# AWS- Install a VM for an Indy Node

Introduction: The following steps are one way to adhere to the Indy Node guidelines for installing an AWS instance server to host an Indy Node. For the hardware requirements applicable for your network, please refer to the steward technical requirements document or the network governance documents for your network.

1. From the AWS EC2 services page, click 'Instances'
2. Click 'Launch Instance'
3. Step 1: Choose AMI
   1. Click the second dropdown arrow in the upper right corner and select a region to run your VM in. Recommended to select the region matching the jurisdiction of your companies corporate offices.
   2. Type “ubuntu” in the search field and hit ‘Enter’ key.
   3. Click 'Select' on the 'Ubuntu Server 16.04 LTS (HVM), SSD Volume Type' AMI. (I had 64-bit (x86) selected, but that choice is up to you)
4. Step 2: Choose an Instance Type:
   1. Select a type with at least 8 vCPUs and 32G RAM or greater then click 'Next: Configure Instance Details'
   2. HINT: t3.2xlarge is sufficient, but you can choose a m or c class server if you prefer.
5. Step 3: Configure Instance Details
   1. Number of instances - 1
   2. Purchasing option - Do NOT request spot instances unless you are sure that it will not incur any possible downtime.
   3. Network - default
   4. Subnet - default (No preference)
   5. Auto-assign Public IP - default (Use subnet setting (enable) )
   6. Placement Group - leave unchecked
   7. Capacity Reservation - default
   8. IAM role - Your choice (default is None)
   9. CPU options - leave unchecked
   10. Shutdown behavior - Stop
   11. Enable termination protection - Your choice
   12. Monitoring - enable
   13. Tenancy - Your choice
   14. Elastic Inference - leave unchecked
   15. T2/T3 Unlimited - (if applicable) Enable (this shouldn't ever incur additional charges given current usage estimates)
   16. File systems - leave blank
   17. Advanced Details - Your choice (you can leave all values at defaults)
   18. Once all of the above is set to your liking, click 'Next: Add Storage'
6. Step 4: Add Storage
   1. Root volume - Your choice, defaults are fine
   2. Click 'Add New Volume'
      1. Volume Type - your choice (EBS is fine)
      2. Device - /dev/sdb
      3. Snapshot - your choice (default - Leave blank)
      4. Size - 1024 GiB
      5. Volume Type - your choice, Magnetic is sufficient (SSD is more expensive)
      6. Delete on Termination - your choice (I checked the box)
      7. Encryption - encryption not required.
   3. Click 'Next: Add Tags'
7. Step 5: Add Tags
   1. Tags are not required
   2. click 'Next: Configure Security Group'
8. Step 6: Configure Security Group
   1. Assign a security group - select 'Create a new security group'
   2. Security group name - your choice, I changed it to be ValidatorClient9702 for future ease of use and that name will be used throughout the rest of this guide.
   3. Description - your choice
   4. Change the default SSH rule that is already in the group.
      1. Source - To restrict SSH access to Admins only, add IP addresses for all admin users of the system in a comma separated list here. This part can be done later and instructions for doing so are included later in this guide.
      2. For Example: I selected My IP from the dropdown choices and elected to add more admins later.
   5. Click Add rule
      1. Type - Custom TCP Rule
      2. Protocol - TCP
      3. Port Range - 9702
      4. Source - Anywhere (ignore the warnings)
      5. Description - your choice (e.g. "Allow all external agents and clients to access this node through port 9702")
9. Click Review and Launch
   1. Check all values for accuracy
   2. Example:

AMI Details

**Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-03ffa9b61e8d2cfda**

Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Root Device Type: ebs Virtualization type: hvm

Instance Type

| Instance Type | ECUs | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance |
| --- | --- | --- | --- | --- | --- | --- |
| t3.2xlarge | Variable | 8 | 32 | EBS only | Yes | Up to 5 Gb |

Security Groups

Security group name ValidatorClient9701

Description ValidatorClient9701 created 2020-06-02T16:22:47.549-06:00

| Type | Protocol | Port Range | Source | Description |
| --- | --- | --- | --- | --- |
|  | | | | |
| SSH | TCP | 22 | 67.199.165.150/32 | SSH for admins |
| Custom TCP Rule | TCP | 9702 | 0.0.0.0/0 | Allow all external... |
| Custom TCP Rule | TCP | 9702 | ::/0 | Allow all external... |

