ISSN:0975-9646

"Firewall Design Techniques and its Development in Linux System"

Prof. Vinit A. Sinha
Assistant Professor (MCA Department)
PRMIT&R, Badnera Amravati (M.S), India

Abstract: - Firewalls are one of the most commonly used security systems to protect networks and hosts. Most researchers have focused on analyzing the latency and throughput of router firewalls. Different from this approach, this research focuses on studying the performance impact and the sensitivity of the Linux firewall (iptables) for a single host. The performance and the sensitivity of the firewall is measure by designing and instrumenting each layer of the Linux TCP/IP stack.

As firewall designed in Linux (OSS), user or requesting person can change the source code of firewall as per our security requirement at any time It is a device or set of device configure to permit , deny, encrypt , decrypt or proxy all (in or out) computer traffic between different security domains based upon a set of rules and other criteria. Firewall can be implemented in either hardware or software, or a combination of both. Firewalls are frequently used to prevent unauthorized internet users from accessing private a network connected to the internet especially intranet. All message entering or leaving the intranet pass through the firewall which examines each message and blocks those that do not meet those that do not meet the specified security criteria.

Keywords: Linux kernel, Firewall technologies, NAT, iptables, socks

I. Introduction

Firewall Introduction:-

Firewalls come in various sizes and flavors. The most typical idea of a firewall is a dedicated system or appliance that sits in the network and segments an "internal" network from the "external" Internet. Most home or SOHO networks use an appliance-based device for broadband connectivity that includes a built-in firewall. In general, firewalls can be categorized under one of two general types:

- Desktop or personal firewalls
- Network firewalls

The primary difference between these two types of firewalls simply boils down to the number of hosts that the firewall protects. Within the network firewall type, there are primary classifications of devices, including the following:

- Packet-filtering firewalls (stateful and non stateful)
- Circuit-level gateways
- Application-level gateways

II. Problem Selection

A person or anyone who is responsible for a private network cannot communicate with public network without using a security program. Firewall is one of the efficient ways to transfer data & information in the safest way.

But now a day there are number of organization provides a firewall security to customer or service requester , while accepting there services users faces some problems like –

1) They bind with service providers for restricted time duration.
2) Many times Real time services are failed to reach users.
3) Users should follow every rule and accept all the conditions generated by firewall service providers.
4) Users cannot be customized firewall as per their own requirements.
5) In readymade firewall some time user cannot be resolved primary errors without help of providers.
6) With personal firewall (Self-Created) person have all the authority to manage each files & folders and has option to configure itself.
7) As linux is open source system the firewall source code is open to modify reliably.
8) Users can create their personal firewall policy as per there security need.
9) In readymade firewall person bonded to configure a firewall as per their policy.
10) With self-creating firewall person have all copyrights to distribute it . also open source system contribute in it.

III. Research Objective

1) To monitor incoming and outgoing security alerts as well as record and track down an intrusion attempt depending on the severity.
2) To shield our computer from outside hacker attacks.
3) To build customized firewall so that user can handle it as per our security requirement.
4) To create user friendly firewall , which is not more complex as compared to day today market's firewall
5) A feeling of increased security that person's PC and contents are being protected.

IV. Firewall Technologies

This section focuses on the technologies used in various firewalls and how they work. The firewall taxonomy in Figure 1 shows the general types of firewalls. This section focuses more on the underlying technologies that devices that fall into those types utilize. In some cases, one technology discussed here can fall into multiple types in the taxonomy tree. The focus is on a wide range of firewall technologies, including the following:

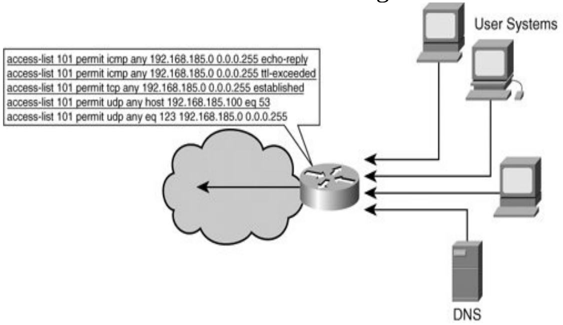
- Personal firewalls
- Packet filters
- Network Address Translation (NAT) firewalls
- Circuit-level firewalls

