**The Data Science Toolkit**

**Goals:**

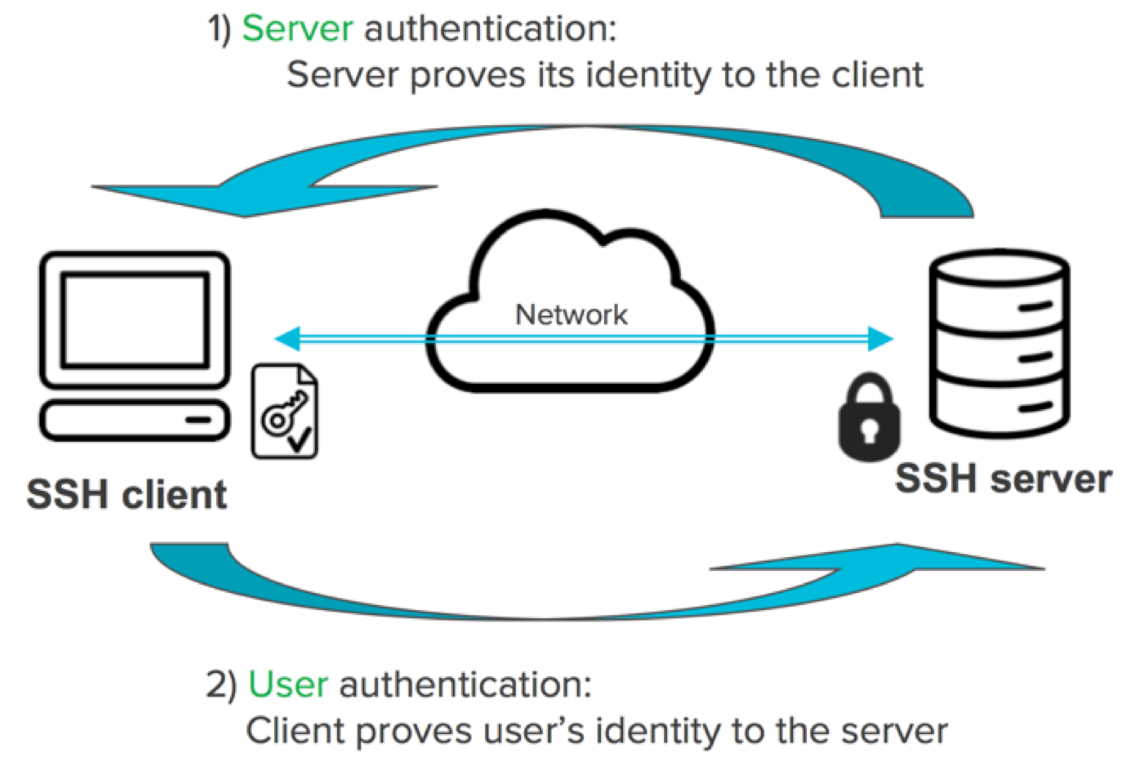
Create a detailed outline of the steps required to configure your own Jupyter Data Science Notebook Server on Amazon Web Services. The idea is that you would be able to revisit these instructions some arbitrary amount of time from now and be able to bring the server online without consulting any additional documentation.

**Steps:**

1. First create AWS account by signing up online at aws.amazon.com (choose free or paid version based on needs).
2. Sign into the AWS console. Use the region closest to you geographically.
3. Under AWS Services, we will **configure EC2 (Elastic Compute Cloud)**
4. Set up Key Pairs and Security groups (not use the default security group)
5. Download gitbash from <https://git-scm.com/download/win>
   1. Use **ssh-keygen** to generate public and private ssh keys
   2. Save in default directory with no passcode. This generates id\_rsa and id\_rsa.pub in the .ssh folder of home directory
   3. Confirm the two keys in default directory using ls ~/.ssh
   4. We want the contents of the public key to upload to AWS, so we type

cat ~/.ssh/id\_rsa.pub and copy the content of the public key

* 1. **Understanding SSH**:
     1. Using ssh users (client) can log into server without password. There is a set of two keys, one is public and other is private. The public key is uploaded to the server and acts as the lock. The private keys stays with the client. The communication is made using the keys.



