International Open Source Network
An Initiative of the UNDP's Asia-Pacific Development Information Programme supported by International Development Research Centre.

Free/Open Source Software Network Infrastructure and Security

Gaurab Raj Upadhaya

V3.0

Published by
the United Nations Development Programme's
Asia-Pacific Development Information Programme (UNDP-APDIP)
Kuala Lumpur, Malaysia

Web: http://www.apdip.net/
Email: info@apdip.net

ISBN:

Table of Contents

Preface

INTRODUCTION

Introduction to FOSS and GNU/Linux

What is open source software?

Software whose inner working code are openly available for public and peer review is referred to as Open Source Software. The source code may be available to the public under different terms and conditions, but the basic premise is that people are free to review the code and make changes.

The basic idea behind open source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.

We in the open source community have learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see the source and everybody else must blindly use an opaque block of bits.

What is Free Software ?

Free software as the name implies, is FREE, or you are neither required to pay for the use of the software nor pay any licenses for multiple. Free software may or may not be open sourced. It is thus getting common for many companies to release basic version of their software for free, but charge for a more advanced Software.

What is FOSS ?

Free and Open Source Software are software that are both open sourced and are free. Thus the emphasis on both aspects of the software.

What is GNU GPL?

GNU stands for GNU's not Unix (It's a recursive acronym). GPL stands for General Public License. These are normally used together to refer to the license provided by the Free Software Foundation. The licenses for most software are designed to take away your freedom to share and change it (the most common being the End User License Agreement - EULA). By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most

of the Free Software Foundation's software and to any other program whose authors commit to using it. The above definition of open source is drawn from this license.[1]
< http://www.gnu.org/philosophy/free-sw.html >

What is Linux?
(From Linux.com)

Linux is an operating system that was initially created as a hobby by a young student, Linus Torvalds, at the University of Helsinki in Finland. Linus had an interest in Minix, a small UNIX system, and decided to develop a system that exceeded the Minix standards. He began his work in 1991 when he released version 0.02 and worked steadily until 1994 when version 1.0 of the Linux Kernel was released. The current full-featured version is 2.6 and development continues.

Linux is developed under the GNU General Public License and its source code is freely available to everyone. This however, doesn't mean that Linux and its assorted distributions are free -- companies and developers may charge money for it as long as the source code remains available. Linux may be used for a wide variety of purposes including networking, software development, and as an end-user platform. Linux is often considered an excellent, low-cost alternative to other more expensive operating systems.

Due to the very nature of Linux's functionality and availability, it has become quite popular worldwide and a vast number of software programmers have taken Linux's source code and adapted it to meet their individual needs. At this time, there are dozens of ongoing projects for porting Linux to various hardware configurations and purposes.

What is GNU/Linux and Linux Kernel ?

In the past few years, the term 'Linux' has evolved to refer to both the main operating system developed initially by Linus, and also other utilities that are part of many Linux distributions. In order to avoid confusion, the term GNU/Linux refers to the the collective Operating system and the term Linux kernel refer to the main operating system code. The Linux Kernel interacts with the hardware, provides networking utilities, memory management and other basic services for other software to be able to run on the hardware. Linux Kernel is the main basis for the GNU/Linux operating environment. The officlal linux kernel is always available from www.kernel.org.

1 A better reference to GNU GPL and open source in general is Free/Open Source Software: A General Introduction, another primer published by the IOSN.

GNU/Linux consists of the Linux Kernel along with other Open source software, all of which are released under the GPL. The Debian Distribution (www.debian.org), is an example of GNU/Linux.

What the the Network Operator and Service Providers's perspective on FOSS ?

All network operators need to provide a set of services, for which they need reliable systems. Services like web surfing, e-mail service, and DNS form the backbone of all network providers. They also need reliability and scalability as the number of users grows. They normally operate in a high volume business. This warrants a reliable system, which can be scaled quickly and that which can run those services efficiently. Free / Open source software provides added advantage over closed source services, not only in terms of monetary value but also in the form of huge flexibility in operations.

On the client side too, a GNU/Linux machine makes it easier to manage and remotely troubleshoot. It also gets rid of many virus problems frequently associated with Microsoft Windows. Software copyright and piracy is an increasing concern worldwide, and using GNU/Linux makes sure that you are always on the side of the law.

Why GNU/Linux for Networking ?

GNU/Linux is the open source operating system endorsed by the Free Software Foundation. While there are other operating systems like *BSD available, which also distributes source codes for free, GNU/Linux is the major operating system which confirms to the GPL.

From a networking perspective, the UNIX operating system is the pioneer in multi user systems. UNIX and UNIX like operating systems like BSD and GNU/Linux have been designed with the network performance and securitty in mind. In past, most network systems have relied on expensive hardware and proprietary Unix platforms to get maximum benefits. With GNU/Linux, the same kind of benefits can be achieved on much cheaply available hardware. At the same time, availability of the source code enables Operators to create their own extensions and tune the software to their own needs.

Also, most servers and programs that are required for network operations are available on GNU/Linux, and a lot of additional utilities are being developed constantly to provide added advantages.

NETWORK CONCEPTS & ARCHITECTURES

What is a Network?

A Network is the mechanism that enables distributed computers and users, to communicate and share resources. Sharing of resources may mean sharing of a printer, scanner or even a large hard disk on a particular computer. A computer network connects two or more computers, printers so that all users on the network have access to other resources.

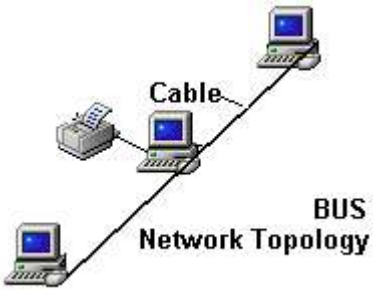
What are the two major types of networks?

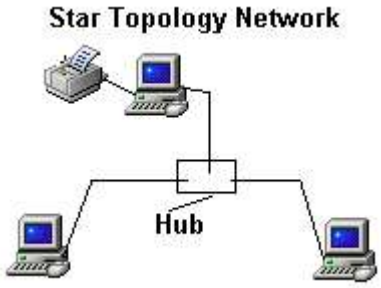
There are two major types of computer networks, viz, a) server based network, b) peer to peer network.
In a *server based network*, there exists one of more central computer(s), on which the resources are centralized. The network is also more or less controlled by the server. Servers are generally superior to normal computer in terms of processing power, memory and hard disk space.
In a *peer to peer network*, all the computers on the network are responsible for sharing of resources. That particular computer shares resources it has. All the computers are servers and all are clients. There is no central control over the resources. Peer to Peer networks are based on peer relationships amongst computers in the network.

What is a Network Topology ?

Topology is the arrangements of computers in the network. There are two main topology used commonly: BUS topology and Star Topology.

The simplest form of a network is the *BUS topology*. All computers are connected in a series to a single stretch of cable. This effectively means the entire length of the cable must be intact for the network to work properly. This makes it prone to failure in larger networks.
A Hub is used as the central exchange point in a *Star topology*. All the computers are connected to the Hub in what form a star - wires going in all directions from the hub. Star topology is easier to maintain and is easier to expand.





What are the different networks Architectures ?

There are different ways in which two computers can communicate at the electronics level. These different ways are effectively different network architectures. The well-known architectures are, Ethernet and Token Ring. But the most popular is Ethernet.

Ethernet

Ethernet is the most popular network architecture that exists today. This mechanism is independent of any vendor, which is why this has proven one of the most successful technology. Basic Ethernet architecture operates at speed of 10 Mbps (Mega Bits per second). Other variations include Fast Ethernet, operating at 100 Mbps, and Gigabit Ethernet operating at 1 Giga bits per second.

What are the different Network Components ?

Network Cabling

Cables play fundamental role in any network. They exists at the most primary level of the network. Two types of cables are in common use - co-axial cables (similar to cable TV cable) and twisted pair cable (similar to telephone cables). Co-axial cables are generally used with T-connectors in a Bus topology network. Twisted pair cables are used mostly in Hub based network.

Network Adapter Cards

Network Interface Cards (NIC) or Network Adapter Cards as they are commonly known provide an interface on the computer to connect to the network cable. Apart from few special computer, most computer do not come ready for networking. So, an additional card is inserted inside the computer which will then connect to the network cable. These are called NIC cards.

How does a Network Functions ?

How do Networks operate?

Computer Networks operate at different layers, each of these layers independent of each other. Network layers as these layers are commonly known provide general understanding of any computer network. A very simple form of these layers is illustrated below.

Operating Software Layer (with TCP/IP protocol)	Network Applications	Application Layer
	TCP / UDP Protocol	Transport Layer
	IP Protocol	Network Layer
Hardware Layer	NIC Drivers	S/w H/w layer
	NIC Card Layer	Electronic Layer

What is a NIC Driver

NIC driver provide a layer between the hardware and the software in any network. All Network card manufacturers provide NIC drivers. NIC drivers sometimes also come with the operating system, but it is always better to use the driver that comes with the card.

Network Protocols

Ans.: Network protocols provide a way in which diverse computer hardware can communicate with each other in a standard way. These protocols follow set rules unique to themselves. The common network protocols are TCP/IP, and NetBEUI. There were other protocols developed and used in the past, but TCP/IP has replaced most others. NetBEUI is developed by Microsoft and is native to the windows environment. TCP/IP is the protocol of the Internet and is the most popular protocol today.

What is TCP/IP

TCP/IP stands for **Transmission Control Protocol / Internet Protocol**. These are the basic protocol of the Internet. There are other protocols in TCP/IP like User Data gram Protocol (UDP). IP is the network layer and provides the basis for the network through the IP addresses. IP addresses on all computer must be unique. TCP and UDP are the transport layer and are responsible for transport of network data packets.

IP addresses follow the format **nnn.nnn.nnn.nnn**, (where **nnn** is less than 255). A network mask separates the network address and the host address.

E.g, An IP address **192.168.0.1** with net mask **255.255.255.0**, means that the network address is **192.168.0** and the host address is **1**.

For internal network use, the IP range defined in RFC[2] 1918 should be used. Examples are 192.168.0.0, 172.16.0.0 and 10.0.0.0

What are part of Network Operations ?

What is a Network Operating System?

Network Operating System (NOS) is a network capable operating system. This operating system handles the network resources, provides services like file and printer sharing. It is also capable of being an application server.

NOS are different from normal desktop operating system, in a sense that the NOS are capable of advanced memory management, network management and are built with a network in mind.

What are Users and groups?

Any network operating system provides some form of to its resources and services. The simplest of this is the username - password pair. All users on the network are given a unique username and they are also given a secret password. Only after verifying that proper username and password has been supplied, the NOS provides access to the designated network services.

A group is a collection of users in any network. A group is created to give similar rights to a group of users. Groups make administration and management easier.

What are rights ?

All resources in a network may not be available to everyone. It is necessary to shield some parts of the network from unauthorized people. So, all NOS provide rights to its users and groups. Users and groups are given only the required rights, while the more powerful rights are limited to the administrators.

2 RFC stands for Request for Comments, which is the series of documentation for to Internet Standards documents. RFC 1918 means RFC number 1918, which defines private use IP addresses. RFCs can be downloaded from http://www.faqs.org/rfcs/. We'll refer to many RFCs for further reference through out the primer.

Who is supreme in a Network?

In most NOS, there's a super user or administrator who has absolute rights in the network system. In Unix and GNU/Linux environment “root” is the super user. An administrator is the supreme user in Windows network. In NetWare network supervisors are super. These super users are capable of doing anything in the NOS. These accounts should be used with care.

How can we monitor network performance?

When users start complaining that the network is too slow, we know that there are problems in the network. It requires monitoring of network performance to find where the fault lies. It may lie in the network hardware or in the network software.

Inferior grade cables and loose cable connection are cause of 80% network problems and performance issues. High quality and prescribed cables and connectors should be always used. Another cause may be the server hardware. Servers are happy when they have more RAM. If the performance is real bad, it might call for the replacement of an older switch.

NOS with better memory management should always be used. Unnecessary services running on the server should be stopped to get the maximum from the server. Every NOS provide some utility to monitor the network performance as well as the server performance.

How do I Troubleshoot my network ?

First step is to check the cables. After that the power connections to the network switch or resources on the network. If the problem is not solved, referring to the different services running on the server should be done.

Can you give me an example of the use of FOSS in networking ?

The Internet is the biggest example of a successful implementation of Free and Open source tools and software. Internet is the largest network in the world. The basis of the Internet is the TCP/IP protocol, major implementation of which are open source. While, it's true that the Internet has helped in the growth of the FOSS in general, FOSS has also made Internet a much more reliable network. A much more detailed information about the Internet standards process and how open source is inherent in it can be found in RFC 1718.

But arent' there some limitations ?

Yes there are limitations. Most open source software are produced confirming to open standards and processes. There are proprietary software and processes which may not have an open source equivalent. Sometimes, it may also be necessary to use proprietary software to access certain types of hardware. Another limitation may be the really rapid pace of development of Free/Open Source Software. To an average user, this may be confusing, but for service providers this would definitely be an advantage to be taken.