Instance Details

| Number of instances | 1 |  |  |
| --- | --- | --- | --- |
| Network | vpc-019e476a |  |  |
| Subnet | No preference (default subnet in any Zone) |  |  |
| EBS-optimized | Yes |  |  |
| Monitoring | Yes |  |  |
| Termination protection | No |  |  |
| Shutdown behavior | Stop |  |  |
| Stop - Hibernate behavior | Disabled |  |  |
| Capacity Reservation | open |  |  |
| IAM role | None |  |  |
| Tenancy | default |  |  |
| T2/T3 Unlimited | Enabled |  |  |
| Host ID |  |  |  |
| Host resource group name |  |  |  |
| Affinity | Off |  |  |
| Kernel ID | Use default |  |  |
| RAM disk ID | Use default |  |  |
| Metadata accessible | Enabled |  |  |
| Metadata version | V1 and V2 (token optional) |  |  |
| Metadata token response hop limit | 1 |  |  |
| User data |  |  |  |
| Assign Public IP | Use subnet setting (Enable) |  |  |
| Assign IPv6 IP | Use subnet setting (Enable) |  |  |

Storage

| **Volume Type** | **Device** | **Snapshot** | **Size (GiB)** | **Volume Type** | **IOPS** | **Throughput (MB/s)** | **Delete on Termination** | **Encrypted** |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |  |
| Root | /dev/sda1 | snap-0af | 8 | gp2 | 100 | N/A | Yes | Not |  |
| ebs | /dev/sdb |  | 1024 | standard | N/A | N/A | Yes | Not |  |

