

## How to Run RStudio & Jupyter Notebook on ARC

Please consider the following guidelines if you plan to use RStudio and Jupyter Notebook on Virginia Tech (VT) ARC ([Advanced Research Computing](#)) for data analysis. The purpose of this document is to provide guidelines for the use of RStudio and Jupyter Notebook on ARC at the Center for Biostatistics and Health Data Science (CBHDS). This document provides a step-by-step guide to demonstrate the various ways to run RStudio and Jupyter Notebook on ARC. Note that you do not need to have any prior experience with high-performance computing (HPC) to follow this document.

### Utilizing ARC Resources for Large Data Sets and Jobs

CBHDS uses ARC for the purpose of storage and data analysis for all bioinformatics projects. ARC is Advanced Research Computing, a unit of VT's Division of Information Technology. ARC provides centralized research computing infrastructure and support for the VT research community. For more information on **ARC services** and **how to request an account, create projects, and manage data** on ARC, please read our "[Bioinformatics Data Transfer](#)" document, which is available on the CBHDS [website](#). Please email Dr. Missi Zhang (email: [missizxm@vt.edu](mailto:missizxm@vt.edu)) if you have any questions about these guidance documents.

VT ARC provides HPC systems, large-scale data storage, big data processing, and software. You can use the [CPU and GPU clusters](#) provided by ARC if you have large data sets and/or large jobs to run. At CBHDS, we often need to run RStudio and Jupyter Notebook on ARC to analyze large data files, especially for bioinformatics projects.

There are two ways of running RStudio and Jupyter Notebook on ARC:

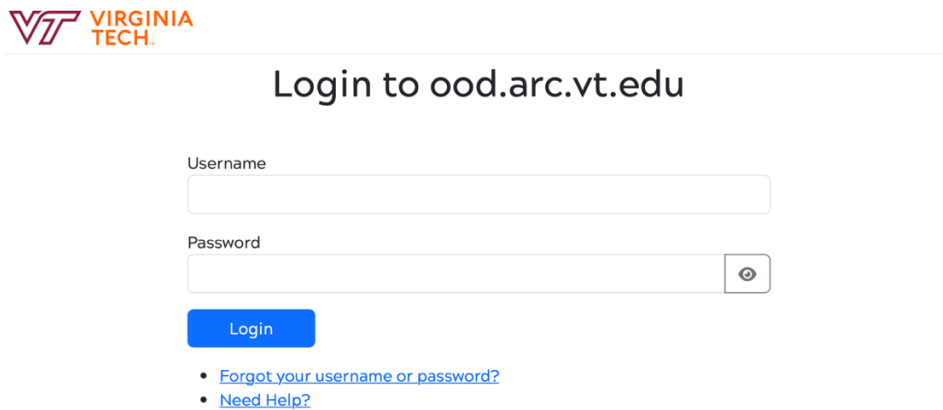
- [Through the interactive apps](#) on "[Open OnDemand](#)": This is the easiest of the two ways. All that you would need to do is to open the links provided in the "Interactive Apps" section and specify the number of nodes and cores you need to use. However, the downside of using this method is that users could only select the R and RStudio versions available on the drop-down list, and users would not be able to install certain packages. If these issues matter to you, you can try the following alternative method listed in the bullet below.
- [Through a remote server on a compute node](#): This method requires requesting computational resources (using sbatch or run salloc/srun commands) through Slurm (the scheduler and cluster resource manager) and run RStudio and Jupyter Notebook on the web browser from a remote server. You can find more information about Slurm and job submission on ARC's [user documentation page](#). Please be aware that although technically this can be done on a login node, data analysis jobs must **NOT** be run on the login nodes and these jobs are subject to **administrative termination**. For more examples about the acceptable and unacceptable use of login nodes, please visit <https://www.docs.arc.vt.edu/usage/abuse.html>.

Below are step-by-step instructions detailing how to run R studio and Jupyter Notebook on ARC for each of the two methods listed above (via Open OnDemand versus remote server).

## 1. “Open OnDemand” Interactive Apps: RStudio and Jupyter Notebook

This section contains instructions for accessing RStudio and Jupyter Notebook through the [“Open OnDemand” interactive apps webpage](#). Please refer to *Section 1* of our [ARC Data Transfer and Management guidance document](#) for information on **requesting VT PID and ARC accounts**. Please make sure you are connected to VT’s campus network before you start Step 1.1. If you are connected to off-campus internet, you can use VT’s [Remote Access VPN](#) to access Blacksburg campus university services.

- 1.1. Go to <https://ood.arc.vt.edu/pun/sys/dashboard/>, and use your VT PID and password to log in to the ARC system.



VT VIRGINIA TECH.

### Login to ood.arc.vt.edu

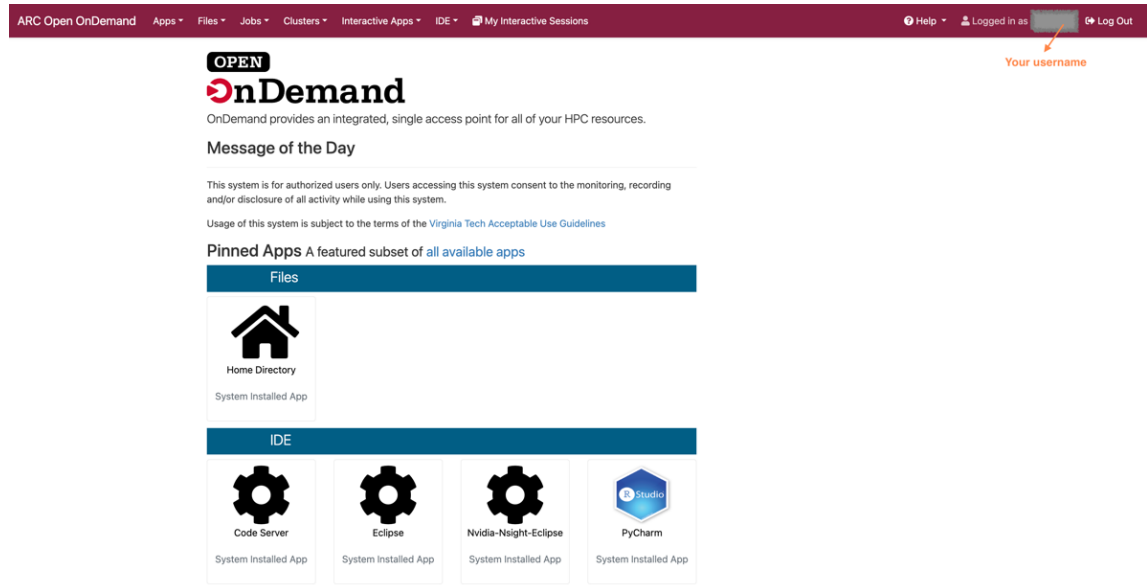
Username

Password

- [Forgot your username or password?](#)
- [Need Help?](#)

Figure 1.1.1. The Login Page for ARC System Access.

Then you’ll find your **PID username** in the top right corner of the Open OnDemand webpage (Fig. 1.1.2).



ARC Open OnDemand Apps Files Jobs Clusters Interactive Apps IDE My Interactive Sessions Help Logged in as Your username Log Out

**OPEN**  
**OnDemand**

OnDemand provides an integrated, single access point for all of your HPC resources.

**Message of the Day**

This system is for authorized users only. Users accessing this system consent to the monitoring, recording and/or disclosure of all activity while using this system.

Usage of this system is subject to the terms of the [Virginia Tech Acceptable Use Guidelines](#)

**Pinned Apps** A featured subset of all available apps

**Files**

Home Directory  
System Installed App

**IDE**

Code Server System Installed App  
Eclipse System Installed App  
Nvidia-Nsight-Eclipse System Installed App  
PyCharm System Installed App

Figure 1.1.2. The Open OnDemand Webpage after logging in.

