Introduction to Unix on Biowulf - January 2023

Alexandra L Emmons Ph.D. & Joe Wu Ph.D.
BTEP/GAU/CCR/NCI/NIH - email ncibtep@mail.nih.gov
Bioinformatics Training and Education Program

Table of Contents

Course overview

Lesson 1 (Overview of Biowulf and signing on to Biowulf)

Lesson 2 (Overview of Biowulf environment and navigating Unix file systems)

Lesson 3 (Copying, file/directory permissions and more on navigating Unix file system)

Lesson 4 (Working with files/folders in Unix - moving, renaming, and removing)

Lesson 5 (Interactive sessions, modules, and bioinformatics applications on Biowulf)

Lesson 6 (Submitting batch jobs and transferring between local machine and Biowulf)

Lesson 6 supplement (Swarm)

Lesson 7 (Viewing file content and data wrangling in Unix)

Help Sessions

Course data

Connecting to Biowulf (additional methods)

Self learning resources

Introduction to Unix on Biowulf 2023

Welcome to this introductory course series on working with Unix on Biowulf. Biowulf *(https://hpc.nih.gov)* is the high performance compute cluster at NIH and runs Unix, which is a command driven operating system. While most are used to working with graphical driven operating systems such as Windows or Mac, working in a completely text and command driven environment can be a daunting task. In addition, most are not used to working in a high-performance computing system where there are lots of computing resources that are shared among many users, so there are some etiquettes that users should follow. In this course series, we will walk through the basics of working in Unix command line on Biowulf. Skills learned in this course will be particularly useful as they are essential to performing bioinformatics work.

Below is an outline of topics covered in this course series. We will meet Tuesdays and Thursdays from January 24 until February 14 between 1 - 2 pm followed by an optional help session from 2 - 3 pm.

Lesson 1 (January 24, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=032d972b0179679e782aa5e91c768424)*)

- Quick overview of Unix and Biowulf
- Discuss Biowulf accounts
- Use of student accounts
- Use of personal account if registrant has one already
- Signing onto Biowulf

Lesson 2 (January 26, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=451ce4c03ea6b33cea1c5555c69df27b)*)

- Overview of the Biowulf environment
- Login node
- Different directory/data storage spaces – home, data, scratch
- Unix directory path structure
- Getting help with Unix commands
- Navigating the Unix file systems (changing directories)
- Listing directory content

Lesson 3 (January 31, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=555c3a0deafe31672b2e7352bb214e5c)*)

- Copying content from one directory to another
- File and directory permissions
- Modifying file and directory permissions
- Continue to learn to navigate the Unix file system and learn to

- List directory content
- Remove files

Lesson 4 (February 2, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=653c656a2f1bcb27dc56c472ad307c70)*)

- Working with files and directories in Unix
- Moving
- Renaming
- More in-depth coverage of removing files and directories

Lesson 5 (February 9, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=7e107abddd45232f8b6078313c97c80b)*)

- Working with an interactive session on Biowulf
- Modules and applications installed on Biowulf
- View available applications
- Load/unload applications
- Change application version used
- Example of using bioinformatics application Biowulf, which includes
- Sratoolkit for downloading sequencing data from NCBI SRA
- FASTQC for assessing sequencing data quality

Lesson 6 (February 14, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=096447ad2fcb8b328b770bc1e7e52df1)*)

- Creating simple shell script that downloads data from SRA and runs fastqc on the data downloaded – will submit this a batch job on Biowulf
- Transfer data between Biowulf and local machine – using the html based fastqc reports generated as an example

Lesson 7 (February 16, 2023) (Recording *(https://cbiit.webex.com/cbiit/ldr.php?RCID=0327e14b365f345c713102af1036a019)*)

- Data wrangling in Unix
- Downloading data from the web
- Viewing of files
- Pattern searching

Student account assignment

For those using student accounts, please click below to view assignment.

Student account assignment *(https://nih-my.sharepoint.com/:x:/g/personal/wuz8_nih_gov/EfgT4_KxMxtMrfWn3FCyegoBKcPPHaAxjzH05Zi7XxgzJw?e=jpupnc)*

Lesson 1: Getting connected to Biowulf

Lesson objectives

After this lesson, we should be able to

- Describe the Unix operating system
- Describe Biowulf
- Connect onto Biowulf via local computer

Unix commands that we will visit in this lesson

- `ssh` (to connect to Biowulf)
- `id` (to check user id and group affiliation)
- `mkdir` (to create directories)

Overview of Unix

In Windows and MacOS, we interact with the computer through a graphical user interface (GUI). On the contrary, in Unix, we interact with the computer by typing commands.

Basic Unix command syntax

The Unix command syntax is composed of

- The command
- Option(s) that will alter how a command functions
- Argument(s), what you want the command to operate on

```
command options argument
```

For instance, to make a new folder in Unix, we use the command `mkdir` . Here, we enter the command followed by the argument(s) that we want the command to operate on. In this case, the argument is the name of the folder that we would like to create. This is different from the graphical based approach that we use to create new folders in Windows *(https://support.microsoft.com/en-us/office/create-a-new-folder-cbbfb6f5-59dd-4e5d-95f6-a12577952e17)* or MacOS *(https://support.apple.com/guide/mac-help/organize-files-with-folders-mh26885/mac)*

```

```

```
mkdir new_folder
```

Above, we just learned our first Unix command, which is just one of many. Before moving further, we should clarify the rationale for using Unix. While there is a steep learning curve, once we have mastered working in Unix, we can perform many of our computing processes. Unix allows for easy file management, editing of text files, and allows us to view tabular data that is too large for Excel. Further, many of the applications used in bioinformatics are made to work in Unix.

Overview of Biowulf

Biowulf is the high performance and Unix-based computing system at NIH. Below are some rationale for using Biowulf.

- Biowulf offers more computing power and space for data storage compared to our local machine.
- Biowulf also houses many applications for bioinformatics *(https://hpc.nih.gov/apps/)*, which are installed and updated by their staff.
- The GUI-based bioinformatics package, Partek Flow *(https://partekflow.cit.nih.gov)* runs on Biowulf.

Visit https://hpc.nih.gov/docs/accounts.html *(https://hpc.nih.gov/docs/accounts.html)* to learn how to obtain a Biowulf account, which requires PI approval and costs $35 a month. Our Biowulf accounts will need to be renewed each year.

Figure 1 shows the hierarchical architecture of Biowulf. This is useful to know so that we know what we are asking for when requesting compute resources.

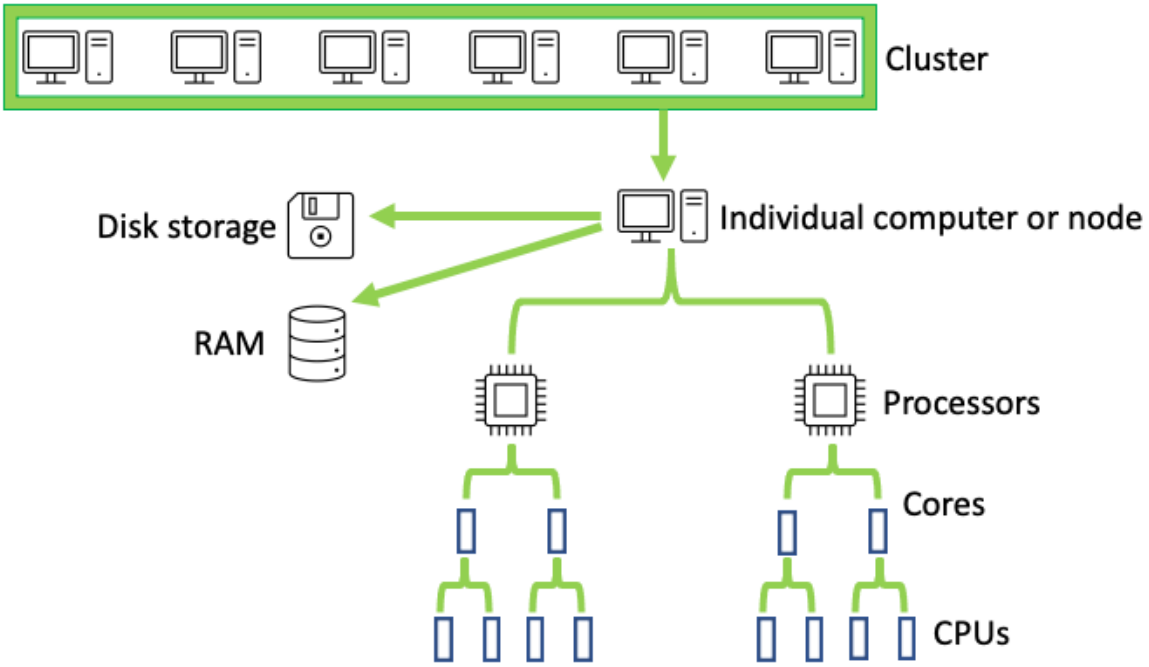


Figure 1: In Biowulf, many computers make up a cluster. Each individual computer or node has disk space for storage and random access memory (RAM) for running tasks. The individual computer is composed of processors, which are further divided into cores, and cores are divided into CPUs. In this example, the individual computer has 2 processors, 4 cores, and 8 CPUs.

Biowulf accounts

If you already have a Biowulf account, please use it for this course series. For those who do not have a Biowulf account, we have access to 30 student accounts.

Signing onto Biowulf

When working on Biowulf, we are working on a remote computer; thus, we need a way to connect to it. We can use Secure Shell Protocol (ssh) to connect to Biowulf. When connecting to Biowulf, we need to either be connected to the NIH network by being on campus or via VPN.

Signing onto Biowulf with a PC

For those using Windows 10 or newer, ssh is built into the command prompt (Figure 2 and Figure 3).

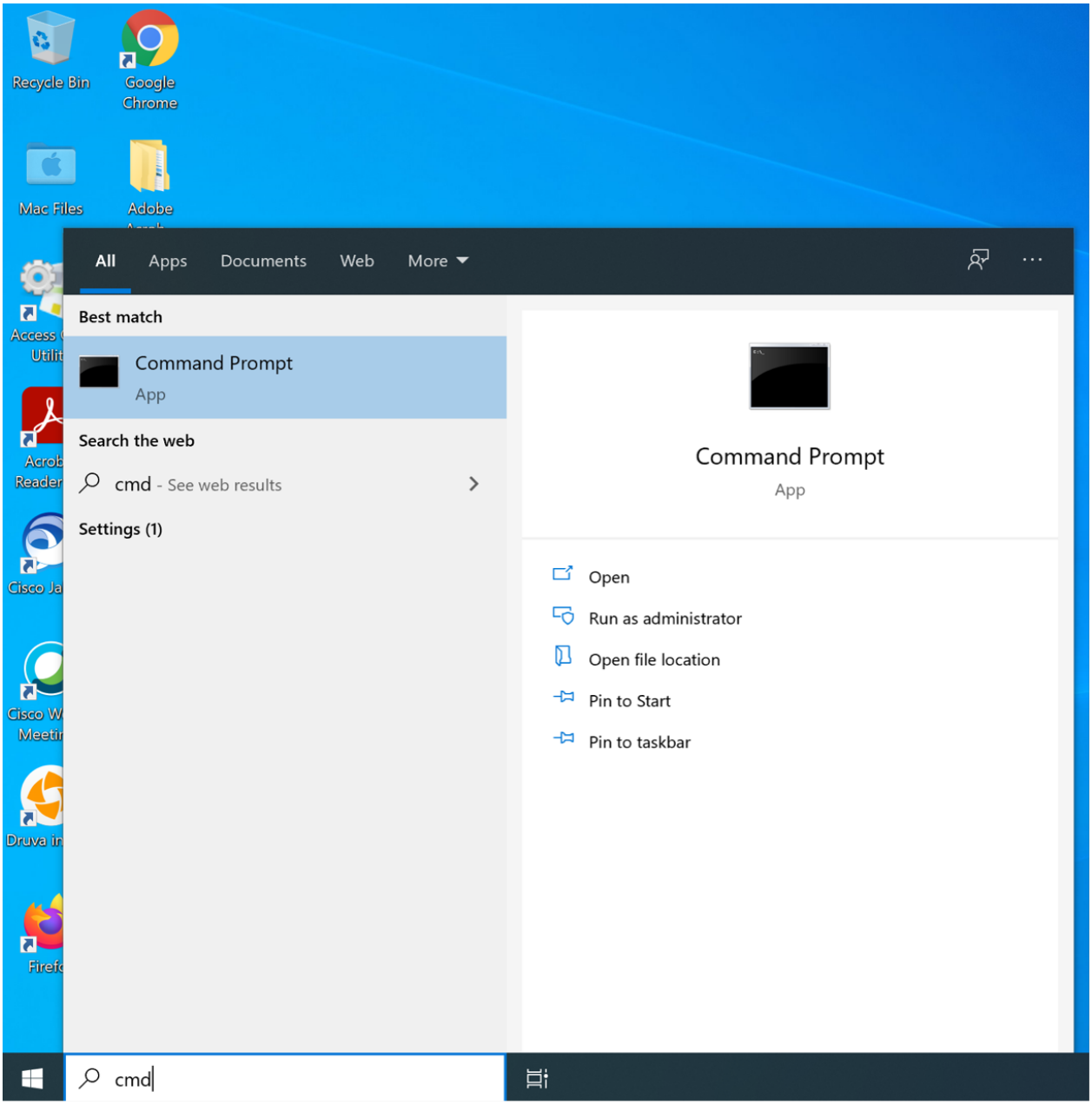


Figure 2: At the search box next to the Windows start menu, type cmd and click on the command prompt application.

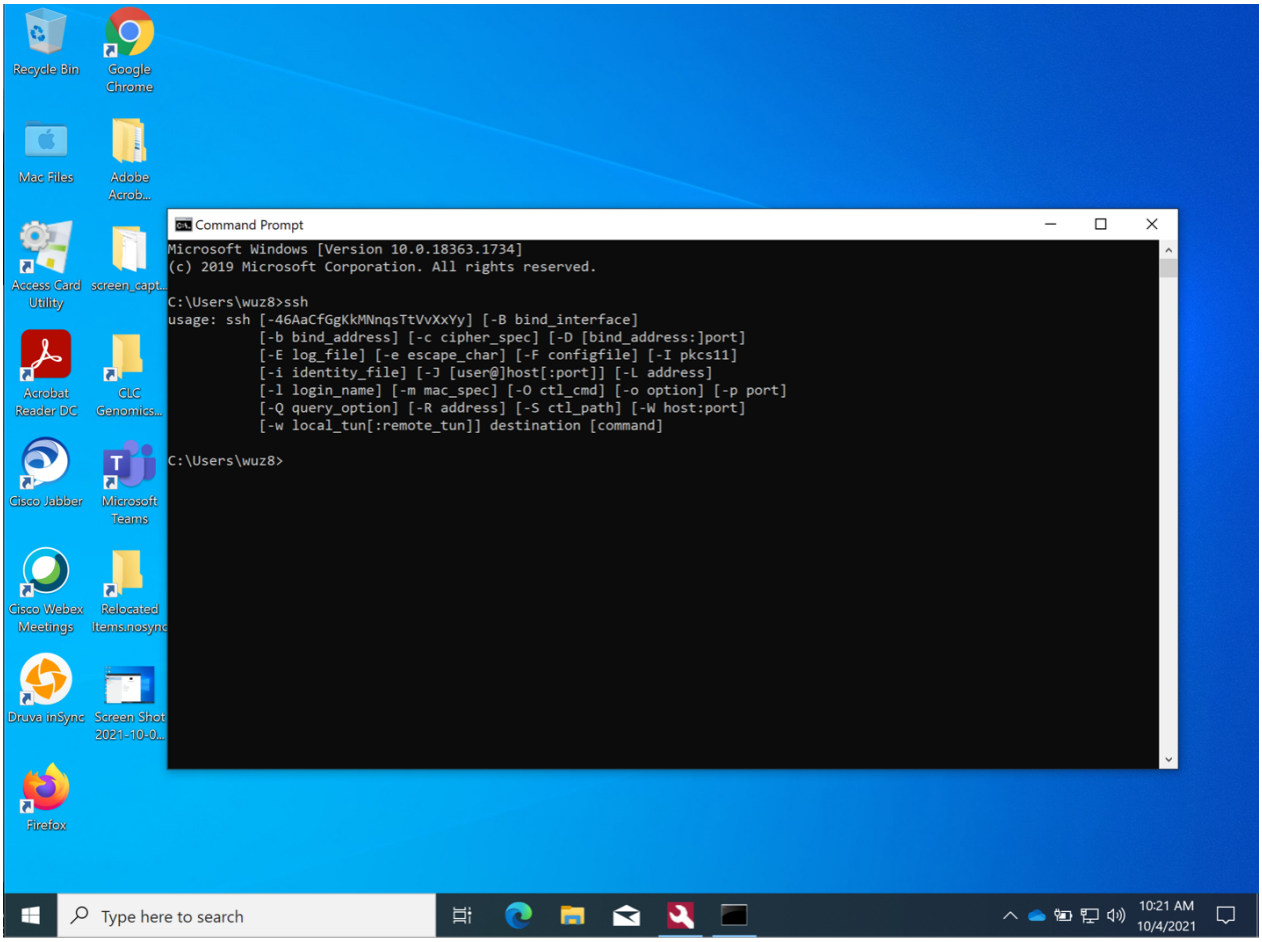


Figure 3: When the command prompt opens, you can type ssh to confirm that it is available

Signing onto Biowulf with a Mac

The best way to sign onto Biowulf from a Mac is to use the built-in terminal (Figure 4). Use the Spot Light search at the Mac menu bar to search for the Terminal application. Click on it to open the Terminal.

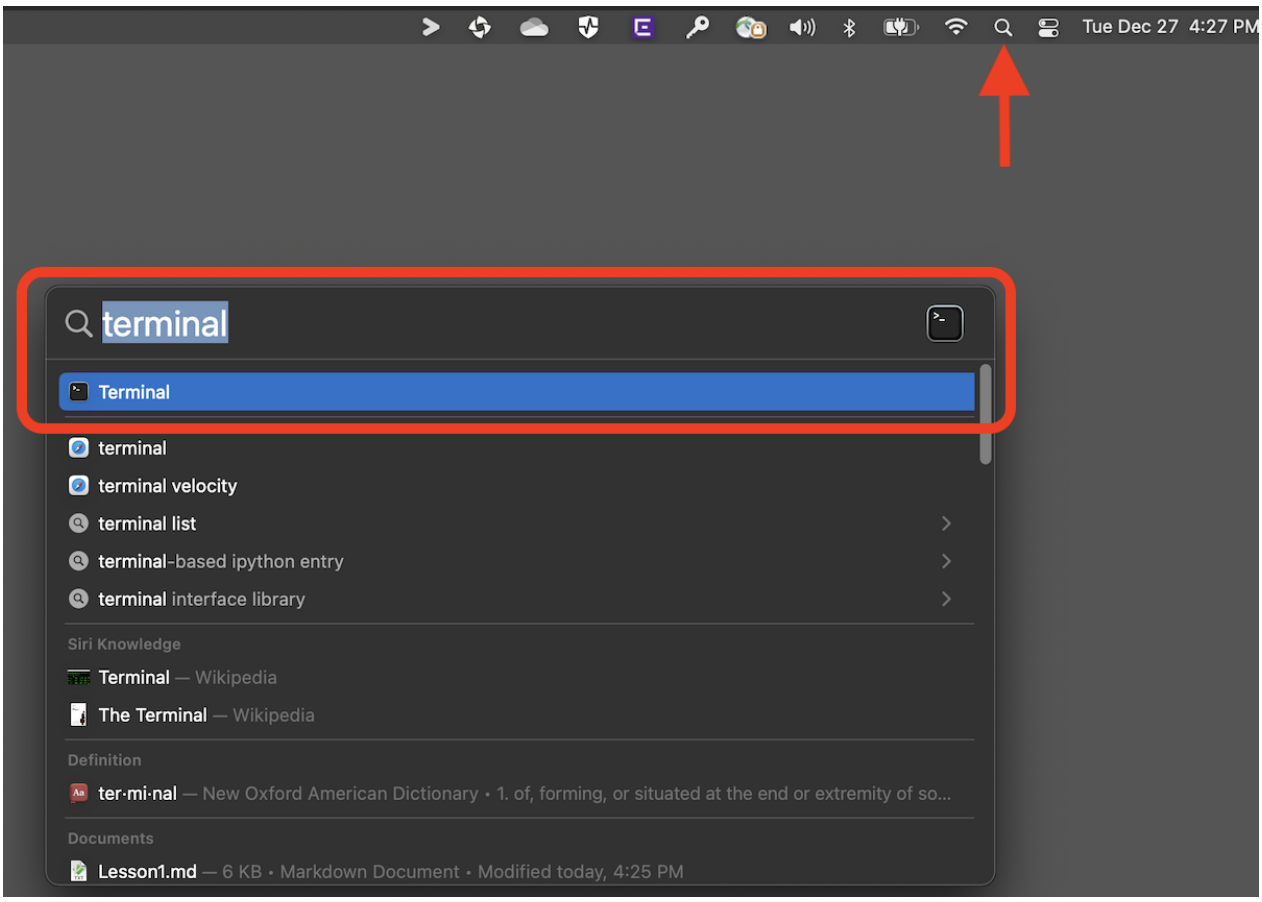


Figure 4: Use the Mac Spot Light search to find the Terminal.

Connect to Biowulf

Remember that if you are not on campus, then you need to connect to the NIH network through VPN. Regardless whether you are using the Windows Command Prompt or Mac Terminal, the construct for ssh to connect to Biowulf is (see Figure 5).

The username in the ssh command is either

- your NIH username if you are using your own Biowulf account for this course **OR**
- one of the student accounts

```
ssh username@biowulf.nih.gov
```

For first time users, when connecting you may see the message below. Respond with yes.

> The authenticity of host 'biowulf.nih.gov (128.231.2.9)' can't be established. ECDSA key fingerprint is SHA256:BoP/KLS17g+gUuQ7mrCHa9oPPO+MHi/h8WML44iA1dw. Are you sure you want to continue connecting (yes/no)? yes

Next, you will see a message warning you that you are accessing a government computer system and that you should not do anything suspicious. At the end of the message, you will be

asked to enter your password, which is either your NIH password (if you are using your own Biowulf account) or the password for the student accounts. The cursor will not move and nothing will be displayed when entering your password, but keep typing.

```
Warning: Permanently added 'biowulf.nih.gov' (ED25519) to the
list of known hosts.
                          ***WARNING***

You are accessing a U.S. Government information system, which
includes (1) this computer, (2) this computer network, (3) all
computers connected to this network, and (4) all devices
and storage media attached to this network or to a computer on
this network. This information system is provided for U.S.
Government-authorized use only.

Unauthorized or improper use of this system may result in
disciplinary action, as well as civil and criminal penalties.

By using this information system, you understand and consent to the
following:

* You have no reasonable expectation of privacy regarding any
communications or data transiting or stored on this information
system. At any time, and for any lawful Government purpose,
the government may monitor, intercept, record, and search and
seize any communication or data transiting or stored on this
information system.

* Any communication or data transiting or stored on this information
system may be disclosed or used for any lawful Government purpose.

--
Notice to users:  This system is rebooted for patches and
maintenance on the first Sunday of every month at 8:00 pm unless
Monday is a holiday, in which case it is rebooted the following
Sunday evening at 8:00 pm.  Running cluster jobs are not
affected by the monthly reboot.

username@biowulf.nih.gov's password:
```

You will be taken to the prompt after successfully entering your password (see below). It is at the prompt where we type commands and interact with Biowulf.

```
[username8@biowulf ~]$
```

The `id` command informs groups that the user might be affiliated with. This is important when collaborating with others Biowulf such that our affiliation with groups will indicate that we have access to the data.

```
id
```

Running the `id` command we see my user id (uid) and primary group id (gid). We also see that I am a part of the GAU and LCP_Omics groups.

```
uid=58740(wuz8) gid=58740(wuz8) groups=58740(wuz8),57888(GAU)
```

Lesson wrap up

In this lesson, we were presented with a high-level overview of Unix and why it is used in bioinformatics. We also learned about the NIH high performance computing system (Biowulf), which runs Unix and why it would be useful to work in this environment for bioinformatics. Finally, we learned how to connect to Biowulf from our local computers.

Even though this was the first lesson, we already learned two Unix commands.

- `mkdir`, which is used to make a new directory
- `id`, which tells users their group affiliation with in a high performance compute system

We also learned the `ssh` command, which is used to connect to Biowulf either from the Windows Command Prompt or Mac Terminal.

Lesson 2: Overview of Biowulf environment and navigating Unix file systems

Quick review

In lesson 1, we saw an overview of the course series and learned of the rationale for using Biowulf. Importantly, we learned to connect to our Biowulf accounts from our local machine using the `ssh` command found in the Windows Command Prompt or Mac Terminal.

Lesson objectives

After this lesson, we should

- Understand the limitations of what we can do in the various spaces within Biowulf, including
    - Login node
    - Home directory
    - Data directory
    - Scratch space
- Understand Unix directory path structure
- Know how to get help with Unix commands
- Be able to navigate directories and list directory contents

Note: do not store Personal Identifiable Information on Biowulf

Unix commands that we will visit in this lesson

- `pwd` (to print present working directory)
- `ls` (to list directory content)
- `cd` (to change directory)

Overview of Biowulf environment

Biowulf user dashboard

A useful feature on the Biowulf website is the user dashboard *(https://hpcnihapps.cit.nih.gov/auth/dashboard/)*. See Figure 1. For those using student accounts, please use the student dashboard *(https://hpcnihapps.cit.nih.gov/auth/student_dashboard)*.

Figure 1: The user dashboard on Biowulf provides useful information for the user's account.

Clicking on the User Dashboard tab will take you to an authentication page (Figure 2). Use your NIH credentials to log in.

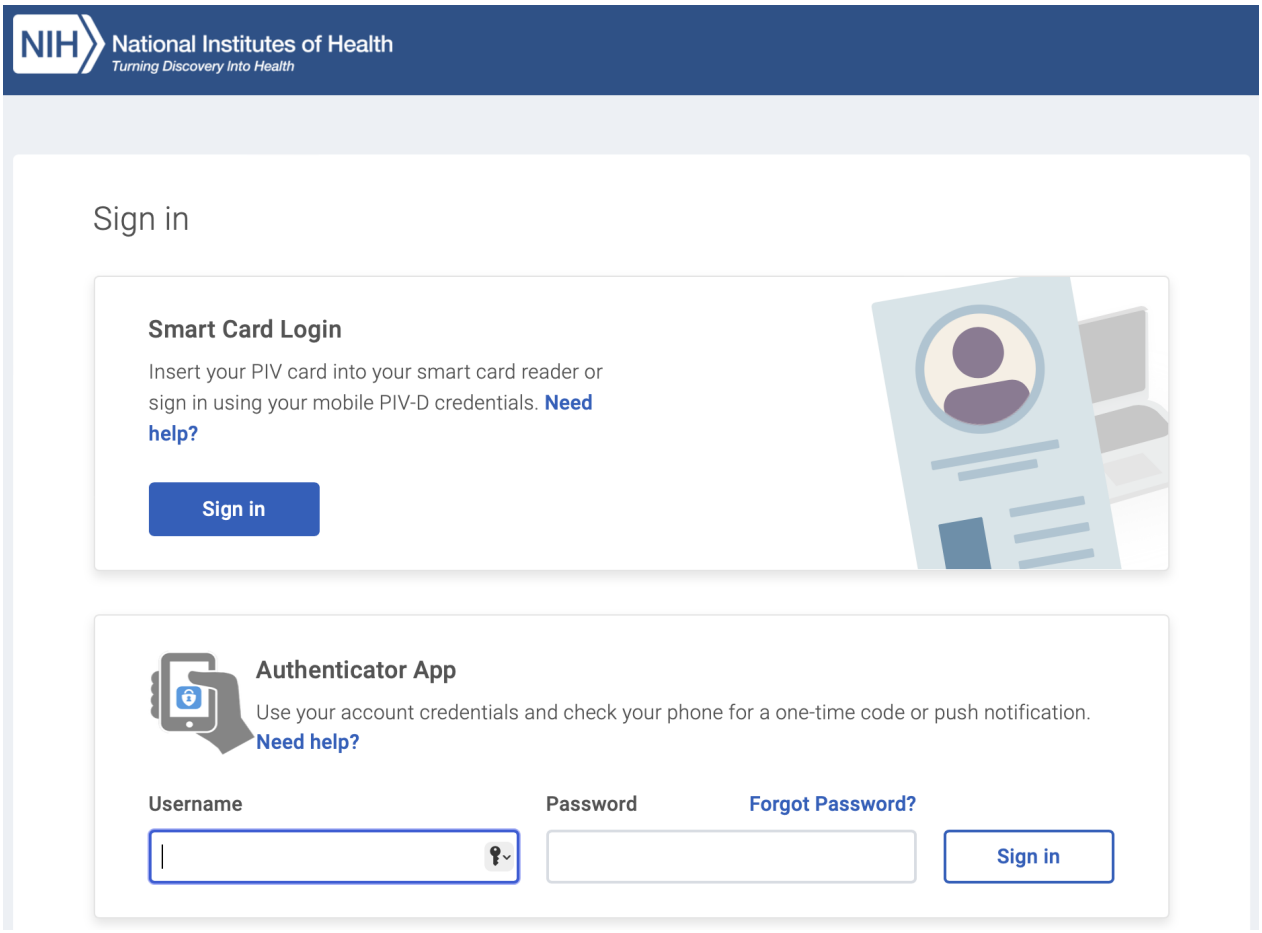


Figure 2: Use your NIH credentials to sign into the user dashboard (even if you are using a student account).

Once logged in, we will be presented with our account information including group affiliations (Figure 3).

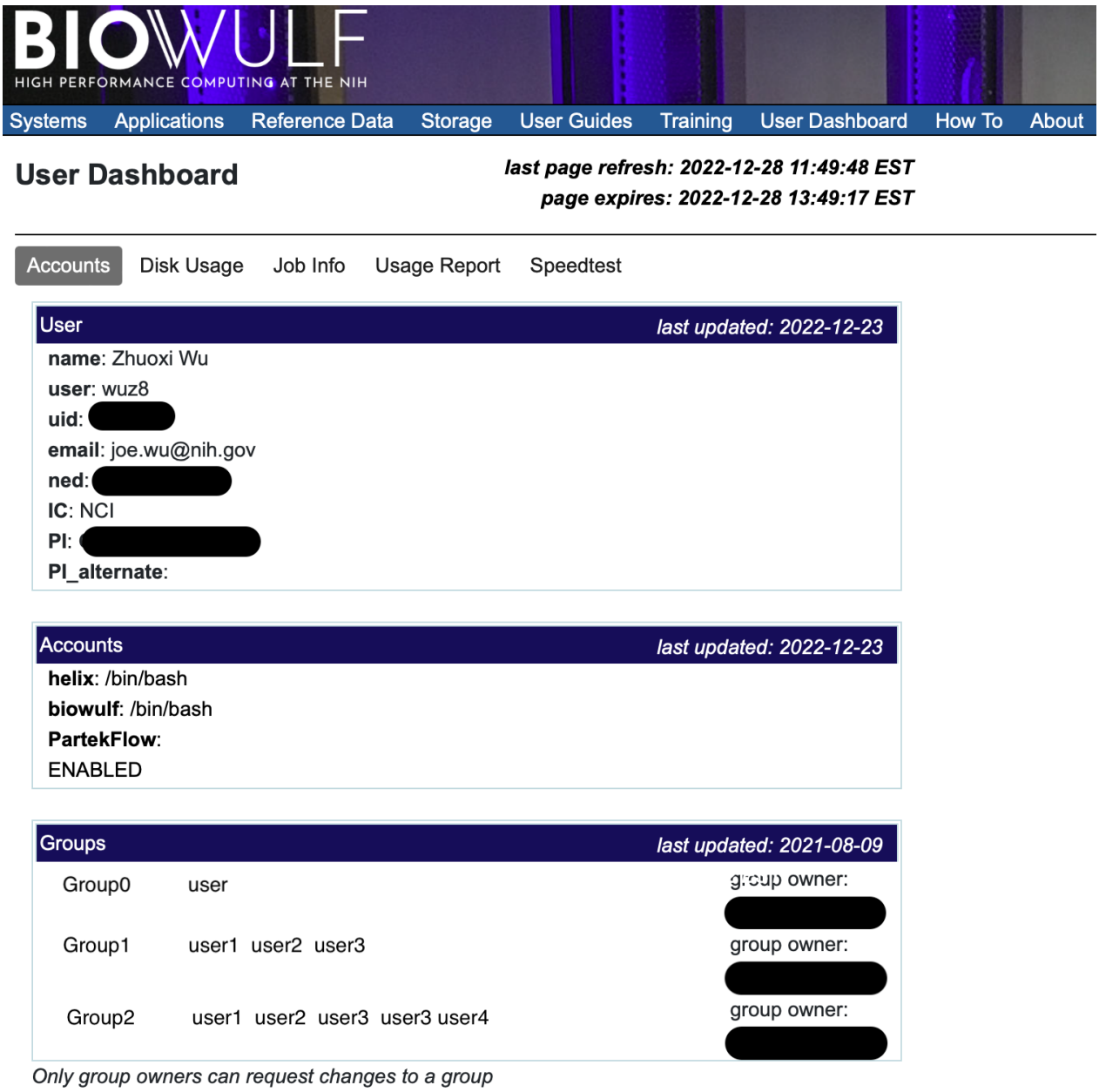


Figure 3: User account information on the Biowulf user dashboard.

Disk quota and usage information is also available in the user dashboard. Note that we can request a quota increase for our data directory (Figure 4) and that the home directory only has 16 gb of space.

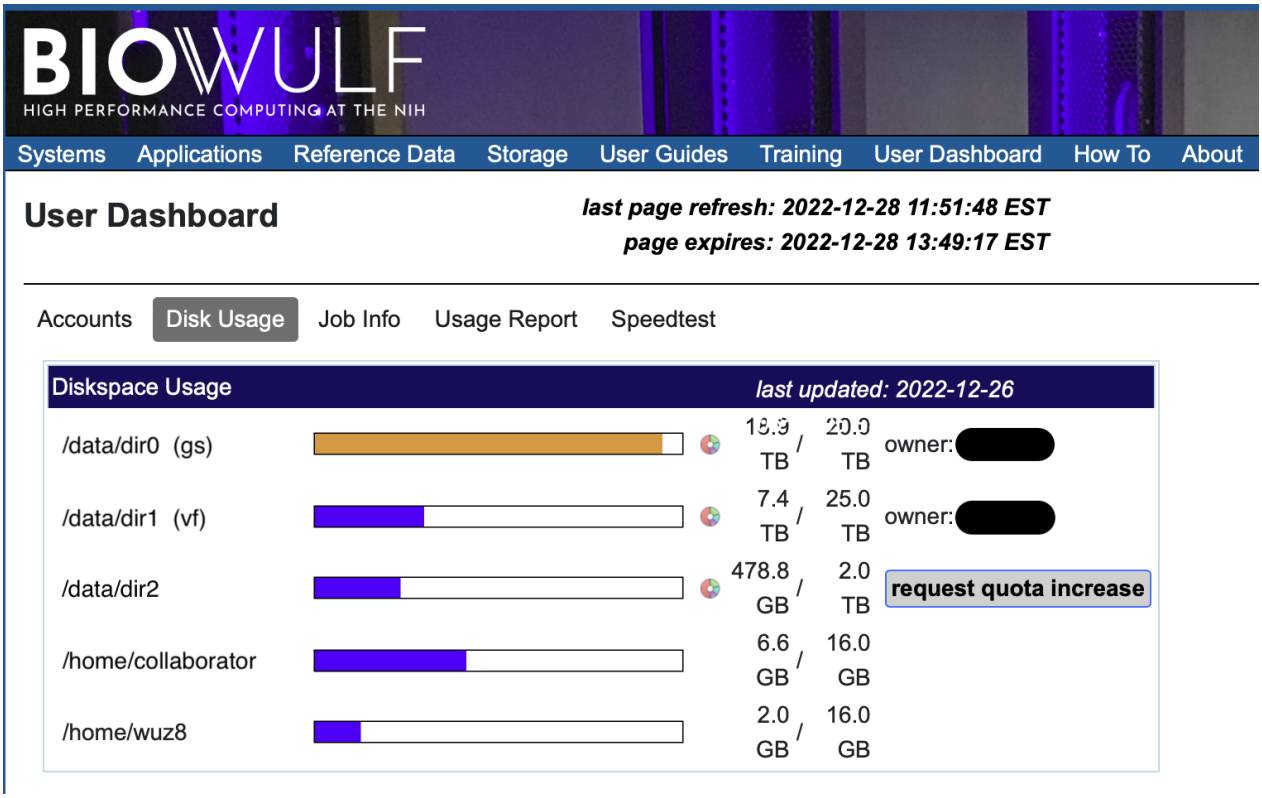


Figure 4: User disk quota and usage shown in the Biowulf user dashboard.

We can also view information and status for the jobs that we have submitted (Figure 5).

