Complete Cloud Lab Walkthrough

In general, this document should not be needed. The following steps will guide you through stetting up the cloud lab resources in Azure in the event that students missed days or need to setup the lab all at once to use it for the upcoming project week.

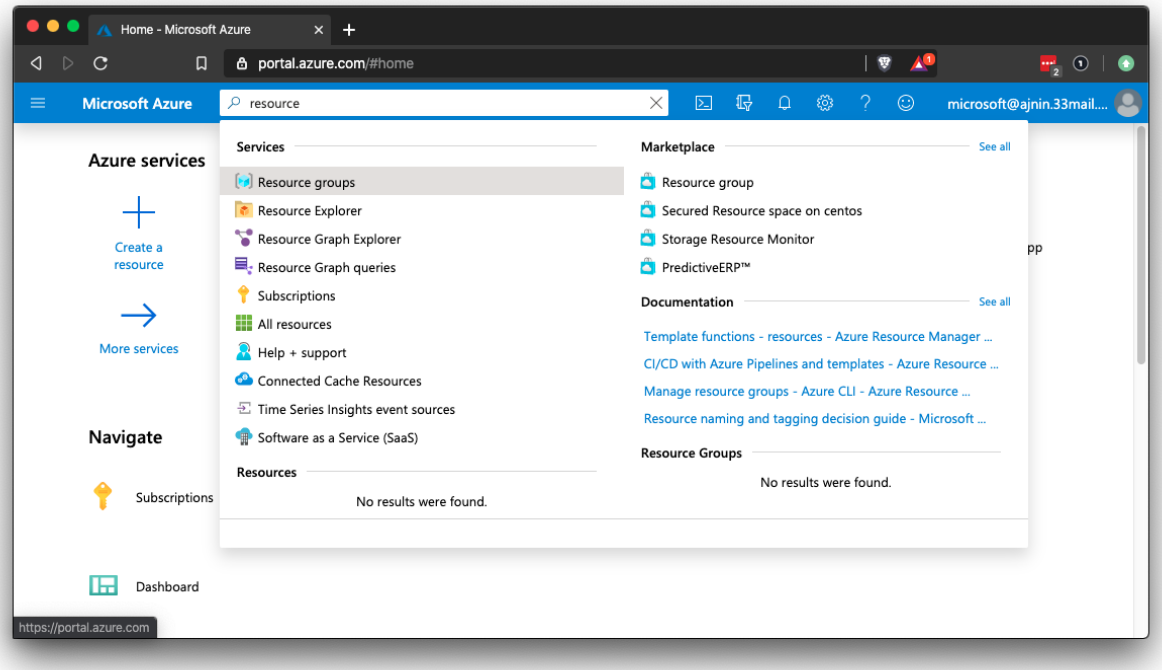
These are the same steps that are covered throughout the activities in the cloud weeks.

VERY IMPORTANT: PAY ATTENTION TO THE USERS THAT YOU CREATE THROUGHOUT THE VARIOUS STEPS. THE EXERCISES BELOW PROVIDE EXAMPLE USERS, BUT MAKE SURE YOU NOTE AND USE THE USER THAT YOU CREATE.

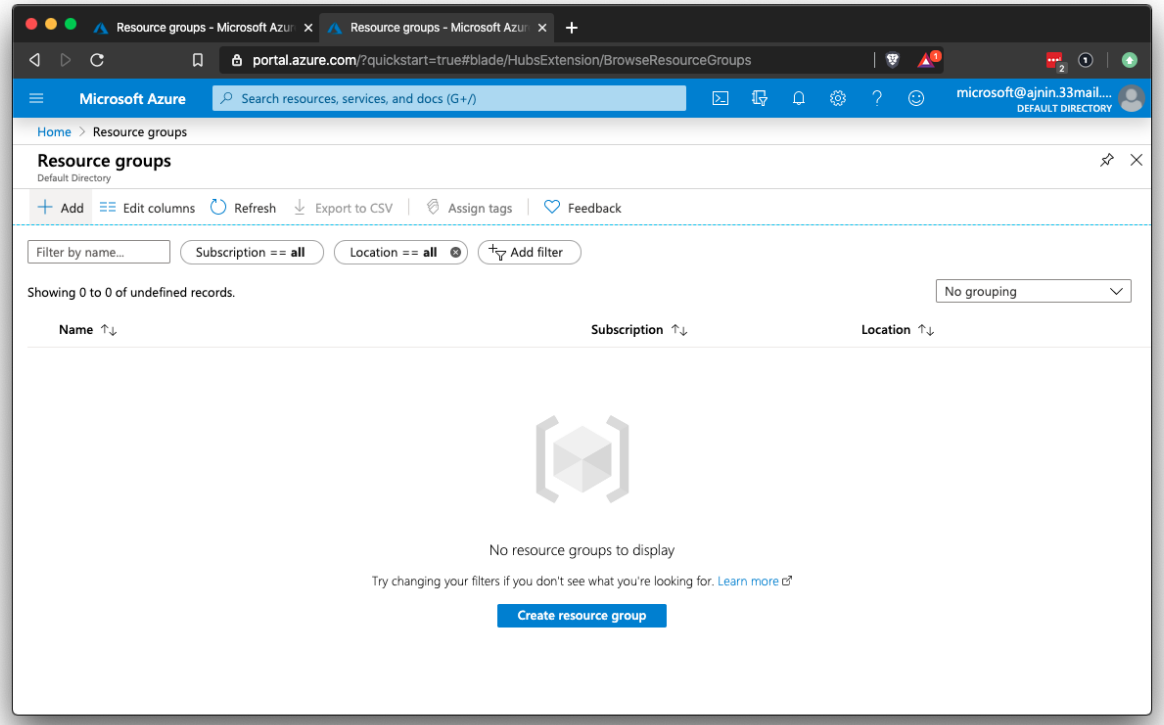
Setting up the Resource Group

Use the Azure portal to create a resource group that will contain everything the Red Team needs in the cloud.

- On the home screen, search for "resource."

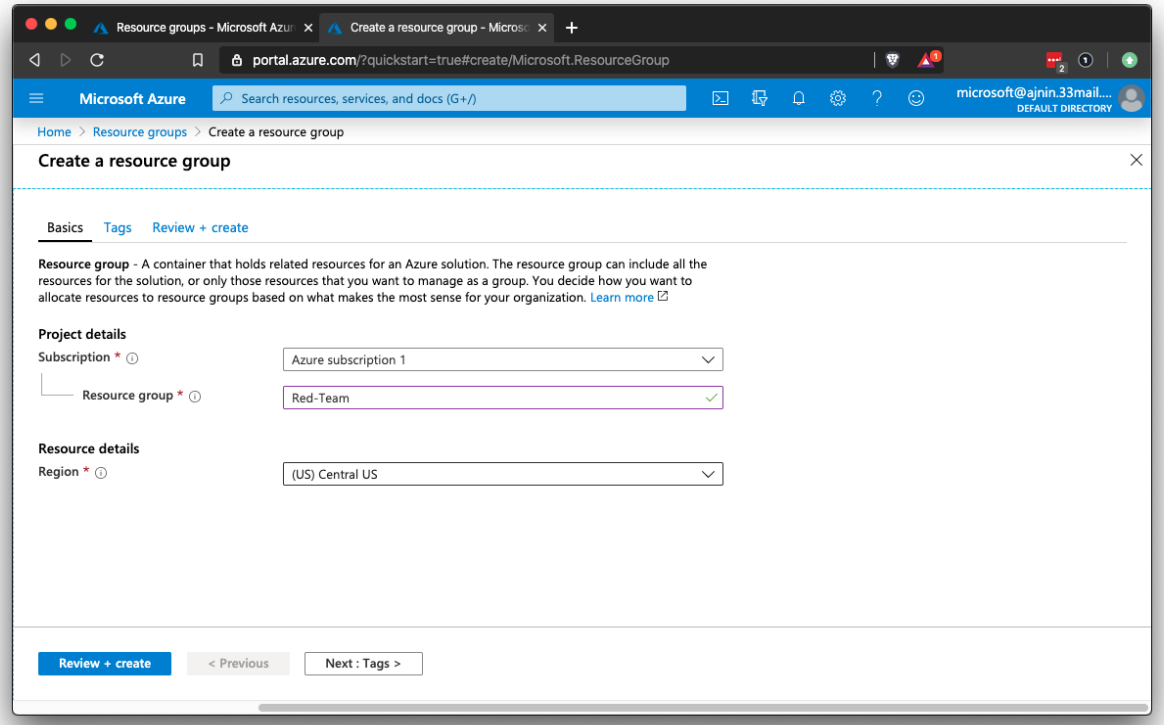


- Click on the + Add button or the Create resource group button.

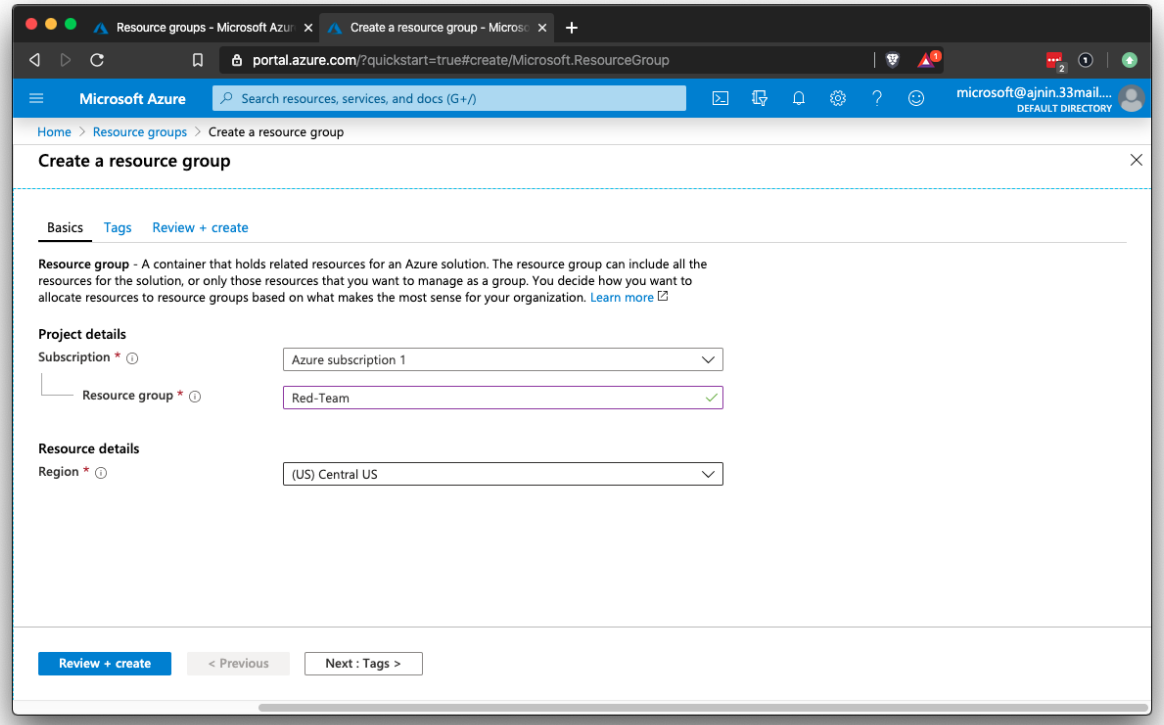


- Create a name for your resource group and choose a region.
    - Note: Choose a region that you can easily remember. Every resource you create after this must be created in the exact same region.

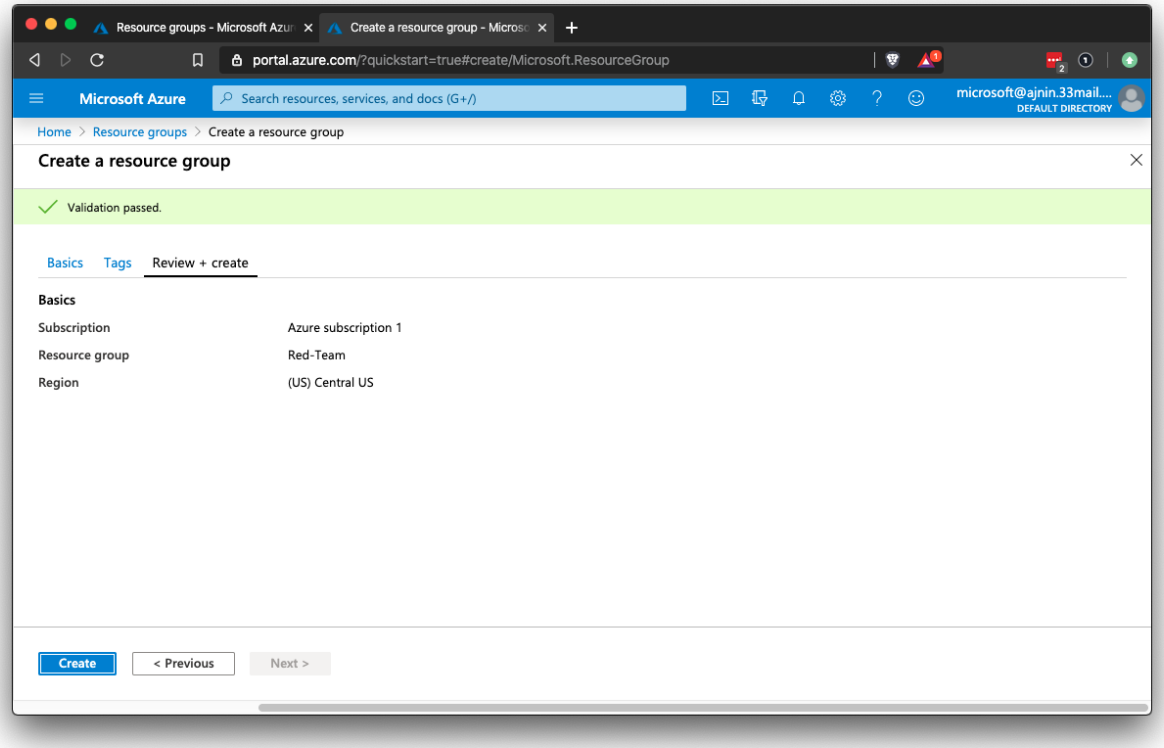
- Click on Review + create.



- Azure will alert you if there are any errors. Click on Create to finalize your settings and create the group.



- Once the group is created, click on Go to resource group in the top-right corner of the screen to view your new resource group.

