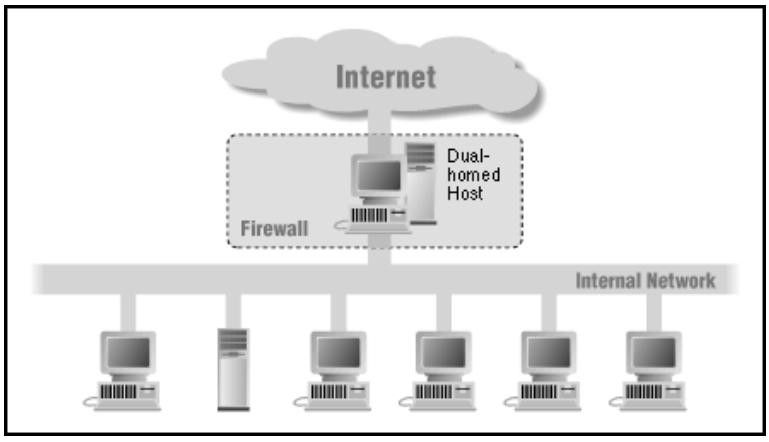

Figure 2: Dual-homed gateway firewall. From [10].

A dual-homed host can provide services only by proxying them, or by having users log into the dual-homed host directly. Yet it is preferable to avoid having users log into the dual-homed host directly as users might bring significant security problems. For instance users may unexpectedly enable services you consider insecure. Furthermore, most users find it inconvenient to use a dual-homed host by logging into it.

Proxying is much better at supporting outbound services, internal users using resources on the Internet, than inbound services, users on the Internet using resources on the internal network. In a dual-homed host configuration, you will normally have to provide services to the Internet by running them on the dual-homed host. But this is not usually advisable because providing services to the Internet is risky, and the dual-homed host is a security-critical machine that you don't want to put risky services on. It might be acceptable to put a minimally functional web server on the dual-homed host but it would clearly be extremely dangerous to provide a normal web server there.

6.3 Screened Host Architecture

The screened host firewall is a more flexible firewall than the dual-homed gateway firewall, however the flexibility is achieved with some cost to security. The screened host firewall is often appropriate for sites that need more flexibility than that provided by the dual-homed gateway firewall.

The screened host firewall combines a packet-filtering router with an application gateway (Bastion host) located on the protected subnet side of the router. The application gateway's proxy services would pass TELNET, FTP, and other

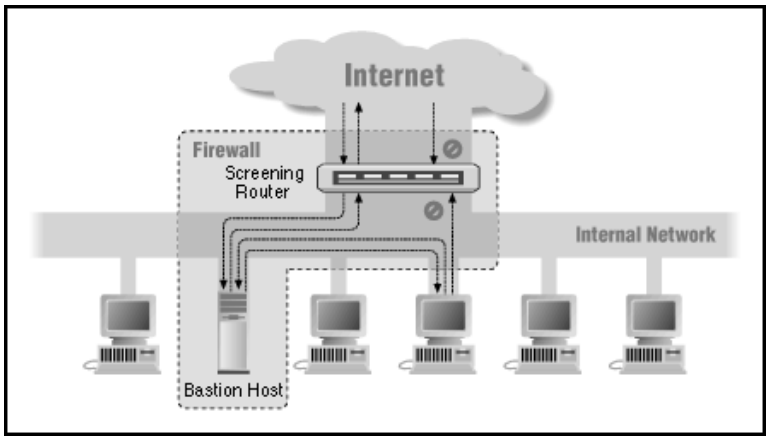

Figure 3: Screened Host Architecture. From [10].

services for which proxies exist, to site systems. The router filters or screens inherently dangerous protocols from reaching the application gateway and site systems. In this architecture, the primary security is provided by packet filtering. For example, packet filtering is what prevents people from going around proxy servers to make direct connections.

Figure 3 shows a simple version of a screened host architecture. The bastion host sits on the internal network. The packet filtering on the screening router is set up in such a way that the bastion host is the only system on the internal network that hosts on the Internet can open connections to. Even then, only certain types of connections are allowed. Any external system trying to access internal systems or services will have to connect to this host. The bastion host thus needs to maintain a high level of host security.

The packet filtering also permits the bastion host to open allowable connections depending on your security policy to the outside world. The packet filtering configuration in the screening router may allow other internal hosts to open connections to hosts on the Internet for certain needed services. These services might be those for which proxy services don't exist, and might be trusted in the sense that the risk of using the services has been considered and found acceptable. You can mix and match these approaches for different services; some may be allowed directly via packet filtering, while others may be allowed only indirectly via proxy. It all depends on the particular policy your site is trying to enforce.

The major disadvantage of Screened host Architecture is that if an attacker manages to break in to the bastion host, there is nothing left in the way of

network security between the bastion host and the rest of the internal hosts. The router also presents a single point of failure; if the router is compromised, the entire network is available to an attacker. For this reason, the screened subnet architecture (to be explained next) has become increasingly popular[5].

6.4 Screened Subnet Architecture

The screened subnet architecture adds an extra layer of security to the screened host architecture by adding a perimeter network that further isolates the internal network from the Internet. In a screened host architecture, your internal network is wide open to attack from your bastion host which makes your bastion host a very tempting target. There are no other defences between it and your other internal machines. If someone successfully breaks into the bastion host in a screened host architecture, he's hit the jackpot [5].

By isolating the bastion host on a perimeter network, you can reduce the impact of a break-in on the bastion host. With the simplest type of screened subnet architecture, there are two screening routers, each connected to the perimeter net. One sits between the perimeter net and the internal network, and the other sits between the perimeter net and the external network (usually the Internet). To break into the internal network with this type of architecture, an attacker would have to get past both routers. Even if the attacker somehow broke in to the bastion host, he'd still have to get past the interior router. There is no single vulnerable point that will compromise the internal network.

Some sites go so far as to create a layered series of perimeter nets between the outside world and their interior network. Less trusted and more vulnerable services are placed on the outer perimeter nets, farthest from the interior network. The idea is that an attacker who breaks into a machine on an outer perimeter net will have a harder time successfully attacking internal machines because of the additional layers of security between the outer perimeter and the internal network.

Figure 4 shows a possible firewall configuration that uses the screened subnet architecture. The next few sections describe briefly the components in this type of architecture.

6.4.1 Perimeter network

The perimeter network is another layer of security, an additional network between the external network and your protected internal network. If an attacker successfully breaks into the outer reaches of your firewall, the perimeter net

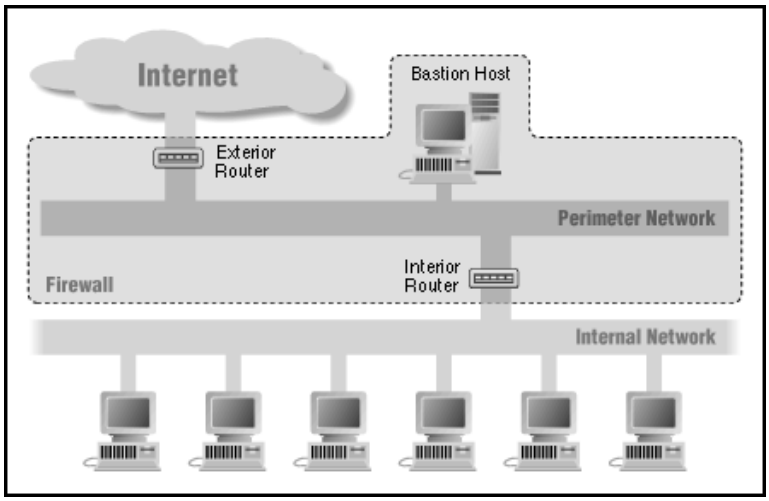

Figure 4: Screened subnet architecture. From [10].

offers an additional layer of protection between that attacker and your internal systems.

With a perimeter network, if someone breaks into a bastion host on the perimeter net, he'll be able to snoop only on traffic on that net. All the traffic on the perimeter net should be either to or from the bastion host, or to or from the Internet. Because no strictly internal traffic (that is, traffic between two internal hosts, which is presumably sensitive or proprietary) passes over the perimeter net, internal traffic will be safe from prying eyes if the bastion host is compromised.

6.4.2 Bastion host

With the screened subnet architecture, you attach a bastion host (or hosts) to the perimeter net; this host is the main point of contact for incoming connections from the outside world; for example: it forwards incoming FTP connections to the site's anonymous FTP server.

