

# Linux Users and Groups

# Users

- Linux is an offshoot of Unix, which was designed to support multiple users from the beginning. (Microsoft DOS started as a single user system)
- Each user has a User ID (UID)
  - It also has a username, but the UID is what matters
  - If a user is created with the same UID as another (by mistake or maliciously) the accounts are essentially the same, even if they had different usernames
- The most powerful user is “root”, with UID 0
- **Do not** perform routine tasks as root, use it only when necessary
- Linux provides easy ways to switch from a non-privileged user to root (privileged user) when needed

# Groups

- Groups are containers that can hold multiple users
- In Linux, each user belongs to at least one group
  - Usually, Linux creates a group for each user, named same as the username
- Groups are used to make management easier
  - Suppose multiple users need access to one directory
  - Create a group that has the needed access
  - Put users in the group to grant access
  - Faster and easier to keep track of than giving access to each user separately
- Most distributions have groups to grant root access
  - The group named root is one, with Group ID (GID) 0
  - Redhat uses the group “wheel”, Ubuntu uses “sudo” (older versions use “adm”)

# Managing Users

- Most distributions have their own GUI tools for users and groups
  - GUI tools vary greatly between distributions
  - CyberPatriots starts with Ubuntu GUI tools
- CLI tools are pretty much standard across distributions
  - Easy to incorporate into scripts (add 100 new users by clicking? I think not.)
- `useradd [username]` creates a new user
  - By default the user has no password, no home dir, and is locked
  - username must be all lower case
- `adduser [username]` also creates a new user (better)
  - Also asks you for a password and creates user's home directory
- `passwd [username]` sets a user's password
  - If you omit the username, it sets your password

# Managing Groups

- `groupadd [groupname]` creates a new group
- `gpasswd -a [user] [group]` adds the user to the group
- `gpasswd -d [user] [group]` deletes the user from the group
- A user can belong to multiple groups but always has one primary group
  - primary group used during login and assigned to new files the user creates
- `usermod` also adds a user to group(s) but has a “gotcha”
  - It replaces existing supplementary groups unless you tell it to “append”
  - It can also be used to change a user’s primary group
  - <https://www.howtogeek.com/50787/add-a-user-to-a-group-or-second-group-on-linux/>
  - <https://www.cyberciti.biz/faq/howto-linux-add-user-to-group/>

# /etc/passwd (1)

- Usernames, UIDs and other user information stored in /etc/passwd
- /etc/passwd is readable by any user or application (world-readable)
- One line per user, fields are separated by colons, this format:

```
username:password:UID:GID:GECOS:home_dir:shell
```

- `password` was a cryptographic hash of the user's password
  - When people learned to crack hashes, the password hash was moved
  - Password hashes are now stored in /etc/shadow, which is readable by root
  - Now the password field contains an "x" to indicate that the password hash is stored in /etc/shadow
  - See Notes for this page (View > Notes Page)

# /etc/passwd (2)

- `UID` is the user's unique ID number
- `GID` is the ID number of the user's primary group
- `GECOS` is a comment field, often the user's full name
  - This dates back to Unix at Bell Labs (1960's), which used print servers that ran the General Electric Comprehensive Operating Supervisor (GECOS). The GECOS field held the user's ID on the print server.
- `home_dir` is the path to the user's home directory (usually `/home/username`)
- `shell` is the path to the user's login shell, the program run when the user opens a terminal, often `/bin/bash` or `/bin/sh`

# Sample /etc/passwd file

```
john@svgs-ubuntu18: ~  
File Edit View Search Terminal Help  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin  
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

<snip>

```
hplip:x:118:7:HPLIP system user,,,:/var/run/hplip:/bin/false  
geoclue:x:119:124::/var/lib/geoclue:/usr/sbin/nologin  
gnome-initial-setup:x:120:65534::/run/gnome-initial-setup:/bin/false  
gdm:x:121:125:Gnome Display Manager:/var/lib/gdm3:/bin/false  
john:x:1000:1000:John,,,:/home/john:/bin/bash
```

Username root has UID and GID 0,  
GECOS/name is root,  
home directory is /root,  
shell is /bin/bash

Username john has UID and GID 1000,  
GECOS/name is John,  
home directory is /home/john,  
shell is /bin/bash

Note that there are several accounts with the shell set to /usr/sbin/nologin or /bin/false. None of these can log in to a terminal. They are system accounts used for internal processes and applications. When trying to find accounts that don't belong, look at a clean installation of the same OS version and look for differences.

- Look especially for accounts that have a shell that allows them to log in (shell other than nologin or false)