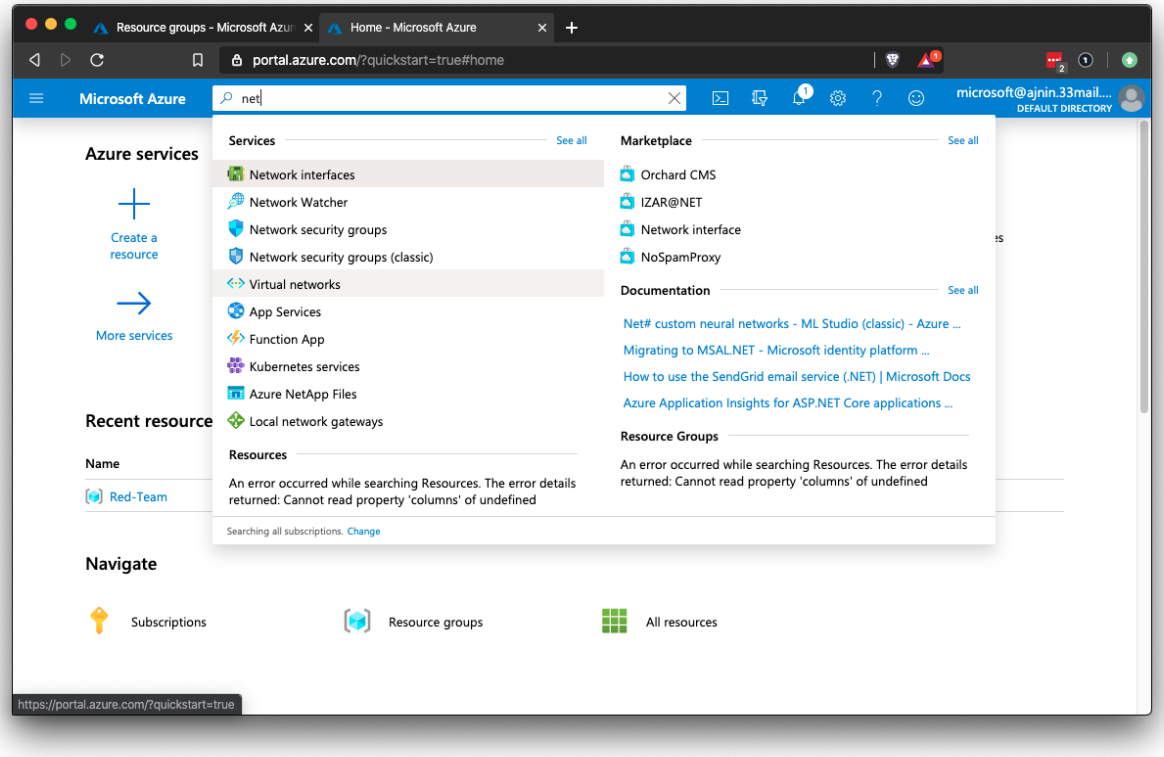


Setting up the VNet

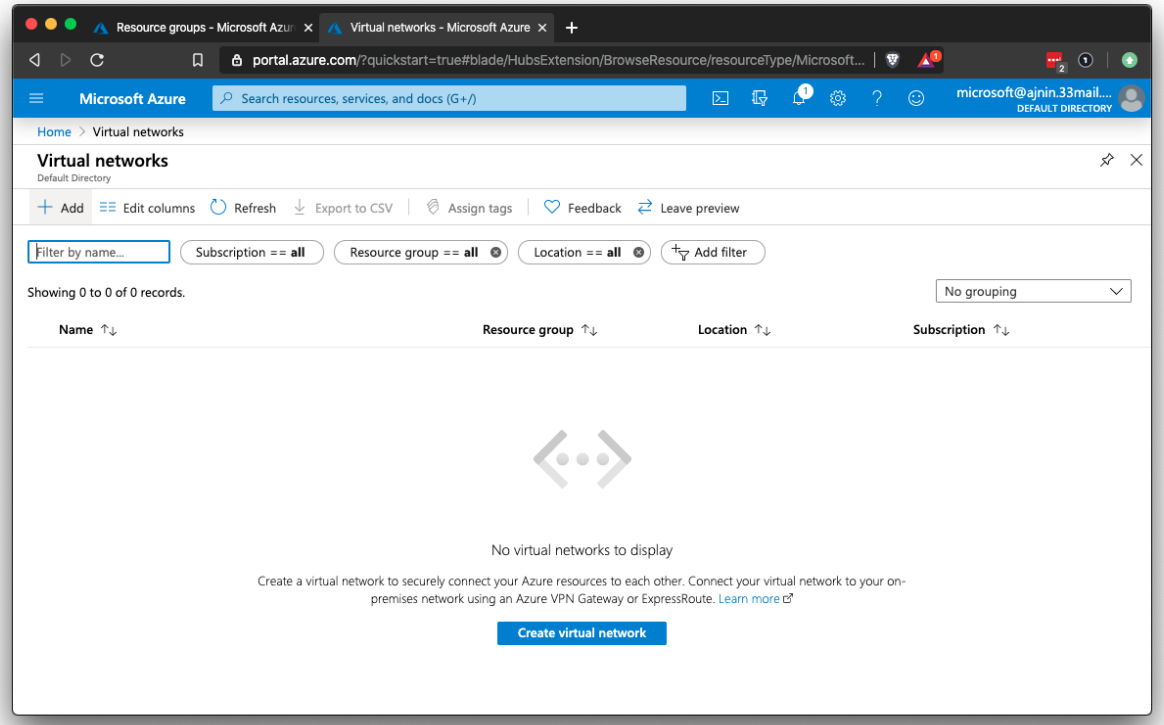
Before you can deploy servers and services, there must be a network where these items can be accessed.

- This network should have the capacity to hold any resource that the Red Team needs, now and in the future.

- Return to the home screen and search for "net." Choose the search result for Virtual networks.

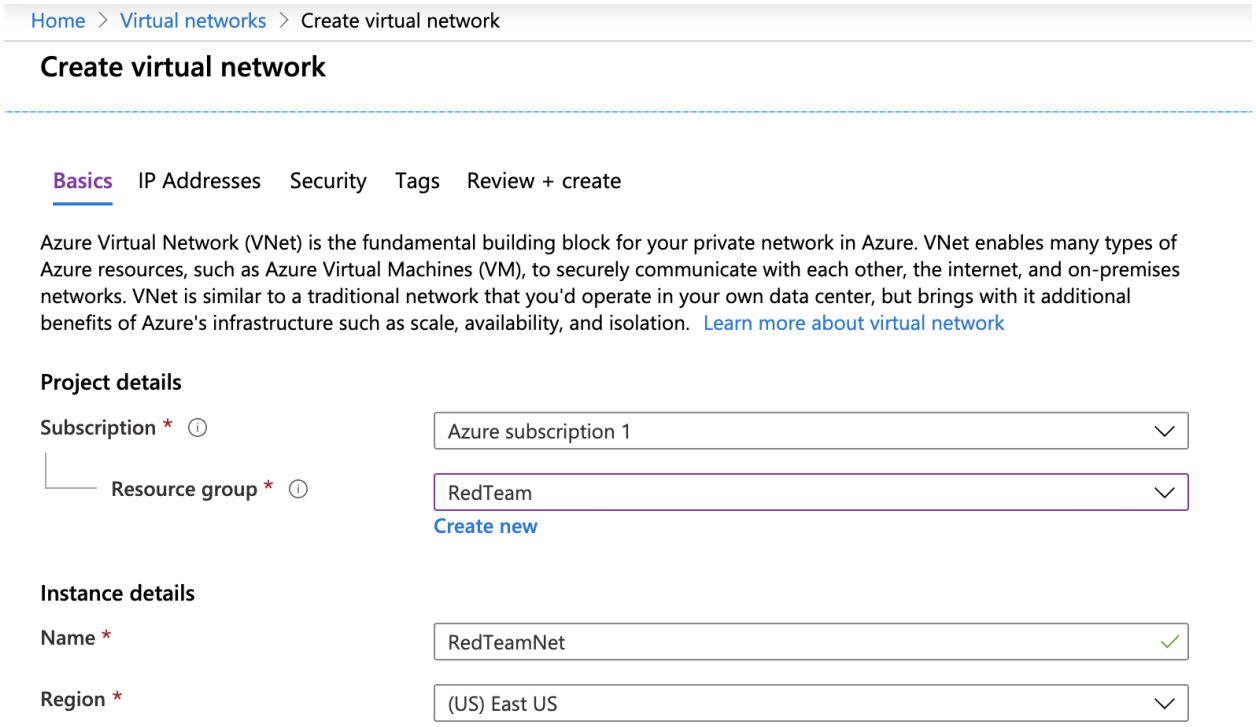


- Click on the + Add button on the top-left of the page or the Create virtual network button on the bottom of the page.

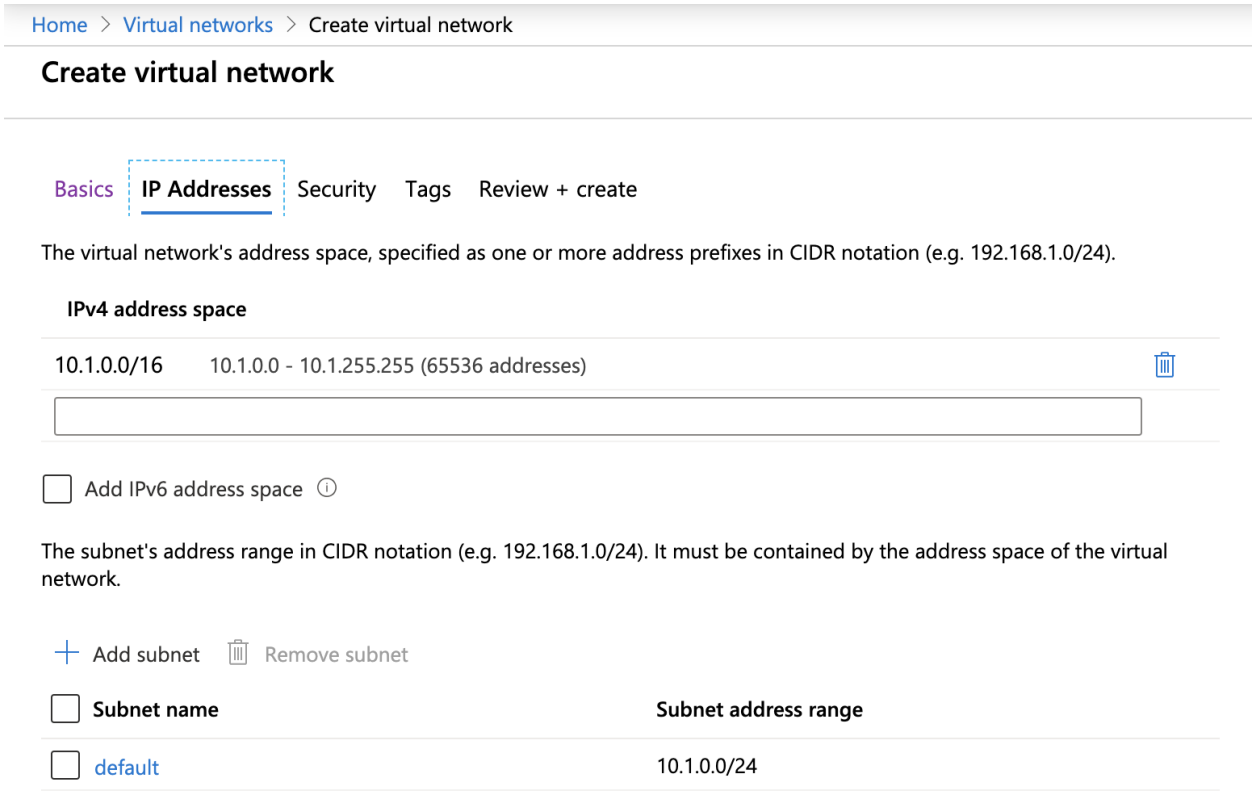


Fill in the network settings:

- Subscription: Your free subscription should be the only option here.
- Resource group: This should be the resource group you created in step two.
- Name: A descriptive name so it will not get confused with other cloud networks in the same account.
- Region: Make sure to choose the same region you chose for your resource group.
    - Carefully configuring the region of your resources is important for ensuring low latency and high availability. Resources should be located as close as possible to those who will be consuming them.

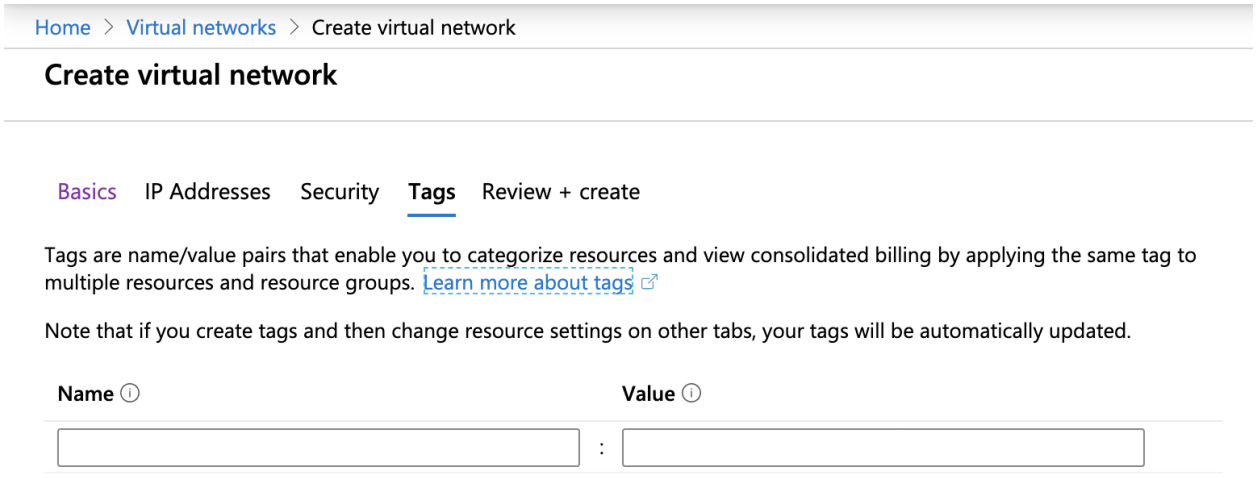


- IP Addresses: Azure requires you to define a network and subnet.
    - Use the defaults on this tab.



- Security: Leave the default settings.

- Tags: No tags are needed.



Click Create.

Create virtual network

Validation passed

Basics | IP Addresses | Security | Tags | **Review + create**

Basics

Subscription	Azure subscription 1
Resource group	RedTeam
Name	RedTeamNet
Region	East US

IP addresses

Address space	10.1.0.0/16
Subnet	default (10.1.0.0/24)

Tags

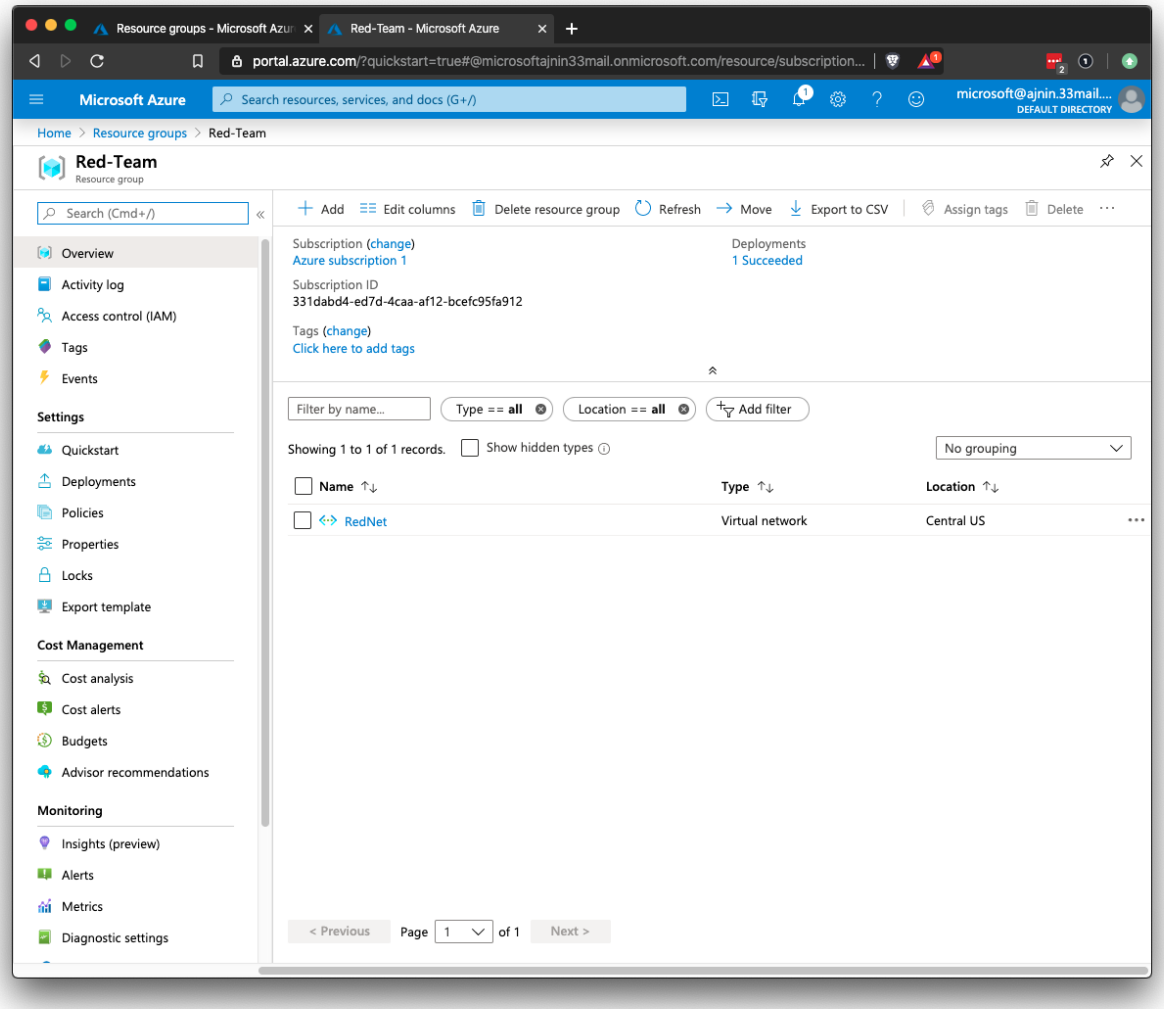
None

Security

BastionHost	Disabled
DDoS protection plan	Basic
Firewall	Disabled

Once you have created your resource group and VNet, return to the home screen and choose the resource group option.

