Users, Group, Guest, Services, Processes, Logging, UFW

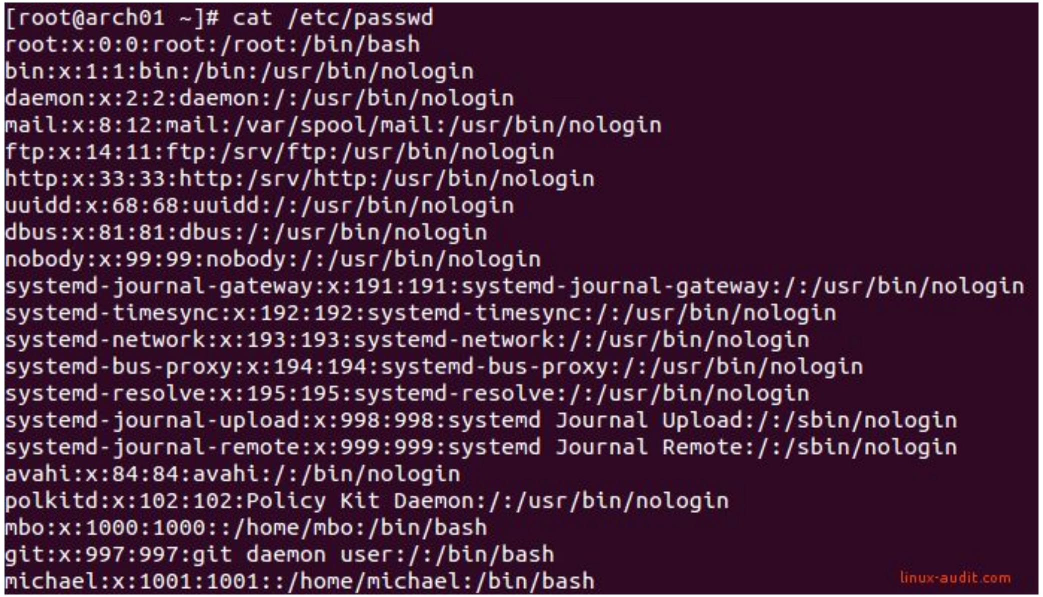
Mostly review, so make sure you know everything in here.

What two types of users are there in Ubuntu and Debian systems?

- What are the differences
- What possible vulnerabilities does that cause?

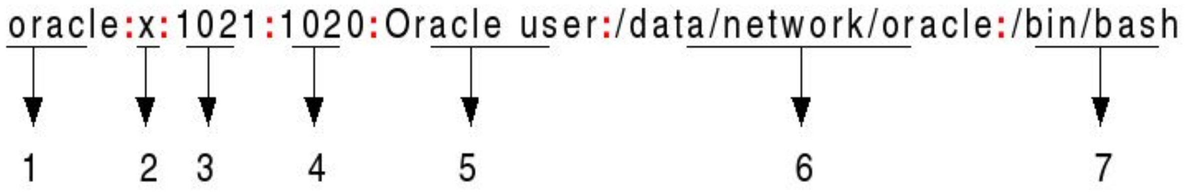
User notation: UID and other user info

- What is the purpose passwd file?
- What does each line mean?
- Any important vulnerabilities in this file?



```
[root@arch01 ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/bin/nologin
daemon:x:2:2:daemon:/:/usr/bin/nologin
mail:x:8:12:mail:/var/spool/mail:/usr/bin/nologin
ftp:x:14:11:ftp:/srv/ftp:/usr/bin/nologin
http:x:33:33:http:/srv/http:/usr/bin/nologin
uuidd:x:68:68:uuidd:/:/usr/bin/nologin
dbus:x:81:81:dbus:/:/usr/bin/nologin
nobody:x:99:99:nobody:/:/usr/bin/nologin
systemd-journal-gateway:x:191:191:systemd-journal-gateway:/:/usr/bin/nologin
systemd-timesync:x:192:192:systemd-timesync:/:/usr/bin/nologin
systemd-network:x:193:193:systemd-network:/:/usr/bin/nologin
systemd-bus-proxy:x:194:194:systemd-bus-proxy:/:/usr/bin/nologin
systemd-resolve:x:195:195:systemd-resolve:/:/usr/bin/nologin
systemd-journal-upload:x:998:998:systemd Journal Upload:/:/sbin/nologin
systemd-journal-remote:x:999:999:systemd Journal Remote:/:/sbin/nologin
avahi:x:84:84:avahi:/:/bin/nologin
polkitd:x:102:102:Policy Kit Daemon:/:/usr/bin/nologin
mbo:x:1000:1000::/home/mbo:/bin/bash
git:x:997:997:git daemon user:/:/bin/bash
michael:x:1001:1001::/home/michael:/bin/bash
```

Passwd file in depth


oracle:x:1021:1020:Oracle user:/data/network/oracle:/bin/bash
1
2
3
4
5
6
7

1. Username
2. Password
   a. What does x mean
3. UID
4. GID
5. Comment field
   a. Full name etc
6. Home directory
7. Shell directory, does not have to be shell

