

Debugging failed systemd services

Publication Date: 27 Sep 2024

Contents

- 1 Environment 2
- 2 Introduction 2
- 3 Requirements 2
- 4 Getting more information on failed services 2
- 5 Manually debugging failed systemd services 4
- 6 Next steps 5

1 Environment

This document applies to the following products and product versions:

- SUSE Linux Enterprise Server 15 SP5, 15 SP4, 15 SP3, 15 SP2, 12 SP5
- SUSE Linux Enterprise Server for SAP Applications 15 SP5, 15 SP4, 15 SP3, 15 SP2, 12 SP5
- SUSE Linux Enterprise High Availability 15 SP5, 15 SP4, 15 SP3, 15 SP2, 12 SP5
- SUSE Linux Enterprise High Performance Computing 15 SP5, 15 SP4, 15 SP3, 15 SP2
- SUSE Linux Enterprise Desktop 15 SP5
- SUSE Linux Enterprise Real Time 15 SP5

2 Introduction

If you are experiencing problems with a failed service, `systemd` gives you commands to help further investigate the issue. Following the instructions below, you can learn how to debug your failed `systemd` services.

3 Requirements

- Basic understanding of `systemd`.
- Root privileges.

4 Getting more information on failed services

When `systemd` fails to start a service, you usually get this very generic error message (shown with Docker as an example):

```
# systemctl start docker
Job for docker.service failed because the control process exited with error code.
See "systemctl status docker.service" and "journalctl -xe" for details.
```

To investigate how to solve such failed service, proceed as follows:

1. Check the status first of the failed service:

```
# systemctl status --full --lines=50 docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: failed (Result: exit-code) since Fri 2021-08-20 10:24:15 CEST; 2min 24s ago
     Docs: http://docs.docker.com
   Process: 2491 ExecStart=/usr/bin/dockerd --add-runtime oci=/usr/sbin/docker-runc $DOCKER_NETWORK_OPTIONS $DOCKER_OPTS >
   Main PID: 2491 (code=exited, status=1/FAILURE)
```

In the above example, the service ran as process ID 2491, but failed. Often you will already see a hint in the error message output. In such a case, follow this hint, fix the issue, and you are done.

However, this is not always possible like in the above example. If the the above output does not reveal anything useful, proceed further.

2. Check the log of the failed Docker service:

```
# journalctl --catalog --pager-end --unit=docker
[...]
Aug 20 10:24:15 localhost.localdomain dockerd[2479]: unable to configure the Docker daemon with file /etc/docker/daemon.json:
cannot unmarshal string into Go value of type map[string]interface {}
```

With the option `--unit` you limit the log entries only to the failed Docker service.

As hinted by the error message, it is a good idea to look into the file `/etc/docker/daemon.json`. In this case, the error was caused by a syntax error. If you fix it and restart Docker you will not get any further errors and can stop with this procedure.

3. If this procedure does not help, proceed further with [Section 5, “Manually debugging failed systemd services”](#).

5 Manually debugging failed systemd services

If the procedure in [Section 4, "Getting more information on failed services"](#) did not work, debug the failed service manually. Generally, use this command:

```
sudo runuser -l USER --group=GROUP -c "cd WD && ENVFILE_CONTENTS ENV COMMAND"
```

The placeholders have the following meaning:

USER and GROUP

The user and group that the systemd service is using. For example, the mariadb service uses the user and group mysql. If the systemd unit file does not contain a user nor a group, remove the -l and -g options.

WD

The working directory. Only needed if it's set in the systemd unit file.

ENVFILE_CONTENTS

Contains a list of key-value pairs without quotes around. Find the key-value pairs in the file /etc/sysconfig/SERVICE_NAME. systemd does not need quotes around values. However, if you really need them, escape the quotes with \.

ENV

Basically the same as ENVFILE_CONTENTS

COMMAND

The command to run. It's usually the ExecStart key in the systemd unit file.

To demonstrate how to use it, investigate it with the Docker service. Use the following procedure:

1. Show the environment file /etc/sysconfig/docker. In most cases you will find this line:

```
DOCKER_OPTS=""
```

If you have other options, memorize them.

2. Show the content of the systemd Docker unit:

```
# systemctl cat docker
```

3. Extract the ExecStart from [Step 2](#).

4. Replace the placeholder and type the following command (replace `DOCKER_OPTS` with the values from *Step 1*):

```
> sudo runuser -c "/usr/bin/dockerd --add-runtime oci=/usr/sbin/docker-runc  
$DOCKER_OPTS"
```

5. Investigate the log and act accordingly. If you want to stop the manual call, press `Ctrl - C`.

6 Next steps

- Learn more about systemd