Personal Firewalls

Personal firewalls are designed to protect a single host. They can be viewed as a hardened shell around the host system, whether it is a server, desktop, or laptop. Typically, personal firewalls assume that outbound traffic from the system is to be permitted and inbound traffic requires inspection. By default, personal firewalls include various profiles that accommodate the typical traffic a system might see.

Packet Filters

Packet filters are network devices that filter traffic based on simple packet characteristics. These devices are typically stateless in that they do not keep a table of the connection state of the various traffic flows through them.

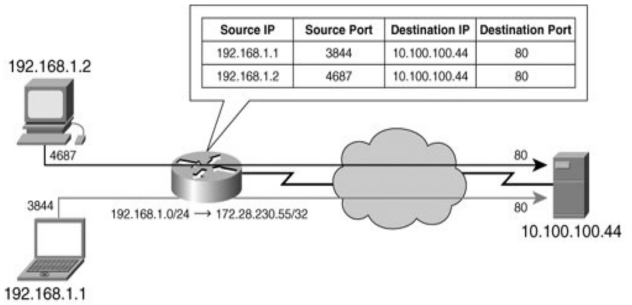


```
access-list 101 permit icmp any 192.168.185.0 0.0.0.255 echo-reply
access-list 101 permit icmp any 192.168.185.0 0.0.0.255 ttl-exceeded
access-list 101 permit tcp any 192.168.185.0 0.0.0.255 established
access-list 101 permit udp any host 192.168.185.100 eq 53
access-list 101 permit udp any eq 123 192.168.185.0 0.0.0.255
```

Figure1. Simple Access List Sample Network

NAT Firewalls

A distinct firewall that existed for a short period is the Network Address Translation (NAT) firewall. In today's firewall market, NAT is a part of almost every firewall product available. From the lowliest SOHO firewall such as the Linksys BEFSX41 to the high-end enterprise PIX 535, NAT is now a function of a firewall.

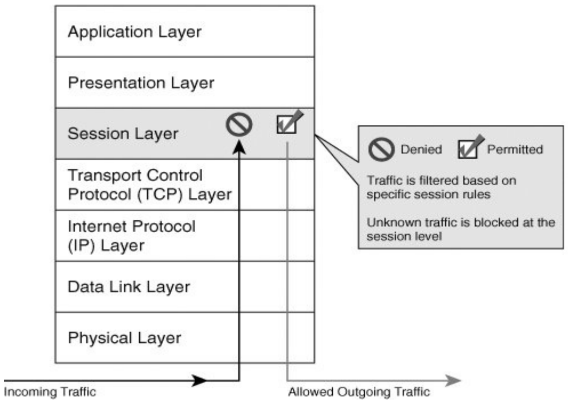


Source IP	Source Port	Destination IP	Destination Port
192.168.1.1	3844	10.100.100.44	80
192.168.1.2	4687	10.100.100.44	80

Figure 2 NAT Firewall

Circuit-Level Firewalls

Circuit-level firewalls work at the session layer of the OSI model and monitor "handshaking" between packets to decide whether the traffic is legitimate. Traffic to a remote computer is modified to make it appear as though it originated from the circuit-level firewall. This modification makes a circuit-level firewall particularly useful in hiding information about a protected network but has the drawback that it does not filter individual packets in a given connection.