Make sure that the shell is `/bin/bash for users ONLY`

Important UID Rules (should be review)

- Any user with UID of 0 is root, make sure no other users besides user root should have UID of 0
- Hidden users have an UID of < 1000

Adding users

- We use the `adduser` command to add users through the terminal
    - another method that is a bit dirtier is adding the line in `/etc/passwd` file but that does not add a home directory or password

Configuring adduser.conf

- Everything in Linux is configurable
- Everything can be made more secure
- The configuration file is found at `/etc/adduser.conf`

Important configs

- `DHOME=[dir]`
  - Sets the directory for all user home directories created by the command, default is `/home`
- `DSHELL=[shell]`
  - Sets the shell for user, default is `/bin/bash`, if set to anything else you will have problems.
- `FIRST_SYSTEM_UID` & `LAST_SYSTEM_UID= 100` & `999` respectively
  - Only values these should ever be set to.
- `FIRST_UID` & `LAST_UID= 1000` & `299999` respectively
  - Only values these should ever be set to.

One last config

- in `adduser.conf` there is a config called `SKEL`
- This is by default set to the directory `/etc/skel`
- This is the skeleton directory for all **NEW** home directories
    - Contains default files like .bashrc
    - .bashrc: runs upon new terminal. Can set aliases here

- To understand what I mean, create a file using touch in the skeleton directory
- Now create a new user and look into their home directory.

Deleting users

- We use the `deluser` command to remove users
- We can also delete the line from the passwd file

- `deluser -group [group]`
  - Equal to `delgroup [group]`
  - Removes a group
- `deluser --remove-home [user]`
  - Removes home directory as well as deleting user
- `deluser --remove-all-files [user]`
  - Removes **ALL** files owned by user on the system

Configuring deluser.conf

- Just like `adduser`, `deluser` also has a config file
- Located at `/etc/deluser.conf`

Here are some important configs and their default values

- `REMOVE_HOME=0`
  - 0 - do not remove home by default
  - 1 - do delete home by default
- `REMOVE_ALL_FILES=0`
  - 0 - do not remove files by default
  - 1 - do delete files by default

Changing passwords

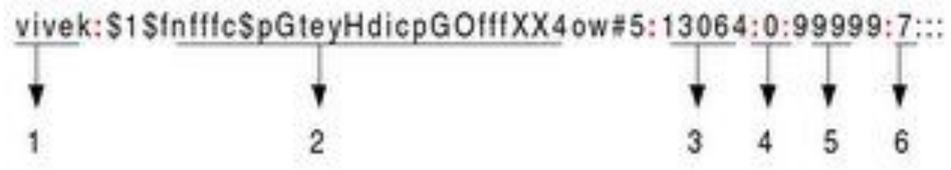
- To change passwords, we use the `passwd` command as a system admin

- `passwd -l [user]`
  - Locks user account, no one can log in to that account
  - Lock everyone who isn't on the readme because those people should not be allowed to log in
- `passwd -u [user]`
  - Unlock user account
- `passwd -S [user]`
  - Show user status
- `passwd -Sa [user]`
  - Show all user status'

Shadow File

- What is the purpose of this file?
    - How does this relate to the passwd file
- What useful information can we obtain from the shadow file?

Understanding the shadow file



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

1. Username
2. 13-char encrypted password
3. Last password change
4. Minimum days for a password
5. Maximum days for a password
6. When the account user is warned about password age
7. Future field, how many days after password expiration that account is disabled
8. Future field, date when account will expire, counted from Jun 1, 1970

More about the shadow file

1TDQFedzX$.kv51AjM.FInu0lrH1dY30

- The beginning of the encrypted password indicates the hash
  - 1 is MD5
  - $2a$ is Blowfish
  - $2y$ is Blowfish
  - 5 is SHA-256
  - 6 is SHA-512

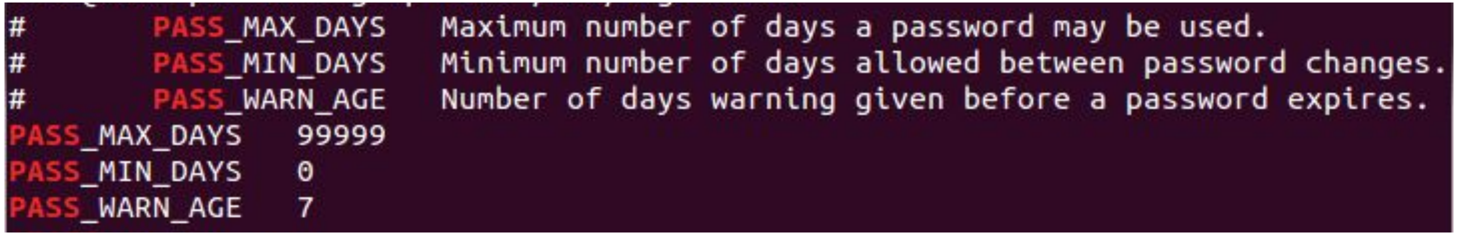
Changing shadow file parameters

- `chage` is used to modify these parameters on a **per user basis**
- Used like this
  - `chage [options] username`

