



# New User Training

Getting Started on HPCC Resources

Eric Rees, Ph.D.


*High Performance Computing Center*



# HPCC User Training Agenda

- Introduction to Parallel Computing
- HPCC Resources
- Logging In and Using the Clusters
- Transferring Data
- Installing Software
- HPCC Policies
- Getting Help

# Introduction to Parallel Computing



# What is Parallel Computing?

- Software is often written and perceived to run **serially**.
  - Execution occurs on a single computer using a single CPU core.
  - A problem is broken down into a discrete series of instructions.
  - Instructions are executed one after another.
  - Only one instruction may execute at any given time.

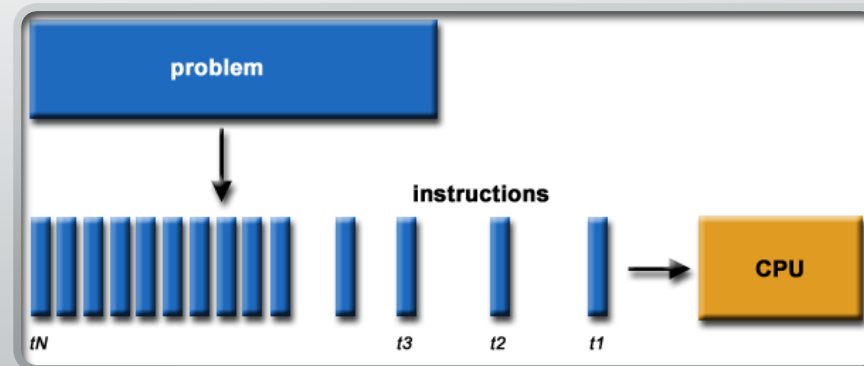


Image provided by Lawrence Livermore National Labs (computing.llnl.gov)

# What is Parallel Computing?

- **Parallel Computing** is the simultaneous use of multiple compute resources to solve a computational problem.
  - Execution occurs across multiple CPU cores.
  - A problem is broken into discrete parts that can be solved concurrently.
  - Each part is further broken down into a series of instruction, executed one after another.
  - Instructions from each part execute simultaneously on different CPU cores.

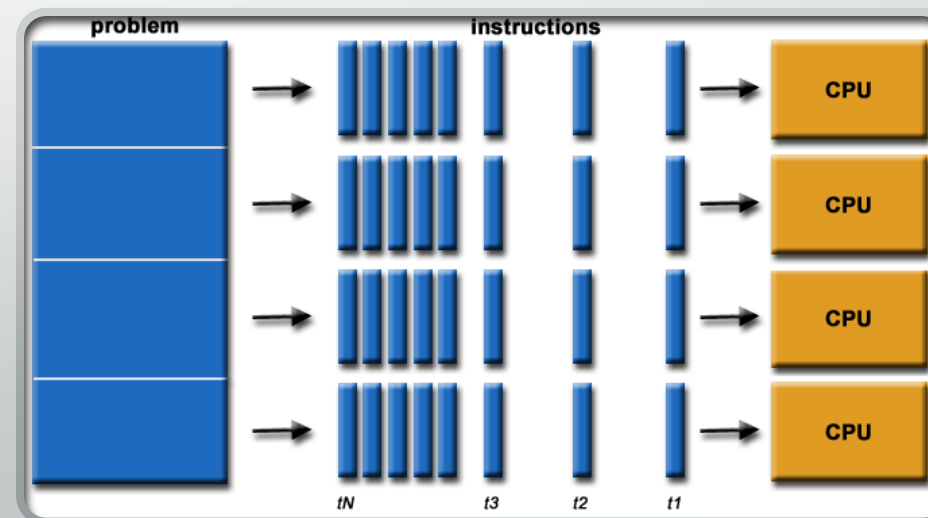


Image provided by Lawrence Livermore National Labs (computing.llnl.gov)

