Slade's Guide to the Yale CS Zoo

For many decades, the Yale Department of Computer Science has provided a computer facility for undergraduate education, known affectionately as "The Zoo." See https://zoo.cs.yale.edu/newzoo/

Note that the Zoo comprises dozens of machines, each of which has an animal name, and a corresponding web address, e.g., lion.zoo.cs.yale.edu. Many of these machines exist IRL (in real life) on the third floor of the Computer Science Building (Arthur K. Watson Hall, 51 Prospect Street, aka, AKW) as well as 17 Hillhouse, Room 111. Additional nodes are available only through virtual connections. Actually, all zoo nodes are available virtually. See below for the secret incantation.

Getting a zoo account and authentication

You need a user id and password to access the zoo. You also will need a duo authentication device, such as a cell phone. If you are enrolled in a computer science course numbered 2000 or higher, you should automatically get an account. You don't need to do anything or ask anyone. Your user id is your Yale netid and your password is your netid password.

The standard method for authenticating yourself to a computer is multi-factor authentication (See https://en.wikipedia.org/wiki/Multi-factor_authentication). You must provide two of the following:

- Something you know, like a user id or password.
- Something you have, like an ATM card or a smart phone app.
- Something you are, like a biometric physical characteristic such as your fingerprint or facial image.

For the zoo you need (1) a user id and a password, and (2) Duo authentication. See https://cybersecurity.yale.edu/mfa

Connecting to the zoo in person

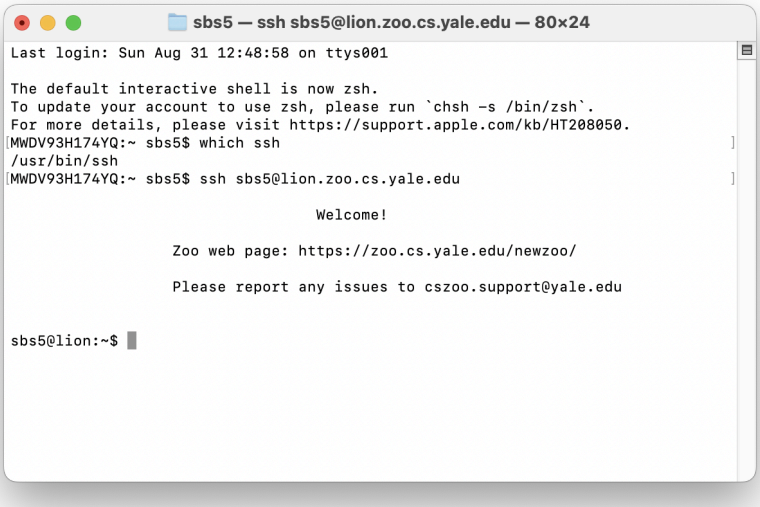
You can simply go up to a physical zoo machine in either AKW or 17 Hillhouse and login, assuming that it is neither occupied nor locked. The latter state occurs when a student has been using a machine in person and needs to be away from the machine briefly. She can then lock the machine, preventing someone else from logging in at the physical machine.

Connecting to the zoo remotely with ssh

You have probably run across this thing called the internet. Well, one of the features of the internet is the ability to login to a remote machine. You can be sitting in your bedroom and login to one of the zoo machines without getting out of your pajamas. In fact, you could be on the beach in Rio (with suitable WiFi service) and likewise login to the zoo. Note: if you are off campus and away from Yale's secure wifi, you will need to set up a Virtual Private Network. See https://en.wikipedia.org/wiki/Virtual_private_network Yale has its own. See https://yale.service-now.com/it?id=kb_article&sysparm_article=KB0000326

