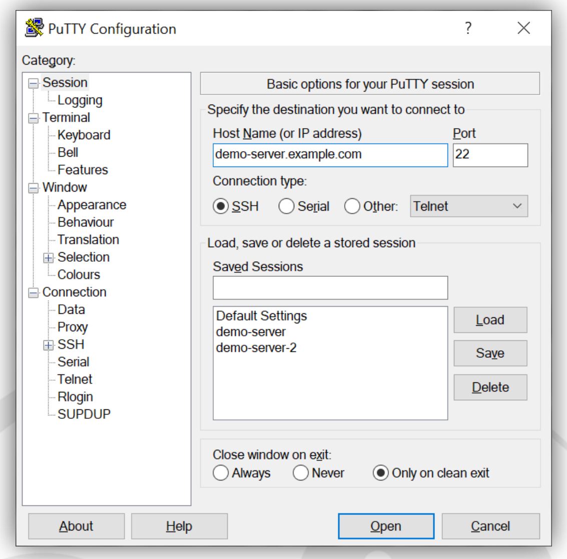
Intro to Scientific Computing



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): demo-server.example.com
Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

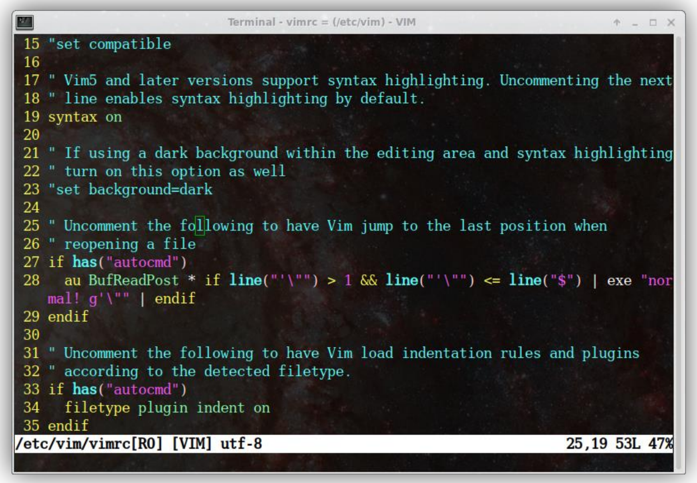
Load, save or delete a stored session

Saved Sessions

Default Settings
demo-server
demo-server-2

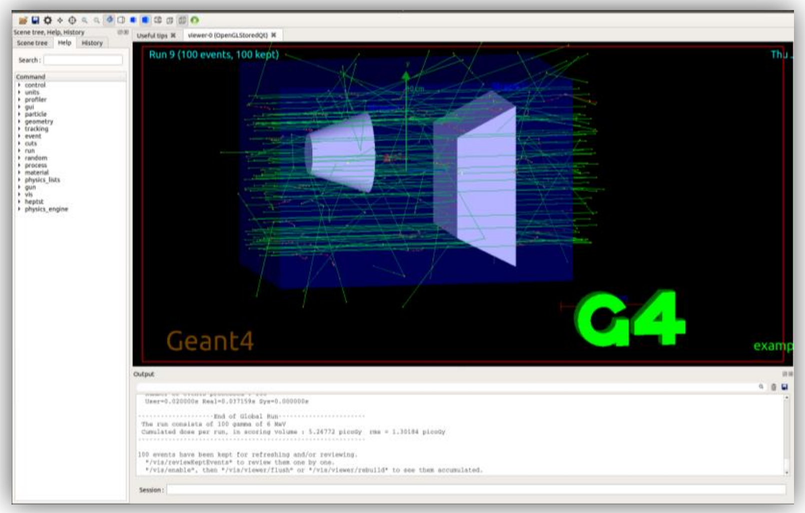
Close window on exit: Always Never [✓ Only on clean exit]

Access High Performance Computing (HPC) Cluster



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "nor
mal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                 25,19 53L 47%
```

Set up scientific computing software on Linux systems



```
 User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Run physics experiments, simulations, and data analysis

Cameron Clarke

Radiation Detector & Imaging Group, Nuclear Physics, Jefferson Lab

BRIC Biomedical Research & Innovation Center

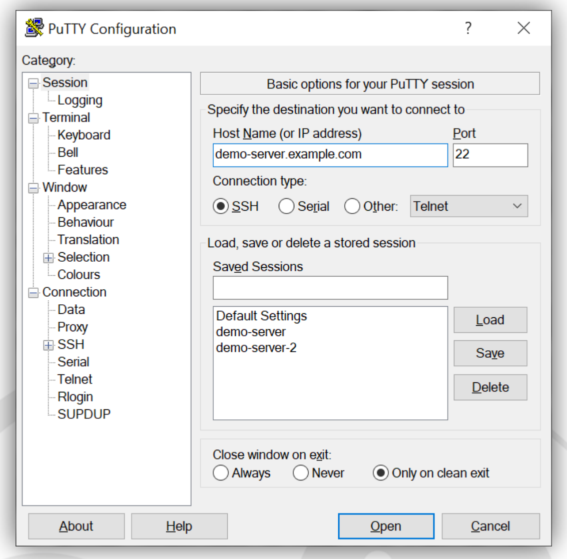
GETTING STARTED AT THE LAB

Slides cover:

- **Account activation and access**
- Accessing scientific computing resources (ifarm)
- Learning to navigate and use Linux computers
- Introduction to scientific computing resources
  (Geant4 simulation and ROOT data analysis)

Friday, August 9, 2024

ACQUIRING AN ACCOUNT



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): demo-server.example.com

Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

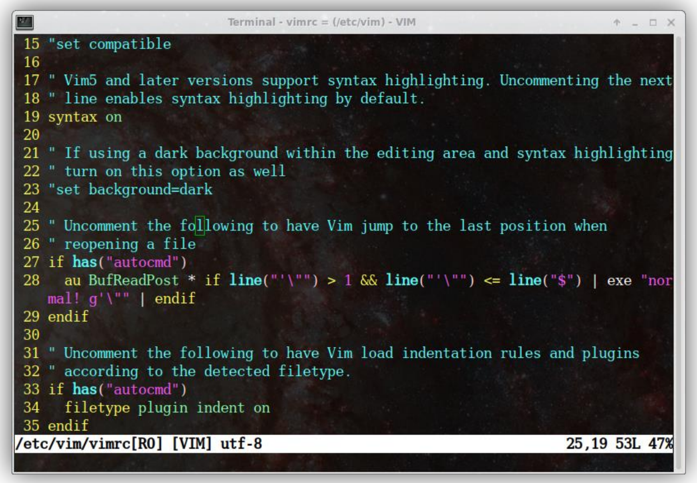
Load, save or delete a stored session

Saved Sessions

Default Settings
demo-server
demo-server-2

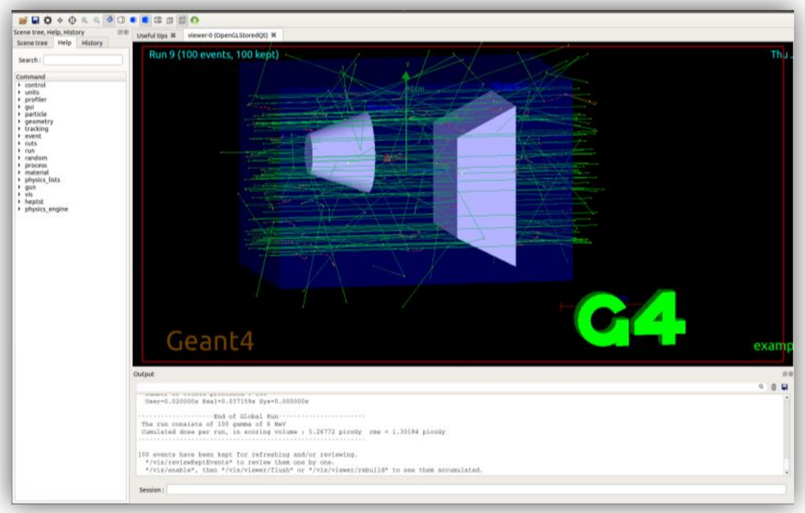
Close window on exit: Always Never [✓ Only on clean exit]

Access High Performance Computing (HPC) Cluster



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "nor
mal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                    25,19 53L 47%
```

Set up scientific computing software on Linux systems



Scene tree, Help, History
Useful tips
viewer-0 (OpenGLStoredQt)
Run 9 (100 events, 100 kept)
Geant4
G4
examp
Output
Session :

```
 User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Run physics experiments, simulations, and data analysis

Jefferson Lab

U.S. DEPARTMENT OF ENERGY | Office of Science

JSA

Getting Started at the Lab

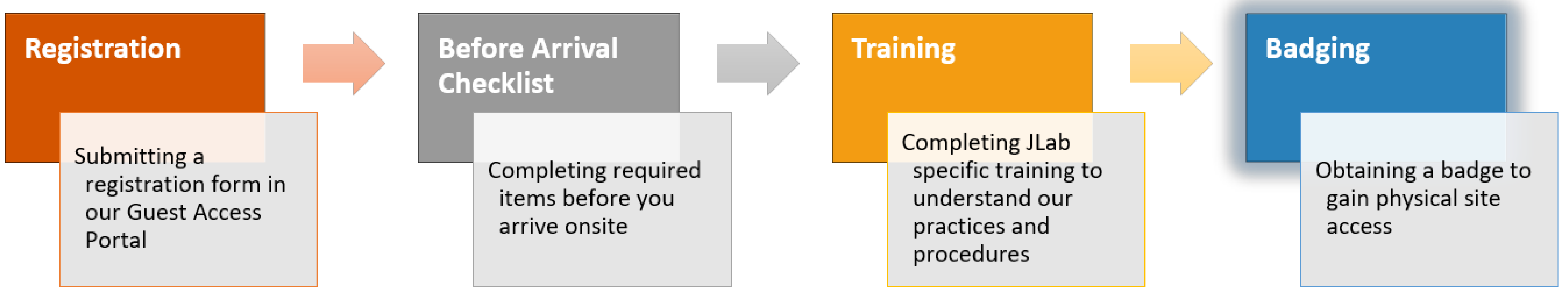
When starting at JLab several administrative steps must be taken to gain access to critical lab resources

- Get a site access name badge
- Get and verify your JLab computer account
- Request access to Microsoft Office (JLab staff and interns only), JLab ifarm, ifarm slurm, VDI, and group file access
- Necessary training (cybersecurity, oxygen deprivation, radiation training, etc.)

Getting Started at the Lab

Get a badge:

- Register – several weeks before visiting, pick a user account you want to have as your permanent email and login
- Checklist – fill out necessary information
- Training – finish basic safety and computer training, advanced training can be done after joining
- Badging – visit the badging office on your first day to get access to JLab buildings


Registration
Submitting a registration form in our Guest Access Portal
Before Arrival Checklist
Completing required items before you arrive onsite
Training
Completing JLab specific training to understand our practices and procedures
Badging
Obtaining a badge to gain physical site access

https://www.jlab.org/facilities/badgingoffice

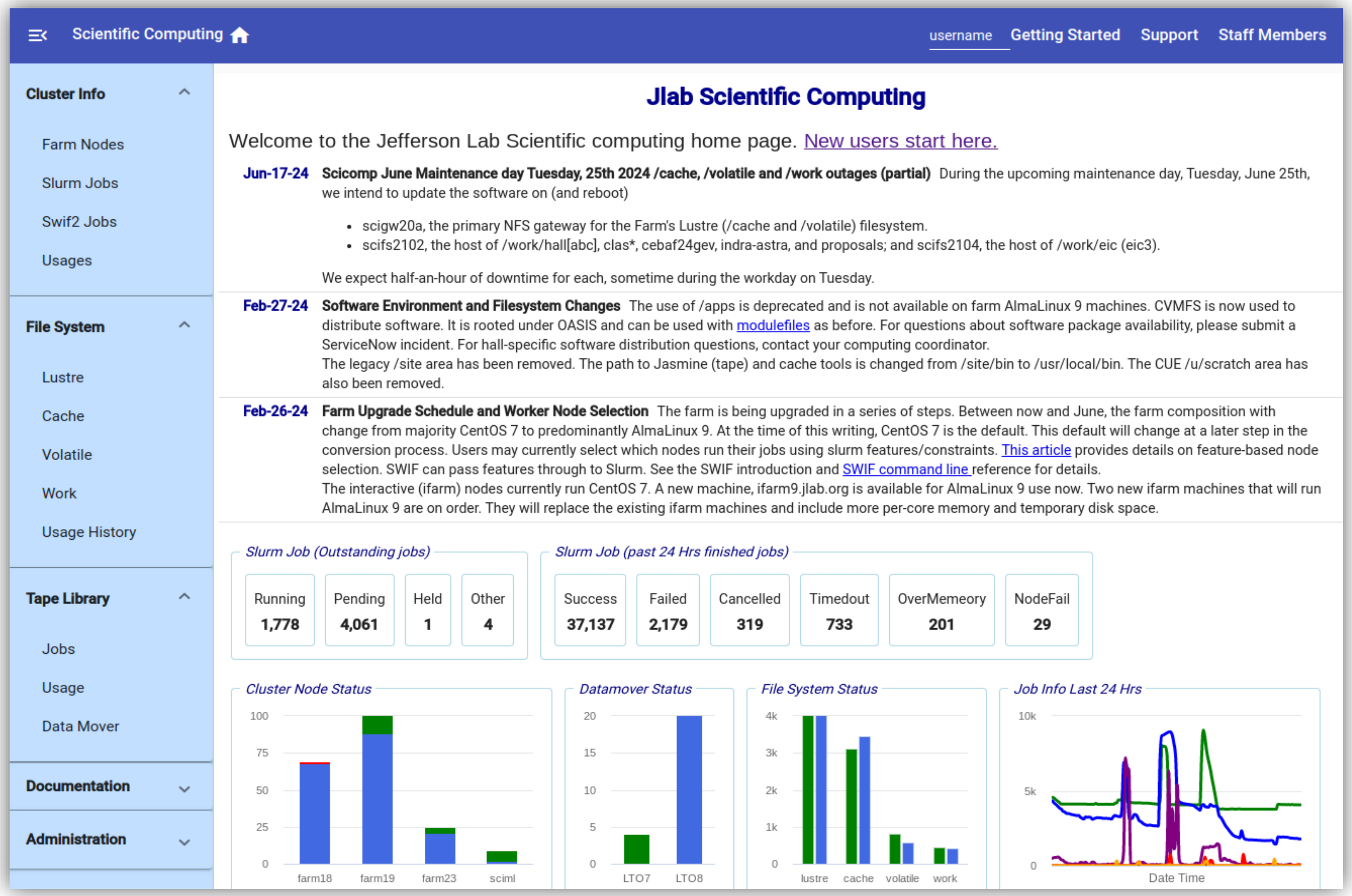
For student interns:

- Mentor needs to finish MGT 202 training
- If outside of formal internship programs (specifically Undergraduate Physics Researchship – UPR students):
    - Mentor needs to formally request site access, a badge, and a computer account for the student

Jefferson Lab

GETTING STARTED AT THE LAB

- Visit computer center helpdesk on 2nd floor of CEBAF Center F wing to receive your computer account
  - Ask for Office 365 access if you are a JLab intern or staff (not available for users)
  - Ask for invitation to MobilePASS SAS MFA for general `login.jlab.org` access
  - Ask to be added to "detimg" computer account user group
  - Ask for VDI RHEL CUE access (Linux web-based virtual machine access)
  - Reject using "SmartCard" USB access if possible (it is required to access personal JLab-managed Windows OS computers)
  - Ask for Jupyterhub Mobile Pass SAS MFA access if you want it



Jlab Scientific Computing

Welcome to the Jefferson Lab Scientific computing home page. New users start here.