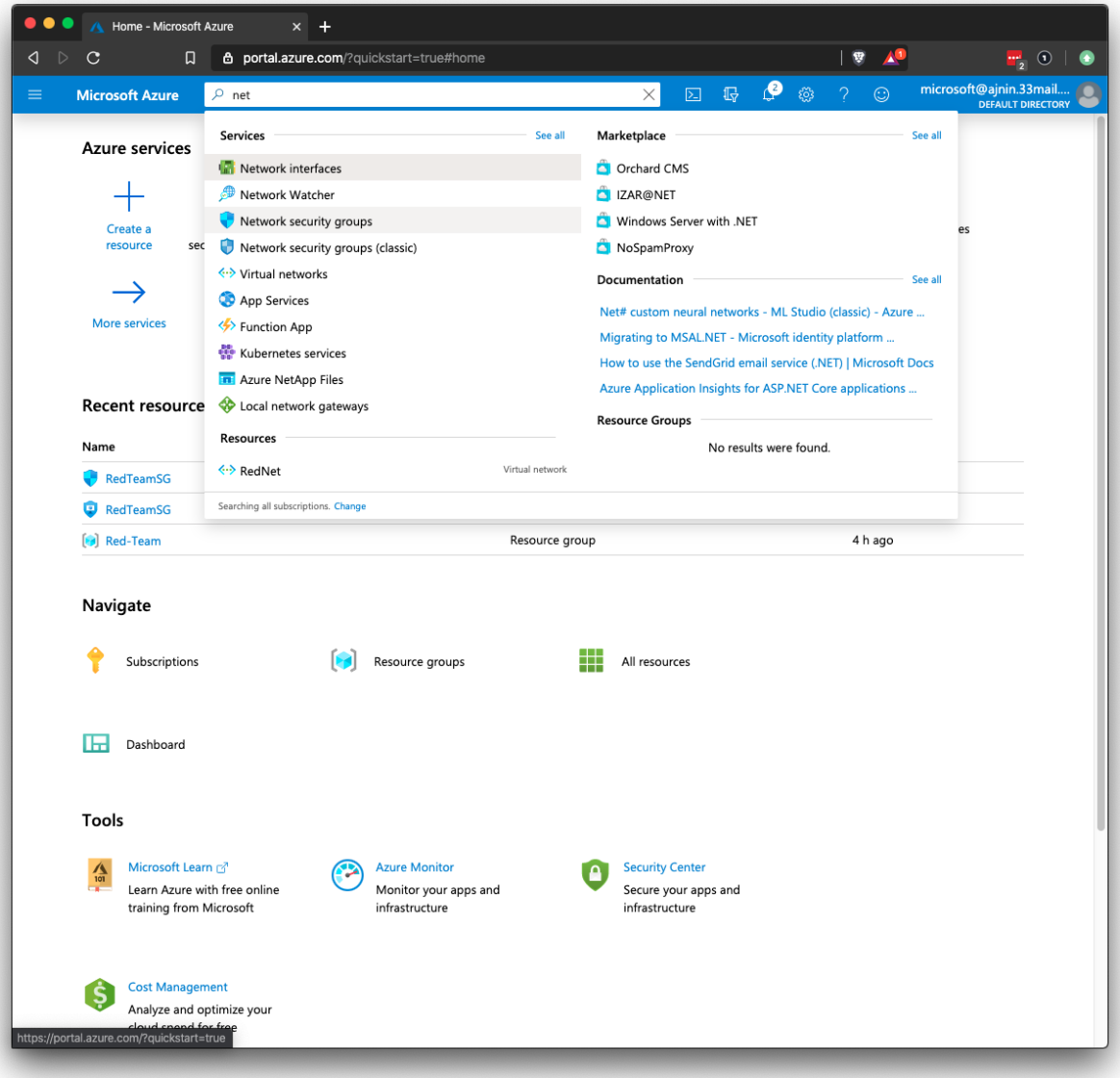
- This provides a list of all resource groups in your account.
- Choose the group that you created and you should see your VNet listed as a resource.



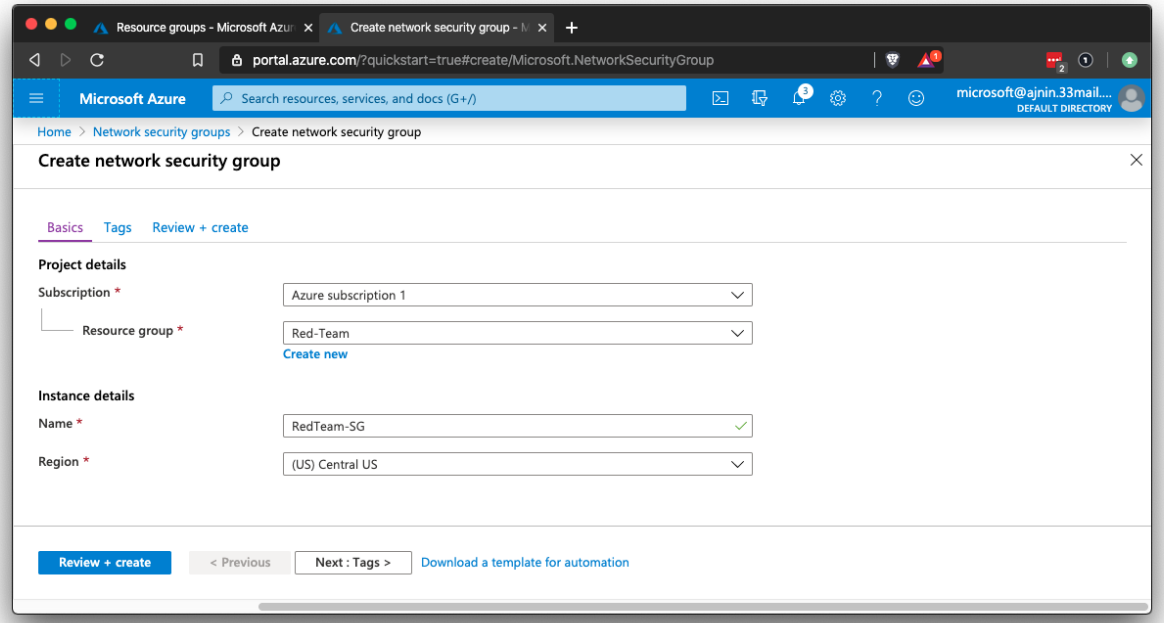
You now have a resource group and VNet that you can use to create the rest of the cloud infrastructure throughout the unit.

Setting up a Network Security Group:

- On your Azure portal home screen, search "net" and choose Network security groups.

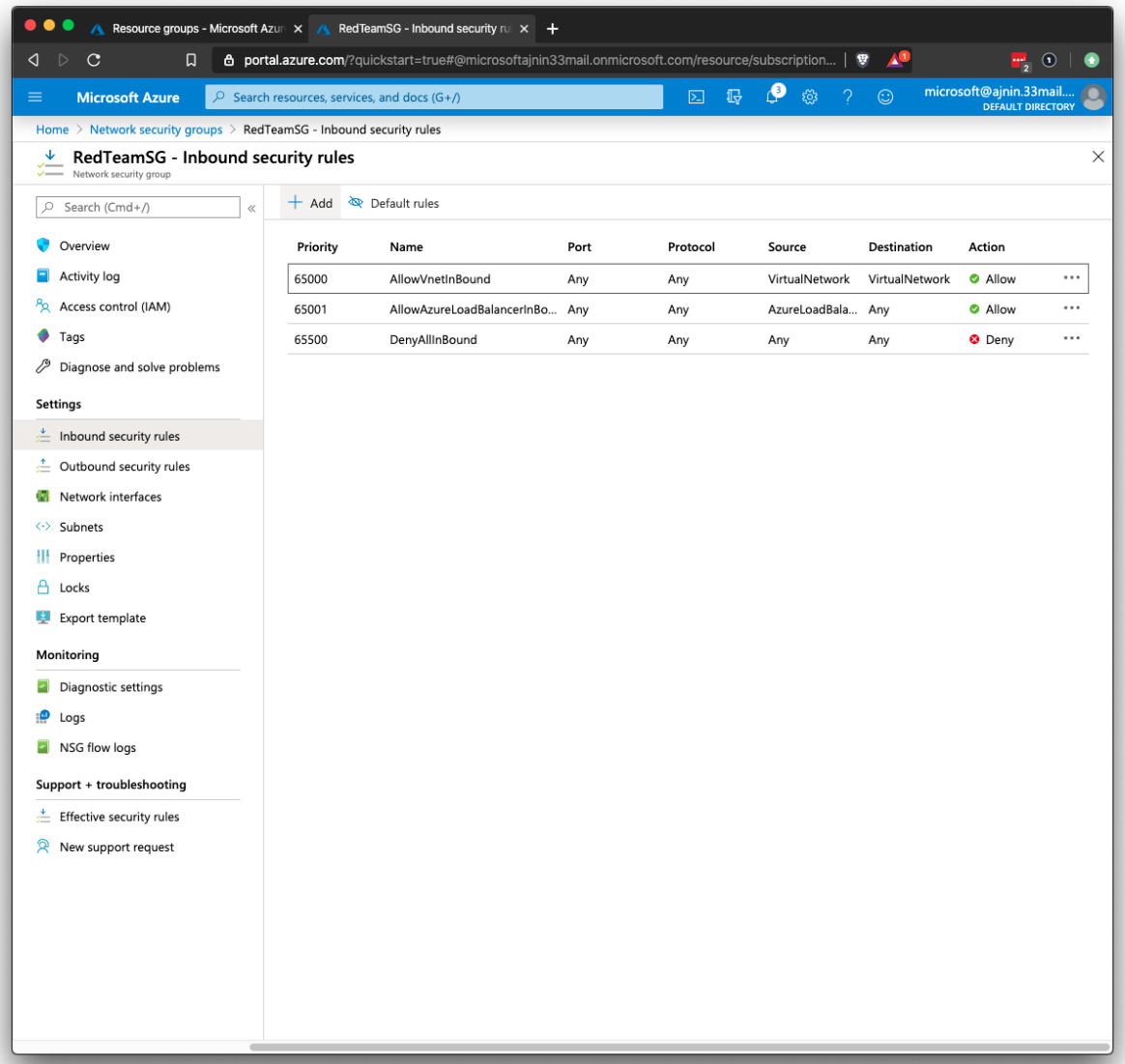


- Create a new security group.
- Add this security group to your resource group.
- Give the group a recognizable name that is easy to remember.
- Make sure the security group is in the same region that you chose during the previous activity.



To create an inbound rule to block all traffic:

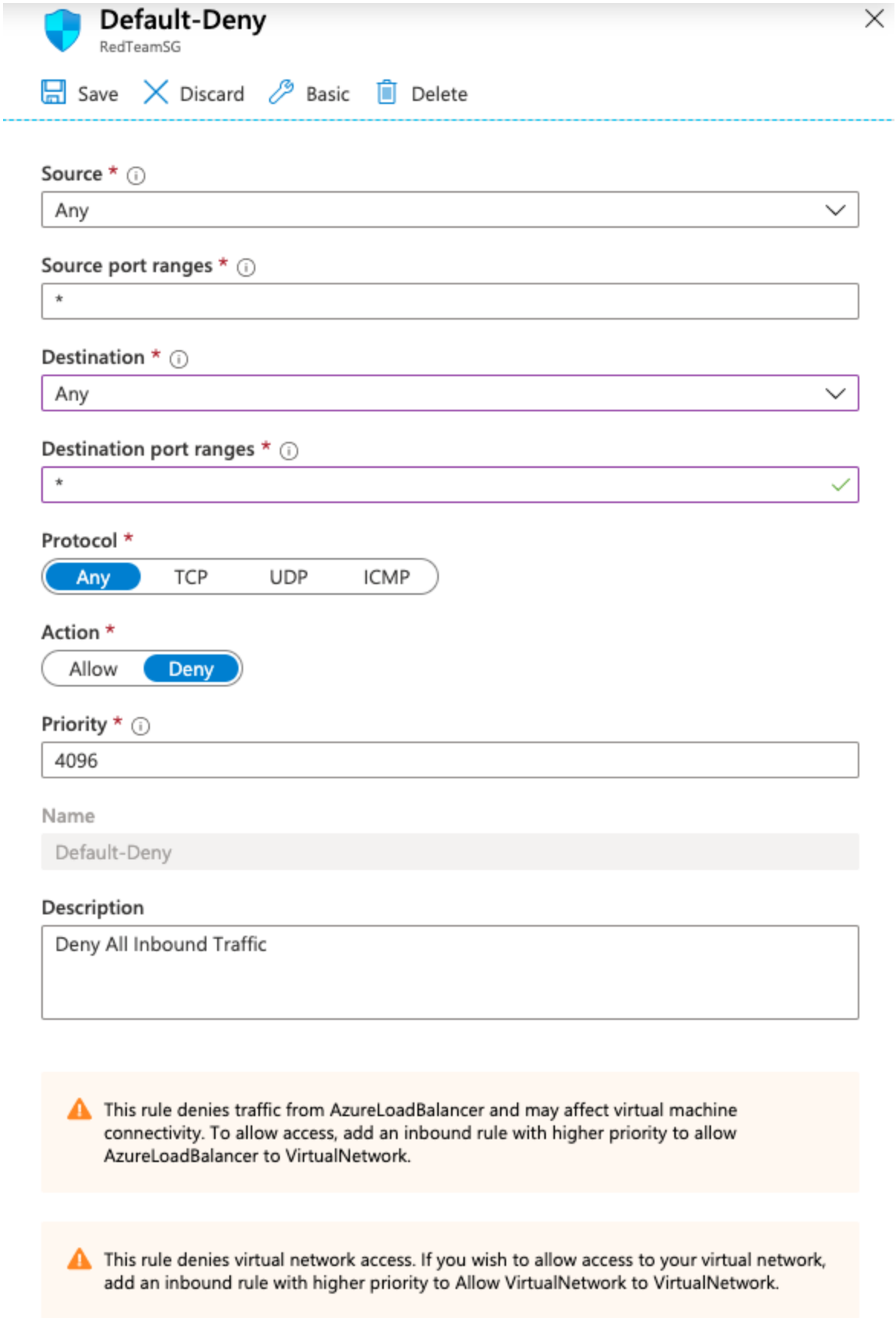
- Once the security group is created, click on the group to configure it.
- Choose Inbound security rules on the left.
- Click on the + Add button to add a rule.



