JLAB SCIENTIFIC COMPUTING ACCESS

ategory.		
Category: Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection	Basic options for your Pu' Specify the destination you want to Host Name (or IP address) demo-server example com Connection type:	TTY session connect to Port 22 Telnet ~
Connection Connection Poata Proxy SSH SSH Serial Telnet -Rlogin SUPDUP	Default Settings demo-server demo-server-2	Load Saye Delete
About Help	Close window on egit: Always Never On	nly on clean exit

Access High Performance Computing (HPC) Cluster



Set up scientific computing software on Linux systems





Run physics experiments, simulations, and data analysis

Cameron Clarke



Radiation Detector & Imaging Group, Nuclear Physics, Jefferson Lab

Biomedical Research & Innovation Center

BRIC









OUTLINE

- **Slide 4**: Logging in to JLab computing resources
- **Slide 11**: Navigating and manipulating files
- Slide 16: Intro to Git, Computing Software
 - Git shared code
 - Shared group installations
 - CVMFS
 - Conda
 - Docker

- Hands on exercises to familiarize yourself with all these things!
 - Slide 33: Logging In
 - Slide 52: Navigating and manipulating files
 - Slide 71: Scripting, Software, and Git

<u>JLab Sci-Comp Knowledge</u> <u>Base - Getting Started</u>



Before you can get an account:

- 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



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

For JLab Farm computing access:

- Need to request permissions from your (experimental hall, theory group, detector group) computing coordinator
- Need to acquire multi-factor authentication (MFA) credentials from the Computer Center Helpdesk

Computing Coordinators Knowledge Base Article:

https://jlab.servicenowservices.com/kb?id=kb_article_view&sysparm_article=KB0014686



Wednesday, May 21, 2025

LECTURE: ACCESSING LAB RESOURCES

Access High Performance Computing (HPC) Cluster



Set up scientific computing software on Linux systems

This section

Run 9 (100 events, 100 kep

-End of Global Hum-The run consists of 100 gamma of 4 MaV Camslated does per run, is scoring volume 1 5,24772 ploudy rms = 1,30184 ploud

vents have been kept for refreshing and/or reviewing. is/reviewEepElvents* to review them one by one. is/readies, then */vis/viewer/flush* or */vis/viewer/

Scene tree Help

Command - control - units - particle -

Run physics experiments, simulations, and data analysis







- Common User Environment (CUE) accounts enable access to many systems at JLab
 - o JLab Insight Web Portal
 - Training (<u>https://misportal.jlab.org/training/people/srl</u>)
 - JList (staff directory)
 - Help-desk, library, and facilities tickets (<u>https://jlab.servicenowservices.com</u>)
 - Requisition system, etc.

Jeffer	son Lab		HOME	INSIGH	T MY LINKS	Welcome C Search	ameron Clarl		OUT Q
Menu	LMS - PERSON SRL	SEAR	CH						
Home Calendar	Person Search								
Search Skills	Re	quired S	kills for Cameron Clar	ke (camero	onc) as of 6/19/20	24 11:02 p	om		
Competencies Courses			Bu	ild SRL in JTA					
Classes Training History	Show 10 v entries					Filter By:			
Skill History			Required Co	mpetencies (s	kill sets)				
Competency Status			Competen	су				^ Sta	itus ^
Revocations	Radiological Worker I						Show	details COM	PLETE
Taken/Scheduled Classes	Showing 1 to 1 of 1 entries						First Pre	vious 1 Ne	xt Last
Other Links	Show 10 v entries					514 B	[
Help	Show 10 V entries					Filter By:			
Training Office			Required Individual S	kills (not part o	of a competency)				
ATL	Skill	Code ^	Category ^	Last	Course Take	en ^	Months	Expiration	Status
SRL Web Brood Training				Acquired ^			Since ^	^	^
vveb-based Training	Timekeeping Principles	ADM001	ADMINISTRATIVE/QUALITY	3/5/2024	Timekeeping Principles	3	3	N/A	Current
Provide Feedback	Salaried Exempt Mechanics	ADM004	ADMINISTRATIVE/QUALITY	3/5/2024	Timekeeping Mechanic	s	3	N/A	Current



misportal.jlab.org/portal/insight



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

- Common User Environment (CUE) accounts enable access to many systems at JLab
 - JLab Insight Web Portal
 - Training (<u>https://misportal.jlab.org/training/people/srl</u>)
 - JList (staff directory)
 - Help-desk, library, and facilities tickets
 - Requisition system, etc.

- \circ $\$ Ask your sponsor or support staff at the lab for guidance
- Email <u>helpdesk@jlab.org</u> for computing assistance
- Email <u>library@jlab.org</u> for journal/book access questions
- Use the *servicenow* portal for facilities and other tickets



https://jlab.servicenowservices.com



- Common User Environment (CUE) accounts enable access to many systems at JLab
 - JLab Insight Web Portal
 - Training (<u>https://misportal.jlab.org/training/people/srl</u>)
 - JList (staff directory)
 - Help-desk, library, and facilities tickets (<u>https://jlab.servicenowservices.com</u>)
 - Requisition system, etc.
 - \circ ~ VPN or VDI for Insight protected pages
 - Gitlab (<u>https://code.jlab.org/</u>)
 - To access interactive computer sessions a Multi-Factor Authentication token (MFA) is required
 - Mobile PASS SAS is the JLab login-portal access method – Help Desk distributed
 - o Phone app based
 - Can ask for a USB stick button as well

🤜 Homepage 🛛 🕇 🌐	Explore / Projects	
다 않 또 Q Search or go to	Explore projects	New project
Explore	Most starred Trending Active Inactive All	
Projects	Search or filter results	Q
 Gloups CI/CD Catalog Topics 	EPSCI / Hydra / Hydra_containers 🔠	☆ 5 😵 0 \$\$0 D 0 Updated 2 months ago
X Snippets	Sawatzky, Brad / Workshop Sept2024	☆ 2 😵 0 🐎 0 D 0 Updated 8 months ago
	J FEDAQ / drivers / jvme Φ	☆ 2 😵 0 🖏 0 D 0 Updated 1 week ago
	FEDAQ / drivers / faV3 ⊕ 型 Other Library for the Jefferson Lab fadc250 V3	☆1 撃0 \$30 ₪0 Updated 2 months ago
	HallB / XYHodo 🌐 Analysis tools for the 2 layer XYHodoscope	☆1 ೪0 \$%0 DP2 Updated 3 months ago
	S DetImg / Streaming Readout PET / StreamingPET StreamingPET with PetIroc ASIC	☆1 ೪ 0 \$% 0 D 0 Updated 3 days ago
	FEDAQ / drivers / ti ① 堕 BSD 3-Clause "New" or "Revised" License Trigger Interface (pipeline) Library	☆1 ೪0 th 0 D 0 Updated 5 days ago
	R PhysDiv / EIC-RUCIO / rucio-eic-docs	☆1 ೪0 ₺0 ₽0 Updated 1 month ago

https://code.jlab.org/



- Common User Environment (CUE) accounts enable access to many systems at JLab
 - JLab Insight Web Portal Ο
 - Training (<u>https://misportal.jlab.org/training/people/srl</u>) Ο
 - JList (staff directory) Ο
 - Help-desk, library, and facilities tickets (<u>https://jlab.servicenowservices.com</u>) Ο