Application Layer
Presentation Layer
Session Layer
Transport Control Protocol (TCP) Layer
Internet Protocol (IP) Layer
Data Link Layer
Physical Layer
Denied
Permitted
Traffic is filtered based on specific session rules
Unknown traffic is blocked at the session level
Incoming Traffic
Allowed Outgoing Traffic

Figure 3. . Circuit-Level Firewall

V. TOOLS FOR FIREWALL - ABOUT LINUX

Linux was first released in 1991 by its author Linus Torvalds at the University of Helsinki. Since then it has grown tremendously in popularity as programmers around the world embraced his project of building a free operating system, adding features, and fixing problems. Linux is popular with today's generation of computer users for the same reasons early versions of the UNIX operating system enticed fans more than 20 years ago. Linux is portable, which means people will find versions running on name-brand or clone PCs, Apple Macintoshes, Sun workstations, or Digital Equipment Corporation Alpha-based computers. Linux also comes with source code, so it is easy to change or customize the software to adapt to required needs. Finally, Linux is a great operating system, rich in features adopted from other versions of UNIX.

Features in Linux

1) No constant rebooting
2) Start/stop services without interrupting others
3) Portable software
4) Downloadable applications
5) No settings hidden in code or registries
6) Mature desktop
7) Freedom

VI . PROPOSED METHODOLOGY

A firewall is a system or router that sits between an external network (i.e. the Internet) and an internal network. This internal network can be a large LAN at a business or user networked home PCs. The firewall in it's simplest form is like a one-way street. It allows people on the internal network to access the external network (the Internet), but it restricts traffic so that no one can use the external network to access the systems or files on the internal network.

What Is iptables?

Originally, the most popular firewall/NAT package running on Linux was ipchains, but it had a number of shortcomings. To rectify this, the Netfilter organization decided to create a new product called iptables

How To Start iptables

start, stop, and restart iptables after booting by using the commands:

[root@bigboy tmp]# service iptables start
[root@bigboy tmp]# service iptables stop
[root@bigboy tmp]# service iptables restart

To get iptables configured to start at boot, use the chkconfig command:.

[root@bigboy tmp]# chkconfig iptables on

Determining The Status of iptables

determine whether iptables is running or not via the service iptables status command. Fedora Core will give a simple status message. For example

[root@bigboy tmp]# service iptables status
Firewall is stopped.
[root@bigboy tmp]#

Designing and Using DMZ Networks to Protect Internet Servers

One of the most useful tools in firewall engineering today is the DMZ, or DeMilitarized Zone, a network where all publicly accessible services are placed so they can be more closely watched and, also, isolated from one's internal network. DMZs, bastion servers and Linux make a particularly good combination.

But what, really, is a DMZ? Is there more than one correct way to design one? Does everyone who hosts internet services need a DMZ network? These are issues really haven't addressed yet, so this month we're going to take a higher-level look at DMZ security.

VII . Design Of Required Server Program

The SOCKS Proxy Server

Setting up the Proxy Server

The SOCKS proxy server available from **http://www.socks.nec.com/**. Uncompressed and un tar the files into a directory on user system, and follow the instructions on how to make it. couple problems are arises while making. Make sure that user Make files are correct.

One important thing to note is that the proxy server needs to be added to /etc/inetd.conf. User must add a line:

socks stream tcp nowait nobody /usr/local/etc/sockd sockd to tell the server to run when requested.

Configuring the Proxy Server

The SOCKS program needs two separate configuration files. One to tell the access allowed, and one to route the requests to the appropriate proxy server. The access file should be housed on the server. The routing file should be housed on every UNIX machine. The DOS and, presumably, Macintosh computers will do their own routing.

Getting the Proxy Server to work with UDP Packets

The SOCKS package works only with TCP packets, not UDP. This makes it quite a bit less useful. Many useful programs, such as talk and Archie, use UDP. There is a package designed to be used as a proxy server for UDP packets called UDP relay, by Tom Fitzgerald <fitz@wang.com.