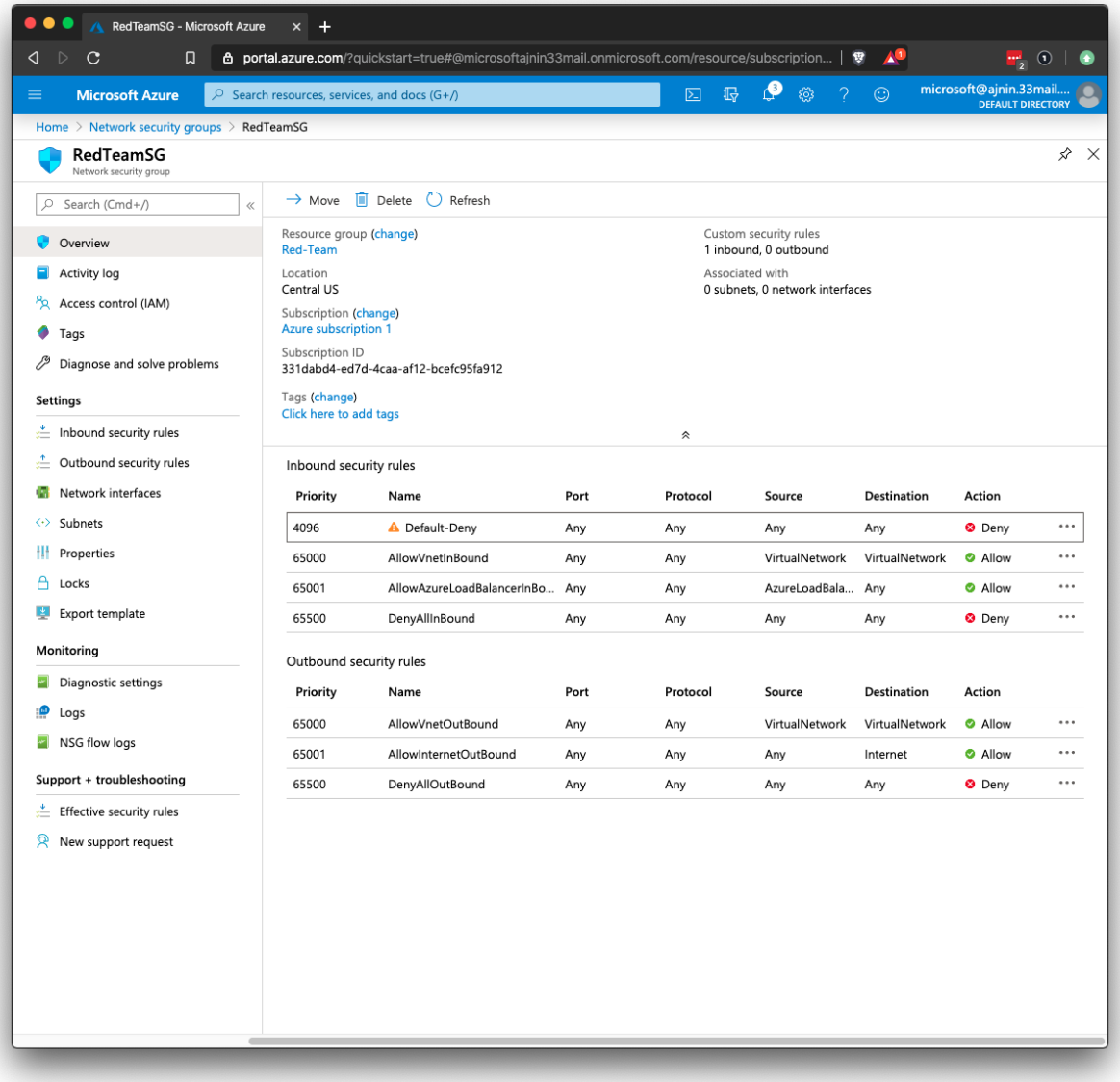
Configure the inbound rule as follows:

- Source: Choose Any source to block all traffic.
- Source port ranges: Source ports are always random, even with common services like HTTP. Therefore, keep the wildcard (*) to match all source ports.
- Destination: Select Any to block any and all traffic associated with this security group.
- Destination port ranges: Usually, you would specify a specific port or a range of ports for the destination. In this case, you can use the wildcard (*) to block all destination ports. You can also block all ports using a range like `0-65535`.
- Protocol: Block Any protocol that is used.

- Action: Use the Block action to stop all of the traffic that matches this rule.
- Priority: This rule will always be the last rule, so it should have the highest possible number for the priority. Other rules will always come before this rule. The highest number Azure allows is 4,096.
- Name: Give your rule a name like "Default-Deny."


Default-Deny
RedTeamSG
Save
Discard
Basic
Delete
Source *
Any
Source port ranges *
*
Destination *
Any
Destination port ranges *
*
Protocol *
Any
TCP
UDP
ICMP
Action *
Allow
Deny
Priority *
4096
Name
Default-Deny
Description
Deny All Inbound Traffic
This rule denies traffic from AzureLoadBalancer and may affect virtual machine connectivity. To allow access, add an inbound rule with higher priority to allow AzureLoadBalancer to VirtualNetwork.
This rule denies virtual network access. If you wish to allow access to your virtual network, add an inbound rule with higher priority to Allow VirtualNetwork to VirtualNetwork.

- Description: Write a quick description similar to "Deny all inbound traffic."
- Save the rule.



You should now have a VNet protected by a network security group that blocks all traffic.

Setting up Virtual Machines

The goal of this activity was to set up your first virtual machines inside your cloud network, which is protected by your network security group. You will use this machine as a jump box to access your cloud network and any other machines inside your VNet.

Remember: Allowing a server to use password authentication for SSH is insecure because the password can be brute forced.

- Therefore, we will only use cryptographic SSH keys to access our cloud servers. Password authentication will not be allowed.
- This is part of the "ground up" security approach that we have been discussing.

Open your command line and run `ssh-keygen` to create a new SSH key pair. - DO NOT CREATE A PASSPHRASE, just press enter twice.

Your output should be similar to:

```
cyber@2Us-MacBook-Pro ~ % ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/cyber/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:r3aBFU50/5iQbbzhqXY+fOIfivRFdMFt37AvLJifC/0 cyber@2Us-MacBook-Pro.local
The randomart image is:
+---[RSA 2048]----+
|         .. . ...|
|          o. =..+|
|         o .o *=+|
|          o  +oB+|
|        So o .*o.|
|        ..+...+ .|
|          o+++.+ |
|        ..oo=+* o|
|       ... ..=E=.|
+----[SHA256]-----+
```

Run `cat ~/.ssh/id_rsa.pub` to display your `id_rsa.pub` key:

Your output should be similar to:

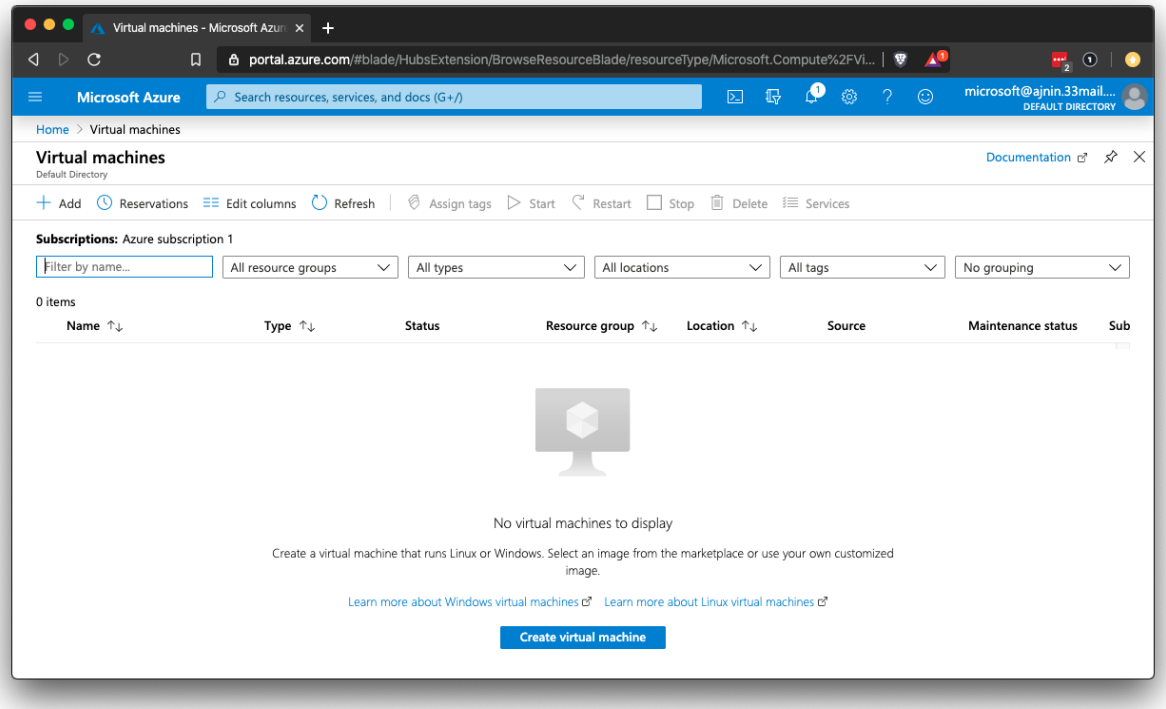
```
cyber@2Us-MacBook-Pro ~ % cat ~/.ssh/id_rsa.pub
```

- `ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDGG6dBJ6ibhgM09U+kn/5NE7cGc4CNHWXein0f+MciKElDalf76nVgFvJQEIImMhAGrtRRJDAd6itlPyBpurSyNOByU6LX7Gl6DfGQKzQns6+n9BheiVLLY9dtodp8oAXdVEGles5EslflPrTrjijVZa9lxGe34DtrjijExWM6hBb0KvwlkU4worPblINx+ghDv+3pdrkUXMsQAht/fLdtP/EBwgSXKYCu/`
- Highlight and copy the SSH key string to your clipboard.

VM 1 - Jump-Box

Open your Azure portal and search for "virtual machines."

- Use the + Add button or the Create virtual machine button to create a new VM.



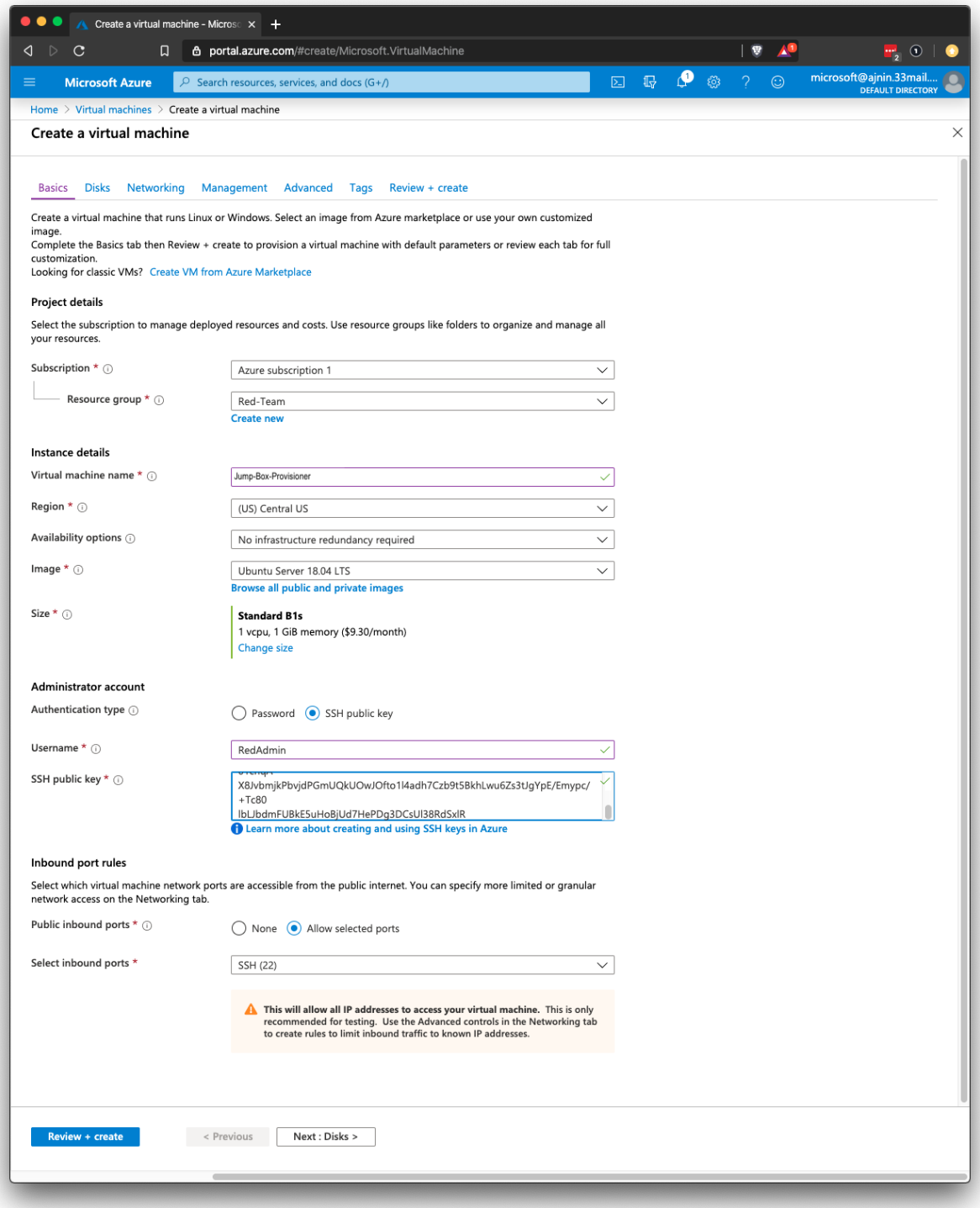
Use the following settings for this VM:

- Resource group: Choose the same resource group that you created for the Red Team.
- Virtual machine name: Use the name "Jump Box Provisioner."
- Region: Use the same region that you used for your other resources.
    - Note that availability of VM's in Azure could cause you to change the region where your VM's are created.
    - The goal is to create 3 machines in the same resource group attached to the same security group. If you cannot add 3 machines to the resource group and security group that you have, a new resource group and security group may need to be created in another region.
- Availability options: We will use this setting for other machines. For our jump box, we will leave this on the default setting.
- Image: Choose the Ubuntu Server 18.04 option.
- Choose the VM option that has:
    - Whose offering is Standard - B1s