Horizon

CUE RHEL7

- Requisition system, etc. Ο
- VPN or VDI for Insight protected pages Ο
- Gitlab (<u>https://code.jlab.org/</u>) Ο
- Multi-Factor Authentication (MFA) login Ο
 - Public JLab computers (jlabl1, etc.) •
 - VMWare Horizon VDI (<u>https://vdi.jlab.org/</u>) •
 - Interactive Farm (ifarm) •
 - Globus •
 - Userweb webpage editing •
- Secondary secure login portals Ο
 - Hall Gateway •
 - Accelerator Gateway •
 - JupyterHub (Google Auth) •



ACCESSING JLAB FILESYSTEMS

- Your personal computer can mount JLab network drives and ifarm job submission tools
 - o Requires OS management by JLab central CUE system
 - Must be on-site
 - Prevents use of local super user permissions
 - Ask the Helpdesk if you want to do this
- Otherwise Accessing JLab resources requires remote access via JupyterHub, VDI or SSH
 - JupyterHub Python Jupyter notebook system, with pre-installed virtual environment kernel options
 - Need to request Google Authenticator MFA token
 - VDI JLab Virtual Desktop Infrastructure (VDI) Red Hat Enterprise Linux 9 (RHEL) Environment
 - Need to request access to non-Windows Virtual machine ask Helpdesk for RHEL9 access
 - Preferred Access Method: <u>SSH Secure Shell command line interface (CLI) login</u>
 - X Windows Forwarding
 - Mac Terminal + XQuartz
 - \circ Windows PuTTy + XMing
 - Linux Terminal
 - SSH Tunneling Manually or with ProxyJump
 - File transfers With manual tunneling, ProxyJump, Userweb, or Globus system



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
- SSH or Linux RHEL9 VDI is recommended (can persist)

Hands-on session slides give detailed instructions for accessing ifarm with all systems:

Secure Shell (SSH) Protocol and Transferring Files

- Linux
- JLab's Virtual Desktop Infrastructure (VDI), from either html web app or desktop app
- Windows
- Mac
- JupyterHub
- ➤ Tomorrow
 - Interactive farm (ifarm) noded with slurm access
 - Farm batch submission access

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

&sysparm_article=KB0015066



LECTURE: NAVIGATE/MANIPULATE FILES

legory:
legory: Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy SSH Serial Tehent Rlogin SUPDUP

Access High Performance Computing (HPC) Cluster





Run physics experiments, simulations, and data analysis

This section







NAVIGATING JLAB IFARM

JLab's shared HPC:

- Computing cluster "farm"
- Has interactive capability through "ifarm"
- Two nodes (ifarm2401, ifarm2402)
- Provides access points for testing, etc. before sending off to production farm
- Interactive nodes:
 - AMD EPYC 9554 (Zen 4 "Genoa")
 - 256 threads (2 sockets × 64 cores × 2 threads/core)
 - 3.1 GHz base / 3.75 GHz max
 - 1.5 TB memory
 - 28 TB striped NVMe /scratch
 - HDR (200 Gb/s) IB



* Backed to tape. Excludes small files.



NAVIGATING JLAB IFARM

- Linux CLI functionality
 - Basic Unix Commands what are they?
 - Unix philosophy (from Wikipedia):
 - Use of plain text for storing data
 - A hierarchical file system
 - Treating devices and certain types of inter-process communication (IPC) as files
 - Use of <u>a large number of software tools</u>, small programs that can be strung together with CLI / as opposed to using a single monolithic program that includes all of the same functionality
- JLab network file system features
 - /group, /work, /home, /volatile,/scratch, /mss tapes, /cache
 - User and group allocations, fairshare between the halls
 - CVMFS as a global shared filesystem, CERN and many other science facilities
 - File Transfer: SCP, SFTP, Globus
- Text editors
 - Vim, Gedit, Emacs, Nano, etc.
- IDEs (Integrated Development Environment)
 - VSCode, Eclipse, IntelliJ, etc.

(https://halldweb1.jlab.org/wiki/index.php/How_to_Setup_Visual_Studio_Code_for_the_ifarm)



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
- Most commands have manuals: type `man <command>` without brackets, to learn more, or `<command> --help` usually works too
- Any command accessible from the CLI can also be utilized in the exact same way inside of a script, enabling efficient operations
- Shells give access to the file system and many system-wide installed commands, such as:
 - cd = change directory, move from one folder to another
 - Is = 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 (be very careful with this command!)
 - In = 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
 - man = show the manual entry for this command



SSH PASSKEYS

- Key and lock pair, where you generate both and distribute the lock to your systems
 - <u>https://blakesmith.me/2010/02/08/understanding-public-key-private-key-concepts.html</u>
 - Useful as an intra-network stable password, git authentication (main point for our later hands-on session), and multi-system login uniformity strategy
 - Generate them on any system, ideally one unique one for each system
 - You must place them on systems you want to access with them
 - Utilize the known_hosts system to save some time and improve your security confidence -<u>https://www.baeldung.com/linux/public-key-known_hosts</u>
- You must maintain their security
 - Verify they have the correct permissions <u>https://www.tecmint.com/set-ssh-directory-permissions-in-linux/</u>
 - Never share your private key
 - Ideally use unique keys for each system, especially unsecured ones
 - Use a password on each one (it is possible to have a no-password key pair, but not safe!)
 - Pay attention to known_hosts authentication notifications for changed server identities (especially for untrusted remote servers)



Wednesday, May 21, 2025

LECTURE: SCRIPTING/SOFTWARE & GIT

Alogory.		
Session	Basic options for your PuTTY se	ession
- Logging	Specify the destination you want to conne	ct to
Keyboard	Host Name (or IP address)	Port
Bell	demo-server.example.com	22
Features	Connection type:	
Window		
Behaviour		ม ~
Translation	Load, save or delete a stored session	
Selection	Saved Sessions	
Colours		7
Data		
Proxy	Default Settings demo-server	Load
⊞ SSH	demo-server-2	Save
Serial		- 54 <u>v</u> c
- Leinet		<u>D</u> elete
SUPDUP		
		_
	Close window on exit:	
		loon ovit

Access High Performance Computing (HPC) Cluster





Run physics experiments, simulations, and data analysis









GETTING STARTED WITH GIT

What is Git?