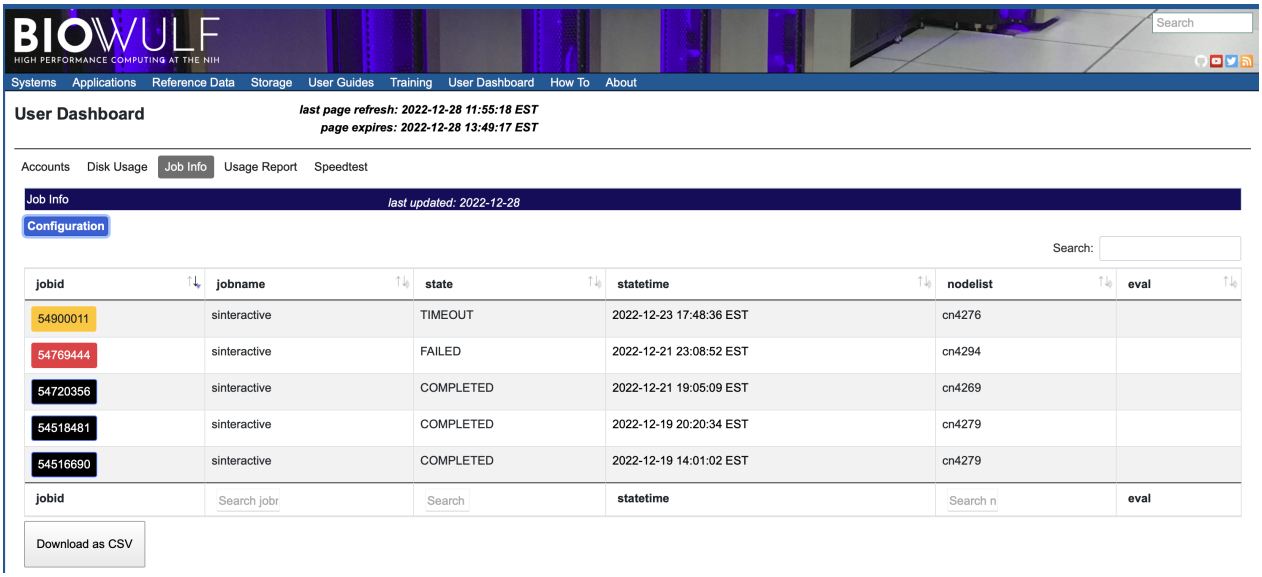


Figure 5: Information and status for jobs submitted to Biowulf.

Connecting to Biowulf

To get started, open the Command Prompt (Windows) or the Terminal (Mac) and connect to Biowulf. Remember you need to be connected to the NIH network either by being on campus or through VPN. Recall from lesson 1 that you use the `ssh` command below to connect to Biowulf,

where username is the username you use to sign in. Remember that when prompted to enter your password, you are not going to be able to see it, but keep typing.

```
ssh username@biowulf.nih.gov
```

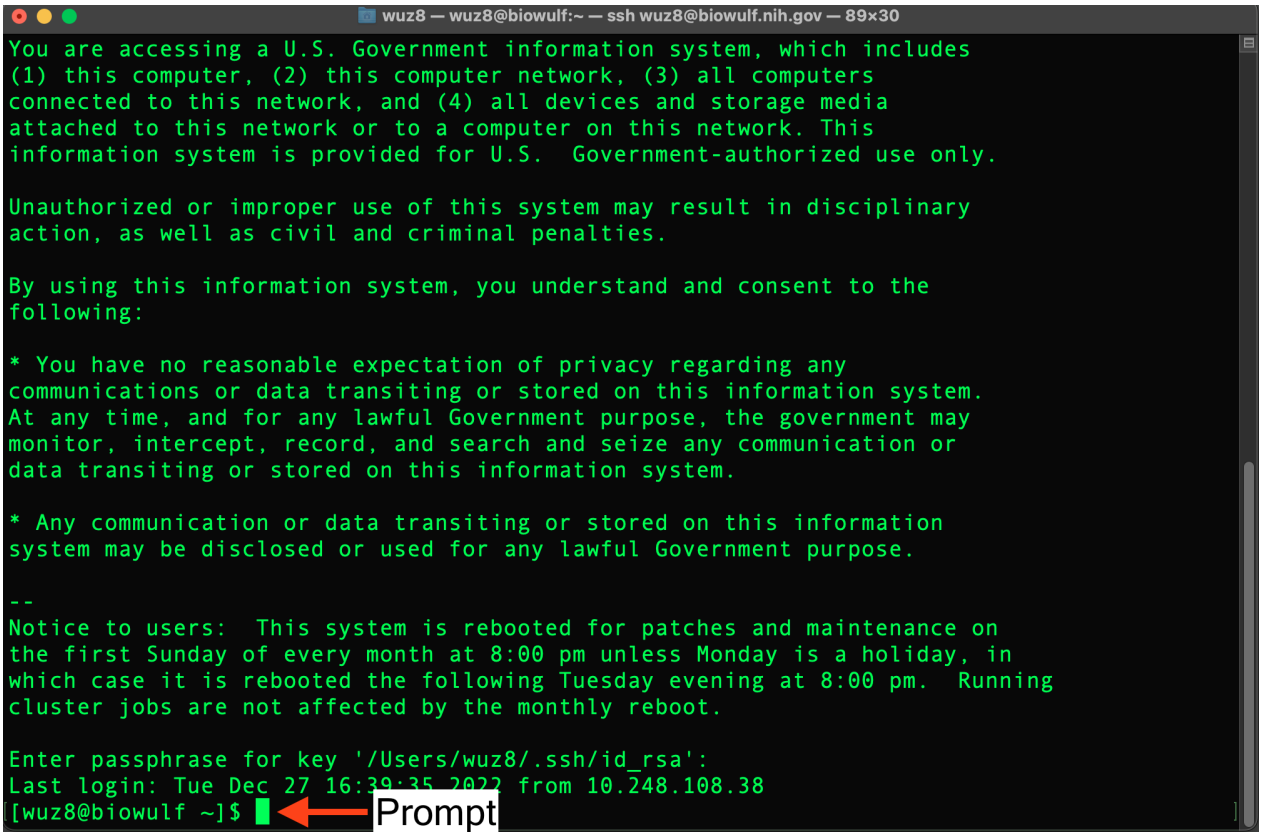


Figure 6: Upon logging in, users will see a prompt where we will interact with Biowulf. The prompt tells us a couple of things that help orient us to where we are. First is what the user is connected to (Biowulf in this case as denoted by username@biowulf). Second, once logged in, we land in our home directory (denoted by ~).

Log in node

We land on the log in node once we connect to Biowulf.

> "The log in node is your point of access to the Biowulf cluster" -- Biowulf accounts and log in node *(https://youtu.be/qiWGxrLI6AY?t=207)*

The log in node is meant for the following (Source: Biowulf accounts and log in node *(https://youtu.be/qiWGxrLI6AY?t=217)*)

- Submitting jobs (main purpose)
- Editing/compiling code
- File management
- File transfer

- Brief testing of code or debugging (under 20 minutes)

There are many users signed on to the log in node at the same time, so do not perform anything that is compute intensive in this space. Request an interactive session or submit a job instead. We will talk about interactive sessions in another lesson.

Home directory

Recall from Figure 4 that users only have 16 gb (gigabytes) of storage space in their home directory. The home directory is your landing spot upon connecting to Biowulf. At the prompt (see Figure 6) it is denoted by "~" and the full directory path is /home/username. As an example, my username is wuz8 so the path to my home directory on Biowulf is /home/wuz8. The home directory does not have much storage space and users cannot request a quota increase for this directory; thus, do not store data or write analysis outputs to the home directory. See the quote below on what the home directory should be used for.

> "Each user has a home directory called /home/username which is accessible from every HPC system. The /home area has a quota of 16 GB which cannot be increased. It is commonly used for config files (aka dotfiles), code, notes, executables, state files, and caches." -- Biowulf *(https://hpc.nih.gov/storage/)*

If we use the `pwd` command, we can identify the present working directory that we are in.

```
pwd
```

```
/home/username
```

Data directory

The data directory is much larger and quota can be increased. The path to the data directory is /data/username. My username is wuz8, so when I do `pwd`, I should see /data/wuz8. We can use the data directory to store our analysis input and output.

```
pwd
```

```
/data/username
```

lscratch

In Biowulf, lscratch is local storage space available on individual nodes. This can be helpful and used for jobs that read or write a lot of temporary files. We will further discuss lscratch in a future lesson.

scratch

The scratch area is a shared storage space accessible to users for storing temporary files. The path to this is /username/scratch where username is the username you use to log into Biowulf. The path to my scratch directory is /wuz8/scratch where wuz8 is my NIH username. A word of caution is that files in scratch are deleted after 10 days. While each user can store up to 10 TB (terabyte) of data in scratch, it is not guaranteed that this amount will always be available. Finally, Biowulf staff will delete files if scratch becomes more than 80% full.

Snapshots

When working in Unix, we need to keep in mind that there is no Recycling Bin (Windows) or Trash can (Mac) that hold deleted items and allow us to recover it. Once we delete something in Unix, it is gone. Fortunately, Biowulf keeps snapshots, which are read-only copy of data at a certain time and we can use these to restore content that we deleted. See here *(https://hpc.nih.gov/storage/backups.html)* for snapshots on Biowulf.

Navigating directories, creating and removing directories, and getting help

Unix directory path structure

Figure 7 shows an example of the file system hierarchy structure in Unix, which starts with the root folder (denoted by /). The root directory is the one where the other directories branch off from. In Figure 7, we see that the home and data directories branch off the root directory. As a matter of fact, if we do `ls /` in Biowulf we will see that the home and data directories are inside the root folder. In Figure 7, the data directory also contains a subfolder, P, which in turn has a folder for the project input (P_in) and a folder for the project output (P_out).

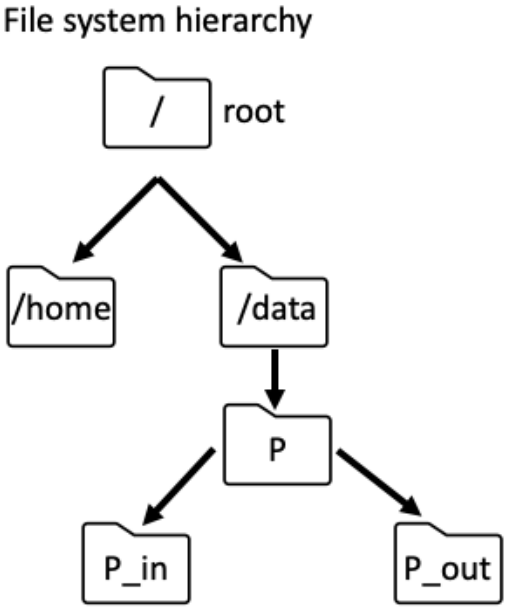


Figure 7: Example of file system hierarchy structure.

```
ls /
```

```
data
home
```

Note that because the home and data directories are both branches of the root, the path to these will be /home and /data respectively. For the P_out folder, the path is /data/P/P_out. Any time we start a path from the root (or "/"), we call it an **absolute path**. Note that each section of a path is separated by "/". This differs from Windows where parts of a path are separated by "\".

> "An absolute path is defined as specifying the location of a file or directory from the root directory (/). In other words,we can say that an absolute path is a complete path from start of actual file system from / directory." -- https://www.geeksforgeeks.org/absolute-relative-pathnames-unix/ *(https://www.geeksforgeeks.org/absolute-relative-pathnames-unix/)*

Getting help with Unix commands

Any time we are unsure of how a command works, we can print the manual for the command using the `man` command followed by the command we want to learn about.

For instance, if we do not know how to use `pwd`, then we can do the following and this prints out the `pwd` manual (Figure 8).

```
man pwd
```

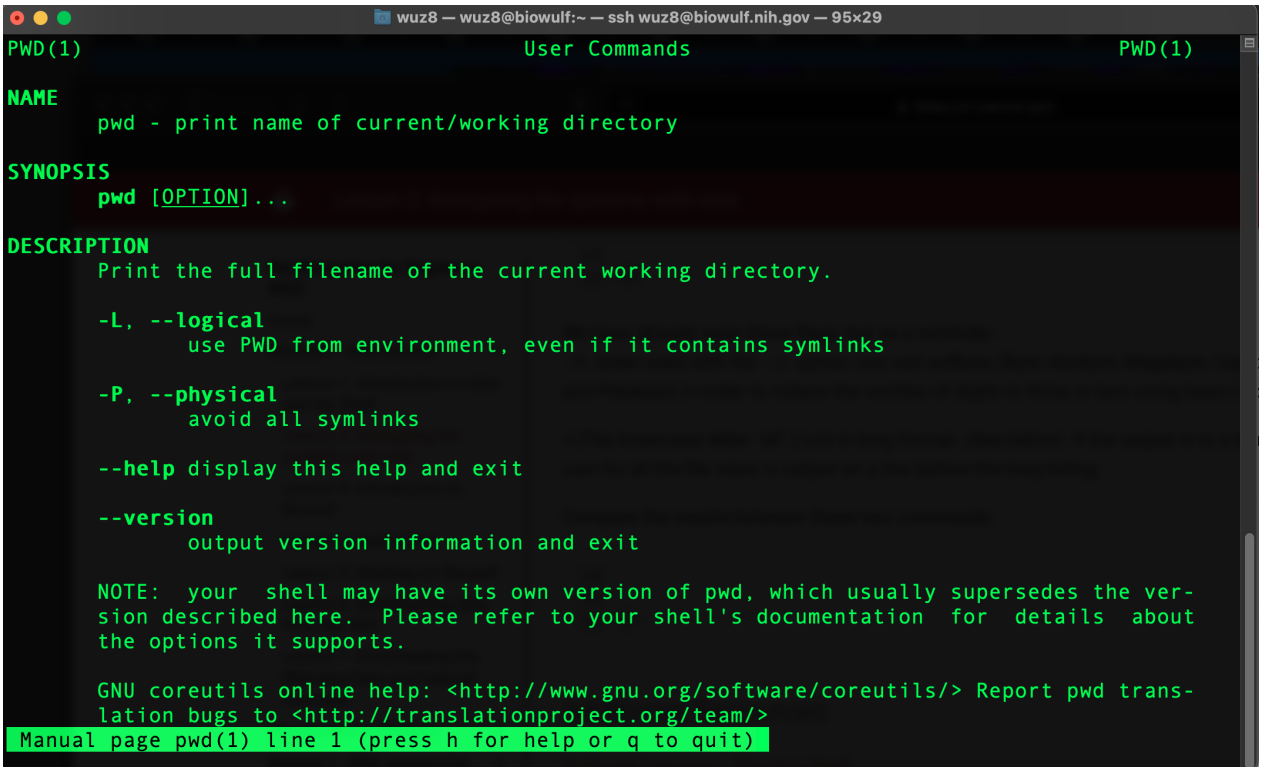


Figure 8: The manual for the `pwd` command

To exit the manual page, hit q.

Changing directory

Recall that upon logging into Biowulf, you land in the home directory, which is limited to only 16 gb of storage space. Thus, you should work in your data folder. Recall that the path to the data folder is /data/username. To change into this directory, we will use a command called `cd`.

```
cd /data/username
```

In my case, since my username is wuz8, I can do the following.

```
cd /data/wuz8
```

Note that once we have changed into the data directory, the "~" (indicating home) is replaced by your username.

```
[username@biowulf username]
```

Now, use `pwd` to confirm that you are in your data directory.

```
pwd
```

```
/data/username
```

To go back to the home directory, we can do either of the following. But let's stay in the data folder though.

```
cd ~
```

or

```
cd
```

or

```
cd /home/username
```

Go back to your data directory and make a new folder called lesson_2.

```
cd /data/username
```

```
mkdir lesson_2
```

Next, change into lesson_2 from your data directory. Note that because you are already in a directory inside the root (ie. /data), we do not need to supply "/" when changing into lesson_2. In essence, we are providing a **relative path**.

> "Relative path is defined as the path related to the present working directory(pwd). It starts at your current directory and never starts with a / ." -- https://www.geeksforgeeks.org/absolute-relative-pathnames-unix/ *(https://www.geeksforgeeks.org/absolute-relative-pathnames-unix/)*

```
cd lesson_2
```

Note that Unix uses "." to denote here in the present working directory and ".." to refer to one directory up. For instance, if you do the following command, we will just get the absolute path to our present working directory (same thing as just doing `pwd`).

```
pwd .
```

```
/data/wuz8/lesson_2
```

Let's make a directory in the lesson_2 folder called unix_on_biowulf_2023. We use `mkdir` to make a new directory.

```
mkdir unix_on_biowulf_2023
```

Change into the unix_on_biowulf_2023.

```
cd unix_on_biowulf_2023
```

To go back one directory to the lesson_2 folder, use the `cd` command.

```
cd ..
```

Listing directory contents

We use `ls` to list the contents of a directory. Staying in lesson_2 folder, we can use `ls` to see what is in it. It should be blank because we have not placed any files or folders in it.

```
ls
```

If you wanted to check the content of a folder other than the present working directory, but do not want to leave the directory you are currently in, you can provide a path to `ls`.

```
ls /data/classes/BTEP/unix_on_biowulf_2023_documents/
```

```
SRR1553606_fastqc  unix_on_biowulf_2023  unix_on_biowulf_2023.zip
```

Remember that you can include options in Unix commands, which will alter how the command runs. In the above, `ls` just spat out the contents of the data folder, but we do not see details

regarding file size, date and time the file or folder was last modified, etc. To see more details we can append `-l` to `ls`.

```
ls -l /data/classes/BTEP/unix_on_biowulf_2023_documents/
```

```
total 56
drwxrwsrwx. 2 wuz8 GAU  4096 Jan  5 10:48 SRR1553606_fastqc
drwxrwsrwx. 2 wuz8 GAU  4096 Jan 12 17:46 unix_on_biowulf_2023
-rwxrwxrwx. 1 wuz8 GAU 41734 Jan  5 10:48 unix_on_biowulf_2023.zip
```

Earlier, we used the `man` command to view the manual for `pwd`. With `ls`, we can also append the `--help` option to pull up help documenations (Figure 9).

```
ls --help
```

```
[wuz8@biowulf wuz8]$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all                  do not ignore entries starting with .
  -A, --almost-all           do not list implied . and ..
      --author               with -l, print the author of each file
  -b, --escape               print C-style escapes for nongraphic characters
      --block-size=SIZE      scale sizes by SIZE before printing them; e.g.,
                               '--block-size=M' prints sizes in units of
                               1,048,576 bytes; see SIZE format below
  -B, --ignore-backups       do not list implied entries ending with ~
  -c                         with -lt: sort by, and show, ctime (time of last
                               modification of file status information);
                               with -l: show ctime and sort by name;
                               otherwise: sort by ctime, newest first
  -C                         list entries by columns
      --color[=WHEN]         colorize the output; WHEN can be 'never', 'auto',
                               or 'always' (the default); more info below
  -d, --directory            list directories themselves, not their contents
  -D, --dired                generate output designed for Emacs' dired mode
  -f                         do not sort, enable -aU, disable -ls --color
  -F, --classify             append indicator (one of */=>@|) to entries
```

Figure 9: Getting help with the ls command using the --help option.

Biowulf status

You can use the Biowulf status page *(https://hpc.nih.gov/systems/status/)* to check for outages.

Lesson 3: File/directory permissions and more on navigating Unix file system

Quick review

In the previous lesson, we learned about limitations on the tasks that we can perform in the various spaces on Biowulf. We also learned to view pertinent user account information on the Biowulf user dashboard. Finally, we introduced ourselves to navigating directories and listing directory content. The following commands were introduced.

- `pwd` (to check the present working directory - ie. the directory that we are currently in)
- `ls` (to list directory content)
- `mkdir` (to make a new directory)
- `cd` (to change between directories)

Lesson objectives

After this lesson, we should

- Be familiar with copying content from one directory to another
- Know how to use the `ls` command with different options to view relevant information regarding files and directories
- Understand file and directory permissions and be able to modify these
- Know how to remove files

Unix commands that we will visit in this lesson

- `cp` (to copy files or directory)
- `ls` (to list directory content)
- `chmod` (to change file and directory permission)
- `mkdir` (to make new directory)
- `cd` (to change directory)
- `rm` (to remove content)

Logging into Biowulf

Before getting started, log into Biowulf using either the student account or your personal account. Open the Command Prompt (Windows) or Terminal (Mac) and use the `ssh` command to connect. Remember that the password does not appear as we are typing it.

```
ssh username@biowulf.nih.gov
```

Change into your data directory

Remember to work in your data directory. Take a moment to change into your data directory using the `cd` command with the arguments below

- Argument: the path of the directory you want to change into (here it is /data/username, an absolute path because it is starting at the root)

```
cd /data/username
```

Use `pwd` to confirm that you have successfully changed into your data folder.

```
pwd
```

```
/data/username
```

Copying of files or directories

There is a folder called unix_on_biowulf_2023_documents in the /data/classes/BTEP folder. We can see it using the `ls` command with the options and arguments below

- Option: `-l` for long and detailed view
- Argument: directory that we want to list contents of (ie. /data/classes/BTEP)

Note: if a directory is not specified with `ls`, we will see the contents of the present working directory

```
ls -l /data/classes/BTEP
```

```
drwxrwsr-x.  4 wuz8       GAU              4096 Feb  9 21:28 unix_on_biowulf_2023_documents
```

Copy a folder

The first task for today is to copy the unix_on_biowulf_2023_documents folder to your data directory. We use the `cp` command with the following options and arguments.

- Option: `-r` to copy recursively (ie. the folder and all of its contents)

- Argument: the path of the folder that we like make a copy of (ie. /data/classes/BTEP/unix_on_biowulf_2023_documents)
- Argument: where to copy the folder to (ie. in the present working directory or ".")

```
cp -r /data/classes/BTEP/unix_on_biowulf_2023_documents .
```

Use `ls -l` to see if the unix_on_biowulf_2023_documents has been successfuly copied. If it has then change into this folder using `cd` and then `ls -l` to see the content of this directory.

```
ls -l
```

```
drwxr-x---.  4 wuz8 _  wuz8           4096 Feb 27 12:44 unix_on_biowulf_2023_documents
```

```
cd unix_on_biowulf_2023_documents
```

```
ls -l
```

```
total 130
drwxr-x---. 2 wuz8 wuz8  4096 Jan 12 17:40 SRR1553606_fastqc
drwxr-x---. 5 wuz8 wuz8  4096 Jan 12 17:40 unix_on_biowulf_2023
-rwxr-x---. 1 wuz8 wuz8 41734 Jan 12 17:40 unix_on_biowulf_2023.zip
```

Next, change into the unix_on_biowulf_2023 folder.

```
cd unix_on_biowulf_2023
```

Then list the contents.

```
ls -l
```

```
total 257
-rwxr-x---. 1 wuz8 wuz8  31666 Jan 12 17:47 counts.csv
-rwxr-x---. 1 wuz8 wuz8 104473 Jan 12 17:47 results.csv
-rwxr-x---. 1 wuz8 wuz8     84 Jan 12 17:47 text_1.txt
```

Copy a file

Earlier, we used `cp` with the `-r` option to recursively copy the unix_on_biowulf_2023_documents directory and all of its contents to the data directory. Suppose we want to make a copy of just one file (the counts.csv file) in the unix_on_biowulf_2023 subfolder of unix_on_biowulf_2023_documents, how would we do this? We could use the `cp` command with the following arguments.

- Argument: File to make a copy of (ie. counts.csv)
- Argument: Name of the copy (ie. counts_copy.csv)

```
cp counts.csv counts_copy.csv
```

Supposed that we want to make a copy of text_1.txt and call it text_1_copy.txt, we can use the `cp` command again.

```
cp text_1.txt text_1_copy.txt
```

Now, if we wanted to make a copy of counts.csv and place it one directory up in the unix_on_biowulf_2023_documents folder then we can use the command below, where "../" represents go back one directory.

```
cp counts.csv ../counts_copy_1.csv
```

Again, "../" denotes one directory back, so the following `ls` command will list the content of the unix_on_biowulf_2023_documents folder and indeed we see counts_copy_1.csv.

```
ls ../
```

```
counts_copy_1.csv  SRR1553606_fastqc  unix_on_biowulf_2023
unix_on_biowulf_2023.zip
```

File and directory permissions

When we use `ls -l` we are listing contents of a directory in the long view and are able to see additional details about a file or directory such as permissions, file size, and date and time of last modification (Figure 1).

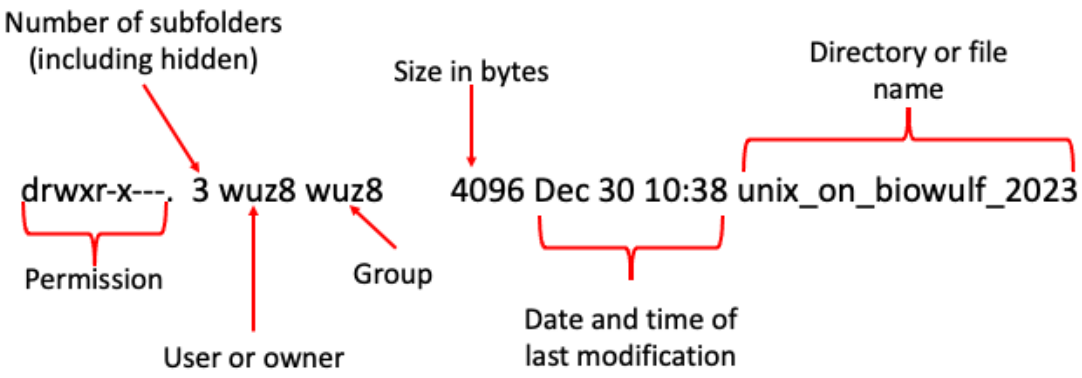


Figure 1: Explanation of the results from `ls -l`.

The column "drwr-x---" in Figure 1 tells us the permission (ie. who can read - r, write - w, or execute - x contents of the file or directory), which is an important aspect of work in Unix systems like Biowulf. Figure 2 gives a breakdown of the information provided in the permission block.

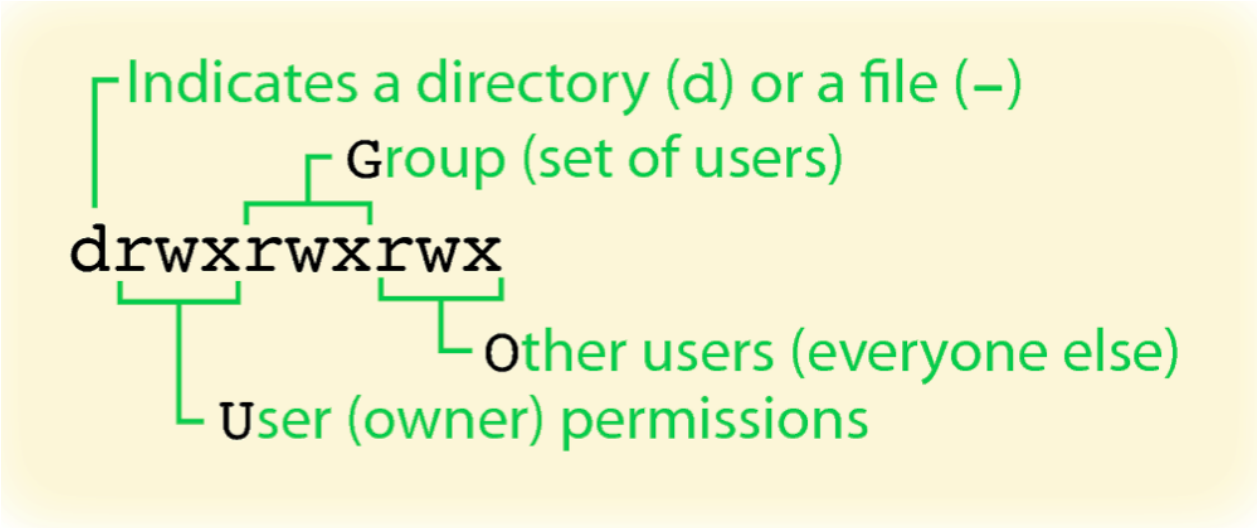


Figure 2: The permissions are divided into three chunks of "rwx", corresponding to read, write, and execution privileges of the file/directory user or owner, others in the group, and everyone else. If the permission begins with d, then we are looking at a directory. If the permission begins with "-", then we are looking at file. Source: UF Research Computing Training *(https://training.it.ufl.edu/media/trainingitufledu/documents/research-computing/RC_UpAndRunning.pdf)*

Modifying permissions

The command for modifying permissions is `chmod`. If we append `--help` to `chmod`, then we can see how to use it.

```
chmod --help
```

```
Usage: chmod [OPTION]... MODE[,MODE]... FILE...
  or:  chmod [OPTION]... OCTAL-MODE FILE...
  or:  chmod [OPTION]... --reference=RFILE FILE...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of RFILE.

  -c, --changes   like verbose but report only when a change is made
  -f, --silent, --quiet  suppress most error messages
  -v, --verbose          output a diagnostic for every file processe
      --no-preserve-root  do not treat '/' specially (the default)
      --preserve-root    fail to operate recursively on '/'
      --reference=RFILE  use RFILE's mode instead of MODE values
  -R, --recursive        change files and directories recursively
      --help     display this help and exit
      --version  output version information and exit

Each MODE is of the form
'[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
For complete documentation, run: info coreutils 'chmod invocation'
```

To use `chmod`, we need to be aware that

- u is user or owner
- g is group
- o is others
- "-" is used to remove a permission
- "+" is used to add a permission
- "=" sets permission

We can also numerically set permissions where

- 0: No permission
- 1: Execute permission
- 2: Write permission
- 3: Execute and write permission (1+2=3)

- 4: Read permission
- 5: Read and execute permission (1+4=5)
- 6: Read and write permission (2+4=6)
- 7: All permission (1+2+4=7)

Now, change back up one directory to the unix_on_biowulf_2023_documents folder. In Unix ".." denotes one directory up.

```
cd ..
```

If we removed the user's read privilege of the unix_on_biowulf_2023 folder and then list the contents of this folder, a "permission denied" message will appear. In the `chmod` command below, we use the following options and arguments

- Option: u denotes user or owner
- Option: - denotes to remove
- Option: r denotes read permission (thus, u-r denotes remove read permission from user or owner)
- Argument: file or folder that we like to modify the permission for (ie. the folder unix_on_biowulf_2023)

```
chmod u-r unix_on_biowulf_2023
```

```
ls -l unix_on_biowulf_2023
```

```
ls: cannot open directory unix_on_biowulf_2023: Permission denied
```

The error permission denied appears because the read or r permission for the owner has been removed from unix_on_biowulf_2023.

```
ls -l
```

```
total 258
-rwxr-x---. 1 wuz8 wuz8 31666 Jan 21 12:26 counts_copy_1.csv
drwxr-x---. 2 wuz8 wuz8  4096 Jan 21 12:25 SRR1553606_fastqc
d-wxr-x---. 2 wuz8 wuz8  4096 Jan 21 12:26 unix_on_biowulf_2023
-rwxr-x---. 1 wuz8 wuz8 41734 Jan 21 12:25 unix_on_biowulf_2023.zip
```

To add the user read permission back we will do the following (ie. replace "-" with "+").

```
chmod u+r unix_on_biowulf_2023
```

Listing the content of our data directory, we will see that the read permission has been added back for the user for the unix_on_biowulf_2023 folder.

```
ls -l
```

```
total 258
-rwxr-x---. 1 wuz8 wuz8 31666 Jan 21 12:26 counts_copy_1.csv
drwxr-x---. 2 wuz8 wuz8  4096 Jan 21 12:25 SRR1553606_fastqc
drwxr-x---. 2 wuz8 wuz8  4096 Jan 21 12:26 unix_on_biowulf_2023
-rwxr-x---. 1 wuz8 wuz8 41734 Jan 21 12:25 unix_on_biowulf_2023.zip
```

We are now able to list the contents of the unix_on_biowulf_2023 directory because we have just reassigned the user the read permission.

```
ls unix_on_biowulf_2023
```

```
counts_copy.csv  counts.csv  results.csv  text_1_copy.txt  text_1.tx
```

Let's use the numbering scheme to add read permission for others to the unix_on_biowulf_2023 folder. The options and arguments for the `chmod` command below is as follows

- Option: the first number assigns user or owner permissions (here 7 is used to assign read, write, and execute permission)
- Option: the second number assigns group permissions (here 5 is used to assign read and execute permission)
- Option: the third number assigns other permissions (here 4 is used to assign read permission)
- Argument: name of the file or folder that we like to modify permissions for (ie. unix_on_biowulf_2023)

```
chmod 754 unix_on_biowulf_2023
```

List the content of our unix_on_biowulf_2023_documents folder to see the changes.

```
ls -l
```

```
```

```
total 258
-rwxr-x---. 1 wuz8 wuz8 31666 Jan 25 10:56 counts_copy_1.csv
drwxr-x---. 2 wuz8 wuz8  4096 Jan 25 10:52 SRR1553606_fastqc
drwxr-xr--. 2 wuz8 wuz8  4096 Jan 25 10:56 unix_on_biowulf_2023
-rwxr-x---. 1 wuz8 wuz8 41734 Jan 25 10:52 unix_on_biowulf_2023.zip
```

Notice that while we added read permission to others for the unix_on_biowulf_2023 folder, the contents inside the folder do not have read permission for others.

```
ls -l unix_on_biowulf_2023
```

```
total 385
-rwxr-x---. 1 wuz8 wuz8  31666 Jan 25 10:56 counts_copy.csv
-rwxr-x---. 1 wuz8 wuz8  31666 Jan 25 10:52 counts.csv
-rwxr-x---. 1 wuz8 wuz8 104473 Jan 25 10:52 results.csv
-rwxr-x---. 1 wuz8 wuz8     84 Jan 25 10:56 text_1_copy.txt
-rwxr-x---. 1 wuz8 wuz8     84 Jan 25 10:52 text_1.txt
```

Note that `chmod` has a recursive mode, denoted by the option `-R`. For a given directory, `-R` will change permission for the directory as well its contents.

Let's use the `-R` option to add read permission to others for the unix_on_biowulf_2023 folder and its contents. The options and arguments for the `chmod` command below are

- Option: `-R` to change permissions recursively
- Option: o to change permission for others
- Option: "+" to add permission
- Option: r denotes read permission (thus, o+r denotes add read permission for others)
- Argument: File or folder to modify permission for (ie. the unix_on_biowulf_2023 folder)

```
chmod -R o+r unix_on_biowulf_2023
```

Listing the content of the unix_on_biowulf_2023_documents folder, we see that the read permission for unix_on_biowulf_2023 has been added for the others.

```
ls -l
```

```
total 258
-rwxr-x---. 1 wuz8 wuz8 31666 Jan 30 22:52 counts_copy_1.csv
drwxr-x---. 2 wuz8 wuz8  4096 Jan 30 22:48 SRR1553606_fastqc
```

```
drwxr-xr--. 2 wuz8 wuz8  4096 Jan 30 22:52 unix_on_biowulf_2023
-rwxr-x---. 1 wuz8 wuz8 41734 Jan 30 22:48 unix_on_biowulf_2023.zip
```

Listing the contents of the unix_on_biowulf_2023 folder, we see that others has read permission for everything inside it.

```
ls -l unix_on_biowulf_2023
```

```
total 385
-rwxr-xr--. 1 wuz8 wuz8  31666 Jan 23 21:05 counts_copy.csv
-rwxr-xr--. 1 wuz8 wuz8  31666 Jan 23 21:05 counts.csv
-rwxr-xr--. 1 wuz8 wuz8 104473 Jan 23 21:05 results.csv
-rwxr-xr--. 1 wuz8 wuz8     84 Jan 23 21:06 text_1_copy.txt
-rwxr-xr--. 1 wuz8 wuz8     84 Jan 23 21:05 text_1.txt
```

Other commands that might come in handy are `chown` and `chgrp`.

- `chown` (to change owner of a file)
- `chgroup` (to change group ownership of a file)

More on the ls command and viewing directory content

Let's stay in the /data/username/unix_on_biowulf_2023_documents folder for this exercise (change into if not in this directory already).

Note that the size of our content are listed as bytes. We can get a more human readable form of the size by appending the `-h` option to `ls -l`. Again we can use `ls --help` to find out about the `-h` flag. Of interest is that options for Unix commands can also be written in the long form preceded by "--" (ie. `-h` is the same as `--human-readable`).

```
ls --help
```

```
-h, --human-readable  with -l, print sizes in human readable format
                        (e.g., 1K 234M 2G)
```

Appending the `-h` option, we see that the content sizes in unix_on_biowulf_2023 are now expressed as kilobytes (K).

```

```

```
ls -lh unix_on_biowulf_2023/
```

```
total 385K
-rwxrwx---. 1 wuz8 wuz8  31K Jan 21 12:26 counts_copy.csv
-rwxrwx---. 1 wuz8 wuz8  31K Jan 21 12:25 counts.csv
-rwxrwx---. 1 wuz8 wuz8 103K Jan 21 12:25 results.csv
-rwxrwx---. 1 wuz8 wuz8   84 Jan 21 12:26 text_1_copy.txt
-rwxrwx---. 1 wuz8 wuz8   84 Jan 21 12:25 text_1.txt
```

Speaking of file sizes, we can use the `checkquota` command to check disk usage. This should give the same result as that shown in the user dashboard.

```
checkquota
```

```
Mount       Used      Quota   Perrcent    Files    Limit  Percent
/data:  478.8 GB     2.0 TB     23.38%     1763 31457280    0.01%
/home:    2.0 GB    16.0 GB     12.44%    14788      n/a    0.00%
```

Deleting files

Let's change into the unix_on_biowulf_2023 folder.

```
cd unix_on_biowulf_2023
```

Then list the content

```
ls
```

```
counts_copy.csv counts.csv results.csv text_1_copy.txt text_1.txt
```

Let's remove counts_copy.csv and text_1_copy.txt using the `rm` command with the following argument.