Jun-17-24 **Scicomp June Maintenance day Tuesday, 25th 2024 /cache, /volatile and /work outages (partial)** During the upcoming maintenance day, Tuesday, June 25th, we intend to update the software on (and reboot)

- scigw20a, the primary NFS gateway for the Farm's Lustre (/cache and /volatile) filesystem.
- scifs2102, the host of /work/hall[abc], clas*, cebaf24gev, indra-astra, and proposals; and scifs2104, the host of /work/eic (eic3).

We expect half-an-hour of downtime for each, sometime during the workday on Tuesday.

Feb-27-24 **Software Environment and Filesystem Changes** The use of /apps is deprecated and is not available on farm AlmaLinux 9 machines. CVMFS is now used to distribute software. It is rooted under OASIS and can be used with modulefiles as before. For questions about software package availability, please submit a ServiceNow incident. For hall-specific software distribution questions, contact your computing coordinator.
The legacy /site area has been removed. The path to Jasmine (tape) and cache tools is changed from /site/bin to /usr/local/bin. The CUE /u/scratch area has also been removed.

Feb-26-24 **Farm Upgrade Schedule and Worker Node Selection** The farm is being upgraded in a series of steps. Between now and June, the farm composition with change from majority CentOS 7 to predominantly AlmaLinux 9. At the time of this writing, CentOS 7 is the default. This default will change at a later step in the conversion process. Users may currently select which nodes run their jobs using slurm features/constraints. This article provides details on feature-based node selection. SWIF can pass features through to Slurm. See the SWIF introduction and SWIF command line reference for details.
The interactive (ifarm) nodes currently run CentOS 7. A new machine, ifarm9.jlab.org is available for AlmaLinux 9 use now. Two new ifarm machines that will run AlmaLinux 9 are on order. They will replace the existing ifarm machines and include more per-core memory and temporary disk space.

Slurm Job (Outstanding jobs)

Running	Pending	Held	Other
1,778	4,061	1	4

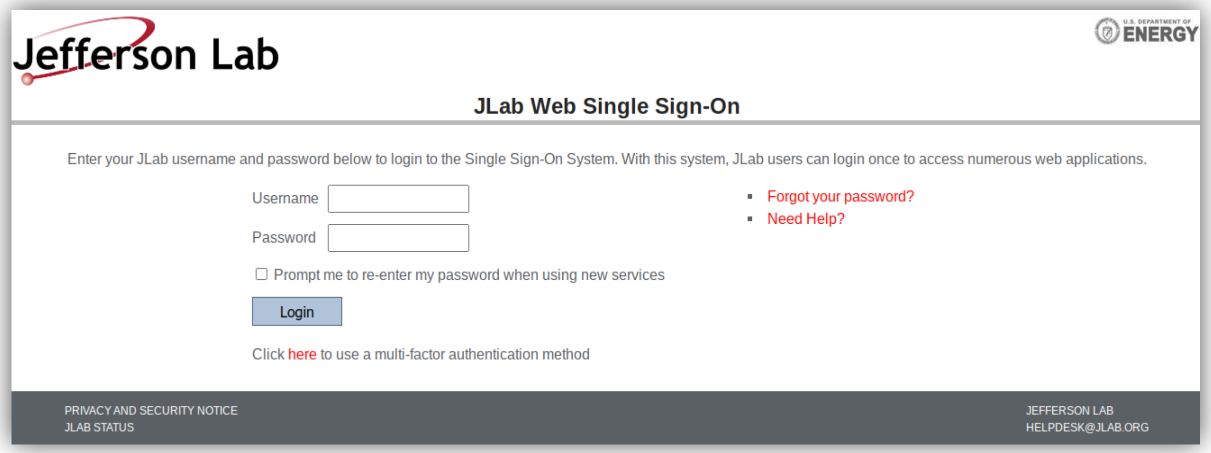
Slurm Job (past 24 Hrs finished jobs)

Success	Failed	Cancelled	Timedout	OverMemeory	NodeFail
37,137	2,179	319	733	201	29

https://scicomp.jlab.org/scicomp/home

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list



JLab Web Single Sign-On

Enter your JLab username and password below to login to the Single Sign-On System. With this system, JLab users can login once to access numerous web applications.

Username

Password

☐ Prompt me to re-enter my password when using new services

Login

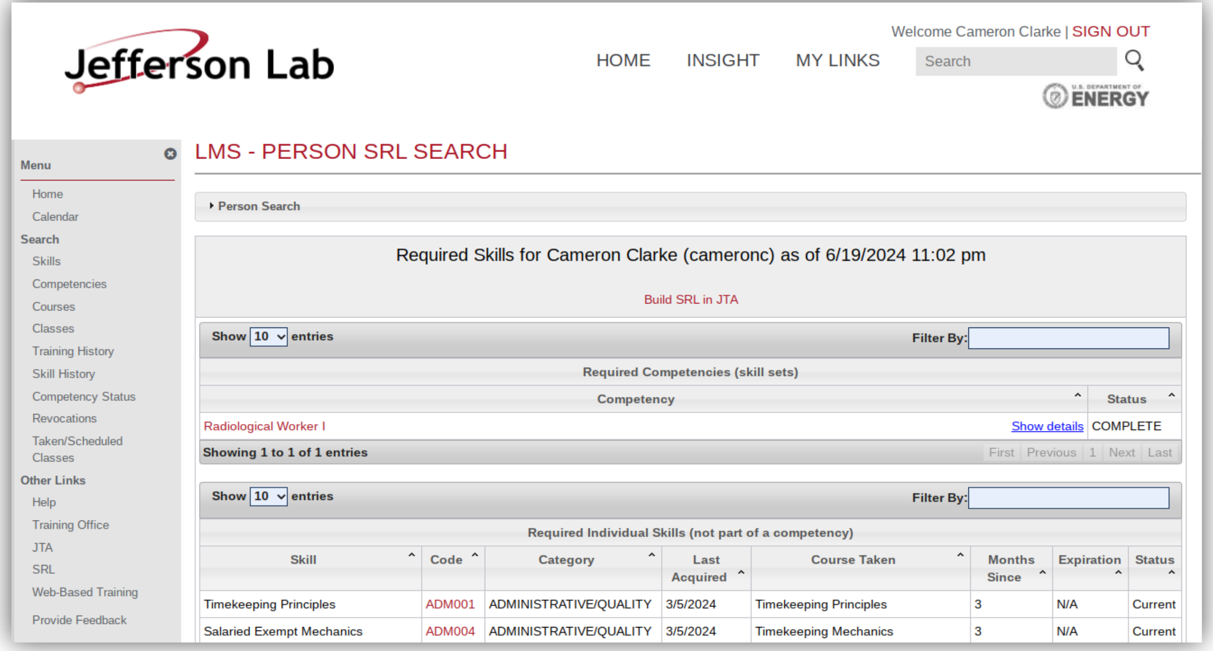
Click here to use a multi-factor authentication method

- Forgot your password?
- Need Help?

PRIVACY AND SECURITY NOTICE
JLAB STATUS

JEFFERSON LAB
HELPDESK@JLAB.ORG

https://jidp.jlab.org/idp/profile/SAML2/Redirect/SSO?execution=e1s2



Welcome Cameron Clarke | SIGN OUT

HOME INSIGHT MY LINKS Search

Menu: Home, Calendar; Search: Skills, Competencies, Courses, Classes, Training History, Skill History, Competency Status, Revocations, Taken/Scheduled Classes; Other Links: Help, Training Office, JTA, SRL, Web-Based Training, Provide Feedback

LMS - PERSON SRL SEARCH

Person Search

Required Skills for Cameron Clarke (cameronc) as of 6/19/2024 11:02 pm

Build SRL in JTA

Show 10 entries — Filter By:

Required Competencies (skill sets)

Competency		Status
Radiological Worker I	Show details	COMPLETE

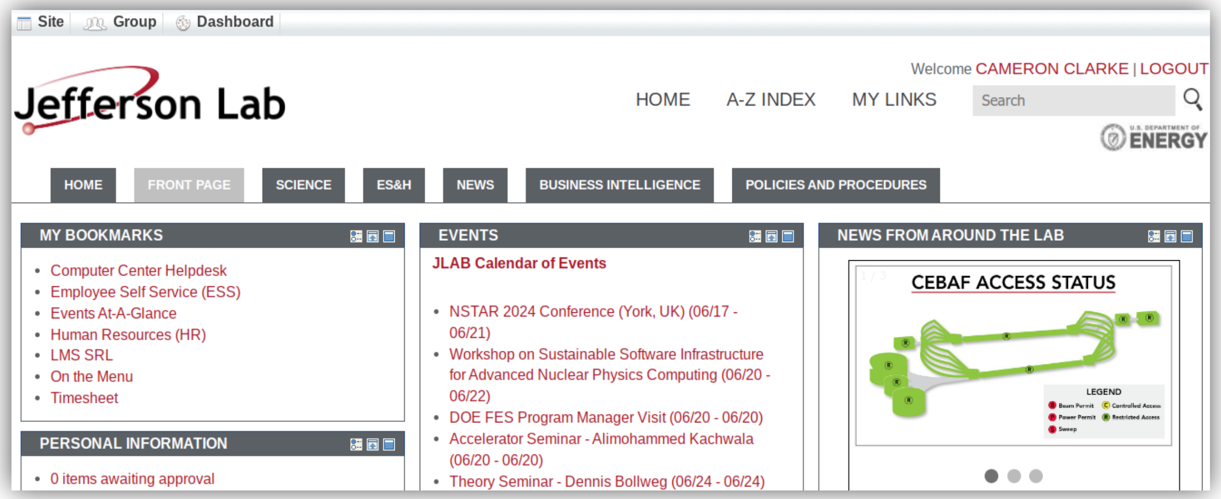
Showing 1 to 1 of 1 entries — First Previous 1 Next Last

Show 10 entries — Filter By:

Required Individual Skills (not part of a competency)

Skill	Code	Category	Last Acquired	Course Taken	Months Since	Expiration	Status
Timekeeping Principles	ADM001	ADMINISTRATIVE/QUALITY	3/5/2024	Timekeeping Principles	3	N/A	Current
Salaried Exempt Mechanics	ADM004	ADMINISTRATIVE/QUALITY	3/5/2024	Timekeeping Mechanics	3	N/A	Current

https://misportal.jlab.org/training/people/srl



Site Group Dashboard

Welcome CAMERON CLARKE | LOGOUT

HOME A-Z INDEX MY LINKS Search

HOME | FRONT PAGE | SCIENCE | ES&H | NEWS | BUSINESS INTELLIGENCE | POLICIES AND PROCEDURES

MY BOOKMARKS
- Computer Center Helpdesk
- Employee Self Service (ESS)
- Events At-A-Glance
- Human Resources (HR)
- LMS SRL
- On the Menu
- Timesheet

PERSONAL INFORMATION
- 0 items awaiting approval

EVENTS

JLAB Calendar of Events
- NSTAR 2024 Conference (York, UK) (06/17 - 06/21)
- Workshop on Sustainable Software Infrastructure for Advanced Nuclear Physics Computing (06/20 - 06/22)
- DOE FES Program Manager Visit (06/20 - 06/20)
- Accelerator Seminar - Alimohammed Kachwala (06/20 - 06/20)
- Theory Seminar - Dennis Bollweg (06/24 - 06/24)