- Git is a file management and version control system: <u>https://git-scm.com/book/en/v2/</u> (Getting started guide)
- <u>Git was designed</u> 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?

- <u>https://code.jlab.org</u> GitLab is the preferred remote Git management system at JLab now (Gitlab backend, similar to GitHub, better than BitBucket)
- The `git` command line program can do version control locally, even if you do not track projects with a remote server system
- Git tools are supported on all OS and in many IDE systems
- There is a "gitbash" terminal emulator on Windows (or you can use Linux Subsystem for Windows to run Linux in Windows 10/11)
- <u>https://github.com</u> is the standard free service





GIT VERSION CONTROL BASICS

- Git skills to improve your analysis and software development work:
 - Create a local blank repository
 - Turn an existing project into a remote git repository
 - Clone (download) an existing remote git repository
 - Check and change git configuration and safely commit code changes to shared repository
- Additional skills to learn before using Git to contribute to a shared 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

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

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



ACCESSING, SHARING, AND TRACKING SOFTWARE

- Software is easily stored:
 - CernVM File System (CVMFS) computing resource for standard packages
 - JLab people share code in /work, /group, /mss, GitLab, etc.
 - Git: code.jlab.org GitLab (preferred), GitHub.com/JeffersonLab
 - Accelerator and hall Logbooks, collaboration elogs and DocDBs
- Conda and Virtual Environments
 - Subject to some version restrictions at JLab
 - (Ana)conda and venv can enable rapid software environment build/rebuild and sharing
- FAIR = Findable, Accessible, Interoperable, and Reusable
 - You will forget how you did your analysis (it doesn't take long) you will thank yourself
 - Always back up your code, your LaTeX files (OverLeaf), your figures, and even your data
 - Add dates, version information, comments, and tag numbers too (all git features)
 - Software versions aren't always backwards compatible and operating systems go end-of-life
 - Virtual machines (Compiled Java, <u>Docker Images</u>, VMWare) are very convenient time-capsule tools!



Wednesday, May 21, 2025

HANDS ON: ACQUIRING AN ACCOUNT

Category:		
Category: Category: Session Category: Category: Session Category:	Basic options for your PuTTY sess Specify the destination you want to connect Host Name (or IP address) demo-server example com Connection type:	sion to Port 22 ~
- Data - Proxy ■ SSH - Serial - Telnet - Rlogin - SUPDUP	Default Settings demo-server demo-server-2	Load Sa <u>v</u> e Delete
	Close window on exit: Always Never Only on cle	an exit

Access High Performance Computing (HPC) Cluster



Set up scientific computing software on Linux systems



Scene tree Help

Command - control - units - particle -

Run physics experiments, simulations, and data analysis

-End of Global Bun The run consists of 100 gamma of 6 Mav Camalated does per run, is scoring volume 1 5.24772 picudy rms = 1.10184 picoty

vents have been kept for refreshing and/or reviewing. is/reviewEepElvents* to review them one by one. is/readies, then */vis/viewer/flush* or */vis/viewer/







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.)



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



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



- 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 all users)
 - Ask for invitation to MobilePASS SAS MFA for general `login.jlab.org` access
 - Ask to be added to relevant 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 Google Authenticator OTP MFA access if you want it

≡ Scientific Computing ↑



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



username Getting Started Support Staff Members

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

	7					Welcome Ca	ameron Clark	ke SIGN C	DUT
Jeffei	rson Lab		HOME	INSIGH	T MY LINKS	Search	ı		Q
							(0 ENER	ĞΥ
C	LMS - PERSON SR	LSEAR	СН						
dar	Person Search								
		Required S	Skills for Cameron Clar	ke (camero	nc) as of 6/19/202	24 11:02 p	om		
tencies									
is			Ви	iild SRL in JTA					
ns ns	Show 10 v entries		Bu	ild SRL in JTA		Filter By:			
is s g History istory	Show 10 v entries		Bu Required Co	ild SRL in JTA	cill sets)	Filter By:			
is s g History istory tency Status	Show 10 v entries		Bu Required Co Competer	nild SRL in JTA	kill sets)	Filter By:		^ Sta	itus
g History istory itency Status ations	Show 10 v entries		Bu Required Co Competer	uild SRL in JTA	till sets)	Filter By:	Show of	^ Sta	itus PLETE
s s g History istory itency Status ations Scheduled s	Show 10 v entries Radiological Worker I Showing 1 to 1 of 1 entries		Bu Required Co Competer	uild SRL in JTA	dil sets)	Filter By:	Show of First Prev	^ Sta details COMP vious 1 Ne	atus PLETE xt Las
g History istory istory tency Status ations Scheduled is s	Show 10 v entries Radiological Worker I Showing 1 to 1 of 1 entries Show 10 v entries		Bi Required Co Competer	nid SRL in JTA	till sets)	Filter By: Filter By:	Show of First Prev	Sta Sta details COM ious 1 Ne	tus PLETE xt Las
g History stercy Status titons Scheduled s g Office	Show 10 v entries Radiological Worker I Showing 1 to 1 of 1 entries Show 10 v entries		Bu Required Co Competer	nild SRL in JTA	till sets) of a competency)	Filter By: Filter By:	Show of First Prev	State details COMF vious 1 Ne	itus PLETE xt Las
g History g History tency Status ations Scheduled s nks g Office	Show 10 v entries Radiological Worker I Showing 1 to 1 of 1 entries Show 10 v entries Skill	^ Code ^	Required Individual S Category ^	niid SRL in JTA empetencies (sł noy skills (not part o Last Acouired ^	dil sets) of a competency) Course Take	Filter By: Filter By:	Show of First Prev	Sta Sta COM Com I Ne Expiration A	ttus PLETE xt Las
g History istory istory itercy Status ations Scheduled is nks g Office ased Training	Show 10 v entries Radiological Worker I Showing 1 to 1 of 1 entries Show 10 v entries Skill Timekeeping Principles	Code ADM001	Required Individual S Category ^ ADMINISTRATIVE/QUALITY	skills (not part of Last Acquired ^	of a competency) Course Take	Filter By: Filter By:	Show of First Prev Months Since ^ 3	Statement Statement	ttus PLETE xt Las Statu

