STAT Cluster

Introduction

Matthew Hielsberg

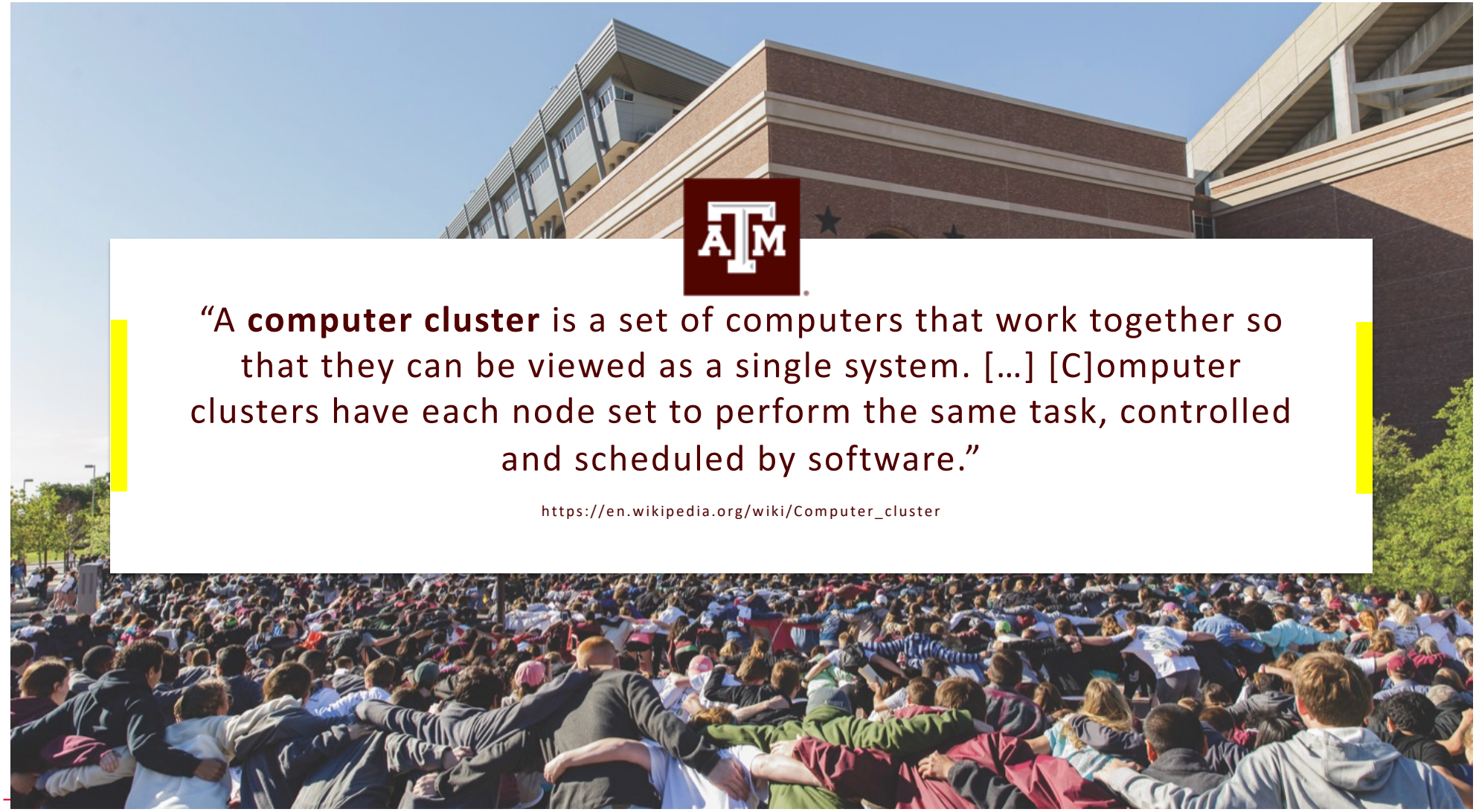
artsci-help@tamu.edu

Technology Services

Agenda

TEXAS A&M UNIVERSITY

- Resource Description
- Requesting an Account
- Usage Policy
- A First Login
- Home Directory and Shell
- Editors
- File Transfer
- Screen Sessions
- Available Software
- Installing Software
- Running Jobs
- Bash Files & Redirection
- Bash Scripting
- R
- Matlab
- Compiling C/Fortran
- Advanced SSH
- Best Practices (Opinionated)

