EECS Tutorial: cslab Linux Environment SSH Access

FAQ for SSH access into cslab Linux environment

General Information regarding accessing cslab Linux environment using SSH

- **Only connect to 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 web-browser interface at cslab-gateway.cs.wichita.edu.**
- SSH uses network port 22. 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.
- You can only access cslab using SSH public key authentication. You cannot access cslab over SSH using your myWSU password.
- The cslab-nodes are located inside a virtual private network. These internal nodes are only SSH accessible via the `cslab-bastion.cs.wichita.edu` proxy host. Any external connection into the cslab environment must first proxy connect through the `cslab-bastion`.
- `cslab-sftp.cs.wichita.edu` can be used as a secure file transfer host with command-line SSH clients, such as *PSFTP* or *SFTP*, and graphical SFTP clients, such as *Filezilla*.
- This tutorial will help you configure your SSH and SFTP clients for access into cslab using SSH public key authentication via a proxy host.
- **ONLY USE CSLAB-BASTION AS A PROXY/JUMPHOST AND CSLAB-SFTP AS AN SCP/SFTP SERVER. DO NOT DIRECTLY SSH INTO CSLAB-BASTION OR CSLAB-SFTP FOR ANY OTHER TASKS!**

How do I access cslab Linux environment via *PuTTY* on Microsoft Windows?

Configuring the *PuTTY* SSH client for cslab access:

1. Download the full *PuTTY* package from Simon Tatham's official download page and install all *PuTTY* utilities on your local Windows computer. You cannot connect to cslab with just the *PuTTY* client.
2. Run the *PuTTY Key Generator*, which is listed in your start menu as *PuTTYgen*.
3. In the *PuTTY Key Generator* window, generate a new RSA key with 4096 bits. You may need to change the settings at bottom of window before clicking **Generate**.
4. Follow the prompts to create randomness and generate your *PuTTY* key.
5. Once your key has been generated, change the following fields:
   - Key comment: `your_mywsu_id@your_local_computer_name`
   - Key passphrase: `choose_passphrase_you_will_remember`
   - Confirm passphrase: `same_passphrase_as_above`

   It is highly recommended to use a passphrase for your new key to keep your Linux user account secure.
6. Select all the text displayed within the box titled *Public key for pasting into OpenSSH authorized_keys file*, starting with `ssh-rsa` and ending with name of your computer.
7. Copy the selected text either by right-clicking within the same box and selecting [copy] or by pressing the key combination **Ctrl+C**.
8. Open a web-browser application and follow the eecs_tutorial_cslab_web_access document to access cslab-gateway.cs.wichita.edu
9. Once logged into *guacamole*, open a **cslab_SSH_CLI_terminal** connection.
10. Within the cslab SSH terminal session in your browser, open the *Guacamole* menu sidebar by pressing the key combination **Ctrl+Alt+Shift**.
11. Paste the copied text to the remote *Guacamole* **Clipboard** field using your preferred method, i.e. **Ctrl+V**.
12. Close the *Guacamole* menu sidebar by pressing the key combination **Ctrl+Alt+Shift**.
13. Within the cslab SSH terminal session, open your `authorized_keys` file by typing `nano ~/.ssh/authorized_keys`

14. Paste your locally copied *PuTTY* 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**.

15. Ensure you have a blank line at end of the text file by pressing **Enter** if required.

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

17. Ensure correct permissions are set on the `authorized_keys` file by typing `chmod 600 ~/.ssh/authorized_keys`

18. NOTE: If you wish to set up more than one local computer with different SSH keys for accessing cslab, then you can append additional SSH public keys in your cslab user `authorized_keys` file. Make sure to remove no longer used SSH public keys from this file.

19. Once you are done with the above steps, make sure to properly disconnect and log out of the cslab *guacamole* web-interface.

20. Back in the *PuTTY Key Generator* window, click **Save private key**.

21. In the *Save private key* window, browse to a local directory where you wish to securely store your new *PuTTY* SSH key and in the **File name** field enter the name `cslab_rsa`. Clicking on **Save** will store your `cslab_rsa.ppk` key file to your local computer. **Keep this file safe and private!**

22. OPTIONAL: If you wish to also store your SSH public key as a simple text file for future reference, then click **Save public key**, browse to the same directory, and in the **File name** field enter the name `cslab_rsa_public_key.txt`

23. Once you are done with the above steps, exit/close the *PuTTY Key Generator* program.

24. Download the cslab_putty_ssh_client_full_configuration.reg registry file from Ben Roose's GitHub tutorials repository. You may need to right-click on the web-page in your browser and select **Save as** or **Save page as**.