Outbound services are handled in either of these ways:

- Set up packet filtering on both the exterior and interior routers to allow internal clients to access external servers directly.
- Set up proxy servers to run on the bastion host (if your firewall uses proxy software) to allow internal clients to access external servers indirectly. You would also set up packet filtering to allow the internal clients to talk to

the proxy servers on the bastion host and vice versa, but to prohibit direct communications between internal clients and the outside world.

6.4.3 Interior router

The interior router does most of the packet filtering for your firewall. It allows selected services outbound from the internal net to the Internet. These services are the services your site can safely support and safely provide using packet filtering rather than proxies.

The services the interior router allows between your bastion host (on the perimeter net itself) and your internal net are not necessarily the same services the interior router allows between the Internet and your internal net. You should limit the services allowed between the bastion host and the internal net to just those that are actually needed. You should further limit services, to the extent possible, by allowing them only to or from particular internal hosts; for example, SMTP might be limited only to connections between the bastion host and your internal mail server or servers. The reason for limiting the services between the bastion host and the internal network is to reduce the number of machines that can be attacked from the bastion host, should it be compromised.

6.4.4 Exterior router

In theory the exterior router protects both the perimeter net and the internal net from the Internet. In practice, exterior routers tend to allow almost anything outbound from the perimeter net, and they generally do very little packet filtering. The packet filtering rules to protect internal machines would need to be essentially the same on both the interior router and the exterior router.

The only packet filtering rules that are really special on the exterior router are those that protect the machines on the perimeter net.

One of the security tasks that the exterior router can usefully perform is, a task that usually can't easily be done anywhere else, the blocking of any incoming packets from the Internet that have forged source addresses. Such packets claim to have come from within the internal network, but actually are coming in from the Internet. The interior router can't tell if packets that claim to be from the perimeter net are forged.

6.5 Internal firewalls

Most of the time Firewalls are built to protect your internal network from the Internet. However, in some situations, you may also be protecting parts of your

internal network from other parts. There are a number of reasons why you might want to do this:

- You have networks that are more secure than the rest of your site, such as secret development project networks or networks where financial data or grades are passed around.
- You have test or lab networks with strange things going on there.
- You have networks that are less secure than the rest of your site, such as demonstration or teaching networks where outsiders are commonly present.

It often makes sense to keep one part of your organisation separate from another. Not everyone in an organisation needs the same services or information, and security is frequently more important in some parts of an organisation (the accounting department, for example) than in others. Many of the same tools and techniques you use to build Internet firewalls are also useful for building these internal firewalls.

6.6 Integrating Modem Pools with Firewalls

Many sites permit dial-in access to modems located at various points throughout the site. This could be a potential backdoor and could negate all the protection provided by the firewall. A much better method for handling modems is to concentrate them into a modem pool, and then secure connections from that pool.

The modem pool likely would consist of modems connected to a terminal server, which is a specialised computer designed for connecting modems to a network. A dial-in user connects to the terminal server, and then connects (e.g., telnets) from there to other host systems. Some terminal servers provide security features that can restrict connections to specific systems, or require users to authenticate using an authentication token. Alternatively, the terminal server can be a host system with modems connected to it [10].

Dual-homed gateway and screened subnet firewalls provide a more secure method for handling modem pools. The terminal server gets located on the inner, screened subnet, where access to and from the modem pool can be carefully controlled by the routers and application gateways. The router on the Internet side protects the modem pool from any direct Internet access except from the application gateway.

A site must rigidly enforce a policy that prevents any users from connecting modems elsewhere on the protected subnet. Even if the modems contain security features, this adds more complexity to the firewall protection scheme and adds another "weak link" to the chain.

7 Firewall Design

In the previous sections, it is explained that firewalls can be put together in different ways. In choosing the right strategy, you need to design a security policy that meets the needs of the organisation with tolerable risk. You have to analyse both your needs and your budget before you start to look at Firewall products. Otherwise you may be influenced by advertising rather than by your own needs.

The first step in securing your system is needs analysis with risk assessment.Here are few tips [8].

- Determine outbound access policy: list the set of services allowed to pass from inside to outside Here it is possible to offer different services for different groups of users.
- Determine inbound access policy: specify any services you want to offer to Internet users. For instance, you might want to give web service.
- Determine if dial-in or dial-out access is required: Dial-in requires a secure remote access PPP server that should be placed outside the firewall. If dial-out access is required by certain users, individual dial-out computers must be made secure in such a way that hostile access to the LAN through the dial-out connection becomes impossible.
- Besides the level of risk you are willing to accept should be clearly defined. What will happen if you are cut off from the Internet? Is it critical or just inconvenient?

Careful analysis is needed when deciding which services and daemons to run on the firewall machine. Each service has its own security considerations. It is important to understand a network service, what it does, and who it's intended for before you run it - especially on a machine connected directly to the Internet.

Once it is determined which services to run on the firewall machine, the next job is to consider your budget. Here you must put into consideration the cost of maintaining the firewall as well. It's important, in other words, to evaluate firewalls not only in terms of what they cost now, but continuing costs such as support. You can then decide whether to buy or implement [3].

There are two default firewall policies. These are: "that which is not expressly forbidden is permitted" and "that which is not expressly permitted is forbidden". Security experts strongly counsel for the latter [6].

At the beginning, it is advisable to create a simple and clear security policy. But your policy might become more complicated with time as your needs increase.

8 Setting up Linux Filtering Firewall

8.1 Packet Filtering

Packet-filtering is a mechanism with traffic screening rules enforcing local policies. It decides whether to route a packet through to its destination, to silently throw the packet away, or reject the packet and return an error message to the sending address based on the designed policies to sort packet header fields. It is also possible to log information about the packet.

Every information that crosses the Internet has to go into a packet at some point. As such it is possible to enforce security policy on every information that crosses your net using Packet Filtering.

A packet refers to an Internet Protocol (IP) network message. It is a piece of information that is sent across a network. Structurally, a packet contains an information header and a message body containing the data being transferred.

There are three types of IP protocol message types:

- An ICMP (Internet Control Message Protocol) packet is a network-level, IP control and status message. ICMP messages contain information about the communication between the two end-point computers.
- A UDP (User Datagram Protocol) IP packet carries data between two network-based programs, without any guarantees regarding successful delivery or packet delivery ordering.
- A TCP (Transmission Control Protocol) IP packet carries data between two network-based programs, as well, but the packet header contains additional state information for maintaining an ongoing, reliable connection. Most Internet services use the TCP communication protocol.

All IP packet headers contain the source and destination IP addresses and the type of IP protocol message, (ICMP, UDP or TCP) this packet contains. Beyond this, a packet header contains slightly different fields depending on the protocol type. ICMP packets contain a type field identifying the control or status message, along with a second code field for defining the message more specifically . UDP and TCP packets contain source and destination service port numbers. TCP packets contain additional information about the state of the connection and unique identifiers for each packet [9].

Every firewall rule has a default Packet Filtering policy. If a packet doesn't match any rule, the default policy is applied. There are two basic approaches:

- Deny everything by default and explicitly allow selected packets,

- Allow Everything by default and explicitly deny selected packets

It is important to know that IP Packet Filtering is a network layer facility. This means that it doesn't understand anything about the application using the network connections, only about the connections themselves. For example, you may deny users access to your internal network on the default telnet port, but if you rely on IP filtering alone, you can't stop them from using the telnet program with a port that you do allow to pass through your firewall. You can prevent this sort of problem by using a proxy server for each service that you allow across your firewall. The proxy servers understand the application they were designed to proxy and can therefore prevent abuse, such as using the telnet program to get past a firewall by using the World Wide Web port.

8.2 Linux for Firewalling

To build a packet filtering firewall under Linux, no special software is required. Linux supports IP Firewalling tools.In all production kernels prior to the 2.2 series, 'ipfwadm' utility tool is used. The 2.2.x kernels marked the release of a program similar to 'ipfwadm' called 'ipchains'. Linux kernels 2.3.15 and later support the Linux IP firewall called 'netfilter'. The netfilter is a multifaceted product providing direct backward-compatible support for both 'ipfwadm' and 'ipchains' as well as a new alternative command called 'iptables' [5].

To build a Linux IP firewall, it is necessary to have a kernel built with IP firewall support and the appropriate configuration utility. For more information on how to compile the kernel refer to [4].