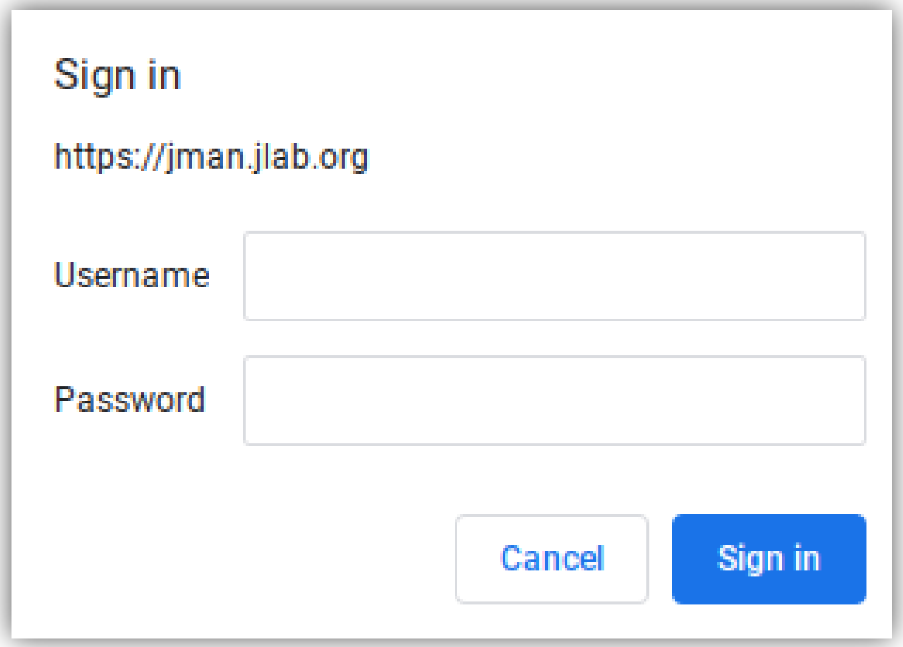
NEWS FROM AROUND THE LAB

CEBAF ACCESS STATUS

https://misportal.jlab.org/portal/insight/frontPage

GETTING STARTED AT THE LAB

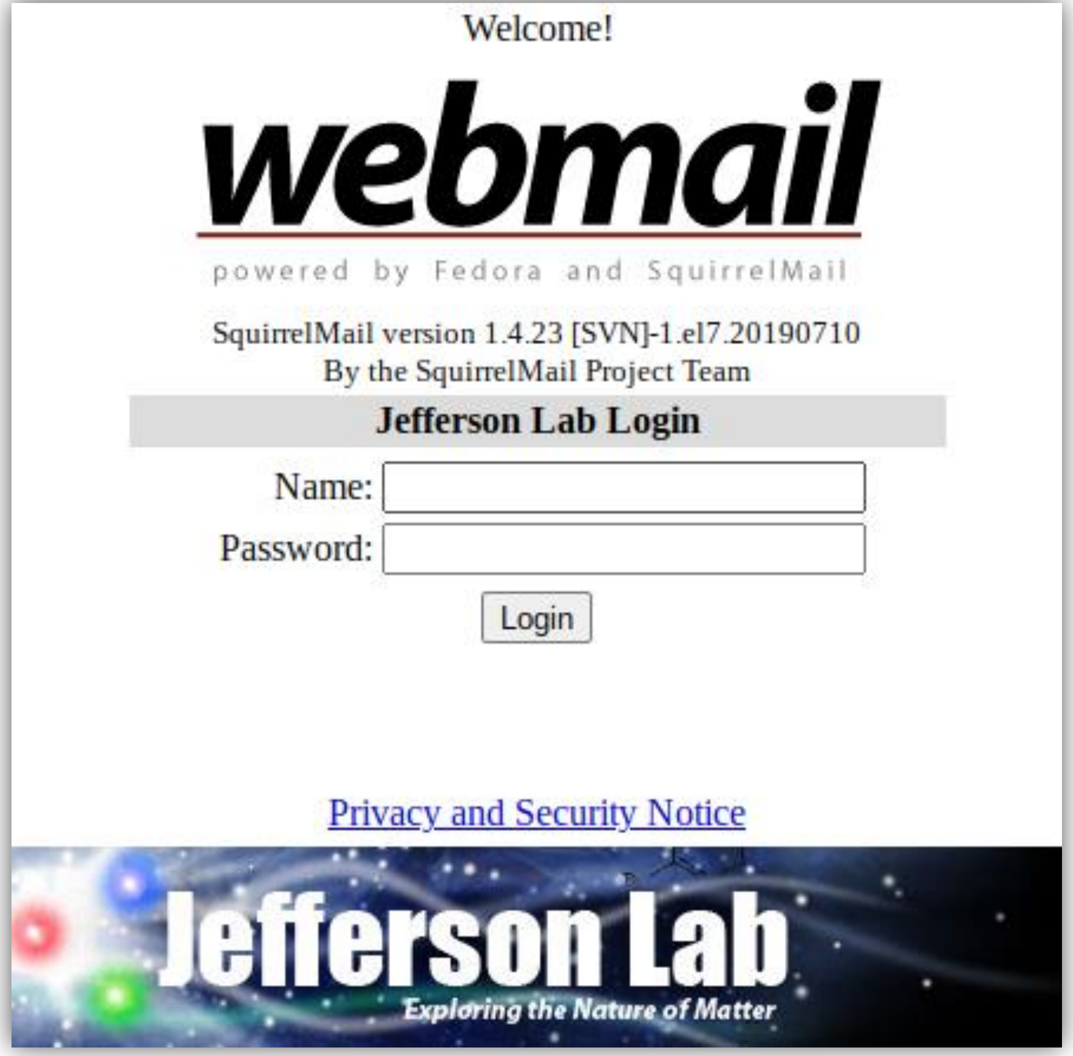
- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command



https://jman.jlab.org/jpasswd

Getting Started at the Lab

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website’s password management system or command line `passwd` command
  - Verify access to squirrel mail (Ignore if using Office)


Welcome!

webmail

powered by Fedora and SquirrelMail

SquirrelMail version 1.4.23 [SVN]-1.el7.20190710
By the SquirrelMail Project Team

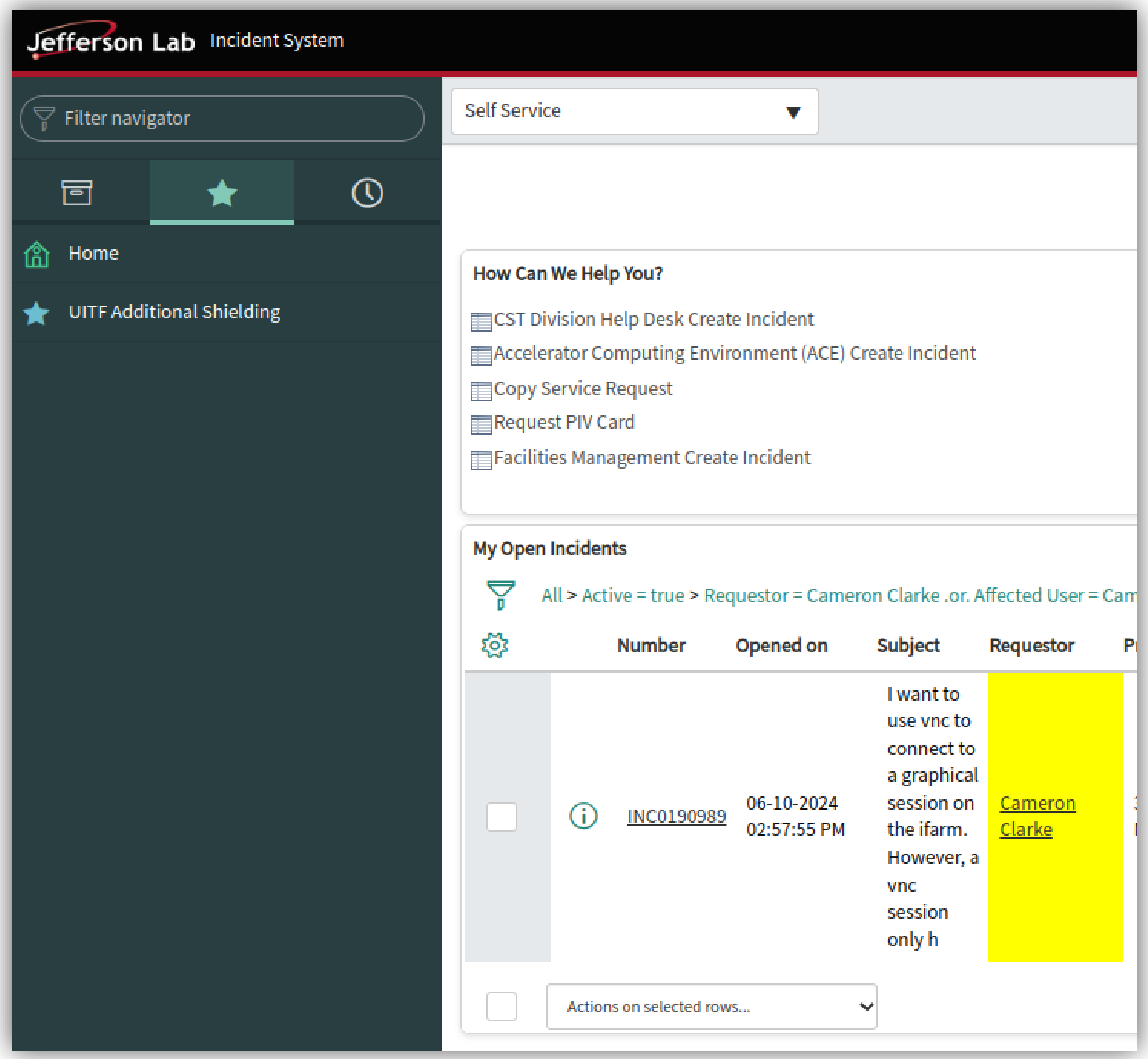
Jefferson Lab Login

Name:
Password:

https://webmail.jlab.org/src/login.php

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help:
    - For any of these steps, if you have any trouble
    - Visiting the help desk in person or calling over the phone (757) 269-7155 works best
    - Emailing helpdesk@jlab.org automatically files a support ticket
    - Or you can file a helpdesk ticket manually at https://jlab.servicenowservices.com/navpage.do



How Can We Help You?

- CST Division Help Desk Create Incident
- Accelerator Computing Environment (ACE) Create Incident
- Copy Service Request
- Request PIV Card
- Facilities Management Create Incident

My Open Incidents

All > Active = true > Requestor = Cameron Clarke .or. Affected User = Cam

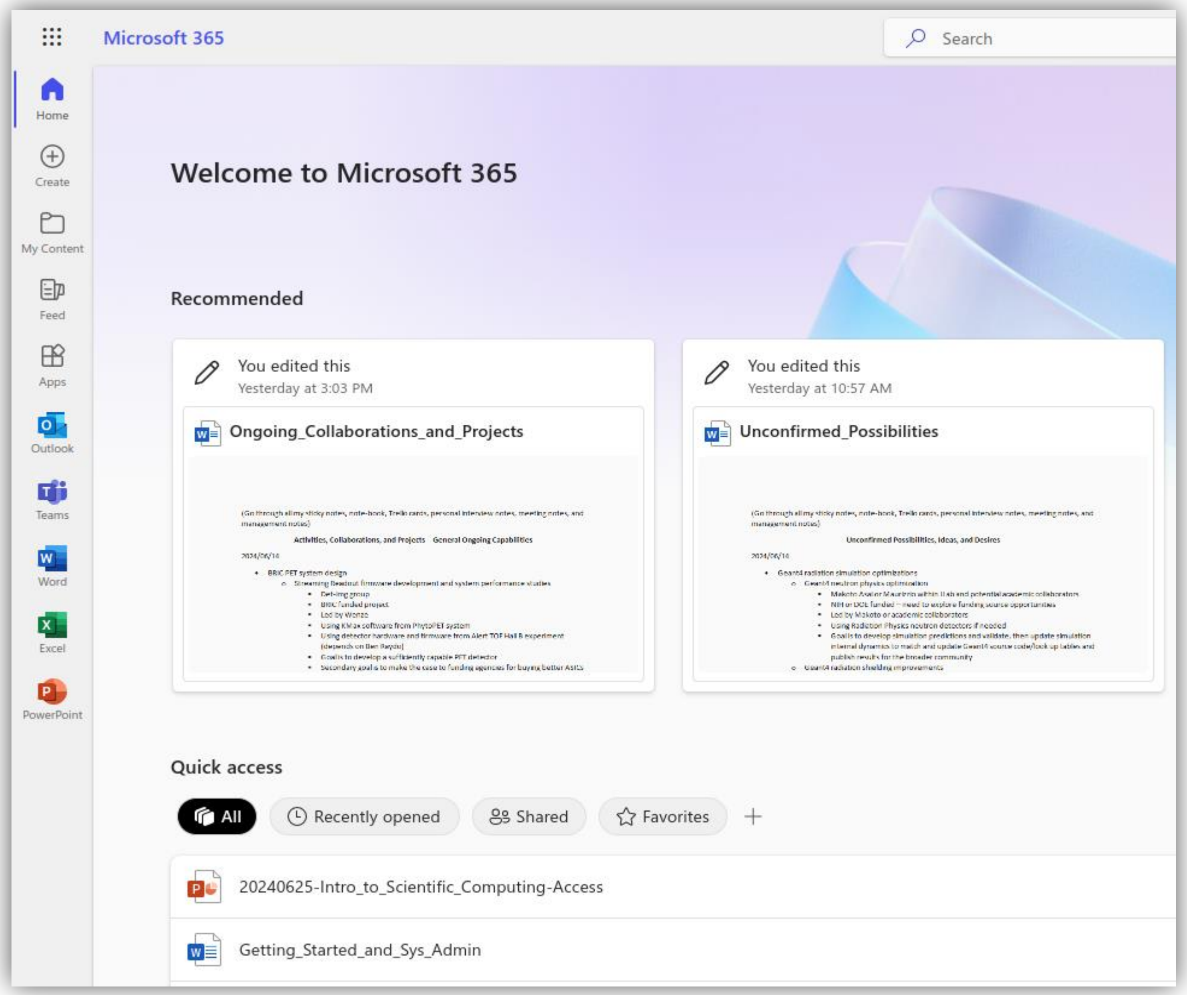
Number	Opened on	Subject	Requestor
INC0190989	06-10-2024 02:57:55 PM	I want to use vnc to connect to a graphical session on the ifarm. However, a vnc session only h	Cameron Clarke

Actions on selected rows...

https://jlab.servicenowservices.com/navpage.do

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA (if you are staff or an intern)
  - Access MS Office + Teams applications (if you are staff or an intern)



Welcome to Microsoft 365

Recommended

You edited this
Yesterday at 3:03 PM

Ongoing_Collaborations_and_Projects

(Go through all my sticky notes, note-book, Trello cards, personal interview notes, meeting notes, and management notes)

Activities, Collaborations, and Projects   General Ongoing Capabilities

2024/06/14

- BRIC PET system design
  - Streaming Readout firmware development and system performance studies
    - Det-img group
    - BRIC funded project
    - Led by Wenze
    - Using KMax software from PhytoPET system
    - Using detector hardware and firmware from Alert TOF Hall B experiment (depends on Ben Raydo)
    - Goal is to develop a sufficiently capable PET detector
    - Secondary goal is to make the case to funding agencies for buying better ASICs

You edited this
Yesterday at 10:57 AM

Unconfirmed_Possibilities

(Go through all my sticky notes, note-book, Trello cards, personal interview notes, meeting notes, and management notes)

Unconfirmed Possibilities, Ideas, and Desires

2024/06/14

- Geant4 radiation simulation optimizations
  - Geant4 neutron physics optimization
    - Makoto Asai or Mauricio within JLab and potential academic collaborators
    - NIH or DOE funded – need to explore funding source opportunities
    - Led by Makoto or academic collaborators
    - Using Radiation Physics neutron detectors if needed
    - Goal is to develop simulation predictions and validate, then update simulation internal dynamics to match and update Geant4 source code/look up tables and publish results for the broader community
  - Geant4 radiation shielding improvements