## 1.2. Click the “Interactive Apps” Icon and you will see a drop-down list (Fig. 1.2.1):

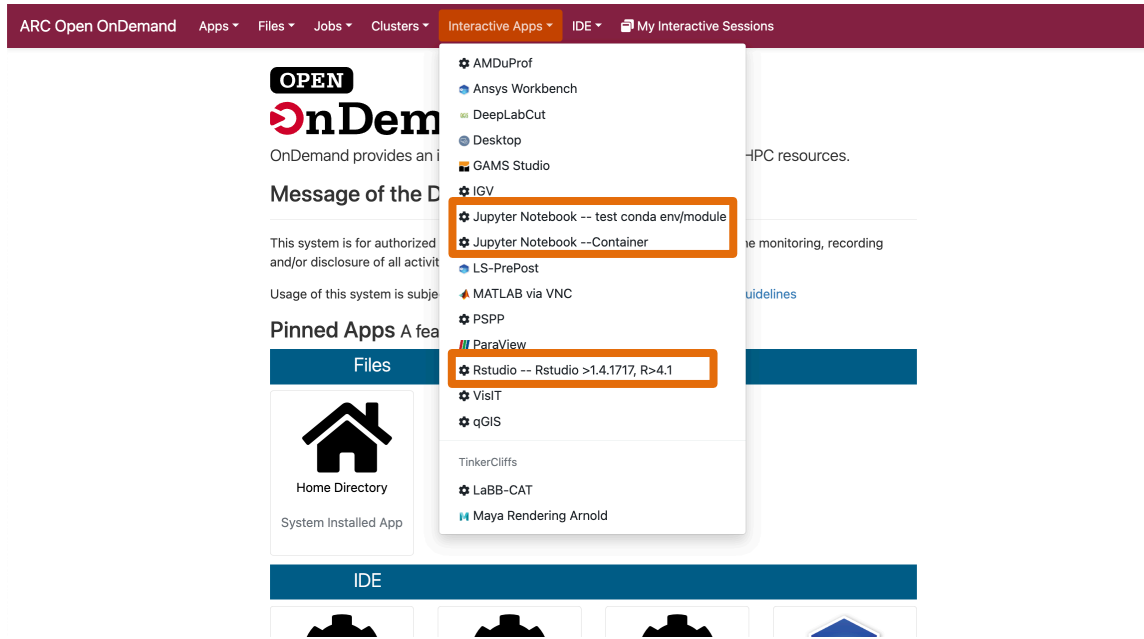


Figure 1.2.1. The drop-down menu for “Interactive Apps”.

## 1.3. Select the app you need, and you will be taken to a resource-requesting page (Fig. 1.3.1).

Provide the slurm account name and specify the computational resources you need to request. For information on slurm accounts, please refer to *Section 4.4* of the “ARC Data Transfer and Management” guidance document.

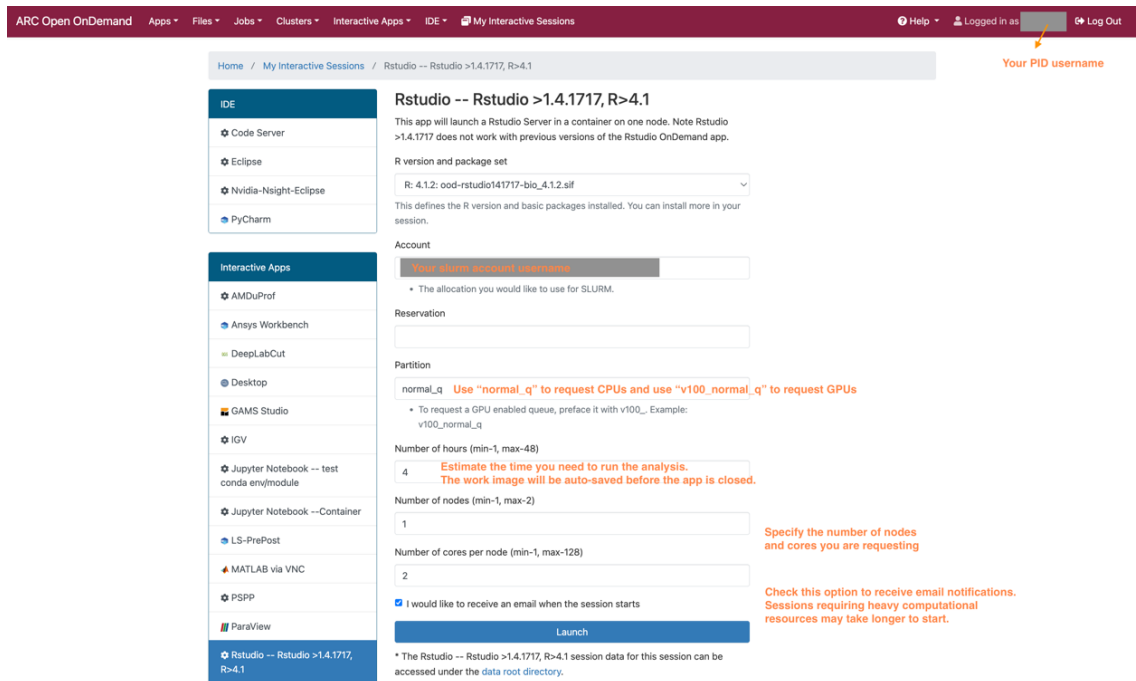


Figure 1.3.1. The resource-requesting webpage for RStudio.

ARC makes interactive RStudio sessions available by running singularity containers. You will be able to select a RStudio version and package set available on the drop-down list (Fig. 1.3.2).

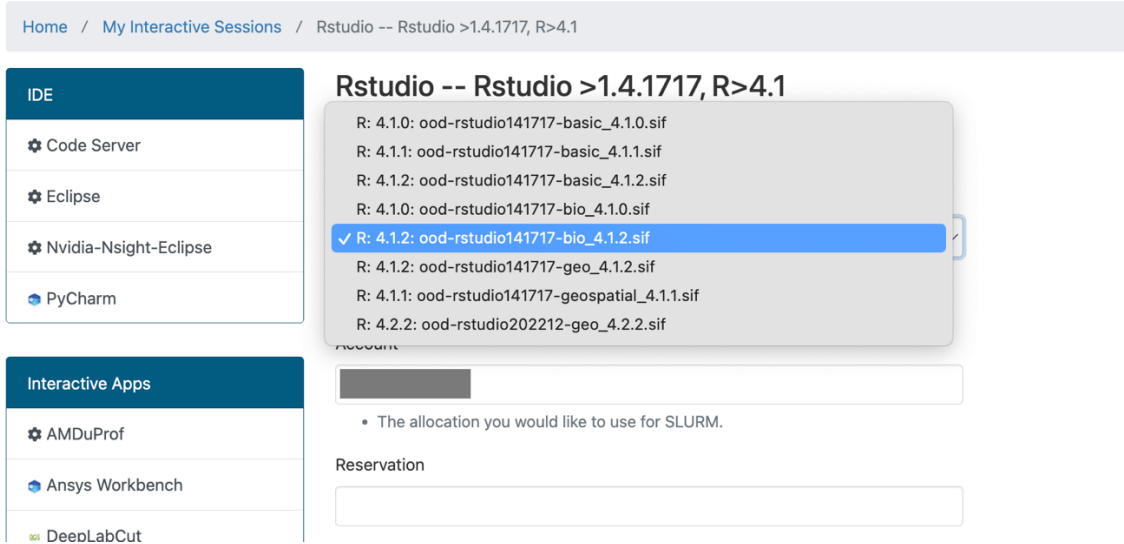


Figure 1.3.2. The drop-down menu for RStudio version selection.

For Jupyter Notebook, the process is similar, except that you can specify the conda environment and project path, if you select “Jupyter Notebook -- test conda env/module”.

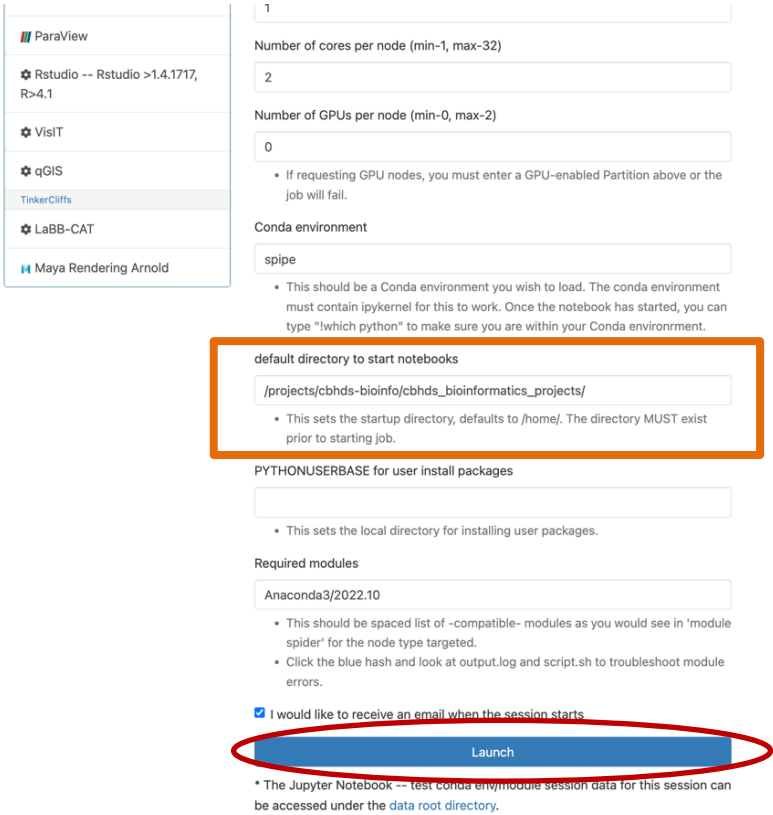


Figure 1.3.3. The resource-requesting webpage for “Jupyter Notebook -- test conda env/module”.

