EECS Tutorial: cslab Linux Environment SSH Access

FAQ for SSH access into cslab Linux environment

How do I access the cslab Linux environment on port 22 via an SSH client?

- The cslab-nodes are located inside a virtual private network. These internal nodes are only SSH accessible via the cslab-bastion.cs.wichita.edu jumphost. Any external connection into the cslab environment must proxy connect through the cslab-bastion and the bastion should be used only as an SSH jumphost/proxy.
- If you are accessing cslab via SSH on WSU campus, ensure you are connected wirelessly to "WSU Secure" or using an Ethernet connected computer. "WSU Guest" wireless prohibits port 22 connections and your SSH session will fail to connect.
- **Please only connect into the cslab environment with an SSH client if you have previous experience in using SSH and the Linux command-line. If you are new to Linux, please use the cslab *Guacamole* web-browser interface.**

Using *PuTTY* SSH client on Microsoft Windows:

1. Download the full *PuTTY* package from Simon Tatham's official download webpage and install all *PuTTY* utilities. You cannot connect to cslab with just the *PuTTY* client.
2. Run the
3. Open *PuTTY* and in [Session] category add a new SSH connection with [Host Name] as `cslab-bastion.cs.wichita.edu` and [Port] as `22`.
4. In [Connection–Data] category add your myWSU_ID into [Auto-login username].
5. In [Connection–SSH] category add the text `ssh cslab` into [Remote command].
6. In [Session] category add a name for the configuration, such as "cslab environment" and click *Save*. Saving as "Default Settings" will always open with this configuration.
7. The first time you connect to the cslab environment using SSH, you will be asked to confirm the authenticity of each SSH remote host.
8. Ensure the cslab-bastion and cslab host key fingerprints match one of the following SHA256 or MD5 hashes before clicking *Yes*:

```
ECDSA key fingerprint is
SHA256:X6dBKj4sqYYPWol6MXSQvGhpIQ6qBxh7mBQhnSw8n64
MD5:d8:ba:c6:1c:86:fa:7f:f6:92:4f:c1:02:30:ce:ab:99

ED25519 key fingerprint is
SHA256:zzozIV7cP1T9C77PLRaevzdzCu21k44lbjd8jaJKS8Q
MD5:6d:3d:8e:3a:db:f6:de:33:af:77:01:40:f3:71:1d:14
```

```
RSA key fingerprint is
SHA256:0CUyGZAYMdOd8vTOK3AtM2XTX3lMaGA2NP73rR7s6Ns
MD5:75:5a:16:53:1a:7c:c2:4b:99:66:2d:e3:1e:76:f9:c9

DSA key fingerprint is
SHA256:7zW122xr+aoBb5yiRI96nvdx8Ml07qLKHYwG2Wu6jIM
MD5:27:59:53:18:5a:67:71:f6:32:f1:e1:15:e9:e5:fe:b1
```

9. You will be prompted twice to enter your myWSU password, once for the cslab-bastion (SSH jumphost/proxy) and once for the internal cslab-node.

