AAiT
ADDIS ABABA INSTITUTE OF TECHNOLOGY
አዲስ አበባ ቴክኖሎጂ ኢንስቲትዩት
ADDIS ABABA UNIVERSITY
አዲስ አበባ ዩኒቨርሲቲ

Enterprise Systems and Network Administration

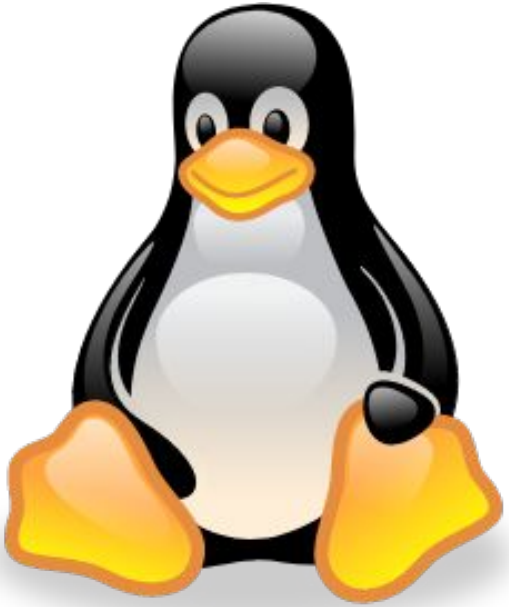
CHAPTER THREE

Linux System Administration

Getnet M.
getnet.mamo@aau.edu.et

2. Linux System Administration

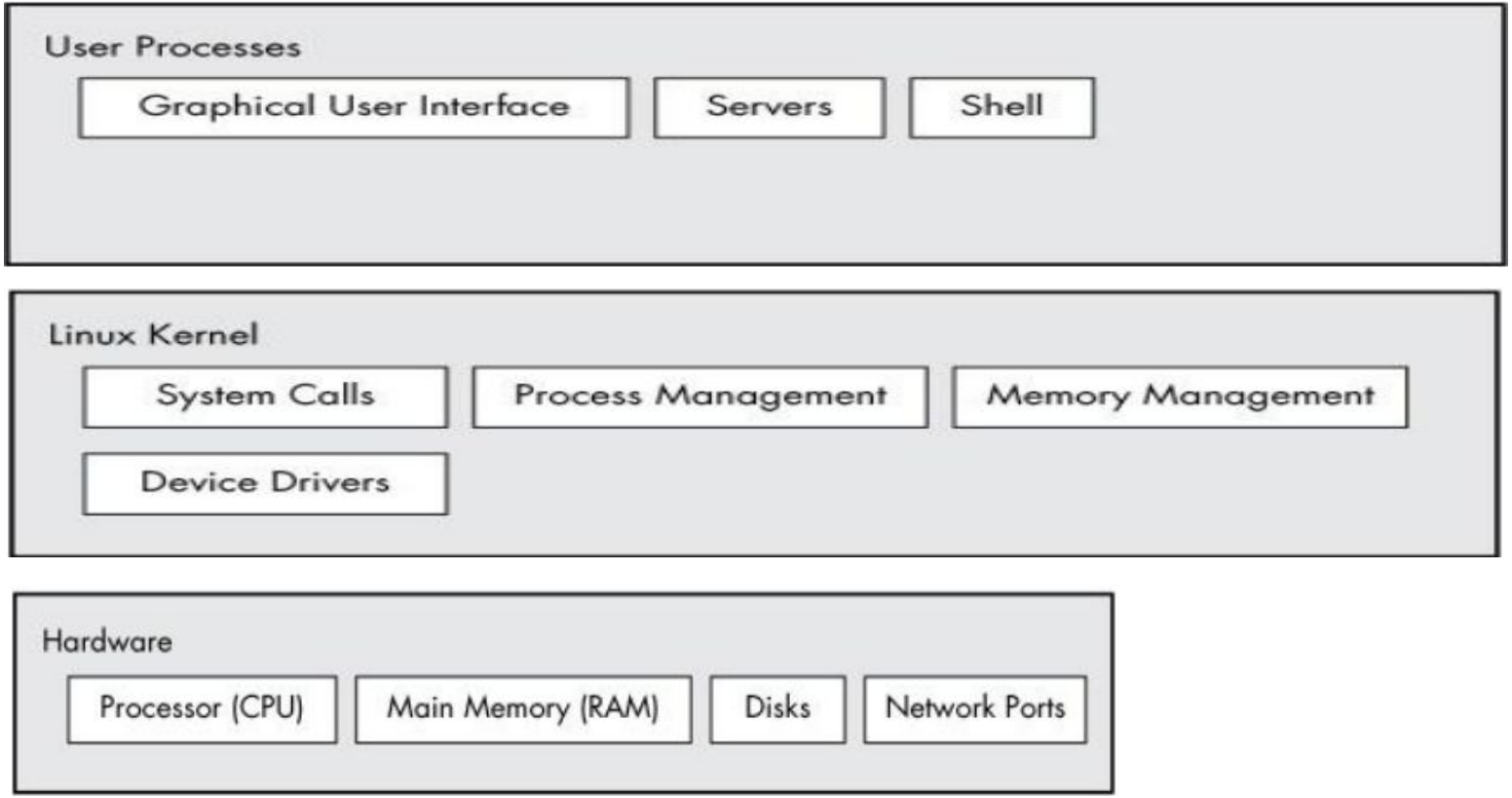
2.1 Introduction to Linux
2.2 Linux Networking
2.2 Linux Firewall
2.3 Linux Proxy



Introduction to Linux

- What is Linux?
  - Operating System
- What is Operating System?
  - Wikipedia: *“the system software responsible for the direct control and management of hardware and basic system operations”*.
- Function of Operating system
  - Memory management
  - Process management
  - Network management
  - File management
  - I/O management

Introduction to Linux



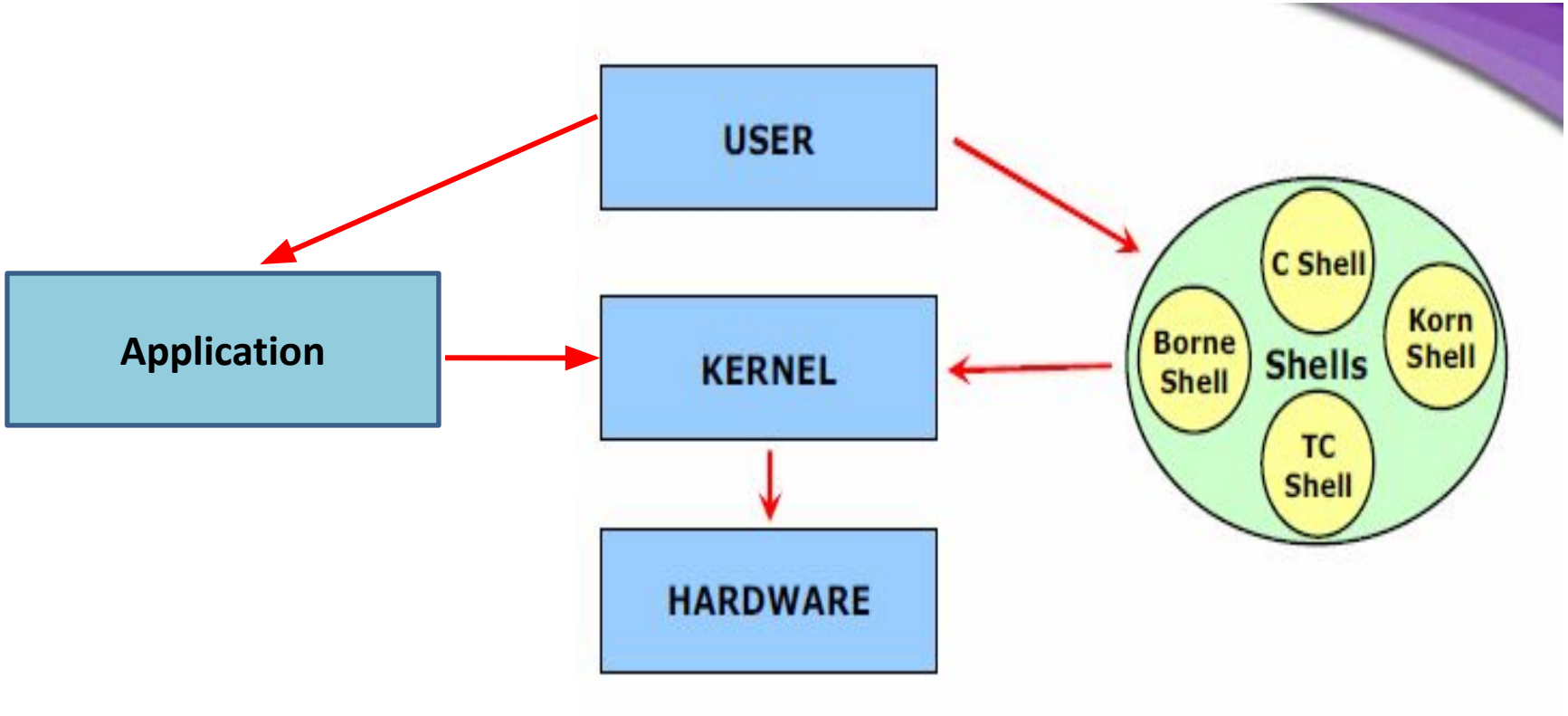
What is kernel?

- **A set of functions that make up the heart of an OS**
- **It is used to provide an application interface between programs and physical devices.**
- **Services provided by the kernel:**
  - Controls execution of processes.
  - Scheduling processes fairly for execution on the CPU.
  - Allocating memory for an executing process.

What is User Process?

- **Processes—the running programs that the kernel manages:**
  - Shell, user program, web-server,
- **Services provided by the shell:**
  - **It interprets all the commands to the kernel**
  - **The kernel after processing the commands gives back to the shell.**

Introduction to Linux



What is Linux?

- **A fully-networked 32/64-Bit Unix-like Operating System**
  - Unix Tools Like sed, awk, and grep
  - Compilers and tools Like C, C++, Fortran, Smalltalk, Ada, JDK, Python,
  - Network Tools Like telnet, ftp, ping, traceroute
- **Multi-user, Multitasking, Multiprocessor**
- **Has the X Windows GUI**
- **Coexists with other Operating Systems**
- **Runs on multiple platforms**
- **Includes the Source Code**

Why Linux?

- **Today Linux has joined the desktop and mobile market.**
- **On the server side, Linux is well-known as a stable and reliable platform.**
- **Linux provides many applications like:**
  - Databases (MySQL,Postgresql),
  - Network services(Web Servers,DNS, Proxy, firewall etc)
  - Software development tools(C, Java, Python,Perl etc.)
  - Office automation tools
  - And many more…

Why Linux?

- There is excellent and free Internet support and documentation available.
- The graphical user interface (GUI) is similar in design to that on any other system
- A very powerful command line alternative is also available.
- Linux *is* user friendly.

Why Linux?

- It is Open Source
  - Today, Linux is ready to accept the challenge of a fast-changing world.
- Linux is free:
  - If you want to spend absolutely nothing, you don't even have to pay the price of a CD.
  - Linux can be downloaded in its entirety from the Internet completely for free.

Why Linux?