- Argument: file that we like to delete (ie. counts_copy.csv and text_1_copy.txt)

```
rm counts_copy.csv
```

```
rm text_1_copy.txt
```

Listing the contents of unix_on_biowulf_2023, we see that counts_copy.csv and text_1_copy.txt was removed.

```
ls
```

```
counts.csv  results.csv  text_1.txt
```

To remove a directory, we can use `rm -r` and this will remove the directory and everything in it recursively. If we have an empty directory, we can use `rmdir`. We will talk more about the rm command in the next class.

Lesson 4: Working with files in Unix - moving, renaming, and removing files and directories

Quick review

In previous the lesson, we learned how to copy content (`cp`), change file and directory permissions (`chmod`), and remove files (`rm`).

Lesson objectives

In this lesson we will achieve the following.

- Know how to move and rename files and directories
- Be able to remove directories.

Unix commands that we will visit in this lesson

- `cp` (to copy files or directories)
- `mv` (to move/rename files or directories)
- `rm` (to remove/delete files or directories)
- `rmdir` (to remove empty directory)
- `tree` (to view directory tree)

Connecting to Biowulf

Before getting started, connect to Biowulf.

```
ssh username@biowulf.nih.gov
```

Moving and renaming files and directories

For this portion of the lesson, change into your data directory.

```
cd /data/username
```

Then, copy over the folder unix_on_biowulf_2023_documents from /data/classes/BTEP.

```
cp -r /data/classes/BTEP/unix_on_biowulf_2023_documents .
```

Change into unix_on_biowulf_2023_documents after copying.

```
cd unix_on_biowulf_2023_documents
```

Then, change into the folder unix_on_biowulf_2023.

```
cd unix_on_biowulf_2023
```

Let's make a copy of counts.csv called counts_copy.csv and a copy of text_1.txt called text_1_copy.txt.

```
cp counts.csv counts_copy.csv
```

```
cp text_1.txt text_1_copy.txt
```

Next, we will create a folder called lesson4 inside the present working directory (ie. unix_on_biowulf_2023).

```
mkdir lesson4
```

Moving files from one directory to another

To move files from one directory to another, we use the `mv` command. Let's move the copied files (counts_copy.csv and text_1_copy.txt) to the lesson4 folder. The command syntax is `mv` followed the following arguments

- Argument: file that we want to move (ie. counts_copy.csv and text_1_copy.txt)
- Argument: folder we like to move the file to (ie. lesson4)

```
mv counts_copy.csv lesson4
```

```
mv text_1_copy.txt lesson4
```

After moving, let's list the contents of unix_on_biowulf_2023 to make sure the move was successful.

```
ls
```

As we hoped, counts_copy.csv and text_1_copy.txt are not in this folder.

```
counts.csv  lesson4  results.csv  text_1.txt
```

Renaming files

Let's now change into the lesson4 folder and list its contents.

```
cd lesson4
```

```
ls
```

Indeed, we see that the two files now reside in the lesson4 folder.

```
counts_copy.csv  text_1_copy.txt
```

Let's rename counts_copy.csv to counts_lesson4.csv using the `mv` command, with the arguments below.

- Argument: the file that we want to rename
- Argument: the name of the new file

```
mv counts_copy.csv counts_lesson4.csv
```

Listing the contents of the lesson4 folder, we will see that we now have the file counts_lesson4.csv rather than counts_copy.csv.

```
ls
```

```
counts_lesson4.csv  text_1_copy.txt
```

Let's cp the counts_lesson4.csv and text_1_copy.txt back to the unix_on_biowulf_2023 folder, which is just one directory up. Recall that we can use ".." to denote one directory up.

```
cp counts_lesson4.csv ../
```

```
cp text_1_copy.txt ../
```

Now, let's go one directory back to the unix_on_biowulf_2023 folder. Again, we will use ".." to indicate one directory up.

```
cd ..
```

Listing the contents of unix_on_biowulf_2023, we see that counts_lesson4.csv and text_1_copy.txt have been copied over.

```
ls
```

```
counts.csv counts_lesson4.csv lesson4 results.csv text_1_copy.txt
text_1.txt
```

Renaming folders

We can also use the `mv` command to rename folders. To see this let's make a copy of the lesson4 folder called lesson4_new. Again, we need to append `-r` to the `cp` command since we are copying a directory.

```
cp -r lesson4 lesson4_new
```

Listing the contents of unix_on_biowulf_2023, we see that lesson4_new has been added.

```
ls
```

```
counts.csv counts_lesson4.csv lesson4 lesson4_new results.csv
text_1_copy.txt text_1.txt
```

Now, we want to use `mv` to rename lesson4_new to lesson4_copy where the arguments are

- Argument: the folder that we want to rename
- Argument: new name for the folder

```
mv lesson4_new lesson4_copy
```

Listing the contents of unix_on_biowulf_2023, we see that we now have a folder called lesson4_copy in place of lesson4_new.

```
ls
```

```
counts.csv counts_lesson4.csv lesson4 lesson4_copy results.csv
text_1_copy.txt text_1.txt
```

Here, we will make a copy of the folder lesson4_copy and name it lesson4_copy_1. Again, using `cp -r` to copy recursively.

```
cp -r lesson4_copy lesson4_copy_1
```

Moving a folder into another folder

Next, we will create a new folder called new_house.

```
mkdir new_house
```

Then we will move lesson4_copy_1 to the folder new_house. Moving a folder to another only works when the name of the destination folder already exists. In the `mv` command below, the arguments are

- Argument: folder we like to move (ie. lesson4_copy_1)
- Argument: destination that we like to move the folder to (ie. new_house)

```
mv lesson4_copy_1/ new_house/
```

Now, if we listed the contents of new_house, we will see that it has as a subfolder, lesson4_copy_1.

```
ls -l new_house
```

```
total 1
drwxr-x---. 2 wuz8 wuz8 4096 Jan 12 20:15 lesson4_copy_1
```

If we used the `tree` command, we can get a hierarchical tree print out of contents inside a folder. For example, doing this for the folder new_house, we will see the lesson4_copy_1 subfolder, as well as the files within lesson4_copy_1.

```
tree new_house
```

```
new_house/
└── lesson4_copy_1
    ├── counts_lesson4.csv
    └── text_1_copy.txt
```

Removing or deleting

The command to remove or delete something in Unix is `rm`. To explore this command, let's change into the lesson4_copy folder.

```
cd lesson4_copy
```

Next, list the contents to see the contents of this folder.

```
ls
```

```
counts_lesson4.csv  text_1_copy.txt
```

To remove text_1_copy.txt, we will use the `rm` command and provide as the following arguments, the file or directory that we wish to remove.

```
rm text_1_copy.txt
```

If we listed the contents of lesson4_copy, we will see that text_1_copy.txt has disappeared.

```
ls
```

```
counts_lesson4.csv
```

Let's copy text_1_copy.txt back to the lesson4_copy folder. There is a copy of the text_1_copy.txt file that resides one directory up in the lesson4 subfolder of the

unix_on_biowulf_2023 folder; thus, with the `cp` command, we can use ".." to denote one directory up, which is the unix_on_biowulf_2023 folder, and then "/lesson4" to get into the lesson4 folder. Because we are still in the folder lesson4_copy, we can tell `cp` to copy to here, in the present working directory, which is denoted by ".".

```
cp ../lesson4/text_1_copy.txt .
```

Listing the content again, we see that text_1_copy.txt has been added back.

```
ls
```

```
counts_lesson4.csv  text_1_copy.txt
```

Recall that in Unix, we do not have a trash can or recycle bin to place stuff that we deleted and then recover. At best, Biowulf provides snapshots that we can use for reovery purposes. Looking the help document for `rm`, we see that there is an option `-i` that will prompt and ask before we remove something. This will allow us to review before deleting.

```
rm --help
```

```
-i                    prompt before every removal
```

Let's try `rm -i` with text_1_copy.txt.

```
rm -i text_1_copy.txt
```

We will be asked whether we really want to remove. We can respond with no at the prompt if we do not want to remove the file.

```
rm: remove regular file ‘text_1_copy.txt’?
```

To remove a folder, we can use the `-r` option with `rm`. The `-r` option will delete recursively (similar logic to `cp -r`). Let's go back up one folder for this exercise, use ".." to denote one folder up.

```
cd ..
```

Let's remove lesson4_copy. We will include `-i` in the `rm` command below.

```
rm -ir lesson4_copy
```

Type yes when prompted and the list the contents of the unix_on_biowulf_2023 folder.

Note that because lesson4_copy is not empty, we will be asked whether we want to descend into the folder and then confirm to delete everything. We will say yes to all.

```
rm: descend into directory ‘lesson4_copy’? yes
rm: remove regular file ‘lesson4_copy/text_1_copy.txt’? yes
rm: remove regular file ‘lesson4_copy/counts_lesson4.csv’? yes
rm: remove directory ‘lesson4_copy’? yes
```

```
ls
```

Indeed, the folder lesson4_copy is gone.

```
counts.csv counts_lesson4.csv lesson4 new_house results.csv
text_1_copy.txt text_1.txt
```

To remove an empty directory, we can use `rmdir`. For this exercise, let's make a new directory called empty_folder.

```
mkdir empty_folder
```

Listing the contents of unix_on_biowulf_2023, we see the empty_folder was added.

```
ls
```

```
counts.csv counts_lesson4.csv empty_folder lesson4 new_house
results.csv text_1_copy.txt text_1.txt
```

Now, we use `rmdir` to delete the empty_folder.

```
rmdir empty_folder
```

Listing the contents of unix_on_biowulf_2023 again, we see that empty_folder is no longer there.

```
ls
```

```
counts.csv counts_lesson4.csv lesson4 new_house results.csv
text_1_copy.txt text_1.txt
```

Lesson 5: Interactive sessions, modules, and bioinformatics applications on Biowulf

Quick review

In the previous lesson, we learned to move, rename, and remove files as well as directories in Unix. Commands that we learned include

- `mv` (to move or rename file or directories)
- `tree` (to generate a directory tree)
- `rm` (to remove files or directories)
- `rmdir` (to remove empty directories)

Lesson objectives

After this lesson, we should be able to

- Request an interactive session on Biowulf
- Know how to find out what applications are available on Biowulf
- Know how to download high throughput sequencing data from NCBI SRA
- Be able to assess quality of high throughput sequencing data

Unix commands that we will learn in this lesson

- `sinteractive` (to request an interactive session on Biowulf)
- `module` (to view, load, or unload applications that are installed on Biowulf)
- `fastq-dump` (to download FASTQ files from NCBI SRA)
- `head` (to view beginning of a file; defaults to the first 10 lines)
- `fastqc` (to assess sequencing data quality)

Requesting an interactive session

Recall that we are not supposed to use the login nodes to perform any computation intensive tasks on Biowulf. Instead, we should either submit a job (if staying in the log in node) with sufficient resources requested or request an interactive session if we are going to be doing some testing and development.

The log in node is meant for the following (Source: Biowulf accounts and log in node *(https://youtu.be/qiWGxrLl6AY?t=217)*)

- Submitting jobs (main purpose)
- Editing/compiling code
- File management
- File transfer
- Brief testing of code or debugging (under 20 minutes)

Today, we are going to learn to request an interactive session, which is suitable for

- testing/debugging cpu-intensive code
- pre/post-processing of data
- use of graphical application

To start an interactive node, type `sinteractive` at the prompt and press Enter/Return on your keyboard.

```
sinteractive
```

You will see a message similar to that shown below as the resource request is being processed and allocated. We only need to use the `sinteractive` command once per session. If we try to start an interactive node on top of another interactive node, we will get a message asking why we want to start another node. Note that our prompt switches from username@biowulf to username@cn#### (where #### is a number) to denote that we are now on a compute note rather than the log in node. In this example, I was connected to cn4269. Note the job ID of 55405280, which we will come back to in a bit. Ignore the errors that show up.

```
[wuz8@biowulf ~]$ sinteractive
salloc: Pending job allocation 55405280
salloc: job 55405280 queued and waiting for resources
salloc: job 55405280 has been allocated resources
salloc: Granted job allocation 55405280
salloc: Waiting for resource configuration
salloc: Nodes cn4269 are ready for job
srun: error: x11: no local DISPLAY defined, skipping
error: unable to open file /tmp/slurm-spank-x11.55405280.0
slurmstepd: error: x11: unable to read DISPLAY value
[wuz8@cn4269 ~]$
```

The default sinteractive allocation is 1 core (2 CPUs) and 0.768 GB/CPU of memory and a walltime of 8 hours. We can use the `jobhist` command followd by `$SLURM_JOBID` (recall the job ID of 55405280 above) and this will give us information on the resources that we asked for as well as the amount of time we have spent on the job.

```
jobhist $SLURM_JOBID
```

Note that while the MemReq shows 2 GB of RAM was requested, it is actually 1.5 GB of RAM (0.768 GB x 2 CPU). Biowulf just rounded to the nearest integer.

```
JobId              : 59427893
User               : wuz8
Submitted          : 20230228 12:14:05
Started            : 20230228 12:17:25
Ended              :

Jobid        Partition       State  Nodes  CPUs      Walltime       Runtime         MemReq  MemUsed  Nodelist
59427893    interactive     RUNNING      1     2       8:00:00         11:07           2GB      1MB  cn4275
```

SLURM_JOBID is known as an environmental variable in the Unix world (see below for the definition). We can set environmental variables for many things including long directory paths that we would not want to repeatedly type. To reference an environmental variable, we prefix the "$" in front of it.

> "Environment variables or ENVs basically define the behavior of the environment. They can affect the processes ongoing or the programs that are executed in the environment." -- https://www.geeksforgeeks.org/environment-variables-in-linux-unix/ *(https://www.geeksforgeeks.org/environment-variables-in-linux-unix/)*.

NCI CCR partition

Note that when we ran `jobhist $SLURM_JOBID` above, a table with information regarding the interactive session appears. One of the columns is labeled partition and it tells us that we were taken to the interactive partition upon requesting an interactive session.

> "Partitions define limitations that restrict the resources that can be requested for a job submitted to that partition. The limitations affect the maximum run time, the amount of memory, and the number of available CPU cores (which are called CPUs in Slurm)." -- https://wiki.hpcuser.uni-oldenburg.de/index.php?title=Partitions *(https://wiki.hpcuser.uni-oldenburg.de/index.php?title=Partitions)*. "Jobs should be submitted to the partition that best matches the required resources." -- https://wiki.hpcuser.uni-oldenburg.de/index.php?title=Partitions *(https://wiki.hpcuser.uni-oldenburg.de/index.php?title=Partitions)*.
>
> "NCI-CCR has funded 153 nodes (4548 physical cores, 9096 cpus with hyperthreading) in the Biowulf cluster, and CCR users have priority access to these nodes. This priority status will last until Febuary 20, 2021 (FY2017 funded nodes), Apr 15, 2022 (FY2018 funded nodes) and May 18, 2023 (FY2019 funded nodes)." -- Biowulf NCI CCR partition *(https://hpc.nih.gov/docs/ccr.html)*

To request an interactive session in the CCR partition use

```

```

```
sinteractive --constraint=ccr
```

In the above, `--constraint` is an option that can be used with `sinteractive` to specify the partition in which we want to run our task. Other useful options can be found in the Biowulf user guide *(https://hpc.nih.gov/docs/userguide.html)*.

To learn more about partitions on Biowulf see https://hpc.nih.gov/docs/userguide.html *(https://hpc.nih.gov/docs/userguide.html)*. Use `freen` to see the available and free resources for the different Biowulf Partitions. To check on limitations for Biowulf partitions, use the `batchlim` command.

To terminate an interactive session, type `exit` at the prompt.

Requesting lscratch space

Remember that each node in Biowulf has some amount of space that could be used to store temporary data (lscratch). These can be used for applications that write many temporary files, such as the sratoolkit. To request lscratch space, include the `--gres` option in `sinteractive`. The option `--gres` stands for generic resource.

If we terminated our current interactive session using `exit` we can then request another interactive session with lscratch space.

```
exit
```

Successful exit of an interactive session produces the message below where interactive_session_job_id is the job ID of the interactive session.

```
exit
salloc: Relinquishing job allocation interactive_session_job_id
```

In the example below, we set `--gres` (ie. generic resource) to lscratch, followed by ":" and then the amount of space we need (in gigabytes or GB, here we ask for 15 GB, so the construct is lscratch:15). GB is the default space size unit when requesting space in lscratch *(https://hpc.nih.gov/docs/userguide.html)*.

```
sinteractive --gres=lscratch:15
```

An application that requires lscratch is the sratoolkit, which can be used to download high throughput sequencing data from NCBI SRA.

Modules

Biowulf staff has installed many applications, including those used in genomic data analysis. In general, to view the applications that are available on Biowulf, we can use the `module` command, with its `avail` subcommand. This will essentially print out a list of applications that are on Biowulf and we can use the up and down arrows to navigate and view the list. We hit "q" to exit this list.

```
module avail
```

To list only the default version of each application, include the `-d` option in `module avail`.

```
module -d avail
```

To check if a specific application is available, you can append the name of the module after `module avail`. For instance, we do that with the genomic sequencing Star aligner Bowtie below.

```
module avail star
```

```
------------------------------------------ /data/classes/BTEP/apps/modules ------------------------------------------
   biostars/1.0

--------------------------------------------- /usr/local/lmod/modulefiles ---------------------------------------------
   STAR-Fusion/1.2.0    STAR-Fusion/1.7.0         STAR/2.5.2a    STAR/2.7.0f    STAR/2.7.6a
   STAR-Fusion/1.3.1    STAR-Fusion/1.9.1         STAR/2.5.2b    STAR/2.7.1a    STAR/2.7.8a
   STAR-Fusion/1.3.2    STAR-Fusion/1.10.0        STAR/2.5.3a    STAR/2.7.2a    STAR/2.7.9a    (D)
   STAR-Fusion/1.5.0    STAR-Fusion/1.11.0 (D)    STAR/2.5.4a    STAR/2.7.3a    starseqr/0.6.7
   STAR-Fusion/1.6.0    STAR/2.4.0k               STAR/2.6.1c    STAR/2.7.5b

  Where:
   D:  Default Module

Module defaults are chosen based on Find First Rules due to Name/Version/Version modules found in the module tree.
See https://lmod.readthedocs.io/en/latest/060_locating.html for details.

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

We can use the `whatis` subcommand to see information regarding a specific tool and also to confirm if Biowulf has it installed. For instance, we can check for fastqc, which is an application used to assess quality of high throughput sequencing data. The output provides a description of what the tool does and the default version if we load the tool. The `whatis` subcommand is case sensitive.

```
module whatis fastqc
```

```
fastqc/0.11.9       : fastqc: It provide quality control functions to next gen sequencing data.
fastqc/0.11.9       : Version: 0.11.9
```

If do we `module whatis fast` and then hit the tab button (ie. to tab complete), we can see the applications that begins with fast. Here, we have several different versions of fastqc available.

```
fastp                 fastqc/0.11.2         fastqc/0.11.9          fastq_screen/0.15.0    fastqtools/0.8.3
fastp/0.20.1          fastqc/0.11.4         fastqc/0.11.9-highmem  fastq_screen/0.15.2    fastsurfer
fastp/0.23.1          fastqc/0.11.5         fastq_screen           fastqtools             fastsurfer/c5e9677
fastp/0.23.2          fastqc/0.11.6         fastq_screen/0.14.0    fastqtools/0.8         fastxtoolkit
fastqc                fastqc/0.11.8         fastq_screen/0.14.1    fastqtools/0.8.1       fastxtoolkit/0.0.14
```

To load an application we can use `module load`. Let's load the sratool kit and fastqc. By default, the latest version of an application is loaded.

```
module load sratoolkit
```

```
Running on cn4303  ...
[+] Loading sratoolkit 3.0.2  ...
```

```
module load fastqc
```

```
[+] Loading fastqc  0.11.9
```

Note that we can change the version of an application that we want to use (provided that Biowulf has it installed). For instance version 0.11.2 of fastqc rather than 0.11.9, all we have to do is to reload the application by including a "/" followed by the version number.

```
module load fastqc/0.11.2
```

```
[-] Unloading fastqc  0.11.9
[+] Loading fastqc  0.11.2

The following have been reloaded with a version change:
  1) fastqc/0.11.9 => fastqc/0.11.2
```

But, we will be using fastqc in a bit so let's reload with the latest version, which is 0.11.9.

```
module load fastqc
```

Modules that you have loaded are unloaded once you exit Biowulf, so you will need to reload again at the next sign in.

Exploring bioinformatics tools

Here, we will download some high throughput genomic sequences from NCBI SRA. The data that we will download were derived from sequencing of the Zaire Ebola virus. See the NCBI SRA page for this study *(https://www.ncbi.nlm.nih.gov/sra/?term=SRR1553606)* for more details.

We will use a command called `fastq-dump` within the `sratoolkit` to grab the first 10000 reads for this sequencing run. In the syntax for `fastq-dump`

- `--split-files` will generate two files that contains the forward and reverse reads from paired-end sequencing.
- `-X` allows us to input how many reads we want to obtain (here, we just want the first 10000 reads to save time and computation resources for this class)
- Finally, we enter the SRA accession number of the sequencing data that we want to download (SRR1553606 in this example).
- We will download this into our data directory, so change into if you are not in the data directory already.

```
cd /data/username
```

```
fastq-dump --split-files -X 10000 SRR1553606
```

Listing the contents of our data directory, we should see the two FASTQ files that were downloaded.

```
ls
```

```
SRR1553606_1.fastq
SRR1553606_2.fastq
```

We can use `head -n 4` view the first 4 lines of the SRR1553606_1.fastq file to see what a FASTQ file looks like. We will talk a bit more about the `head` command in lesson 7.

```
head -n 4 SRR1553606_1.fastq
```

Essentially, FASTQ files contain our high throughput sequencing data. Each sequence read starts with a metadata header line that begins with "@", followed by the actual sequence, then a "+" followed by a line with quality scores that tells us the error likelihood for each of the bases in the sequencing read. This pattern of four lines repeats for all of sequencing reads we have in

our FASTQ file (in this case, we should have 10000 sequencing reads because that is how many we asked fastq-dump to download).

```
@SRR1553606.1 1 length=101
ATACACATCTCCGAGCCCACGAGACCTCTCTACATCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAACAGGAGTCGCCCAGCCCTGCTCAACGAGCTGCAG
+SRR1553606.1 1 length=101
@@@FDFDFHHHHHIJGIJJHHIIJIGHGHIJGFI9DDFH?FFHIGGGH>EHGIJEECCABBDABD##################################
```

One of the things we need to do after receiving our sequencing data is to assay the quality of the data. We can do this using fastqc.

```
fastqc --help
```

From the fastqc help documents, we see that in general, to run fastqc, we just need to provide it the names of the FASTQ files.

```
fastqc seqfile1 seqfile2 .. seqfileN
```

Let's wrap up this lesson by running fastqc for SRR1553606_1.fastq and SRR1553606_2.fastq

```
fastqc SRR1553606_1.fastq SRR1553606_2.fastq
```

As it runs, we can see the analysis progress.

```
Started analysis of SRR1553606_1.fastq
Approx 10% complete for SRR1553606_1.fastq
Approx 20% complete for SRR1553606_1.fastq
Approx 30% complete for SRR1553606_1.fastq
Approx 40% complete for SRR1553606_1.fastq
Approx 50% complete for SRR1553606_1.fastq
Approx 60% complete for SRR1553606_1.fastq
Approx 70% complete for SRR1553606_1.fastq
Approx 80% complete for SRR1553606_1.fastq
Approx 90% complete for SRR1553606_1.fastq
Approx 100% complete for SRR1553606_1.fastq
Analysis complete for SRR1553606_1.fastq
Started analysis of SRR1553606_2.fastq
Approx 10% complete for SRR1553606_2.fastq
Approx 20% complete for SRR1553606_2.fastq
Approx 30% complete for SRR1553606_2.fastq
Approx 40% complete for SRR1553606_2.fastq
Approx 50% complete for SRR1553606_2.fastq
Approx 60% complete for SRR1553606_2.fastq
Approx 70% complete for SRR1553606_2.fastq
```

```
Approx 80% complete for SRR1553606_2.fastq
Approx 90% complete for SRR1553606_2.fastq
Approx 100% complete for SRR1553606_2.fastq
Analysis complete for SRR1553606_2.fastq
```

The quality assessment reports for SRR1553606_1.fastq and SRR1553606_2.fastq are written into SRR1553606_1_fastqc.html and SRR1553606_2_fastqc.html, respectively as evident when we list the contents of our data folder after fastqc has completed. To view these, we will need to transfer these to our local desktop (will discuss in Lesson 6).

In the `ls` command below, we use `-1` to list one item per row.

```
ls -1
```

```
SRR1553606_1.fastq
SRR1553606_1_fastqc.html
SRR1553606_1_fastqc.zip
SRR1553606_2.fastq
SRR1553606_2_fastqc.html
SRR1553606_2_fastqc.zip
SRR1553606_fastqc_log
SRR1553606_fastqc.sh
```

Lesson 6: Submitting batch jobs and transferring between local machine and Biowulf

Quick review:

In the previous lesson, we learned to request an interactive session on Biowulf so that we can perform more compute intensive tasks such as downloading sequencing data from NCBI SRA and subsequently assessing quality of the downloaded sequencing data. Commands that we learned include

- `sinteractive` (to request an interactive session)
- `module avail` (to view a list of software that are installed on Biowulf)
- `module whatis` (to obtain details about a specific software)
- `module load` (to load software)
- `fastq-dump` (to download sequencing data from NCBI SRA)
- `fastqc` (to assess sequencing data quality)

Lesson objectives:

After this lesson, we will

- Be able to develop a short shell script that will
    - download sequencing data from NCBI SRA
    - assay the quality of the sequencing data
- Know how to submit shell scripts as a batch job.
- Be able to transfer files between our local computer and Biowulf.

Unix commands that we will visit in this lesson

- `nano` (to open the nano editor to edit files)
- `sbatch` (to submit jobs to Biowulf)
- `squeue` (to check job status)
- `scancel` (to cancel jobs)
- `scp` (to copy content between local computer and Biowulf)

Creating shell scripts and submitting batch jobs

First, sign into Biowulf

```
ssh username@biowulf.nih.gov
```

Change into the data directory

```
cd /data/username
```

Let's then create a folder called SRR1553606_fastqc to store the FASTQ files for SRR1553606 and its fastqc output.

```
mkdir SRR1553606_fastqc
```

Next, change into SRR1553606_fastqc

```
cd SRR1553606_fastqc
```

Let's create a shell script called SRR1553606_fastqc.sh. Shell scripts have extension ".sh". This script will accomplish the following

- Download sequencing data (FASTQ files) for NCBI SRA accession SRR1553606 *(https://www.ncbi.nlm.nih.gov/sra/?term=SRR1553606)*. The study in which SRR1553606 was derived used paired end sequencing methods so we should get two FASTQ files.
- After downloading the FASTQ files, we will run fastqc to assess sequencing data quality.

Creating SRR1553606_fastqc.sh using the nano editor

Nano is a built-in Unix text editor. We can use nano to create SRR1553606_fastqc.sh. The syntax here is `nano file_to_edit` so in this example do the following.

```
nano SRR1553606_fastqc.sh
```

This will open a blank editor (since SRR1553606_fastqc.sh is a new file). At the top of the editor we see the name of the file that is opened in nano.

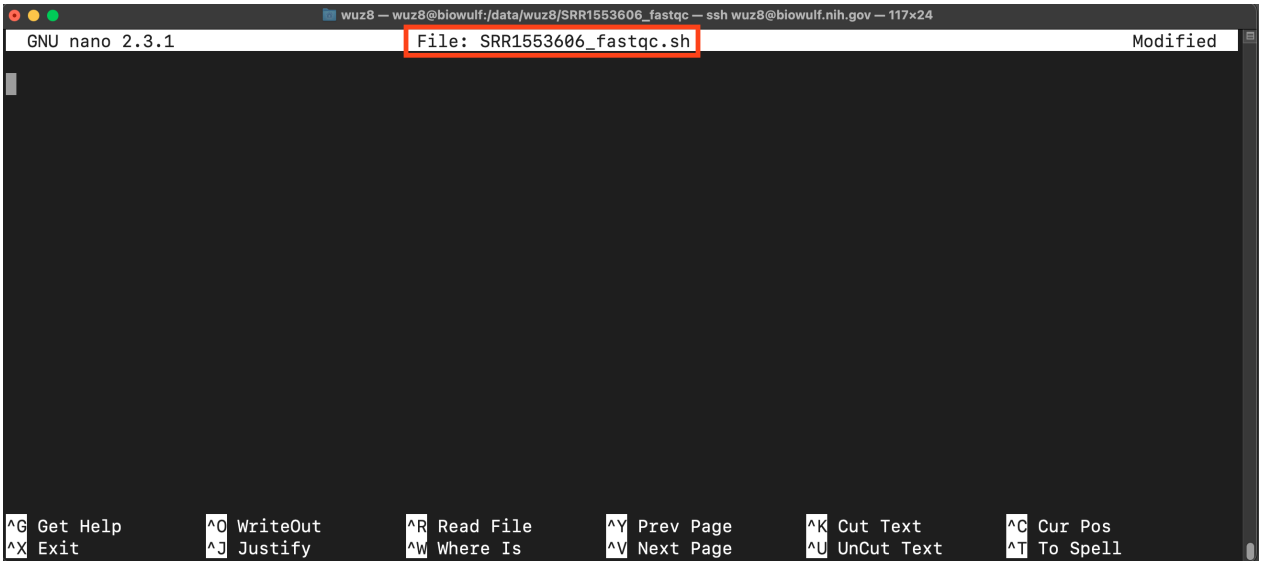


Figure 1: Nano opens a blank editor for us to start constructing SRR1553606_fastqc.sh.

Copy and paste the code below into the editor

```
#!/bin/bash
#SBATCH --job-name=SRR1553606_fastqc
#SBATCH --mail-type=ALL
#SBATCH --mail-user=wuz8@nih.gov
#SBATCH --mem=1gb
#SBATCH --partition=student
#SBATCH --time=00:05:00
#SBATCH --output=SRR1553606_fastqc_log
#SBATCH --gres=lscratch:5

export TMPDIR=/lscratch/$SLURM_JOB_ID

#LOAD REQUIRED MODULES
module load sratoolkit
module load fastqc

#USE FASTQ-DUMP FROM THE SRATOOLKIT TO DOWNLOAD FASTQ FILES FROM NCBI
#FOR PAIRED END SEQUENCING, --split-files WILL WRITE THE FORWARD AND
##WE WILL END UP WITH TWO FASTQ FILES IN THIS DOWNLOAD
#-X 10000 WILL ONLY DOWNLOAD THE FIRST 10000 READS
fastq-dump --split-files -X 10000 SRR1553606

#RUN FASTQC
fastqc -o /data/$USER/SRR1553606_fastqc SRR1553606*.fastq
```

After pasting the code into the editor, hit control-x and a message shows up at the bottom of the editor and it asks whether we like to save the file. Hit "Y" for yes in this case.

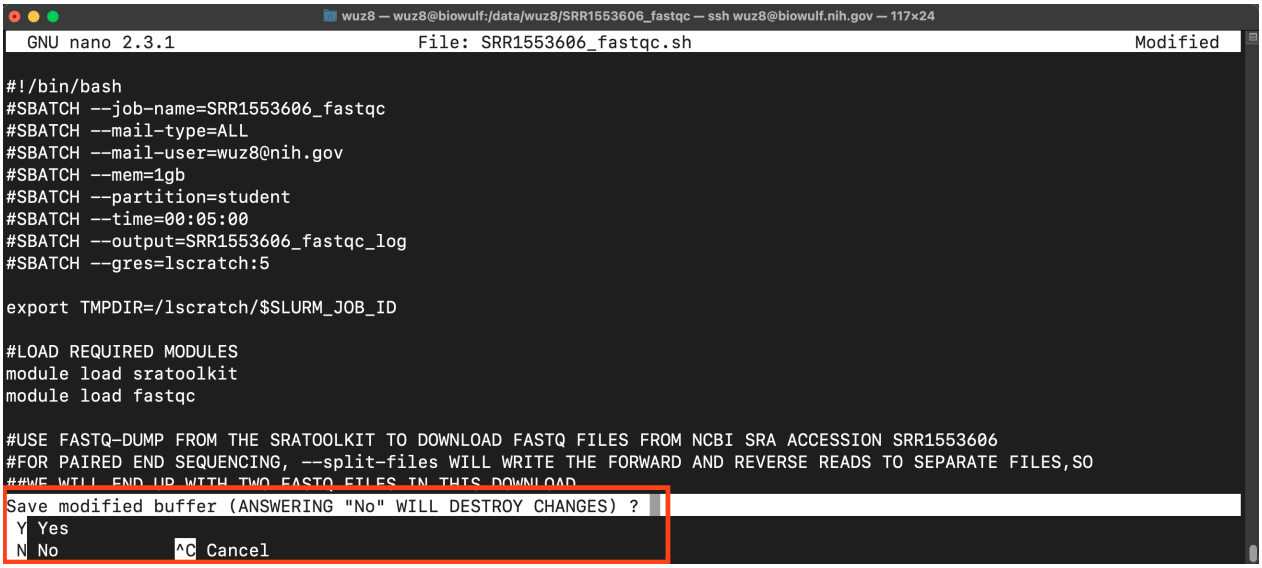


Figure 2: Hit control-x after editing to exit nano. It will ask whether you like to save.

Next, we will be asked what name to save the file as. Hit enter to save as SRR1553606_fastqc.sh.

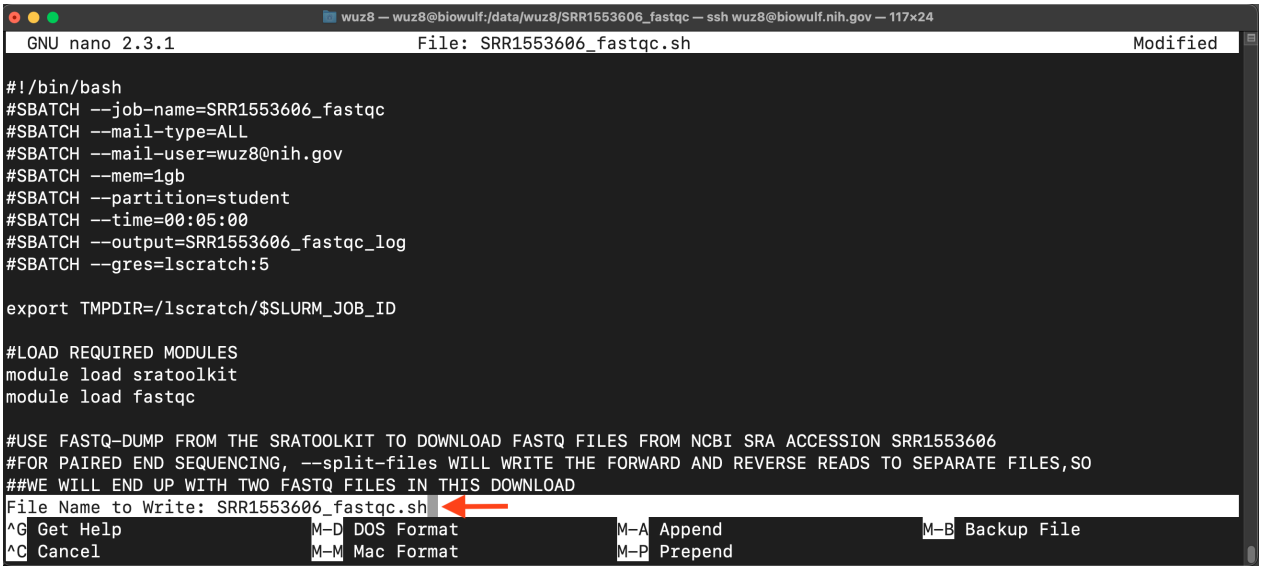


Figure 3: If we select to save, nano will ask which file name we like to save to.

Now, let's break down SRR1553606_fastqc.sh

- Lines that start with "#" are comments and are not run as a part of the script
- A shell script starts with #!/bin/bash, where
    - "#!" is known as the sha-bang
    - following "#!", is the path to the command interpreter (ie. /bin/bash)
- Next, we have lines that start with #SBATCH. Because these lines start with "#", they will not be run as a part of the script. However, these lines are important because they

instruct Biowulf on when and where to send job notification as well as what resources need to be allocated.
    - job-name: (name of the job)
    - mail-type: (type of notification emails to receive, here we want Biowulf to email to us all notifications regarding a job)
    - mail-user: (where to send notification emails, this should be your NIH email)
    - mem: (RAM or memory required, we want 1gb)
    - partition: (which partition to use; **student accounts will need to use the student partition; if you are using your own Biowulf account, please remove this line**)
    - time: (how much time should be alloted for the job, we want 5 minutes)
    - output: (name of the log file)
    - gres: (request lscratch space of 5 gb - recall the sratoolkit requires us to allocate lscratch space because it has to write temporary files)
- export TMPDIR=/lscratch/$SLURM_JOB_ID sets the TMPDIR or temporary directory path to lscratch. Temporary files will be written to TMPDIR.
- Load sratoolkit and fastqc using `module load`
- Download the FASTQ data using `fastq-dump` from sratoolkit
- Run `fastqc` - from the previous class, we know that the files downloaded are SRR1553606_1.fastq and SRR1553606_2.fastq so we can use the "*" to denote wildcard or anything between the accession (SRR1553606) and the file extension ".fastq" to provide input for `fastqc`. We specify the output directory using the `-o` option and provide the path /data/$USER/SRR1553606_fastqc where $USER is the environmental variable that points to your Biowulf username.

To submit the shell script as a job, we will do

```
sbatch SRR1553606_fastqc.sh
```

After submission, we can use `squeue -u username` to check on job status.

