FANSHAWE

INFO-6003

O/S & Application Security

Week 12

FANSHAWE

Agenda

- Introduction to Linux, Unix and Unix-Like Operating Systems
- Linux File Security
- Access Control
- Introduction to Pluggable Authentication Modules (PAM)

FANSHAWE

Introduction to Linux & Unix

FANSHAWE

UNIX Distributions

- UNIX as an operating system was developed at Bell Labs
  - Developed in the C programming language
  - Used widely in Universities in 70s and 80s
  - University of California at Berkley
    - Developed BSD Unix (Berkley Software Distribution)
- There are many commercial versions of UNIX
  - Oracle, Solaris
  - IBM, AIX
  - Hewlett Packard, HP-UX
  - Apple, Mac OS X

FANSHAWE

UNIX Distributions

- The Open Group is responsible for a common standard definition for UNIX
  - Consortium of commercial & government agencies
  - Sun, IBM, HP, NEC, NASA, etc.
- Linux has its roots in UNIX
  - Based on the POSIX (Portable Operating System Interface) rules developed for UNIX
  - Reversed engineered from UNIX

Linux Roots

- Minix was a Unix like OS written by Andrew Tanenbaum
  - Designed to work on intel386 processor
  - Linus Torvalds used Minix as his inspiration to create Linux
- Development of Linux started by Linus Torvalds
  - Posted the kernel on Internet and it was picked up by many others who helped contribute code
  - Many individuals and groups contribute utilities and applications
    - This is a strength and a weakness

FANSHAWE

UNIX-Like Operating Systems

- FreeBSD is a Unix-like operating system
  - Not a clone of UNIX
  - But works like UNIX
- OpenBSD project develops many tools and is considered the most secure UNIX-Like system
  - OpenSSH – Secure Shell
  - OpenNTPD – Network Time Protocol Daemon
  - OpenOSPFD – Open Shortest Path First routing protocol
  - OpenSMTPD – Simple Mail Transfer Protocol daemon

Linux Distributions

FANSHAWE

Linux Distributions

- A distribution is a collection of software applications running on top of a Linux Kernel
  - Distro for short
- Distributions are maintained by companies or communities of volunteers, with some crossover
  - Commercial – entirely commercial version
  - Commercially Backed – community version, but supported by company
  - Community – entirely community driven

Linux Distributions

Page Hit Ranking

Data span: Last 6 months Go

Rank	Distribution	H.P.D*
1	Mint	3109▲
2	Debian	1941▲
3	Ubuntu	1531▲
4	openSUSE	1452▲
5	Fedora	1115▲
6	Manjaro	1044▲
7	Mageia	989▼
8	CentOS	872▬
9	Arch	757▲
10	Kali	732▬
11	Android-x86	681▲
12	LXLE	666▬
13	Zorin	648▲
14	PCLinuxOS	581▲
15	Puppy	575▬
16	elementary	529▬
17	Lubuntu	526▬
18	Lite	500▲
19	Ubuntu MATE	481▲
20	deepin	471▲

- There are many distributions available
- DistroWatch keeps a ranking of the top 100

http://distrowatch.com

Notice that Kali Linux is in the top 10?

Linux Distributions

- Various distributions do have common characteristics
  - Same kernel releases
  - Same source code
  - Same basic commands and applications
- Many differences
  - Installed utilities and tools
  - Default services installed
  - Cryptographic methods used for passwords and other security features

Security Challenges

Linux Can Be Complex

- At its core, Linux is a networked operating system
  - Many network services are installed by default
  - Some not secure by default – ftp, telnet, rsh
  - Hackers can target these services, or the scripts that start & configure services
  - The original goal of developers was to provide easy connectivity between systems
    - Security has been added on over time

Security Challenges

- A full installation of Red Hat Enterprise Linux has over 1000 application and library packages
- An experienced admin would only install the required packages and server applications
- An inexperienced admin may install more packages than necessary
  - DNS, DHCP, telnet,
- These could run on start up with out admin knowledge
  - Increases attack surface