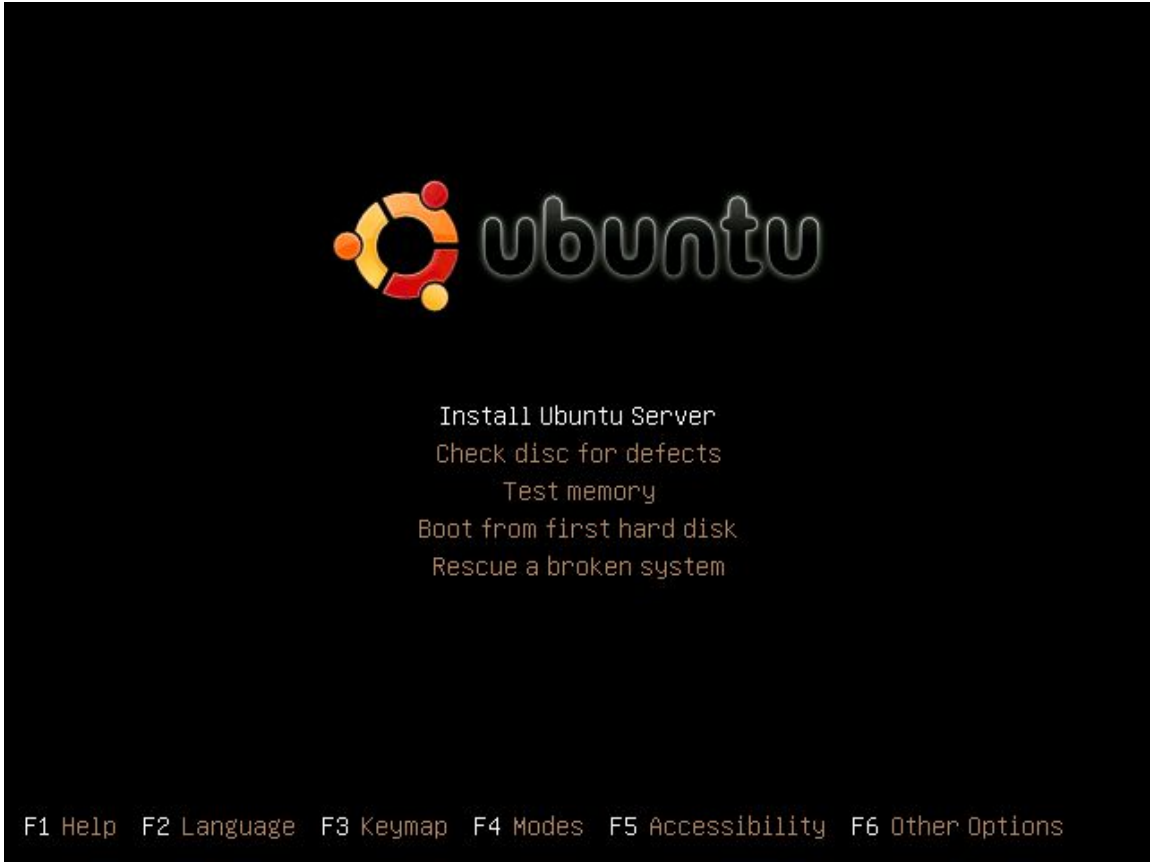
- **Linux is portable to any hardware platform.**
  - Laptop, Desktop, Server, Mobile, embedded-system,
- **Linux was made to keep on running.**
  - As with UNIX, a Linux system expects to run without rebooting all the time.
  - Tasks can be scheduled to run at suitable times.
- **Linux is secure and versatile.**
  - The security model used in Linux is based on the UNIX idea of security which is robust.
  - It is less prone to virus attacks.
- **Linux is scalable**

Linux Distributions

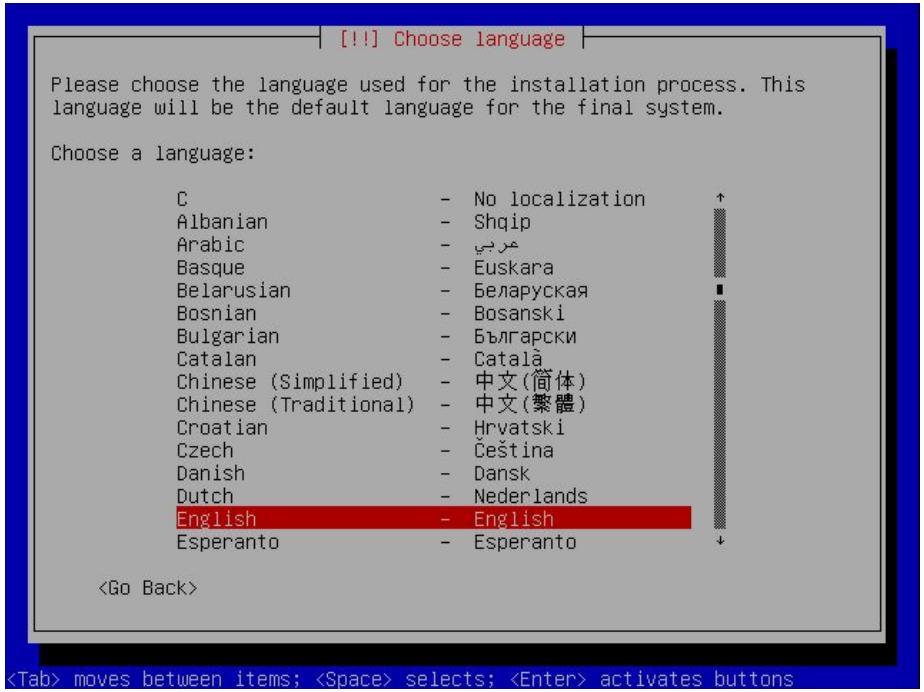
Linux Distributions

- **Some individual students or companies add some applications and tools to Linux then they began to distribute their own choice of packages bound around Linux basic kernel.**
- We call this individual sets as distributions.
- **Today there are hundreds of different distributions available popular Linux distributions include**
  - **SUSE Linux**
  - **Fedora Linux**
  - **RedHat Linux**
  - **Debian Linux**
  - **Linspire**
  - **Gentoo Linux**
  - **Slackware Linux**
  - **TurboLinux**
  - **Mandrake Linux**
  - **Lycoris Linux**
  - **CentOS**
  - **ALT Linux**
  - **Ubuntu**