    - 1 CPU
    - 1 RAM

For SSH, use the following settings:

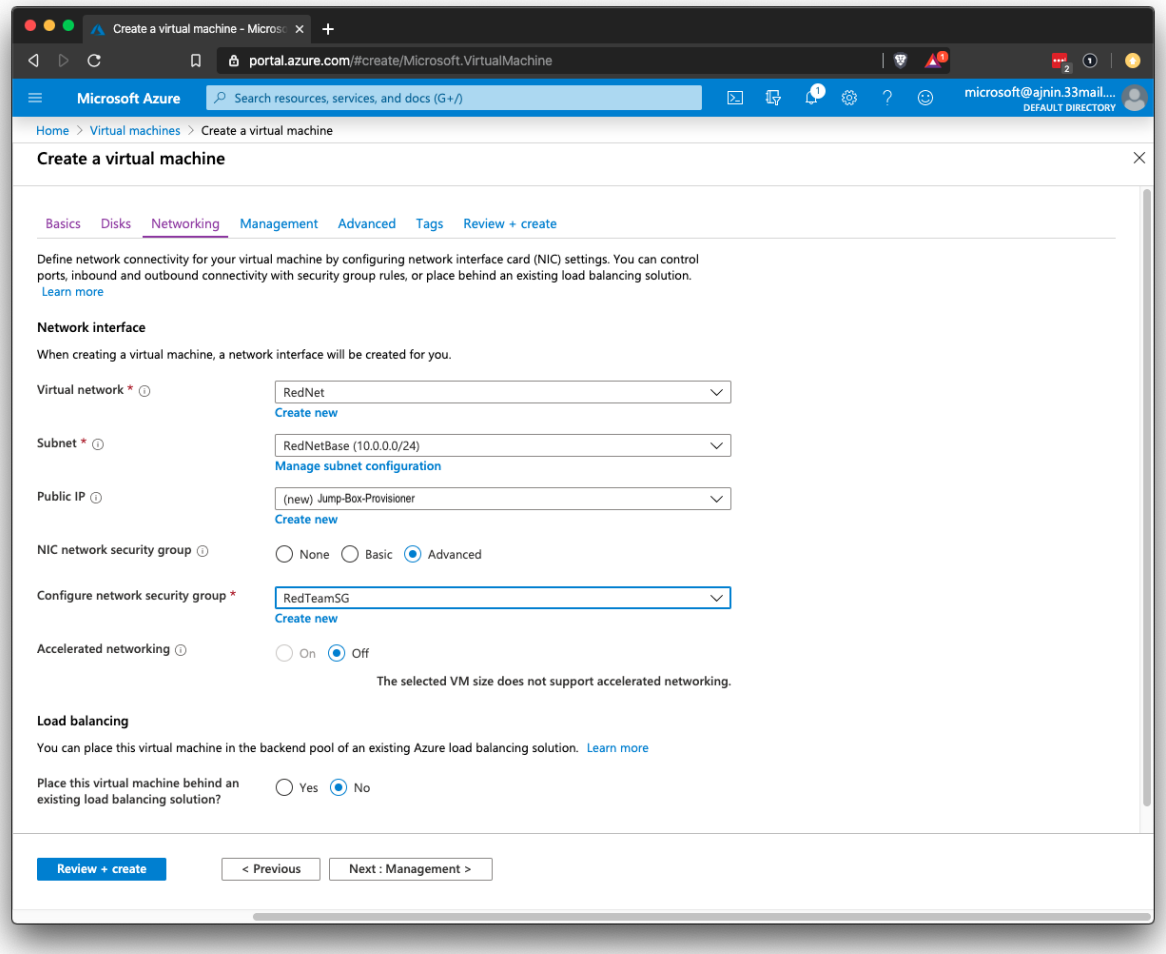
- Authentication type: SSH public key.
- Username: Create any username you like.
- SSH public key: Paste the public key string that you copied earlier.
- Public inbound ports: Ignore this setting. It will be overwritten when you choose your security group.
- Select inbound ports: Ignore this setting. It will be overwritten when you choose your security group.



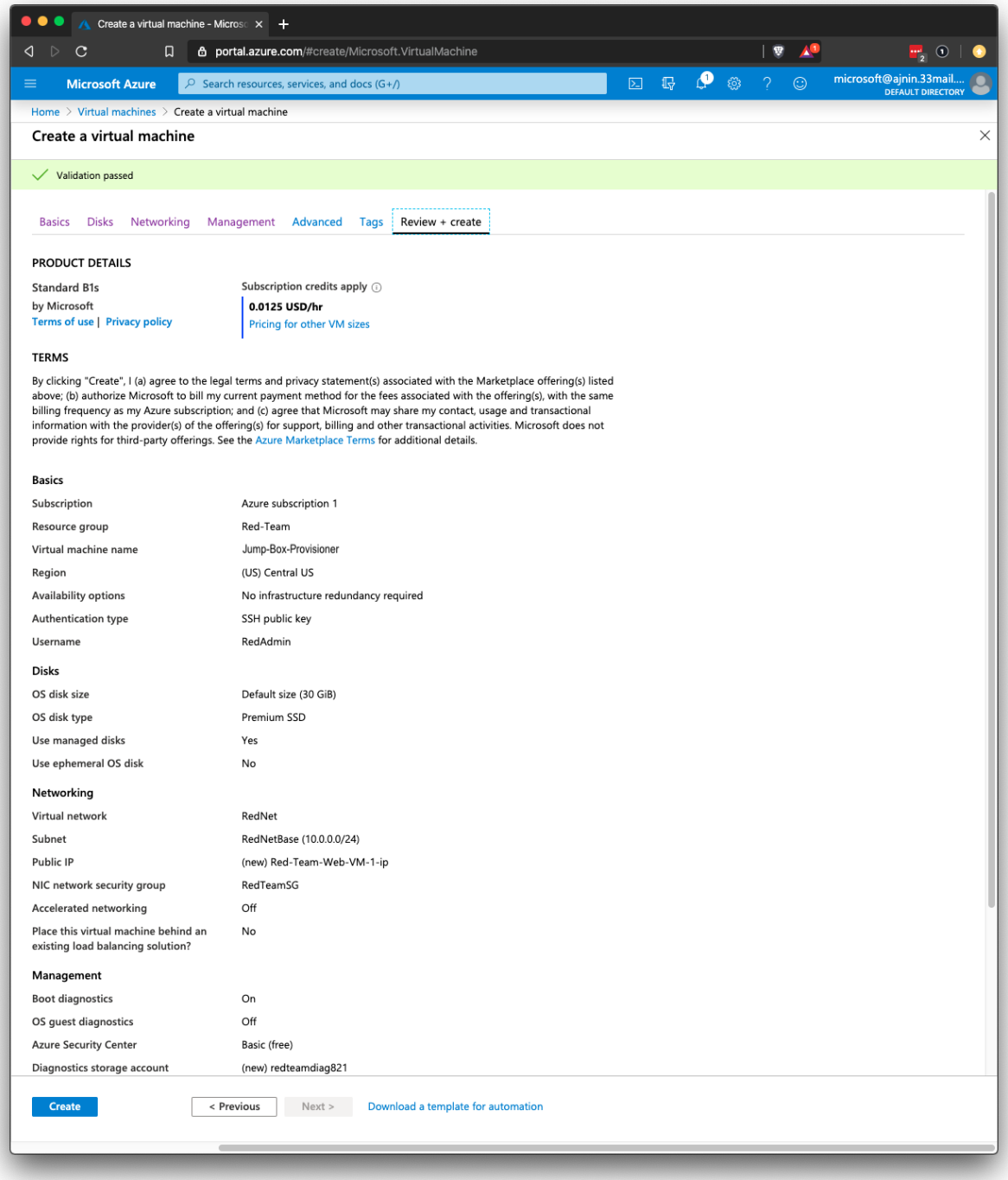
Move to the Networking tab and set the following settings:

- Virtual network: Choose the VNet you created for the Red Team.
- Subnet: Choose the subnet that you created earlier.

- Public IP: This can be kept as default.
- NIC network security group: Choose the Advanced option so we can specify our custom security group.
- Configure network security group: Choose your Red Team network security group.
- Accelerated networking: Keep as the default setting (Off).
- In the Networking settings, take note of the VM URL. You may use it later.
- Load balancing: Keep as the default setting (No).



- Click on Review + create.



- Finalize all your settings and create the VM by clicking on the Create button.

VM's 2 and 3 - Web VM's

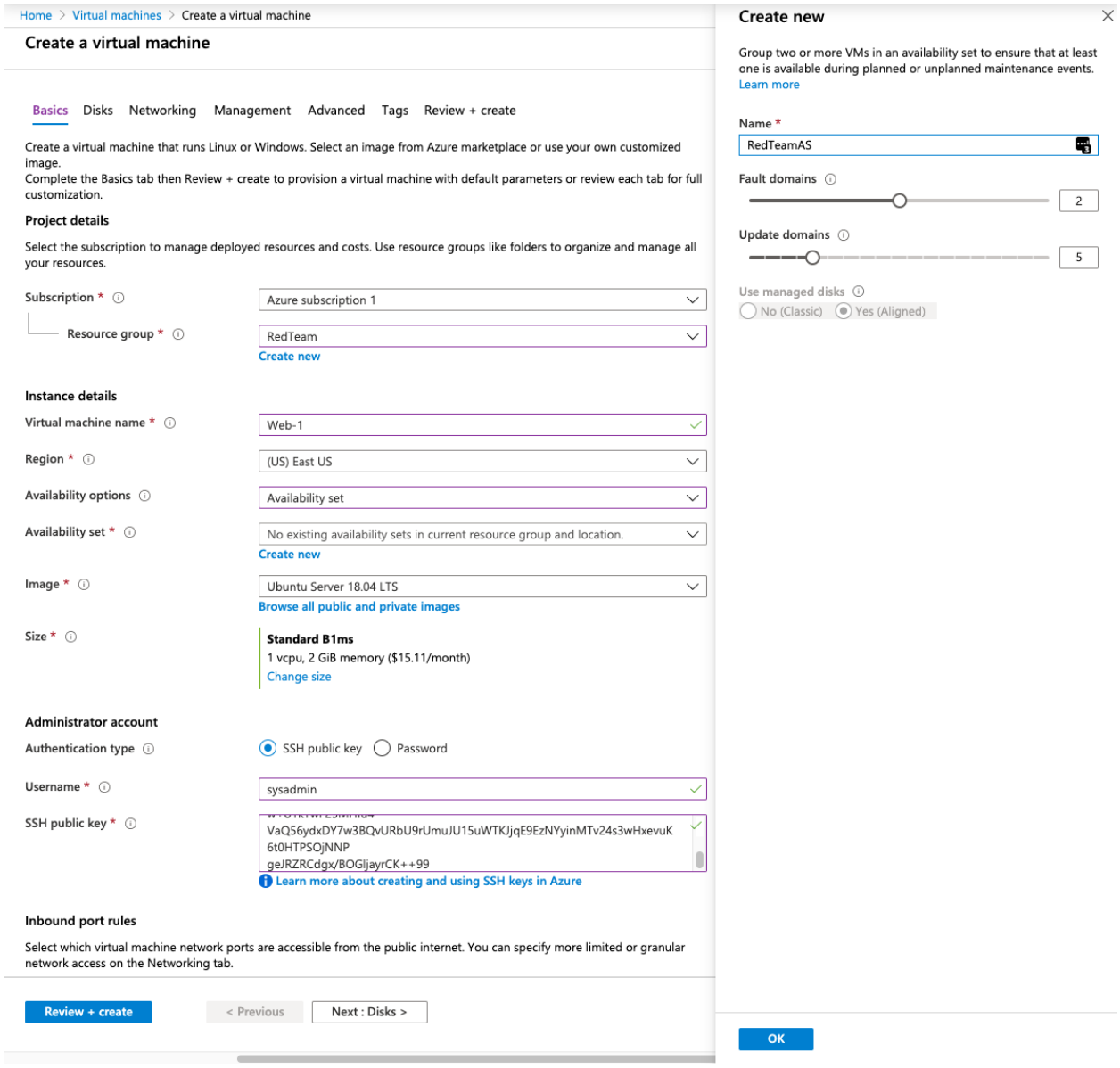
Create 2 more new VMs. Keep the following in mind when configuring these VM's:

- Each VM should be named "Web-1" and "Web-2"
- These VM's need to be in the same resource group you are using for all other resources.
- The VM's should be located in the same region as your resource group and security group.
    - Note that availability of VM's in Azure could cause you to change the region where your VM's are created.
    - The goal is to create 3 machines in the same resource group attached to the same security group. If you cannot add 3 machines to the resource group and security group that you have, a new resource group and security group may need to be created in another region.
- The administrative username should make sense for this scenario. You should use the same admin name for all 3 machines. Make sure to take a note of this name as you will need it to login later.
- SSH Key: Later in the lab setup, we will overwrite these SSH keys. For now, use the SSH key that you created in the first VM setup.
    - Run: `cat ~/.ssh/id_rsa.pub` and copy the key.
- Choose the VM option that has:
    - Whose offering is Standard - B1ms
    - 1 CPU
    - 2 RAM

Note: These web machines should have *2 GB* of RAM and the Jump-Box only needs *1 GB*. All 3 machines should only have *1 vCPU* because the free Azure account only allows *4 vCPU's* in total per region.

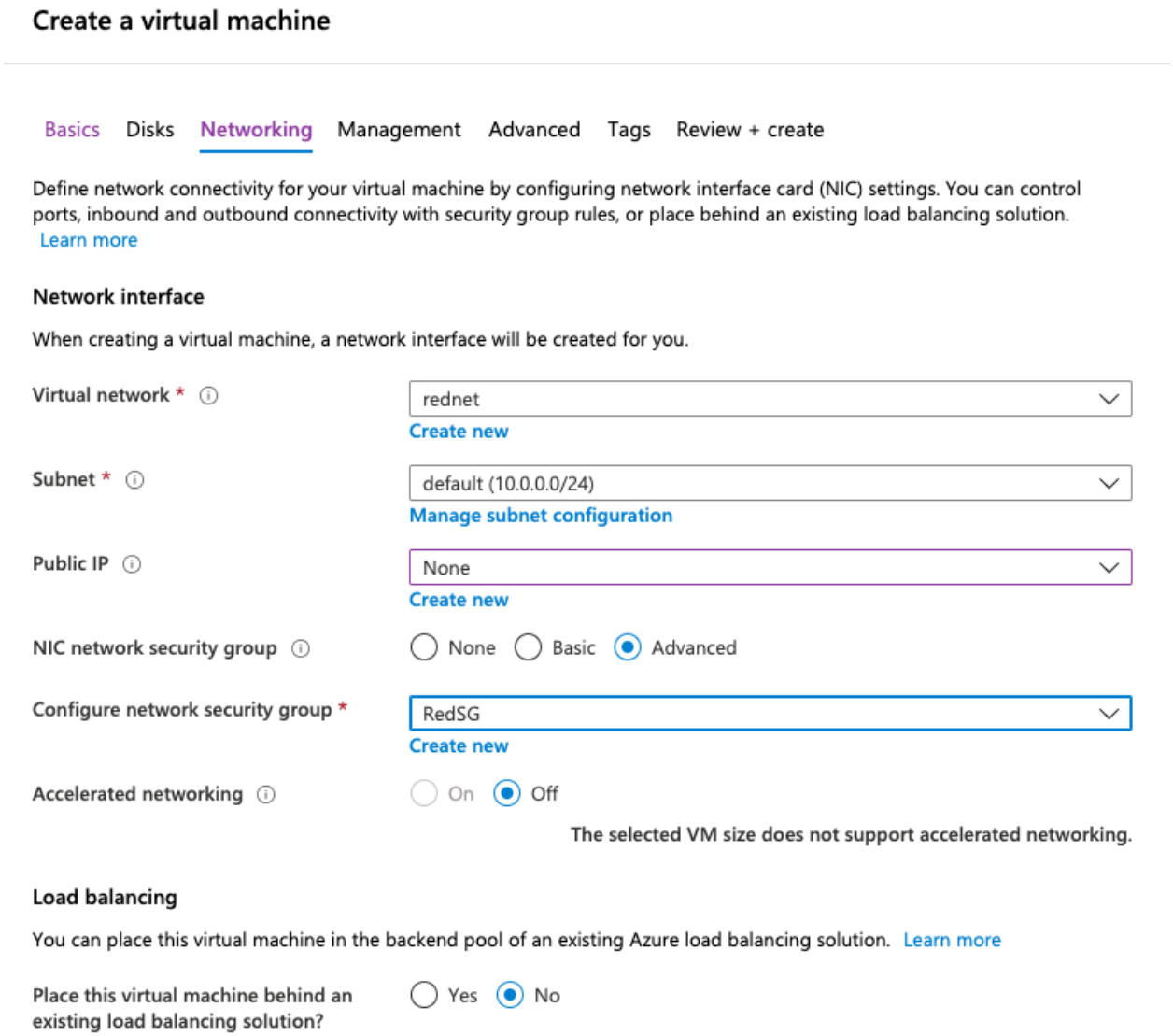
VERY IMPORTANT: Make sure both of these VM's are in the same availability Set. Machines that are not in the same availability set cannot be added to the same load balancer later, and will have to be deleted and recreated in the same availability set.

