

Linux

# Linux Facts

- Linux is not an OS
- Its a Kernel
- There are OS'es based on linux kernel
- Even android is based on linux kernel
- Android is Malware base of Linux
  - Google succeeded in getting malware to linux.
- Another mobile OS: Firefox OS

# The Linux FileSystem

- Minix File System

- First linux file system
- File size limit of 64 Mb
- File name limit of 14 char

- Ext (Extended File System)

- Solved the 2 problems
- Allowed 2 GB data, & name upto 255 chars
- Had modification problems

# Linux FS continued

## •Ext2

- Solved the previous problems
- Max volume size support of 2TB
- But no journalling

## •Ext3

- Journalled file system
- Takes less CPU power
- Adds file system growth & indexing

# Journalling

- Keeps track of the changes that will be made before committing it to the main File System.
- In event of crash and failures, its easier to get it back online.
- Files less likely to get corrupted.
- Very simple
- Wider Testing base.

# Linux FS Continued

## •Ext4

- Backward compatible to ext3
- Extends storage limits
- Performance Improvements
- Supports volumes upto 1 Exbibyte = 1 million TeraBytes
- File sizes upto 1TB
- New block allocation algorithm

# More File Systems

## •FAT

- Used in floppys and USBs
- Prominent usage of an index table
- Each entry contains the number of the next cluster in the file, or a marker indicating end of file.

## •NTFS

- Supersedes FAT
- Better performance, ACL & journalling



# Boot Process

- The PC's power supply brings all the required voltages for the Motherboard and peripheral components to acceptable levels
  - + & - 12V lines, + and - 5.00V, and likely + & - 3.30V
- Once a good voltage is present, the Motherboard will turn on and fans will start spinning (can take half a second)
- The motherboard's clock pulses begin synchronizing all interactions of the peripherals
- RAM will be clear\*



# Boot Process

- *Non-volatile ram may not be clear*
- The processor points to the start of the BIOS boot program (usually 0xFFFF0), right at the end of the system memory.
  - this usually is just a jump instruction
- The BIOS performs the power-on self test (POST), and if there are any fatal errors, the boot processor stops.

# Boot Process

- The BIOS then looks for the video card (specifically the video card's own BIOS code which is usually at 0xC000 (C-thousand))
  - If any other BIOSes are detected (secondary, etc) they are executed as well...
  - what could go wrong? :)
- Video cards can have non-volatile ram
- Rootkits \*can\* hide here (very rare)
- The BIOS then looks for any other devices ROM to see if they have BIOSes as well..
  - IDE/ATA hard disk BIOSes are usually at 0xC8000
  - Infecting these BIOS requires supply chain attack

# Boot Process

- The BIOS then displays its startup screen
- BIOS tests the system memory (RAM count)
- BIOS then does hardware probing to detect what sorts of hardware is plugged in
- BIOS then will detect & configure Plug & Play
- BIOS then displays a summary screen, and then proceeds to look for a drive to boot from
- BIOS looks for main boot record (MBR) to start the OS boot process.

– There are MBR viruses

– If it is on a HD then it looks for master boot record at cylinder 0.

# Boot Sequence

(Broadly Speaking)

- BIOS Initialization
- Boot Loader(Read boot loader from mbr)
- Kernel Initialization
- init (execution of system startup scripts)
  - /etc/rc.d/rc.sysinit
  - /etc/rc.d/rc
  - /etc/rc.d/rc\*.d/
  - /etc/rc.d/rc.local

# Installation

- Requirements:

- PenDrive min 8 GB (for Live USB installation)
- PenDrive min 32 GB (for full installation)
- VirtualBox
- ISO file
- USB Installer Application

# Package Management

- apt-get update
- apt-get upgrade
- apt-get autoremove
- apt-cache search <software name>
- apt-get install <software name>
- Other means : (.deb), (.tar), (.zip), (.tar.gz), (.tar.bz2)
- Use software centre

# Linux commands

- Man – very helpful, followed by info
- Tab completion
- Id, groups, uptime
- Uname
- Date, cal, time, times
- Whoami, pwd, cd, mkdir, rmdir, rm, cp, mv, ls

Fun- aplay /bin/bash ;)



# File System

- /etc – configuration files
  - passwd: user info
  - shadow: encrypted passwords
- /home – users files
- /bin – binaries (ls,kill,chmod,cp,mv)
- /sbin – system binaries(shutdown,adduser,mount)
- /dev - sda SCSI drive, hda IDE drive, fda floppy drive

# File System

- /lib – shared libraries used by programs
- /proc – virtual file system, files stored in memory not on drive, user can get info of running prog.
- /var – log files, man pages, mails
- /usr – subdirectories with user & admin tools
- /boot – static files for boot loader
- command 'man hier' will list the hierarchy

# File permissions

- Do a long listing.
- -rwxrwxrwx where :
  - could be d – directory
  - normal file
  - l – link
  - First rwx shows read write permission of file owner
  - Next rwx of group, the last of the other users.

