User Management

Types of Users:

- Super User / root user
- System User / In built user
- User created users

The Superuser

- By default, one account has elevated privileges to issue any command, access any file, and perform every function
- Superuser, a.k.a. root
  - Technically, can change to anything – but don't
- User and group number 0
- Must limit use of root
  - Inexperienced users can cause serious harm
  - Use of root for non-privileged tasks unnecessary and can be open to attack
  - Security and privacy violations – root can look at anyone's files
- Limit/restrict the root ssh or remote login to system
- Ensure a strong password

Superuser Privileges

- What usually works best is short periods of superuser privilege, only when necessary
- Obtain privileges, complete task, relinquish privileges
- Most common ways are su and sudo
- Can also use the setuid/setgid method (Ch. 4), but not recommended

Su

- Short for *substitute* or *switch user*
- Syntax:
  - `su [options] [username]`
  - If `username` is omitted, `root` is assumed
- After issuing command, prompted for that user’s password
- A new shell opened with the privileges of that user
- Once done issuing commands, must type exit

sudo

- Allows you to issue a single command as another user
- Syntax:
```
sudo [options] [-u user] command
```
- Again, if no user specified, root assumed
- New shell opened with user’s privileges
- Specified command executed
- Shell exited

sudoers

- Must configure a user to run commands as another user when using `sudo`
- Permissions stored in `/etc/sudoers`
- Use utility `visudo` to edit this file (run as root)
- Permissions granted to users or groups, to certain commands or all, and with or without password being required

User Administration

- User configuration stored in `/etc/passwd`
- File got it’s name because it originally contained passwords as well
  - Security problem – too many processes need to read `passwd`
  - A shadow file used now instead (more in a sec)
- Each line contains info for one user

Adding User:

- Syntax:
  `useradd [options] [-g group] [-d home] [-s shell] username`
- -g to define user’s initial group
- -d to define user’s home directory
- -s to define user’s default shell
- Other options for expiration, using defaults, etc

Deleting User:

- Syntax: `userdel [-r]` *`username`*
- -r to delete home directory and it’s contents

Modifying User:

- Syntax:
  - `usermod [options]` *`username`*
- Options are pretty much identical to those of useradd
- Also, -l to change the user’s login name
- And –G to list additional groups to add user to

Group management:

- Group info housed in `/etc/group`
- Similar to user management
- Common commands:
  - `groupadd`
  - `groupdel`
  - `groupmod`

Passwd

```
jsmith:x:1001:1001:Joe Smith,(234)555-8910,email:joe@test.com:/home/jsmith:/bin/bash
```

- First field is username
- Second was password – now a dummy char
- Third is userid (uid)
- Fourth is groupid (gid)
- Fifth is GECOS field
  - Full name, contact info
  - Gen. Elec. Comprehensive OS
- Sixth is user’s home directory
- Seventh is user’s default shell

Passwd

- Originally `passwd` contained a user's password information
- How it works
  - User picks a password
  - A random number is generated (called the salt)
  - The salt and the password is passed into a hash function (a one-way cryptographic algorithm)
  - The salt and result are stored in ASCII
- Problem – user-level programs need to read `passwd`
  - Get user name, location
  - Home directory, shell
- So `passwd` was world readable
- So anyone on system could see a user's salted hash
- It's encrypted – what's the big deal???

Shadow

`jsmith:$1$CzzxUSse$bKJL9wAns39vlxQlBZ8wd/:13744:0:99999:7:::`

- Wasn't acceptable to have `passwd` world readable if it contained hashes
- So salted hashes moved to a new file
- `/etc/shadow`
- Format similar to `passwd`, one user per line
- Readable only by root
- First field is username
- Second is the salted hash or account status
  - `NP` or `!` or null for blank password
  - LK or `*` for locked/disabled account
  - !! for account with expired password
- Third is days since last password change
  - Measured from epoch (midnight UTC 1/1/1970)

Shadow

```
jsmith:$1$CzzxUSse$bKJL9wAns39vlxQlBZ8wd/:13744:0:99999:7:::
```

- Fourth is days until password is eligible to be changed
- Fifth is days before change is required
- Sixth is days before expiration to warn
- Seventh is days before account expires
- Eighth is days since epoch when account expires
- Ninth is unused/reserved