Vanier College

SysAdmin & DevOps

Tassia Camoes Araujo
araujot@vanier.college

420-321-VA Unix
Winter 2024

Outline

System Administrator

- The person responsible for installing, configuring and maintaining the system in good health.

System Administrator

- The person responsible for installing, configuring and maintaining the system in good health.

"Your network is **secure**, *your computer is* **up and running**, *and your printer is* **jam-free**. *Why? Because you've got an awesome sysadmin (or maybe a whole IT department) keeping your business up and running."*

`https://www.redhat.com/sysadmin/sysadmin-devops`

Sysadmin tasks

- User administration: accounts, space, support

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management
- Application installation/update/compatibility/license

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management
- Application installation/update/compatibility/license
- Service configuration/maintenance

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management
- Application installation/update/compatibility/license
- Service configuration/maintenance
- Scheduling tasks/jobs

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management
- Application installation/update/compatibility/license
- Service configuration/maintenance
- Scheduling tasks/jobs
- Backup and disaster recovery

Sysadmin tasks

- User administration: accounts, space, support
- System maintenance and health monitoring
- Network and security management
- Application installation/update/compatibility/license
- Service configuration/maintenance
- Scheduling tasks/jobs
- Backup and disaster recovery
- User training

User accounts

- User and group management: *adduser* & *usermod*

User accounts

- User and group management: *adduser* & *usermod*
- The root user has full administrative power

User accounts

- User and group management: *adduser* & *usermod*
- The root user has full administrative power
- Avoid working directly as root, only when needed!

User accounts

- User and group management: *adduser* & *usermod*
- The root user has full administrative power
- Avoid working directly as root, only when needed!
- Execution of a command as another user with *sudo*

User accounts

- User and group management: *adduser* & *usermod*
- The root user has full administrative power
- Avoid working directly as root, only when needed!
- Execution of a command as another user with *sudo*
- The *sudo* group and */etc/sudoers* file

Special file permissions

A fourth access level in addition to user, group, and other

- SUID (s) = file is executed as the user who owns the file
  - eg. /usr/bin/passwd needs to write to /etc/shadow
- SGID (s) = file is executed as the group who owns the file
- Sticky bit (t) = restricts file deletion to its owner (eg. /tmp)

Special file permissions

A fourth access level in addition to user, group, and other

- SUID (s) = file is executed as the user who owns the file
  - eg. /usr/bin/passwd needs to write to /etc/shadow
- SGID (s) = file is executed as the group who owns the file
- Sticky bit (t) = restricts file deletion to its owner (eg. /tmp)

- Passed as a fourth preceeding digit to chmod
  - *$ chmod X### file* (X is the special permission)
  - SUID(4), GUID (2), Sticky (1)

Special file permissions

A fourth access level in addition to user, group, and other

- SUID (s) = file is executed as the user who owns the file
    - eg. /usr/bin/passwd needs to write to /etc/shadow
- SGID (s) = file is executed as the group who owns the file
- Sticky bit (t) = restricts file deletion to its owner (eg. /tmp)

- Passed as a fourth preceeding digit to chmod
    - *$ chmod X### file* (X is the special permission)
    - SUID(4), GUID (2), Sticky (1)

- Learn more:
  `https://www.redhat.com/sysadmin/suid-sgid-sticky-bit`

Application management

- The task of installing, removing and upgrading software

Application management

- The task of installing, removing and upgrading software
- GNU/Linux distributions provide *software packages* to facilitate application management

Application management

- The task of installing, removing and upgrading software
- GNU/Linux distributions provide *software packages* to facilitate application management
- Package development has different stages:

Application management

- The task of installing, removing and upgrading software
- GNU/Linux distributions provide *software packages* to facilitate application management
- Package development has different stages:
  - Just after the build of a new version, a package is considered **unstable**

Application management

- The task of installing, removing and upgrading software
- GNU/Linux distributions provide *software packages* to facilitate application management
- Package development has different stages:
  - Just after the build of a new version, a package is considered **unstable**
  - After some time, if that version survives, it goes under **testing**

Application management

- The task of installing, removing and upgrading software
- GNU/Linux distributions provide *software packages* to facilitate application management
- Package development has different stages:
  - Just after the build of a new version, a package is considered **unstable**
  - After some time, if that version survives, it goes under **testing**
  - At some point, the package is considered mature and becomes **stable**

Package sets - the Debian example

- Each project has a different workflow in regards to packaging and packages life cycle

Package sets - the Debian example

- Each project has a different workflow in regards to packaging and packages life cycle
- A *distribution* or *release* is the set of all packages that composes a version of the OS