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

Jefferson Lab		ENERGY
	JLab Web Single Sign-On	
Enter your JLab username and passwo Username	d below to login to the Single Sign-On System. With this system, JLab users can login once to access numerous web Forgot your password? Need Help?	applications.
Password	me to re-enter my password when using new services	
Login Click here	to use a multi-factor authentication method	
PRIVACY AND SECURITY NOTICE JLAB STATUS	JEFFEF HELPD	RSON LAB IESK@JLAB.ORG

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

📑 Site 🔊 Group 🍈 Dashboard				
Jefferson Lab	HOME	A-Z INDEX	Welcom MY LINKS	Search
	NEWS BUSINESS INTELLIGENCE			
Computer Center Helpdesk Employee Self Service (ESS) Events At-A-Glance Human Resources (HR) LMS SRL On the Menu Timesheet	JLAB Calendar of Events NSTAR 2024 Conference (York, UK) (06 06/21) Workshop on Sustainable Software Infra for Advanced Nuclear Physics Computin 06/22) DOE EFE Decement Macage Visit (600)	5/17 - structure ig (06/20 -		F ACCESS STATUS
PERSONAL INFORMATION I Information Informatio Information Information Information Informat	 Accelerator Seminar - Alimohammed Ka (06/20 - 06/20) Theory Seminar - Dennis Bollweg (06/24) 	chwala 4 - 06/24)		

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



- Verify account access
 - Go to <u>www.jlab.org</u>, 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

Sign in https://jman.jlab.org	
Username	
Password	
	Cancel Sign in

https://jman.jlab.org/jpasswd



- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - 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 <u>helpdesk@jlab.org</u> automatically files a support ticket
 - Or you can file a helpdesk ticket manually at <u>https://jlab.servicenowservices.com/navpage.do</u>

Jefferson Lab Incident System	
Filter navigator	Self Service
合 Home	How Can We Help You?
★ UITF Additional Shielding	 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 = Cameron C
	Image: Number Opened on Subject Requestor P Image: Number
	Actions on selected rows

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



- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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)





- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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
- \circ $\;$ Otherwise follow instructions in subsequent set of slides



Jefferson Lab

- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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 ifarm.jlab.org access:

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



- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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 ifarm.jlab.org access:

- Ask the helpdesk staff to help you test your ifarm.jlab.org access by terminal on their work-station at the helpdesk
- \circ $\;$ 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: <u>https://jlab.servicenowservices.com/kb</u> <u>?id=kb_article_view&sysparm_article=KB0014685</u>
- To verify complete farm batch job access log in to ifarm, 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=22d 2c5db1b4a09506a9e85dae54bcbcc)

- 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 (<u>https://scicomp.jlab.org/docs/farm_slurm_batch_interactive_jobs</u>)
 Iefferson Laboration

- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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 <u>https://vdi.jlab.org</u> (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=dec 16b0ddb7f0410ee4a3889fc961944 ³¹

- Verify account access
 - Go to <u>www.jlab.org</u>, 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
 - If you need help: <u>helpdesk@jlab.org</u> tickets
 - Verify Office 365 access log in to <u>www.office.com</u> 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

	Jefferson Lab Exploring the Nature of Matter	
	Log In	
Usernar	me	
Passwor	rd	
	Log In	

Log in to <u>https://jupyterhub.jlab.org</u> – first using your common user environment (CUE) password – second using your Google Authenticator OTP code



Wednesday, May 21, 2025

HANDS-ON: ACCESSING LAB RESOURCES

X PuTTY Configuration	? ×
Category:	
Session Logging Terminal Heyboard Bell Features Window Appearance Bobatour	Basic options for your PuTTY session Specify the destination you want to connect to Host Name (or IP address) Port demo-server example.com 22 Connection type: SSH O Sterial Other:
Benavour Translation Selection Colours Connection Data Proxy SSH Serial Teinet Rlogin SUPDUP	Load, save or delete a stored session Savgd Sessions Default Settings demo-server demo-server-2 Save Delete
	Close window on exit: Always Never Only on clean exit

Access High Performance Computing (HPC) Cluster



Set up scientific computing software on Linux systems

This section





Run physics experiments, simulations, and data analysis





ACCESSING JLAB IFARM

Hands-On Exercises to master:

- SSH to JLab login portal
- SSH to jlabl1 public machine
- SSH to JLab ifarm
- SSH between jlabl1 and ifarm
- Verify ifarm group and job submission permissions
- Perform a manual SSH ProxyJump into the ifarm
- Perform a manual SSH tunnel into the ifarm
- Recommended: Set up a permanent SSH ProxyJump
- Optional: Connect to JLab JupyterHub <u>https://jupyterhub.jlab.org</u>
- Optional: Connect to JLab VDI system <u>https://vdi.jlab.org</u>



34

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
- SSH or Linux RHEL9 VDI is recommended (can persist)

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
- ProxyJump SSH and Port Forwarding
- JupyterHub
- Tomorrow
 - Interactive farm (ifarm) nodes with slurm access
 - Farm batch submission access

JLab Service Now Article on remote Access:

https://jlab.servicenowservices.com/scicomp?id=kb_article

<u>&sysparm_article=KB0015066</u>

GlueX Wiki on VSCode ifarm access

https://halldweb1.jlab.org/wiki/index.php/How_t

o_Setup_Visual_Studio_Code_for_the_ifarm

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

٠



- application and open it
- Execute `ssh –Y <u><username>@login.jlab.org</u>` ٠
 - Use MobilePass SAS MFA personal 6-8 digit code + 6 ٠ digit encrypted code output in one line as the password
- Then: Execute `ssh –Y <username>@ifarm` ٠
 - Use your usual JLab Common User Environment (CUE) ٠ password, or your ssh-key if you have set one up
- Linux VDI (shown on the right):
 - Same as using Linux, managed by JLab ٠
 - Accessed using <u>https://vdi.jlab.org</u> login website ٠
 - See notes from prior set of slides for instructions ٠
 - Then: open a terminal from inside the VDI session •

JLab RHEL9 VDI screen – activities button and terminal


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

Administrator: Windows PowerShell	-	×
Try the new cross-platform PowerShell https://aka.ms/pscore6		^
PS C:\WINDOWS\system32> <mark>ssh</mark> usage: ssh [-46AaCfGgKkmnnqs[tVvXxYy] [-B bind_interface] [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]		
[-E log_file] [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file] [-J [user@]host[:port]] [-L address] [-l login name] [-m mac spec] [-O ctl cmd] [-o option] [-p port]	1	
[-Q query_option] [-R address] [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]] destination [command]		
PS C:\WINDOWS\system32>		
		~