```
squeue -u username
```

For instance, below is the status of a job that I submitted, the job status (ST column) is PD (pending). Note that because I am not signed on with a student account, I can use the "norm" partition for this job.

```
[wuz8@biowulf wuz8]$ squeue -u wuz8
   JOBID  PARTITION     NAME  USER  ST  TIME  NODES  NODELIST(REASON)
55421634       norm unix_on_  wuz8  PD  0:00      1  (None)
```

To cancel a job, use `scancel` followed by the job id

```
scancel job_id
```

Figure 4 and Figure 5 shows the emails that Biowulf will send us upon the start of a submitted job and then when it completes. Because we set `--mailtype=ALL`, Biowulf will email to inform us of all statuses regarding a batch job. For instance, if we cancel a job, Biowulf will send us an email (Figure 6).

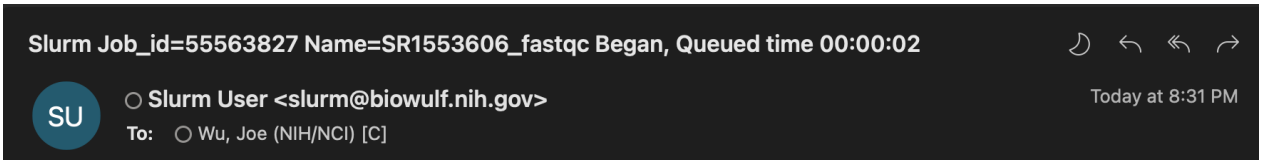


Figure 4: Email sent by Biowulf to our NIH emails when a job starts.

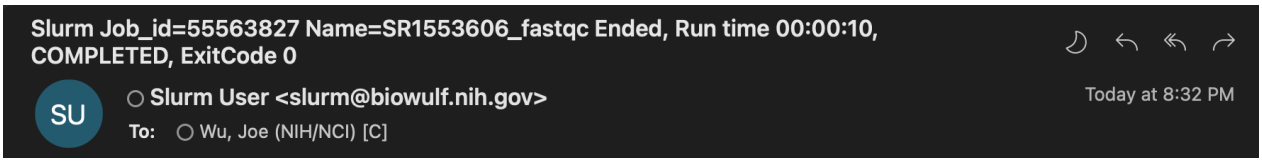


Figure 5: Email sent by Biowulf to our NIH emails when a job completes.

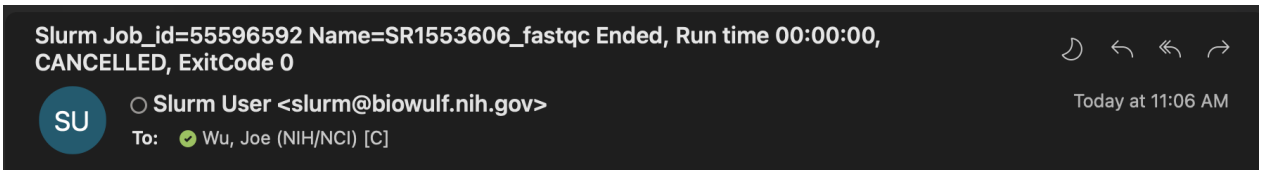


Figure 6: Email sent by Biowulf to our NIH emails when a job is cancelled.

Note that we can always go into the Biowulf User Dashboard to get information about our jobs. For those on student accounts use the student dashboard *(https://hpcnihapps.cit.nih.gov/auth/student_dashboard)* but sign in with your own NIH username not the student1, student2, etc. usernames. At the User Dashboard, click on "Job Info" and a table listing the jobs will appear (Figure 7). Clicking on a job (ie. 55599045 shown in Figure 7) we will see useful information such as a plot of memory and CPU usage (Figure 8), which is important because depending on the usage, we can fine tune the amount of resources that we request.

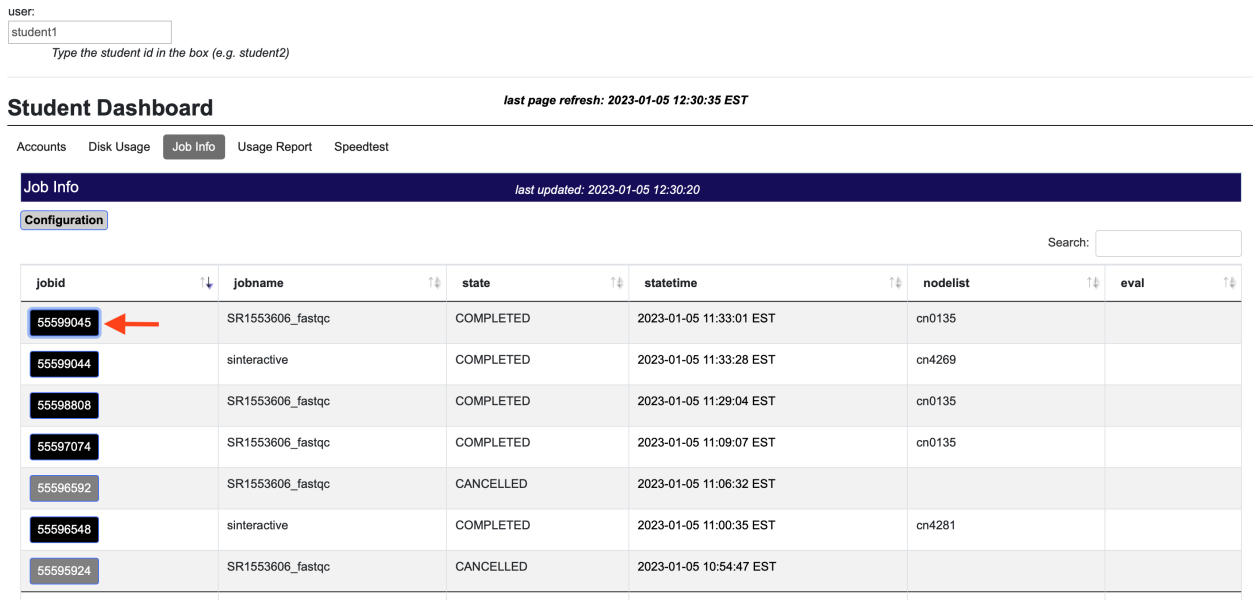


Figure 7: A listing of student1's jobs in the Biowulf User Dashboard.

Figure 8: Memory and CPU usage for one of student1's jobs in the Biowulf User Dashboard.

Transferring data between Biowulf and local machine

Now that we have generated FASTQC reports for SRR1553606, we need to transfer them to our local machine so we can view in a web browser. There are many options for transferring between our computer and Biowulf. See Transferring data to and from the NIH HPC systems *(https://hpc.nih.gov/docs/transfer.html)* for details.

Options for transferring between local and Biowulf are listed below.

- Mount HPC drives to local machine (slow) *(https://hpc.nih.gov/docs/hpcdrive.html)*
- Command line tools
    - secure copy (scp)
    - sftp
- Use graphical sftp or scp client *(https://hpc.nih.gov/docs/transfer.html#GUI)*
    - WinSCP
    - Fugu
    - Mobaxterm
- Globus is recommended for transferring large datasets between local and HPC *(https://hpc.nih.gov/docs/globus/)*

Biowulf also provides options for transferring to and from NIH Box and NIH OneDrive. See Transferring data between NIH Box or NIH OneDrive and HPC systems *(https://hpc.nih.gov/docs/box_onedrive.html)* or Globus on NIH HPC (Biowulf) *(https://hpc.nih.gov/docs/globus/od_box.php)* for instructions.

Some of the data transfer options involve installing new software. Submitting a ticket with service.cancer.gov *(https://service.cancer.gov/ncisp)* will help you get that done.

Biowulf file transfer tutorials on YouTube:

- mounting *(https://youtu.be/H8ZksTK3EtE?t=88)*
- secure copy or scp *(https://youtu.be/H8ZksTK3EtE?t=258)*
- Globus *(https://youtu.be/mg9-a1OuDqo)*

Transferring data using Globus

Globus is the recommended tool for transferring large datasets between local computer and HPC. Refer to https://hpc.nih.gov/docs/globus/ *(https://hpc.nih.gov/docs/globus/)* for instructions on how to setup your Globus account. Use Chrome when working with Globus.

Once you have setup your Globus account, goto https://www.globus.org/ *(https://www.globus.org/)* to log in (see Figure 9.)

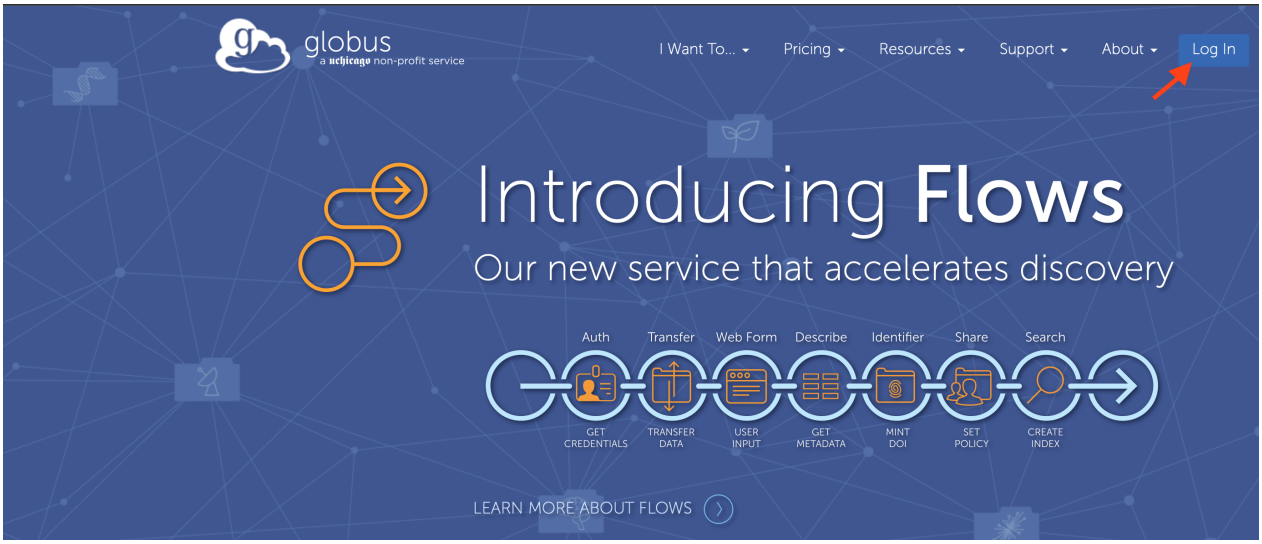


Figure 9: Log in to your Globus account at https://www.globus.org/ *(https://www.globus.org/)*.

Clicking on the Log In button (Figure 9) will take a you to a page where you can search for your institution or organization. It auto suggests so when your institution or organization (ie. National Institutes of Health) shows up, select it (Figure 10).

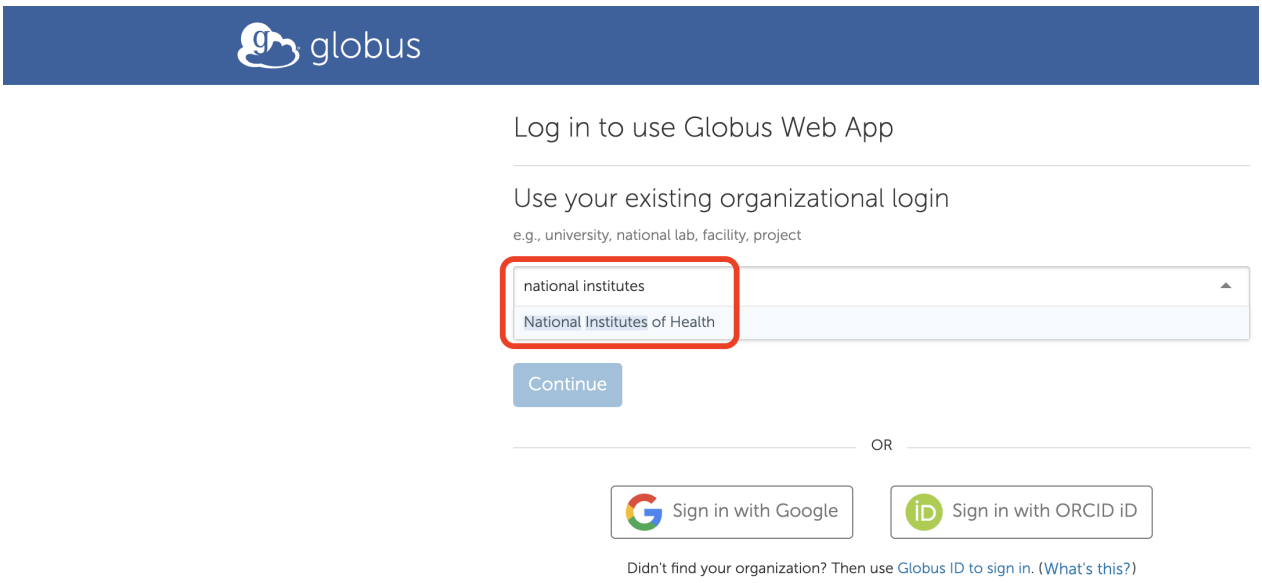


Figure 10: Select your institution or organization.

Once you have selected your institution or organization, click on Continue (Figure 11) and you will be directed to complete the Globus log in process by entering your NIH credentials (Figure 12).

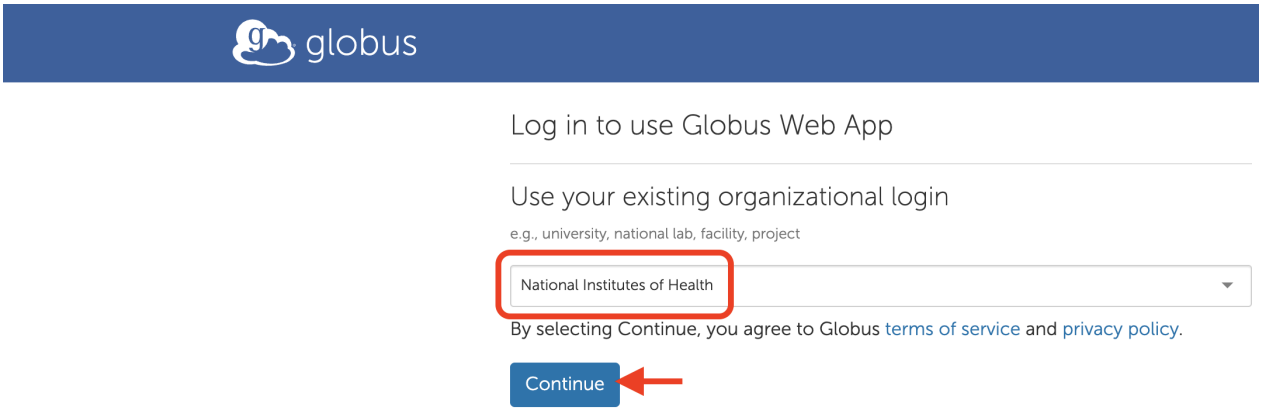


Figure 11: Click continue to proceed to entering your NIH credentials.

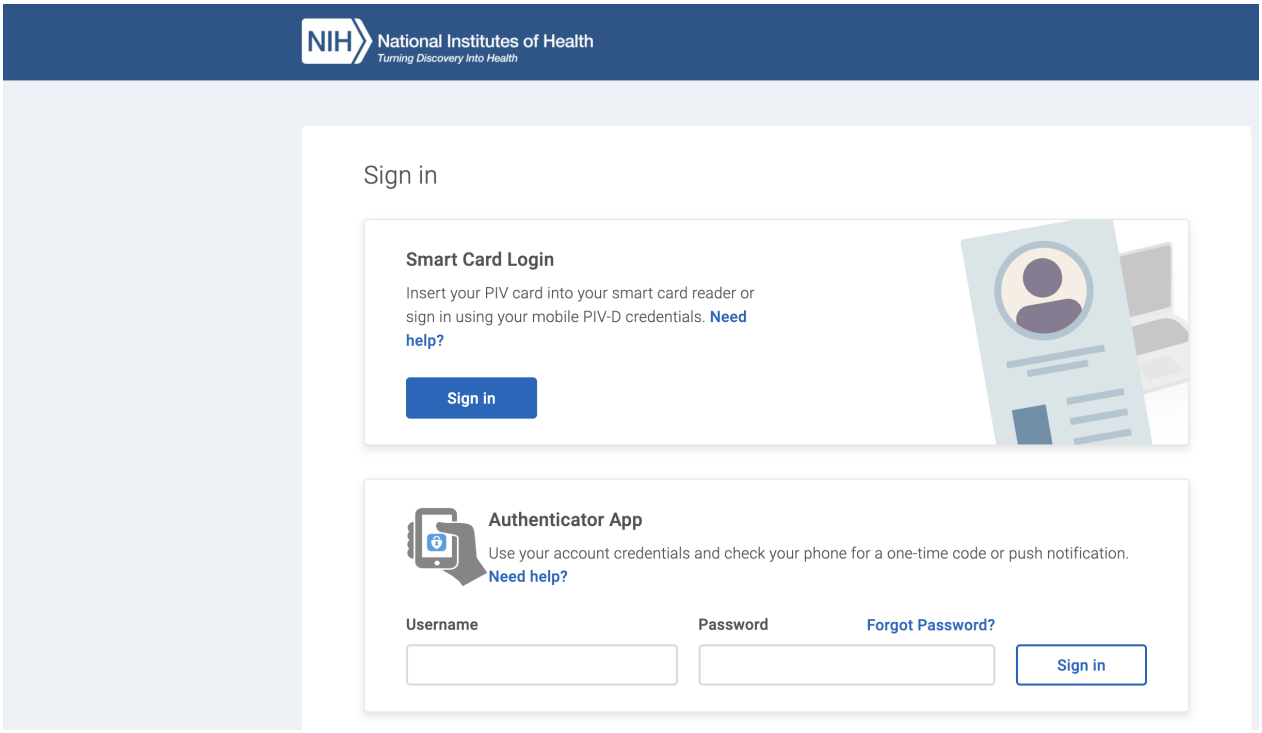


Figure 12: Complete the Globus log in process by entering your NIH credentials.

Hit agree at the screen that appears after you have provided your NIH credentials (Figure 13). This takes to you the Globus File Manager (Figure 14).

The service or system you're connecting to requires that NIH share a limited set of basic information about you before you can access it. You must explicitly give consent to release this information before you can access these services.

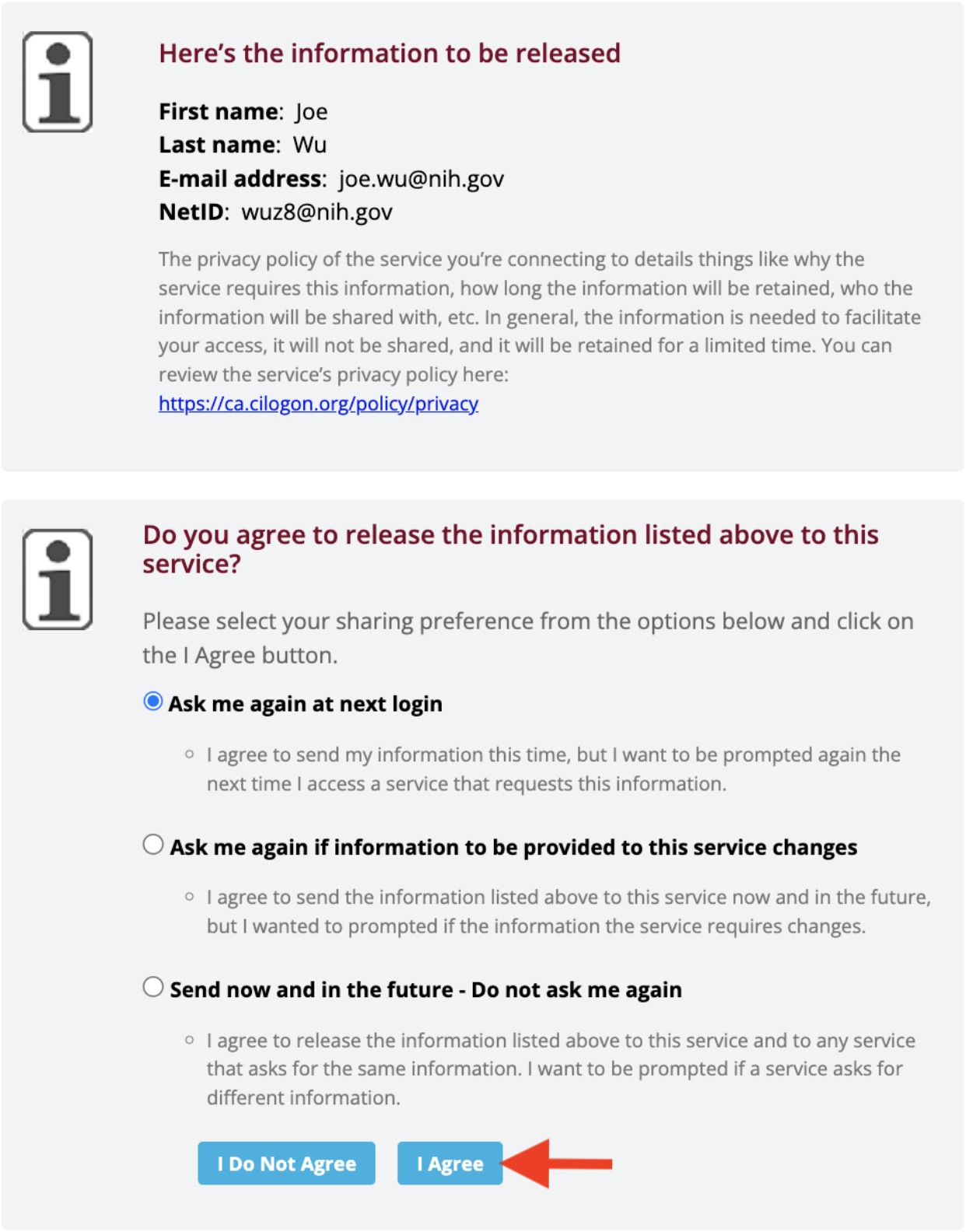


Figure 13: Hit agree to proceed.

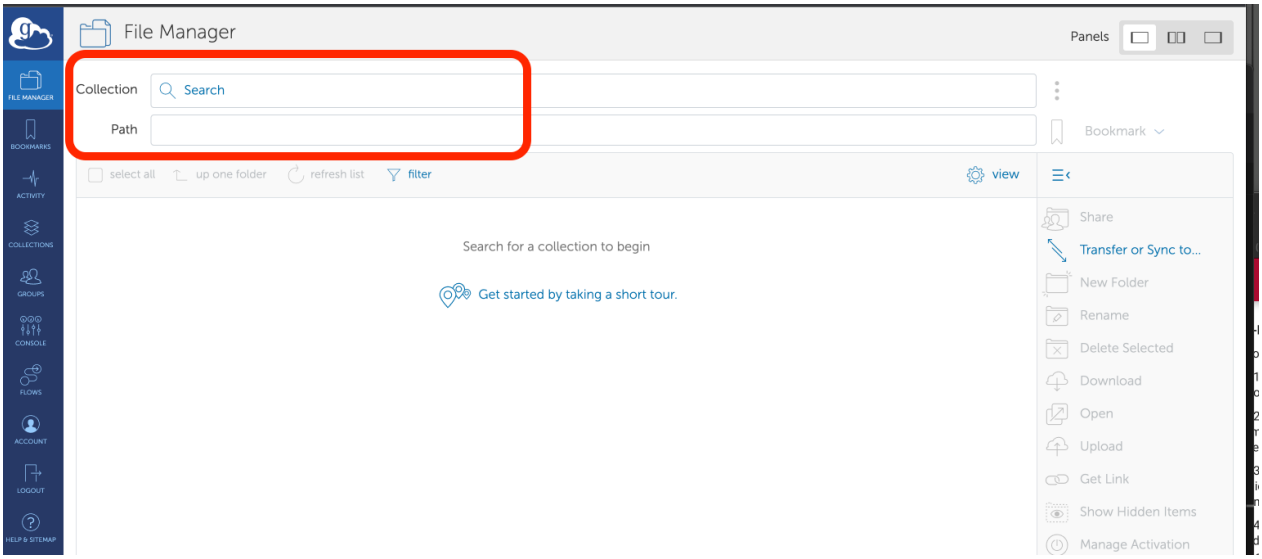


Figure 14: Globus file manager.

At the Gloubs file manager (Figure 15), search for your local file collection (see https://hpc.nih.gov/docs/globus/ for creating a local collection *(https://hpc.nih.gov/docs/globus/)*). Next, select Transfer or Sync to and the File Manager splits into two panes. On the pane to right (Figure 16), search for NIH HPC Data Transfer (Biowulf) collection and this will take you to your Biowulf home directory, but you can change into your data directory below the search box. Select the files that you want to transfer and hit Start to transfer to local. Upon a successful transfer, the files should show up on the pane to the left, which is your local file collection.

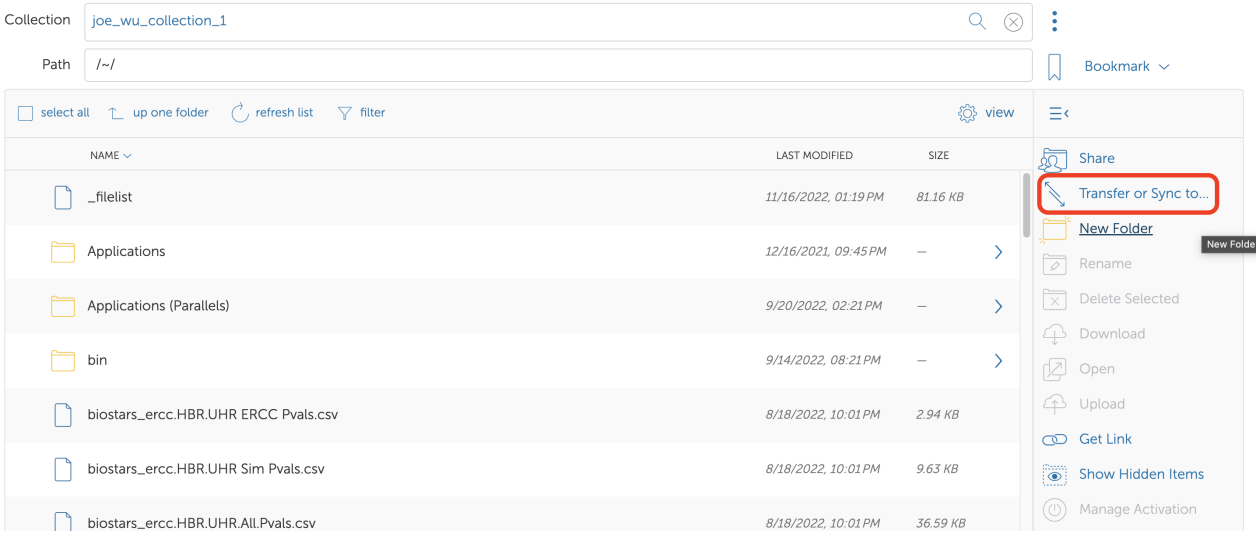


Figure 15: Search for the contents of your local computer at the Globus File Manager.

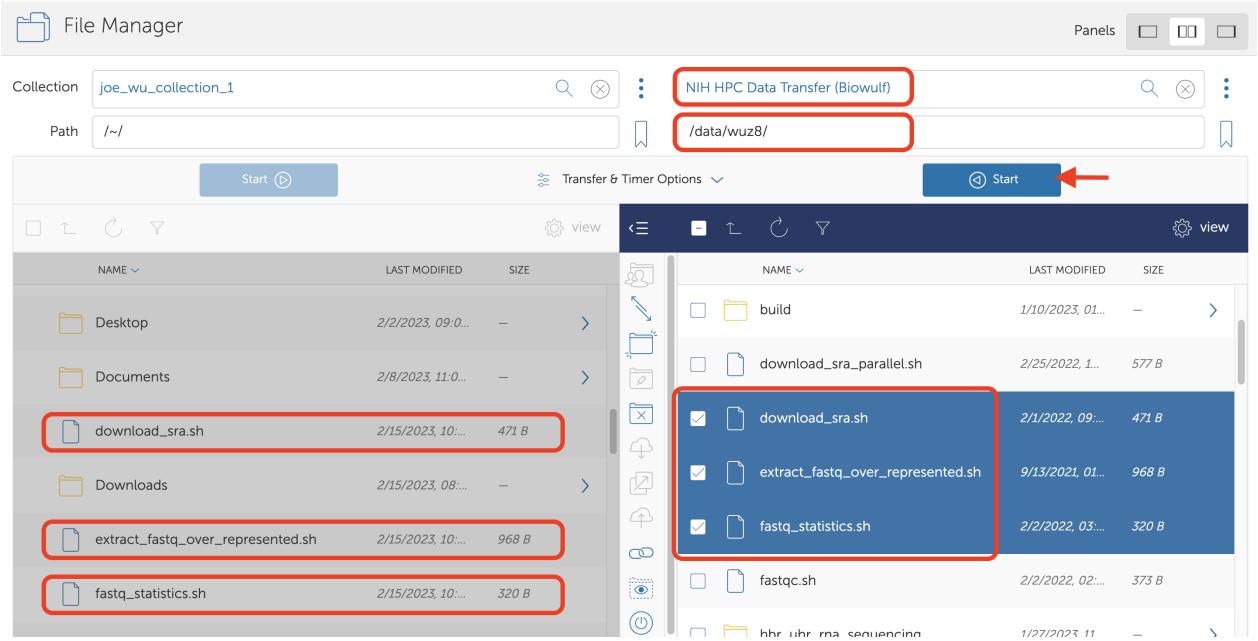


Figure 16: Find the NIH HPC Data Transfer (Biowulf), change into you Biowulf data directory, select files and hit Start to transfer to local.

Transferring data using scp

For the remainder of this lesson, we will learn how to use secure copy (`scp`) to transfer the FASTQC html reports from SRR1553606 to our local computers to view. While the syntax for `scp` is the same whether we use Windows or Mac, there are subtle differences like the way directory paths are expressed in Windows versus Mac. Below, we break down the instructions for using `scp` for both operating systems.

Directory path structure: Mac versus Windows

Before using the `scp` command to transfer data from Biowulf to local computer, we should understand the directory path structure in Macs and Windows.

Mac directory path structure

Mac directory path structure follows that of Unix. For instance, the absolute path of the local Downloads folder follows the structure below. This path is absolute because it is starting at the root, which is denoted by "/" at the beginning of the path. Replace username with your NIH username since this is what you use to sign onto your local machine.

```
/Users/username/Downloads
```

Windows directory path structure

The path to the local Downloads folder in Windows is shown below. It starts with the name of the disk that the folder is in. Windows uses letters in the alphabet to name disks. Replace username

with your NIH username since this is what you use to sign onto your local machine. Note that Windows directory paths start with the name of the disk drive followed by a ":". Also, the parts of a directory path are separated by "\" and not "/" as seen Macs and Unix.

```
disk_name:\Users\username\Downloads
```

scp for Windows users

We are going to place the FASTQC html reports for SRR1553606 in our Windows Downloads folder. The next step is to change into the Windows Downloads folder. Upon opening the command prompt, you should be in the `disk_drive:\Users\username` folder, where

- `disk_drive` is a letter in the alphabet (ie. `O` in Figure 9). This disk drive is that in which Windows is installed and is also known as the systems drive.
- `username` is your NIH username since you are now working locally and you use your NIH username to log into your local computer (in Figure 17, you see `wuz8` because that is my NIH username)

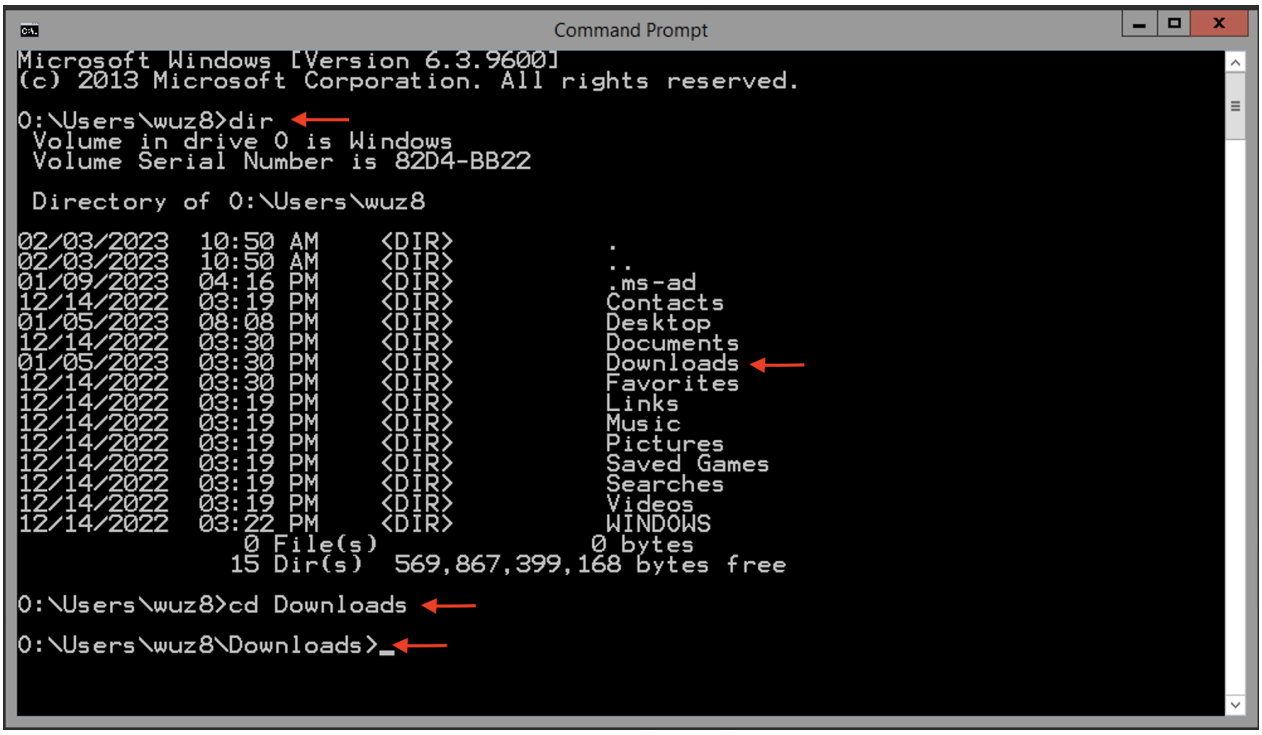


Figure 17: Upon opening the Windows Command Prompt, you will land in your `Users\username` folder (where username is your NIH username).

At the Windows Command Prompt, type `dir` to list the contents of the `Users\username` folder and you will see a subfolder called `Downloads` (Figure 9). In Unix, we used `ls` to list directory content. For those using Windows, `dir` is used to list directory content in the Command Prompt and it is just one of the many commands used in the Microsoft Disk Operating System or MS-DOS *(https://www.computerhope.com/overview.htm)*.

```
dir
```

Next, type `cd Downloads` to change into your local Windows Downloads directory. Here, we see a common command between Unix and MS-DOS (ie. `cd`).

```
cd Downloads
```

In the Windows Downloads folder, use the the following commands to transfer the FASTQC reports from Biowulf. Again, "." at the end of the commands denotes "here, in the current directory". Enter your the password you used to sign into Biowulf if prompted.

```
scp username@helix.nih.gov:/data/username/SRR1553606_fastqc/SRR15536(
```

```
scp username@helix.nih.gov:/data/username/SRR1553606_fastqc/SRR15536(
```

After download completes, the two FASTQC html reports for SRR1553606 will appear in the Windows Downloads folder and you can view these using a web browser (Figure 18).

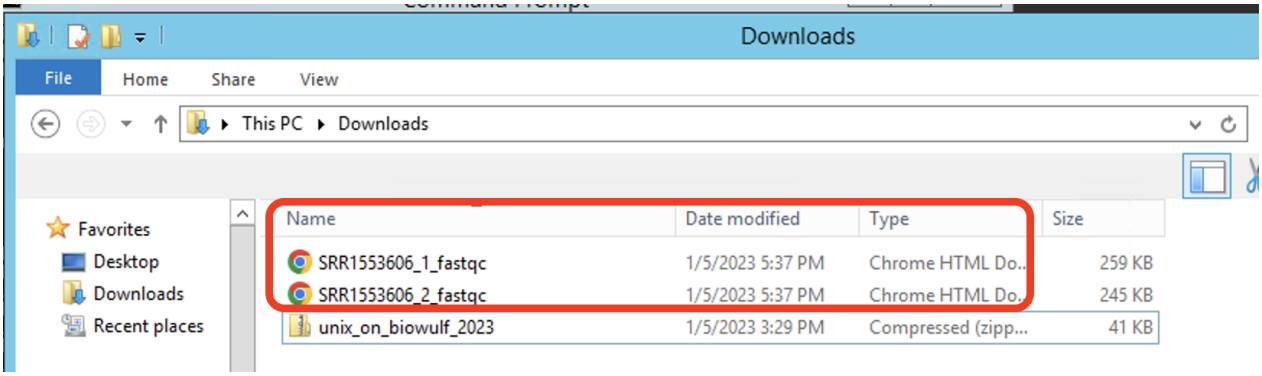


Figure 18: FASTQC html reports for SRR1553606 appear in the Windows Downloads folder after successful scp.

scp for Mac users

Mac users will need to open a Terminal window. Then change into the local Downloads folder which should be /Users/username/Downloads (where username should be your NIH username).

