Exercise 2.6: Create and customize a systemd service

This exercise is going to explore the various configuration directories for a **systemd** service. The application **stress** will be used, install the package **stress** with the appropriate package installer.

The contents of the **stress** package does not include a **systemd** unit configuration so one must be created. The package installed on the test system has the binary for **stress** as `/usr/bin/stress`. Create a **systemd** vendor unit file as `/usr/lib/systemd/system/foo.service`

```
$ cat /usr/lib/systemd/system/foo.service

[Unit]
Description=Example service unit to run stress
[Service]
ExecStart=/usr/bin/stress --cpu 4 --io 4 --vm 2 --vm-bytes 1G
[Install]
WantedBy=multi-user.target
```

Once the unit file is created **systemd** will be able to start and stop the service. Use the **top** command to verify that **stress** is working. The following commands may be useful:

```
# systemctl start foo
# systemctl status foo -l
# systemd-delta
# systemctl stop foo
```

Since the `/usr/lib/systemd/system/foo.service` may be altered by the vendor at update time, create a unit file in `/etc/systemd/system/foo.service` for the **stress** service and change the parameters slightly so it is easy to identify which unit file is being used. It is normal to copy the vendor unit file into the `/etc/systemd/system/` directory and make appropriate customizations.

```
$ cat /etc/systemd/system/foo.service

[Unit]
Description=Example service unit to run stress
[Service]
ExecStart=/usr/bin/stress --cpu 2 --io 2 --vm 4 --vm-bytes 1G
[Install]
WantedBy=multi-user.target
```

Start or restart the service and examine the differences in the following command output.

```
# systemctl status foo -l
# systemd-delta
```

Often times it is desirable to add or change features by program or script control, the drop-in files are convenient for this. One item of caution, if one is changing a previously defined function (like ExecStart) it must be undefined first then added back in. Create a drop-in directory and file for out **stress** service and verify the changes are active. Here is our example file:

```
$ cat /etc/systemd/system/foo.service.d/00-foo.conf

[Service]
ExecStart=
ExecStart=/usr/bin/stress --cpu 1 --vm 1 --io 1 --vm-bytes 1G
```


Start or restart the service and examine the differences in the following command output.

```
# systemctl status foo -l
# systemd-delta
```

With **systemd** additional features and capabilities can be easily added. As an example **cgroups** controls can be added to out service. Here is an example of adding a **systemd slice** to the example service and adding a resource limit to that slice. The slice is then attached to the service drop-in file. First setup a `<service>.slice` unit file:

```
$ cat /etc/systemd/system/foo.slice

[Unit]
Description=stress slice
[Slice]
CPUQuota=30%
```

Then connect our service to the **slice**. Add the following to the bottom of the unit file in `/etc/systemd/system/foo.service.d/00-foo.conf`

```
Slice=foo.slice
```

Restart the services and examine the differences with:

```
# systemctl daemon-reload
# systemctl status foo -l
# systemd-delta
# top
```

Bonus step: In our example there are no unique values in the `/etc/systemd/system/foo.service` file so in this example it is redundant. We can get rid of the extra file.

```
# mv /etc/systemd/system/foo.service /root/
# systemctl daemon-reload
# systemctl restart foo
# systemctl status foo
```

Consult the man pages **systemd.resource-control(5), systemd.service(5)** and other systemd man pages for additional information.

