APCC Computing System

Sang-Cheol Kim

System Analyst

Science Division, APCC

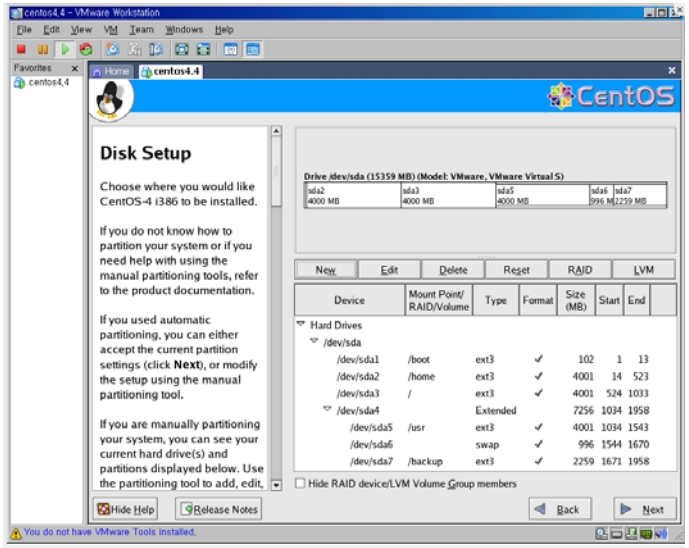
Contents:

Linux

LInux is :

Linux is a free open-source operating system based on Unix. Linux was originally created by Linux Torvalds with the assistance of developers from around the globe.

A freeware version of Unix, Linux is becoming popular as a powerful, low-cost operating system for running servers.



Set partition :

Linux partition set is more important. According to Linux server use purpose, must divide partition.

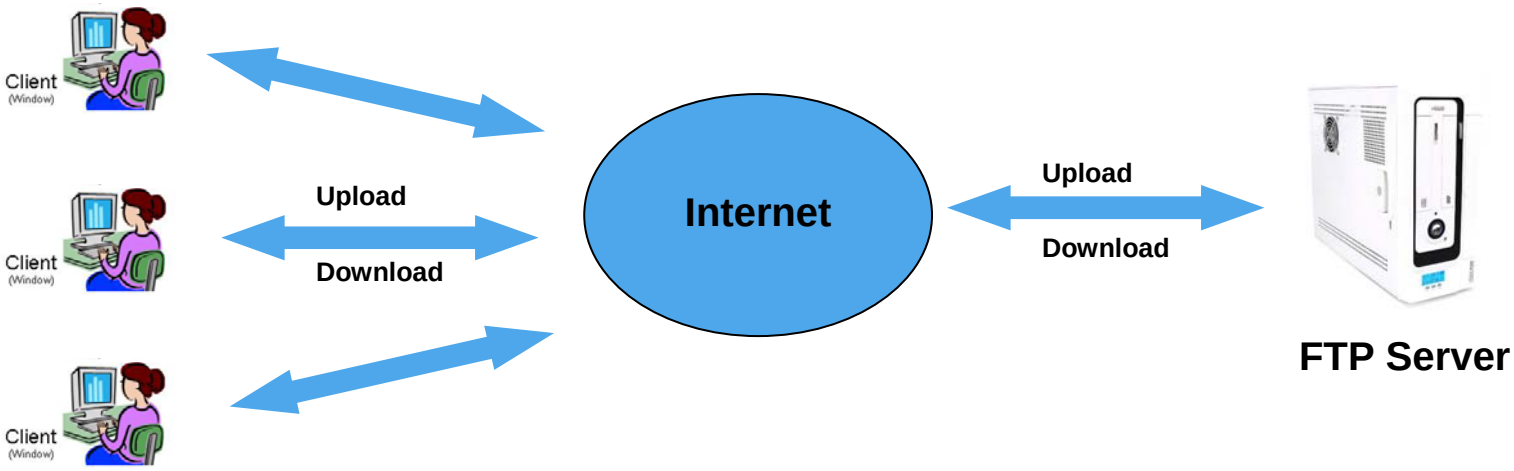
Basically, /boot, swap, / by three divide. But , have better divide by a serveral partition according to necessary use purpose.