Drawbacks with Proxy Servers

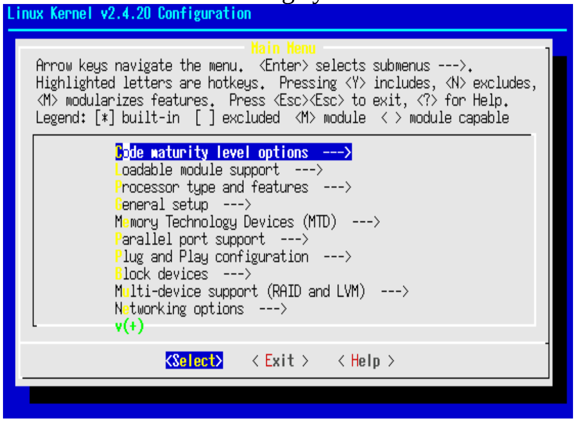
The proxy server is, above all, a security device. Using it to increase internet access with limited IP addresses will have many drawbacks. A proxy server will allow greater access from inside the protected network to the outside, but will keep the inside completely inaccessible from the outside. This means no servers, talk or archive connections, or direct mailing to the inside computers.

VIII . Compilation Of Linux Kernel

The Linux kernel is essentially what Linus Torvalds started work on in 1991. A kernel is the part of the operating system that handles the communication between software and user machine's hardware. First, user need to know what hardware is in user machine in order to get a good kernel at the end of the process. There are also other concerns. For example, user need to decide what kinds of file systems user machine needs to work with (ext2, Reiser, VFAT, NTFS, etc), what character support to include in the kernel, what networking protocols user need to use or whether the machine will be used as a router or firewall, to name a few.

There are three main interfaces that user can choose from when user are preparing the configuration of a new kernel. They are 'config', 'menuconfig' and 'xconfig'. The first one is an old-style command-line interface that is not very attractive nor do I recommend it if this is something that user've never done before.

The last one, xconfig, is a GUI based configuration tool that requires user have X-window running. This may not be the best option if user have a server because there's no need to be running x-window in that environment. As a general rule, the middle option, 'menuconfig' because it has an ncurses based GUI (text mode with colors) and therefore it is visual and straightforward. It can also be used to configure a kernel on a remote machine, something that xconfig is just not suited to. Here's what the menuconfig system looks like:



```
Linux Kernel v2.4.20 Configuration
                         Main Menu
 Arrow keys navigate the menu.  <Enter> selects submenus --->.
 Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
 <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
 Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

          Code maturity level options  --->
          Loadable module support  --->
          Processor type and features  --->
          General setup  --->
          Memory Technology Devices (MTD)  --->
          Parallel port support  --->
          Plug and Play configuration  --->
          Block devices  --->
          Multi-device support (RAID and LVM)  --->
          Networking options  --->
          v(+)

               <Select>    < Exit >    < Help >
```

To use this, just type: 'make menuconfig' inside user /usr/src/linux/ directory. User will see something very similar to the screenshot above. What user have here are the major categories of kernel configuration options. Since I don't know what each machine out there is going to need, it would be impossible to go over every option.

A firewall effectively puts all of its systems inside a fort with the door closed.

The socks program can be thought of as an intelligent sentinel that checks IP addresses as they pass through the firewall. If an illegal origin IP address comes up to the firewall and tries to enter the network, then the socks program will deny the entry and even warn the super user of the intended intrusion.

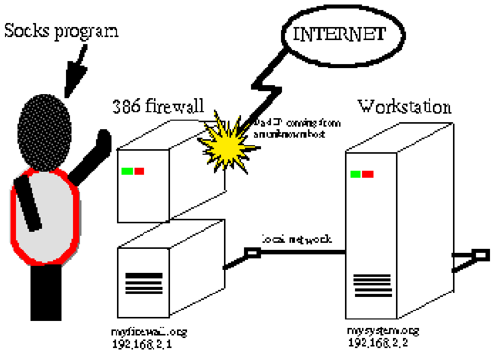


Figure 4. Getting on internet

IX . Finalize the Firewall

Firewalls are a prime example of an opportunity to use a special-purpose Linux distribution. Linux firewall distributions typically

1. Are tuned to include primarily those components needed to be a firewall.
2. Contain scripts for easily configuring firewall settings.
3. Don't include X, which requires that user use the command line or a Web browser from another machine on the network, allowing the distribution to fi t in a much smaller space.
4. Include a few other tools for diagnosing network problems or serving the local network in some way.

