Contents

Getting Started

Most of your work in 6.191 (6.004) will happen on the Athena machines. However, there are a few steps that you will need to do to setup your account: adding the 6.191 (6.004) toolchain, and setting up an SSH key for MIT GitHub.

1 Login to Athena

To access athena, you have a few options. You can setup tools for your computer, access Athena via a web browser, or visit an Athena workstation.

1.1 Connect locally

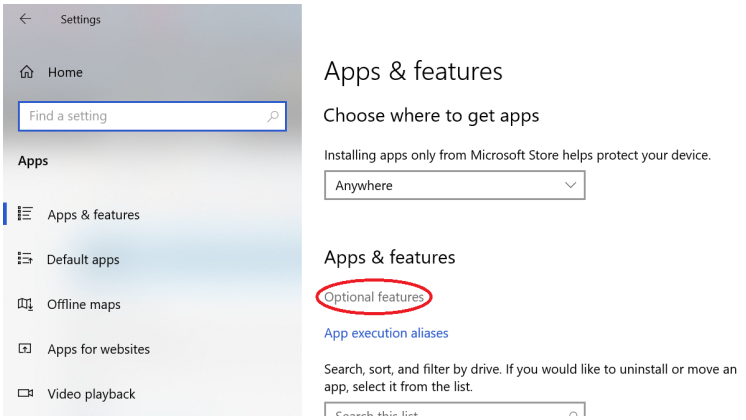
To access Athena on your PC/Mac, you can use SSH. However, if you have a Windows or Mac computer, there are a few things you need to do first.

1.1.1 Windows

To access Athena on Windows, you have need to enable SSH client on your local machine. You can enable the Windows Native SSH Client without additional downloads, or if native SSH client is unavailable, MIT still supports secureCRT as a nalternative SSH client for windows.

Native SSH Client In order to enable SSH client natively on windows without using SecureCRT, the following steps are required.

1. Go to Settings > Apps > Optional features



2. Click Add a feature
3. Search for OpenSSH Client and download the feature
4. Once the feature is downloaded, you can go to windows Command prompt (Searching `cmd` in the window search bar or access through other ways). And type in `ssh` to test whether Window have SSH client installed. If your command prompt recognize this command, it should output the following on your command prompt output or something similar.

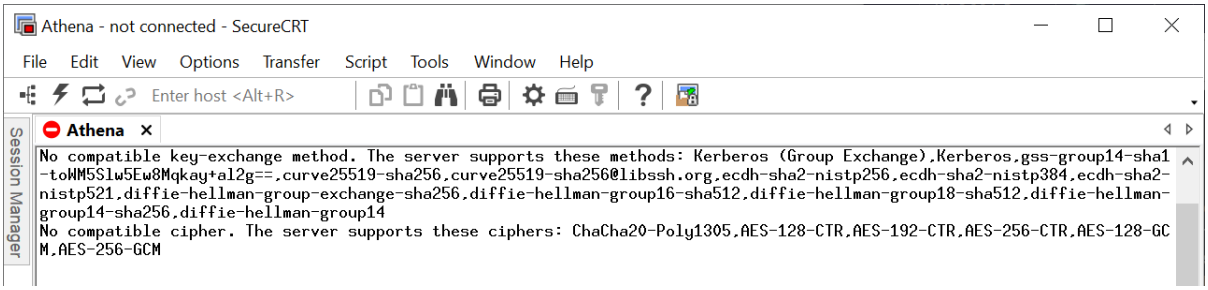
```
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```

5. To connect to Athena, type: `ssh {kerberos}@athena.dialup.mit.edu` and replace `{kerberos}` with your Athena username. After authenticating with your password and duo, you will be on Athena.

SecureCRT If your windows version does not support the native SSH client, you can still download the alternative SSH client shell through the following steps.

1. **Download and install SecureCRT**.
2. **Setup SecureCRT:** On the same page, follow the SecureCRT for Windows: Quick Start Guide instructions (certificates required).
3. **Access Athena** Open SecureCRT and click on **Athena** After providing your password and authenticating with duo, you will be on Athena.

Fixing SecureCRT connection errors Sometimes, after installing and configuring SecureCRT, you still cannot connect to Athena. In this case, you might see the error below. To fix this, you can do the following:



1. Right click the Athena session and click **Properties**.
2. Under **SSH2**, in the **Authentication** section, check **Keyboard Interactive** (for entering your password and Duo authentication), and in the Key exchange section, check **curve25519-sha256**.
3. Under **SSH2** / **Advanced**, check the **AES-256-CTR** cipher.

1.1.2 MacOS

MacOS comes with a terminal app and SSH installed:

1. **Open a terminal**. You can open any terminal application. For example, **Terminal.app** is pre-installed.
   At the terminal prompt, type: `ssh {kerberos}@athena.dialup.mit.edu` and replace `{kerberos}` with your Athena username. After authenticating with your password and duo, you will be on Athena.

1.1.3 Linux

For most Linux systems, no additional setup should be needed. You just need to open a terminal window, and type: `ssh {kerberos}@athena.dialup.mit.edu` and replace `{kerberos}` with your Athena username. After authenticating with your password and duo, you will be on Athena.

1.1.4 Via a web browser

Visit `https://athena.dialup.mit.edu`, where you can use a web-based SSH client to access the dialup servers. While this method requires no configuration, the web-based SSH client may not have some advanced terminal features.

1.2 From a workstation

At any unoccupied workstation that displays the words “Welcome to Athena", enter your username (kerberos) and password.

2 Setup on Athena

2.1 Setting up the 6.191 (6.004) Toolchain

6.191 (6.004) has an AFS locker containing binaries that it needs for you to test and compile your labs, however you need to add this manually. Additionally, we strongly recommend that you configure Athena to use the latest version of Git rather than the default Athena version of Git.

1. **Access your Athena account.**

2. **Add Course Locker Manually.** Run the following commands in your terminal. This will configure your current session to use the 6.004 toolchains required for labs.