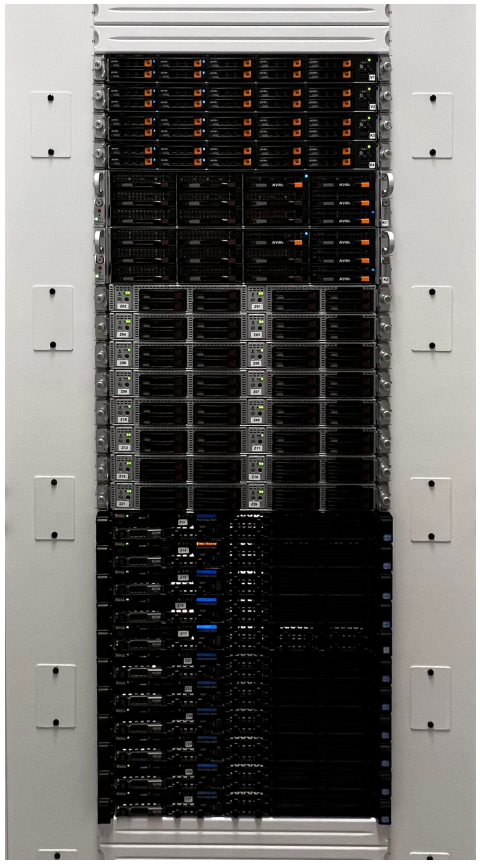


"A **computer cluster** is a set of computers that work together so that they can be viewed as a single system. […] [C]omputer clusters have each node set to perform the same task, controlled and scheduled by software."

https://en.wikipedia.org/wiki/Computer_cluster

STAT Cluster

TEXAS A&M UNIVERSITY



Heterogeneous Cluster

- 27 Compute Nodes
  - z1-z12 28-core Broadwell 64 GB
  - z13-16 24-core Ivy Bridge 128 GB
  - z17 28-core Haswell 112 GB
  - z18-z21 28-core Skylake 96 GB
  - z22-z27 16-core Sandy Bridge 128 GB
- Redundant Management Nodes
- File Server (~7.25 TB total storage)
- Backup + Offsite Backup
- Gigabit Ethernet

To be decommissioned no later than May 2024!

Requesting an Account

TEXAS A&M UNIVERSITY

Statistics Faculty / Staff / Students

Email artsci-help@tamu.edu, and include the following information:

- Full Name
- Net ID

Non-Department Faculty / Staff / Students

All non-department users must be sponsored by a department faculty or staff member and will maintain an account only for the duration of the project(s) specified when the account was created. Extensions/reactivations may be granted but will require a new account request.

Email artsci-help@tamu.edu, and include the following information:

- Sponsor's Full Name
- Applicant Full Name
- *Applicant Net ID
- Applicant Email (@tamu.edu)
- Short description of the research to be performed using the department's computing resources.
- Duration of need: How long will the applicant need to use the department's resources.

* Guest Net ID's may be requested from Identity Management Office http://u.tamu.edu/netidrequest, with approval from Dept. Head.

Policy

This computer system and data herein are available only for authorized purposes by authorized users in accordance with TAMU SAP 29.01.03.M0.02, Rules for Responsible Computing. Use for any other purpose may result in administrative/disciplinary actions and/or criminal prosecution against the user. Usage may be subject to security testing and monitoring. Users have no expectation of privacy except as otherwise provided by applicable privacy laws.

By using this computer system, you acknowledge and affirm that all data stored/used is only public data (DC-3) or university-internal data (DC-4) as defined in the Texas A&M Information Security Controls Catalog:

https://it.tamu.edu/policy/it-policy/controls-catalog/index.php#controls-DC

No data categorized as confidential (DC-5) or critical (DC-6) may be used, stored, or transmitted by this computer system.

This computer system is a shared resource, and we ask that users use only as many resources (cores and memory) that are necessary for their computation. Furthermore, users are limited to running processes on a maximum of **FOUR** compute nodes at any given time, and no process may exceed **SEVEN** days without prior approval. Users exceeding these limits will find their processes terminated. Additionally, this computer resource is not intended to be an ssh jump-host, and we ask that users do not initiate ssh connections from the computer resource to other systems with the exception version control systems (git/GitHub) or for using scp/sftp to move data.

Policy

Logging in to r1:

$ ssh NetID@r1.stat.tamu.edu

****

This computer system and the data herein are available only for authorized purposes by authorized users: use for any other purpose is prohibited and may result in administrative/disciplinary actions or criminal prosecution against the user. Usage may be subject to security testing and monitoring to ensure compliance with the policies of Texas A&M University, Texas A&M University System, and the State of Texas. There is no expectation of privacy on this system except as otherwise provided by applicable privacy laws.

Users should refer to Texas A&M University Standard Administrative Procedure 29.01.03.M0.02, Rules for Responsible Computing, for guidance on the appropriate use of Texas A&M University information resources.