IMPORTANT NETWORK FUNCTIONS USING FOSS

The Domain Name System

The Domain Name system (DNS) is the glue that keep the technical side of the Internet connected to the users of the network. Simply stating, DNS provides names to numbers translation and vice versa. Internet is based on the TCP/IP protocol, and as such computers on the Internet know path to each other only through the IP Addresses. For simple human folks, remembering the numerical IP address of each computer on the Internet is not possible. There in comes DNS.

The main purpose of DNS is to map names to objects on the Internet. The objects may be an IP Address or it may be the identification of mail servers, name servers and even personal telephone numbers. It is common sense that names are easier to remember than numbers.

The earlier version of DNS was the hosts.txt file, which was manually maintained by the SRI-NIC (Stanford Research Institute-Network Information Center) in the early days of ARPANET[3] in 1970s. This hosts.txt file was updated on a single computer and pulled by computers all over the world. While this method worked for some time, name collision was imminent when more hosts were added to the network. The flat single file mapping was simply not scalable. Thus DNS was created and has been defined in RFC 1034 and 1035.

DNS consists of three components.

-- DNS Name space – the domain names
-- DNS Server – the server which hosts the namespace
-- Resolver – the client which uses the server.

The three components work in tandem to create a viable solution to name resolution. DNS is designed in a way that all data is maintained locally but is retrievable globally. The data is distributed amongst different name servers and no single computer has all the data. The data is internally always consistence, thereby providing a stable system. DNS is designed in a way that any device can send DNS queries to a server.

DNS Name Space

DNS name space is a concept. Names are references to address, objects and physical presences. Names normally refer to a human understandable

3 Advanced Research Program Agency Network is considered the precusor to the current Internet.

reference that identifies an endpoint. DNS Namespace, as the word itself says, is the name space defined for Domain Name Service. on the Internet, domain names provide an heirarchy for DNS Name space. They provide an order to how Internet addresses are identified.

Let's take an comparitive example. If someone where to send me a letter, it would say

Gaurab Raj Upadhaya
205/8 Sahayogi Marg, Kathmandu, Nepal

In the above example, the address provides a way in which a letter can be sent to me from anywhere in the world. At the same time, if someone were to send me an e-mail, it could be either of these

gaurab@wlink.com.np
gaurab@lahai.com

In both cases, while the e-mail ends up in the same mailbox, it travels differently to that address. What needs to be understood is the domain heirarchy that makes the e-mail work. It is part of the DNS name space. Domain names are the implementation of DNS Namespace. The domain name is a tree shaped structure. The top most level of the the DNS tree is called the 'root'. The 'root' is referenced with a '.' (dot). Immediately below the 'root' are the country code top level domain (ccTLD) and the gobal top level domain (gTLD). These two top levels are pre-defined and fixed on a global scale. The ccTLDs are asssinged as per the ISO xxx standard. The gTLd are decided by ICANN (Internet Corporation Assigned Name and Numbers). Examples of ccTLDs are .np, .in, .my, .uk, .se etc. ccTLDs are two letter codes. Examples of gTLDs are .com, .org, .net, .gov, .edu, .mil, .info, .name, and .aero.

Domains and sub domains

Below the TLDs are the user level space. These are commonly referred to as the domain names. For example, anything under .net is in the net domain, any thing under .uk is under the UK domain. An extension to this concept is anything under the lahai.com, i.e, www.lahai.com, evo.lahai.com, are under the lahai.com domain.

Every domain created under an upper level domain is referred to as a sub domain. Thus in above example, evo.lahai.com is a subdomain under lahai.com.

Zones and Delegation

In computer terms, each DNS namespace is reflected by it's zone file. It is also referred to as the 'administrative namespace'. Each domain or sub domain on a name server has a respective zone file. The zone file is the main file which provides the mapping.

But, what makes DNS so scalable is the ability to define delegation for sub domains to other servers and other zone files. Thus, the root zone file delegates ccTLD and gTLD functions to their respective servers. And each cctLD or gTLD server furhter delegates specific domain information further to their registered owners. Thus the actual names to object mapping will be provided only by the authorative zone for that domain. It can be equalled to a parent delegating authority to its child.

---------- Box ----------

Some DNS terminology

DNS = Domain Name Service
FQDN = Fully Qualified Domain Name
RR = Resouce Record
ccTLD = country code Top Level Domain
gTLD = global Top Level Domain

--------- box closed ---------

Name Servers

Name servers hosts the DNS zone files. They answer queries placed to them. Name servers are of two common types.

- Authoritative Name Servers
    - master
    - Slave
- Non-Authoritative Name Servers
    - caching nameservers
    - caching forwarders

Most implementation are a combination of two or more types.

Authoritative Name Servers

Authoritative name servers host the main zone files for the designated domain. The authoritativeness of the name server is based on the delegation from the upper level domain. Thus for any server to be authoritative for evo.lahai.com domain, it has to be delegated in the lahai.com zone file.

The master file is where the main file is hosted. The slave mirrors the file from the master. There can be multiple slave servers to one master servers. A single master server can support more than 20 million names, but it might not be a good idea. Different DNS server software are capable of handling upto huge number of DNS queries. Commonly cited example is of 300,000 queries per second. Changes to the master copy of the database are replicated according to timing set by the administrator

Recursive Name Server

Recursive Name server are not authoritative for all domains for which it is serving data. They act on behalf of other clients and cache the result in it's memory. If the same query is sent within a pre-defined time period, then instead of searching the entire DNS structure, it serves the data from the cache. In case of caching forwarders, the server users another DNS server to get it's result. When the data are forwarded to the client, it's marked as non-authritative.

Mixed implementation

In smaller organizations, a single Name server can be used for multiple purpose. A server can be authorative of a select few domains but may also serve as a non authoritative caching server for other domains.

Resolver

The Resolver is the client that asks the server for the DNS data. The resovler is normally implemented at the operating system level in form of a library, so that multiple applications can use it.

Using BIND for DNS

BIND (Berkeley Internet Name Domain) is an implementation of the Domain Name System (DNS) protocols and provides an openly re-distributable reference implementation of the major components of the Domain Name System, including:

- a Domain Name System server (named)
- a Domain Name System resolver library
- tools for verifying the proper operation of the DNS server

The BIND DNS Server is used on the vast majority of name serving machines on the Internet, providing a robust and stable architecture on top of which an organization's naming architecture can be built. The resolver library included in the BIND distribution provides the standard APIs for translation between domain names and Internet addresses and is intended to be linked with applications requiring name service.

Getting and Installing BIND

BIND is normally installed by default in most GNU/Linux Installation. Otherwise, you can always get a copy from the BIND home page at http://www.isc.org/products/BIND/. The installation in different distributions of GNU/Linux may differ. It's best to follow the distribution guide for installation.

Configuration of BIND

The BIND configuration has to be done again in three stages.

First the client side or the resolver library needs to be configured followed by the server itself and finally the tools.

Resolver configuration

Resolver is the client side of the DNS system. Even if you do not run a DNS sever on the computer, you'll need the resolver installed. Naturally, n order to configure BIND, you first need to configure the resolver library. This is done by configuration of following files

`/etc/host.conf`

This file specifies how the host name resolution is performed. This has been made obsolete, but older installation may still use it.

/etc/nsswitch.conf

This file has replaced host.conf. This file specifies the order in which name resolution takes place. It tells the computer the order in which it should try to convert names into IP addresses.

```
# /etc/nsswitch.conf
# Any line starting with a '#' sign is a comment.
# In this example, hosts are resolved through DNS, then from
files

hosts:          dns files

# only files are used for network name resolution

networks:       files
```

The default file created during installation is most often sufficient.

/etc/resolv.conf

Resolv.conf is the basic DNS configuration file, which specifies the DNS server and the domain name. The three key words here are 'domain' 'search' and 'nameserver'.

```
# /etc/resolv.conf
# Our domain
domain       gaurab.org.np
# Default search domains in order of priority
search      gaurab.org.np lahai.com.np
#
# We use the local server as the first name server.
nameserver  127.0.0.1
# we have second name server at up stream provider.
nameserver  202.52.255.47
```

This file is also created during the installation, and if your network configurations haven't changed, you can leave it unchanged.

Server Configuration

'**named**' is the name given to the DNS server daemon. A daemon is a unix software that runs continiously on the server as a service. The DNS server is thus commonly referred to as the 'name daemon' or just 'named'. The file name of the DNS server is also 'named'

For BIND versions 4.x.x, named used the configuration file /etc/named.boot. But in the later versions of BIND (8.x.x), the configuration file is / etc/named.conf.

For our purpose we use the /etc/named.conf. In the following example, a master DNS for the domains gaurab.org.np and mos.com.np is specified. A slave DNS for domain wlink.com.np is also shown.

```
//
// /etc/named.conf file for ns.lahai.com
// in this file '//' is the comment.

// you specify the default data directory for DNS. Now all DNS
// related files should go into /var/named or any other
// directory as specified.

 options {
        directory "/var/named";
 };

// First you need to add the DNS roon zone file name. It's
there
// by default.

zone "." {
        type hint;
        file "named.ca";
 };

// Now we are specifiying a master domain called lahai.com
// whose information is stored in the file 'named.lahai.com'

 zone "lahai.com" {
        type master;
        file "named.lahai.com";
 };

// the whole thing can also be done in a single line.

zone "gaurab.org.np" { type master; file
"named.gaurab.org.np"; };


// Now this server is also a slave for another domain
“wlink.com.np'
```

```
zone "wlink.com.np" { type slave; masters { 202.79.32.33; };
      file "slave/named.wlink.com.np"; };

 zone "0.0.127.in-addr.arpa" { type master; file
"named.local"; }
```

This file sets of the stage for adding the real data about host name and IP addresses. The /etc/named.conf file can take a lot of additional configurations, but we are skipping them here.

After relevant entries have been made in the named.conf file, it is necessary to create the host name records for the corresponding domains. All the files should be placed in the directory specified by the `directory "/var/named";` in the named.conf file.

The **`named.local`** file provides reverse zone lookup for the loopback interface or the 127.0.0.0 network used by the loopback addresses. . The default file should be left unchanged.

The **`named.ca`** provides the root server information to the DNS server. The default should never be edited.

Now let's look at a sample DNS file for domain lahai.com (**`named.lahai.com`**)

```
; file /var/named/named.lahai.com

@               IN  SOA   ns.lahai.com. Gaurab.lahai.com. (
                          2004050801      ; serial number
                          86400           ; refresh: once per day
(1D)
                          3600       ; retry: one hour (1H)
                          3600000         ; expire: 42 days (6W)
                          604800          ; minimum: 1 week (1W)
                          )

; we are specifying three Name servers.

                         IN  NS    ns.lahai.com.
                              IN  NS    a.ns.hopcount.ca.
                              IN  NS    ns1.lahai.com.

; local mail is distributed on another server
```

```
                                IN  MX    10 mail.lahai.com.
                                IN  MX    20 ns.lahai.com.

; loopback address
localhost.       IN  A     127.0.0.1

; The glue records so that the NS records can resolve.

ns         IN    A     204.61.208.110
ns1        IN    A     202.79.55.14

; main DNS entry
www           IN  A     207.189.222.2
mail         IN  A    202.51.76.8

; Aliases  for the www machine.

ftp   IN  CNAME www
```

The above file is the main file for the domain 'lahai.com' . If i want to add additional names for the domain 'lahai.com' like pop3.lahai.com and smtp.lahai.com, then i should add it in the above file.

The order in which the name servers are listed in this file makes them master and slave. The first NS is always the master server and the other two are slave servers. Each time the master server is updated, it automatically sends a notification to the other NS servers listed in the file.

A Note about Reverse DNS

Most often the majority of DNS related problems are due to mis-configured Reverse DNS. Reverse DNS is the mapping of numbers into names, or the opposite of the the forward name resolution. Many applications use this property to verify that the network source IP address is valid. A common expample are SPAM or unsolicited commercial e-mail (junk e-mail) prevention software, which will refuse to accept mails from any domain that doesn't have a reverse DNS configured.

The reverse DNS works through delegation of the particular group of IP address from one of the Regional Internet Registries (RIR), which in the Asia Pacific Region is APNIC (www.apnic.net). Since ISPs are normally the APNIC members, they are responsible for configuring the appropriate reverse DNS for the IP addresses under use by them and their clients.

Since each computer has it's own loopback interface and the IP address associated with it, BIND comes with the default installation of the **`named.local`** file, which is the reverse DNS for 127.0.0.0 network. This file looks like the following.

```
; /var/named/named.local

$TTL    86400
@       IN      SOA     localhost. root.localhost.  (
                                      1997022700 ; Serial
                                      28800      ; Refresh
                                      14400      ; Retry
                                      3600000    ; Expire
                                      86400 )    ; Minimum
              IN      NS      localhost.

1       IN      PTR     localhost.
```

Administrating BIND DNS.

BIND includes a utility called **rndc** that allows you to administer the named daemon, locally or remotely, with command line statements. The rndc program uses the '**/etc/rndc.conf'** file for its configuration options, which can be overridden with command line options.