```
source /mit/6.004/setup.sh
```

3. **Add Course Locker Automatically.** To add the course locker automatically whenever you log in, you should add the line "`source /mit/6.004/setup.sh`" (without the quotes) to your `.bashrc`. You can do this through any editor (`emacs`, `nano`, `vim`, ...), or by running the following command:

```
echo "source /mit/6.004/setup.sh" >> /mit/{kerberos}/.bashrc
```

   Note that the change to `.bashrc` is applied after you restart the current Athena session.

   If you prefer to not modify your `.bashrc`, you should run "`source /mit/6.004/setup.sh`" manually whenever you login.

4. **Get Latest Version of Git.** The default version of Git on Athena is old and will not be able to show Didit build status locally. To resolve this issue, you will need to update the version of Git in your Athena locker.

   - Similar to above, you can update the Git version manually every time you log by running the command "`add -f git`" (without the quotes) in your terminal.
   - On the other hand, Git can be updated to the latest version automatically whenever you login by adding the "`add -f git`" command to your `.bashrc`. Similar to above, this can be accomplished by any editor (`emacs`, `nano`, `vim`, ...), or by running the following command in your terminal:

```
echo "add -f git" >> /mit/{kerberos}/.bashrc
```

     Note again that the change to `.bashrc` is applied after you restart the current Athena session.

     If you prefer to not modify your `.bashrc`, then you should run "`add -f git`" manually whenever you login.

2.2 Generating an SSH Key

To submit your code for grading, you will be pushing it to MIT's GitHub. However, to access GitHub, you will need to create an SSH key and add it to your account. (An SSH key is a piece of data that can be used to authenticate to many services on the command line; we will use it in 6.004 to authenticate to MIT GitHub so you can pull and submit code.)

On your Athena account (and locally, if you want to push to GitHub), you would run the command `ssh-keygen`. It will ask you where to store the file, and a password. For a good default, paste the following, replacing {kerberos} with your Athena username.

```
ssh-keygen -t rsa -b 4096 -C "{kerberos}@mit.edu"
```

- When you're prompted to "Enter a file in which to save the key," press Enter. This accepts the default file location (`.ssh/id_rsa`). Athena automatically loads this key name to the ssh-agent.
- When you're prompted to "Enter passphrase," you can create a passphrase to secure your SSH key. You will be prompted to type it a second time as a double-check, and will need to enter it again every time in the future you want to use this SSH key. Note that nothing will appear on the screen as you enter your passphrase; this is normal and is how many command-line applications accept passphrases. Also note that entering a passphrase is optional.

2.3 Link your MIT GitHub account

Once you have created your SSH key, you will need to link it with GitHub.

1. Output the content of the public key on your terminal using

   ```
   cat ~/.ssh/id_rsa.pub
   ```

   Then copy the output (without the trailing white spaces) to your clipboard (manually).

2. Login to MIT GitHub or `https://github.mit.edu` with your MIT Kerberos and Password.

3. Go to **Settings** (click on your icon in the top right) → **SSH and GPG keys**.

4. Click **New SSH key**.

5. Paste the content on the Key field.

6. Click **Add SSH Key**.

3 Some basic git commands

In this class, Git is used to track all of your code submissions and automatically send your code to our graders every time you update it. You will only need to use a few simple git commands.

- `git clone` : After following the Create Repository link for the lab you are working on, a repo will be created for you with the initial contents of the lab. The website will also show you a `git clone` command that you need to run in your terminal to make a local copy of your lab repo. This command automatically configures your local repo to print a feedback message from Didit (our lab grading system) on your Terminal when you make a submission.

  Alternatively, you can clone a local repo without the feedback message feature by running the `git clone` command in your terminal:

  ```
  git clone git@github.mit.edu:6191-fa23/lab1-{YourMITUsername}.git
  ```

  replacing lab1 with whichever lab you are working on. While this configuration does not print a Didit feedback message on your terminal on new submission, Didit will grade your new submissions. Refer to the end of this section for accessing Didit results.

- `git commit` : The next instruction is `git commit`. This will take a set of files and package them up and create a point that you can come back to. Once any file tracked by Git is changed, you can commit all your changes by using the command:

  ```
  git commit -am "Message here: commits all changes on files that are tracked"
  ```

  This will take ALL the files you have edited and package them up together.

- `git add`: If you would like to control which files are included in the commit, you can use `git add` together with a `commit -m "Message"` (without -a option, to commit only the files that you added explicitly. For example,

  ```
  git add File1 File2 File3
  git commit -m "Message here: commits only changes in File1 File2 File3"
  ```

  will add and commit only files `File1`, `File2`, and `File3`.

- `git push`: The last git command you will need to learn is `git push` which takes your code and submits it to github.mit.edu where we collect your labs.

  ```
  git push
  ```

You may submit your code as many times as you like. We urge you to submit often so that your work is backed up in Git, and so that the course staff can see your latest version of your code when trying to answer your questions on piazza.

You can see the status of your latest lab submission by going to the course Didit website, clicking on your repo name, followed by clicking on your build. Each build shows you exactly which tests have passed and which have failed.

4 Editting files

The repos that you clone from github will have code templates that you will need to fill in in order to complete the lab. In order to modify these templates, you will need to use an editor. There are multiple editors available on athena including: vi, vim, emacs, and nano. If you are new to using text editors, then you will probably want to use nano which is a basic text editor which lists commonly used commands at the bottom of the editor. If you would like to use vi/m or emacs you may want to do a search for their most commonly used commands on the web.

You may also find our short bash and git tutorial helpful in manipulating your files on athena.

As you get into the later labs (lab 3 on), you will be writing minispec code. There are configuration files available to help properly highlight these files in your editors. See the course resources page (Minispec syntax setup section) for details.

5 Setup for local development

While most of the tools used for 6.004 are only available on Athena, it is possible to write code locally using your favorite text editor and then sync your files to Athena.

5.1 Enable syntax highlighting (optional)

If you want to enable syntax highlighting, you can acquire the files at the course resources pages (Minispec syntax setup) section.

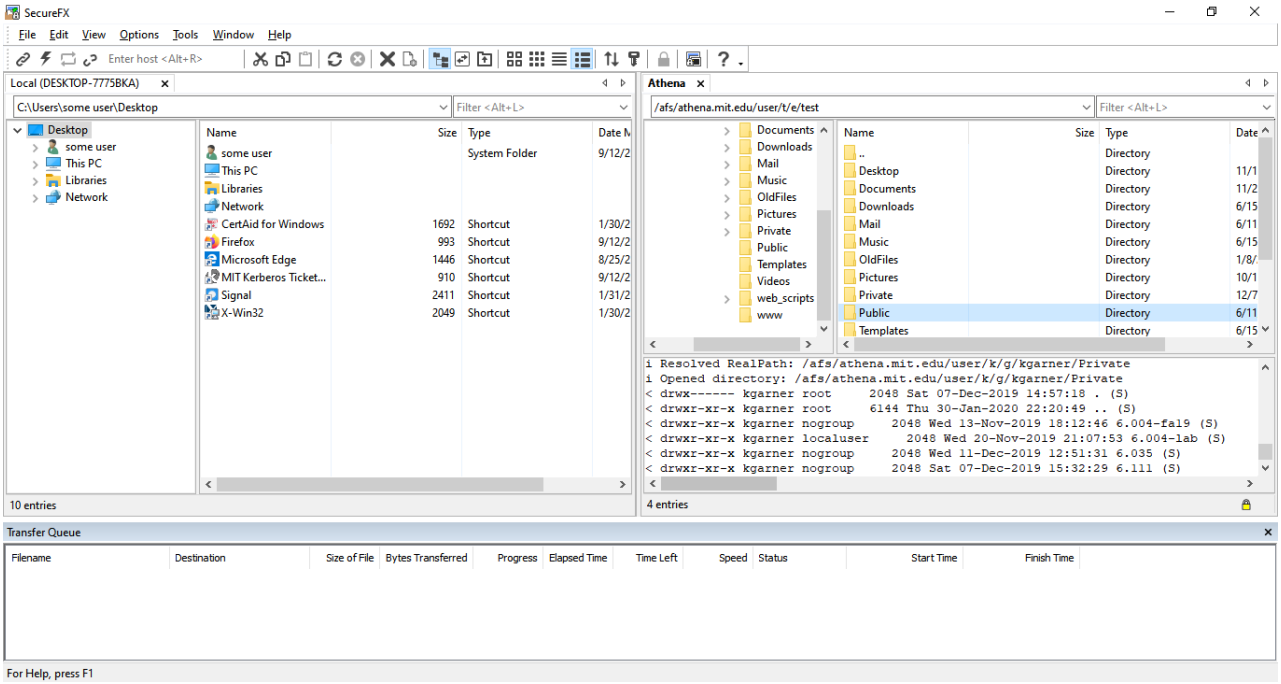