1.4. Click “Launch” at the bottom of the resource-requesting page (Fig. 1.3.3).

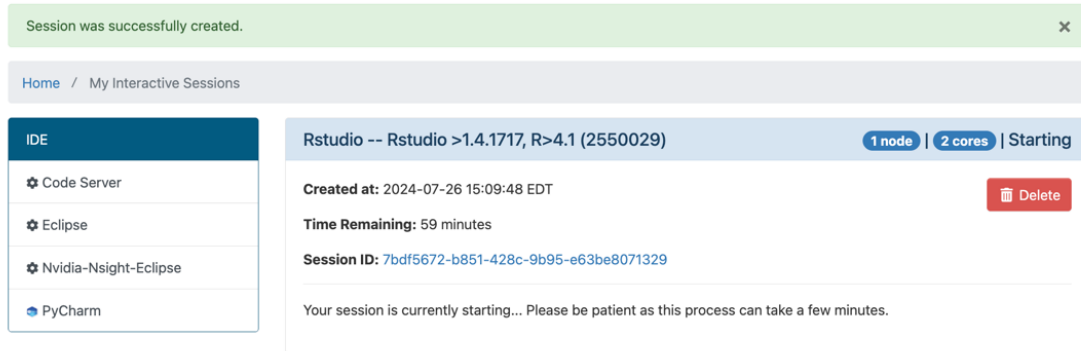


Figure 1.4.1. Starting an interactive RStudio session.

You will see a message saying “your session is currently starting”. When it’s ready, click “Connect to RStudio Server”.

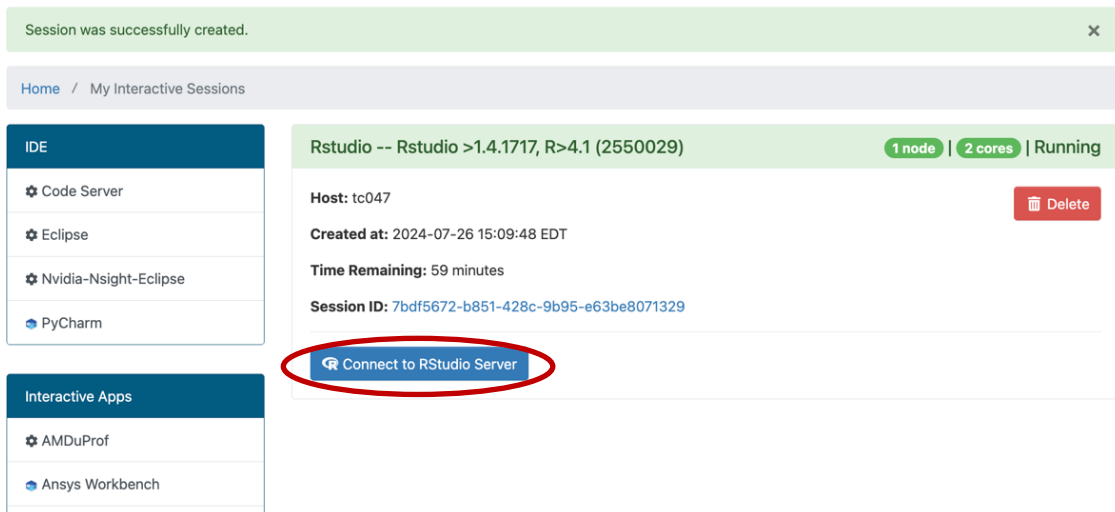


Figure 1.4.2. A remote RStudio server is ready.

- 1.5. Then a new tab will be open on your browser, and you can work in the interactive RStudio session as you would work on your local desktop.

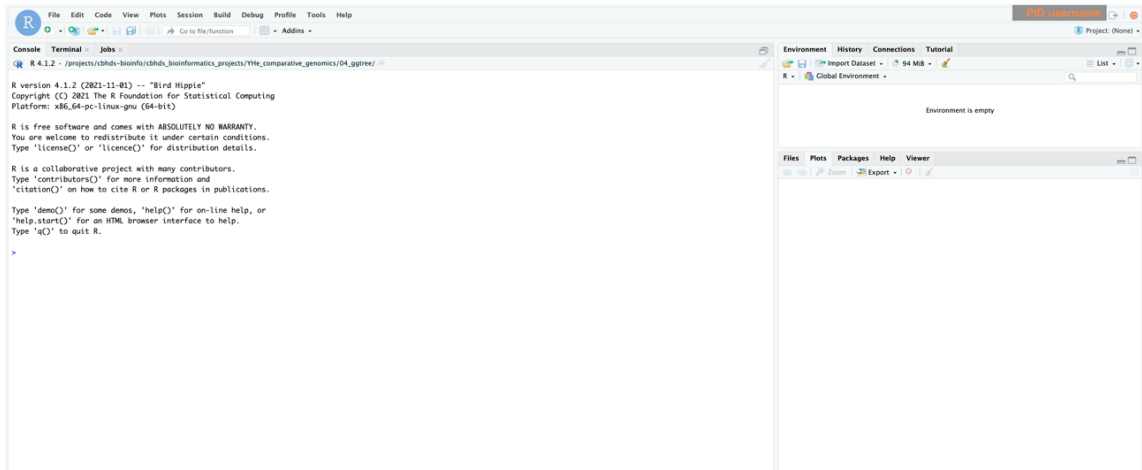


Figure 1.5.1. A remote RStudio session.

Set your working directory with your project path. For example, `“setwd(“/projects/cbhds/project_001”)”`. Then you are ready to go!

## 2. Running RStudio and Jupyter Notebook Through A Remote Server on A Compute Node

This section contains instructions for requesting computational resources from ARC and launching Jupyter Notebook and RStudio on a compute node. Please make sure you are connected to VT’s network before you start following our examples. If you are connected to off-campus internet, you can use VT’s [Remote Access VPN](#) to access Blacksburg campus university services.

### 2.1. Access HPC Compute Resources (Clusters)

There are multiple ways to request computational resources from ARC. You can submit a sbatch job script (**Recommended**) for non-interactive execution or run interactive jobs using “salloc” and “srun” commands. VT ARC provided [an example script](#) to launch Jupyter Notebook on a compute node. Since the hpc cluster “huckleberry” was retired in 2022, we are providing an updated example modified based on this existing example.

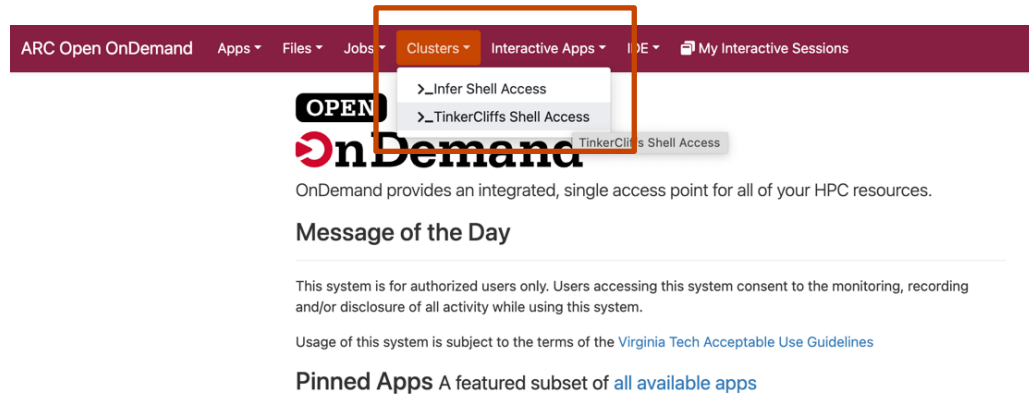
You can get access to ARC clusters through [the “Open OnDemand” website](#) or via SSH connections. ARC provided video tutorials for browser-based cluster access ([https://video.vt.edu/media/ARCA+Open+OnDemand+for+Browser-based+Cluster+Access/1\\_nkp1ebuu/176584251](https://video.vt.edu/media/ARCA+Open+OnDemand+for+Browser-based+Cluster+Access/1_nkp1ebuu/176584251)) and SSH-based cluster access ([https://video.vt.edu/media/ARCA+Accessing+clusters+from+the+command+line+via+SSH/1\\_nkojfb72/176584251](https://video.vt.edu/media/ARCA+Accessing+clusters+from+the+command+line+via+SSH/1_nkojfb72/176584251)).

