LINUX FIREWALL

Feasibility and Design Report

Aman Agarwal	(15114006)
Anubhav Jain	(15114014)
Devesh Masane	(15114023)
Dhanraj Sahu	(15114024)
Harsh Kumar Bansal	(15114033)

FEASIBILITY

We want to built a firewall to satisfy the following requirements:-

1) All datagrams entering or leaving the intranet pass through the firewall, which examines each datagrams and blocks those that do not meet the specified security criteria. Our objective is to build a firewall that blocks unauthorized access while permitting authorized communications using packet filtering.

2) The firewall can BLOCK or UNBLOCK packets according to a set of rules. The rule will be added to the firewall and all incoming TCP packets from a certain source IP network to a destination port will be blocked. And we will manipulate the network packets according to these rules.

Possible Solutions:-

There are mainly 2 types of firewall hardware and software and we are going to built only software firewall over hardware firewall due to following reasons::

1. Normally, a dedicated hardware firewall costs more than a software firewall.

2. Hardware firewall is Difficult to install, and upgrade.

3. Hardware firewall takes up physical space, and involves wiring.

4. The individual system's operating system-based firewall can provide a great security.

5. Software firewall are ideal for personal or home use , They can be configured or reconfigured easily.

6. Through the software firewall, we can restrict some specific application from the Internet. This makes the software firewall more flexible.

7. The software firewall give users complete control on their Internet traffic through a nice user friendly interface that requires little or no knowledge.

There are several types of software firewall techniques and we are going to use packet filter techniques due to following reasons:-

1. Packet filter technique looks at each packet entering or leaving the network and accepts or rejects it based on user-defined laws. Packet filtering is fairly effective and transparent to users.

2. Filter based firewalls are configured with a table of laws that characterize the packets that they will, and will not, forward. These laws take into account the IP addresses of source, destination, ports, protocol etc.

3. A packet filter is the mechanism requires the manager to specify how the router should dispose off each datagram. For example, the manager might choose to filter (i.e. block) all datagrams that come from a particular source or those used by a particular application, while choosing to route other datagrams to their destination. When a datagram first arrives, the router passes the datagram through its packet filter before performing any processing, if the filter rejects the datagram, the router drops it immediately.

4. The filter can be configured in two modes. In the first mode, the default action can be defined to route the datagram while the IP combinations in the laws define the datagrams that need to be blocked. The other configuration is the reverse, i.e. the default action be blocking for the datagram and only the datagrams which abide by the filter laws are routed correctly. We are going to use first mode because it is more faesible and effective in small linux system than second mode.

So, we are finally going to use software firewall using packet filter technique on linux module for our purpose to satisfy the mentioned requirements.This is most effective and feasible solution for given solution.

APPLICATIONS

1. A Firewall Protects Your Computer From Unauthorized Remote Access:

One of the worst things that could happen to your computer is if someone attempts to take control remotely. Seeing the mouse move around on your monitor as a remote intruder usurps your digital kingdom, assuming control of your data, is petrifying. With a correctly configured firewall (and a modern OS) you should have remote desktop access disabled, thus preventing hackers from taking over your computer.

2. Firewalls Can Block Messages Linking to Unwanted Content:

The Internet has a lot of bad code traversing the cyberspace, waiting to pounce on unprotected PCs. While your ISP can help prevent this, its unlikely that theyll be able to long-term. Now-a-days,operating systems check if there built-in firewall is active or not,or if a third party firewall like an anti-virus is installed or not.

3. Firewalls Make Online Gaming Safer:

Various malware has been developed that targets online gamers, existing on unsecured or recently compromised game servers. While game publishers usually keep on top of security on their servers, its always a good idea to have a firewall enabled before you start playing online games. Any attempts by hackers to use their malware to get into your system will be blocked, leaving your system secure.

4. You Can Block Unsuitable or Immoral Content With a Firewall:

This type of blocking is usually found in parental control applications. The content filtering comes with some domestic security suites (mirroring the actions of the corporate-focused firewalls), usually in close proximity to the firewall.

5. Firewalls Can Be Hardware or Software:

Firewalls dont necessarily have to be software. Hardware firewalls are found in most homes, built into the router. Accessing these firewalls is possible by using the administrator credentials for the router (make sure youve changed the default password), and once youve signed in you should be able to review the options and change them if necessary.

CLASSES AND FUNCTIONALITIES

Firewall is a system designed to prevent unauthorized access to or from a private network. All datagrams entering or leaving the internet pass through the firewall, which examines each datagrams and blocks those that do not meet the specified security criteria. Our objective is to build a firewall that blocks unauthorized access while permitting authorized communications using packet filtering.

