

Spark Environment Configuration in SoC Compute Cluster

SoC Compute Cluster and Slurm

What are the SoC Compute Cluster and Slurm?

SoC Compute Cluster is a high-performance Linux server cluster designed to support high-performance and/or distributed computing applications. This is available to all SoC users and users from other faculties that are taking modules in SoC. You can check [Compute Cluster Guide](#) for more information.

Slurm, the Slurm Workload Manager is the system to manage all jobs in this cluster. You can check [Slurm Cluster Information](#) for more information.

Here we will focus on how to log in Compute Cluster, configure an environment for Spark and use Slurm to submit your jobs.

Log in

First of all, you must have an [SoC account](#) and enable "SoC Compute Cluster" in MySoC to use service. When you visit MySoC, go to Home -> Service, you can see a page like this

The screenshot shows the 'My SoC Services' section of the MySoC interface. On the left, there's a sidebar with links: 'mySoC-apps' (selected), 'MySoC Account', 'MySoC Email Home', 'MySoC Resources', and 'Logout'. The main area has a blue header bar with 'My SoC Services'. Below it, there's a table-like structure with two columns. The first column contains 'MySoC Account', 'MySoC Email Home', 'MySoC Resources', and 'Logout'. The second column contains '+ HOME > Services' and a detailed description: 'The SoC Compute Cluster is a Unix based compute cluster on which you can run your compute intensive jobs. You can also use this to do parallel computing experiments.' At the bottom right of this section is a blue 'Disable' button. A note below the table says 'This service is currently enabled'.

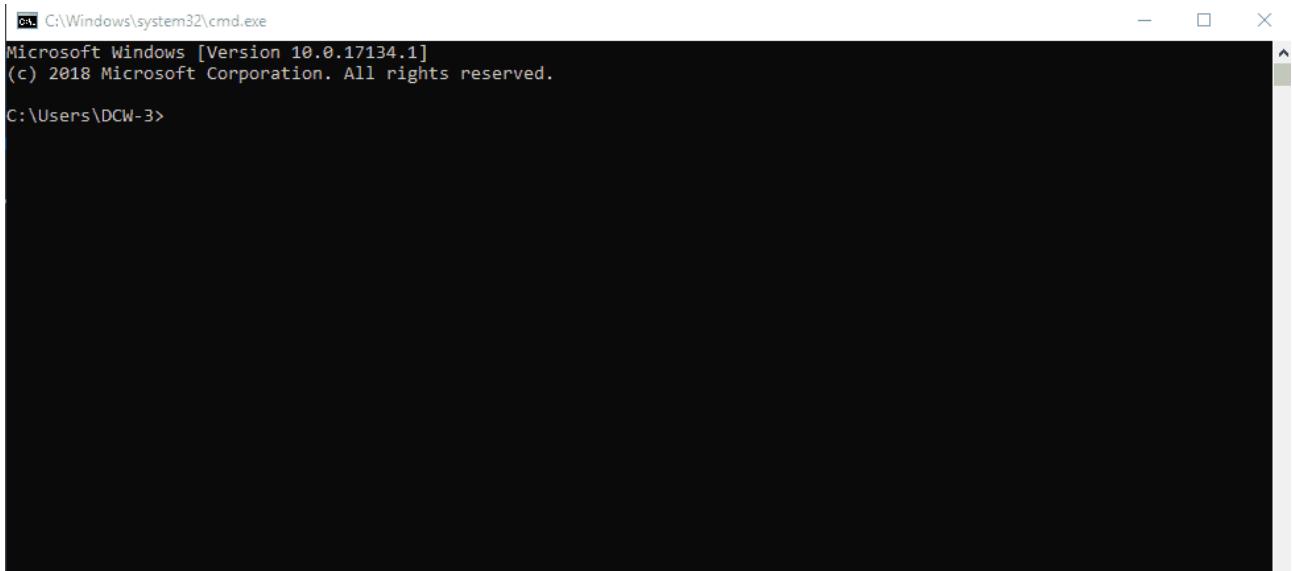
After that, we can try to log in.

The cluster nodes only accepts [SSH](#) connections from within SoC. If you are not connecting from inside SOC, then you must use SoC VPN first before logging in to the cluster.