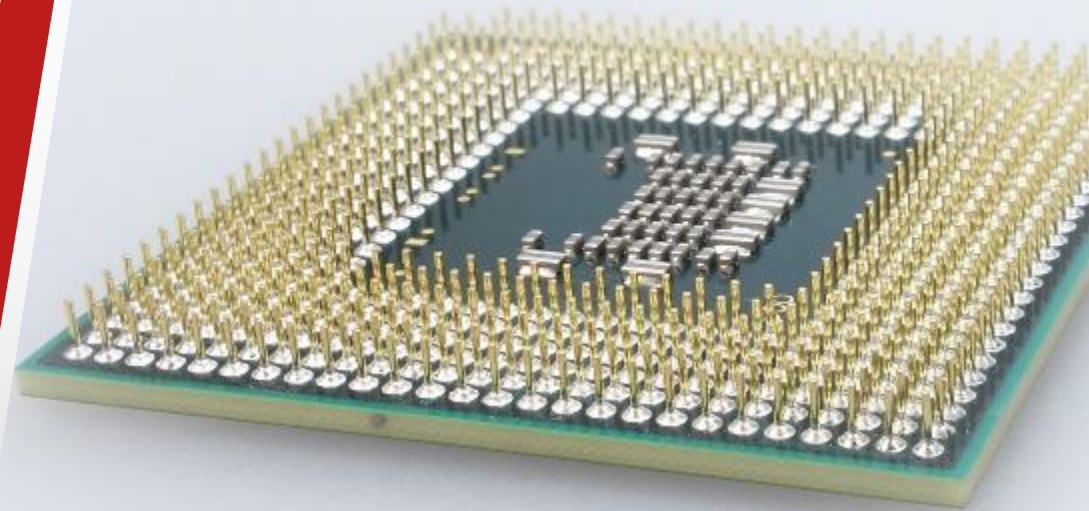
# Classes of Parallel Computers

- **Multi-core Computing**

- Multi-core processors contain multiple 'processing units' (called cores) on a single chip.
- Allows for parallel execution across cores – each able to reach the same system resources (RAM, Keyboard, Monitor, etc...)

- **Symmetric Multiprocessor (SMP)**

- A symmetric multiprocessor is a computer system with multiple identical processors.
- Each processor likely has multiple cores.
- Allows for parallel execution across cores – each able to reach the same system resources (RAM, Keyboard, Monitor, etc...)



# Classes of Parallel Computers

- **Clusters**

- A group of loosely coupled computers working together closely.
- Processes can be spread across multiple nodes but processes are unable to reach the same system resources (RAM, Keyboard, Monitor, etc...)

- **Massively Parallel Processors (MPP)**

- A group of tightly coupled computers working together closely across a specialized high-speed interconnect.
- Processes can be spread across multiple nodes but processes are unable to reach the same system resources (RAM, Keyboard, Monitor, etc...)



# Classes of Parallel Computers

- **Grid Computing**
  - Highly distributed form of parallel computing.
  - Clusters or single resources are spread across multiple sites using the Internet for connectivity.
  - Plagued by high latency issues.
- **Cloud Computing**
  - Same as Grids but often run by for-profit entities.
  - Sites are often spread over large geographic areas but most processing occurs within a single site – to help alleviate latency issues.





Why does it matter?

- Exposes 3 kinds of programming models:
  - **Serial programming**
    - Executes serially using a single core / thread
    - *Single core machines*
  - **Multi-core / Multi-threaded Programming**
    - Executes in parallel using multiple cores / threads
    - All threads are running on the same machine and access the same RAM
    - *Multicore & Symmetric Multiprocessing*
  - **Massively Parallel Programming**
    - Executes in parallel using multiple machines
    - *Clusters, Massive Parallel Processors, & Grid/Cloud*

# Parting Words

- If your program is not written to use a certain model, it will not “*just work*” in that model.
  - Running serial code on a 36-core machine will use 1 core and leave 35 cores sitting idle.
  - Attempting to run multi-threaded code across 10 nodes will result in 1 node being overutilized and 9 nodes sitting idle.
- Not all multi-threading/MPP is equal!
  - Try to understand how your program works at a small scale before attempting to “*scale up*”.
  - Keep in mind that programming language, developer decisions and even user input data can greatly alter how well an application scales.