5.2 Syncing files with Athena (Graphically)

To sync files to Athena from your computer, you have a couple of options: graphically and through the command line.

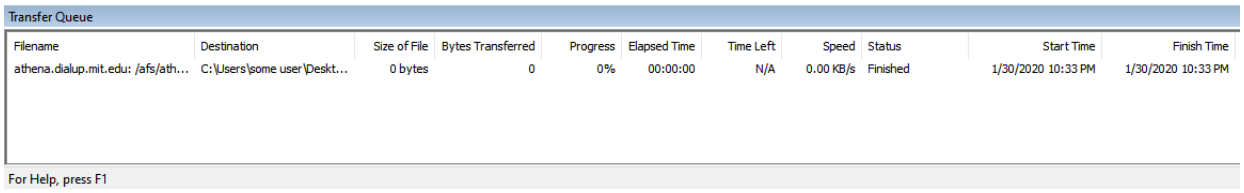
5.2.1 Windows

One of the easiest tools you can use to sync files between Athena and your Windows computer is SecureFX. This comes installed with **SecureCRT**.

Once you have installed and authenticated into SecureFX, you should see a screen similar to below:



On the left, you will see the directories that are on your computer, and on the right, you will see files and directories that are on Athena. To copy files from your computer to Athena or vice-versa, you can just drag files from one window to another. When copying files, you should see information about the progress in the Transfer Queue.



5.2.2 MacOS

On MacOS, you can install Fetch to sync files between Athena and your computer:

1. **Download Fetch** (credentials required).
2. **Install Fetch**. Follow the Installation instructions (certificates required).
3. **Start Fetch**. You can follow Using Fetch at MIT for examples.

5.3 Syncing files with Athena (Command line)

If you would prefer to use the command line, there are a few options: **SFTP** and **SCP**.

5.3.1 Using SFTP

SFTP (SSH File Transfer Protocol) is a tool for sharing files via SSH. It can be used in a similar fashion as SSH. The nice thing about SFTP is that it presreves the session (so long as you are connected).

1. **Open SFTP**. Where this is found depends on the operating system you run:
   - **Windows:** Open the Command Prompt. This can be found by opening the Start menu and searching for **Command Prompt**.

- **MacOS:** Open the **Terminal** app. This can be found in **Applications**.
- **Linux:** Open any Terminal application.

2. Run the following command: `sftp {kerberos}@athena.dialup.mit.edu` and replace {kerberos} with your Athena username. You will be asked to provide a password and login with Duo.
3. When you see **sftp>**, you are in the SFTP shell and can run SFTP commands (described below).

SFTP Commands

- `?` gets a description of all commands
- `ls {path}` lists files and directories on Athena. You can provide a file path to list the files in that directory.
- `lls {path}` lists files and directories on your computer. You can provide a file path to list the files in that directory.
- `get -r {path}` downloads a file from Athena to your computer. For directories, you should provide the **-r** option to download all files in a directory.
  Sample command: **get -r test.txt**
- `put -r {path}` uploads a file to Athena from your computer. For directories, you should provide the **-r** option to download all files in a directory.
  Sample command: **put -r test.txt**
- `cd {path}` changes the current directory on Athena. For example, you can run **cd Desktop**.
- `lcd {path}` changes the current directory on your computer. For example, you can run **lcd Desktop**.

5.3.2 SCP

If you prefer to copy files using a single-line command instead of **sftp**, you can also use the **scp** command in your local terminal or prompt.

The general syntax to **upload** a file is as follows:

```
scp {local path} {kerberos}@athena.dialup.mit.edu:{remote path}
```

For example, to upload the file **temp.txt** in your local current directory to **temp.txt** in your home directory in Athena, run:

```
scp temp.txt {kerberos}@athena.dialup.mit.edu:temp.txt
```

To **download** a file from Athena, the general syntax is as follows:

```
scp {kerberos}@athena.dialup.mit.edu:{remote path} {local path}
```

For example, to download the file **discussions.txt** in your lab1-|{kerberos}| directory, which is in your Athena home directory, to **discussions.txt** in your local current directory, run

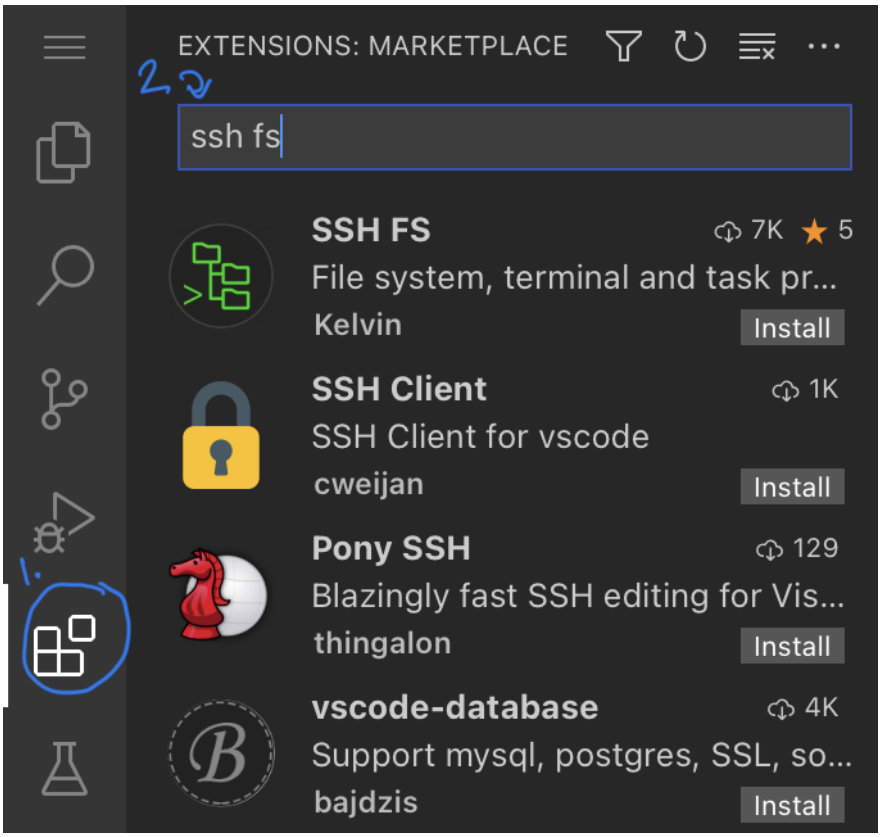
```
scp {kerberos}@athena.dialup.mit.edu:lab1-{kerberos}/discussions.txt discussions.txt
```

Note that if the destination file exists in the destination directory, **scp** will overwrite the file.

5.4 Syncing files with Visual Studio Code (VSCode)

This section describes some features of VSCode that allow you to edit your code in your local VSCode editor and sync files with Athena directly without having to manually transfer files back and forth.

1. Install the SSH FS extension. Click on the extensions button (1), and search for "SSH FS" and install the one developed by Kelvin.

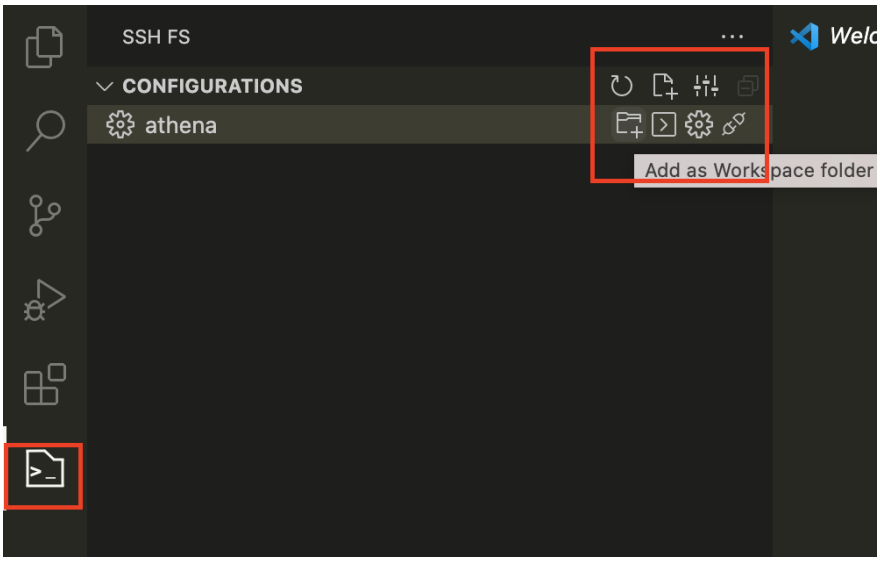


2. Open SSH FS and click on **New Configuration**. You may give the configuration a name of your choice (e.g., Athena).

3. Edit the configuration to populate the fields as shown below:

   - Host : `athena.dialup.mit.edu`
   - Root : `~/`
   - Username : `YOUR_KERBEROS`
   - Password : Should remain blank since this passwrod field is not stored safely

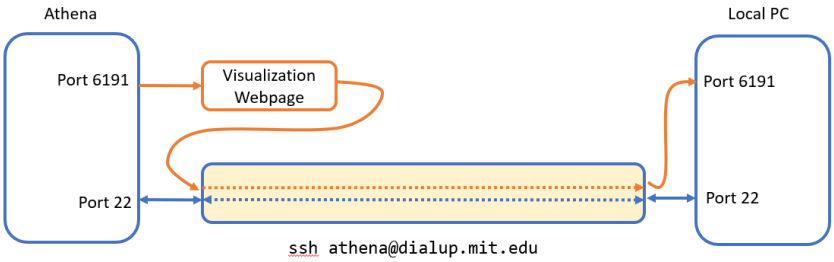
4. Once configured, click on **Add as Workspace folder** (this is the first icon that shows up if you hover over your new configuration) and follow the prompt to authenticate by providing your Kerberos password and via Duo. Note that you may be asked to authenticate twice.



5. Once authenticated, you should be able to use the VSCode File Explorer to edit Athena files directly.
6. If you need to download an Athena file to your local machine (e.g., an HTML file produced by `visual` or an SVG file produced by `synth`), you need simply right-click on the file in File Explorer and choose **Download...**.

6 Setup Port Forwarding (Optional)