1. Click ‘Launch’
2. A Key Pair selection window appears
   1. From the top dropdown, select 'Create a new key pair'
   2. Key pair name - Your choice (I selected the alias name that I will eventually assign to this Validator Node, opsnode)
   3. Click 'Download Key Pair'
   4. Copy the downloaded file to a "secure and accessible location" as indicated. I use a ~/pems directory for my .pems files.
3. Wait for the message “Your instances are now Launching” (Hint: This might take a few minutes)
   1. ! Launch Failed - This message appeared the first time I tried to launch a VM in a new region. After a few minutes (the time it took me to type this in) I clicked 'Retry Failed Tasks' and it worked fine.
   2. Read the information on the 'Launch Status' screen, become familiar with it and click on any links that you need.
   3. Scroll down and click the 'View Instances' button (bottom right) to proceed.
4. On the Instance view screen select your instance (check only the box next to the instance you just created)
   1. The Instance information appears at the bottom of your screen. Find and record the public and private(local) IP addresses for later use. These are the Client IP’s.
   2. Record the Instance ID, Availability Zone, VPC ID, and the Subnet ID. You will need these when you add the second NIC.
5. Stop your Virtual machine so that you can add a new Network Interface (NIC).
   1. Click 'Actions' dropdown at the top and select 'Instance State' -> 'Stop' -> 'Yes, Stop'
   2. Wait for the whirly-gig next to your instance in the instance list to stop spinning and for the Instance State to be 'stopped' before proceeding. (Hint: This could take several minutes, you can create a new Node IP address while you wait, but you won’t be able to add the new NIC until the VM is stopped)
6. Create a new Subnet for the second NIC
   1. Scroll down in the instance details of your new VM and click on your VPC ID link.
   2. Select 'Subnets' from the new left menu, then click 'Create subnet'.
      1. Name tag - your choice (e.g. ValidatorNode9701-subnet). Providing a name here makes a subsequent step easier.
      2. VPC - select the same VPC as your new VM (recorded in the previous step).
      3. Availability Zone - must select the same Availability zone as your new VM (recorded in the previous step).
      4. IPv4 CIDR block - Type in a valid new subnet block similar to the CIDR already showing. (e.g. 172.31.128.0/24)
      5. Click 'Create'
      6. Return to the EC2 services main page.
7. Create a new security group for the second NIC
   1. On the EC2 side menu, click 'Security Groups' (under Network & Security)
   2. Click 'Create security group'
   3. Security group name - ValidatorNode9701-nsg
   4. Description -Your choice
   5. VPC - default (this might be different for your use case)
   6. Before performing the next set of steps, obtain a list of Node IP addresses from your network administrator. LATER: To get your own list of nodes on your network, run the following command from your validator node after installation is complete and the node is added to the network:  
       > **sudo current\_validators --writeJson | node\_address\_list**
   7. Inbound rules:
      1. Repeat the following steps for each IP address in your Nodes list.
      2. Click 'Add rule'
      3. Type - Custom TCP
      4. Port range - 9701 (Must match the port that you will set up later in the Node software configuration and must be the same for all rules added to the whitelist)
      5. Source - Custom -> The next IP address from the Nodes list.
      6. Description - Enter the Alias matching the Node from the list for ease of future management.
   8. Click 'Create security group' when you have added all of the Node IPs from your list.
   9. Record the security group Name and ID (e.g. ValidatorNode9701-nsg, sg-09c5205a3af5fb5c6)
8. Create a Network Interface (NIC)
   1. On the EC2 left side menu - Under 'Network & Security' click 'Network Interfaces'
   2. Click 'Create Network Interface'
      1. Description - your choice (e.g. "NIC for the Node IP and port on OpsNode")
      2. Subnet -> Select the new subnet created in a previous step of these instructions. Double check that it is in the exact same availability zone as your instance. (e.g. us-east-2b)
      3. IPv4 Private IP -> auto assign
      4. Elastic Fabric Adapter - leave unchecked
      5. Security groups - Select the Group created during a previous step of these instructions (e.g. ValidatorNode9701-nsg)
      6. Click 'Create'
      7. Select the new interface (only) and click the Attach button in the top menu bar.
         1. Find and select the instance ID (recorded in an earlier step)
         2. Click 'Attach'
9. Record the Network Interface ID of each network interface
   1. On EC2 left side menu - INSTANCES -> Instances
   2. Select your instance
   3. At the bottom of the screen in the description tab, scroll down to ‘Network interfaces’
   4. Click on each interface and then record the ‘Interface ID’ and the ‘Private IP Address’ for the Client and Node interfaces for later use. In my case eth0 is the Client interface and eth1 the Node interface.
10. Create 2 Elastic IP’s and associate them with the network interfaces
    1. For AWS we create Elastic IP addresses because we want the address to be static and the default is for them to be dynamically assigned. We do not want the IP address to change every time you have to reboot your server.
    2. On EC2 left side menu - Network & Security ->Elastic IPs
       1. Click 'Allocate Elastic IP Address'
       2. Click 'Allocate'
       3. Repeat the above 2 steps to allocate another IP address
       4. At this point you will not see both addresses created! When I did this a filter appeared that was blocking me from seeing any more new addresses created. If you have created too many addresses, as I did, simply select the ones you want to remove, click 'Actions', then select 'Release Elastic IP addresses' and follow the prompts for removal.
    3. Give your new addresses appropriate names so that you can identify them later. (i.e. OpsNode Client and OpsNode Node)
    4. For each new Elastic IP do the following:
       1. Select one of the Elastic IP’s you just created
       2. Click Actions -> Associate address
          1. Resource type - ‘Network interface’
          2. Network Interface - <use one of the network interface IDs noted in previous step>
          3. Private IP - Select the IP from the list (there should only be one option and it should match the internal IP address of the chosen interface)
          4. Leave 'Ressociation' checkbox empty
          5. Click 'Associate'
          6. Click 'Clear filters' (if you don't see the second Elastic IP in the list)
       3. Repeat the above steps for the other interface.
    5. Click 'Clear filters' again if needed.
    6. Check to make sure that both Elastic IP's have been associated, and then record and label the Public/Private IP address combinations in a place where you get to it later.
    7. Click 'Instances' in the left menu and then select your instance.
    8. Scroll down in the bottom pane and click on each of the network interfaces to double check and be sure that you have recorded the Public and Private IP addresses associated with each named interface. (e.g. Client - eth0, 13.58.197.208, 172.31.26.65 and Node - eth1, 3.135.134.42, 172.31.128.42) This information will be used when you install the Validator on your instance.
11. Start your instance
    1. Select your instance - click the 'Actions' dropdown from the top bar -> Instance State -> Start.
    2. Wait for the 'Instance State' of your instance to be 'running' before performing the next step.
12. Log in to your VM
    1. From your workstation do the following: (Windows 10 will be different)
    2. ssh -i <public rsa key file> ubuntu@<Client IP Address>
    3. Where rsa key file was the ssh key .pem file generated earlier
    4. And where Client IP is the public address from Nic #1 (Client Public IP from your Node Installation Info spreadsheet)
    5. for example: ssh -i ~/pems/opsnode.pem ubuntu@13.58.197.208
    6. NOTE: I got an error the first time I ran the above to login: "Permission denied" because "Permissions are too open" <for your pem file>. To correct the issue I ran chmod 0600 ~/pems/opsnode.pem and then I was able to login successfully.
13. Configure networking to the second NIC
    1. From your instance's command prompt, run the command $ ip a and verify that you have 2 internal IP addresses that match what you have in your Node Installation Info spreadsheet. Note the names of the network interfaces. (mine were ens5 and ens6) The remaining instructions in this section assume ens5 is your original primary NIC (Client NIC) and ens6 is the secondary NIC (Node NIC).
    2. route -n
       1. Record the default gateway for later use. (e.g. mine is 172.31.16.1)
    3. Disable automatic network management by AWS. Run the following:
       1. sudo su -
       2. echo 'network: {config: disabled}' > /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg
    4. cd /etc/network/interfaces.d
    5. vim 50-cloud-init.cfg
       1. Cut the existing ens5 lines from this file in preparation for moving them to a new file in this same directory. (Mine only had 2 lines to remove.)
       2. Save the changed file.
       3. Example 50-cloud-init.cfg now looks like:   
             
