

Rstudio on the Curnagl cluster

Rstudio can be run on the curnagl cluster from within a singularity container, with an interactive interface provided on the web browser of any given workstation.

Running interactively with Rstudio on the clusters is only meant for testing. Development must be carried out on the users workstations, and production runs must be accomplished [from within R scripts/codes in batch mode](#).

The command Rstudio is now available in r-light module. You have to do a reservation first with Sinteractive, ask the right amount of resources and then launch the command 'Rstudio'.

Procedure

```
Sinteractive  # specify here the right amount of resources
module load r-light
Rstudio
```

The procedure below is now deprecated !!

Preparatory steps

1. If the workstation is outside of the campus, first [connect to the VPN](#)
2. [Login to the cluster](#)
3. Create/choose a folder under the **/scratch** or the **/work** filesystems under your project (ex. **/work/FAC/.../rstudio**); this folder will appear as your HOME inside the Rstudio environment, and we will refer to it as **\${WORK}**
4. *(This step is **optional** and only applies if you need a R version not available in the r-light module)* Create the singularity image inside the cluster (substitute **\${WORK}** appropriately):

```
[me@curnagl ~]$ module load singularityce
[me@curnagl ~]$ singularity pull --dir="${WORK}" --name=rstudio-server.sif
docker://rocker/rstudio
```

This last step might take a while...

The batch script

Create a file **rstudio-server.sbatch** with the following contents (it must be on the cluster, but the exact location does not matter):

```
#!/bin/bash -l

#SBATCH --account ACCOUNT_NAME
#SBATCH --mail-type BEGIN
#SBATCH --mail-user <first.lastname>@unil.ch

#SBATCH --chdir ${WORK}
#SBATCH --job-name rstudio-server
#SBATCH --signal=USR2
#SBATCH --output=rstudio-server.job.%j
#SBATCH --partition interactive
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1
#SBATCH --mem 8G
#SBATCH --time 01:59:59
#SBATCH --export NONE

set -e

RVERSION=4.4.1 #See module spider r-light to get all available versions
LOCAL_PORT=8787
RSTUDIO_CWD=$(pwd)
RSTUDIO_SIF="/dcsrsoft/singularity/containers/r-light.sif"

module load python singularityce

# Create temp directory for ephemeral content to bind-mount in the container
RSTUDIO_TMP=$(mktemp --tmpdir -d rstudio.XXX)

mkdir -p -m 700 \
    ${RSTUDIO_TMP}/run \
    ${RSTUDIO_TMP}/tmp \
    ${RSTUDIO_TMP}/var/lib/rstudio-server
```

```

mkdir -p ${RSTUDIO_CWD}/.R

cat > ${RSTUDIO_TMP}/database.conf <<END
provider=sqlite
directory=/var/lib/rstudio-server
END

# Set OMP_NUM_THREADS to prevent OpenBLAS (and any other OpenMP-enhanced
# libraries used by R) from spawning more threads than the number of processors
# allocated to the job.
#
# Set R_LIBS_USER to a path specific to rocker/rstudio to avoid conflicts with
# personal libraries from any R installation in the host environment

cat > ${RSTUDIO_TMP}/rsession.sh <<END
#!/bin/sh

export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK:-1}
export R_LIBS_USER=${RSTUDIO_CWD}/.R
export PATH=${PATH}:/usr/lib/rstudio-server/bin
exec rsession "\${@}"
END

chmod +x ${RSTUDIO_TMP}/rsession.sh

SINGULARITY_BIND+="${RSTUDIO_CWD}:${RSTUDIO_CWD},"
SINGULARITY_BIND+="${RSTUDIO_TMP}/run:/run,"
SINGULARITY_BIND+="${RSTUDIO_TMP}/tmp:/tmp,"
SINGULARITY_BIND+="${RSTUDIO_TMP}/database.conf:/etc/rstudio/database.conf,"
SINGULARITY_BIND+="${RSTUDIO_TMP}/rsession.sh:/etc/rstudio/rsession.sh,"
SINGULARITY_BIND+="${RSTUDIO_TMP}/var/lib/rstudio-server:/var/lib/rstudio-server,"
SINGULARITY_BIND+="/users:/users,/scratch:/scratch,/work:/work"
export SINGULARITY_BIND

# Do not suspend idle sessions.
# Alternative to setting session-timeout-minutes=0 in /etc/rstudio/rsession.conf
export SINGULARITYENV_RSTUDIO_SESSION_TIMEOUT=0

export SINGULARITYENV_USER=$(id -un)
export SINGULARITYENV_PASSWORD=$(openssl rand -base64 15)