The kernel starts with three lists of rules; these lists are called firewall chains. The three chains are called input, output and forward. When a packet comes in the kernel uses the input chain to decide its fate. If it survives that step, then the kernel decides where to send the packet next. If it is destined for another machine, it consults the forward chain. Finally, just before a packet is to go out, the kernel consults the output chain.

A chain is a checklist of rules. If the rule doesn't match the packet, then the next rule in the chain is consulted. Finally, if there are no more rules to consult, then the kernel looks at the chain policy to decide what to do. In a security-conscious system, this policy usually tells the kernel to reject or deny the packet. The implementation in this thesis is based on 'ipchains' utility. For detailed information on the 'ipchains' command syntax refer [7].

8.3 Configuring Firewalling Chains

The implementation uses a Packet Filtering mechanism, and it is installed on a gateway machine with two network interfaces (called a dual-homed gateway). One of the interfaces (external interface) is connected to the Internet and the other (internal interface) is connected to the LAN. The host name of this machine is 'asmara'. It is a pentium II with 64 megabyte of memory and 2 GB partitined hard disk.

The external interface 'eth0' is assigned a valid Class C IP number, 130.238.12.197. The LAN uses some of the reserved Class C IP addresses. The internal interface 'eth1' on the dual homed gateway is assigned an IP number of 192.168.1.1. On this net it is possible to connect 254 computers. I have two laptops in the LAN. Their IP number is 192.168.1.2 and 192.168.1.3. They don't have direct TCP/IP traffic with other networks.

IPMASQ, an advanced form of NAT (Network Address Translation) supported by Linux, is used to allow all hosts on a private network to use the Internet at the price of the valid IP number in 'asmara'. The dual-homed system(asmara) masquerades the LAN traffic. This makes the machine act as a simple screened-subnet firewall due to the proxying gateway effect of IP masquerading.

For LAN setup information refer to [5].

Throughout the configuration, the following symbolic constants are used to make things more readable.

For more information on IP address assignment, see [11]. Internet Assigned Numbers Authority(IANA) is responsible for the allocation and registration of IP address assignments. Port number-to-service mappings are also coordinated by IANA. Refer the same site.

```
PATH="/sbin"
LOOPBACK_INTERFACE="lo"            # loopback interface
LOCAL_INTERFACE="eth1"             # internal LAN interface
EXTERNAL_INTERFACE="eth0"          # Internet connected interface
ASMARA="130.238.12.197"            # asmara IP address
LAN="192.168.1.0/24"               # LAN (private network)
NAME_SERVER1="130.238.8.10"        # primary nameserver used by asmara
NAME_SERVER2="130.238.12.42"       # secondary nameserver used by asmara
NAME_SERVER3="130.238.4.133"       # third nameserver used by asmara
NEWS_SERVER="news.uu.se"           # News server of the university
LOOPBACK="127.0.0.0/8"             # reserved loopback address range
```

```
CLASS_A="10.0.0.0/8"                # class A private networks
CLASS_B="172.16.0.0/12"             # class B private networks
CLASS_C="192.168.0.0/16"            # class C private networks
CLASS_D_MULTICAST="224.0.0.0/4"     # class D multicast addresses
CLASS_E_RESERVED="240.0.0.0/5"      # class E reserved addresses
BROADCAST_SRC="0.0.0.0"             # broadcast source address
BROADCAST_DEST="255.255.255.255"    # broadcast destination address
PRIV_PORTS="0:1023"                 # privileged port range
UNPRI_PORTS="1024:65535"           # unprivileged port range
XWINDOW_PORTS="6000:6063"           # X windows port range
SSH_PORTS="513:65535"               # SSH port range
```

8.3.1 Flushing the Chain

The first thing to be done is to flush any existing rules from all chains. Otherwise, any new rules you define would be added to the end of existing rules. Packets could easily match a preexisting rule and this might produce undesired results. The following command flushes the rules of all three built in chains: input, output, and forward.

```
ipchains -F
```

Now the chains are empty and the system is in its default accept-everything policy state.

8.3.2 Defining the Default Policy

It is advisable to consider blocking everything by default, and only specifically allowing what services you need on a case-by-case basis when security is critical. This policy makes the job much easier as it is only required to worry about every security problem of services which are allowed to run instead of having to worry about every security problem with every product and service around.

The command '-P' is used to Change the policy for the built-in chains: input, output and forward. The target 'DENY' tell the kernel to silently drop a matching packets without any notification to the remote sender. Simillarly the target 'REJECT' drops a matching packets but it generates an ICMP error message telling the destination is unreachable. Denying a remote connection gives no information to attackers.

```
ipchains -P input    DENY
ipchains -P output   REJECT
ipchains -P forward  REJECT
```

Now all network traffic is blocked.

8.3.3 Enabling Loopback Interface

Unrestricted access is needed through the loopback interface. This allows to run any local network-based services, such as the X Windows system, without having to worry about getting all the firewall rules specified.

The command 'A' is used to append a new rule to a chain. The '-i' option specifies the name of an interface (physical device the packet came in on, or is going out on) to match. And the target 'ACCEPT' allows the packet through. ipchains use '-j' for target specification.

```
ipchains -A input -i $LOOPBACK_INTERFACE -j ACCEPT
ipchains -A output -i $LOOPBACK_INTERFACE -j ACCEPT
```

System logging, X-Windows, and other local, socket based services are now available.

8.3.4 Enable Unlimited LAN access to the Firewall's Internal Network

For small sites direct access to the Firewall machine from the Internal LAN is okay to allowed.The next set of rules allows Unlimited traffic within the local network. The flag '-s' and '-d' are used to specify source and destination IP addresses respectively.

```
ipchains -A input  -i $LOCAL_INTERFACE -s $LAN -j ACCEPT
ipchains -A output -i $LOCAL_INTERFACE -d $LAN -j ACCEPT
```

8.3.5 Enabling LAN access to the Internet: IP Forwarding and Masquerading

Communication between the LAN and the Internet is a two-step process. It needs to be forwarded and masqueraded.

IP forwarding is a kernel service allowing the Linux machine to act as a router between two networks, forwarding traffic from one network to the other. With a LAN, IP forwarding must be enabled in the routing section of the network configuration and local packets permitted.

IP addresses taken from the Class A, B, C private address ranges require IP masquerading (another kernel service) or application-level proxying to substitute the private LAN IP address with the valid IP address of the firewalls

machine's external interface. Actually this adds extra security by transparently isolating internal machines from the Internet. The first command turns on forwarding capability and the second one masquerades.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
ipchains -A forward -i $EXTERNAL_INTERFACE -s $LAN -j MASQ
```

8.3.6 Anti-Spoofing

IP spoofing is a technique where a host sends out packets which claim to be from another host. Since packet filtering makes decisions based on this source address, IP spoofing is used to fool packet filters.
The Linux kernel offers support against incoming spoofed packets in addition to what can be done at the firewall level. To do that, Source Address Verification should be turned on and get spoof protection before any network interfaces are initialised. TCP SYN Cookies protection should also be enabled, in case it is not done.

SYN cookies is built in the TCP stack of a Linux. It protects the Linux box. It is an implementation of TCP which respond to TCP SYN request with a cookies to interdict half-open TCP connections.

```
echo 1 >/proc/sys/net/ipv4/tcp_syncookies
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do        #do source validation
  echo 1 > $f #by reverse path
       done
```

At the packet-filtering level, it is possible to minimise IP spoofing. Some addresses are recognised with certainty as a forgery.

- Deny incoming packets to your external interface claiming to be from you you can have the matching packet logged using the The '-l' flag can be used to have matching packet logged. It is a useful feature for analysing exceptional events.

  ```
  ipchains -A input -s $ASMARA -j DENY -l
  ```

- Deny incoming packets claiming to be from the reserved IP addresses set aside in each of the class A, B, or C private network address.

  ```
  ipchains -A input -i $EXTERNAL_INTERFACE   -s $CLASS_A  -j DENY
  ipchains -A input -i $EXTERNAL_INTERFACE   -s $CLASS_B  -j DENY
  ipchains -A input -i $EXTERNAL_INTERFACE   -s $CLASS_C  -j DENY
  ```

- Block broadcast packets containing illegal source or destination broadcast addresses. The address 255.255.255.255 is reserved for broadcast destination address and 0.0.0.0 is reserved as a broadcast source address.

```
ipchains -A input   -s $BROADCAST_DEST -j DENY -l
ipchains -A input   -d $BROADCAST_SRC -j DENY -l
```

- Refuse special addresses defined as reserved by the IANA(Internet Assigned Numbers Authority). Some reserved addresses are not included as reserved blocks are being allocated more often. For updates refer [11].