# HPCC Resources



# Current Resources



## Clusters

Quanah Cluster  
Ivy Cluster  
Community Clusters



## Parallel File System

Lustre File System  
6.9 PB in total space

# Ivy Cluster

- Commissioned in 2014
- Running CentOS 7.4
- Currently consists of
  - 100 Nodes
  - 2000 Cores (20 cores/node)
  - 6.25 TB Total RAM (64 GB/node)
  - Xeon E5-2670v2 **Ivy** Bridge Processors
  - QDR 40 GB/second InfiniBand fabric



# Quannah Cluster

- Commissioned in 2017
- Running CentOS 7.4
- Currently consists of
  - 467 Nodes
  - 16,812 Cores (36 cores/node)
  - 87.56 TB Total RAM (192 GB/node)
  - Xeon E5-2695v4 Broadwell Processors
  - Omni-Path (100 Gbps) Fabric
- Benchmarked at 485 Teraflops/sec



# Red Raider Cluster

- Arrives Q1 2020
- Commission planned for mid 2020
- Will consist of
  - 240 CPU Nodes
    - 30,720 Cores (128 cores / node)
    - 120 TB Total RAM (512 GB / node)
    - AMD Rome Processors
  - 20 GPU Nodes
    - 40 NVIDIA Tesla V100 GPUs (2 V100 / node)
    - 10 TB Total RAM (512 GB / node)
  - HDR 200 GB/second Infiniband fabric



# Lustre Storage System

- High speed parallel file system
- 6.9 PB of storage space
- Users may purchase dedicated storage space:
  - With Backup – \$80 /TB / year
  - Without Backup– \$40 /TB / year





# Hrothgar West Cluster (Decommissioned)

- Commissioned in 2011
- Decommissioned in November 2019
- Consisted of
  - 563 Nodes
  - 6,756 Cores (12 cores/node)
  - 13.19 TB Total RAM (24 GB/node)
  - Xeon X5660 Westmere Processors
  - DDR 20 GB/second InfiniBand fabric



# Lustre File System

- You have access to three storage areas
  - Home - /home/<eraider>
    - Quota: 300GB
    - Backed up nightly
    - Never purged
  - Work - /lustre/work/<eraider>
    - Quota: 700GB
    - Never backed up nor purged
  - Scratch - /lustre/scratch/<eraider>
    - Quota: None
    - Never backed up
    - Purged

## Quota / Backup / Purge per Lustre Area

Area	Quota	Backup	Purged
/home/<eraider>	300 GB	Yes	No
/lustre/work/<eraider>	700 GB	No	No
/lustre/scratch/<eraider>	None	No	Monthly

## Example Quota Results

```
Current Storage Usage for errees:  
    /home - Currently using 19 of 300 GB ( 6%).  
    /lustre/work - Currently using 2 of 700 GB ( 0%).
```

# Logging In and Using the Clusters





# Getting Started


- Account Request
  - Faculty/Staff account
  - Student account
  - Research Partner account
  - <http://www.depts.ttu.edu/hpcc/accounts/index.php>
- User Guides
  - The best place to get detailed information regarding using the clusters
  - <http://www.depts.ttu.edu/hpcc/userguides/index.php>
- Queue Status
  - Current view of the pending and running jobs on each cluster
  - <http://charlie.hpcc.ttu.edu/qstat/qstat.html>



# Getting Started

- Logging into HPCC Resources
  - User Guide: <http://tinyurl.com/ttu-hpcc-login>
- On or Off Campus?
  - On Campus: Wired TTU network & TTUnet wireless network
  - Off Campus: Any other network connection, including:
    - TTUHSC networks
    - TTUguest wireless network
- Logging in from Off Campus
  - Log in via the SSH gateway – **Not owned/maintained by HPCC**
  - Establish a VPN - <https://goo.gl/4LbuWG> - **Preferred Method**





# Environment Settings

- User Environments
  - .bashrc
    - Bash script that runs whenever you start an interactive login.
    - Often used to set up user environments.
    - While you may add to your .bashrc file, **do not remove any default settings!**
  - Modules
    - The primary way to change your user environment.
    - Running on Quanah and Ivy

User Guide:

[http://www.depts.ttu.edu/hpcc/userguides/general\\_guides/software\\_environment.php](http://www.depts.ttu.edu/hpcc/userguides/general_guides/software_environment.php)

# Modules

- Modules commands:
  - module avail
  - module list
  - module load <module\_name>
  - module unload <module\_name>
  - module spider <keyword>
  - module purge

```
quanah:$ module avail
----- /opt/apps/nfs/module/modulefiles -----
gnu/5.4.0   gurobi/v751   intel/17.3.191   java/1.8.121   matlab/R2017b   perl/5.16.3

Use "module spider" to find all possible modules.
Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".

quanah:$ module spider python

python:
-----
Description:
  Python-2.7.9 compiled with Intel

Versions:
  python/2.7.9-gnu
  python/2.7.9-intel

-----
For detailed information about a specific "python" module (including how to load the modules) use the
name.
For example:

$ module spider python/2.7.9-intel
-----
```