Security Challenges

- Linux distributions still more or less require a computer geek to deploy a secure usable machine
  - Knowledge of installation options
  - Many distros only have command line interfaces
- New tools have been developed to make it easier for inexperienced users to deploy a Linux system, but there is always a tradeoff
  - Security Tradeoff:
    - Easier to use = Easier for a hacker to exploit

Security Challenges

- Many subtle & major differences
  - Different tools installed by default at startup depending on the distribution
  - New users often chose to install all services rather than pick from a list of hundreds of tools modules and libraries
- Some newer Linux distributions now give users access to commands that previously had been limited to root
  - Increases the chance for misconfigurations

Security Challenges

- Many Linux components are written by 3rd parties
  - Commercial, academic, freelance
  - Each would be responsible for patches to the software they have written
  - Quality of secure programming practices will vary
  - Users may have to search for updates from each vendor or group responsible for the software
- Major distributions such as Debian have centralized methods for updating the OS and software packages

Security Challenges

- Open source programming model
- Many programmers contribute to the Linux application base
  - Are all contributors following secure coding practices?
    - Buffer overflows
    - Checks for code injections
- Who is responsible for patching and notification of security problems with free software
  - 1000s of programmers from over a 100 different countries

Type of Distro & Security

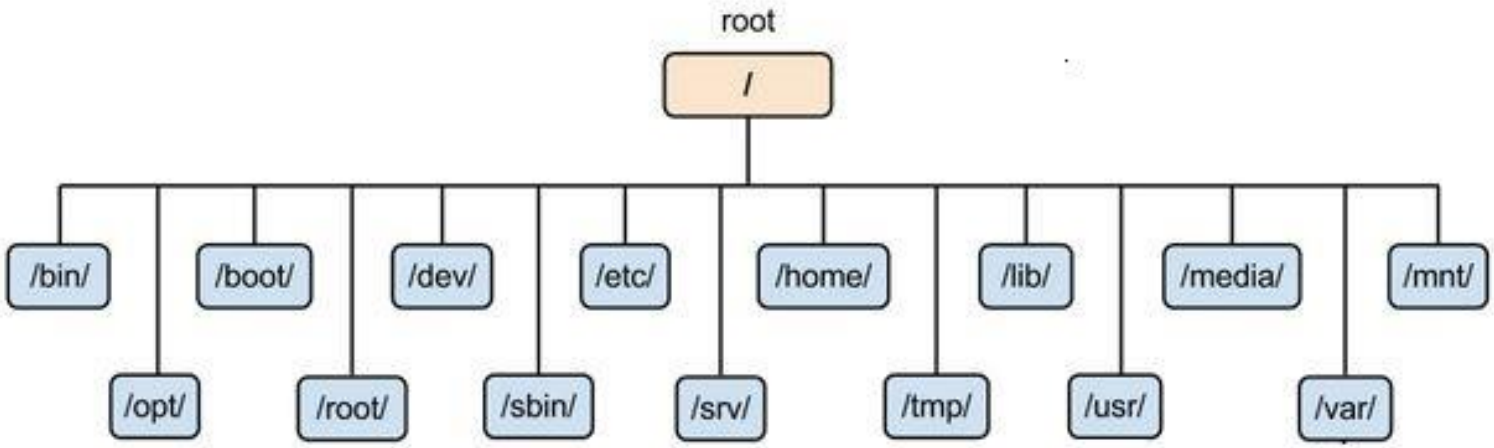
- Most server operating systems and applications have been tested over the years
  - This makes them very secure and more trustworthy
- Newer software for client desktops may not be subjected to the rigorous testing required to find software bugs
  - Linux suffers from the same coding problems as programs developed for Windows or Mac OSX

Linux File Security

FANSHAWE

Linux File System

- Everything in Linux is a File
  - The same access controls can be used to lock down any file
- No concept of individual “Drives” like Windows
- Has a single file system based on an inverted tree starting at the / (root)



FANSHAWE

Linux File Permissions

- The Linux file system has 3 basic permissions
  - Read
  - Write
  - Execute
- These permissions can be granted to
  - User – Owner of the file (or directory)
  - Group – All members of a group
    - When a new user account is created a group account with the same user name is also created
  - Others (world) – All other users