FTP Server:

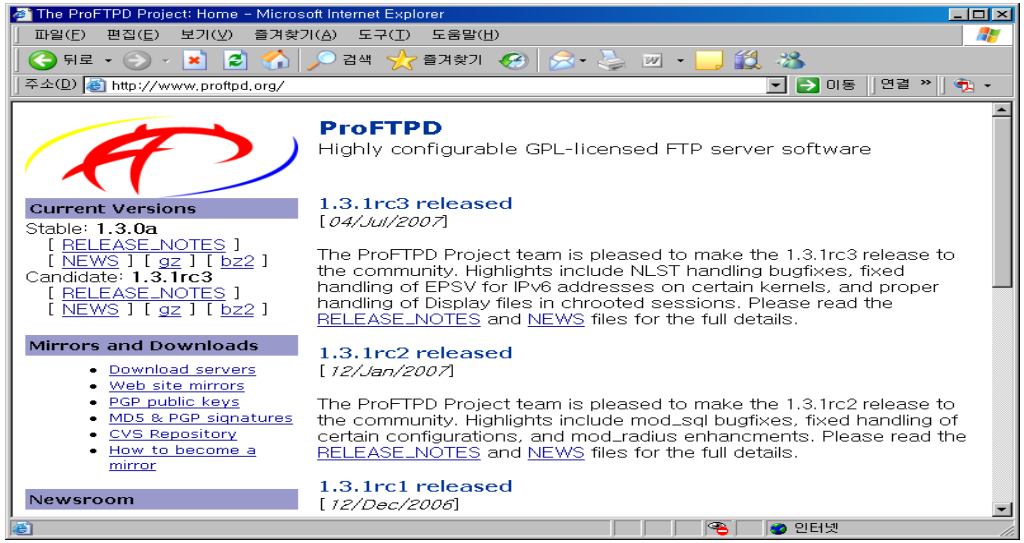
FTP Server is :

Short for **File Transfer Protocol**, the protocol for exchange files over internet.

FTP uses the internet's TCP/IP protocols to enable data transfer



FTP Server – install guide (ProFTPD):



Get source code :

- **ProFTPD website**
- **use wget util**

```
[root@test local]# wget ftp://ftp.proftpd.org/distrib/source/proftpd-1.3.0a.tar.gz
--10:46:39--  ftp://ftp.proftpd.org/distrib/source/proftpd-1.3.0a.tar.gz
           => `proftpd-1.3.0a.tar.gz'
Resolving ftp.proftpd.org... 81.223.20.36
Connecting to ftp.proftpd.org|81.223.20.36|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /distrib/source ... done.
==> PASV ... done.    ==> RETR proftpd-1.3.0a.tar.gz ... done.
Length: 1,858,160 (1.8M) (unauthoritative)

100%[=====================================================================================>] 1,858,160    306.75K/s    ETA 00:

10:46:50 (291.38 KB/s) - `proftpd-1.3.0a.tar.gz' saved [1858160]

[root@test local]#
```

FTP Server – install guide (ProFTPD) :

```
[root@test local]# tar xvfz proftpd-1.3.0a.tar.gz
proftpd-1.3.0a/
proftpd-1.3.0a/contrib/
proftpd-1.3.0a/contrib/dist/
proftpd-1.3.0a/contrib/dist/rpm/
proftpd-1.3.0a/contrib/dist/rpm/ftp.pamd
```

decompress using tar

```
[root@test proftpd-1.3.0a]# ./configure --prefix=/usr/local/proftpd --enable-autoshadow --enable-shadow
```

configure

option	Explain
--prefix=PREFIX	Install architecture-independent files in PREFIX
--enable-autoshadow	Enable run-time auto-detection of shadowed passwords (requires shadow)
--enable-shadow	Force compilation of shadowed password support