```
#Can't be blocked for DHCP users.
ipchains -A input   -s 0.0.0.0/8 -j DENY -l
#LoopBack
ipchains -A input   -s 127.0.0.0/8 -j DENY -l
#Link Local Networks
ipchains -A input   -s 169.254.0.0/16 -j DENY -l
#TEST-NET
ipchains -A input   -s 192.0.2.0/24 -j DENY -l
# Classes D, E,and unallocated.
ipchains -A input   -s 224.0.0.0/3 -j DENY -l
```

8.3.7 Filtering ICMP Control and Status Messages

ICMP control messages are generated in response to a number of error conditions, and they are produced by network analysis programs, such as 'ping' and 'traceroute'.

Four ICMP message types need to pass through the firewall:Source Quench, Parameter Problem, incoming Destination Unreachable, and outgoing Destination unreachable subtype Fragmentation. The other four ICMP message types: Echo request, Echo Reply, other outgoing Destination Unreachable subtypes, and Time Exceeded are optional.

- Source Quench, ICMP message type 4, is sent when a connection source, usually a router, is sending data faster than the next destination router can handle it. It is used as a flow control at the IP network layer, usually between two adjacent machines. Below is a rule to allow both incoming and outgoing Source Quench messages.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
                   --icmp-type source-quench \
```

```
                    -d $ASMARA -j ACCEP
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
                    -s $ASMARA source-quench -j ACCEPT
```

The originating router responds by sending packets at a slower rate, gradually increasing the rate until it receives another Source Quench message.

- Destination Unreachable, ICMP message type 3, is a general error status message. It contains an error code field identifying the particular kind of error. Incoming messages has to be allowed. It is preferable to drop outgoing message because:
  - The message might be sent in response to a hacker's attempt to map the service port or address space which might enable him/her to get some useful information.
  - An attacker can also create a denial-of-service condition by forcing the system to generate large numbers of these messages by bombarding your unused UDP ports. Worse: an attacker can spoof the source address, forcing the system to send reply to the spoofed hosts.

    But outgoing message of subtypes Fragmentation Needed should be allowed. It is used to negotiate packet fragment size, otherwise the network performance can be seriously degraded without this negotiation. And if you want to respond to incoming traceroute requests, you must allow outgoing ICMP Destination Unreachable message, subtype code Port Unreachable. The rules used in the implementation are below. The '-p' flag is used to specifying a protocol (TCP, UDP, ICMP).

```
 ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
                    --icmp-type destination-unreachable \
                    -d $ASMARA -j ACCEPT
           ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
                    -s $ASMARA fragmentation-needed -j ACCEPT
```

- Parameter Problem, ICMP message type 12, is sent when a packet is received containing illegal or unexpected data in the header, or when the header checksum doesn't match the checksum generated by the receiving machine. This rule is required for a smooth data transfer.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
                --icmp-type parameter-problem \
                -d $ASMARA -j ACCEPT
 ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
                -s $ASMARA parameter-problem -j ACCEPT
```

- Time Exceeded, ICMP message type 11, indicates a timeout condition, or more accurately, that a packet's maximum hop count has been exceeded.Here is the rule to accept incoming time Exceeded messages.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
                --icmp-type time-exceeded \
                -d $ASMARA -j ACCEPT
```

  Outgoing Time Exceeded messages has to be dropped as it might be a good source of information for hackers. But to respond to incoming 'traceroute' requests, it is required to allow outgoing ICMP Time Exceeded messages.

- Echo request(type 8) and Echo Reply(type 0) Control messages
  Echo request message broadcast to all machines in a network address space and an Echo Reply messages, in return, is generated from all hosts. A network tool Ping is used for generating these messages. In the implementation incoming Echo Reply and outgoing Echo request are allowed. It is also allowed for these hosts belonging to the local network to ping the machine 'asmara' but other incoming Echo Requests are denied as 'ping' is used in denial-of-service attacks. The rules follow:

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
   --icmp-type echo-reply \
   -d $ASMARA -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
   -s $ASMARA echo-request -j ACCEPT
ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
   -s $LAN echo-request \
   -d $ASMARA -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
   -s $ASMARA echo-reply \
   -d $LAN -j ACCEPT
```

8.3.8 Blocking Incoming and Outgoing Smurf Attacks

Smurf attacks have used 'ping' packets continually broadcasting Echo Request messages to intermediate hosts with the sources address spoofed to be the intended victim's IP address. As a result, every machine in the intermediary's network continually bombards the victims machine with Echo Reply messages, choking off all available bandwidth.

The next rule protects smurf attack generated with a Broadcast source address.

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp  \
                -d $BROADCAST_DEST -j DENY -l
```

8.3.9 Protecting Services on Assigned Unprivileged ports

LAN services often run on unprivileged ports. Hence incoming connection attempts to these ports have to be blocked. It is also good to block outgoing connection attempts to protect from mistakes on your end, and log potential security problems.

For TCP-based services, a connection attempt to one of these services can be distinguished from an ongoing connection with a client using one of these unprivileged ports through the state of the SYN and ACK bits.

While TCP services on privileged ports are reasonably safe, UDP services are inherently less secure, and some services are assigned to run on unprivileged ports. Some services are offered through an officially registered, well-known unprivileged port. Additionally, some services, such as FTP and IRC, use complicated communication protocols that don't lend themselves well to packet filtering.

FTP is a good example of how the deny-by-default policy isn't always enough to cover all possible cases. FTP allows connections between two unprivileged ports. The rules allowing FTP inadvertently allow incoming connections to these other, local services as well. This shows how firewall rules are logically hierarchical and order-dependent. The rules explicitly protecting a LAN service running on unprivileged port must preceed the FTP rules allowing access to the entire unprivileged port range.

- Disallowing X Windows Connection(TCP port 6000:6063)

  The Connection to the remote X Window server should be made over SSH, which automatically supports X Windows Connection tunneling.

  The next rules rejects outgoing connection attempts to remote X Windows Manager and block and log incoming connection attempts to remote X

Windows Managers. In this rule he '-y' flag is used to specify connection initiation. It matches only TCP datagrams with the SYN bit set and ACK and FIN bits clear. The target DENY is preferable for incoming connection attempts to avoid giving useful information for hostile sites.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
        --destination-port $XWINDOW_PORTS -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
        --destination-port $XWINDOW_PORTS -j REJECT
```

- Disallowing SOCKS Server Connection(TCP Port 1080)

  SOCKS is a freely available local proxy server. SOCKS-aware client programs connect to the server instead of directly connecting to remote servers. The SOCKS server connects to remote servers, as a client, on your behalf.
  Attempts to connect to remote SOCKS servers are fairly common and often involve intrusion exploits.
  The following rules ensure no outgoing connection attempts to remote SOCKS servers and it blocks incoming connection attempts to your SOCKS server.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
   --destination-port $SOCKS_PORT -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
   --destination-port $SOCKS_PORT -j REJECT
```

- Disallowing NFS(UDP/TCP Port 2049) Connections As a datagram service, UDP doesn't have a connection state associated with it. Access to UDP services should simply be blocked.

  NFS is the main UDP service to be concerned with. NFS runs on unprivileged port 2049. It can be configured to run as a TCP-based services. The first rule blocks NFS UDP port 2049 from any incoming access. You shouldn't run NFS on a firewall machine, but if you are, external access has to be denied. The next two TCP rules block both incoming and outgoing connection attempts.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p udp \
         -d &ASMARA $NFS_PORT -j DENY -l
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
```

