Today, we are going to talk about Python and Jupyter Notebooks. This will be the first of a few hands-on sessions discussing the basics of data science leading up to analyzing our own honeypot data.

We will start by learning what Jupyter Notebooks are and how to get started with them, and then we will fill in the blanks on a pre-made notebook to learn the basics of Python.

1 Jupyter Notebooks

Jupyter is a web application that allows us to mix code, text, and images into a single, dynamic document, called a "Notebook". From a data scientist's point of view, Jupyter Notebooks are useful since they allow us to execute code in segments. This becomes important when working with large amounts of data. We can execute long-running parts of our code, such as loading in our data set, only once and then make changes to our analysis code very quickly.

1.1 Installing Jupyter

To install Jupyter Notebooks to our system, we are going to use Python's package manager "pip". Pip works in a very similar way to the Linux package manager, `apt`, that we talked about last week. However, we must first install pip using the following command:

```
user@wse380-22fa:~$ sudo apt install python3-pip
```

Notice how we specify that we want to install the Python3 version of pip. Python currently has two major versions in use: 2 and 3. Because official support for Python2 has already ended, it is important to use Python3. Thus, when we try to install a package using pip, we must make sure we are using the Python3 version, pip3. Use the following command to install Jupyter for Python3 and then resolve a broken dependency:

```
user@wse380-22fa:~$ pip3 install jupyter
user@wse380-22fa:~$ pip3 install markupsafe==2.0.1
```

We use the `install` command with the package name `jupyter` to install the latest version. Attempting to run `jupyter` from the command line should display a help message, signifying a successful install. However, if we try to do so now, we will encounter an error from the command line telling us that it does not know what `jupyter` is. We need to tell the command line where to find the `jupyter` program because we installed it in a local, user-specific directory. Run the following, and then try executing `jupyter` again.

```
user@wse380-22fa:~$ echo 'export␣PATH="$PATH:$HOME/.local/bin"' >> ~/.bashrc
user@wse380-22fa:~$ source ~/.bashrc
user@wse380-22fa:~$ jupyter
usage: jupyter [-h] [--version] [--config-dir] [--data-dir] [--runtime-dir]
               [--paths] [--json]
               [subcommand]
```

1.2 Starting a Jupyter Notebook Server

Now that we have successfully installed Jupyter to our computers, we can start it up to create or open an existing notebook. When we start Jupyter, it will use the current directory as it's project directory. Therefore, we want to first create a new directory so we can keep all of our notebooks and data together. We will use a couple of the commands we learned last week to do so:

```
user@wse380-22fa:~$ mkdir notebooks
user@wse380-22fa:~$ cd notebooks
```

Here, we simply created a new directory called "notebooks" with the `mkdir` command, and changed our current directory to it. Our last step before starting Jupyter would be to place any notebooks we already have and would like to edit into this directory. Once we have done that, we can start Jupyter with the following command:

```
user@wse380-22fa:~/notebooks$ jupyter notebook
[I 20:24:53.114 NotebookApp] Writing notebook server cookie secret to...
[I 20:24:53.285 NotebookApp] Serving notebooks from local directory:...
[I 20:24:53.285 NotebookApp] Jupyter Notebook 6.4.8 is running at:
[I 20:24:53.285 NotebookApp] http://localhost:8888/?token=...
[I 20:24:53.285 NotebookApp]  or http://127.0.0.1:8888/?token=...
[I 20:24:53.285 NotebookApp] Use Control-C to stop this server...
[W 20:24:53.288 NotebookApp] No web browser found:...
[C 20:24:53.288 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/user/.local/share/jupyter/runtime/...
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=...
     or http://127.0.0.1:8888/?token=...
```

As you can see, running this command prints a bunch of information to the terminal, mostly describing the runtime parameters and how to access the notebook using a browser. However, because we are **running Jupyter on a remote machine in the cloud**, we do not have a graphical user interface (GUI) with which to interact with the notebook. Thus, we need to establish a method by which a browser on our local machine can connect to the Jupyter server on the remote machine. We can do this securely by establishing a local port forwarding tunnel using SSH.

1.3 Connecting to the Remote Jupyter Server

The command we will be using for local port forwarding over SSH is as follows:

```
user@wse380-22fa:~$ ssh -L 1234:localhost:5678 foobar@1.2.3.4
```

This instructs our SSH client to establish a connection to an SSH server at IP address `1.2.3.4` on user `foobar`, and then forward all traffic we receive on our local machine on port `1234` to `localhost:5678` on the (remote) SSH server. Thus, to access our remote Jupyter server, we will execute the following command in a seperate terminal:

```
user@wse380-22fa:~$ ssh -L 8888:localhost:8888 foobar@1.2.3.4
```

which will forward all local traffic on port `8888` to `localhost` on port `8888` (i.e., our Jupyter server) on our remote machines. Next, go back to the output from the `jupyter notebook` command, copy one of the URLs, and paste it into a browser on your local machine. You should now be connected to the remote Jupyter server!

This page displays your current directory. If you placed an existing notebook here before starting, you can open it by clicking on its name. Otherwise, you can create a new notebook using the ”Create” button the top right, and then selecting ”Python 3” as the type of notebook. Either of these options will open the notebook in a new browser tab.

1.4 Example Notebooks

Let us turn our attention to some example Jupyter Notebooks to become more familiar with both Python and Jupyter. We will need to move our example notebooks into the Jupyter Notebook directory so that we can interact with them. Open a new terminal, connect to the machine, and execute:

```
user@wse380-22fa:~$ cp wse380Lessons/Session2/*.ipynb notebooks/
```

which will copy all the files in the directory `wse380Lessons/Session2` with an extension of `.ipynb` (the Jupyter Notebook extension) to our Jupyter Notebook directory. Switch back to your browser tab that is connected to the remote Jupyter server, refresh the notebook list, and you should now see the two notebooks `IntroToPython.ipynb` and `IntroToPandas.ipynb`. Feel free to follow along on your own device and let us know if you have any questions or run into any problems!