Open your Command Line (CLI)

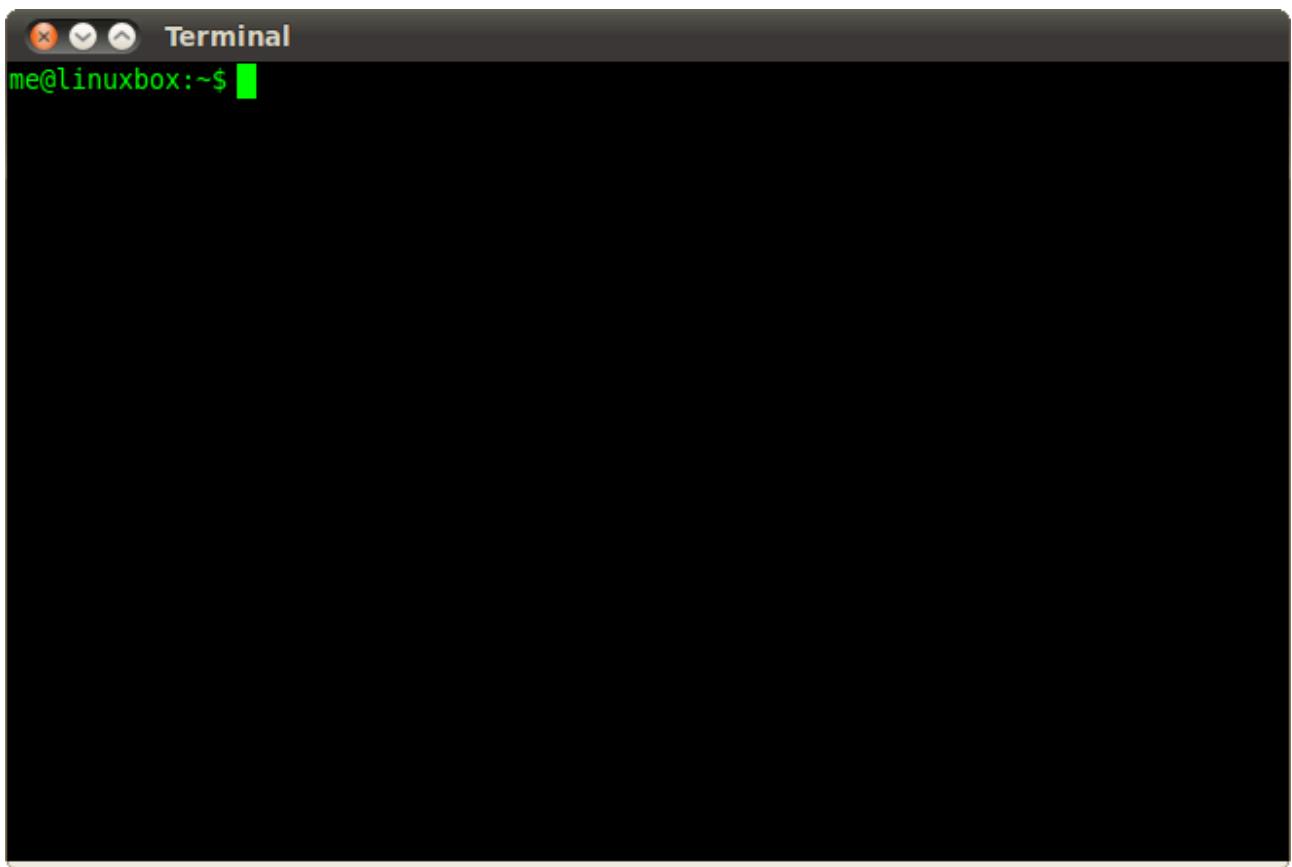
Command line is a text interface to your computer. It is also called the [shell](#), [terminal](#), [console](#), [prompt](#) or some other names.

When you opened the command line, you will see something like this



A screenshot of a Windows Command Prompt window titled "cmd". The window shows the system information: "Microsoft Windows [Version 10.0.17134.1]" and "(c) 2018 Microsoft Corporation. All rights reserved." followed by the prompt "C:\Users\DCW-3>". The window has standard minimize, maximize, and close buttons at the top right.

if you are using a Unix like system e.g., macOS or Unbutu,



A screenshot of a Linux terminal window titled "Terminal". The window shows the user's login information: "me@linuxbox:~\$". The window has standard minimize, maximize, and close buttons at the top left.

Anyway, you basically get yourself a blank window where you can input some commands.

Log in with SSH

You can check the [SSH](#) Guide for more about SSH. But right now, let's log in.
Input this command into the command line

ssh user1234@xlog0

Please replace **user1234** with your own account. If you are not sure about your account, please check this [SoC account](#), or simply try every account you have that is given NUS.