```
        --destination-port $NFS_PORT -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
        --destination-port $NFS_PORT -j REJECT
```

8.3.10 Allowing DNS

Domain Name Services(DNS) is one of the required Internet services. It translates between hostnames and their associated IP addresses. It is not possible to locate a remote host without IP, and hence DNS.

DNS uses a communication protocol that relies on both UDP and TCP. Connection modes include regular client-to-server connections, peer-to-peer traffic between forwarding servers and full servers, and primary and secondary name server connections.

Allowing DNS Lookups as a client: when a hostname requires a lookup , the resolver requests the lookup from a named server. DNS sends a lookup request as a UDP datagram but if error occurs because the data is too large to fit a UDP datagram DNS retries using a TCP connection.
Below is the configuration to allow DNS lookups as a client for one of primary nameservers. Similar rules are needed for the other two nameservers. The '!' character is used to nagate values. In this case '! y' filters TCP connection request packets.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
         -s $ASMARA $UNPRIVPORTS \
         -d $NAMESERVER_1 53 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
         -s $NAMESERVER_1 53 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT

ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $UNPRIVPORTS \
         -d $NAMESERVER_1 53 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         -s $NAMESERVER_1 53 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

If you offer a DNS service to the outside world, you need to allow traffic to the DNS port from the remote machines. But in this implementation DNS service

is not provided to any machine.

8.3.11 The AUTH User Identification Service

The AUTH, or IDENT, user identification service (TCP port 113) is most often used when sending mail or posting a Usenet article. Some FTP sites are also configured to require a resolvable AUTH lookup to get the account name of the user who initiated the service. Here is the script..

- Allowing outgoing AUTH Requests
  The machine should act as an AUTH client if you run a mail or FTP Server.
  The rules follow:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
        -s $ASMARA $UNPRIVPORTS \
        --destination-port 113 -j ACCEPT


ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
        --source-port 113 \
        -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

- Filtering Incoming AUTH Requests to Your Server
  Offering AUTH service is the subject of ongoing debate. Few FTP sites require this service. If you run the identd server, the following rules enable incoming identd connection request.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp
                --source-port $UNPRIVPORTS \
                -d $ASMARA 113 -j ACCEPT



ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
                -s $ASMARA 113 \
                --destination-port  $UNPRIVPORTS -j ACCEPT
```

  If it is decided not to offer the AUTH service, reject the connection request instead of denying the incoming requests each time a mail is sent or a Usenet article is post. It avoids a long wait for the TCP connection timeout otherwise client won't be notified that the mail or article was received for delivery until the identd request timed out.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
      --source-port $UNPRIVPORTS \
       -d $ASMARA 113 -j REJECT
```

8.4 Enabling Common Internet services

8.4.1 WEB

Web services are based on Hypertext Transfer Protocol(HTTP). Client and Server connections use the standard TCP conventions. Several higher-level, special-purpose communication protocols are available. Secure access over SSL is one of them.

- Standard HTTP Access (TCP Port 80) The next two rules allow access to a remote Web servers.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
          -s $ASMARA $UNPRIVPORTS \
          --destination-port 80 -j ACCEPT

 ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
          --source-port 80 \
          -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

  If you want to run a web server of your own and host a Web site for the Internet users, you need to use the following rules.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
          --source-port $UNPRIVPORTS \
          -d $ASMARA 80 -j ACCEPT

 ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
          -s $ASMARA 80 \
          --destination-port $UNPRIVPORTS -j ACCEPT
```

- Secure Web Access (SSL) (TCP Port 443)

  Secure Socket Layer is used for secure, encrypted Web access. The SSL protocol uses TCP Port 443. You will most often encounter this if you access a commercial Web site, use online banking services, or enter a protected Web area where you'll be prompted for personal information. The following two rules allow a client access a secure Web sites.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $UNPRIVPORTS \
         --destination-port 443 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         --source-port 443 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

If you need to conduct some form of e-commerce, you need to allow incoming connections to SSL-protected areas of your Web site. Below are the rules.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
                --source-port $UNPRIVPORTS \
                -d $ASMARA 443 -j ACCEPT

        ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
                -s $ASMARA 443 \
                --destination-port $UNPRIVPORTS -j ACCEPT
```

8.4.2 Usenet News

Usenet news is accessed over NNTP running on top of TCP through service port 119. Reading news and posting articles need the following rules to be included.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
   -s $ASMARA $UNPRIVPORTS \
   -d $NEWS_SERVER 119 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
   -s $NEWS_SERVER 119 \
   -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

A small site is very unlikely to host a news server for the outside world. But whenever it is required to give service, rules should be added to allow incoming connections from a set of clients to this port.

8.4.3 Telnet

Telnet is a standard means of login to remote machines over the Internet. It gives an insecure service, because it communicates in ASCII clear text. People

can snoop and detect password of users login from remote. It is preferable to use an encrypted service, such as 'SSH' rather than telnet. In my implementation I decided to allow outgoing telnet access to remote sites but denied to allow incoming services for security reasons. Below are the rules.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $UNPRIVPORTS \
         --destination-port 23 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         --source-port 23 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

8.4.4 SSH

SSH, secure shell, is considered by far preferable to using 'telnet' for remote login access, because both ends of the connection use authentication keys for both hosts and users, and data is encrypted. Additionally, SSH is more than a remote login service. It can automatically tunnel Window connections between remote sites, and FTP and other TCP-based connections can be tunnel through the more secure SSH connection. It is considered as poor man's virtual private network(VPN).

The ports used by SSH are highly configurable. The default connections are initiated between a client's unprivileged port and the server's assigned service port 22. But later the client is reassigned to a privileged port in the descending range from 1023 to 513 in order to support some authentication services. The range port you allow depends to the number of simultaneously incoming SSH connections you allow.

The following two rules allow you to connect to remote sites using SSH and the last two enable remote clients access your local SSH server.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $SSH_PORTS \
         --destination-port 22 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         --source-port 22 \
         -d $ASMARA $SSH_PORTS -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
```

```
        --source-port $SSH_PORTS \
        -d $ASMARA 22 -j ACCEPT

ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
        -s $ASMARA 22 \
        --destination-port $SSH_PORTS -j ACCEPT
```

8.4.5 FTP

FTP is the most common way of transferring files between two networked machines. FTP uses two privileged ports, one (port 21) for sending commands and another (port 20) for sending data. FTP has two modes for exchanging data. Normal port mode is the original mechanism when using the ftp client program and connecting to a remote FTP site. Passive mode is a newer mechanism, and is the default when connecting through a Web browser. Follows the FTP rules used in this paper together with their explanation.

- Allowing outgoing FTP requests to a remote FTP server

  Most sites want FTP client connections to a remote server. Below are the rules. In this case remote TCP connection request is not permitted. The text '! -y ' is used to protect remote server from initiating a connection.

  ```
  ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
     -s $ASMARA $UNPRIVPORTS \
     --destination-port 21 -j ACCEPT
  ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
     --source-port 21 \
     -d $ASMARA $UNPRIVPORTS -j ACCEPT
  ```

- Normal Port mode FTP data channels

  The remote server callbacks to establish Normal Port FTP data channels. This unusall process makes FTP difficult to secure. There is no mechanism to assure that the incoming connection is truly originating from the remote FTP server you have contacted. Hence you need to explicitly block incoming connections to local services running on unprivileged ports. The data connection rules follow:

  ```
  ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
     --source-port 20 \
     -d $ASMARA $UNPRIVPORTS -j ACCEPT
  ```

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
   -s $ASMARA $UNPRIVPORTS \
   --destination-port 20 -j ACCEPT
```

- Passive mode FTP data channel

  Passive mode is considered more secure than Normal port mode because the FTP client initiates both the control and data connections, even though the connection is made between two unprivileged ports.

  The rules below allow the newer passive data channel mode.

```
ipchains -A output  -i $EXTERNAL_INTERFACE -p tcp  \
   -s $ASMARA $UNPRIVPORTS
   --destination-port $UNPRIVPORTS -j ACCEPT

ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
   --source-port $UNPRIVPORTS \
   -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

  You need to implement a similar configuration to allow access to your local FTP server.

8.4.6 Email

Email is one of the most common Internet services. Email is sent across a network using the SMTP protocol assigned to TCP service port 25 and it is commonly received locally using one of the different protocols, SMTP, POP (port 110) and IMAP (port 143).

In this implementation SMTP is used both for sending and receiving email. The rules are listed below.

- Sending mail to any external mail server. You can restrict to limited mail servers.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
       -s $ASMARA $UNPRIVPORTS \
       --destination-port 25 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
        --source-port 25 \
       -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

- Receiving mail sent directly to your local machines from anywhere in the world.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
        --source-port $UNPRIVPORTS \
        -d $ASMARA 25 -j ACCEPT

ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
        -s $ASMARA 25 \
        --destination-port $UNPRIVPORTS -j ACCEPT
```

8.4.7 Finger

Finger (TCP port 79) provides user account information, including such as user login name, pending mail, real name and user furnished personal information in a '.plan' file. As such, it is regarded as insecure. My implementation allows accessing remote finger server but denies giving such service to remote sites.It might also be a good idea to allow local finger server service to restricted trusted sites.

Here are the rules:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $UNPRIVPORTS \
         --destination-port 79 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         --source-port 79 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

8.4.8 Traceroute

Traceroute is a less secure UDP service that causes intermediate systems to generate ICMP Time Exceeded messages to gather hop count information, and the target system to return a Destination Unreachable message, indicating the endpoint of the route to the host. It can be configured to use any port range. It often uses source ports in the range from 32769 to 65535 and destination port in the range from 33434 to 33523. As a less secure service, outgoing traceroute requests is enabled while incoming traceroute requests from any source are denied.