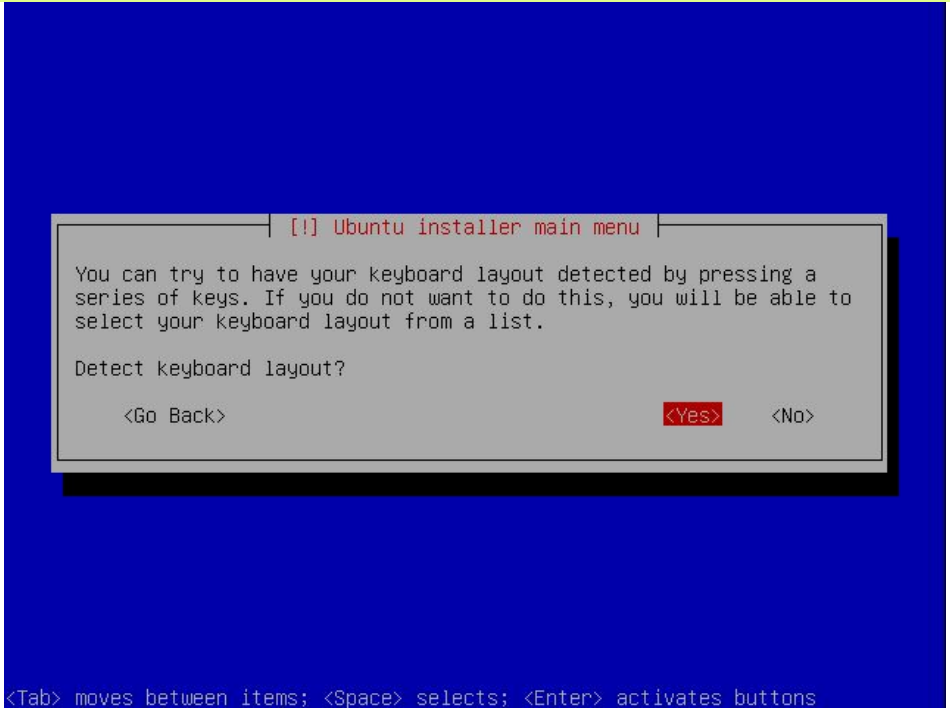
Linux Installation

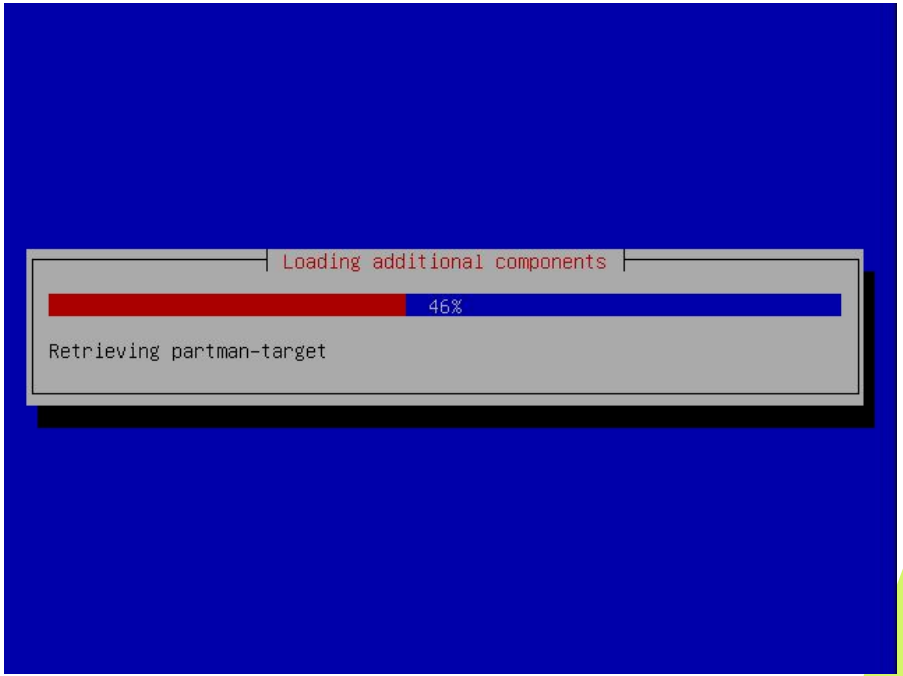


Linux Installation

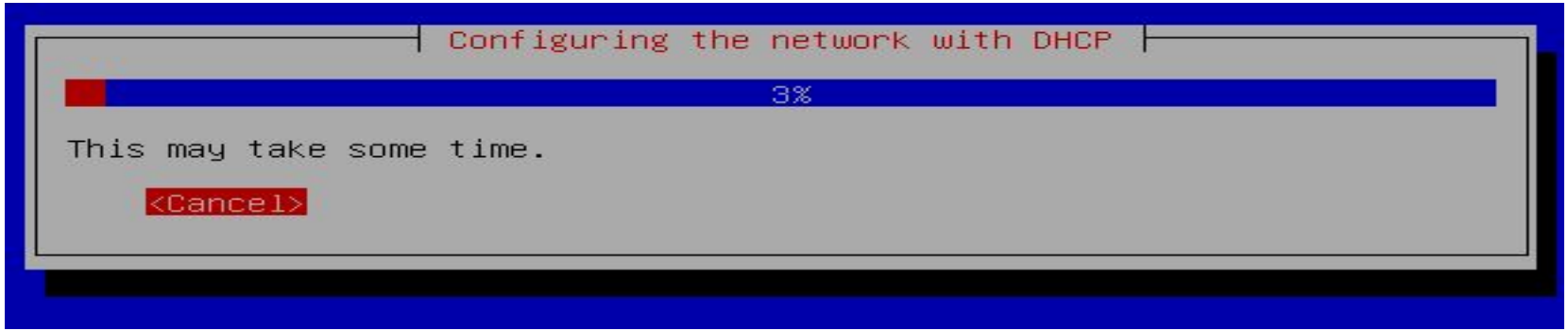


Linux Installation

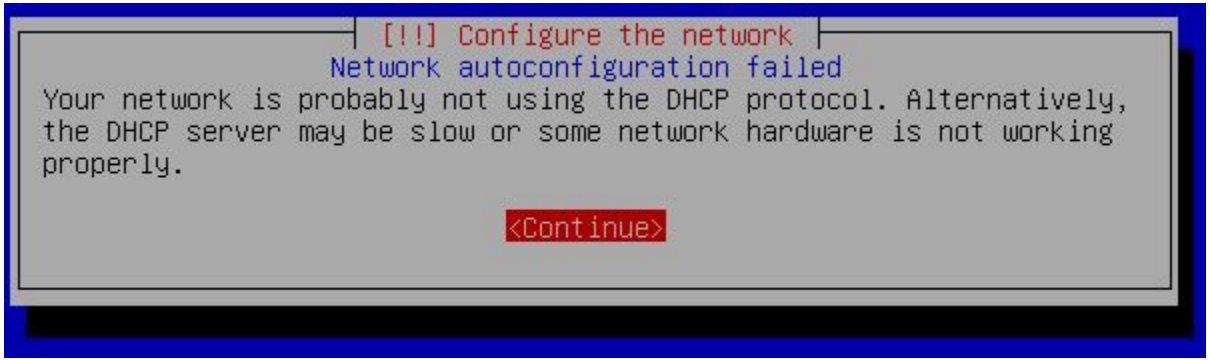




Linux Installation

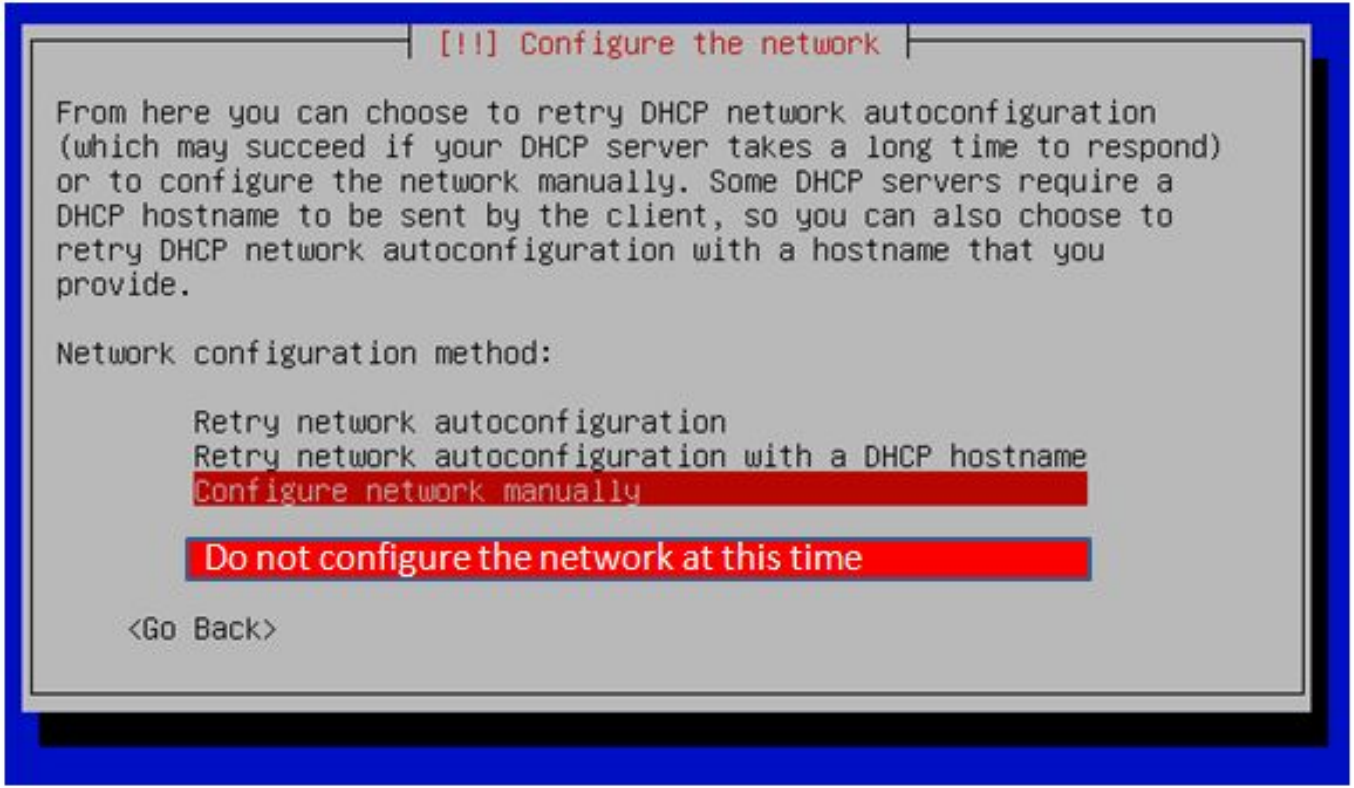


Press “Cancel” to configure your network manually.

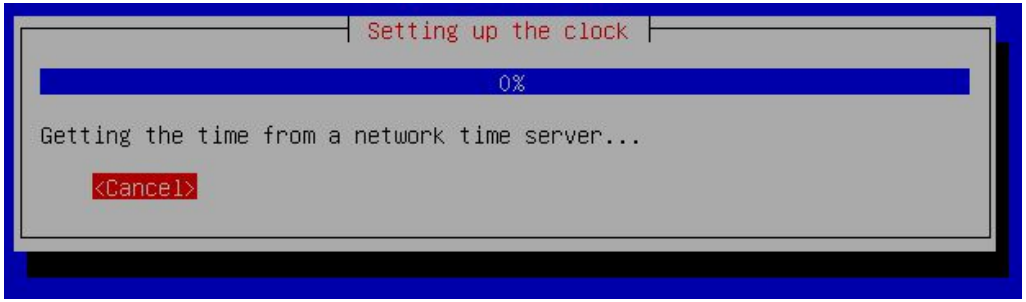


Press “Continue” to continue.

Linux Installation



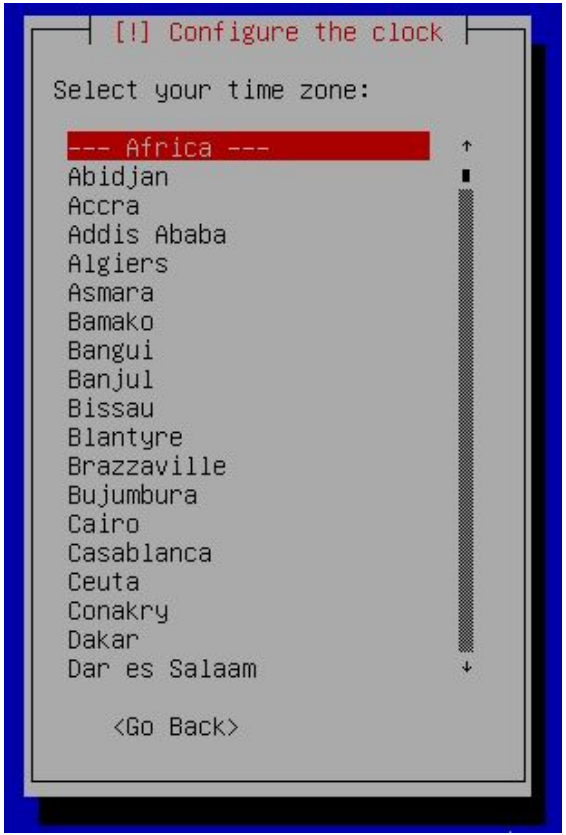
Linux Installation



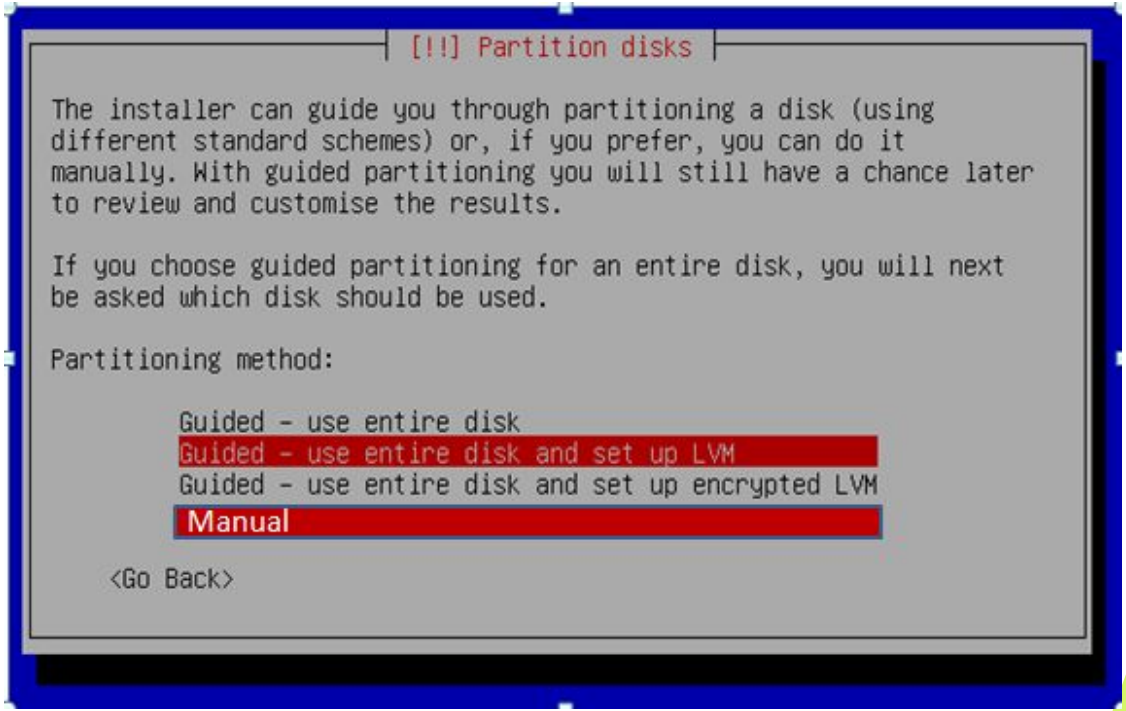
It's important that you configure your clock for the time zone of Addis Ababa.

If you choose another time zone some of your server settings will not be optimal – including the locations where you obtain additional software for your installation.

Linux Installation



If you chose “Select from worldwide list” in the previous step, then scroll down the screen until get you Addis Ababa. Highlight your choice and press <ENTER> to continue.

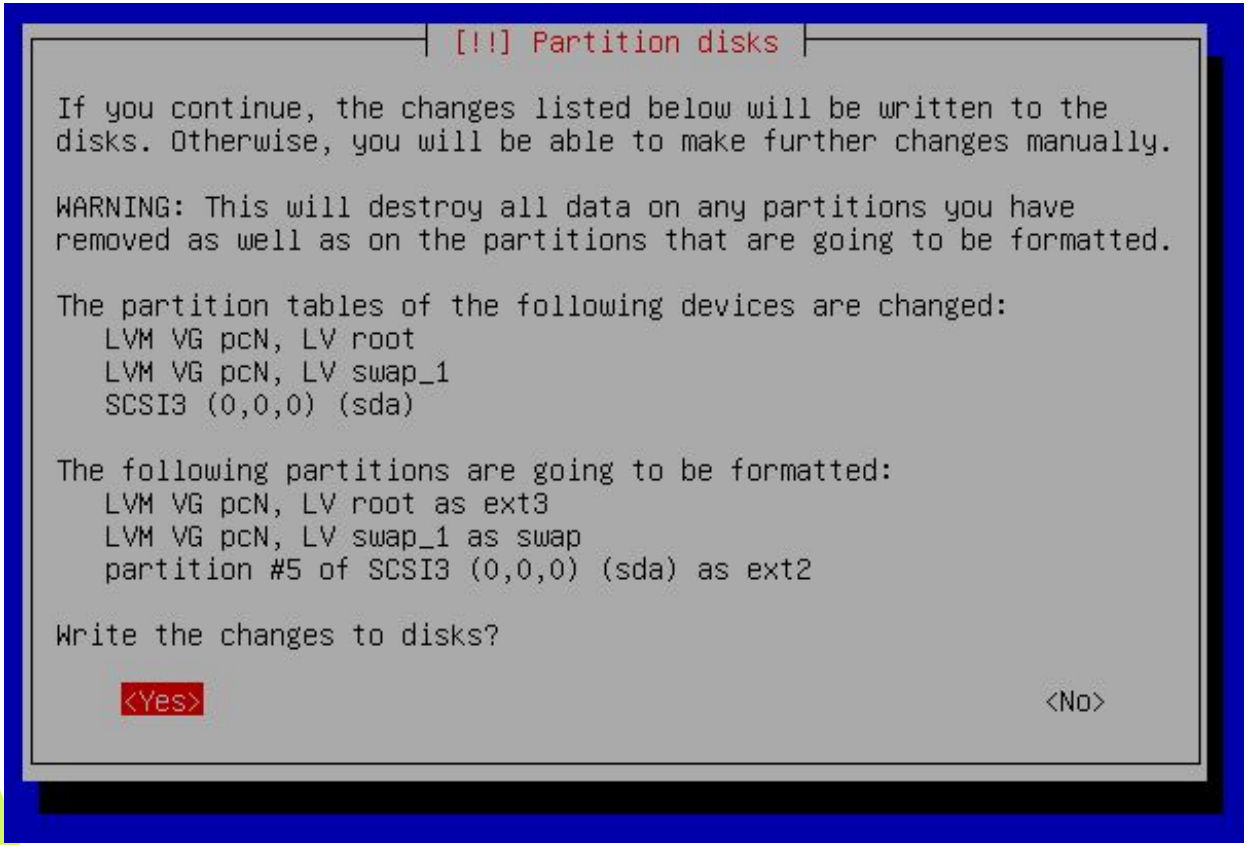


Linux Installation

- In Windows, each drive is often a partition
  - The C: drive often represents one hard drive
- In Linux, there are a minimum of two partitions needed
  - '/' is the root partition, and can include all files and directories
  - A *swap space* is also needed, at least equal in size to your RAM
- To make dual-boot system, use a separate partition for linux.