## Modules Tips and Recommendations

- Try to keep all of your module load commands as part of your job submission scripts.
  - Makes debugging and changing between experiments easier.
  - Prevents collisions or accidentally running jobs with the wrong environment.
  - Provides yourself and collaborators with a way of tracking the exact software and versions used in any given experiment.
- Always contain the version number of a module in the module load command.
  - Makes version tracking easier.
  - Prevents unanticipated changes in version during an experiment.
  - This will make it easier for you to track which experiments and jobs used which versions of different software, which will in turn make writing your research paper's "methods" section easier.
  - Example: Use **module load nwchem/6.6-intel** instead of just **module load nwchem**

# Submission script layout

## User Guide:

[http://www.depts.ttu.edu/hpcc/userguides/general\\_guides/job\\_submission.php](http://www.depts.ttu.edu/hpcc/userguides/general_guides/job_submission.php)

- V** instructs the scheduler to keep the current environment settings
- cwd** instructs the scheduler to start the commands from the current directory
- S /bin/bash** instructs the scheduler to use the /bin/bash as the shell for the batch session
- N <job name>** sets the name for the job. Can be referenced in the script using \$JOB\_NAME
- o \$JOB\_NAME.o\$JOB\_ID** indicates the name of the standard output file.
- e \$JOB\_NAME.e\$JOB\_ID** indicates the name of the standard error file.
- q <queue name>** instructs the scheduler to use the queue defined by <queue name>.
- pe <parallel environment>** instructs the scheduler to use the parallel environment defined by <parallel environment>.
- l h\_vmem = <float>G** instructs the scheduler to reserve <float> gb of memory per slot
- l h\_rt = HH:MM:SS** instructs the scheduler to set the maximum job time to HH:MM:SS
- P <project name>** instructs the scheduler to use the project defined by <project name>.

# Parallel Environments

## Quanah

- Omni queue
  - -pe mpi
    - Must request in increments of 36.
    - All MPI jobs should use this PE.
  - -pe sm
    - Can request between 1 and 36.
    - Slots are guaranteed to be on one node.

## Ivy

- Ivy queue
  - -pe ivy
    - Must request in increments of 20.
- Community Clusters
  - -pe mpi
    - Can request any number of job slots.
  - -pe fill
    - Can request any number of job slots.
  - -pe sm
    - Can request between 1 and max number of cores per node.

# Requesting Memory

- QuanaH limits memory usage per slot (essentially per core).
- By default, your processes will be limited to ~5.3 GB per slot.
  - Each node has 192GB of memory and 36 slots:  
 $192 \div 36 = 5.333 \dots$
- You can increase or decrease amount using the `-l h_vmem=#G` option.
- Examples:

```
#!/bin/sh
#$ -V
#$ -cwd
#$ -S /bin/bash
#$ -N MPI_Test_Job
#$ -o $JOB_NAME.o$JOB_ID
#$ -e $JOB_NAME.e$JOB_ID
#$ -q omni
#$ -pe mpi 36
## -l h_vmem=5.3G
#$ -l h_rt=48:00:00
#$ -P quanaH
```

Uses 190.8 GB

```
#!/bin/sh
#$ -V
#$ -cwd
#$ -S /bin/bash
#$ -N MPI_Test_Job
#$ -o $JOB_NAME.o$JOB_ID
#$ -e $JOB_NAME.e$JOB_ID
#$ -q omni
#$ -pe mpi 36
## -l h_vmem=192G
#$ -l h_rt=48:00:00
#$ -P quanaH
```

Uses 6.75 TB



# Requesting Runtime Limits

- Recommended that you set the max runtime you expect a job will take.
- The scheduler behaves more efficiently as more jobs give it accurate runtime, slot and memory requirements.
  - Setting an accurate runtime allows your job to possibly be scheduled earlier than it would otherwise.
- You can set the max runtime for a job using `-l h_rt=HH:MM:SS`

- Examples:

```
#!/bin/sh
#$ -V
#$ -cwd
#$ -S /bin/bash
#$ -N MPI_Test_Job
#$ -o $JOB_NAME.o$JOB_ID
#$ -e $JOB_NAME.e$JOB_ID
#$ -q omni
#$ -pe mpi 36
#$ -l h_vmem=5.3G
## -l h_rt=48:00:00
#$ -P quanah
```