```
ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
```

```
        -s $ASMARA 32769:65535 \
        --destination-port 33434:33523 -j ACCEPT -l
```

8.4.9 ICQ

ICQ is a revolutionary, user-friendly Internet tool that informs you who's online at any time and enables you to contact them at will. No longer will you search in vain for friends or associates on the Net. With ICQ, you can chat, send messages, files and URL's, play games, or just hang out with your fellow 'Netters' while still surfing the Net.

But it is an insecure service. You can be crushed from outside. For better security it is preferable to disable the service though there are security tools desined for ICQ which are said to protect from intruders.

Below is the rule for enabling outgoing ICQ client request but it is very risk like the FTP command. At some point it opens the local unprivilaged ports of the client. You need to explicitly block incoming connections to local services running on unprivilaged ports. This clarifys that firewall rules are logically hierarchical and order-dependent. Besides it makes clear that is a deny-by-default policy isn't always enough to cover all possible cases

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
         -s $ASMARA $UNPRIVPORTS \
         --destination-port 2000:4000 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
         --source-port 2000:4000 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT

ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
         -s $ASMARA $UNPRIVPORTS \
         --destination-port 4000 -j ACCEPT

ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
         --source-port 4000 \
         -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

8.5 Installing the Firewall

The installation process is quite simple for systems with fixed IP address. Here are the steps:

- Rename the script file to be 'rc.firewall' and move it to the directory /etc/rc.d.
- The script should be owned by root and may be readable and executable by the group The commands are:

```
chown root.root /etc/rc.d/rc.firewall
chmod 0754 /etc/rc.d/rc.firewall
```

- The command 'sh /etc/rc.d/rc.firewall has to be added to the end of the file /etc/rc.d/rc.local.

A different set of configurations is needed for a Firewall with Dynamic IP address.

8.6 Debugging

Firewall rules are notoriously difficult to get right. It is a advisable to test the specified set of rules one by one. Here are general tips which might help fixing errors.

- Work on one service at a time.
- The first matching rule wins. Order is important. Use the ipchains -L chain -v commands to see how the rules are ordered.
- Always execute the rules from a complete test script. Be sure the script flushes all existing rules and resets the default policies first. Otherwise, you won't be sure which rules are in effect.
- Don't execute new rules from the command line. Especially don't execute the flush and default policy rules from the command line. If you are logged in using X Windows system or remotely from another system, you might be cut off.
- If a service works on your local network but not externally, turn on logging for accepted packets for your internal traffic to see what packet are being sent in both directions.
- If a service doesn't work at all, temporarily change the default incoming and outgoing policies to accept and log incoming accepted packets to get help in fixing the error.

8.6.1 Checking Network Interfaces and Network connectivity

A debugging tool 'ifconfig' is used for reporting the status of the network interface. The same tool is used in configuring and activating the network interfaces. The '-a' option should be used to get a report of the status of the inactive interfaces as well.

The tool 'ping' checks whether packets are passing through the interfaces. But a negative ping response doesn't prove that a remote site is down. The site might not allow Echo Request ICMP messages.

8.6.2 Checking Forwarding Rules

The command 'ipchains -L forward -v' is used. It lists Forwarding rules. The -n option reports all fields as numeric values rather than as symbolic names. This option can save time if your rules use a lot of symbolic hostnames that would require DNS lookups before being listed. The -v option produces more verbose output, including the interface name.

```
[root@asmara /root]# ipchains -L forward -v
Chain forward (policy DENY: 241 packets, 16204 bytes):
 pkts bytes target     prot opt    tosa tosx  ifname  source        destination ports
   26  2147 MASQ       all  ------ 0xFF 0x00  eth0    192.168.1.0/24 anywhere    n/a
```

This output implies the default forwarding rule is to reject. It also includes a rule which says: Any packets coming from the local network, destined for anywhere, are masqueraded and passed on through the external interface. The symbol 'n/a' stands for not defined port.

8.6.3 Checking Input Rules

The command used is 'ipchains -L input -v' The output is huge but I tried to paste selected output for giving a sample explanation.

```
Chain input (policy DENY: 602334 packets, 99057656 bytes):
 pkts bytes target     prot opt    tosa tosx  ifname source              destination          ports
    0     0 ACCEPT     all  ------ 0xFF 0x00  lo     anywhere            anywhere             n/a
  325 36243 ACCEPT     all  ------ 0xFF 0x00  eth1   192.168.1.0/24      anywhere             n/a
    0     0 DENY       all  ----l- 0xFF 0x00  any    10.0.0.0/8          anywhere             n/a
    0     0 DENY       all  ----l- 0xFF 0x00  any    172.16.0.0/12       anywhere             n/a
    0     0 DENY       all  ----l- 0xFF 0x00  any    192.168.0.0/16      anywhere             n/a
    0     0 DENY       all  ----l- 0xFF 0x00  any    localhost           anywhere             n/a
    0     0 DENY       all  ----l- 0xFF 0x00  any    anywhere            0.0.0.0              n/a
   61 10942 ACCEPT     udp  ------ 0xFF 0x00  eth0   fenix.it.uu.se      asmara.csd.uu.se     domain ->   1024:65535
    0     0 ACCEPT     tcp  !y---- 0xFF 0x00  eth0   fenix.it.uu.se      asmara.csd.uu.se     domain ->   1024:65535
   16  4751 ACCEPT     udp  ------ 0xFF 0x00  eth0   meryl.it.uu.se      asmara.csd.uu.se     domain ->   1024:65535
    0     0 ACCEPT     tcp  !y---- 0xFF 0x00  eth0   meryl.it.uu.se      asmara.csd.uu.se     domain ->   1024:65535
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   anywhere            asmara.csd.uu.se     echo-reply
    4   224 ACCEPT     icmp ------ 0xFF 0x00  eth0   anywhere            asmara.csd.uu.se     destination-unreachable
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   anywhere            asmara.csd.uu.se     source-quench
 5821  788K DENY       udp  ----l- 0xFF 0x00 eth0   anywhere            anywhere             any ->   0:1023
    0     0 DENY       udp  ----l- 0xFF 0x00  eth0   anywhere            anywhere             any ->   1024:65535
    0     0 DENY       icmp ----l- 0xFF 0x00  eth0   anywhere            anywhere             redirect
```

Here some interpretation of the selected results from the list above.

- The default input rule is to deny.
- Rule 1 says any packets incoming through local interface is accepted. Rules 6 says any other packet claiming from localhost coming by any interface is rejected. This fits my stated rules. Order of rules is very important.
- Rules 3,4 and 5 deny packets having source address of reserved Class A, B, C private network.
- You can also clearly check that selected incoming ICMP packets are accept and the rest are denied.

8.6.4 Checking Output Rules

The command 'ipchains -L output -v' is used to Check Output rules. Below is a list of selected results. A similar analysis of the result can help check whether the rules are defined as you desired.

```
Chain output (policy REJECT: 183638 packets, 14617237 bytes):
 pkts bytes target     prot opt    tosa tosx  ifname source               destination          ports
    0     0 ACCEPT     all  ------ 0xFF 0x00  lo     anywhere             anywhere             n/a
    0     0 REJECT     tcp  -y---- 0xFF 0x00  eth0   anywhere             anywhere             any ->   2049
    0     0 REJECT     tcp  -y---- 0xFF 0x00  eth0   anywhere             anywhere             any ->   socks
   85  5840 ACCEPT     udp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     fenix.it.uu.se       1024:65535 ->   domain
    0     0 ACCEPT     tcp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     fenix.it.uu.se       1024:65535 ->   domain
   20  1273 ACCEPT     udp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     meryl.it.uu.se       1024:65535 ->   domain
    0     0 ACCEPT     tcp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     meryl.it.uu.se       1024:65535 ->   domain
    4   280 ACCEPT     udp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     DNS1.UU.SE           1024:65535 ->   domain
    0     0 ACCEPT     tcp  ------ 0xFF 0x00  eth0   asmara.csd.uu.se     DNS1.UU.SE           1024:65535 ->   domain
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     UpUNet.UU.SE/16      echo-reply
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     anywhere             fragmentation-needed
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     anywhere             source-quench
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     anywhere             echo-request
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     anywhere             parameter-problem
    0     0 ACCEPT     icmp ------ 0xFF 0x00  eth0   asmara.csd.uu.se     UpUNet.UU.SE/16      time-exceeded
    8  1052 REJECT     all  ----l- 0xFF 0x00  eth0   anywhere             anywhere             n/a
```

8.6.5 Testing Individual Packets Against the Firewall Rules

Individual rules and packet types can be tested. The -C option will report whether the packet would be accepted or denied by the firewall rules.
There are some ipchains syntax differences between defining actual rules and defining test packet descriptions. You can't use default values when using the '-C' option. Exact command-line arguments must be specified. The test packet description must use the -i option to specify an interface. Explicit source and destination ports and addresses must be specified. Below are some sample tests.