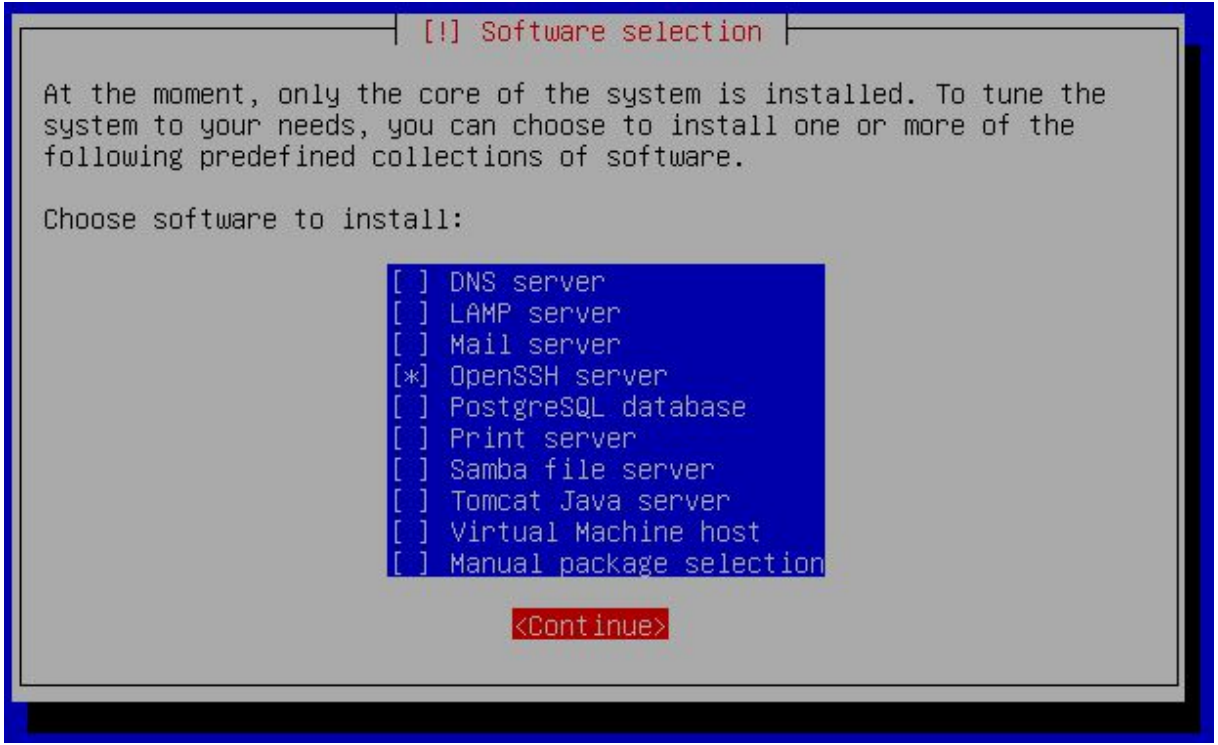
- Make sure you manually partition the hard-disk, most Linux installers use the *whole disk* as the **default** option.
- In that case, you may **lose all your data**, and your **Windows installation** if you have one.

Linux Installation

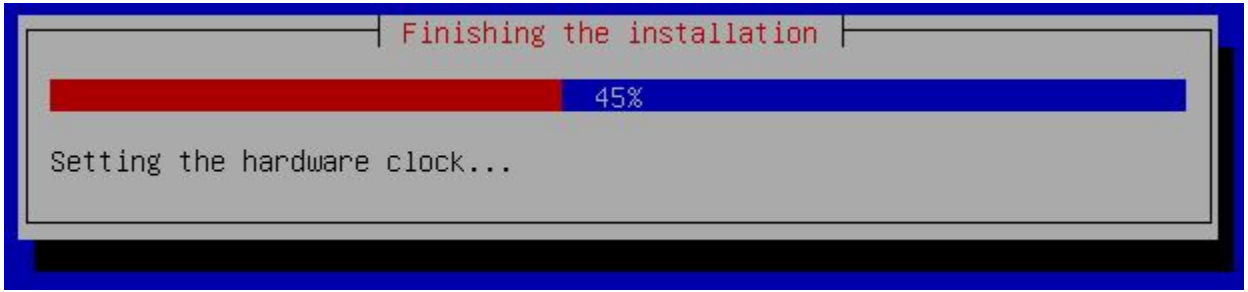


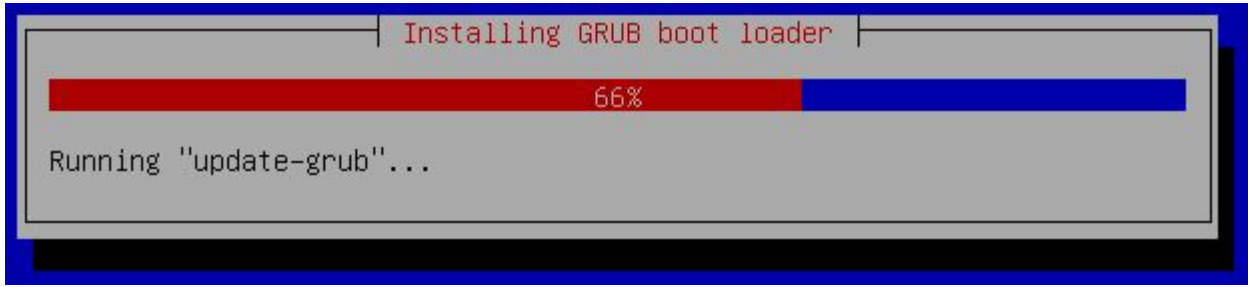
Select “<Yes>” and press <ENTER> to continue.

Linux Installation



Linux Installation



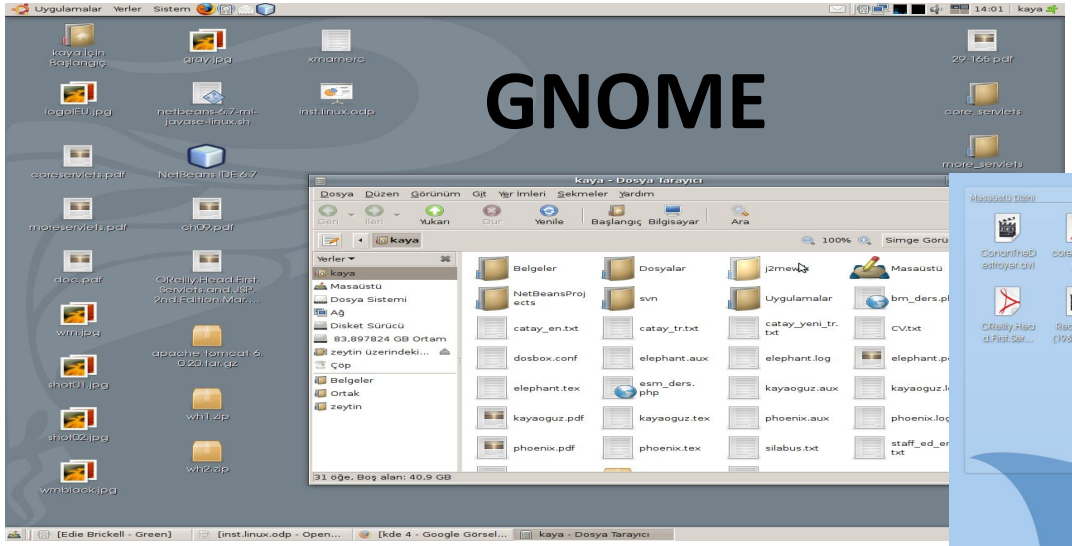


These should appear on your screen as Ubuntu finishes its installation process.

Linux Installation

- Depending on your op. sys. choice, you'll likely to get one of the following desktop environments: GNOME (Ubuntu) or KDE (Pardus).
- There are other desktop environments (XFCE, Window Maker, Fluxbox, IceWM etc.) But Unity, GNOME and KDE are the most popular.
- Both of them will look familiar with a 'Start' menu, taskbar, system tray, icons on desktop, drag and drop etc.

Linux Installation



Managing Software

What is software package management?

- A way to distribute software and configuration
- Eg.
  - .tar.gz or tgz (Slackware)
  - .rpm (Red Hat, Fedora, SUSE, …)
  - .deb (Debian, Ubuntu)
  - .exe or msi (Windows)
- Meta-package managers
  - Locate packages on the Internet, download, install and analyze inter-package dependencies. eg.
  - yum (rpm)
  - apt-get (deb and rpm)

Managing Software

- Debian binary package file names use the following convention:

```
<foo>_<VersionNumber>-<DebianRevisionNumber>.deb
```

- A `.deb` file is a GNU archive file containing several mandatory files:
  - `debian-binary` (contains format version number)
  - `control.tar.gz` (contains series of plain files, of which the file `control` is mandatory and contains the core control information)
  - `data.tar.gz` (contains the files-system archive of the items to be installed)

The Debian package system has a range of package "dependencies”:

- Package A **depends** on Package B if B absolutely must be installed in order to run A.
- Package A **recommends** Package B, if the package maintainer judges that most users would not want A without also having the functionality provided by B.
- Package A **suggests** Package B if B contains files that are related to (and usually enhance) the functionality of A.
- Package A **conflicts** with Package B when A will not operate if B is installed on the system.
- Package A **replaces** Package B when files installed by B are removed and (in some cases) over-written by files in A.
- Package A **provides** Package B when all of the files and functionality of B are incorporated into A.

Managing Software

Tools:

- **dselect** – menu-driven package management tool
- **dpkg** – install package (package-file centric)
- medium-level tool set to install, build, remove, and manage Debian packages
- **apt-get** – install package (package-archive centric)
- **tasksel** – install task (a set of packages)
- **aptitude** – install package (package & task)
- **synaptic**, **gsynaptic** – GUI APT alternatives

Managing Software

Tools:

- When the Debian distribution was small, `dselect` and `dpkg` were adequate tools for package management.
- As the collection has grown in size and complexity, these tools are no longer provide a smooth and easy installation.
- The APT system provides a new suite of tools to augment `dpkg` and provide a better installation tool than `dselect`.
- The primary tool of the APT system is the command **`apt-get`** `or simply` **`apt`**.

Managing Software

- The list of mirrors and sources are kept in `/etc/apt/sources.list`
- The syntax of a mirror record is
- `deb uri distribution [component1] [componenent2] […]`
- Examples
- `deb http://http.us.debian.org/debian stable main contrib non-free`
- `deb ftp://ftp.debian.org/debian stable contrib`
- `deb-src file:/home/jason/debian unstable main contrib non-free`
- Notice
  - `deb` and `deb-src` (deb-source)
  - Distribution: `stable`, `unstable`, and `testing`
  - Branch: `main`, `desktop`, and so on
  - free, non-free, contrib

Managing Software

- The primary APT commands are :
  - **`apt-cache`** and
  - **`apt-get`**
  - `apt-cache` manipulates packages stored in the APT cache (see the man page for details)
  - `apt-get` retrieves and installs applications from a source listed in `/etc/apt/sources.list`

apt-get Commands

apt-get update – updates the list of available packages

- **apt-get install foo** – gets the latest version of a package named **foo**
- **apt-get** remove **foo** – removes the package **foo**
- **apt-get upgrade** – upgrades any installed packages
- **apt-get dist-upgrade** – upgrades the distribution

Managing Software

- To keep your system current, all you have to do is periodically run the following from **`cron`**:
  - `apt-get update`
  - `apt-get upgrade`
- And occasionally, run `apt-get dist-upgrade`

```
debian:/etc/apt# apt-get upgrade
+ apt-get upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
1 packages upgraded, 0 newly installed, 0 to remove and 0  not upgraded.
Need to get 194kB of archives. After unpacking 0B will be used.
Do you want to continue? [Y/n] y
Get:1 http://ftp.kulnet.kuleuven.ac.be woody/main debhelper 4.0.2.openoffice [19
4kB]
Fetched 194kB in 6s (31.8kB/s)
(Reading database ... 62633 files and directories currently installed.)
Preparing to replace debhelper 4.0.2 (using .../debhelper_4.0.2.openoffice_all.d
eb) ...
Unpacking replacement debhelper ...
Setting up debhelper (4.0.2.openoffice) ...

debian:/etc/apt#
```

Managing Software

- To install a package, you only need to know its common name like **ssh, openssh, apache2, mysql-Server, pdfsam,** and so on.
- To install **apache** web server:
  - **apt-get install apache2**
- If you don't know the common name of the package enter the first few letters and press **Tab**, system will give names of possible packages.
- The best place to get a package name is:
  - http://www.debian.org/distrib/packages.

Managing Software

```
debian:~# apt-get install sendmail
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  exim
The following NEW packages will be installed:
  sendmail
0 packages upgraded, 1 newly installed, 1 to remove and 0  not upgraded.
1 packages not fully installed or removed.
Need to get 0B/918kB of archives. After unpacking 1778kB will be used.
Do you want to continue? [Y/n]
```

File System

- Filesystem is a method for storing and organizing computer files and the data they contain to make it easy to find and access.
- Different operating systems normally use different file systems.
- Consists of files/directories and information needed to locate/access those objects
- Linux file system ext2 similar to ext3
  - Major difference is that ext3 is a journaling file system

File System

- The file system in Linux stores
  - the kernel
  - the executable commands supported by the OS
  - configuration information
  - user data
  - and special files that are used to give controlled access to system hardware and OS functions.