Max Runtime: 48 hours

```
#!/bin/sh
#$ -V
#$ -cwd
#$ -S /bin/bash
#$ -N MPI_Test_Job
#$ -o $JOB_NAME.o$JOB_ID
#$ -e $JOB_NAME.e$JOB_ID
#$ -q omni
#$ -pe mpi 36
#$ -l h_vmem=5.3G
## -l h_rt=3:00:00
#$ -P quanah
```

Max Runtime: 3 hours

# Projects



## What is a project?

Construct in Univa Grid Engine that provides a means to organize joint computational tasks from multiple users.



## What does it do?

Defines resource usage policies for all jobs that belong to a project.

# Projects on Quanah

## quanah

- Maximum amount of running cores: 16,812
- Default run time: 48 hours
- Maximum run time: 48 hours
- Allowed Parallel Environments: 'sm' and 'mpi'

## xlquanah

- Maximum amount of running cores: 144
- Default run time: 72 hours
- Maximum run time: 120 hours
- Allowed Parallel Environments: 'sm'

# Projects on Ivy

## ivy

- Available only on the 'ivy' queues.
- Maximum amount of running cores: 2,000
- Default run time: 48 hours
- Maximum run time: 48 hours
- Allowed Parallel Environments: 'ivy'

## communitycluster

- Available only on the community cluster queues:
  - Chewie, R2D2, Yoda, ancellcc, blawzcc, caocc, dahlcc, phillipsc, tang256cc, & tangcc



# Submitting Jobs

- User Guide: [http://www.depts.ttu.edu/hpcc/userguides/general\\_guides/job\\_submission.php](http://www.depts.ttu.edu/hpcc/userguides/general_guides/job_submission.php)
- Example Script
  - Set the name of our job to “MPI\_TEST\_JOB”
  - Requests to run on the Quanah cluster using the omni queue.
  - Sets the parallel environment to “mpi” and requests 36 cores.

```
#!/bin/sh
#$ -V
#$ -cwd
#$ -S /bin/bash
#$ -N MPI_Test_Job
#$ -o $JOB_NAME.o$JOB_ID
#$ -e $JOB_NAME.e$JOB_ID
#$ -q omni
#$ -pe mpi 36
#$ -l h_vmem=5.3G
#$ -l h_rt=48:00:00
#$ -P quanah

module load intel impi

mpirun --machinefile machinefile.$JOB_ID -np $NSLOTS ./mpi_hello_world
```

# Submitting Jobs

- Submit job
  - `qsub <job submission script>`
- Check job status
  - Command: **qstat**
  - job-ID, prior, name, user, **state**, submit/start at, queue, jclass, slots, ja-task-ID
    - "r": running
    - "qw": waiting in the queue
    - "E": error

```
quanah:/mpi_tutorial$ qsub mpi.sh
Your job 51516 ("MPI_Test_Job") has been submitted
quanah:/mpi_tutorial$ qstat
job-ID      prior    name         user      state submit/start at   queue                                jclass      slots ja-task-ID
-----
      51516  0.00000 MPI_Test_J   Test     qw    07/26/2017 15:41:16                                36
quanah:/mpi_tutorial$ qstat
job-ID      prior    name         user      state submit/start at   queue                                jclass      slots ja-task-ID
-----
      51516  8.79812 MPI_Test_J   Test     r     07/26/2017 15:41:16 omni@compute-20-4.localdomain        36
quanah:/mpi_tutorial$
```

# Interactive Login

- Start an interactive job using QLOGIN
  - `qlogin -P <cluster> -q <queue> -pe <pe> <#>`
  - Examples:
    - `qlogin -P quanah -q omni -pe sm 1`
    - `qlogin -P hrothgar -q ivy -pe ivy 20`
- While using QLOGIN
  - Make sure the prompt changes to `compute-#-#`
  - If it doesn't change, check your `qlogin` and try again.
  - Make sure to run "exit" when you are finished
  - Keep in mind: **Runtime limits apply to qlogin!**

```
quanah:$ qlogin -P quanah -q omni -pe sm 1
Your job 75025 ("QLOGIN") has been submitted
waiting for interactive job to be scheduled ...
Your interactive job 75025 has been successfully scheduled.
Establishing /export/uge/scripts/qlogin_wrapper session to host compute-8-24.localdomain ...
Last login: Tue Nov  7 11:06:15 2017
compute-8-24:$
```