Before you can use the rndc, you can need to add the following to you named.conf file

```
/etc/named.conf

controls {
   inet 127.0.0.1 allow { localhost; } keys { <key-name>; };
 };

key "<key-name>" {
   algorithm hmac-md5;
   secret "<key-value>";
 };
```

In this case, the <*key-value*> is a HMAC-MD5[4] key. You can generate your own HMAC-MD5 keys with the following command:

```
dnssec-keygen -a hmac-md5 -b <bit-length> -n HOST <key-file-name>
```

A key with at least a 256-bit length is good idea. The actual key that should be placed in the <*key-value*> area can found in the <*key-file-name*>.

Configuration file /etc/rndc.conf

```
options {
   default-server  localhost;
   default-key     "<key-name>";
 };

server localhost {
   key  "<key-name>";
 };

key "<key-name>" {
   algorithm hmac-md5;
   secret "<key-value>";
```

4 A popular way to encrypt. It uses a one way Hash algorithm for encryption.

```
};
```

The <key-name> and <key-value> should be exactly the same as their settings in `/etc/named.conf.`

To test all of the settings, try the `rndc reload` command. You should see response similar to this:

```
rndc: reload command successful
```

You can also use the `rndc reload to` reload any changes made to your DNS files.

DNS tools

There are two common tools to test DNS. They are **`nslookup`** and **`dig`**. `Nslookup` is the older of the two, and is less preferred. You can use the **`dig`** utility to test the DNS service. Refer to **`man dig`** for details. Some people still prefer to use nslookup.

The Mail Server

Internet and e-mail were taken as synonym at earlier days of the Internet. Even today, more than quarter of the total Internet traffic is still e-mails. And it's not surprising that open source softwares rule the world of e-mail. On the Internet, e-mail messages work on the basis of SMTP (Simple Mail Transfer Protocol) which is defined in RFC 2821. Simple Mail transfer protocol is a really a simple protocol designed to make the transfer of e-mail messages between mail servers as easy as possible.

SMTP works in plain text, and communicates between the mail servers, which are also referred to as Mail Transfer Agents or MTAs. The most popular mail server software is 'sendmail'. Other example also includes Exim, qmail and postfix amongst others. Their closed source alternatives are Lotus Notes and Microsoft Exchange.

The beauty of the open source is the level of software complexity available. While Exim and postfix have a smaller footprint and consume a little amount of memory on the servers, sendmail is a complex beast that runs the busiest mail servers of the world.

A common example cited about the strength of open source mail servers is hotmail.com. In 2000, when hotmail.com was bought by Microsoft, they immediately tried to change the mail server software from sendmail based to a microsoft exchange based system. It was immediately brought back to the original state, as exchange couldn't cope with the loads of hotmail.com, which provides free e-mail for more than 10 million users.

Another great benefit of open source mail servers is the modularity of the software. Sendmail itself provides for a large number of extensions and provision for including modules. This makes it easier for developers to extend the software for their in-house needs. If you need to develop extension to your e-mail server to automatically handle different types of e-mails, open source software provides this function.

Other mail related protocols

There are other additional protocols that makes a fullly functioning e-mail system. The two main ones are Post Office Protocol (POP) and Internet Mail Access protocol (IMAP). These two protocol provides the end unser functionality for users. POP and IMAP protocols are used by e-mail software for accessing e-mails stored on a server. So if you use an e-mail client like Eudora or Thunderbird, then it'll use either the POP or the IMAP protocol to pull e-mails from your mail server to the local machine.

Anti SPAM features

One of the biggest advantage of free and open source software has been its extensibility. Nothing highlights this more than the available anti-spam features in mail servers. Today almost 80% of all e-mail messages are thought to be Unsolicited Commercial E-mail (UCE), commonly referred to as junk e-mail or simply SPAM. UCE not only consumes a lot of bandwidth and network resources, it is also a nuinsence to users and decreases productivity for organizations.

The best anti-SPAM tools available today are all open source. In order to stop spammers from sending junk e-mail, it is ncessary to identify their tools and origin, and what better than thousands of users contributing to identifying those people. The open source concept makes sure that not a single junk e-mail goes un-reported so that the origin can be idenified easily.
A common anti-SPAM technique is the use of RBL or Real Time Block Lists. The different RBLs lists the IP address of networks that are known as origin to huge amount of SPAM. Again the open nature of these lists, and the software like spam assassin makes it easier to tune the software to their own needs.

Eg. In a corporate environment that i consulted for, the users needed to send e-mail in BLOCK letters due to their nature of work. Now, if the entire e-mail were in BLOCK letters, most anti-spam tools identify them as junk e-mail. But because the company was using an open source solution, we were easily able to modify the code for the software to remove this criteria, only for mail originating within their network.

Using Sendmail for SMTP

Sendmail is one of the SMTP servers avialable under Linux. It is also one of the oldest open source software that is highly used. Many people consider sendmail to be too complicated and difficult to use. Sendmail has it's own pros and cons. Because it has a lot of features, it's complex piece of software. But at the same time, the basic operations of sendmail can be managed easily.

Sendmail configuration is handled either by directly editing the sendmail configuration file (no recommended) or through the use of the M4 marco language in creating a new configuration file from a set of variables.

Now we will deal with sendmail. Given here are the minimum necessary changes to the default sendmail installation.

Enabling network -ability in Sendmail.

Default installation of software in many distribution is limited to the mail server listening only on the loopback address[5], i.e, the server is not reacheable over the network. For sendmail to be reacheable from the network, you'll need to edit the appropriate like in the **`/etc/mail/sendmail.mc`** file. You should edit to remove 127.0.0.1 from the following line

```
DAEMON_OPTIONS(`Port=smtp, Name=MTA')
```

After that you'll have to run the m4 macro to create a new sendmail configuration files

```
[root@mail /etc/mail]#  m4 sendmail.mc > sendmail.cf
[root@mail /etc/mail]#  service sendmail restart
```

This should enalbe network reacheability for the sendmail daemon. There are also a lot of other options on the `sendmail.mc` file, that you can play around with.

Local Domain Names

Edit `/etc/mail/local-host-names` and add all domain and domain aliases that your site uses.

```
  # local-hosts-names -
       # include all aliases for your machine here.
  lahai.com
  gaurab.org.np
  ns.lahai.com

# some examples
   mail.you.com
  yoursite1.com
  mail.yoursite1.com
  yoursite2.com
  mail.yoursite2.com
  yoursite3.com
  mail.yoursite3.com
```

These are necessary so that sendmail accepts mail for these domains.

5 The IP address of the loopback interface is 127.0.0.1

Virtual Domain Users

But the above configuration does not entirely solve the problem of virtual domain users. For that use the virtsusertable feature. For that go to `/etc/mail/virtusertable`

```
# /etc/mail/virtusertable

#virtual e-mail address real username

user1@yoursite1.com           user1_yoursite1

# for domain pop, i.e, all e-mail in a domin into a single account

@yoursite2.com          yoursite2
```

Be sure to restart the sendmail daemon, after making the changes.

```
[root@mail /etc/mail]#  service sendmail restart
```

Access Control

Sendmail provides an access control feature through the /etc/mail/access file.

```
# /etc/mail/access

spam@cybermail.com      REJECT
aol.com                 REJECT
207.46.131.30           REJECT
postmaster@aol.com      OK
GNU/Linux.org.au        RELAY
192.0.21.               OK
```

OK Accept the mail message.

RELAY Accept messages from this host or user even if they are not destined for our host; that is, accept messages for relaying to other hosts from this host.

REJECT Reject the mail with a generic message.

It's is required that you allow RELAY from your own network. Otherwise, computers on the network using the server as their out going smtp server will not be able to send e-mails.

Running Sendmail as System Daemon

The script is located at `/etc/rc.d/init.d/sendmail` and is started automatically when the computer is started. You can also start it using other commands

```
[root@mail /etc/mail]#  /etc/init.d/sendmail start
[root@mail /etc/mail]#  service sendmail restart
```

Running Sendmail From xineted

It is a good idea (good from the security standpoint) to have sendmail run from xinetd.conf and not as a standalone daemon. For that we need to add it to /etc/xinetd.d directory and remove it from /etc/rc.d/init.d, add the sendmail queue processing to cron. Here is what you have to do:

1. When using xinetd create a file sendmail in /etc/xinetd.d/ similar to

```
default: on
service sendmail
{
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/bin/sendmail -bs
}
```

2. Edit `/etc/rc.d/init.d/sendmail` to have exit 0 somewhere in the very beginning (might not be the best way, be sure to document the changes you do to these files) so that this file does nothing instead of starting sendmail

3. By editing your (root's) crontab[6] (to edit do crontab -e) add a line like this

```
*/20 * * * * /usr/sbin/sendmail -q
```

That would process sendmail queue every 20 min (if it exists).

6 Cron, is a software used to schedule different other software on the computer.

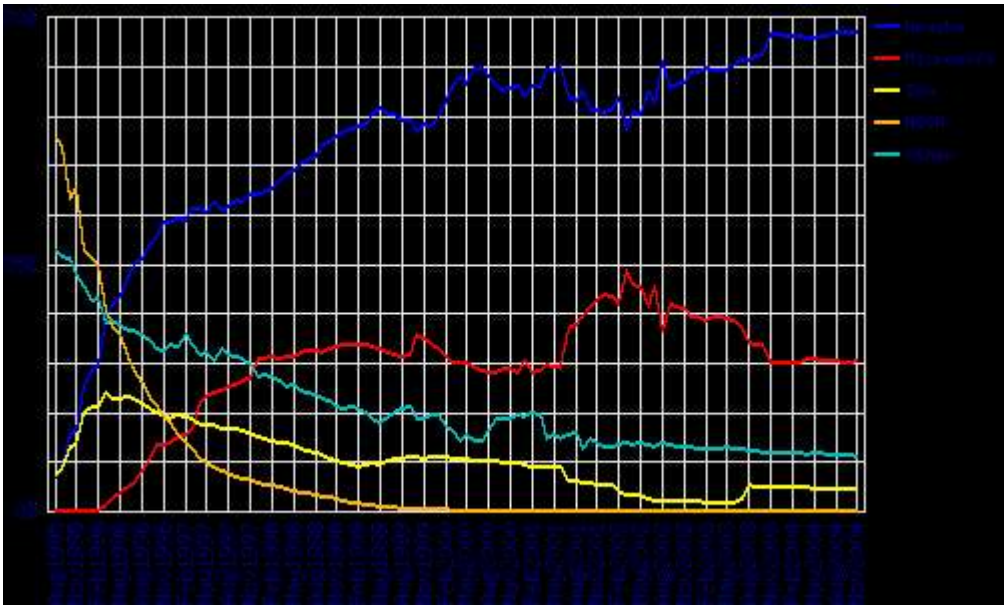
The Web Server - Apache

The dominance of free and open source software in the web server arena is well known. Apache web server has been the undisputed leader in any and all web server surveys conducted on the Internet. Apache web server also has many strengths. It was the leader in introducing name based virtual hosts, was the first truly modular web server that would work seamlessly with database servers. It also has a fully built Authentication, Authorization and Access Control functions as well as scripting support.

Apache also fully integrates with OpenSSL, which provides secure sockets layer[7], thereby enabling use of apache web server for e-commerce and secure transaction. And best of all, apache web server can be fully configured through a text based configuration file – httpd.conf.

Box---

November 2004 Web Server Survey



The latest survey stats are alwayw available at
http://news.netcraft.com/archives/web_server_survey.html

---box close ----------

7 Secure Sockets Layer (SSL) encrypts data that is travelling over the public network.

Configuring Apache

Configuring Apache is also failry easy, unless you want to run complex software on the web server. The configuration files are most often located by default in `'/etc/httpd/conf'.` Additional configuraiton files are located underneath `'/etc/httpd'.` It's also common for Apache to be installed in `'/usr/local/apache'.`

The main configuraiton file for apahce is the httpd.conf. Most often apache works out of the box.

The httpd.conf file

Directives are the settings that define how Apache should actually run where files are located on your server, how much of the machine's resources Apache may use, which content visitors are allowed to see, how many concurrent visitors the server can handle and other parameters.

Let's look at the main directives

`/etc/httpd/conf/httpd.conf` – the main configuration file

Server Identification

`ServerName`: construct self-referential URLs
`ServerAdmin`: email of server admin displayed on error messages

File Locations

`DocumentRoot` - the location where the static content of your web site lives
`ServerRoot` – for relative location of files that don't begin with a slash "/"
`ErrorLog` – to log server wide error messages
`PidFile` – contains process ID of the httpd process
`Alias` – to serve files outside the DocumentRoot
`ScriptAlias` – the location for CGI scripts, dynamic content generators
`DirectoryIndex` - the file specified in the DirectoryIndex directive to display by default
`Userdir` – to serve files from public_html dir in user's home dir as www.site.com/~user

Process Creation

`MaxClients` – no. of simultaneous connections allowed from clients
Server-Pool Regulation - Apache under Unix is multi-process

balances the overhead required to spawn child processes with system resources

MinSpareServers, MaxSpareServers, StartServers, MaxClients

Watch, tune, watch, tune

`User`,`Group` – set the privileges of the Apache child processes

Network Configuration

`BindAddress` - restrict the server to listening to a single IP address
`Listen` - specify multiple IP addresses and/or Ports
`KeepAlive` - an extension to HTTP, provides a persistent connection
`Port` – TCP port number the web server runs on, can be changed to an unused port

URL redirection:

To redirect requests to another URL

```
Redirect permanent /foo/ http://www.example.com/bar/
```

Virtual Hosts:

Virtual Hosts is the practice of maintaining more than one web server name on one physical machine. For example, the same physical machine can hosts both the http://www.apdip.net and http://www.iosn.net.