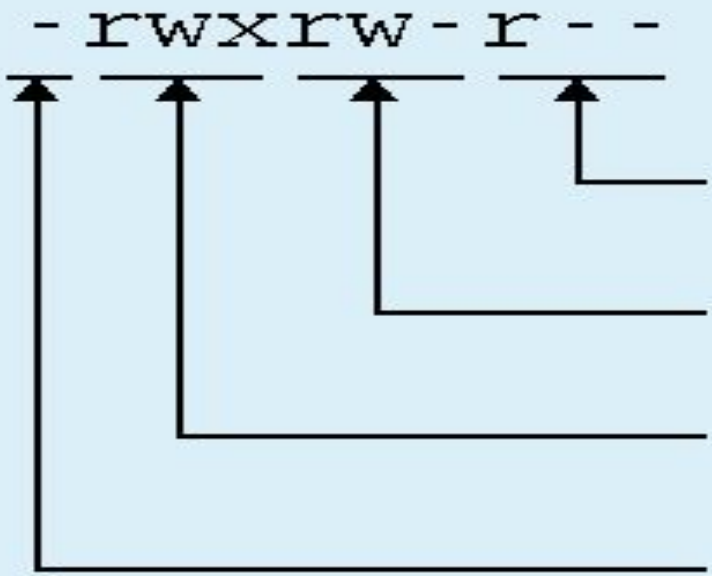
File System

- Items stored in the filesystem are of four types:
  - **Ordinary files** contain text, data, or program information. Files can not contain other files or directories.
  - **Directories** containers that hold files, and other directories.
  - **Devices** are used in the same way as ordinary files providing applications with easy access to the hardware devices.
  - **Links** which is a pointer to another file

File System

- File Permissions help you protect your files against other users on the system
- Three access levels:
  - User: The user that created the file
  - Group: All users in the group that owns the file
  - Others: All others
- Three permissions:
  - Read (**r**): Read content of file or list content of directory
  - Write (**w**): Change content of file or create/delete files in directory
  - Execute (**x**): Execute file as program or use directory as active directory

File System



Read, write, and execute permissions for all other users

Read, write and execute permissions for members of the group owning the file.

Read, write and execute permissions for the owner of the file.

File type. "–" indicates a regular file. A "d" indicates a directory.

File System

chmod: command is used to change the permissions of a file or directory

:~# chmod 777 myFile

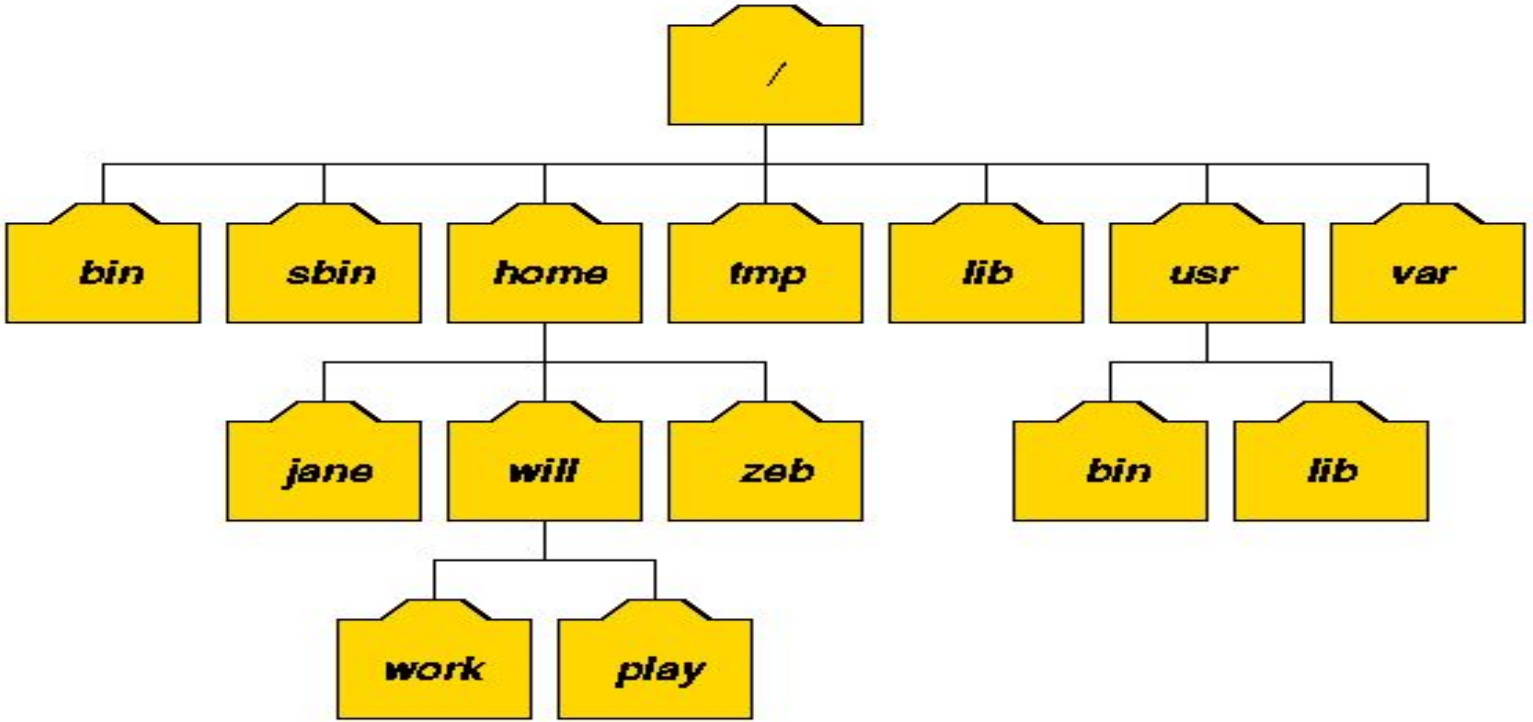
:~# chmod a+w myFile

:~# chmod u=rx myFile

File System

- The system is laid out as a hierarchical tree structure. The top-level directory is the 'root' designated by a slash '/'.
- Each directory can have many child directories, but only one parent directory.
- Physical devices are mounted on mount points
  - Floppy disks
  - Hard disk partition
  - CD-ROM drives
- No drive letter like A:, C:, …

File System



File System

- The path to a certain location can be specified as:
  - Absolute path from root

    e.g.

    /root/home/will
  - Relative path

    e.g.

Accessing play from user “zeb”

../will/play

File System

Directory	Stands for	Content
/	root	Top-level directory in the hierarchical tree
/bin	Binaries	Contains binaries used by both the system administrator and non-privileged user e.g. command ‘ls’ is stored here
/dev	Devices	Contains hardware devices directories. It is a virtual directory
/etc	Et cetra	Contains configuration files for running applications
/home		Contains user subdirectories
/lib	Libraries	Contains shared libraries e.g. C, Perl, Python general libraries
/mnt	Mount	
/proc	Processes	Contains information about the system e.g. process that are running. It is a virtual directory
/root		Default home directory for the system administrator. Isolated to increase security

File System

Directory	Stands for	Content
/sbin	Secure Binaries	Contains secured binaries that are only accessed by privileged users e.g. fdisk, partitioning tool is kept here
/tmp	Temporaries	Contains temporary files
/usr	Unix System Resources	Contains subdirectories such as /usr/doc which contains system documentations, /usr/local the local hosts directory
/var	Variables	Contains log and spool files
/boot		Contains Linux kernel

Manage users

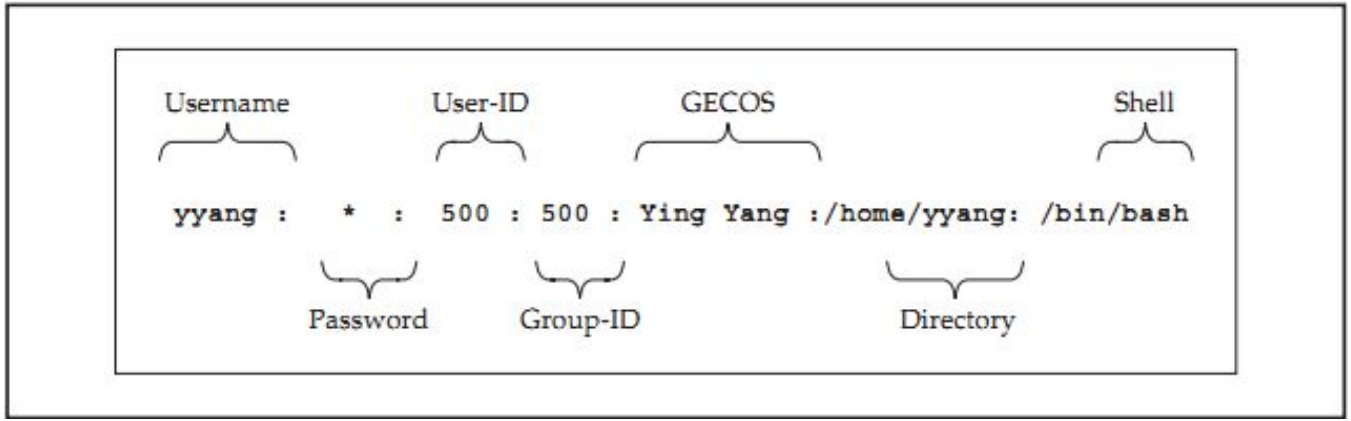
- Under Linux, every file and program must be owned by a user.
- Each user has a unique identifier called a user ID (UID).
- Each user must also belong to at least one group (collection of users established by the system administrator).
- Users may belong to multiple groups.
- Like users, groups also have unique identifiers, called group IDs (GIDs).
- A running program inherits the rights and permissions of the user who invokes it

Manage users

- All user information is stored in straight text files
  - enable you to edit user information using simple text editor
  - enables administrators to develop user management tools
- Common user information files
  - /etc/passwd
  - /etc/shadow
  - /etc/group

The /etc/passwd File

- The /etc/passwd file stores the user's login, encrypted password entry, UID, default GID, name, home directory, and login shell.



Manage users

The /etc/shadow File

- contains encrypted password for user accounts
- also contains optional password aging or expiration information
- Each line represents a single user with fields:
  - Login name
  - Encrypted password
  - Days password was last changed
  - Days before password may be changed
  - Days after which password must be changed
  - Days before password is to expire that user is warned
  - Days after password expires that account is disabled
  - Days account is disabled
  - A reserved field

mmel:1HEWdPIJ.$qX/RbB.TPGcyerAVDlF4g.:12830:0:99999:7:::

Manage users

The /etc/group File

- contains a list of groups, with one group per line
- Each group entry in the file has four standard fields, with each field colon-delimited
- The fields are:
  - Group name -The name of the group
  - Group password-This is optional, but if set, it allows users who are not part of the group to join
  - Group ID (GID)-The numerical equivalent of the group name
  - Group members-A comma-separated list

bin:x:1:root,bin,daemon

Command-Line User Management

- useradd-As the name implies, useradd allows you to add a single user to the system.

```
usage: useradd [-u uid [-o]] [-g group] [-G group,...]
               [-d home] [-s shell] [-c comment] [-m [-k template]]
               [-f inactive] [-e expire ] [-p passwd] [-M] [-n] [-r] name
   useradd -D [-g group] [-b base] [-s shell]
               [-f inactive] [-e expire ]
```

Manage users

usermod

- The usermod command allows you to modify an existing user in the system.
- It works in much the same way as useradd.

```
usage: usermod [-u uid [-o]] [-g group] [-G group,...]
               [-d home [-m]] [-s shell] [-c comment] [-l new_name]
               [-f inactive] [-e expire ] [-p passwd] [-L|-U] name
```

Manage users

userdel

- The userdel command does the exact opposite of useradd—it removes existing users

usage: userdel [-r] username

groupadd

- The groupadd command adds groups to the /etc/group file

usage: groupadd [-g gid [-o]] [-r] [-f] group

groupdel

- Even more straightforward than userdel, the groupdel command removes existing groups specified in the /etc/group file.