- Under Availability Options, select 'Availability Set'. Click on 'Create New' under the Availability set. Give it an appropriate name. After creating it on the first VM, choose it for the second VM.



In the Networking tab and set the following settings:

- Virtual network: Choose the VNet you created for the Red Team.
- Subnet: Choose the subnet that you created earlier.
- Public IP: NONE! Make sure these web VM's do not have a public IP address.



- NIC network security group: Choose the Advanced option so we can specify our custom security group.
- Configure network security group: Choose your Red Team network security group.
- Accelerated networking: Keep as the default setting (Off).
- In the Networking settings, take note of the VM URL. You may use it later.
- Load balancing: Keep as the default setting (No).

NOTE: Notice that these machines will not be accessible at this time because our security group is blocking all traffic. We will configure access to these machines in a later activity.

The final WebVM's should resemble the following:

PRODUCT DETAILS

Standard B1ms
by Microsoft
Terms of use | Privacy policy

Subscription credits apply
0.0207 USD/hr
Pricing for other VM sizes

TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.

Basics

Subscription	Azure subscription 1
Resource group	RedTeam
Virtual machine name	Web-1
Region	East US
Availability options	Availability set
Availability set	(new) RedTeamAS
Authentication type	SSH public key
Key pair name	sysadmin

Disks

OS disk type	Premium SSD
Use managed disks	Yes
Use ephemeral OS disk	No

Networking

Virtual network	rednet
Subnet	default (10.0.0.0/24)
Public IP	None
NIC network security group	RedSG
Accelerated networking	Off
Place this virtual machine behind an existing load balancing solution?	No

Management

Boot diagnostics	On
OS guest diagnostics	Off
Azure Security Center	None
Diagnostics storage account	redteamdiag583
System assigned managed identity	Off
Auto-shutdown	Off

Advanced

Extensions	None
Cloud init	No
Proximity placement group	None

Setting up your Jump Box Administration

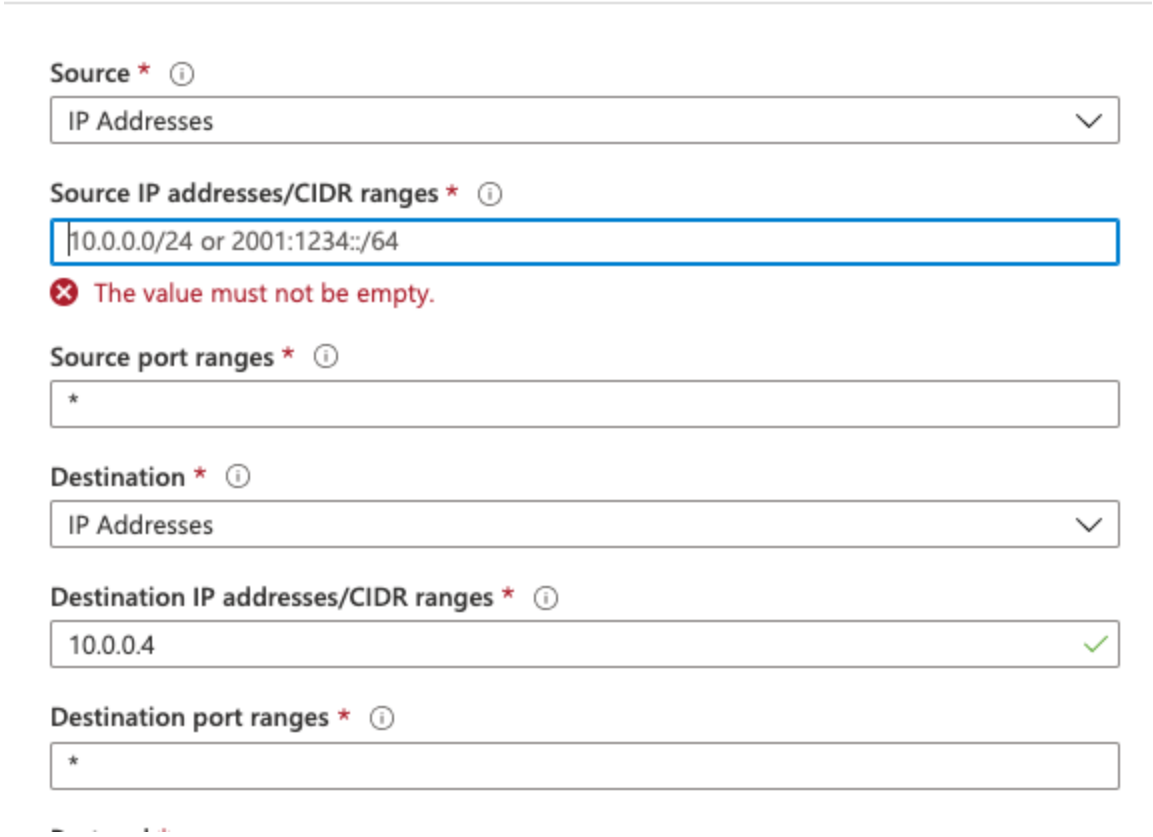
The goal of this activity was to create a security group rule to allow SSH connections only from your current IP address, and to connect to your new virtual machine for management.

---

1. Visit `whatsmyip.org` to get the IPv4 address of the network you are currently using.

Next, log into portal.azure.com to create a security group rule to allow SSH connections from your current IP address.

2. Find your security group listed under your resource group.
3. Create a rule allowing SSH connections from your IP address.
    - Choose Inbound security rules on the left.
    - Click + Add to add a rule.
        - Source: Use the IP Addresses setting, with your IP address in the field.
        - Source port ranges: Set to Any or * here.
        - Destination: This can be set VirtualNetwork but a better setting is to specify the internal IP of your jump box to really limit this traffic.
        - Destination port ranges: Since we only want to allow SSH, designate port `22`.
        - Protocol: Set to Any or TCP.
        - Action: Set to Allow traffic.
        - Priority: This must be a lower number than your rule to deny all traffic, i.e., less than 4,096.
        - Name: Name this rule anything you like, but it should describe the rule. For example: `SSH`.
        - Description: Write a short description similar to: "Allow SSH from my IP."

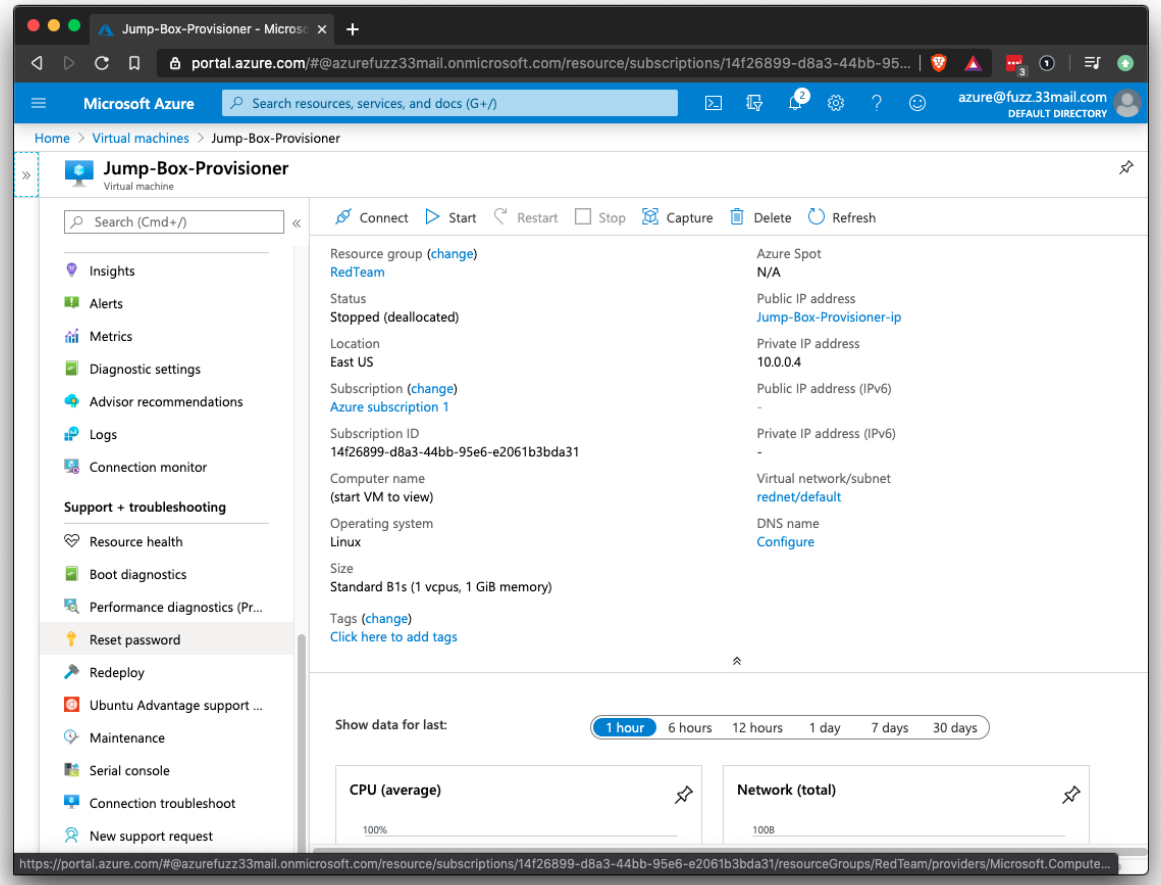


4. Use your command line to SSH to the VM for administration. Windows users should use GitBash.
    - The command to connect is `ssh admin-username@VM-public-IP`.
    - Use the username you previously set up. (Your SSH key should be used automatically.)
5. Once you are connected, check your `sudo` permissions.
    - Run the command `sudo -l`.
    - Notice that your admin user has full `sudo` permissions without requiring a password.

Please note that your public IP address will change depending on your location.

- In a normal work environment, you would set up a static IP address to avoid continually creating rules to allow access to your cloud machine.
- In our case, you will need to create another security rule allowing your home network to access your Azure VM.

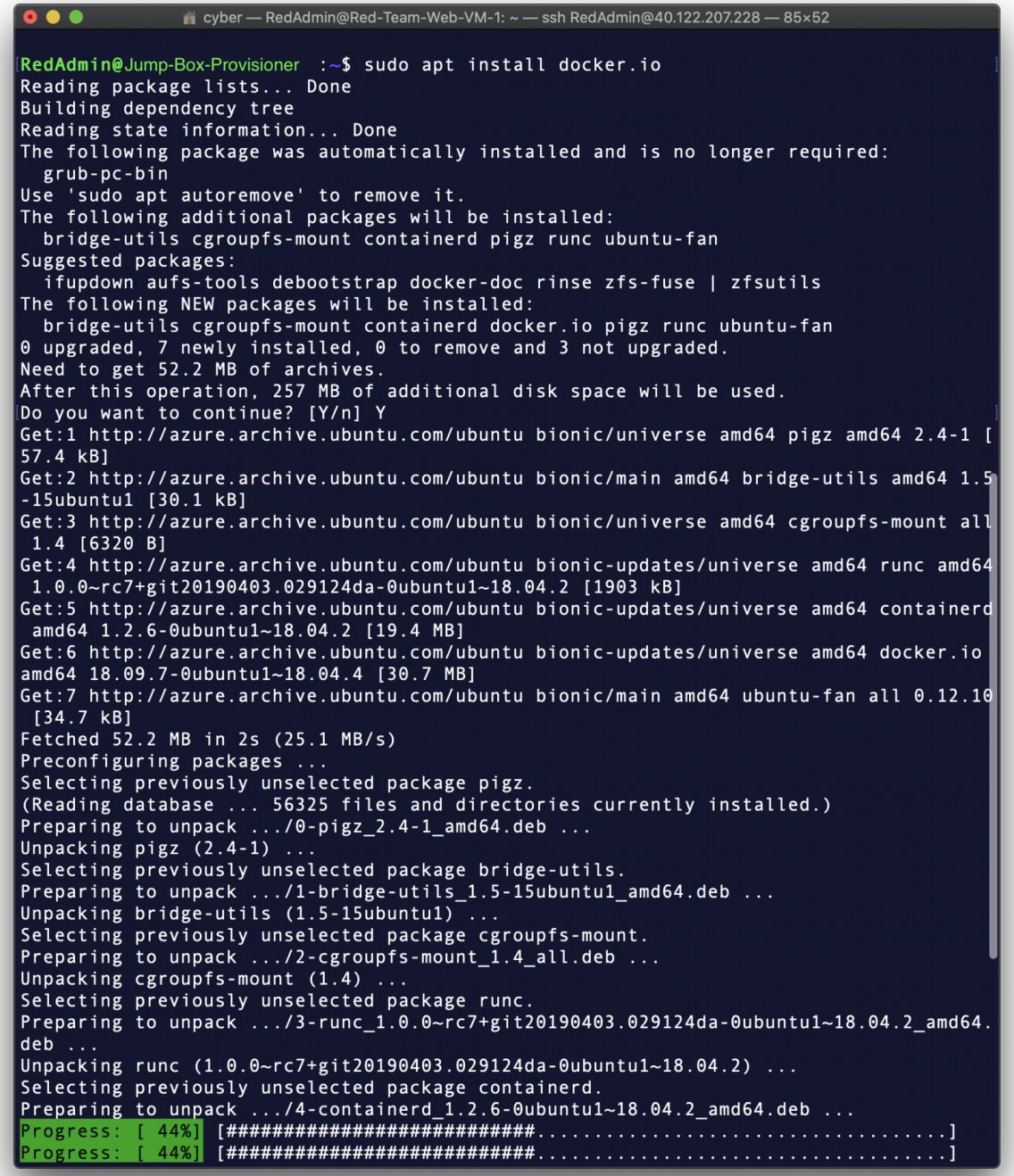
NOTE: If you need to reset your SSH key, you can do so in the VM details page by selecting 'Reset Password' on the left had column.