```
cd /Users/username/Downloads
```

If you cannot remember the username for your government furnished Mac, then do the following, where "~" denotes home directory because technically the Downloads folder is

located in the home directory which is /Users/username (where again, username is your NIH username). Another option is to type `echo $USER` to find the username for your government furnished Mac.

```
cd ~/Downloads
```

Once in our Downloads folder, we do the following to copy over the FASTQC html files. Where username, is your NIH username or the student accounts (ie. student1, student2, student3, etc.). Enter your password when prompted.

```
scp username@helix.nih.gov:/data/username/SRR1553606_fastqc/SRR15536(
```

```
scp username@helix.nih.gov:/data/username/SRR1553606_fastqc/SRR15536(
```

If `scp` was successful, we should see the two FASTQC html reports for SRR1553606 in the Mac Downloads folder (Figure 19).

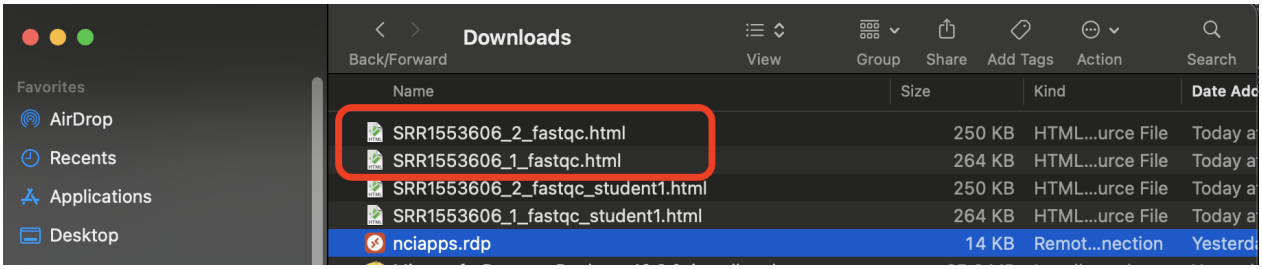


Figure 19: FASTQC html reports for SRR1553606 appear in the Mac Downloads folder after successful `scp`.

To place SRR1553606_1_fastqc.html and SRR1553606_2_fastqc.html from local to Biowulf using `scp, do the following (we will put it in our data folder). Note, we should stay in our local Downloads directory for this. Enter your password if prompted.

```
scp SRR1553606_1_fastqc.html username@helix.nih.gov:/data/username
```

```
scp SRR1553606_2_fastqc.html username@helix.nih.gov:/data/username
```

Lesson 6 supplement (Swarm)

Swarm in Biowulf

In Biowulf, we can create a swarm script to help with parallelization of tasks. For instance, we can use a swarm script to download multiple sequencing data files from the NCBI SRA study Zaire ebolavirus sample sequencing from the 2014 outbreak in Sierra Leone, West Africa *(https://trace.ncbi.nlm.nih.gov/Traces/?view=study&acc=SRP045416)* in parallel, rather than one file after another. Here, we will download the first 10000 reads the following sequencing data files in this study

- SRR1553606
- SRR1553416
- SRR1553417
- SRR1553418
- SRR1553419

While we can run five invidual `fastq-dump` commands (one after another) in an interactive session to download sequencing data files, it would be easier to do this using the swarm script below where each of the `fastq-dump` commands are run as an individual sub-job and thus, allowing us to download in parallel.

We can copy and paste the following swarm script into the nano editor, save it, and then submit the script as a job to Biowulf to accomplish our download. Save this script as SRP045416.swarm.

```
#SWARM --job-name SRP045416
#SWARM --sbatch "--mail-type=ALL --mail-user=wuz8@nih.gov"
#SWARM --gres=lscratch:15
#SWARM --module sratoolkit

fastq-dump --split-files -X 10000 SRR1553606
fastq-dump --split-files -X 10000 SRR1553416
fastq-dump --split-files -X 10000 SRR1553417
fastq-dump --split-files -X 10000 SRR1553418
fastq-dump --split-files -X 10000 SRR1553419
```

The first four lines in the script start with #SWARM are not run as part of the script and are directives for requesting resources on Biowulf. The four swarm directives are interpreted as below:

- --job-name
  - assigns job name (ie. SRP045416)
- --sbatch "--mail-type=ALL --mail-user=wuz8@nih.gov"
  - asks Biowulf to email all job notifications
- --gres=lscratch:15
  - asks for 15 GB of local scratch space for temporary files
- --module sratoolkit
  - loads the sratoolkit so we can run fastq-dump

We can request other compute resources for swarm jobs (see https://hpc.nih.gov/apps/swarm.html *(https://hpc.nih.gov/apps/swarm.html)*).

To submit the swarm script, we do

```
swarm -f SRP045416.swarm
```

Note that upon submitting of the swarm script, Biowulf assigns an overall job ID (58101981).

```
[wuz8@cn4269 wuz8]$ swarm -f SRP045416.swarm
58101981
```

We can use `squeue` to check the status of this job. By doing so, we see that each of the `fastq-dump` commands are run as a sub-job labeled as 58101981_0, 58101981_1, 58101981_2, 58101981_3, and 58101981_4. The number of sub-jobs reflex the number of commands in the swarm script. Another way of interpreting swarm is that it offers efficiency by enabling the submission of multiple jobs that run in parallel through the submission of one script to the Biowulf batch system.

```
[wuz8@cn4269 wuz8]$ squeue -u wuz8
```

```
JOBID       PARTITION     NAME  USER ST  TIME  NODES NODELIST(REASON)
58101981_2      norm    swarm   wuz8  R  0:06      1 cn4308
58101981_3      norm    swarm   wuz8  R  0:06      1 cn4270
58101981_4      norm    swarm   wuz8  R  0:06      1 cn4316
58101981_0      norm    swarm   wuz8  R  0:07      1 cn4303
58101981_1      norm    swarm   wuz8  R  0:07      1 cn4305
```

Lesson 7: Downloading data, viewing file content, and data wrangling in Unix

Quick review:

In this course series, we have learned how to connect to and navigate around Biowulf. In addition, we have learned how to use applications installed on Biowulf to download sequencing data from the SRA (using `fastq-dump`) and subsequently, assess the quality of the downloaded sequencing data (using `fastqc`). Further, we learned to transfer files from Biowulf to our local computer (using `scp`). Finally, we learned to request an interactive session (using `sinteractive`) or submit a batch job (using `sbatch`) to perform compute intensive tasks.

Lesson objectives:

After this lesson, we should

- Be able to download data from the web
- Know how to view file content
- Know how to perform pattern search

Unix commands that we will learn in this lesson

- `wget` (to download data from the web)
- `curl` (to download data from the web)
- `tar` (to unpack tape archives)
- `unzip` (to unpack zipped files)
- `cat` (to display file content)
- `head` (to display beginning of file content; defaults to first 10 lines)
- `tail` (to display end of file content; defaults to last 10 lines)
- `zcat` (to display compressed file content)
- `less` and `more` (to scroll through files)
- `grep` (to search for patterns)

Downloading data from URL

In Unix, we can use `wget` or `curl` to download data from URL.

As an example, let's download the sequncing data (FASTQ) files for the Human Brain Reference (HBR) and Universal Human Reference (UHR) from the Griffith lab RNA sequencing

tutorial *(https://rnabio.org/module-01-inputs/0001/05/01/RNAseq_Data/)*. You can read more about the HBR-UHR dataset on that page.

Before getting started, let's use `pwd` to make sure that we are in our data directory

```
pwd
```

```
/data/username
```

If not change into it

```
cd /data/username
```

The next step is to create a folder to store the HBR-UHR sequencing data. Let's call this folder hbr_uhr_rna_sequencing. To create this folder, we will use the `mkdir` command.

```
mkdir hbr_uhr_rna_sequencing
```

After creating hbr_uhr_rna_sequencing, change into it.

```
cd hbr_uhr_rna_sequencing
```

To download the FASTQ files for the HBR-UHR dataset, type `wget` at the command line followed by the URL of the dataset, which is http://genomedata.org/rnaseq-tutorial/HBR_UHR_ERCC_ds_5pc.tar.

```
wget http://genomedata.org/rnaseq-tutorial/HBR_UHR_ERCC_ds_5pc.tar
```

During the download process, we will see a download progress bar in the terminal (Figure 1).

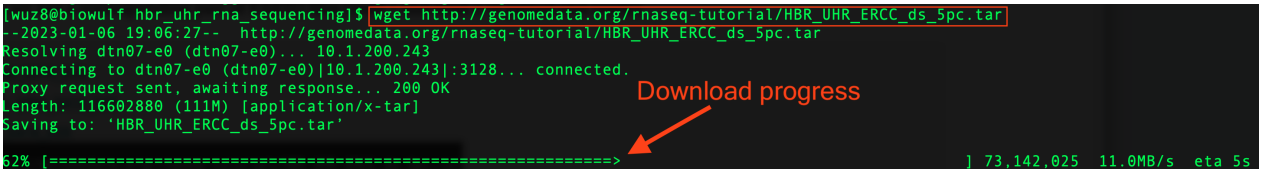


Figure 1: Unix `wget` download progress.

Now, if we list the contents of the hbr_uhr_rna_sequencing folder

```
```

```
ls -l
```

We will see a tape archive or ".tar" file that we need to unpack to get to the HBR-UHR sequences. More on ".tar" files in a bit.

```
-rw-r-----. 1 wuz8 wuz8 116602880 Oct 23  2018 HBR_UHR_ERCC_ds_5pc.ta
```

First, let's remove HBR_UHR_ERCC_ds_5pc.tar and download it again using `curl`.

To delete a file, recall that we use `rm` followed by the name of the file that we want to delete.

```
rm HBR_UHR_ERCC_ds_5pc.tar
```

With the `curl` command, we need to specify an output file name. We see two options for specifying the name of the output file with `curl` if we look into help documents.

```
curl --help
```

```
-o, --output FILE   Write output to <file> instead of stdout
-O, --remote-name   Write output to a file named as the remote file
```

```
curl -O  http://genomedata.org/rnaseq-tutorial/HBR_UHR_ERCC_ds_5pc.ta
```

Listing the contents of the hbr_uhr_rna_sequencing directory, we see that the file HBR_UHR_ERCC_ds_5pc.tar appears when we use `curl` with the `-O` option, which writes a file that has the same name as that from the URL (ie. HBR_UHR_ERCC_ds_5pc.tar).

```
ls -l
```

```
-rw-r-----. 1 wuz8 wuz8 116602880 Jan  7 12:10 HBR_UHR_ERCC_ds_5pc.ta
```

Let's try downloading using `curl` but specifying a file name of our choice using the `-o` (lower case o) option. We will name the file HBR_UHR_READS.tar.

```
curl http://genomedata.org/rnaseq-tutorial/HBR_UHR_ERCC_ds_5pc.tar -o
```

Listing the contents of the hbr_uhr_rna_sequencing folder, we see that in addition to the file HBR_UHR_ERCC_ds_5pc.tar, which we downloaded using `curl -O`, we have the file HBR_UHR_READS.tar, which we downloaded using `curl` with the `-o` option (where we specified an output file name of our choice, rather than the one provided by the URL).

```
ls -l
```

```
-rw-r-----. 1 wuz8 wuz8 116602880 Jan  7 12:10 HBR_UHR_ERCC_ds_5pc.ta
-rw-r-----. 1 wuz8 wuz8 116602880 Jan  7 12:16 HBR_UHR_READS.tar
```

Let's go ahead and remove HBR_UHR_ERCC_ds_5pc.tar because it is the same as HBR_UHR_READS.tar.

```
rm HBR_UHR_ERCC_ds_5pc.tar
```

Tar files and how to unpack them

Earlier, we mentioned that the ".tar" extension stands for Tape Archive. Tape Archive allows us to package many files and folders into a single file for easy transfer and sharing. We use the `tar` command to unpack these files. Options for the `tar` command can be found by using the command below.

```
tar --help
```

The options that we will use for unpacking are below. Note that we can use a single "-" to string together options in Unix commands.

```
-x, --extract, --get       extract files from an archive
-v, --verbose              verbosely list files processed
-f, --file=ARCHIVE         use archive file or device ARCHIVE
```

```
tar -xvf HBR_UHR_READS.tar
```

Because we included the `-v` option in the `tar` command above, we see the files that are unpacked as the command runs.

```
HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.fastq.gz
HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.fastq.gz
```

```
HBR_Rep2_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.fastq.gz
HBR_Rep2_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.fastq.gz
HBR_Rep3_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.fastq.gz
HBR_Rep3_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.fastq.gz
UHR_Rep1_ERCC-Mix1_Build37-ErccTranscripts-chr22.read1.fastq.gz
UHR_Rep1_ERCC-Mix1_Build37-ErccTranscripts-chr22.read2.fastq.gz
UHR_Rep2_ERCC-Mix1_Build37-ErccTranscripts-chr22.read1.fastq.gz
UHR_Rep2_ERCC-Mix1_Build37-ErccTranscripts-chr22.read2.fastq.gz
UHR_Rvep3_ERCC-Mix1_Build37-ErccTranscripts-chr22.read1.fastq.gz
UHR_Rep3_ERCC-Mix1_Build37-ErccTranscripts-chr22.read2.fastq.gz
```

Note that sequencing data from the HBR-UHR dataset come as FASTQ files but are g-zipped (.gz). We will learn how to view data in the Unix terminal and point out how to view FASTQ files that are g-zipped without having to unzip them. Note that some bioinformatics software such as FASTQC can take g-zipped FASTQ files (ie. fastq.gz) as input. Check with the help documents for each software to find out.

Viewing file content in Unix

This portion of the lesson will focus on viewing contents of files in Unix. We will focus on the three types of files below.

- Plain text
- Tabular data (ie. data files that have many columns and many rows, like a matrix); these data tables can have columns that are
    - comma separated (csv)
    - tab separated (these will come in the form of txt files)
- FASTQ files, which contain high throughput sequencing data

Viewing plain text files in Unix

For this portion of the lesson, let's change back into our data directory. Again, username is the username you used to sign into Biowulf, this could be your NIH username if you have Biowulf a account or one of the student accounts that were setup for us.

```
cd /data/username
```

Next, we are going to the course documents and use `wget` to grab the file unix_on_biowulf_2023.zip (this is under the section labeled Course data in the course documents)

```
wget https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/data/unix_
```

We will then use `unzip` to unpack the contents of unix_on_biowulf_2023.zip.

```
unzip unix_on_biowulf_2023.zip
```

Note that we get a status of what is being unpacked as the unzipping occurs.

```
Archive:  unix_on_biowulf_2023.zip
   creating: unix_on_biowulf_2023/
  inflating: unix_on_biowulf_2023/text_1.txt
  inflating: unix_on_biowulf_2023/counts.csv
  inflating: unix_on_biowulf_2023/results.csv
```

Listing the contents of our data folder, we will see a new folder called unix_on_biowulf_2023 (let's change into this).

```
ls -l
```

```
drwxr-xr-x.  2 wuz8 wuz8        4096 Jan  2 14:44 unix_on_biowulf_20
```

```
cd unix_on_biowulf_2023
```

We will list the contents of the unix_on_biowulf_2023 directory to see what we have to work with.

```
ls
```

We have a gene expression counts table from the HBR-UHR dataset (counts.csv), the differential expression analysis results from the HBR-UHR dataset (results.csv), and a random text file (text_1.txt).

```
counts.csv  results.csv  text_1.txt
```

Let's see what is in text_1.txt by using `cat`

```
cat text_1.txt
```

```
oranges
blue
```

```
bananas
cats
dogs
apple
florida
gators
gainesville
alachua
county
btep
```

The `head` command can be used to view the top several lines of a file (default is 10 lines). We can use the `-n` option to specify how many lines we want (for instance `-n 5` will show the first five lines).

```
head -n 5 text_1.txt
```

```
oranges
blue
bananas
cats
dogs
```

Opposite of `head`, `tail` will show the bottom 10 lines of a file by default. Again, we can use `-n` to specify the number lines other the default.

```
tail -n 5 text_1.txt
```

```
gators
gainesville
alachua
county
btep
```

We can use the `zcat` command to view contents of compressed files without uncompressing them. For instance, the FASTQ files that we downloaded for the HBR-UHR dataset. We will stay in the unix_on_biowulf_2023 directory for this but will append the "../hbr_uhr_rna_sequencing" to reference the directory in which the FASTQ files are in. ".." tells Unix to go up one directory and then look in the folder hbr_uhr_rna_sequencing. We will use "|" or the pipe to send the output of `zcat` to the `head -n 4` command, to get the first four lines of the FASTQ file HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.fastq.gz.

```
zcat ../hbr_uhr_rna_sequencing/HBR_Rep1_ERCC-Mix2_Build37-ErccTransc
```

The FASTQ file contains many sequencing reads and these come in 4 lines each, which are

- Metadata header that starts with "@"
- Actual sequence
- "+"
- Error likelihood of each of bases along the read

```
@HWI-ST718_146963544:7:2201:16660:89809/1
CAAAGAGAGAAAGAAAAGTCAATGATTTTATAGCCAGGCAAAATGACTTTCAAGTAAAAAATATAAAG
+
CCCFFFFFHHHHHJJJJJHIHIJJIJJJJJJJJJJJJIJJJJJJJJJJJJJIJJIIJJJJJJJJJJJ
```

For larger files that have a lot of rows, we can use `less` to scroll. For instance, if we `cat counts.csv`, it will print the entire file to the terminal. So to view parts of it at a time while being able to scroll we can use the `less` command, which is known as a terminal pager. Note that we can use the down arrow to scroll down a file and the up arrow to scroll up when using `less`.

```
less counts.csv
```

The counts.csv file is a gene expression counts table and has seven columns, where the first column contains gene IDs. Note that the columns are separated by commas as suggested by the ".csv" extension.

Hit q to get out of `less` and return to the prompt.

```
Geneid,HBR_1.bam,HBR_2.bam,HBR_3.bam,UHR_1.bam,UHR_2.bam,UHR_3.bam
U2,0,0,0,0,0,0
CU459211.1,0,0,0,0,0,0
CU104787.1,0,0,0,0,0,0
BAGE5,0,0,0,0,0,0
ACTR3BP6,0,0,0,0,0,0
5_8S_rRNA,0,0,0,0,0,0
AC137488.1,0,0,0,0,0,0
AC137488.2,0,0,0,0,0,0
CU013544.1,0,0,0,0,0,0
CT867976.1,0,0,0,0,0,0
CT867977.1,0,0,0,0,0,0
CT978678.1,0,0,0,0,0,0
CU459202.1,0,0,0,0,0,0
AC116618.1,0,0,0,0,0,0
CU463998.1,0,0,0,0,0,0
```

```
CU463998.3,0,0,0,0,0,0
CU463998.2,0,0,0,0,0,0
U6,0,0,0,0,0,0
LA16c-60D12.1,0,0,0,3,2,0
LA16c-13E4.3,0,0,0,0,0,1
LA16c-60D12.2,0,0,0,0,4,1
ZNF72P,0,0,0,0,1,0
```

We can also use the `more` command to scroll through counts.csv. At the bottom of the page, `more` prints out the percentage of the file content shown in the screen. We can hit enter to scroll line by line or the space bar to scroll page by page. Hit q to exit `more` and return to the prompt. Note that on Biowulf, we cannot scroll up with `more`.

```
more counts.csv
```

```
Geneid,HBR_1.bam,HBR_2.bam,HBR_3.bam,UHR_1.bam,UHR_2.bam,UHR_3.bam
U2,0,0,0,0,0,0
CU459211.1,0,0,0,0,0,0
CU104787.1,0,0,0,0,0,0
BAGE5,0,0,0,0,0,0
ACTR3BP6,0,0,0,0,0,0
5_8S_rRNA,0,0,0,0,0,0
AC137488.1,0,0,0,0,0,0
AC137488.2,0,0,0,0,0,0
CU013544.1,0,0,0,0,0,0
CT867976.1,0,0,0,0,0,0
CT867977.1,0,0,0,0,0,0
CT978678.1,0,0,0,0,0,0
CU459202.1,0,0,0,0,0,0
AC116618.1,0,0,0,0,0,0
CU463998.1,0,0,0,0,0,0
CU463998.3,0,0,0,0,0,0
CU463998.2,0,0,0,0,0,0
U6,0,0,0,0,0,0
LA16c-60D12.1,0,0,0,3,2,0
LA16c-13E4.3,0,0,0,0,0,1
LA16c-60D12.2,0,0,0,0,4,1
ZNF72P,0,0,0,0,1,0
--More--(1%)
```

The `less` command allows for horizontal scrolling if we append the `-S` option. We can also combine it with the `column` command to print tabular data with the columns nicely aligned. The `-t` option in `column` counts the number of columns and creates a table, while `-s` option tells

`column` the column separators in a data table (comma in this case, denoted by ',' in the command below). We pipe or send, using "|" the output of `column` to `less -S`. Hit q to get out of the following command and return to the prompt.

```
column -t -s ',' results.csv | less -S
```

```
name                baseMean  baseMeanA  baseMeanB  foldChange  log2I
SYNGR1              526.9     1012.5     41.3       0.04        -4.6
SEPT3               500.7     960.8      40.7       0.042       -4.6
YWHAH               797.4     1361.1     233.8      0.172       -2.5
RPL3                1710.7    828.2      2593.2     3.139       1.7
```

Pattern searching in Unix

We can use `grep` to search for patterns in files. For instance, the command below will find the word alachua in text_1.txt. The syntax for `grep` is the command, followed by the pattern, and where we like to find the pattern (text_1.txt in this case).

```
grep gainesville text_1.txt
```

```
gainesville
```

If we use the `-v` option, we can select lines in a file that does not contain a pattern. In the `grep` command below, we will print out every line in text_1.txt that does not contain alachua.

```
grep -v gainesville text_1.txt
```

```
oranges
blue
bananas
cats
dogs
apple
florida
gators
alachua
county
btep
```

Help Sessions

Lesson 1: Help session

Lesson recap

In this lesson, we achieved the following

- Learned about the Unix operating system
- Obtained an overview of Biowulf, which is the high performance and Unix-based computing cluster at NIH
- Learned how to sign onto Biowulf, either through using
    - student accounts OR
    - personal accounts

Practice session goals

The main goal for this practice session is to make sure that everyone can sign onto Biowulf. There are also practice questions to help reinforce what we have learned today.

Practice questions

Question 1:

Name the softwares for Windows and Macs that we use to connect to Biowulf.

{{Sdet}}{{Ssum}}Solution{{Esum}}

For Windows 10 or beyond users, we have the Command Prompt application. Alternatively, Windows users can use PuTTy or MobaXterm.

Mac users can use the built in Terminal application.

{{Edet}}

Question 2:

Connect to your Biowulf account, how do we do this?

{{Sdet}}{{Ssum}}Solution{{Esum}}

Via the Windows Command Prompt or Mac Terminal

```
ssh username@biowulf.nih.gov
```

{{Edet}}

Question 3:

What is the hierarchical architecture of Biowulf?

{{Sdet}}{{Ssum}}Solution{{Esum}}

- Cluster
- Computer/node
- Processor
- Core
- CPU

{{Edet}}

Question 4:

What is the Unix command for checking your username and group affiliation?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
id
```

{{Edet}}

Question 5:

What is the Unix command for making a new directory?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
mkdir
```

{{Edet}}

Lesson 2: Help session

Lesson recap

In this lesson, the following were accomplished.

- We obtained an understanding of limitations to tasks that we can perform in the various Biowulf spaces such as
    - Log in versus compute nodes
    - Home, data, and scratch directories
- Explored the Biowulf user dashboard
- We started to learn how to navigate Biowulf by
    - Checking the present working directory (`pwd`)
    - Listing directory content (`ls`)
    - Moving to different directories (`cd`)

Practice questions

Question 1:

What are some of the information that we can find from the Biowulf user dashboard?

{{Sdet}}{{Ssum}}Solution{{Esum}}

- Group affiliation
- Disk storage usage and request storage quota increase for the user's data directory
- Information on submitted jobs

{{Edet}}

Question 2:

True or False: Upon logging into Biowulf, we land in a compute node where we can start to do compute intensive tasks.

{{Sdet}}{{Ssum}}Solution{{Esum}}

False. Upon logging into Biowulf, we are taken to the login node where we can do the following

- Submit jobs
- Edit/compile code
- Manage files
- Transfer files

- Briefly test code

{{Edet}}

Question 3:

True or False: We should store our data and analysis outputs in our Biowulf home directory.

{{Sdet}}{{Ssum}}Solution{{Esum}}

False. A user's home directory is only 16 GB in size and cannot be increased. The home directory is meant for storing config files, code, state files, cache, etc. Store data and analysis output in the data directory.

{{Edet}}

Question 4:

True or False: We are doomed if we removed something that should not be removed.

{{Sdet}}{{Ssum}}Solution{{Esum}}

False. We can try to restore from the user's snapshot.

{{Edet}}

Question 5:

What command do we use to check which directory we are currently in on Biowulf?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
pwd
```

{{Edet}}

Question 6:

What command do we use to view the content of a directory in Biowulf?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
ls
```

{{Edet}}

Question 7:

What command do we use to navigate to different directories in Biowulf?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cd
```

{{Edet}}

Lesson 3: Help session

Lesson recap

In this lesson, we continued to learn about the Biowulf environment and should now be able to

- Copy content from one directory to another
- Use options of the `ls` command to view various directory content information
- Determine file and directory permissions and modified these
- Remove files

Practice questions

Question 1:

Can you copy the following folder to your data directory?

/data/classes/BTEP/unix_on_biowulf_2023_practice_sessions/lesson3_practice

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cp -r /data/classes/BTEP/unix_on_biowulf_2023_practice_sessions/lesso
```

If you are in your data directory then you can do the following, where "." denotes here in the present working directory

cp -r /data/classes/BTEP/unix_on_biowulf_2023_practice_sessions/lesson3_practice/ .

{{Edet}}

Question 2:

Can you change into lesson3_practice and then list its contents so that you see permissions?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cd lesson3_practice
```

```
ls -l
```

```
total 2
drwxr-s---. 2 student2 student2 4096 Jan 16 15:17 sample_sequence_dat
-rwxr-x---. 1 student2 student2   46 Jan 16 15:17 text1.txt
```

There are three sets of rwx permissions - a set for the file/directory owner, a set for those in the group, and a set for everyone else.

{{Edet}}

Question 3:

How many files and directories are in the folder lesson3_practice?

{{Sdet}}{{Ssum}}Solution{{Esum}}

There is 1 file (text1.txt) and a folder (sample_sequence_data)

{{Edet}}

Question 4:

What permission is set for the file in lesson3_practice?

{{Sdet}}{{Ssum}}Solution{{Esum}}

Owner has read, write, and execute Group has read and execute

{{Edet}}

Question 5:

Can you add write permission to the group for the file in lesson3_practice

{{Sdet}}{{Ssum}}Solution{{Esum}}

chmod g+w text1.txt

{{Edet}}

Question 6:

Can you make a copy of text1.txt and call it text1_copy.txt?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cp text1.txt text1_copy.txt
```

{{Edet}}

Question 7:

Can you remove text1_copy.txt?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
rm text1_copy.txt
```

{{Edet}}

Lesson 4: Help session

Lesson recap

In this lesson, we learned how to move files from one directory to another, rename files and folders. We also learned how to remove directories and use the `rm` command so that it confirms with us before removing.

Practice questions

For these exercises, copy the lesson4_practice_20230202 folder in the BTEP classes directory to your data directory by following the steps below.

First, sign into Biowulf

```
ssh username@biowulf.nih.gov
```

After connecting to Biowulf, change into your data directory

```
cd /data/username
```

Then, copy over the lesson4_practice_20230202 folder to your data directory (which should be your present working directory - denoted by ".")

```
cp -r /data/classes/BTEP/unix_on_biowulf_2023_practice_sessions/lesso
```

Question 1:

Change into the lesson4_practice_20230202 folder after you copied it to your data directory. What is in this folder? How many subfolders does it have?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cd lesson4_practice_20230202/
```

```
ls -l
```

The lesson4_practice_20230202 folder has one subfolder (example_rna_seq_counts)

```
total 1
drwxr-s---. 2 student1 student1 4096 Jan 17 10:41 example_rna_seq_cou
```

{{Edet}}

Question 2:

Change into the example_rna_seq_counts folder. How many files are in this folder?

{{Sdet}}{{Ssum}}Solutions{{Esum}}

```
cd example_rna_seq_counts/
```

```
ls -l
```

There are three files in the example_rna_seq_counts folder. These are gene expression counts tables for NCBI SRA studies SRP025982, SRP048685, and SRP011233 obtained from recount2 *(https://jhubiostatistics.shinyapps.io/recount/)*, which is a repository of RNA sequencing count tables.

```
total 360960
-rwxr-x---. 1 student1 student1 366932592 Jan 17 10:41 counts_gene_1
-rwxr-x---. 1 student1 student1    883910 Jan 17 10:41 counts_gene_2
-rwxr-x---. 1 student1 student1   1590821 Jan 17 10:41 counts_gene_3
```

{{Edet}}

Question 3:

Unfortunately, file names for gene expression counts data from recount2 are very generic (like what we see in the example_rna_seq_counts directory). Thus, we would like to rename these files to make them more informative. So knowing that

- counts_gene_1.tsv is derived from study SRP025982
- counts_gene_2.tsv is derived from study SRP048685
- counts_gene_3.tsv is derived from study SRP011233

Can you rename the files counts_gene_1.tsv to SRP025982.tsv, counts_gene_2.tsv to SRP048685.tsv, and counts_gene_3.tsv to SRP011233.tsv so that the file names tells us the study in which they came from?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
mv counts_gene_1.tsv SRP025982.tsv
```

```
mv counts_gene_2.tsv SRP048685.tsv
```

```
mv counts_gene_3.tsv SRP011233.tsv
```

{{Edet}}

Question 4:

Go back up one directory to the lesson4_practice_20230202 folder and make a new folder called rna_seq_recounts and then move the expression counts tables into this folder

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
mkdir rna_seq_recounts
```

```
mv example_rna_seq_counts/SRP025982.tsv rna_seq_recounts
```

```
mv example_rna_seq_counts/SRP048685.tsv rna_seq_recounts
```

```
mv example_rna_seq_counts/SRP011233.tsv rna_seq_recounts
```

{{Edet}}

Question 5:

Change into the rna_seq_recounts folder and delete SRP025982.tsv, but make sure that we are asked whether we really want to delete.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cd rna_seq_recounts
```

```
rm -i SRP025982.tsv
```

{{Edet}}

Question 6:

The example_rna_seq_counts folder in the lesson4_practice_20230202 directory should now be empty. How do you remove an empty directory in Unix.

{{Sdet}}{{Ssum}}Solution{{Esum}}

Go back up one directory to lesson4_practice_20230202

```
cd ..
```

```
rmdir example_rna_seq_counts
```

{{Edet}}

Lesson 5: Help session

Lesson recap

In this lesson, we learned how to request an interactive session to perform compute intensive tasks on Biowulf. We also learned about bioinformatics applications that are installed on Biowulf and explored tools used in high throughput sequencig analysis.

Practice questions

Be sure to stay in your data directory for this exercise. In your data directory, create a folder called srr1553423_fastqc.

```
mkdir srr1553423_fastqc
```

We are going to download sequencing data for NCBI SRA study SRR1553423 *(https://trace.ncbi.nlm.nih.gov/Traces/index.html?view=run_browser&acc=SRR1553423&display=metadata)* using the sratool kit and assay quality using fastqc.

Question 1:

Can you request an interactive session with 5 gb of lscratch space?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
sinteractive --gres=lscratch:5
```

{{Edet}}

After the interactive session has been granted, change into the srr1553423_fastqc directory.

```
cd srr1553423_fastqc
```

Question 2:

How do we load the sratoolkit and fastqc?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
module load sratoolkit
```

```
module load fastqc
```

{{Edet}}

Question 3:

Can you download the first 10,000 reads for SRR1553423? This is paired end sequencing data.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
fastq-dump --split-files -X 10000 SRR1553423
```

{{Edet}}

Question 4:

How many files where downloaded?

{{Sdet}}{{Ssum}}Solution{{Esum}}

Two fastq files were downloaded.

```
ls
```

```
SRR1553423_1.fastq  SRR1553423_2.fastq
```

{{Edet}}

Question 5:

How do we assess sequencing data quality?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
fastqc SRR1553423_1.fastq  SRR1553423_2.fastq
```

{{Edet}}

Question 6:

We are done with our work on the interactive session, how do we terminate this interactive session?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
exit
```

{{Edet}}

Lesson 6: Help session

Lesson recap

In this lesson, we learned to submit a short script that downloads sequencing data from NCBI SRA and subsequently assess sequencing data quality. We then transferred the QC data to our local computer for viewing.

Practice questions

The exercises below will further help you develop proficiency in using the Unix nano editor. You will modify the script we used in the lesson to download and assess the quality of sequencing data for NCBI SRA study SRR1553423. You will also become more comfortable with transferring data from Biowulf to your local computer for viewing.

Question 1:

What is the first step to editing a new or an existing file using nano.

{{Sdet}}{{Ssum}}Solution{{Esum}}

If you are editing a new file, then the command below will open a blank editor

```
nano filename
```

{{Edet}}

Question 2:

After you are done editing, what is are the steps to exiting nano and returning to the prompt?

{{Sdet}}{{Ssum}}Solution{{Esum}}

Hit control x

- If you made edits, nano will ask if you like to save.
    - If you choose yes to save, nano will ask you to confirm the file name.
    - If you choose no to not save, then you will return to the prompt.
- If you did not make edits then you will just be returned to the prompt.

{{Edet}}

Question 3:

Create new directory in your data folder called srr1553423_fastqc and then change into this.

{{Sdet}}{{Ssum}}Solution{{Esum}}

Change into your data directory if you are not in it

```
cd /data/username
```

```
mkdir srr1553423_fastqc
```

```
cd srr1553423_fastqc
```

{{Edet}}

Question 4:

Stay in the srr1553423_fastqc folder. Copy SRR1553606_fastqc.sh from /data/classes/BTEP/unix_on_biowulf_2023_documents/SRR1553606_fastqc to the srr1553423_fastqc folder. Change the file name to SRR1553423_fastqc.sh.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
cp /data/classes/BTEP/unix_on_biowulf_2023_documents/SRR1553606_fastc
```

```
mv SRR1553606_fastqc.sh SRR1553423_fastqc.sh
```

{{Edet}}

Question 5:

We need to make a few edits to SRR1553423_fastqc.sh before we can submit it as a batch job. Open the script with nano and make the necessary edits.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
nano SRR1553423_fastqc.sh
```

Before submitting this script we need to

- change the job-name to SRR1553423_fastqc
- change the user email to your NIH email
- for those connecting to Biowulf using their own accounts, remove the following line
    - "#SBATCH --partition=student"
- change the name of the output log file to SRR1553423_fastqc_log
- in the first comment line after loading modules, replace SRR1553606 with SRR1553423
- replace SRR1553606 with SRR1553423 in the fastq-dump command
- replace
    - Replace the output directory for the fastqc results to /data/$USER/srr1553423_fastqc
    - SRR1553606 with SRR1553423 in the fastqc command

Save these changes and exit

{{Edet}}

Question 6:

Now, submit SRR1553423_fastqc.sh as a batch job.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
sbatch SRR1553423_fastqc.sh
```

{{Edet}}

Question 7:

List the contents of the srr1553423_fastqc folder, what are the names of the html fastqc reports?

{{Sdet}}{{Ssum}}Solution{{Esum}}

Below are the two html fastqc reports generated for the sequencing data belonging to NCBI SRA SRR1553423.

```
SRR1553423_1_fastqc.html
SRR1553423_2_fastqc.html
```

{{Edet}}

Question 8:

Can you use `scp` to copy SRR1553423_1_fastqc.html and SRR1553423_2_fastqc.html to your local computer for viewing. Use either the Mac Terminal or Windows Command Prompt for this and save it to your downloads folder.

{{Sdet}}{{Ssum}}Solution{{Esum}}

Mac users: open the Terminal and change into your Download directory

```
cd ~/Downloads
```

Replace username below with the username you used to connect to Biowulf

```
scp username@helix.nih.gov:/data/username/srr1553423_fastqc/SRR15534
```

```
scp username@helix.nih.gov:/data/username/srr1553423_fastqc/SRR15534
```

Enter your password when promoted during the secure copy process.

Windows users:

Refer to Figure 17 in the Lesson 6 documentation to change into your Windows downloads folder. Then, use the `scp` command to copy the html fastqc reports for SRR1553423 to your local.

Replace username below with the username you used to connect to Biowulf

```
scp username@helix.nih.gov:/data/username/srr1553423_fastqc/SRR15534
```

```
scp username@helix.nih.gov:/data/username/srr1553423_fastqc/SRR15534
```

Enter your password when promoted during the secure copy process.

{{Edet}}

Lesson 7: Help session

Lesson recap

This lesson has taught us how to download data from the web in Unix. We are also able to view file content and to search for patterns in files.