usage: groupdel group

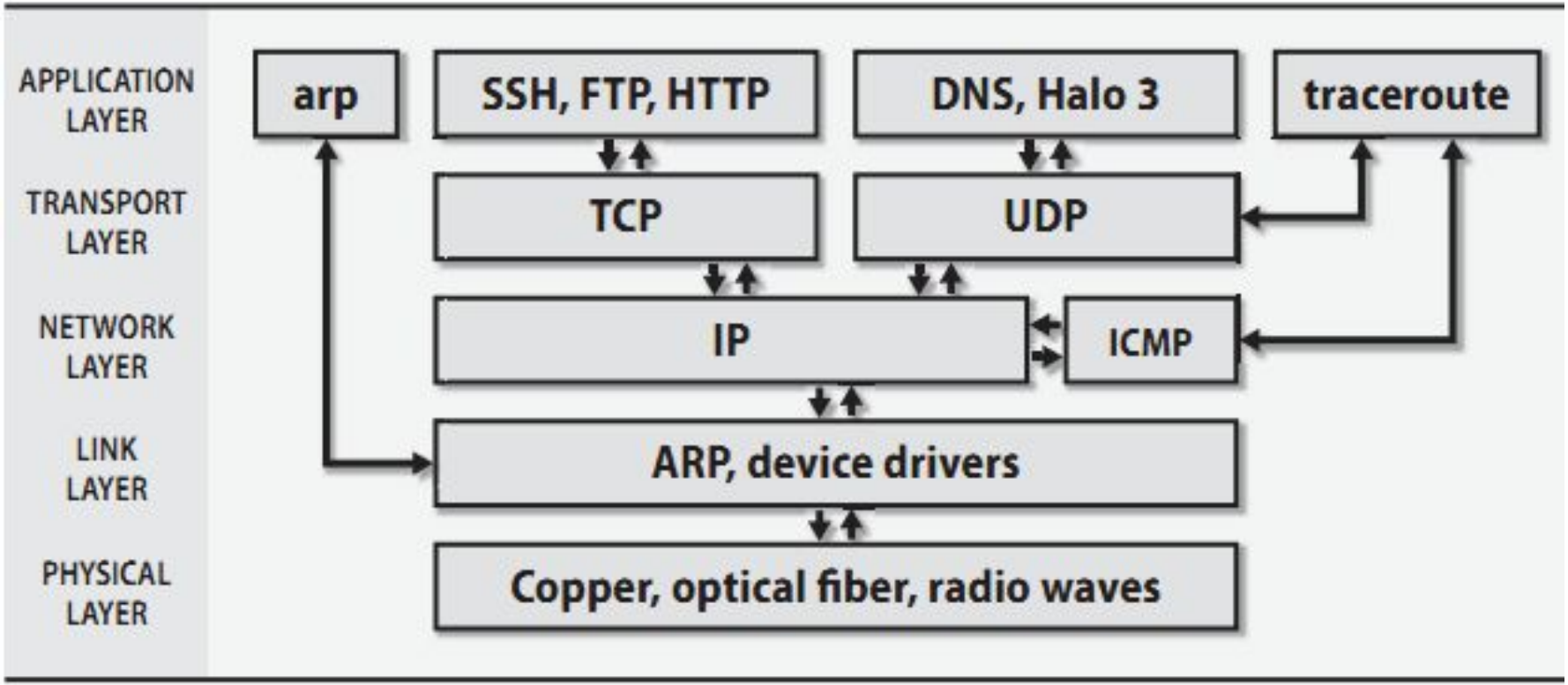
groupmod

- The groupmod command allows you to modify the parameters of an existing group.
- The options for this command are

usage: groupmod [-g gid [-o]] [-n name] group

Linux network

TCP/IP Protocol Stack



TCP/IP Packet Encapsulation

A typical network packet[4]

Ethernet header	IPv4 header	UDP header	Application data	Ethernet CRC
14 bytes	20 bytes	8 bytes	100 bytes	4 bytes

UDP packet (108 bytes)

IPv4 packet (128 bytes)

Ethernet frame (146 bytes)

Connecting to a Network

- Hostname and IP Address assignment
- Configuration of hardware
- Default route (gateway) assignment
- Name Service Configuration
- Testing and troubleshooting

Addressing Mechanism

- Like letters or email messages, network packets must be properly addressed in order to reach their destinations.
- Several addressing schemes are used in combination:
  - MAC (media access control) addresses for use by hardware
  - IPv4 and IPv6 network addresses for use by software
  - Hostnames for use by people

Address Types

- The IP layer defines several broad types of address, some of which have direct counterparts at the link layer:
  - Unicast – addresses that refer to a single network interface
  - Multicast – addresses that simultaneously target a group of hosts
  - Broadcast – addresses that include all hosts on the local subnet

Hostnames

- Uniquely identifies each system
- Fully Qualified Domain Name
  - **`hostname.site.domain[.country]`**
  - Country: 2 letter identifier for country (et, uk)
  - Domain: Type of site (edu, com, org, gov)
  - Site: Unique name of organization (aau, bdu)
  - Hostname: Unique name of system (www, mail)
- **`hostname`**: Display or set system name

Interface Configuration

- Hardware to connect to network
- Common interfaces
  - Ethernet
  - Modem
- **`ifconfig`** – View/Configure interface (Linux)
- **`ipconfig`** – View interface (Windows)
- **`netplan`** - network configuration (Linux)

Interface Configuration

- Any device use symbol to determine
  - eth0: Ethernet device number 0
  - eth1: Ethernet device number 1
  - lo : local loopback device
  - Wlan0 : Wireless LAN device 0

ipconfig

- ifconfig eth0 add 10.133.120.153 broadcast 10.133.120.255 netmask 255.255.255.0
- Netmask forces TCP/IP to go only to the router interface for any address except those in 10.133.120.
- Broadcast limits broadcasts to the 10.133.120. subnet

Configuring /etc/hosts file

- The /etc/hosts file is just a list of IP addresses and their corresponding server names.
- Check this file before querying DNS.
- Update this file if hostname or IP changed
  - Impractical to use company-wide
- Use a centralized DNS server to handle most of the rest.
  - Use this file if you are not managing DNS

Configuring /etc/network/interfaces

- The file /etc/network/interfaces file stores permanent IP address assignment to interfaces

  auto eth0

  iface eth0 inet dhcp/static

  address *ip4*

  gateway *defaultgatewayIP*

  netmask *subnetmask*

  broadcast *broadcastaddress of subnet*

Configuring /etc/network/interfaces

- Example configuration

```
auto eth0
iface eth0 inet static
    address 10.133.120.20
    netmask 255.255.255.0
    gateway  10.133.120.1
    broadcast 10.133.120.255
```

- Save file and restart network

```
#/etc/init.d/networking restart
#service network restart
```

Name Services

- **`/etc/hosts`**
  - Local configuration
  - Localhost – 127.0.0.1
- **`/etc/resolv.conf`**
  - Domain Name Service (DNS) lookup
  - **`search`**: domains to search if not FQDN
  - **`nameserver`** (3): Nameservers to consult for name/IP resolution

Network Testing

- Localhost reachability
- Hostname reachability
- Local network reachability
- Internet network reachability
- DNS resolution

Network Tools

- **`ping`** – Reachability test
- **`traceroute`** – Routing performance
- **`netstat`** – Network performance stats
- **`tcpdump`** – Packet sniffing
- **`nslookup/dig`** – DNS Queries

SSH

- As a system administrator, you spend most of your time at the datacenter
- If you have the necessary tools, you don’t need to be at the datacenter physically
- You can log into any of the servers remotely from your personal computer
- One of the tools for remote login is **telnet**
  - not secured

SSH

- Secured SHell
- Connecting to internet increases vulnerability
  - Firewalls are not enough
  - telnet send username and password as simple text
  - ssh is a secured telnet which encrypts commands, usernames and passwords in a remote login
- OpenSSH is commonly used tool
- Others: putty, freeSSH, secureCRT

SSH

- To install SSH-client

  apt-get install ssh

- To install SSH-server

  apt-get install openssh-server

- To login into a remote machine (IP:10.5.192.30) with username Abebe onto

  ssh abebe@10.5.192.30

Secured Copy (scp)

- Copy file remotely from one machine to the other using ssh
- Syntax:
  - scp fileName username@IPaddressORhostname:DestinationLocation
- Example to copy the file **myMusic.mp3** from current working directory
  - ***scp myMusic.mp3 abebe@10.2.191.30:/home/abebe/music/***
- Using ***-r*** we can copy the whole directory recursively
- Read all the other options using ***man*** page.

- Reading assignments:
  - Windows network configuration
  - netplan network configuration
  - Secured FTP

Dynamic Host Configuration Protocol (DHCP)

- DHCP - centrally control IP-related information and eliminate the need to manually keep track of where individual IP addresses are allocated
- Two basic functions:
  - Provide a mechanism for assigning addresses to hosts
  - A method by which clients can request addresses and other configuration data from server
- In a DHCP-enabled host, a special message is sent out requesting an IP address and a subnet mask from a DHCP server
- DHCP server responds with information the client requests such as IP address, default gateways, NetBios
- DHCP provides static and dynamic address allocation that can be manual or automatic.

DHCP Operation

IP lease request

- First step in obtaining an IP address under DHCP
- It is initiated by a host with TCP/IP, configured to obtain an IP address automatically
- Since the requesting client is not aware of its own IP address, or that belonging to the DHCP server,
  - it will use 0.0.0.0 for client and 255.255.255.255 for DHCP server with UDP ports 67 (client) and 68 (server)
  - Message includes MAC address of client for the reply

  known as a DHCP discover

DHCP Operation

IP lease offer

- DHCP offer consist of an IP address, subnet mask, lease period (in seconds), and the IP address of the proposing DHCP server
- Offer sent to requesting MAC address
- The pending IP address offer is reserved temporarily to prevent it from being taken simultaneously by other machines

DHCP Operation

IP lease selection

- client machine selects the first IP address offer it receives.
- The client replies by broadcasting an acceptance message, requesting to lease IP information.
- Just as in stage one, this message will be broadcast as a DHCP request, but this time, it will additionally include the IP address of the DHCP server whose offer was accepted.
- All other DHCP servers will then revoke their offers

DHCP Operation

IP lease acknowledgment

- The accepted DHCP server proceeds to assign an IP address to the client, then sends:
  - DHCPACK – positive acknowledgment
  - DHCPNACK - negative acknowledgment
    - If the client is attempting to lease its old IP address, which has since been reassigned elsewhere.
    - The requesting client has an inaccurate IP address, resulting from physically changing locations to an alternate subnet
- Negative acceptance messages can also mean that the requesting client has an inaccurate IP address, resulting from physically changing locations to an alternate subnet.
- The client machine integrates the new IP information into its TCP/IP configuration.

DHCP Operation

Lease renewal:

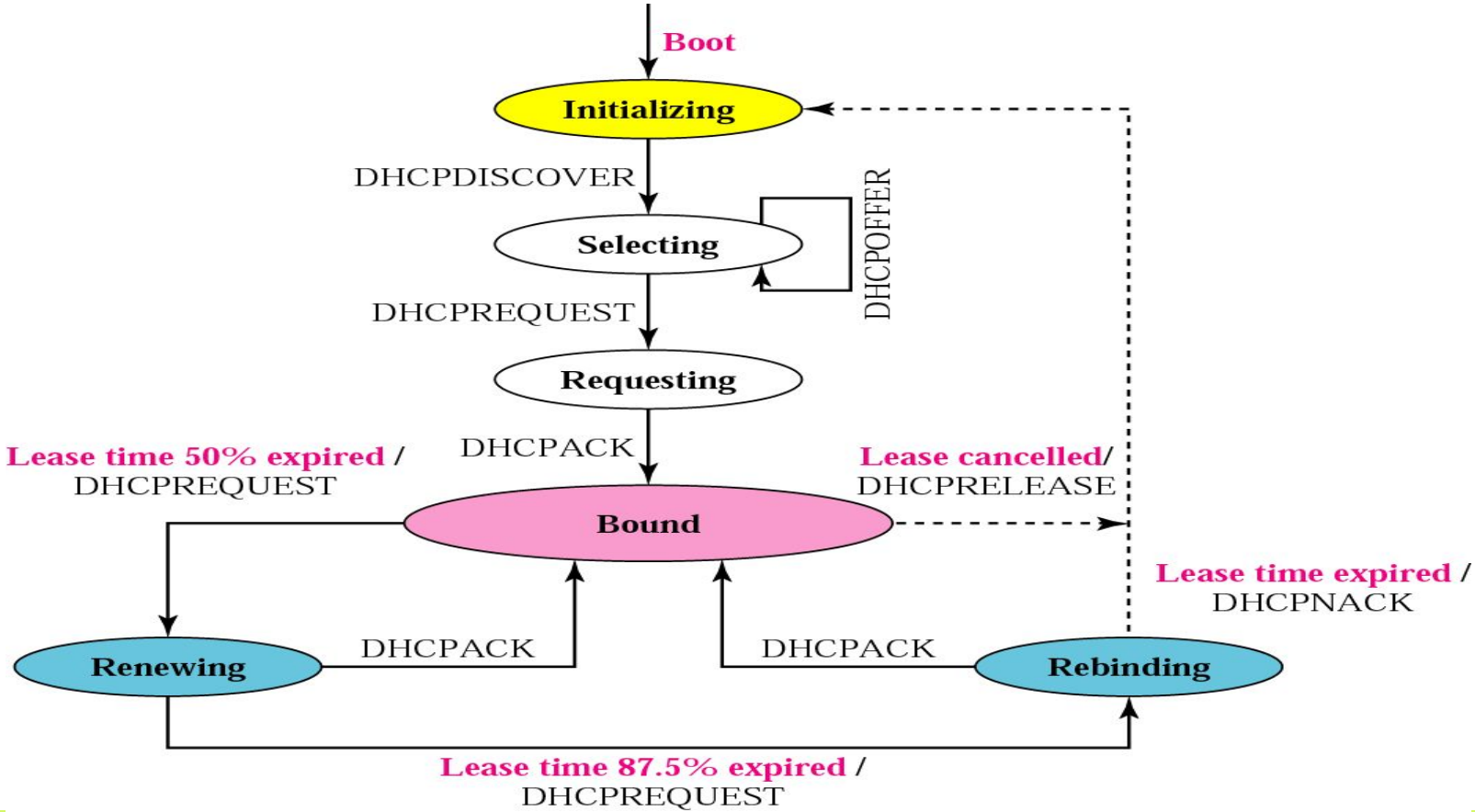
- The leasing client will send
  - DHCPREQUEST - to the DHCP server when its lease period has elapsed by 50%.
- If the DHCP server is available, and there are no reasons for rejecting the request, a DHCP acknowledge message is sent to the client, updating the configuration and resetting the lease time.
- If the server is unavailable, the client will receive an ‘eviction’ notice stating that it had not been renewed.
  - Client would still have a remaining 50% lease time to use the IP
  - React by sending out an additional lease renewal attempt when 87.5%
  - if DHCPACK is received, renew the lease.
- If the client received a DHCPNACK (negative) message, it would have to stop using the IP address immediately, and start the leasing process over, from the beginning.