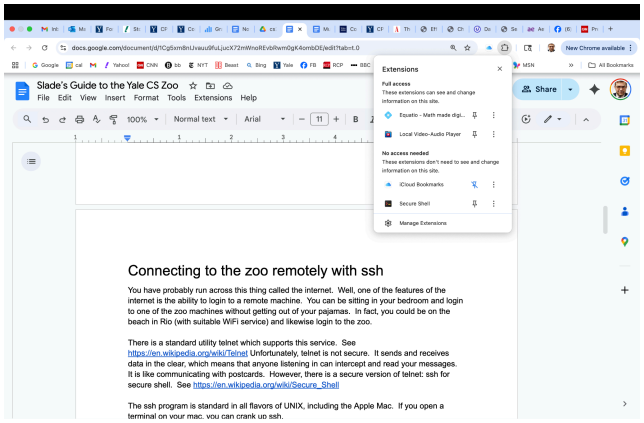
There is a standard utility **telnet** which supports this remote login service. See https://en.wikipedia.org/wiki/Telnet Unfortunately, **telnet** is not secure. It sends and receives data in the clear, which means that anyone listening in can intercept and read your messages. It is like communicating with postcards. However, there is a secure version of **telnet** which encrypts the traffic: **ssh** for secure shell. (Note "shell" is the UNIX term for the command interpreter.) See https://en.wikipedia.org/wiki/Secure_Shell

The **ssh** program is standard in all flavors of UNIX, including the Apple Mac. If you open a terminal on your mac, you can crank up **ssh**.

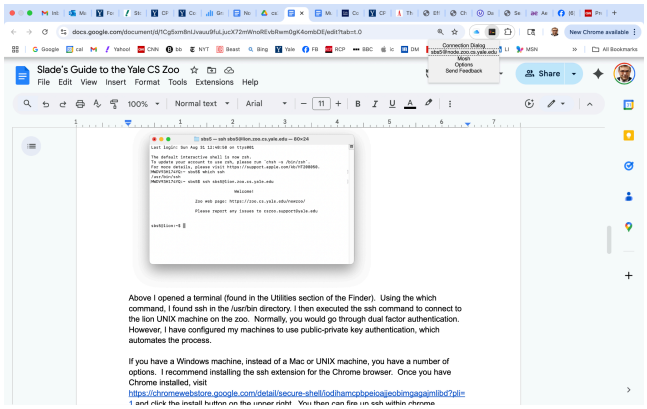


Above I opened a terminal (found in the Utilities section of the Finder). Using the **which** command, I found **ssh** in the **/usr/bin** directory. I then executed the **ssh** command to connect to the lion UNIX machine on the zoo. Normally, you would go through dual factor authentication. However, I have configured my machines to use public-private key authentication, which automates the process. See https://en.wikipedia.org/wiki/Public-key_cryptography and https://zoo.cs.yale.edu/newzoo/pubkey_guide.html

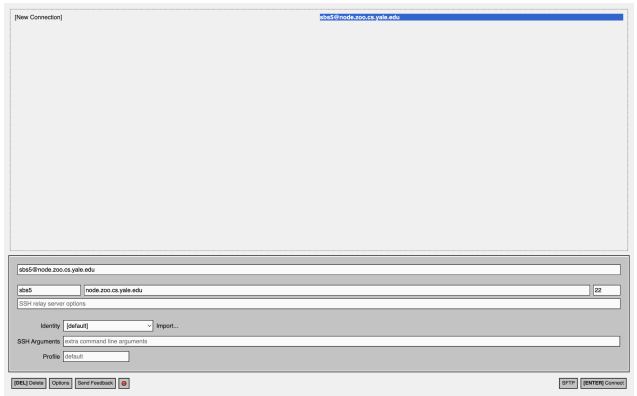
If you have a Windows machine, instead of a Mac or UNIX machine, you have a number of options. I recommend installing the **ssh** extension for the Chrome browser. Once you have Chrome installed, visit https://chromewebstore.google.com/detail/secure-shell/iodihamcpbpeioajjeobimgagajmlibd?pli=1 and click the install button on the upper right. You then can fire up **ssh** within chrome.