Practice questions

Question 1:

Make a folder in your data directory called lesson7_practice and change into it.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
mkdir lesson7_practice
```

```
cd lesson7_practice
```

{{Edet}}

Question 2:

Next, goto the course data section *(https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/unix_introduction_2023_data/)* of the class documentation. Download 22_transcriptome.fa into the lesson7_practice folder using `wget`. This file contains sequences corresponding to the transcripts found in human chromosome 22.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
wget https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/data/22_t
```

{{Edet}}

Question 3:

Next, goto the course data section *(https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/unix_introduction_2023_data/)* of the class documentation. Download 22.gtf into the lesson7_practice folder using `curl`. This is the genomic annotation file for human chromosome

22, which tells us where features such as genes, transcripts, exons, and coding sequences are found along a genome.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
curl https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/data/22.gt
```

OR

```
curl -O https://btep.ccr.cancer.gov/docs/unix-on-biowulf-2023/data/22
```

{{Edet}}

Question 4:

Print the first six lines of 22_transcriptome.fa.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
head -n 6 22_transcriptome.fa
```

```
>ENST00000615943.1 loc:chr22|10736171-10736283|- exons:10736171-10736
ATCACTTCTCGGCCTTTTGGCTAAGATCAACTGTAGTATCTGTTGTTATTAATATAATATTGTATATTC
ACCAATTGTCAATACAAGGCTGTTTGTATCTGATATGAACCAA
>ENST00000618365.1 loc:chr22|10936023-10936161|- exons:10936023-10936
AGCATGCCCAGTTAATTTGAAATTTCAGATAAACAAATACTTTTTTCAGTGTAAGTATATCCCATACAA
ATTTGGGACATGCTTATACTAAAATATTATTCCTTATTTATCTGAAATTGAAATTTAACTGGGTATTAC
```

{{Edet}}

Question 5:

Print the last eight lines of 22_transcriptome.fa.

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
tail -n 8 22_transcriptome.fa
```

```
>ENST00000427528.1 loc:chr22|50798655-50799123|+ exons:50798655-50799
ATGGCACCAAAAGCGAAGGAAGCTCCTGCTCATCCTAAAGCCGAAGCCAAAGCGAAGGCTTTAAAGGCC
AGAAGGCAGTGTTGAAAGGTGTCCGCAGCCACACGCAAAAAAGAAGATCCGCATGTCACTCACCTTCAC
```

```
CGGCCCAAGACACTGCGACTCCGGAGGCAGCCCAGATATCCTCGGAAGAGCACCCCCAGGAGAAACAA(
TTGGCCACTATGCTATCATCAAGTTTCCGCTGGCCACTGAGTCGGCCGTGAAGAAGATAGAAGAAAAC/
CACGCTTGTGTTCACTGTGGATGTTAAAGCCAACAAGCACCAGATCAGACAGGCTGTGAAGAAGCTCT/
GACAGTGATGTGGCCAAGGTCACCACCCTGATTTGTCCTGATAAGGAGAACAAGGCATATGTTCGACT
CTCCTGATTATGATGCTTTCGATGTTGTAACAAAATTGGGATCACCTAA
```

{{Edet}}

Question 6:

Can you find the transcript ENST00000615943.1 in the file 22.gtf? What is the name of the gene in which it is derived?

{{Sdet}}{{Ssum}}Solution{{Esum}}

