CSN09703
Networked Services
Introduction/Linux Basics

Module Leader: Dr Gordon Russell
Lecturers: G. Russell, Petra Leimich

Networked Services

- Main focus is Linux system administration.
- Uses “linuxzoo.net” for practical exercises.
- Pre-requisites for this module are:
  - Basic OS concepts (partitions, virtual memory, processes, etc).
  - Networking basics (IP, Ethernet)

Why Linux

- Linux is a powerful “server” platform.
  - Many web sites use Linux as the operating system
  - Even Steve Ballmer of Microsoft said Linux has 60% of the server market in 2008.
  - Microsoft used to be a heavy user...
  - Considered by many to be more secure and reliable than comparable windows server solutions.
- Servers need Administrators
- Web developers need system knowledge
- Security Specialists securing servers is an important activity.
- Computer Forensics need to be able to consider server forensics.

Recommended Text

- Variety of good books on system administration.
- Recommended book for general admin:

  UNIX SYSTEM ADMINISTRATION HANDBOOK:
  Third Edition – EVI NEMETH et all
  Prentice Hall, ISBN 0-13-020601-6

- Nemeth also wrote “Linux System Administration Handbook”. Its just as good, cheaper, and lighter.
- Apache and DNS is best read online.

Elements Covered

- The module covers some important aspects of system administration for Linux machines:
  - Basic Unix / command prompt - revision
  - Linux user and network administration
  - Network Security
  - Linux-based DNS
  - Apache Web Server administration.
  - Hacking Techniques
  - Email
  - SELinux

Weeks 1 & 2

see moodle for additional activities

Week	day	Lecture	Linuxzoo
1	Mon	Intro / Linux basics Users, Permissions, Processes, Pipes	Intro1, intro2 wildcard, permission
	Tue	Basic Administration Concepts	pipe vi essential
	Wed	Networking	admin
2	Mon	SELinux	
	Tue	Firewalls + Security	net fwall
	Wed	DNS	dns

Weeks 3 & 4

see moodle for additional activities

Week	day	Lecture	Linuxzoo
3	Mon	Essential Apache	diag
	Tue	Using Apache	Apache1
	Wed	Revision	** PRACTICAL EXAM **
4	Mon	Email	Apache2
	Tue	Hacking	Revision
	Wed	Revision Short answer exam	

Lectures

- Lectures are not the source of all knowledge.
- You need to do some reading on your own, and to practice with the Linux machines.
- Otherwise you will struggle with this module.

Assessment

- The assessments involve both practical exam and a written exam.
- Both pieces of work are OPEN BOOK.
- Practical exam:
  - 90 minute Linux network and Linux configuration and troubleshooting.
  - you need to make specified changes to a Linux installation.
  - in Linuxzoo
  - 50% of module mark

- Written exam:
  - 90 minutes
  - 20 short-answer questions in moodle
  - 50% of module mark

Practicals

- Labs involve you being the administrator on your own Linux machine.
- We will use linuxzoo again http://linuxzoo.net
  - “Linux” tutorials
  - Linux Centos 7 machines
- Check moodle for your new class / auth code

Linuxzoo: update your setup

In the Account Links drop-down, choose “Your Profile”

Update the programme info to CSN09703 with auth code given in moodle

Linuxzoo reminder

- Your linuxzoo password is independent of your Napier password
- Each user needs an individual IP address so that Linuxzoo can match the user to the VM
- To create this, use a VPN connection created with Cisco anyconnect Server name: napier-sslvpn.napier.ac.uk
  - Needed for SSH and telnet.
  - You can VNC with or without a VPN

Running a tutorial Machine

- Your machine is a VIRTUAL machine.
- Your VM uses a shared computer resource.
- The resource is limited!
- Do not go crazy (do not recompile the world).
- Your virtual disk is usually preserved between sessions. BUT this is not reliable - Do not save your life work on it.

- If you mess up (e.g. delete something essential), go back to a fresh image

Telnet in the virtual machines

- Telnet is quite clever and usually no matter what OS and keyboard you have things just seem to “work”.
- Sometimes however telnet gets confused.
- If you ever have a problem where cursor keys stop working, or your editor corrupts the screen try these magic commands (you don’t type the “>”):

```
> export TERM=vt100
> tset
```

Which User

- “root” is the administrator, password is “secure”.
- If you need a basic user, use “alice”, password is “secure”.
- The first tutorials use a user “demo”, password “demo”
  - This user is created using a button
  - If you reboot and the user is missing, recreate it by pressing the button again.

- If you are logged in as one user and want to be another, use “su –”.
  - If you are “root” and want to be “demo”, then
    su - demo
  - After an “su” you can return to being “root” by pressing
    CTRL D
  - Note pressing it twice logs you out!
  - The command “whoami” tells you which user you are currently using.

```
[root@host-1-81 ~]# whoami
root
[root@host-1-81 ~]# su - alice
[alice@host-1-81 ~]$ whoami
alice
[alice@host-1-81 ~]$ logout
[root@host-1-81 ~]# whoami
root
[root@host-1-81 ~]#
```

- Always use the right user for the right question. Creating files as alice when you should be demo will be marked wrong!

Unix Flavours

- There are many flavours of unix and Linux.
- Linux “distributions” include:
  - Fedora
  - Centos
  - Redhat
  - Novell SUSE
  - Gentoo
- Different distributions have things in common but some differences. The distributions selection is often down to personal choice and “what my friend uses”.

Linux Flavours

Linux is a file-based system

- Everything on linux is either a file or a directory.
- A file which is executed becomes a process.
- Processes can be seen as files too.
- Devices, such as scanners and hard drives, are also files.

Directories

- /bin : This contains commands a user can run, like ‘ls’, but which might be needed during boot.
- /boot: Things needed during the boot process
- /dev : This contains devices, like the mouse.
- /home : This is where users store their files.
- /tmp : Temporary storage for users and the system
- /var : System files which can change.
- /etc : System config files which don’t change
- /lib and /lib64: Where all the system libraries live
- /proc : Files which represent the running system (like processes).
- /sbin : Commands which only an administrator would want.
- /usr : Commands which are never needed during bootup.

Redirection and pipes

- If you end a command with “>” or “>>”, its output goes to a file.
  - What’s the difference?
- If you end a command with “<“, its input comes from a file.
- “|” is a “pipe”, used to redirect output into another command

Command Parameters and flags

- Some commands change behaviours with different parameters.
- If a parameter relates to a file, then it is called a “parameter”.
- However, if the parameter changes the behavour of the program, it is instead called an “option” or “flag”.

Flags example

```
# cal

   February 2017
Mo Tu We Th Fr Sa Su
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28
```

```
# cal -s

 February 2017
Su Mo Tu We Th Fr Sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28
```

Man pages

- If you don't know what options or flags are possible for a command, use "man"
- For instance, to find out what flags cal uses, do:

```
$ man cal
```

- To get out of man, press "q".
- Enter shows more, line by line
- Space bar shows more – next page.

root@host-1-81:~

```
CAL(1)                          User Commands                          CAL(1)

NAME
       cal - display a calendar

SYNOPSIS
       cal [options] [[[day] month] year]

DESCRIPTION
       cal  displays  a  simple  calendar.  If no arguments are specified, the
       current month is displayed.

OPTIONS
       -1, --one
              Display single month output.  (This is the default.)

       -3, --three
              Display prev/current/next month output.

       -s, --sunday
              Display Sunday as the first day of the week.

       -m, --monday
              Display Monday as the first day of the week.

       -j, --julian
              Display Julian dates (days one-based, numbered from January 1).

       -y, --year
              Display a calendar for the current year.

 Manual page cal(1) line 1 (press h for help or q to quit)
```

man -k

- You can keyword search for commands
- For instance, what commands show a calendar?

```
$ man -k calendar
cal            (1)  - displays a calendar
cal            (1p) - print a calendar
difftime       (3p) - compute the difference…
```

HARD and SOFT LINKS

File links

- Sometimes we want to have the same file contents in two or more different places.
  - Edit one version and you edit all versions
  - You only use up the disk space for one version no matter how many copies you have.
  - This is done using a file link
- Links are common in the system directories, and are used for configuration as well as dynamic libraries.

Hard Link

- Lets assume that the current working directory is /home/jane.
- We wish to create a link 'hardfile2' within the sub-directory projects from the file 'hardfile'.

```
% date > hardfile ( create the file )
% ls -l
-rwx-xr-x. 1 jane users 605 Nov 18 12:25 hardfile

% ln hardfile projects/hardfile2

% ls -l projects/hardfile2
-rwx-xr-x. 2 jane users 605 Nov 18 12:25 projects/hardfile2
```

```
% ls -l projects/hardfile2
-rwx-xr-x. 2 jane users 605 Nov 18 12:25 projects/hardfile2
```

- The file 'hardfile' and its link 'hardfile2' are indistinguishable
- if 'hardfile' is updated then 'hardfile2' is updated
- Notice the link number has increased to 2.
  This would occur in both listings.

Soft Links (symbolic links)

- Let us assume that the current working directory is /home/john and we wish to create a link 'softfile2' within the subdirectory projects to the file 'softfile'.
- Notice the '-s' switch:

```
% date > softfile ( create the file )
% ls -l
-rwx-xr-x. 1 john users 605 Nov 18 12:25 softfile

% ln -s /home/john/softfile projects/softfile2

% ls -l projects/softfile2
lrwx-xr-x. 1 john users 605 Nov 18 12:25 projects/softfile2
                      ->     /home/john/softfile
```

Soft Links

```
lrwx-xr-x. 1 john users 605 Nov 18 12:25
                    softfile2 -> /home/john/softfile
```

- Notice the appended pathname on the long listing, the link number has not changed, but the permissions show an 'l' at the beginning of the long listing rather than a '-'.
- Again any updates in 'softfile' will be reflected in 'softfile2'.

Hard and soft links

Hard link

```
ln fileA hardlinkA
```

A hard link points to the same disk block(s) [inode] as the original file.
Still works when original file is deleted.

Soft link

```
ln -s fileB softlinkB
```

A soft link points to the original file,
like a windows shortcut.
Left "hanging" if original file is deleted.

Discussions

- Who is using Linux?

Discussions

- What is Linux for?
  - Desktop
  - Software Developers
  - Servers

CSN09703
Networked Services
Users, Permissions, Processes

Module Leader: Dr Gordon Russell

Lecturers: Gordon Russell, Petra Leimich

This lecture

Concepts

- Users
- File permissions
- Processes
- Extracting info from files (cut)

USERS

UID and GID

- In Unix, there are User IDs and Group IDs.
- User IDs uniquely identify a particular user.
- Group IDs allow users to be collected into groupings.
- Groups could be used to allow friends to share files, while stopping people not in that “group” of friends from reading the files.

Users

- User details are stored in 4 files.
  - /etc/passwd - General User details.
  - /etc/shadow - User passwords.
  - /etc/group - The users' groups.
  - /etc/gshadow - Passwords for groups.

> cat **/etc/passwd**

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
…
```

- Username, *x*, UID, GID, text description (finger information), home directory, login shell.
- Field delimiter is a colon (:)

> head –3 **/etc/shadow**

```
root:$1$RcFIaOlb$bwl5dvTECg3M1ZgMQ7e6I.:12663:0:99999:7:::
bin:*:12621:0:99999:7:::
daemon:*:12621:0:99999:7:::
```

- Passwords are md5 hashed. (with random seed)
- * means no password – these “users” can’t log in
- Shadow passwords can expire and have rules.

Question: passwords

- The users root and alice have the same password
- What are the implications for the hashes stored?

demo

- (logged in as root)
- cat /etc/shadow | grep -E 'root|alice'

> tail –3 **/etc/group**

```
gdm:x:42:
dovecot:x:97:
mysql:x:27:
```

- Group contains group names, x, and the number which defines that group uniquely.
- After the last : can be a list of users who are in that group.

```
friends:x:500:gordon,andrew
```

> tail –3 **/etc/gshadow**

```
gdm:x::
dovecot:x::
mysql:x::
```

- Allows people to change groups on a password.
- Not often used, but when done the password is placed here where the ‘x’ is.

EXTRACTING INFORMATION

How can we extract information from structured files?

- You know how to use > for redirection, and how to use grep, possibly with a pipe, to extract relevant rows of information.
- But that doesn't allow us to, for example, find all the usernames beginning with "a".

- We can use a regular expression, "^a", to extract the relevant rows from /etc/passwd:

```
$ grep "^a" /etc/passwd
adm:x:3:4:adm:/var/adm:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
andrew:x:501:500:Andrew Cumming:/home/andrew:/bin/bash
```

cut

- The grep got the information, but also lots of other pieces of info. How can we extract only the usernames?

- The command “cut” chops things out of a line.
- It will split a field out of a line so long as it knows what character is the delimiter (marks the end of one field and the start of another).
- In /etc/passwd, “:” is the delimiter, so cut is –d”:”
- The username is in field 1, so –f1

```
$ grep "^a" /etc/passwd | cut –d”:” –f1
adm
apache
andrew
```

PERMISSIONS

Permissions

- A file or directory has various permissions and ownerships applied to it.
- Three file permissions:
  - r – read permission
  - w – write permission
  - x – execute permission
- Three permission levels:
  - u – User (the creator of the object)
  - g – Group (a group identifier)
  - o – Other (everyone not in the User or Group specified)

chmod

- Used to change permissions
- Can use numeric or symbolic notation

Syntax

chmod <new permissions> <file>

example

```
> touch /tmp/test
> ls -l /tmp/test
-rw-r--r--.  1 root root 0 Sep 23 15:47 /tmp/test

> chmod og+wx /tmp/test

> ls -l /tmp/test
-rw-rwxrwx.  1 root root 0 Sep 23 15:47 /tmp/test
```

chown

- Used to change owner, optionally also group

Syntax

chown <new owner>[.<new group>] <file>

example

```
> ls -l /tmp/test
-rw-rwxrwx.  1 root root 0 Sep 23 15:47 /tmp/test

> chown ftp.mem /tmp/test
> ls -l /tmp/test
-rw-rwxrwx.  1 ftp mem 0 Sep 23 15:47 /tmp/test
```

chgrp

- Used to change group only

Syntax

chgrp <new group> <file>

example

```
> ls -l /tmp/test
-rw-rwxrwx.  1 ftp mem 0 Sep 23 15:47 /tmp/test

> chgrp root /tmp/test
> ls -l /tmp/test
-rw-rwxrwx.  1 ftp root 0 Sep 23 15:47 /tmp/test
```

User(Owner) Group Other

-rwxrwxrwx.

type of object:

- means normal file

d means directory

c means a character device (mouse, keyboard)

b means a block device (ide disk, scsi disk)

Alternative Access methods

There are more types to discover!

> ls –ld /home

```
drwxr-xr-x.  2 root root 4096 Jul 27 13:38 /home
```

- /home is a directory, owned by root in group root.
- UID root can do anything, group root can rx, others can rx.
- Size is not really useful for directories.
- The "." immediately after the permissions indicates that alternative access methods exist.
  - If this is a " " (space) there are no additional methods.
  - "." (dot) indicates a SELinux security context
  - "+" (plus) indicates a combination of access methods.
- The "2" indicates that there are two hard links to the file

Alternative Access Methods

- Cover this in more detail in a later lecture.
- ACL access methods allow you to set fine-grained permissions:

```
> touch test
> setfacl –m user:root:rwx test
> ls -l test
-rw-rw-r--+ 1 gordon gordon 0 Aug 30 15:25 test
> getfacl test
   user::rw-
   user:root:rwx
   group::rw-
   mask::rw-
   other::r--
```

- SELinux access methods map complex process rules to file context information, e.g. The web server can only see files in the “httpd_user_context_t” context.

Numeric Notation

- An older way of looking at permissions.
- Still needed for some commands, and a fast way of changing multiple permissions.
- Based on octal, 4 digits long.
- Digit 0 is usually 0, 1 is OWNER, 2 GROUP, 3 OTHER.
- Values:

Octal	Binary	Perms	Octal	Binary	Perms
7	111	rwx	3	011	-wx
6	110	rw-	2	010	-w-
5	101	r-x	1	001	--x
4	100	r--	0	000	---

Example

- If User rwx, Group rx, Other rx,
  - Symbolic –rwxr-xr-x
  - Numeric 0755
- If User rwx, Group x, Other none
  - Symbolic –rwx--x---
  - Numeric 0710

> umask 022

- When a command creates a file or directory the default is:
  - rwxrwxrwx – for directories
  - rw-rw-rw- – for files
- The value of your umask is SUBTRACTED from the numeric protection code.
- So removing write for group and other you need to know that 2 stands for w, and thus for:
  - rw-r--r-- (644) - Write only for owner.
  - This is numerically, 666-022 => 644
  - So the umask is 022.

The umask mask

- 0022
  - Col 0 is always 0, Col 1 is OWNER
  - Col 2 is GROUP, Col 3 is OTHER
- Values:

Octal	Binary	Perms	Octal	Binary	Perms
0	000	rwx	4	100	-wx
1	001	rw-	5	101	-w-
2	010	r-x	6	110	--x
3	011	r--	7	111	---

Question

Give the chmod command to set the permissions for the file myfile such that owner can read and write, group can read, and all permissions are denied for others. Your command should work regardless of what the previous permissions were.

(a) Using numeric notation

(b) Using symbolic notation – write a single statement

(c) Which notation do you prefer? Why?

Question

- Alice created file1 and directory mydir,
  then the umask was changed and she created file2 and mydir_new.