****

Policy

Logging in to r1 (cont.):

```
…
NetID@r1's password:
Last login: Fri Jul 22 16:43:15 2022 from connect-172-31-54-160.vpn.tamu.edu

*** Maintenance is on the 3rd Thursday of each month,
*** and all nodes will be rebooted every month.
*** Public key SSH logins have been disabled for this system.
*** Type 'load' to see available systems.
*** Type 'quota' to see your storage quota.

*** R-4.3.0 has been installed, however 4.2.0 is the default R.
*** Use /usr/local/bin/R-4.3.0 explicitly to use the new R.
*** Matlab R2022a has been installed and is the default.
*** TeXLive 2022 has been installed and is the default.
*** julia-1.6.5 LTS and julia-1.7.2 have been installed.

*** R packages must be installed on this login node.

*** Do not run jobs/simulations on this system.

*** September 21 maintenance has been completed.
*** Next scheduled maintenance: October 19.
[NetID@r1:~]$
```

Current policy is to ALWAYS reboot!

Passwords are always required

Don’t install on the compute nodes

Jobs run on r1 will be terminated!

Policy

Logging in to a compute node (e.g. z12) from r1:

```
[NetID@r1:~]$ ssh z01
…
Last login: Mon Aug 28 08:20:44 2023 from r1.stat.tamu.edu

*** Jobs running for more than 7 days may be killed.
*** Jobs running on more than 4 nodes may be killed.
*** You must prefer compute nodes that are not in use.
*** Type 'load' to see available systems.

[NetID@z01:~]$
```

Notifications are emailed prior to terminating jobs exceeding these limits.

Policy

TEXAS A&M UNIVERSITY

Noted in output of load

[NetID@r1:~]$ load

System	Uptime	Load	CPU	Mem
z01 up	22+09	6.0	28	64G
.				
.				
.				
z27 up	22+08	0.0	16	128G
Total:		287.6	668	

Storage quota for NetID: 4.1G / 10G ← Quota Reminder: Used / Total

Note: Simulations running longer than 7 days may be killed. ← Notifications are emailed prior to terminating jobs exceeding these limits.

Limit your jobs to 4 compute nodes. If possible, fully load a node and use nodes that are not being used.

A First Login

TEXAS A&M UNIVERSITY

- Only accessible from within University network (department computers, TAMU wifi, eduroam, or VPN)

```
$ ssh NetID@r1.stat.tamu.edu

The authenticity of host ‘r1 (128.194.13.129)' can't be established.
…
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added ‘r1,128.194.13.129' to the list of known hosts.

[Policy Banner Omitted]

[NetID@r1:~]$
```

Home Directory + Shell

TEXAS A&M UNIVERSITY

The home directory is a directory (or folder) for a particular user (you) of the system.
Upon logging in it is set as the current working directory, and is referred to by a tilde (e.g. ~/) or its full path /home/NetID (where NetID is replaced with YOUR NetID).
This folder holds:

- All of your data (any files you have created)
- Any software you have installed (not installed by an admin)

By default, this directory is accessible only to you.

```
[NetID@r1:~]$                          # Home Directory (/home/NetID)

[NetID@r1:~]$ quota
Storage quota for NetID : 4.1G / 10G   # Quota: Used Space / Total Space

[NetID@r1:~]$ cd test                  # Change the current directory to '~/test'

[NetID@r1:~]$ mkdir subdir             # Make a new directory 'subdir' in ~/test
```

The Unix Shell (https://swcarpentry.github.io/shell-novice/)

Home Directory + Shell

TEXAS A&M UNIVERSITY

```
[NetID@r1:~/test]$ ls                    # List the contents of the current directory (~/test)

a.txt    b.txt    c    subdir

[NetID@r1:~/test]$ ls -l                 # List the contents of the current directory in long format

total 2
-  rw-  r-- r--.   1  NetID NetID 15 Jul 25 10:56  a.txt
-  rw-  r-- r--.   1  NetID NetID  0 Jul 25 10:56  b.txt
-  rwx r-x r-x.  1  NetID NetID  0 Jul 25 10:57  c
d rwx r-x r-x.  2  NetID NetID  2 Jul 25 11:01  subdir

[ spacing added above for readability ]
```

The Unix Shell (https://swcarpentry.github.io/shell-novice/)

Home Directory + Shell

TEXAS A&M UNIVERSITY

```
[NetID@r1:~/test]$ cat a.txt              # Concatenate files and print them to standard output

Hello, World!

[NetID@r1:~/test]$ rm a.txt               # Deletes a.txt (non-recoverable, except from backup)

[NetID@r1:~/test]$ ls -l                  # List the contents of the current directory in long format

total 1
-  rw-  r-- r--.   1  NetID NetID 0 Jul 25 10:56  b.txt
-  rwx r-x r-x.  1  NetID NetID 0 Jul 25 10:57  c
d rwx r-x r-x.  2  NetID NetID 2 Jul 25 11:01  subdir

[ spacing added above for readability ]
```