```
grep ENST00000615943.1 22.gtf
```

This transcript comes from the gene U2, which codes for a snRNA

{{Edet}}

Course data

Data for Introduction to Unix on Biowulf 2023

unix_on_biowulf_2023.zip

SRR1553606_fastqc.sh

22_transcriptome.fa

22.gtf

Connecting to Biowulf (additional methods)

Interfacing with Biowulf using Putty

Putty is an open source and grpahical based ssh client that is also capable of scp and sftp. It is one of the ways to interface with Biowulf for Windows users. To obtin Putty, goto https://www.putty.org *(https://www.putty.org)* (Figure 1). At the Putty website, click on the Download Putty link (Figure 1).

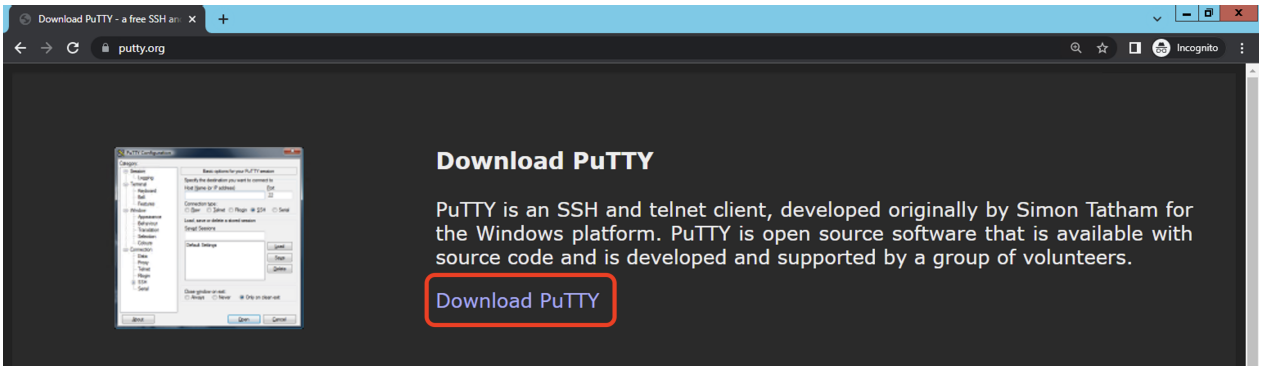


Figure 1

Subsequently, we will be taken to a page that houses several download options. To avoid having to install anything, we can grab the ".exe" files under the "Alternative binary files" section (Figure 2). Make sure to get the 64-bit x86 versions. Download putty.exe (the ssh client), pscp.exe (for scp), and psftp.exe (for sftp).

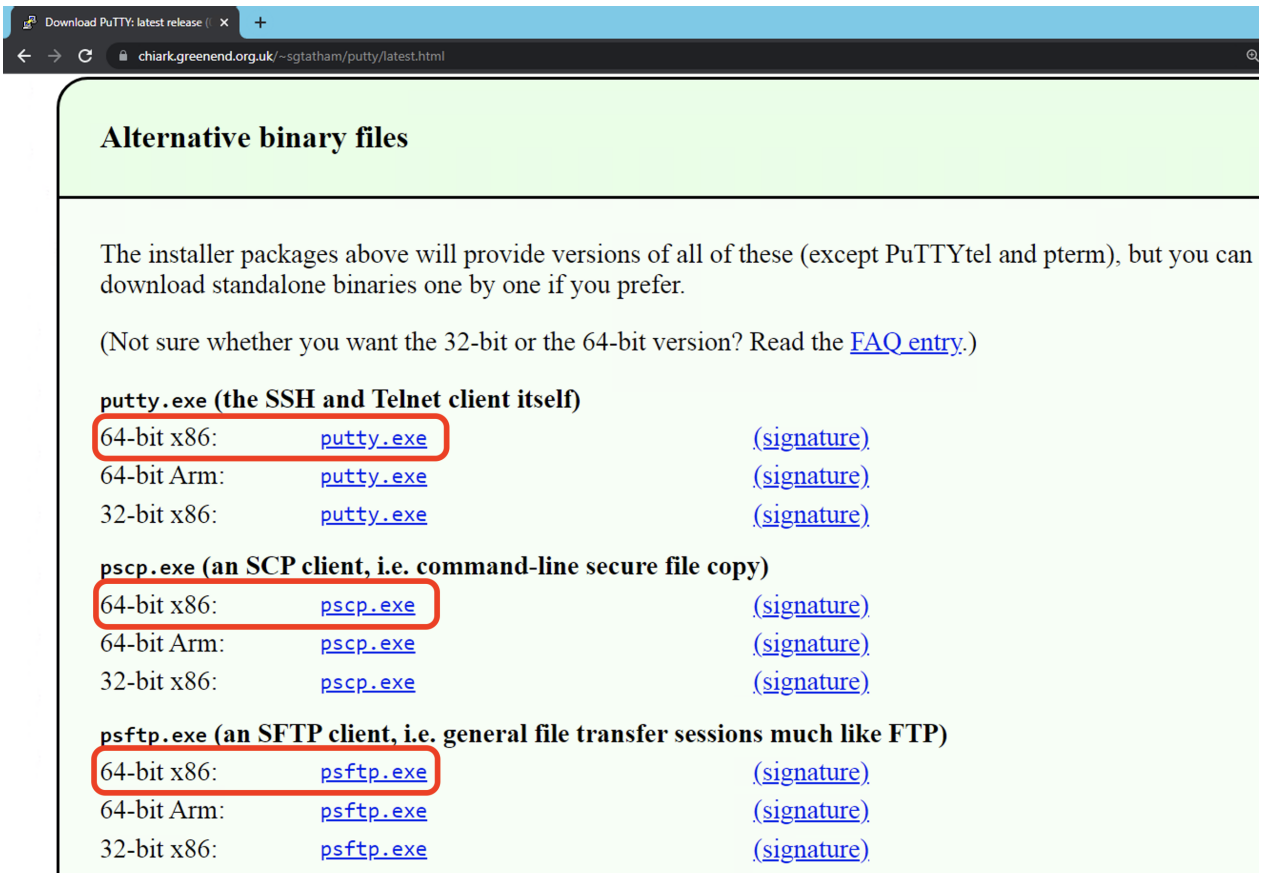


Figure 2

In this example, I have downloaded putty.exe, pscp.exe, and psftp.exe onto the Windows desktop.

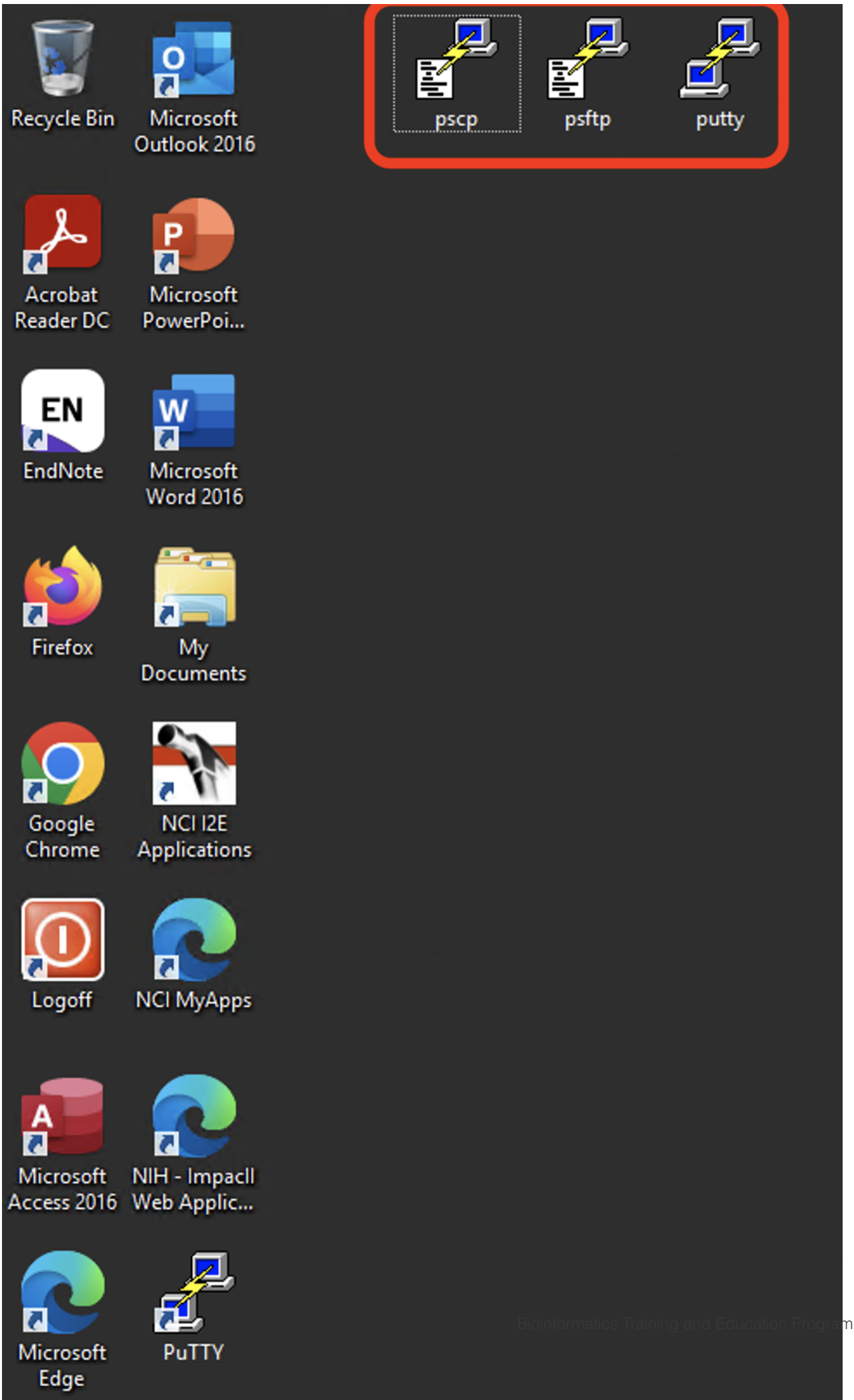

pscp
psftp
putty
Recycle Bin
Microsoft Outlook 2016
Acrobat Reader DC
Microsoft PowerPoi...
EndNote
Microsoft Word 2016
Firefox
My Documents
Google Chrome
NCI I2E Applications
Logoff
NCI MyApps
Microsoft Access 2016
NIH - ImpacII Web Applic...
Microsoft Edge
PuTTY

Figure 3

To connect to Biowulf, open putty and in the dialogue box that appears, enter biowulf.nih.gov in the box labeled "Host Name (or IP address), make sure the "Port" is set to 22, and that we choose SSH as the "Connection type". See Figure 4. Once the information has been entered, hit "Open".

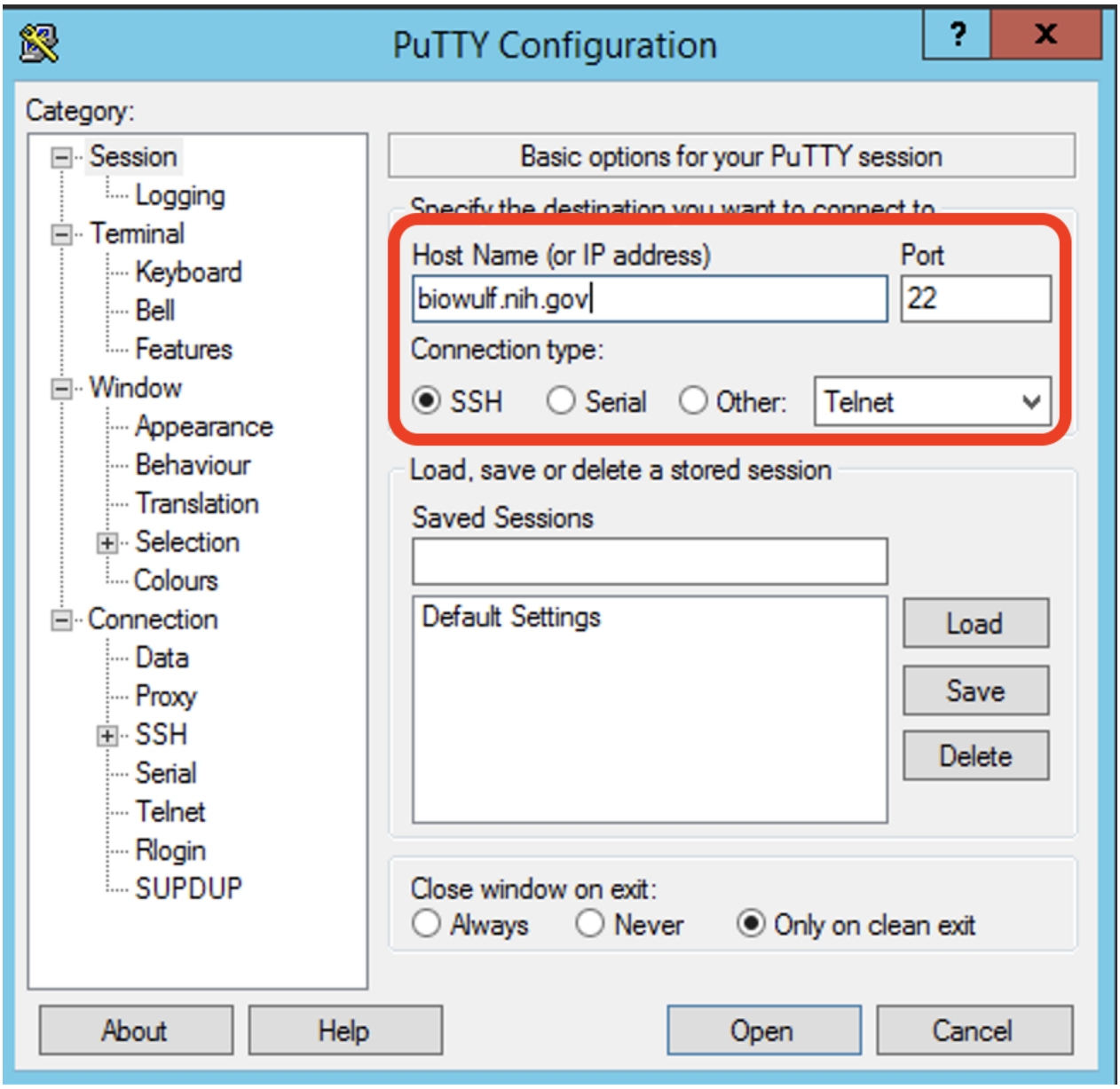


Figure 4

We will then be taken to a terminal where we entering our login credentials (Figur 5 and Figure 6)

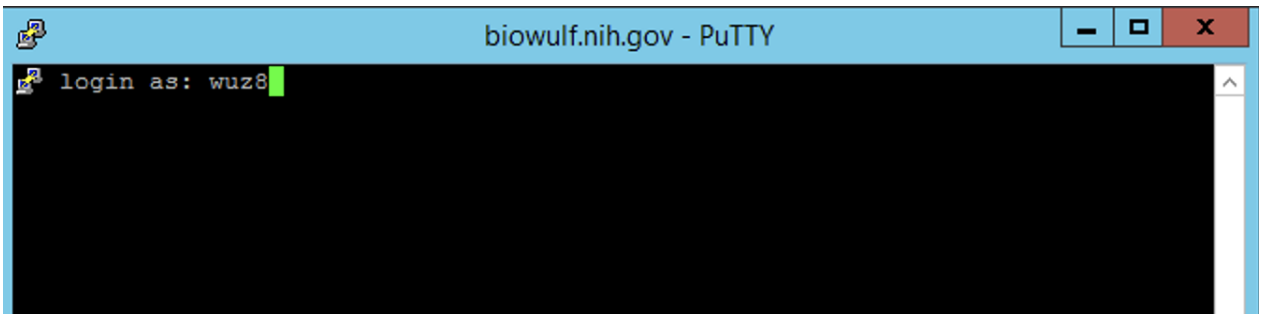


Figure 5

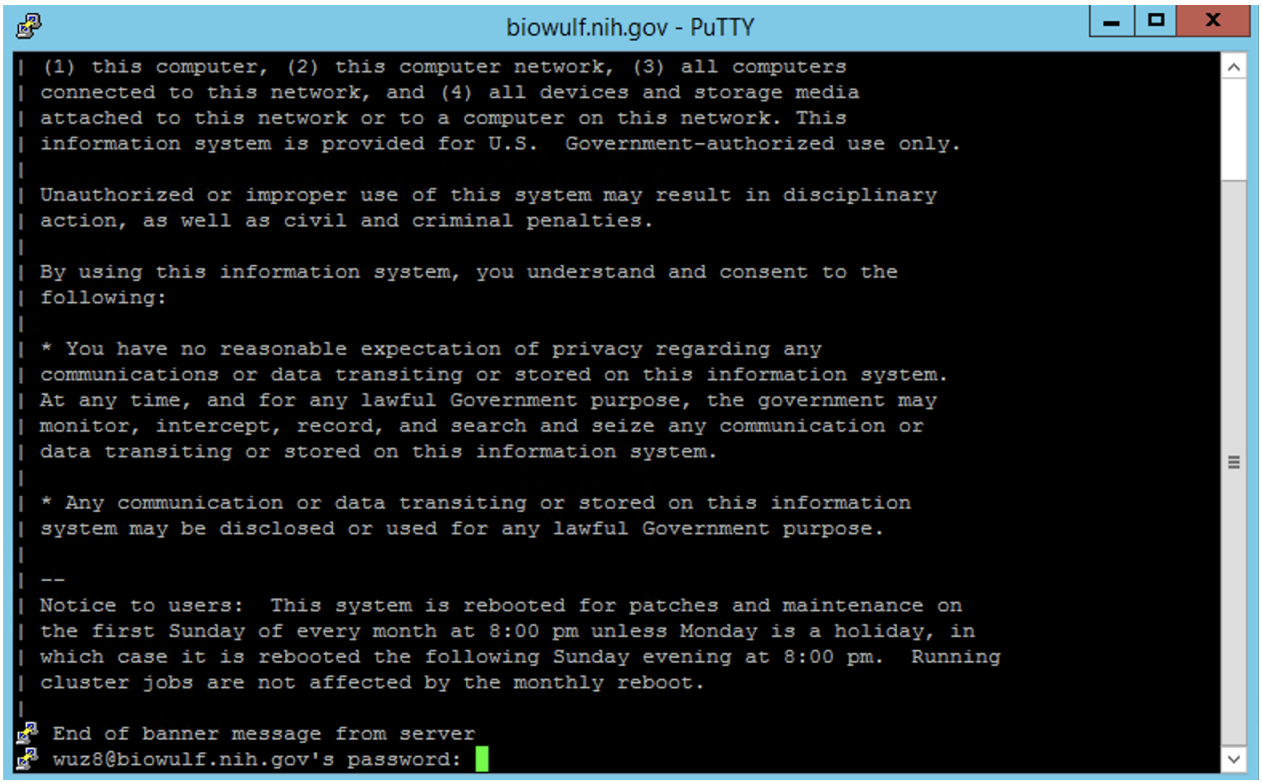


Figure 6

We will reach the Biowulf prompt after successfully logging in (Figure 7).

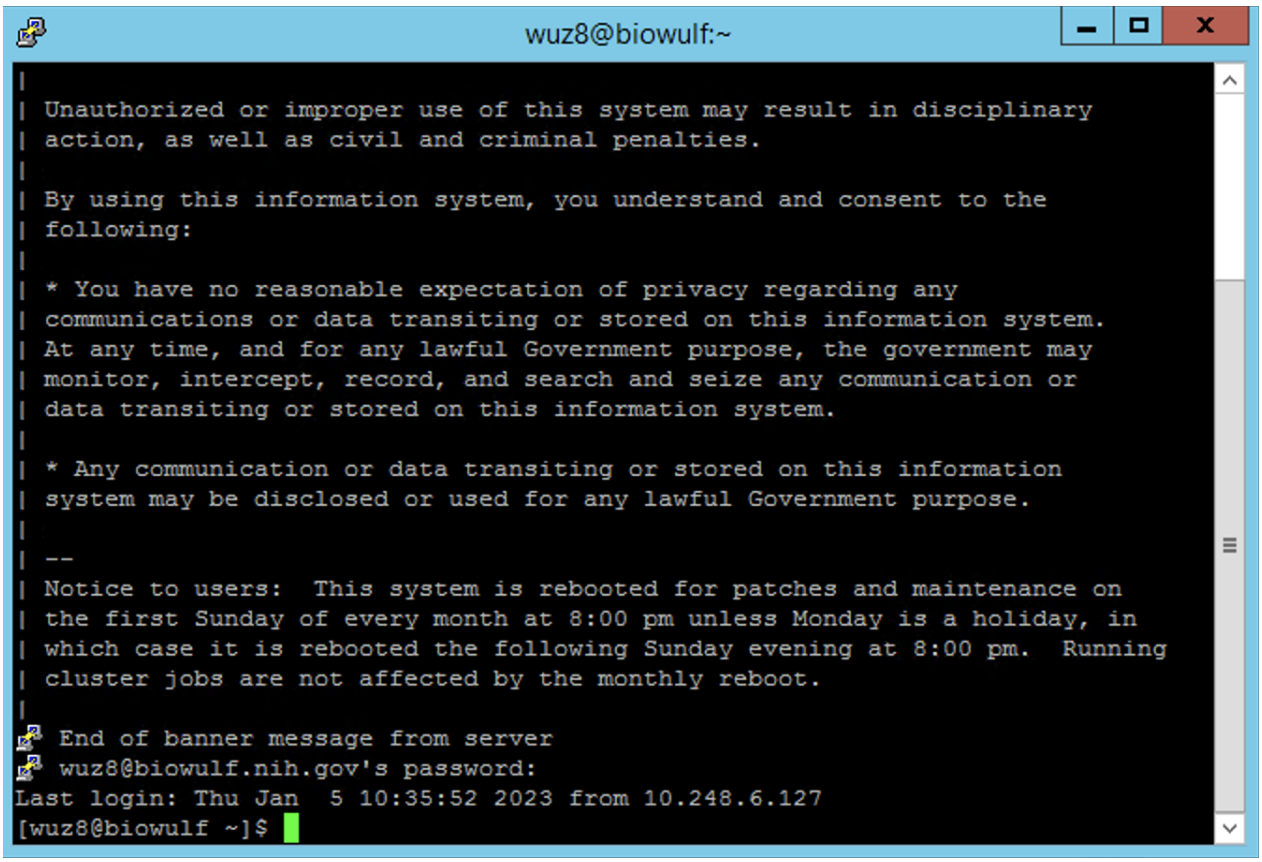


Figure 7

If we open a Windows Command Prompt, and change into the directory where pscp.exe was downloaded (in this case O:\Users\wuz8\Desktop), we can transfer files from Biowulf to our local machine using the Putty version of scp, which is pscp (Figure 8).

Figure 8

Interfacing with Biowulf using Mobaxterm

Mobaxterm is another open source and graphical based ssh client that Windows users can use to interact with Biowulf. To obtain Mobaxterm, goto https://mobaxterm.mobatek.net *(https://mobaxterm.mobatek.net)* and click the Download tab at the top (Figure 1).

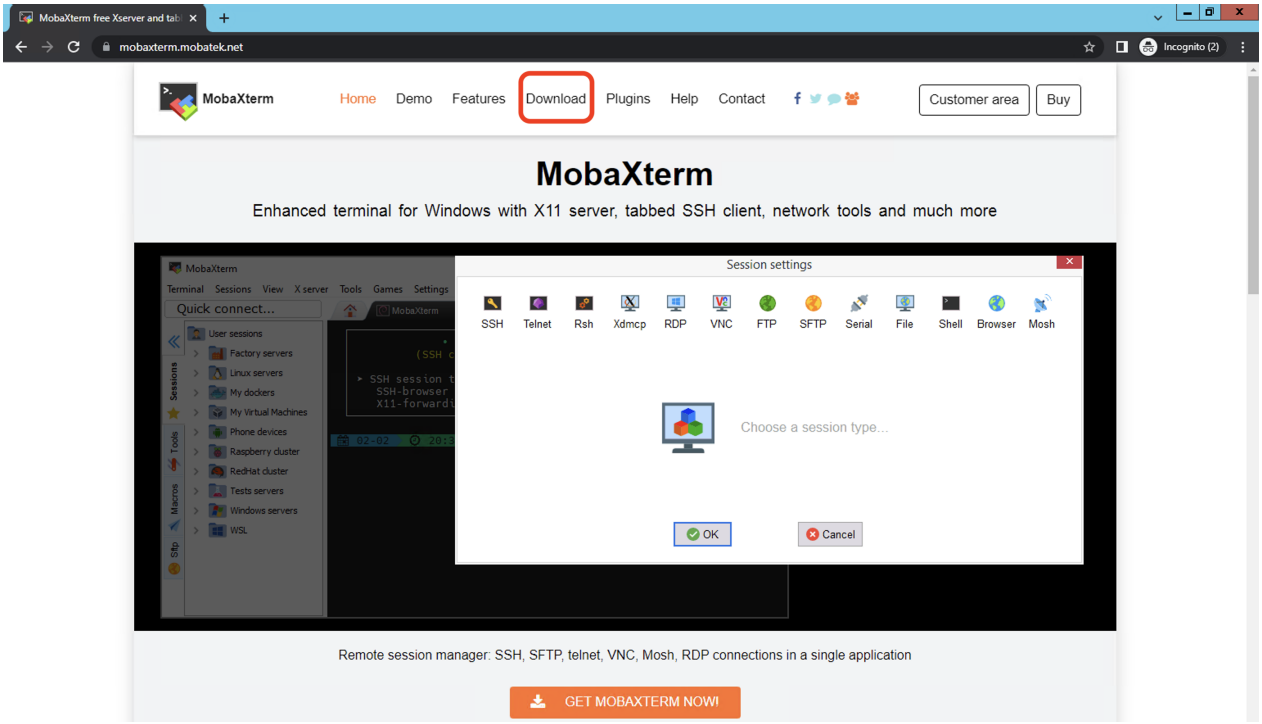


Figure 1

Subsequently, select to download the Home Edition, which is free (Figure 2).

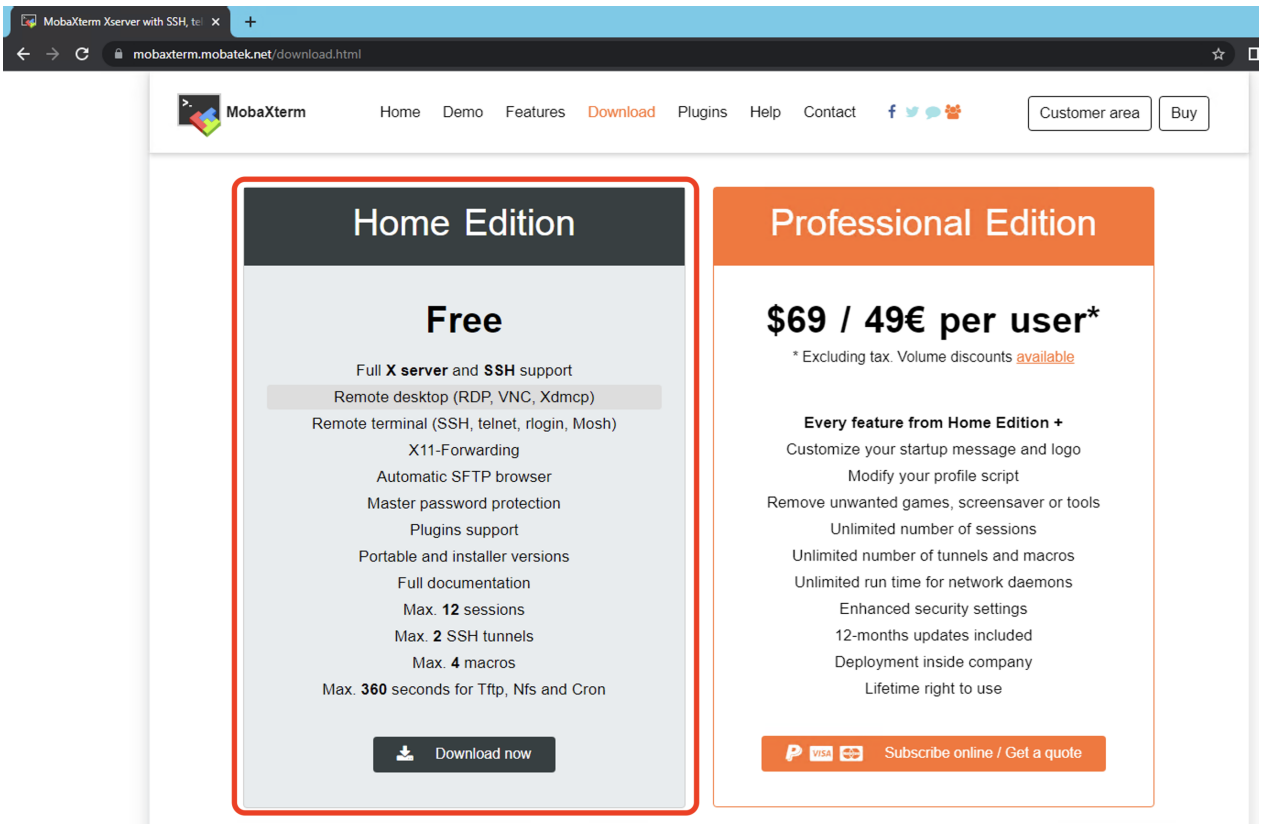


Figure 2

To avoid installing, choose to download the Portabl edition (Figure 3).

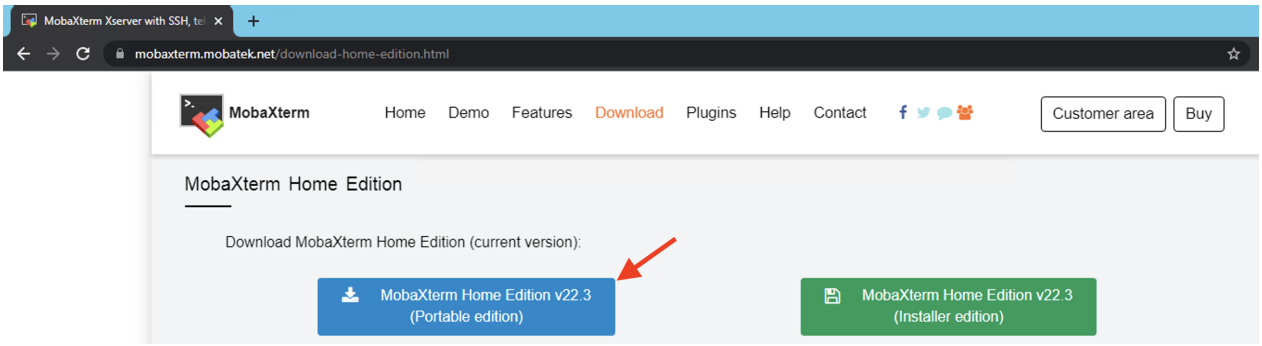


Figure 3

The Portable version was downloaded onto Windows desktop in zip folder (Figure 4).

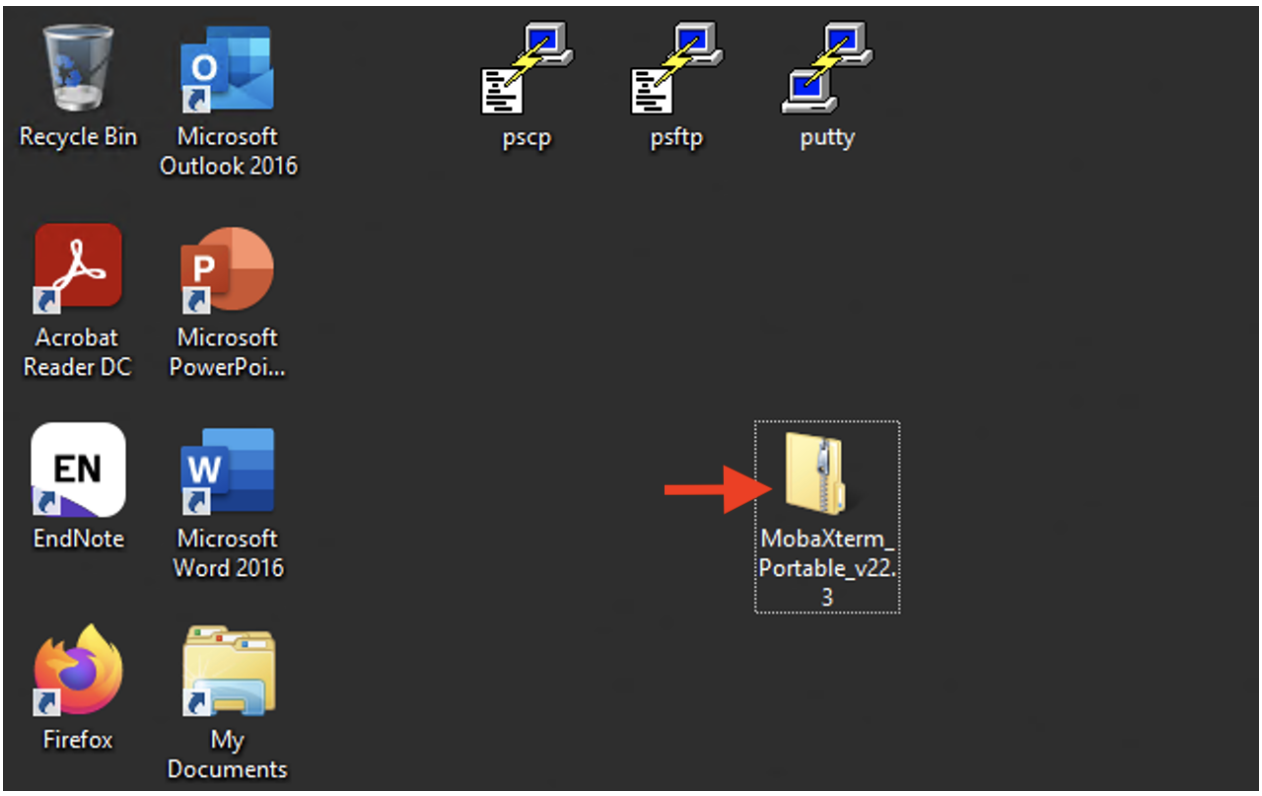


Figure 4

Right click on the zip folder and choose Extract All (Figure 5).

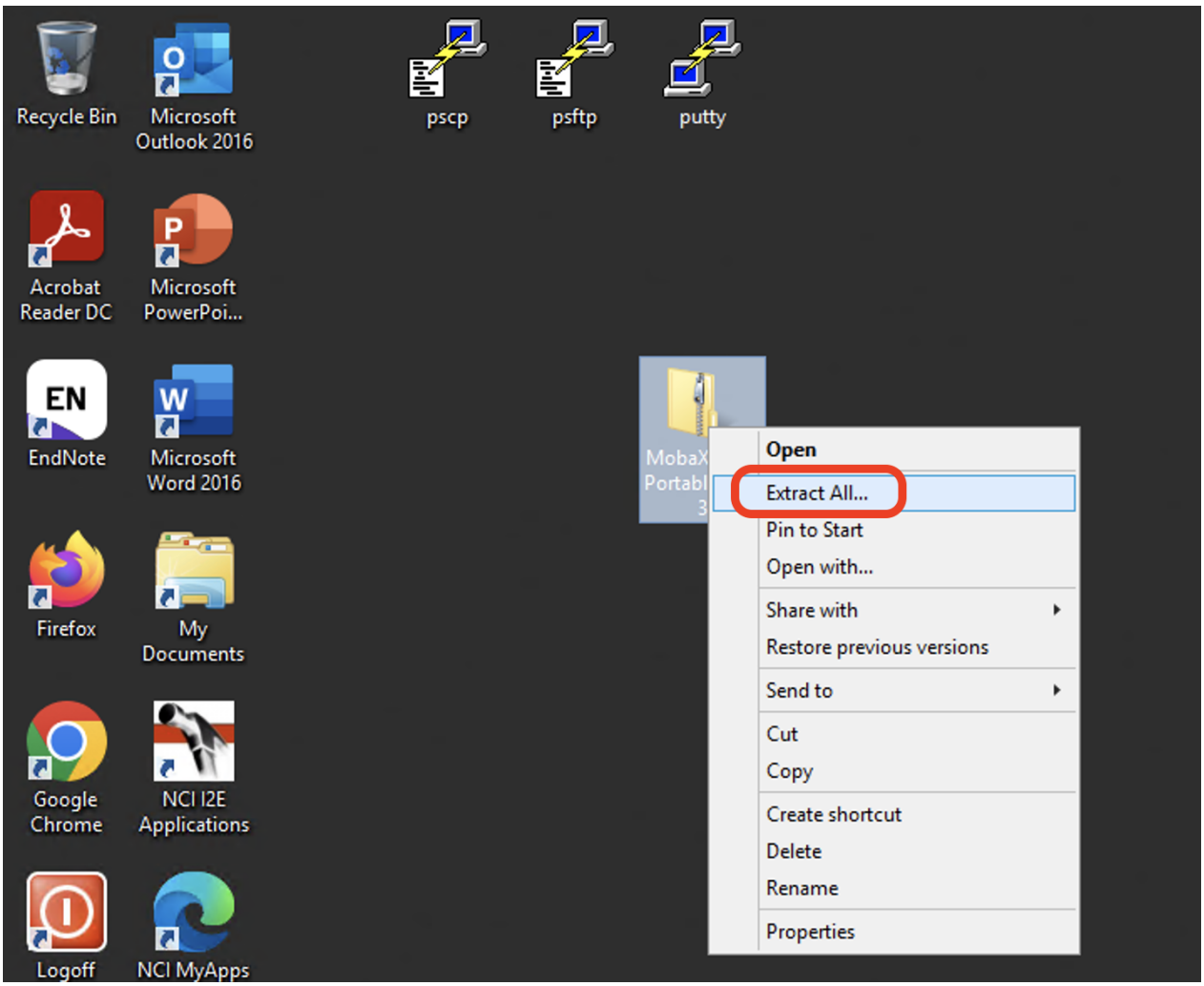


Figure 5

Confirm where you want the unzipped folder to go and then select Extract (Figure 6). In this example, the contents will be extracted onto the Windows desktop.

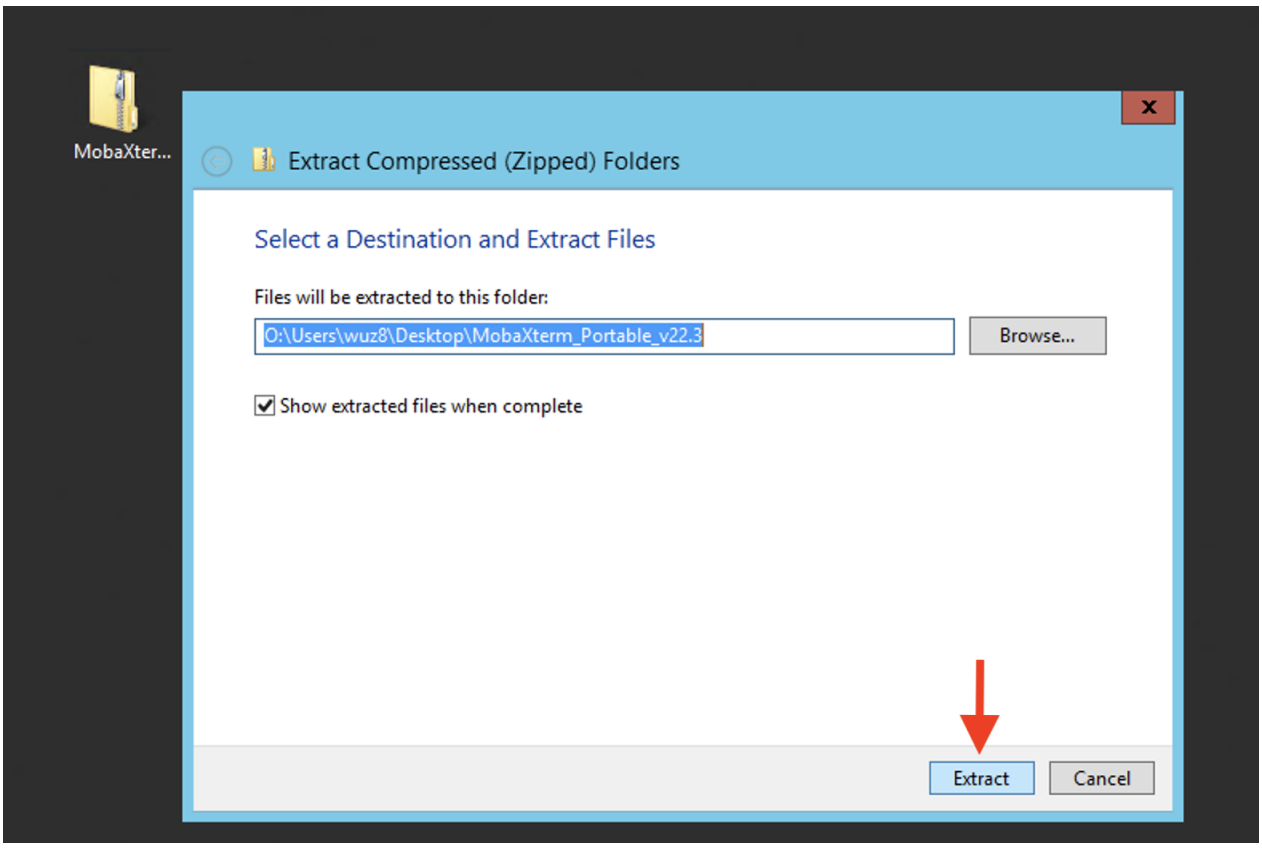


Figure 6

Once extracted, we will see a folder with the Mobaxterm icon (Figure 7).

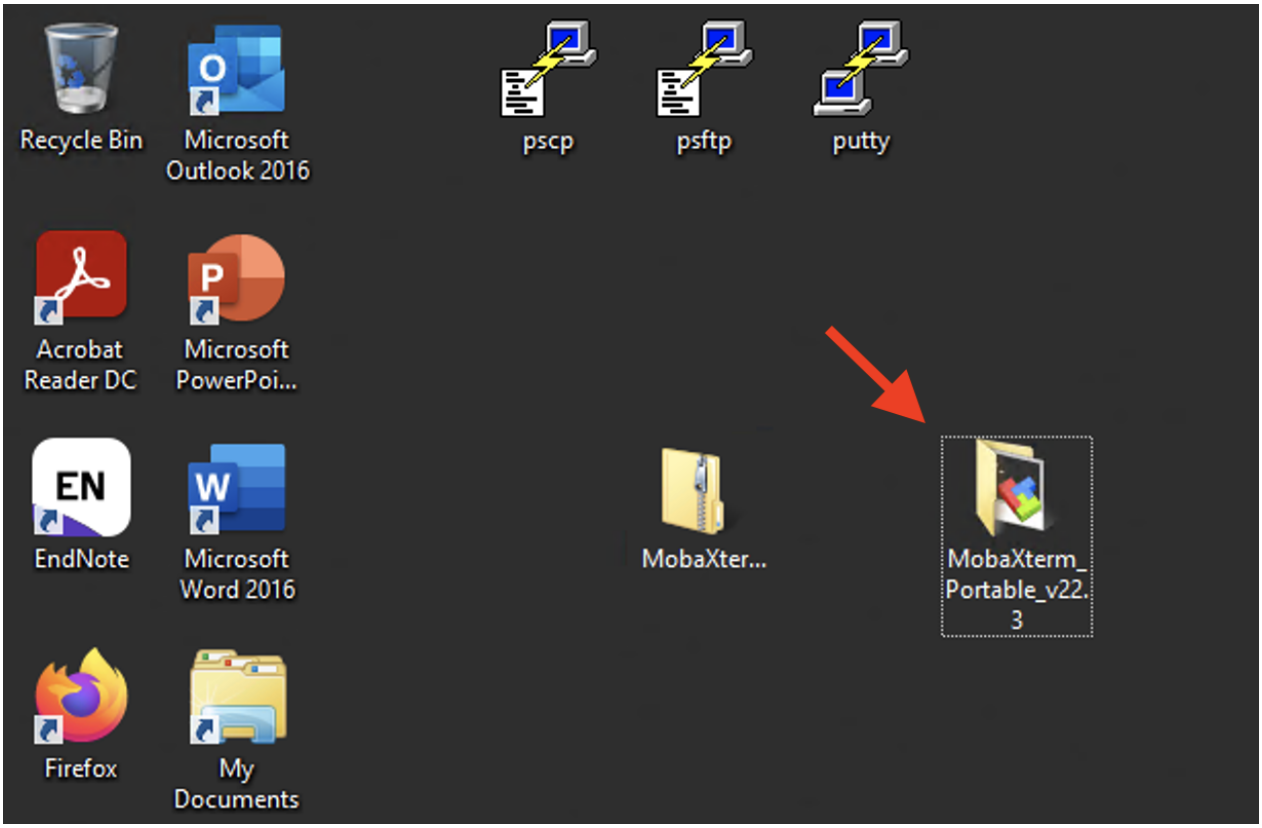


Figure 7

Open the unzipped folder and click on the Mobaxterm application file (MobaXterm_Personal_22.3) (Figure 8).

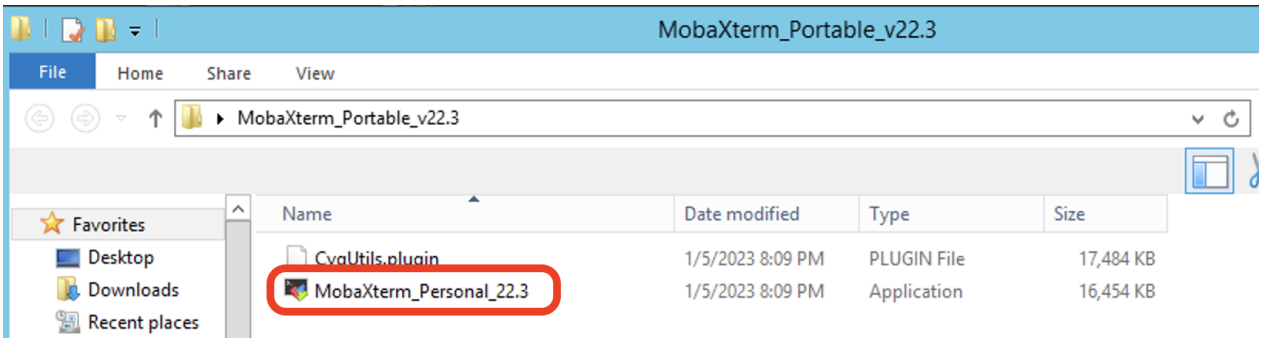


Figure 8

We will see the Mobaxterm client interface. To connect to Biowulf, we can click on Terminal and select "Open new tab" (Figure 9).

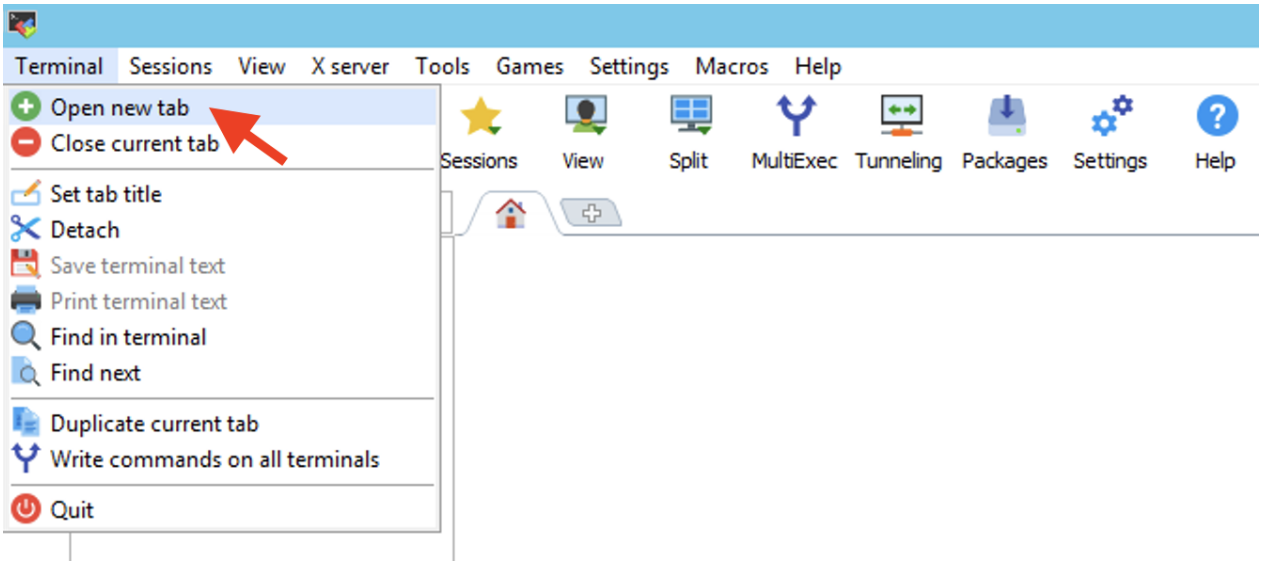


Figure 9

A local Unix terminal should open. From here, we can use the `ssh` command to connect to Biowulf (Figure 10).

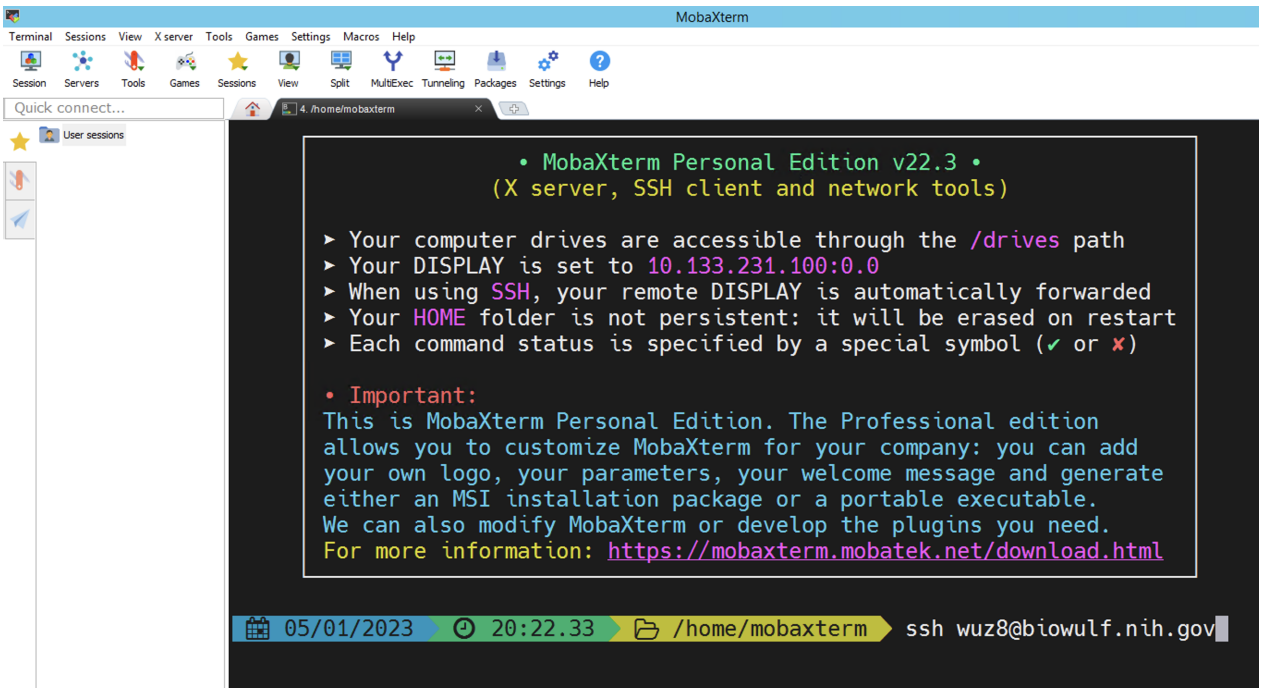


Figure 10

When asked whether we want to continue, select yes (Figure 11). The message in Figure 11 appears when your machine connects to Biowulf for the first time.

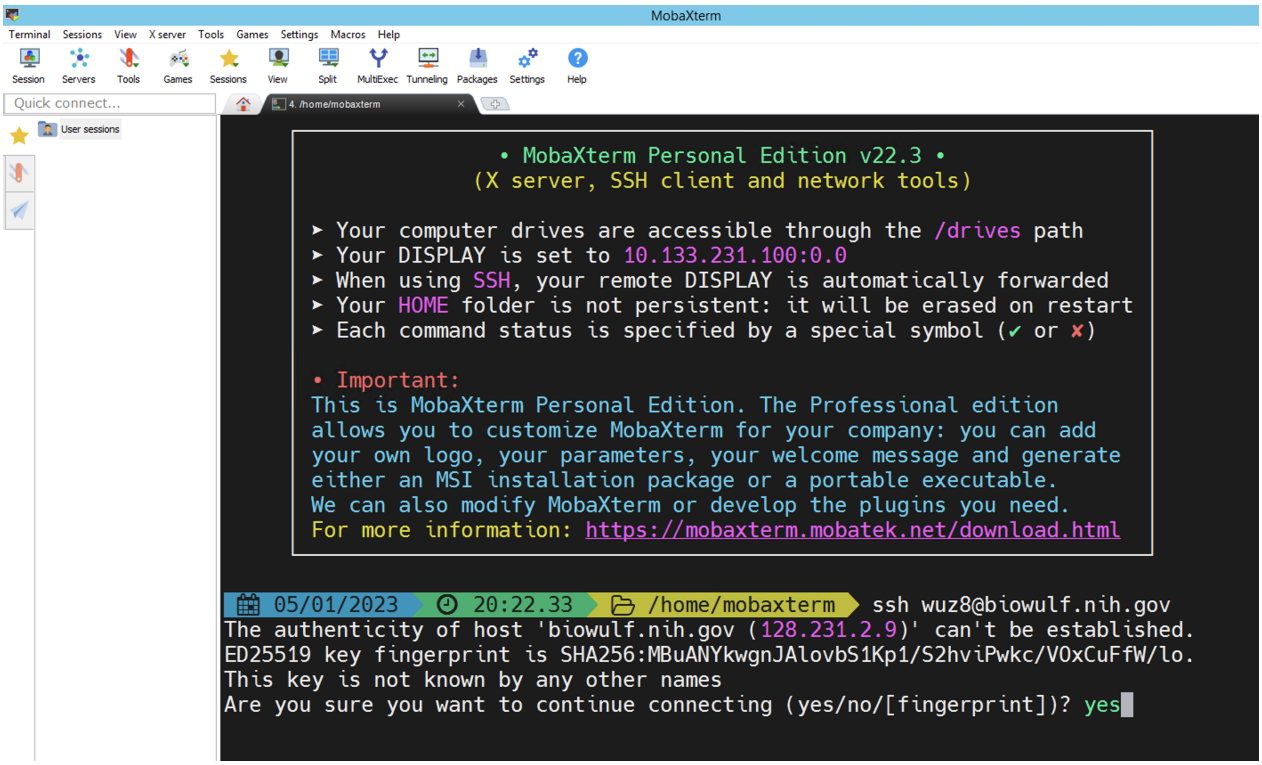


Figure 11

Enter Biowulf password (Figure 12).

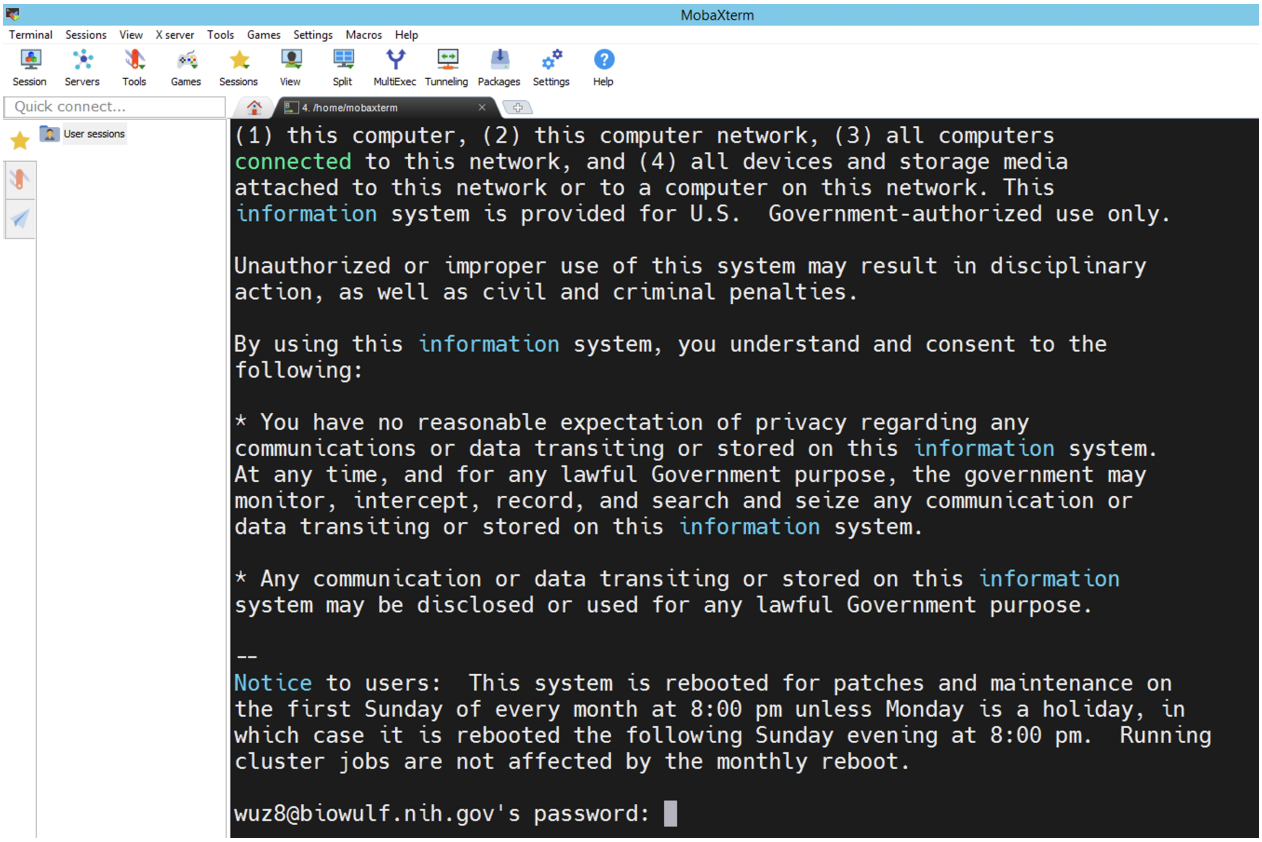


Figure 12

Hit No to not save the password (Figure 13) and we should land in our Biowulf terminal prompt.

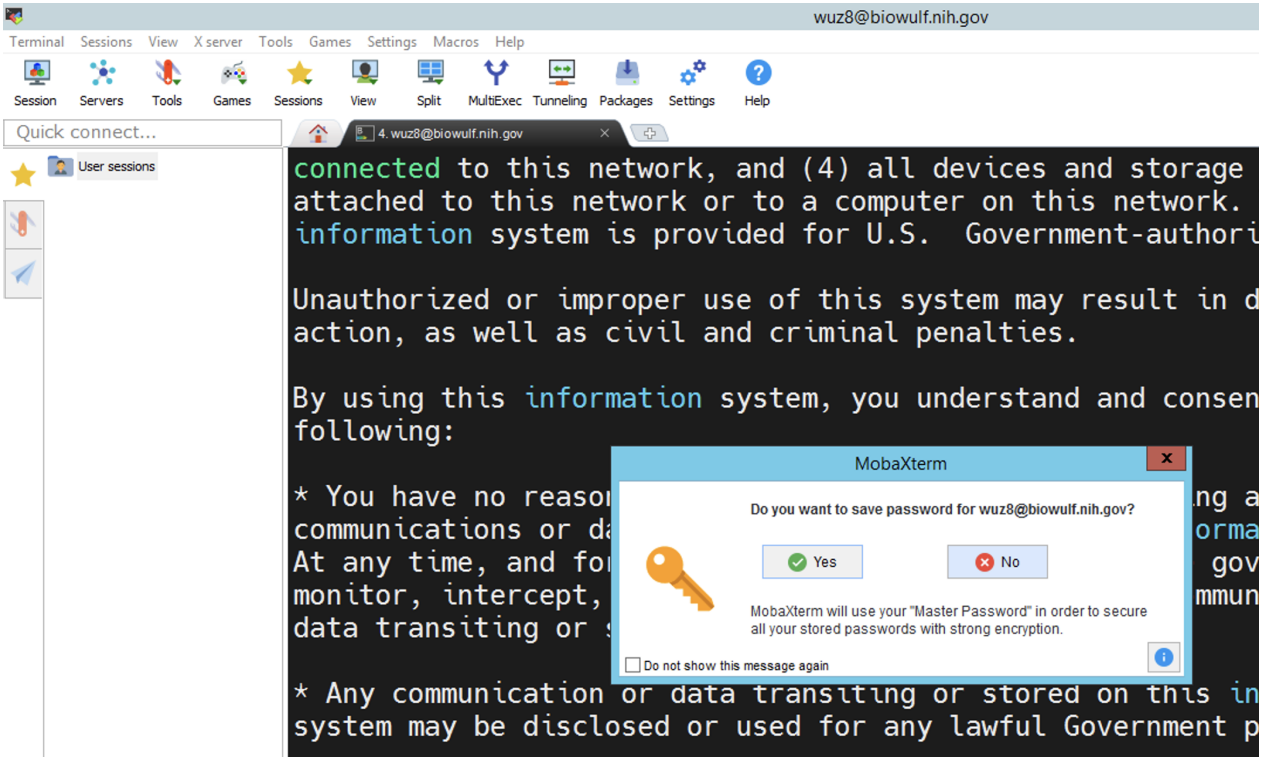
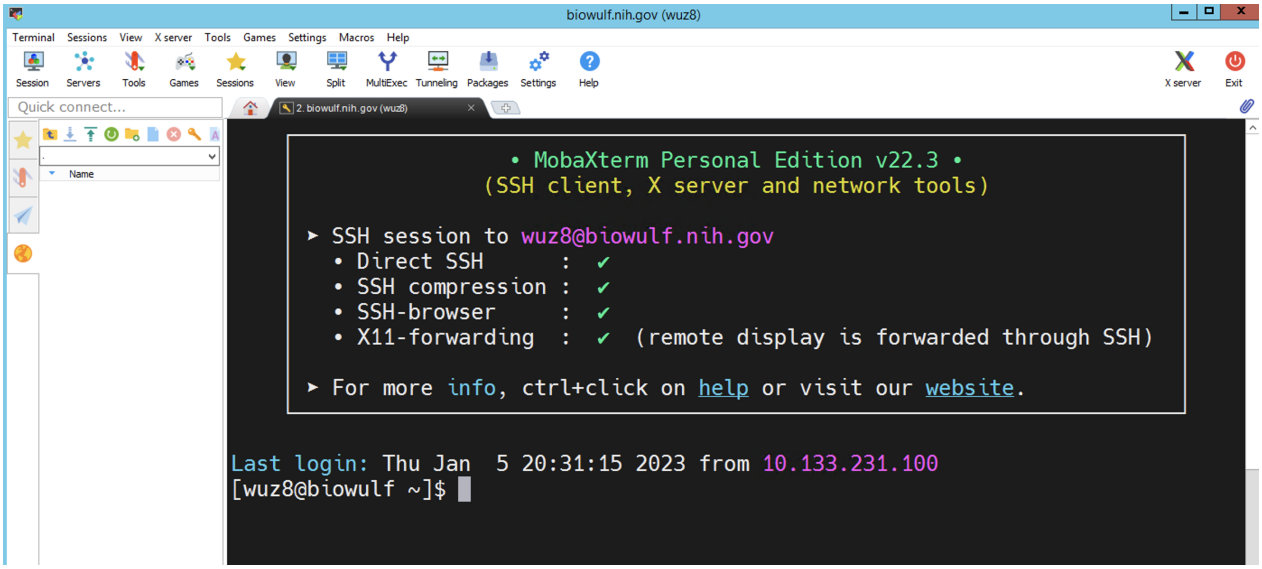


Figure 13



Mobaxterm - Biowulf connection successful

An alternative method for connecting to Biowulf is to select the Session tab (Figure 14) and choose to start a new SSH session (Figure 15).

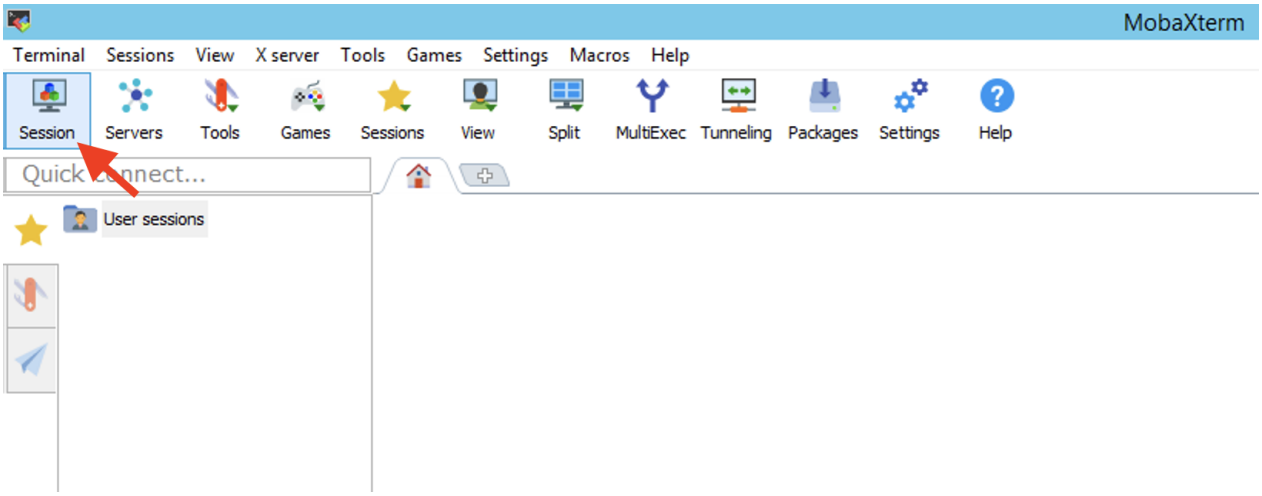


Figure 14

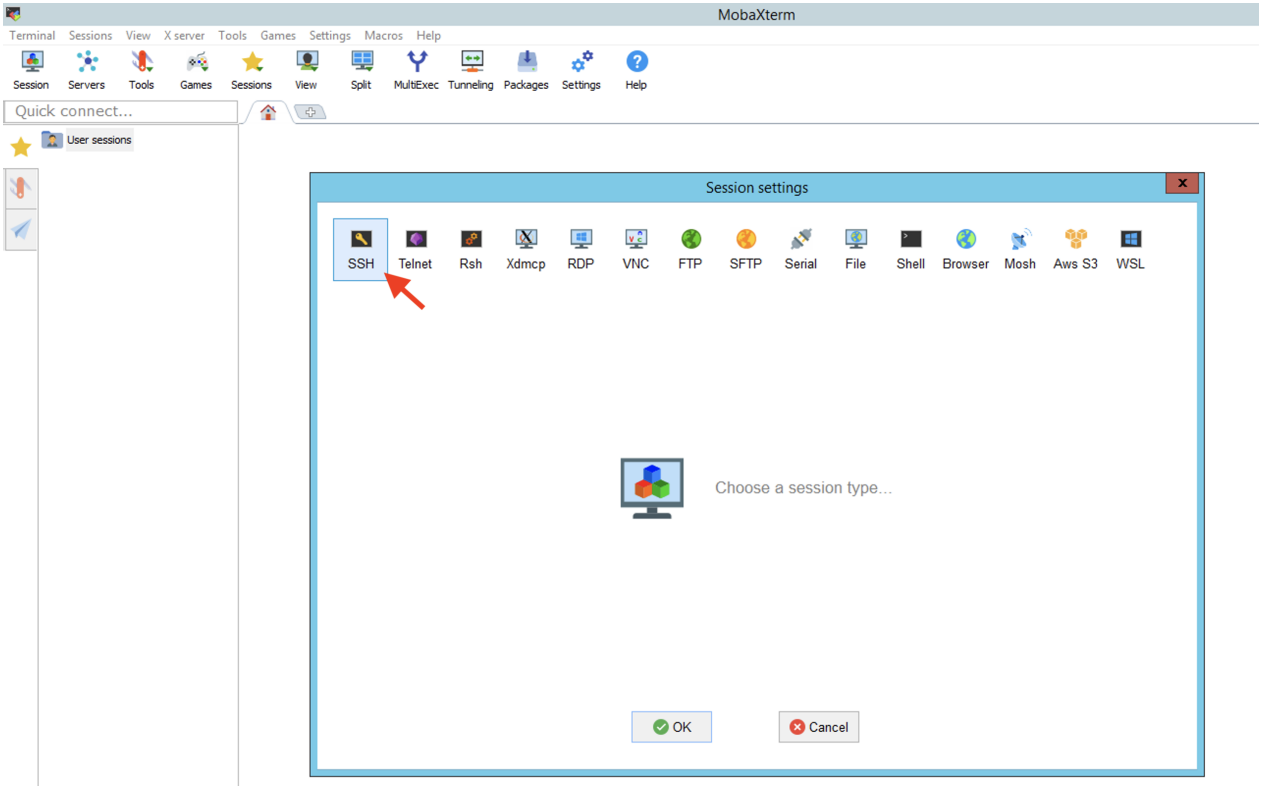


Figure 15

In the basic setting box, enter the Remote host (biowulf.nih.gov), followed by the username (remember to check the box Specify username), and again, stay on Port 22 (Figure 16).

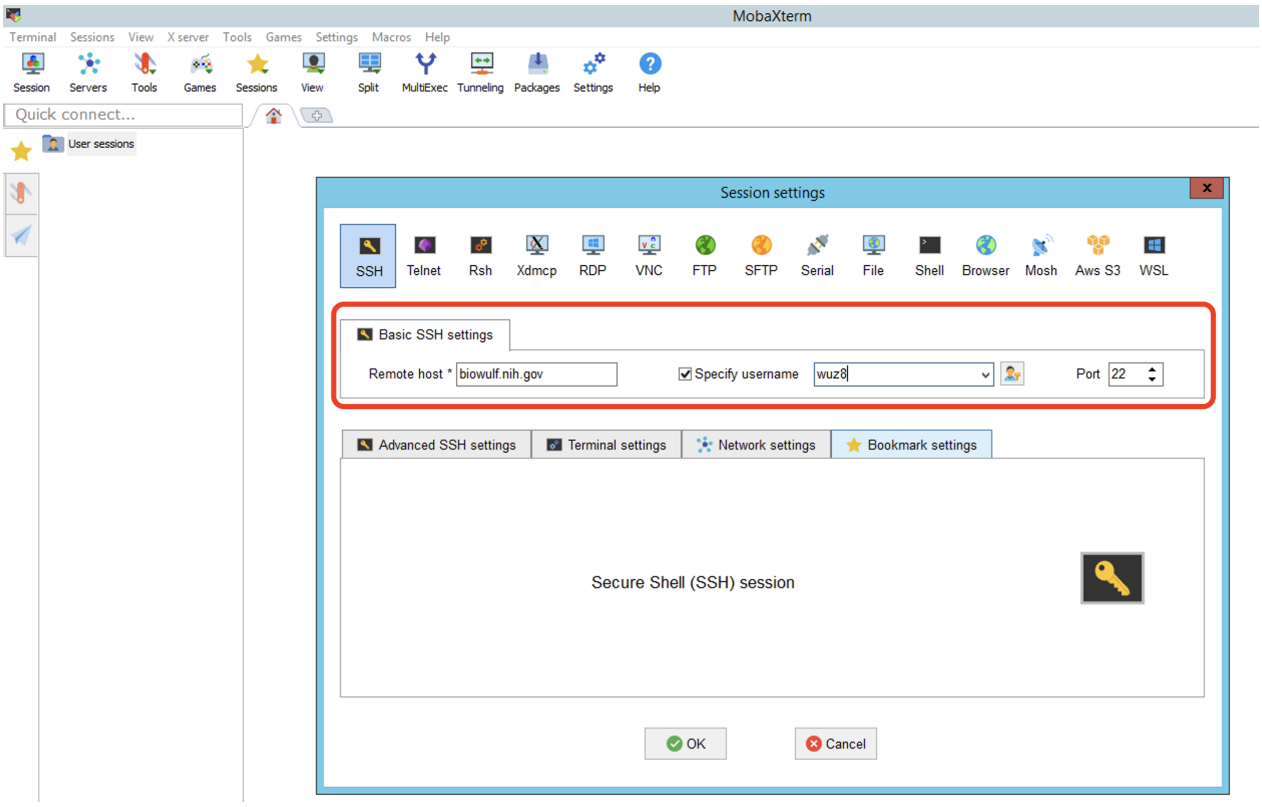


Figure 16

If this is the first time conecting to Biowulf, accept the certificate shown in Figure 17 and we will be taken to our Biowulf terminal prompt.

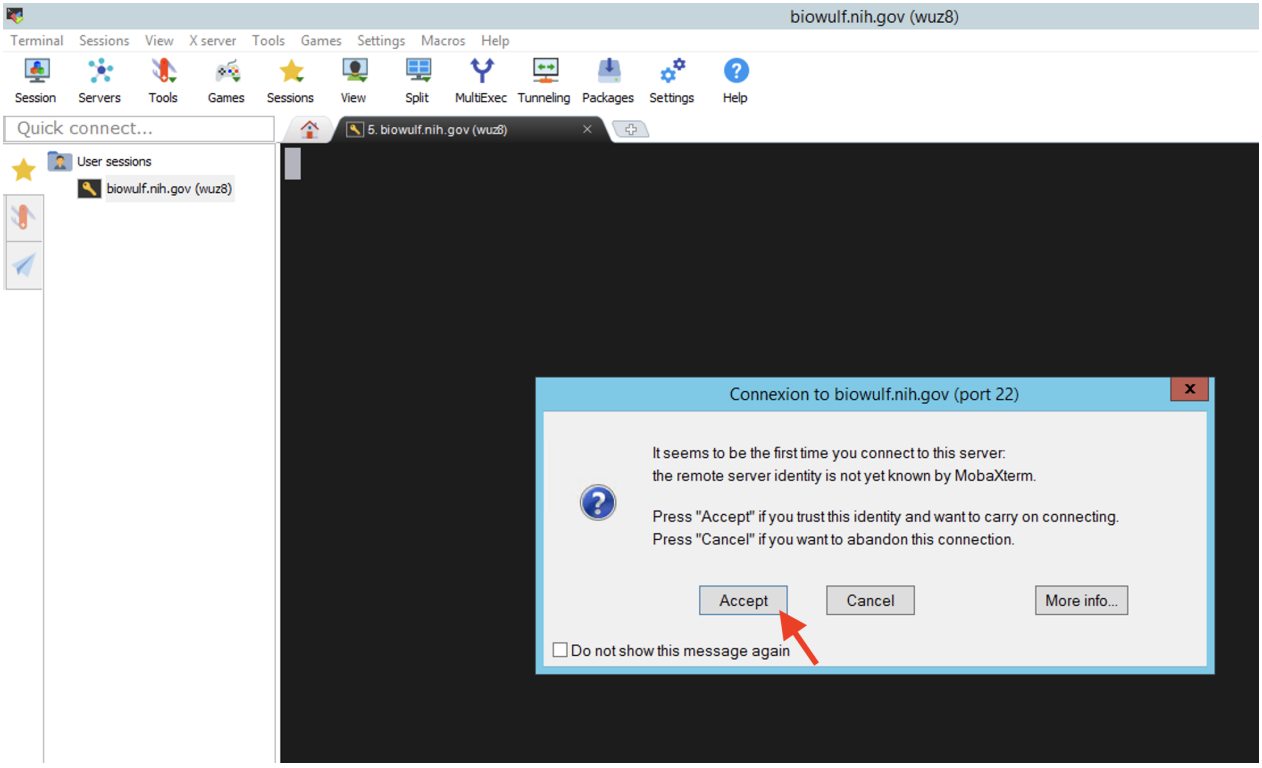
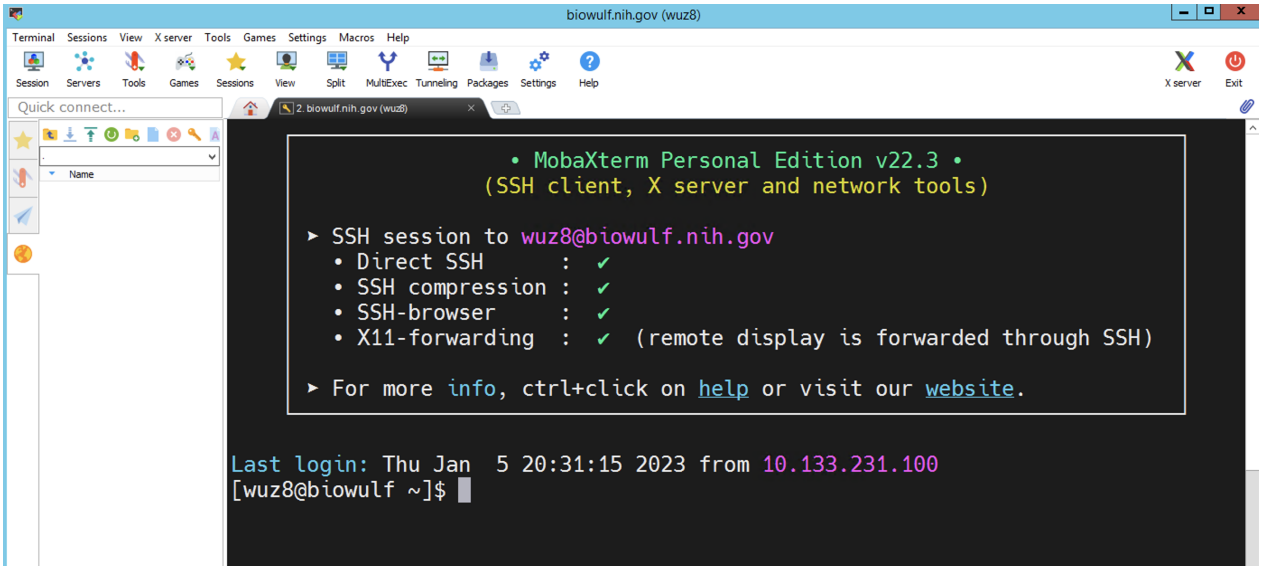


Figure 17



Mobaxterm - Biowulf connection successful

At the local terminal, we can use the `scp` command to transfer data from Biowulf to local and the other way around (Figure 18). See Figure 9 for opening a new local terminal.

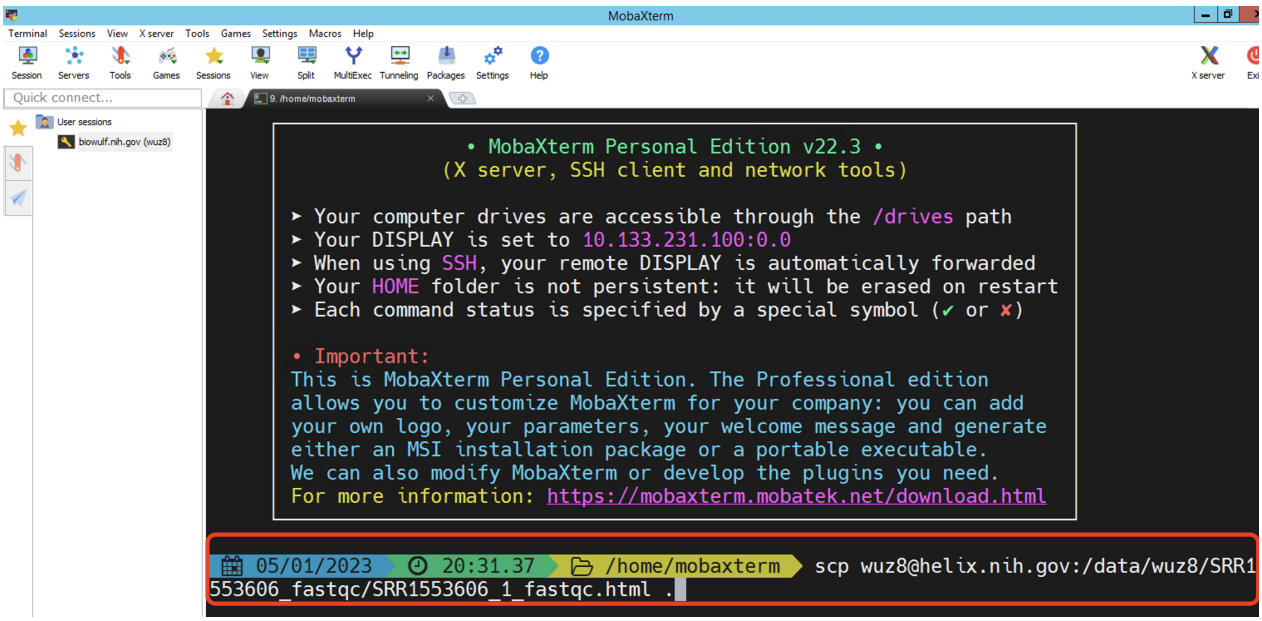


Figure 18

When clicking on the Session tab, we can also request a sftp session to help with data transfer (Figure 19).

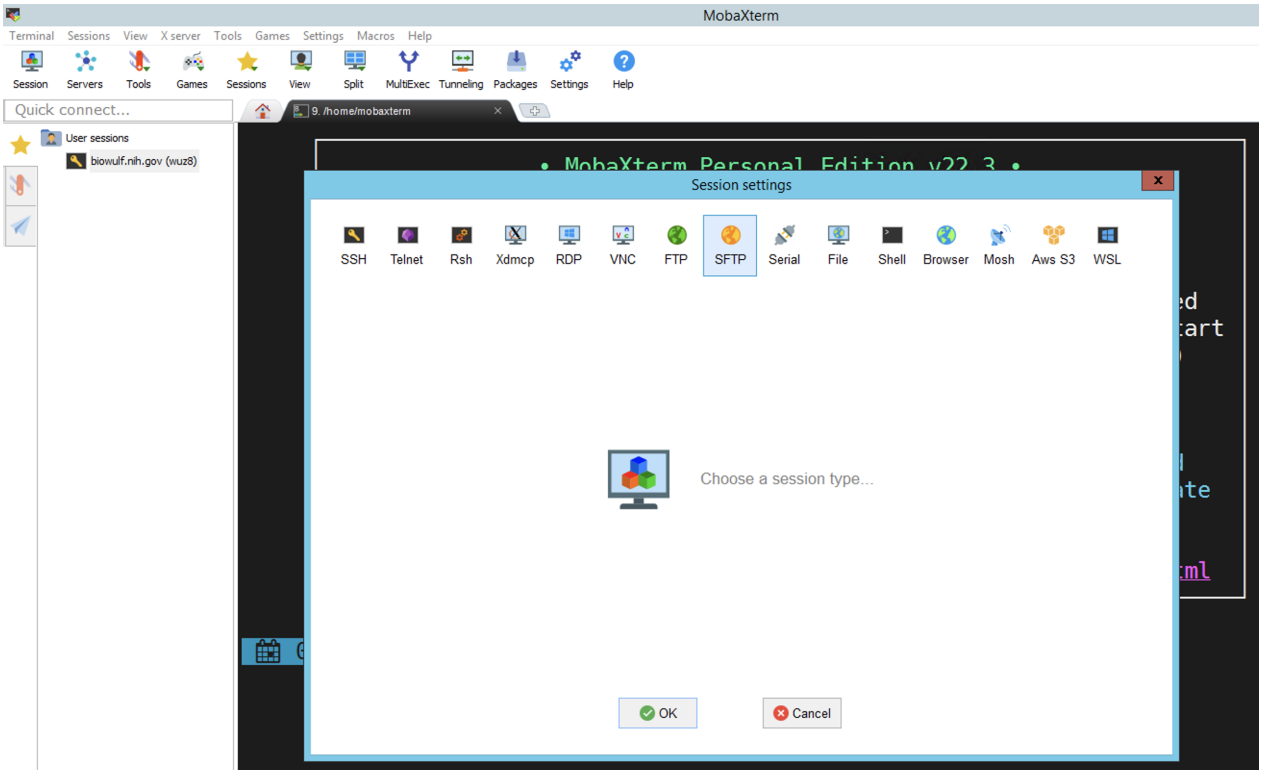


Figure 19

Again, provide the Remote host (helix.nih.gov for file transfer), followed by username, and remember to stay on Port 22 (Figure 20). Hit ok after the information has been entered.

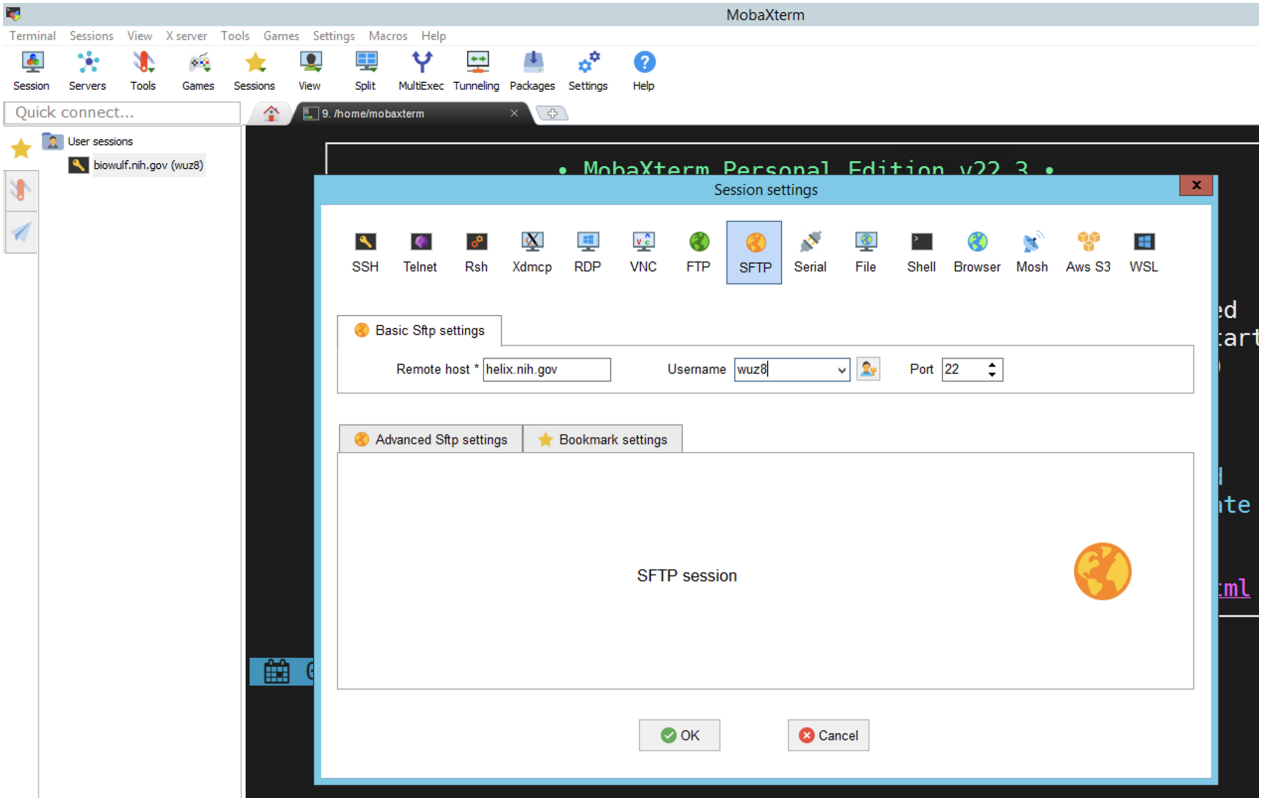


Figure 20

We will see the message to accept the certificate because its our first time logging in (go ahead and hit Accept) (Figure 21)