10. You may see a `Could not chdir to home directory....` warning. Do not be concerned, this is not a critical error.

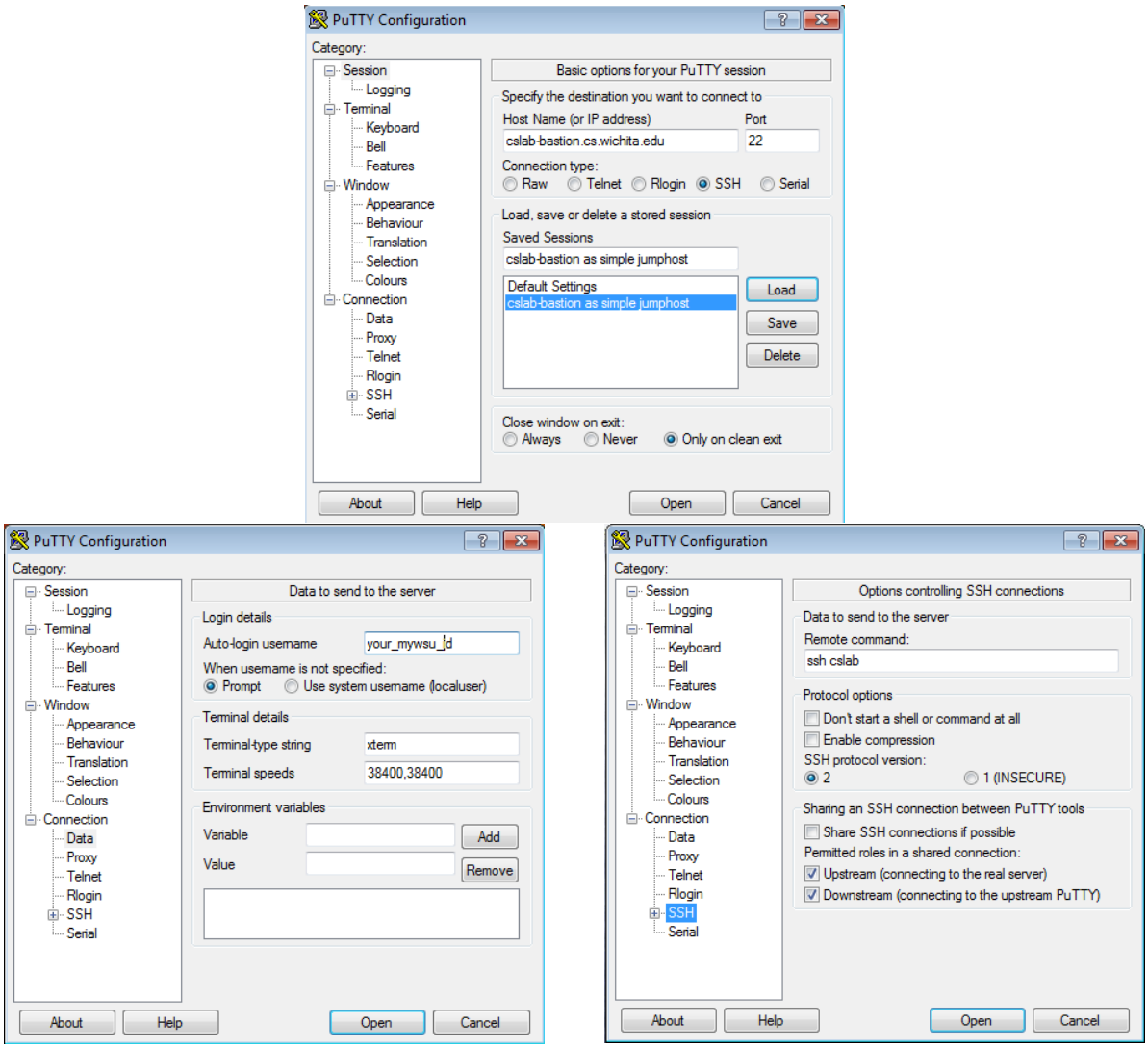

Figure 1: *PuTTY* configuration for cslab

11. If the SSH connection completed successfully, then you will be presented with a standard shell prompt:

    `your_mywsu_id@cslab-node-#:~$`

Configuring *OpenSSH* client on Linux or Mac OSX for use with cslab:

1. On your Linux or Mac desktop/laptop computer, add the following host entry into your local user `~/.ssh/config` file:

```
Host cslab cslab-last cslab.cs.wichita.edu cslab-last.cs.wichita.edu
  ProxyCommand ssh your_mywsu_id@cslab-bastion.cs.wichita.edu ballast %h
  User your_mywsu_id
  IdentityFile ~/.ssh/cslab_rsa
  HostKeyAlias cslab.cs.wichita.edu
```

(Note: replace "your_mywsu_id" with your own 8 character ID number in both `ProxyCommand` and `User` lines.)