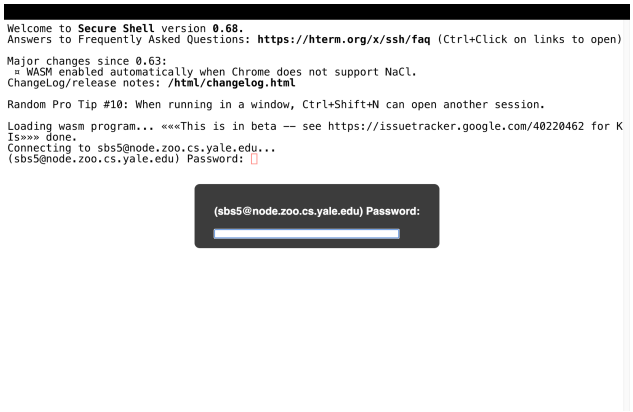
Click on the extensions icon in the address bar, and select **Secure Shell** from the popup menu.



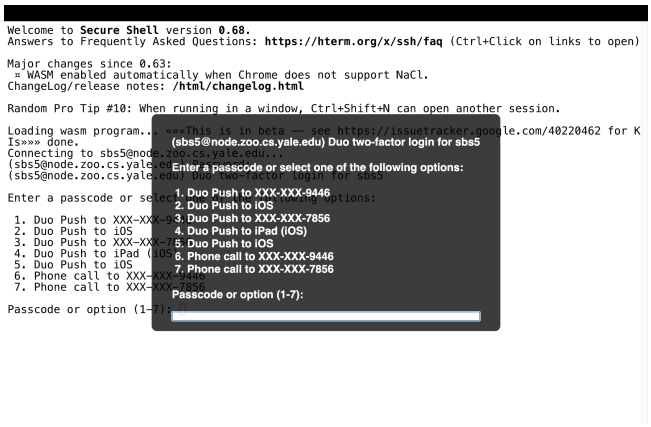
Then, select **Connection Dialog** from the new popup menu, which takes you to the following screen.



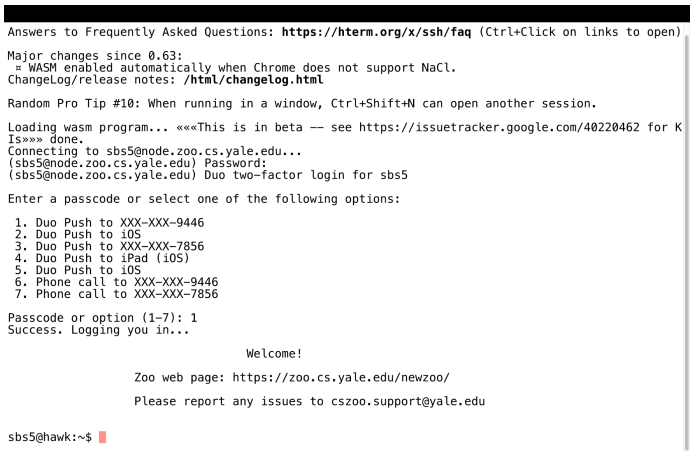
Fill in your netid@nodename information, then click the **[Enter] Connect** button at the lower right.



Input your netid password in the dialog box.



Select a Duo authentication option from the dialog box.



You are now talking to a zoo node!

More information about the zoo is available at https://zoo.cs.yale.edu/dsac/docs/zoo/

Learning UNIX

More information about UNIX is found in Slade's LINUX tutorial:
https://zoo.cs.yale.edu/classes/cs201/UNIX.html

Also, if you are a student in CPSC 2000 or CPSC 2010, you can run the interactive online UNIX tutorial written in Python by a former ULA, which mirrors the above document. It is found a /c/cs2000/www/unixtutorial.py or /c/cs2010/www/unixtutorial.py

The best way to learn UNIX is not by reading, but by typing. You can ssh to the zoo and type in the commands from the tutorial or from class transcripts.

Running Staff Solutions

Compiled versions of the homework assignments for CPSC 2000 and CPSC 2010 are available online in the respective homework directories.