25. Double-click on the downloaded file in your file explorer, and click **Yes** when asked if you wish to add this new information to your Windows registry. You will not be able to continue configuration of PuTTY until this file has been added into your Windows registry.

26. Run the *PuTTY* program and you should now see two new **Saved Sessions** entries in your *PuTTY* Configuration Window listed as *cslab access* and *cslab last node accessed*.

27. Click on *cslab access* and click on **Load**. DO NOT click **Open**.

28. Click on the *Connection–Data* Category tab on the left of the window and enter your myWSU ID in the **Auto-login username** field.

29. Click on the *Connection–Proxy* Category tab on the left of the window and enter your myWSU ID in the **Username** field.

30. Click on the *Session* Category tab on the left of the window and click on **Save**. DO NOT click **Open**.

31. Perform the last 4 steps again for the *cslab last node accessed* entry.

32. Once you are done with the above steps, exit/close the *PuTTY* program.

33. Congratulations! You have now configured your local Windows computer to directly access cslab with *PuTTY*. However, you will need to add your new SSH key to the *Pageant (PuTTY Authentication agent)* prior to logging into cslab. Running *Pageant* and adding your SSH key will be explained in the next section.

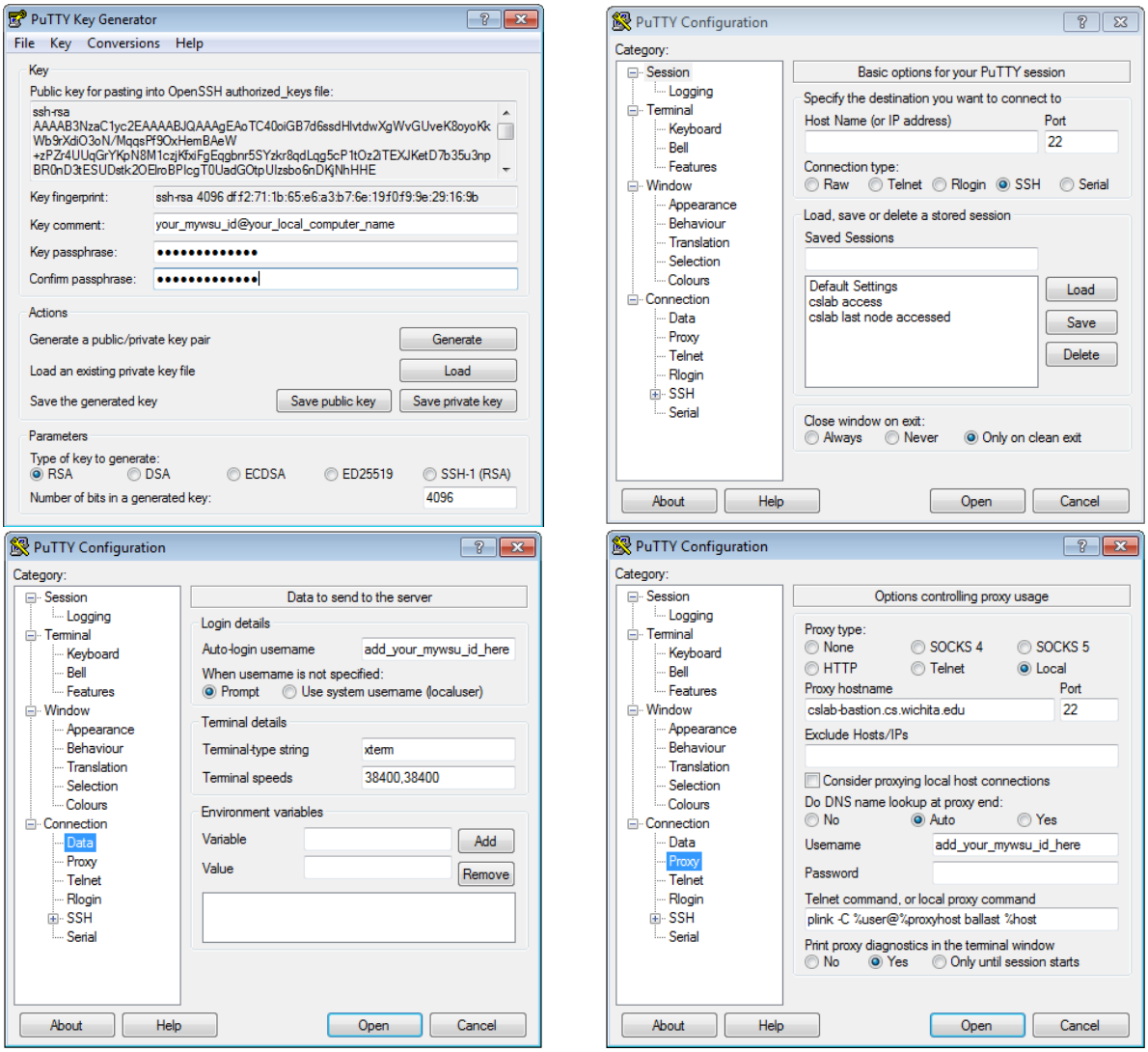

Figure 1: *PuTTY* screenshots for cslab configuration

Logging into cslab using the *Pageant* agent:

1. Run the *Pageant (PuTTY Authentication agent)*, listed in your start menu as *Pageant*.
2. On the far right of your Windows taskbar, you should see a new notification icon for the running *Pageant* program.
3. Right-click on the *Pageant* notification icon and select the menu option **Add key**.
4. In the *Select private Key File* window, browse to the local directory with your stored `cslab_rsa.ppk` file. Select the `cslab_rsa` key file and click on **Open**.