```
[alice@host-5-137 ~]$ ls -lt
total 8
-r--r-----. 1 alice alice 43 Mar  6 18:40 file2
dr-xr-x--x. 2 alice alice  6 Mar  6 18:40 mydir_new
-rw-rw-r--. 1 alice alice 45 Mar  6 18:37 file1
drwxrwxr-x. 2 alice alice  6 Mar  6 18:36 mydir
```

- What was Alice's original umask?
- What was the command used to change it?

PROCESSES

Processes

- Processes are running programs.
- They have their own ID (pid)
- Some processes are part of the filesystem and can be found.
- Some processes are special, and cannot be found, and these are usually described [brackets].
- The INIT process is the boss process in linux.

> ps aux

```
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
root       1    0.1  0.8  1480  496 ?        S    12:57   0:00 init [5]
root       2    0.0  0.0     0    0 ?        SWN  12:57   0:00 [ksoftirqd/0]
root       3    0.0  0.0     0    0 ?        SW<  12:57   0:00 [events/0]
root       4    0.0  0.0     0    0 ?        SW<  12:57   0:00 [khelper]
root       16   0.0  0.0     0    0 ?        SW   12:57   0:00 [kjournald]
…
root       527  0.0  0.9  1464  576 ?        S    12:58   0:00 syslogd -m 0
rpc        553  0.0  0.9  1544  584 ?        S    12:58   0:00 portmap
rpcuser    573  0.1  1.3  1644  812 ?        S    12:58   0:00 rpc.statd
root       658  0.4  2.4  3656 1484 ?        S    12:58   0:00 /usr/sbin/sshd
gordon   15521  0.0  0.1  3992  760 pts/1    R    20:41   0:00 ps aux
```

State Codes

- Standard Codes
  - D uninterruptible sleep (usually I/O)
  - R runnable (on run queue)
  - S sleeping
  - T traced or stopped
  - W paging
  - X dead
  - Z a defunct ("zombie") process

- Additional Codes
  - W has no resident pages
  - < high-priority process
  - N low-priority task (nice)
  - L has pages locked into memory (for real-time and custom IO)

Process Relationships

- Processes form “trees” of parentage.
- All processes have the parent INIT.
- If a process starts another process, that new process has a parent of the old process.
- I now run pstree, in the bash shell, after logging in to the machine using ssh (controlled by sshd).

> pstree

```
[root@host-1-97 ~]# pstree
systemd─┬─ModemManager───2*[{ModemManager}]
        ├─NetworkManager─┬─dhclient
        │                └─3*[{NetworkManager}]
        ├─2*[abrt-watch-log]
        ├─abrtd
        ├─accounts-daemon───2*[{accounts-daemon}]
        ├─acpid
        ├─alsactl
        ├─at-spi-bus-laun─┬─dbus-daemon───{dbus-daemon}
        │                 └─3*[{at-spi-bus-laun}]
        ....
        ├─smartd
        ├─sshd───sshd───bash───pstree
        ├─systemd-journal
        ├─systemd-localed
        ├─systemd-logind
        ├─systemd-udevd
        ├─tuned───4*[{tuned}]
        └─upowerd───2*[{upowerd}]
```

/proc

- Processes are represented as files in /proc
- They appear as a directory with a name equal to the PID of the process.
- They have things in the directory which define the process in question.

> ls –l /proc/668

```
-r--------.  1 root root 0 Sep 21 16:32 auxv
-r--r--r--.  1 root root 0 Sep 21 16:31 cmdline
lrwxrwxrwx.  1 root root 0 Sep 21 16:32 cwd -> /
-r--------.  1 root root 0 Sep 21 16:32 environ
lrwxrwxrwx.  1 root root 0 Sep 21 16:32 exe -> /usr/sbin/sshd
dr-x------.  2 root root 0 Sep 21 16:32 fd
-r--r--r--.  1 root root 0 Sep 21 16:32 maps
-rw-------.  1 root root 0 Sep 21 16:32 mem
-r--r--r--.  1 root root 0 Sep 21 16:32 mounts
lrwxrwxrwx.  1 root root 0 Sep 21 16:32 root -> /
-r--r--r--.  1 root root 0 Sep 21 16:31 stat
-r--r--r--.  1 root root 0 Sep 21 16:32 statm
-r--r--r--.  1 root root 0 Sep 21 16:31 status
dr-xr-xr-x.  3 root root 0 Sep 21 16:32 task
-r--r--r--.  1 root root 0 Sep 21 16:32 wchan
```

> ls –l /proc/668/fd

```
lrwx------.  1 root root 64 Sep 21 16:32 0 -> /dev/null
lrwx------.  1 root root 64 Sep 21 16:32 1 -> /dev/null
lrwx------.  1 root root 64 Sep 21 16:32 2 -> /dev/null
lrwx------.  1 root root 64 Sep 21 16:32 3 -> socket:[4230]
```

- Files which that process has open.
- 0,1,2 are STDIN,STDOUT,STDERR.
- 668 : sshd – listening on a socket for people logging in with ssh.

> sleep 20 > /tmp/hia &
[1] 854
> ls –l /proc/854
> ls –l /proc/854/fd

```
lrwxrwxrwx.  1 root root 0 Sep 21 16:45 cwd -> /root
-r--------.  1 root root 0 Sep 21 16:45 environ
lrwxrwxrwx.  1 root root 0 Sep 21 16:45 exe -> /bin/sleep
dr-x------.  2 root root 0 Sep 21 16:45 fd

lrwx------.  1 root root 64 Sep 21 16:45 0 -> /dev/pts/0
l-wx------.  1 root root 64 Sep 21 16:45 1 -> /tmp/hia
lrwx------.  1 root root 64 Sep 21 16:45 2 -> /dev/pts/0
```

Daemons

- A Daemon is a process started when you boot which runs in the background.
- Not all things started when booting stay running (e.g. they set something up and then die).
- To help us, daemons usually have a name which ends with a “d”. (e.g. syslogd, sshd).

> top

```
top - 16:03:17 up  1:04,  1 user,  load average: 0.00, 0.00, 0.00
Tasks:  35 total,   2 running,  33 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0% us,  0.0% sy,  0.0% ni, 100.0% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:     59764k total,    52308k used,     7456k free,     6192k buffers
Swap:   205816k total,        0k used,   205816k free,    32472k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
  807 root      16   0  1624  728 1412 R  0.0  1.2   0:00.02 in.telnetd
  934 root      16   0  1828  872 1628 R  0.0  1.5   0:00.00 top
```

SYSLOG

- The syslogd daemon is your friend.
- It helps other daemons record what is going on into a file.
- On the website, you can click on “syslog output” and see what syslogd has noticed.
- This output, known as the syslog, can also be seen from the prompt using “dmesg”.

> who

root pts/0 Sep 21 15:59 (hub1-gw)

- I am root, logged on from hub1-gw.
- My session is linked to a device which handles my screen and keyboard, called pts/0
- This refers to /dev/pts/0

> ls –l /dev/pts/0

crw--w----. 1 root tty 136, 0 Sep 21 16:49 /dev/pts/0

- This is a character device.
- The person connected via it always owns it.
- There are no sizes with block or char devices.
- 136 is the major device number
- 0 is the minor device number.

mknod

- Devices are usually created automatically.
- To create a new file to represent a device, use mknod.

```
> mknod /tmp/screen c 136 0
> echo “hello there” > /tmp/screen
hello there
```

- Find things out for yourself!

> man ps

Telnet linuxzoo.net

```
PS(1)                        Linux UserôÇös Manual                        PS(1)

NAME
       ps - report process status

SYNOPSIS
ps [options]

DESCRIPTION
ps gives a snapshot of the current processes. If you want
a repetitive update of this status, use top. This man
page documents the /proc-based version of ps, or tries to.

COMMAND-LINE OPTIONS

This version of ps accepts several kinds of options.

Unix options may be grouped and must be preceeded by a dash.
BSD options may be grouped and must not be used with a dash.
Gnu long options are preceeded by two dashes.

Options of different types may be freely mixed.

Set the I_WANT_A_BROKEN_PS environment variable to force BSD syntax even
:
```

Discussion

Future of file permission:

- Is User/Group/Other sufficient?
- Simple control methods? ACL…
- Complex control methods? SELinux

Discussion

What happens to owner and permissions when alice copies a file that belongs to root?

Labs

You should now be able to do the following Linuxzoo labs:

- Permission

- If you have time, do the additional activity described on the following slide

CSN09703
Networked Services
Apache – Basic Authentication

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Basic Authentication

Basic Authentication

- Often you might want simple usernames and passwords to control access you parts of a website.
- There are many approaches for this.
- The easiest way is to use Basic Authentication.
- This, when required, asks the browser to ask you for a username and password for accessing protected pages.
- The username and password is sent as clear text for every page request made by the browser.

.htaccess

- The best way to control basic authentication is via an .htaccess file in the directory to protect.
- To allow this the <directory> definition which includes the directory to be protected must have

  AllowOveride AuthConfig

Building a Password File

- You have to create a file with usernames and passwords.
- It is a good idea if this file is not one which someone can access via a URL.

```
> htpasswd –c /home/gordon/password andrew
New Password: *******
Retype New Password: *******
Adding password for user andrew.
```

-c is only the first time running the command, as this creates the file too. Miss out –c after the first run.

.htaccess

```
AuthType Basic
AuthName "Restricted Files"
AuthUserFile /home/gordon/password
Require user andrew
```

- Authtype Digest
  - This is another option, which requests the passwords in an encrypted format. It is not as widely supported as Basic.

The password file

- The password file created is just a text file.
- As a text file it does not scale well…
  - As more users are added the file gets bigger.
  - On every page request the file has to be parsed again.
- There are other formats available using hashed files (either db or dbm). These are faster to access but more complex to manage.

Any valid user

Require user andrew

- Can be changed to

Require valid-user

- In this way any user in the password file can access the directory.

Groups

- Just as in passwd users are also in groups, you can use the same idea for apache.
- Create a plain text file with the following format:

Groupname: user1 user2 user3 …

- If users gordon and andrew exists, and you want them to be known as group staff…

staff: gordon andrew

Add to .htaccess

AuthType Basic
AuthName "By Invitation Only"
AuthUserFile /home/gordon/password
AuthGroupFile /home/gordon/groups
Require group staff

Basic Auth Problems

- Its simple protection.
- Passwords in the clear.
- Every request need the password file lookup
- Large numbers of users difficult to manage
- Not a good idea for commercial systems
  - Yet some big sites use it!
- However, users recognise it and understand it.

Control by IP

- .htaccess can offer more control than just Basic Authentication.
- You can also restrict access to directories by IP.
- To do this you need to use variations of the “require” directive.
  - Require user gordon
  - Require group staff
  - Require all granted
  - Require all denied
  - Require ip 10.0.0.1
  - Require host linuxzoo.net
  - Require not ip 10 172.16 192.168.200
- Multiple require statement can be combined together
  - The default is that ANY can be true to allow access

Example

- Stop 10.0.0.1 accessing a directory…
- Edit the .htaccess in that directory:

```
Require not ip 10.0.0.1
```

Combining rules

- By default the rules are combined together as any. So

```
Require ip 10.0.0.1
Require ip 10.0.0.2
```

- Is actually

```
<RequireAny>
  Require ip 10.0.0.1
  Require ip 10.0.0.2
</RequireAny>
```

- This may make statements easier to understand.

Combining rules – Other methods

- RequireAny basically ORs the
- Other directives exist
  - RequireNone: None of the directives must succeed
  - RequireAll: All of the directives must succees

- So to stop 10.0.0.1 and .2…

```
<RequireNone>
  Require ip 10.0.0.1
  Require ip 10.0.0.2
</RequireNone>
```

More complex example

- Given a basic authentication definition, allow user gordon, or group magic, or anyone from IP 10.0.0.50:

```
<RequireAny>
      Require ip 10.0.0.50
      Require group magic
      Require user gordon
</RequireAny>
```

More More complex example

- Given a basic authentication definition, allow user gordon from 10.0.0.1, or user jim from IP 10.0.0.50:

```
<RequireAny>
      <RequireAll>
            Require ip 10.0.0.50
            Require user jim
      </RequireAll>
      <RequireAll>
            Require ip 10.0.0.1
            Require user gordon
      </RequireAll>
</RequireAny>
```

Discussion

Question 1

- The following is an .htaccess file of a fictitious student on a student's web account.

```
Auth Type Basic
AuthTitle "Password Required"
AuthUserFile home/jim/.www-password
Required user jim
```

- Spot 4 errors

Question 2

- The following is an .htaccess file of a fictitious student on a student’s web account.

```
AuthType Basic
AuthName "Password Required"
AuthUserFile /home/09006754/.www-password
Require user server_admin
```

- Provide the code to change the password for server_admin.

CSN09703
Networked Services
Day 2a: Basic Administration Concepts part 1

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

This lecture

- Disks and partitions
- The boot process

Disks and Partitions

Disks in Linux

- Disk storage is a layered system

Disks

- In Linux a disk is often a SATA disk, and is identified by:
  - /dev/sda – lowest numbered SCSI device
  - /dev/sdb – next lowest SCSI device
  - …
- The CDROM drive, which is also a block storage device, frequently has a name /dev/sr0.
  - For convenience /dev/cdrom is a softlink to the actual cdrom device

```
$ ls -l /dev/cdrom
lrwxrwxrwx. 1 root root 3 Sep 25 13:10 /dev/cdrom -> sr0
```

Partitions

- Rather than use the whole disk for one purpose…
- Split disk up into chunks.
- The chunks are known as partitions.
- Partitions can be primary or secondary.
- This is partially a hang-over from when DOS could only handle 4 partitions…

```
>  sfdisk -l /dev/sda

Disk /dev/sda: 19449 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0

   Device Boot Start     End   #cyls    #blocks   Id  System
/dev/sda1   *      0+   1274-   1275-  10240000   83  Linux
/dev/sda2       1274+   3824-   2550-  20480000   82  Linux swap / Solaris
/dev/sda3       3824+  19449-  15625- 125506560   83  Linux
/dev/sda4          0       -       0          0    0  Empty
```

Common Partitions

- At a minimum a linux box has
  - “/”, the partition which holds the main directory tree, such as the operating system files
  - “/boot”, a partition which stays close to the start of the disk structure, designated to be easily accessible from a variety of BIOS configurations and disk formats, used for bootstraping.
  - “swap”, an area for holding virtual memory which has been swapped out of main memory.
- Options include
  - “/home”, to keep users and system files physically separate.
  - “/var”, as some feel that /var is changed at a higher rate than other partitions and thus there may be more security in formatting is separately.

Logical Volume Manager LVM

- If you want to change your partitions after installing, then you are doing something very risky…
- It is possible to destroy every partition by damaging the partition table.
- Often the current partitions use 100% of the disk, so repartitioning to make something bigger means shrinking other partitions.
- Formatted partitions don’t like being shrunk, and need to be compressed and migrated before being shrunk.
- LVM and other tools tries to make this simpler, by adding dynamic run-time controls to partition management.

LVM

```
$ lvdisplay
 --- Logical volume ---
  LV Path        /dev/centos_lvm/root
  LV Name        root
  VG Name        centos_lvm
  LV UUID        xnNIQ2-ct1m-UqaR-BkjO-GrNI-FUwt-e2ciYM
  LV Size        6.51 GiB
```

LVM Management

- You can do things dynamically.. But you still have to have care.
- Create a new partition in an LVM?

```
$ lvcreate -L 2G -n newstuff centos_lvm
```

- Shrink an existing partition?

```
$ lvreduce -L-1G -n mypartition centos_lvm
```

- Add a new harddrive… effectively glue its space onto LVM and make the dynamic area bigger.

```
$ vgextend centos_lvm /dec/sdb1
```

- To make resizing work, you still have to carefully resize the partition format, but at least there is no danger in deleing all the partitions accidentally.

Partition Format

- When you format a partition for use, in say EXT3, or NTFS, of XFS, the partition gets a block device id. This uniquely (hopefully) identifies the partition.
- In the past you had to identify partitions by their number and disk device name, such as partition 2 on sda being /dev/sda2.
- Better to identify them by block id, so if partitions are moved (say by lvm), or disks rearranged (so sda becomes sdb), you can still identify a partition easily, reliability, and consistently…

Block identifiers

```
$ blkid
```

```
/dev/sda1: UUID="f3b744e4-e754-4842-93d1-43b06de64b66" TYPE="xfs"
/dev/sda2: UUID="kIogiH-f548-AELA-NMVK-sbr7-9u7j-2K6MDz" TYPE="LVM2_member"
/dev/sdb1: UUID="971d09b6-8ce8-49c7-9ec9-16b0155f42cf" TYPE="swap"
/dev/mapper/centos_lvm-root: UUID="b66fdf9b-16f0-4648-9663-536881db0ab1" TYPE="xfs"
```

> cat /etc/fstab

- When the system boots the fstab file tells the kernel what filesystems to load

```
/dev/mapper/centos_lvm-root                /      xfs     defaults        1 1
UUID=f3b744e4-e754-4842-93d1-43b06de64b66 /boot  xfs     defaults        1 2
UUID=971d09b6-8ce8-49c7-9ec9-16b0155f42cf swap   swap    defaults        0 0
```

> df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/centos_lvm-root	6813696	4100528	2713168	61%	/
devtmpfs	241940	0	241940	0%	/dev
tmpfs	250952	80	250872	1%	/dev/shm
tmpfs	250952	4752	246200	2%	/run
tmpfs	250952	0	250952	0%	/sys/fs/cgroup
/dev/sda1	508588	136588	372000	27%	/boot

- “df -h” is also useful, translating bytes in MB or GB as appropriate…

Disk Usage

- If you want to find out how much disk space a directory is using, the “du” command does this easily.

```
$ du -s /usr/lib
477464  /usr/lib
$ du -sh /usr/lib
467M    /usr/lib
```

- “-s” is useful, otherwise it tells you about all subdirectories too.
- “-h” puts it into human readable form.

Linux Boot Process

Booting to kernel

- From switch-on:
  - PC BIOS selects a boot disk
  - BIOS loads the boot block and executes it.
  - This loads a stage 1 boot loader.
  - Stage 1 loads stage 2 loader.
  - Linux loader (e.g. Grub, lilo) runs
  - Operator selects from loader menu
  - Kernel loaded with device ramdisk

GNU GRUB version 2.00-15
Linux /boot/vmlinuz-3.2.0-55-generic
Linux /boot/vmlinuz-3.2.0-55-generic (single)
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line. ESC to return
previous menu.

Startup Commands

- As linux boots, it runs various system scripts.
- In Redhat/Centos/Fedora, this is controlled largely by systemd.
  - In the olden days this was controlled by init.d scripts, based on System V.
- Systemd forms a tree of dependencies:
  - Start the network before starting sshd…
  - Start the firewall after the network…
- The internals of systemd are very complex, so best to always use the system commands to control how systemd runs

Target tree

multi-user.target
multi-user.target.wants/
systemd-ask-password-
dbus.service
dbus-connect.timer
basic.target
basic.target.wants/
system-udevd.service
audio-daemon.service
graphics-daemon.service
udisks.service

systemd.analyze

Controlling services in systemd

- There are many services you may wish to control, such as sshd, apache, databases, etc.
- Things which are services are named “something.service”, such as sshd.service.
- Control is via systemctl
- Start a service: systemctl start sshd.service
- Stop a service: systemctl stop sshd.service
- Restart sshd: systemctl restart sshd.service
- Reload sshd: systemctl reload sshd.service.

Make things start when you boot

- Start a service: systemctl start sshd.service
  - The sshd service will run only till you reboot.
- If you want something to run every time you reboot then enable it.
  - systemctl enable sshd.service
- Change your mind and want it disabled next time you boot?
  - systemctl disable sshd.service
- Stop a service: systemctl stop sshd.service

Status

- The information about a service is stored in a few locations.
- You can do that yourself, but systemctl can build the info for you…
- systemctl status sshd.service

```
[root@host-19-17 grub.d]# systemctl status sshd.service
sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled)
   Active: active (running) since Mon 2014-09-29 14:55:19 BST; 1h 23min ago
  Process: 1014 ExecStartPre=/usr/sbin/sshd-keygen (code=exited, status=0/SUCCES
S)
 Main PID: 1023 (sshd)
   CGroup: /system.slice/sshd.service
           └─1023 /usr/sbin/sshd -D

Sep 29 14:55:19 host-0-0.linuxzoo.net systemd[1]: Started OpenSSH server daemon.
Sep 29 14:55:20 host-0-0.linuxzoo.net sshd[1023]: Server listening on 0.0.0.0...
Sep 29 14:55:20 host-0-0.linuxzoo.net sshd[1023]: Server listening on :: port...
Sep 29 14:55:36 host-19-17.linuxzoo.net sshd[1306]: Accepted password for roo...
Sep 29 14:57:30 host-19-17.linuxzoo.net sshd[1362]: Accepted password for roo...
Sep 29 15:13:20 host-19-17.linuxzoo.net sshd[1665]: Accepted password for roo...
Sep 29 15:17:37 host-19-17.linuxzoo.net sshd[1746]: Accepted password for roo...
Sep 29 16:02:23 host-19-17.linuxzoo.net sshd[2409]: Accepted password for roo...
Hint: Some lines were ellipsized, use -l to show in full.
[root@host-19-17 grub.d]#
```

Configuration file: /usr/lib/systemd/system/sshd.service

```
[Unit]
Description=OpenSSH server daemon
After=syslog.target network.target auditd.service

[Service]
EnvironmentFile=/etc/sysconfig/sshd
ExecStartPre=/usr/sbin/sshd-keygen
ExecStart=/usr/sbin/sshd -D $OPTIONS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

System V init.d

- This is still in use in some parts of CentOS, and in Linux in general.
- Originally there were “runlevels”, each numbered and associated with a concept like “boot with a graphical interface”, “boot with no networking”, etc. Booting with a GUI is considered runlevel 5.
- By default in sysv, runlevel 5 is considered normal.
- What runs at runlevel 5 is controlled by soft links in /etc/rc5.d.
  - Each link indicates what runs (or not) at this runlevel.
  - During boot or shutdown the system transitions between runlevels.
  - During boot it enters runlevel 5, and during shutdown it enters runlevel 6.
  - It softlinks to scripts in /etc/init.d, which control each service.
  - The name of the link indicates what happens in that runlevel

> ls –l rc5.d

```
lrwxrwxrwx. 1 root root 20 Sep  8 09:20 K50netconsole -> ../init.d/netconsole
lrwxrwxrwx. 1 root root 17 Sep  8 13:06 S10network -> ../init.d/network
lrwxrwxrwx. 1 root root 17 Sep  8 09:35 S20iprinit -> ../init.d/iprinit
lrwxrwxrwx. 1 root root 19 Sep  8 09:35 S20iprupdate -> ../init.d/iprupdate
lrwxrwxrwx. 1 root root 17 Sep  8 13:06 S21iprdump -> ../init.d/iprdump
```

S/K priority service-name

- S10network :
- ls –l S10network

lrwxrwxrwx. 1 root root 17 Sep 8 13:06 S10network -> ../init.d/network

- Starts at priority 10 – runs after 9 and before 11
- “S” indicates the system should start networking at this runlevel
- “K” indicates the system should stop something instead.