The Unix Shell (https://swcarpentry.github.io/shell-novice/)

Editors

Command Line Editors:

- nano (https://www.nano-editor.org)
- vi (https://vimhelp.org)

[NetID@r1:~/test]$ nano b.txt ← Enter text
Ctrl+O to save
Ctrl+X to exit

[NetID@r1:~/test]$ cat b.txt

New text entered via nano.

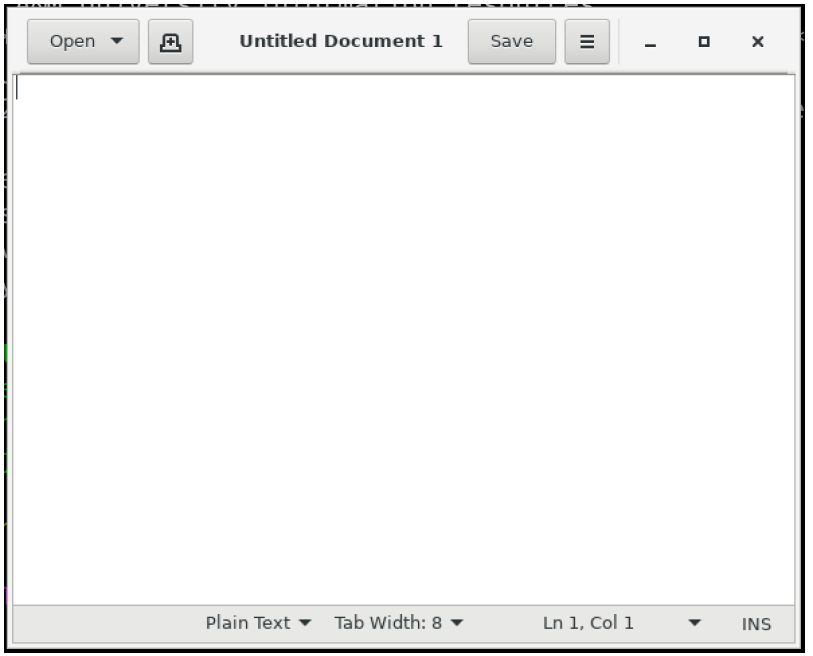
[NetID@r1:~/test]$ logout ← Ctrl+D to exit

Editors + X forwarding

GUI Editors:

- Emacs (https://www.gnu.org/software/emacs/)
- gedit (https://wiki.gnome.org/Apps/Gedit)

```
$ ssh -X NetID@r1.stat.tamu.edu
[…]
[NetID@r1:~]$ gedit
```



- Linux
  Should work by default
- MacOS
  XQuartz (https://www.xquartz.org)
- Windows :
  MobaXterm (https://mobaxterm.mobatek.net/download.html)

Note that GUI applications (i.e. Matlab) running on a compute node require X forwarding from the node as well (e.g. [NetID@r1:~]$ ssh -X z01)

Remote Editors: VSCode

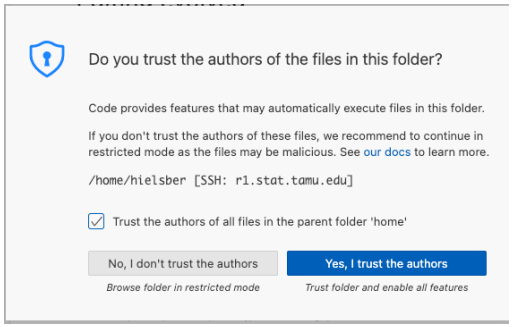
TEXAS A&M UNIVERSITY

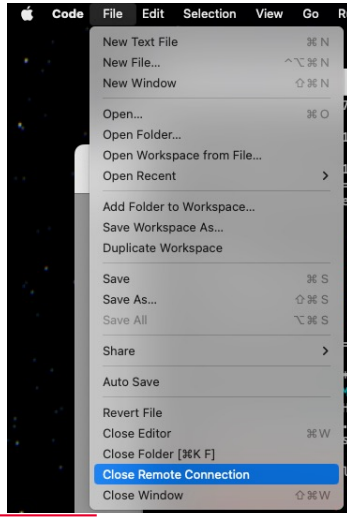
VSCode

- Download + Installation Instructions
  - https://code.visualstudio.com/docs/remote/ssh
- Basic Usage
  - Press F1, and search for “Remote-SSH: Connect to Host”
  - Select + Add New SSH Host…
  - Enter NetID@r1.stat.tamu.edu
  - Enter your password at top when prompted