After log in, you will see a prompt like this:

user1234@xlog0:~\$

This means you, the **user1234**, are logged into the node **xlog0** and is currently in your home directory (denoted by **~**). You can log into the node **xlog0**, **xlog1**, **xlog2** or **xlog3**. These nodes are just for log in, not computation. Then we will configure the environment for Spark here and finally submit our jobs to Slurm here.

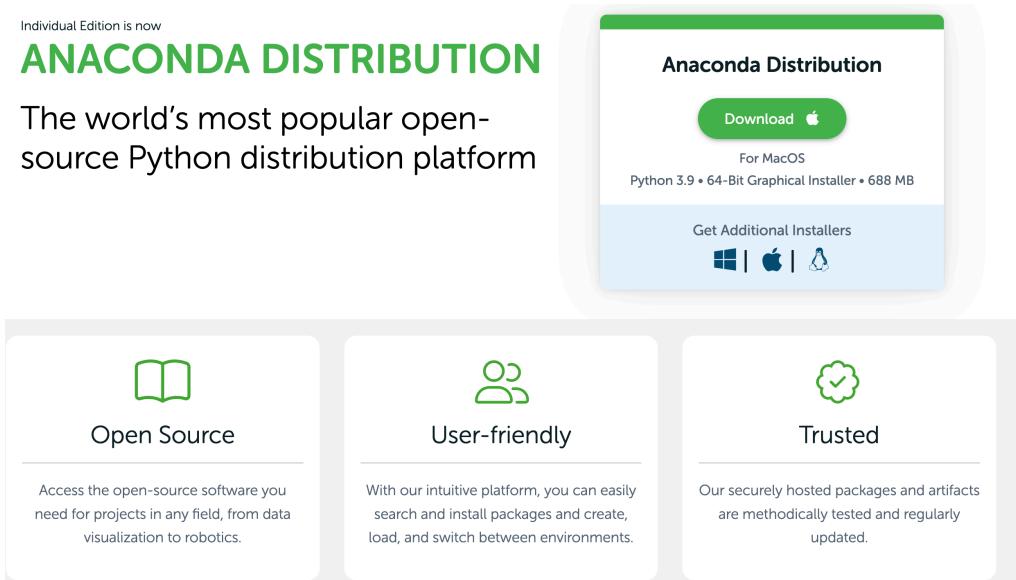
Since Soc Compute Cluster uses a Network File System, all files under your home directory are shared among different servers, which means you only need to configure the environment on one server and it is available in all servers. But it also means that the I/O operation could be very slow if you are dealing with some very large files.

Environment Configuration for Spark

Install Anaconda

Download Anaconda individual edition at <https://anaconda.com/>, and install. Here is the official instruction about how to do that, [installing on Linux](#).

1. Visit the Anaconda downloads page



The screenshot shows the Anaconda Distribution website. At the top, it says "Individual Edition is now" and "ANACONDA DISTRIBUTION". Below that, it says "The world's most popular open-source Python distribution platform". To the right, there is a call-to-action button "Download" for Mac OS, with the note "Python 3.9 • 64-Bit Graphical Installer • 688 MB". Below the download button, there is a link "Get Additional Installers" followed by icons for Windows, Mac, and Linux. At the bottom, there are three cards: "Open Source" (with a book icon), "User-friendly" (with a person icon), and "Trusted" (with a checkmark icon). Each card has a brief description below it.

Individual Edition is now
ANACONDA DISTRIBUTION

The world's most popular open-source Python distribution platform

Anaconda Distribution

Download

For Mac OS
Python 3.9 • 64-Bit Graphical Installer • 688 MB

Get Additional Installers

Open Source

Access the open-source software you need for projects in any field, from data visualization to robotics.

User-friendly

With our intuitive platform, you can easily search and install packages and create, load, and switch between environments.

Trusted

Our securely hosted packages and artifacts are methodically tested and regularly updated.

