Lab Exercise – systemd 1

Plan: in the following exercises we will explore important systemd commands, that help to setup a debugginging environment for systemd

prerequisites:

IPADDRESS of your VM named "trouble" is 192.168.2.160

as root generate a public/private RSA key pair on your desktop

```
ssh-keygen

ssh-copy-id 192.168.2.160
```

now you are able to login passwordless to the VM "trouble" via

ssh -X root@192.168.2.160

ssh -X root@trouble

add an entry into /etc/hosts if hostname "trouble" is not listed

- 0.9 there are over 130 man pages.

How do you quickly determine the right manpage for an unknown option, parameter, directive?

- 1.0 is the journal persistent?

Per default logs are saved in /run/log/journal/, run is ephemeral

we have already booted the VM once, execute:

```
journalctl --list-boots
```

It will only display the most recent boot, so all journal information from previous boots is lost per default!

Directory /var/log/journal/ needs to exist, so that systemd-journald-service can save its data there.

Directory /etc/systemd lists some essential configuration files with its defualt values.

Open /etc/systemd/journald.conf and change the line that starts with Storage=

```
vi /etc/systemd/journald.conf

[...]

[Journal]

Storage=persistent

#Compress=yes

[…]
```

restart the journal service and display the last boots:

```
systemctl restart systemd-journald.service
```

verify, that directory /var/log/journal has been created.

Reboot and verify:

```
journalctl--list-boots
```

From now on all boots will be listed

During boot startup no messages are displayed.

To change this adapt the grub2 configuration:

edit /etc/default/grub

```
GRUB_CMDLINE_LINUX_DEFAULT="resume=/dev/vda1 quiet showopts"
```

and execute:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

and reboot.

Never edit `/boot/grub2/grub.cfg directly!`

• 1.1 In your KVM VM within virt-manager execute Ctrl+Alt+F10

Where does this output come from?

Which configuration file is uses to enable it?

• 1.2 check if lamp server is installed as pattern

```
systemctl status apache2
```

execute:

```
systemctl
```

apache2 is not listed

• 1.3 Now execute:

```
systemctl list-unit-files|grep apache2
```

where does systemd define that a service is disabled although it is installed(as in this case apache2)?

• 1.4 What files are changed when you execute the following commands?

```
systemctl enable apache2
```

Take good note, which files were created by the command

```
systemctl disable apache2

systemctl mask apache2

systemctl start apache2
```

this will not work, because service is masked

```
systemctl unmask apache2

systemctl enable apache2

systemctl start apache2
```

rc scripts for Backward compatibility

• 2.0 execute:

```
rccron status
```

for Backward compatibility it is still possible to execute rc scripts via a symlink

• 2.1 what does the command actually execute ?

Hint: softlink

```
ls -l /usr/sbin/rc*
```

• 2.3 are these real init scripts?

• what do they do?

hint: use shell debugging

• include a "set -x" statement in /usr/sbin/service after the initial comments:

```
cp -p /usr/sbin/service /usr/sbin/service.orig

sed -i '4 a set -x' /usr/sbin/service
```

• execute the command again:

```
rccron status
```

• examine the output

to display the interesting lines do:

```
rccron status  2>&1|grep systemctl

# systemctl --full --no-legend --no-pager --type=service
--property=LoadState show cron.service
```

• 2.5 look up the options in the systemctl man page, some of the options above allow for automating sytemctl commands and assigning them to variables.

• 2.5.1 Init scripts use DefaultDependencies=No

What is the main difference this is causing?

Hint: Read systemd.service(5) and search for DefaultDependencies

• 2.6 what other types are available for systemctl command?

Hint: use bash completion

A:

```
systemctl -t <tab>
```

• 2.7 Display the contents of the cron service:

• 2.8 what does it depend on? (Look for Wants and Requires lines) Is there any ordering imposed on those dependencies? (After and Before lines - if there is an After line, the current unit should be started after the indicated unit.)

• 2.9 what are the units that depend on it (Look for WantedBy and RequiredBy)

• 2.10 when will the cron service be restarted?

• 2.11 what other options are available for restart of a service?

• 2.12 which manpage describes this?

• 2.13 kill the cron command

hint: ps aux|grep cron

• 2.14 what happens to the service?

• 2.16 was the service restarted? Which commands help you to see log/error messages

• 2.17 stop the unit using systemctl stop cron<tab> .