Parameters are specific to a virtual host, which override some of the main server configuration defaults. There can be two types of virtulhosts – IP Based and Name based.

In an IP based virtualhost the IP address of the connection is used to determine the correct virtualhost to serve. The approach requires a separate IP address for each virtualhost

In name based virtual hosts, the host name are sent as part of the HTTP headers, which means many different hosts can share the same IP address. But you'll need to map each host to the IP address in DNS. This eases the demand for scarce IP addresses. Name based virtual hosts cannot be used used with SSL secure servers and older software may not be compatible.

Virtualhost Directives:

`NameVirtualHost` - designate the IP address and port no to listen (optional)
`<VirtualHost>` - same argument as NameVirtualHost

`ServerName` - designate which host is served
`DocumentRoot` - where in the filesystem the content for that host lives
`ServerAlias` – make the host accessible by more than one name

Taking an example.

```
 NameVirtualHost *

<VirtualHost *>
ServerName www.domain.tld
DocumentRoot /www/domain
</VirtualHost>

<VirtualHost *>
ServerName www.otherdomain.tld
DocumentRoot /www/otherdomain
serverAlias otherdomain.tld *.otherdomain.tld
</VirtualHost>
```

If no matching virtual host is found, then the first listed virtual host that matches the IP address will be used. All other standard Apache directives can be used inside the virtualhost directive.

Access Control per Directory using .htaccess file

.htaccess file is a text file containing Apache directives

`AccessFileName .htaccess` …in httpd.conf

.htaccess file contents

```
AuthName "restricted stuff"
AuthType Basic
AuthUserFile /usr/local/etc/httpd/htusers
AuthGroupFile /usr/local/httpd/htgroup
require valid-user
require group staff
require user lahai gaurab

AuthName "restrict posting"
AuthType Basic
AuthUserFile /usr/local/etc/httpd/htusers
```

```
AuthGroupFile /usr/local/httpd/htgroup
<Limit POST>
require group admin
</Limit>
```

htpasswd – to manage users for access control

htpasswd -c /usr/local/etc/httpd/users martin

htpasswd /usr/local/etc/httpd/users ritesh

```
/usr/local/etc/httpd/htusers contents:
martin:WrU808BHQai36
jane:iABCQFQs40E8M
art:FAdHN3W753sSU
```

/usr/local/httpd/htgroup contents:

```
staff:martin jane
admin:lahai gaurab
```

References:

http://httpd.apache.org/docs/vhosts/index.html
http://httpd.apache.org/docs/vhosts/examples.html

Proxy and web caching with Squid:

Just like the mail server and the web server, free and open source software also set the standard in the area of proxy and cache servers. SQUID is synonyms with proxy services in the networking world. SQUID is a very modular high performance proxy and web caching server. The squid website is http://www.squid-cache.org . Squid proxy caches can be clustered to provide much better speed and access. Squid cache was also one of the first cache system to implement hierarchial cache system.

Some advantages of Squid
⇒ High-performance proxy caching server for web clients
⇒ A full-featured Web proxy cache
⇒ Designed to run on Unix systems
⇒ Free, open-source software
⇒ Handles all requests in a single, non-blocking, I/O-driven process
⇒ Keeps meta data and especially hot objects cached in RAM
⇒ Caches DNS lookups
⇒ Implements negative caching of failed requests
⇒ Supports SSL, extensive access controls, and full request logging
⇒ Using ICP, caches can be arranged in a hierarchy or mesh for additional bandwidth savings

Squid consists of:
⇒ Main server program *squid*
⇒ Domain Name System lookup program *dnsserver* for faster DNS lookups
⇒ Optional programs for rewriting requests and performing authentication
⇒ Some management and client tools

squid.conf – the main configuration file

⇒ Default configuration file denies all client requests
⇒ Configure to allow access only to trusted hosts and/or users
⇒ Carefully design your access control scheme
⇒ Check it from time to time to make sure that it works as you expect
⇒ People will abuse if proxy allows access from untrusted hosts or users
⇒ To make their browsing anonymous
⇒ Intentionally use your proxy for transactions that may be illegal
⇒ Web sites exist with a list of open-access HTTP proxies

The squid configuration example. To run the basic squid, the only thing configurable is the proxy port. The default squid proxy port is 3128. but you can always change it.

Network options:

```
http_port port        Hostname: port
```

Squid Access Control

Squid is must better known for it's complex access control system. You can not only allow and restrict access based on IP addresses but also based on domain name. The use of regular expression let you create complex rules for access through the proxy server. To use access control in squid, a sophisticated access control system, similar to the one used in routers is used. It's basically a two step process.

- Defining the access listed through use of acl command.
- Allowing or denying access based on the access list created earlier.

i. `acl` used for defining an Access List. '

'acl' literally stands for Access control list. The default ACLs are

```
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl SSL_ports port 443 563
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl CONNECT method CONNECT
```

ii. http_access – to control http access to clients

If there are no "access" lines present, the default is to allow the request

```
Default
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
```

```
http_access deny CONNECT !SSL_ports
http_access deny all
```

deny all line is very important

After all the http_access rules, if access isn't denied, it's ALLOWED !!

If none of the "http_access" lines cause a match, the default is the opposite of the last line in the list
A good idea to have a "deny all" or "allow all" entry at the end of your access lists

Examples:

Restrict access to work hours (9am - 5pm, Monday to Friday) from IP 192.168.2/24

```
acl ip_acl src 192.168.2/24
acl time_acl time M T W H F 9:00-17:00
http_access allow ip_acl time_acl
http_access deny all
```

Rules are read from top to bottom

```
acl xyz src 172.161.163.86
acl morning time 06:00-11:00
acl lunch time 14:00-14:30

http_access allow xyz morning
http_access deny xyz
http_access allow xyz lunch
```

Be careful with the order of allowing subnets

```
acl mynetwork src 10.0.0.0/255.0.0.0
acl servernet src 10.0.1.0/255.255.255.0

http_access deny servernet
http_access allow mynet
```

***always_direct** and **never_direct** tags*

```
# always go direct to local machines
always_direct allow my-iplist-1
always_direct allow my-iplist-2
# never go direct to other hosts
never_direct allow all
```

iii. cache_dir: dir to store cached data

```
cache_dir /usr/local/squid/cache/ 100 16 256
```

Can support more than one disk with multiple mount points
```
cache_dir /usr/local/squid/cache1/ 10000 16 256
cache_dir /usr/local/squid/cache2/ 20000 16 256
```

iv. *cache_mgr: email of the Cache Admin*

Appended to the end of error pages returned to users

```
cache_effective_user squid
cache_effective_group squid
Changes user and group ID's once it has bound to the incoming
network port
```

`ftp_user`: set the email address that is used for FTP proxy
Client: Connects to a cache and request a page, and prints out useful timing information

```
client http://squid.nlanr.net/
client -h cache.qualica.com -p 8080 http://www.ora.com/

client -h www.ora.com -p 80 /

client -g 0 -h www.ora.com -p 80 /
```

v. *Squid Logs:*

```
/usr/local/squid/logs/cache.log
/usr/local/squid/logs/access.log
```

Transparent Caching / Transparent Proxy

Transparently pick up the appropriate packets and cache requests and solves the biggest problem with caching: getting users to use the cache server. Four things needs to be considered.

- *Correct network layout* - all network traffic needs to pass through a *filter device*
- *Filtering* out the appropriate packets
- *Kernel Transparency*: redirecting port 80 connections to Squid
- *Squid settings*. Squid needs to know that it's supposed to act in transparent mode.

A detailed explanation of how to achieve transperant proxy is given at http://www.linuxdoc.org/HOWTO/mini/TransparentProxy.html

SECURITY FUNCTIONS USING FOSS

Security Paradigms, Security Policies and Infrastructure issues

Network and information security is one of th emost important things under anyones consideration when designing and deploying a network. Before we can get into the details of network security implementaiton with free and open source software, we need to look at the different security paradigms that have been in existance for some time now. The older paradigm supported the notion that perimeter security was the ultimate form of security for any network. That myth has now been dissolved, and the new paradigm is to secure every device itself in addition to the perimeter security imposed by firewalls and packet filters.

Seucurity as a technical solution is also no longer a valid assumption. Seucurity now involves not only stopping malafide packets from entering your network, but also stopping their origination from your own network, deploying tools to detect intrusion attempts, intelligent network sniffers and use of individualized security policy.

Security policy is now one of the most important security tool for any organization. Security policy helps in maintaining consistency in security implementation in any organization. Without the use of a security policy, it makes a administrator work difficult in identifying breaches in security. It also helps an organization to take actions against people who defy the policy. In absence of any such policy, the works is always done on an achoc manner, which can create more possibilities for security breaches.

There is no single bullet that can solve you security situation or problems. What is more useable are best practices, that document ways in which common security problems can be avoided. There are also security best practices for events when a security breach occurs. Let's look at common reasons for security breach and some of the best practices remedies.

Network security risks

Open architecture of TCP/IP (the protocol of the Internet) :

While TPC/IP is a highly efficient, cost-effective, and flexible communications protocol for local and global communications and is widely adopted on the global Internet and in the internal networks of large corporations, it was designed twenty years ago when the Internet consisted of a few hundred closely controlled hosts with limited security. The Internet now connects millions of computers, controlled by millions of individuals and organizations

in which the core network is administered by thousands of competing operators. This complex network spans the whole globe, connected by fibers, leased lines, dial-up modems, and mobile phones while very tolerant of random errors, TCP/IP is vulnerable to a number of malicious attacks.

The most common types of threats & attacks are

- Unauthorized access – insecure hosts, cracking
- Eavesdropping a transmission – access to the medium
- looking for passwords, credit card numbers, or business secrets
- Hijacking, or taking over a communication
- Inspect and modify any data being transmitted
- IP spoofing, or faking network addresses
- Impersonate to fool access control mechanisms
- Redirect connections to a fake server
- Denial of Service (DoS) attacks, when huge amount of useless data coming to the target makes it unable to function properly.
- Interruption of service due to system destruction or using up all available system resources for the service CPU, memory, bandwidth

Mistakes People Make that Lead to Security Breaches [8]

Technological holes account for a great number of the successful break-ins, but people do their share, as well:

The Five Worst Security Mistakes End Users Make

- Failing to install anti-virus, keep its signatures up to date, and perform full system scans regularly.
- Opening unsolicited e-mail attachments without verifying their source and checking their content first, or executing games or screen savers or other programs from untrusted sources.
- Failing to install security patches-especially for Microsoft Office, Microsoft Internet Explorer, Outlook, Windows OS.
- Not making and testing backups.
- Using a modem while connected through a local area network.

The Seven Worst Security Mistakes Senior Executives Make

- Assigning untrained people to maintain security and providing neither the training nor the time to make it possible to learn and do the job.

8 Most of the points under this topics traces it's origin back to the SANS Institute . SANS Institute is located at www.sans.org. Some text have been modified to suit changing times.

- Failing to understand the relationship of information security to the business problem-they understand physical security but do not see the consequences of poor information security.
- Failing to deal with the operational aspects of security: making a few fixes and then not allowing the follow through necessary to ensure the problems stay fixed
- Relying primarily on a firewall
- Failing to realize how much money their information and organizational reputations are worth
- Authorizing reactive, short-term fixes so problems re-emerge rapidly.
- Pretending the problem will go away if they ignore it.

The Ten Worst Security Mistakes IT People Make

- Connecting systems to the Internet before hardening them.
- Connecting test systems to the Internet with default accounts/passwords
- Failing to update systems when security holes are found
- Using telnet and other unencrypted protocols for managing systems, routers, firewalls, and PKI.
- Giving users passwords over the phone or changing user passwords in response to telephone or personal requests when the requester is not authenticated.
- Failing to maintain and test backups.
- Running unnecessary services : ftpd, telnetd, finger, rpc, mail, rservices
- Implementing firewalls with rules that don't stop malicious or dangerous traffic - incoming and outgoing.
- Failing to implement or update virus detection software
- Failing to educate users on what to look for and what to do when they see a potential security problem.