2. Select Linux

Anaconda Installers

Windows 

Python 3.9

64-Bit Graphical Installer (621 MB)

MacOS 

Python 3.9

64-Bit Graphical Installer (688 MB)

Linux 

Python 3.9

64-Bit (x86) Installer (737 MB)

3. Copy the download link

Windows 

Python 3.9

64-Bit Graphical Installer (621 MB)

MacOS 

Python 3.9

64-Bit Graphical Installer (688 MB)

Linux 

Python 3.9

64-Bit (x86) Installer (737 MB)



4. Use wget to download

`wget https://repo.anaconda.com/archive/Anaconda3-2022.10-Linux-x86_64.sh`

5. Run the script to install

`bash Anaconda3-2022.10-Linux-x86_64.sh`

6. source the .bash-rc file to add Anaconda to your PATH. Note that .bashrc file is in your home directory

`cd ~
source .bashrc`

Install Java

1. Create an environment.

`conda create --name myenv python=<version>`

You can specify the python version here. Replace <version> with the version number you want, i.e., 3.6

```
(base) e0514929@xlog2:~/ta$ conda create --name cs5344 python=3.6
```

For example, the environment name is cs5344 here.

Then it will show a list of new packages need to be installed, just type 'y' and wait for it to finish.

```
The following NEW packages will be INSTALLED:
```

```
_libgcc_mutex      pkgs/main/linux-64::__libgcc_mutex-0.1-main
_omp_mutex         pkgs/main/linux-64::__openmp_mutex-5.1-1_gnu
ca-certificates    pkgs/main/linux-64::ca-certificates-2022.10.11-h06a4308_0
certifi             pkgs/main/linux-64::certifi-2021.5.30-py36h06a4308_0
ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.38-h1181459_1
libffi              pkgs/main/linux-64::libffi-3.3-he6710b0_2
libgcc-ng           pkgs/main/linux-64::libgcc-ng-11.2.0-h1234567_1
libgomp             pkgs/main/linux-64::libgomp-11.2.0-h1234567_1
libstdcxx-ng        pkgs/main/linux-64::libstdcxx-ng-11.2.0-h1234567_1
ncurses             pkgs/main/linux-64::ncurses-6.3-h5eee18b_3
openssl             pkgs/main/linux-64::openssl-1.1.1s-h7f8727e_0
pip                 pkgs/main/linux-64::pip-21.2.2-py36h06a4308_0
python               pkgs/main/linux-64::python-3.6.13-h12debd9_1
readline             pkgs/main/linux-64::readline-8.2-h5eee18b_0
setuptools          pkgs/main/linux-64::setuptools-58.0.4-py36h06a4308_0
sqlite              pkgs/main/linux-64::sqlite-3.40.0-h5082296_0
tk                  pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0
wheel               pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
xz                  pkgs/main/linux-64::xz-5.2.8-h5eee18b_0
zlib                pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0
```

```
Proceed ([y]/n)?
```

When finished

```
|Proceed ([y]/n)? y
```

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate cs5344
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

2. Activate Conda Environment

After log in one of xlog nodes, activate the environment you just created. For example,

```
(base) e0514929@xlog0:~$ conda activate cs5344
(cs5344) e0514929@xlog0:~$
```

(check the command to do that in [Conda Cheat Sheet](#) based on your operating system). Now you are in the cs5344 conda environment.

3. Install OpenJDK

To install OpenJDK (a version of Java), use the following command,

conda install openjdk=<version>

Again, replace <version> with the version number you want, e.g., **11**. Like this,
Just input 'y' and wait it to finish.

```
==> WARNING: A newer version of conda exists. <==  
    current version: 4.10.3  
    latest version: 22.11.1  
  
Please update conda by running  
  
$ conda update -n base -c defaults conda  
  
## Package Plan ##  
  
environment location: /home/e/e0514929/anaconda3/envs/cs5344  
  
added / updated specs:  
- openjdk  
  
The following NEW packages will be INSTALLED:  
  
dbus                  pkgs/main/linux-64::dbus-1.13.18-hb2f20db_0  
expat                pkgs/main/linux-64::expat-2.4.9-h6a678d5_0  
freetype              pkgs/main/linux-64::freetype-2.12.1-h4a9f257_0  
glib                 pkgs/main/linux-64::glib-2.69.1-h4ff587b_1  
libpng               pkgs/main/linux-64::libpng-1.6.37-hbc83047_0  
libxcb               pkgs/main/linux-64::libxcb-1.15-h7f8727e_0  
openjdk              pkgs/main/linux-64::openjdk-11.0.13-h87a67e3_0  
pcre                 pkgs/main/linux-64::pcre-8.45-h295c915_0  
  
Proceed ([y]/n)?  
  
Preparing transaction: done  
Verifying transaction: done  
Executing transaction: done  
(cs5344) e0514929@xlog0:~$ █
```

Install PySpark

The last thing we need to install is pyspark. You do not need to worry about Python, Spark or Scala. Conda will take care of them for you. The command is

conda install pyspark

Like this,

Again, input 'y' and wait. This could take some time.

