CompBio HandBook

CompBio Lab

2023-08-07

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

```
1 + 1
```

```
[1] 2
```

1 Introduction

This is a book created from markdown and executable code.

See Knuth (1984) for additional discussion of literate programming.

```
1 + 1
```

```
[1] 2
```

Part I

Sockeye

2 Sockeye

Sockeye is a High Performance Computing (HPC) platform, use it to run large computational analysis or works here. Using this platform requires some knowledge to Linux system, also setup in your local computer. See UBC Advanced Research Computing (2019) for additional information of the platform.

In the following sections, for simplicity, we will differentiate these platforms by the following:

- Your PC/Laptop = **Local**
- Sockeye = **Remote**

Terminologies

keyword: as

- Node: Computer
- Job: Complex computation task
- Walltime: Maximum time allowed to run a job (give $x + 1$ hours for safe)
- Resources: Memory/Time/Nodes/Cores use to run a job
- Batch job: Usage for non-interactive scripts scheduled to be run
- Interactive job: Usage for testing scripts that requires some amount of resources
- Directory: folder
- Access: Read or write permission from user
- I/O: Input and Output
- Modules: `built-in` softwares, need to manual load for your use
- GUI: Graphical User Interface , any form of graphical "viewable" softwares
- Binary: Executable programs in the terminal, usually can be found by `which BIN_NAME`, i.e. `which ls`, and output is the path of the program.
- CLI: Command Line Interface, access from terminal to run binaries and any types of unix-style commands.
- "X" Space
  - Home: your personal directory (private)
  - Scratch: directory to carry most computations
  - Project: directory to store large files of any kind

Important

Sockeye is a bit special in terms of write and read access from different nodes (more on this later). And, internet connection is not guaranteed everywhere.

2.1 Login

To login into the remote server of **Sockeye** for any platform of OS, use the following code as template and prompt it in your terminal:

```
ssh cwl@sockeye.arc.ubc.ca
```

Replace `cwl` with your actual campus-wide-login username

Tip

Advanced usage is to configurate your `ssh` locally and setup a key to login into Sockeye without prompting password (2FA is still required). This complements with VSCode quite well

2.1.1 Setup SSH configuration for Sockeye (Optional)

Follow these steps depending on your OS (Windows/Linux/MacOs) to setup a proper SSH configuration for Sockeye server. Note, you could apply this same process to any other remote server of your choice:

1. Generate a new pair of ssh keys in your local

   ```
   ssh-keygen -t rsa
   ```

   You could accept the default setting of where the key is stored, by default it is in your home directory of `~`.

   Note

   The program is also going to ask you if add a passphrase for more security, it is completely optional and up to your preference. This passphrase serves like another 2FA.

2. Copy the **public** key from the generated pair to the remote

   For Windows Powershell:

   ```
   type $env:USERPROFILE\.ssh\id_rsa.pub | ssh cwl@sockeye.arc.ubc.ca "cat >> .ssh/autho
   ```

   For Mac:

   ```
   cat ~/.ssh/id_rsa.pub | ssh cwl@sockeye.arc.ubc.ca 'cat >> /home/user/.ssh/authorized
   ```

   For Linux:

   ```
   ssh-copy-id -i ~/.ssh/id_rsa.pub cwl@sockeye.arc.ubc.ca
   ```

   Caution

   Remember to change the **CWL** argument here. And, careful with accidentally copying your private `id_*` key without the `.pub` extension to the server of your choice.

3. Add `Host`, `HostName`, `User` into a file called `~/.ssh/config` in your local.

   The general flow is the following:

   ```
   Host <shorthand_or_alias_of_your_choice>
     HostName  <server_host_domain>
     User  <username_on_server>
   ```

   For sockeye, you could use the following snippet, remember to change the **cwl** argument

   ```
   echo "Host  arc
     HostName sockeye.arc.ubc.ca
     User  cwl" > ~/.ssh/config
   ```

   > `~/.ssh/config` is the preffered place to safe ssh related configurations in most OS.

After completing this setup, you dont have to prompt for password anymore when ssh into a server, however the 2FA is still required, any passphrase if any. In the case of Sockeye, the login simplifies to:

```
ssh arc # provided you followed previous steps
# or any other server you setup
```

```
ssh <Host> # after providing Host, HostName, and User of a server in the config file
```

2.2 Job Submission

3 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

References

Knuth, Donald E. 1984. "Literate Programming." *Comput. J.* 27 (2): 97–111. https://doi.org/10.1093/comjnl/27.2.97.

UBC Advanced Research Computing. 2019. "UBC ARC Sockeye." UBC Advanced Research Computing. https://doi.org/10.14288/SOCKEYE.