           auto lo  
           iface lo inet loopback
    6. Create 2 new network interface files using the following suggestions. These example files are configured so that an AWS instance can use 2 IP addresses on 2 different interfaces and subnets.
    7. vim ens5.cfg (use <interface name>.cfg if your interface name is not ens5)
       1. For the following, substitute the Gateway and interface names recorded earlier for <Gateway> and <interface name> respectively. <interface name> should also match the name of the file it is in. Paste the ens5 lines cut from the 50-cloud-init.cfg file and then add the following lines, indented 3 spaces:  
            
           up ip route add default via <Gateway> dev <interface name> tab 1  
           up ip rule add from <local IP addr of <interface name>>/32 tab 1  
           up ip rule add to <local IP addr of <interface name>>/32 tab 1  
           up ip route flush cache

* 1. Example ens5.cfg  
       
     auto ens5

iface ens5 inet dhcp

up ip route add default via 172.31.16.1 dev ens5 tab 1

up ip rule add from 172.31.26.65/32 tab 1

up ip rule add to 172.31.26.65/32 tab 1

up ip route flush cache

* 1. Repeat previous steps but for the second network interface. The simplest way to do that is probably:
     1. cp ens5.cfg ens6.cfg
     2. vi ens6.cfg
        1. Replace all instances of ens5 with ens6
        2. Change <local IP addr> to the one corresponding to ens6
        3. Change ‘tab 1’ to ‘tab 2’
        4. Change the Gateway to the correct value (I used 172.31.128.1, but see below for how I found out what it was)
  2. Example ens6.cfg  
       
     auto ens6

iface ens6 inet dhcp

up ip route add default via 172.31.128.1 dev ens6 tab 2

up ip rule add from 172.31.128.42/32 tab 2

up ip rule add to 172.31.128.42/32 tab 2

up ip route flush cache

* 1. ifup ens6
     1. Check to make sure ens6 came up and is working properly using the command ip a. NOTE: When I ran ifup, I got the error "Failed to bring up ens6." because I had the wrong Gateway in ens6.cfg. At this point if you run route -n again, you will see the correct gateway and can adjust ens6.cfg to use it.
     2. If the ens5 interface becomes unusable, you should now be able to log in through ens6 to fix it (if you first allow ssh to the second NIC in the EC2 console).
  2. Restart your instance.
     1. reboot
  3. ssh to your instance again as described earlier.
     1. ssh -i <public rsa key file> ubuntu@<Client IP Address>

