

1st CCU Workshop on remote access



Today's Schedule

- 09:00 to 10:00 – SSH, SSH keys, SCP and rsync
- 10:00 to 10:30 – Tunneling and port forwarding
- 10:30 to 11:00 – Intermission/Questions
- 11:00 to 11:30 – remote text editors: vim and nano
- 11:30 to 12:30 – Mosh and Tmux

You might remember me from ...



You might remember me from ...

Data analysis meetings began in 2022 with Jacob Davidson and myself as organizers



Jacob Davidson



Daniel S. Calovi

IMPRS – Introduction to Scientific Coding workshop



The Data Science Consultancy

Join our
CASCB Slack
channel



Schedule a
meeting with me



Why are we offering this workshop?

- Most of the High Performance Computing (HPC) infrastructure available are only accessible through ssh/terminal
- We would like to remove this barrier that prevents users from fully using these solutions



IP addresses

- Private and public IP addresses
 - Private IP addresses are the ones you have inside your own network
 - Public IP addresses is what the internet sees
- E.g. At home you usually have one router, and everyone connected to it. While accessing a website the website will register your public IP, but if you need to connect to a different computer in your home network, you will need their private IP
- Virtual Private Networks (VPNs)
 - A VPN creates a secure tunnel between your machine and a remote server, often assigning you a new public IP and/or providing you with a private IP within a different Network
 - Used for security reasons to make your connection opaque to your network manager
 - Used for security reasons to grant you access to a network so that you can access computers within

SSH security measures

- Normally your computer will only be able to make outgoing connections to other computers
- The next step is allowing you to receive incoming connections from the local network
 - This usually means installing/managing a firewall, in linux `ufw` would be most common choice
 - `apt install ufw ; ufw allow ssh; ufw enable`

SSH security measures

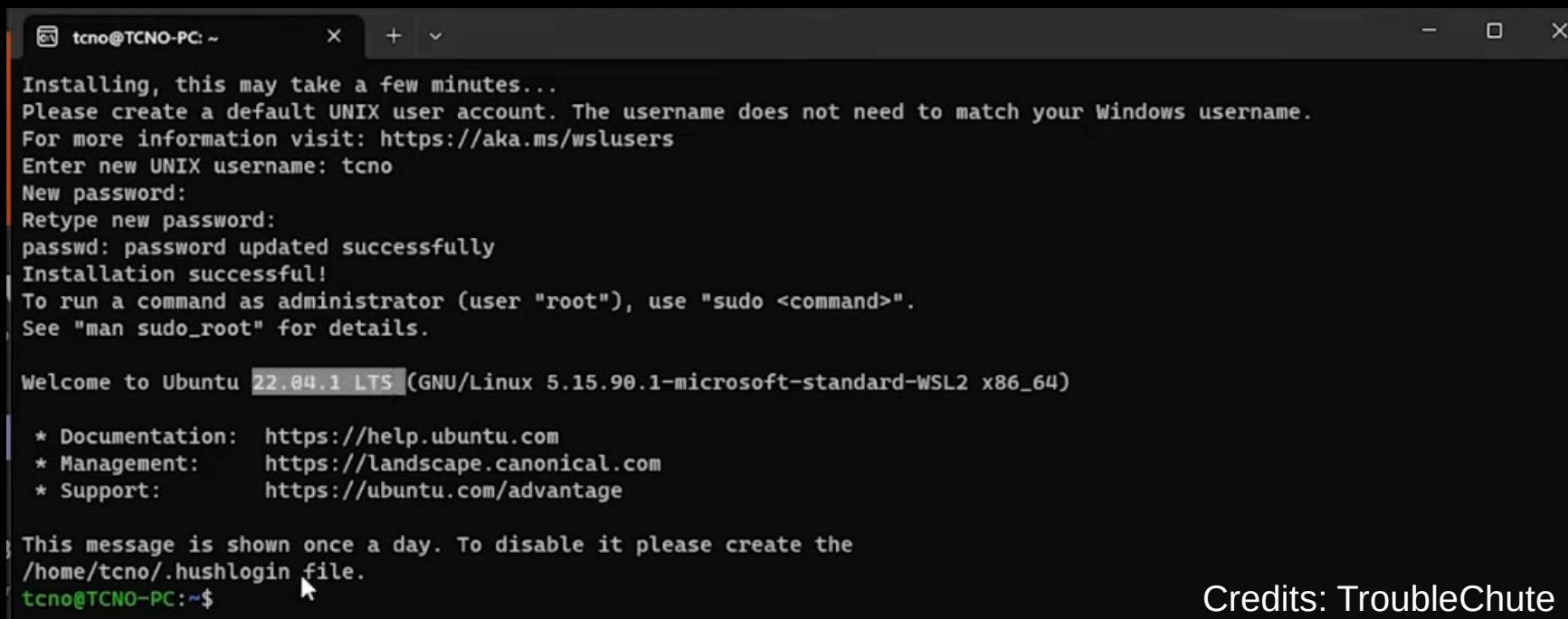
- The third, and very unusual and unsafe step, would be to open your local machine to be accessible from outside your local network (internet-facing)
 - Usually this is not a problem, since most of the time we are behind routers, knowing your public ip would only allow them to see your router, not your local machine
 - Complicated steps would be necessary to make your local machine accessible from an external network → Network Address Translation (NATs, old school gamers will probably remember these)

