COURSEWARE

Systemd Service Configuration

Contents

Overview

This module discusses configuring Systemd Unit files, *Service* type files in particular.
These files will be referred to as services and units throughout the module.

Configuration

Here is a basic configuration file:

```
[Unit]
Description=Foo

[Service]
ExecStart=/usr/sbin/foo-daemon

[Install]
WantedBy=multi-user.target
```

Here are the sections of the example above broken down:

- `[Unit]`: This section is for generic information about the Unit such as its description.
- `[Service]`: This section is for defining things like what to actually run for the service and which user to run it as.
- `[Install]`: Not used during runtime, this section will only be used when installing the service.
  The `systemctl enable` and `systemctl disable` commands will utilise this section.
  A common configuration here is the `WantedBy=muti-user.target`, so this means the service will only start once the system has reached that system state on boot up.

  > `multi-user.target` normally defines a system state where all network services are started up and the system will accept logins, but a local GUI is not started.

ExecStart

`ExecStart` can be set as a part of the `[Service]` section.
What you set this to is basically just what the service is going to run, this unit for instance will just append the current date the end of a file every second:

```
[Unit]
Description=Save Date to /tmp/dates.txt

[Service]
ExecStart=/bin/sh -c 'while true; do sleep 1; date >> /tmp/date.txt; done'

[Install]
WantedBy=multi-user.target
```

This doesn't have to be a shell command that your are running here, it could be any application such as a web server for instance.

User

In many cases when deploying an application as a service to a system, you will want to have a separate user for running the service.
This user would have more restricted privileges to make the system less vulnerable should that service be compromised.

Just like `ExecStart`, this option can set as a part of the `[Service]` section:

```
[Unit]
Description=Save Date to /tmp/dates.txt

[Service]
User=service-user
ExecStart=/bin/sh -c 'while true; do sleep 1; echo "$(date): Running as:
$(whoami)" >> /tmp/date.txt; done'

[Install]
WantedBy=multi-user.target
```

> Notice that this Unit would also include the user's name in the `/tmp/date.txt` file.

WorkingDirectory

If your application needs to access resources from a relative path then you may want to consider using the `WorkingDirectory` configuration.
This will set the current working directory of the process upon running it.
You would usually set this to where your application exists and is operating.

Environment

Quite often you will need to parameterise properties using environment variables, to allow your application to operate on more than one environment.
Environment variables can be set in Unit files for your application by using the `Environment` configuration in the `[Service]` section.

More than one environment variable can be set this way; this example is setting which file to save the outputs to and also the 'environment' which is included as a part of the message:

```
[Unit]
Description=Save Date to /tmp/dates.txt

[Service]
User=service-user
Environment=OUTPUT_FILE=/tmp/dates.txt
Environment=ENVIRONMENT='development'
ExecStart=/bin/sh -c 'while true; do sleep 1; echo "$(date): Running on
${ENVIRONMENT} as: $(whoami)" >> ${OUTPUT_FILE}; done'

[Install]
WantedBy=multi-user.target
```

Installation

Installing a Unit is very simple, you just need to put the `*.service` file in a certain directory and then reload the systemd unit files.
There are several places you can put the unit file but a common place is in `/etc/systemd/system/`.

So to install a unit called `my-application.service` you would copy it to `/etc/systemd/system/my-application.service` and then run this command to reload the systemd units:

```
sudo systemctl daemon-reload
```

Tutorial

This tutorial is going to show you how to setup and install a very simple systemd unit.

Create the Unit File

Create new file in `/tmp/tutorial.service` and enter the following:

```
[Unit]
Description=Tutorial Service

[Service]
User=tutorial-user
Environment=OUTPUT_FILE=/tmp/output.txt
Environment=ENVIRONMENT='development'
ExecStart=/bin/sh -c 'while true; do sleep 1; echo "$(date): Running on
${ENVIRONMENT} as: $(whoami)" >> ${OUTPUT_FILE}; done'

[Install]
WantedBy=multi-user.target
```

Setup the Service User

Notice that the service configured in the previous step is being run by a user called `tutorial-user`.
For this service to work that user will need to exist so create them using this command:

```
sudo useradd --system tutorial-user
```

Install the Unit

With the user now added and the configuration prepared, it can now be installed by copying the file into `/etc/systemd/system` and then *reloading* the systemd unit files:

```
sudo cp /tmp/tutorial.service /etc/systemd/system/tutorial.service
sudo systemctl daemon-reload
```

Start the Service

Once the service has been installed it can then be started:

```
sudo systemctl start tutorial
```

Check your new Service

You can run this command to see the current status of your service:

```
sudo systemctl status tutorial
```

An output like this should be displayed:

```
● tutorial.service - Tutorial Service
   Loaded: loaded (/etc/systemd/system/tutorial.service; disabled; vendor
preset: enabled)
   Active: active (running) since Sun 2020-03-01 20:13:25 UTC; 2s ago
 Main PID: 820 (sh)
    Tasks: 2 (limit: 4915)
   CGroup: /system.slice/tutorial.service
           ├─820 /bin/sh -c while true; do sleep 1; echo "$(date): Running on
'development' as: $(whoami)" >> /tmp/output.txt; done
           └─827 sleep 1k
```

The actual command in the `ExecStart` is running a shell command which outputs information to `/tmp/output.txt`, you can watch this file get updated by using the `tail` command:

```
tail -f /tmp/output.txt
```

You will see an output with the date, the 'environment' being used and the current user:

```
Sun Mar  1 20:13:31 UTC 2020: Running on 'development' as: tutorial-user
Sun Mar  1 20:13:32 UTC 2020: Running on 'development' as: tutorial-user
Sun Mar  1 20:13:33 UTC 2020: Running on 'development' as: tutorial-user
Sun Mar  1 20:13:34 UTC 2020: Running on 'development' as: tutorial-user
Sun Mar  1 20:13:35 UTC 2020: Running on 'development' as: tutorial-user
```

Clean Up

To clean up run the following commands:

```
# stop the service
sudo systemctl stop tutorial
# remove the unit
sudo rm -rf /etc/systemd/system/tutorial.service
# reload the systemd units
sudo systemctl daemon-reload
# remove the service user
sudo userdel tutorial-user
```

Exercises

There are no exercises for this module.