Remote Editors: VSCode

- On First log in you may be prompted to trust the authors
- In the top left select the File Explorer, and then Open Folder
  - Choose your home directory (or other folder) and click OK.
  - This may prompt for your password at the top.
- You can edit files, utilizing any/most installed extensions
- When finished, save your files and click File > Close Remote Connection





File Transfer (scp)

TEXAS A&M UNIVERSITY

SCP – Secure Copy (https://linux.die.net/man/1/scp)

- scp [OPTION] [[user@]SRC_HOST:]file1 [[user@]DEST_HOST:]file2

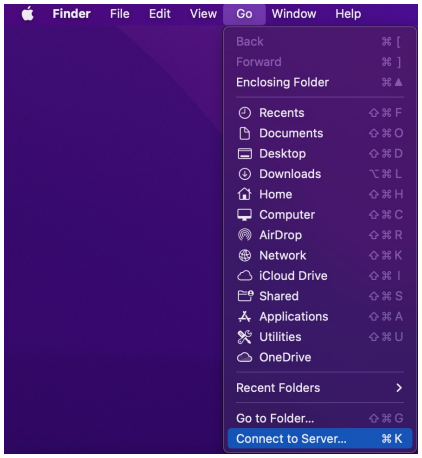
- Copy a single file from one system to another, for example: Copy local_file.txt to r1 and change its name on the remote system to remote_file.txt.

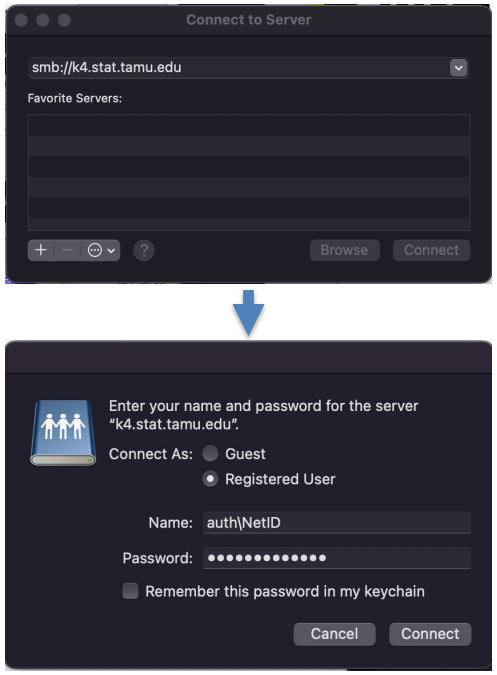
- To copy a directory, you must use the recursive option (-r).

```
$ scp local_file.txt NetID@r1.stat.tamu.edu:test/remote_file.txt
[…]
NetID@r1.stat.tamu.edu's password:
scp my_local_file.txt                    100%    0     0.0KB/s   00:00

$ ssh NetID@r1.stat.tamu.edu
[…]

[NetID@r1:~/test]$ ls

b.txt    c    remote_file.txt      subdir
```

File Transfer (macOS)

Home directories are shared from k4.stat.tamu.edu using Samba.





In macOS:

1. Select Go > Connect to Server…
2. Enter the following in the dialog: smb://k4.stat.tamu.edu
3. Click Connect
4. Select Registered User, and in the Name field enter "auth\NetID".
5. Enter your NetID password in the password field.
6. Click Connect

This will mount your Linux home directory as a folder in the Finder. You can then drag/drop files as you normally would in the Finder. Alternatively, applications like Cyberduck may be used.

https://support.apple.com/guide/mac-help/connect-mac-shared-computers-servers-mchlp1140/mac

File Transfer (Windows)

For Windows users there are several applications that may be used to simplify file transfer:

- MobaXterm (https://mobaxterm.mobatek.net/documentation.html)
  - MobaXterm includes a graphical SSH-browser: when you log to a remote server using SSH, a graphical SSH-browser pops up on in the left sidebar allowing you to drag and drop files directly from or to the remote server.
  - In the SSH side browser, double-click on a remote file to edit it directly. Changes are automatically saved to the remote server.
- WinSCP (https://winscp.net/eng/docs/start)

Use one of the above and connect to r1.stat.tamu.edu.

Screen Sessions

Screen allows you to disconnect from a server while all your processes continue to run (after detaching from the screen session).