FANSHAWE

Linux File Permission Basics

- **Read** – Gives the user the ability to read the contents of the file or directory
- **Write** – Gives the user the ability to modify or write to a file or directory
- **Execute** – Gives the user the ability to execute a file (such as a script) or view the contents of a directory

FANSHAWE

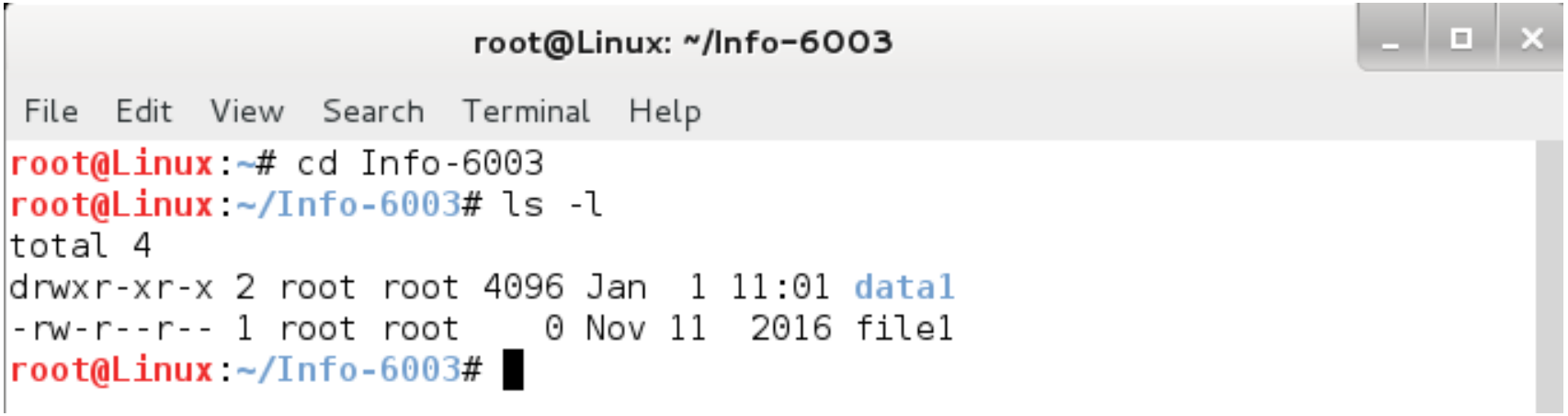
Linux File Permissions

User Permissions

Group Permissions

Others (All) Permissions

Linux File Permissions



- The screen capture above shows the content of the Info-6003 directory file
  - sub directory file data1 (**d** for directory type file)
    - read/write/execute for the user
    - read/execute for group & other
  - file file1 (- indicates file type file)
    - read/write for the user
    - read only for group & other

FANSHAWE

Linux File Permissions

root@Linux: /bin

File Edit View Search Terminal Help

```
1308242 -rwxr-xr-x  1 root root  26792 Aug  4  2012 ntfscat
1308243 -rwxr-xr-x  1 root root  30816 Aug  4  2012 ntfsck
1308244 -rwxr-xr-x  1 root root  30888 Aug  4  2012 ntfscluster
1308245 -rwxr-xr-x  1 root root  34984 Aug  4  2012 ntfscmp
1308246 -rwxr-xr-x  1 root root  22592 Aug  4  2012 ntfsdump_logfile
1308247 -rwxr-xr-x  1 root root  39088 Aug  4  2012 ntfsfix
1308248 -rwxr-xr-x  1 root root  55480 Aug  4  2012 ntfsinfo
1308249 -rwxr-xr-x  1 root root  31984 Aug  4  2012 ntfsls
1308250 -rwxr-xr-x  1 root root  30832 Aug  4  2012 ntfsmftalloc
1308251 -rwxr-xr-x  1 root root  30888 Aug  4  2012 ntfsmove
1308252 -rwxr-xr-x  1 root root  39016 Aug  4  2012 ntfstruncate
1308253 -rwxr-xr-x  1 root root  39568 Aug  4  2012 ntfswipe
1308254 lrwxrwxrwx  1 root root     14 Jan  8  2014 pidof -> /sbin/killall5
1308255 -rwsr-xr-x  1 root root  36136 Apr 12  2011 ping
1308256 -rwsr-xr-x  1 root root  36896 Apr 12  2011 ping6
1308257 -rwxr-xr-x  1 root root  93120 Mar 28  2013 ps
1308258 -rwxr-xr-x  1 root root  35360 Jan 26  2013 pwd
```