Configure option

FTP Server – install guide (ProFTPD) :

```
[root@test proftpd-1.3.0a]# make
```

make – compile source code

```
[root@test proftpd-1.3.0a]# make install
```

make install – file copy, set permission and right

► **Check the directory**

- **/usr/local/proftpd : proftpd program home directory**
- **/usr/local/proftpd/etc : exist configure file**
- **/usr/local/proftpd/sbin : execution file**

FTP Server – install guide (ProFTPD) :

► **set proftpd.conf file**

```
# This is a basic ProFTPD configuration file (rename it to
# 'proftpd.conf' for actual use.  It establishes a single server
# and a single anonymous login.  It assumes that you have a user/group
# "nobody" and "ftp" for normal operation and anon.

ServerName                      "ProFTPD Default Installation"
ServerType                      standalone
DefaultServer                   on

# Port 21 is the standard FTP port.
Port                            21

# Umask 022 is a good standard umask to prevent new dirs and files
# from being group and world writable.
Umask                           022

# To prevent DoS attacks, set the maximum number of child processes
# to 30.  If you need to allow more than 30 concurrent connections
# at once, simply increase this value.  Note that this ONLY works
# in standalone mode, in inetd mode you should use an inetd server
# that allows you to limit maximum number of processes per service
# (such as xinetd).
MaxInstances                    30

# Set the user and group under which the server will run.
User                            nobody
Group                           nogroup

# To cause every FTP user to be "jailed" (chrooted) into their home
# directory, uncomment this line.
#DefaultRoot ~

# Normally, we want files to be overwriteable.
AllowOverwrite          on

# Bar use of SITE CHMOD by default
<Limit SITE_CHMOD>
  DenyAll
</Limit>

# A basic anonymous configuration, no upload directories.  If you do not
# want anonymous users, simply delete this entire <Anonymous> section.
<Anonymous ~ftp>
  User                          ftp
  Group                         ftp

  # We want clients to be able to login with "anonymous" as well as "ftp"
  UserAlias                     anonymous ftp

  # Limit the maximum number of anonymous logins
  MaxClients                    10

  # We want 'welcome.msg' displayed at login, and '.message' displayed
  # in each newly chdired directory.
  DisplayLogin                  welcome.msg
  DisplayFirstChdir             .message

  # Limit WRITE everywhere in the anonymous chroot
  <Limit WRITE>
"proftpd.conf" 62L, 1862C
```

Item	comment
ServerName	FTP server name set
ServerType	standalone / inetd
Port	Set use port (Default : 21)
MaxInstances	Number of child process
User, Group	Set user and group
RootLogin	Whether or not root connect ftp server
DisplayLogin	Set message. FTP connection sucess.
<Directory> ~ </Directory>	Set individual directory
<Anonymous ~ftp> ~ </Anonymous>	Set anonymous ftp
<Limit> ~ </Limit>	Set limitation of ftp command
MaxLoginAttempts	Max number of connect times

FTP Server – install guide (ProFTPD) :

► **Server Type : standalone / inetd**

- **xinetd use**

```
[root@test etc]# cat /etc/xinetd.d/proftpd
service ftp
{
        disable             = no
        flags               = REUSE
        protocol            = tcp
        socket_type         = stream
        instances           = 50
        wait                = no
        user                = root
        server              = /usr/local/proftpd/sbin/in.proftpd
        log_on_sucess       = HOST PID
        log_on_failure      = HOST RECORD
}
[root@test etc]#
```

```
[root@test etc]# /etc/rc.d/init.d/xinetd restart
Stopping xinetd:                                           [  OK  ]
Starting xinetd:                                           [  OK  ]
[root@test etc]#
```