# Current Cluster Usage

- Viewing the current queue status and running/pending jobs
  - For a queue overview, run the command: “qstat -g c”
  - Visit the queue status webpage: <http://charlie.hpcc.ttu.edu/qstat/qstat.html> (Updates every 2 minutes)

## Job Runtime Limits

Cluster	Queue	Project	Runtime Limit	# of Cores per node	Memory per node
Quanah	omni	quanah	48 hours	36 cores	192 GB
Quanah	omni	xlquanah	120 hours	36 cores	192 GB
Quanah	omni	hep / cbg	∞ hours	36 cores	192 GB
Ivy	ivy	hrothgar	48 hours	20 cores	64 GB
Ivy	community cluster queues (Chewie, R2D2, Yoda, ancellcc, blawzcc, caocc, dahlcc, phillipsc, tangcc, tang256cc)	communitycluster	∞ hours	vary	vary

# Debugging Failed Jobs

- **Job output**

- Standard: `$JOB_NAME.o$JOB_ID`
- Error: `$JOB_NAME.e$JOB_ID`

- **When debugging:**

1. Check the output files for errors
2. Check the output of `qacct -j <job_ID>`
  - failed
  - exit\_status
  - maxvmem
  - start\_time & end\_time (<runtime limit)
  - low
3. Contact Support: [hpccsupport@ttu.edu](mailto:hpccsupport@ttu.edu)

# X Windows

## Interactive GUI using Linux/Mac

- Install X Server (Xquartz)
- Add the following option to your normal ssh command:
  - -Y
- Example:
  - `ssh -Y eraider@quanah.hpcc.ttu.edu`
- Run a test command like `xclock`.

## Interactive GUI using Window

- Install MobaXterm
- Open a local terminal
- Log into the cluster using a standard ssh command:
  - `ssh eraider@quanah.hpcc.ttu.edu`
- Run a test command like `xclock`.

# Transferring Data

A decorative graphic in the bottom right corner of the slide, consisting of several overlapping, parallel lines. The lines are colored in a gradient from dark red to light grey, creating a sense of depth and movement. The lines are oriented diagonally, starting from the bottom left and extending towards the top right.

# Transferring Data

- Transfer files using Globus Connect
  - User Guide: <http://tinyurl.com/hpcc-data-transfer>
- Whenever possible, refrain from using:
  - scp,
  - sftp,
  - rsync,
  - Or any other data transfer tool.



# Transferring Data via Globus Connect

- Why use Globus?
  - Globus Connect service is well connected to the campus network.
  - The data transfer nodes are better positioned for transferring user data.
  - Globus connect service eliminates the data transfer load from the cluster login node.
  - Globus connect works with Linux, Mac and Windows and is controlled through a web GUI.
  - Numerous other sites (including TACC) support Globus Connect data transfers.



# Software Installations



# Local R Package Installations

- Install an R package into your home folder:
  - module load intel R
  - R
  - `install.packages('<package name>')`
    - Example: `install.packages('readr')`
  - Select a mirror
- The R application will ask if you want to install it locally the first time you do this.



# Local Python Package Installations

- Install a Python package into your home folder:
  - *module load intel python*
  - *pip install --user <package name>*
    - Example: *pip install --user matplotlib*
- Install a local copy of Python using Conda:
  - */lustre/work/examples/InstallPython.sh*
  - *. \$HOME/conda/etc/profile.d/conda.sh*
  - *conda activate*
  - *conda install <package name>*
    - Example: *conda install biopython*