- To Install a Linux operating system in **Windows Subsystem for Linux** (WSL) utilize Windows official channels
 - Ubuntu 20.04.6: <u>https://apps.microsoft.com/detail/9mttcl66cpxj?hl=en-us&gl=en</u>
 - 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"
 - Simplify X11 windows forwarding for Linux and gives you the power of Linux within the Windows ecosystem



SSH with Windows the old way:

- Download and Install PuTTY and Xming
 - PuTTY: Download the application from the Microsoft Apps store to install PuTTY.
 - <u>https://apps.microsoft.com/detail/xpfnzksklbp7rj?amp%3Bgl=US&hl=en-us&gl=US</u>
 - Xming: Download from SourceForge be careful to not click on advertisements inspect all links first!
 - <u>https://sourceforge.net/projects/xming/</u>
 - 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"

Session	Basic options for your Pull	TY session		
Logging	Specify the destination you want to connect to			
- Teminal	Host Name (or IP address)	Port		
Bell		22		
Window Appearance Behaviour Translation	Raw Telnet Rlogin Load, save or delete a stored session Saved Sessions	● <u>S</u> SH ○ Serial		
- Colours	Default Settings	Load		
- Data		Sa <u>v</u> e		
Froxy Telnet Blogin		Delete		
Serial	Close window on exit: Always Never O Onl	y 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)



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

ategory:			Category:	
	Basic options for your PuTTY	session	Features	Options controlling SSH X11 forwarding
Terminal Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Poxy Telnet Rlogin SH	Specify the destination you want to cor Host <u>Name</u> (or IP address) Connection type: ○ Raw ○ Ielnet ○ Riogin ● S Load, save or delete a stored session Saved Sessions Default Settings	Inect to Port 22 ISH O Segial Load Save Delete		X11 forwarding Grable X11 forwarding X display location Remote X11 authentication protocol MIT-Magic-Cookie-1 X authority file for local display Browse
I Serial	Close window on exit: Always Never Only or Only or	n clean exit	- TTY X11 - Tunnels V	
About H	elp Open	Cancel	About Help	Open Cancel

39

PuTTY GUI "Session" page

Connection > SSH > Auth > X11 Page



SSH with Windows the old way:

 \bigotimes

- 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



<-- A PuTTY prompt after successful logging into a remote host

xclock test-command demonstrating successful windows forwarding -->





SSH with Mac:

- Install XQuartz <u>https://www.xquartz.org/</u>
 - 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



XQuartz splash page and included xterm terminal emulator



Farm login and batch submission access:

- ssh to ifarm following earlier instructions
- Perform an interactive job submission test (instructions on the right)
- Batch submission, monitoring, and debugging will be covered later
- Successful verification requires your collaboration/group's computing coordinator to have given you permissions (see details on the right)

- <u>To test ability to submit jobs, then execute:</u>
 - 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)

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: <u>https://jlab.servicenowservices.com/kb</u> <u>?id=kb_article_view&sysparm_article=KB0014685</u>
- To verify complete farm batch job access log in to ifarm, 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=22d 2c5db1b4a09506a9e85dae54bcbcc)



Advanced:

- SSH command chaining
- SSH 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



Advanced: SSH ProxyJump

- SSH command chaining via manual ProxyJump
 - `ssh -Y -J <user>@login.jlab.org <user>@ifarm`
 - J indicates manual ProxyJump
 - Y indicates X forwarding with strict security check (can do –X instead to not check security)
 - This is quite convenient: it simplifies to one line and requires entering only the MFA password



Advanced: SSH Tunnels

- SSH 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, or any other unused port number
 - Example bashrc: alias tunnellogin="ssh -L 2201:ifarm:22 < username>@login.jlab.org"
 - SSH over tunnel: `ssh -p <forwarding port> localhost`
 - <use the same forwarding port as set up earlier>
 - Example bashrc: alias ifarmssh="ssh -p 2201 localhost"
 - What is this doing?
 - What you are doing here is attaching a remote port (in this case 22) onto a local port (in this case 2201)
 - With this attached port you can now interact with the localhost (your own computer) as if it is the remote (because it is!)
 - Primary convenience is you can attach <u>many activities</u> through this one opened port, <u>all without needing to type a</u> <u>password</u> (including additional SSH windows, sftp, scp, VNC, etc.)
 - Make sure the terminal executing 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



Advanced: Permanent ProxyJump

- SSH ProxyJump permanent proxy configuration
 - General guide: <u>https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0015066</u>
 - Specific guide, Linux: <u>https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0014918</u>
 - Specific guide, Windows:

https://jlab.servicenowservices.com/kb?id=kb_article_view&sysparm_article=KB0015113



Advanced: JupyterHub

- <u>https://JupyterHub.jlab.org</u>
- Provides access to JLab ifarm compute power, GPUs, and file systems
- Requires a Google Authenticator MFA token (generated when initially joining JupyterHub – ask Help Desk for help)
- You can set up your own Notebook Images using venv Virtual Environment tools (example right)

Jefferson Lab	Jefferson Lab
Log In	Log In
Password	
	Log In Cancel
Log In	Back

You can use the jlab ifarm to hold temp files, home folder, etc.

- Just need to configure symlinks from the main jlabl1 ~/home/<username> folder to point where you want things to go
- To avoid running out of space on your ~/home folder put them in a /work/ or /group/ space
 - \circ .local -> /work/detimg/cameronc/local
 - $\circ \qquad . jupyter \ -> / work/detimg/cameronc/jupyter \\$
 - $\circ \qquad . {\tt cache} \mathbin{\mathchar`->/work/detimg/cameronc/cache}$
 - .conda -> /work/detimg/cameronc/conda
- You can use conda virtual environments to set up your own custom designed environments (or just use the ones pre-made by the SciComp team)
- To make your own venv-kernel (example case):
 - `conda activate pytorch_env`
 - `python -m ipykernel install --user --name=pytorch_env --display-name="Python (pytorch_env)"`
 - Restart your Jupyter Notebook server for the new kernel to appear in the list.
 - Select Kernel: In your Jupyter Notebook, go to Kernel -> Change Kernel and select "Python (pytorch_env)".