• 2.18 What is its status now?

• 2.19 Disable the unit. What does the output of systemctl show?

• 2.20 List the directory that the indicated symlink used to be in. Re-examine the unit file and see which directive created this symlink. This symlink was created in response to a WantedBy directive.

hint: use systemctl list-unit-files

• 2.21 Re-enable the unit. List the directory that the symlink was contained in.

Have a look at the timestamp of symlink

• 2.22 Now list the units that depend on this one using systemctl --reverse list-dependencies

• 2.23 Finally, list the units that this unit depends on using the previous command without the --reverse option. Is there a surprise?

• 2.24 is the cron unit started yet?

• 2.25 Start the unit.

• 3.0 Look at the symlink named default.target. What is it symlinked to?

• 3.1 Check the contents of /etc/systemd/system for unit files and dependency directories. Verify the default target

using systemctl get-default.

• 3.2 name another(shorter) way of doing:

less /usr/lib/systemd/system/graphical.target

• 3.3 Change your default target to multi-user mode using

• 3.4 Did either of the default.target symlinks change? You just changed the

system 'initdefault' state effectively from 5 to 3. Did you disable the GUI console?

• 3.5 Issue the command:

```
systemd-cgls | tee /tmp/graphical-target.txt
```

• We will now go immediately to multi-user.target. This will not be apparent unless you can see

the GUI console.

• 3.6 Issue the command

```
systemctl isolate multi-user.target
```

If you can see the GUI console, verify that the console has changed to a text prompt. Otherwise, look in the output of systemd-cgls and verify that the gdm processes have exited

execute the following commands:

```
systemd-cgls| tee /tmp/multi-target.txt
```

• 3.7 Execute:

```
vimdiff <(cat /tmp/graphical-target.txt) <(cat /tmp/multi-
target.txt)
```

• 3.8 which services have also been used in the graphical target but not in multi-user.target ?

hint: use

```
journalctl -xb
```

• to inquire journalctl options look up its man page

- 3.9 what do above services have in common?
- 3.10 look up the Type of service of accounts-daemon

Lab exercise 3 – systemd listing units

Plan: In this exercise we will see which units are installed and enabled on your system.

In the previous exercise, you listed the unit files with the extension .service in the main systemd directory /usr/lib/systemd/system.

• Recreate that list.

• Ask systemd for a list of all the unit files using the option --type=service and the command list-unit-files. If you see any discrepancies, you should know that there is a second directory of systemd information /etc/systemd/system. This second directory is reserved for implementation-specific data.

• execute systemd-delta

• list-unit-files does not tell you what units are enabled. For that, use the list-units command (again, with --type=service). This indicates which units are enabled on your system. In the listing you will find some service units that are not enabled (disabled). Can you find any that you recognize?

• Repeat the above experiments changing the type of the units to target.

The isolate command for systemctl can be used on any target that can 'stand on its own', or, in other words, can trigger a valid change in system state. List all the target units in /usr/lib/systemd/system that allow the use of the isolate command (they have the keyword/value pair AllowIsolate=yes in their unit file.

A:

```
            grep -l "AllowIsolate=yes" /usr/lib/systemd/system/*.target
```

• Unit generators:

all units based on rules in /usr/lib/udev/rules.d/ contain rules with

```
            TAG+=systemd entries
```

execute:

```
grep -rc 'TAG+="systemd"' /usr/lib/udev/rules.d/|grep -v ":0"
```

Lab exercise 3 – systemd listing units

have a look at 99-systemd.rulesudev

• which kernel modules are loaded at boot time?

Hint: modules-load.d(5)

• which services are using configuration options from /etc/sysconfig ?

List them!

Hint:

```
alias A1=' awk '\''{print $1}'\'''
```

```
systemctl -t service  --no-legend|A1|while read x; do systemctl cat $x|
grep "/etc/sysconfig" && echo $x; done
```

• Will a service be started nevertheless when those services do not contain a configuration in

/etc/sysconfig ?

Hint:

yes

EnvironmentFile=-

man systemd.exec

The argument passed should be an absolute filename or wildcard expression, optionally prefixed with "-", which indicates that if the file does not exist, it will not be read and no error or warning message is logged.

Note your findings:

Lab exercise 3 – systemd listing units