- So rc6.d holds things like:

lrwxrwxrwx. 1 root root 17 Sep 8 13:06 K90network -> ../init.d/network

The syslog

- dmesg shows the messages which have recently been generated by the kernel…
- This and service messages are collated by the syslogd.
- It sorts messages out according to a config file in /etc, and then categorises the messages and stores them somewhere in /var/log/…
- For instance
  - systemd messages end up in “messages”.
  - sshd messages end up in “secure”.
  - Web server diagnostics will end up in /var/log/httpd/…
- The location of some files and their function does change between distributions…
- You can always use “systemctl status” to locate recent messages.

The xinetd super-daemon

- Some of the services (e.g. sshd) are processes.
- They start running from an rc script.
- They wait on their own for comms.
- They terminate only when the machine does down.
- Some people say this wastes resources.
- The super-server concept was born.

XINETD

- Xinetd waits for requests from the internet.
- From the requests it works out what program would like to deal with that request.
- It then starts that program running and gives it the waiting requests.
- In this way resources are only used if someone actually requests access to a particular service.

- systemd supports this type of service too, but without XINETD.
  - However rather than “service” this is known as a “socket”.
- For our virtual machines, telnet has been set up to use this.
  - telnet.socket
  - When connections arrive, it triggers in turn telnet.service for that connection.

Terminating a process

- If you know the process id (the PID) of a program you can terminate it quickly and easily
- You send it a message (a signal) to tell it to end.
- The message to end now is called SIGKILL.

> ps aux | grep sshd

root 1796 …… /usr/sbin/sshd

> kill –s SIGKILL 1796

Discussion

- Here are some past exam questions you should now be able to answer:

Question 1

- The following commands are typed on a Unix computer.

```
$ mkdir temp
$ cd temp/
$ mkdir txt.txt/
$ cd txt.txt/
$ touch hello
$ cd ..
$ ls *.*
```

What is printed on the screen in response to the last line of the commands?

CSN09703
Networked Services
Email Management - SMTP

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

SMTP

SMTP

- Email is send between source and destination using a simple protocol.
- SMTP is the basic protocol used
  - Simple Mail Transport Protocol (RFC 821 and 1123)
- SMTP deals mostly with simple email without attachments.
- Attachment emails (plus email delivery using some advanced features) uses ESMTP
  - Extended/Enhanced SMTP (RFC 1869).

SMTP

- Very simple protocol
- Text Based
- You can send email using TELNET.
- Easy to forge an email…

telnet grussell.org 25

Connected to grussell.org (10.10.5.5).

Escape character is '^]'.

220 grussell.org ESMTP Sendmail 8.12.11/8.12.11; Sun, 14 Nov 200819:01:01 GMT

> **helo pc236b.napier.ac.uk**

250 grussell.org Hello pc236b.napier.ac.uk [10.4.5.6], pleased to meet you

> **mail from: g.russell@napier.ac.uk**

250 2.1.0 g.russell@napier.ac.uk … Sender ok

> **rcpt to: me@grussell.org**

250 2.1.5 me@grussell.org.. Recipient ok

telnet grussell.org 25

> **data**
354 Enter mail, end with “.” on a line by itself
> From: “Santa" claws@northpole.com
> To: Gordon Russell <me@grussell.org>
> **Date: Tue, 15 Jan 2008 16:02:43 -0500**
> Subject: SMTP

Hello gordon.
I am Santa
> .
250 Ok: queued as 5555
> QUIT
221 Bye

Envelope and Headers

- The email information about from and to supplied using SMTP (except the contents of the DATA command) makes up what is known as the Message Envelope.
- The from, to, and other initial information in the email itself (the part in DATA) is known as the email header.
- The email instructions in the envelope DOES NOT have to match that in the email headers.
- This is useful when, for instance, dealing with email mailing lists: The envelope directs it to you, but the headers state it is actually to “the name of the email group”.

Forged Emails

- Note that the envelope FROM was:
  > **mail from: g.russell@napier.ac.uk**

- The data FROM was:
  From: “Santa" claws@northpole.com

- This is perfectly valid. It will be delivered.
- The envelope is used through the delivery process, but it is discarded when it is finally delivered to the recipient.
- The final recipient cannot recover the information in the envelope. However, the headers can give useful information.

Email headers

```
From g.russell@napier.ac.uk Sun Nov 15 11:12:21 2009
Received: from pc236b.napier.ac.uk [10.2.4.5]
        by grussell.org (8.18.11) id PDQ666
        Sun Nov 15 11:12:20 2008 -0000
Received: (gor@localhost)
        by pc236b.napier.ac.uk (8.18.11) id LXY123
        Sun Nov 15 11:12:16 2008 -0000
Date: Sun Nov 15 11:12:15 2008 -0000
From: g.russell@napier.ac.uk
To: me@grussell.org
Message-Id: <20041115111215.LXY123@pc236b.napier.ac.uk>
Subject: Wow

Message body is here. This is the message.
```

- From “Date” down is the data added by the original sender.
- As the email moves from machine to machine, extra information is added to the data.

```
From g.russell@napier.ac.uk Sun Nov 15 11:12:21 2009
Received: from pc236b.napier.ac.uk [10.2.4.5]
        by grussell.org (8.18.11) id PDQ666
        Sun Nov 15 11:12:20 2008 -0000
Received: (gor@localhost)
        by pc236b.napier.ac.uk (8.18.11) id LXY123
        Sun Nov 15 11:12:16 2008 -0000
```

Received

Received: from pc236b.napier.ac.uk [10.2.4.5]
by grussell.org (8.18.11) id PDQ666
Sun Nov 15 11:12:20 2008 -0000
Received: (gor@localhost)
by pc236b.napier.ac.uk (8.18.11) id LXY123
Sun Nov 15 11:12:16 2008 -0000

- The first “hop” the email went through was at pc236b.
- The email was written by someone on that machine (user gor).
- Sendmail handled that hop, version 8.18.11

Received

Received: from pc236b.napier.ac.uk [10.2.4.5]
by grussell.org (8.18.11) id PDQ666
Sun Nov 15 11:12:20 2008 -0000
Received: (gor@localhost)
by pc236b.napier.ac.uk (8.18.11) id LXY123
Sun Nov 15 11:12:16 2008 -0000

- The second “hop” the email went through was at grussell.org. It received the email from a server with ip 10.2.4.5.
- Pc236b and 10.2.4.5 should be the same thing.
- It took 4 seconds to be delivered between servers.
- Sendmail handled that hop, version 8.18.11

Spotting forged emails

- You are looking for “funnies” in the headers.
  - Dates and times that go backwards (taking into account the timezone)
  - Hops which don’t match up
  - Strange strings in the hop data
  - DATA and hop data which makes no sense.
  - HOP routes which sound strange (like a bank delivering emails via yahoo).

MX Records

- When you email “linuxzoo.net” the delivery process will look for a MX record for linuxzoo.net.
- If it doesn’t find one, email is directed to the A record.
- If it finds a MX record, email id delivered to the machine described in the MX record.
- This allows a whole domain to delegate email reception to one or more key servers, without having to have email servers on every single possible host.

Linux Email

MUA, MTA, and MDA

- Email in Linux is controlled via three types of services:

  1. MUA – Mail User Agent
  2. MTA – Mail Transfer Agent
  3. MDA – Mail Delivery Agent

MUA

- The email “client”.
- Users use the Mail User Agent to read and send emails.
- It takes email messages which have been delivered to a particular user’s mailbox and displays them to the user.
- It takes new messages and passes these to the MTA for delivery.
- Examples include mutt, mail, and pine.

MTA

- The Mail Transfer Agent is the mail equivalent to an IP router.
- It takes messages given by an MUA or another MTA, and depending on the delivery address passes them onto another MTA or to an MDA for delivery.
- Examples include sendmail, qmail, and postfix.
- Each MTA hop inserts its own data at the start of the email data section.

MDA

- The MDA or Mail Delivery Agent takes email messages from the MTA and delivers it to a particular user or to a MTA.
- Once delivered it is held until an MUA for that user reads the email.
- Examples include mail and procmail for local delivery, and sendmail itself for network delivery.

Example

- Lets consider an example of g.russell@napier.ac.uk delivering an email to me@grussell.org.

1. g.russell MUA on pc236b.napier.ac.uk send the email to the MTA (sendmail) on localhost.
2. The localhost MTA looks up the MX record for grussell.org.
3. The record indicates that email is delivered to grussell.org itself.
4. Sendmail uses its MDA (SMTP) agent to deliver the email to the MTA on grussell.org

5. The MTA on grussell.org looks at the destination user (me) and decides that this is a local user.
6. It uses its local delivery agent MDA (procmail) to put this email into the user's mailbox.
7. The email is stored in /var/spool/mail/me.
8. Next time "me" logs into grussell.org, the email will be waiting for him.

pc236b.napier.ac.uk
MUA
mutt
MTA
sendmail
MDA
(sendmail)
grussell.org
MTA
sendmail
MDA
procmail
mailbox

Discussion

Question 1

With respect to email delivery, discuss the difference between a header and an envelope.

Question 2

- When examining an email header, what header elements could be useful in deciding if it is a forged email. Briefly explain your reasoning.

CSN09703
Networked Services
Essential Apache

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Apache

- Very well known and respected http server.
- Used commercially.
- Freely available from http://www.apache.org
- Plenty of plugins.
- Relatively easy and flexible to configure.
- Fast and Reliable.

Server Architectures

- In most designs of server, you either use
  - Threaded model
  - Forking model
  - Asynchronous Architecture
- A threaded model needs special OS support to provide lightweight threads.
- Forking means that each new request which arrives is handled by a whole process. This is the traditional Apache way.
- Asynchronous. Some web servers exist with this model, where one process handles everything with complex IO code. Good for fast processing of simple web pages. See NGINX.
- Hybrid – Using more than one design simultaneously.

Traditional Forking Model

Forking Initial Settings

StartServers	8
MinSpareServers	5
MaxSpareServers	10
MaxClients	768
MaxConnectionsPerChild	4000

Worker Threaded Forking Model

Worker Initial Settings

StartServers	2
MaxRequestWorkers	150
MinSpareThreads	25
MaxSpareThreads	75
ThreadsPerChild	25
MaxConnectionsPerChild	0

Important Files

- /etc/httpd/conf/http.conf – the main conf file.
- /etc/httpd/conf.d/*.conf – Where your configuration changes go.

- Remember when changing the configurations it is only reread on a server reload or restart.
- Errors and other details are logged by default in /var/log/httpd/ as access_log, error_log, as suexec.log.

Reload or Restart

- Reload is the best option to use.
- With a reload, apache checks your configuration file, and switches to it only if it contains no errors.
- If it has errors, it keeps using the old configuration.
- This allows you to reconfigure a server with no downtime.
- Restart shuts down then starts the server…
- Look in the error log for help (e.g. /var/log/httpd/error_log), or syslog (e.g. /var/log/messages).
- Remember to use the service command for this:
  - systemctl start|stop|reload|restart|status httpd.service
- You can easily make errors in the config file. You can check for errors using
  - httpd -t

Mimic a Browser

- To understand how a sever is running is it sometimes useful to make requests at the keyboard of a server and see the results as text.
- Telnet can do this, so long as you have learned some basic HTTP commands.
- The two important ones are:
  - HEAD – Give information on a page.
  - GET – Give me the whole page.

- In HTTP 1.1 we can use virtual hosts.
- This allows multiple hosts to share a single server.
- Each host has a different name.
- The name of the host you want to answer a query is given as part of a page request.
- This is only supported in HTTP 1.1 and beyond.

$ telnet linuxzoo.net 80
HEAD / HTTP/1.1
Host: linuxzoo.net

```
HTTP/1.1 200 OK
Date: Wed, 15 Oct 2014 10:31:49 GMT
Server: Apache/2.4.6 (CentOS) PHP/5.4.16 SVN/1.7.14
Content-length: 6876
Last-Modified: Tue, 14 Oct 2014 15:40:15 GMT
Content-Type: text/html; charset=UTF-8
```

$ telnet linuxzoo.net 80
HEAD / HTTP/1.1
Host: db.grussell.org

```
HTTP/1.1 200 OK
Date: Wed, 15 Oct 2014 10:32:48 GMT
Server: Apache/2.4.6 (CentOS) PHP/5.4.16 SVN/1.7.14
Last-Modified: Thu, 28 Feb 2008 12:40:43 GMT
ETag: "c81-447373ada48c0"
Accept-Ranges: bytes
Content-Length: 3201
Content-Type: text/html; charset=UTF-8
```

VirtualHosts

- The sharing of a single IP to provide multiple hostnames is well supported in Apache.
- The part of the conf file which handles this is called <VirtualHost>
- Each part holds a list of hostnames it can handle
- The first host found in the file is always considered the default, so if no VirtualHost section matches the first block is done instead.

```
<VirtualHost *:80>
  ServerAdmin me@grussell.org
  DocumentRoot /home/gordon/public_html
  ServerName grussell.org
  ServerAlias www.grussell.org grussell.org.uk
  ErrorLog logs/gr-error_log
  CustomLog logs/gr-access_log combined
</VirtualHost>
```

public_html

- Where apache runs on a server used by many different servers, it would be useful for each user to be able to build their own web pages which the server could serve.
- But the virtualhost configuration takes only a single document root, and each user has their own directories in /home.
- You could make the root /home
  - All of the files in /home would be accessible, not just web pages.
  - It's a bit disgusting…
- Instead, apache supports web pages appearing in a users home directory, under the subdirectory public_html.

public_html access

- Urls of the form
  - http://linuxzoo.net/~gordon/file.html
- Refer to
  - /home/gordon/public_html/file.html
- This feature must first be switched on in httpd.conf.
- To activate it, find the line
  - UserDir disable
- Then either delete the line, or put “#” (the comment character) in front of it.
- Then find the following line and delete the ‘#’ character.
  - #UserDir public_html
- Remember to reload the server.

Linuxzoo tutorials

- Each time you book a linuxzoo machine, you will likely get a different IP and hostname.
- Each time you come in, check your hostname with “hostname”.

```
$ hostname
host-5-5.linuxzoo.net
```

- In this example, virtual hosts vm-5-5.linuxzoo.net, as well as host-5-5 and web-5-5 will be proxied to your machine.
- Warning: If the server on which your virtual machine fails, you will be moved to a different machine and a different IP. You need to check your hostname when you boot!

Web access from the prompt

- The prompt is fast and convenient for admin purposes, but when you are debugging http sometimes “telnet” is not sufficient.
- There are a few other tools you can use at the prompt.
  - elinks
  - lwp-request
  - wget
- However, there is no simple replacement for actually using a real browser to check your pages.

$ elinks http://linuxzoo.net

Edinburgh Napier UNIVERSITY

root@lzmain:/var/named/chroot/var/named

```
                                                          Welcome to linuxzoo
  If you can see this check that

    * Javascript is enabled

  Linuxzoo Penguin Icon

                              Welcome to linuxzoo

  Learn Linux from the safety of your chair using a remote private linux
  machine with root access.

    * Welcome to linuxzoo

          * Our environment
          * Essential Linux
          * System Administration

  Status: Web system is operating normally. FREE server restored and in
  testing. Everything should be back to normal.

  Look at the Our Environment link, and then Running Your Machine for
  getting started.

  Quick start hints: register/login, Join Queue, Switch On (in Control tab),
  Wait for successful boot, click the Connect tab, and then click "telnet:
  linuxzoo.net" (or type telnet linuxzoo.net at your command prompt).
  Username root, password secure.

  áááááá
```

Copy http to your directory

- lwp-request http://linuxzoo.net > file.html
  - The data is obtained and then printed to the screen.
  - In this case that is redirected to file.html
- wget http://linuxzoo.net

```
$ wget http://linuxzoo.net
--19:20:11--  http://linuxzoo.net/
Resolving linuxzoo.net... 146.176.166.1
Connecting to linuxzoo.net|146.176.166.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4785 (4.7K) [text/html]
Saving to: `index.html'
100%[=========================================>] 4,785       --.-K/s   in 0s
19:20:11 (304 MB/s) - `index.html' saved [4785/4785]
```

SELinux and Apache

- SELinux secures apache, and SELinux security of files in public_html is by default quite strong.
- Check if SELinux allows files to be published from public_html by
  - getsebool httpd_read_user_content
  - If this is 0 then publishing files is forbidden.
- Set SELinux to allow public_html publishing using:
  - setsebool -P httpd_read_user_content 1
  - This may take 20 or more seconds. Be patient.
  - The setting will be forgotten if you get a new image in the linuxzoo interface.
- SELinux requires the file security (shown by ls –Z) to be:
  - unconfined_u:object_r:httpd_user_content_t:s0
  - However this should happen automatically provided you create files in public_html
  - You can set the type of say filename.html (but remember you should not have to) using:
    - chcon –t httpd_user_content_t filename.html

Discussions

Discussion

- Apache runs as a user, usually “apache” or “httpd”. For apache to serve a file from a user’s public_html directory, what permissions would be required?

Discussion

- Here are some mock exam questions you should now be able to answer:

Question 1

- To test a web server which is hosting the virtual host “grussell.org”, using only telnet, what would you type at the telnet prompt?

Question 2

What fields would you expect to have to define in a VirtualHost definition in apache?

CSN09703
Networked Services
Essential Firewalls

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Firewalls

Firewalls

- Firewalls and the implementation of Management Policy plus proactive security methods
- Management Policy can include
  - Network Services allowed to run, and under what conditions
  - Client access to external resources
  - Quality of service
  - Security and maintaining availability

Corporate Firewalls

- Cisco ASA
- Cisco router ACLs
- Linux router iptables

- ASA syntax is strange, yet ASA solutions are popular!
- ACLs are weak for policy implementations
- iptables is powerful but uncommon.

Linux Firewalls

- Security issues which Linux firewalls can deal with:
  - Denial of service
  - Differences between external and internal users
  - Hiding local-only services
  - Traffic management
  - Virus and hacking protection
  - Implementing management policy
- Rules are needed in terms of packets arriving, leaving, and passing through.

Packet Information: IP

<table>
<tr><th colspan="4">0
0 1 2 3 4 5 6 7 8 9</th><th colspan="4">1
0 1 2 3 4 5 6 7 8 9</th><th colspan="6">2
0 1 2 3 4 5 6 7 8 9</th><th>3
0 1</th></tr>
<tr><td colspan="2">Version</td><td colspan="2">IHL</td><td colspan="4">TOS/DSCP/ECN</td><td colspan="8">Total Length</td></tr>
<tr><td colspan="8">Identification</td><td colspan="2">Flags</td><td colspan="6">Fragment Offset</td></tr>
<tr><td colspan="4">Time To Live</td><td colspan="4">Protocol</td><td colspan="8">Header Checksum</td></tr>
<tr><td colspan="16">Source Address</td></tr>
<tr><td colspan="16">Destination Address</td></tr>
<tr><td colspan="12">Options</td><td colspan="4">Padding</td></tr>
</table>

- Decisions can be made about packets based on their IP header:
  - Packets from 10.0.0.1 are to be trusted
  - Packets sent to 10.1.1.1 should be blocked
  - ICMP protocol packets should be blocked
  - etc

Packet Information: TCP

- Decisions can be made about TCP packets based on their TCP header:
  - Packets sent to port 80 (http requests) should be allowed
  - Packets sent from port 57890 should be blocked
  - SYN packets with FIN (an invalid combination) should be blocked.
  - etc

iptables

- Since kernel 2.4, the standard interface for firewall configuration is iptables.
- It implements its rules using many “tables”
  - Filter – handles standard “firewall” things
  - NAT – rewriting of source/destination IPs
  - Mangle – specialised hacking of packet info
  - RAW – low-level modifications to packets
- Most firewalls only need to be involved with the filter table

Chains

- Within each table is a number of chains.
- Think of each table as containing different types of rules you might want to use (like “filter” rules).
- Think of chains as defining WHEN a packet will have those rules applied to them.
- Chains are done in a particular order.
- Some packets only go to some chains and not others (depending on how the packet was made).