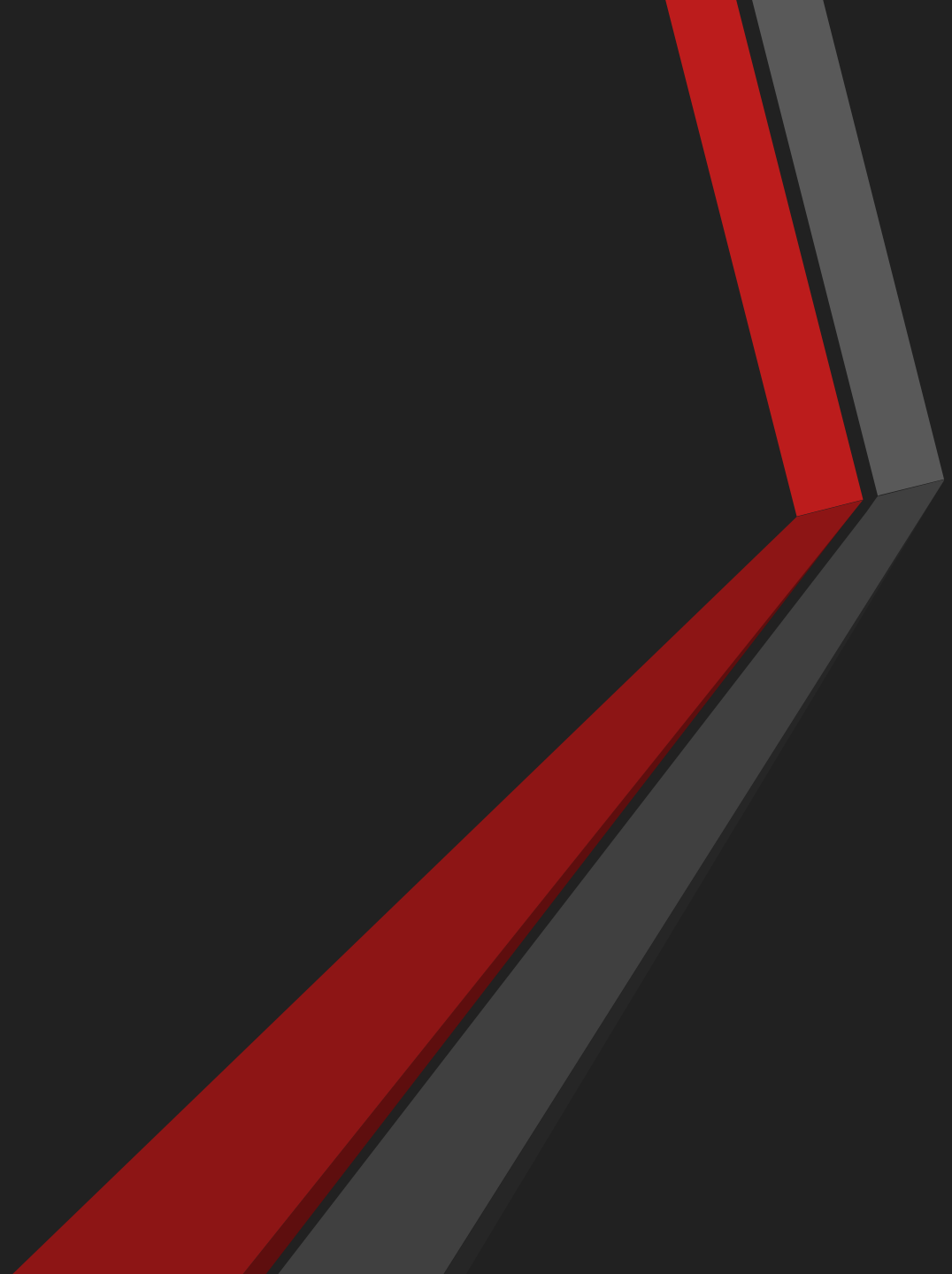


# Frustrated with compiling?

- For software you know doesn't need to be system wide but you want help getting installed
  - Contact us at [hpcsupport@ttu.edu](mailto:hpcsupport@ttu.edu)
- Need your software accessible to other users or believe the software would help the general community?
  - Submit a [software request!](#)
- **Software Installation Requests**
  - Software requests are handled on a case by case basis.
  - Requesting software does not guarantee it will be installed "cluster-wide".
  - May take two or more weeks to complete your request.



# HPCC Policies





# Usage Policies

- Login nodes (Quanah and Ivy)
  - No jobs are allowed to run on the login node.
- SSH Access
  - No direct SSH access allowed to the nodes.
- Software Installation Requests
  - Software requests are handled on a case by case basis.
  - Requesting software does not guarantee it will be installed “cluster-wide”.
  - Reminder: May take two or more weeks to complete your request.