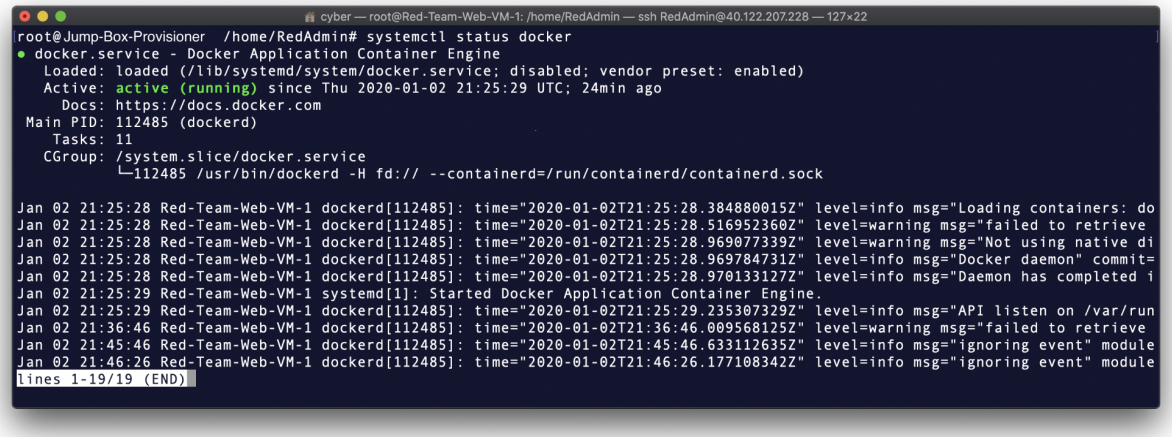
Docker Container Setup

The goal of this activity was to configure your jump box to run Docker containers and to install a container.

1. Start by installing `docker.io` on your Jump box.
    - Run `sudo apt update` then `sudo apt install docker.io`



2. Verify that the Docker service is running.
    - Run `sudo systemctl status docker`
        - Note: If the Docker service is not running, start it with `sudo systemctl start docker`.



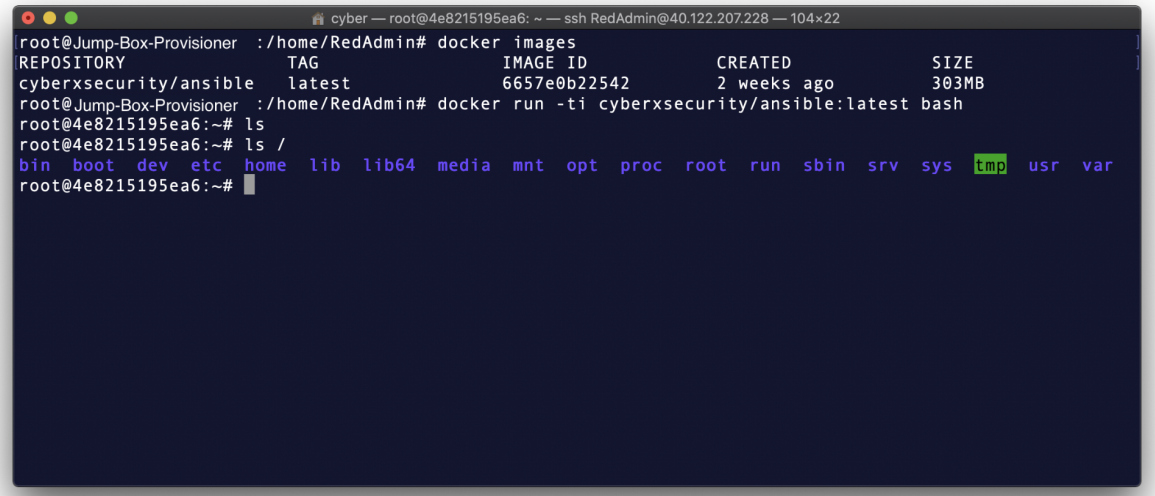
3. Once Docker is installed, pull the container `cyberxsecurity/ansible`.
    - Run `sudo docker pull cyberxsecurity/ansible`.

```
RedAdmin@Jump-Box-Provisioner :~$ sudo docker pull cyberxsecurity/ansible
Using default tag: latest
latest: Pulling from cyberxsecurity/ansible
7ddbc47eeb70: Pull complete
c1bbdc448b72: Pull complete
8c3b70e39044: Pull complete
45d437916d57: Pull complete
78440e84779a: Pull complete
b6f0115afe25: Pull complete
Digest: sha256:a245954c7eda5d15ec6bc8cca2ab129e12491e711de0db63344ebf40fa35d26b
Status: Downloaded newer image for cyberxsecurity/ansible:latest
RedAdmin@Jump-Box-Provisioner :~$
```

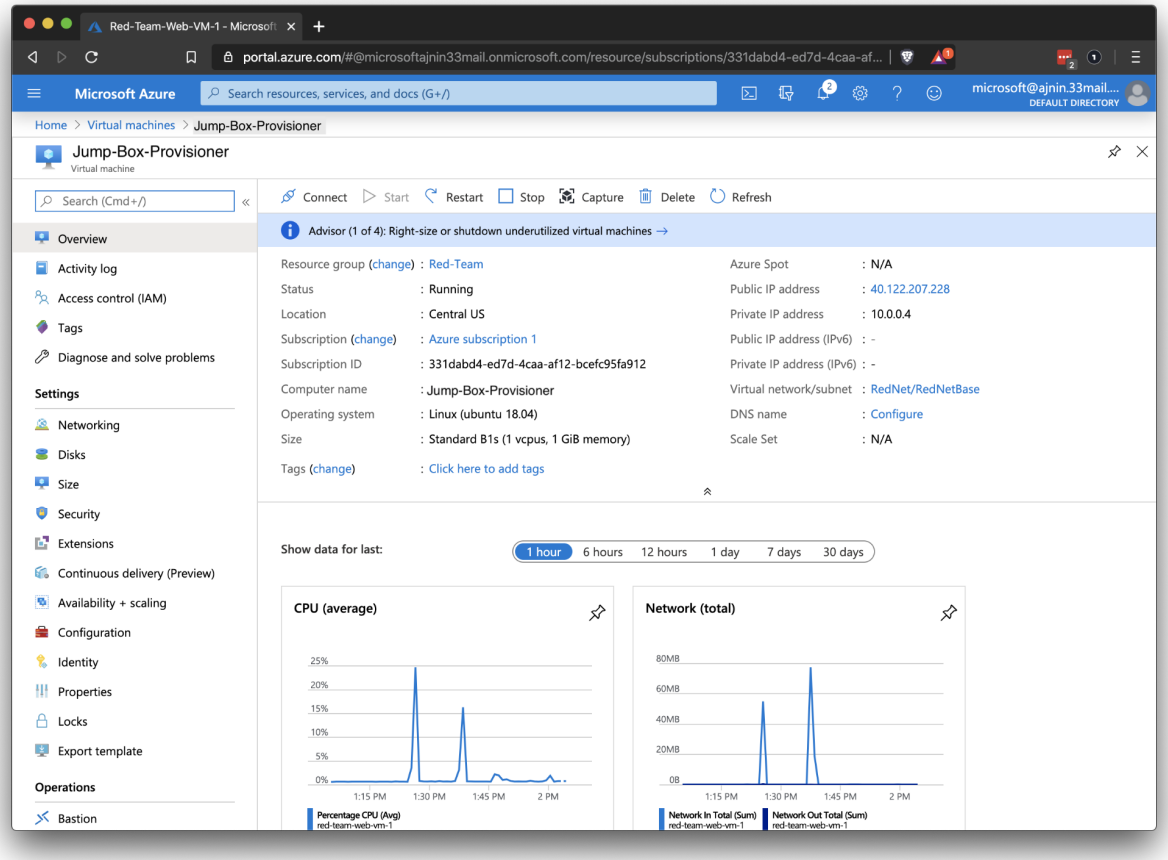
- You can also switch to the root user so you don't have to keep typing `sudo`.

- Run `sudo su`.

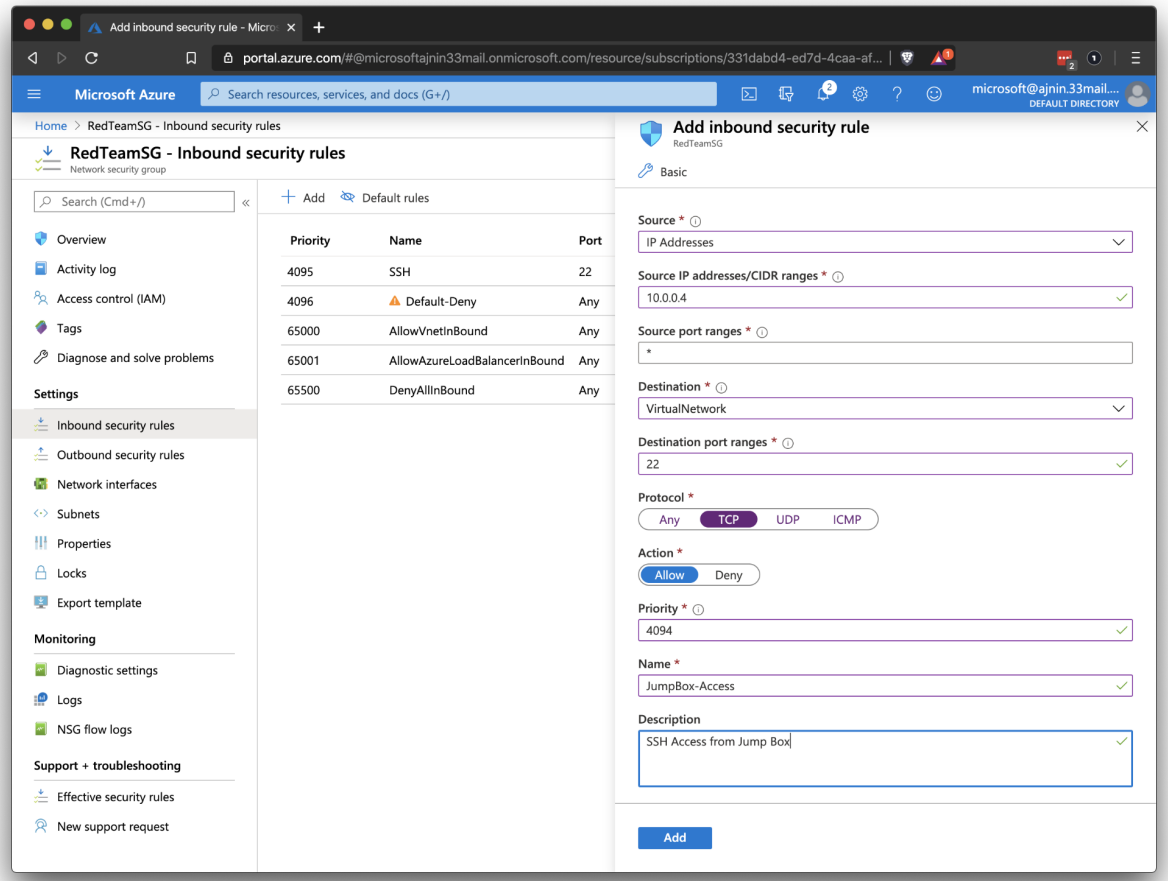
4. Launch the Ansible container and connect to it using the appropriate Docker commands.
    - Run `docker run -ti cyberxsecurity/ansible:latest bash` to start the container.
    - Run `exit` to quit.



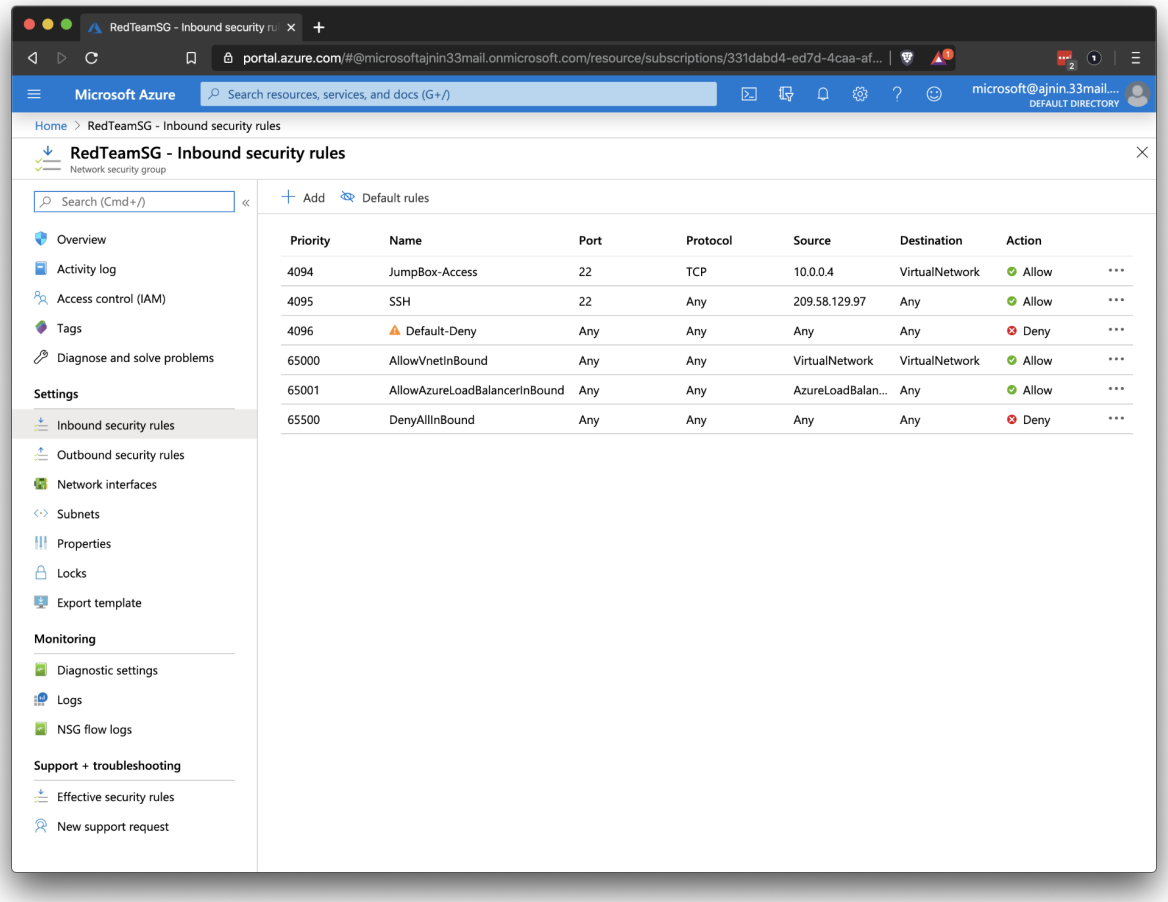
5. Create a new security group rule that allows your jump box machine full access to your VNet.
    - Get the private IP address of your jump box.



- Go to your security group settings and create an inbound rule. Create rules allowing SSH connections from your IP address.
    - Source: Use the IP Addresses setting with your jump box's internal IP address in the field.
    - Source port ranges: Any or * can be listed here.
    - Destination: Set to VirtualNetwork.
    - Destination port ranges: Only allow SSH. So, only port `22`.
    - Protocol: Set to Any or TCP.
    - Action: Set to Allow traffic from your jump box.
    - Priority: Priority must be a lower number than your rule to deny all traffic.
    - Name: Name this rule anything you like, but it should describe the rule. For example: `SSH from Jump Box`.
    - Description: Write a short description similar to: "Allow SSH from the jump box IP."