Why I am explaining all this?

- **Private IPs** matter
- It is often the case that we want to transfer files to and from remote machines
- When machines aren't on the same network, it's usually easier to initiate a file transfer from the local machine — either by requesting files from the remote machine or by sending files to it.
- This approach generally simplifies the process compared to initiating the transfer from the remote machine (avoid setting up a NAT)

Learning by doing it

- MacOS and Linux users should open a terminal
- Windows users should open the ubuntu installation mentioned in the e-mail last week
 - This is something similar to what Windows users should be seeing:



```
tcno@TCNO-PC: ~  
Installing, this may take a few minutes...  
Please create a default UNIX user account. The username does not need to match your Windows username.  
For more information visit: https://aka.ms/wslusers  
Enter new UNIX username: tcno  
New password:  
Retype new password:  
passwd: password updated successfully  
Installation successful!  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.90.1-microsoft-standard-WSL2 x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
This message is shown once a day. To disable it please create the  
/home/tcno/.hushlogin file.  
tcno@TCNO-PC:~$
```

Credits: TroubleChute

Secure Shell (SSH) Protocol

- `ssh-keygen`
 - most of us don't use them to avoid having to type passwords later
- `ssh localhost`
 - `localhost` is an abstraction to avoid using the private IP of the local machine
- Syntax of ssh is normally “`ssh login@host`”
 - Login can be omitted if they are the same between local and remote machines
- If you have different logins in your local machine that you know the password, you can try
 - `ssh different_login@localhost`

Secure Shell (SSH) Protocol

- `cd ~/.ssh/` #linux and wsl users
 - `~/` means `/home/login/` in linux
- `Cd /users/login/.ssh/` #macOS
- What are `id_rsa` and `id_rsa.pub`
 - Your identifiers besides a password, once your public key is copied to a remote machine, **normally** you don't need to use your password into it anymore
- `authorized_keys` file
 - These are keys from other computers that can access your machine without **normally** using a password
- `known_hosts` file
 - Every computer you access has some specific information (host key), to avoid spoofing (phishing, man-in-the-middle attacks), your ssh will always warn you if a remote computer has changed their host key

Secure copy (SCP) protocol

- Syntax
 - `scp path/to/your/files login@host:path/at/the/remote/location`
- Example time, considering the login each one of you have receive, in a terminal type:
 - `touch attendance.txt` #creates an empty file
 - `scp attendance.txt login@64.226.126.98:~/`
 - `ssh login@64.226.126.98`
 - `ls -l`
- If you were copying a folder with subfolders you should use the recursive option
 - `scp -r path/to/your/folder login@host:/path/at/the/remote/location`
 - Question time, how would you transfer a file from that server to your computer?