- The above listing of the /bin directory shows the files are owned by the root user but read & execute permissions have been granted to any other user for most files

FANSHAWE

Linux File Permissions

root@Linux: /etc

File Edit View Search Terminal Help

```
1048922 drwxr-xr-x   2 root   root      4096 Jan  8  2014 sgml
1047550 -rw-r-----   1 root   shadow    1323 Jan  8  2014 shadow
1048928 -rw-r--r--   1 root   root       103 Jan  8  2014 shells
1048929 drwxr-xr-x   2 root   root      4096 Jan  8  2014 siege
1048932 drwxr-xr-x   2 root   root      4096 Jan  8  2014 skel
1048937 -rw-r--r--   1 root   root      7059 Jun 19  2011 smartd.conf
1048938 drwxr-xr-x   3 root   root      4096 Jan  8  2014 smartmontools
1048942 -rw-r--r--   1 root   root      1132 Feb 25  2013 smi.conf
1048943 drwxr-xr-x   2 root   root      4096 Jan  8  2014 snmp
1048947 drwxr-xr-x   3 root   root      4096 Jan  8  2014 sound
1048952 drwxr-xr-x   4 root   root      4096 Jan  8  2014 speech-dispatcher
1048971 drwxr-xr-x   2 root   root      4096 Jan  8  2014 ssh
1048975 drwxr-xr-x   4 root   root      4096 Jan  8  2014 ssl
1049463 -rw-r--r--   1 root   root       771 Jan  8  2014 staff-group-for-usr-local
1049464 drwxr-xr-x   2 root   root      4096 Jan  8  2014 stunnel
1049466 drwxr-xr-x   2 root   root      4096 Jan  8  2014 subversion
1049469 -r--r-----   1 root   root       669 Mar  1  2013 sudoers
```

- The above listing of the /etc directory shows that the “others” user is not allowed to read the shadow or sudoers files

chmod

- The chmod command allows the owner of the file to change permissions
  - Each permission is given a numeric value based on the octal numbering system
  - R W X
  - 4 2 1
- The sum of the permission values in the chmod command assigns the permission
  - 4 – read only
  - 5 – read/execute
  - 6 – read/write
  - 7 – read/write/execute

FANSHAWE

chmod Octal Examples

- chmod 755 file1
  - Will give user (owner) rwx (7=4+2+1)
  - Group r-x (read & execute) (5=4+1)
  - Other r-x (read & execute) (5=4+1)
- chmod 764 file1
  - Will give user (owner) rwx (7=4+2+1)
  - Group rw- (read & write) (6=4+2)
  - Other r-- (read) (4=4)

FANSHAWE

chmod Octal Examples

- chmod 755 file1
  - Will give user (owner) rwx (7=4+2+1)
  - Group r-x (read & execute) (5=4+1)
  - Other r-x (read & execute) (5=4+1)
- chmod 764 file1
  - Will give user (owner) rwx (7=4+2+1)
  - Group rw- (read & write) (6=4+2)
  - Other r-- (read) (4=4)

chmod Operators

- In addition to the octal settings for permissions in Linux systems, you can also use Operators
  - u = user
  - g = group
  - o = others (world/all)
  - a = everyone of the above (ugo)

chmod Operator Examples

- chmod o+w data1
  - Will give others w (write on data1 directory)

```
File  Edit  View  Search  Terminal  Help
root@Linux:~/Info-6003# ls -l
total 4
drwxr-xr-x 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003# chmod o+w data1
root@Linux:~/Info-6003# ls -l
total 4
drwxr-xrwx 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003#
```