Here we are providing a step-by-step guide for accessing ARC clusters from a web browser.

**2.1.1** Go to [the “Open OnDemand” website](#) and log in as described in [Step 1.1](#).

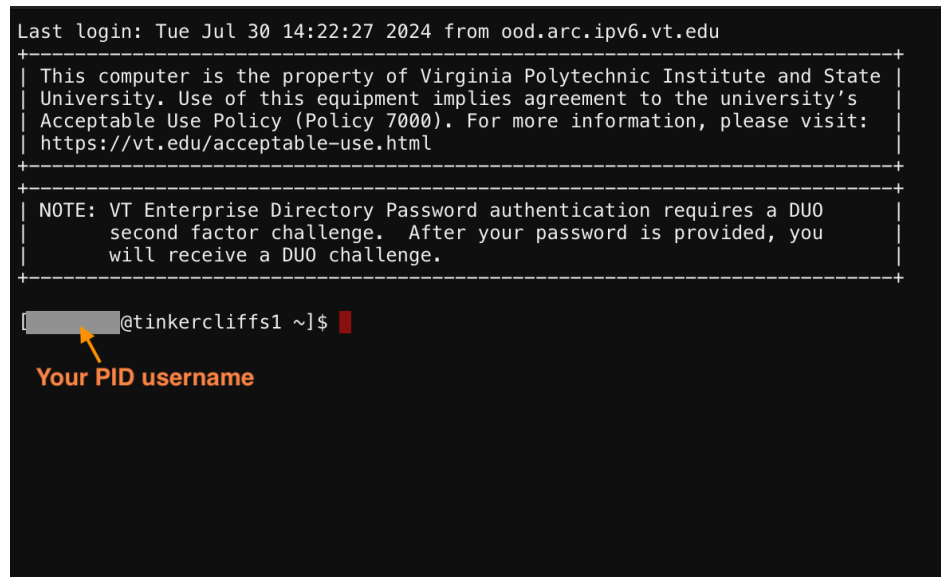
Click “Clusters” to select the cluster you need. As of July 2024, there are two clusters available: “TinkerCliffs” and “Infer”. Please click to review the hardware resource and usage policy for “[TinkerCliffs](#)” and “[Infer](#)”. Since “Infer” is a GPU cluster nearing end of life, we will use “TinkerCliffs” in this instruction.

Now, please click “TinkerCliffs Shell Access” on the drop-down list.



*Figure 2.1.1. Accessing the cluster.*

**2.1.2** You might need to verify your login activity on the Duo App. After that, you will see a Command Line Interface (CLI) similar to Fig 2.1.2. You have gained the access to the tinkercliffs cluster.



*Figure 2.1.2. The Command Line Interface of TinkerCliffs1*

## 2.2 Run Jupyter Notebook on A Compute Node using A Slurm Script

This example is built upon the example script provided by VT ARC (<https://www.docs.arc.vt.edu/resources/compute/huckleberry.html#jupyter-notebooks>).

### 2.2.1 In the Command Line Interface, please navigate to your project directory by running “cd /projects/your\_project\_path”.

```
Last login: Wed Jul 31 16:47:36 2024 from ood.arc.ipv6.vt.edu
+-----+
| This computer is the property of Virginia Polytechnic Institute and State |
| University. Use of this equipment implies agreement to the university's   |
| Acceptable Use Policy (Policy 7000). For more information, please visit:  |
| https://vt.edu/acceptable-use.html                                       |
+-----+
+-----+
| NOTE: VT Enterprise Directory Password authentication requires a DUO     |
| second factor challenge. After your password is provided, you          |
| will receive a DUO challenge.                                           |
+-----+
+-----+
[username@tinkercliffs1 ~]$ pwd      pwd: print the current directory path
/home/username
[username@tinkercliffs1 ~]$ cd /projects/cbhds/072024_Missi_test/
[username@tinkercliffs1 072024_Missi_test]$ pwd
/projects/cbhds/072024_Missi_test   cd: navigate to your
[username@tinkercliffs1 072024_Missi_test]$
```

Figure 2.2.1. Navigating to your project directory.

### 2.2.2 Create a text named as “jupyter\_example.sh” by running “nano ./ jupyter\_example.sh”.

```
[Your@tinkercliffs1 ~]$ cd /projects/cbhds/072024_Missi_test/
[username@tinkercliffs1 072024_Missi_test]$ nano ./jupyter_example.sh
```

Figure 2.2.2. Creating a text file.

### 2.2.3 This will open a new window for you to edit the text file.

Please copy and paste the following slurm script and press “control” and “O” simultaneously on your keyboard and then hit “return” to save the edited file. Then press “control” and “X” simultaneously on your keyboard and then hit “return” to exit the editing window. Now, run “cat ./jupyter\_example.sh” to review the script to make sure it was successfully edited and saved.

```

@tinkercliffs1 ~]$ cd /projects/cbhds/072024_Missi_test/
@tinkercliffs1 072024_Missi_test]$ nano ./jupyter_example.sh
@tinkercliffs1 072024_Missi_test]$ cat ./jupyter_example.sh
#!/bin/bash
#SBATCH --job-name=start-jupyter
#SBATCH --partition=normal_q
#SBATCH --nodes=1 #number of nodes
#SBATCH --ntasks-per-node=16 #maximum number of tasks per node
#SBATCH --mem=16G #memory per node
#SBATCH --time=00:00:00 #specify the runtime using HH:MM:SS
#SBATCH --account= slurm_account #replace it with your slurm account name
#SBATCH --mail-user PID@vt.edu #replace it with your own email address
#SBATCH --mail-type BEGIN
#SBATCH --mail-type END
#SBATCH --output=start-jupyter.%J.out
#SBATCH --error=start-jupyter.%J.err

# Create an ssh tunnel
XDG_RUNTIME_DIR=""
node=$(hostname -s)
user=$(whoami)
cluster="tinkercliffs1"
port=8888 # you can replace 8888 with 8889, 8890, 8891, or 8892.

ssh -N -f -R ${port}:localhost:${port} ${user}@${cluster}.arc.vt.edu

# Print the instructions
echo -e "
# Note: default port: 8888. In case port 8888 is not available, please go to
start-jupyter.%J.err to find the alternative port info. Or simply replace 8888
with 8889, 8890, 8891, or 8892 in the above code line.

# Run this command in the terminal on your laptop:
ssh -N -f -L 8888:${node}:8888 ${user}@${cluster}.arc.vt.edu

# Use a browser on your local machine to go to:
http://localhost:8888/
"

# Load the module and activate the conda environment
module load Anaconda3/2024.02-1 #you just run module load Anaconda3 without
specifying the version
source activate test #replace "test" with the conda environment you intend to work
in

# If jupyter is not installed, please remove the "#" in the following line:
# conda install jupyter -y

# Run Jupyter
echo "starting jupyter notebook"
jupyter-notebook --no-browser --port=${port} --ip=${node}

# Keep the interactive session alive for x seconds: 600 seconds=10 minutes
sleep 600

exit

```

Figure 2.2.3. The editing window opened by nano.