```
 [root@asmara /root]# ipchains -C forward -i eth0 -p tcp -s 192.168.1.0/24 5000 -d any/0 80
     masqueraded
[root@asmara /root]# ipchains -C forward -i eth0 -p tcp -s 193.168.1.0/24 5000 -d any/0 80
denied
```

The above commands assure that any packet coming from the a machine on the LAN and destined to any address is masqueraded while packet coming from outside of my LAN is denied.

Here is another test rule.

```
[root@asmara /root]# ipchains -C input -i eth0 -p tcp -y -s 192.168.1.1 5000 -d 130.238.12.197 25
denied
```

Clearly it matches with the IP spoofing rule. Packets claiming to be from my source address should be denied.

8.6.6 Checking Network processes

The command "netstat -a -p -A inet" is a way to check which programs were running and listening on which network interfaces, over which transport protocols (TCP or UDP), and on which service ports. The -A inet option limits the report to services and ports related to remote network communications. Eliminate that option to include UNIX reports.

The command 'ps -ax' lists all programs running on the machine. With the exception of few system daemons; init, kflushd, kpiod, kswapd, mdrecoveryd, khubd, and mingetty, at other daemons should be services explicitly enabled under the runlevel manager, /etc/rc.d/rc.local, or /etc/inetd.conf. Undesired services should be disabled.

8.6.7 Firewall Log messages

Individually matched packets are logged to /var/log/messages and the consol by default. But firewall log messages could be duplicated to a different file by configuring the /etc/syslog.conf file.

Log analysis tools such as logcheck, switchs, etc. could be used to monitor what is written to the system logs. These tools can run continually to identify potential security problems and notify the authorised person.

9 Intrusion Detection

No matter how security minded you are, no matter how many updates and patches you apply, there's always a chance that someone will crack one of your systems. Security developers and hackers are in a continual race to keep one step ahead of each other. Hackers are continually looking for new vulnerabilities and new approaches. What is secured today might not be secure tomorrow.

9.1 How do intruders get into systems?

Intruders get into a system in a number of ways:

- Software bugs will crop up forever. System Administrators and Programmers can never track down and eliminate all possible holes. Intruders have only to find one hole to break in.
- Configuration mistakes are going to be made. For instance, the default, easy-to-use setup which comes with the computer can be hacked in easily.
- Password cracking: guessing, dictionary attack and brute force attack can be used.
- Sniffing unsecured traffic.
- Design flaws also lead to possible security problems.

Hence an "Intrusion Detection System (IDS)" has to be run continually to recognise attacks against the network that firewalls are unable to see and fix bugs immediately. In addition you must have a backup plan in the event that security is compromised.

9.2 How are intrusions detected?

Here are ways to increase intrusion detection.

- Examine log files (/var/logmaillog, /var/log/messages ...) for connections from unusual locations or other unusual activity. But don't forget that many intruders edit log files in an attempt to hide their activity.
- Look for setuid and setgid files (especially setuid root files) everywhere on your system. Intruders often leave setuid copies of /bin/sh or /bin/time around to allow them root access at a late time.

- Check your system binaries to make sure that they haven't been altered. The use of MD5, Tripwire, and other cryptographic checksum tools can be used to detect any change.
- Check your systems for unauthorised use of a network monitoring program, commonly called a sniffer or packet sniffer. Intruders may use a sniffer to capture user account and password information.
- Examine all the files that are run by 'cron' and 'at.' Intruders leave back doors in files run from 'cron' or submitted to 'at.'
- Check for unauthorised services. Inspect /etc/inetd.conf for unauthorised additions or change.
- Examine the /etc/passwd file on the system and check for modifications to that file. In particular, look for the unauthorised creation of new accounts, accounts with no passwords, or UID changes (especially UID 0) to existing accounts.
- Check your system and network configuration files for unauthorised entries. In particular, check /etc/hosts, /etc/hosts.equiv, /etc/hosts.deny /etc/hosts.allow, /etc/hosts.lpd, and so on.
- Look everywhere on the system for unusual or hidden files as these can be used to hide tools and information.
- System integrity tools such as SATAN, tripwire, Cops, Crack, ifstatus, MD5, etc. [2] has to be used in analyses, audits, and vulnerability checks.

A Appendix-Firewall Script

```
#!/bin/sh
# --------------------------------------------------------------------------
# Author: Yordanos G. Beyene
# February 24, 01.
# This configurations is implemented on a Packet Filtering dual-homed gateway.
# It is a simple configuration to be used for sites who don't give much service
# to Internet(remote) users.
# Throughout the configuration variouse attacks are attenuated and the most
# common Internet services are enabled.
# --------------------------------------------------------------------------

echo "Starting firewalling... "

# --------------------------------------------------------------------------
#  Symbolic constants for making  things more readable and easy maintenance
# --------------------------------------------------------------------------

PATH="/sbin"
LOOPBACK_INTERFACE="lo"            # loopback interface
LOCAL_INTERFACE="eth1"             # internal LAN interface
EXTERNAL_INTERFACE="eth0"          # Internet connected interface
ASMARA="130.238.12.197"            # asmara IP address
LAN="192.168.1.0/24"               # LAN (private network)
NAME_SERVER1="130.238.8.10"        # primary nameserver used by asmara
NAME_SERVER2="130.238.12.42"       # secondary nameserver used by asmara
NAME_SERVER3="130.238.4.133"    # third nameserver used by asmara
NEWS_SERVER="news.uu.se"           # News server of the university
LOOPBACK="127.0.0.0/8"             # reserved loopback address range
CLASS_A="10.0.0.0/8"               # class A private networks
CLASS_B="172.16.0.0/12"            # class B private networks
CLASS_C="192.168.0.0/16"           # class C private networks
CLASS_D_MULTICAST="224.0.0.0/4"    # class D multicast addresses
CLASS_E_RESERVED="240.0.0.0/5"     # class E reserved addresses
BROADCAST_SRC="0.0.0.0"            # broadcast source address
BROADCAST_DEST="255.255.255.255"   # broadcast destination address
PRIV_PORTS="0:1023"                # privileged port range
UNPRIV_PORTS="1024:65535"          # unprivileged port range
```

```
XWINDOW_PORTS="6000:6063"          # X windows port range
SSH_PORTS="513:65535"              # SSH port range

# ------------------------------------------------------------------------------
# Default policy is DENY
# Remove all existing rules belonging to this filter
    ipchains -F
# Set the default policy to DENY.
    ipchains -P input  DENY
    ipchains -P output REJECT
    ipchains -P forward DENY
# ------------------------------------------------------------------------------

# Enable TCP SYN Cookie Protection
    echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Enable IP spoofing protection
    # turn on Source Address Verification
    for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
        echo 1 > $f
    done
# Disable ICMP Redirect Acceptance
    for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
        echo 0 > $f
    done

    for f in /proc/sys/net/ipv4/conf/*/send_redirects; do
        echo 0 > $f
    done

# Disable Source Routed Packets
    for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
        echo 0 > $f
    done

# ------------------------------------------------------------------------------
# Allow Unlimited traffic on the loopback interface.
    ipchains -A input  -i $LOOPBACK_INTERFACE  -j ACCEPT
```

```
    ipchains -A output -i $LOOPBACK_INTERFACE  -j ACCEPT
# ------------------------------------------------------------------------------
# Allow internal machines to access the fireall machine.
    ipchains -A input  -i $LOCAL_INTERFACE -s $LAN -j ACCEPT
    ipchains -A output -i $LOCAL_INTERFACE -d $LAN -j ACCEPT
# ------------------------------------------------------------------------------
# Masquerade internal traffic.
    ipchains -A forward -i $EXTERNAL_INTERFACE -s $LAN -j MASQ
# ------------------------------------------------------------------------------
# Refuse spoofed packets.