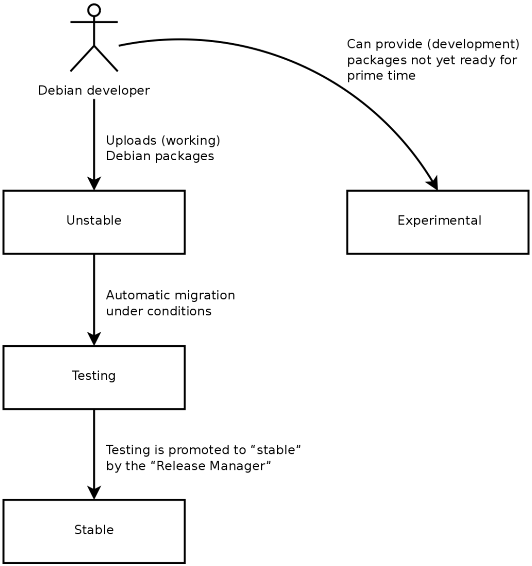
Package sets - the Debian example

- Each project has a different workflow in regards to packaging and packages life cycle
- A *distribution* or *release* is the set of all packages that composes a version of the OS
- A *stable* distribution is the set of packages that has reached the desired maturity to be published to the general public (usually, new versioning)

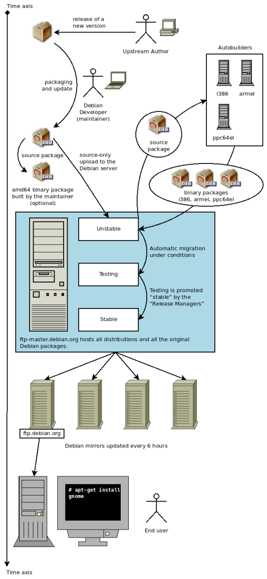
Package sets - the Debian example

- Each project has a different workflow in regards to packaging and packages life cycle
- A *distribution* or *release* is the set of all packages that composes a version of the OS
- A *stable* distribution is the set of packages that has reached the desired maturity to be published to the general public (usually, new versioning)
- Before getting there, there are intermediate stages that individual packages (and dependencies) pass through

A package's path through the various releases


Debian developer
Can provide (development) packages not yet ready for prime time
Uploads (working) Debian packages
Unstable
Experimental
Automatic migration under conditions
Testing
Testing is promoted to “stable” by the “Release Manager”
Stable

A package's life cycle



Time axis
release of a new version
Upstream Author
packaging and update
Debian Developer (maintainer)
Autobuilders
i386
armel
ppc64el
source package
source package
source-only upload to the Debian server
amd64 binary package built by the maintainer (optional)
binary packages i386, armel, ppc64el
Unstable
Testing
Stable
Automatic migration under conditions
Testing is promoted "stable" by the "Release Managers"
ftp-master.debian.org hosts all distributions and all the original Debian packages.
ftp.debian.org
Debian mirrors updated every 6 hours
# apt-get install gnome
End user
Time axis

Network configuration

- Desktop environments take care of it with *network manager*

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*
- The new *ip* command has replaced old *ifconfig* and *route* from net-tools

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*
- The new *ip* command has replaced old *ifconfig* and *route* from net-tools
  - `$ ip address show`

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*
- The new *ip* command has replaced old *ifconfig* and *route* from net-tools
  - `$ ip address show`
  - `$ ip route`

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*
- The new *ip* command has replaced old *ifconfig* and *route* from net-tools
  - `$ ip address show`
  - `$ ip route`
- Name of the machine is set in */etc/hostname*

Network configuration

- Desktop environments take care of it with *network manager*
- For servers, network settings are saved in */etc/network/interfaces*
- The new *ip* command has replaced old *ifconfig* and *route* from net-tools
  - `$ ip address show`
  - `$ ip route`
- Name of the machine is set in */etc/hostname*
- Domain and other hosts configuration in */etc/hosts*

Init Systems

Initialization daemon

- The mother of all processes (PID 1) - try *pstree*
- Drives the system/services initialization at boot time
- Two major init systems in Linux:
  - System V: relies on init scripts and runlevels (rc0, rc1, ..., rc6 - in order, not in parallel)
  - System D: based on unit files and targets
    - Services can be started in parallel
    - Quicker boot-up time
    - Better handling of hot-plug devices (added to a running computer, eg. USB drive)
    - Standard service configuration and management across Linux distributions (*systemctl* command)

Initialization with System D

- Services are started/stopped based on target units
- Target units "want" or "require" other targets or services
- At boot, systemd activates the default.target unit (the alias in `/lib/systemd/system/default.target`)
- `$ systemctl get-default`
  - Servers will default to *multi-user.target*
  - Desktop systems will default to *graphical.target*