1. Configure and mount the data disk.
   1. Find the name of your data disk:
      1. sudo fdisk -l
      2. In most cases **/dev/nvme1n1** will be the name of the 1Tib data disk created during the EC2 instance setup.
   2. The following steps assume that your disk size is less than 2 TiB, that your disk is /dev/nvme1n1 and that you will be using MBR partitioning.
   3. sudo fdisk /dev/nvme1n1
      1. Create a new partition
         1. n
         2. p
         3. <defaults for the rest> TIP: press enter 3 times to accept the defaults and complete the process of creating a partition.
         4. Now, print and write the partition and exit.
         5. p
         6. w
   4. Update the kernel:
      1. partprobe
   5. Add a filesystem to your new disk partition:
      1. sudo mkfs -t ext4 /dev/nvme1n1p1
   6. Mount the disk to the directory where the Node software does the most writing (/var/lib/indy):
      1. sudo mkdir /var/lib/indy
      2. sudo mount /dev/nvme1n1p1 /var/lib/indy
   7. Add the drive to /etc/fstab so that it mounts at server startup.
      1. sudo blkid
      2. Record the UUID of /dev/nvme1n1p1 for use in the /etc/fstab file.
      3. sudo vim /etc/fstab
      4. Add the following line to the end of the fstab file (substituting in your own UUID):
         1. UUID=189d7b71-99b6-44fa-8b26-5ea4db3c9ffd /var/lib/indy ext4 defaults,nofail 1 2
         2. Vim Hint: In vim, arrow down to the last line of the file, press the ‘o’ key and then paste in the above line. As before, <esc> then :wq will write and exit the file.
2. Restart the instance to check for NIC and Disk persistence.
   1. From EC2 select your instance, click ‘Actions’ -> 'Instance State' -> 'Reboot'
   2. Login to your VM as before:
      1. ssh -i <public rsa key file> ubuntu@<Client IP Address>
   3. Check the NIC
      1. ip a
      2. The output of the above command should have 2 NICS with the correct IP addresses displayed.
      3. df -h
      4. The output of the above command should show /var/lib/indy mounted to the /dev/nvme1n1p1 disk with the correct size (1T).
      5. More NIC and disk verifications will occur during the Indy Node install process.
3. Optional: Add a temporary administrative user as a safety net during Two Factor Authentication (2FA) setup.
   1. sudo adduser tempadmin
      1. You can safely ignore messages like “sent invalidate(passwd) request, exiting“
   2. sudo usermod -aG sudo tempadmin
   3. Setup sshd\_config to temporarily allow password login for the tempadmin user.
      1. sudo vim /etc/ssh/sshd\_config
      2. Comment out the line containing ‘ChallengeResponseAuthentication’.
         1. #ChallengeResponseAuthentication no
      3. Make sure this line exists and is set to yes:
         1. PasswordAuthentication yes
      4. :wq to save and exit.
      5. sudo systemctl restart sshd
      6. The above lines will be altered again when you set up 2FA.
   4. To be able to login, you will also likely need to setup an ssh key
      1. sudo mkdir /home/tempadmin/.ssh
      2. sudo chown tempadmin:tempadmin /home/tempadmin/.ssh
      3. sudo vim /home/tempadmin/.ssh/authorized\_keys
      4. Paste the users public key into the open file and then save it (:wq) (You can use the same key as you used for the ubuntu user in this case, since it is a temporary user)
      5. sudo chown tempadmin:tempadmin /home/tempadmin/.ssh/authorized\_keys
4. Setup 2FA for SSH access to the Node for your base user.
   1. Optional: Login in a separate terminal as your tempadmin user (that has sudo privileges) to have a backup just in case something goes wrong during setup.
      1. ssh tempadmin@<Client IP Addr>
   2. Install Google Authenticator, Duo, or Authy on your phone.
   3. As your base user on the Node VM, run the following to install the authenticator:
      1. sudo apt-get install libpam-google-authenticator
   4. Configure the authenticator to allow both password and SSH key login with 2FA by changing 2 files:
      1. sudo vim /etc/pam.d/common-auth
      2. Add the following line as the first uncommented line in the file
         1. auth sufficient pam\_google\_authenticator.so
         2. :wq
      3. sudo vim /etc/ssh/sshd\_config
         1. add/configure the following lines:
            1. ChallengeResponseAuthentication yes
            2. UsePAM yes
            3. AuthenticationMethods publickey,keyboard-interactive
            4. PasswordAuthentication no
         2. If you see any of the above lines commented out, remove the # to uncomment them. If you don't see any of the above lines, make sure to add them. If you see those lines configured in any different way, edit them to reflect the above.
         3. :wq
      4. sudo systemctl restart sshd
   5. Setup your base user to use 2FA by running the following from a terminal:
      1. google-authenticator