Quick access

All | Recently opened | Shared | Favorites | +

20240625-Intro_to_Scientific_Computing-Access

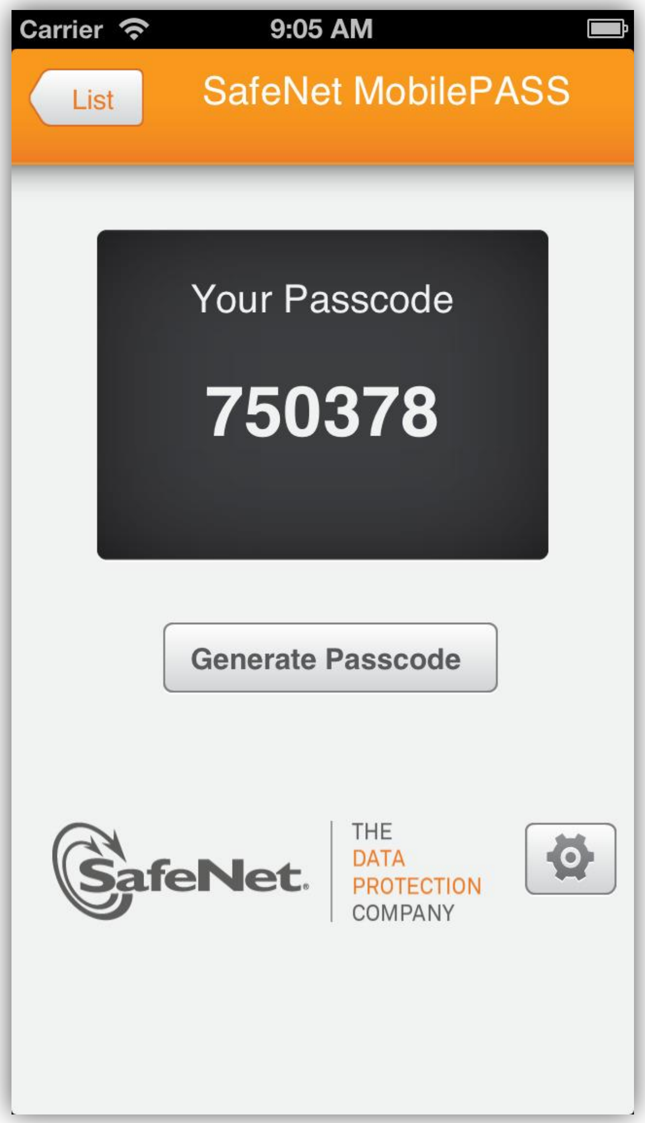
Getting_Started_and_Sys_Admin

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA
  - Access MS Office + Teams applications
  - Set up and verify login.jlab.org access with the SAS MobilePass multifactor app invitation

To test login.jlab.org access:

- Ask the helpdesk staff to help you test your login.jlab.org by terminal on their work-station at the helpdesk
- Otherwise follow instructions in subsequent set of slides



MobilePass Phone App

Or – USB Code Generator (not a smart card, just a convenient dongle that you can press to get a code)

Uses SafeNet Authentication Service (SAS)

Jefferson Lab

Getting Started at the Lab

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA
  - Access MS Office + Teams applications
  - Set up and verify login.jlab.org access with the SAS MobilePass multifactor app invitation
  - Set up and verify interactive farm (ifarm) slurm and batch access (optional, can wait until later)

  To test ifarm9.jlab.org access:
  - Ask the helpdesk staff to help you test your ifarm9.jlab.org access by terminal on their work-station at the helpdesk
  - Otherwise follow instructions in subsequent set of slides

Getting Started at the Lab

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA
  - Access MS Office + Teams applications
  - Set up and verify login.jlab.org access with the SAS MobilePass multifactor app invitation
  - Set up and verify interactive farm (ifarm) slurm and batch access (optional, can wait until later)

  To test ifarm9.jlab.org access:
  - Ask the helpdesk staff to help you test your ifarm9.jlab.org access by terminal on their work-station at the helpdesk
  - Otherwise follow instructions in subsequent set of slides

Interactive farm (ifarm) slurm access pre-requisites:

- Ask scicomp/helpdesk to add you to needed user group
  - Check access by logging in to any Linux system and executing `groups <username>`
- A slurm account manager will need to add you by executing the following ifarm commands
  - `sacctmgr -i create user <username> account=<group name>`
  - `sacctmgr show user <username>`
  - `sacctmgr list assoc account=<group name>`

  Servicenow slurm account manager guide: https://jlab.servicenowservices.com/kb?id=kb_article_view&sysparm_article=KB0014685
- To verify complete farm batch job access – log in to ifarm9, execute following commands from ifarm node:
  - To enable submitting farm jobs you must run `/site/bin/jcert -create` on ifarm

  From (https://jlab.servicenowservices.com/scicomp?id=kb_article_view&sys_kb_id=22d2c5db1b4a09506a9e85dae54bcbcc)
- To test ability to submit jobs, then execute:
  - ifarm> salloc -p ifarm
  - ifarm> srun --pty bash
  - bash-4.2$ echo "This is running on host `hostname`"

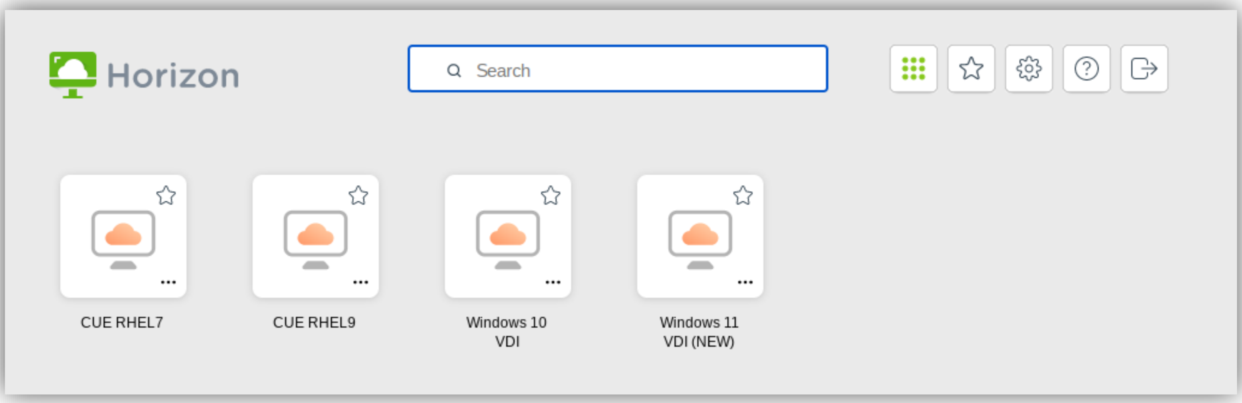
  From (https://scicomp.jlab.org/docs/farm_slurm_batch_interactive_jobs)

Jefferson Lab

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA
  - Access MS Office + Teams applications
  - Set up and verify login.jlab.org access with the SAS MobilePass multifactor app invitation
  - Set up and verify interactive farm (ifarm) slurm and batch access (optional, can wait until later)
  - Verify VDI access

Log in to https://vdi.jlab.org (download the app or use the easier html web-version) using your common user environment (CUE) username and password



Once you get to this screen – select CUE RHEL9

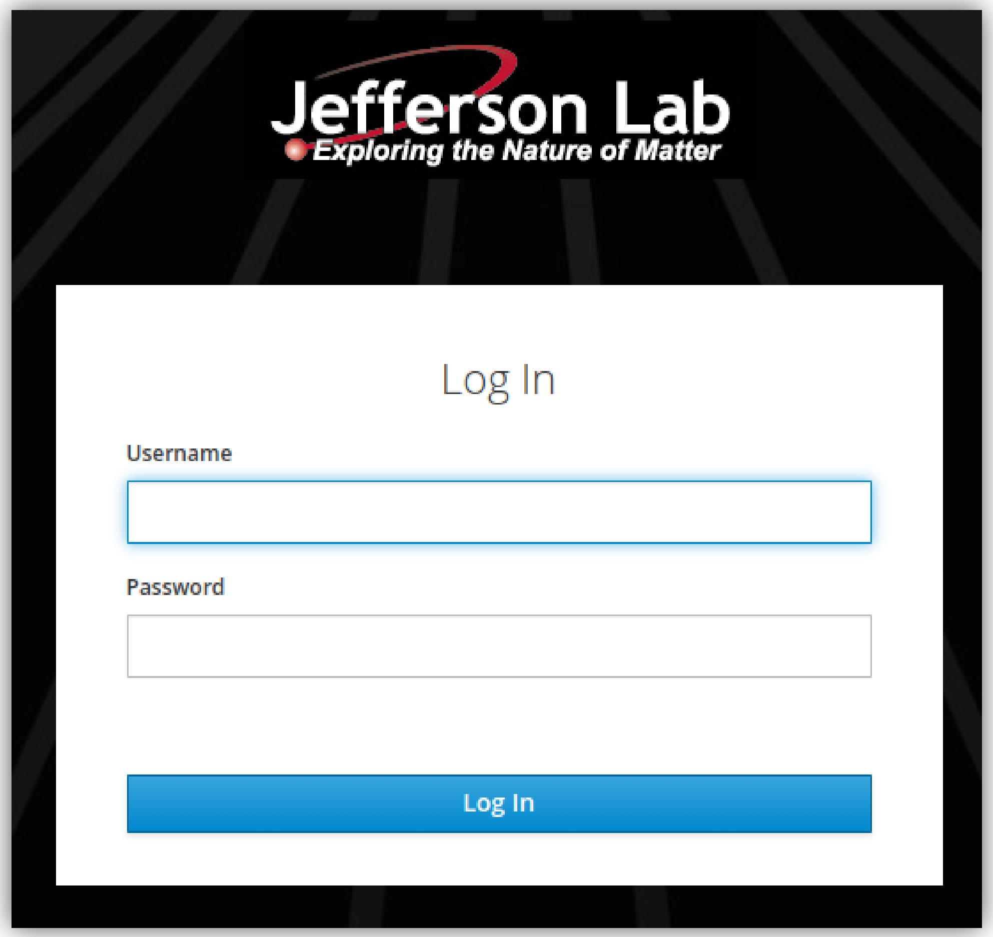
When prompted, log in to RHEL9 system welcome screen using your MobilePASS SAS personal PIN plus 6 digit MFA code as the password

(a SmartCard is required for Windows Systems – avoid!)

JLab VDI Support Article:
https://jlab.servicenowservices.com/sp?id=kb_article&sys_id=dec16b0ddb7f0410ee4a3889fc961944

GETTING STARTED AT THE LAB

- Verify account access
  - Go to www.jlab.org, log in to your user account, visit the insight page, and check your skills list
  - Change your temporary password, using the website's password management system or command line `passwd` command
  - Verify access to squirrel mail
  - If you need help: helpdesk@jlab.org tickets
  - Verify Office 365 access – log in to www.office.com with JLab credentials, set up Microsoft MFA
  - Access MS Office + Teams applications
  - Set up and verify login.jlab.org access with the SAS MobilePass multifactor app invitation
  - Set up and verify interactive farm (ifarm) slurm and batch access (optional, can wait until later)
  - Verify VDI access
  - Set up and verify access to JupyterHub



Log in to https://jupyterhub.jlab.org – first using your common user environment (CUE) password – second using your MobilePASS SAS OTP personal PIN plus code

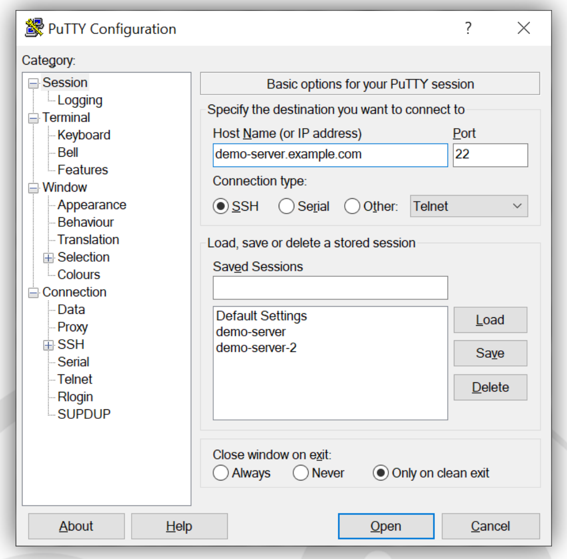
GETTING STARTED AT THE LAB

Slides cover:

- Account activation and access
- **Accessing scientific computing resources (ifarm)**
- Learning to navigate and use Linux computers
- Introduction to scientific computing resources
  (Geant4 simulation and ROOT data analysis)

Friday, August 9, 2024

Accessing Scientific Computing Resources



Specify the destination you want to connect to

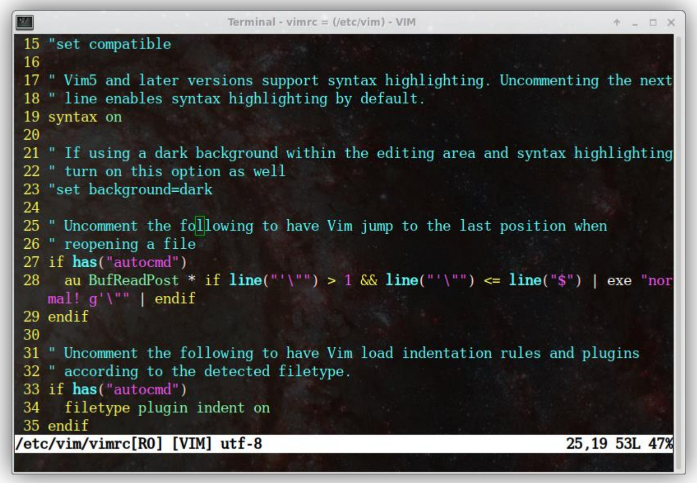
Host Name (or IP address): demo-server.example.com
Port: 22

Connection type: [✓ SSH] Serial Other: Telnet

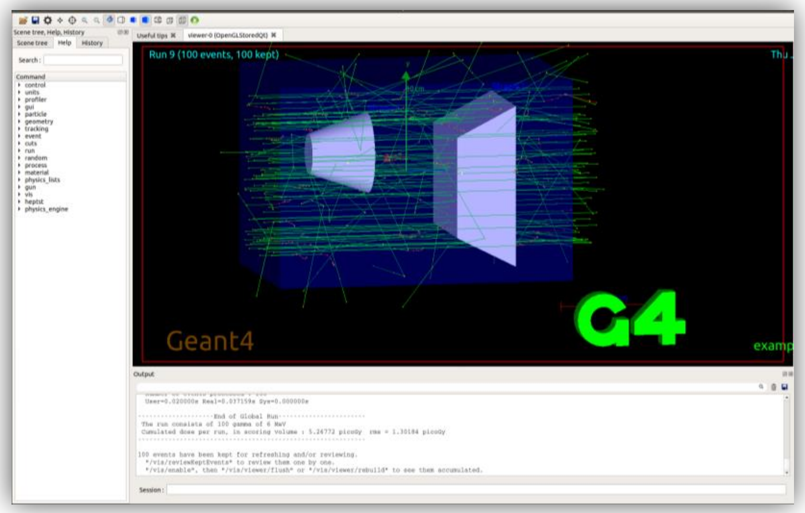
Load, save or delete a stored session

Saved Sessions:
- Default Settings
- demo-server
- demo-server-2

Close window on exit: Always Never [✓ Only on clean exit]