In certain parts of the lab we will be doing in this class, we will require some visulization, which would require us to browse an HTML file from the Athena server. Our tools can setup visualization server on Athena which will serve the visualization web page. In order to access such web page, we need to setup a **Port forwarding** to allow our local PC to connect to the Athena web server. Port forwarding is a feature that allow Local PC to connect to a temporary web server launched on Athena through the initial SSH connection that we used to log into Athena. The mechanic of how Port forwarding work is summarized in the picture below.



To exapnd further, the SSH server on Athena will pull all network packet sent on the port 6191 (or other ports) on Athena and send it through the same SSH connection that we used for connecting to Athena. Subsequently, the SSH client on our local machine will then forward the file to the corresponding port (In this case port 6191) on our local machine, which we can view by typing `localhost:6191` in your browser.

6.1 Port-forwarding for Command line SSH connection

If you connect to Athena thourgh SSH command line (through windows command prompt, Mac Terminal, or linux terminal), we can setup port forwarding through the following steps.

Port-forwarding by configuring SSH mid-session

1. Once you are connected to Athena, press enter and type `~C` to open an SSH configuration shell.
2. Enter

   ```
   -L 6191:127.0.0.1:6191
   ```

   to forward the web page over ssh from athena to your computer via port forwarding.
3. Press the enter key twice to exit from SSH configuration shell.
4. You can now continue using Athena like usual, and anytime you run a visualization server on Athena, you can open `http://localhost:6191/` in your local web browser on your computer to view the visualization served on Athena.

Port-forwarding by configuring initial connection

You can also add certain command line option to allow port forwarding when you first connect to Athena. You can connect to Athena using the following command.

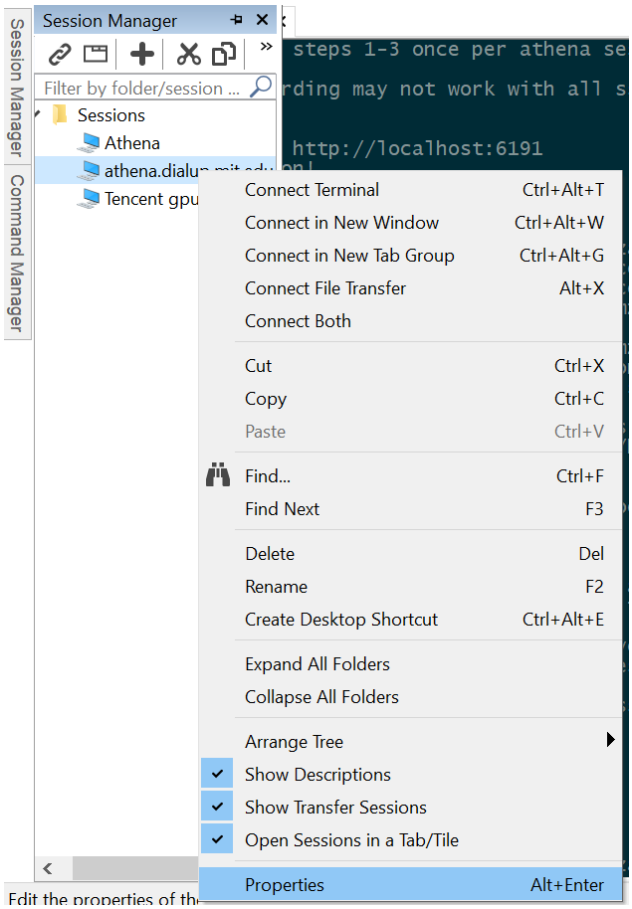
```
ssh -L 6191:127.0.0.1:6191 {kerberos}@athena.dialup.mit.edu
```

Once you are authenticated through password and Duo, the port forwarding will be automatically established.

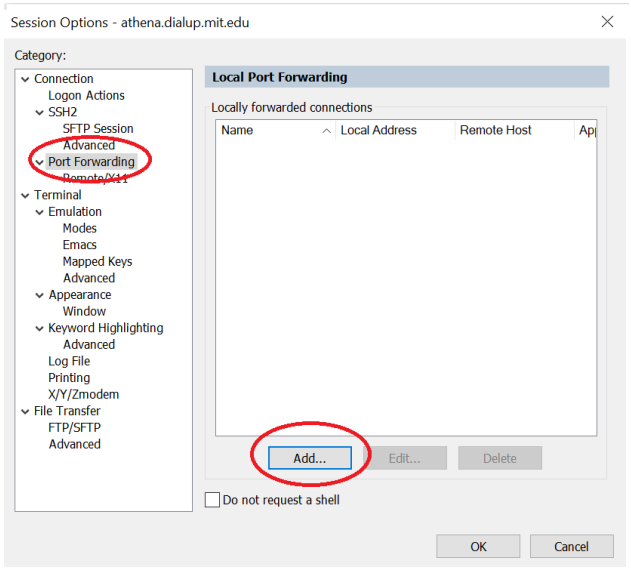
6.2 Port-Forwarding for SecureCRT

To enable Port forwarding in secureCRT, you need to perform the following steps.