Security Best Practices

- Some set a goal to fully and completely secure a system
- But this is impractical and usually an impossible goal to make a system full-proof
- A realistic goal is to set up a regular routine where you identify/correct as many vulnerabilities as practical

Benefits of implementing best security practices:

To make it so difficult for an attacker to gain access that he gives up before he gets in

- Many sites have minimal or no security - attackers usually gain access relatively quickly and with a low level of expertise

- With some security, chances of an attacker exploiting its systems are decreased significantly - the intruder will probably move on to a more vulnerable site
- “The idea is not that you should protect a system to the point it cannot be compromised, but to secure it at least enough so that most intruders will not be able to break in, and will choose to direct their efforts elsewhere” e.g. it is just like putting iron bars and locks on our windows and doors - we do it not to "keep the robbers out", but to persuade them to turn their attention to our neighbors
- Rather than directing our efforts at protecting against the thousands of specific threats (this exploit, that Trojan virus, these mis-configurations), focus our energies into tasks that provide the most comprehensive protection against the majority of threats
- Best Security Practices are very dynamic, constantly changing and evolving
- Administrators should include their own Best Security Practices and modify those mentioned here to best fit their environment

Points to ponder:

- Take into consideration your needs risks, resources, and then apply to your systems to most effectively protect them from intrusion or disruption
- Information systems are unavoidably complex and fluid, so the most effective way to apply security is in layers
- You should place security measures at different points in your network, allowing each to do what it does best
- From an attacker's perspective, you have constructed a series of obstacles of varying difficulty between the attacker and your systems
- Secure each component in your system (firewalls, routers, servers, hosts, and appliances) so that even if an attacker works their way through your obstacle-course, at the end they will find systems that are resistant to attack

Backup

- Maintain full and reliable backups of all data, log files
- Archive all software (purchased or freeware), upgrades, and patches off-line so that it can be reloaded when necessary
- Backup configurations, such as the Windows registry and text/binary configuration files, used by the operating systems or applications
- Consider the media, retention requirements, storage, rotation, methods (incremental, differential, full) and the scheduling
- Keep copy of a full backup in a secure off-site location for disaster recovery

----------box ----

⇒ FOSS Vs. Proprietary Software

A big debate when talking about security is the relative benefits of free and open source software in comparasion to closed source or proprietary software. Both have their own advantages, but the structure of free and open sources software means that security is much more of an necessity rather than an after thought.

If a free and open source product is not secure, there'll be hundreds if not thousands of people reviewing the source code of the application in question immediately. If the product is not designed in a secure way, newer implementation which are more secure are common to come by. The good example is the TCP/IP protocol stack on GNU/Linux itself. From the early days to the present day, the networking protocol stack in GNU/Linux has seen tremendous change, not only for security reasons but also for enhancement and inclusion of new features.

When talking about free and open source vs. closed source software from a security perspective, a common thing that comes into discussion is full disclousre vs. non-disclousre. In case of full-disclousre, any and all security vulnerabilities that have been identified are publicly disclosed, after a fix has been found. If the situation is critical, vulnerability disclousre is done, even if a fix has not been released. The effect is an immediate attention by thousands of people who use the software. This results in quick fixes and patches made available.

Whereas, in case of closed source software, since others cannot look at the software, the users are at the vendors mercy on when the security fix will be available.

Interesting to note that, in case of closed software, the same people who wrote the unsecure software are trusted to fix their problems, where as in open source others are able to have a different perspective when reviewing at the software.

close box --------

Securing your network and hosts properly

Firewall

Many people might think that a firewall is a single device on your network configured to protect your internal network from the external world. In reality, A firewall is a system (or a group of systems) that enforces an access control policy between two networks. It disallows unauthorized and/or malicious traffic from traveling on your network – in both directions. Users have to realize that Firewalls can't protect you from attacks that don't go through it. Specially, if there's another entry point to your network not protected by a firewall, then your network isn't secured. Also, contrary to the common mis-conception, firewalls do not verify the content of the traffic through it.

Types of firewalls:

Packet filtering firewalls

- examines the source and destination address of the data packet and either allows or denies the packet from traveling the network
- blocks access through the firewall to any packets, which try to access ports which have been declared "off-limits"

Application layer firewalls

- Also known proxy firewalls, application gateway
- attempts to hide the configuration of the network behind the firewall by acting on behalf of that network/servers
- All requests for access are translated at the firewall so that all packets are sent to and from the firewall, rather than from the hosts behind the firewall

Stateful inspection firewalls

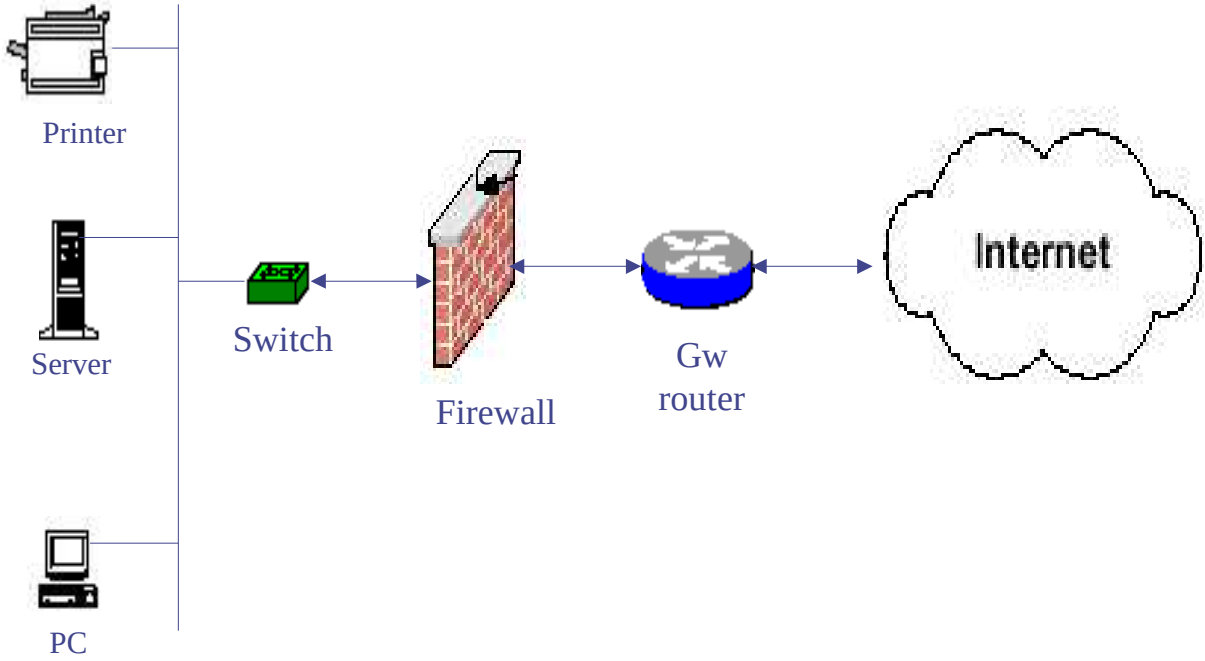
- Examines the state and the context of the packets
- Remembers what outgoing requests have been sent and only allow responses to those requests back through the firewall
- Attempts to access the internal network that have not been requested by the internal network will be denied

Firewall Best Practices

Regardless of which type of firewall, someone has to configure the firewall to make it work properly, the rules for access must be defined and entered into the firewall for enforcement. A security manager is usually responsible for the firewall configuration. But again, if the policy is not made properly, then there is not much the security manager can do. Some rule of thumbs are .

- Explicitly deny all traffic except for what you want
- The default policy should be that if the firewall doesn't know what to do with the packet, deny/drop it
- Don't rely only on your firewall for the protection of your network, remember that it's only a device, and devices do fail
- Make sure you implement what's called "defense in depth." - multiple layers of network protection
- Make sure all of the network traffic passes through the firewall
- If the firewall becomes disabled, then disable all communication
- If there's another way in to the network (like a modem pool or a maintenance network connection), then this connection could be used to enter the network completely bypassing the firewall protection
- Disable or uninstall any unnecessary services and software on the firewall
- Limit the number of applications that run on the firewall
- Consider running antivirus, content filtering, Virtual Private Network, etc. on different hardware
- Do not rely on packet filtering alone. Use stateful inspection and application proxies if possible
- Ensure that you're filtering packets for illegal/incorrect addresses – to avoid IP spoofing
- Ensure that physical access to the firewall is controlled
- Use firewalls internally to segment networks between different departments and permit access control based upon business needs
- Remember that firewalls won't prevent attacks that originate from inside your network

A typical firewall setup



Free and Open Source Firewall software

Iptables www.iptables.org

IPTABLES, by far, is the most popular firewall tool available on GNU/Linux. It works in combination with netfilter, which is included as an integral part of the Linux Kernel. It has been in the kernel since release 2.4.x as the firewalling subsystem. IP Tables is a much improved version of previoulsy available IP Chains and IP fwadm software.

IP Tables provides functionality of packet filtering (stateless or stateful) as well as NAT (Network Address Translation) and IP Masquerading . It also can perform packet mangling (manipulation).

IPTables itself consistes of multiple sets of things. They are

- userspace command for runtime configuration of the firewalling subsystem
- Kernel modules and the user interface application. Kernel modules are components that handle various tasks and are compiled as loadable modules in the GNU/Linux kernel
- Generic table structure for the definition of rulesets
- Consists out of a number of classifiers (matches) and one connected action (target).

Iptables – Tables , Chains and rules

IPTables normally has multiple tables defined in the kernel. The default is the 'filter' table. The GNU/Linux Kernel starts with three lists of rules in the `filter' table. A chain is a checklist of rules. There are 3 built-in chains which are INPUT, OUTPUT and FORWARD chains. The Linux Kernel examines each chain to decide the fate of the packet. In turn, each rule says `if the packet header looks like this, then here's what to do with the packet'

Using iptables:

iptables syntax

```
[root@mail /]#  iptables  [options] [table] rules TARGET
```

Main iptables Options

1. Create a new chain (-N).
2. Delete an empty chain (-X).
3. Change the policy for a built-in chain. (-P).
4. List the rules in a chain (-L).
5. Flush the rules out of a chain (-F).
6. Zero the packet and byte counters on all rules in a chain (-Z).

You can do man iptables or iptables –h for a list of command options

To manipulate rules in a chain:

1. Append a new rule to a chain (-A).
2. Insert a new rule at some position in a chain (-I).
3. Replace a rule at some position in a chain (-R).
4. Delete a rule at some position in a chain (-D).
5. Delete the first rule that matches in a chain (-D).

The default policies:

The default policy is to accept all packets to, from, through it.

```
INPUT ACCEPT any any
FORWARD ACCEPT any any
OUTPUT ACCEPT any any

To set the default policies to DROP all packets:
# iptables –P INPUT DROP
# iptables –P FORWARD DROP
# iptables –P OUTPUT DROP
```

Targets – what to do if a packet matches a rule

ACCEPT
DROP
REJECT
LOG

Example rules:

```
# iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP
# iptables -D INPUT 1
# iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP

# iptables -A INPUT -s 0/0 -j DROP

# iptables –A INPUT –p tcp –dport 25 –j ACCEPT
# iptables –A FORWARD –p tcp dport 25 –j ACCEPT
# iptables –A FORWARD –I eth0 –o eth1 –p tcp –dport 25 –j ACCEPT
# iptables –A FORWARD –s 192.0.2.0/24 –d 202.52.255.1 –p tcp –dport 25 –j ACCEPT
# iptables –A FORWARD –s 192.0.2.0/24 –d 202.52.255.5 –p udp –dport 53 –j ACCEPT
# iptables –A FORWARD –d 202.52.255.5 –p tcp –j LOG –log-prefix “TCP log ”

To get help:
# iptables -p tcp -h
# iptables -m state -h
# iptables –j ACCEPT -h
```

Connection Tracking:

Stateful connection tracking of traversing packets with –m state –state options

NEW – packets that create a NEW connection
ESTABLISHED – packets belonging to an existing connection – reply packets
RELATED – packets related to an existing connection – ICMP error messages/ FTP data conns
INVALID – packets not corresponding to any existing conn

```
# iptables –A FORWARD –d 192.0.2.0/24 –p tcp –m –state –state ESTABLISHED –j REJECT
# iptables –A FORWARD –m –state –state INVALID –j DROP
```

Sample Iptables setup:

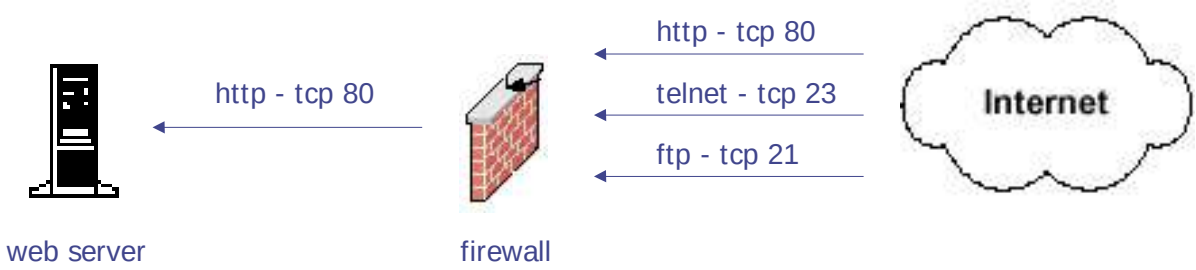
```
-------------------------------------------------------------
iptables –F  flush all the chains
iptables –X  delete all the empty chains

# Changing Default policy
iptables –P INPUT DROP
iptables –P FORWARD DROP
iptables –P OUTPUT DROP

# INPUT chain
```

```
iptables –A INPUT –p icmp –j ACCEPT
iptables –A INPUT –I eth1 –p tcp –dport 22 –j ACCEPT
iptables –A INPUT –I eth1 –m state –state ESTABLISHED,RELATED –j ACCEPT

# FORWARD chain
iptables –A FORWARD –p icmp –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.1 –p tcp –dport 25 –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.2 –p tcp –dport 110 –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.3 –p tcp –dport  80 –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.4 –p tcp –dport 53 –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.4 –p udp –dport 53 –j ACCEPT
iptables –A FORWARD –I eth0 –o eth1 –d 192.0.2.0/24 –m state –state
ESTABLISHED,RELATED –j ACCEPT

iptables –A FORWARD –I eth1 –o eth0 –s 192.0.2.1 –p tcp –dport 25 –j ACCEPT
iptables –A FORWARD –I eth1 –o eth0 –s 192.0.2.4 –p tcp –dport 53 –j ACCEPT
iptables –A FORWARD –I eth1 –o eth0 –s 192.0.2.4 –p udp –dport 53 –j ACCEPT
iptables –A FORWARD –I eth1 –o eth0 –s 192.0.2.0/24 –p tcp –dport 80 –j ACCEPT
iptables –A FORWARD –I eth1 –o eth0 –s 192.0.2.0/24 –m state –state
ESTABLISHED,RELATED –j ACCEPT

# OUTPUT chain
iptables –A OUTPUT –p icmp –j ACCEPT
iptables –A OUTPUT –o eth1 –p tcp –dport 22 –j ACCEPT
iptables –A INPUT –o eth1 –m state –state ESTABLISHED,RELATED –j ACCEPT

# Enabling NAT
iptables -t nat -A POSTROUTING -s 192.0.2.0/24 -j MASQUERADE
```

References:

http://www.unixreview.com/documents/s=1237/urm0103c/0103c.htm
http://www.knowplace.org/netfilter/index.html
http://www.GNU/Linuxguruz.org/iptables/
http://www.boingworld.com/workshops/GNU/Linux/iptables-tutorial/iptables-tutorial/iptables-tutorial.html

Packet flow diagram: How Packets Traverse The Filters?



- Allow only http - tcp 80
- Drop ip any

Server Security, Host / Network Scanning and Intrusion Detection System

It is often advised that some kind of detection and prevention measure be run in addition to the firewall. This is commonly referred to as Network Scanning and Intrusion Detection System (IDS). There are also proactive measures which can make your system and servers secure. Let's look at some of those.

Server security do's and don't

- Run the server on a hardened and routinely updated operating system
- Keep current on software / application updates
- make sure you test these updates in a controlled, non-production environment whenever possible
- one server patch may undo a correction a previous patch applied so scan the server after the patching up to make sure
- hackers usually attack servers with security bugs that are well known and around for a long time
- Disable file sharing an all critical machines – as it makes them vulnerable to bother information theft and certain types of quick-moving viruses
- Improper sharing configuration can expose critical systems files or give full file system access to any hostile party

Host / Network Scanning

Network scanning refers to the activity of scanning your own networks for vulnerabilities. There are tools available for this purpose. Incidently, these are most often the same tools used by crackers and external parties to gauge your network security. Thus, using them yourself can led you to quite a few of those security holes, which would have gone unnoticed otherwise. There are some tips in what to scan for.

Scanning Systems tips

- Scans will help determine that only the required ports are open
- Services running on the open ports are not vulnerable to known security bugs/holes
- Will help you determine if your systems have been compromised – if new open ports are found
- Perform full port scans using a tool like nmap/ndiff, nessus, fscan on a regular basis
- Port scans should cover all ports (1-65,535), both UDP and TCP, on all systems: both clients and servers devices such as routers, switches, printers and anything else connected (physically through wire or wireless) to your network

Host scanning software : Nmap : www.insecure.org/nmap

Nmap is a Network MAPper. It's a powerful utility for network exploration or security auditing, which can rapidly scan large networks or single host to determine what hosts are available on the network, what services (ports) they are offering, what operating system (and OS version) they are running, and what type of packet filters/firewalls are in use. It runs on most types of computers, and both console and graphical versions are available. It's also GNU GPL oftware, available with full source code.

Nmap Usage examples

1. This option scans all reserved TCP ports on the machine target.example.com . The -v means turn on verbose mode.

```
nmap -v target.example.com
```

2. Launches a stealth SYN scan against each machine that is up out of the 255 machines on class ´C´ where target.example.com resides. It also tries to determine what operating system is running on each host that is up and running. This requires root privileges because of the SYN scan and the OS detection.

```
nmap -sS -O target.example.com/24
```

3. Sends an Xmas tree scan to the first half of each of the 255 possible 8 bit subnets in the 198.116 class ´B´ address space. We are testing whether the systems run sshd, DNS, pop3d, mapd, or port 4564. Note that Xmas scan doesn't work on Microsoft boxes due to their deficient TCP stack. Same goes with CISCO, IRIX, HP/UX, and BSDI boxes.

```
nmap -sX -p 22,53,110,143,4564 198.116.*.1-127
```

4. Rather than focus on a specific IP range, it is sometimes interesting to slice up the entire Internet and scan a small sample from each slice. This command finds all web servers on machines with IP addresses ending in .2.3, .2.4, or .2.5

```
nmap -v --randomize_hosts -p 80 ´*.*.2.3-5´
```

5. Launch a stealth scan with OS detection on all privileged ports against 255 hosts in the network, output the results into the file /root/nmap.scan

```
nmap –sS –O 192.0.2.0/24 –oN /root/nmap.scan
```

6. Launch a stealth scan with OS detection on specified ports against 255 hosts in the network, in verbose mode.

- `nmap –sS –O –v 192.0.2.0/24 –p '1-1024,1080,3128'`

Network Monitoring, Scanning and reporting Tool : Nessus

The mantra for doing Network Monitoring and scanning can be "Prevention is ideal, but detection is a must" We must realize that “No prevention technique is full-proof” and new vulnerabilities are discovered every week that you may not be aware of. Thus, constant vigilance is required to detect new unknown attacks. Once you are attacked, without logs, you have little chance of finding what the attackers did. It is also common sense that you can not detect an attack if you do not know what is occurring on your network and for that purpose Logs provide the details of what is occurring, what systems are being attacked, and what systems have been compromised If any log entries that don't look right, and investigate them immediately. But since the everyday thousands of lines of logs are produced, using appropriate tools to diagnose these is necessary. It's also necessary that these logs and scans are compared against the vulnerability databases, so that the security gap in the network can be known immediately.

What is Nessus ?

Nessus is a security scanner that can remotely audit a given network or servers. It can determine whether bad guys (aka 'crackers') may break into it, or misuse your network or servers in some way

Unlike others, Nessus does not take anything for granted and will not consider that a services only run on fixed ports. i.e, if you run your web server on port 1234, Nessus will detect it and test its security. It will not make its security tests by the version number, but will really attempt to exploit the vulnerability. It is very fast, reliable and has a modular architecture that allows you to fit it to your needs. Nessus also has front ends for both Windows and macintosh.

Nessus Installation:

Nessus is made up of two parts : a client and a server. You need a Unix-like system to use the server (GNU/Linux is recommended). In this test, I used the standard client nessus, mainly because the same auther wrote it and because it is the only one that supports the cipher layer.
The Nessus Security Scanner relies on the following items:

GTK - The Gimp Toolkit, version 1.2

GTK is a set of Widgets (like Motif) which are used by many open-sourced programs such as The Gimp. GTK is used by the POSIX client nessus.

Download it at : ftp://ftp.gimp.org/pub/gtk/v1.2.

Note : If your system comes with GTK, make sure that you have the gtk-config program installed. If you do not, install the gtk-devel package that should come on your distribution CDROM.
Note #2: If you do not want to install GTK and/or if your system lacks X11, then you can compile a command-line client by doing

```
./configure –disable-gtk
```

in nessus-core

Nmap, which we already covered earlier is an excellent portscanner and which is available at http://www.insecure.org/nmap/.

OpenSSL (optional but heavily recommended). OpenSSL is used for the client - server communication as well as in the testing of SSL-enabled services. Get it at http://www.openssl.org.

Nessus also comes as a standalone package that auto-installs itself. To use it, download the script nessus-installer.sh http://www.nessus.org/download.html

Using Nessus:

1. Create a nessusd account

The nessusd server has its own users database, each user having a a set of restrictions. This allows you to share a single nessusd server for a whole network and different administrators who will only test their part of the network.

The utility nessus-adduser takes care of the creation of a new account :

```
# nessus-adduser
Using /var/tmp as a temporary file holder

Add a new nessusd user
----------------------
Login : sunil
Authentication (pass/cert) [pass] :
Login password : nessus

User rules
----------
```

```
nessusd has a rules system which allows you to restrict the hosts
that sunil has the right to test. For instance, you may want him to
be able to scan his own host only.

Please see the nessus-adduser(8) man page for the rules syntax. Enter
the rules for this user, and hit ctrl-D once you are done : (the user
can have an empty rules set)

Login              : sunil
Password          : nessus
DN                :
Rules             :

Is that ok ? (y/n) [y]
user added.
```

2. Configure your nessus daemon

In the file /usr/local/etc/nessus/nessusd.conf, I can set several options for nessusd. Typically this is where you can define that you want nessusd to use your favorite language (french for me). Anyway, I kept the standard configuration file for this demonstration.

3. Start nessusd

Once all of this is done, I can safely start nessusd as root :

```
nessusd -D
```

4. Fire up nessus in X windows graphical environment

Click on Login, since this setup is correct. Since this is the first time connecting to this server, it will ask the password. The next time you connect to it, the public key will be enough.

Once connected, the Log in button changes to Log out, and a Connected label appears at its left.

5. The security checks configuration

Let all the security check to be performed, except the Denial of Service attacks, because you do not want hosts to crash.

Clicking on a plugin name will pop up a window explaining what the plugin does.

6. The plugins preferences

Some security checks will require extra arguments. For instance, the pop2 overflow security test needs a valid pop account. The plugin which tests whether a FTP directory is writeable or not asks if it should just trust the permissions or really attempt to store a file. And so on... This screen shot shows the configuration of Nmap.

7. The scan options

Here you choose which port scanner you want to use. Prefer to use the Nmap tcp connect scanner, since it's the fastest.

8. Define the targets

Uncheck the 'Perform a DNS transfer zone' option, since it would make DNS transfer on fr.nessus.org and nessus.org, and it would be useless, since it would not gain any new hosts.

Use the following options to define the targets:

```
192.0.2.1             A single IP address.
192.0.2.1-7           A range of IP addresses.
192.0.2.1-192.0.3.50  Another range of IP addresses.
192.0.2.1/29          Again a range of IP addresses in CIDR[9]
prof.fr.nessus.org    A hostname in FQDN[10] notation.

prof                  A hostname (as long as it is resolvable
on the server).

prof, 192.0.2.1/29, ...    Any combination of the above
mentioned forms separated by a comma.
```

9. The rules section

The rules allow a user to restrict his test. For instance, if you want to test 192.0.2.0/29, except 192.0.2.2. The ruleset entered allows you to do that. Once all of this is done, start the scan…

10. The Nessus scan report

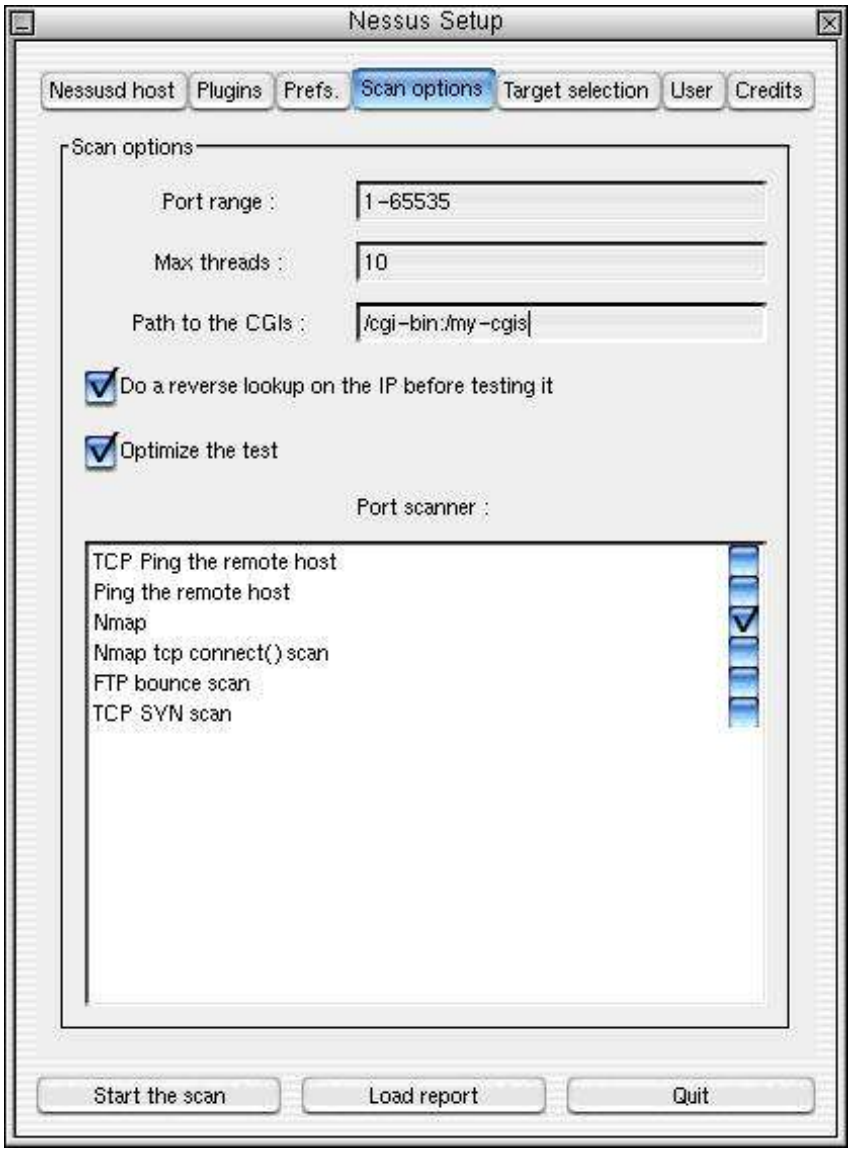
Now that the scan is over, the report window just pops up

9 Classles Inter Domain Routing
10 Fully Qualified Domain Name.

11. Resolving the security issues

Go through the scan report and try to resolve reported security holes on the respective servers by following the suggestions given in the report OR by referring to the vendor's site.