      2. Answer ‘y’ to all questions asked during the setup
      3. Save the secret key, verification code and scratch codes in a safe place. These are all just for your user and can be used to login or to recover as needed.
   6. On your phone app add an account and then scan the barcode or enter the 16 character secret key from the previous steps output.
   7. You should now be able to login using 2FA. First, check that login still works for your base user in a new terminal. If that doesn’t work, double check all of the configuration steps above and then restart sshd again. If it still doesn’t work, it’s possible that a server restart is required to make 2FA work (NOTE: It is dangerous to restart at this point, because then all of your backup terminals that are logged in will be logged out and there is a chance that you will lose access. Please check that all other steps have been executed properly before restarting.)
5. Add other administrative users:
   1. Send the other new admin users the following instructions for generating their own SSH keys:
      1. ssh-keygen -P "" -t rsa -b 4096 -m pem -f ~/pems/validatornode.pem
      2. Have the new users send you their public key (e.g. validatornode.pem.pub if they do the above command)
      3. Also have them send you their Public IP address so that you can add it to the EC2 firewall to allow them access. Optionally, have them send a preferred username also.
   2. Add their IP addresses to the EC2 firewall:
      1. From the EC2 instance screen, select your instance and scroll down to find and click on the primary security group. (e.g. ValidatorClient9702)
      2. Click the Inbound rules tab just below the middle of the screen and click the 'Edit inbound rules' button.
      3. In the new window that pops up, click in the 'Source' field of the port 22 rule to add the new users' IP addresses separated by commas.(no spaces)
      4. Click ‘Save’ (Note: Restart is not needed. As soon as you save, they should have access.)
   3. Add the users to the server:
      1. Login to the Node as the base user.
      2. Run the following commands, substituting the username in for <newuser>
      3. sudo adduser <newuser>
         1. You can safely ignore messages like “sent invalidate(passwd) request, exiting“
      4. sudo usermod -aG sudo <newuser>
      5. Then create a file in the newusers home directory:
         1. sudo mkdir /home/<newuser>/.ssh
         2. sudo chown <newuser>:<newuser> /home/<newuser>/.ssh
         3. sudo vim /home/<newuser>/.ssh/authorized\_keys
         4. Paste the users public key into the open file and then save it (:wq)
         5. sudo chown <newuser>:<newuser> /home/<newuser>/.ssh/authorized\_keys
      6. Repeat the above for each new admin user you create.
   4. The new users are now able to login. Since 2FA is required, when you send the password to each of the new users, also send the following instructions (HINT: fill in the username, Client IP address, and password for them with the correct values):
      1. Thanks for agreeing to help with the administration of our Indy Validator Node. Please login to the node, change your password, and setup Two Factor Authentication (2FA) using the following instructions:
         1. ssh -i <your private SSH key file> <username>@<Client IP Addr>
         2. Type in <password> for your password
         3. On successful login, type in ‘passwd’ to change your password on the Validator Node. Please use a unique password of sufficient length and store it in a secure place (i.e. a password manager).
         4. To set up 2FA, type in ‘google-authenticator’
            1. Answer ‘y’ to all questions asked during the setup
            2. Save the secret key, verification code, and scratch codes in a safe place. These are all for your user and can be used to login or to recover as needed.
         5. Install Google Authenticator, Duo, Authy, or other google-authenticator compatible app on your phone or device.
         6. On your 2FA phone app, add an account, and then scan the barcode or enter the 16 character secret key from step 4’s output.
         7. Log out and then log back in to check and make sure it worked!
   5. All of your secondary admin users should be setup now.
6. You can now begin the Indy Node installation.

# Troubleshooting, Tips, and Acknowledgements

1. Helpful links (and acknowledgements to the authors of the content on these sites).
   1. <https://aws.amazon.com/premiumsupport/knowledge-center/ec2-ubuntu-secondary-network-interface/>
   2. <https://www.digitalocean.com/community/tutorials/how-to-add-and-delete-users-on-ubuntu-16-04>
   3. <https://www.techrepublic.com/article/how-to-combine-ssh-key-authentication-and-two-factor-authentication-on-linux/>
   4. For adding a second user’s 2FA for them<https://www.linux.com/topic/desktop/how-set-2-factor-authentication-login-and-sudo/>