- `chage -m 6 [user]`
  - Changes min days of user to 6
- `chage -M 15 [user]`
  - Changes max days of user to 15
- `chage -W 7 [user]`
  - `Warn days`
- `chage -I 5 [user]`
  - `Inactive days`

Changing shadow files parameters

- Global settings can be set globally by editing `/etc/login.defs`



```
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

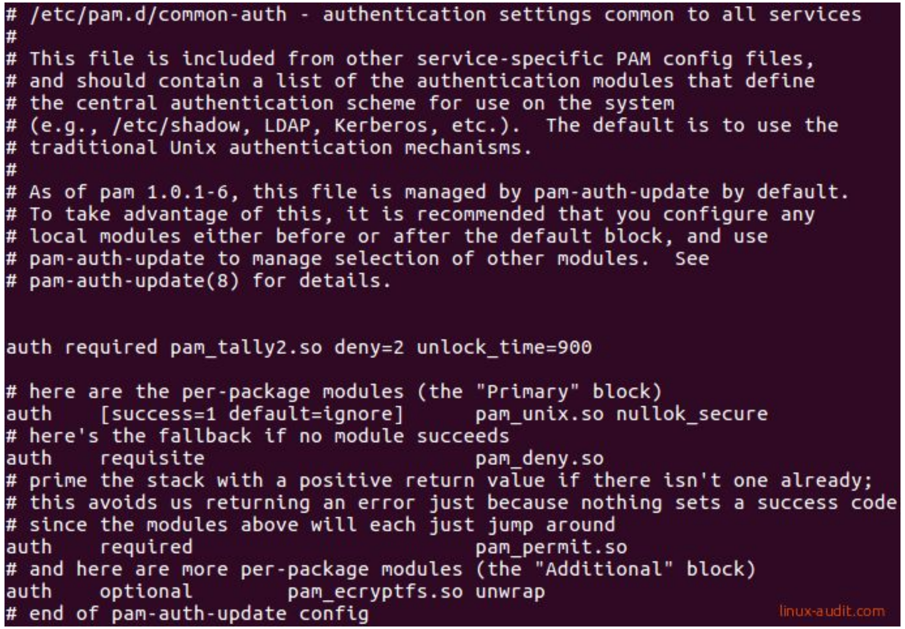
Pluggable Authentication Module

- Controls password authentication for most applications
- `/etc/pam.d/` contains all the configuration for all pam

- `sudo apt install libpam-cracklib`
  - Installs a PAM module that helps us ensure new passwords are secure, we'll go over how to use this

Common-auth

- /etc/pam.d/common-auth
- This file sets the authentication settings common to all applications, hence the name



```
# /etc/pam.d/common-auth - authentication settings common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of the authentication modules that define
# the central authentication scheme for use on the system
# (e.g., /etc/shadow, LDAP, Kerberos, etc.).  The default is to use the
# traditional Unix authentication mechanisms.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.