Chain Names

- Some chain names you might see are
  - PREROUTING
  - INPUT
  - OUTPUT
  - FORWARD
  - POSTROUTING

- Summary of the tables and chains a packet will traverse in iptables.

Source:
http://iptables-tutorial.frozentux.net/iptables-tutorial.html

FILTER TABLE CHAINS: INPUT, OUTPUT, FORWARD

FILTER TABLE

- In the filter table a packet will only go through 1 chain.
  - Packets created within the computer go through OUTPUT
  - Packets which need to be routed from one eth device to another eth device go through FORWARD.
  - Packets received by the computer for processing locally go through INPUT

A Chain

- Each chain is made up of 0 or more rules.
- A rule is a set of tests and an action.
- If the tests are all true then the action is performed.
- If a test is partially or completely false then the next rule is looked at instead.
- If all the rules are used up without an action taking place, then the chain POLICY is done instead (i.e. a default rule).

Tests

- Each rule has 0 or more tests.
- There are many tests possible, such as:
  - Is this TCP?
  - Is this from 10.0.0.5?
  - Is this from port 22?
  - Is this going to port 23?
  - Is this going to ip 50.0.0.1?
  - Am I receiving packets faster than 10 per second?
  - Which interface is it going out/in on?
- Remember you can combine these tests, e.g. TCP and from port 22.

Actions

- If all the tests are true then the action is carried out.
- Some actions are terminating, meaning that once they are done no more rules are looked at.
- A few actions are non-terminating. This is useful if you want to say print a message if the test is true, but continue on with the next rule anyway.

- Example actions are:
  - DROP – delete a packet immediately and terminate
  - ACCEPT – the packet is good and terminate
  - REJECT – delete the packet and terminate, but send back an ICMP message to the sender
  - LOG – print to syslog a message and move onto the next rule.

Some tests:

• Is this TCP ?	-p tcp
• Is this from 10.0.0.5?	-s 10.0.0.5
• Is this from port 22?	--sport 22
• Is this going to port 23?	--dport 23
• Is this going to ip 50.0.0.1?	-d 50.0.0.1
• Is this going out on eth0?	-o eth0
• Is this coming in from eth0?	-i eth0

Default Policy

Rules:
1) Accept Packet “Wanted”
2) Drop Packet “Unwanted”

What about packets where there are no rules which match?

You need a default action…

Default Policy: ACCEPT?

Rules:

1) ~~Accept Packet "Wanted"~~
2) Drop Packet "Unwanted"

With default ACCEPT, all packets are accepted unless you have a rule to DROP or REJECT them.

You must have a rule for any bad packets you could receive.

Default Policy: DROP?

Rules:

1) Accept Packet “Wanted”
2) ~~Drop Packet “Unwanted”~~

With default DROP, all packets are DROPPED unless you have a rule to ACCEPT them.

You must have a rule for any good packets you could receive.

More secure but easier to lock yourself out by accident

Setting the policy

```
$ iptables -P INPUT ACCEPT
$ iptables -P OUTPUT ACCEPT
$ iptables -P FORWARD DROP
```

- This is a typical unsecured machine configuration. Typical machines only have 1 eth device, so don't forward. Otherwise, all packets are allowed.
- Better to have INPUT DROP and have the ACCEPT rules, but remember INPUT DROP without rules then drops everything!

```
$ iptables -P INPUT DROP
```

Editing firewalls

- iptable does allow you to edit firewalls dynamically.
- However, this is very problematic and difficult.
- Instead, I recommend putting all your rules in a file and running that file to change the firewall.
- This allows you to use your favourite editor to write the firewall.
- At the start of the file, delete all current firewall rules in each table using “-F”.

```
$ touch firewall
$ chmod +x firewall
$ vi firewall
/sbin/iptables -F INPUT
/sbin/iptables -F OUTPUT
/sbin/iptables -F FORWARD

# Set the default policies for the chains
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT ACCEPT
/sbin/iptables -P FORWARD DROP
```

- To load these rules do

$ ./firewall

- However, don’t do that yet. The default is DROP for INPUT. Without more rules you will be kicked out of the server never to return…
- This is bad if the server is 5 minutes walk away. But if it is 500miles away you are in trouble!
- This type of firewall is INGRESS ONLY. No rules for going out (OUTPUT/EGRESS). Kind of like the XP firewall…

Discussion

- How would iptables firewall rules, combined with a Linux based router, compare with a custom firewall appliance, such as a Cisco ASA5500?

Discussion

- If a packet arrives at a linux server, and it needs to be delivered onto a local process on that server, then which chain in the filter table would it traverse?

CSN09703
Networked Services
Essential Networking

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

End System Networking

Linux Networking

- Linux is a capable networking platform
- It runs many server applications, so is often seen as a prime platform for server applications.
- It has extensive level 2 and 3 networking support.
- It supports multiple network connections.

- In this course the old network commands, such as ifconfig and route, are deprecated. They may be removed from linux distributions at any time…

Default Networking

- Linux is a system which needs networking in order to work correctly.
- Even a system with no network has networking.
- The basic network is the loopback network.
- Every computer has an IP on the loopback network named *localhost*.

```
> telnet localhost
> telnet 127.0.0.1
> ping localhost
```

localhost

- The IP of localhost is 127.0.0.1
- It operates as a true network, and anything which can be done on a network in linux can operate on the localhost network.
- Linux operates a *priority* networking system, and localhost has the highest priority. If a packet can be delivered using localhost then it will always be delivered with localhost.

The localhost network device

- “lo” is often thought of as the localhost network device.
- It is rarely actually implemented as a /dev device.
- However, all the commands which expect a network device will take lo as a device name.
- It is handled internally in the kernel.

$ ip link show lo

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue
   state UNKNOWN mode DEFAULT
     link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

- LOOPBACK - It’s a loopback device.
- UP – The network is running at layer 2
- LOWER_UP – Active at layer 1
- qdisk – Queueing discipline – How queued things are processed.

The Network Device

- In many systems /dev/eth* is the ethernet network device.
- In such systems with only one network connection, /dev/eth0 is the standard device name.
- Some distributions are sometimes renaming eth0..n to reflect the hardware bus number of the device
  - This makes the name the same no matter how many hardware devices are plugged in later.
  - On my server I have enp3s0 and enp2s0…
- A basic network needs
  - IP number of the host
  - Netmask for the network
  - Gateway IP for the gateway
  - Broadcast address

- The modern way to specify an IPv4 is the normal IP number and a /n value informing you of the netmask.

  10.0.1.20/24

- This indicates:
  - An IP of 10.0.1.20
  - A netmask of the first 24 bits (255.255.255.0)
  - Sensibly a broadcast of 10.0.1.255
  - Sensibly a gateway of 10.0.1.254

$ ip link show ens3

```
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
  pfifo_fast state UP mode DEFAULT qlen 1000
    link/ether 00:0c:11:00:13:90 brd ff:ff:ff:ff:ff:ff
```

- At layer 2, it supports broadcast and multicast.
- Queuing defaults to a prioritised fifo queuing discipline.
- The mac address is 00:0c:11:00:13:90

$ ip addr show ens3

```
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
  pfifo_fast state UP qlen 1000
    link/ether 00:0c:11:00:13:90 brd ff:ff:ff:ff:ff:ff
    inet 10.0.19.17/29 brd 10.0.19.23 scope global
  dynamic ens3
       valid_lft 863827sec preferred_lft 863827sec
    inet6 fe80::20c:11ff:fe00:1390/64 scope link
       valid_lft forever preferred_lft forever
```

- Now can see the layer 3 information.. IP, broadcast, etc.
- Dynamic indicates the address is learned via dhcp.

$ ip -s link show ens3

```
2...
    RX: bytes  packets  errors  dropped overrun mcast
    44073      425      0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    71632      336      0       0       0       0
```

- Various statistics can be obtained which may help with diagnostics.

Ethernet Errors

- Difficult to find out exact meanings, but it is likely that:
  - Errors – CRC Error in packet
  - Dropped – Kernel buffers overflowed
  - Overruns – Card buffer overflowed
  - Frame – Frame length not a multiple of 8 bits
  - Carrier – Probably a fault in the card
  - Collisions – tx collided with another frame

ip address add IFADDR dev DEVICE

- The IFADDR can contain information for setting many things, including:
  - IP
  - Broadcast address
  - Netmask

```
ip address add 10.0.50.10/24 broadcast 10.0.50.255
                    dev eth0
```

Interface selection

- The IP command describes how each device should be used.
- The selection of the interface, and the rules as to what interface is used for each packet, are stored in the routing table
- Linux has an advanced rule-based routing table
  - We will only be doing simple routing
  - We will use the “ip route” command for routing table manipulation.
- It is possible to have a large number of routing tables…
  - Which table to use depends on rules.
  - By default, the rules are simple!

> ip rule show

```
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

- Rules point to tables, which are like subroutines in a program.
- The number is the priority.
- In this case table local is first, then main, then default.
- If the network packet is handled in a particular table, it is not passed on to any other tables.

> ip route show table local

```
broadcast 10.0.19.16 dev ens3  proto kernel  scope link  src 10.0.19.17
local 10.0.19.17 dev ens3  proto kernel  scope host  src 10.0.19.17
broadcast 10.0.19.23 dev ens3  proto kernel  scope link  src 10.0.19.17
broadcast 127.0.0.0 dev lo  proto kernel  scope link  src 127.0.0.1
local 127.0.0.0/8 dev lo  proto kernel  scope host  src 127.0.0.1
local 127.0.0.1 dev lo  proto kernel  scope host  src 127.0.0.1
broadcast 127.255.255.255 dev lo  proto kernel  scope link  src 127.0.0.1
```

- So…all packets are tested in the local table first.
- If a packet is obviously handled using what we know about a link, then make the obvious choice.

> ip route show table main

```
default via 10.0.19.22 dev ens3  proto static  metric 1024
10.0.19.16/29 dev ens3  proto kernel  scope link  src
   10.0.19.17
```

- For an end-network PC, packets are either for the local network or must be forwarded to the gateway.
- Rules move from most specific to least specific.

1. Destination in 10.0.19.16/29, send directly via ens3
2. Other destination? – ask 10.0.19.22 to forward the packet.

Route

- For a simple example of: 10.0.50.10/24:

```
$ ip addr add 10.0.50.10/24 broadcast 10.0.50.255 dev eth0
$ ip route append 10.0.50.0/24 dev eth0 table main
$ ip route append default via 10.0.50.254
```

- Table main is the default, so can be left out of ip route.
- By default, the “default” route works out which device to send on.

Discussion

- Assessment questions

1. Given a desktop Linux machine with an IP of 10.0.1.14/24, and a gateway which is 1 less than the broadcast address, write down the ip commands needed to define the interface and routing table.

CSN09703
Networked Services
Hacking

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

This lecture

- Hacking Techniques
- DNS cache poisoning
- Discussions

Hacking Techniques

Security Concerns

- Security matters to most people
- You want to stop people doing things you do not want them to do with your resources.
- There are different approaches to security.
  - Secure things when problems occur
  - Proactive protection measures

Big Danger

- There is a real danger in providing secure environments – they will become less usable to the real users…
  - Example – stop viruses spreading by blocking outgoing SMTP stops people sending emails without a proxy.
- If users think your security is in the way, they will take steps to bypass your security.
- The ideal security is invisible to real users, but unavoidable to hackers.

Hacking

- Perhaps the best way to understand security is to understand hacking.
- There are black hats, white hats, and some in between…
  - Black hats break systems maliciously
  - White hats break systems without damaging things, and help admins become more secure.
  - Grey hats tend to break “some” systems maliciously, like pornography sites, or break things accidentally, like students doing coursework…

Cost

- All proactive measures, and all active hacking attempts incur an administrative cost.
- Hat colour does not reduce cost.
- Proactive measures are hard to cost-justify.
  - “Yes boss, I did spend a week securing our systems. You wont notice the change, but we are 30% more secure now…”
- Reactive measures are hard to tolerate.
  - “Yes maybe I should have taken security seriously, but how was I to know we were going to be hacked? We have never been hacked before!”

The approach

- There are three approaches…
  1. Social engineering
  2. Brute force
  3. Technical intrusions

- Most attacks are about escalation of privileges.
  - Do not necessary attack ROOT, but get some sort of privilege on the remote machine and use that to become more powerful.
  - Hacking into a basic account then getting more privileges is know as privilege escalation.

Social Engineering

- This is becoming very common.
- Relies on human nature.
- Good examples are:
  - Emails from your bank asking for username and password details.
  - Phone calls from an “administrator”.
  - Visits from offsite technicians.

Hacking BLOB University

- Firstly I get the name of a user and their telephone number.
- I also need a pay as you go phone.
- I pretend to be someone in support.

```
$ whois blob.ac.uk
…
Registrant Contact:
        Jimmy Smith

Registrant Address:
        Director of IT Services
        Blob University
        Blobby House
        +44 141 555 5555
```

Hacking BLOB University

- Its best if Jimmy is not in when you phone. But a Director will never answer their phone anyway…
- Phone your target, “Hi, Jimmy Smith here from Blobby House, your user account may have been hacked as your account has sent obscene messages to a secretary in accounts. Unless you want to admit to it now? Am I talking to the right person? Whats your userid? Blah blah. For auditing reasons, you better stop using your account for the rest of the day. What’s the password, as I need to supply that to the police.”

As strong as the weakest link

- Visit some universities at the start of the year, and you may even see their password policy.
  - “Your surname plus the year of your birth”.
- Passwords for chocolate:
  - http://news.bbc.co.uk/1/hi/technology/3639679.stm
  - When surveyed on the street by attractive “surveyors”, 70% of people gave up their password for a bar of chocolate. In fact, 34% of those questioned gave their password without the offer of chocolate…

Brute Force

- There are plenty of software tools to do this.
- An example of this could be a password cracker testing passwords from a dictionary to gain user access to a site.
- Another example is Denial of Service, with the sheer number of something causing performance degradation.

DoS

- Denial of service is popular.
- Stopping a site doing its work costs real and calculatable amounts of money.
- If a gaming site takes in 100,000 per day, then it may be justifiable to pay someone 50,000 to stop them bringing it down for days in a row.
- Simple DoS is easy to protect against, as all you need is more bandwidth than the attacker.

Example: SMURF

- If you sent a PING to a machine, it pings back a response.
- If you PING a machine, but forge the SRC address to someone you don’t like, this other machine gets the PING replies…
- On some systems, a PING to a.b.c.255 (the network broadcast address) will get all machines on the network to PING back.
- Combine this together and you have an untraceable DoS attack!

Distributed DoS

- If a hacker has taken over a few thousand computers using a trogon or virus, they can build themselves a bot farm.
- On command each machine can launch a hard-to-trace DoS attack on you.
- This can be much harder to block
  - You can block the src address but if this is forged you are stuffed.
  - Best you can do is rate limit syn packets, so that at least legitimate connections are treated normally after the tcp handshake.

Technical Exploits

- Exploit deficiencies in system design, configuration, or management.
- Most involve 5 problem areas:
  1. Inherent security defects
  2. Misuse of legitimate tools
  3. Improper maintenance
  4. Ineffective Security
  5. Inadequate detection systems

Security Defects

- Software is now so complex that all software ships with with unexpected “features”.
- Problems are often reported publicly (e.g. on CERT, SANS, CVS).
- Vendors will eventually release “fixes”.
- Some time later system admins will install the new version…
- Not all problems can be fixed (e.g. protocol weaknesses).

Misusing Tools

- Many useful tools in standard installs can be used to break security if misused..
  - ping – find victims
  - traceroute – find network topologoes.
  - dig – DNS information
  - whois – background information on target.
  - finger – who is logged in.
  - rpcinfo – what rpc services are running
  - showmount – what NFS mounts are exported
  - telnet – play with any TCP protocol service.

> showmount –a orion.napier.ac.uk | grep gor

artemis:/export/home/o2/staff/gor

pc236nt:/export/home/o2/staff/gor

> mount –t nfs orion.napier.ac.uk:/export/home/o2/staff/gor /mnt/a

- I have access to all I: drive files for all users.
- No password required.
- Moral: do not use I: for your vital stuff…

Improper Maintenance

- An example of this could be firmware in a router not be updated, or critical updates to a system being missed.
- Lack of time is often the cause.
- Lack of priority may also be an issue.
- Perhaps highlights the need to prioritise security matters as determined by a risk assessment.

Ineffective Security

- Perhaps caused by having no security policy…
- An example could be spending all of ones time on firewall management, and then leaving the root password as “rootroot” for speed.
- Also causes by conflicts between users and admin people.
  - Users want CGI and write scripts which bypass admin security.

Detection

- Many sites rely on audit trails to detect problems.
- This does nothing to detect Trojans, backdoors, and viruses.
- New tools on the market to detect more subtle problems.
  - For example, checksums of system files checked against remote records.

The Process

- A dedicated hacker will have many targets on the go at once, all in different stages of being hacked.
- A commonly held list of stages is:
  1. Casing
  2. Scanning
  3. Enumeration

Casing:

- Gather information on the target.
- Often called fingerprinting.
  - IPs, services running, routing tables, domain information, authentication scheme, user details, admin names, contact information, telephone numbers, connection type, etc.
  - For instance, has an admin discussed their firewall configurations on a newsgroup or admin forum? Could be interesting…

Scanning

- In the scanning phase, individual machines are identified by direct communication.
- Tools are available to tell you about OS type, open ports, firewall configurations, and even version numbers.
- Scanning should also involve the routers and firewall devices, as these may be remotely configurable.

Example: nmap

> nmap linuxzoo.net

PORT	STATE	SERVICE
22/tcp	open	ssh
23/tcp	open	telnet
53/tcp	open	domain
80/tcp	open	http
81/tcp	open	host2-ns
123/tcp	closed	ntp
5900/tcp	closed	vnc
5901/tcp	closed	vnc-1
5902/tcp	closed	vnc-2
5903/tcp	closed	vnc-3

Enumeration

- This really covers getting some sort of “access”.
- It could be discovering a username and then a password (brute force perhaps).
- It could be a badly written NFS or other share.
  - NULL Shares
  - Zone transfers

Failed SSH logins

- /var/log/secure contains SSH attempts.
- Failed attempts can be a sign of a problem:

head -3 /var/log/secure

```
Failed password for root from ::ffff:219.232.?.? port 40731 ssh2
Illegal user eaguilar from ::ffff:61.129.?.?
Failed password for illegal user eaguilar from ::ffff:61.129.?.? port 53785 ssh2
```

- Not exactly readable… “?” inserted to spare embarrassments.

A little perl

```
#!/usr/bin/perl

open(my $file,"</var/log/secure.1");
my %ip;
foreach my $line (<$file>) {
  if ($line =~ m/Failed/) {
    if ($line =~ m/(\d+\.\d+\.\d+\.\d+)/) {
      $ip{$1}++;
    }
  }
}

foreach my $k (keys %ip) {
  my $what = `dig -x $k | grep PTR`;
  my $dig = "?";
  $dig = $1 if ($what =~ m/PTR\s+([^\s]+)$/);
  print "$k \t: $ip{$k} fails : $dig\n";
}
```

The data

202.118.?.? : 73 fails : ?

140.125.?.? : 1 fails : ?.?.edu.tw.

61.129.?.? : 23 fails : ?

78.110.?.? : 239 fails : ?

152.104.?.? : 24 fails : static-ip-?.rev.dyxnet.com.

219.232.?.? : 1 fails : ?

- Not everyone has a PTR record. The 78.110.?.? has a lot of attempts.
- Even without PTR, we can use WHOIS or RIPE to find out.. It comes to a dedicated server company in the UK.

Next Step

- Cannot really take action.
- Even UK site is probably the result of being hacked itself.
- Some people have these scripts add iptables rules so that more than “n” failed logins blocks the IP for a certain period of time.

Any successful login is the end…

- Once a hacker can actually run commands on a machine, the end is likely to be close…
- There are a great many bugs in normal everyday system commands which allows people to become root, access things they are not allowed to, and to modify things we would rather they didn't.
- Security ends with a successful login.

Discussion

Question 1

You have detected 15 failed ssh logins from 10.0.0.1. What should you do next and why?

Question 2

A firewall audit using nmap was performed, and produced the following results:

PORT	STATE	SERVICE
22/tcp	open	ssh
23/tcp	open	telnet
53/tcp	open	domain
80/tcp	open	http
123/tcp	closed	ntp

Comment on the quality of your server security.

CSN09703
Networked Services
Domain Name Service - DNS

Module Leader: Dr Gordon Russell

Lecturers: G. Russell, P. Leimich

DNS

Basics

- DNS – Domain Name Service
- Translates between machine names and IP.
- Two main types
  - Forward (domain to IP translation)
  - Reverse (IP to domain translation)

Terminology

- Zone
  - A collection of hostnames and their IPs
- Nameserver
  - The server which responds to DNS queries. A question could be “Give me the IP of grussell.org”.
- Authoritative Nameserver
  - The server which has all the information for a zone stored locally
- Recursive Nameserver
  - If the nameserver is not authoritative for a zone, it is willing to go and ask other nameservers until it has asked an authoritative nameserver for the answer on your behalf. It then tells you the answer to your query.

- Resolver
  - The part of an OS which sends the DNS questions to nameservers. It's a library which other programs will use. For instance, "ping grussell.org" would ask the resolver to "resolve" grussell.org. It goes on to ask a nameserver for the answer.
- Delegation
  - Sometimes a server does not know how to answer a query, but knows a server that can. The process of delegation effectively says that another server is delegated to answer your query, and you need to speak to them instead.
- Resource Record (RR)
  - Part of an answer to a query. An answer could be the IP for grussell.org, but there are other resource records (e.g. for email delivery and delegation).

WHOIS

- When you register a domain, you have to give information to the registrar.
- This includes a contact name, address, and other contact details.
- You also have to give at least 2 authoritative nameservers.

