UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

ziti

Tools 2025 – Apptainer Introduction

Shigehiko Schamoni

12.06.2025

Overview

- Introduction
- Installing Apptainer
- Using Apptainer
- Exercises

- Introduction
- Installing Apptainer
- Using Apptainer
- Exercises

Introduction to Apptainer

- Apptainer is a *container* platform.
- Apptainer is focused on:
  - Effective security model: same user inside a container as outside
  - Integration over isolation: cluster/server system accessible by default
  - SIF container format is easy to share
- Secure alternative to Docker
- Designed for high-performance computing
- Compatible with Docker and OCI images
- Documentation:
  `https://apptainer.org/docs/user/latest/`

Apptainer Features

- Provides all benefits of a container platform, e.g.
  - Bundles application together with all required software.
  - Based on single image files that are transferable.
  - Images are independent on library version of the host system
- Does not need a daemon:
  - Easier to handle
  - No security concerns: container runs within the user context
- Bring Your Own Software
  - Install what you want inside the image and then run it.
  - User outside the container is the same user inside.
    $\Rightarrow$ Not possible to escalate privileges this way!

- Introduction
- Installing Apptainer
- Using Apptainer
- Exercises

Installing Apptainer

- Installed via the distribution's repository or via GitHub `https://apptainer.org/docs/admin/main/installation.html`
- After installation, Apptainer can be started as a normal user
- Command line interface to interact with containers
- From within a container:
  - Redirect I/O, use pipes
  - Pass arguments and shell variables
  - Access files
  - etc.
- Information about interface and individual subcommands

```
:~$ apptainer help [subcommand]
```

Apptainer Bind Mounts

By default, Apptainer activates the following bind mounts:

- User's home directory (`$HOME`)
- Current working directory (`pwd`)
- System mounts:
  - `/dev`
  - `/etc/hosts`
  - `/etc/localtime`
  - `/proc`
  - `/sys`
  - `/tmp`
  - `/var/tmp`
- System configuration files:
  - `/etc/resolv.conf`
  - `/etc/groups`
  - `/etc/passwd`

- Introduction
- Installing Apptainer
- Using Apptainer
- Exercises

Running and Converting Docker Images

- Apptainer works with Docker images:
  1. Convert "on-the-fly"
  2. Permanently convert to Apptainer's SIF format
- Convert "on-the-fly"

```
:~$ apptainer run docker://hello-world
```

  Very similar to Docker:

```
:~$ docker run hello-world
```

- Permanently convert an image, e.g. save to `hello-world.sif`

```
:~$ apptainer build hello-world.sif docker://
hello-world
```

Disabling and Enabling Bind Mounts

- The option `--no-mount` allows (specific) mounts to be disabled.
- To disable multiple mounts, separate them by commas:

```
:~$ apptainer run --no-mount tmp,sys,dev
mycontainer.sif
```

- The option `--mount` allows mounting of host directories using a Docker compatible syntax.
- Apptainer provides an easier way of mounting directories via the `--bind` option.
- To bind a directory `/data` on the host to `/mnt` in the container:

```
:~$ apptainer exec --bind /data:/mnt
mycontainer.sif
```

  The directory `/mnt` does not need to exist in the container.

Building Containers from Definition Files

- Apptainer can work with *definition files* that describe how to build a container image. Example:

```
Bootstrap: docker
From: ubuntu:16.04

%post
apt-get -y update
apt-get -y install cowsay lolcat

%environment
export LC_ALL=C
export PATH=/usr/games:$PATH

%runscript
date | cowsay | lolcat
```

- Very similar to Docker definition files
- Creating an image requires root-privileges:

```
:~$ sudo apptainer build myimage.sif myimage.
def
```

Bring Your Own Software (I)

- Apptainer containers are read-only by default.
- Filesystem overlay mechanism makes immutable images writeable.
- Create an overlay image with 100MB free space

```
:~$ apptainer overlay create --size 100 /tmp/
overlay.img
```

- Possible to install software permanently without changing the original image
- Create an overlay image with `fakeroot` option

```
:~$ apptainer overlay create --size 100 --
fakeroot /tmp/overlay.img
```

`fakeroot` enables the user to act within the container as root.

Bring Your Own Software (II)

- Using the overlay image, we can apply modifications to the container

```
:~$ apptainer shell --overlay /tmp/overlay.img
myimage.sif
```

- To install software, root access is required:

```
:~$ apptainer shell --fakeroot --overlay /tmp/
overlay.img myimage.sif
```

- We can now install software as usual, e.g.:

```
:~$ apt update
```

```
:~$ apt install cowsay lolcat
```

```
:~$ date | /usr/games/cowsay | /usr/games/
lolcat
```

- Introduction
- Installing Apptainer
- Using Apptainer
- Exercises

Hello World!

Let's run the "Hello World" Docker example in Apptainer.

- Login to the pool-PC in front of you.
- Check if Apptainer is installed, e.g. via `apptainer version`.
- Run the `hello-world` Docker image using Apptainer.
- Convert the `ubuntu` Docker container image to Apptainer's SIF format and save it to `/tmp/ubuntu.sif`.

```
:~$ apptainer build ...
```

(hint: `man apptainer`)

- Now execute a `ls` command within the container:

```
:~$ apptainer exec /tmp/ubuntu.sif /bin/ls ~
```

What might be the output in Docker?

- Try out a few other Apptainer subcommands, e.g. `shell`, `inspect`, etc.

Accessing Host Resources

- Execute an `ls` command within the container:

```
:~$ apptainer exec /tmp/ubuntu.sif /bin/ls ~
```

  What might be the output in Docker?
- Now run the Ubuntu image with an interactive shell (subcommand `shell`).
- Write something to your home directory, e.g.

```
:~$ touch ~/write_test
```

- Exit the container and verify the created file is still present.
- Is this behavior useful?

Modifying a Container

- Run the `ubuntu` container in an interactive session.
- Try to install the binary `sl` as described in the Docker exercise.
- Why is this not possible?

```
:~$ apptainer exec /tmp/ubuntu.sif /bin/ls ~
```

  What might be the output in Docker?
- Create a temporary overlay image of size 100 MB:

```
:~$ apptainer overlay create --fakeroot --size
100 /tmp/overlay.img
```

  Note: `fakeroot is needed to mimic root access inside the container.`
- Now run the container together with the overlay file:

```
:~$ apptainer shell --fakeroot --overlay /tmp/
overlay.img /tmp/ubuntu.sif
```

- Install the binary `sl` again and run it.