auth required pam_tally2.so deny=2 unlock_time=900

# here are the per-package modules (the "Primary" block)
auth    [success=1 default=ignore]      pam_unix.so nullok_secure
# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
auth    required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
auth    optional        pam_ecryptfs.so unwrap
# end of pam-auth-update config
```

Common-auth

Here is a sample configuration for the common-auth, it is long so we will go over the important configurations:

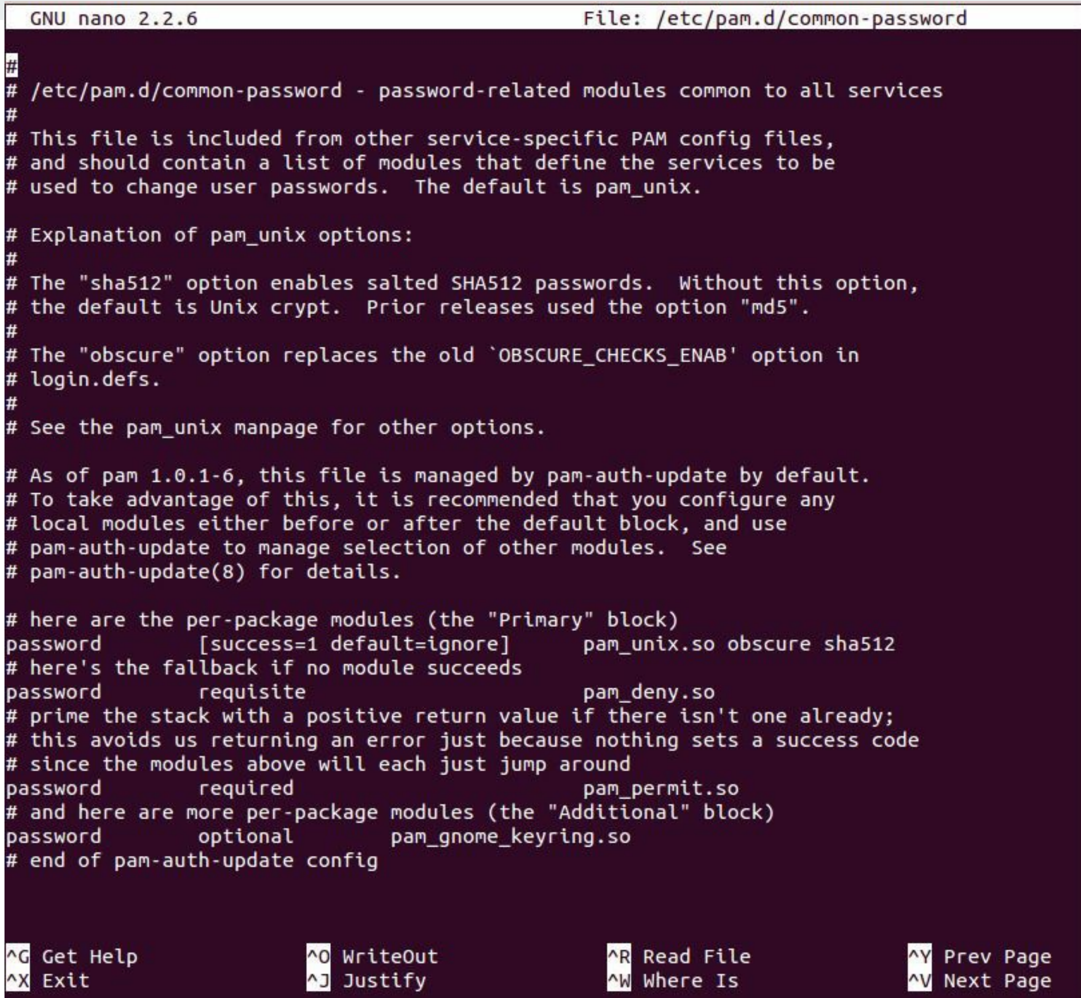
```
auth optional pam_tally.so deny=5 unlock_time=900 onerr=fail audit even_deny_root_account silent
```

- deny=5
  - Deny user if attempts exceed 5
- unlock_time=900
  - Locks user out for 900 seconds attempts exceed the amount defined
- audit
  - Logs account into syslog if attempts exceed the amount defined
- even_deny_root_account
  - Locks the root account, can't login with root

Note: never have the line "nullok" or even "nullok_secure". Think about what null means and what ok means

Common-password

- `/etc/common-password`
- This file controls how passwords are created, set, stored, and recalled throughout the system.



```
  GNU nano 2.2.6                                    File: /etc/pam.d/common-password

#
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords.  The default is pam_unix.

# Explanation of pam_unix options:
#
# The "sha512" option enables salted SHA512 passwords.  Without this option,
# the default is Unix crypt.  Prior releases used the option "md5".
#
# The "obscure" option replaces the old `OBSCURE_CHECKS_ENAB' option in
# login.defs.
#
# See the pam_unix manpage for other options.

# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
password        [success=1 default=ignore]      pam_unix.so obscure sha512
# here's the fallback if no module succeeds
password        requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password        required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
password        optional        pam_gnome_keyring.so
# end of pam-auth-update config



^G Get Help         ^O WriteOut         ^R Read File        ^Y Prev Page
^X Exit             ^J Justify          ^W Where Is         ^V Next Page
```

Common-password

There are two important lines we must edit, here are their configurations

```
password requisite pam_cracklib.so retry=3 minlen=8 difok=3 reject_username
minclass=3 maxrepeat=2 dcredit=-1 ucredit=-1 lcredit=-1 ocredit=-1 gecoscheck
enforce_for_root
```

- retry=3
  - allow three attempts at a good password before the passwd program shuts down
- minlen=8
  - Minimum password length of 8
- difok=3
  - Sets the amount of characters that must be different from the last password
- Reject_username
  - Reject username as password
- minclass=3
  - Minimum types of characters that must be used
- maxrepeat=2
  - Maximum number of repeating characters

Common-password

There are two important lines we must edit, here are their configurations