FANSHAWE

chmod Operator Examples

- chmod g-x data1
  - Will take away execute on data1 directory to group

```
File  Edit  View  Search  Terminal  Help
root@Linux:~/Info-6003# ls -l
total 4
drwxr-xr-x 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003# chmod g-x data1
root@Linux:~/Info-6003# ls -l
total 4
drwxr--r-x 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003#
```

FANSHAWE

Keep in mind…

- Taking away execute (x) permissions will prevent that user from listing the contents of the directory

```
File  Edit  View  Search  Terminal  Help
root@Linux:~/Info-6003# ls -l
total 4
drwxr-xr-x 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003# chmod g-x data1
root@Linux:~/Info-6003# ls -l
total 4
drwxr--r-x 2 root root 4096 Jan  1 11:01 data1
-rw-r--r-- 1 root root    0 Nov 11  2016 file1
root@Linux:~/Info-6003#
```

More File Permissions

- Only the owner of a file should have write permissions for the file
- Allowing others to change or modify a file can be a security issue
  - Especially important for configuration files
- Linux has many configuration files (.conf) that only the root has rwx permissions for
- There are many other commands to manipulate the file system access
  - chown, chattr, umask, etc.

umask

- umask can be used to restrict the default permissions on files created by users
- Prevents unintentional assignment of higher permissions
- Default umasks:
  - Root umask 022
  - User umask 002
- Default Permissions:
  - 777 for directory type files
  - 666 for files type files

FANSHAWE

umask

- To lock down root's files a umask 077 can be set for root
  - 0-user, 7-group(rwx) 7-other(rwx)
- The mask value is inverted and then a Boolean AND function is used with the default file permissions to get the resultant permission
  - 7 – 111 inverted is 000
  - When 000 is ANDed to any number the resultant will be 000 no read/write/execute permissions assigned by default

FANSHAWE

umask

- Typically when a new file is created by a user the system will assign permissions 666 (rw-rw-rw-)
  - Permission 4(r) 2(w) 1(x)
  - 5 = r-x, 6 = rw-, 7 = rwx
- Umask is set in the /etc/profile file to apply to all users
- Typical umask setting for users is 002
  - 2 in binary is 010 (gives -w-) inverted is101 (5)
  - Would allow r-x permission if set by user
  - When 101 (mask) is ANDed to 110 the default permissions
  - Result is 100 or (4) read only

Access Control

FANSHAWE

User Accounts

- User accounts fall into three main categories
  - Super User / Root
  - System Accounts
    - Such as mail, nobody, apache, syslog, etc.
    - Used to control the access these services have to other system resources
  - Normal User Accounts
    - Typical end user accounts

User Accounts

- Root account is a Super User account
  - Other Super User accounts can be created
  - Any account with the userid and groupid set to 0 is a Super User account
- Best practice is to login as a normal user for most tasks
- Change to a Super User account only to perform tasks denied to normal users
  - Modifying configuration files, creating new users, resetting passwords, etc.

FANSHAWE

Switching Users

- su – substitute user
  - You will sometimes see this referred to as super user because it is often used to switch to the root account
- Allows you to switch to another account without having to logout of current session
  - su TEST2 (would switch to the TEST2 account)
- Opens a shell in the environment as the new user
  - You need to know the password
- Normal practice is to logon with a normal user account and use su to switch to the root account as required

FANSHAWE

Super User Accounts

- Changing to super user account with su, requires the user to enter root's password and a new shell is created
  - Anything run in that shell is run as root
  - When exiting su the directory path will be returned to the previous user shell
- There are methods that can be used to perform tasks as root, without logging on as root
  - SUID/SGID bits & sudo command
  - Both ways *can* give users root permissions to run commands or executable files

FANSHAWE

SUID & SGID

- Sets an executable file to take on permissions of the owner or group of the file rather than the user that executed file
  - User that executes the file temporarily assumes the permissions of the owner of the file (often root)
- For Example
  - passwd program uses SUID to allow users to change their passwords
  - /etc/password is only accessible as root in some systems