Server Options

elect a notebook image
scipy
pecify runtime (HH:MM:SS format, Max: 24hr)
12:00:00
pecify CPUs per task (Max: 16)
4
pecify Memory per CPU (Max: 4000 MB)
1000
elect GPU type
NVIDIA Telsa T4
pecify GPUs per task (Max: 4)
0

Advanced: VDI

• VDI is Virtual Desktop Infrastructure – VMWare Horizon Virtual Machine connected to JLab CUE Account shared /home

Let site edit files? **vdi ilab org** will be able to edit files in **VI**

older Sharir

- Must first request Linux RHEL9 access from Computer Center (helpdesk)
- File-sharing through VDI -> Under settings enable "edit files" and pick a local folder to share
 - Mount <your folder> from your local your computer
 - Now accessible inside VDI at ~/tsclient/<your folder> (can copy/paste files, no editing!)
- Persistence closed tab can be reconnected to, within a reasonable time-frame
 - Settings and files are persistent, this is == your farm or jlabl1 /home/<username> directory!



ACCESSING JLAB IFARM WITH VIRTUAL NETWORK COMPUTING (VNC)

Advanced: VNC

- 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): <u>https://hallaweb.jlab.org/wiki/index.php/Using_a_VNC_Server/Client#Setting_up_the_VNC_Server</u>
 - 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

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

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

• For

Windows: <u>https://jlab.servicenowservices.com/sp?sys_kb_id=ab0602eadb844810ee4a3889fc961942&id=kb_art_icle_view&sysparm_rank=6&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361</u>

- 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>`



ACCESSING JLAB IFARM WITH VIRTUAL NETWORK COMPUTING (VNC)

Advanced: VNC

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

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

• For

Windows: <u>https://jlab.servicenowservices.com/sp?sys_kb_id=ab0602eadb844810ee4a3889fc961942&id=kb_art_icle_view&sysparm_rank=6&sysparm_tsqueryId=eeab4bc8478fca10281bbd51026d4361</u>

- 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"



Wednesday, May 21, 2025

HANDS-ON: NAVIGATE/MANIPULATE FILES

Category:		
Category: Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Tanslation Selection Colours Connection Data Proxy Proxy Colours	Basic options for your PuTTY's Specify the destination you want to conner Host Name (or IP address) demo-server example com Connection type:	ession ct to Port 22 et ~
- Froxy ■ SSH - Serial - Telnet - Rlogin - SUPDUP	demo-server demo-server-2	Sa <u>v</u> e Delete
	Close window on exit: Always Never Only on Close	clean exit
About Help	Open	Cancel

Access High Performance Computing (HPC) Cluster





Run physics experiments, simulations, and data analysis

This section







NAVIGATING IFARM FILESYSTEMS

Hands-On Exercises to master:

- Test a round of ssh and scp/sftp/globus of a simple example file
- Generate an SSH key (with password protection)
- Look at your SSH key, using the text editor of your choice
- Tweak your .*shell rc (Run Commands) and .login files to your taste
- Edit the comment, contact information, computer identity of the SSH Pub-Key as needed
- Place SSH keys in the right place on your local and remote computers filesystems
- Modify and verify <u>r/w/x</u> and <u>global/group/user</u> security for your ~/ home folder, .ssh/ folder, and contents
- Practice parsing of the example-file, change the scripts to your taste
 - bash
 - tcsh
 - Python





NAVIGATING IFARM FILESYSTEMS

- Basic Unix CLI Commands
- JLab network file system features
- SFTP, SCP, Globus file transfers
- Text editors Vim, Gedit, Emacs, Nano
- IDEs VSCode, Eclipse, IntelliJ

JLab computing cluster "farm" has interactive capability through "ifarm", with two nodes (ifarm2401, ifarm2402) providing access points for testing, etc.

- AMD EPYC 9554 (Zen 4 "Genoa")
- 256 threads (2 sockets × 64 cores × 2 threads/core)
- 3.1 GHz base / 3.75 GHz max
- 1.5 TB memory
- 28 TB striped NVMe /scratch
- HDR (200 Gb/s) IB

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

Exercises

- Place your SSH public key on remote and local computers
- Verify security, explain chmod, chgrp, ls, bits, etc.
- Test a round of ssh and scp/sftp/globus(?)



Transferring files: The Easy Way – Download from the internet with `wget`

- wget Linux command, download a known internet file to your local machine
- Please download the example text file and bash, tcsh, python, and ROOT scripts for later use!
- Execute these commands in a folder you want to work out of, try executing those scripts (if you know how!):
 - wget https://userweb.jlab.org/~cameronc/files/Talks/20250522-SciComp_Workshop/raw/lorem.txt
 - wget https://userweb.jlab.org/~cameronc/files/Talks/20250522-SciComp_Workshop/raw/print_first_and_every_45th.py
 - wget https://userweb.jlab.org/~cameronc/files/Talks/20250522-SciComp_Workshop/raw/print_first_and_every_45th.tcsh
 - wget https://userweb.jlab.org/~cameronc/files/Talks/20250522-SciComp_Workshop/raw/printFirstAndEvery45th.C
 - wget https://userweb.jlab.org/~cameronc/files/Talks/20250522-SciComp_Workshop/raw/print_first_and_every_45th.sh
 - (/work/<groupname>/<username>/SciComp_Examples/ would be a good starting place, or just use your home directory)
- Next task:
 - Copy the lorem.txt file between the ifarm folder and your local personal computer using your preferred method (full instructions for each method are described in the following few slides)



56

Transferring files:

- Globus standalone software: file transfer system that can be operated from a browser
 - Requires CUE login
 - Requires software installation in order to transfer files to and from your personal computer
 - Works for Mac, Windows, and Linux
 - <u>https://jlab.servicenowservices.com/scicomp?id=kb_article&sysparm_article=KB0014843</u>
 - <u>https://docs.globus.org/guides/tutorials/manage-files/transfer-files/</u>

Log	g in to use Globus Web App
Use e.g., u	e your existing organizational login
Univ By se	ersity of Chicago electing Continue, you agree to Globus terms of service and privacy policy.
	ntinue
¢	Globus uses CILogon to enable you to Log In from this organization. By clicking Continue, you agree to the CILogon privacy policy and you agree to share your username, email address, and affiliation with CILogon and Globus. You also agree for CILogon to issue a certificate that allows Globus to act on your behalf.
	OR
	Sign in with GitHub
	G Sign in with Google
	Didn't find your organization? Then use Clobus ID to sign in (What's this?)