The slurm script:

```

#!/bin/bash
#SBATCH --job-name=start-jupyter
#SBATCH --partition=normal_q
#SBATCH --nodes=1 #number of nodes
#SBATCH --ntasks-per-node=16 #maximum number of tasks per node
#SBATCH --mem=16G #memory per node
#SBATCH --time=00:10:00 #specify the runtime using HH:MM:SS
#SBATCH --account= slurm_account #replace it with your slurm account name
#SBATCH --mail-user PID@vt.edu #replace it with your own email address
#SBATCH --mail-type BEGIN
#SBATCH --mail-type END
#SBATCH --output=start-jupyter.%J.out
#SBATCH --error=start-jupyter.%J.err

# Create an ssh tunnel
XDG_RUNTIME_DIR=""
node=$(hostname -s)
user=$(whoami)
cluster="tinkercliffs1"
port=8888 # you can replace 8888 with 8889, 8890, 8891, or 8892.

ssh -N -f -R ${port}:localhost:${port} ${user}@${cluster}.arc.vt.edu

# Print the instructions
echo -e "
# Note: default port: 8888. In case port 8888 is not available, please go to
start-jupyter.%J.err to find the alternative port info. Or simply replace 8888
with 8889, 8890, 8891, or 8892 in the above code line.

# Run this command in the terminal on your laptop:
ssh -N -f -L 8888:${node}:8888 ${user}@${cluster}.arc.vt.edu

# Use a browser on your local machine to go to:
http://localhost:8888/
"

# Load the module and activate the conda environment
module load Anaconda3/2024.02-1 #you just run module load Anaconda3 without
specifying the version
source activate test #replace "test" with the conda environment you intend to work
in

# If jupyter is not installed, please remove the "#" in the following line:
# conda install jupyter -y

# Run Jupyter
echo "starting jupyter notebook"
jupyter-notebook --no-browser --port=${port} --ip=${node}

# Keep the interactive session alive for x seconds: 600 seconds=10 minutes
sleep 600

exit

```

## 2.2.4 Run the following code to set a password:

```
module load Anaconda3/2024.02-1 #you just run "module load Anaconda3" without
specifying the version
source activate test #replace "test" with the conda environment you intend to work
in
jupyter notebook --generate-config
jupyter notebook password
```

Then type any password you wish to set and verify it by typing it again.

```
[redacted@tinkercliffs1 072024_Missi_test]$ module load Anaconda3/2024.02-1
[redacted@tinkercliffs1 072024_Missi_test]$ source activate test
(test) [redacted@tinkercliffs1 072024_Missi_test]$ jupyter notebook --generate-config
Writing default config to: /home/[redacted]/.jupyter/jupyter_notebook_config.py
(test) [redacted@tinkercliffs1 072024_Missi_test]$ jupyter notebook password
Enter password:
Verify password:
[JupyterPasswordApp] Wrote hashed password to /home/missizxm/.jupyter/jupyter_server_config.json
```

Figure 2.2.4. Setting a password for the Jupyter notebook access.

## 2.2.5 Run “sbatch jupyter\_example.sh” and you will see the job ID.

You can run “`queue --job the_job_ID`” to review the job status and run “`scancel the_job_ID`” to cancel the job.

```
(test) [redacted@tinkercliffs1 072024_Missi_test]$ sbatch jupyter_example.sh
Submitted batch job 2565023
(test) [redacted@tinkercliffs1 072024_Missi_test]$ queue --job 2565023
      JOBID PARTITION  NAME  USER ST  TIME  NODES NODELIST(REASON)
      2565023  normal_q  start-ju [redacted]  R    0:47    1  tc003
```

Figure 2.2.5. Running the slurm script and reviewing the job status.

## 2.2.6 Run “ls” and you will see the “.err” and “.out” files. Run “cat” followed by the output file name, and you will see the URL for Jupyter notebook connection and the command you’ll need to run in your local terminal (highlighted in green).

```
(test) [redacted@tinkercliffs1 072024_Missi_test]$ ls
jupyter_example.sh start-jupyter.2565023.err start-jupyter.2565023.out
(test) [redacted@tinkercliffs1 072024_Missi_test]$ cat start-jupyter.2565023.out
# Note: default port: 8888. In case port 8888 is not available, please go to start-jupyter.%J.err to find the alternative port info.
# Run this command in the terminal on your laptop:
ssh -N -f -L 8888:tc003:8888 [redacted@tinkercliffs1.arc.vt.edu]
# Use a browser on your local machine to go to:
http://localhost:8888/
starting jupyter notebook
```

Figure 2.2.6. Reviewing the instructions in the slurm output file.

## 2.2.7 Open a local terminal window and run the ssh command printed in the .out file. The command should look like “ssh -N -f -L port#:node#:port# username@tinkercliffs1.arc.vt.edu”. Then type your PID password and you might need to verify the login in the DUO Mobile app.

```
[redacted]@STAT-MZ-440903 ~ % ssh -N -f -L 8888:tc003:8888 [redacted@tinkercliffs1.arc.vt.edu]
Enter passphrase for key '/Users/[redacted]/.ssh/id_rsa':
```

Figure 2.2.7. Running the ssh command in your local terminal window.

## 2.2.8 Copy the URL (<http://localhost:8888/>) provided in the slurm output and paste it in your web browser. Type the password you set in Step 2.2.4.

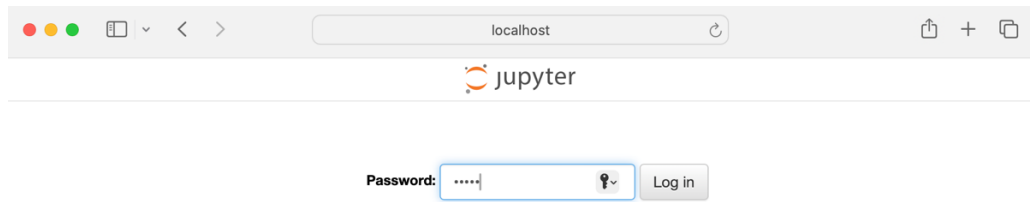


Figure 2.2.8. Accessing the Jupyter notebook server using your password.

Then you will see the Jupyter Notebook dashboard. Now, you are ready to roll!

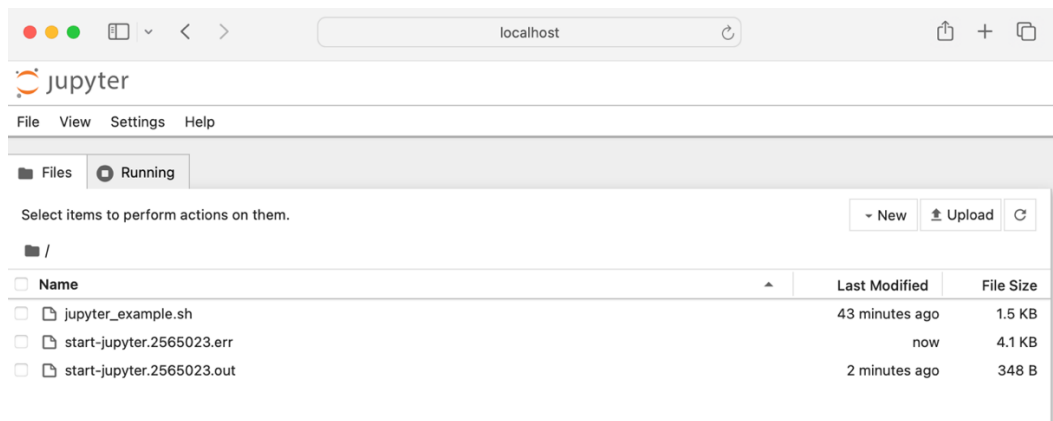


Figure 2.2.9. A Jupyter Notebook dashboard.

### 2.3 Run RStudio on A Compute Node using a Slurm Script

Running RStudio on ARC requires a different approach, because there is not a RStudio module available on ARC. However, you can run R scripts using a R module, and here is an [example submission script](#) provided by VT ARC.

In order to run RStudio on ARC, we will need to create or find a singularity image (a SIF, Singularity Image Format, file) that packs the R and RStudio versions along with most of the packages you need. Then we can use the SIF file to launch a container on ARC, in which we can run a RStudio remote session. VT ARC provided [quick instructions](#) for the usages of singularity on ARC. For more information about singularity (also known as apptainer), please visit <https://apptainer.org/user-docs/master/introduction.html> or <https://apptainer.org/docs/user/latest/introduction.html>.

The following instructions were created based on information from the multiple online resources including <https://rocker-project.org/use/singularity.html>, <https://www.rc.virginia.edu/userinfo/howtos/rivanna/launch-rserver/>, <https://www.hpc.iastate.edu/guides/containers/rstudio>, and <https://jiayiliu.me/post/2023-07-18-deploy-container-on-hpc/#how-to-run-a-container-on-hpc>.

**2.3.1** In the Command Line Interface, please navigate to your project directory by running “cd /projects/your\_project\_path”.

```
Last login: Wed Jul 31 16:47:36 2024 from ood.arc.ipv6.vt.edu
+-----+
| This computer is the property of Virginia Polytechnic Institute and State |
| University. Use of this equipment implies agreement to the university's  |
| Acceptable Use Policy (Policy 7000). For more information, please visit: |
| https://vt.edu/acceptable-use.html                                     |
+-----+
| NOTE: VT Enterprise Directory Password authentication requires a DUO    |
| second factor challenge. After your password is provided, you          |
| will receive a DUO challenge.                                         |
+-----+

[username@tinkercliffs1 ~]$ pwd      pwd: print the current directory path
/home/username
[username@tinkercliffs1 ~]$ cd /projects/cbhds/072024_Missi_test/
[username@tinkercliffs1 072024_Missi_test]$ pwd
/projects/cbhds/072024_Missi_test   cd: navigate to your
[username@tinkercliffs1 072024_Missi_test]$
```

Figure 2.3.1. Navigating to your project directory.