```
/etc/shadow
```

- Password hashes and other info were moved to /etc/shadow when people learned that dictionary attacks could find many passwords
- Only privileged users (root) can read /etc/shadow
- Format (from <https://www.cyberciti.biz/faq/understanding-etcshadow-file/> )

vivek:\$1\$fnfffc\$pGteyHdicpGOfffXX4ow#5:13064:0:99999:7:::

<snip>

1 2 3 4 5 6

1 is the username, 2 is the password hash

3 – 6 explained in the Notes for this slide ( View > Notes Page)

The link above does a good job of showing how to set all the values in `/etc/shadow` for a user

# Sample /etc/shadow file

```
john@svgs-ubuntu18:~$ sudo cat /etc/shadow
root:!:17760:0:99999:7:::
daemon*:17737:0:99999:7:::
```

<snip>

```
john:$6$C6hSVPRs$axByh0xtSvCrZAwG/6ezojgVi76majPyX0.YfGamFQuB3SnH.dOTGI0au3ifHYPHT./
4oKyYFNWGFylpqMLq/:17765:0:99999:7:::
```

The hash for the user account root is set to “!”, which means the account is locked

The hash for the user account daemon is set to “\*”, which means no password set. You cannot log in to that account, but the OS may use it for internal tasks.

The hash for the user account john has three components separated by “\$”

6                      the hash function is the SHA512 algorithm

C6hSVPRs              the salt is C6hSVPRs \*see notes below\*

axByh0xtSvCrZAwG/6ezojgVi76majPyX0.YfGamFQuB3SnH.dOTGI0au3ifHYPHT./4oKyYF

NWGFylpqMLq/              this is the hash

CyberPatriots: When checking for unauthorized users on the system, check the ones that have hashes, since they can log in.

# /etc/group

- The /etc/group file lists all the group names, optional group passwords (x below means no group password) group IDs (GID), and group members

```
john@svgs-ubuntu18: ~  
File Edit View Search Terminal Help  
root:x:0:  
daemon:x:1:  
bin:x:2:  
sys:x:3:  
adm:x:4:syslog,john  
tty:x:5:  
<snip>  
cdrom:x:24:john  
floppy:x:25:  
tape:x:26:  
sudo:x:27:john  
<snip>  
geoclue:x:124:  
gdm:x:125:  
john:x:1000:  
sambashare:x:126:john
```

- Members of the group “root” (GID 0) have the same privileges as the root user (no members in this example)
- In Ubuntu, the adm and sudo groups also give root privileges (adm is replaced by sudo, and may be removed in the future)
- User john is a member of several groups in this example, including adm and sudo
- There are many groups created by the OS for control of devices and applications
- Use gpasswd and usermod to make changes

# su—substitute user

- Allows you to switch to a privileged account when needed
  - Use privileged accounts only when necessary to reduce risk
  - Changes your terminal to the new user until you type “exit”
- `su [username]`
  - Switch to the account specified in `username`
  - If `username` is omitted, switch to root
- `su -`
  - The single dash “-” option gives you the user’s environment as well
  - Some apps run better this way
- When `su` prompts for a password, use the password of the account you are switching to.

# sudo— “super user do”

- Execute one command as a super user
- Who can use `sudo`, and what commands they can execute, is configured in `/etc/sudoers`
  - edit `/etc/sudoers` with `sudoedit` or `visudo`
- Provides auditing—each command is logged
- `sudo` asks for your password
  - you are still using your account, just with more privileges
- `sudo -l` (lower case “L”) gives a listing of who can use `sudo` and what they can execute
- `sudo -i` gives you a root shell

# Ubuntu Uniqueness

- Ubuntu is used by brand new Linux users as well as power users
- Ubuntu encourages use of `sudo` for all actions requiring elevated privileges, so it locks the root account by default
- `sudo -i` will give you a full root shell for multiple commands
- `sudo su -` works the same as `su -` does in other distributions

```
john@svgs-ubuntu18:~$ sudo cat /etc/shadow
root:!:17760:0:99999:7:::
daemon:*:17737:0:99999:7:::
```

<snip>

```
john:$6$C6hSVPRs$axByh0xtSvCrZAWG/6ezojgVi76majPyX0.YfGamFQuB3SnH.d0TGI0au3ifHYPhT./
4oKyYFNWGFylpqMLq/:17765:0:99999:7:::
```

The root user has an “!” where the hash should be, so it has no password and cannot log in. To make Ubuntu work like other distributions, set the root password with `sudo passwd root`. To return to normal, lock the account with `sudo passwd -l root`.