- **standalone use**

```
[root@test etc]#
[root@test etc]# /usr/local/proftpd/sbin/proftpd
```

FTP user command guide :

► Command on used FTP server

Command	comment	Command	comment
ascii	Transmission mode set ascii mode	mdelete	Delete several files (ex : mdelete *.html)
binary	Transmission mode set binary mode	mget	Get several files
bye	ftp connection close and quit	mput	Upload server files on remote system
cd	Change directory on remote system (ex : cd directory-name)	open	Try ftp connect (ex : open xxx.xxx.xxx.xxx)
cdup	Move up directory on remote system	put	Upload file on remote system
chmod	Change file permission on remote system (ex : chmod 755 index.html)	pwd	Display now directory on remote system
close	ftp connection close	quit	ftp connection close and quit
delete	Delete file on remote system (ex : delete index_old.html)	type	Set transmission mode (ex : type ascii or type binary)
dir	Display directory list on remote system	rmdir	Delete directory on remote system
disconnect	ftp connection close	size	Show file size by byte on remote system
exit	ftp connection close and quit	status	Show ftp session mode now connected
get	Get file (ex : get index.hmtl)	restatus	Show status on remote system
help	Can see command help	rename	Change file name on remote system
lcd	Change directory on local system	ls	Display directory list on remote system

SSH

SSH is :

Secure Shell - A way to access another machine. Data is sent encrypted between the machine making it hard to grab information like passwords. And traffic can receive faster transfer efficiency being compressed. Port (22)

Why use SSH :

- Strong security
- Privacy protection, All communications are encoded by automatic and clearly
- Safe X11 session, establish DISPLAY variable automatically to remote server and all X11 connections do forwarding through security channel
- TCP/IP port can do forwarding freely to other port from both directions.
- Alternate perfectly rlogin, rsh, rcp etc

SSH install guide

SSH information :

- sshd server daemon : /usr/sbin/sshd
- ssh client : /usr/bin/ssh
- ssh files(directory) relation service : /etc/ssh

SSH download :

- http://www.ssh.com

SSH install :

- download software
- decompress file (tar , gzip)
- configure
- make (source compile)
- make install (file copy, set permission and right)
- run sshd daemon

SSH :

```
[root@apccweb ~]# ssh -l root 190.1.1.234
The authenticity of host '190.1.1.234 (190.1.1.234)' can't be established.
RSA key fingerprint is 8b:15:ba:10:57:f1:c7:0b:40:0b:3f:d4:f3:8f:0b:33.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '190.1.1.234' (RSA) to the list of known hosts.
root@190.1.1.234's password:
Last login: Fri Jul 27 16:56:10 2007 from
[root@test ~]#
```

Example : How to connect using ssh

```
[root@test ssh]# cd /etc/ssh
[root@test ssh]# ll
total 160
-rw-------  1 root root 111892 May  2  2006 moduli
-rw-r--r--  1 root root   1417 May  2  2006 ssh_config
-rw-------  1 root root   3025 May  2  2006 sshd_config
-rw-------  1 root root    668 Apr 10 16:39 ssh_host_dsa_key
-rw-r--r--  1 root root    590 Apr 10 16:39 ssh_host_dsa_key.pub
-rw-------  1 root root    515 Apr 10 16:39 ssh_host_key
-rw-r--r--  1 root root    319 Apr 10 16:39 ssh_host_key.pub
-rw-------  1 root root    883 Apr 10 16:39 ssh_host_rsa_key
-rw-r--r--  1 root root    210 Apr 10 16:39 ssh_host_rsa_key.pub
[root@test ssh]#
```

ssh configuration file

sshd_config

- ssh server configuration file

ssh_config

- ssh client configuration file

ssh_host_dsa_key, ssh_host_key, ssh_host_rsa_key

- These three files contain the private parts of the host keys

SSH :

```
#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6

#RSAAuthentication yes
#PubkeyAuthentication yes
#AuthorizedKeysFile     .ssh/authorized_keys

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#RhostsRSAAuthentication no
# similar for protocol version 2
#HostbasedAuthentication no
                                                          35,0-1        35%
```