```
password requisite pam_cracklib.so retry=3 minlen=8 difok=3 reject_username
minclass=3 maxrepeat=2 dcredit=-1 ucredit=-1 lcredit=-1 ocredit=-1 gecoscheck
enforce_for_root
```

- gecoscheck:
  - Some fields are optional, like full name, number, address, etc. in adduser
  - Don’t allow these to show up in the password
- enforce_for_root
  - Root must obey password policies

Common-password

- lcredit
    - lowercase
- ucredit
    - uppercase
- dcredit
    - digits
- ocredit
    - Alphanumeric characters, !?>< etc.

This assigns value to types of characters, negative value means that they are required

Common-password

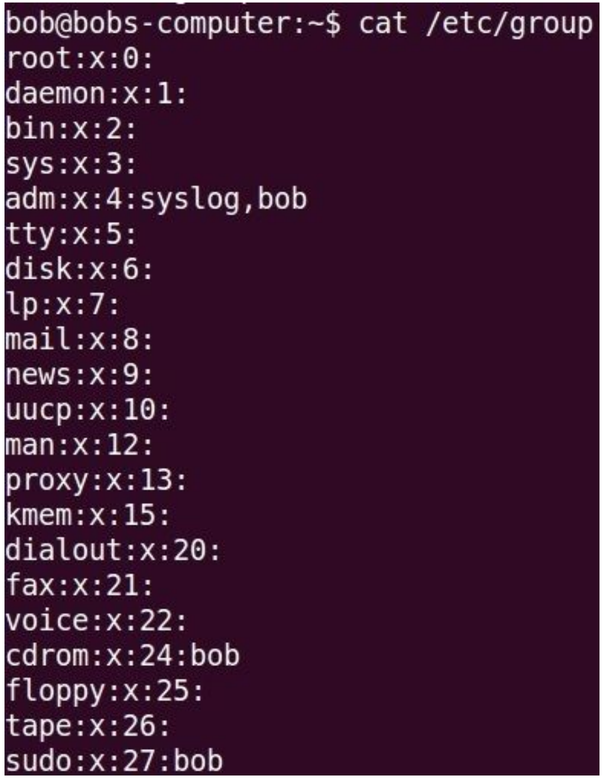
There are two important lines we must edit, here are their configurations

```
password sufficient pam_unix.so use_authtok obscure rounds=80000 sha512 shadow remember=7
```

- obscure
  - Extra password checks, you can google them if you want.
- sha12
  - Encrypt passwords with SHA12
- rounds
  - number of rounds of encryption
- remember=7
  - Remember the last 7 passwords to prevent alternating passwords too often

The group file

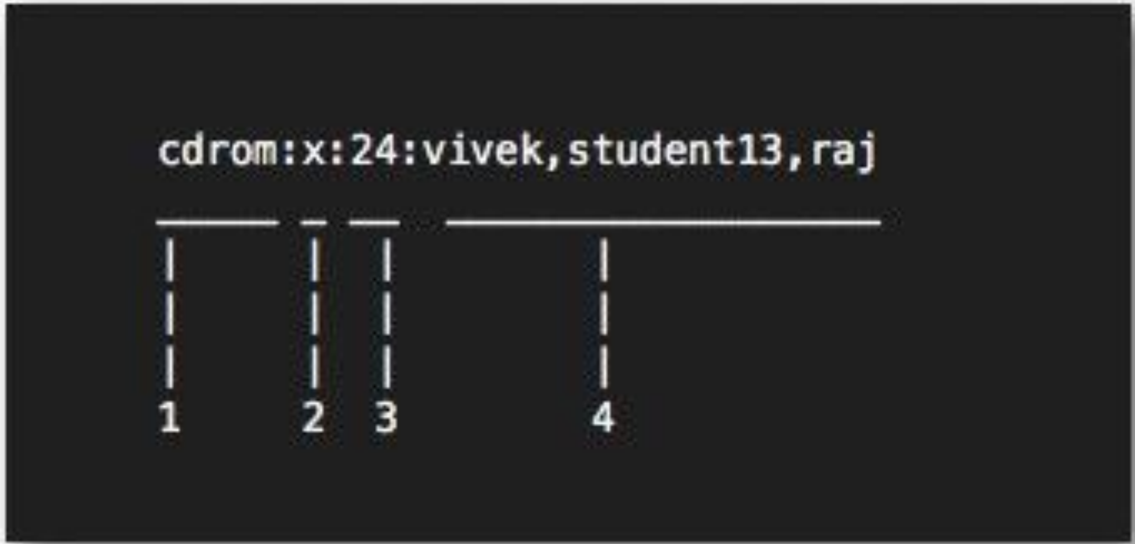
- Similar to the passwd file
  - Both in etc
- What does control? What important groups are there?



```
bob@bobs-computer:~$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,bob
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:bob
floppy:x:25:
tape:x:26:
sudo:x:27:bob
```

Group file in depth

1. Group name
2. Password
3. GID
4. Users in group

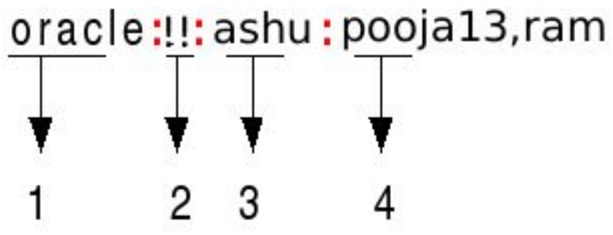