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "normal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                    25,19 53L 47%
```



```
 User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Access High Performance Computing (HPC) Cluster → Set up scientific computing software on Linux systems → Run physics experiments, simulations, and data analysis

This section

ACCESSING JLAB IFARM

Logging into and accessing the JLab job submission (batch) and interactive farm (ifarm) computing systems can be done in many ways

- Depends on your personal system and needs
- Find the method in the following slides that is most convenient to you
- **Linux RHEL9 VDI is recommended**

The following slides cover accessing JLab's ifarm with various systems:

- ➢ Secure Shell (SSH) Protocol
  - ➢ Linux
  - ➢ JLab's Virtual Desktop Infrastructure (VDI), from either html web app or desktop app
  - ➢ Windows
  - ➢ Mac
- ➢ Interactive farm (ifarm) slurm access
- ➢ Farm batch submission access
- ➢ Transferring files

JLab Service Now Article on remote Access:
https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0015066

Jefferson Lab

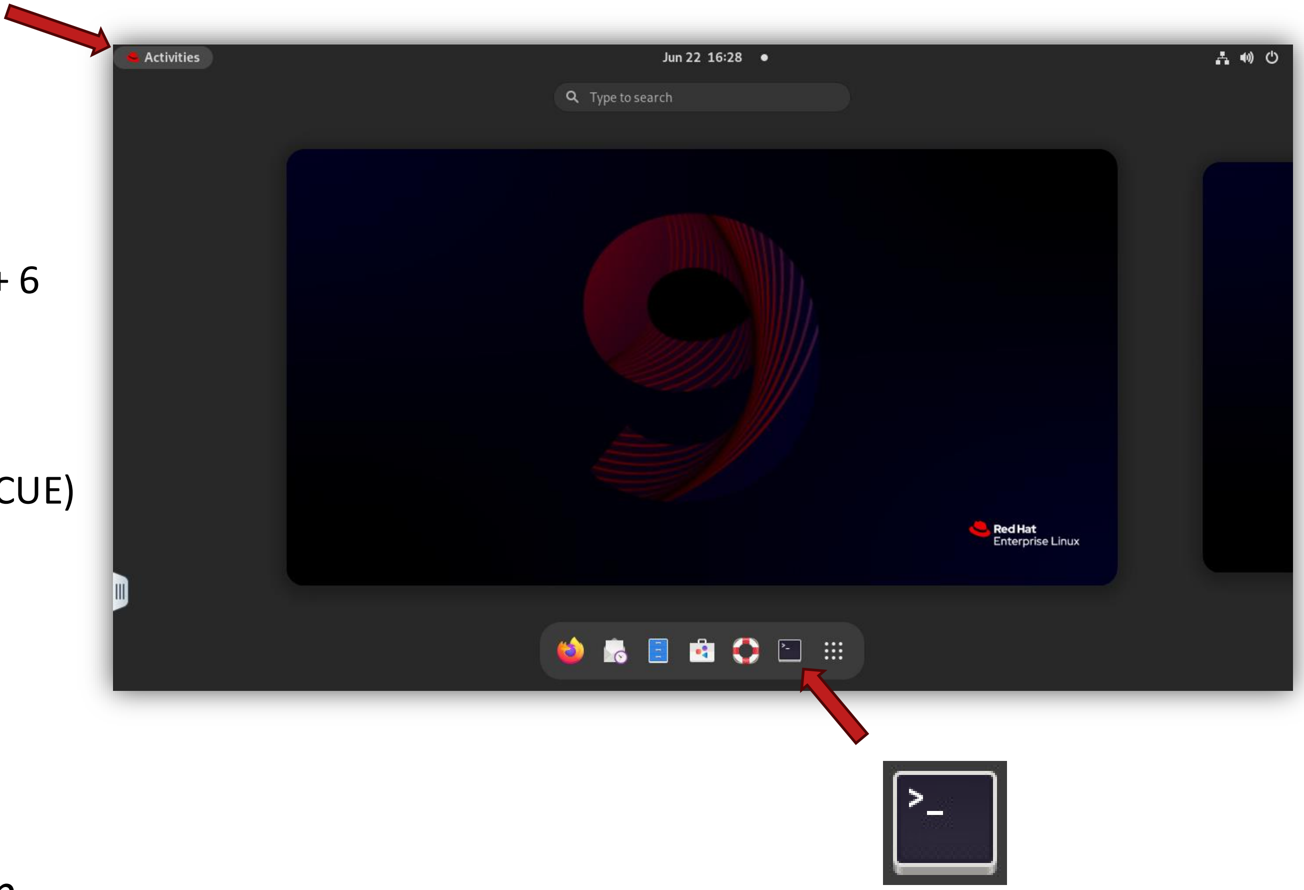
ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Linux:

- Once inside a Linux system
  - Click Activities or find an applications drop down menu
  - Find the terminal application and open it
  - Execute `ssh –Y <username>@login.jlab.org`
    - Use MobilePass SAS MFA personal 6-8 digit code + 6 digit encrypted code output in one line as the password
  - Then: Execute `ssh –Y <username>@ifarm`
    - Use your usual JLab Common User Environment (CUE) password, or your ssh-key if you have set one up

- Linux VDI (shown on the right):
  - Same as using Linux, managed by JLab
  - Accessed using https://vdi.jlab.org login website
  - See notes from prior set of slides for instructions
  - Then: open a terminal from inside the VDI session

JLab RHEL9 VDI screen – activities button and terminal



ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Windows the new way:

- There is a ssh client now installed by default in modern Windows operating systems
- Open any terminal in Windows: "cmd", PowerShell, or any Windows Subsystem for Linux (WSL) prompt
- Execute SSH commands following instructions on prior Linux instructions slide

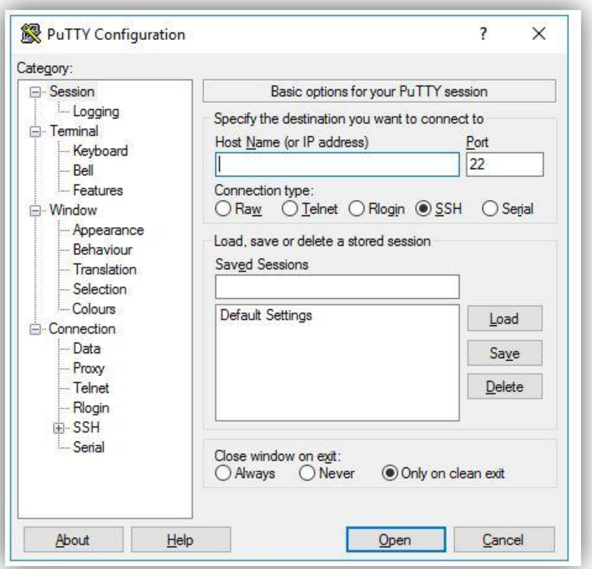
- To Install a Linux operating system in Windows Subsystem for Linux (WSL) utilize Windows official channels
  - Ubuntu 20.04.6: https://apps.microsoft.com/detail/9mttcl66cpxj?hl=en-us&gl=en
  - These can come in many Linux flavors – Ubuntu and Debian are the same thing and are most newby friendly
  - Enable installation of many scientific computing resources such as Python, ROOT, and even Geant4 using package management tools such as "Snap" or "Miniconda" - to be discussed in subsequent sets of slides
  - Simplify X11 windows forwarding for Linux and gives you the power of Linux within the Windows ecosystem

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Windows the old way:

- Download and Install PuTTY and Xming
  - PuTTY: Download the application from the Microsoft Apps store to install PuTTY.
    - https://apps.microsoft.com/detail/xpfnzksklbp7rj?amp%3Bgl=US&hl=en-us&gl=US
  - Xming: Download from SourceForge – be careful to not click on advertisements – inspect all links first!
    - https://sourceforge.net/projects/xming/
    - Xming will run in the background, only acting when you try to open GUIs on remote servers via PuTTY

Host Name for Accessing JLab scientific computing resources is "login.jlab.org"



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

PuTTY GUI "Session" page

Enter the hostname you want to access in "Host Name (or IP address) field

Port 22 is standard ssh protocol port

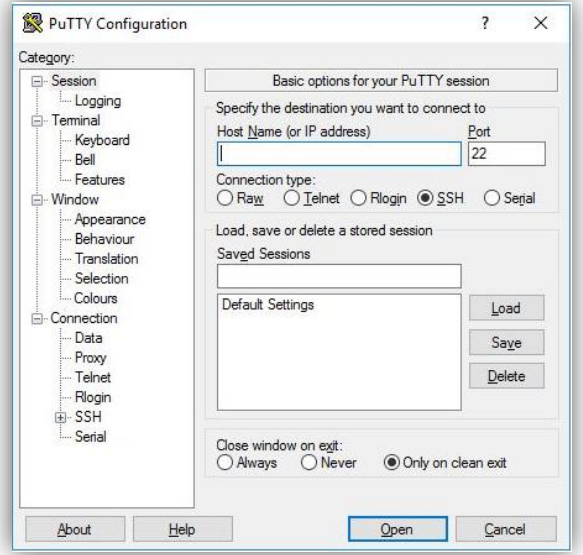
IPv4 protocol port numbers range from 0 - 65535 (16 bit integer range)

Jefferson Lab

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Windows the old way:

- Configure PuTTY for X11 window forwarding to work
  - Open PuTTY, filling in the IP address or DNS host name
    - Type in "login.jlab.org" (no quotes) to access the JLab scientific computing resources from on or off-site
    - A specified hostname, like "jlabl1" (no quotes), if ifarm access is not the goal
  - Go to the Connection > SSH > Auth > X11 Page menu
    - Check the box "Enable X11 forwarding" and type in the "X display location" to be "localhost:0" (no quotes)
  - Go back to "Session" page, type some useful name (like "jlab login"), and 'Save' your configuration for later loading



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

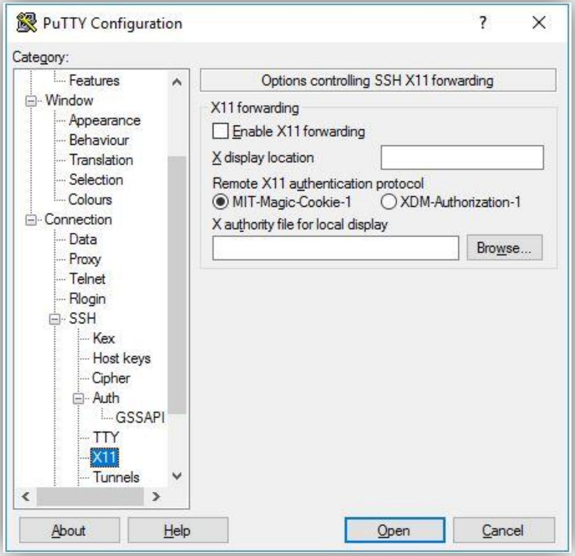
Connection type: Raw / Telnet / Rlogin / [✓ SSH] / Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: Always / Never / [✓ Only on clean exit]

PuTTY GUI "Session" page



PuTTY Configuration

Options controlling SSH X11 forwarding

X11 forwarding

☐ Enable X11 forwarding

X display location:

Remote X11 authentication protocol: [✓ MIT-Magic-Cookie-1] / XDM-Authorization-1

X authority file for local display:

Connection > SSH > Auth > X11 Page

Jefferson Lab

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Windows the old way:

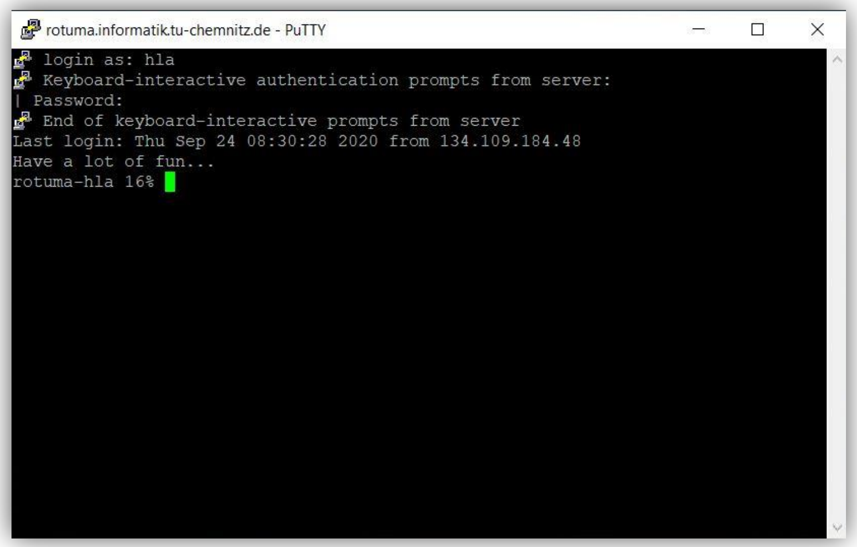
- Try X11 forwarding with XMing
  - You will need to execute the XMing program (double click it's start menu icon) so that it runs in the background
  - You may need to allow Window’s firewall access – if so then accept the pop-up request

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Windows the old way:

- Try X11 forwarding with XMing
  - You will need to execute the XMing program (double click it's start menu icon) so that it runs in the background
  - You may need to allow Window's firewall access – if so then accept the pop-up request

- ssh into the "login.jlab.org" portal
  - Connect by using the "open" button on PuTTY
    - Use MobilePass SAS MFA personal 6-8 digit code + 6 digit encrypted code output in one line as the password to enter the login portal
  - Then: execute `ssh –Y <username>@ifarm` on the command line
    - Use your usual JLab Common User Environment (CUE) password, or your ssh-key if you have set one up



rotuma.informatik.tu-chemnitz.de - PuTTY

```
login as: hla
Keyboard-interactive authentication prompts from server:
| Password:
End of keyboard-interactive prompts from server
Last login: Thu Sep 24 08:30:28 2020 from 134.109.184.48
Have a lot of fun...
rotuma-hla 16%
```

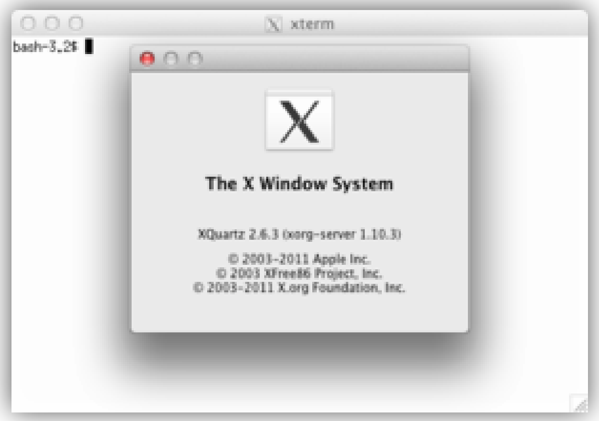
A PuTTY prompt after successful logging into a remote host

Jefferson Lab

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

SSH with Mac:

- Install XQuartz https://www.xquartz.org/
  - XQuartz must be run before attempting to open any process with forwarded graphical display (similar to Windows' XMing program, but easier)
- Open a standard Mac terminal and SSH the same way as with Linux
  - Execute `ssh –Y <username>@login.jlab.org`
    - Use MobilePass SAS MFA personal 6-8 digit code + 6 digit encrypted code output in one line as the password
  - Then: execute `ssh –Y <username>@ifarm` command
    - Use your usual JLab Common User Environment (CUE) password, or your ssh-key if you have set one up



```
bash-3.2$
```

XQuartz 2.6.3 (xorg-server 1.10.3)
© 2003-2011 Apple Inc.
© 2003 XFree86 Project, Inc.
© 2003-2011 X.org Foundation, Inc.