**2.3.2** First, we need a singularity image (.SIF file) that packs most of the things you need for running a RStudio session. You can pull (download) an image from [DockerHub](#), or [build your own image](#).

Since [the Rocker Project team](#) provided [many images](#) where RStudio server and basic packages are installed, and those images are available in the DockerHub container image library: <https://hub.docker.com/u/rocker>. For example, this image “[tidyverse:latest](#)” contains R, RStudio, and tidyverse packages.

Now, let us pull this image to our working directory by running “module load containers/apptainer/1.1.8” and then “apptainer pull docker://rocker/tidyverse:latest”. If the system you’re using does not apptainer v1.1.8. You can just run “**module load containers/apptainer**” and it will load the latest version of apptainer that is available.

```

[missizxm@tinkercliffs1 072024_Missi_test]$ module load containers/apptainer/1.1.8
[missizxm@tinkercliffs1 072024_Missi_test]$ apptainer pull docker://rocker/tidyverse:latest
INFO: Converting OCI blobs to SIF format
INFO: Starting build...
Getting image source signatures
Copying blob aae3293b611a done
Copying blob 7646c8da3324 done
Copying blob 2a532d0e6beb done
Copying blob 7ccbc6ce8ef9 done
Copying blob db5aa940be08 done
Copying blob a2977a275248 done
Copying blob 565a86b8dc8d done
Copying blob 7bd8d0b63f2f done
Copying blob 08516011f9e4 done
Copying blob 2041332265ca done
Copying blob 5b9c39d18b55 done
Copying blob 286ca6f97446 done
Copying blob b5c2e531b988 done
Copying blob dc90a1a57065 done
Copying blob 84068eccbae4 done
Copying blob b2f9dd331a53 done
Copying blob 8670ebf4d2d3 done
Copying blob 8063b8a0feb1 done
Copying blob e76623234471 done
Copying blob baf0ee110225 done
Copying config 7d6bfa77fc done
Writing manifest to image destination
Storing signatures
2024/08/07 16:25:30 info unpack layer: sha256:7646c8da332499ae416b15479ce832db32e39a501c662e24324f595509a0d3db
2024/08/07 16:25:31 info unpack layer: sha256:2a532d0e6beb480150ec53012d158da517e109bdab9e9bdf5a29c407a58d5a8f
2024/08/07 16:25:31 info unpack layer: sha256:db5aa940be0845edf2594157eceb27c48eca1ea15c4916b736b3f5543225d431
2024/08/07 16:25:36 info unpack layer: sha256:7ccbc6ce8ef929270247e2546a97c2f4a7a5d98d7a107aeec3b15c79590a69c3
2024/08/07 16:25:36 info unpack layer: sha256:a2977a27524881350809d399c4d3f5d96a8c4ced9708a2c7fd3f9835d2969af1
2024/08/07 16:25:36 info unpack layer: sha256:aae3293b611a9aa0fed0bc35d2c07741bfffef2fec8ead86c707177daee0fe
2024/08/07 16:25:37 info unpack layer: sha256:565a86b8dc8de93b38058a5d7740d399ecfb6507d959c0681c9d85925fc0deb6
2024/08/07 16:25:37 info unpack layer: sha256:08516011f9e4651ae6846f31abe6dbab09ef21413d72233ea12e7f240dccb8f5
2024/08/07 16:25:41 info unpack layer: sha256:7bd8d0b63f2fa9d0d6e17c3117c429e51485f04f7e145ebae4c12c8700189c0c
2024/08/07 16:25:41 info unpack layer: sha256:2041332265ca27f2c5d7b802d1e1dcfe1839c435ec324f551ea23cccd2398a52
2024/08/07 16:25:41 info unpack layer: sha256:286ca6f97446c1816260c59531a633252268e5b8e762deb56b1ebc6a96446f3
2024/08/07 16:25:42 info unpack layer: sha256:5b9c39d18b55123d05e82eb148f0e299ce4f1728b57a983b33600514d587074a
2024/08/07 16:25:42 info unpack layer: sha256:b5c2e531b988aacf132881f5320bb5d33390ba048a7ea28d791f70dd74c53217
2024/08/07 16:25:42 info unpack layer: sha256:dc90a1a570651815136ffffc2b101a8e6eefff330adb7a7d10a0deed467fd1616b
2024/08/07 16:25:42 info unpack layer: sha256:84068eccbae4a9a8fa8a787e3e29a190615bf61cfd22b321b0912e2eed0704
2024/08/07 16:25:50 info unpack layer: sha256:b2f9dd331a53a103c680f19a57f8e07544e57a1fcd5134cb5c38faa0eadc5188
2024/08/07 16:25:50 info unpack layer: sha256:8670ebf4d2d3685744b7d1fd51c0699bd271d64b810f6850f226e5d5bfbcbbe1
2024/08/07 16:25:50 info unpack layer: sha256:8063b8a0feb1699bd0ce6ab0c93bf0c8ec20edaf6f130e23a8bbbbb1f817d6de
2024/08/07 16:25:50 info unpack layer: sha256:e766232344718498aa27a6753c5640ab8c1d0179ae2d638bac81e5d7527faaa
2024/08/07 16:25:50 info unpack layer: sha256:baf0ee110225f6ff717e490c3aa0ea7d90e7e1c11b96d90860d52a478a64b91
INFO: Creating SIF file...

```

Figure 2.3.2. Pulling a docker image from DockerHub and creating a SIF file based on that.

### 2.3.3 Run “ls” and you will see a SIF in the current directory.

```

[username@tinkercliffs1 072024_Missi_test]$ ls
jupyter example.sh start-jupyter,2565023.err start-jupyter,2565023.out tidyverse_latest.sif

```

Figure 2.3.3. A SIF file is present in the current working directory.

### 2.3.4 Create a text named as “rstudio\_1time\_setup.sh” by running “nano ./rstudio\_1time\_setup.sh”.

```

[username@tinkercliffs1 072024_Missi_test]$ nano ./rstudio_1time_setup.sh

```

Figure 2.3.4. Creating a text file named “rstudio\_1time\_setup.sh”.

### 2.3.5 This will open a new window for you to edit the text file.

Please copy and paste the following bash script and press “control” and “O” simultaneously on your keyboard and then hit “return” to save the edited file. Then press “control” and “X” simultaneously on your keyboard and then hit “return” to exit the editing window.

```

GNU nano 2.3.1 File: ./rstudio_ltime_setup.sh
#!/bin/bash
workdir=$1
TMPDIR=${workdir}
mkdir -p $TMPDIR/tmp/rstudio-server
uuidgen -e $TMPDIR/tmp/rstudio-server/secure-cookie-key
chmod 600 $TMPDIR/tmp/rstudio-server/secure-cookie-key
mkdir -p $TMPDIR/var/{lib,run}
  
```

Figure 2.3.5. Writing the bash script.

The bash script:

```

#!/bin/bash

workdir=$1
TMPDIR=${workdir}
mkdir -p $TMPDIR/tmp/rstudio-server
uuidgen -e $TMPDIR/tmp/rstudio-server/secure-cookie-key
chmod 600 $TMPDIR/tmp/rstudio-server/secure-cookie-key
mkdir -p $TMPDIR/var/{lib,run}
  
```

Technically, “TMPDIR” can be set as any directory where you have write permissions. Here we use the “workdir=\$1” to take the path you will provide when execute the bash script as your “TMPDIR”. This script sets up a secure and organized directory structure for RStudio server's temporary and variable data storage, and those directories will be bind-mounted at runtime when you launch the container.

**2.3.6 Run “cat ./rstudio\_ltime\_setup.sh” to review the script to make sure it was successfully edited and saved.**

```

GNU nano 2.3.1 File: ./rstudio_ltime_setup.sh
#!/bin/bash
workdir=$1
TMPDIR=${workdir}
mkdir -p $TMPDIR/tmp/rstudio-server
uuidgen -e $TMPDIR/tmp/rstudio-server/secure-cookie-key
chmod 600 $TMPDIR/tmp/rstudio-server/secure-cookie-key
mkdir -p $TMPDIR/var/{lib,run}
  
```

Figure 2.3.6. Reviewing the bash script.