DHCP Operation

Lease release

- If the client elects to cancel the lease, or is unable to contact the DHCP server before the lease elapses, the lease is automatically released.
- Note that DHCP leases are not automatically released at system shutdown.
- A system that has lost its lease will attempt to re-lease the same address that it had previously used.

DHCP Operation



DHCP Operation

How does DHCP work?

1. Server discovery
   - **Client** DHCPDISCOVER packet asking "Who can give me DHCP information?"

     Hello

2. Servers make an offer
   - **All servers** on the subnet unicast a DHCPOFFER packet saying "I can supply you with DHCP information, if you like." **What Do You need**

3. Client requests
   - **The client** selects one of the responses, and broadcasts a DHCPREQUEST packet saying "I choose server XYZ. Server XYZ, here's my MAC address, what's my IP address?" **Give Me An Address**

DHCP Operation

4. **Server** responds **Here It is and for How long**
   - The server responds with a DHCPACK packet saying "Here is your IP address. It's good for 24 hours."
   - The response can contain additional information, if the client asked for it.
   - The server records that the IP address is in use.

5. **Client** releases **You Can Have it Back**
   - The client finishes it's work, and send a DHCPRELEASE packet saying "I'm done with the IP address."
   - The server records that the IP address is not in use.

DHCP Client

- ***dhclient -*** DHCP client daemon, included with many popular Linux distributions,
  - is the software component used to talk to a DHCP server
- If invoked, it will attempt to obtain an address from an available DHCP server and then configure its networking configuration accordingly.
- Configuration of /etc/network/interfaces:

  auto eth0
  iface eth0 inet dhcp

Configuring the DHCP Client

- The client is typically run from the startup files, but it can also be run manually.
- It's typically started prior to other network-based services, since other network services are of no use unless the system itself can get on the network.
- On the other hand, the client can be invoked at the command line any time after startup.
- The command to invoke the client is: ***dhclient***

DHCP Client

```
[root@clientB ~]# dhclient
.....<OUTPUT TRUNCATED>.....
Sending on LPF/eth0/00:0c:29:f8:b8:88
Sending on Socket/fallback
DHCPDISCOVER on lo to 255.255.255.255 port 67 interval 7
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.1
SIOCADDRT: File exists
bound to 192.168.1.36 -- renewal in 138238 seconds.
```

DHCP Client

- Optionally, the client daemon can be started with additional flags that slightly modify the behavior of the software.
- For example, you can optionally specify the interface (such as eth0) for which an address lease should be requested.
  - ***dhclient eth0***
- For the full syntax of the command is use read the manual page: ***man dhclient***

DHCP Client

- In order to keep track of leases across system reboots and server restarts, dhclient keeps a list of leases it has been assigned in the dhclient.leases(5) file.
- On startup, after reading the dhclient.conf file, dhclient reads the dhclient.leases file to refresh its memory about what leases it has been assigned.

DHCP Client

- It is also possible to specify interfaces by name in the ***dhclient.conf*** file.
- If interfaces are specified in this way, then the client will only configure interfaces that are either specified in the configuration file or on the command line, and will ignore all other interfaces.
- On startup, dhclient reads the ***dhclient.conf*** for configuration instructions. It then gets a list of all the network interfaces that are configured in the current system.

DHCP Server

- The DHCP server, is responsible for serving IP addresses and other relevant information upon client request.
- Since the DHCP protocol is broadcast-based, a server will have to be present on each subnet for which DHCP service is to be provided.
- Installation on ubuntu: ***apt-get install isc-dhcp-server***
- The main configuration file is ***/etc/dhcp/dhcpd.conf***
- The configuration file consists of a set of global directives followed by one or more subnet definitions.
- Comments are prefixed with hash marks (#).

DHCP Server

- Like most configuration files in UNIX, the file is ASCII text and can be modified using your favorite text editor.
- The general structure of the configuration file is as follows:

```
Global parameters;
Declaration1
    [parameters related to declaration1]
    [nested sub declaration]
Declaration2
    [parameters related to declaration2]
    [nested sub declaration]
```

DHCP Server

- Global Settings of sample dhcpd.conf configuration

DHCP Server

- ***dns-update-style*** - specifies that our DHCP server will not do DNS updates for addresses that it hands out
- ***default-lease-time***: directive specifies how long a DHCP lease will be active if a connecting client does not specify a time.
- ***max-lease-time:*** specifies the maximum lease time allowed if the client does specify a time
- Both settings specify a time in seconds.
- ***log-facility:*** specifies how the system logger should handle log entries generated by the DHCP server

DHCP Server

option domain-name “aau.edu.et";

- This global setting specifies the domain name of the organization that name-servers are authoritive for

option domain-name-servers 10.5.5.15, 10.5.120.15

- This specifies list of DNS servers
- All those global settings are common to all clients acquiring IP from this server

DHCP Server

- To each subnet within the network, the default gateway and IP address range should be specified.

```
subnet 10.5.120.0 netmask 255.255.255.0
{
    option range 10.5.120.5 10.5.120.250;
    option router 10.5.120.1
}
```

Linux Firewall

Linux Firewall: iptables

- What is firewall?
- A system that sits between two networks:
  - Used to protect one from the other
  - Places a bottleneck between the networks
  - All communications must pass through the bottleneck this gives us a single point of control

Linux Firewall: iptables

- **How does it work?**
  - Packet Filtering
    - Rejects TCP/IP packets from unauthorized hosts and/or connection attempts by unauthorized hosts
  - Network Address Translation (NAT)
    - Translates the addresses of internal hosts so as to hide them from the outside world
    - Also known as IP masquerading
  - Proxy Services
    - Makes high level application level connections to external hosts on behalf of internal hosts to completely break the network connection between internal and external hosts

Linux Firewall: iptables

- **Other services:**
  - Encrypted Authentication
    - Allows users on the external network to authenticate to the Firewall to gain access to the private network
    - Virtual Private Networking
    - Establishes a secure connection between two private networks over a public network
      - This allows the use of the Internet as a connection medium rather than the use of an expensive leased line

Linux Firewall: iptables

- **Other services:**
  - Virus Scanning
    - Searches incoming data streams for virus signatures so they may be blocked
    - Done by subscription to stay current
      - McAfee / Norton
  - Content Filtering
    - Allows the blocking of internal users from certain types of content.
      - Usually an add-on to a proxy server
      - Usually a separate subscription service as it is too hard and time consuming to keep current

Linux Firewall: iptables

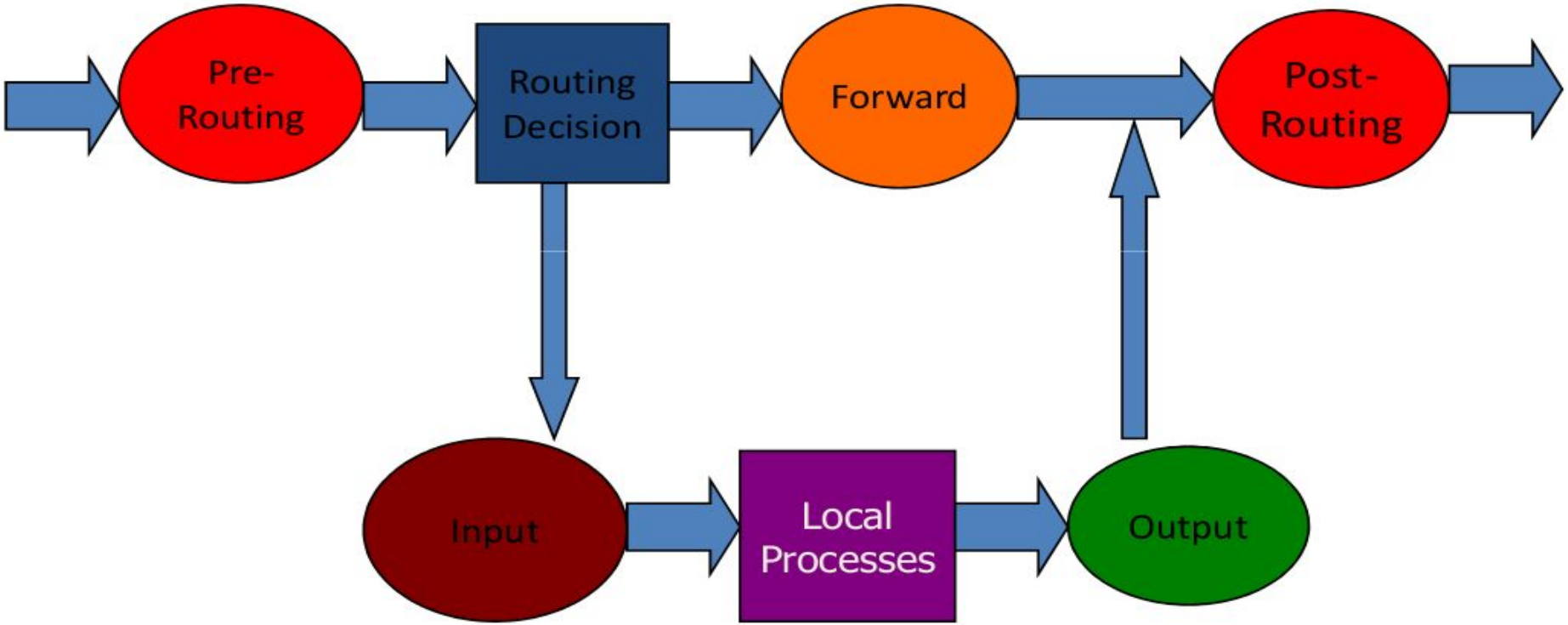
- Iptables is a Linux firewall that also is capable of doing NAT
- Consists of a set of rules
- Rules are normally in a config- script and are written as Iptables-commands.
- The two most important tables in Iptables are FILTER and NAT

Linux Firewall: iptables

Filter:

- Consists of chains:
  - INPUT
  - FORWARD
  - OUTPUT
- If the rules in the chain matches, decision will be:
  - DROP – packet will be dropped and don't send error message
  - ACCEPT – let the packet through
  - REJECT – sends ICMP error message
  - MASQ – masquerade
  - RETURN – end of chain; stop traversing this chain and resume the calling chain
  - QUEUE – pass the packet to the user space

Linux Firewall: iptables



Linux Firewall: iptables

- **INPUT:**
  - INPUT deals with all packets received and that have the machine that runs iptables as destination.
  - This means that only packets that are meant for the machine that runs iptables will be processed by this chain

- **FOWARD:**
  - Deals with the packets that are incoming to the machine that runs iptables, but are meant to be forwarded to other machines.
  - They can be forwarded to a machine on the local network or to a machine on an external network.

- **OUTPUT:**
  - Deals with packets that has their origin in the machine that runs iptables and are going out to another machine.
  - Packets coming from the local net and going out, will not be processed in this chain but in the FORWARD chain.