# Changing file permissions

- `chmod` command in 2 ways:

- Alpha : `gu+r`, `gu-r`, `u+x`

- Octal : `644`, `777`, `755`

- Where :

- $7 = 4(\text{read}) + 2(\text{write}) + 1(\text{execute})$

- $6 = 4(\text{read}) + 2(\text{write}) + 0(\text{execute})$

- $5 = 4(\text{read}) + 0(\text{write}) + 1(\text{execute})$

- $4 = 4(\text{read}) + 0(\text{write}) + 0(\text{execute})$

# Additional permissions(NVM)

- d denotes a directory
- b denotes a block special file-that move data in blocks
- c denotes a character special file- through which system transfers data
- l denotes a symbolic link
- p denotes a named pipe
- During IPC.
- s denotes a domain socket – IPC on same host.

# Setuid, setgid, sticky bits

- set user id – file(or command) with SUID set enables users to be treated temporarily as privileged when run
- set group id – files or commands with SGID will run with group id of owner
- sticky bits - prevents users from renaming, moving or deleting contained files owned by other users.

# Examples

- passwd command has SUID set, allows changing password in /etc/shadow for anyone which only root can.
  - chmod u+s "/<command path>/command"
  - Find / -perm -4000 -print (find all setuid files)
- sgid on commands run with group id of owner
  - chmod g+s "/<path>/command binary"
- Sticky bits enabled files can only be deleted by root or the owner
  - chmod o+t /<path>/directory



# Extended File Attributes

- attr / lsattr / chattr interesting uses:
- chattr +i = immutable (means no one, not even root can change/delete/link the file)
- chattr +a = make file append-only (great for logs security!)
- chattr +s = secure deletion for file(used for kernels, not supported by ext2, ext3 or ext4)

# Advanced commands

- du – disk usage in Kb
- df – free space
- w – combination of who, ps, uptime
- free – system resources
- top
- cron – used for autorun, 7 fields :
  - minute hour dom month dow user cmd

# Users/Passwords

- /etc/passwd contains user list
- /etc/shadow contains passwords
- man 3 crypt shows:
  - 1 | MD5 (22 characters)
  - 2a | Blowfish
  - 5 | SHA-256 (43 characters)
  - 6 | SHA-512 (86 characters)

# Command History

- `history`
- `..bash_history`
- `!!` - repeat last command
- `!wget` – repeat the wget command
- `!!:p` or `!wget:p`
  - To print command, and not repeat it.

# Bash Almighty

- Bourne Again SHell (Bash)
- Default in most systems
- Belongs to the 'sh' lineage
- Very powerful & very useful to administrators
- And... very easy to use.

# The Shell: What does the shell do?

- Read and Execute commands
  - Built-in commands
  - Other commands from programs stored in some directory
- Provides support for better interaction with the OS
- Supports Scripting

# Starter tip/Basic help

- When in doubt:
  - consult the man pages
- find the file or program you are looking for via:
  - `find / -name "target file"`
- Program acting weird?
  - make sure it is the right program:
  - which "program"



# Finding about commands

- **type** – tells you if a command is a built-in, an alias, or a program (external to the shell)
- **which** – tells in which directory a utility is located
- **help** – displays information about built-in commands (it is a builtin itself) --help
- **info bash** – a good place to read about the BASH shell

# Shell Scripting Basics

- Pipe & Redirections

- STDIN (0) <

- mutt -s “mail sub” appy < message.txt

- STDOUT (1) >, >>

- any\_command > out.txt

- STDERR (2) 2>, 2>>

- any\_command 2> error.txt

- |

- echo “mail body” | mutt -s “sub” appy

# find, ps, grep, cut

## •Finding files

–find / -name log

–find / -name log 2> error.txt

–find / -name log > logiles 2> /dev/null

## •Process Management

–ps -ef

–ps -ef | grep ssh

–Ps -ef | grep ssh | cut -d “ “ -f1

# User Management

- Adding users
- Adding to groups
- Modifying user groups
- Changing default login shell (chsh -s)
- Disable root login
- Enable password ageing (chage -m -M -E -W)

# Service Management

- Using Netstat to list services
- fuser 22/tcp to identify process, or
- lsof -i:22
- Use ps for more info
- Shutdown unnecessary services, disable on startup
- Well known services with ports @ /etc/services

# ssh

- Secure Shell
- Encrypted version of telnet, secure
- ssh-keygen for first user (on server)
- scp for secured file copy over network
- ssh [appy@ipaddr](#) without password (creating backdoor)
- Use ssh as an encrypted proxy (ssh -D <localport> remotehost)

# Archive, Compress, Encrypt

- tar
- Zip
- Gzip
- Bzip
- Ccrypt
- Gpg <-- SAVIOR!
- Truecrypt/Veracrypt for volume encryption