

Eigene Software

Singularity Container

VERSION • 3.7.0

Sofern Software benötigt wird die nicht auf dem Cluster installiert ist und sich nicht im Home-Verzeichnis installieren lässt, weil beispielsweise Root-Rechte benötigt werden, so kann ein Singularity-Container verwendet werden. Hierzu ist es vorerst notwendig einen Container zu bauen. Eine Anleitung findet sich innerhalb der [offiziellen Dokumentation](#). Da das Bauen von Containern ausschließlich mit `root`-Rechten möglich ist, muss dies außerhalb des Clusters geschehen.

Alternativ ist es möglich den Container mittels eines [Remote Builders in der Sylabs Cloud](#) zu bauen. Hierfür wird ein entsprechender Account benötigt.

Container außerhalb des Clusters erstellen

Nachstehend sind einige grundlegende Schritte für das Erstellen eines Containers.

Ubuntu 20.04 Image herunterladen und einen Container daraus erstellen

```
singularity pull ubuntu.sif library://ubuntu:20.04
```

Container lokal erkunden

```
# Start the container, and enter the shell
singularity shell ./ubuntu.sif

# Now check the operating system of the container, you would see Ubuntu
head /etc/os-release

# your user id inside container
whoami

# exit the container
exit
```

Convert the ubuntu.sif file into a Singularity sandbox

```
singularity build --fakeroot --sandbox myubuntu ./ubuntu.sif
```

Start the sandbox as container with a writable shell and install nano

```
singularity shell --fakeroot --writable myubuntu
apt update
apt upgrade
apt install nano
exit
```

Convert the sandbox into a Singularity image file

```
singularity build --fakeroot myubuntu.sif myubuntu/
```

Create an image from a def file

```
Bootstrap: docker
From: ubuntu:18.04

%post
    apt-get update -y
    apt-get -y install libpmem1 librpmem1 libpmemblk1 libpmemlog1 libpmemobj1 libpmempool1
    apt-get -y install libpmem-dev librpmem-dev libpmemblk-dev libpmemlog-dev libpmemobj-dev libpmempool-dev
    apt-get -y install build-essential libmemkind-dev libvmem-dev nano git
```

Container im Clusters nutzen

Nachdem der Container gebaut und auf das Cluster übertragen wurde ist er auf allen Knoten verfügbar und kann mit den folgenden Befehlen gestartet werden.

Interaktives arbeiten in einem Singularity Container

```
student@node80:~/containers$ singularity shell my_custom_container.sif
Singularity>
```

Im Container wird das Home-Directoy gemountet, sodass man auf alle Dateien darin normal zugreifen kann.

Der standardmäßig gesetzte Prompt `singularity>` kann durch setzen der `SINGULARITYENV_PS1` Umgebungsvariable angepasst werden. Das folgende Beispiel ergänzt zu Anfang den Namen des Containers. Die Umgebungsvariable kann auch in `~/bashrc` gesetzt werden.

Anpassen des Prompts

```
student@node80:~/containers$ export SINGULARITYENV_PS1='(${SINGULARITY_NAME/.sif/}) \[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
student@node80:~/containers$ singularity shell my_custom_container.sif
(my_custom_container) student@node80:~/containers$
```