PermitRootLogin

- **yes : root connect possible**
- **no : root connect impossible**
- **line delete : root connect impossible**

/etc/ssh/sshd_config : server configuration file

TCP Wrapper :

TCP Wrapper is :

TCP Wrapper is a host-based network ACL system, used to filter network access to internet protocol services run on operation system such as linux or BSD. It allows host or subnetwork IP addresses, names and / or ident query replies, to be used as tokens on which to filter to for access control purposes.

TCP Wrapper consist of :

- daemon process : /usr/sbin/inetd
- configuration file : /etc/inetd.conf
- etc use file : /etc/rc.d/init.d/inet, /etc/hosts.allow, /etc/hosts.deny

TCP Wrapper :

TCP Wrapper use or reference file :

- inetd : kerenl version 2.2
  : /etc/inetd.conf file reference
    - filed of inetd.conf file
      service name, socket type, protocol, flag, user, server program, parameter
  : start / stop
    - /etc/rc.d/init.d/inet start or /etc/rc.d/init.d/inet stop
- xined : kernel version 2.4
  : start / stop
    - /etc/rc.d/init.d/xinetd start or /etc/rc.d/init.d/xinted stop

TCP Wrapper :

TCP Wrapper control host access method :

I. TCP Wrapper whole service stop

- /etc/rc.d/init.d/inet stop (kernel 2.2) or /etc/rc.d/init.d/xinetd stop (kernel 2.4)

II. Stop of selective service

- example

telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd

➔ #telnet stream tcp nowail root /usr/sbin/tcpd in.telnetd

(stop service)

III. Host access control

- using /etc/hosts.allow and /etc/hosts.deny
- /etc/hosts.allow : Record list of hosts that access is admited.
- /etc/hosts.deny : Record list of hosts that access is denied.

TCP Wrapper :

TCP Wrapper control host access method :

-/usr/sbin/tcpd : access control facility for internet services. The tcpd program can be set up to monitor incoming requests for telnet, finger, ftp, exec, rsh, rlogin, tftp, talk, comsat and other services that have a one-to-one mapping onto executable files.

- tcpd read two files (/etc/hosts.allow and /etc/hosts.deny). Then provide services specification client by result that read files.

- tcpd read and apply order

I. hosts.allow

II. hosts.deny

example

ALL : ALL (allow all)

ALL : localhost, .aaa.com (allow all of localhost and aaa.com hosts)

in.telnetd : 192.168.0.2 (allow telnet connection in 192.168.0.2)

TCP Wrapper :

→ hosts.allow and hosts.deny wildcards

Wildcards	Comment
ALL	All services or All hosts
LOCAL	Matches any host whose name does not contain a dot character
KNOWN	Matches any user whose name is known, and matches any host whose name and address are known. This pattern should be used with care:host names may be unavailable due to temporary name server problems. A network address will be unavailable when the software cannot figure out what type of network it is talking to.
UNKNOWN	Matches any user whose name is unknown, and matches any host whose name or address are unknown. This pattern should be used with care : host names may be unavailable due to temporary name server problems. A network address will be unavailable when the software cannot figure out what type of network it is talking to.
PARANOID	Matches any host whose name does not match its address. When tcpd is built with-DPARAMOID (default mode), it drops requests from such clients even before looking at the access control tables. Build without-DPARANOID when you you want more control over such requests.
B EXCEPT A	Intended use is of the form: ‘list_1 EXECPT list2 ? This construct matches list_! Unless it matches list2. The EXCEPT operator can be used int daemon_lists and in client_lists. The EXCEPT operator can be nested: if the control language would permit the use of parentheses, ‘ a EXCEPT b EXCEPT c? would parse as ‘(a EXCEPT (b EXCEPT c))?