Charlie Cloud Notes

By Craig Walker May 10, 2023

Some things about CharlieCloud

- to build charliecloud you need python3 and, specifically, the module “requests” and its dependencies
- “ch-image build” will build the image from a Dockerfile….docker is not needed for this
  - USER is not needed in this file and is ignored. I don’t recommend setting up users at all
  - You will need to make a home/USER directory, however, as it makes things more simple
    - for such things as .bashrc etc…
    - most programs are expecting some notion of a USER
  - You will need to add --force to your build line. The default workaround mode “seccomp” is fine.
    - basically, this is needed for installing applications (apt-get, yum, zypper)
    - More info can be read at https://hpc.github.io/charliecloud/ch-image.html#privilege-model
- “ch-convert” will convert your image to a filesystem
  - you can convert to a tar.gz but you can’t ch-run from it
  - you will need to “ch-run” from :
    - either an inbuilt ch-image,
    - a filesystem ← I choose this one as it is easily editable
      - a directory with your typical linux filesystem (/bin /etc /usr /home /tmp /var …)
    - a squashfs file
      - though this is supposed to be common, squashfs was not available on ac-cluster
      - is basically a tar but can be mounted
  - ch-convert <in-built image name> <folder not made yet>
    - so the in-built image name is the name you gave the image with “ch-image build”
      - name is set with -t flag
    - folder not made yet ← make sure that the leaf to this path has not been created yet
      - for instance: /home/cwalker/deployment/my_fs
        - here my_fs has not been created yet, but will be by the ch-convert command

Some things about CharlieCloud

- "ch-run" will run your image
  - typically I run it as '`ch-run <path to charliecloud filesystem> -- /bin/bash -c "<commands>"` '
  - you will want to --bind parts of the normal filesystem to your charliecloud filesystem
    - I usually use charliecloud's /mnt to bind, though you can use any folder in the charliecloud filesystem that doesn't already have files in it
    - multiple --bind flags are allowed
    - syntax: --bind <abs path normal fs>:<charliecloud filesystem path>
      - example: `ch-run ./batsim_ch --bind /home/cwalker/deployment:/mnt -- /bin/bash -c ' echo "hello" '`
  - you will want to put commands with a lot of quotation marks in a file and run it, as this one-liner gets confusing since the command to -c needs to be quoted also
  - you will want to set the HOME environment variable
    - ex: --set-env=HOME=/home/sim
      - multiple --set-env= flags are allowed
    - I also set the TERM environment variable
      - on ac-cluster it was normally set to 'xterm-256color'
      - gets rid of warnings, not sure if it is absolutely needed
  - you will want to use the --write flag so you can write to your filesystem during a run
    - With batsim, this is needed for some temporary files
    - As far as my tests go, this is fine and does not require any kind of locking of files
      - But always be aware that when multiple invocations of ch-run are running your filesystem, they are spinning up the SAME filesystem. This is different than how docker works.
  - you will want to source your .bashrc before anything, as you can see below (dark gray)
  - example: `ch-run ../batsim_ch --bind ${curDir%/basefiles}:/mnt/ --write --set-env=HOME=/home/sim -- /bin/bash -c "source /home/sim/.bashrc; cd /mnt/basefiles;source /home/sim/python_env/bin/activate; python3 generate_config_docker.py -i /mnt/basefiles/tests/charliecloud/test1_consv_bf_resv.config -o /mnt/experiments/test1_slurm --output-config"`
  - I remember in the past that running slurm commands inside of the charliecloud filesystem did not work correctly.
    - sbatch a script that runs ch-run inside of it, instead

Installing with apt-get after making an image

- if you want to install stuff after the fact(at least in an ubuntu based image)
- first off you need to run ch-run with:
  - --write
    - obviously you need to be able to write to its filesystem to install stuff
  - -u 0 -g 0
    - you need root privileges from root user and root group (zero is the uid and gid of root)
  - ex:
    - `ch-run ./batsim_ch -u 0 -g 0 --write --set-env=HOME=/home/sim -- /bin/bash -c "source /home/sim/.bashrc; bash"`
    - gives you an interactive bash shell
- Once you have an interactive shell do the following

```
cat <<EOF > /etc/apt/apt.conf.d/sandbox-disable
APT::Sandbox::User "root";
EOF
sed -i '/ssl-cert/d' /var/lib/dpkg/statoverride
```

Should be good to go after that. run apt-get update && apt-get install <pkg>