**2.3.7 Run “ls” and you will find the “rstudio\_ltime\_setup.sh” file in the current directory. Run “chmod +x rstudio\_ltime\_setup.sh” to make this script exceptionable. Notice the text color of this file has changed to green?**

```

[username@tinkercliffs1 072024_Missi_test]$ ls
jupyter_example.sh  rstudio_ltime_setup.sh  start-jupyter.2565023.err  start-jupyter.2565023.out  tidyverse_latest.sif
[username@tinkercliffs1 072024_Missi_test]$ chmod +x rstudio_ltime_setup.sh
[username@tinkercliffs1 072024_Missi_test]$ ls
jupyter_example.sh  rstudio_ltime_setup.sh  start-jupyter.2565023.err  start-jupyter.2565023.out  tidyverse_latest.sif
[username@tinkercliffs1 072024_Missi_test]$
  
```

Figure 2.3.7. Changing the file permission of the bash script.

**2.3.8 Execute the bash script “rstudio\_1time\_setup.sh”. Note that this script is to be executed as a one-time setup on the frontend. You may need to rerun this script for running a new rocker container. And you will need to provide a path for holding RStudio server's temporary and variable data (See 2.3.5).**

In this case, we are creating a new directory in the current working directory to store the RStudio server data. So, run

```
“./rstudio_1time_setup.sh /projects/cbhds/072024_Missi_test/rstudio_example”.
```

Please replace “/projects/cbhds/072024\_Missi\_test/rstudio\_example” with any path where you would like to store the RStudio server data.

```
@tinkercliffs1 072024_Missi_test]$ mkdir rstudio_example
@tinkercliffs1 072024_Missi_test]$ ./rstudio_1time_setup.sh /projects/cbhds/072024_Missi_test/rstudio_example
```

Figure 2.3.8. Executing the one-time setup bash script.

Here is a snapshot of the current directory structure (you’ll need run the “tree” command to view the directory structure, but it is not necessary). Now, we have set up the RStudio server tmp and var directories.



**2.3.9 Now, let us write a slurm script to launch RStudio server. Unlike the one-time setup bash script, which we only run it once, we need to submit the slurm script every time we need to run a RStudio session.**

This process is similar to Step 2.3.4-2.3.6:

Run “nano ./launch\_rstudio\_server\_example.sh”. Please copy and paste the following slurm script and press “control” and “O” simultaneously on your keyboard and then hit “return” to save the edited file. Then press “control” and “X” simultaneously on your keyboard and then hit “return” to exit the editing window.

The slurm script:

```
#!/bin/bash
#SBATCH --job-name=start-rstudio
#SBATCH --partition=normal_q
#SBATCH --nodes=1 #number of nodes
#SBATCH --ntasks-per-node=16 #maximum number of tasks per node
#SBATCH --mem=16G #memory per node
#SBATCH --time=00:10:00 #specify the runtime using HH:MM:SS
#SBATCH --account=cbhds #replace it with your slurm account name
#SBATCH --mail-user missizxm@vt.edu #replace it with your own email address
#SBATCH --mail-type BEGIN
#SBATCH --mail-type END
#SBATCH --output=start-rstudio.%J.out
#SBATCH --error=start-rstudio.%J.err

# Specify path to container
myworkdir=$1
mytmpdir=$2
# Get the SIF
SIF=${myworkdir}/tidyverse_latest.sif

# Specify path to tmp directory created in previous section
TMPDIR=${mytmpdir}
```

```

# Load the apptainer module, you can just run module load containers/apptainer if
v1.1.8 is not available
module load containers/apptainer/1.1.8

# Configure RStudio Server behavior and user credentials
node=$(hostname -s) # Get the node info
user=$(whoami) # Get the current user's username
port=8787
cluster="tinkercliffs1" # Set the cluster name
export APPTAINERENV_RSTUDIO_SESSION_TIMEOUT=0 # Do not suspend idle sessions.

# Set up container username and password
export APPTAINERENV_USER=${user}
export APPTAINERENV_PASSWORD=$(openssl rand -base64 15)

# Print debugging information
echo "Debugging Information:"
echo "Node : ${node}"
echo "User: ${user}"
echo "Cluster: ${cluster}"
echo "Port: ${port}"
echo "Container Path: ${SIF}"
echo "Temporary Directory: ${TMPDIR}"

# Print instructions for setting up an SSH tunnel and logging into RStudio Server
echo -e "

1. SSH tunnel from your workstation using the following command (macOS, Linux and
Windows PowerShell):

    ssh -N -L ${port}:${node}:${port} ${user}@${cluster}.arc.vt.edu

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

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

    user: ${APPTAINERENV_USER}
    password: ${APPTAINERENV_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}

"

# Run RStudio server in a container
# If you need to access files in multiple project directories, replace --bind
${myworkdir} with --bind project_directory_absolute_path1, path2, path3
apptainer exec \
    -B ${myworkdir} \
    -B ${TMPDIR}/var/lib:/var/lib/rstudio-server \
    -B ${TMPDIR}/var/run:/var/run/rstudio-server \
    -B ${TMPDIR}/tmp:/tmp \
    $SIF \
    rserver \
    --server-user ${user} \
    --auth-none=0 \
    --auth-pam-helper-path=pam-helper \
    --auth-stay-signed-in-days=1 \
    --auth-timeout-minutes=0

echo -e "rserver exited"

exit

```

Run “cat ./launch\_rstudio\_server\_example.sh” to review the script to make sure it was successfully edited and saved.

```
[username@tinkercliffs1 072024_Missi_test]$ cat launch_rstudio_server_example.sh
#!/bin/bash
#SBATCH --job-name=start-rstudio
#SBATCH --partition=normal_q
#SBATCH --nodes=1 #number of nodes
#SBATCH --ntasks-per-node=16 #maximum number of tasks per node
#SBATCH --mem=16G #memory per node
#SBATCH --time=00:10:00 #specify the runtime using HH:MM:SS
#SBATCH --account=cbhds #replace it with your slurm account name
#SBATCH --mail-user missizxm@vt.edu #replace it with your own email address
#SBATCH --mail-type BEGIN
#SBATCH --mail-type END
#SBATCH --output=start-rstudio.%J.out
#SBATCH --error=start-rstudio.%J.err

# Specify path to container
myworkdir=$1
mytmpdir=$2
# Get the SIF
SIF=${myworkdir}/tidyverse_latest.sif

# Specify path to tmp directory created in previous section
TMPDIR=${mytmpdir}

# Load the apptainer module
module load containers/apptainer/1.1.8

# Configure RStudio Server behavior and user credentials
node=$(hostname -s) # Get the node info
user=$(whoami) # Get the current user's username
port=8787
cluster="tinkercliffs1" # Set the cluster name
export APPTAINERENV_RSTUDIO_SESSION_TIMEOUT=0 # Do not suspend idle sessions.

# Set up container username and password
export APPTAINERENV_USER=${user}
export APPTAINERENV_PASSWORD=$(openssl rand -base64 15)

# Print debugging information
echo "Debugging Information:"
echo "Node : ${node}"
echo "User: ${user}"
echo "Cluster: ${cluster}"
echo "Port: ${port}"
echo "Container Path: ${SIF}"
echo "Temporary Directory: ${TMPDIR}"

# Print instructions for setting up an SSH tunnel and logging into RStudio Server
echo -e "
1. SSH tunnel from your workstation using the following command (macOS, Linux and Windows PowerShell):
ssh -N -L ${port}:${node}:${port} ${user}@${cluster}.arc.vt.edu"

```

Figure 2.3.9. Reviewing the slurm script.

### 2.3.10 Now, we are ready to submit the slurm script and launch RStudio server.

Here you will need two directory path:

Path1: where your sif is stored. In this case, path1 is the current working directory:  
“/projects/cbhds/072024\_Missi\_test”

Path2: where the RStudio server’s tmp and var directories are located in. In other words, path2 is the “TMPDIR” in Step 2.3.5 and also the path you used in Step 2.3.8. In this case, path 2 is “/projects/cbhds/072024\_Missi\_test/rstudio\_example”

Run “sbatch ./launch\_rstudio\_server\_example.sh /projects/cbhds/072024\_Missi\_test /projects/cbhds/072024\_Missi\_test/rstudio\_example”

```
[username@tinkercliffs1 072024_Missi_test]$ sbatch ./launch_rstudio_server_example.sh /projects/cbhds/072024_Missi_test /projects/cbhds/072024_Missi_test/rstudio_example
Submitted batch job 2577427
```

Figure 2.3.10. Submitting the slurm script.

### 2.3.11 Run “ls” and find the slurm output file “start-rstudio.%J.out”. Run “cat

start-rstudio.%J.out” to review the connection instructions provided in this file.