CPSC 2000:

```
sbs5@cardinal:/c/cs2000/hws$ ls
hw0a.pyc  hw0.py  hw1a.pyc  hw1.py  __pycache__
```

The compiled solutions are `hwXa.pyc`. For example, you could enter the following:

```
sbs5@cardinal:/c/cs2000/hws$ python3
Python 3.12.3 (main, Aug 14 2025, 17:47:21) [GCC 13.3.0] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>> import hw0a
>>> hw0a.main()
hours
# is it greater than 0?
 OK  got: 2 expected: <function main.<locals>.<lambda> at
0x78c70228fb00>
productorsum
 OK  got: 2 expected: 2
 OK  got: 300 expected: 300
 OK  got: 90 expected: 90
 OK  got: -2000 expected: -2000
food
# is it a string?
```

```
 OK  got: 'bacon' expected: <function main.<locals>.<lambda> at
0x78c70228fb00>
```

CPSC 2010:

```
sbs5@cardinal:/c/cs2010/hws$ ls
hw0.rkt  hw0_rkt.zo  hw1.rkt  hw1_rkt.zo
```

The compiled solutions are `hwX_rkt.zo`. For example, you could enter the following:

```
sbs5@cardinal:/c/cs2010/hws$ racket
Welcome to Racket v8.10 [cs].
> (enter! "hw0_rkt.zo")
"hw0_rkt.zo"> (runtests)
'((testing
   hours
   OK
   got:
   2
   expected:
   #<procedure:...ws/.answers/hw0.rkt:152:22>)
  (testing kg2lb OK got: (1 pound 1 ounce) expected: (1 pound 1
ounce))
  (testing kg2lb OK got: (2 pounds 3 ounces) expected: (2 pounds 3
ounces))
  (testing kg2lb OK got: (4 pounds 6 ounces) expected: (4 pounds 6
ounces))
  (testing kg2lb OK got: (6 pounds 9 ounces) expected: (6 pounds 9
ounces))
  (testing kg2lb OK got: (8 pounds 12 ounces) expected: (8 pounds 12
ounces))
  (testing kg2lb OK got: (19 pounds 12 ounces) expected: (19 pounds
12 ounces))
  (testing kg2lb OK got: (22 pounds 0 ounces) expected: (22 pounds 0
ounces))
  (testing kg2lb OK got: (220 pounds 0 ounces) expected: (220 pounds
0 ounces))
  (testing
   timezone
   OK
   got:
   "EST"
   expected:
   #<procedure:...ws/.answers/hw0.rkt:163:30>))
```

Transferring Files

One of the earliest Internet utilities was FTP - the file transfer protocol. See https://en.wikipedia.org/wiki/File_Transfer_Protocol It makes it easy to copy files from one computer on the network to another. Like the telnet utility mentioned above, the FTP program was not secure. It was easy to eavesdrop on or even corrupt transmissions. Thus was born SFTP - secure FTP. See https://en.wikipedia.org/wiki/SSH_File_Transfer_Protocol

sftp is available on UNIX. See the man page: https://linux.die.net/man/1/sftp

Below is a simple example in which I copy a file from my UNIX workstation to the zoo and then copy a file from the zoo back to my UNIX workstation.

```
$ cat .bashrc

echo "This is .bashrc"
PATH=$PATH:$HOME/racket/bin
$ sftp sbs5@python.zoo.cs.yale.edu
Connected to python.zoo.cs.yale.edu.
sftp> put .bashrc dot.bashrc
Uploading .bashrc to /home/accts/sbs5/dot.bashrc
.bashrc                                       100%   52    74.8KB/s
00:00
sftp> cd /c/cs2010/hws
sftp> ls
hw0.rkt      hw0_rkt.zo   hw1.rkt      hw1.rkt~     hw1_rkt.zo
sftp> get hw0.rkt
Fetching /home/classes/cs201/hws/hw0.rkt to hw0.rkt
hw0.rkt                                       100% 6146     1.3MB/s
00:00
sftp> quit
$ file hw0.rkt
hw0.rkt: ASCII text
```

An alternative to **sftp** is the secure copy or **scp** command. See https://linux.die.net/man/1/scp

For Windows, see
https://sftpcloud.io/learn/sftp/windows-sftp-command-line-a-comprehensive-guide-for-secure-file-transfers