When it is finished, you have a conda environment for spark named cs5344.

```

long      up 3-00:00:00      1  down amdgpu2
test      up    10:00      2  down* xgpf[2,11]
test      up    10:00     28  idle xcne[0-7],xgpf[0-1,3-10],xgpg[0-9]
(base) e0514929@xlog0:~$ squeue
      JOBID PARTITION      NAME      USER ST          TIME  NODES NODELIST(REASON)
(base) e0514929@xlog0:~$ █

Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 4.10.3
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: /home/e/e0514929/anaconda3/envs/cs5344

added / updated specs:
- pyspark

The following NEW packages will be INSTALLED:

abseil-cpp      pkgs/main/linux-64::abseil-cpp-20210324.2-h2531618_0
arrow-cpp        pkgs/main/linux-64::arrow-cpp-3.0.0-py36hb21186_4
aws-c-common    pkgs/main/linux-64::aws-c-common-0.4.57-he6710b0_1
aws-c-event-stream pkgs/main/linux-64::aws-c-event-stream-0.1.6-h2531618_5
aws-checksums   pkgs/main/linux-64::aws-checksums-0.1.9-he6710b0_0
aws-sdk-cpp     pkgs/main/linux-64::aws-sdk-cpp-1.8.185-hce553d0_0
blas            pkgs/main/linux-64::blas-1.0-mkl
boost-cpp       pkgs/main/linux-64::boost-cpp-1.73.0-h27cf23_11
brotli          pkgs/main/linux-64::brotli-1.0.9-h5eee18b_7
brotli-bin     pkgs/main/linux-64::brotli-bin-1.0.9-h5eee18b_7
bzip2           pkgs/main/linux-64::bzip2-1.0.8-h7b6447c_0
c-ares          pkgs/main/linux-64::c-ares-1.18.1-h7f8727e_0

```

Submit your job to Slurm

Let's say you have done some Spark coding and you want to run your script in Slurm. Since we do not use xlog0~2 to do computation. We need to ask Slurm to allocate us some resources to do the computation.

There are some common helpful Slurm commands.

sinfo: Show status of nodes and partitions in the Slurm cluster.

```

(base) e0514929@xlog0:~$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
standard*  up    10:00      1  down* xgpd0
standard*  up    10:00     15  mix xcnd[19-25],xgpc2,xgpd[3-4],xgph[5-9]
standard*  up    10:00      5  alloc xcnc24,xcnd[15-18]
standard*  up    10:00     72  idle amdgpu[0-1],xcna[0-6],xcnb[0-9],xcnc[0-15,17-23],xcnd[0-8,10,12-14,26-29],xgpc[0-1],xgpd2,xgpe[0-4],xgph[0-4]
medium    up    3:00:00     35  mix xcnc[28-32,34-35],xcnd[19-25],xgpc[2-4],xgpd[3-6],xgpe[6-9],xgph[5-14]
medium    up    3:00:00      9  alloc xcnc[24-27],xcnd[15-18],xgpd7
medium    up    3:00:00     49  idle amdgpu1,xcna[4-6,8-11],xcnb[5-14],xcnc[13-15,17-23,36-37],xcnd[26-44]
medium    up    3:00:00      1  down amdgpu2
long      up    3:00:00:00    25  mix xcnc[28-32,34-35],xgpc[3-4],xgpd[5-6,8-9],xgpe[10-11],xgph[10-19]
long      up    3:00:00:00      4  alloc xcnc[25-27],xgpd7
long      up    3:00:00:00    61  idle xcna[8-15],xcnb[10-14,16-19],xcnc[36-49],xcnd[30-59]
long      up    3:00:00:00      1  down amdgpu2
test      up    10:00      2  down* xgpf[2,11]
test      up    10:00     28  idle xcne[0-7],xgpf[0-1,3-10],xgpg[0-9]

```

squeue: Show the Slurm job queue.

This is the first semester that all nodes are managed by Slurm. IT service guys are also trying to find the best way to set the system. So, sometimes Slurm commands may not behave like it should. To find out more, please check [Slurm Guide](#).