XQuartz splash page and included xterm terminal emulator

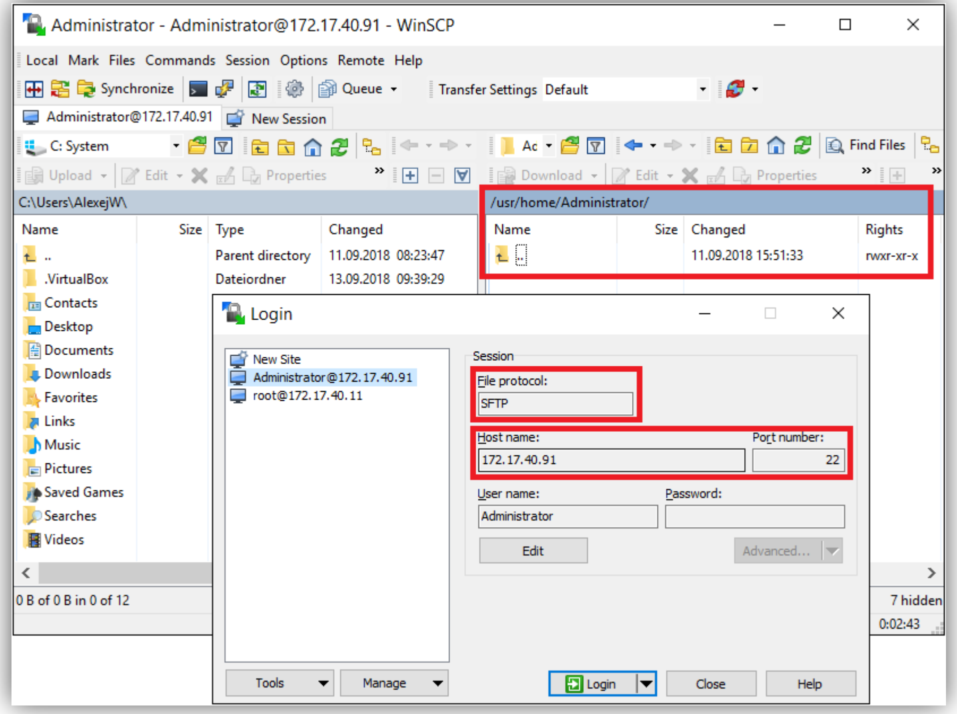
Jefferson Lab

Accessing JLab ifarm with SSH X windows forwarding

Transferring files:

- Windows:
  - Download WinSCP: https://winscp.net/eng/index.php
  - To tunnel through login.jlab.org and access jlabl1 or ifarm9 computers
    - From your command prompt ("cmd", PowerShell, or WSL)
    - Execute: `ssh -L 23:ifarm9:22 <username>@login.jlab.org`
  - Then open WinSCP to localhost:23 and login with JLab account to ifarm9

WinSCP across the login portal from outside JLab network requires ProxyJump or SSH port forwarding tunneling



Administrator - Administrator@172.17.40.91 - WinSCP

Transfer Settings Default

C:\Users\AlexejW\

Name	Size	Type	Changed
..		Parent directory	11.09.2018 08:23:47
.VirtualBox		Dateiordner	13.09.2018 09:39:29
Contacts			
Desktop			
Documents			
Downloads			
Favorites			
Links			
Music			
Pictures			
Saved Games			
Searches			
Videos			

0 B of 0 B in 0 of 12

/usr/home/Administrator/

Name	Size	Changed	Rights
..		11.09.2018 15:51:33	rwxr-xr-x

7 hidden

0:02:43

Login

New Site
Administrator@172.17.40.91
root@172.17.40.11

Session
File protocol: SFTP
Host name: 172.17.40.91
Port number: 22
User name: Administrator
Password:

Example WinSCP GUI screenshot

Left is your local computer file tree

Right is the remote computer file tree

Port "23" here is chosen for convenience, it may be already occupied, in which case chose another unoccupied port

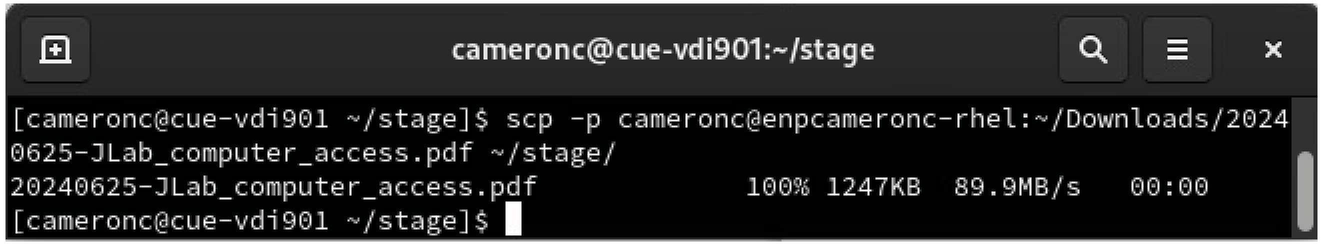
Jefferson Lab

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Advanced scp – Using Manual ProxyJump:
Upload: scp -J <user>@login.jlab.org. file_to_upload <user>@ifarm:/dest/path/
Download: scp -J <user>@login.jlab.org. <user>@ifarm:/file/to/download ./

Transferring files:

- From terminals (any Operating System)
- Secure Copy Protocol: `scp -rp <files to copy> <destination>`
    - "-r" allows recursive copy, meaning all sub-folders and their contents will be copied as well
    - "-p" allows preserving permissions and meta-data like date of file creation, etc. when copying (recommended)
    - You can copy from the local computer to a remote destination, then:
        - <destination> = <username>@<remote host name>:<path on remote computer>
        - <path on remote computer> should be something like ~/Downloads/ (where ~ means "my home folder")
        - <files to copy> would be the path to the files, or just the file's name if it is in the current directory
    - You can copy from a remote computer if you know the exact path, then:
        - <files to copy> would be <username>@<remote host name>:<path on remote computer>
    - Example: `scp –p cameronc@enpcameronc-rhel:~/Downloads/20240625-JLab_computer_access.pdf ~/stage/`
        - ` ` tick marks denote a block of code
        - < > brackets denote something you should replace with your own choice
        - scp through login portal requires ProxyJump or tunneling



```
[cameronc@cue-vdi901 ~/stage]$ scp -p cameronc@enpcameronc-rhel:~/Downloads/2024
0625-JLab_computer_access.pdf ~/stage/
20240625-JLab_computer_access.pdf            100% 1247KB  89.9MB/s   00:00
[cameronc@cue-vdi901 ~/stage]$
```

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Farm login and batch submission access:

- ssh to ifarm9 following earlier instructions
- Perform an interactive job submission test (instructions on the right)
- Batch submission, monitoring, and debugging will be covered later

Interactive farm (ifarm) slurm access pre-requisites:

- Ask scicomp/helpdesk to add you to needed user group
  - Check access by logging in to any Linux system and executing `groups <username>`
- A slurm account manager will need to add you by executing the following ifarm commands
  - `sacctmgr -i create user <username> account=<group name>`
  - `sacctmgr show user <username>`
  - `sacctmgr list assoc account=<group name>`

  Servicenow slurm account manager guide: https://jlab.servicenowservices.com/kb?id=kb_article_view&sysparm_article=KB0014685
- To verify complete farm batch job access – log in to ifarm9, execute following commands from ifarm node:
  - To enable submitting farm jobs you must run `/site/bin/jcert -create` on ifarm

  From (https://jlab.servicenowservices.com/scicomp?id=kb_article_view&sys_kb_id=22d2c5db1b4a09506a9e85dae54bcbcc)
- To test ability to submit jobs, then execute:
  - ifarm> salloc -p ifarm
  - ifarm> srun --pty bash
  - bash-4.2$ echo "This is running on host `hostname`"

  From (https://scicomp.jlab.org/docs/farm_slurm_batch_interactive_jobs)

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Advanced:

- SSH key generation and installation
- SSH command chaining
- SSH and SCP tunneling through login.jlab.org
- SSH ProxyJump with linux SSH config
- SSH ProxyJump with Windows PuTTY
- Virtual Network Computing (VNC) initialization
- VNC port forward tunneling through login.jlab.org and local access

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Advanced:

- SSH key generation and installation: Linux and Mac (including Windows Subsystem for Linux)
  - GitHub guide (useful for github access instructions): https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
  - nixCraft guide (preferred): https://www.cyberciti.biz/faq/how-to-set-up-ssh-keys-on-linux-unix/

- SSH key generation and installation: Windows ("cmd", PowerShell)
  - Microsoft guide: https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_keymanagement
  - Atlassian guide: https://support.atlassian.com/bitbucket-cloud/docs/set-up-personal-ssh-keys-on-windows/

- PuTTY ssh keys in Windows:
  - DigitalOcean guide: https://docs.digitalocean.com/products/droplets/how-to/add-ssh-keys/create-with-putty/

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Advanced:

- SSH command chaining via manual ProxyJump
    - `ssh -Y -J <user>@login.jlab.org <user>@ifarm`

    - J indicates manul ProxyJump
    - Y indicates X forwarding with strict security check
    - This is quite convenient: **it simplifies to one line and requires entering only the MFA password**

ACCESSING JLAB IFARM WITH SSH X WINDOWS FORWARDING

Advanced:

- SSH and SCP tunneling through login.jlab.org
  - This permits only entering the MobilePass SAS MFA key one time and no subsequent authentication is needed
  - Set up tunnel: `ssh -L <forwarding port>:ifarm:22 <username>@login.jlab.org`
    - <forwarding port> can be 22
    - alias tunnellogin="ssh -L 2201:ifarm:22 <username>@login.jlab.org"
  - SSH over tunnel: `ssh -p <forwarding port> localhost`
    - <same forwarding port as before>
    - alias ifarmssh="ssh -p 2201 localhost"
  - Make sure the terminal hosting the tunnel does not close, or else all traffic will fail
  - Verify that the port you chose to forward over is allowed (not bind rejected error message after logging in) - it may be occupied by another user or process

Accessing JLab ifarm with SSH X windows forwarding

Advanced:

- SSH ProxyJump with linux SSH config
  - Specific guide, including Windows, Mac, and Linux instructions: https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0014918
  - General guide: https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0015066

Accessing JLab ifarm with Virtual Network Computing (VNC)

Advanced:

- Virtual Network Computing (VNC) initialization
  - Old JLab instructions (requires sudo access): https://jlab.servicenowservices.com/sp?sys_kb_id=d4520b731b2f2d106a9e85dae54bcbe6&id=kb_article_view&sysparm_rank=1&sysparm_tsqueryId=9a6ccb0c478fca10281bbd51026d4392
  - Hall A wiki instructions (a bit old, but still works): https://hallaweb.jlab.org/wiki/index.php/Using_a_VNC_Server/Client#Setting_up_the_VNC_Server

  - Official instructions (requires sudo):
    - Read /usr/share/doc/tigervnc/HOWTO.md
    - Set user defaults in /etc/tigervnc/vncserver.users - set default VNC session number "y" (should be y=1 for primary user)
    - `systemctl start vncserver@:1`
    - `systemctl enable vncserver@:1`
    - Or as sudo replace 1 with "x" to start all users

  - **Without sudo: just do**
    - `vncserver :1`
    - `vncserver -kill :1`
    - Replace 1 with whatever number is available

ACCESSING JLAB IFARM WITH VIRTUAL NETWORK COMPUTING (VNC)

Advanced:

- VNC port forward tunneling through login.jlab.org and local access
  - For Mac: https://jlab.servicenowservices.com/sp?sys_kb_id=63d78ae2dbc888107d37365e7c961986&id=kb_article_view&sysparm_rank=2&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361
  - For Windows: https://jlab.servicenowservices.com/sp?sys_kb_id=ab0602eadb844810ee4a3889fc961942&id=kb_article_view&sysparm_rank=6&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361
  - For Linux:
    - Similar to Mac, or follow these instructions
    - <vnc port> = 5900 + VNC session ID number (typically 1 is available) = 5901
    - `ssh -L <vnc port>:localhost:<vnc port> <username>@<remote host name>`
    - `vncviewer localhost:<vnc port>`
    - Advanced, 3 terminal system to access ifarm with login.jlab.org tunnel included:
      (recommend to use ProxyJumps instead)
      (<forwarding port> is anything free other than 22)

Terminal 1: `ssh -L <forwarding port>:ifarm:<ssh port> <username>@login.jlab.org`
Terminal 2: `ssh -p <forwarding port> -L <vnc port>:localhost:<vnc port> <username>@localhost`
Terminal 3: `vncviewer localhost:<vnc port>`

Jefferson Lab

Accessing JLab ifarm with Virtual Network Computing (VNC)

Advanced:

- VNC port forward tunneling through login.jlab.org and local access

  - For Mac: https://jlab.servicenowservices.com/sp?sys_kb_id=63d78ae2dbc888107d37365e7c961986&id=kb_article_view&sysparm_rank=2&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361

  - For Windows: https://jlab.servicenowservices.com/sp?sys_kb_id=ab0602eadb844810ee4a3889fc961942&id=kb_article_view&sysparm_rank=6&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361