```
cdrom:x:24:vivek,student13,raj
_____ _ __  _________________
|     | |          |
|     | |          |
|     | |          |
1     2 3          4
```

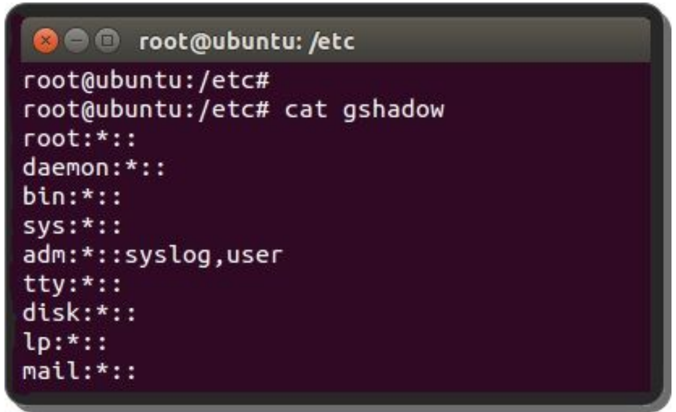
How to configure groups

- Groups have passwords and administrators
- `/etc/group` does not actually show these, to see group admins and **encrypted** group passwords you must look at `/etc/gshadow`

- `members [group]`
    - Returns all members of a group
    - might have to install members, but it's a nice cmd



1. Group name
2. Group password
    a. !! no pass
3. Group admin(s)
4. Regular users



```
root@ubuntu:/etc#
root@ubuntu:/etc# cat gshadow
root:*::
daemon:*::
bin:*::
sys:*::
adm:*::syslog,user
tty:*::
disk:*::
lp:*::
mail:*::
```

How to configure groups

- To edit group properties, we use the `gpasswd` command to edit admins, group passwords, and regular users

- `gpasswd [group]`
  - Prompts for new password when called by group administrator
- `gpasswd -a [user]`
  - Add a regular user to group
- `gpasswd -A [users,users2]`
  - Set administrator list
- `gpasswd -d [user]`
  - Remove a user from group

Sudoers file

- What is the importance of this file?
- How do we safely edit this file?
- What other directories does this file include per se?

What to look for in the sudoers file

- Always edit with the visudo command
  - Visudo checks the file with basic sanity checks, makes sure the file saves correctly otherwise you cannot use administrative privileges
- Any user with [username] ALL=(ALL:ALL) ALL
- Instances of NOPASSWD
- Instances of Defaults !authenticate
- /etc/sudoers.d/
  - Any file can have these lines INCLUDING THE README
  - Use ls -al to look for files with blank names

What is LightDM?

- LightDM is our Display Manager for Ubuntu v16.04 and below
  - For newer versions of Ubuntu we use GDM
- LightDM launches our X servers, greeter (Login screens), and user sessions
  - Default greeter is Unity Greeter
- We are concerned with the greeter

Configuring LightDM

- LightDM configurations are stored in three places
  - `/usr/share/lightdm/lightdm.conf.d/*.conf`
  - `/etc/lightdm/lightdm.conf.d/*.conf`
  - `/etc/lightdm/lightdm.conf`
- First location is restricted and only for the system, we don’t edit that one
- Second location is editable and we may choose to edit this file
- Third location is editable and preferable to edit.

Configuring LightDM

- LightDM combines the configurations at the end
- Let’s pick `/etc/lightdm/lightdm.conf`

Here are some important configs

- `allow-guest=false`
- `greeter-hide-users=true`
  - Hides the users list, the list of users shown at login. Explicitly declare manual login below
- `greeter-show-manual-login=true`
  - Shows the manual login

Package management

- What is our main package managers? What commands can we use to install/remove packages?
    - How are they different? What makes us choose one over the other?
- What is a repo?
- Where are repos stored?
- How do we edit repos?

Apt Basics

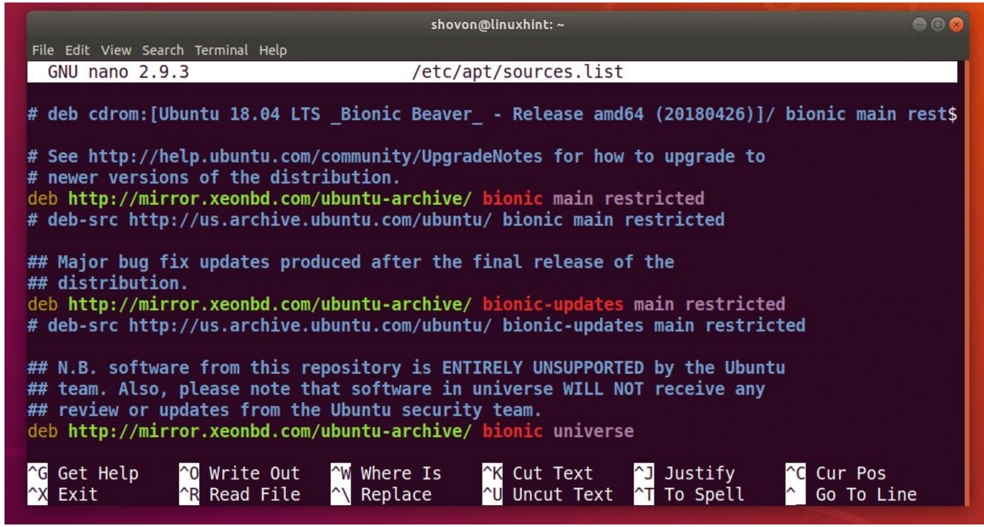
- Apt is short for Aptitude
- `apt` vs `apt-get`
  - Apt command is supposed to be “More pleasant for end user”
  - Recommended to use apt over apt-get.
- Do not confuse apt for dpkg
  - Apt handles packages from internet repos, while dpkg simply installs .deb files.
  - Apt downloads the deb file in a package, and installs it with dpkg.

Apt options

- `apt install [packagename]`
  - Install package
- `apt update`
  - Updates package index, defined by your repos in sources.list and sources.list.d
- `apt upgrade`
  - Install updates to packages from the APT package index. (we'll cover what this is soon.)
- `apt remove [packagename]`
  - Remove packages

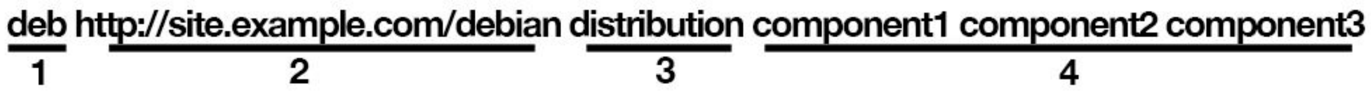
Repositories

- Ubuntu stores its repos in the `/etc/sources.list` and corresponding files in `/etc/sources.list.d/`
- Repos control your APT package index