Secure copy (SCP) protocol

- Syntax
 - `scp path/to/your/files login@host:path/at/the/remote/location`
- Example time, considering the login each one of you have receive, in a terminal type:
 - `touch attendance.txt` #creates an empty file
 - `scp attendance.txt login@64.226.126.98:~/`
 - `ssh login@64.226.126.98`
 - `ls -l`
- If you were copying a folder with subfolders you should use the recursive option
 - `scp -r path/to/your/folder login@host:/path/at/the/remote/location`
 - Question time, how would you transfer a file from that server to your computer?

`scp login@host:/path/at/the/remote/location/attendance.txt path/to/your/folder/`

remote sync (rsync)

- Ever had a huge list of files in a folder and you wanted to only copy the ones that do not exist in the destination location?
- **rsync** is your friend (but a bit of a high maintenance one)
- **rsync -azulv files/folder login@host:/path/at/remote/location/**

remote sync (rsync)

- What is azulv? And what are the MANY options available?
- **-a** → Archive mode, makes an exact copy
- **-z** → Compression, compresses the data during transmission, high cpu usage, good for text files, bad for videos or binaries
- **-u** → Update, skip files that are newer at the destination
- **-l** → Copy symlinks as symlinks, copies links as links and not the files they are pointing to (maybe too complex for most)
- **-v** → Verbose, provides detailed output of the process

Test time

- Copy the folder “useless_but_plentifull_files” and all contents inside it to the server in your home

Tunnels and port forwarding

- Sometimes it is useful to create a tunnel between your local machine and a remote one
- Test case scenario: running a jupyter notebook in your office desktop, but accessing it at home
 - `ssh -L 8080:localhost:8000 user@host`
 - This will create a port forwarding between your remote machine and your local one.
 - It is exposing the remote machine port 8000 to your local machine port 8080
- We can open a python webserver by typing:
 - `python3 -m http.server 8000`
- And then in a browser we can open the url
 - `localhost:8080`

ssh -L vs ssh -R

- These tunnels can be incoming or outgoing
- The previous example with `ssh -L` was requesting a port from the remote machine
- If we want to offer a port to the remote machine, we can use `ssh -R`

Tunnels as a means to increase security

- In HPC systems, it is common to have a Frontdoor machine.
- This frontdoor is internet facing (dangerous), and controls access to the powerful computers inside (internal machine)
- Instead of doing two normal ssh connections (one to the frontdoor, and then one to the internal machine) it is worth defining a tunnel for easier access:
- `ssh -L 9090:internal_machine_ip:22 user@frontdoor-ip`
 - Where `9090` is the port where you are routing the ssh connections (port `22`)
- Now you can access that internal machine or copy files to it by using:
 - `ssh -p 9090 internal_user@localhost`
 - `scp -P 9090 /path/to/local/file internal_user@localhost:/path/to/remote/destination`

Advanced uses of ssh

- You can execute commands directly through the ssh by adding a command after the address:
 - `ssh login@64.226.126.98 'mkdir test; ls -lth'`
- You can enable graphical interface forwarding by using:
 - `ssh -X login@host`
 - This is very slow, and should only be used sparingly

Words of caution

- Different administrators can change ssh configurations to disable some of these features
- The root only accessible file `/etc/ssh/sshd_config` contains a list of several services that can be enabled or disabled
- e.g. some remote machines can only be accessed by pre-authorized machines (`authorized_keys` file), some tunnel features might not work as intended in this scenario
- Xforwarding – allowing access to a remote graphical interface might be disabled, only allowing you to use the terminal

Intermission/Questions