Syntax of iptables command

- **iptables –t TABLE –A CHAIN –[i|o] IFACE –s w.x.y.z –d a.b.c.d –p PROT –m state --state STATE –j ACTION**
  - TABLE = nat | filter | mangle
  - CHAIN = INPUT | OUTPUT | FORWARD |PREROUTING| POSTROUTING
  - IFACE = eth0 | eth1 | ppp0 | ...
  - PROT = tcp | icmp | udp | ...
  - STATE = NEW | ESTABLISHED | RELATED | ..
  - ACTION = DROP | ACCEPT | REJECT

Specifying IP addresses

- Source: -s, --source or –src
- Destination: -d, --destination or –dst
- IP address can be specified in four ways.
  - (Fully qualified) host name (e.g., site, site.aau.edu.et)
  - IP address (e.g., 127.0.0.1)
  - Group specification (e.g., 130.108.27.0/24)
  - Group specification (e.g., 130.108.27.0/255.255.255.0)
- ‘–s ! IPaddress’ and ‘–d ! IPaddress’: Match address not equal to the given

Specifying Protocol

- -p protocol
- Protocol number
  - 17-UDP
  - 6-TCP
  - 1-ICMP
- Protocol can be a name
  - TCP
  - UDP
  - ICMP
- -p ! protocol

Iptables examples

1. iptables --flush
   Delete all rules
2. iptables -A INPUT -i lo -j ACCEPT
   Accept all packets arriving on lo for local processes
3. iptables -A OUTPUT -o lo -j ACCEPT
4. iptables --policy INPUT DROP
   Unless other rules apply, drop all INPUT packets
5. iptables --policy OUTPUT DROP
6. iptables --policy FORWARD DROP
7. iptables -L -v -n
   List all rules, verbosely, using numeric IP addresses etc.

The Log Target

- LOG
  - --log-level
  - --log-prefix
  - --log-tcp-sequence
  - --log-tcp-options
  - --log-ip-options

1. iptables -A OUTPUT -o eth0 -j LOG

   Jump the packets that are on OUTPUT chain intending to leave from eth0 interface to LOG

2. iptables -A INPUT -m state --state INVALID -j LOG --log-prefix “INVALID input: ”

   Jump the packets that are on INPUT chain with an INVALID state to to LOG and have the logged text begin with “INVALID input: ”

iptables syntax examples

- iptables -A INPUT -i eth1 -p tcp -s 192.168.17.1 --sport 1024:65535 -d 192.168.17.2 --dport 22 -j ACCEPT
  - Accept all TCP packets arriving on eth1 for local processes from 192.168.17.1 with any source port higher than 1023 to 192.168.17.2 and destination port 22.
- iptables –A INPUT –p tcp –s 0/0 –d 0/0 –dport 0:1023 –j REJECT
  - *Reject all incoming TCP traffic destined for ports 0 to 1023*
- iptables –A OUTPUT –p tcp –s 0/0 –d ! comp –j REJECT
  - Reject all outgoing TCP traffic except the one destined for comp
- iptables –A INPUT –p TCP –s osis110 --syn –j DROP
  - Drop all SYN packets from host osis11

Operations on chains

- Operations to manage whole chains
  - N: create a new chain
  - P: change the policy of built-in chain
  - L:list the rules in a chain
  - F:flush the rules out of a chain
- Manipulate rules inside a chain
  - A: append a new rule to a chain
  - I: insert a new rule at some position in a chain
  - R: Replace a rule at some position in a chain
  - D: delete a rule in a chain

Linux Proxy

Linux Proxy

What is proxy?

- Firewall device; internal users communicate with the proxy, which in turn talks to the big bad Internet
  - Get private address space (RFC 1918) into publicly routable address space
- Allows one to implement policy
  - Restrict who can access the Internet
  - Restrict what sites users can access
  - Provides detailed logs of user activity

What is a caching proxy?

- Stores a local copy of objects fetched
  - Subsequent accesses by other users in the organization are served from the local cache, rather than the origin server
  - Reduces network bandwidth
  - Users experience faster web access

How do proxies work?

- User configures web browser to use proxy instead of connecting directly to origin servers
  - Manual configuration for older PC based browsers, and many UNIX browsers (e.g., Lynx)
  - Proxy auto-configuration file for Mozilla Firefox or Internet Explorer
    - Far more flexible caching policy
    - Simplifies user configuration, help desk support, etc.

How do proxies work?

- User requests a page: *http://www.google.com/*
- Browser forwards request to proxy
- Proxy optionally verifies user's identity and checks policy for right to access *www.google.com*
- Assuming right is granted, fetches page and returns it to user

What is Squid?

- A caching proxy for
  - HTTP, HTTPS (tunnel only)
  - FTP
  - WAIS (requires additional software)
  - WHOIS (Squid version 2 only)
- Supports transparent proxying
- Supports proxy hierarchies (ICP protocol)
- Squid is not an origin server!

Squid's page fetch algorithm

- Check cache for existing copy of object (lookup based on MD5 hash of URL)
- If it exists in cache
  - Check object's expire time; if expired, fall back to origin server
  - Check object's refresh rule; if expired, perform an **If-Modified-Since** against origin server
  - If object still considered fresh, return cached object to requester

Squid's page fetch algorithm

- If object is not in cache, expired, or otherwise invalidated
  - Fetch object from origin server
  - If 500 error from origin server, and expired object available, returns expired object
  - Test object for cacheability; if cacheable, store local copy

Cacheable objects

- HTTP
  - Must have a **Last-Modified**: tag
  - If origin server required HTTP authentication for request, must have **Cache-Control**: public tag
  - Ideally also has an **Expires** or **Cache-Control**: max-age tag
  - Content provider decides what header tags to include
    - Web servers can auto-generate some tags, such as **Last-Modified** and **Content-Length**, under certain conditions
- FTP
  - Squid sets Expires time to fetch timestamp + 2 days

Non-cacheable objects

- HTTPS
- HTTP
  - No Last-Modified: tag
  - Authenticated objects
  - Cache-Control: private, no-cache, and no-store tags
  - URLs with cgi-bin or ? in them
  - POST method (form submission)

Implication for content providers

- Caching is a good thing for you!
- Make cgi and other dynamic content generators return **Last-Modified** and **Expires/Cache-Control** tags whenever possible
  - If at all possible, also include a **Content-Length** tag to enable use of persistent connections
- Consider using **Cache-Control**: public, must-revalidate for authenticated web sites

Implication for content providers

- If you need a page hit counter, make one small object on the page non-cacheable.
- FTP sites, due to lack of **Last-Modified** timestamps, are inherently non-cacheable. Put (large) downloads on your web site instead of on, or in addition to, an FTP site.

Implication for content providers

- Microsoft's IIS with ASP generates non-cacheable pages by default
- Other scripting suites (e.g., Cold Fusion) also require special work to make cacheable
- Squid doesn't implement support for **Vary**: tag yet; considers object non-cacheable
- Squid currently treats **Cache-Control**: must-revalidate as **Cache-Control**: private

Transparent proxying

- Router forwards all traffic to port 80 to proxy machine using a route policy
- Pros
  - Requires no explicit proxy configuration in the user's browser

Transparent proxying

- Cons
  - Route policies put excessive CPU load on routers on many (Cisco) platforms
  - Kernel hacks to support it on the proxy machine are still unstable
  - Often leads to mysterious page retrieval failures
  - Only proxies HTTP traffic on port 80; not FTP or HTTP on other ports
  - No redundancy in case of failure of the proxy

Squid hardware requirements

- UNIX operating system
- 128M RAM minimum recommended (scales by user count and size of disk cache)
- Disk
  - 512M to 1G for small user counts
  - 16G to 24G for large user counts
  - Squid 3.x is latest

File system recommendations

- Use Veritas' vxfs if you have it
- Disable last accessed time updates (for example, noatime mount option on Linux)
- Consider increasing sync frequency
- If using UFS (Unix File System)
  - Optimize for space instead of time

Installing Squid

- Get distribution from http://www.squid.org/
- Ubuntu/Debian ***apt-get install squid***
- Edit ***squid.conf*** file
- Run Squid -z to initialize cache directory structure
- Start Squid daemon
- Test
- Migrate users over to proxy

squid.conf runtime settings

- Default squid.conf file is heavily commented! Read it!
- Must set
  - cache_dir (one per disk)
  - cache_peer (one per peer) if participating in a hierarchy
  - cache_mem (8-16M preferred, even for large caches)
  - acl rules (default rules mostly work, but must reflect your address space)

squid.conf runtime settings

- Recommendations
  - ipcache_size, fqdncache_size to 4096
  - log_fqdn off (use Apache's logresolve offline)
  - Increase dns_children, redirect_children, authenticate_children based on usage statistics (see cachemgr.cgi front-end)
  - Tweak refresh_pattern rules (Danger Will Robinson! -- I suggest starting with examples found in the squid mailing list archives)

squid.conf runtime settings

- Recommendations (continued)
  - quick_abort_min 128 KB, quick_abort_max 4096 KB, quick_abort_pct 75
    - Tailor based on your bandwidth to the Internet
    - By default, squid will complete retrieval of any object requested, regardless of size; can burn considerable amounts of bandwidth!
- Too many other options in squid.conf to cover here; you really should read all the embedded comments!

squid.conf ACL example

- acl manager proto cache_object
- acl localhost src 127.0.0.1/32
- acl managerhost src 204.248.51.34/32
- acl managerhost src 204.248.51.39/32
- acl managerhost src 204.248.51.40/32
- acl officernet src 204.248.51.0/24
- acl officernet-internal src 172.16.0.0/16
- acl all src 0.0.0.0/0.0.0.0

squid.conf ACL example

- acl SSL_ports port 443 563
- acl gopher_ports port 70
- acl wais_ports port 210
- acl whois_ports port 43
- acl www_ports port 80 81
- acl ftp_ports port 21
- acl Safe_ports port 1025-65535
- acl CONNECT method CONNECT
- acl FTP proto FTP
- acl HTTP proto HTTP
- acl WAIS proto WAIS
- acl GOPHER proto GOPHER
- acl WHOIS proto WHOIS

squid.conf ACL example

- http_access deny manager !localhost !managerhost
- http_access deny CONNECT !SSL_ports
- http_access deny HTTP !www_ports !Safe_ports
- http_access deny FTP !ftp_ports !Safe_ports
- http_access deny GOPHER !gopher_ports !Safe_ports
- http_access deny WAIS !wais_ports !Safe_ports
- http_access deny WHOIS !whois_ports !Safe_ports
- http_access allow localhost
- http_access allow officernet
- http_access allow officernet-internal
- http_access deny all

Group Presentation

- Delay pools
- Dansguardian
- MRTG
- LDAP

Access Control List

- Allow to control access based on:
  - Source/Destination IP address
  - Source/Destination domain
  - Protocol
  - Port number
  - Method
- Syntax:
  - ***acl name type value***

Access Control List

- Ip address: 10.20.30.0/24 , 10.2.3.4
- Domain: .google.com, .yahoo.com, .edu.et, .gov.uk,
- Port number: 80, 1024 – 6455, 20 30 40,
- Method: GET, POST, CONNECT
- Protocol: HTTP, FTP, SMTP, HTTPS

Access Control List

- acl aaitNet src 10.1.0.0/16
- acl mcNet src 10.2.0.0/16
- acl presidentNet src 10.3.0.0/16
- acl myPc src 10.1.20.220/32
- acl aaitNet dst 10.1.0.0/16
- acl mcNet dst 10.2.0.0/16
- acl presidentNet dst 10.3.0.0/16
- acl myPc dst 10.1.20.220/32
- acl all src 0.0.0.0/0.0.0.0

Access Control List

- acl safe_sites dstdomain **.google.com**
- acl safe_sites dstdomain **.yahoo.com**
- acl bad_sites dstdomain **.sex.com**
- acl bad_sites dstdomain **.useless.com**

Access Control List

- acl safePorts port 1025-65535
- acl SSL_ports port 443 563
- acl gopher_ports port 70
- acl wais_ports port 210
- acl whois_ports port 43
- acl www_ports port 80 81
- acl ftp_ports port 20 21

Access Control List

- acl conn_method method **CONNECT**
- acl post_method method **POST**
- acl get_method method **GET**
- acl safe_protocols proto **http**
- acl safe_protocols proto **https**
- acl safe_protocols proto **ftp**
- acl safe_protocols proto **smtp**

http_access

- Access operator where you allow/deny internet access that matches a particular acl rule
- If multiple acl names are provided, it uses ***and*** logical operator
- Syntax: http_access deny/allow aclName aclName aclName ....

http_access

- http_access deny bad_sites
- http_access deny !safe_ports
- http_access allow mcNet aaitNet safe_ports
- http_access allow presidentNet safe_ports
- http_access deny HTTP !www_ports !
- Safe_ports

Thank you.

Getnet M.
getnet.mamo@aau.edu.et