1. Go back to AWS Console and Under Key Pair 🡪import keypair 🡪 name the keypair and paste the content of the public key that we got from bash (note: when using on windows it is recommended to first copy on notepad, eliminate spaces and then upload). Now our keypair is set up.
2. Now configure **security group** to connect to our system via ssh (port 22) and http (port 80)
3. Add rule🡪ssh🡪source (anywhere) and name the security group
4. AWS is now configured for launching our Jupyter server.
5. We launch **AWS** **instance**. It is a 7 step process.

* Step 1: choose AMI (template that has the configuration required to launch our instance). we choose ubuntu server 16.4
* Step 2: choose t2.micro with 1 CPU and 1GB Ram
* Step 3: configure instance🡪use default setting
* Step 4: add storage🡪30 GB of space
* Step 5: add tags🡪none
* Step 6: add security group🡪select existing one created using steps #7 and #8
* Step 7: review and launch🡪review all requirements are met🡪launch🡪select the keypair we created in #6
* View Instance🡪shows details of running instance and we will use the public ip address from here

1. We will now **install docker** to manage our jupyter nobebook
2. Go to bash and type ssh [ubuntu@52.53.182.225](mailto:ubuntu@52.53.182.225) (ubuntu is the user on EC2 instance @public ip address from our instance). The keypairs we configured will match with our system and we will connect.
3. Say yes to security warning as we are connecting for the first time. We are now connected to ubuntu@privateip
4. We used ssh to connect to AWS system and now we will install docker using

Curl -sSL <https://get.docker.com> | sh (pipe directly into shell)

1. We now associate ubuntu user to docker user group using

sudo usermod -aG docker ubuntu. Log out (exit) and log back in (ssh [ubuntu@52.53.182.225](mailto:ubuntu@52.53.182.225)) for this to take into effect.

Verify everything works using docker -v (shows the version of docker)

1. We pull jupyter image that we use to run out jyputer server notebook using

docker pull jupyter/datascience-notebook

1. Verify the pull using **docker images** that shows the image id and size
2. We now run the image using docker run -v /home/ubuntu:/home/jovyan -p 80:8888 -d jupyter/datascience-notebook

(volume command to write to aws connecting jovyan user directory to port 80 for AWS and jupyter defaults to port 8888 running in detached mode)

1. We get a long string that is **container** id, we get container id using docker ps
2. To get the token to log into the server via browser use docker exec “container id” jupyter notebook list. Copy the token.
3. Access the server in the browser, type public ip address and paste the token to log in
4. We now see our jupyter notebook server running and we can launch Julia, Python or R notebook and save contents.

**Security Concerns:**

Since access to the Jupyter notebook server means access to running arbitrary code, it is important to restrict access to the notebook server. For this reason, notebook 4.3 introduces token-based authentication that is **on by default**

As Jupyter notebooks become more popular for sharing and collaboration, the potential for malicious people to attempt to exploit the notebook for their purposes increases. IPython 2.0 introduced a security model to prevent execution of untrusted code without explicit user input.

When a notebook is executed and saved, a signature is computed from a digest of the notebook’s contents plus a secret key. This is stored in a database, writable only by the current user. Each signature represents a series of outputs which were produced by code the current user executed, and are therefore trusted.

When you open a notebook, the server computes its signature, and checks if it’s in the database. If a match is found, HTML and Javascript output in the notebook will be trusted at load, otherwise it will be untrusted.

**A detailed budget of the costs of running a Jupyter Data Science Notebook Server for three months using at least three different kinds of EC 2 instances**



**Schema**

Local Machine port 80

Jupyter port 8888

Dockers

Ubuntu

Id\_rsa

Id\_rsa.pub

ssh

AWS