FANSHAWE

SUID & SGID

- SUID
  - When a file with SUID is executed, the resulting process will assume the effective user ID given to the user
  - Enables users to be treated temporarily as root (or another user)
- SGID
  - When a file with SGID is executed, the resulting process will assume the group ID given to the group

Privilege Escalation

- Can be a security problem if the executable file or program access granted with SUID has a buffer overflow or other security vulnerability
  - Once exploited, attacker now has root access because the program ran in the root security context
- Note: Linux kernel will not honor SUID bits set in a script file

Security Measures

- Users should not be allowed to run SUID executables from their home directories
- The /etc/fstab configuration file options can be used to prevent the following from the home directory
  - nosuid prevents SUID and SGID enabled executables from running
  - noexec to prevent executable programs
  - nodev to stop devices files from being recognized in the /home directory

FANSHAWE

SUID

- SUID is denoted on a user with an **s** or **S** for the execute (x) permission
  - rw**s**r-xr-x
    - A lower case s if the user had execute (x) permissions before SUID was set
  - rw**S**r-xr-x
    - An upper case S if the user didn't have the execute (x) permissions before SUID was set

Setting SUID

- The chmod command is used to set SUID on an executable file or script
- chmod u+s filename
  - File Before: rwxr-xr-x filename
  - File After: rwsr-xr-x filename
  - The user has su permissions on file execution but the group & other still have normal user permissions
- Alternate command format
  - chmod 4755 filename
  - Would have the same result

FANSHAWE

Special Permissions

- Special Bits
  - 4, SUID
  - 2, SGID
  - 1, sticky bit
- chmod 2755 file1
  - rwx r-s r-x file1
  - The SGID permission is now set for file1

FANSHAWE

SUID

- The SUID bit can be set for a number of functional commands such as: su, ping, mount, umount to control access
- The command will be executed without a prompt for super user or owner password
- Can be a security issue because no control is placed on which user can execute the command
- Should not be used with commands that could allow users to elevate permissions
- The ls command on a directory will mark the SUID files with a different colour
  - If supported, or set

Sticky Bit

- On files the sticky bit is used to encourage the OS to keep an executable program resident in memory after it has been executed so that it is available for immediate use at next request
  - Does not have to reload
- On directories the sticky bit is used to lock the files in a shared directory so that only the owner or root can change the files

FANSHAWE

Sticky Bit

- The sticky bit is generally used at the directory level
  - Normally, if a user is given write permissions to a directory but not the files in the directory they can still delete or modify all the files in the directory!
- Sticky bit only allows a user to edit or delete the files they added to the shared directory
- Sticky bit is denoted with the letter t
  - chmod 1777 dirA
  - drwxrwxrwt dirA

FANSHAWE

File Attributes

- File Attributes can be viewed with the lsattr command
- File attributes can be assigned with the chattr command
- These are different than file permissions!

File Attributes

- Security Related Attributes
  - Immutable attribute
  - File cannot be changed in any way including permissions
    - chattr + i file1
    - adds the immutable attribute to file1
  - The immutable attribute can only be set by root account

File Attributes

- Security Related Attributes
  - Secure Delete
  - When a file is deleted with the rm command the entire file is overwritten with zeros
  - A normal file delete does not actually remove the data 1s & 0s from the disk
    - chattr + s file1
    - adds the secure delete attribute to file1

Storing Passwords

Passwd Entry

- Modern distros store user information like passwords in /etc/shadow
- When /etc/shadow is in use, the location where you would normally see the hashed password in /etc/passwd is replaced with an x
  - Example of /etc/passwd entry

amack:x:1001:1001:Art,Mack,,:/home/amack:/bin/bash

- **amack** is the username
- **x** is the placeholder for the password that is stored in /etc/shadow
- **1001:1001** are the user and group IDs
- **Art,Mack** is the user's name
- **/home/amack** is the location of the user's home directory
- **/bin/bash** identifies which shell the user will be using