In packet filtering, we look at each packet entering or leaving the network and accepts or rejects it based on user-defined laws. Packet filtering is fairly effective and transparent to users, but it is difficult to configure. In addition, it is susceptible to IP spoofing.

Our project can be divided into two functionalities :

1. Kernel Module
2. UserSpace Program (Configuration Utility)

Our firewall can BLOCK or UNBLOCK packets according to a set of rules. The rules are set by a user space configuraiton utility program. For example,

```
./mf in scrip 10.0.2.15 srcnetmask 255.255.0.0 destport 80 proto TCP action
BLOCK
```

The rule will be added to the firewall and all incoming TCP packets from source IP 10.0.0.0/16 network to destination port 80 will be blocked. This configuration tool parses the user commands and sends instructions to the kernel module mf_km.ko through a proc file /proc/firewall. Then based on user commands, mf_km.ko can add/delete/print firewall policy. The mf_km.ko intercepts network packets arriving or leaving system network interface, and filters (either pass or drop) the packets based on firewall policy set by user.

Thus the firewall configuration utility can add, delete and print rules and the kernel module will manipulate the network packets according to these rules.

TOOLS AND PROGRAMMING LANGUAGES

The firewall implementation is done as a Linux kernel module. The implementation also has a simple configuration program for users to configure the firewall in user space and procfs virtual file system is used to pass information between user space and kernel space.

All programming is done in C and shell script.

Various tools that are used are follows:

- **Command line parsing API in GNU glibc** : Many Linux programs are command line based and sometimes the options can be complicated. Luckily, the GNU C library glibc provides some APIs to simplify the command line option parsing.
  Specifically, there are two methods for parsing the commands, getopt and getopt_long. getopt() is used to parse the single character option, and getopt_long() works with both long options and single-character options.

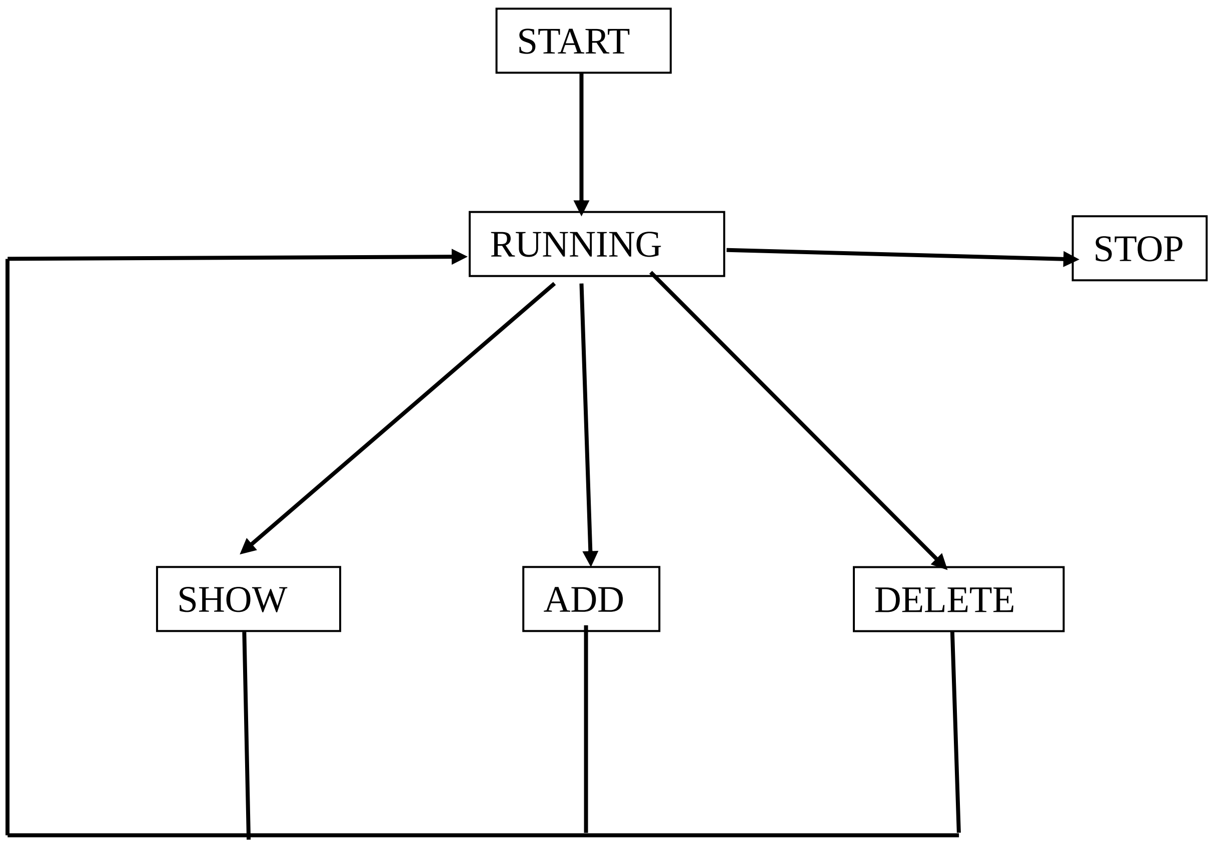
- **A linux kernel module** : Programming a linux kernel module is a tricky part. It has some predefined data structures only which we can use. We will use linked-list data structure predefined in linux kernel. Linux kernel has a special linked-list as a built-in data structure defined in /lib/modules/$(uname -r)/build/include/linux/list.h. Compared with common implementations of linked list, the Linux kernel version allows one to embed the pre-defined linked-list node into any data structure to form a linked list.