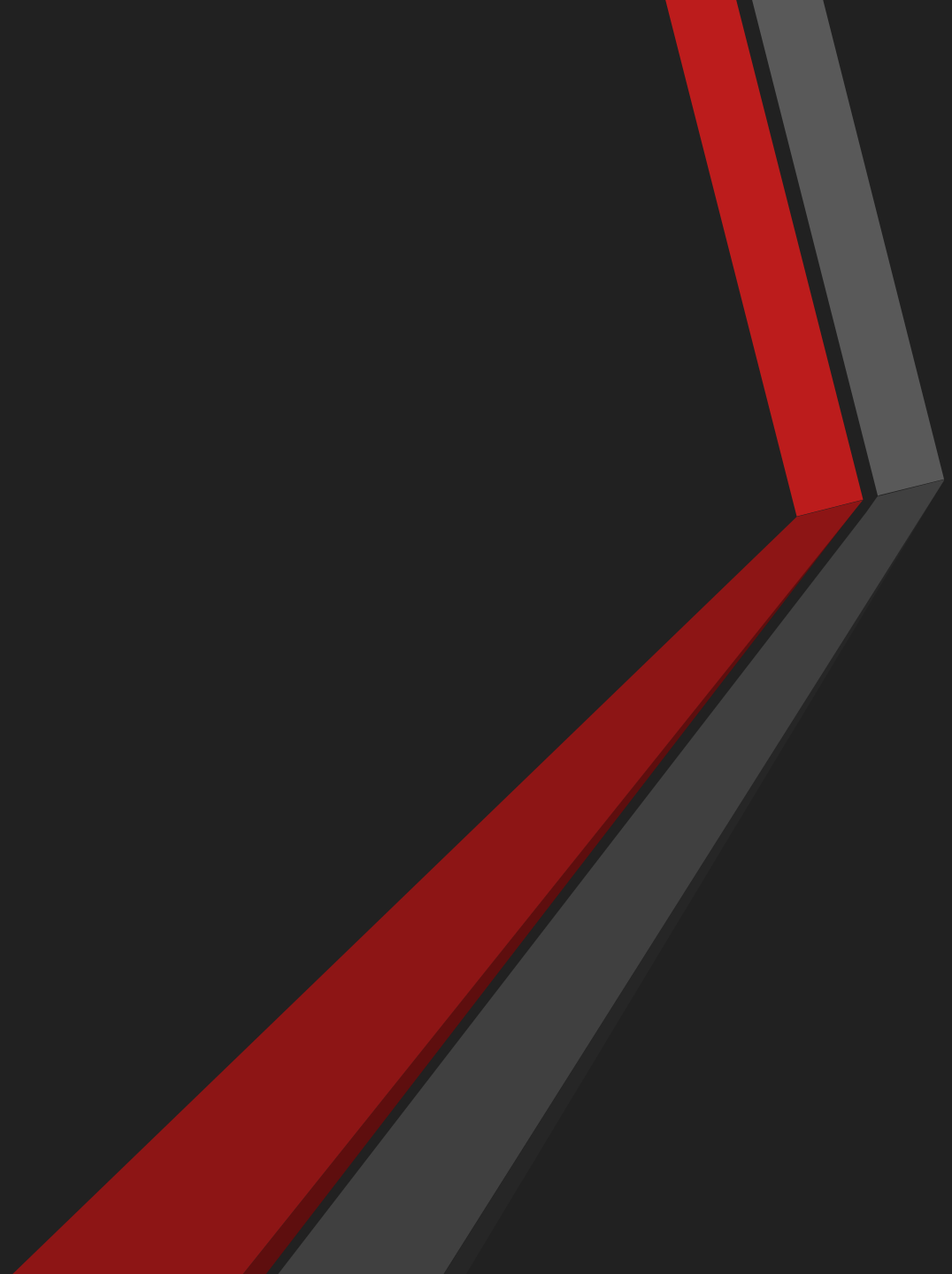
# Scratch Purge Policy

- Scratch will be purged monthly.
- Automatic removal of all files not accessed within the past year
- Purge will aim to drop scratch usage below 65%
  - This may remove files accessed in past year
- User **may not** run “touch” or similar commands for the purpose of circumventing this policy.
  - Doing so can result in a ban.





Getting Help



# Getting Help

- Visit our website - [www.hpcc.ttu.edu](http://www.hpcc.ttu.edu)
  - Most user guides have been updated
  - New user guides are being added
- Submit a support ticket
  - Send an email to [hpccsupport@ttu.edu](mailto:hpccsupport@ttu.edu)





TEXAS TECH UNIVERSITY

Information Technology Division™