12. Nessus screen shot



Intrusion Detection System (IDS)

Intrusion Detection is the art of detecting inappropriate, incorrect, or anomalous activity. IDS will Inspects/sniffs all network traffic passing thru it for any abnormal content. It has built in signature-base and anomaly detection, providing the capability to look for set "patterns" in packets. It can also string search signature (i.e. look for confidential), log and TCP reset features. IDS provides worthwhile information about malicious network traffic and help identify the source of the incoming probes, scans or attacks. It is similar to a security "camera" or a "burglar alarm“ as it alerts security personnel that someone is picking the "lock“ i.e, alerts security personnel that a Network Invasion maybe in progress

SNORT

The most common open source IDS in use is SNORT (www.snort.org). It does the following.

- Monitors network traffic for predefined suspicious activity or patterns
- Alert system administrators when potential hostile traffic is detected
- A cross-platform, lightweight network intrusion detection tool
- Can be deployed to monitor small TCP/IP networks
- Detect a wide variety of suspicious network traffic as well as outright attacks
- Deployed rapidly to fill potential holes in a network's security coverage

The 3 modes of Snort:

1. Sniffer mode:

Display the packect headers or data

```
Snort –v
Snort –vd
```

2. Packet logger mode:

Record packets to the disk

```
snort –dev –l ./log
snort -l ./log –b                log all packets in binary mode
snort –dvr ./log/snort.log       to display packets in ascii
                                      mode from the binary log
```

3. Network Intrusion Detection Mode:

Doesn't record every single packet sent down the wire

```
snort -d -h 192.0.2.0/24 -l ./log -c snort.conf
```

This will configure Snort to run in it's most basic NIDS form, logging packets that the rules tell it to in plain ASCII to a hierarchical directory structure (just like packet logger mode).

To send NIDS alerts to syslog use -s:

```
snort -s -h 192.0.2.0/24 –l ./log -c snort.conf
```

High Performance Configuration:

```
snort -b -A fast -c snort.conf
```

This log packets in tcpdump format and produce minimal alerts

To read this file back and break out the data in the familiar Snort format

```
snort -d -c snort.conf -l ./log -h 192.0.2.0/24 -r snort.log
```

To run Snort in daemon mode (provide full paths):

```
usr/local/bin/snort -d -h 192.0.2.0/24 -l /var/log/snortlogs
-c /usr/local/etc/snort.conf -s -D
```

To post packet logs to public mailing lists try –O obfuscates" your the IP addresses

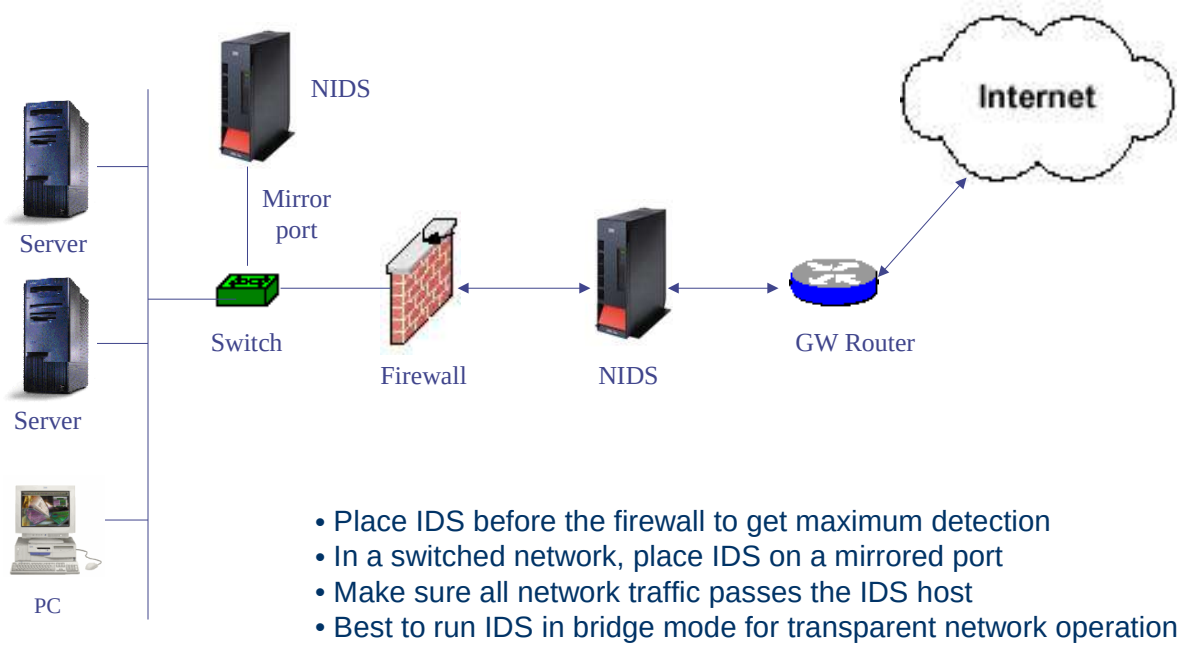
```
snort -d -v -r snort.log -O -h 192.0.2.0/24
```

To get help options:

```
snort –h
```

References:

http://www.snort.org/docs/writing_rules/
http://www.snort.org/docs/lisapaper.txt



Secure Shell **OpenSSH:** http://www.openssh.org

One of the biggest advantages of Unix and GNU/Linux is the ability to work remotely on the server systems. This feature is an inherent part of any Unix like operating system. In earlier days, most people used *Telnet* to log remotely into servers, but now telnet has been made obsolete by **ssh** or secure shell.

The main difference between telnet and ssh is the secure communication protocol of the later. In telenet all the communication was done in plain text, where as in ssh all communication is encrypted.

Openssh is both a ssh server and client. It is installed by default on most GNU/Linux distributions. Since it even ecncrypts the password sent to set up the session, it effectively eliminate eavesdropping, connection hijacking, and other network-level attacks. It includes the following different server sub-systems.

Sshd – ssh daemon run on the server machine
Sftp-server – sftp server sub-system
Ssh-agent – authentication agent that holds the keys
Ssh-add – used to register new keys with the Agent
Ssh-keygen – used to create public authentication keys

OpenSSH Config files:

Since OpenSSH is both a client and a server, there are two configuration files for each purpose.

/etc/ssh/ssh_config – ssh client systemwide configuration file, provides defaults for users
/etc/ssh/sshd_config - sshd server system-wide configuration file

OpenSSH server configuration:

Installation and Basic Operation

Install the openssh-server and openssh rpm package included in your Linux Distribution. Most Linux distribution will have option to install OpenSSH. Please note that the openssh-server package depends on the openssh package. OpenSSH packages also require the OpenSSL package (openssl) which installs several important cryptographic libraries that help OpenSSH provide encrypted communications

OpenSSH daemon, sshd, uses the configuration file `'/etc/ssh/sshd_config'`

The default config file is sufficient. For customization, refer to sshd man page for config options.

Editing /viewing the sshd_config file

```
[root@mail /] # vi /etc/ssh/sshd_config
```

#Port 22 - Specifies the port number that sshd listens on

#Protocol 2,1 - Specifies the protocol versions sshd supports

#ListenAddress 0.0.0.0 - Specifies the local addresses sshd should listen on

#HostKey /etc/ssh/ssh_host_rsa_key - Specifies a file containing a private host key used by SSH

#SyslogFacility AUTH - Gives the facility code that is used when sshd logs messages

#LogLevel INFO - Gives the verbosity level that is used when sshd logs messages

#LoginGraceTime 600 – time limit for a user to log in

#PermitRootLogin yes - Specifies whether root can login using ssh

#StrictModes yes - Specifies whether sshd should check file modes and ownership of the user's files and home directory before accepting login

#PubkeyAuthentication yes - Specifies whether public key authentication is allowed

#PasswordAuthentication yes - Specifies whether password authentication is allowed

#PermitEmptyPasswords no - When password authentication is allowed, it specifies whether the server allows login to accounts with empty password strings

#MaxStartups 10 - Specifies the maximum number of concurrent unauthenticated connections to the sshd daemon

#KeepAlive yes - Specifies whether the system should send TCP keepalive messages to the other side

#VerifyReverseMapping no - Specifies whether sshd should try to verify the remote host name

Subsystem sftp /usr/libexec/openssh/sftp-server

Managing the sshd service

To start the service: `#/sbin/service sshd start`
To stop the service:`#/sbin/service sshd stop`
To restart the service: `#/sbin/service sshd restart`

To automatically run the service: `# chkconfig sshd on`

OpenSSH Client configuration:

To connect to an OpenSSH server from a client machine, you must have the openssh-clients and openssh packages installed on the client machine.

System wide ssh client configuration is stored in **`/etc/ssh_config`** and is used as the default for all system wide users

The configuration values can be changed in per-user configuration files or on the command line. The default home directory is **`~/.ssh`**

Using the ssh with password authentication:

1. Make sure you have the following enabled in **`/etc/sshd_config`** file on the ssh server

```
PasswordAuthentication     yes
```

2. To log in to a host named server.com, type the following:

```
$ ssh gaurab@server.com
```

The first time you ssh to a remote machine, you will see a message similar to the following:

```
The authenticity of host 'server.com (192.0.2.22)' can't be
established.
RSA key fingerprint is
0f:41:6c:52:31:59:43:0f:dd:49:5f:3d:47:9d:b5:9e.
Are you sure you want to continue connecting (yes/no)? yes
```

Type **yes** to continue.
This will add the server to your list of known hosts as seen in the following message:

```
Warning: Permanently added 'server.com,192.0.2.22' (RSA) to
the list of knownhosts.
gaurab@server.com's password:
Last login: Sat Oct 16 19:34:13 2004 from gw-ktm.lahai.com
[gaurab@server.com gaurab]$
```

3. **`known_hosts`** file in **`.ssh`** folder contains the remote servers' host keys:

```
$ more known_hosts
server.com,192.0.2.22 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA5IwKqHiqSeXjrX3dCpS1gXZo9GqJ0hkk+c
lf+WmABYbEH6IGMyy2CeARtaR6QLpqB1SaGEFsn84dqA6kWLYfn4FuDVDc8KTy
ABLVEMOm6NnLZkHPKr3Cb0RgivDYSYHlwgWuDi7XBvmoC44WA2EbM7eBy5h1kH
rXZ5yPXq3rxI0=
```

Using the ssh command with public key authentication:

1. Make sure you have the following enabled in `/etc/sshd_config` file on the ssh server.

PubkeyAuthentication yes

2. Key pair generation:

ssh-keygen - authentication key generation, management and conversion

To generate a RSA key pair to work with version 2 of the protocol:

```
$ ssh-keygen -t rsa
```

Generating public/private rsa key pair.
Enter file in which to save the key (**/home/gaurab/.ssh/id_rsa**):

Enter a passphrase different from your account password and confirm it by entering it again

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gaurab/.ssh/id_rsa.
Your public key has been saved in /home/gaurab/.ssh/id_rsa.pub.
The key fingerprint is:
e5:a1:ac:ce:8d:16:3a:b9:3a:e4:e9:06:9c:90:b6:da gaurab@lahai.com
```

The public key is written to **~/.ssh/id_rsa.pub.**
The private key is written to **~/.ssh/id_rsa.**
Never distribute your private key to anyone and change the permissions of your .ssh directory using the command **chmod 755 ~/.ssh**

```
$ ll .ssh/
total 8
-rw-------    1 gaurab   gaurab        951 Oct 16 19:37 id_rsa
-rw-r--r--    1 gaurab   gaurab        236 Oct 16 19:37 id_rsa.pub
```

Copy the contents of **~/.ssh/id_rsa.pub** to **~/.ssh/authorized_keys** on the machine to which you want to connect.

If the file **~/.ssh/authorized_keys** does not exist, you can copy the file **~/.ssh/id_rsa.pub** to the file **~/.ssh/authorized_keys** on the remote SSH server.

Securely copy your public key file to the remote ssh server:

```
$ scp ~/.ssh/id_rsa.pub gaurab@server.com:
```

Logon to the remote ssh server using password authentication, one more time:

```
$ ssh gaurab@server.com
```

Copy the content of the public key file to authorized_keys file on the remote host:

```
$ cat id_rsa.pub >> ~/.ssh/authorized_keys
```

Now, logon to the remote ssh server via public key authentication:

```
$ ssh gaurab@server.com

Enter passphrase for key '/home/gaurab/.ssh/id_rsa':
Last login: Sat Oct 16 19:40:11 2004 from myhost.com
[gaurab@server.com gaurab]$
```

To run a command on the remote host with ssh and exit:

```
$ ssh ns.lahai.com ls /usr/share/doc
```

Using the scp command

The scp command is used to transfer files between machines over a secure, encrypted connection

To transfer the local file shadowman to /home/username/shadowman on penguin.example.net:

```
$ scp shadowman username@penguin.example.net:/home/username
```

To transfer a remote file to the local system:

```
$ scp username@tohostname:/remotefile /newlocalfile
```

To transfer the contents of the directory /downloads to an existing directory called uploads on the remote machine penguin.example.net:

```
$ scp /downloads/* username@penguin.example.net:/uploads/
```

Using the sftp command

The sftp command is used to open a secure, interactive FTP session via an encrypted channel.