5. When prompted, enter the previously defined passphrase for your key.
6. OPTIONAL: if you wish to see the SSH keys already added into *Pageant*, right-click on the *Pageant* notification icon and select the menu option **View keys**.
7. Right-click on the *Pageant* notification icon and, under the **Saved Sessions** sub-menu, select **cslab remote access**.
8. If everything is correctly configured, a *PuTTY* terminal emulator window will open and automatically log you into an available cslab node! You should be presented with the shell prompt: `your_mywsu_id@cslab-node-#:~$`
9. Each time you shutdown or reboot your computer, you will need to manually run *Pageant* and then add your SSH key before you can connect to the cslab environment.
10. OPTIONAL: You can have *Pageant* run automatically at startup by copying the *Pageant* Start menu shortcut file into your Windows **Startup** directory.
11. OPTIONAL: To have your cslab SSH key automatically added into *Pageant* when it runs, open the **Properties** box for the *Pageant* Start menu shortcut (or the previously created **Startup** shortcut). In the **Target** field append the full filepath of your `cslab_rsa.ppk` key file to the end of the `pageant.exe` line, i.e.

```
"C:\Program Files\PuTTY\pageant.exe" "C:\Users\localuser\cslab_rsa.ppk"
```

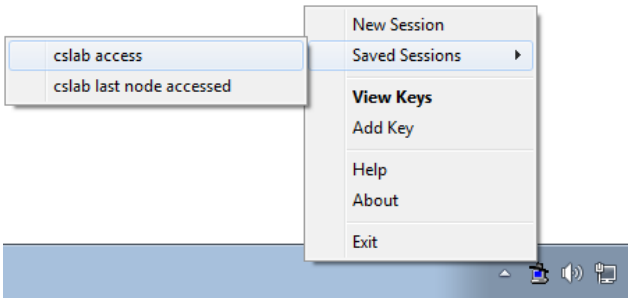

Figure 2: *Pageant* taskbar menu for cslab access

Logging into your last used cslab-node-# using the *PuTTY* client:

- When connecting to the cslab Linux environment using an SSH client, the *ballast* load-balancer 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 each time you connect.
- **Use the *cslab last node accessed* saved session to connect to the last used cslab-node only when you need access into a previously running SSH session. For normal use the best option is to connect using the *cslab access* saved session and let the *ballast* load-balancer automatically connect to an available node.**

Running graphical (GUI) applications using the *PuTTY* client:

1. SSH allows for graphical applications to run on a local computer from the remote cslab Linux environment using X11 forwarding.
2. You will need to install the Xming software before using X11 forwarding with *PuTTY*. Download the Xming X Server for Windows and install the software package.
3. Run the *PuTTY* program, click on **cslab access** in Saved Sessions, and click on **Load**.
4. Click on the *Connection–SSH–X11* Category tab on the left of the window.
5. Turn on the **Enable X11 Forwarding** check box and ensure the **X display location** contains `localhost:0.0`. If you change the display location within *Xming Launch* settings, then also change the **X display location** in *PuTTY* to ensure consistency.
6. Click on the *Session* Category tab on the left of the window. Save this change as a different **Saved Session** by entering a new name for the session, such as `cslab with graphical access`, and click on **Save**.
7. Once you are done with the above steps, exit/close the *PuTTY* program.
8. To use cslab with X11 forwarding, first run *Xming* in the background and then select your new **Saved Session** from the *Pageant* notification menu.
9. Once a connection into cslab has been established, you can type the name of a GUI application you wish to run, such as `geany &`

How do I copy files from/to cslab Linux environment via SSH on Windows?

Copying files remotely using *PuTTY* command-line tools:

- Ensure you have first followed the directions in the previous section to configure your *PuTTY* client and *pagant* is running with your SSH private key added.
- Open a Windows command prompt window.
- To copy a file from your local computer to your user home directory on cslab using *PuTTY Secure Copy (PSCP)*, type
  `pscp local_file mywsu_id@cslab-sftp.cs.wichita.edu:remote_dir`
- To copy a file from your user home directory on cslab to a local directory on your local computer using *PuTTY Secure Copy (PSCP)*, type
  `pscp mywsu_id@cslab-sftp.cs.wichita.edu:remote_file local_dir`
- To use *PuTTY SSH File Transfer Protocol (PSFTP)* interactive command-line tool, type
  `psftp mywsu_id@cslab-sftp.cs.wichita.edu`
- To see a list of available interactive commands once in `psftp>`, type `help`
- For further information on `pscp` and `psftp`, see the online *PuTTY* documentation for Chapter 5 and Chapter 6 respectively.

Copying files remotely using *SFTP* with the *FileZilla* graphical client:

1. Ensure you have first followed the directions in the previous section to configure your *PuTTY* client.
2. Download `FileZilla` software from the FileZilla download page and install the SFTP client on your local Windows computer.
3. Run the *FileZilla* graphical client program.
4. In the *FileZilla* window, open the **Site Manager** via the **File** menu, by clicking on the upper-left icon, or by pressing **CTRL+S**.
5. In *Site Manager* window, click **New Site** and type in a site name, i.e. `cslab-sftp`
6. In *Site Manager* window, change the following fields:
   - Host: `cslab-sftp.cs.wichita.edu`
   - Protocol: `[SFTP - SSH File Transfer Protocol]`
   - Logon Type: `[Key file]`
   - User: `your_mywsu_id`

7. *FileZilla* can use *PuTTY* SSH keys. To the right of the Key file field click **Browse**.
8. In the *Choose a key file* window, browse to the local directory with your stored `cslab_rsa.ppk` file. Select the `cslab_rsa` key file and click on **Open**.
9. Click **Connect** and enter the passphrase/password to unlock your *PuTTY* private key when prompted.
10. If you have set up *FileZilla* correctly, it will automatically connect to `cslab-sftp` and display your home directory on cslab in the right pane.
11. Drag-and-drop files to transfer them between your local computer and the remote cslab environment.
12. If you wish to hide "hidden" files which start with a period, then select **Directory Listing Filters** under the **View** menu, enable *Configuration files* on the Remote filter, and click **OK** or **Apply**.
13. For further information on how to use *FileZilla*, see the FileZilla documentation page.