```
  GNU nano 2.9.3                                  /etc/apt/sources.list

# deb cdrom:[Ubuntu 18.04 LTS _Bionic Beaver_ - Release amd64 (20180426)]/ bionic main rest$

# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://mirror.xeonbd.com/ubuntu-archive/ bionic main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ bionic main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://mirror.xeonbd.com/ubuntu-archive/ bionic-updates main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ bionic-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team. Also, please note that software in universe WILL NOT receive any
## review or updates from the Ubuntu security team.
deb http://mirror.xeonbd.com/ubuntu-archive/ bionic universe

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

Understanding the sources.list file



```
deb http://site.example.com/debian distribution component1 component2 component3
```

1. Type of repo
   a. deb is user packages
   b. deb-src is source code
2. Location of repo
3. Ubuntu version
   a. Ubuntu 16.04 is xenial
4. Section name, or component

Adding and removing repos

- `add-apt-repository [repo]`
    - A full line in quotes will append to the sources.list
- `add-apt-repository ppa:[developer]/[ppaname]`
    - This adds a Personal Package Archives to your package index
    - This is unsupported software, be wary of what you add.

Apt settings

- `dpkg-reconfigure -plow unattended-upgrades`
- `/etc/apt/apt.conf.d/20auto-upgrades:`
  - `APT::Periodic::Update-Package-Lists "1";`
  - `APT::Periodic::Download-Upgradeable-Packages "1";`
  - `APT::Periodic::AutocleanInterval "7";`
  - `APT::Periodic::Unattended-Upgrade "1";`
- This is the same thing as doing apt thru GUI settings

Services, Processes, and Logging

Hello there

How to start, stop, and restart services

- There are multiple commands that can control services, the two main commands are `service` and `systemctl`
- We are gonna go over `service`

- `service [service] start`
    - Starts a service
- `service [service] stop`
    - Stops a service
- `service [service] restart`
    - Restarts a service
- `Service --status-all`
    - Lists the status of all services

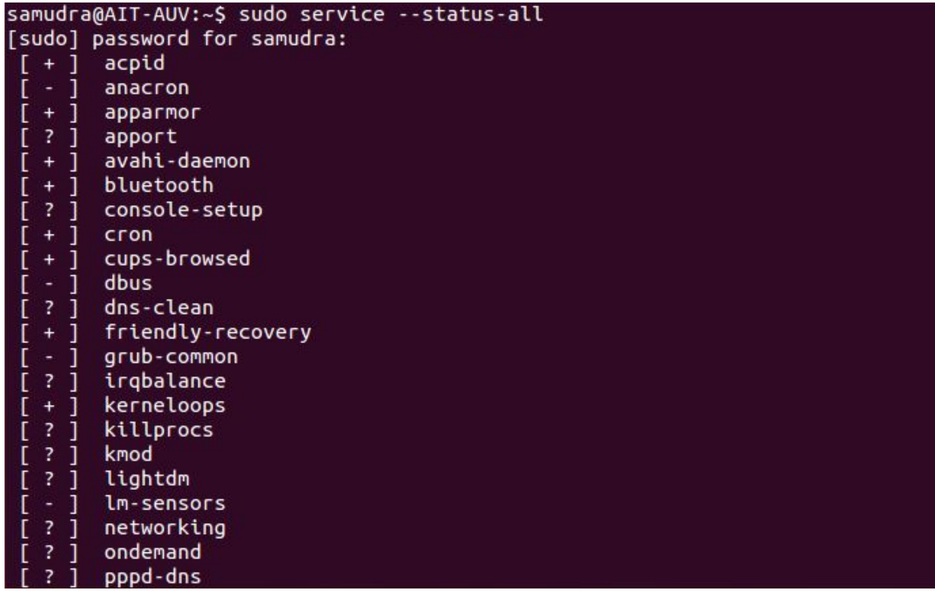
More about the service command

- The service command works by calling a script in the `/etc/init.d/` directory
- This directory holds many scripts that control services like apache and sshd

