This document is intended to be an instructional guide to our usual workflow when running analyses in remote Jupyter notebooks. I encourage you all to review the introductory content to Python and Jupyter in Session 2.

1 Establishing SSH Connections

We need to establish an SSH connection so that we can run `jupyter notebook` and forward the connection to our local computer to access the Jupyter software running on the remote machine. I recommend keeping another terminal open so that you can use the command line and leave Jupyter open. We can run the following:

```
# in terminal 1 (for jupyter)
user@local-machine:~$ ssh -L 8888:localhost:8888 -p 2905 remote-user@1.2.3.4
remote-user@1.2.3.4:~$ cd notebooks/
remote-user@1.2.3.4:~$ jupyter notebook
# and then copy-paste in your browser (or click the link to open it)

# in terminal 2 (to use the command line)
user@local-machine:~$ ssh -p 2905 remote-user@1.2.3.4
remote-user@1.2.3.4:~$
```

2 Preparing Session Notebooks

All the content for the hands-on sessions are in the folder `wse380Lessons` in the user home directory. To prepare for any Session X, do the following:

```
# in terminal 2
## fetch any updates and return to home
remote-user@1.2.3.4:~$ cd wse380Lessons && git pull
...
remote-user@1.2.3.4:~/wse380Lessons$ cd ..

## list the content of the directory to see if there is a notebook
remote-user@1.2.3.4:~$ ls ~/wse380Lessons/SessionX
<lesson_plan>.pdf <notebook>.ipynb requirements.txt

## install any required libraries if there is a *.ipynb file
remote-user@1.2.3.4:~$ pip3 install -r \
    ~/wse380Lessons/SessionX/requirements.txt

## make a new copy of the notebook if you
## have not yet (skip if you already have the notebook for this session)
remote-user@1.2.3.4:~$ cp ~/wse380Lessons/SessionX/*.ipynb ~/notebooks/
```

You should now be able to see the notebook in the browser tab that you have connected to the Jupyter server! To run it, just click on the name and it should open in a new tab.

3 Running Notebooks

To run a notebook, you can click into a code cell and hit `Shift+Enter` to run the cell and move to the next cell. Alternatively, you can use the `Run` menu toolbar. You may have to change parts of the code to reflect your setup. Most commonly, we needed to do the following:

- change variables with file paths because the home directory will depend on your username (e.g., change `/home/user` to `/home/remote-username` for new notebooks)
  - also, we needed to change the file path for the geolocation database

```
# change this from
reader = maxminddb.open_database('GeoLite2-City.mmdb')
# to
reader = maxminddb.open_database(
    '/home/remote-user/wse380Lessons/Session3/GeoLite2-City.mmdb'
)
```

- export log files from the low-interaction honeypot that we configured using Docker. Note that this honeypot has been stopped because we are currently running the medium-interaction honeypot `cowrie`, but you can stop `cowrie` as described in Session 4 and restart the low-interaction honeypot if desired as described in Session 1. We were able to exports the file by running

```
# in terminal 2
## find out what the name of the docker container is
remote-user@1.2.3.4:~$ sudo docker ps -a
CONTAINER ID    IMAGE      COMMAND    CREATED    STATUS
    PORTS      NAMES
... justinazoff/ssh-auth-logger ... ... ... ... name-1

## export the logs from stderr to ~/ssh_logs.json
remote-user@1.2.3.4:~$ sudo docker logs \
    name-1 > ~/ssh_logs.json 2>&1
```