To submit a job to Slurm, **srun**. This command will automatically get you a job allocation and submit your job. Remember to activate your environment. Like this,

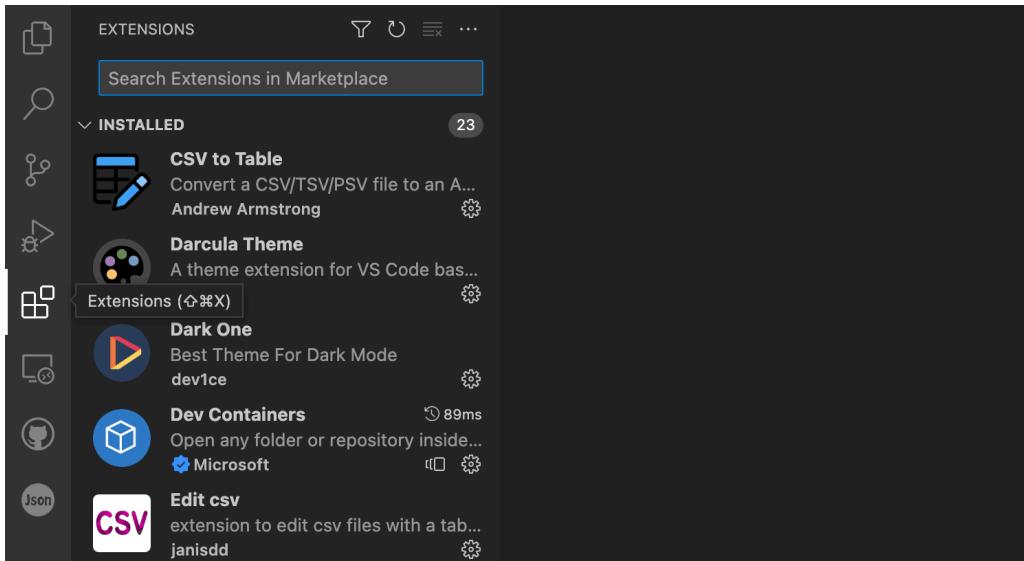
```
(base) e0514929@xlog0:~/ta/CS5344/Lab1$ conda activate cs5344
(cs5344) e0514929@xlog0:~/ta/CS5344/Lab1$ srun spark-submit wordcount.py in.txt outfile
```

Use Compute Server with an IDE

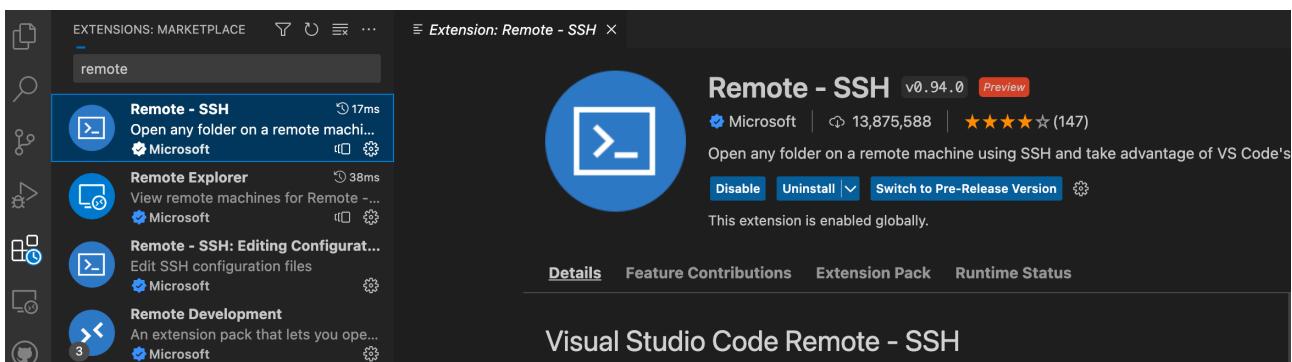
You may want to coding in an IDE with the remote servers. Most IDEs provide the function to connect to remote servers. We recommend VS Code to do that. Compared with other IDEs, it is easy and straightforward to use and it provides all kinds of extensions which could be very helpful in coding.

You can download VS Code in [here](#).