  - For Linux:
    - Similar to Mac, or follow these instructions
    - <vnc port> = 5900 + VNC session ID number (typically 1 is available) = 5901
    - `ssh -L <vnc port>:localhost:<vnc port> <username>@<remote host name>`
    - `vncviewer localhost:<vnc port>`

    - Advanced, 3 terminal system to access ifarm with login.jlab.org tunnel included:
      (recommend to use ProxyJumps instead)
      (<forwarding port> is anything free other than 22)

Example alias for bashrc. Assumes you ran `vncserver :2` on the ifarm2402 server and your local username is the same as the jlab one (otherwise tag <username>@ in front of login.jlab.org and localhost) and assumes ports 2201 and VNC:2 are available:

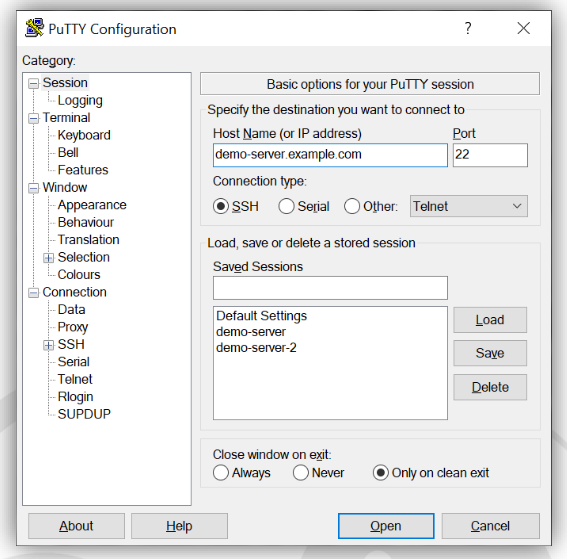
- alias tunnelscilogin="ssh -L 2201:ifarm2402:22 login.jlab.org"
- alias tunnelifarm="ssh -p 2201 -L 5902:localhost:5902 localhost"
- alias ifarmvnc="vncviewer localhost:5902"

Getting Started at the Lab

Slides cover:

- Account activation and access
- Accessing scientific computing resources (ifarm)
- **<u>Learning to navigate and use Linux computers</u>**
- Introduction to scientific computing resources
  (Geant4 simulation and ROOT data analysis)

Learning to navigate and use Linux



Specify the destination you want to connect to

Host Name (or IP address): demo-server.example.com

Port: 22

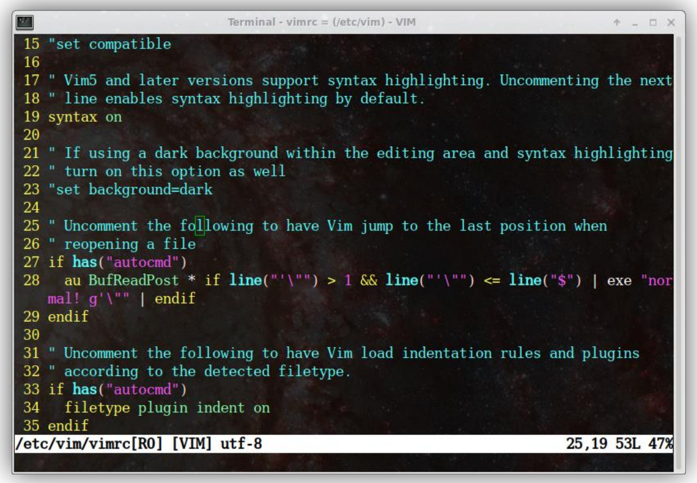
Connection type: [✓ SSH] Serial Other: Telnet

Load, save or delete a stored session

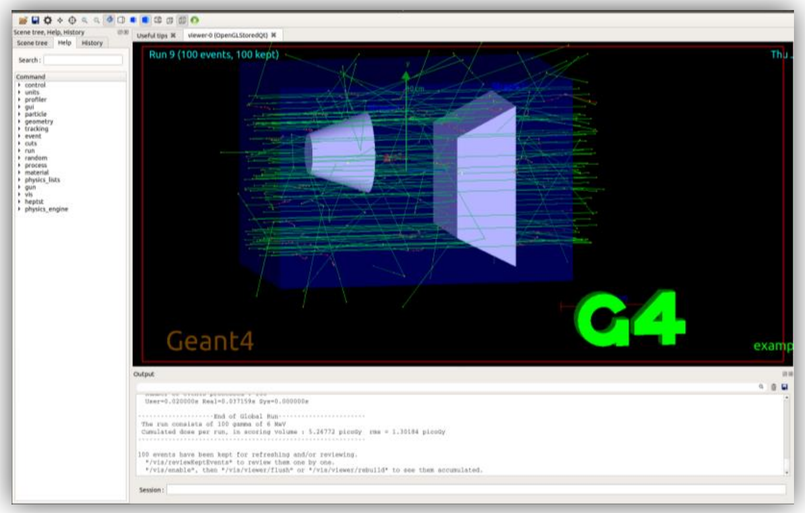
Saved Sessions:

Default Settings
demo-server
demo-server-2

Close window on exit: Always Never [✓ Only on clean exit]



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "nor
mal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                  25,19 53L 47%
```



```
 User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Access High Performance Computing (HPC) Cluster → Set up scientific computing software on Linux systems → Run physics experiments, simulations, and data analysis

This section

LEARNING LINUX

In this set of slides we will go over how to utilize Linux for scientific computing:

- Shell scripting
- Text editors
- Environment setup
- Basic scripting example

Shell Scripting

Linux terminals utilize interactive "shell" environments:

- There are many acceptable "shells" pre-installed on most linux systems
- The Bourne Again Shell "BASH" is most popular
- Jefferson Lab scientific computing systems default to the "C" shell, called "tcsh", so called because it is "like" the C language
  - Please read the Ubuntu website beginner's guide: https://ubuntu.com/tutorials/command-line-for-beginners#1-overview
  - You may find the FreeCodeCamp guide additionally informative: https://www.freecodecamp.org/news/command-line-for-beginners/

Shell Scripting

Linux terminals utilize interactive "shell" environments:

- There are many acceptable "shells" pre-installed on most linux systems
- The Bourne Again Shell "BASH" is most popular
- Jefferson Lab scientific computing systems default to the "C" shell, called "tcsh", so called because it is "like" the C language
- Shells give access to the file system and many system-wide installed commands, such as:
  - cd = change directory, move from one folder to another
  - ls = list the contents of the directory
  - pwd = state the full path of the current working directory (path working directory = pwd)
  - cp = copy the first argument file to the location of the second argument
  - mv = move the first argument file to the location of the second argument
  - rm = remove, delete the file passed in as an argument
  - ln = create a symbolic link, "shortcut"
  - cat = concatenate, display the contents of the file passed as an argument to the screen
  - which = give the full path of the location for a command passed as an argument
  - echo = print back to the screen, this is useful in scripting for printing results to the screen
  - grep = a powerful program for searching text strings for sub-strings, and much more
  - ps = list all processes running in this terminal shell session
  - top = list all processes running on the computer, with useful information and live updating
  - less = like cat, but lets you scroll around and do text searching, like Vim text editor
  - tmux and screen = convenient shell session preservation and re-attachment tools

Shell Scripting

Linux terminals utilize interactive "shell" environments:

- There are many acceptable "shells" pre-installed on most linux systems
- The Bourne Again Shell "BASH" is most popular
- Jefferson Lab scientific computing systems default to the "C" shell, called "tcsh", so called because it is "like" the C language
- Shells give access to the file system and many system-wide installed commands
- **Now, login to the JLab ifarm to proceed to practice using text editors**
    - **ssh to through the login.jlab.org portal, then on to ifarm**
    - **Find your group's work folder and make yourself a sub-directory:**
        - **`cd /work/<group name>/`**
        - **`mkdir <my username>` (no < > brackets)**
        - **`cd <my username>`**
        - **`touch helloworld.txt`**
        - **`<editor> helloworld.txt` where <editor> = vim, gedit, or emacs as you prefer – described on the follow slides**

TEXT EDITORS

Text Editors – Vim vs. Gedit vs. Emacs:

- StackShare has a guide to the three and their differences: https://stackshare.io/stackups/emacs-vs-gedit-vs-vim
- Vim (my preferred): https://thevaluable.dev/vim-commands-beginner/ - Or just type the `vimtutor` command at the terminal!
- Gedit: https://help.gnome.org/users/gedit/stable/index.html.en
- Emacs: http://www.jesshamrick.com/2012/09/10/absolute-beginners-guide-to-emacs/
- Learn about the three of them (or throw in nano as a fourth option) and pick the one you would like to learn
  - This is a personal choice
  - You can stick to GUI based Integrated Development Environments (IDE's) line Eclipse or VSCode if you chose
  - However, knowing how to use one of these is critical
  - Just in case you need to access a system with no GUI or software installation capabilities or need to quickly edit code

ENVIRONMENT SETUP

Setting up your shell environment:

- There is a "rc" = "run commands" file for each kind of shell
  - tcsh uses ~/.tcshrc
  - They are executed at each new shell open, new terminal, or new remote login
  - They can be relied upon, along with a ~/.login file and other more niche configuration files, to set up your environment
- For the JLab ifarm: to successful set up a user environment that gives access to ROOT and Geant4
  - Load modules stored in the /cvmfs system
  - /cvmfs is the CernVM File System, which gives global access to scientific collaborations to scientific computing software
  - Edit your .tcshrc on the jlabl1 or ifarm9 server to contain, at the bottom:

```
set hostnamex="`hostname`"
if ( $hostnamex =~ *'farm'* ) then
   module use /cvmfs/oasis.opensciencegrid.org/jlab/scicomp/sw/el9/modulefiles
   module load root
   setenv SIM_HOME /cvmfs/oasis.opensciencegrid.org/jlab/geant4
   module use "${SIM_HOME}"/ceInstall/modulefiles
   module load sim
endif
```

BASIC SCRIPTING EXAMPLE

Scripting in Linux allows for complex chains of shell commands to accomplish your goals

- Let's work on an example to learn the basics of using Bash, tcsh, Python, and ROOT on the JLab scientific computing systems
- Go to your "work" folder: `cd /work/<group name>/<my username>`
- Bash – parse an example text, skip lines and print each 45th line to the screen

➢ Open a blank text file print_first_and_every_45th.sh with your text editor
➢ Paste in the following bash code

```
#!/bin/bash
# Check if a file was provided as an argument
if [ "$#" -ne 1 ]; then
   echo "Usage: $0 filename"
   exit 1
fi
# File to read
file=$1
# Check if the file exists
if [ ! -f "$file" ]; then
   echo "File not found!"
   exit 1
fi
# Counter for lines
count=1
# Read the file line by line
while IFS= read -r line
do
   # Check if the current line number is the first line or every 46th line thereafter
   if [ $(( (count - 1) % 45 )) -eq 0 ]; then
      echo "$line"
   fi
   # Increment the counter
   count=$((count + 1))
done < "$file"
```

➢ chmod +x print_first_and_every_45th.sh
➢ ./print_first_and_every_45th.sh yourfile.txt

To generate a text file to parse, first execute the following (from the ifarm or your local computer, depending on vim version number):

- cat /usr/share/vim/vim82/tutor/tutor >> yourfile.txt

BASIC SCRIPTING EXAMPLE

Scripting in Linux allows for complex chains of shell commands to accomplish your goals

- Let's work on an example to learn the basics of using Bash, tcsh, Python, and ROOT on the JLab scientific computing systems
- Go to your "work" folder: `cd /work/<group name>/<my username>`
- tcsh – parse an example text, skip lines and print each 45th line to the screen

  - ➢ Open a blank text file print_first_and_every_45th.tcsh with your text editor
  - ➢ Paste in the following tcsh code

```
#!/bin/tcsh
# Check if a file was provided as an argument
if ( $#argv != 1 ) then
   echo "Usage: $0 filename"
   exit 1
endif
# File to read
set file = $argv[1]
# Check if the file exists
if ( ! -f $file ) then
   echo "File not found!"
   exit 1
endif
# Counter for lines
set count = 1
# Read the file line by line
foreach line ( `cat $file` )
   # Check if the current line number is the first line or every 46th line thereafter
   if ( ( $count - 1 ) % 45 == 0 ) then
      echo $line
   endif
   # Increment the counter
   @ count++
end
```

  - ➢ chmod +x print_first_and_every_45th.tcsh
  - ➢ ./print_first_and_every_45th.tcsh yourfile.txt

Basic Scripting Example

Scripting in Linux allows for complex chains of shell commands to accomplish your goals

- Let's work on an example to learn the basics of using Bash, tcsh, Python, and ROOT on the JLab scientific computing systems
- Go to your "work" folder: `cd /work/<group name>/<my username>`
- Python – parse an example text, skip lines and print each 45th line to the screen

  - ➢ Open a blank text file print_first_and_every_45th.py with your text editor
  - ➢ Paste in the following python code

```
import sys
def print_first_and_every_45th_line(filename):
    try:
        with open(filename, 'r') as file:
            for count, line in enumerate(file, start=1):
                if (count - 1) % 45 == 0:
                    print(line.strip())
    except FileNotFoundError:
        print(f"File {filename} not found!")
if __name__ == "__main__":
    if len(sys.argv) != 2:
        print(f"Usage: {sys.argv[0]} filename")
        sys.exit(1)
    filename = sys.argv[1]
    print_first_and_every_45th_line(filename)
```

  - ➢ python print_first_and_every_45th.py yourfile.txt

Basic Scripting Example

Scripting in Linux allows for complex chains of shell commands to accomplish your goals

- Let's work on an example to learn the basics of using Bash, tcsh, Python, and ROOT on the JLab scientific computing systems
- Go to your "work" folder: `cd /work/<group name>/<my username>`
- ROOT (interpretted C++) – parse example text, skip lines and print each 45th line to the screen

  - Open a blank text file printFirstAndEvery45th.C with your text editor
  - Paste in the following ROOT C code

```
#include <iostream>
#include <fstream>
#include <string>
int printFirstAndEvery45th(const char* filename) {
   std::ifstream file(filename);
   if (!file.is_open()) {
      std::cerr << "File not found!" << std::endl;
      return 0;
   }
   std::string line;
   int count = 1;
   while (std::getline(file, line)) {
      if ((count - 1) % 45 == 0) {
         std::cout << line << std::endl;
      }
      count++;
   }
   file.close();
   return 1;
}
```