```
$ whois napier.ac.uk
Registered For:
        Napier University
Servers:
        dns0.napier.ac.uk       146.176.1.5
        dns1.napier.ac.uk       146.176.2.5
Registrant Address:
        Napier University
        C&IT
        219 Colinton Road~Edinburgh
```

DNS Distributed Database

- By way of an example, consider the following:

  $ ping www.napier.ac.uk
- The resolver in Linux is asked to find out the IP for www.napier.ac.uk
- The resolver contacts its local recursive nameserver and send it a DNS query.
- The resource record needed to translate a domain name into an IP is known as an “A” record.

- Step 1: The resolver contacts the local ISP nameserver.
- It is not authoritative. If it has been asked before it might have cached the answer.
- In this case, no cached answer…

- Step 2: ISP nameserver asks a root server...
- There are more than a dozen root servers.
- Their job is to direct your local nameserver to a nameserver which will help resolve the request.
- The root servers are pre-configured in the local nameserver, with names like:

a.root-servers.net.

b.root-servers.net.

c.root-servers.net.

d.root-servers.net.

- Step 2: ISP nameserver asks a random root server...
- It doesn't know either, but offers a referral to nameservers which can help.

- The root server refers to another set of servers which can answer the query:
- NS are nameserver resource records.
- Note that you also get the A records of the nameservers for free.
- This extra information is referred to as the glue records.
- Without this we would have to look them up with a different query, so this saves time…

```
uk.             172800  IN      NS      ns1.nic.uk.
uk.             172800  IN      NS      ns2.nic.uk.
uk.             172800  IN      NS      ns3.nic.uk.
uk.             172800  IN      NS      ns4.nic.uk.
;; ADDITIONAL SECTION:
ns1.nic.uk.         172800  IN      A       195.66.240.130
ns2.nic.uk.         172800  IN      A       217.79.164.131
ns3.nic.uk.         172800  IN      A       213.219.13.131
ns4.nic.uk.         172800  IN      A       194.83.244.131
```

- Step 3: ISP nameserver asks one of the referral nameservers at random.
- Another referral.
- Receive NS list for ac.uk nameservers

- Step 4: ISP nameserver asks one of the referral nameservers at random for.
- Another referral.
- Receive NS list for napier.ac.uk nameservers

- The ja.net nameserver responded with the authoritative nameservers for Napier.
- The last step is to ask one of the authoritative nameservers for www.napier.ac.uk

```
napier.ac.uk.        86400   IN    NS    dns1.napier.ac.uk.
napier.ac.uk.        86400   IN    NS    dns0.napier.ac.uk.

;; ADDITIONAL SECTION:
dns0.napier.ac.uk.   86400   IN    A     146.176.1.5
dns1.napier.ac.uk.   86400   IN    A     146.176.2.5
```

- Step 5: ISP nameserver one of the napier nameservers
- Finally it gets the answer…

- The answer from Napier was AUTHORITATIVE.
- It is cached in the ISP.
- The cached version is returned to you.
- As it is cached, it is non-authoritative.

Manual Lookups

- If we are running our own DNS servers, or just having trouble making the resolver work, we can make our own queries using “dig”

```
$ whois napier.ac.uk
 Domain servers in listed order:
 dns0.napier.ac.uk 146.176.1.5
 dns1.napier.ac.uk 146.176.2.5
```

> dig www.napier.ac.uk @dns0.napier.ac.uk

```
www.napier.ac.uk.        86400   IN      A       146.176.222.174

;; AUTHORITY SECTION:
napier.ac.uk.            86400   IN      NS      dns0.napier.ac.uk.
napier.ac.uk.            86400   IN      NS      dns1.napier.ac.uk.

;; ADDITIONAL SECTION:
dns0.napier.ac.uk.       86400   IN      A       146.176.1.5
dns1.napier.ac.uk.       86400   IN      A       146.176.2.5
```

Reverse Lookup

- Reverse is working out that given 146.176.222.174, the host is www.napier.ac.uk.
- A special domain name is used for IP to Domain Name translation
- The domain is the IP in reverse, ending with IN-ADDR.ARPA
- You need to take the first 3 elements of the IP first to find the right server, then query that server with the full IP.
- The resource record for reverse DNS is PTR.

```
> dig 222.176.146.IN-ADDR.ARPA
…
;; AUTHORITY SECTION:
222.176.146.IN-ADDR.ARPA. 86400 IN      SOA     dns0.napier.ac.uk.
    root.central.napier.ac.uk. 200808271 28800 7200 604800 86400

> dig 174.222.176.146.IN-ADDR.ARPA @dns0.napier.ac.uk -t any
…
;; ANSWER SECTION:
174.222.176.146.IN-ADDR.ARPA. 86400 IN  PTR     www.napier.ac.uk.
```

Question

- Given the IP number 50.60.70.80, what hostname would be queried to perform a reverse lookup on this IP?
- What RR would be expected from this?

CSN09703
Networked Services
SELinux

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

SELinux

Linux security model

- Linux by default uses DAC for its file access model
  - DAC – Discretionary Access Control
- DAC is based on UID or GID and uses file access controls based on RWX
  - DAC also includes finer-grained ACLs.
- DAC has some issues…
  - If permission is granted to a user then this is independent of user behaviour. So for instance if a good process was hacked in some way, it can continue to access what it could before even if it was operating in a different way.
  - ACLs are time consuming to maintain
  - Grant and revoke of permissions requires finding all the files related to the change
  - Cannot easily stay negative statements, i.e. what are you not allowed to access.
  - Non file-based permissions are tricky to manage… e.g. what ports can be opened by a process, what permissions do child processes have, or what processes can a process talk to.

SELinux

- SELinux was originally developed by the NSA (National Security Agency).
- It is an implementation of the Flask operating system security architecture…
- SELinux implements MAC
  - MAC – Mandatory Access Control
  - MAC is a centrally managed security policy based on objects.
- Ultimately the idea is to give processes “least privilege”, i.e. only exactly what that process needs to get the job done.
- Each file is given an SELinux label, using xattr.
- Each process is also labelled.
- Basically, SELinux holds rules about what process labels can do to each file label.

SELinux Decision Process

- https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-selinux.html

Modes of Operation

- SELinux can be run in
  - Enforcing Mode (default)
  - Permissive Mode
  - Disabled
- SELinux can also operate in a few types of enforcement
  - Strict
  - Targeted (default)
  - Minimum
  - mls

Permissive Mode

- Often seen in advice “run in permissive mode to fix this”…
- Permissive mode does not enforce the rules, but does report when it would normally stop an action…
- There is a danger however that while in permissive mode the file labels could be changed into a state which is unwanted:
  - If a process tried to create a file but choses the wrong file label, permissive mode will allow this to happen.
  - If you then return to enforcing mode, the new file will have the wrong label

- cat /sys/fs/selinux/enforce
  - 1 indicates currently enforcing, 0 otherwise.
- getenforce Reports “Enforcing” if enforcing…
- setenforce 0 Changes mode to permissive
- setenforce 1 Changes mode to enforcing

Permissive Mode

- Usually best to fix things without leaving enforcing mode, but people find this hard!
- You can create the file /.autorelabel then reboot.
  - On next boot the whole filesystem will be relabelled with the default labels for each file
  - Default labels may not be what you want, and may override legitimate changes, but labels will now be at least strictly correct.
  - So if on boot default is enforcing, then at least the labels will work (e.g. the system should be bootable).

Security Concepts

- Subjects
  - The thing performing an action. In Linux these are processes.
  - Since processes are related to users, you can consider users subjects too.
- Access
  - What access is permitted, such as read, write, append, delete, open, change…
- Object
  - The resource on which the action applies

Subjects use Access on Objects.

Domain

- In SELinux, a label assigned to a process is also called a domain.

```
$ ps -Zx | grep sshd
system_u:system_r:sshd_t:s0-s0:c0.c1023 2141 ? Ss     0:00 /usr/sbin/sshd
```

SELinux user : SELinux role : SELinux type [: SELinux MLS]

So here the domain is `system_u:system_r:sshd_t`

For convenience, this is often shortened to type part, `sshd_t`

Type

- A type is a label for an object, such as a file.

```
# ls -Z /etc/ssh/ssh_host_ecdsa_key
system_u:object_r:sshd_key_t:s0 /etc/ssh/ssh_host_ecdsa_key
```

- When making decisions about an action, SELinux considers the domain and the type, but also the class.

- An object class is the description about what sort of object it is. For instance, a file, a directory, a soft link, etc. This allows for fine grained decision making.

SELinux: the processing

- SELinux makes decisions from the AVC (access vector cache).
- If the subject/type/action pair is not in the AVC, it is loaded from the policy database first.
  - If the ACTION is ok in the cache for a subject/type, then the ACTION is allowed.
  - If the ACTION is not ok then an AVC: Denied message is generated.

- https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-selinux.html

MLS

- Sensitivity is the 4th field in a domain or type:

`system_u:system_r:sshd_t:s0-s0:c0.c1023`

- The fourth field has two concepts: sensitivity and category.

- Sensitivity is equivalent to document classifications like internal, secret, top secret, etc, represented by a number.
  - s0 is the least secret, and higher numbers represent more secrecy
- Category relates to concepts like department names or project titles. Again these are just numbers like c0 or c43
  - Unlike sensitivity c1 is not more or less than c0, just different.

- MLS information can be written in a number of ways:

```
system_u:system_r:sshd_t:s0-s2:c0.c1023
```

- Sensitivity of s0 to s2 and all categories from c0 to c1023

```
system_u:system_r:sshd_t:s0
```

- Sensitivity of s0 and a default categories from c0

```
system_u:system_r:sshd_t:s0:c5,c9,c44
```

- Sensitivity of s0 and membership of categories c5,c9, and c44.

MLS in practice - dominance

- If a process has all the categories of the object (or more), and a sensitivity equal to or more than the object, then it is said to *dominate*. Known as *dom*.
- If a process has some (at least 1) but not all of the categories, and a sensitivity equal to or less than the object, then it is said to be *dominated*. Known as *domby*.
- Process has the same sensitivity and categories as the object. Known as *eq*.
- Process has categories which have nothing in common with the object it is said to be *incompatible*. Known as *incomp*.

- This is then used in an MLS policy. Such a policy might be that:
  - Read operations allowed if the domain dominates the object
  - Write operations are allowed if the domain is dominated by the resource

MLS – The last word

- MLS policy is not currently used in SELinux by default.
- Type enforcement (i.e. using the other three fields of the label) is much more common, and MUCH easier.
- However, you can expect to see the MLS fields appear wherever SELinux is used.
- Safe to ignore ☺ (for now…)

SELinux Users

- Domains have an SELinux user entry
  - Provides functionality which relates to something users cannot change themselves
  - Each Linux user is in reality also an SELinux user, based on a mapping

```
# semanage login -l

Login Name           SELinux User         MLS/MCS Range
Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

unconfined

- Unconfined in SELinux indicates that SELinux will not enforce any rules in policing the interactions
- In effect, if an user is unconfined, they operate largely as if SELinux is disabled.
- Generally, if a user or process becomes unconfined, then there should be no way back for the process to later become “confined”.

- In the targeted mode, systems generally use SELinux to enforce system-level concepts, such as daemons.
  - Users are left unconfined.

Roles

- Roles in SELinux are what actually carry permissions for users
  - Roles get permissions
  - Users get roles

```
# semanage users -l

SELinux User    ...         SELinux Roles

root                        staff_r sysadm_r system_r unconfined_r
staff_u                     staff_r sysadm_r system_r unconfined_r
system_u                    system_r unconfined_r
unconfined_u                system_r unconfined_r
user_u                      user_r
```

Default Roles

Role	Description
user_r	End user role meant to be assigned to users with interactive logon privileges to the system, but without additional role requirements
staff_r	End user role meant to be assigned to users who also need to transition to other roles. The staff_r role by itself isn't much more privileged than user_r except that it is allowed to do role transitions
sysadm_r	Powerful role for generic system administration
system_r	System role meant for daemons and system services

Users to roles to domains

Example: the Apache daemon

- The Apache daemon, httpd:

```
$ ps auxZ | grep httpd
system_u:system_r:httpd_t:s0   /usr/sbin/httpd
```

When Apache wants to log errors, it writes it to /var/log/httpd/error_log

```
$ ls -Z /var/log/httpd/error_log
-rw-r--r--. system_u:object_r:httpd_log_t:s0 error_log
```

Allow

- For httpd_t to access httpd_log_t, it needs a rule which Allows this:

```
$ sesearch --allow --source httpd_t –-target
      httpd_log_t --class file


allow httpd_t httpd_log_t : file { lock append open …} ;
```

SELinux ports

- SELinux also has rules about what types can open which network ports.
- Again, for httpd, it would naturally want to use port 80

```
# sesearch -s httpd_t -c tcp_socket -AC -p name_bind
allow httpd_t http_port_t : tcp_socket name_bind ;

# semanage port -l | grep http_port_t
http_port_t     tcp      80, 81, 443, ...
```

Transitions

- Processes can move from one type to another by executing a program of that new type
  - Of course there must be a rule which permits that transition

- For example, the init process which starts all daemons has type init_t.
- You may wish to run a command in /etc/init.d so that you end up running that daemon unconfined.
- In this case a special type exists, unconfined_service_t, which has no transitions back to a confined state

- To find the right type:

```
$ sesearch -T -s init_t | grep unconfined
type_transition init_t bin_t : process unconfined_service_t;
type_transition init_t usr_t : process unconfined_service_t;
```

So if the file /etc/init.d/oracle has either bin_t or usr_t, the process will become type unconfined_service_t.

```
$ ls -Z /etc/init.d/oracle
-rwxr-x---. system_u:object_r:bin_t:s0       oracle
```

Booleans

- Some rules are considered part of the base rules.
- However, other rules can be switched on or off using Booleans
  - Adding more rules usually means adding more ways to access something
  - More access could mean less security
  - So you should only enable Booleans when they are actually needed.
- So for instance, to enable httpd to read stuff in /home/*, this feature needs to be switched on:

```
$ setsebool -P httpd_enable_homedirs 1
```

Of course you do need to know which bool you need ☺

- httpd_enable_homedirs can be explored using sesearch.
- You can discover the type of /home/demo:

```
$ ls -Z /home/demo
unconfined_u:object_r:user_home_dir_t:s0 vmd

$ sesearch --allow -s httpd_t -b httpd_enable_homedirs
…
allow httpd_t user_home_dir_t : lnk_file { read getattr } ;
…
```

SELinux Administration

- OK so SELinux is very complex…
- Fortunately, the low-level rules and transitions are generally things never needing touched, but it is good to understand the basics of how it works.

- In general terms, all that is generally needed is
  - An ability to set labels to suit particular scenarios
  - An understanding of the error messages when things go wrong
  - Some high-level configuration via the booleans

Discussion

- How does SELinux control process access to tcp ports? List the commands you would use along with their parameters to explain what ports a process would be permitted to use.

CSN09703
Networked Services
Apache Log Analysis

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Log Analysis

Logs

- Apache produces two types of log files
  - Error Logs
  - Access Logs
- Error logs are useful for debugging
- Access logs are excellent for monitoring how your site is being used.
  - Fun for people who have hobby sites
  - Life or death if your business relies on the web site.

Where are the logs

- Normally they go to /var/log/httpd/access_log and error_log
- In a virtual host we set them to what we liked:

```
<VirtualHost>
…
    ErrorLog logs/gr-error_log
    CustomLog logs/gr-access_log combined
</VirtualHost>
```

Logging in /var/log/http access file

- The normally used log format is called “combined”.
- It contains significant amounts of information about each page request.
- Specifically, the log format is:

%h %l %u %t %r %>s %b Referrer UserAgent

%h %l %u %t %r %>s %b Referrer UserAgent

- h – IP of the client
- l – useless ident info
- u – username in basic authentication
- t – time of request
- r – the request itself
- s – The response code (e.g. 200 is a successful request)
- b – size of the response page
- Referrer – who the client things told it to come here
- User Agent – identification info of the browser

Analysing the log

- The log is useful in itself for checking the proper function of the server.
- However, traffic analysis is also valuable.
- There are a number of tools available to do this.
- One of the best free ones is webaliser.

Webaliser Summary

Analysis

- The summer is quiet for linuxzoo.
- Students are enthusiastic in October…
- After that it settles down to “kept busy”.

Per day activity – October

- I wonder which day was the first tutorial?
- Look at the 7 day oscillations. This is common in many web sites.
- Who stole all my web site data on the 25th?

Hour analysis – October

- Peak learning time (so they say) is 11am.
- Students here seem to like 9am-4pm.
- American students produce another bump later at night.