In this case, we should run “cat start-rstudio.2577427.out” and you will find the following information from this file:

- The command you need to type in a terminal window on your local machine to establish a connection between your local machine and the remote machine through an ssh tunnel. It looks like “ssh -N -L 8787:tc002:8787 username@tinkercliffs1.arc.vt.edu”.
- A URL for you to run a RStudio interactive session on your web browser: “http://localhost:8787”
- The login credentials for you to connect to RStudio Server.

```
[username@tinkercliffs1 072024_Missi_test]$ ls
jupyter_example.sh          rstudio_time_setup.sh    start-jupyter.2565023.err  start-rstudio.2577427.err  tidyverse_latest.sif
launch_rstudio_server_example.sh  rstudio_example          start-jupyter.2565023.out  start-rstudio.2577427.out
[username@tinkercliffs1 072024_Missi_test]$ cat start-rstudio.2577427.out
Debugging Information:
Node : tc002
User : username
Cluster: tinkercliffs1
Port: 8787
Container Path: /projects/cbhds/072024_Missi_test/tidyverse_latest.sif
Temporary Directory: /projects/cbhds/072024_Missi_test/rstudio_example

1. SSH tunnel from your workstation using the following command (macOS, Linux and Windows PowerShell):
ssh -N -L 8787:tc002:8787 username@tinkercliffs1.arc.vt.edu
and point your web browser to http://localhost:8787

2. Log in to RStudio Server using the following credentials:
user: username
password: vLz3dX7IvePYbz1fjf2h

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 2577427
```

Figure 2.3.11. Reviewing the slurm output file to find instructions for connecting to RStudio Server from a local machine.

**2.3.12** Open a local terminal window and run the ssh command printed in the .out file. The command should look like “ssh -N -L 8787:tc002:8787 username@tinkercliffs1.arc.vt.edu”. Then type your PID password and you might need to verify the login in the DUO Mobile app.

```
local username @STAT-MZ-440903 ~ % ssh -N -L 8787:tc002:8787 username@tinkercliffs1.arc.vt.edu
Enter passphrase for key '/Users/username/.ssh/id_rsa':
```

Figure 2.3.12. Creating an ssh tunnel on a local machine.

**2.3.13** Copy the URL (http://localhost:8787/) provided in the slurm output and paste it in your web browser. Type the RStudio Server login credentials (See Step 2.3.11).

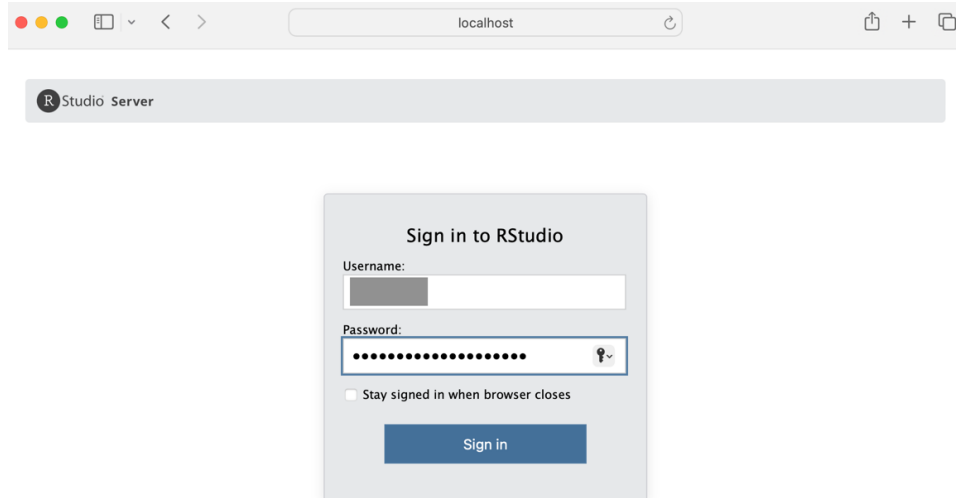


Figure 2.3.13. Connecting to RStudio Server using the login credentials from the slurm output file.

### 2.3.14 Voilà! You are connected to the RStudio Server.

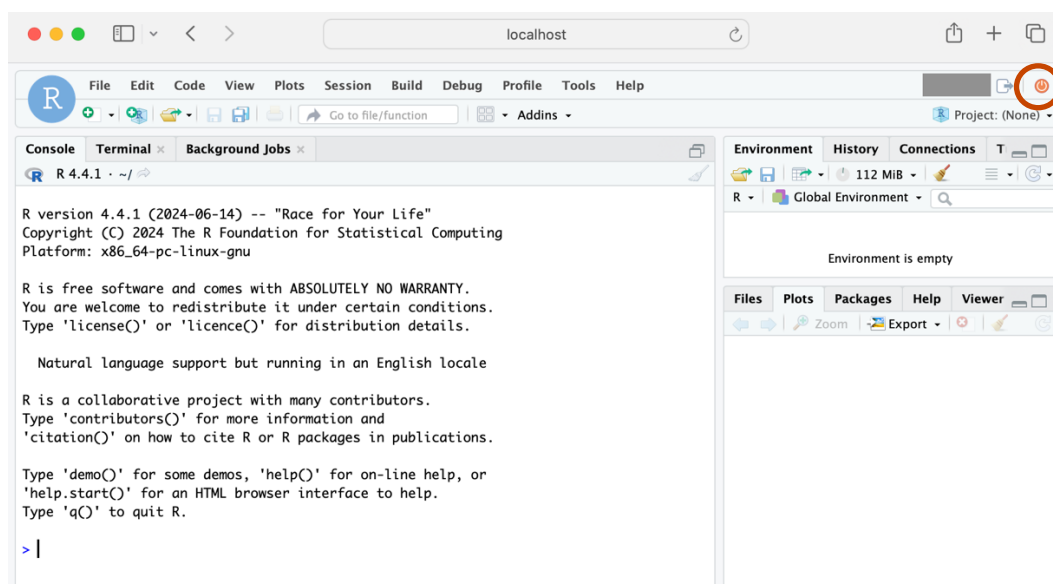


Figure 2.3.14. Running an interactive RStudio session.

After you finish your data analysis job, please **save your files and click the power icon in the top right corner and to the current session (see in red circle in the top right corner of Fig 2.3.14)**. Then, please run “**scancel -f**” followed by the job ID to terminate the job, as shown in the slurm output file (see 2.3.11).

## References & Additional Resources:

For quick guidance on ARC services and how to request an account, create projects, and manage data on ARC, please read our “[Data Transfer and Management](https://biostat.centers.vt.edu/process.html)” document (<https://biostat.centers.vt.edu/process.html>).

- VT ARC User Documentation: <https://www.docs.arc.vt.edu/index.html>
- VT ARC Workshops: <https://www.docs.arc.vt.edu/usage/workshops.html>
  - Intro to ARC
  - Connect to ARC and Run Your 1st Jobs
  - Using Software on ARC Systems
  - Launching in Parallel
  - Monitoring Job Efficiency and Resource Utilization
- Running non-interactive jobs using slurm scripts: <https://www.docs.arc.vt.edu/usage/slurm.html>
- Interactive and batch jobs: [https://video.vt.edu/media/ARCA+Interactive+and+Batch+Jobs/1\\_doz5ylqg/176584251](https://video.vt.edu/media/ARCA+Interactive+and+Batch+Jobs/1_doz5ylqg/176584251)
- Singularity/Apptainer:
  - [https://hsf-training.github.io/hsf-training-singularity-webpage/#:~:text=Apptainer%20\(formerly%20known%20as%20Singularity,%2C%20fast%2C%20and%20secure%20manner](https://hsf-training.github.io/hsf-training-singularity-webpage/#:~:text=Apptainer%20(formerly%20known%20as%20Singularity,%2C%20fast%2C%20and%20secure%20manner)
  - <https://apptainer.org/user-docs/master/introduction.html>
  - <https://apptainer.org/docs/user/latest/>
  - <https://www.docs.arc.vt.edu/software/singularity.html#singularity>
- Running RStudio on hpc:
  - <https://www.rc.virginia.edu/userinfo/howtos/rivanna/launch-rserver/>
  - <https://www.hpc.iastate.edu/guides/containers/rstudio>
  - <https://jiayiliu.me/post/2023-07-18-deploy-container-on-hpc/#how-to-run-a-container-on-hpc>
  - <https://rocker-project.org/use/singularity.html>
- Running Jupyter Notebook on hpc:
  - <https://www.docs.arc.vt.edu/resources/compute/huckleberry.html>
  - <https://sna.cs.colostate.edu/hpc/samples/gpu-jobs/jupyter/>
  - <https://services.dartmouth.edu/TDClient/1806/Portal/KB/ArticleDet?ID=76952>