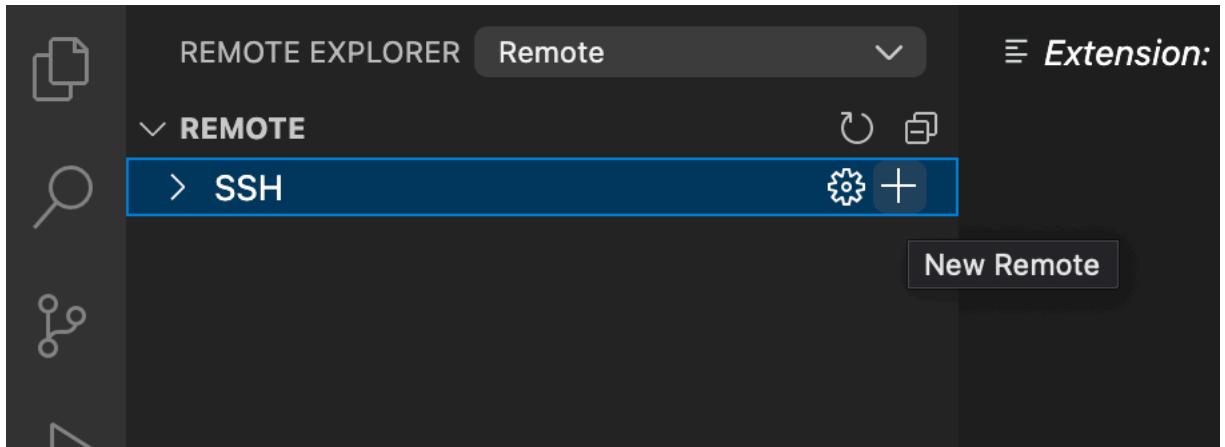
After installation, open VS Code and choose Extensions on the left panel.



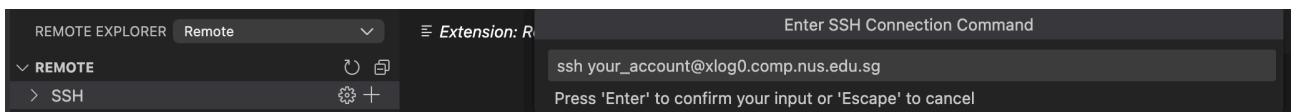
Then search 'remote' and install 'Remote - SSH'.



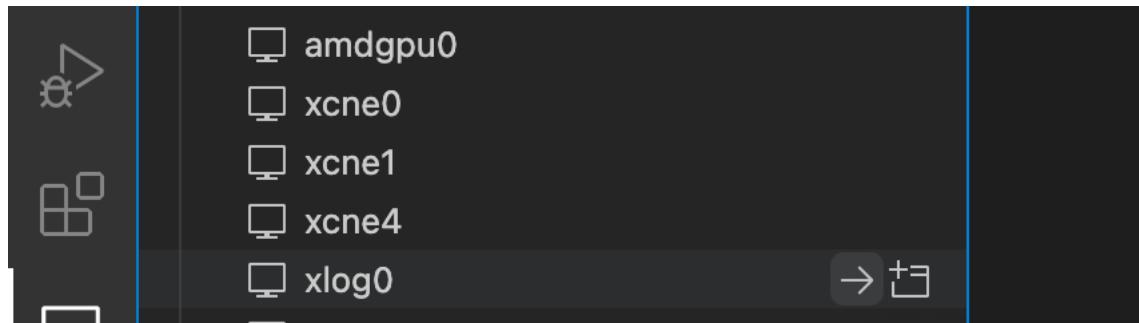
After installing the ‘Remote - SSH’, choose it on the left panel and click the adding symbol to add a ‘New Remote’.