- **Linux proc file system** : procfs is a software created virtual file system that mounted to /proc directory at boot time. It was originally designed to provide information about running process of the Linux system, but has gone far beyond its original purpose as Linux kernel development proceeds. It can act as a bridge connecting the user space and the kernel space. User space program can use proc files to read the information exported by kernel.

- **Iptables/ Netfilter** : The basic firewall software most commonly used in Linux is called iptables. The iptables firewall works by interacting with the packet filtering hooks in the Linux kernel's networking stack. These kernel hooks are known as the netfilter framework. Every packet that enters networking system (incoming or outgoing) will trigger these hooks as it progresses through the stack, allowing programs that register with these hooks to interact with the traffic at key points. The kernel modules associated with iptables register at these hooks in order to ensure that the traffic conforms to the conditions laid out by the firewall rules.

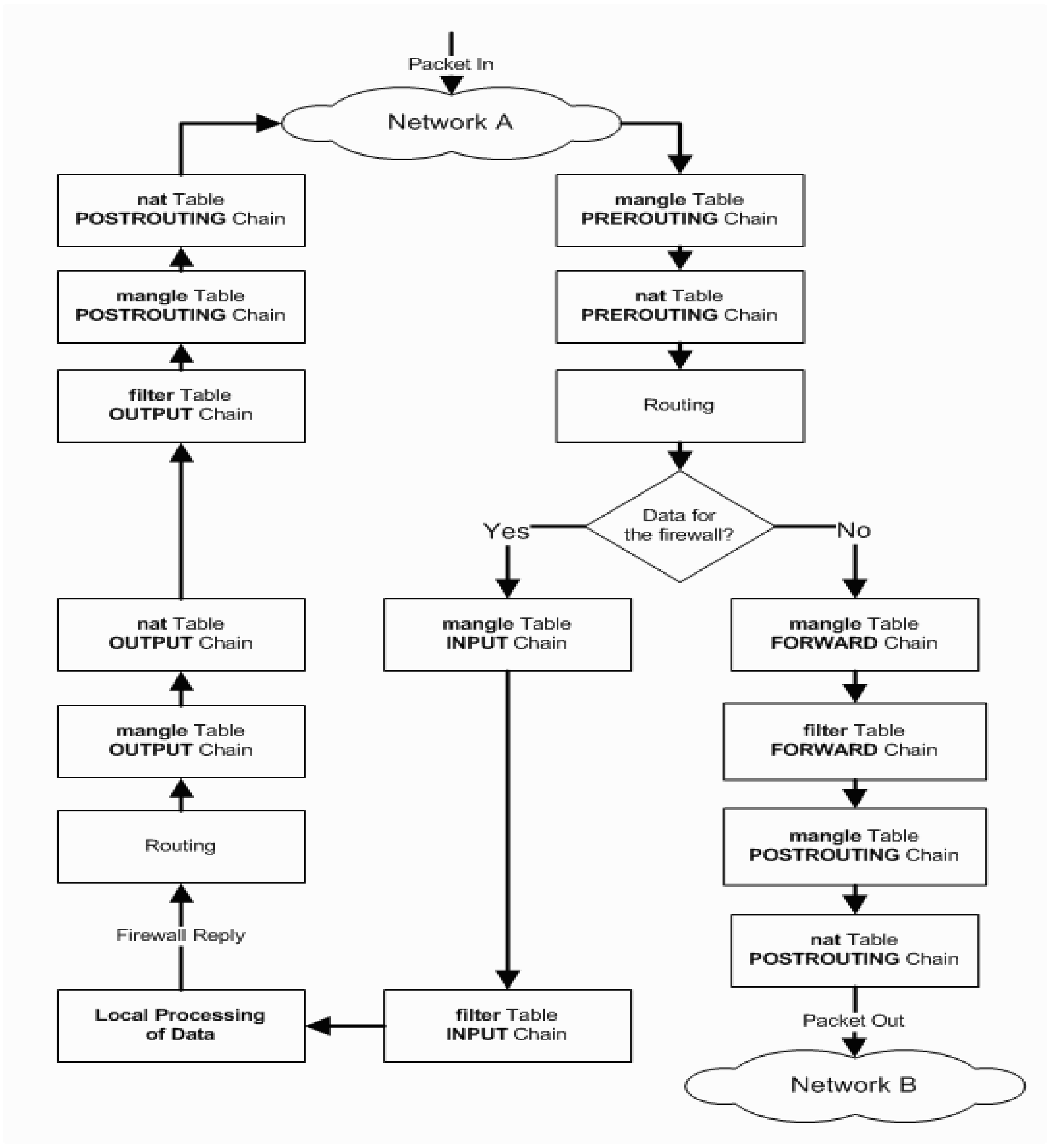
DESIGN

Linux Firewall Configuration Utility



Linux Firewall kernel module working

PREROUTING: Triggered by the NF_IP_PRE_ROUTING hook.
INPUT: Triggered by the NF_IP_LOCAL_IN hook.
FORWARD: Triggered by the NF_IP_FORWARD hook.
OUTPUT: Triggered by the NF_IP_LOCAL_OUT hook.
POSTROUTING: Triggered by the NF_IP_POST_ROUTING hook.


Packet In
Network A
nat Table POSTROUTING Chain
mangle Table PREROUTING Chain
mangle Table POSTROUTING Chain
nat Table PREROUTING Chain
filter Table OUTPUT Chain
Routing
Yes
Data for the firewall?
No
nat Table OUTPUT Chain
mangle Table INPUT Chain
mangle Table FORWARD Chain
mangle Table OUTPUT Chain
filter Table FORWARD Chain
Routing
mangle Table POSTROUTING Chain
Firewall Reply
nat Table POSTROUTING Chain
Local Processing of Data
filter Table INPUT Chain
Packet Out
Network B

CHALLENGES

These are the challenges a company generally faces when designing a firewall:

1. **Cost** : The cost of adding firewall "brains" to the inside of the network is substantial, especially compared to the continued cost reduction of standard networking switches and routers.