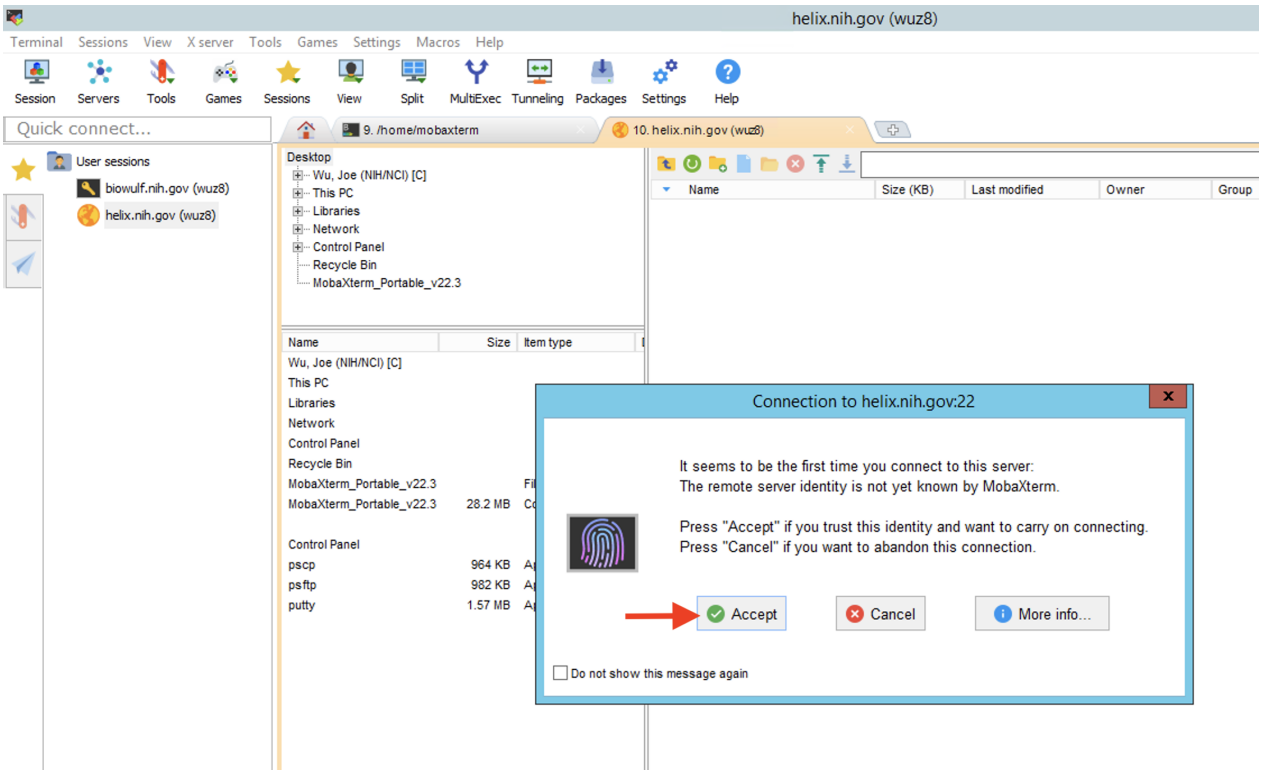


Figure 21

This will take us to an interface where on the right side we can navigate the directories in our Helix/Biowulf account and on the left, we have our local directories and files. We can then drag and drop files from local to Biowulf or from Biowulf to local (Figure 22).

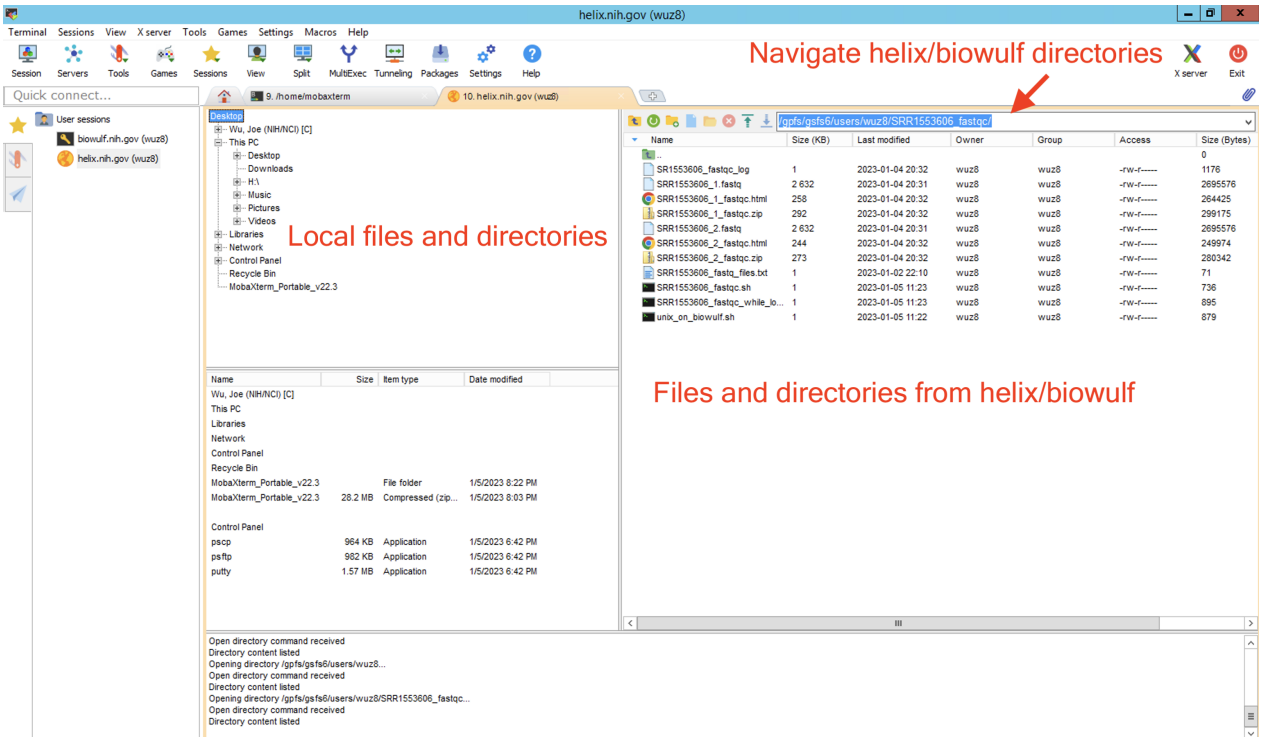


Figure 22

Interfacing with Biowulf using Fugu

Fugu is an open source and graphical based application for Mac users that can be used for data transfer between local and high performance compute systems such as Biowulf. To obtain Fugu, refer to the instruction from the Biowulf website for GUI file transfer applications *(https://hpc.nih.gov/docs/transfer.html#GUI)*.

> "Fugu is a graphical frontend to the commandline Secure File Transfer application (SFTP). SFTP is similar to FTP, but unlike FTP, the entire session is encrypted, meaning no passwords are sent in cleartext form, and is thus much less vulnerable to third-party interception. Fugu allows you to take advantage of SFTP's security without having to sacrifice the ease of use found in a GUI. Fugu also includes support for SCP file transfers, and the ability to create secure tunnels via SSH.

- Download Fugu from the U. Mich. Fugu website.
- For OSX 10.5 and above, download from cnet.com.
- Doubleclick on the downloaded Fugu_xxxx.dmg file to open. A small window with the Fugu icon will appear" -- Biowulf GUI file transfer applications *(https://hpc.nih.gov/docs/transfer.html#GUI)*.

Upon opening Fugu, we will see two panels. One allows us to navigate our local directories and files while the other allows us to connect to a remote host (ie. Helix/Biowulf) (Figure 1). Following Figure 1, do the following to connect to Helix

- Enter helix.nih.gov in the box that says Connect to
- Enter Helix/Biowulf username (mine is wuz8 so that is what we see)
- Make sure that Port is set to 22
- Specify the directory in Helix/Biowulf that we like to goto (ie. /data/wuz8, which is my data directory)
- Hit connect when done entering credentials

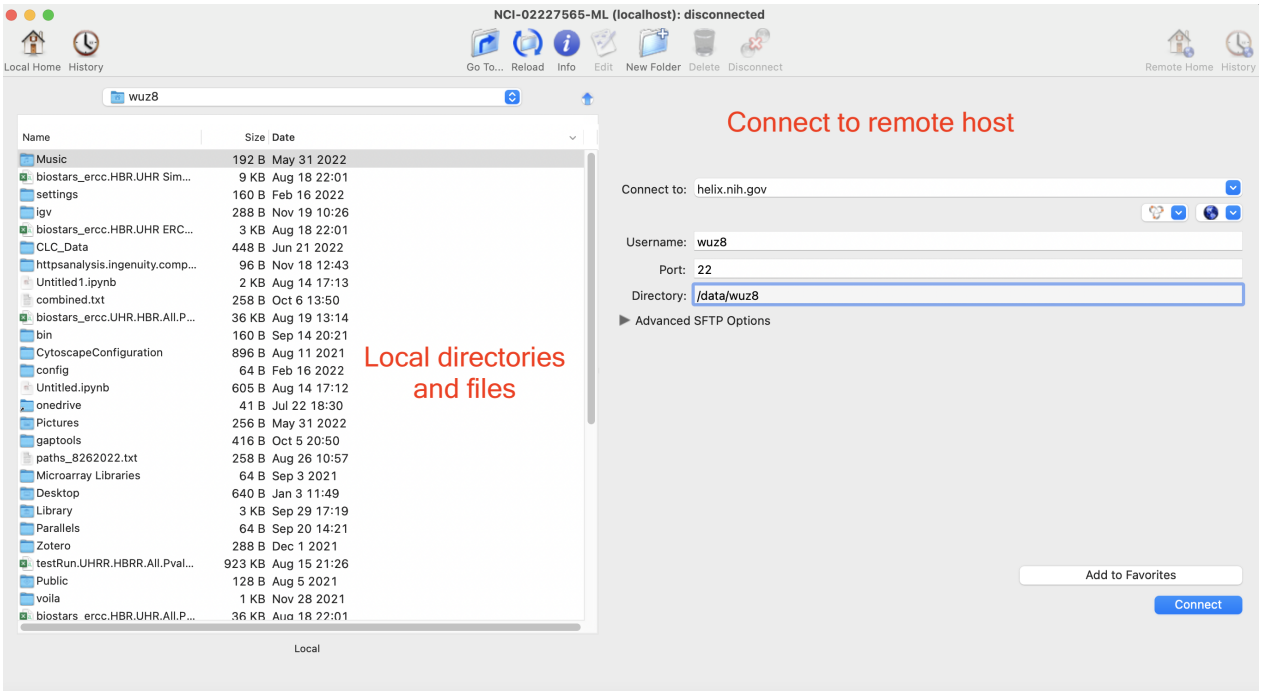


Figure 1

At the next screen, enter the password used to log into Helix/Biowulf and click the Authenticate button (Figure 2).

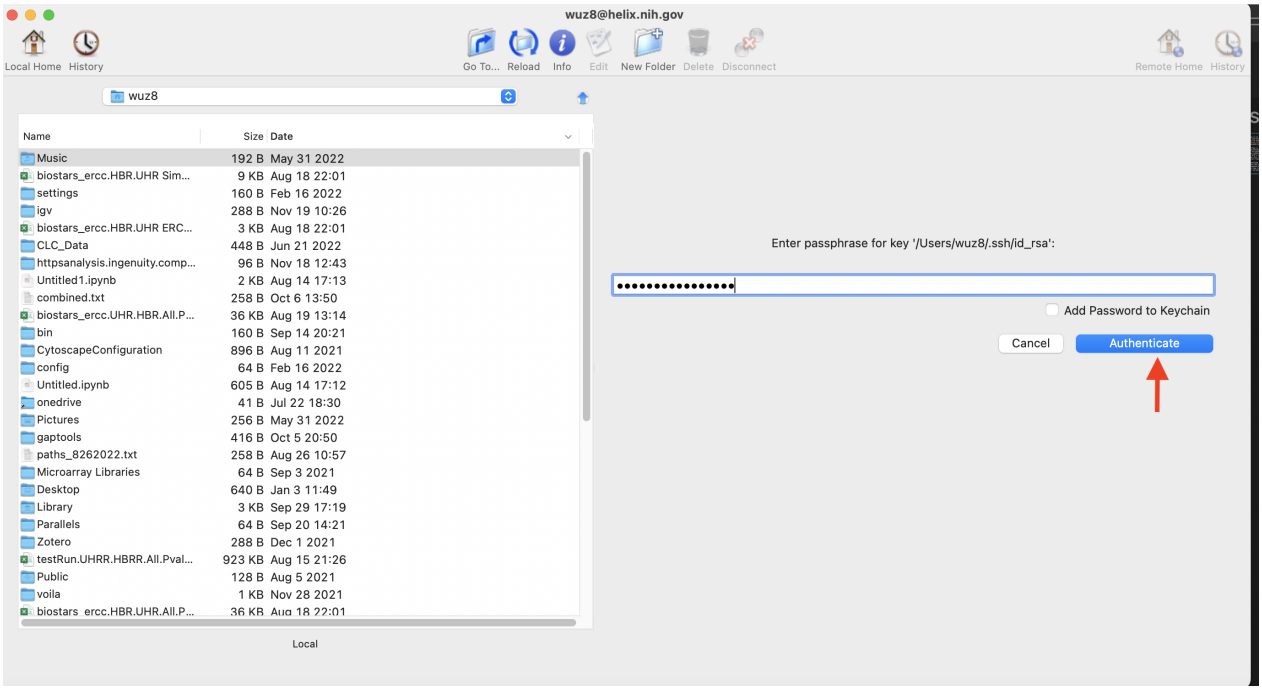


Figure 2

Once signed in to Helix, we will see our Helix/Biowulf directories and files in one panel and our local directories and files in another panel. From here we can select and then drag and drop either from Helix/Biowulf to local or from local to Helix/Biowulf (Figure 3).

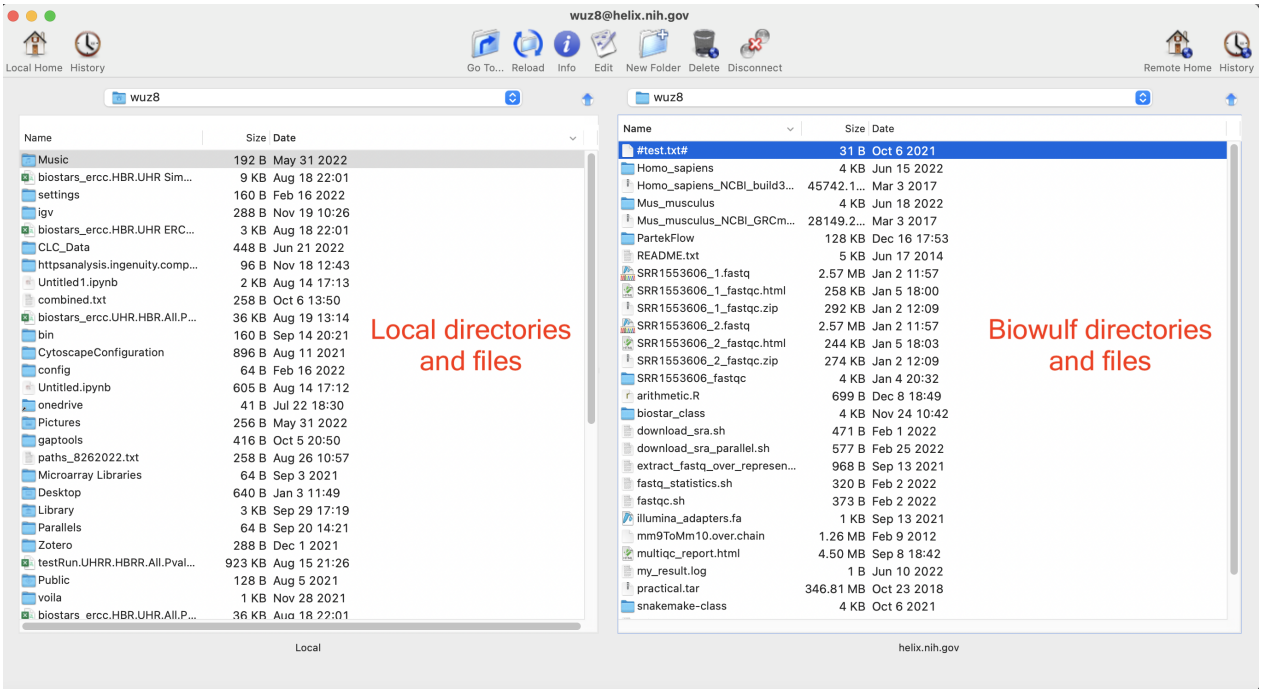

wuz8@helix.nih.gov
Local Home
History
Go To...
Reload
Info
Edit
New Folder
Delete
Disconnect
Remote Home
History
wuz8
wuz8
Name
Size
Date
Music 192 B May 31 2022
biostars_ercc.HBR.UHR Sim... 9 KB Aug 18 22:01
settings 160 B Feb 16 2022
igv 288 B Nov 19 10:26
biostars_ercc.HBR.UHR ERC... 3 KB Aug 18 22:01
CLC_Data 448 B Jun 21 2022
httpsanalysis.ingenuity.comp... 96 B Nov 18 12:43
Untitled1.ipynb 2 KB Aug 14 17:13
combined.txt 258 B Oct 6 13:50
biostars_ercc.UHR.HBR.All.P... 36 KB Aug 19 13:14
bin 160 B Sep 14 20:21
CytoscapeConfiguration 896 B Aug 11 2021
config 64 B Feb 16 2022
Untitled.ipynb 605 B Aug 14 17:12
onedrive 41 B Jul 22 18:30
Pictures 256 B May 31 2022
gaptools 416 B Oct 5 20:50
paths_8262022.txt 258 B Aug 26 10:57
Microarray Libraries 64 B Sep 3 2021
Desktop 640 B Jan 3 11:49
Library 3 KB Sep 29 17:19
Parallels 64 B Sep 20 14:21
Zotero 288 B Dec 1 2021
testRun.UHRR.HBRR.All.Pval... 923 KB Aug 15 21:26
Public 128 B Aug 5 2021
voila 1 KB Nov 28 2021
biostars_ercc.HBR.UHR.All.P... 36 KB Aug 18 22:01
Local directories and files
Local
Name
Size
Date
#test.txt# 31 B Oct 6 2021
Homo_sapiens 4 KB Jun 15 2022
Homo_sapiens_NCBI_build3... 45742.1... Mar 3 2017
Mus_musculus 4 KB Jun 18 2022
Mus_musculus_NCBI_GRCm... 28149.2... Mar 3 2017
PartekFlow 128 KB Dec 16 17:53
README.txt 5 KB Jun 17 2014
SRR1553606_1.fastq 2.57 MB Jan 2 11:57
SRR1553606_1_fastqc.html 258 KB Jan 5 18:00
SRR1553606_1_fastqc.zip 292 KB Jan 2 12:09
SRR1553606_2.fastq 2.57 MB Jan 2 11:57
SRR1553606_2_fastqc.html 244 KB Jan 5 18:03
SRR1553606_2_fastqc.zip 274 KB Jan 2 12:09
SRR1553606_fastqc 4 KB Jan 4 20:32
arithmetic.R 699 B Dec 8 18:49
biostar_class 4 KB Nov 24 10:42
download_sra.sh 471 B Feb 1 2022
download_sra_parallel.sh 577 B Feb 25 2022
extract_fastq_over_represen... 968 B Sep 13 2021
fastq_statistics.sh 320 B Feb 2 2022
fastqc.sh 373 B Feb 2 2022
illumina_adapters.fa 1 KB Sep 13 2021
mm9ToMm10.over.chain 1.26 MB Feb 9 2012
multiqc_report.html 4.50 MB Sep 8 18:42
my_result.log 1 B Jun 10 2022
practical.tar 346.81 MB Oct 23 2018
snakemake-class 4 KB Oct 6 2021
Biowulf directories and files
helix.nih.gov

Figure 3

Self learning resources

Introduction to Unix on Biowulf 2023: Self learning resources

Biowulf training and learning resources

For a list of Biowulf online classes, see https://hpc.nih.gov/training/intro_biowulf/ *(https://hpc.nih.gov/training/intro_biowulf/)*. These online classes are pre-recorded and are accompanied by exercise questions. These online classes cover topics that include Biowulf basics, swarm, and submission of batch jobs.

Biowulf also offers monthly Zoom consultations *(https://hpc.nih.gov/training/#upcoming)*.

You can always refer to the Biowulf website *(https://hpc.nih.gov/systems/)* as a good reference.

Dataquest

Below are two Unix courses offered by Dataquest. Learners are able to use an browser-integrated Unix terminal to gain hands-on experience in the two Dataquest classes below. You will need a license to access Dataquest courses. Please see https://btep.ccr.cancer.gov/licenses/ *(https://btep.ccr.cancer.gov/licenses/)* for instructions on obtaining a license.

Command Line for Data Science *(https://www.dataquest.io/course/command-line-elements/)*

Intermediate Command Line for Data Science *(https://www.dataquest.io/course/command-line-intermediate/)*

Useful Unix commands for Bioinformatics

See Stephen Tuner's Bioinformatics one-liners page *(https://github.com/stephenturner/oneliners)* for commands that can help your data wrangling tasks that are often needed when conducting bioinformatics analysis.