Transferring files:

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

🖺 Administrat	or - Adm	inistrator@172.	17.40.91 - WinSCP				- 0	×
Local Mark Files	Command	ls Session Option	s Remote Help					
🖶 🚼 📮 Synch	ronize 🗾	P 💽 🛞	🗿 Queue 🔹 🛛 Transf	er Settings Default		- 🥩 -		
📮 Administrator@	172.17.40.9	1 💣 New Sessio	n					
🐛 C: System	- 🖪	🔽 🖻 🗖 🍙	2 🔒 🖛 - 🔶 -	📕 Ac 🝷 🚰 🔽	◆ - →	- 12 7	ት 🎜 🖾 ቦ	ind Files 🛛 🗣 📥
📳 Upload 👻 📝	Edit 👻 🗙	🛃 🕞 Propertie	s 🎽 🕂 🗖	Download 👻	🖉 Edit 👻	🗙 🛃 🕞 Pro	operties	» + »
C:\Users\AlexejW\				/usr/home/Administ	rator/			
Name	Size	Туре	Changed	Name	Size	Changed		Rights
<u>t</u>		Parent directory	11.09.2018 08:23:47	<u>₹</u> .		11.09.2018 15:	51:33	rwxr-xr-x
.VirtualBox		Dateiordner	13.09.2018 09:39:29					
🔚 Contacts							×	7
🔚 Desktop		Login						
📔 Documents		New Site		Session				1
其 Downloads		Administrato	r@172.17.40.91	File protocol:				
Ravorites		root@172.1	7.40.11	SFTP				
tinks				Host name:		Port	number:	
Music				172.17.40.91			22	
Pictures								
Saved Games				User name:	Pa	ssword:		
Judeos				Administrator				
ing videos				Edit		Advar	nced 🔽	
<								>
0 B of 0 B in 0 of 12								7 hidde
		4						0:02:43
		Tools	▼ Manage ▼	Login	▼	Close	Help	

Example WinSCP GUI screenshot

Left is your local computer file tree

Right is the remote computer file tree

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

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



Transferring files:

Using Manual ProxyJump though Login portal: 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 ./

- 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/` ٠
 - ``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



SSH PASSKEYS

- History and what they are
- Why they are useful (intra-network stable password, git authentication, multi-system login uniformity)
- How to generate them (multi-system)
- How to place them (multi-system, and git)
- How to maintain security (permissions, never share private key, unique keys, passwords, authentication notifications)



SSH PASSKEYS

- Key and lock pair, where you generate both and distribute the lock to your systems
 - <u>https://blakesmith.me/2010/02/08/understanding-public-key-private-key-concepts.html</u>
 - Useful as an intra-network stable password, git authentication (main point for our later hands-on session), and multi-system login uniformity strategy
 - Generate them on any system, ideally one unique one for each system
 - You must place them on systems you want to access with them
 - Utilize the known_hosts system to save some time and improve your security confidence https://www.baeldung.com/linux/public-key-known_hosts
- You must maintain their security
 - Verify they have the correct permissions https://portal.perforce.com/s/article/6210
 - .ssh directory should be 700 (drwx-----), The public key (.pub file) should be 644 (-rw-r--r--). The private key (id_rsa) on the client host, and authorized_keys on the server, should be 600 (-rw------).
 - Never share your private key
 - Ideally use unique keys for each system, especially unsecured ones
 - Use a password on each one (it is possible to have a no-password key pair)
 - Pay attention to known_hosts authentication notifications for changed server identities (especially for untrusted remote servers)



ACCESSING JLAB IFARM WITH SSH KEYS

SSH Key Generation: Generate your SSH key

- 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): <u>https://www.cyberciti.biz/faq/how-to-set-up-ssh-keys-on-linux-unix/</u>
- SSH key generation and installation: Windows ("cmd", PowerShell)
 - Microsoft guide: <u>https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_keymanagement</u>
 - Atlassian guide: <u>https://support.atlassian.com/bitbucket-cloud/docs/set-up-personal-ssh-keys-on-windows/</u>
- PuTTY ssh keys in Windows:
 - DigitalOcean guide: <u>https://docs.digitalocean.com/products/droplets/how-to/add-ssh-keys/create-with-putty/</u>
- Register your key with the ssh-agent: (use ssh-add command: https://support.atlassian.com/bitbucket-cloud/docs/set-up-personal-ssh-keys-on-linux/)
- Verify security, utilize chmod, chgrp, Is -Isa, etc., and amend the comment information on the pub-key
 - verify you have the correct permissions https://portal.perforce.com/s/article/6210
 - .ssh directory should be 700 (drwx-----), The public key (.pub file) should be 644 (-rw-r--r--). The private key (id_rsa) on the client host, and authorized_keys on the server, should be 600 (-rw------).
 - Always verify the permissions bits before proceeding 3 digt numbers correspond to read, write, execute access: 1+2+4 = 7
- Place .ssh key on the remote server, test a round of ssh and scp/sftp/globus



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: <u>https://ubuntu.com/tutorials/command-line-for-beginners#1-overview</u>
 - You may find the FreeCodeCamp guide additionally informative: <u>https://www.freecodecamp.org/news/command-line-for-beginners/</u>



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
- Most commands have manuals: type `man <command>` without brackets, to learn more, or `<command> --help` usually works too
- Any command accessible from the CLI can also be utilized in the exact same way inside of a script, enabling efficient operations
- Shells give access to the file system and many system-wide installed commands, such as:
 - cd = change directory, move from one folder to another
 - Is = 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 (be very careful with this command!)
 - In = 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
 - man = show the manual entry for this command



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: <u>https://stackshare.io/stackups/emacs-vs-gedit-vs-vim</u>
- Vim (my preferred): <u>https://thevaluable.dev/vim-commands-beginner/</u> Or just type the `vimtutor` command at the terminal!
- Gedit: <u>https://help.gnome.org/users/gedit/stable/index.html.en</u>
- Emacs: <u>http://www.jesshamrick.com/2012/09/10/absolute-beginners-guide-to-emacs/</u>
- 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 ifarm 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
endif

Once satisfactorily set up – next few slides: execute the every 45th line scripts on the lorem.txt (or your own .txt file)



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

```
> 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"
```

Open a blank text file print_first_and_every_45th.sh with your text editor

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



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 (\$ = 1) then
  echo "Usage: $0 filename"
  exit 1
endif
# File to read
setfile = $argv[1]
# Check if the file exists
if (!-f $file) then
 echo "File not found!"
  exit 1
endif
# Counter for lines
setcount = 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



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 File NotFoundError:
    print(f"File {filename} not found!")
if name == " main ":
 if len(sys.argv) != 2:
    print(f"Usage: {sys.argv[0]} filename")
   sys.exit(1)
  file name = sys.argv[1]
  print_first_and_every_45th_line(filename)
```

python print_first_and_every_45th.py yourfile.txt



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