- To invoke a screen session: [NetID@z01:~]$ screen
  - You can name a screen session using (-S), e.g. [NetID@z01:~]$ screen -S myApp
- After a screen session starts, simply run your processes/jobs as normal
- Detach from the screen session (leaving your processes running) by pressing Ctrl-a followed by d.
- Reattach to a screen session with [NetID@z01:~]$ screen -r
  or [NetID@z01:~]$ screen -r myApp if you have multiple sessions running.
- List your running screen session with [NetID@z01:~]$ screen -ls
- Exit a screen (stop the session entirely) using the command [NetID@z01:~]$ exit

- Don't forget to clean up your screen sessions!

Software

- gcc/gfortran – versions 9.3.1, 10.2.1 and 11.2.1 are available. The system default is version 4.8.5. In order to use a later version, enable the corresponding toolset e.g.:
  - source /opt/rh/devtoolset-9/enable # for version 9.3.1
  - source /opt/rh/devtoolset-10/enable # for version 10.2.1
  - source /opt/rh/devtoolset-11/enable # for version 11.2.1

  After this you should get a more appropriate version of gcc (e.g. gcc --version). Note that this change will only be for the current session. If you wish for this to be the default then you will need to add one of the above source lines to your .bashrc file.
- Julia
- LaTeX – Basic installation available, but not a recommended use of a compute cluster.
- Matlab
- Python – Users may install Anaconda (https://www.anaconda.com) locally for Jupyter, etc.
- R – versions 3.6.3, 4.0.5, 4.1.3, 4.2.0 (default) and 4.3.0 are available. To use a specific version type R-4.0.5 (i.e. R, a dash, and then the full version number).
  - Packages are installed locally by users (stored in ~/R).

Installing Software

TEXAS A&M UNIVERSITY

Users may install software in their home directory, e.g. Anaconda, R packages, etc.

Compiling/Installing should only be done on r1, otherwise the application may not work on all compute nodes.

Running installed software should only be done on a compute node.

Running Jobs

TEXAS A&M UNIVERSITY

Running installed software should only be done on a compute node. That is all executions of your code/applications/simulations should be performed on a compute node (z01, …, z27), and not the login node (r1). Jobs found on r1 will be terminated without warning.

Use the load command to identify an unused node. Nodes that have 0.0 in the load column are available for your application.

SSH to an unused node and run your jobs (don't forget to use screen to prevent processes from being terminated after disconnecting).

System		Uptime	Load	CPU	Mem
z01	up	24+04	0.0	28	64G
z02	up	24+04	110.1	28	64G
z03	up	24+04	0.0	28	64G
z04	up	24+04	10.0	28	64G
z05	up	24+04	60.1	28	64G
z06	up	24+04	0.0	28	64G
z07	up	24+04	0.0	28	64G
z08	up	24+04	0.0	28	64G
z09	up	24+04	0.0	28	64G
z10	up	24+04	0.0	28	64G
z11	up	24+04	1.0	28	64G
z12	up	24+04	0.0	28	64G
z13	up	24+04	0.0	24	128G
z14	up	24+04	0.0	24	128G
z15	up	24+04	0.0	24	128G
z16	up	24+04	0.0	24	128G
z17	up	24+04	0.0	28	112G
z18	up	24+04	52.6	28	96G
z19	up	24+04	52.9	28	96G
z20	up	24+04	0.0	28	96G
z21	down	--	--	28	96G
z22	up	24+04	6.0	16	128G
z23	up	24+04	0.0	16	128G
z24	up	24+04	0.0	16	128G
z25	up	24+04	0.0	16	128G
z26	up	24+04	0.0	16	128G
z27	up	24+04	0.0	16	128G
Total:			292.7	668	

Bash Files & Redirection

TEXAS A&M UNIVERSITY

I/O File Descriptors

- stdin (0) Standard Input, e.g. input from the keyboard
- stdout (1) Standard Output, e.g. the console, where a program writes messages to the user
- stderr (2) Standard Error, e.g. the console, where a program writes error messages

Linux Special Files

- /dev/null Accepts and discards all input
- /dev/zero Accepts and discards all input, and produces zeros (null characters) as output when read from
- /dev/random Produces random bytes when read from

Console Redirection

- > Redirects output (stdout, but not stderr)
- < Get input from this source (e.g. file)
- 2> Redirects file descriptor 2 (stderr)
- &> Redirects all output (stdout and stderr)
- | (pipe) Pass output as input to next command

```
[NetID@r1:~/test]$  ls | sort > list.txt
# Pass the output of ls as the input to sort, and
# redirect the output of sort to the file list.txt

[NetID@r1:~/test]$ cat list.txt
b.txt
c
list.txt
remote_file.txt
subdir

[NetID@r1:~/test]$ grep -r hello / 2> /dev/null
# Recursively search for the string hello in files,
# starting from the root directory (/), and
# redirect all errors to /dev/null
[output omitted]
```

