Remote access and run Jupyter Notebook using SSH without port forwarding on router

A personal note
Created Date: 2021-Nov-22
Last Edited: 2022-Sept-20
Author: MT

1 Purpose

The purpose of this note is to record the steps of running a Jupyter Notebook session via remote access using SSH without the need of setting up port forwarding on a router. This will be useful if one has no access to the router's administration page.

2 Background Setting

We want to remotely run a Jupyter Notebook session on the `HOME` Ubuntu computer by using the `WORK` Ubuntu computer that is not within the same network as `HOME`. In this note, the `HOME` computer can be either a headless Linux server, or a desktop machine with a monitor, keyboard, mouse, or other peripherals. The `WORK` computer is usually a laptop (or a PC) that one may carry with her/him when working outside.

The solution includes the use of a **free** AWS (Amazon Web Services) cloud Linux server (although other cloud services are also applicable as long as it can hold a Ubuntu or a Linux server).

3 The Workflow

The workflow includes the following procedures:

0. (Prerequisite) Create a Ubuntu AWS instance, or any other cloud server (A post on how to create a free AWS cloud server can be found in Adrian's post). In this case, the version of Ubuntu used is 20.04 LTS (this will also work for other versions of Ubuntu and likely other distributions of Linux). Installing the `openssh-client` and `openssh-server` applications on the cloud server (if it has not been done before) by running the following two commands in the terminal (one guide on the installation):

```
    sudo apt install openssh-client
    sudo apt install openssh-server
```

1. Connect the `HOME` computer to the `CLOUD` server by *reverse SSH Tunneling* (see this post on Reserve SSH Tunneling). Run the following command on the `HOME` machine:

```
ssh -R 19999:localhost:22 -i ~/.ssh/<key.pem> cloud@<cloud_public_ip>
```

Note: port 19999 could be any unused port on the `HOME` machine; 'cloud@' should be replaced by the name of the `CLOUD` server followed by the address declaration symbol '@'; and the contents within the angle brackets <> should be replaced by the corresponding paths.

2. Connect from `CLOUD` to `HOME` through SSH tunneling. Run the following command on the `CLOUD` Linux server:

   ```
   ssh home@localhost -p 19999
   ```

   Note: again, 19999 should be the same port ID that we occupied in step 1; and, 'home@' should be replaced by the user name of the `HOME` machine followed by the address declaration symbol '@'.

3. Starts a Jupyter Notebook session on the `HOME` machine (choose one of the following two scenarios):

   3.1. either directly (i.e., locally) on the `HOME` computer, if one has access to the `HOME` computer (Read Section 3.1 for detailed steps);

   3.2. or remotely by using the `WORK` computer first connect to the `CLOUD` server then connect to the `HOME` computer, if one has no access to the `HOME` machine (Read Section 3.2 or Section 3.3 for detailed steps).

   In either case (3.1 or 3.2), the Jupyter Notebook URL with authentication token should be recorded/copied, as it will be used in Step 6.

4. Open up a new terminal window and connect the `WORK` computer to the `CLOUD` server again by running the following commands on the `WORK` computer (read also Pranav's post for more details):

   ```
   ssh -i ~/.ssh/<key.pem> cloud@cloud_public_ip
   ```

   Note: 'cloud@' should be replaced by the user name of the `CLOUD` server followed by the address declaration symbol '@'; and, the contents within the angle brackets <> should be replaced by the corresponding paths.

   After connecting to the cloud server, we run the following command:

   ```
   ssh -NL 1234:localhost:1234 home@localhost -p 19999
   ```

   Note: port 1234 here should be the same port used in Step 3; port 19999 here should be the same port used in Step 1.

5. Open up a new terminal in `WORK` and run the following command:

   ```
   ssh -NL 1234:localhost:1234 -i ~/.ssh/<key.pem> cloud@<cloud_public_ip>
   ```

   Note: port 1234 here should be the same port used in Step 3; 'cloud@' should be replaced by the user name of the `CLOUD` server followed by the address declaration symbol '@'; and the contents within the angle brackets <> should be replaced by the corresponding paths.

6. Finally, paste the Jupyter Notebook URL with authentication token to a browser (e.g., Firefox or Google Chrome). The connection should be established successfully.

3.1 Starting Jupyter Notebook on the HOME computer locally

If one has access to the HOME machine, to start a Jupyter Notebook locally without the need of showing the session in a browser, run the following command (read also Pranav's post for more details):

```
jupyter notebook --no-browser --port 1234
```

Note: port 1234 here can be any unused port; if the Jupyter Notebook is installed in an Anaconda environment, one should first activate the correct conda environment and then run the above command.

3.2 Starting Jupyter Notebook on the HOME computer remotely

If one has no access to the HOME computer, we need to first make a connection between the WORK computer and the HOME computer, and then we start a Jupyter Notebook session remotely.

In this scenario, we first run the following command on the WORK computer to connect to the cloud:

```
ssh -i ~/.ssh/<key.pem> cloud@cloud_public_ip
```

Note: 'cloud@' should be replaced by the user name of the CLOUD server followed by the address declaration symbol '@'; and, the contents within the angle brackets <> should be replaced by the corresponding paths.

Then, we connect from CLOUD to HOME by running the command:

```
ssh home@localhost -p 19999
```

Note: port 19999 should be the same port ID that we occupied in step 1; and, 'home@' should be replaced by the user name of the HOME machine followed by the address declaration symbol '@'.

After connecting to the home server, we can start a Jupyter Notebook session under an anaconda environment by running the command:

```
jupyter notebook --no-browser --port 1234
```

Note: port 1234 here can be any unused port.

3.3 Starting Jupyter Notebook on the HOME computer remotely in one command

As an alternative to the first two commands in Sec. 3.2, we can establish a connection from WORK to HOME in one line of command. On the WORK computer, run the following command:

```
ssh -t -i ~/.ssh/<key/pem> cloud@<cloud_public_ip> ssh home@localhost -p
   19999
```

Note: port 19999 should be the same port ID that we occupied in step 1; 'cloud@' should be replaced by the user name of the CLOUD server followed by the address declaration symbol '@'; 'home@' should be replaced by the user name of the HOME machine followed by the address declaration symbol '@'; and, the contents within the angle brackets <> should be replaced by the corresponding paths.

After connecting to the home server, we can start a Jupyter Notebook session under an anaconda environment by running the command:

```
jupyter notebook --no-browser --port 1234
```

Note: port 1234 here can be any unused port.

4 Beyond The Above Notes

In the above workflow, Steps 1 and 2 should be run upfront (i.e., not related to the `WORK` computer), and one would start with Step 3 when one would want to connect from `WORK` to `HOME`. That is starting from Step 3.2, all the commands will be run remotely on `WORK` without the need for physical access to `HOME`.

However, one should note that the above workflow relies on a continuous connection between `HOME` and `CLOUD`, otherwise, one would need to run Steps 1 and 2 again when the connection is broken. One solution (beyond this document) is to write a bash script to periodically check the connection between `HOME` to `CLOUD` and perform Steps 1 and 2 automatically when the connection is dropped by first terminating/killing the previous connection and starting over a new `HOME-CLOUD` connection.