stackoverflow.com/questions/13069634/python-daemon-and-systemd-service

I have simple Python script with works as a daemon. I am trying to create systemd script to be able to start this script during startup.

Current systemd script:

```
[Unit]
Description=Text
After=syslog.target

[Service]
Type=forking
User=node
Group=node
WorkingDirectory=/home/node/Node/
PIDFile=/var/run/zebra.pid
ExecStart=/home/node/Node/node.py

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

node.py:

```
if __name__ == '__main__':
    with daemon.DaemonContext():
        check = Node()
        check.run()
```

`run` contains `while True` loop.

I try to run this service by `systemctl start zebra-node.service`. Unfortunately service never finished stating sequence - I have to press Ctrl+C. Script is running, but status is activating and after a while it change to deactivating. Now I am using python-daemon (but before I tried without it and the symptoms were similar).

Should I implement some additional features to my script or is systemd file incorrect?

5 Answers

The reason, it does not complete the startup sequence is, that for Type `forking` your startup process is expected to fork and exit (see $ man systemd.service - search for forking).

Simply use only the main process, do not daemonize

One option is to do less. With systemd, there is often no need to create daemons and you may directly run the code without daemonizing.

```
#!/usr/bin/python -p
from somewhere import Node
check = Node()
check.run()
```

This allows using simpler Type of service called `simple`, so your unit file would look like.

```
[Unit]
Description=Simplified simple zebra service
After=syslog.target

[Service]
Type=simple
User=node
Group=node
WorkingDirectory=/home/node/Node/
ExecStart=/home/node/Node/node.py
StandardOutput=syslog
StandardError=syslog

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

Note, that the `-p` in python shebang is not necessary, but in case you print something out to the stdout or stderr, the `-p` makes sure, there is no output buffering in place and printed lines will be immediately caught by systemd and recorded in journal. Without it, it would appear with some delay.

For this purpose I added into unit file the lines `StandardOutput=syslog` and `StandardError=syslog`. If you do not care about printed output in your journal, do not care about these lines (they do not have to be present).

`systemd` makes many daemons obsolete

While the title of your question explicitly asks about daemonizing, I guess, the core of the question is "how to make my service running" and while **using main process seems much simpler** (you do not have to care about daemons at all), it could be considered answer to your question.

I think, that many people use daemonizing just because "everybody does it". With systemd the reasons for daemonizing are often obsolete. There might be some reasons to use daemonization, but it will be rare case now.