Edinburgh Napier UNIVERSITY

Users

#	Hits		Files		KBytes		Visits		Hostname
1	25319	9.79%	24605	10.69%	94474	2.07%	1051	6.33%	gtw-12.nhs.uk
2	16906	6.54%	16906	7.35%	14933	0.33%	1	0.01%	dsl081-020-069.nyc1.dsl.speakeasy.net
3	6400	2.48%	6381	2.77%	32857	0.72%	122	0.73%	200.182.252.4
4	6005	2.32%	5887	2.56%	24881	0.54%	245	1.47%	200.217.233.139
5	4903	1.90%	2230	0.97%	8144	0.18%	39	0.23%	mvx-200-196-55-148.mundivox.com
6	4506	1.74%	4506	1.96%	6580	0.14%	4	0.02%	200-221-128-3.corp.uolinc.com
7	3679	1.42%	9	0.00%	20	0.00%	2	0.01%	59.145.136.1
8	2728	1.06%	2726	1.18%	10462	0.23%	47	0.28%	59.163.124.54.static.vsnl.net.in
9	2690	1.04%	2590	1.13%	4733	0.10%	103	0.62%	58.68.28.66
10	2647	1.02%	2645	1.15%	11453	0.25%	193	1.16%	glenlivet.spc.eeng.liv.ac.uk
11	2479	0.96%	2455	1.07%	11568	0.25%	69	0.42%	mx2.queirozgalvao.com
12	2381	0.92%	2376	1.03%	50698	1.11%	1	0.01%	200.4.171.30
13	2156	0.83%	270	0.12%	2494	0.05%	6	0.04%	186.112-84-212.staticip.namesco.net
14	1657	0.64%	1657	0.72%	5821	0.13%	8	0.05%	80-41-249-174.dynamic.dsl.as9105.com
15	1606	0.62%	1601	0.70%	7577	0.17%	42	0.25%	146.176.242.35
16	1479	0.57%	347	0.15%	2545	0.06%	6	0.04%	r200-40-197-174.static.adinet.com.uy
17	1478	0.57%	1478	0.64%	0	0.00%	1476	8.89%	host.avidnetwork.com
18	1323	0.51%	1056	0.46%	12408	0.27%	3	0.02%	80-192-78-217.cable.ubr13.edin.blueyonder.co.uk
19	1157	0.45%	1155	0.50%	5840	0.13%	42	0.25%	146.176.242.54

Referrer Info

Top 30 of 831 Total Referrers

#	Hits		Referrer
1	45948	17.77%	- (Direct Request)
2	1774	0.69%	http://www.google.com/search
3	425	0.16%	http://mail.google.com/mail/
4	343	0.13%	http://www.dicas-l.com.br/dicas-l/20061005.php
5	323	0.12%	http://www.google.co.uk/search
6	182	0.07%	http://www.google.ca/search
7	151	0.06%	http://en.wikipedia.org/wiki/System_administrator
8	142	0.05%	http://www.google.co.in/search
9	140	0.05%	http://www.ligeirinhorj.blogger.com.br/
10	135	0.05%	http://sqlzoo.net/
11	113	0.04%	http://146.176.165.229/my-netlab-s.cgi
12	99	0.04%	http://www.stumbleupon.com/refer.php
13	97	0.04%	http://www.google.com/linux
14	92	0.04%	http://www.google.com.au/search
15	79	0.03%	http://grussell.org/
16	78	0.03%	http://www.dicas-l.com.br/
17	73	0.03%	http://www.google.de/search
18	66	0.03%	http://www.google.nl/search
19	55	0.02%	http://www.grussell.org/

What search terms?

Top 20 of 1160 Total Search Strings			
#	Hits		Search String
1	176	8.17%	umask 022
2	68	3.16%	linuxzoo
3	46	2.13%	unix file structure
4	45	2.09%	apache2 authentication
5	31	1.44%	rndc.key
6	30	1.39%	apache basic authentication
7	27	1.25%	apache public_html
8	20	0.93%	generate rndc key
9	18	0.84%	linux zoo
10	18	0.84%	named service
11	16	0.74%	iptables -f
12	16	0.74%	mysql replicate
13	16	0.74%	symbolic link example
14	15	0.70%	apache2 basic authentication
15	15	0.70%	setting umask
16	14	0.65%	apache basic auth
17	13	0.60%	named forward
18	12	0.56%	rndc key

Where from?

Usage by Country for October 2006
Unresolved/Unknown (24%)
United Kingdom (23%)
Brazil (16%)
US Commercial (15%)
Network (13%)
India (2%)
Uruguay (1%)
US Educational (1%)
Other (5%)

Google Analytics

- Another approach to web logging is to use JavaScript embedded in each web page.
- This does away with the need to access the web log.
  - Good if you don’t have access!
- It does mean that
  - You only get logs where there is javascript switched on.
  - Each page is slowed by having extra stuff on it.
  - It’s a little more complex.

Edinburgh Napier UNIVERSITY

Analytics Settings | View Reports: db.grussell.org | My Analytics Accounts: ActiveSQL

- Dashboard
  - Saved Reports
- Visitors
- Traffic Sources
- Content
- Goals

Settings
- Email

Help Resources
- About this Report
- Conversion University
- Common Questions

Dashboard

Jul 28, 2008 - Aug 27, 2008
Comparing to: Site

Export | Email

Graph by: Day | Week | Month | Visits

Site Usage

11,998 Visits	55.00% Bounce Rate
60,750 Pageviews	00:04:18 Avg. Time on Site
5.06 Pages/Visit	77.18% % New Visits

Visitors Overview

Map Overlay

11,998 visits came from 141 countries/territories

Logging Summary

- What is best?
- I have used both and have mixed feelings…
- Things to consider
  - Convenience
  - Reliability
  - Availability
  - Performance
  - Cost
  - Privacy
  - Complexity

Discussion

Question 1

- Below is a line from a webserver logfile:

157.55.18.25 - - [31/Aug/2011:12:48:04 +0100] "GET /robots.txt HTTP/1.1" 200 48 "-" "Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm)"

- What kind of request was this? Was this a successful request (i.e. was a document found)?

CSN09703
Networked Services
Exploits – heartbleed

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Apache certificate leakage

heartbleed

Apache weakness due to OpenSSL

- This is an issue which caused Apache to leak SSL certificates to the internet…
- It was caused by a weakness in OpenSSL, which Apache uses to manage its HTTPS connections.
- The buggy version of OpenSSL was released in March 2012.

- As a result of this OpenSSL has been put through the wringer quite a bit, and now there are options available instead of OpenSSL. But it is still popular!

Timeline

• OpenSSL released	March 2012
• Reported Vulnerability	1 April 2014
• Patch Released	21 March 2014
• First proven exploit attempt	8 April 2014

- To be fair, some unofficial fixes had been released before the patch was available…

Background – the Heartbeat Extension

- OpenSSL introduced an heartbeat extension.
- This enabled an endpoint of a secure session to detect whether the peer is still operating
  - Sort of an encrypted ping.

heartbeatRequest

- To run the “ping”, one endpoint send a heartbeatRequest message.
- In response, the peer responds with a heartbearResponse.

Example

SERVER, ARE YOU STILL THERE? IF SO, REPLY "POTATO" (6 LETTERS).
secure connection using key "4538538374224"
User Meg wants these 6 letters: POTATO.
da wants pages about "irl games". Unlocking
ecure records with master key 5130985733435

The problem

- The protocol is sound, but when it was implemented the developers assuming that the heartbeatRequest payload length value would be honest…
  - Consider if a request was send and the length was 100, but the actual length in the request was only 1.
  - In the response, the peer would return the 1 byte which was originally sent, and then another 99 bytes from the memory after that.
  - This memory could belong to anything, and it would not be zerod since it was last used, so it could hold details of a previous request or any other sort of internal data structure.
  - The length could be set up to 64k, allowing a hacker to peer into around 64k of memory space which it shouldn’t have been able to access… woops.

Example

SERVER, ARE YOU STILL THERE? IF SO, REPLY "HAT" (500 LETTERS).
User Meg wants these 500 letters: HAT.
User Meg wants these 500 letters: HAT.

The result

- Provided you sent the bad query often enough, then there is plenty of opportunity to look at a whole range of private memory variables used by apache to do things internally.
- Surprisingly, OpenSSL used its own memory management functions, and this may have had the effect of making useful private data more likely to be accessible.
- In general, repeated queries could turn up anything:
  - Passwords submitted by another user
  - Personal data on a page which another user was sent
  - A slew of session cookies
- If it was done often enough, the private SSL keys for the server could also be obtained, leaving sites open to man in the middle attacks.

The patch

- Compare the payload length field to the actual length of the payload
- Discard the request if payload length > actual length of payload.

- OK, it was a little more complex, as the request packet length was stored in one location, whereas the length held within the request was accessed at a different level of abstraction in the code. Still easy though.

How fast were servers patched?

- Once the vulnerability was reported:

Patching behaviour over time

Reality..

- What should have happened is a 3 stage process:

1. Patch OpenSSL
2. Create a new private key and generate a new certificate
3. Revoke the old key.

However, by the 9/5/2014:

Conclusion

- Simple errors can have big consequences
- Just patching without understanding the risks is problematic
- Patching can be slow…

Discussion

Question 1

Consider a server which was active for the duration of the heartbleed bug. If this server was for a bank, discuss the risks if the only action was to install the patch and reboot.

CSN09703
Networked Services
Linux as a Router

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Linux as a “Router”

- If Linux has more than 1 network connection, it can perform layer 3 routing, just like a Cisco router.
- Cisco routers often have only 2 or 3 network connections, and it is easy to build a PC to replicate this.
- Cisco argue that their routers are far superior…

- Physically, Cisco routers have ASICs that make routing fast, with various hardware “things” to cache decisions and process data.
  - For generic routing Cisco will most likely outperform Linux.
- But physical routers may need other things, like services, firewalls, IDS and IPS, and advanced traffic management features.
  - Such features may be better handled by Linux, but with a performance/cost ratio lower than hardware devices.

Ref: http://www.linuxjournal.com/node/5826/print

Configuration

- Multiple networks are no different from single network configurations.
- You need “ip address” for each interface.
- You need a route for each interface
  - Some configuration approaches may add route information automatically. But you should still be able to do it manually, and understand what is going on, as the automation may not be right. It is also good for exam questions!
  - In the tutorials the routes do not seem to be automatically added...
- You also need 1 default route.

- General Rule: If you have “n” ethernet devices, you need “n” ip route commands (one per device) PLUS 1 route for the default gateway.

Example: Simple Gateway

Add this example

```
> ip addr add 10.0.50.10/24 broadcast 10.0.50.255 dev eth0
> ip route append 10.0.50.0/24 dev eth0
> ip addr add 10.0.1.254/24 broadcast 10.0.1.255 dev eth1
> ip route append 10.0.1.0/24 dev eth1
> ip route append default via 10.0.50.254
```

> ip route show

10.0.50.0/24 dev eth0 scope link

10.0.1.0/24 dev eth1 scope link

default via 10.0.50.254 dev eth0

Class Exercise:

1. List ip addr commands for eth0, eth1, eth2
2. Show ip commands in the main table

The netmask

- The netmask can be any size from /0 to /32.
- Perhaps you considered only /8, /16, /24 masks.
- These are fixed-length masks, matching the IP type (like Class A, B, etc).
- Complex networks use variable-length subnet masks.

VLSM

- Variable length subnet masks:
- Subdivide the host part of the network mask into smaller pieces.
- Each subdivision has its own network
- So if you need to run 2 networks, but only have 10.1.1.0/24, you can create 2 networks as:
  - 10.1.1.0/25
  - 10.1.1.128/25
- Remember that first and last host is reserved for “network” and “broadcast”. Thus you cannot use 10.1.1.0 or 10.1.1.127 or 10.1.1.128 for host addresses.

VLSM is “borrowing bits”

- Problem: You need 5 networks, but you only have 10.10.10.0/24.
- You cannot split into an number of networks which is not a power of 2 (2,4,8,16,etc), so split into 8.
- 8 needs 3 bits in binary (000-111 is 8 combinations)
- So borrow 3 bits from /24, making it /27.
- The new network numbers are:
  - 10.10.10.0/27 10.10.10.32/27
  - 10.10.10.64/27 10.10.10.96/27
  - 10.10.10.128/27 10.10.10.160/27
  - 10.10.10.192/27 10.10.10.224/27

VLSM for minimum hosts

- Sometimes you have a problem which states that you need n hosts per network.
- Consider the example of 10.1.1.0/24, where you need to divide your network into as many subnets as possible, where each subnet can hold at least 10 hosts.
- Increase “10” by 2, then increase to the next power of 2 (i.e. 16).
- 16 needs 4 bits (0000-1111 is 16 combinations).
- Take 32-4, giving 28. Network is 10.1.1.0/28, or:
  - 10.1.1.0/28, 10.1.1.16/28, 10.1.1.32/28, etc.

Class Exercise

- You have 10.20.1.0/24. Split the network into subnets so that each net can support at least 31 hosts.

Broken VLSM

- Some legacy systems don't understand VLSM (e.g. RIP)
- Sometimes called the "subnet zero" problem
- This leads to 2 points of confusion, concerning the first and last network:
  - With 10.10.10.0/24 split into /27, networks 10.10.10.0/27 and 10.10.10.224/27 cause problems.
- For 10.10.10.0/27, 10.10.10.0 is the network number, and 10.10.10.255 is the broadcast address. But in VLSM, it's the network number for network 1, and the broadcast for network 8.
- Take care with legacy systems!!!

P2P netmask

- A point to point network is a little weird…
  - 10.0.0.14/32
  - Netmask 255.255.255.255
  - Broadcast 10.0.0.255
  - Gateway is likely to still be 10.0.0.254

- The gateway IP can be reused multiple times on each p2p link without difficulties.

- Really small netmasks > 1 IP …
- 10.0.0.5/30
  - 2 bits unset thus only 4 IPs in this net
  - IPs are 10.0.0.4,10.0.0.5,10.0.0.6,10.0.0.7
  - Broadcast will be highest ip, 10.0.0.7
  - The network has its own address (all bits zero) which reserves 10.0.0.4 for the network.
  - Max-1 is often the gateway, 10.0.0.6
  - Only 1 IP for host, 10.0.0.5

- Other than p2p, biggest netmask must be /30.

VLSM with mixed networks

- Consider the topology shown. You only have 10.1.1.0/24 to play with:

Net1 needs 50 hosts

Net2 needs 50 hosts

- 50 hosts suggests 6 bits, leaving 2 bits, /26
- This provides 4 networks.
- However there are 5 networks:
  - Net1
  - Net2
  - Router A-B
  - Router B-C
  - Router B-D

- Solution is to divide up one /26, and use that for router-router links.
- For Net1+2 50 hosts suggests 6 bits, leaving 2 bits, /26
- For routers, 2 hosts suggests 2 bits or /30.
- Use 10.1.1.0/26 for Net1, 10.1.1.64/26 for net2.
- Split 10.1.1.128/26 into multiple /30 links:
  - Net1 – 10.1.1.0/26
  - Net2 – 10.1.1.64/26
  - Router A-B – 10.1.1.128/30
  - Router B-C – 10.1.1.132/30
  - Router B-D – 10.1.1.136/30

Exercise

- Consider the topology shown. You only have 10.1.1.0/24 to play with:

Net1 needs 30 hosts

Net2 needs 30 hosts

Net3 needs 100 hosts

Solution:

- 10.1.1.0/24 gets split into:
  - Net3 – 10.1.1.0/25
  - Net1 – 10.1.1.128/27
  - Net2 – 10.1.1.160/27
  - Router A-B – 10.1.1.192/30
  - Router B-C – 10.1.1.196/30
  - Router B-D – 10.1.1.200/30

Discussion

- Here are some past exam questions you should now be able to answer:

Question 1

- Consider the following output from ip route show.

default via 192.168.255.1 dev eno3

10.0.1.24/29 dev eno2

192.168.255.0/24 dev eno3

192.168.0.0/16 dev eno4

What device will the following packets leave on:

a) destined for 10.0.1.1

b) destined for 10.0.1.25

c) destined for 192.168.255.4

d) destined for 192.168.14.254

CSN09703
Networked Services
Linux DNS configuration

Module Leader: Dr Gordon Russell

Lecturers: G. Russell, P.Leimich

Linux DNS

Resolver in Linux

- You now know how DNS forward and reverse works. But you do not have to do all the repeated queries yourself!
- The resolver looks after all the lookups.

- As an example, consider a command
  > ping www.linuxzoo.net
- The computer needs to find the IP number…

$ cat /etc/host.conf

order hosts,bind

- This is the schemes used to translate DNS.
  - hosts – use the /etc/hosts file
  - bind – use dns

/etc/hosts

- This file is the simplest lookup.
- It is called “host” resolution.
- The file has lines like

127.0.0.1 localhost.localdomain localhost

- It is IP, then hostname, then aliases for the host.
- Its fast and simple, but only lists machines you edit yourself into the file.
- It is good for “kickstarting” finding key machines.

> cat /etc/resolv.conf

```
search linuxzoo.net net
nameserver 10.200.0.1
```

- Not found in /etc/hosts? Check resolve.conf and use bind.
- nameserver – who to ask (our nameserver)
- search – add these to the host if not found.
  - Looking for www.linuxzoo? This search would try “www.linuxzoo”, “www.linuxzoo.linuxzoo.net”, “www.linuxzoo.net”. If still not found then fail.
  - This is a convenience for users.

> dig www.linuxzoo.net @10.200.0.1

- Ask your nameserver.

1. If nameserver knows the answer, return it.
2. If unknown, nameserver recurses (plus store the answer in a cache).
3. If no one knows after recursion, return a failure.

- If resolv.conf has multiple nameserver entries, each one is tried until all are tried or the answer is returned from one of them.

Your own nameserver

- You might want to run your own nameserver if:
  - You perform lookups frequently and want to cache the queries locally
  - You want to query the root servers directly without having to talk via a local nameserver.
  - You want to add your own entries to DNS
    - You run your own domain and want to control your DNS entries directly.
    - You have local IPs and want to name them on your own network.

Nameserver daemons

- The most popular is “named”.
- It is installed on the UML simulations.
- It is part of the bind9 distribution (www.isc.org).
- It is popular, but other services are available.
  - Common systems are often targetting for hacking attacks.
  - NAMED is often cited as a security problem.
  - It needs to be patched often.
  - It should be secured using addition security technology.
    - It can be run in a chroot.
    - It can use SELinux if available.
    - It can use both! In our labs we will use SELinux security.

What is a chroot

- Many Linux services can be run “chroot”.
- This gives a new “/” directory just for that service.
- It contains only the minimum files and directories.
- The contents are (when possible) owned by a different user than the one who owns the service.
- The service should not be executed as root.
- Service runs as a user who does nothing else but run that service.
- Hack the service and you are stuck in a directory with little contents, you can change very little, with a user which can do nothing… If possible, always run services in a chroot!

SELinux protection

- In our experiments we will not be using a chroot for additional security.
- Instead we will use SELinux.
  - This gives the kernel a set of rules which named must obey, including what files can be opened and what sort of network connections can be made.
  - Fortunately this is all pre-configured in Fedora and thus completely invisible to us as administrators.

RNDC

- RNDC allows you to administer NAMED remotely.
- Obviously this has to happen with some security.
- RNCD uses a signed key to validate its security credentials.
- In a non-chroot solution this needs only to be stored in /etc/rndc.key
- If you are (for some reason) using a chroot it must also be copied to /var/named/chroot/etc/rndc.key

Generate the key

- In a normal installation, you should generate your own key.
- In linuxzoo, you get the key generated automatically. This is a good thing, as generating the key in linuxzoo turns out to be problematic. DON'T.
- However, if you want to generate your own key (ignoring my advice) then:

$ rndc-confgen –a –b 128 –r keyboard

  - -b 128 – Sets the bits to 128 (fast but weak)
  - -r keyboard – Random keys require "entropy". Normally done with /dev/random, but in UML this does not work well. This option asks you to type randomly for a while, and use your keyboard rhythm to generate the random number!
  - The only time you want to do this in linuxzoo is if you deleted the key file!

/etc/named.conf

- Really two parts – options and zone
  - Options allow run time configurations and global defaults to be set.
  - Zone entries allow us to set up a forward or reverse entry for a domain.

Master and slave

- There are two distinct types of zone:
  - Master- they have the zone definitions and you can edit that information if you wish
  - Slave- they copy the zone definition automatically from the master. Their copy is read only, so you cannot edit the records on a slave.
- Slave nameservers are needed to give DNS higher reliability and redundancy. You edit on a master and the change is copied to all your slaves. Slave configuration is not considered further here.
- The master is often called a PRIMARY, and any slaves called SECONDARIES. But these names are badly abused, so stick with master/slave.