2. **Performance** : Firewalls have proven themselves on Internet-speed links, but most enterprises have significantly higher flow rates within the network than towards the Internet. Common tasks such as file sharing and backups would bring a firewall designed for Internet speeds to its knees on a 100 Mbps Ethernet link.

3. **Management :** Most firewall vendors have found it challenging to define management in terms of many-to-many relationships. Generally, the three-legged firewall is about as sophisticated as they get, and having multiple firewalls in a single configuration has been a difficult problem to solve elegantly. While some vendors now cleanly handle dozens of ports, extending this to thousands and managing access control dynamically across hundreds of network elements is a challenge.

4. **Policy** : Network managers find it easy to define security policy as it relates to the Internet, but find it much more difficult to describe what are permitted and denied flows within the organization itself. If you can't define policy, then you can't design a firewall to implement that policy.

5. **Authentication :** Users on the network have traditionally not authenticated themselves at layers 2 and 3; they connect to applications and authenticate at that level. However, for network-layer security, authentication of "who is out there" must be tightly bound to the user.

6. **Binding :** As packets flow through a network, it's difficult to assign security policies anywhere but at the extreme edge. The basic unit of access control is usually the user, but packets aren't bound to a particular user. Making a tight binding between a user-based policy and a packet that has an ephemeral IP address is a problem for which there's no standards-based solution.

REFERENCES

- https://www.veracode.com/security/firewall-security
- http://public.lanl.gov/cdi/projects/firewall/index.html
- https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture
- http://searchsecurity.techtarget.com/feature/Top-challenges-facing-defense-in-depth-firewall-technology
- http://www.juniper.net/us/en/products-services/what-is/firewall-design/