```

```
# get unused socket per https://unix.stackexchange.com/a/132524
# tiny race condition between the python & singularity commands
readonly PORT=$(python -c 'import socket; s=socket.socket(); s.bind("", 0));
print(s.getsockname()[1]); s.close()')
cat 1>&2 <<END
```

1. SSH tunnel from your workstation using the following command:

```
ssh -n -N -J ${SINGULARITYENV_USER}@curnagl.dcsr.unil.ch -L ${LOCAL_PORT}:localhost:${PORT}
${SINGULARITYENV_USER}@${HOSTNAME}
```

and point your web browser to `http://localhost:${LOCAL_PORT}`

2. log in to RStudio Server using the following credentials:

```
user: ${SINGULARITYENV_USER}
password: ${SINGULARITYENV_PASSWORD}
```

When done using RStudio Server, terminate the job by:

1. Exit the RStudio Session ("power" button in the top right corner of the RStudio window)
2. Issue the following command on the login node:

```
scancel -f ${SLURM_JOB_ID}
END
```

```
singularity exec --home ${RSTUDIO_CWD} --cleanenv ${RSTUDIO_SIF} \
  /usr/lib/rstudio-server/bin/rsrver --www-port ${PORT} \
    --auth-none=0 \
    --auth-pam-helper-path=pam-helper \
    --auth-stay-signed-in-days=30 \
    --auth-timeout-minutes=0 \
    --auth-encrypt-password=0 \
    --rsession-path=/etc/rstudio/rsession.sh \
    --server-user=${SINGULARITYENV_USER} \
    --rsession-which-r /opt/R- $\{RVERSION\}$ /bin/R
```

```
SINGULARITY_EXIT_CODE=$?
echo "rsrver exited $SINGULARITY_EXIT_CODE" 1>&2
exit $SINGULARITY_EXIT_CODE
```

You need to carefully replace, at the beginning of the file, the following elements:

- On line 3: **ACCOUNT_NAME** with the project id that was attributed to your PI for the given project
- On line 5: **<first.lastname>@unil.ch** with your e-mail address
- On line 7: **\${WORK}** must be replaced with the **absolute path** (ex. */work/FAC/.../rstudio*) to the chosen folder you created on the preparatory steps
- On line 21: you can modify the R version. All available versions can be obtained from the following command `module spider r-light`
- On line 24: if (and only if) you went through the optional fourth preparatory step, then you need to redefine **RSTUDIO_SIF** so that the line reads
RSTUDIO_SIF=\${RSTUDIO_CWD}/rstudio-server.sif

Running Rstudio

Submit a job for running Rstudio from within the cluster with:

```
[me@curnagl ~]$ sbatch rstudio-server.sbatch
```

You will receive a notification by e-mail as soon as the job is running.

A new file `${WORK}/rstudio-server.job.###` (with `###` some given job id number) is then automatically created. Its contents will give you instructions on how to proceed in order to start a new Rstudio remote session from your workstation.

You will have 2h time to test your code.

Révision #21

Créé 22 février 2023 16:24:25 par Flavio Calvo

Mis à jour 11 septembre 2024 16:58:30 par Emmanuel Jeanvoine