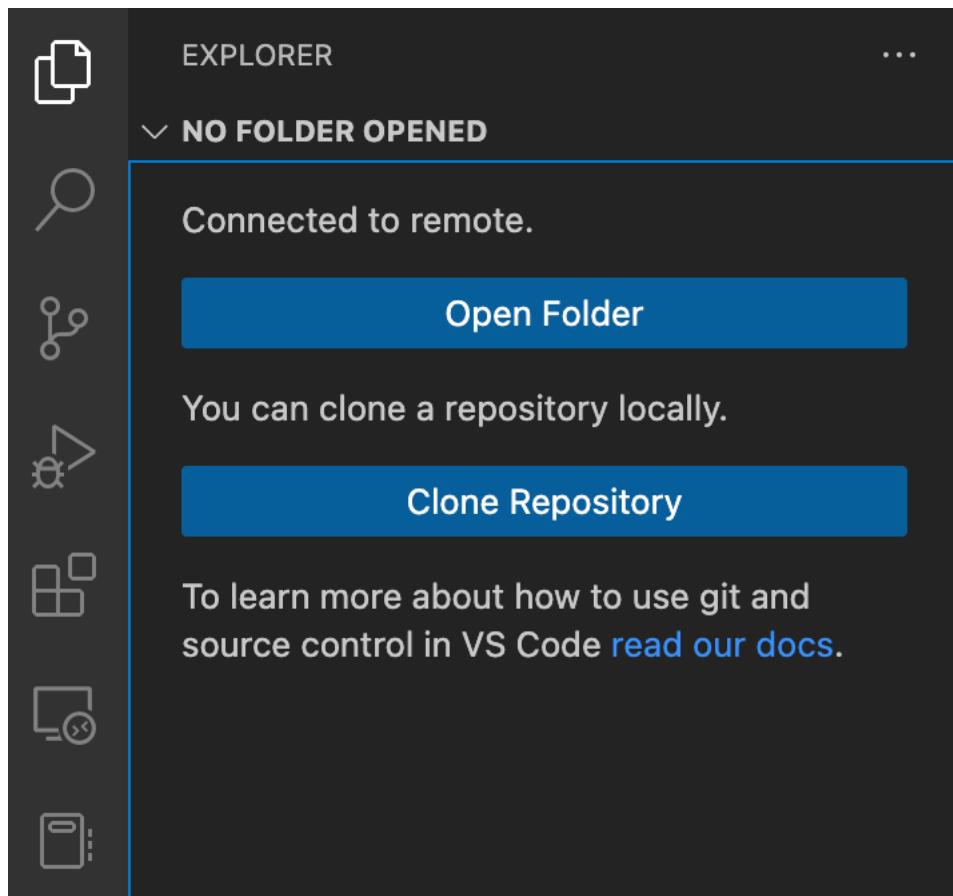
Then it will ask for a SSH command. Let’s take xlog0 for an example. Input your account and the host address of xlog0.



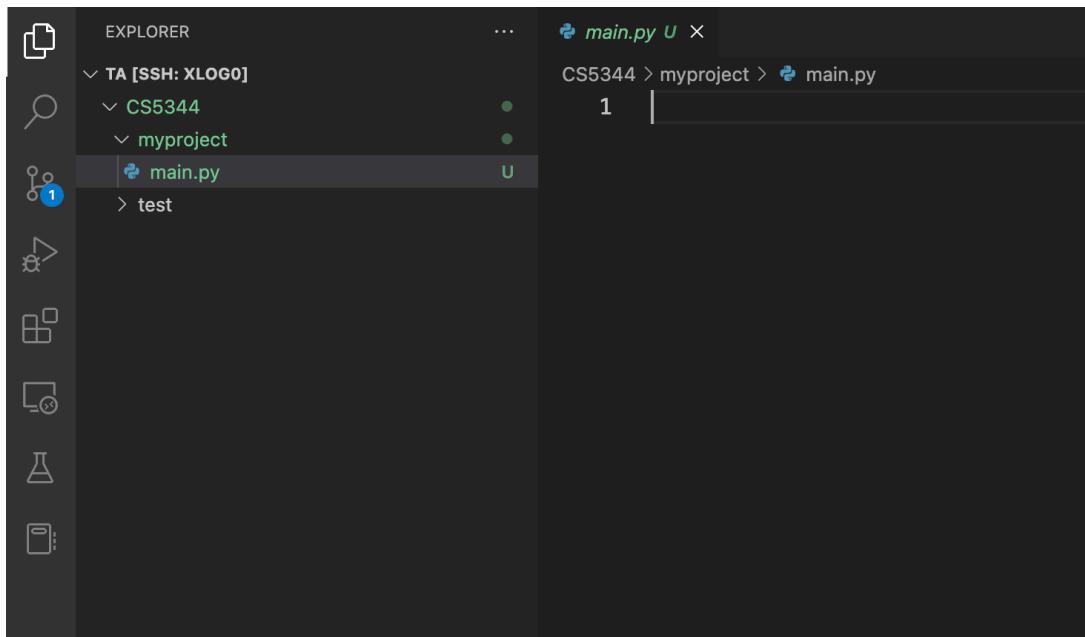
Click the arrow next to SSH, you will see xlog has been added into remote server list.



Open xlog0, then input your password. You will see



It means you have successfully connected to xlog0. Now you can open folders and edit files like being in a local server. For example,



And you can choose which python you want to use to run your code in the right corner. For example, here the python 3.6 in the conda environment cs5344 we have created is chosen.

A screenshot of a terminal window. The top portion is black. Below it is a blue header bar containing the following text from left to right: "Ln 1, Col 1", "Spaces: 4", "UTF-8", "LF", a small icon of a computer monitor with a dot on it, "Python", and "3.6.13 ('cs5344': conda)".