FANSHAWE

File Permissions

- Permissions for /etc/passwd and /etc/shadow
  - /etc/shadow has the more restrictive permissions
- Only root has access to /etc/shadow

```
root@artmack:~# ls -l /etc/passwd
-rw-r--r-- 1 root root 2879 Feb 16 00:36 /etc/passwd
root@artmack:~# ls -l /etc/shadow
-rw-r----- 1 root shadow 1637 Feb 16 00:36 /etc/shadow
root@artmack:~#
```

FANSHAWE

Hash Algorithms

- Passwords are stored in the /etc/shadow with a variety of cryptographic methods
  - DES, Blowfish
  - MD5 Hash with password salt
  - Ubuntu uses SHA256-SHA512
- Method used can be identified by the characters at the beginning of the hash
  - MD5 represented by 1
  - Blowfish represented by 2
  - DES represented by _ (underscore)
  - SHA256 represented by 5 SHA512 uses 6
- Hackers use this to know which brute force method or rainbow table is required

FANSHAWE

Password Salts

- Salt is a random number of bits added the password to change the hash output
- Makes it harder to guess real input to hash
- Keeps users from guessing other user's passwords
- If 2 users have the same password they won't have the same hash value when a salt is added

Changing Hash

- By default Ubuntu uses a salted SHA512 hash
  - Strong Hash Algorithm with 512 bit hash
  - Shown in the /etc/shadow file as 6followed by a 512 bit character string that represents the password
- Administrators can modify the Pluggable Authentication Modules to change the cryptographic method used to store the password in the /etc/shadow file

Pluggable Authentication Modules

PAM

- Quote from Linux-PAM systems admin guide
  - "It is the purpose of the Linux-PAM project to separate the development of privilege granting software from the development of secure and appropriate authentication schemes
  - This is accomplished by providing a library of functions that an application may use to request that a user be authenticated

PAM

- Pluggable Authentication Modules
- PAM provides a library containing functions for proper authentication procedures
- Allows for a separate module to provide authentication so it does not have to be in the program API
- The PAM configuration files for all services are found in /etc/pam.d directory

PAM

- The services using PAM to authenticate the user are listed in the pam.d directory
  - Will vary by install, based on applications and services installed
- Examples
  - su
  - sudo
  - passwd
  - login
  - sshd

PAM

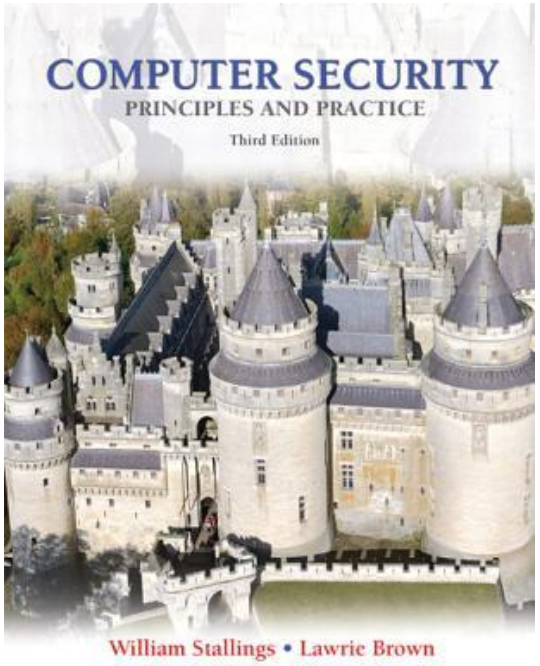
- PAM configuration files have 3 entries
  - The first entry indicates one of four categories which identify different types of modules for controlling access to a particular service
  - The second field in each entry is called the control flag and determines the action taken when the module succeeds or fails
  - The third filed contains the values

```
password   [success=2 default=ignore]   pam_unix.so obscure sha512
```

  - More detail on PAM after Test-02

Homework

- Read Chapter 12 Sections:
- 12.6 – Linux/Unix Security



Lab 09 – Linux Security

FANSHAWE

Lab 09 Details

- Configure a Linux VM
- Manage users
- Edit file permissions
- Change file attributes
- Modify PAM Module to change hashing mechanism