```
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;</pre>
    return 0;
  std::string line;
  int count = 1;
  while (std::getline(file, line)) {
   if ((count - 1) % 45 == 0) {
      std::cout << line << std::endl;</pre>
    count++;
  file.close();
  return 1;
```

Open a blank text file printFirstAndEvery45th.C with your text editor

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



Wednesday, May 21, 2025

HANDS-ON: SCRIPTING/SOFTWARE & GIT

Category:		
Ession Logging	Basic options for your PuTTY se Specify the destination you want to connect	ssion :t to
 Keyboard Bell Features Window Appearance Behaviour Translation Selection 	Host Name (or IP address) demo-server example com Connection type: SSH Osejial Other: Telne Load, save or delete a stored session Saved Sessions	Port 22 t ~
Colours Connection Connection Pota Proxy SSH SSH Serial Telnet Riogin SUPDUP	Default Settings demo-server demo-server-2	Load Sa <u>v</u> e Delete
	Close window on exit: Always Never Only on c	ean exit
	0	0 1

Access High Performance Computing (HPC) Cluster





Run physics experiments, simulations, and data analysis

This section







SCRIPTING/SOFTWARE AND GIT

Hands-On Exercises to master:

- Clone a software repository
- Set up your user information in your git configuration
- Set up ROOT module from CVMFS
- Practice executing scripts in bash, tcsh, python, and ROOT
- Try to debug the tcsh script problem
- Either create your own branch with the bugfix or checkout the bugfix-tcsh branch
- Fork the repository to your personal code.jlab.org account
- Create a pull-request in your personal account for the bugfix-tcsh branch
- Approve the PR and merge



72
TYPICAL GIT VERSION CONTROL EXAMPLE

Code.jlab.org repo-creation instructions

- You can create an empty new repository
- You can create a repository on your computer and push it into code.jlab.org ("Push an Existing Folder")
 - We will use this later, plus creating an empty repository on code.jlab.org, to hold the results of your "fork"
- Can clone an existing repository and push it to your personal space (manual "Forking")
- You can also do these actions from the code.jlab.org website interface (sometimes takes more effort)
- Note: Web interface enables CD/CI tools (commits, PRs, etc.)

Assignment: Clone my example repo with every-45th line parsing scripts in it

- First set up your ssh key on code.jlab.org, following these instructions: <u>https://docs.gitlab.com/user/ssh/#add-an-ssh-key-to-your-gitlab-account</u>
- Execute git clone command (easiest approach):
- `git clone git@code.jlab.org:cameronc/SciComp_Workshop_Git_Example`

Git local setup

Configure your Git identity locally to use it only for this project:

git config --local user.name "Clarke, Cameron" git config --local user.email "cameronc@jlab.org"

Add files

Push files to this repository using SSH or HTTPS. If you're unsure, we recommend SSH.

SSH HTTPS

How to use SSH keys?

Create a new repository

git clone git@code.jlab.org:cameronc/scicomp_workshop_git_example.git
cd scicomp_workshop_git_example
git switchcreate main
touch README.md
git add README.md
git commit -m "add README"
git pushset-upstream origin main

Push an existing folder

Go to your folder

cd existing_folder

Configure the Git repository

```
git init --initial-branch=main
git remote add origin git@code.jlab.org:cameronc/scicomp_workshop_git_example.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```

Push an existing Git repository

Go to your repository

cd existing_repo

Configure the Git repository

git remote rename origin old-origin git remote add origin git@code.jlab.org:cameronc/scicomp_workshop_git_example.git git push --set-upstream origin --all git push --set-upstream origin --tags



73

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: <u>https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account</u>
- Gitlab: <u>https://docs.gitlab.com/ee/user/ssh.html</u> (This is what code.jlab.org runs)
- More intro to Git: <u>https://www.atlassian.com/git/tutorials/setting-up-a-repository</u>

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



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
- sit pull --ff-only (use this to avoid unwanted merge conflicts and better monitor recent remote changes)
- git checkout -b <new branch name here>
- 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



ACCESSING SHARED SOFTWARE

- Shared software can reside in many places
 - /group/ disks
 - /mss/ tapes
 - More recently: CVMFS (CernVM File System)

Load ROOT from CVMFS

- From the command line, execute:
 - module use /cvmfs/oasis.opensciencegrid.org/jlab/scicomp/sw/el9/modulefiles
 - module load root
- Or, add that line to your .tcshrc file (in ~/ your home folder, use `ls –lsa` to see the "hidden" . files)

set hostnamex="`hostname`"
if (\$hostnamex =~ *'farm'*) then
 module use /cvmfs/oasis.opensciencegrid.org/jlab/scicomp/sw/el9/modulefiles
 module load root
endif



76

SCRIPTING

- Execute the scripts from the Git Repo you cloned earlier
 - Simply type ./<script name> <filename> for .sh and .tcsh (bash and C-shell scripts)
 - Use `python3` to run the .py python scripts, also with <filename> as an argument
 - Use `root –l –b –q <script name>.C'("<filename>")' ` to execute ROOT scripts
- Identify the bug (with the tcsh script)
- Try to debug it
- Create your own branch with your bug fixes, or `git checkout` the bugfix-tcsh branch
- Fork the repository to your personal account
 - (Or you can create an empty repository and push it there, as in the prior example)
- Create a Pull Request on the code.jlab.org website (personal repository) and approve



JLAB SCIENTIFIC COMPUTING ACCESS

ategory:			
Category: Session Coging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy SSH Serial Telnet Rilogin SUPDUP	Basic options for your PuTT Specify the destination you want to cor Host Name (or IP address) demo-server example com Connection type: SSH Segial Other: Load, save or delete a stored session Savgd Sessions 	Y session nnect to Port 22 elnet	~
	Default Settings demo-server demo-server-2	Load Sa <u>v</u> e Delete	
	Close window on exit. Always Never Only of O	on clean exit	

Access High Performance Computing (HPC) Cluster



Set up scientific computing software on Linux systems



Run physics experiments, simulations, and data analysis

Feel free to ask any follow up questions: <u>cameronc@jlab.org</u>





Tomorrow



OUTLINE

- Different ways of using the farm VNC, ssh, VDI, VSCode, etc, Introduce scp, sftp, globus
- Walk people through accessing the farm and making their own work directory
 - Navigating the farm using Linux commands
 - Make sure people can use at least the following commands
 - Proxy jump
 - Help people set up globus or scp or sftp
 - If time, make git ssh key
 - Sanitation permissions, gitlab intro, etc.
- Git repo on your work directory, make changes and pull request, execute script
 - Git basics! Clone repository to your work directory, Edit a script or text file within the cloned repository, Add, commit, push change to the forked repo, Create a pull request for the change
- Load some basic modules, CVMFS
 - Write a "Hello World!" script and execute it use ROOT/C++, Building software and debugging