```
$ sftp username@hostname.com
```

Once authenticated, you can use a set of commands similar to using FTP

NETWORK PLANNING WITH FOSS

A network in any computing environment is always a long term investment. It is imperitive that proper planning be done before going all out on deploying any network. A few pointers on network design and development.

Network Planning Basics

Capacity
: Plan for at least 2-3 years

Infrastructure
: Build for at least 1 year

Business Models
: Look after the next quarter

As illustrated above, there are three things to consider when planning a network – Capacity both in terms of bandwidth, as well as human resource. Second you need to consider the infrastrucure you will build to support the capacity. Ultimately, your network is only good as loong as it can help you meet business needs and costs. For both service providers as well as non profits, their planning should always look at 2-3 years in advance but at the same time, the infrastructure should be able to handle at least another year of operation.

Major Considerations

What we said above were the basic ruels but here are some more points to consider.

- Identify the components of a LAN/WAN and determine the type of network design most appropriate for a given site.
- Identify the different media used in network communications, distinguish between them, and determine how to use them to connect servers and workstations in a network.
- Differentiate between the different networking standards, protocols, and access methods, and determine which would be most appropriate for a given situation.
- Recognize the primary network architectures, identify their major characteristics, and determine which would be most appropriate for a proposed system.
- Identify the primary functions of network operating systems and distinguish between a centralized computing environment and a client/server environment.

- Determine how to implement and support the major networking components (including the server, operating system, and clients), and propose a system for adequately securing data on a given LAN and protecting the system's components.
- Distinguish between LANs and WANs, identify the components used to expand a LAN into a WAN, and determine how to implement an appropriate modem in the larger LAN/WAN environment.
- Identify strategic LAN support tools and resources, and determine how to use these in troubleshooting basic network problems

Services Planning

Mail

Choose a reliable MTA, but at the same time be cautious about SPAM, as it is a big headache.
Make redundant servers
Separate user servers from real servers

DNS

Use latest BIND releases
Plan nomenclature properly, but don't make it too obvious
Redundancy is most important
Arrange to host alternative DNS servers at off site / multiple locations.

User services

E-mail Access - POP, Web
Web Access
Transparent Caching / Proxy

Core Services – Infrastructure

Multi home, try to buy bandwidth from an IX facility.
Buying capacity is cheaper then managed capacity.
Plan to peer with other ISPs as much as possible

Infrastructure - Routers

Use loopback address in a separate subnet
Use the lookback address as RouterID.
For core routers – memory is important
For edge routers – ports are important
Take configuration backups regularly

Infrastructure - Switching

Switching capacity is never enough, invest in large switches, if that's what you'll need in future
Use VLANs to separate different groups of machines / networks
for Backbones, gigabit Ethernet is now more common

Infrastructure - Backbone

Switched Vs. Routed Backbone
The same decisions apply as in LAN connections,
Backbone itself can be switched, the traffic between different subnets can be routed.
Switching has its advantages if large local broadband use.

Infrastructure – Branch Offices

BOs need to be planned well in advance
BO tend to grow faster then you think they will

Basic considerations for BOs
Multihoming
Distributed user services
Authentication / remote management

Infrastructure - Hosts

Core Services
Use separate servers for separate functions
Free and Open Source software provide the best tools
1U servers are more manageable and also consume less power and space
Use standardized platform as much as possible.

Using FOSS

In an integrated environment, FOSS tools are used alongside proprietary software and tools. This is a common scenario, but when it comes to network infrastructure, resources and security, FOSS provides the best of all the software that are available. In an networked environment, the ability to quickly diagnose and solve problems are critical. Experienced network administrators know that 80% of all network related problems are to do with cabling and physical problems. A lot of problems can be minimized by using the best of the breed software in each category, and today as seen above, the best software in most cases are free and open source softwares.

...

FURTHER REFERENCES

The most useful resource for further reading is the Internet. Here are some useful sites on the Internet.

Resources on the Web.

General Reading / References

a. Free Software Foundation <www.fsf.org>
b. IOSN <www.iosn.net>
c. Linux Documentation Project <www.ldp.org>, www.linuxdoc.org.
d. Internet Standards and RFCs <www.faqs.org/rfcs>
e. Internet Society / NSRC Workskhop Resources <www.ws.isoc.org>
f. Network Startup Resource Center <www.nsrc.org>
g. UNESCO Free Software Portal http://www.unesco.org/webworld/portal_freesoft/index.shtml
h. Networking and Information Technology Observatory http://www.sdnp.undp.org/observatory/ - click on the "open source" theme

Linux Software Download and documentation

a. Source Forge <www.sourceforge.net>
b. Kernel .org <www.kernel.org>
c. Freshmeat <www.freshmeat.net>
d. RPMFind <www.rpmfind.net>
e. DNS/BIND <www.isc.org/bind>
f. Sendmail <www.sendmail.org>
g. SSL <www.openssl.org>
h. SSH <www.openssh.org>

Security Related sites

a. Security Focus <www.securityfocus.com>
b. CERT (www.cert.org>

Forums

a. Slashdot <www.slashdot.org>
b. Linux Online <www..linux.com>

Events

a. APRICOT <www.apricot.net>
b. NANOG <www.nanog.org>
c. AfNOG <www.afnog.org>
d. SANOG <www.sanog.org>
e. O'reilly Open Source Conference (www.oreilly.com)

Useful Mailing Lists

a. ISP-Linux <www.isp-linux.com>
b. NANOG <www.nanog.org>
c. AfNOG <www.afnog.org>
d. SANOG <www.sanog.org>

List of FOSS most commonly used by ISPs

Operating System:
Debian Linux : www.debian.org,

Accounting Authorization Authentication:
Cistron Radius: http://www.radius.cistron.nl
FreeRadius: http://www.freeradius.org
Freeside: http://www.sisd.com/freeside

Domain Name Server
BIND: http://www.isc.org/products/BIND

MAIL Servers
Sendmail: www.sendmail.org
Qmail: www.qmail.org
CourierImap - http://www.inter7.com/courierimap
Majordomo: http://www.greatcircle.com/majordomo

Web Related Software
Apache: www.apache.org
PHP: http://www.php.net

Proxy and Caching Servers
Squid: http://www.squid-cache.org

Security Related
Iptables: http://www.netfilter.org
Snort: http://www.snort.org
OpenSSH: http://www.openssh.org
OpenSSL: http://www.openssl.org

Network Management System:
MRTG: http://www.mrtg.org
Netsaint: http://www.netsaint.org
Nmap: http://www.nmap.org
Ntop: http://www.ntop.org

Browsers and E-mail Clients :
Firefox Internet Browser http://www.firefox.com
Mozilla http://www.mozilla.org
Thunderbird E-mail Client http://www.firefox.com

Books

Linux Books tends to get obsolete very quickly. But they are still a handy reference, when your access to the Internet is limited. Some books

O'reilly Books

Practical Internet and Unix Security
Linux in a Nut Shell,
Building Secure Servers with Linux,
DNS and BIND
Essential System Administration
Linux Security Cookbook
Network Troubleshooting Tools

Other O'reilly books on FOSS are also highly recommended. Though I've only included DNS/BIND, as it forms the basis of most other software operations.

Other Books

- Unix Unleashed - System Administrator's Edition - (Techmedia, India)
- Special Edition - Using Linux, 3rd Edition. (Prentice Hall of India Ltd.)

- Computer Networks by A. Tanenbum, (Prentice Hall)
-

GLOSSARY

Server : A server is the powerful computer which serves resources to other computers on the network. The server can also be a big storage space, as well as a database. The key here is the special software, which enables the hardware to provide services.

Client : A client is the computer on the network, which uses the server to get resources which may be e-mails, docments or even data.

Inernet : Internet is commonly referred to as 'networks of networks'. In technical terms, Internet is the global network of computers with are able to communicate in TCP/IP with each other.

Intranet : A private network, which operates on the same principles of the Internet is called the Intranet. Intranets also use the TCP/IP protocol.

SSL : Secure Sockets Layer, is a secure communications layer, that makes it possible to use the web for e-commerce transastion. It makes sure that all communications between a client and server is encrypted.

RFC: Requests for Comments are

IETF : Internet Engineering Task Force, is the Internet Standards making organization. IETF publishes the RFC series documents as Internet Standards. Www.ietf.org.

VLAN : Virtual LAN is a feature provided by many Ethernet switches, to allow to create multiple logical networks over the same physical connections. A single switch with three VLANs defined is same as three different physical switches.

VPN : Virtual Private Network are encrypted private networks which are running on top of the public network like the Internet.

Daemon: In Unix and GNU/Linux, Daemons are server software that constantly run on top of the operating system. In other words, Daemons are server software.

Loopback : Loopback is a virtual interface defined in all TCP/IP computers. This virtual interface is assigned the IP address 127.0.0.1 and is required for TCP/IP to work properly.

Localhost : Computers with TCP/IP installed refer to their loopback interface as the localhost. This is defnied by default in the DNS software.

Hosts : Any device connected to the Internet or an TCP/IP network is referred to as the host.

PoP: Point of Presence – an Internet Service Providers branch office. Not to be confused with POP3, which is an e-mail protocol and stands for Post Office Protocol.

Port : The points of contacts with any TCP/IP host. TCP/IP computers with IP address have ports ranging from 1 to 65336, which act as point of contacts for different services running on that computer. Eg, the SMTP protocol uses port 25, http users port 80 and so on.

ABOUT THE AUTHOR

Gaurab Raj Upadhaya

gaurab@lahai.com

Gaurab Raj Upadhaya, is currently employed as Internet Economics Analyst / Staff Engineer, at Packet Clearing House (www.pch.net), a research non-profit based in Berkeley, California. He works mostly in Internet backbone operations, analysing peering relationships between operators and roles of Internet Exchange Points in different parts of Asia. Much of the work involves training ISPs in developing countries about best practices on network operations. He also runs the PCH INOC-DBA (www.pch.net/inoc-dba) hotline phone system for service providers. He initiated the Nepal Internet Exchange (npIX) and currently serves as its voluntary CEO.

In 2003, Gaurab started the South Asian Network Operators Group (SANOG), a non-profit educational event and forum for ISPs in the South Asian Region (www.sanog.org). SANOG has a track dedicated to in-depth hands on workshops on ISP operations using open source, along side other workshops on BIND/DNS and Security. He currently chairs SANOG, and is the vice-chair of Asia Pacific Internet Association (www.apia.org), and Management Committee member for Asia Pacific Regional Internet Conference on Operational Technologies (APRICOT). (www.apricot.net)

In the past he has served as a National UN Volunteer, working for the UNDP/Cisco/USAID Least Developing Countries Initiative (www.cisco.com/edu/ldci) as Unites volunteer (www.unites.org). He has been using Linux and open source software since 1996. He previously worked as system and network administrator for United Mission to Nepal (www.umn.org.np).

He is Cisco Certified Network Associate (CCNA). His personal web site is www.gaurab.org.np. He likes travelling, photography and reading books when he is not on the network.

ACKNOWLEDGEMENT

I'd like to acknowledgement all those who've helped and contributed to the preparation of this primer. My collegues at PCH, Bill Woodcock, and Daniel S. Silverstein have provided both structural and content inputs. Mr. Ritesh Raj Josti, Mercantile Communications, who is my co-instructors at APRICOT, has provided the basis for a lot of security related materials.

I'd also like to thank the IOSN Team, ken, sunil, phet and Shahid on being patient and persistent with the primer.

Lastly thanks to all the reviewers, .. <names to be added>