## Linux command "systemctl status" is not working inside a Docker container

The command systemctl status is not working. It never has. My container is on CentOS 7. When I issue systemctl status I get results Failed to get D-Bus connection: operation not permitted.

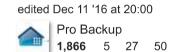
## Regression

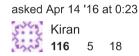
- 1. I then looked into upgrading systemd. I removed the /etc/yum/protected.d/system.conf file. I then used yum remove systemd. I see that systemd version 219-19.el7\_2.4 has been installed. I choose N to not actually remove systemd. I then installed systemd-libs-219-19.el7\_2.7.x86\_64.rpm. I then installed systemd version 2.7. I then used yum remove systemd just to determine the version. I see that systemd version 219-19.el7\_2.7 is installed. I choose NO to abort the removal. systemctl status still does not work. I get the same error: Failed to get D-Bus connection: operation not permitted.
- 2. I tried creating a new Docker container with the -privileged flag. When I used the -p 80:80 option, the Docker run command failed.
- 3. When I left out the -p 80:80 option in my docker run command, the new container had the same problem.
- 4. I created a Docker container with a docker run ... -v /sys/fs/cgroup:/sys/fs/cgroup:ro " option. I had the same problem.

I expect systemctl status to work inside a Docker container.

What should I do to get systemctl status to work in a Docker container?

/ centos / systemd / docker





## 2 Answers

I've managed to fix this issue in a CentOS:7 Docker container. I've followed mainly the Guide on CentOS Docker image project.

```
FROM centos:7

ENV container docker
RUN (cd /lib/systemd/system/sysinit.target.wants/; for i in *; do [ $i == \
    systemd-tmpfiles-setup.service ] || rm -f $i; done); \
    rm -f /lib/systemd/system/multi-user.target.wants/*;\
    rm -f /etc/systemd/system/*.wants/*;\
    rm -f /lib/systemd/system/local-fs.target.wants/*; \
    rm -f /lib/systemd/system/sockets.target.wants/*udev*; \
    rm -f /lib/systemd/system/sockets.target.wants/*initctl*; \
    rm -f /lib/systemd/system/basic.target.wants/*;\
    rm -f /lib/systemd/system/anaconda.target.wants/*;

# Install anything. The service you want to start must be a SystemD service.

CMD ["/usr/sbin/init"]
```

Now, build the image, and run it using at least the following arguments to docker run command: -v /run -v /sys/fs/cgroup:/sys/fs/cgroup:ro

Then main point is that /usr/sbin/init must be the first process inside the Docker container.

So if you want to use a custom script that executes some commands before running /usr/sbin/init, launch it at the end of your script using exec /usr/sbin/init (in a bash script).

Here is an example:

```
ADD cmd.sh /usr/local/bin/
RUN chmod +x /usr/local/bin/cmd.sh

CMD ["/usr/local/bin/cmd.sh"]

And here is the content of cmd.sh:

#!/bin/bash

# Do some stuffs

exec /usr/sbin/init # To correctly start D-Bus thanks to https://forums.docker.com/t/any-simple-and-safe-way-to-start-services-on-centos7-systemd/5695/8
```

You could have System is booting up. See pam\_nologin(8) if your using the PAM system, in that case, delete /usr/lib/tmpfiles.d/systemd-nologin.conf in your Dockerfile because it creates the file /var/run/nologin Which generates this specific error.

answered Mar 1 '17 at 14:25

Anthony O.

245 2 6

Run the docker as following, It's works for me,

```
docker run -it --privileged -p 10081:80 centos7 /bin/bash
```