Service management practice

- First, check all services offered by your system

```
# systemctl list-units --type=service
```

  - If sshd is not available, install the package *openssh-server*

- Check the status/log of a particular service

```
# systemctl status ssh.service
# journalctl -u ssh.service
```

- Stop, start, restart

```
# systemctl stop ssh.service
# systemctl start ssh.service
# systemctl restart ssh.service
```

- Enable/disable persistent services

```
# systemctl enable ssh.service
# systemctl disable ssh.service
```

Add a new service

1. Create a bash script to be run as a service (eg. /usr/bin/mydaemon.sh)
2. Add execution permission to the script file

```
#!/bin/bash
while true
do
    echo "HelloWorld @ $(date)" >> /tmp/bash.log
    sleep 3
done
```

Sample service unit configuration

1. Create a service unit configuration file
2. Place the file in /etc/systemd/system/mydaemon.service

```
[Unit]
Description=My bash daemon

[Service]
ExecStart=/usr/bin/mydaemon.sh

[Install]
WantedBy=graphical.target
```

Start/Enable the service

- Start the service immediately:

```
$ systemctl start mydaemon.service
```

- Enable at boot using systemctl:

```
$ systemctl enable mydaemon.service
```

- **Or** enable by creating yourself a symbolic link on the desired target “wants” directory

```
$ cd /etc/systemd/system/graphical.target.wants/
$ ln -s /etc/systemd/system/mydaemon.service
```

Unit configuration: Sections & directives

[Unit]

- Description
- After: this comes after which other unit
- Requires: units it depends on
- Wants: activated in parallel, but independent

[Service]

- ExecStart: the command to start the service
- ExecReload: the command to reload it

[Install]

- WantedBy: the target unit to which this service belongs to

https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files

Scheduling jobs with crond

- The "classic" method
- System-wide + users crontab
- Edit your scheduler with *crontab -e*
- List your jobs with *crontab -l*

```
# .------------ minute (0-59)
# | .------------ hour (0-23)
# | | .------------ day of the month (1-31)
# | | | .------------ month (1-12)
# | | | | .------------ day of the week (0-6) (Sun to Sat)
# | | | | |
# | | | | |
# | | | | |
# * * * * * <user-name> <command to be executed>
```

Scheduling tasks as Systemd timers

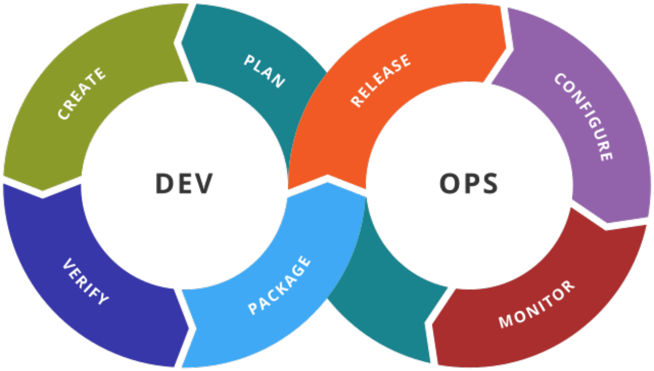
- More flexibility for time specification to trigger job:
  - eg. specific amount of time after startup, completion of a task, or completion of a service unit
- The job is defined as a regular service unit
- The scheduler is a timer unit, which triggers the job at the specified time
- List your existing timers: *$ systemctl status *timer*

Scheduling tasks as Systemd timers

- More flexibility for time specification to trigger job:
  - eg. specific amount of time after startup, completion of a task, or completion of a service unit
- The job is defined as a regular service unit
- The scheduler is a timer unit, which triggers the job at the specified time
- List your existing timers: *$ systemctl status *timer*
- Learn more: `https://opensource.com/article/20/7/systemd-timers`

Sysadmin *vs.* DevOps

Devops toolchain



DevOps

"DevOps represents a change in IT culture, focusing on rapid IT service delivery through the adoption of **agile**, *lean practices in the context of a system-oriented approach. DevOps emphasizes people (and culture) and seeks to improve* **collaboration between operations and development teams**. *DevOps implementations utilize technology — especially automation tools that can leverage an increasingly programmable and dynamic infrastructure from a life cycle perspective."*

`https://www.redhat.com/sysadmin/sysadmin-devops`

DevOps

- Development & Operations united
- Automation: building, deploying, and monitoring
- CI/CD = Continuous Integration / Continuous Delivery
- Containers - app isolation from its environment (eg. *Docker*)
- Orchestration - lifecycle management (eg. *Kubernetes*)
- Configuration management - automation (eg. *Ansible*)

`https://roadmap.sh/devops`

Thanks!