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

This tells the daemon to use the root servers listed in named.ca to resolve things not solved by other entries. This can be considered the “default”.

```
options {
        directory "/var/named";
        forward only;
};
```

- Nothing exciting in this part.
- Note that in linuxzoo, DNS requests to the roots (or anywhere else) are intercepted my the linuxzoo filewall and redirected to 10.200.0.1.
- This keeps the load on the root servers down, and makes it harder for people to use linuxzoo to hack the planet…
- Also allows my name service to falsify records – needed to make things work right in the UMLs.

```
zone "localhost" IN {
        type master;
        file "localhost.zone";
        allow-update { none; };
};
```

- The file localhost.zone gives forward resolving for the domain “localhost”.

```
zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "named.local";
        allow-update { none; };
};
```

- The named.local file give reverse lookups for the 127.0.0.0/24 IP range.

localhost.zone

```
$TTL    86400
$ORIGIN localhost.
@                    1D IN SOA       @ root (
                              42              ; serial (d. adams)
                              3H              ; refresh
                              15M             ; retry
                              1W              ; expiry
                              1D )            ; minimum

                     1D IN NS        @
                     1D IN A         127.0.0.1
```

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.
```

Example : grussell.org, in IP 50.1.1.0/24

- /etc/named.conf zone for this:

```
zone "grussell.org" IN {
        type master;
        file "grussell.zone";
        allow-update { none; };
};
zone “1.1.50.in-addr.arpa" IN {
        type master;
        file “grussell.rev";
        allow-update { none; };
};
```

> cat /var/named/grussell.zone

```
$TTL    86400
$ORIGIN grussell.org.
@                       1D IN SOA       ns1 admin.grussell.org. (
                                2004101701              ; serial
                                3H                      ; refresh
                                15M                     ; retry
                                1W                      ; expiry
                                1D )                    ; minimum

                        1D IN NS        ns1
                        1D IN A         50.1.1.1
www                             CNAME           grussell.org.
ns1                             1D IN A         50.1.1.10
```

> cat /var/named/grussell.rev

```
$TTL    86400
@       IN      SOA     ns1.grussell.org. admin.grussell.org.  (
                                1997022700 ; Serial
                                28800      ; Refresh
                                14400      ; Retry
                                3600000    ; Expire
                                86400 )    ; Minimum
     IN      NS       ns1.grussell.org.

1   IN      PTR     grussell.org.
10  IN      PTR     ns1.grussell.org.
```

MX (Mail Exchange) records

host1	IN	MX	10	host1
	IN	MX	20	backuphost
	IN	MX	30	mx.easydns.com.

- Priority goes to lowest number.
- No dot and end – add the origin.

Load Balancing

- With 1 server providing a service, only that server can handle requests.
- Multiple servers can handle requests in parallel.
- But how can a single server name be made automatically share out requests to multiple servers?
- Load balancing does this…

Email server balancing

host1	IN	MX	10	smtp1
	IN	MX	10	smtp2
	IN	MX	10	smtp3
smtp1	IN	A		10.0.0.1
smtp2	IN	A		10.0.0.2
smtp3	IN	A		10.0.0.3

- Equal priority MX records are usually randomly utilised
- Selection mechanism is mail application dependent

Email server balancing with A

host1	IN	MX	10	smtp
smtp	IN	A		10.0.0.1
	IN	A		10.0.0.2
	IN	A		10.0.0.3

- Chosen using rrset-order (default is random)
- Make sure reverse of .1,.2,.3 -> smtp.domain.com

Server balancing with A

www	IN	A	10.0.0.1
	IN	A	10.0.0.2
	IN	A	10.0.0.3
ftp	IN	A	10.0.0.10
ftp	IN	A	10.0.0.11

- Chosen using rrset-order (default is random)
- The effect of caching needs to be considered
- The distribution of load needs to be considered

DNS record types

- SOA – Start of authority, gives params for zone
- A6 – handle ipv6 references
- NS – useful in delegation but basically ignored.
- CNAME – an alias
- HINFO – Hardware and OS being run.
- RP – who to send emails to.
- TXT – useful text – for instance can be used to certify email server IP number for some spam detection software.

Discussion

Discussion

- Spot the error(s)

```
$TTL    86400
$ORIGIN broken.net.
@                       1D IN SOA       ns1 admin@grussell.org. (
                                2004101701                      ; serial
                                3H                      ; refresh
                                15M                     ; retry
                                1W                      ; expiry
                                1D )                    ; minimum

                        1D IN NS        ns1
                        1D IN A         10.0.0.1
www                     CNAME           broken.net.
ns1                     1D IN A         10.0.0.10.
ns2.                    1D IN A         10.0.0.11.
```

Question 1

Provide a forward DNS file for the domain test.com. The parameters of the SOA are unimportant. Make sure of the following:

test.com maps to 1.0.0.1

www.test.com is an alias to test.com

email.test.com is 1.0.0.2

A nameserver exists at 1.0.0.10

Email to test.com goes to the email host.

Answer: test.zone

```
$TTL    86400
$ORIGIN test.com.
@                   1D IN SOA       ns1 root (
                                ...)        ; minimum

                    1D IN NS        ns1
                    1D IN A         1.0.0.1
                  MX 10           email
www               CNAME           test.com.
email             A               1.0.0.2
ns1               A               1.0.0.10
```

Question 2

In a server using DNS round robin load balancing across three different A records, discuss what would happen if one of the machines associated with one of the A records failed. How could such a problem be managed?

CSN09703
Networked Services
Apache mod_rewrite

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

mod_rewrite

URL Rewriting

- A useful module in apache is mod_rewrite.
- This allows us to change URLs dynamically.
- This can be useful to, for example,
  - Change the URL of aliases in a domain so that they always give the name you want.
  - Support directories and files being moved without breaking bookmarked URLs.
  - Provide a variety of proxying methods.

URL Parts

- URL:
  - http://linuxzoo.net/mystuff/file.html

- This is made up of
  - Protocol: “http”
  - Hostname: “linuxzoo.net”
  - Path: “/mystuff/file.html”

- In URL rewriting, you often only get to operate on part of a URL in any one instruction, so take care you know which bit you want to use!

Methods

- mod_rewrite has many functions…
- The key functions are:
  - RewriteCondition – an IF statement
  - RewriteRule – an action (doit) statement.
- These can be placed almost anywhere in the apache configuration files.
- We will concentrate on their use in VirtualHost areas of httpd.conf.
- To work, the area must also have:

```
RewriteEngine on
```

rewriteRule

- Basic for of this rule is:

RewriteRule URL-PATH New-PATH

- For instance, you have moved /dvd1.iso into a subdirectory /dvd…

RewriteRule /dvd1.iso /dvd/dvd1.iso

- The first parameter is a Regular Expression.
- This is an “internal” rewrite. In this case only Apache knows the changed location, and from the browser’s point of view the file appears to still be /dvd1.iso.

Regular Expressions

- The match comparison is a regular expression.
- Useful aspects of regular expressions include:
- Text matching:

.	Any single Character
[chars]	One of the characters in chars
[^chars]	None of the characters in chars

Quantifiers and Grouping

- Quantifiers:

?	0 or 1 of the preceding text
*	0 or N of the preceding text
+	1 or N of the preceding text

- Grouping

(text)	A text group – Can mark the border of an alternative or for RHS reference as $N

Anchors and Escaping

- Anchors:

  ^ Start of the URL

  $ End of the URL

- Escaping

  \char Allows you to use a character as the “char”. For instance, \^ is the ^ character and not the start of the URL.

Back References

- $N corresponds to a group from the URL match.
- For example, rewrite any URL ending in .txt to .html one could write:

```
RewriteRule        (.*)\.txt        $1.html
```

More complex example.

- You have directories “/android/1.txt”, “/android/2.txt”, and “/gordon/android/here/3.txt”.
- Your manager has told you for marketing reasons you need to start using “droid” rather than “android”, so you rename the 2 directories.
  - But you want the old URLs to still work without having to keep copies of the old directories.

```
RewriteRule ^(.*)/android/(.*)$    $1/droid/$2
```

Additional Flags

- At the end of the RewriteRule can be a number of flags.
- The Flags are listed in [brackets], eg [F,G] for flags F and G.
- These change or enhance the behaviour of the match.

Options:

- R or R=code - This sends the browser the new URL as an external REDIRECTION. The code can be the type or redirection, such as 302 for MOVED TEMPORARILY (the default).
- F - Send back FORBIDDEN.
- G - Send back GONE
- C - Chained… if the test is false skip the next rule too.
- L - Last – do not look at any more rules.
- NC - case insensitive.

- There are many more options, but these are some of the important ones.

Complex example

- If the URL starts /drgordon/, rewrite /drgordon/ to /lordgordon/.
- In addition, if the URL did have /drgordon/ in it, replace any use of “hi.txt” with “greetings.txt”.

```
RewriteRule ^/drgordon(.*)        /lordgordon$1    [C]
RewriteRule ^(.*)/hi\.txt$        $1/greetings.txt [L]
```

- If rule 1 matches then rule 2 is checked
- If rule 2 also matches do not do any other rules (e.g. rule 3 onwards).
- If rule 1 does not match then go onto rule 3+.

RewriteCond

- This command performs tests or RULES.
- If the test matches, then the next test is checked.
- If all tests match, then the first RewriteRule which follows the tests is performed (or more than 1 if they are Chained together).
- If any Cond does not match, processing skips on till after the Rule(s) in this block.

- Basic Form of RewriteCond

RewriteCond TestString ConditionString

- The value of the TestString is compared to the conditionstring.
- Condition String can be any type of regular expression.
- TestString can be one of a huge variety of things, including variables and file tests.

Variables:

- Here are some of the important variables:
  - REMOTE_ADDR
  - REQUEST_METHOD
  - HTTP_HOST
  - REQUEST_URI (e.g. a path such as /index.html) (Its URI not URL)
  - REQUEST_FILENAME (e.g. /home/gordon/…)

- You use these by using a percent sign, then enclosing the variable name with curly brackets, such as %{REMOTE_ADDR}.
- There are many variables available for a variety of purposes.

Flags

- RewriteCond can also take flags in the same way as RewriteRule.
- There are only 2 flags:
  - NC – case insensitive
  - OR – or the Conds together.
- Normally all conditions have to be true before the RewriteRule is done, with OR the RewriteRule is done if ANY condition is true.

Example 1:

- If 10.20.0.5 tries to view any file, redirect the page reference to /gordon/bye.html.

```
RewriteCond %{REMOTE_ADDR} ^10\.20\.0\.5$
RewriteRule .*   /gordon/bye.html [L]
```

Example 2:

- If 10.20.0.5 tries to view /gordon/index.html, redirect the page reference to /gordon/bye.html.

```
RewriteCond %{REMOTE_ADDR} ^10\.20\.0\.5$
RewriteRule ^/gordon/index\.html$ /gordon/bye.html [L]
```

Example 3:

- My VirtualHost has grussell.org, www.grussell.org, and www.grussell.org.uk.
- Rewrite all requests to grussell.org.

```
RewriteEngine on
RewriteCond %{HTTP_HOST}   !^grussell\.org$
RewriteRule ^(.*)$ http://grussell.org$1 [L,R]
```

Example 4:

- Rewrite *.grussell.org to grussell.org, and *.grussell.org.uk to grussell.org.uk.

```
RewriteEngine on
RewriteCond %{HTTP_HOST} ^.+grussell\.org$
RewriteRule ^(.*)$ http://grussell.org$1 [L,R]

RewriteCond %{HTTP_HOST} ^.+grussell\.org\.uk$
RewriteRule ^(.*)$ http://grussell.org.uk$1 [L,R]
```

Example 5:

- If the request is for host “try.com”, and the path is “/here.dat”, then redirect externally to http://there.org.

```
RewriteEngine on
RewriteCond %{HTTP_HOST} ^try\.com$
RewriteCond %{REQUEST_URI} ^/here\.dat$
RewriteRule ^(.*)$ http://there.com [L,R]
```

Discussions

Question 1

Supply mod_rewrite instructions such that a request for http://grussell.org/~uta gets redirected externally and permanently to http://upriss.org.uk.

CSN09703
Networked Services
SELinux Usage

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

SELinux Usage

- The theory of SELinux is tricky…
- However, as a system administrator, day to day SELinux is generally much more high level and mechanical.
- To help, essential SELinux offers
  - An automatic labelling service
  - Commands to perform manual type changes
  - Control over Boolean activation of additional rules
  - A straight forward diagnostic tool with suggested changes
  - A simple tool to extend SELinux to new problems

Automatic labelling

- When a file is created it needs a type immediately.
  - The average user needs SELinux support without needing any knowledge…
  - By default its type is inherited from the parent.

- It is possible to create a type transition based on a filename…

```
$ filetrans_pattern(staff_t, user_home_dir_t, httpd_user_content_t,
dir, "public_html")
```

- This is best left to the policy writers rather than for everyday use.
- However it is useful…

Warning

- Inheriting the parent type only happens when a file is created.
- Similarly, file transition patterns also only happen at creation time.
- If a file is renamed (e.g. using mv) the current type stays as it is…
  - This can lead to confusion if you accidentally created a file, directory, or whole directory tree initially in the wrong place.

```
$ mkdir public_html
# ls -Z
… unconfined_u:object_r:httpd_user_content_t:s0     public_html
# rmdir public_html
# mkdir public_error
# ls -Z
… unconfined_u:object_r:user_home_t:s0              public_error
# mv public_error public_html
# ls -Z
… unconfined_u:object_r:user_home_t:s0              public_html
```

Automatic-ish labelling

- One cannot expect the average user to write their own transition policy…
  - Instead, users have to accept the automatic type but can then relabel easily.
  - To help, a database of “default” types is maintained.
    - These are not automatically assigned at creation time though.
  - Each user needs to run a command in order to get the default type set…
    - Otherwise the get the inherited type or the name transition type.

Automatic-ish labelling

- A database of default types is held.
  - At relabelling time the full pathname is used to look up the default type.
  - Database holds rules as regular expressions

- You can simply list all the rules:

```
$ semanage fcontext -l | grep passwd
/etc/passwd[-\+]?  regular file  system_u:object_r:passwd_file_t:s0
```

- You can get the default relabelling for a specific file

```
$ matchpathcon /etc/passwd
/etc/passwd     system_u:object_r:passwd_file_t:s0
```

Auto RE-typing

- So if you move the file and know you got the wrong type, what to do?
- For basic things, you can always reset a file to its default type with *restorecon*.

```
$ matchpathcon /root/test
/root/test      system_u:object_r:admin_home_t:s0

$ ls -Z /root/test
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 /root/test

$ restorecon -v /root/test
restorecon reset /root/test context unconfined_u:object_r:user_tmp_t:s0->unconfined_u:object_r:admin_home_t:s0

$ ls -Z /root/test
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 /root/test
```

Auto RE-Typing a whole directory tree

- Sometimes you might create directories and files, then realise it has to be moved…
  - Simply mv them as normal to the new location
  - Then restorecon with -r (for recursive), and give the pathname to the top directory.
  - Use care… remember you get the defaults
  - If a user deliberately changed a type and you relabel the whole of /home, you will not be popular.
  - You can use –n if you are concerned, which just tells you what would change without actually changing anything.

```
$ restorecon -rnv /home
```

Manual Changes

- Users who know what they are doing may with to change labels manually. This is done using *chcon*.
  - Note this changes only the specific file type, not the default type
  - If after manually changing a file you do restorecon, the file changes back to the default…

```
$ ls -Z test
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 test
$ chcon -t bin_t test
$ ls -Z test
-rw-r--r--. root root unconfined_u:object_r:bin_t:s0   test
```

Permanent manual changes

- You can reprogram the restorecon type for things if you wish.
  - That way restorecon will set the type to what you really want.

- So if you always want /root/test to be bin_t, you can do:

```
$ semanage fcontext -a -t bin_t "/root/test"
```

- Do remember that the database is a list of regular expressions, so a default for a file “test.dat” would need to be “test\.dat”.

- You can also use regular expressions to do complex matching

```
$ semanage fcontext -a -t bin_t "/root/(.*/)?test\.dat"
```

Using Booleans

- As you remember, Booleans allow rules to be switched on and off in groups.
  - Each group of rules linked to a Boolean effectively allow or block a feature you may be interested in.
  - If you can find a Boolean to allow what you want to do, this is much easier than writing your own SELinux rules.
  - getsebool allows you to read current Boolean states.
  - setsebool allows you to set current states.

- List all Booleans and their states

```
$ getsebool -a
…
httpd_enable_homedirs --> on
…
```

Setting booleans

- Get the value of a named Boolean

```
getsebool httpd_enable_homedirs
httpd_enable_homedirs --> off
```

- Set the value of a named Boolean (remembered till next reboot)

```
setsebool httpd_enable_homedirs 1
getsebool httpd_enable_homedirs
httpd_enable_homedirs --> on
```

- Set the value of a named Boolean permanently

```
setsebool -P httpd_enable_homedirs 1
```

But which Boolean do you need???

- From the theory you could work through the rules…
- However, often you can get this sort of information from a manual.
- You can get a little commenting text using the following:

```
$ semanage boolean -l | grep httpd_enable_homedir
httpd_enable_homedirs   (on,on)  Allow httpd to enable homedirs
```

- There are also commands which analyse errors in the selinux log and propose solutions, and these may be Booleans.
  - In the remaining slides diagnostic approaches are discussed, and you will see that Boolean changes are sometimes proposed automatically.

Diagnostics

- If something means an action is blocked by SELinux then it is logged.
- The log location is /var/log/audit/audit.log

- The file can contain many issues, some of which are not interesting!
- Lines which start

```
type=AVC
```

are the ones which are significant.

- A useful command is ausearch, which is a common tool to analyse log files.

ausearch -m avc -ts recent

- This command files AVC errors in the audit.log file which have occurred within the last 10 minutes.

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file

$ date -d @1490555793.938
Sun 26 Mar 20:16:33 BST 2017
```

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- Denied { read } – Action which was blocked
- Pid – the PID of the process affected.
- Comm – the short name of the process running
- Name – The name of the target which was being accessed
- Dev - The device of the object

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- Ino - The inode number of the object.
- The device dm-0 indicates lvm filesystem, and you can find out more by doing:

```
$ dmsetup info /dev/dm-0
```

- In this case, dm-0 is the root filesystem /, so to find the file do:

```
$ find / -xdev -inum 26080829
/etc/httpd/conf/httpd.conf
```

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- Scontext – the label of the process making the request.
- Tcontext – the label of the target
- Tclass – what was the target (in the case it is just a file).

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- It could be an error, and you can find out:

```
# ausearch -m avc -ts recent > file
[root@host-5-65 ~]# cat file | audit2why
 Was caused by:
        Missing type enforcement (TE) allow rule.
```

- Often the error message is suggesting SELinux Booleans which if enabled would allow this action, and this generally can be very useful.

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- If it is not an error, you can actually add more rules to the database and permit this action:

```
# ausearch -m avc -ts recent > file
[root@host-5-65 ~]# cat file | audit2allow
```

```
#============== httpd_t ===============
allow httpd_t named_conf_t:file read;
```

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="httpd.conf"
dev="dm-0" ino=26080829
scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- However, in this case it is just a mislabelled file:

```
# restorecon -v /etc/httpd/conf/httpd.conf
restorecon reset /etc/httpd/conf/httpd.conf context
system_u:object_r:named_conf_t:s0-
>system_u:object_r:httpd_config_t:s0
```

SELinux for Systems

- People are scared of SELinux..
  - It is complex sounding!
- Good error messages, but can be cryptic
- Good range of diagnostic tools, but knowledge of them is poor.
- Excellent layer of security, but many still like running in permissive mode…
- Try to work with it switched on, and instead use unconstrained for processes which cause problems in SELinux:
  - This at least secures some aspects of Linux
  - You can also extend SELinux to support new custom processes, but that is another lecture…

Discussion

- An SELinux alert is stopping a process from operating. Analysing the audit.log file shows the following information:

```
type=AVC msg=audit(1490555793.938:152): avc:  denied
{ read } for  pid=1800 comm="httpd" name="ftpd.conf"
dev="dm-0" ino=26 scontext=system_u:system_r:ftpd_t:s0
tcontext=system_u:object_r:named_conf_t:s0 tclass=file
```

- How would you discover which file has the named_conf_t label discussed in this alert?