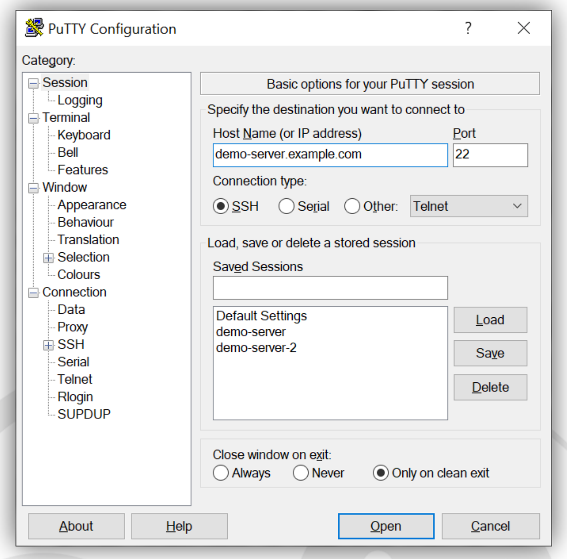
  - root -l 'printFirstAndEvery45th.C("yourfile.txt")'

GETTING STARTED AT THE LAB

Slides cover:

- Account activation and access
- Accessing scientific computing resources (ifarm)
- Learning to navigate and use Linux computers
- **<u>Introduction to scientific computing resources</u>**
  (Geant4 simulation and ROOT data analysis)

Introduction to Scientific Computing Work



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): demo-server.example.com   Port: 22

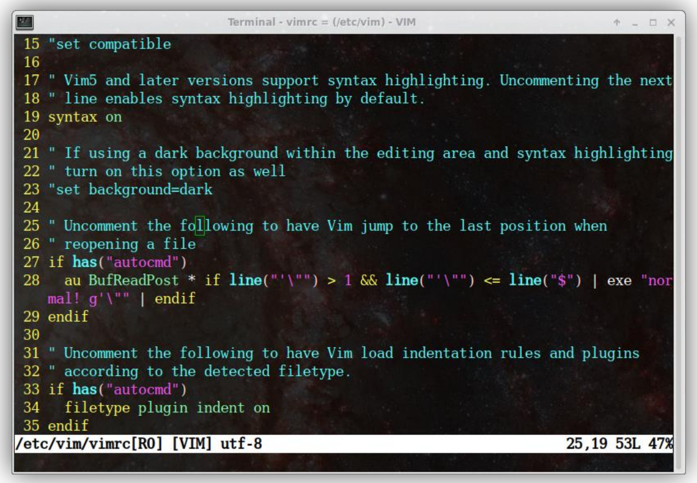
Connection type: [✓ SSH] Serial Other: Telnet

Load, save or delete a stored session

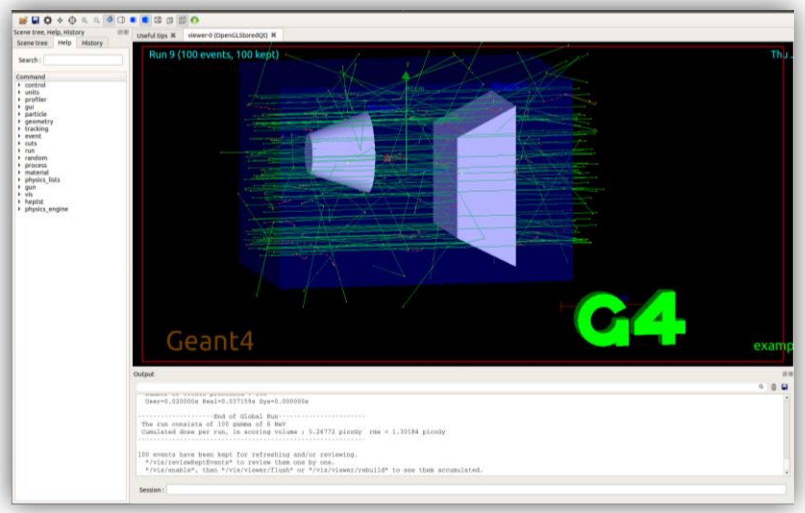
Saved Sessions

Default Settings
demo-server
demo-server-2

Close window on exit: Always Never [✓ Only on clean exit]



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "nor
mal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                25,19 53L 47%
```



```
User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Access High Performance Computing (HPC) Cluster $\rightarrow$ Set up scientific computing software on Linux systems $\rightarrow$ Run physics experiments, simulations, and data analysis

This section $\nearrow$

Jefferson Lab

U.S. DEPARTMENT OF ENERGY | Office of Science

JSA

GETTING STARTED WITH GEANT4

To utilize Scientific Computing:

- Run Geant4 simulations
- Analyze data with ROOT
- Track software with Git

What is Geant4?

- General getting started guide: https://geant4-userdoc.web.cern.ch/UsersGuides/IntroductionToGeant4/html/index.html
- Advanced developer guide: https://geant4-userdoc.web.cern.ch/UsersGuides/ForApplicationDeveloper/html/index.html

/cvmfs/oasis.opensciencegrid.org/jlab/geant4/1.2/almalinux9-gcc11/geant4/11.2.1/source/examples

Want to install Geant4 locally on your home computer?

- https://geant4-userdoc.web.cern.ch/UsersGuides/InstallationGuide/html/

GEANT4 AS A PHYSICS LABORATORY

Geant4 emulates the physical world

- Samples the phase space of possible interactions for particles
- Treats them the same as if they were propagating through physical material
- The advantage is you get to know the kinematics along the way
- Need to be careful about how the particle "hit" information is recorded and stored to output files
- Geant4 takes short-cuts, only report aggregated results when crossing a geometry boundary, for example
- GDML (markup language – XML) allows for convenient run-time defined geometries if implemented

Examples included in the Geant4 installation cover nuclear physics, accelerator design, and medical imaging and treatment scenarios

- From the JLab ifarm: /cvmfs/oasis.opensciencegrid.org/jlab/geant4/1.2/almalinux9-gcc11/geant4/11.2.1/source/examples

GETTING STARTED WITH ROOT

To utilize Scientific Computing:

- Run Geant4 simulations
- Analyze data with ROOT
- Track software with Git

What is ROOT?

- General getting started guide: https://root.cern.ch/root/htmldoc/guides/primer/ROOTPrimer.html
- Learning scripting section https://root.cern.ch/root/htmldoc/guides/primer/ROOTPrimer.html#learn-c-at-the-root-prompt

/cvmfs/oasis.opensciencegrid.org/jlab/scicomp/sw/el9/root/6.30.06-gcc11.4.0/tutorials

Want to install ROOT locally on your home computer?

- https://root.cern/install/#install-via-a-package-manager

ROOT is useful for interactive scripting – to debug or quickly investigate data
ROOT is useful as a data processing system for DAQ, including real-time analysis capabilities

Getting Started with Git

To utilize Scientific Computing:

- Run Geant4 simulations
- Analyze data with ROOT
- Track software with Git

What is Git?

- Git is a file management and version control system: https://git-scm.com/book/en/v2/ (Getting started guide)
- Git was designed in 2005 to support Linux kernel development by Linus Torvalds:
- Github is just a website that hosts Git repositories, like Bitbucket, Gitlab, or your local machine operating as a file server

Want to use Git?

- https://code.jlab.org is the preferred remote Git management system at JLab now (Gitlab backend, similar to GitHub)
- The `git` command line program can do version control locally, even if you do not track projects with a remote server system
- https://github.com is the standard free service

TYPICAL GIT VERSION CONTROL EXAMPLE

To create a local blank repository:

- $ git init
- $ cat "Hello World" >> README.md
- $ git add README.md
- $ git commit -m "initial commit"

Before pushing or pulling any commits to/from a remote repository, set up your git ssh key config first:

- Github: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account
- Gitlab: https://docs.gitlab.com/ee/user/ssh.html
- More intro to Git: https://www.atlassian.com/git/tutorials/setting-up-a-repository

To turn an existing project into a remote git repository:

- Do the same steps above for creating a local repository, then attach it to a remote host like Gitlab as follows:
- Create an empty repository on Gitlab: fill out information on https://github.com/new with <reponame>
- $ git remote add origin git@code.jlab.org:<username or organization name>/<reponame>.git
- $ git remote –v *(verify)*
- $ git branch –a *(to find default initial branch-name, can be changed with `git branch -m <branch-name> <new branch-name>`)*
- $ git config --global user.name <username>
- $ git config --global user.email <email>
- $ git push --set-upstream origin <branch-name>
- $ git remote show origin *(verify)*

To clone (download) an existing remote git repository:

- $ git clone git@code.jlab.org:<username or organization name>/<reponame>.git

Jefferson Lab

Typical Git Version Control Example

Example sequence of commands to check git configuration and safely commit changes:

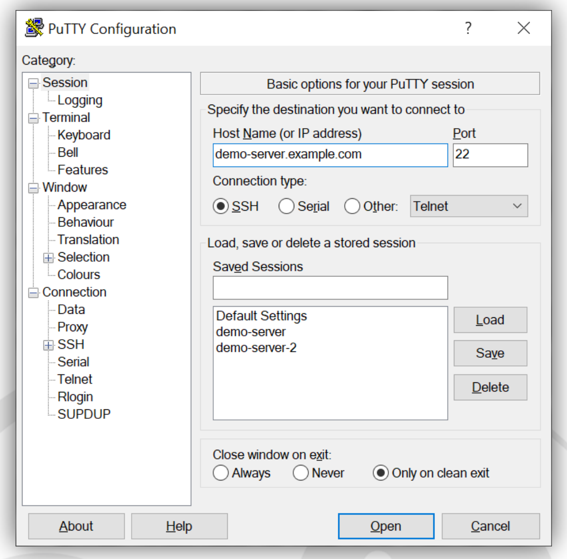
- git remote show origin *(verify your local repository is connected to the remote you think it should be)*
- git branch -a
- git fetch
- git pull --ff-only *(use this to avoid unwanted merge conflicts and better monitor recent remote changes)*
- git status *(can also pass the folder of file of interest as an argument to simplify outputs)*
- git log -N (N = number of recent commit logs to read)
- git diff [some local file that has changes]
- git blame [some local file you want change history of]
- git add [some local file that has changes you want to push]
- git commit --author=[your git remote server username]
- git push -u [origin name] [your branch name] *(the origin names – typically "origin" or "remote" - are set in **.git/config**)*

Additional challenges to plan to address before using Git to contribute to a project include:

- Using the right branch and managing issues and pull requests with collaborators
- Setting up username and password settings for remote git server access
- Resolving push/pull/merge conflicts
- Writing good commit messages

Friday, August 9, 2024

Introduction to Scientific Computing



Specify the destination you want to connect to

Host Name (or IP address): demo-server.example.com
Port: 22

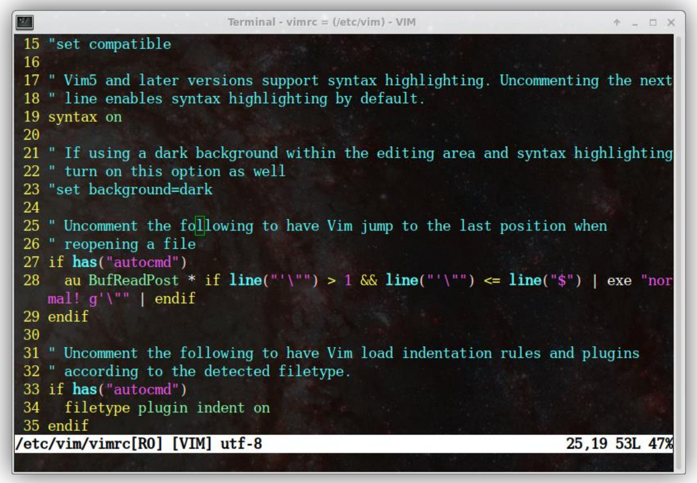
Connection type: [✓ SSH] Serial Other: Telnet

Load, save or delete a stored session

Saved Sessions:
- Default Settings
- demo-server
- demo-server-2

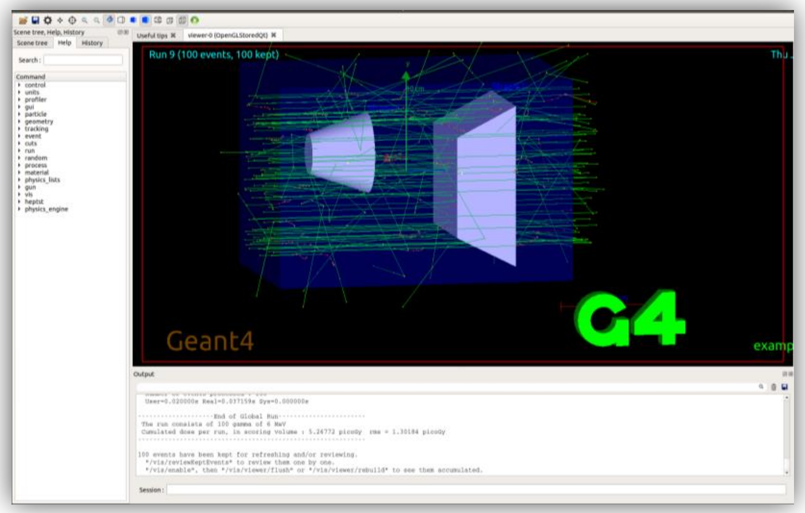
Close window on exit: Always Never [✓ Only on clean exit]

Access High Performance Computing (HPC) Cluster



```
"set compatible

" Vim5 and later versions support syntax highlighting. Uncommenting the next
" line enables syntax highlighting by default.
syntax on

" If using a dark background within the editing area and syntax highlighting
" turn on this option as well
"set background=dark

" Uncomment the following to have Vim jump to the last position when
" reopening a file
if has("autocmd")
  au BufReadPost * if line("'\"") > 1 && line("'\"") <= line("$") | exe "normal! g'\"" | endif
endif

" Uncomment the following to have Vim load indentation rules and plugins
" according to the detected filetype.
if has("autocmd")
  filetype plugin indent on
endif
/etc/vim/vimrc[RO] [VIM] utf-8                                    25,19 53L 47%
```

Set up scientific computing software on Linux systems



```
 User=0.020000s Real=0.037159s Sys=0.000000s

--------------------End of Global Run-----------------------
 The run consists of 100 gamma of 6 MeV
 Cumulated dose per run, in scoring volume : 5.24772 picoGy  rms = 1.30184 picoGy
------------------------------------------------------------

100 events have been kept for refreshing and/or reviewing.
  "/vis/reviewKeptEvents" to review them one by one.
  "/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

Run physics experiments, simulations, and data analysis

Feel free to ask any follow up questions:
cameronc@jlab.org

Jefferson Lab

U.S. DEPARTMENT OF ENERGY | Office of Science

JSA