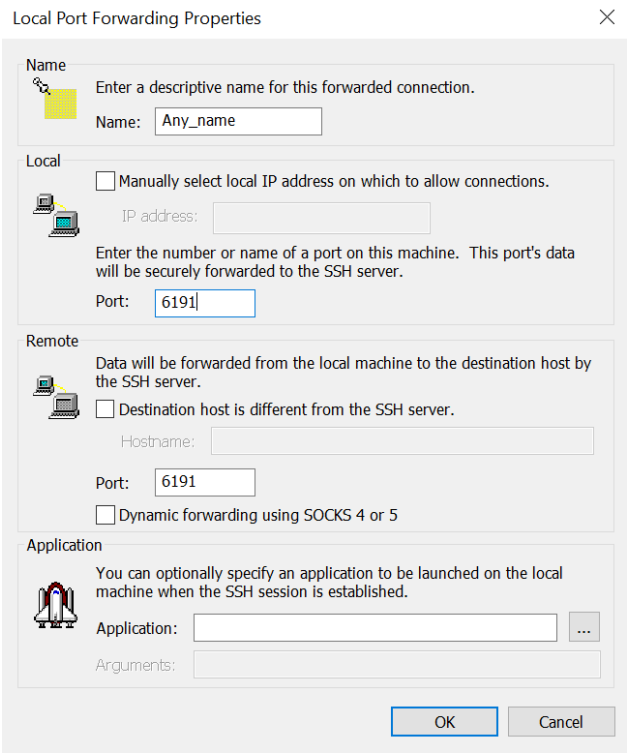
1. Right click on the connection that you use to connect to Athena in the Session Manager tab. Then click property.



2. Click Port Forwarding on the left window and click Add button as shown in the picture below



3. Add the port forwarding property according to the following picutre. Name the port forwarding connection with any namethat you want. The important part is to make sure that both Local Port and Remote Port are 6191.



4. Click OK to close the port forwarding windows, and continue to connect to SSH through the same secureCRT session. Once authenticatd, the port forwarding willbe automatically established.

7 Tools for Unreliable Internet Connections (optional)

An unreliable internet connection can make it difficult to maintain an SSH connection to Athena. In this section, we will install a tool called Mosh (mobile shell) which will keep the SSH connection alive even if the network connection fails temporarily. Please follow the instructions in each section below to install, configure, and start a Mosh connection.

7.1 Installing Mosh

To use Mosh, it first needs to be installed on your local machine. Follow the instructions below that correspond to your operating system.

7.1.1 Windows 10

Sadly, there is no Window-specific mosh available yet (The Google chrome version does not support logging into our Athena server). Right now, our best solutions are to enable Linux subsystem for Window and install mosh on Linux.

You can enable Window subsystem for Linux by following these steps from here

1. Open PowerShell as Administrator and run this exact command:

   ```
   Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
   ```

2. Restart your computer when prompted.
3. Go to your Microsoft Store and Download Ubuntu 18.04 LTS
4. Run Ubuntu 18.04 LTS app from Start Menu and Follow the Linux installation guideline from subsubsection 7.1.3

7.1.2 MacOS

If you have a package manager installed on your Mac such as Homebrew or MacPorts, you can simply use that to install Mosh.

- Homebrew: `brew install mosh`
- MacPorts: `sudo port install mosh`

Alternatively, if you do not have a package manager, you can download and install the .pkg file from the mosh installation website.

7.1.3 Linux

Use the package manager for your distribution of Linux. For example, if you are on Debian or Ubuntu, run