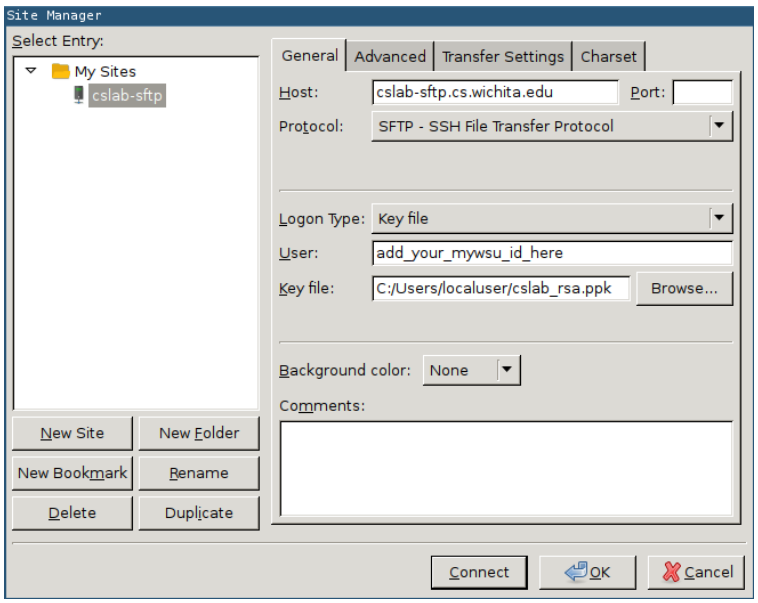

Figure 3: *FileZilla* Site Manager settings for cslab-sftp

Why is *PuTTY* or *FileZilla* asking to confirm the authenticity of cslab host keys?

Automatically adding cslab host keys to your registry for *PuTTY*:

- SSH host keys are essential to securing an SSH connection into a remote server. If you see an incorrect host key, then it may mean a cyber-attacker has attempted to compromise the remote server.
- The cslab_putty_ssh_client_full_configuration.reg registry file will automatically add the cslab host keys into your Windows registry during your initial configuration and set up of *PuTTY*. However, if your configuration loses its cached host keys for any reason, then it may ask you to reconfirm the authenticity of cslab hosts when you try to open an SSH connection.
- You can also download the cslab_putty_ssh_host_keys_only.reg registry file and re-add just the cslab host keys into your Windows registry.

Manually adding the cslab host keys into *PuTTY* or *FileZilla*:

- You can manually add SSH host keys to your *PuTTY* and *FileZilla* configurations. When adding new host keys you should always ensure the host key fingerprint is correct. `cslab`, `cslab-bastion`, and `cslab-sftp` host key fingerprints must match one of the following SHA256 or MD5 hashes before you add or accept the host key:

```
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
```

How do I access cslab Linux environment via *OpenSSH* on Linux or Mac?

Configuring the *OpenSSH* client for cslab access:

1. Many of the following commands need to be typed into a local command-line terminal, so open a command-line terminal emulator window first.

2. Install the `openssh-client` software package using your specific distribution's package management tools if not already installed.

3. Download the cslab_openssh_client_config file from Ben Roose's GitHub tutorials repository. You may need to right-click on the web-page in your browser and select **Save as** or **Save page as**, or you can download this file directly using `wget`

4. The downloaded cslab configuration can be added into your SSH client config while substituting your own myWSU ID into it by changing into the directory where you downloaded the `cslab_openssh_client_config` file and typing

   ```
   sed "s/mywsu_placeholder/ENTER_YOUR_OWN_MYWSU_ID_HERE/g" \
    cslab_openssh_client_config >> ~/.ssh/config
   ```

5. Alternatively, you can manually copy the following host entries into your local `~/.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 $HOME/.ssh/cslab_rsa
  Compression yes
  HostKeyAlias cslab.cs.wichita.edu

