Rstudio on the Urblauna cluster

Rstudio can be run on the Urblauna cluster from within a singularity container, with an interactive interface provided on the web browser of a Guacamole session.

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.

Preparatory steps on Curnagl side

A few operations have to be executed on the Curnagl cluster:

1. Create a directory in your /work project dedicated to be used as an R library, for instance: `mkdir /work/FAC/FBM/DBC/mypi/project/R_ROOT`
2. Optional : install required R packages, for instance `ggplot2`
   `module load gcc r`
   `export R_LIBS_USER=/work/FAC/FBM/DBC/mypi/project/R_ROOT`
   `R`
   `>>>install.packages("ggplot2")`

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 --job-name rstudio-server
#SBATCH --signal=USR2
#SBATCH --output=rstudio-server.job
#SBATCH --nodes 1
#SBATCH --ntasks 1
```

```
#SBATCH --cpus-per-task 1
#SBATCH --mem 8G
#SBATCH --time 02:00:00
#SBATCH --partition interactive
#SBATCH --export NONE

RLIBS_USER_DIR=<<<RLIBS_PATH>>>
RSTUDIO_CWD=~
RSTUDIO_SIF="/dcsrsoft/singularity/containers/rstudio-4.2.1.sif"

module load gcc python singularity
module load r
RLIBS_DIR=${R_ROOT}/rlib/R/library
module unload r



# 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=${SLURM_CPUS_PER_TASK:-1}
export R_LIBS=${RLIBS_DIR}
export R_LIBS_USER=${RLIBS_USER_DIR}
export PATH=${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,/dcsrsoft"
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=$(python -c 'import socket; s=socket.socket(); s.bind(("", 0)); print(s.getsockname()[1]);
s.close()')
cat 1>&2 <<END
1. open the Guacamole web browser to http://${HOSTNAME}:${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 --env R_LIBS=${RLIBS_DIR} --home ${RSTUDIO_CWD} --cleanenv ${RSTUDIO_SIF} \
singularity exec --home ${RSTUDIO_CWD} --cleanenv ${RSTUDIO_SIF} \
   rserver --www-port ${PORT} \
           --auth-none=0 \
           --auth-pam-helper-path=pam-helper \
           --auth-stay-signed-in-days=30 \
           --auth-timeout-minutes=0 \
           --rsession-path=/etc/rstudio/rsession.sh \
           --server-user=${SINGULARITYENV_USER}

SINGULARITY_EXIT_CODE=$?
echo "rserver 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 14: ***<<<RLIBS_PATH>>>*** must be replaced with the **absolute path** (ex. */work/FAC/.../R_ROOT*) to the chosen folder you created on the preparatory steps

Running Rstudio

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

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

Once the job is running (you can check that with Squeue), a new file rstudio-server.job is then automatically created. Its contents will give you instructions on how to proceed in order to start a new Rstudio remote session from Guacamole.

In this script we have reserved 2 hours

Révision #3
Créé 23 mai 2023 16:01:14 par Emmanuel Jeanvoine
Mis à jour 24 mai 2023 06:08:06 par Emmanuel Jeanvoine