```
sudo apt-get install mosh
```

For a more extensive list of distribution options, check out the mosh installation website.

7.2 Configuration

Mosh needs to be enabled on Athena before you can connect to it. To do so, you will need to add the `mosh_project` locker. First SSH into Athena as usual:

```
ssh {kerberos}@athena.dialup.mit.edu
```

Then run the following command to update your .bashrc file:

```
echo "add mosh_project" >> /mit/{kerberos}/.bashrc
```

After doing so, you can close the SSH connection.

7.3 Staring a Mosh Session

The instructions in the previous two sections only need to be completed once. This section describes the steps you will take each time you want to start a Mosh session.

To use Mosh, simply run the following command on your local machine:

```
mosh {kerberos}@athena.dialup.mit.edu
```

You will be prompted to log in with your Kerberos password and 2-factor authentication. Afterwards, this session should remain persistent even if the network connection drops.

7.4 Limitations and Issues

There are a couple issues to be aware of when working with Mosh:

- If you use Mosh and leave an SSH connection open for a long time (over 10 hours), your Kerberos tickets will expire, and you will be unable to access any of your files. For example, `ls` will be unable to display information about the files, and you will not be able to write text to your files. To fix this, run the command `renew` to renew your Kerberos tickets. You will be prompted for your Kerberos password, but you will not need 2-factor authentication.
- Mosh does not work with Port forwarding yet. Therefore, to do a part of lab that requires visualization, you might need to download the webpage through file transfer instead of viewing it using port forwarding.