Host cslab-sftp cslab-sftp.cs.wichita.edu
  HostName cslab-sftp.cs.wichita.edu
  User your_mywsu_id
  IdentityFile $HOME/.ssh/cslab_rsa
  Compression yes
  HostKeyAlias cslab.cs.wichita.edu
```

(Ensure to replace `your_mywsu_id` with your own 8 character myWSU ID number in all three `ProxyCommand` and `User` lines.)

6. The cslab SSH host keys must be added into your local `~/.ssh/known_hosts` file. Download the cslab_openssh_known_hosts file from Ben Roose's GitHub tutorials repository. You can download this file directly using `wget` You may need to right-click on the web-page in your browser and select **Save as** or **Save page as**, or you can download this file directly using `wget`

7. The host keys can be added into your known_hosts by changing into the directory where you downloaded the `cslab_openssh_known_hosts` file and typing
`cat cslab_openssh_known_hosts >> ~/.ssh/known_hosts`

8. Generate a new *OpenSSH* public/private key pair for your local user by typing
`ssh-keygen -t rsa -b 4096 -f ~/.ssh/cslab_rsa`

9. Follow the prompts in the command-line as your new *OpenSSH* key is generated and enter a passphrase you will remember!
It is highly recommended to use a passphrase for your new key to keep your Linux user account secure.

10. Open the newly generated *OpenSSH* public key by typing
`less ~/.ssh/cslab_rsa.pub`

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

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

13. Open a web-browser application and follow the eecs_tutorial_cslab_web_access document to access cslab-gateway.cs.wichita.edu

14. Once logged into *guacamole*, open a **cslab_SSH_CLI_terminal** connection.

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

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

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

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

19. Paste your locally copied *OpenSSH* 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**.

20. Ensure you have a blank line at end of the text file by pressing **Enter** if required.

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

22. Ensure correct permissions are set on the `authorized_keys` file by typing
`chmod 600 ~/.ssh/authorized_keys`

23. NOTE: If you wish to set up more than one local computer with different SSH keys for accessing cslab, then you can append additional SSH public keys in your cslab user `authorized_keys` file. Make sure to remove no longer used SSH public keys from this file.

24. Congratulations! You have now configured your local Linux or Mac computer to directly access cslab with *OpenSSH*. Make sure to properly disconnect and log out of the cslab *guacamole* web-interface once you are done with the above steps.

Logging into cslab using the *OpenSSH* client:

1. Ensure you have first followed the directions in the previous section to configure your *OpenSSH* client.

2. In your local computer open a command-line terminal emulator window, such as the Mac OSX Terminal application, *xterm*, *lxterminal*, or *terminator*.

3. Start the ssh-agent in the background by typing
   `eval "$(ssh-agent -s)"`

4. Add your cslab_rsa private key to the ssh-agent by typing
   `ssh-add ~/.ssh/cslab_rsa`

5. When prompted, enter the passphrase to unlock your local SSH private key.

6. Connect to the cslab Linux environment by typing
   `ssh cslab`

7. If everything is correctly configured, you will be automatically connected to an available cslab node! You should be presented with the shell prompt:
   `your_mywsu_id@cslab-node-#:~$`

8. NOTE: Mac OSX and many Linux distributions can run the ssh-agent and unlock SSH keys during user login using a *keychain* program. Check online for specific details on how to store SSH keys in your distribution. If not, then there are many script functions that can be added to your local user `.bashrc` file for running the ssh-agent at login, such as tstellanova's start_ssh_agent.sh on GitHubGist.

Logging into your last used cslab-node-# using the *OpenSSH* client:

- When connecting to the cslab Linux environment using `ssh cslab`, the *ballast* load-balancer 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 each time you connect.
- **Only connect to the last used cslab-node if you need access into a previously running SSH session. For normal use the best option is always let the *ballast* load-balancer automatically connect to an available node.**
- To connect to the last node you previously accessed using SSH type
  `ssh cslab-last`

Running graphical (GUI) applications using the *OpenSSH* client:

- SSH allows for graphical applications to run on a local computer from the remote cslab Linux environment using X11 forwarding.
- If you are using Mac OSX, then you may need to install the *XQuartz* software before using X11 forwarding. Download XQuartz for Mac and install the software package.