# Refuse incoming packets pretending to be from the external address.
    ipchains -A input   -s $ASMARA -j DENY -l

# Refuse incoming packets claiming to be from a Class A, B or C private network
    ipchains -A input   -i $EXTERNAL_INTERFACE -s $CLASS_A -j DENY -l
    ipchains -A input   -i $EXTERNAL_INTERFACE -s $CLASS_B -j DENY -l
    ipchains -A input   -i $EXTERNAL_INTERFACE -s $CLASS_C -j DENY -l

# Refuse broadcast address source  packets
    ipchains -A input   -s $BROADCAST_DEST -j DENY -l
    ipchains -A input   -d $BROADCAST_SRC -j DENY -l

# Refuse special addresses defined as reserved by the IANA.
    ipchains -A input   -s 0.0.0.0/8 -j DENY -l
    ipchains -A input   -s 127.0.0.0/8 -j DENY -l
    ipchains -A input   -s 169.254.0.0/16 -j DENY -l
    ipchains -A input   -s 192.0.2.0/24 -j DENY -l
    ipchains -A input   -s 224.0.0.0/3 -j DENY -l
# ------------------------------------------------------------------------------
# Avoid ports subject to attacks.

# Disallow NFS connection

    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $NFS_PORT -j DENY -l
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $NFS_PORT -j REJECT
```

```
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
             --destination-port $NFS_PORT -j DENY -l

# Disallow SOCKS connection
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $SOCKS_PORT -j DENY -l
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $SOCKS_PORT -j REJECT

# Disallow X Windows Connection
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $XWINDOW_PORTS -j DENY -l
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -y \
             --destination-port $XWINDOW_PORTS -j REJECT

# ------------------------------------------------------------------------
# Allow outgoing traceroute request but accept incoming traceroute from
# trusted sites or Local network
    ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
                -s $ASMARA 32769:65535 \
                --destination-port 33434:33523 -j ACCEPT -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
             -s LAN  32769:65535 \
             -d $ASMARA  -j ACCEPT -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
             --source-port  32769:65535  \
             --destination-port 33434:33523 -j DENY -l

# ------------------------------------------------------------------------
# DNS client service (the rules included here apply for the Primary NAMESERVER only
#                     only but similar rules should be implemented fot the other.
    ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
             -s $ASMARA $UNPRIVPORTS \
             -d $NAMESERVER_1 53 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
             -s $NAMESERVER_1 53 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
```

```
            -s $ASMARA $UNPRIVPORTS \
            -d $NAMESERVER_1 53 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
            -s $NAMESERVER_1 53 \
            -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allowing outgoing AUTH Requests
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
            -s $ASMARA $UNPRIVPORTS \
            --destination-port 113 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
            --source-port 113 \
            -d $ASMARA $UNPRIVPORTS -j ACCEPT
# REJECT incoming  AUTH Requests
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
            --source-port $UNPRIVPORTS \

# ------------------------------------------------------------------------------
# Allows accessing remote  WEB servers
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
            -s $ASMARA $UNPRIVPORTS \
            --destination-port 80 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
            --source-port 80 \
            -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allows accessing remote secure WEB servers
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
            -s $ASMARA $UNPRIVPORTS \
            --destination-port 443 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
            --source-port 443 \
            -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allow accessing local WEB server
```

```
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
             --source-port $UNPRIVPORTS \
             -d $ASMARA 80 -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
             -s $ASMARA 80 \
             --destination-port $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allow accessing USENET NEWS
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             -d $NEWS_SERVER 119 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             -s $NEWS_SERVER 119 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allowing sending Email to any external mail server
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 25 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 25 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT

# Allowing receiving direct Emails
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
             --source-port $UNPRIVPORTS \
             -d $ASMARA 25 -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
             -s $ASMARA 25 \
             --destination-port $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Enabling SSH connection to remote sites
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $SSH_PORTS \
             --destination-port 22 -j ACCEPT
```

```
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 22 \
             -d $ASMARA $SSH_PORTS -j ACCEPT

# Enabling remote sites access your local SSH server
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
             --source-port $SSH_PORTS \
             -d $ASMARA 22 -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
             -s $ASMARA 22 \
             --destination-port $SSH_PORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allow outgoing TELNET request
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 23 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 23 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allows accessing remote finger server
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 79 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 79 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Allowing normal port outgoing FTP requests
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 21 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 21 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT
```

```
# Allowing  remote server callbacks to establish Normal Port FTP data channel
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  \
             --source-port 20 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 20 -j ACCEPT

# ------------------------------------------------------------------------------
#enabling outgoing ICQ client request.
    ipchains -A output -i $EXTERNAL_INTERFACE -p tcp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 2000:4000 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp ! -y \
             --source-port 2000:4000 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p udp  \
             -s $ASMARA $UNPRIVPORTS \
             --destination-port 4000 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
             --source-port 4000 \
             -d $ASMARA $UNPRIVPORTS -j ACCEPT

# ------------------------------------------------------------------------------
# Configuring ICMP messages Control and Status Messages to prevent denial of
# service while getting a smooth connection.
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
             --icmp-type source-quench \
             -d $ASMARA -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
             -s $ASMARA source-quench -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
             -s 130.238.0.0/16 echo-request \
             -d $ASMARA -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
             -s $ASMARA echo-request -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
```

```
            --icmp-type echo-reply \
            -d $ASMARA -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
            -s $ASMARA echo-reply \
            -d LAN -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
            --icmp-type destination-unreachable \
            -d $ASMARA -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
            --icmp-type time-exceeded \
            -d $ASMARA -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
            -s $ASMARA time-exceeded \
            -d 130.238.0.0/16 -j ACCEPT
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
            --icmp-type parameter-problem \
            -d $ASMARA -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
            -s $ASMARA parameter-problem -j ACCEPT
    ipchains -A output -i $EXTERNAL_INTERFACE -p icmp  \
            -s $ASMARA fragmentation-needed -j ACCEPT

# ------------------------------------------------------------------------------
# Enable logging for selected denied packets, helps to detect intrusion or bugs.
    ipchains -A input  -i $EXTERNAL_INTERFACE -p tcp  -j DENY -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
            --destination-port $PRIVPORTS -j DENY -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p udp  \
            --destination-port $UNPRIVPORTS -j DENY -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
            --icmp-type 5 -j DENY -l
    ipchains -A input  -i $EXTERNAL_INTERFACE -p icmp  \
            --icmp-type 13:255 -j DENY -l
    ipchains -A output -i $EXTERNAL_INTERFACE  -j REJECT -l
# ------------------------------------------------------------------------------

echo "done"
exit 0
```

References

[1] Avolio,F., *1999, Firewalls and Internet Security, the Second Hundred (Internet) Years*, Internet Protocol Journal (IPJ), V2.http://www.avolio.com/fw2hundred.html

[2] Cert Coordination Centre., *2000, Internet security issues*, Carnegie Mellon: Software Engineering Institute, http://www.cert.org/tech-tips/

[3] Curtin,M. and Ranum,M.J., *2000, Internet Firewalls: FRequently Asked Questions*. http://www.interhack.net/pubs/fwfaq/

[4] Grennan, M., *2000, Firewall and Proxy Server HOWTO*. http://www.linuxdoc.org/HOWTO/Firewall-HOWTO.html

[5] Kirch,O. and Dawson,T., *2000, Linux: Network Administrator's Guide*, 2nd ed., O'Reilly, Beijing, 474pp.

[6] Pfleeger,C.P., *1997, Security in Computing*, 2nd ed., Prentice-Hall, London, 574pp.

[7] Russell,R., *2000, Linux Ipchains Howto*. http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html

[8] Vicom Technology Ltd, *2001, Firewalls*. http://www.vicomsoft.com/knowledge/reference/firewalls1.html

[9] Ziegler,R.L., *2000, Linux Firewalls*, New Riders, Indiana, 470pp.

[10] Zwicky,E.D., Cooper,S. and Chapman,D.B., *2000, Internet and Web Security: Building Internet Firewalls*, O'Reilly, Beijing, 869pp. http://www.sunworld.com/sunworldonline/swol-01-1996/swol-01-firewall-2.html

[11] *2000,The Internet Assigned Numbers Authority*, http://www.iana.org/