Bash Scripting

TEXAS A&M UNIVERSITY

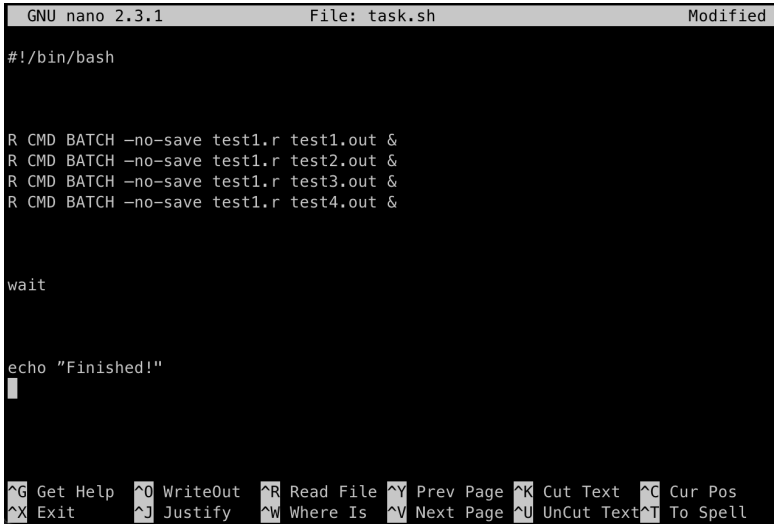
Instead of manually entering commands on the command line, we can use scripts.
Scripts are really text files containing executable code, and typically named using the extension “sh”, e.g. task.sh.
Scripts should be given execute permission (chmod u+x task.sh), so they can be run from the command line directly.

The first line should be a “shebang” line (https://en.wikipedia.org/wiki/Shebang_(Unix))

The body of the script should perform some task(s). In this example we are running four instances of R (non-interactively) in the background (note the &), with input from test1.r and the output of each going to a different file (test1.out, test2.out, …)

The script then waits for all background processes to complete.

And finally, the script echo’s (writes) “Finished!” to the console.



Be careful not to overwrite output (consider what happens when you omit testX.out from one or more lines above).
You can also use GNU Parallel to help run many simulations.

Bash Scripting Tutorial (https://linuxconfig.org/bash-scripting-tutorial-for-beginners)
GNU Parallel (https://www.gnu.org/software/parallel/parallel_tutorial.html)

R

TEXAS A&M UNIVERSITY

Interactive Session:

- [NetID@z01:~]$ R # This opens the R interpreter

Non-Interactive Session

- To run a non-interactive session as follows you must provide one of the following options:
  - --save Data sets are saved at the end of the session
  - --no-save Data sets are not saved at the end of the session
  - --vanilla Equvalient to –no-save, –no-environ, –no-site-file, –no-init-file and –no-restore
- Following the options redirect (<) stdin to come from your R script (e.g. test.R).
- Optionally, redirect stdout and/or stderr to a file (e.g. &> test.R.out)
- [NetID@z01:~]$ R --no-save < test.R &> test.R.out

Batch Execution of R

- Similar to the non-interactive session, except ‘proc.time()’ will be executed after the input script.
- [NetID@z01:~]$ R CMD BATCH --no-save test.R test.R.out

An Intro to R: Appendix B - Invoking R (https://colinfay.me/intro-to-r/appendix-b-invoking-r.html)
Batch Execution of R (https://stat.ethz.ch/R-manual/R-devel/library/utils/html/BATCH.html)

Matlab

TEXAS A&M UNIVERSITY

Interactive Session:

- [NetID@z01:~]$ matlab -nodesktop # This opens the Matlab in the console

Non-Interactive Session

- Redirect (<) stdin to come from your M script (e.g. test.m).
- Optionally, redirect stdout and/or stderr to a file (e.g. &> test.m.out)
- [NetID@z01:~]$ matlab –nodesktop < test.m &> test.m.out

GUI Session

- Follow the X forwarding instructions (using 'ssh -X' to r1 and then again to a compute node ssh –X z01). This should open the Matlab interface on your local machine.
- [NetID@z01:~]$ matlab

Matlab Documentaiton(https://www.mathworks.com/help/matlab/)

Matlab GUI: Errors

TEXAS A&M UNIVERSITY

For libGL errors:

- Execute the following line prior to running Matlab or add it to your .bashrc

  export LIBGL_ALWAYS_INDIRECT=1

For graphics errors (e.g. display flicker/flashing):

- Create the file, java.opts, in your home directory
- Add the following lines to that file:

  -Dsun.java2d.xrender=false

  -Dsun.java2d.pmoffscreen=false

Matlab Documentaiton(https://www.mathworks.com/help/matlab/)

Compiling C/Fortran

Refer to the Software slide for gcc/gfortran, and select an appropriate version of the compilers, for example:

```
source /opt/rh/devtoolset-11/enable  # for version 11.2.1
```

Using optimization flags will produce much faster code, for example:

```
gfortran -O3 –o pgm file1.f file2.f … -lm
```

The compute nodes do not all support the same microarchitectures, which is why for portability across compute nodes all compilation/installation should be performed on r1. If you wish to see the supported instruction sets on each node `cat /proc/cpuinfo`

Advanced SSH

TEXAS A&M UNIVERSITY

For those of you that install Anaconda locally and wish to run a Jupyter Notebook server on one of the compute nodes, do the following:

1. Choose a compute node by running the **load** command. For this example, we will use z02.
2. From your local system, run:

   ssh -L 42221:localhost:42221 NetID@r1.stat.tamu.edu -t ssh -L 42221:localhost:42221 NetID@z02.stat.tamu.edu

   This will require you to enter your password twice, once for r1 and a second time for z02. The port numbers above, (e.g. 42221) can be any numbers greater than 1024, but to avoid potential conflicts the user should choose values like these. If different ports are chosen for each forward be sure the middle two port numbers match.

3. From z02.stat.tamu.edu (or which ever z node was chosen) start a Jupyter server and specify the port to be the **last** port from the above command line (e.g. 42221). Depending on their setup (in their bash profile) they may be required to first run 'conda activate' or 'source ~/anaconda3/bin/activate'.

   jupyter notebook --port 42221

4. On your local system, open a browser and navigate to http://127.0.0.1:42221/?token=... where the port 42221 is the **first** port specified in the forwarding command in step (1), and the token value for the URL comes from the output of the Jupyter notebook server, for example:

```
(base) [NetID@z02:~]$ jupyter notebook --port 42221
  ...
To access the notebook, open this file in a browser:
        file:///home/NetID/.local/share/jupyter/runtime/nbserver-41099-open.html
Or copy and paste one of these URLs:
        http://z02.stat.tamu.edu:42221/?token=77a91bd48eea3fefdde943d1fdb2d63c6946b810fe863662     or
        http://127.0.0.1:42221/?token=77a91bd48eea3fefdde943d1fdb2d63c6946b810fe863662
```

Best Practices (Opinionated)

TEXAS A&M UNIVERSITY

- Reproducibility
  - Be organized
  - Documentation!
  - Parameterize your code - NO hard-coded values
  - Keep config/parameters with results
  - Include version control info with your results
    - e.g. GitHub commit hash
- Version Control
  - Commit often and document changes
  - Use GitHub for all code and configurations
    - Collection of tutorials: https://www.atlassian.com/git/tutorials
    - Branching tutorial: https://learngitbranching.js.org/
    - GitHub Skills: https://skills.github.com/
    - GitHub Actions: https://github.com/features/actions

Best Practices (Opinionated)

TEXAS A&M UNIVERSITY

- Checkpoints, checkpoints, checkpoints
  - [Wikipedia] Checkpointing is a technique that provides fault tolerance for computing systems. It basically consists of saving a snapshot of the application's state, so that applications can restart from that point in case of failure. This is particularly important for long running applications that are executed in failure-prone computing systems.
  - Created manually, or tools exist that can help (e.g. https://cran.r-project.org/web/packages/checkpoint/vignettes/checkpoint.html)

- Write, review and periodically run testing (automate whenever possible)
  - Unit tests – low level testing of functions, methods, classes, components, etc. does each piece satisfy pre/post conditions, etc.
  - Integration tests – do the modules/components/etc. work together, interaction testing
  - Functional tests – does the application produce the expected output
  - End-to-end tests – does the application work as expected in its intended environment
  - Acceptance testing – does the application satisfy the customer/PI/etc. requirements
  - Performance testing – how does the application perform under various conditions/workloads
  - Smoke testing – basic/quick checks that show the application is working

HPRC

TEXAS A&M UNIVERSITY

Apply for or Renew a High Performance Research Computing Account:

- https://hprc.tamu.edu/apply/

Training

- New User Info: https://hprc.tamu.edu/user_services/new_user_information.html
- Short Courses: https://hprc.tamu.edu/training/
- Online: https://hprc.tamu.edu/training/online.html
- Primers: https://hprc.tamu.edu/training/primers_popup.html
- Workshops: https://hprc.tamu.edu/events/workshops/

Available Resources

- https://hprc.tamu.edu/resources/

Questions?

artsci-help@tamu.edu