2. Open a command-line terminal emulator window and generate a new SSH public/private key pair for your local user by typing
   `ssh-keygen -t rsa -b 4096`

3. Follow the prompts in the command-line as your new SSH key is generated:

   ```
   Generating public/private rsa key pair.
   Enter file in which to save the key (../id_rsa):
   ```
   Type `~$/.ssh/cslab_rsa` and press Enter
   `Enter passphrase:` Enter a passphrase you will remember!
   ```
   Your identification has been saved in .../.ssh/id_rsa.
   Your public key has been saved in .../.ssh/id_rsa.pub.
   The key fingerprint is:
   SHA256:  [fingerprint and randomart image]
   ```

   It is highly recommended to use a passphrase for your SSH key to keep your Linux user account on the cslab system secure.

4. Open the newly generated SSH public key by typing
   `less ~$/.ssh/cslab_rsa.pub`

5. Select all the text displayed within *less*, starting with `ssh-rsa` and ending with the hostname of your local computer.

6. Copy the text either by right-clicking on the terminal emulator window and selecting copy or by pressing the key combination **Ctrl+Shift+C**.

7. Open a web-browser application and follow the ADDTUTORIALCMD to access cslab-gateway.cs.wichita.edu

8. Once logged into the cslab *guacamole* web-interface, open a [cslab_SSH_CLI_terminal] connection.

9. Within the cslab SSH terminal session in your browser, open the *Guacamole* menu sidebar by pressing the key combination **Ctrl+Alt+Shift**.

10. Paste the copied text to the remote *Guacamole* [Clipboard] field using your preferred method, i.e. **Ctrl+V**.

11. Close the *Guacamole* menu sidebar by pressing the key combination **Ctrl+Alt+Shift**.

12. Within the cslab SSH terminal session, open the file `~/.ssh/authorized_keys` by typing
    `nano ~/.ssh/authorized_keys`

13. Paste your locally copied SSH public key into the terminal session by right-clicking on the browser window with your mouse or by pressing the key combination **Ctrl+Shift+V**.

14. Ensure you have a blank line at end of the text file by pressing **Enter** after the text.

15. Quit *nano* by pressing **CTRL+X** and follow the prompts at the bottom of the screen to ensure you save the `~/.ssh/authorized_keys` file.

16. In a local CLI terminal connect to the cslab Linux environment by typing

    ```
    ssh cslab
    ```

17. The first time you connect to the cslab environment using SSH, you will be asked to confirm the authenticity of each SSH remote host, i.e.

    ```
    The authenticity of host 'cslab.cs.wichita.edu' can't be established.
    ECDSA key fingerprint is [SHA256 or MD5 hash value].
    Are you sure you want to continue connecting (yes/no)?
    ```

18. Ensure the cslab-bastion and cslab host key fingerprints match one of the following SHA256 and MD5 hashes before typing `yes`:

    ```
    ECDSA key fingerprint is
    SHA256:X6dBKj4sqYYPWol6MXSQvGhpIQ6qBxh7mBQhnSw8n64
    MD5:d8:ba:c6:1c:86:fa:7f:f6:92:4f:c1:02:30:ce:ab:99

    ed25519 key fingerprint is
    SHA256:zzozIV7cP1T9C77PLRaevzdzCu21k44lbjd8jaJKS8Q
    MD5:6d:3d:8e:3a:db:f6:de:33:af:77:01:40:f3:71:1d:14

    RSA key fingerprint is
    ```

```
SHA256:0CUyGZAYMdOd8vTOK3AtM2XTX3lMaGA2NP73rR7s6Ns
MD5:75:5a:16:53:1a:7c:c2:4b:99:66:2d:e3:1e:76:f9:c9

DSA key fingerprint is
SHA256:7zW122xr+aoBb5yiRI96nvdx8Ml07qLKHYwG2Wu6jIM
MD5:27:59:53:18:5a:67:71:f6:32:f1:e1:15:e9:e5:fe:b1
```

19. You will be prompted twice to enter your myWSU password, once for the cslab-bastion (SSH jumphost/proxy) and once for the internal cslab-node.