CSN09703
Networked Services
Stateful Firewalls

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

SSH server example

- Allow someone to SSH and TELNET into your machine

Standard stateless INGRESS FILTER RULES, then

```
iptables -t FILTER  -A INPUT -p tcp --dport 22 -j ACCEPT
iptables            -A INPUT -p tcp --dport 23 -j ACCEPT
```

SSH Client Example

- Allow local machine to ssh and telnet to other machines:

Standard stateless INGRESS FILTER RULES, then

```
iptables -t FILTER  -A INPUT -p tcp --sport 22 -j ACCEPT
iptables            -A INPUT -p tcp --sport 23 -j ACCEPT
```

- When you SSH to elsewhere, your first packet goes out on the OUTPUT chain to dport 22 on that remote machine. However the reply comes FROM dport 22 on that remote machine to your computer via the INPUT chain.

sport or dport

Server End

$ iptables –A INPUT –p tcp --dport 22 –j ACCEPT

Your End

$ iptables –A INPUT –p tcp --sport 22 –j ACCEPT

SSH Client Example

```
iptables -t FILTER  -A INPUT -p tcp --sport 22 -j ACCEPT
```

- So this rule intends to allow the reply (and all subsequent packets) to be ACCEPTed where it relates to someone on the local machine using SSH to connect to a remote machine.
- But a hacker from any machine could just send packets with an sport 22, and these would be accepted by the firewall even though they are unwanted.
  - This could allow an exploit to be used against our computer…
- We really need a way of knowing a packet is part of a TCP stream we actually want…

Stateful Rules

- If you could find out what packets are the start of a stream and which packets are not, then you could have rules to control the first packet.
  - If packet 1 then check to see if the rules say it is ok
  - If packet 2 then it must have been ok, otherwise packet 1 would have been blocked…
- Linux holds a state table of packets to allow such things to be monitored…

Example: What is TCP state?

- In TCP/IP, TCP goes through a number of states:

Stateful Rules

- You can add iptables rules to detect what state a packet is in.
- This feature used to be called “state”, but now is known as “conntrack” or Connection Tracking.
- Tracking tests available include:
  - NEW – never seen the stream before
  - ESTABLISHED – traffic in both directions in the stream.
  - RELATED – associated with an established stream but is actually a different stream. For instance an established FTP connection may transfer each file using a new stream (active connections). Each new connection is RELATED.
  - INVALID – worrying and should probably be dropped!

/proc/net/nf_conntrack

```
ipv4     2 tcp      6 431978 ESTABLISHED src=146.176.164.219
dst=146.176.166.41 sport=56749 dport=22 ...
```

- Here for instance a TCP stream has exchanged SYN, so is ESTABLISHED..

```
ipv4     2 tcp      6 81 TIME_WAIT src=10.200.0.1
dst=10.200.0.19 sport=63040 dport=80 ...
```

- Streams can end up in many states. TIME_WAIT means the stream is closed, but the table remembers closed connections for a time so that left-over retransmissions can be handled properly.

Iptable Rules based on network state

- If a connection has been going a while…

```
$ iptables -A INPUT -m conntrack --ctstate
  RELATED,ESTABLISHED -j ACCEPT
```

- But a new incoming connection to the local ssh server needs to be allowed?

```
$ iptables -A INPUT -m conntrack --ctstate
  NEW -p tcp --dport 22 -j ACCEPT
```

DROP or REJECT

- DROP just blocks the packet.
- REJECT blocks and also sends an ICMP message to the sender saying it is blocked.
- DROP gives nothing away, and a hacker finds it difficult to get useful information about your network if packets are dropped.
- REJECT gives away information about your firewall ruleset.
- With DROP and IP, where packets are lossy, some protocols may retransmit a packet which you have dropped thinking it has been lost. This means dealing with the same packet again and again.
- A good rule: intranet traffic should be REJECT, internet traffic should be DROP.
- Note you can only ACCEPT or DROP in a policy, but you can REJECT in a rule.

Example: tight SSH server example

- Allow someone to SSH into your machine, provided their IP is 10.0.0.1.

Standard stateful INGRESS FILTER RULES, then

```
iptables -A INPUT -p tcp --dport 22 -s 10.0.0.1 -j ACCEPT
```

- Allow someone to SSH into your machine, provided their IP is between 10.0.0.1 and 10.0.0.255.

```
iptables -A INPUT -p tcp --dport 22 -s 10.0.0.1/24 -j ACCEPT
```

Basic client machine

- Allow local machine to ssh only to 10.0.0.1
- You have been asked to use only the INPUT chain and leave OUTPUT as ACCEPT

Standard stateful INGRESS FILTER RULES, then BEFORE RELATED/ESTABLISHED

```
iptables -A INPUT -p tcp --sport ssh ! -s 10.0.0.1
                -j DROP
```

Add a rule to permit ping

- Anyone can ping this machine:

Add to the end of the file

```
iptables -A INPUT -p icmp
          --icmp-type echo-request -j ACCEPT
```

Rate limiting

- You can do statistics on connections and use that in the rules.
- The “limit” module allows you to make checks to see how times per second a particular rule is run.
- For instance, it is nice to handle PINGs, but if you were asked to handle too many they perhaps it is ok to ignore some of those?
- It uses the “limit” module, and the test is “--limit” and a rate, such as 3/second, 5/minute, etc.
- Exceeding the rate is FALSE, and within the limit is TRUE.

Add a rule to permit safe ping

- Anyone can ping this machine, but I will only respond if the ping requests are slower than 2 per second:

Add to the end of the file, but before RELATED/ESTABLISHED

```
iptables –A INPUT –p icmp
          --icmp-type echo-request
          -m limit --limit 2/second –j ACCEPT
```

- Might help protect a system from a “ping of death”.

Monitor safe ping

- Anyone can ping this machine, but I will only respond if the ping requests are slower than 2 per second. Faster than that gets a logged message

Add to the end of the file, but before RELATED/ESTABLISHED.

```
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 2/second -j ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request -j LOG
```

Monitor Monitor safe ping

- Anyone can ping this machine, but I will only respond if the ping requests are slower than 2 per second. Faster than that gets a logged message, but don't log more than 10 per minute…

Add to the end of the file, before RELATED/ESTABLISHED

```
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 2/second -j ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 10/minute -j LOG
```

Impact of Policy DROP and ACCEPT

```
iptables -P INPUT ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 2/second -j ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 10/minute -j LOG
```

What happens to ICMP packets faster than 2/second?

Impact of Policy DROP and ACCEPT

```
iptables -P INPUT ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 2/second -j ACCEPT
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -m limit --limit 10/minute -j LOG
```

```
iptables -A INPUT -p icmp
          --icmp-type echo-request
          -j DROP
```

Basic client machine

- Allow local machine to ssh and telnet out
- …using INPUT policy of ACCEPT, leaving all other chains empty and ACCEPT.

```
/sbin/iptables -P INPUT ACCEPT
#
iptables –A INPUT –m conntrack --ctstate NEW –j DROP
iptables –A INPUT –p tcp --sport 22 –j ACCEPT
iptables –A INPUT –p tcp --sport 23 –j ACCEPT
iptables –A INPUT –p tcp –j DROP
```

- …using INPUT policy of DROP:

```
/sbin/iptables -P INPUT DROP
#
iptables –A INPUT –m conntrack --ctstate NEW –j DROP
iptables –A INPUT –p tcp --sport 22 –j ACCEPT
iptables –A INPUT –p tcp --sport 23 –j ACCEPT
```

Basic client machine

- Allow local machine to ssh only to 10.0.0.1,
- …but using INPUT policy of ACCEPT:

```
iptables -P INPUT ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW -j DROP
iptables -A INPUT -p tcp --sport ssh ! -s 10.0.0.1
            -j DROP
iptables -A INPUT -p tcp ! --sport ssh -j DROP
```

- …but using INPUT policy of DROP:

```
iptables -P INPUT DROP
iptables -A INPUT -m conntrack --ctstate NEW -j DROP
iptables -A INPUT -p tcp --sport ssh -s 10.0.0.1
            -j ACCEPT
```

INPUT ACCEPT

- To be clear, a policy of ACCEPT is seriously silly…
- Good to learn on but wide open to attack.
- ACCEPT policy means you need rules to block. Anything you didn’t think about is accepted.
- DROP policy means you need rules to allow. Anything you didn’t think about is dropped.
- Security relies on a good policy, and that must be DROP.

subroutines

- Sometimes, complex rules in a single chain are difficult to manage.
- My server firewall has 200 rules…
- In the same way as programs benefit from subroutines, so do firewall rules.
- Subroutines are created with –N, deleted with –X, and can be reached with a target of “-j subroutinename” and returned with a target of “-j RETURN”.

Monitor Monitor safe ping subroutine

- Anyone can ping this machine, but I will only respond if the ping requests are slower than 2 per second. Faster than that gets a logged message, but don't log more than 10 per minute… Use a SUBROUTINE

```
iptables –X ECHO
iptables –N ECHO
Iptables –A ECHO -m limit --limit 2/second –j ACCEPT
iptables –A ECHO -m limit --limit 10/minute –j LOG
iptables –A ECHO –j DROP

iptables –A INPUT –p icmp --icmp-type echo-request –j ECHO
```

Extra tests

- There are many other simple tests.
- Use man iptables to find more
- For example:
  - Stop apache surfing the web:

    iptables –A OUTPUT state --state NEW -p tcp
    --sport 80 -m owner --uid-owner=apache -j DROP
  - Specify port ranges like:

    --sport 137:139
  - Specify ip masks like:

    -d 10.0.0.1/24

Basic Stateful FORWARDING

- You are running your firewall machine with 2 network cards, eth0 and eth1.
- Eth0 connects to the internet, Eth1 to the intranet.
- In this regard Eth1 is a gateway for your local network.

- Eth0 is 10.0.1.1/24, Eth1 is 10.0.2.254/24
- You have two servers in your intranet.
  - M1 is 10.0.2.1/24, running an ssh server
  - M2 is 10.0.2.2/24, running an http server
- GW Firewall FORWARD would be:

```
iptables -F FORWARD
iptables -P FORWARD DROP
iptables –A FORWARD –m conntrack--ctstate RELATED,ESTABLISHED
    –j ACCEPT
iptables –A FORWARD –m conntrack--ctstate NEW –p tcp –i eth0 --dport
    ssh –d 10.0.2.1 –j ACCEPT
iptables –A FORWARD –m conntrack--ctstate NEW –p tcp –i eth0 --dport
    http –d 10.0.2.2 –j ACCEPT
```

Egress filtering

- Up till now you have been using ACCEPT as the default policy for OUTPUT.
- However, this is not as secure as having a DROP policy.
- DROP as the policy in OUTPUT is called *egress filtering*.
- Although easy to completely mess up it is no harder than INPUT DROP policy.
- It limits OUTPUT packets to only those which you explicitly define.
- It could help reduce hacking attempts, and the spread of viruses.

Complete EGRESS Example

- Configure a non-routing server firewall which runs telnet and http servers. Users on the server can ssh out. Use EGRESS filtering.

```
/sbin/iptables -F INPUT
/sbin/iptables -F OUTPUT
/sbin/iptables -F FORWARD
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
```

```
iptables -A INPUT -m conntrack --ctstate \
            RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW \
            -p tcp --dport telnet -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW \
            -p tcp --dport http -j ACCEPT

iptables -A OUTPUT -m conntrack --ctstate \
            RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m conntrack --ctstate NEW\
            -p tcp --dport ssh -j ACCEPT
```

Other Firewall Ideas

- Block private IPs leaving the gateway for the internet:

iptables -A FORWARD -o eth0 -s 192.168.0.0/16 -j DROP

iptables -A FORWARD -o eth0 -s 172.16.0.0/12 -j DROP

iptables -A FORWARD -o eth0 -s 10.0.0.0/8 -j DROP

- Only the loopback device can have IPs 127.0.0.1/8
- Blocking packets with a source port of smtp cuts down on viruses if the machine has no email server.
- Block netbios (ports 136/137) leaving for the internet.

Discussion

- Here are some past exam questions you should now be able to answer:

Question 1

Show the iptables commands relevant in defining an egress filter allowing only related or established connections, as well as outgoing http, to be accepted and all other egress traffic to be rejected. You can assume egress only involves eth0.

Question 2

Consider the following iptables configuration:

```
iptables –P INPUT drop
iptables –A INPUT –m conntrack --ctstate
    RELATED,ESTABLISHED –j ACCEPT
iptables –A INPUT –p tcp --sport ssh –j
    ACCEPT
```

Would incoming ssh connection requests be blocked? Give an explanation for your answer.

Question 3

- Consider the output below:

```
$ cat /proc/net/nf_conntrack | grep "port=22"
ipv4     2 tcp      6 431102 ESTABLISHED src=10.200.0.1
dst=10.0.5.33 sport=13073 dport=22 src=10.0.5.33 dst=10.200.0.1
sport=22 dport=13073 [ASSURED] mark=0
secctx=system_u:object_r:unlabeled_t:s0 zone=0 use=2
ipv4     2 tcp      6 10 TIME_WAIT src=192.168.1.68
dst=146.176.166.6 sport=39934 dport=22 src=146.176.166.6
dst=192.168.1.68 sport=22 dport=39934 [ASSURED] mark=0
secctx=system_u:object_r:unlabeled_t:s0 zone=0 use=2
```

What does this information tell you, given the current machine IPs are 192.168.1.68 and 10.200.0.5?

CSN09703
Networked Services
Day 2b: Basic Administration Concepts Part 2

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

This lecture

- User Management
- Searching (revision)
- Discussions

User Management

User Management

- A wide topic…
  - Adding/Removing/Changing current users
  - Default Scripts
  - Global Scripts

Manual Creation

- User entries in passwd,shadow, group,gshadow.
- Home directory in /home.
- Copy basic .files into their home directory.
- Make new user own their own directory and files.

> adduser gordon

- This does all the magic for you.
- It copies the default .files from /etc/skel/

> ls –a /etc/skel/

.bash_logout .bash_profile .bashrc .gtkrc .kde

- In bash, .bashrc is executed in non-login shell, and .bash_profile in a login shell.

Skel files

- These files are the basic .files created for a new user.
- Users are free to edit these when they log in.
- This allows them to control their own path, env, and other settings (such as aliases).
- However, if you install a new package which needs something set for each user at login, editing all these copies by hand would be tiresome.

> ls /etc/profile.d

colorls.csh	gnome-ssh-askpass.csh	krb5.csh	less.csh	vim.csh
colorls.sh	gnome-ssh-askpass.sh	krb5.sh	less.sh	vim.sh
glib2.csh	kde.csh	lang.csh	qt.csh	which-2.sh
glib2.sh	kde.sh	lang.sh	qt.sh	

- If you log in with bash, all the .sh files are executed before your .files
- If you log in with csh, all the .csh files are executed before your .files.

> cat /etc/profile.d/vim.sh

```
if [ -n "$BASH_VERSION" -o -n "$KSH_VERSION" -o -n
   "$ZSH_VERSION" ]; then
  # for bash, pdksh and zsh, only if no alias is already set
  alias vi >/dev/null 2>&1 || alias vi=vim
fi
```

- I.e. if this is bash, and you have not set an alias for “vi”, then set one to run “vim” when you type “vi”.

Example

- Create a user jim, in group staff
- But how to set the group?
- You could do:

$ man adduser

- Usually commands also take the flag “-h”

```
$ adduser –h
adduser: invalid option -- h
Usage: useradd [options] LOGIN

Options:
  -b, --base-dir BASE_DIR       base directory for the new user account
  …
  -g, --gid GROUP               force use GROUP for the new user account
  …
```

```
$ adduser jim –g staff
$ tail -1 /etc/passwd
jim:x:502:100::/home/jim:/bin/bash
$ grep 100 /etc/group
staff:x:100:
```

Moving a uid or gid

```
$ tail -1 /etc/passwd
jim:x:502:100::/home/andrew:/bin/bash

                                $ grep 100 /etc/group
                                staff:x:100:

$ ls -lnd /home/jim
drwx--x--x. 6 502 100 4096 Mar 27 11:53 /home/jim
$ ls -lan /home/jim
drwx--x--x. 6 502 100   4096 Mar 27 11:53 .
drwxr-xr-x. 6   0   0   4096 Jul 13  2007 ..
-rw-------. 1 502 100  10553 Apr  8 14:48 .bash_history
-rw-r--r--. 1 502 100 747954 Dec 20  2007 planner.zip
```

Useful Commands

```
$ chown jim.staff filename
$ chown jim filename
$ chgrp staff filename
```

When a User logs in

- When a user logs in, the appropriate . files are executed (.login, .cshrc, etc).
- If you want to change to a different user, you could log out and log in again, or you could do
  - su – gordon (change to the gordon user)
  - su – (change to root)
- Without the “-”, you still change users, but the . scripts don’t get executed.
- To go back to the previous user, press CTRL-D

FILE SEARCHING

A file CONTAINING something

- You are looking for a file containing “gordon”
- You think it is in /etc/ something

```
$ grep “gordon” /etc/*
/etc/group:gordon:x:500:
/etc/group-:gordon:x:500:
/etc/gshadow:gordon:!!::
/etc/gshadow-:gordon:!!::
/etc/passwd:gordon:x:500:100:Dr Gordon …
```

A FILENAME containing something

- E.g. You know somewhere in /etc there is a filename with the word “host” in it.

$ find /etc –name ‘*host*’

/etc/hosts.deny

/etc/ghostscript

/etc/ssh/ssh_host_dsa_key.pub

- Find can find on a range of metadata, not just names.
  This includes include sizes, permissions, types, ownership, and combinations of tests.

Find to do something

- Usually find prints the things which match.
- You can get it to execute instead.
- Here you want to find all files called core, and delete them:

```
$ find . -name core -print -exec rm {} \;
```

- I didn’t invent the syntax, so don’t blame me…

Discussion

- A user keeps getting logged out each time they log in… why?

Discussion

- A user finds their ls command is “broken”… why?

Discussion

- Here are some past exam questions you should now be able to answer:

Question 1

- What is the function of “su”, and what is the difference between “su – gordon" and “su gordon"?

Question 2

- What type of files would you expect to find in /sbin?

Question 3

- Consider the following line:

  gordon:x:44:

  In which file in /etc would you expect to see such a line, and what does it mean?

Discussion / practical exercise

What happens to file permissions and owner when copying?

To test your answer:

- Make sure you are logged in as demo or alice, NOT as root
- Make a new directory in your home directory called "etccopy". Copy all files which begin with a "u" and end with ".conf" from /etc to your new directory. (you may need sudo)
- Do a long listing of the original and the copied files. What do you find?

CSN09703
Networked Services
Apache – Security Discussion

Module Leader: Dr Gordon Russell

Lecturers: G. Russell

Apache Security

Security

- Hackers often consider a web server a good hacking target
- You should be very careful how apache is configured.
- The main problem is CGI scripts
  - CGI is a program which runs when you view a page.
  - Its output is sent back to the user's browser.
  - As it is an active process it can do permanent things to your server.

Simple CGI: who.cgi

```
#!/bin/sh
echo 'Content-Type: text/html; charset=ISO-8859-1'
echo
echo '<body><pre>'
whoami
env
echo '</pre></body>'
```

http://servername/who.cgi

apache SERVER_SIGNATURE=*Apache/2.0.51 (Fedora) Server at servername Port 80*
UNIQUE_ID=umn4CZKwogYAADNFYkcAAAAI
HTTP_KEEP_ALIVE=300
HTTP_USER_AGENT=Mozilla/5.0 (Windows; U; Windows NT 5.1; en-GB; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1
SERVER_PORT=80
HTTP_HOST=servername
DOCUMENT_ROOT=/home/gordon/public_html

Issues

- This cgi program only prints.
- However, it could also delete things, or transfer data, copy passwords, etc.
- A hacker is rarely wanting distruction.
- Hackers want access! This requires either
  - Transferring hacking programs to the server
  - Copying files from the server (e.g. /etc/passwd).

Ideas

- Make sure apache runs as a user just for the server
  - The user “apache” is commonly used here.
  - In the httpd.conf, make sure there is:

    user apache

    group apache
- Hide the apache version number.
  - Might be useful if a hacker is searching for a buggy apache version.
  - In httpd.conf

    ServerSignature Off

    ServerTokens Prod

- Don’t allow apache to ever give pages from “/”

```
<Directory />
    Require all denied
</Directory>
```

- Do you really need directory browsing?

```
Options -Indexes
```

- The apache user should not own its conf files

```
$ chown -R root:apache /etc/httpd
$ chmod -R u=rwx,g=r,o-rwx /etc/httpd
```

- Do not allow apache to surf the web:

```
$ iptables -A OUTPUT -m owner
                    --uid-owner apache
              -m state --state NEW
                    -j DROP
```

Discussion

Question 1

- How can iptables-based firewalls in Linux help improve the security of an Apache server?

Question 2

- You want to secure apache so that all web requests can only use the characters a-z, “.”, and “/”. If they don’t then display the contents of “/noway.html”.