For desktop Linux system, a simplified GUI firewall tool is a good way to begin protecting the computer. At the very least, firewall to explicitly allow others to use selected services from person computer, while blocking requests for other services. Tools that come with Mandriva, openSUSE, Fedora, and Red Hat Enterprise Linux (RHEL) illustrate a few ways to configure a firewall using GUI tools. During installation of Fedora or RHEL, the anaconda installer offers a screen for selecting a simple, secure firewall. Mandriva offers a firewall tool with its Control Center. X. Limitations, Conclusion & Limitation:-

Limitations :-

1. As linux is little known to general users, the end users find it difficult to configure the firewalls.
2. The command line interface to configure linux firewalls is difficult to operate by general users as commands are not easy to remember.
3. Many third party firewall application are available for windows platform in open market which are easy to operate , makes end users lured to these applications .
4. General users are less aware of security issues . Hence they are not bothered using such application.

Conclusion:-

As firewall design in linux system, It is easy to implement on any other platform oriented system , because linux is based on open source system means provide the source code for desire program.

Personal firewall provides more security for surfing on internet as well as option to programmer to activate & deactivate the firewall.

Designing firewall in linux system also helps to know unique features of linux OS , bonding between Internet & Firewall and safe surfing of websites over the large network.

Future Scope: -

1. In Future Firewall will also work like Intrusion Detection Device.
2. Multifunction firewall can be designed to share workload.
3. Linux based firewall can implement newer technology like AJAX.
4. Firewall policies can minimized tension between web based innovation & strict network security system.

References

Books :-

1. Wes Noonan, Ido Dubrawsky --- Firewall Fundamentals
2. Philip Costigan P.C. SCADA LINK Supervisory Control and Data Acquisition --- Firewall for linux
3. Christopher Negus – Linux Bible 2010 Edition
4. Sandra Palumbo – Secure Works : The Information Security Experts
5. Raymond Blair & Arvind Durai – CISCO secure firewall : service module
6. Wes Noonan, Ido Dubrawsky – Firewall fundamentals
7. Elizabeth D. Zwicky, Simon Cooper, and D. Brent Chapman - Building Internet Firewalls, Second Edition , June 2000
8. Lucian Gheorghe - **Designing and Implementing Linux Firewalls and QoS using Netfilter, Iproute2, NAT and L7-Filter**
9. Martin Krzywinski -- Port knocking
10. S. Patton , D. Doss and W. Yurcik -- Open source versus commercial firewalls: functional comparison
11. **Jeff Regan -- An Intro. to Using Linux as a Multipurpose Firewall**

Websites:-

1. Essential of firewall -- http://www.checkpoint.com/resources/firewall/
2. Firewall distribution -- http://www.thegeekstuff.com/2010/02/top-5-best-linux-firewalls/
3. Homemade firewall with linux http://www.schaik.com/wwwillem.html
4. Build a firewall in linux http://www.ehow.com/how_5089603_build-firewall-linux.html
5. How to build a simple linux firewall http://www.cyberciti.biz/
6. Creating a linux firewall for home network http://www.governmentsecurity.org/forum/index.php?act=idx&s=78854aea2a226f417e798e1c857244eb
7. Firewall builder user guide http://www.fwbuilder.org/docs/users_guide/compile-install-detail.htm
8. The Perfect linux firewall PART I http://www.howtoforge.com/forums
9. Firewalls with linux http://linuxconsultant.info/
10. White papers for firewall http://www.bulwarks.com
11. Setting up your own firewall http://rr.sans.org/firewall/IPF.php
12. How to build an open BSD firewall http://www.cpio.org/obsd_firewall.html
13. Guide to free firewalls http://www.vpnlabs.com/vpn-categories/Firewall/2/index.html
14. How to create Linux computer firewalls. http://www.suite101.com/profile.cfm/FleurHup1234