20. You may see a `Could not chdir to home directory....` warning. Do not be concerned, this is not a critical error.

21. If the SSH connection completed successfully, then you will be presented with a standard shell prompt:

    ```
    your_mywsu_id@cslab-node-#:~$
    ```

How do I use graphical (GUI) applications in cslab via an SSH client?

Graphical applications cannot currently be accessed in Microsoft Windows using the *PuTTY* SSH client. Please use the *Guacamole* web-browser interface instead.

Using *OpenSSH* client with X11 forwarding on Linux or Mac OSX:

- Ensure you have followed the directions in using *OpenSSH* client first.
- SSH allows for graphical applications to run on a local computer from the remote cslab Linux environment using X11 forwarding.
- To use X11 forwarding on a per session basis append the `-X` option flag to your SSH command, i.e.

  ```
  ssh -X cslab
  ```

- To always use X11 forwarding for connections to cslab, instead of using the `-X` option flag, add the following line to the `Host cslab cslab-last...` entry in your `~/.ssh/config` file:

  ```
  ForwardX11 yes
  ```

- If you are using Mac OSX, then you may need to install XQuartz before using X11 forwarding. Download XQuartz for Mac and install the software package.

How do I copy files from/to the cslab Linux environment via an SSH client?

Using *OpenSSH* client on Linux or Mac OSX:

- Ensure you have followed the directions in using *OpenSSH* client first.
- To copy a file from your local computer to your user home directory on cslab using Secure Copy (SCP), type

  ```
  scp local_filename_or_path cslab:~
  ```

- To copy a file from your user home directory on cslab to a local directory on your local computer, type

  ```
  scp cslab:~/remote_filename_or_path local_directory
  ```

How do I access the last accessed cslab-node via an SSH client?

- When connecting to the cslab Linux environment using an SSH client, the ballast load-balancer on cslab-bastion will redirect you to one of the available and least used cslab-nodes at time of connection. Since load-balancing is calculated by ballast on a one minute cycle, you may not be redirected to the same cslab-node the next time you connect into cslab using SSH.
- **Use the following instructions to connect to the last used cslab-node only if/when you need access into a previously running SSH connection. For normal use, your best option is always to connect via `ssh cslab` and let the ballast load-balancer automatically connect to an available node.**

Using *OpenSSH* client on Linux or Mac OSX:

1. If you need to connect to the last cslab-node you previously accessed using SSH, just append `-last` to the SSH command, i.e.

   ```
   ssh cslab-last
   ```

Using *PuTTY* SSH client on Microsoft Windows:

1. Open *PuTTY* and load the previously created "cslab environment" session.
2. Change [Connections–Data] configuration category [Remote command] entry to `ssh cslab-last`.
3. *Save* changed configuration as a new [Session] with a different name, such as "cslab environment last used node".
4. If you need to connect to the last cslab-node you previously accessed using SSH, connect using this newly created *PuTTY* session.

What are the currently known bugs/glitches within the cslab Linux environment?

Due to the cslab environment being so new, there are a few software bugs and glitches which you may see at times. Most of these bugs occur during operations that involve the new cslab environment interacting with the older CS servers and functions and are not critical issues.
Known bugs include:

- A policykit error message pops up when user first logs into a *Guacamole* RDP desktop session if other users are also logged in. This is a minor bug within policykit and can be easily mitigated by clicking *Okay* in the pop-up window.
- When the *handin* command is used in the cslab environment for programming assignment submission it may produce a `segmentation fault` error. This is caused by the older 32-bit *handin* program running on the cslab 64-bit processor architecture. The error is minor and does not affect submission of student assignments for grading.

If you experience a bug or error when using the cslab Linux environment which stops you from completing your programming assignments, please inform your instructor at your earliest convenience.