- `service apache2 stop` equals `/etc/init.d/apache2 stop`

Reading service status’

- The output from the command `service --status-all` can be confusing to read
- `[+]`
  - Means the service is running
- `[-]`
  - Means the service is not running
- `[?]`
  - Means the service is not responding to the query, the service isn’t sending anything back when asked if running.



```
samudra@AIT-AUV:~$ sudo service --status-all
[sudo] password for samudra:
 [ + ]  acpid
 [ - ]  anacron
 [ + ]  apparmor
 [ ? ]  apport
 [ + ]  avahi-daemon
 [ + ]  bluetooth
 [ ? ]  console-setup
 [ + ]  cron
 [ + ]  cups-browsed
 [ - ]  dbus
 [ ? ]  dns-clean
 [ + ]  friendly-recovery
 [ - ]  grub-common
 [ ? ]  irqbalance
 [ + ]  kerneloops
 [ ? ]  killprocs
 [ ? ]  kmod
 [ ? ]  lightdm
 [ - ]  lm-sensors
 [ ? ]  networking
 [ ? ]  ondemand
 [ ? ]  pppd-dns
```

How to view running processes

- There are a couple ways to view running processes
- Each one is useful for different purposes

- `ps aux`
  - Outputs a snapshot of running processes, good to pipe into `grep` or `less`
- `top`
  - Outputs live running processes, updating in real time. Good for managing cpu usage and memory management
- `htop`
  - Better version of `top`, not usually installed
- `pgrep [process]`
  - Returns process ID

How to kill processes

- Just like viewing processes, killing processes is done in a variety of ways

- `kill [PID]`
  - Kills process using process id, Useful when you want to kill a single very specific process
- `killall [process]`
  - Kills all processes that match name

Logging in Ubuntu

- System logging is an important part of auditing a Ubuntu system.
- System logs are stored in `/var/log/`
- These files have many lines, it is generally recommended to use `grep` to cut down on the sheer volume

Important System Logs

- `/var/log/auth.log`
  - Logs all authentication related information, anything with PAM
- `/var/log/daemon.log`
  - Logs all things to do with daemons, background processes, useful for troubleshooting certain daemons
- `/var/log/syslog`
  - Contains a great deal of information from various parts of the system, refer to this when you can't find what you want in other logs.

Application Logs

- Many applications log in the `/var/log/` directory as well

Example Application Logs

- `/var/log/apache2/access.log`
  - Logs all pages and files served by the server
- `/var/log/apache2/error.log`
  - Logs all errors reported by the server
- `/var/log/samba/log.[IP_ADDRESS]`
  - Logs various samba requests from that IP

Un-readable logs

- Some logs are not cat-able, and have special commands.

- `faillog`
  - Shows login failures log
- `lastlog`
  - Shows listing of logins, use with `less`

Firewall configuration

UFW in Ubuntu

- UFW stands for Uncomplicated Firewall, and is the main firewall tool in Ubuntu.
- UFW has a GUI but that's disgusting. WE USE THE COMMAND LINE.

- `ufw enable`
  - Enables firewall and enables firewall on boot
- `ufw disable`
  - Disable firewall and disable firewall on boot
- `ufw default allow|deny|reject [incoming|outgoing|routed]`
  - Sets the default for all incoming, outgoing, or routed packets (depending on option) to allow, deny, or reject (depending on the option)
- `ufw reject [args]`
  - Reject any packet following these arguments (We will go over args in next slide)
- `ufw allow [args]`
  - Allow any packet following these arguments (We will go over args in next slide)
- `ufw deny [args]`
  - Deny any packet following these arguments (We will go over args in next slide)
- `ufw logging [on|off]`
  - Enables logging, logs can be found at `/var/log/ufw.log`

Valid UFW Arguments

- When deny,rejecting or allowing packets, there are various different valid arguments.

Example arguments

- `24/tcp`
  - This is a port number and a protocol
- `sshd`
  - A service name
- `apache2/udp`
  - A service name and a protocol

Deny, reject, and allow

- Deny means do not allow that package to travel in the specified direction (incoming,outcoming). Default direction is incoming
- Reject means do not allow that package to travel in specified direction and inform the sender that the packet was rejected
- Allow means do allow that packet to travel.