- To use X11 forwarding on a per session basis append the `-X` option flag to your SSH command:
  `ssh -X cslab`
- To always use X11 forwarding for connections into cslab, instead of using the `-X` option flag, add the following line to the `Host cslab cslab-last` entry in your local user `/.ssh/config` file:
  `ForwardX11 yes`
- Once a connection into cslab has been established, you can type the name of a GUI application you wish to run, such as `geany &`

How do I copy files from/to cslab Linux environment via SSH on Linux or Mac?

Copying files remotely using *OpenSSH* command-line tools:

- Ensure you have first followed the directions in the previous section to configure your *OpenSSH* client.
- 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-sftp:~`
- To copy a file from your user home directory on cslab to a local directory on your local computer using *Secure Copy (SCP)*, type
  `scp cslab-sftp:~/remote_filename_or_path local_directory`
- To use the *SSH File Transfer Protocol (SFTP)* interactive command-line tool, type
  `sftp cslab-sftp`
- To see a list of available interactive commands once in `sftp>`, type `help`

Copying files remotely using *SFTP* with the *FileZilla* graphical client:

1. Ensure you have first followed the directions in the previous section to configure your *OpenSSH* client.
2. Install the `filezilla` SFTP client software package using your distribution's package management tools, or via the FileZilla download page.
3. Run the *FileZilla* graphical client program.
4. In the *FileZilla* window, open the **Site Manager** via the **File** menu, by clicking on the upper-left icon, or by pressing **CTRL+S**.
5. In the *Site Manager* window, click **New Site** and type in a site name, i.e. `cslab-sftp`.
6. In the *Site Manager* window, change the following fields:
   - Host: `cslab-sftp.cs.wichita.edu`
   - Protocol: `[SFTP - SSH File Transfer Protocol]`
   - Logon Type: `[Key file]`
   - User: `your_mywsu_id`
7. *FileZilla* cannot use *OpenSSH* keys without your key being converted into a different format first. To the right of the Key file field click **Browse**.

8. In the *Choose a key file* window, click the "paper & pencil" icon in the upper-left to open *Location:*. In the Location field type `~/.ssh/cslab_rsa` and click **Open**.

9. *FileZilla* will ask you to convert the *OpenSSH* key into a supported *PuTTY* key format. Click **Yes** and enter the passphrase/password to unlock your *OpenSSH* private key when prompted.

10. In the *Select filename for converted key file* window, at the Name field type `~/.ssh/cslab.ppk` and click **Save**.

11. Click **Connect** and enter the passphrase/password to unlock your *FileZilla* supported private key when prompted.

12. If you have set up *FileZilla* correctly, it will automatically connect to `cslab-sftp` and display your home directory on cslab in the right pane.

13. Drag-and-drop files to transfer them between your local computer and the remote cslab environment.

14. If you wish to hide "hidden" files which start with a period, then select **Directory Listing Filters** under the **View** menu, enable *Configuration files* on both Local and Remote filters, and click **OK** or **Apply**.

15. For further information on how to use *FileZilla*, see the FileZilla documentation page.

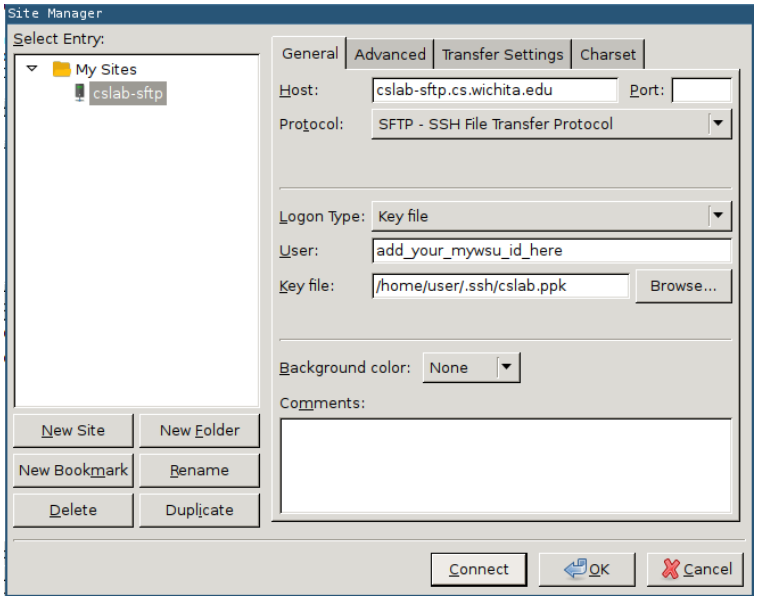

Figure 4: *FileZilla* Site Manager settings for cslab-sftp

Why is *SSH* or *FileZilla* asking to confirm the authenticity of cslab host keys?

Automatically adding the cslab host keys to your known_hosts file:

- SSH host keys are essential to securing an SSH connection into a remote server. If you see an incorrect host key, then it may mean a cyber-attacker has attempted to compromise the remote server. However, if your configuration loses its cached host keys for any reason, then it may ask you to reconfirm the authenticity of cslab hosts when you try to open an SSH connection.
- You can download the cslab_openssh_known_hosts file and re-add the cslab host keys into you local `known_hosts` file by typing
  `cat cslab_openssh_known_hosts >> ~/.ssh/known_hosts`

Manually adding the cslab host keys into *OpenSSH* or *FileZilla*:

- If you do not have the cslab host keys in your `known_hosts` file when you initially connect to cslab, you will likely see the following warning:

```
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)?
```

- You can manually add SSH host keys to your *OpenSSH* and *FileZilla* configurations but you should always ensure the host key fingerprint is correct first. `cslab`, `cslab-bastion`, and `cslab-sftp` host key fingerprints must match one of the following SHA256 or MD5 hashes before you add or accept the host key:

```
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
```