Your final security group rules should be similar to this:



Setup your Provisioner

In this activity, you launched a new VM from the Azure portal that could only be accessed using a new SSH key from the container running inside your jump box.

1. Connect to your Ansible container. Once you're connected, create a new SSH key and copy the public key.
    - Run `docker images` to view your image.

Run `docker run -it cyberxsecurity/ansible /bin/bash` to start your container and connect to it. (Note that the prompt changes.)
`root@Red-Team-Web-VM-1:/home/RedAdmin# docker run -it cyberxsecurity/ansible /bin/bash`

    - `root@23b86e1d62ad:~#`

Run `ssh-keygen` to create an SSH key.

```
root@23b86e1d62ad:~# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:gzoKliTqbxvTFhrNU7ZwUHEx7xAA7MBPS2Wq3HdJ6rw root@23b86e1d62ad
The key's randomart image is:
+---[RSA 2048]----+
|  . .o+*o=.      |
|   o ++ . +      |
|    *o.+ o .     |
|  . =+=.+ +      |
|.. + *.+So .     |
|+ . +.* ..       |
|oo +oo o         |
|o. o+.  .        |
| .+o.  E         |
+----[SHA256]-----+
```

- `root@23b86e1d62ad:~#`

Run `ls .ssh/` to view your keys.

```
root@23b86e1d62ad:~# ls .ssh/
```

- `id_rsa  id_rsa.pub`

Run `cat .ssh/id_rsa.pub` to display your public key.

```
root@23b86e1d62ad:~# cat .ssh/id_rsa.pub
```

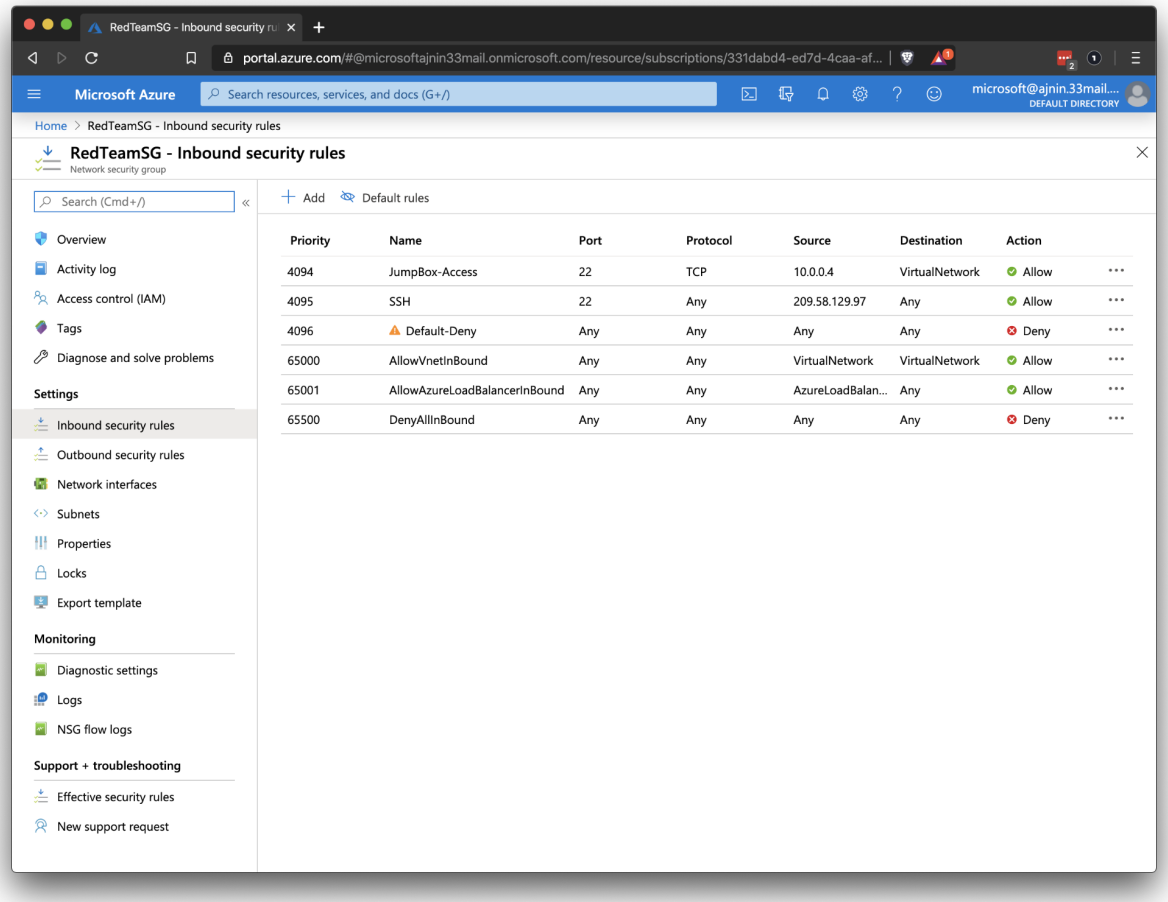
- `ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDz5KX3urPPKbYRKS3J06wyw5Xj4eZRQTcg6u2LpnSsXwPWYBpCdF5lE3tJlbp7AsnXlXpq2G0oAy5dcLJX2anpfaEBTEvZ0mFBS24AdNnF3ptan5SmEM/`
- Copy your public key string.

Return to the Azure portal and locate one of your web-vm's details page.

- Reset your Vm's password and use your container's new public key for the SSH user.

2.

- Get the internal IP for your new VM from the Details page.



3. After your VM launches, test your connection using `ssh` from your jump box Ansible container.
    - Note: If only TCP connections are enabled for SSH in your security group rule, ICMP packets will not be allowed, so you will not be able to use `ping`.

```
root@23b86e1d62ad:~# ping 10.0.0.6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
^C
--- 10.0.0.6 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3062ms

root@23b86e1d62ad:~#
root@23b86e1d62ad:~# ssh [username]@[IP Address]
The authenticity of host '10.0.0.6 (10.0.0.6)' can't be established.
ECDSA key fingerprint is SHA256:7Wd1cStyhq5HihBf+7TQgjIQe2uHP6arx2qZ1YrPAP4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.6' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 5.0.0-1027-azure x86_64)
```

```
* Documentation:  https://help.ubuntu.com
* Management:     https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of Mon Jan  6 18:49:56 UTC 2020

System load:  0.01              Processes:           108
Usage of /:   4.1% of 28.90GB   Users logged in:     0
Memory usage: 36%               IP address for eth0: 10.0.0.6
Swap usage:   0%


0 packages can be updated.
0 updates are security updates.


Last login: Mon Jan  6 18:33:30 2020 from 10.0.0.4
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

4. `ansible@Pentest-1:~$`
    - Exit this SSH session by running `exit`.

Locate the Ansible config file and hosts file.

```
root@1f08425a2967:~# ls /etc/ansible/
```

5. `ansible.cfg  hosts  roles`
    - Add this machine's internal IP address to the Ansible hosts file.
    - Open the file with `nano /etc/ansible/hosts`.
    - Uncomment the `[webservers]` header line.
    - Add the internal IP address under the `[webservers]` header.
        - Add the python line: `ansible_python_interpreter=/usr/bin/python3` besides each IP.

```
# This is the default ansible 'hosts' file.
 #
 # It should live in /etc/ansible/hosts
 #
 #   - Comments begin with the '#' character
 #   - Blank lines are ignored
 #   - Groups of hosts are delimited by [header] elements
 #   - You can enter hostnames or ip addresses
 #   - A hostname/ip can be a member of multiple groups
 # Ex 1: Ungrouped hosts, specify before any group headers.

 ## green.example.com
 ## blue.example.com
 ## 192.168.100.1
```

```
## 192.168.100.10

# Ex 2: A collection of hosts belonging to the 'webservers' group

[webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
10.0.0.6 ansible_python_interpreter=/usr/bin/python3
              10.0.0.7 ansible_python_interpreter=/usr/bin/python3
```

6.
7. Change the Ansible configuration file to use your administrator account for SSH connections.
    - Open the file with `nano /etc/ansible/ansible.cfg` and scroll down to the `remote_user` option.
    - Uncomment the `remote_user` line and replace `root` with your admin username using this format: - `remote_user = <user-name-for-web-VMs>`

Example:

```
# What flags to pass to sudo
# WARNING: leaving out the defaults might create unexpected behaviours
#sudo_flags = -H -S -n

# SSH timeout
#timeout = 10

# default user to use for playbooks if user is not specified
# (/usr/bin/ansible will use current user as default)
remote_user = sysadmin

# logging is off by default unless this path is defined
# if so defined, consider logrotate
#log_path = /var/log/ansible.log

# default module name for /usr/bin/ansible
#module_name = command
```

8.
9. Test an Ansible connection using the appropriate Ansible command.

If you used `ansible_python_interpreter=/usr/bin/python3` your output should look like:

```
10.0.0.5 | SUCCESS => {
"changed": false,
"ping": "pong"
}
```

```
10.0.0.6 | SUCCESS => {
            "changed": false,
            "ping": "pong"
}
```

If that line isn't present, you will get a warning like this:

```
root@1f08425a2967:~# ansible -m ping all
[DEPRECATION WARNING]: Distribution Ubuntu 18.04 on host 10.0.0.6 should use
/usr/bin/python3, but is using /usr/bin/python for backward compatibility with
prior Ansible releases. A future Ansible release will default to using the
discovered platform python for this host. See https://docs.ansible.com/ansible/
2.9/reference_appendices/interpreter_discovery.html for more information. This
feature will be removed in version 2.12. Deprecation warnings can be disabled
by setting deprecation_warnings=False in ansible.cfg.
10.0.0.6 | SUCCESS => {
            "ansible_facts": {
                        "discovered_interpreter_python": "/usr/bin/python"
            },
            "changed": false,
            "ping": "pong"
}
```

- Ignore the `[DEPRECATION WARNING]` or add the line `ansible_python_interpreter=/usr/bin/python3` next to each Ip address in the hosts file.

Setup your Ansible Playbooks

The goal here is to create an Ansible playbook that installed Docker and configure a VM with the DVWA web app.

---

1. Connect to your jump box, and connect to the Ansible container in the box.
    - If you stopped your container or exited it in the last activity, find it again using `docker container list -a`.

```
root@Red-Team-Web-VM-1:/home/RedAdmin# docker container list -a
CONTAINER ID        IMAGE                              COMMAND
CREATED             STATUS                             PORTS               NAMES
Exited (0) 2 minutes ago                               hardcore_brown
```

2. `a0d78be636f7        cyberxsecurity/ansible:latest   "bash"`
   `3 days ago`

- NOTE: In this example, the container is called `hardcore_brown` and this name is randomly generated. The name of your container will be different.
- Start the container again using `docker start [container_name]`.

```
root@Red-Team-Web-VM-1:/home/RedAdmin# docker start hardcore_brown
```

3. `hardcore_brown`
    - Get a shell in your container using `docker attach [container_name]`.

```
root@Red-Team-Web-VM-1:/home/RedAdmin# docker attach hardcore_brown
```

4. `root@1f08425a2967:~#`
5. Create a YAML playbook file that you will use for your configuration.

```
root@1f08425a2967:~# nano /etc/ansible/pentest.yml
```

The top of your YAML file should read similar to:

```
---
- name: Config Web VM with Docker
    hosts: web
    become: true
    tasks:
```

Use the Ansible `apt` module to install `docker.io` and `python3-pip`: Note: `update_cache` must be used here, or `docker.io` will not install. (this is the equivalent of running `apt update`)

```
  - name: docker.io
    apt:
                  update_cache: yes
      name: docker.io
      state: present

  - name: Install pip3
    apt:
      force_apt_get: yes
      name: python3-pip
    ●       state: present
```

Note: `update_cache: yes` is needed to download and install docker.io

Use the Ansible `pip` module to install `docker`:

```
  - name: Install Python Docker module
    pip:
      name: docker
    ●       state: present
```

Note: Here we are installing the Python Docker Module, so Ansible can then utilize that module to control docker containers. More about the Python Docker Module HERE

- Use the Ansible `docker-container` module to install the `cyberxsecurity/dvwa` container.
    - Make sure you publish port `80` on the container to port `80` on the host.

```
- name: download and launch a docker web container
   docker_container:
     name: dvwa
     image: cyberxsecurity/dvwa
     state: started
     restart_policy: always
```

- `published_ports: 80:80`

NOTE: `restart_policy: always` will ensure that the container restarts if you restart your web vm. Without it, you will have to restart your container when you restart the machine.

You will also need to use the `systemd` module to restart the docker service when the machine reboots. That block looks like this:

```
   - name: Enable docker service
     systemd:
       name: docker
       enabled: yes
```

Run your Ansible playbook on the new virtual machine.
Your final playbook should read similar to:

```
---
- name: Config Web VM with Docker
  hosts: webservers
  become: true
  tasks:
  - name: docker.io
    apt:
      force_apt_get: yes
      update_cache: yes
      name: docker.io
      state: present

  - name: Install pip3
    apt:
      force_apt_get: yes
      name: python3-pip
      state: present

  - name: Install Docker python module
```

```
    pip:
      name: docker
      state: present

  - name: download and launch a docker web container
    docker_container:
      name: dvwa
      image: cyberxsecurity/dvwa
      state: started
      published_ports: 80:80

  - name: Enable docker service
    systemd:
      name: docker
```

3. `enabled: yes`

Running your playbook should produce an output similar to the following:

```
root@1f08425a2967:~# ansible-playbook /etc/ansible/pentest.yml

PLAY [Config Web VM with Docker]
***************************************************************

TASK [Gathering Facts]
*********************************************************************
ok: [10.0.0.6]

TASK [docker.io]
*****************************************************************************
[WARNING]: Updating cache and auto-installing missing dependency: python-apt

changed: [10.0.0.6]

TASK [Install pip3]
*****************************************************************************
changed: [10.0.0.6]

TASK [Install Docker python module]
*********************************************************
changed: [10.0.0.6]

TASK [download and launch a docker web container]
*********************************************
changed: [10.0.0.6]

PLAY RECAP
*****************************************************************************
******
```

- `10.0.0.6 : ok=5 changed=4 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0`

4. To test that DVWA is running on the new VM, SSH to the new VM from your Ansible container.
    - SSH to your container:

```
root@1f08425a2967:~# ssh sysadmin@10.0.0.6
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 5.0.0-1027-azure x86_64)

* Documentation:  https://help.ubuntu.com
* Management:     https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

  System information as of Mon Jan  6 20:01:03 UTC 2020

  System load:  0.01              Processes:              122
  Usage of /:   9.9% of 28.90GB   Users logged in:        0
  Memory usage: 58%               IP address for eth0:    10.0.0.6
  Swap usage:   0%                IP address for docker0: 172.17.0.1


18 packages can be updated.
0 updates are security updates.
```

5. `Last login: Mon Jan  6 19:33:51 2020 from 10.0.0.4`
    - Run `curl localhost/setup.php` to test the connection. If everything is working, you should get back some HTML from the DVWA container.

```
ansible@Pentest-1:~$ curl localhost/setup.php

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

    <title>Setup :: Damn Vulnerable Web Application (DVWA) v1.10
*Development*</title>

    <link rel="stylesheet" type="text/css" href="dvwa/css/main.css" />

    <link rel="icon" type="\image/ico" href="favicon.ico" />

    <script type="text/javascript" src="dvwa/js/dvwaPage.js"></script>
```

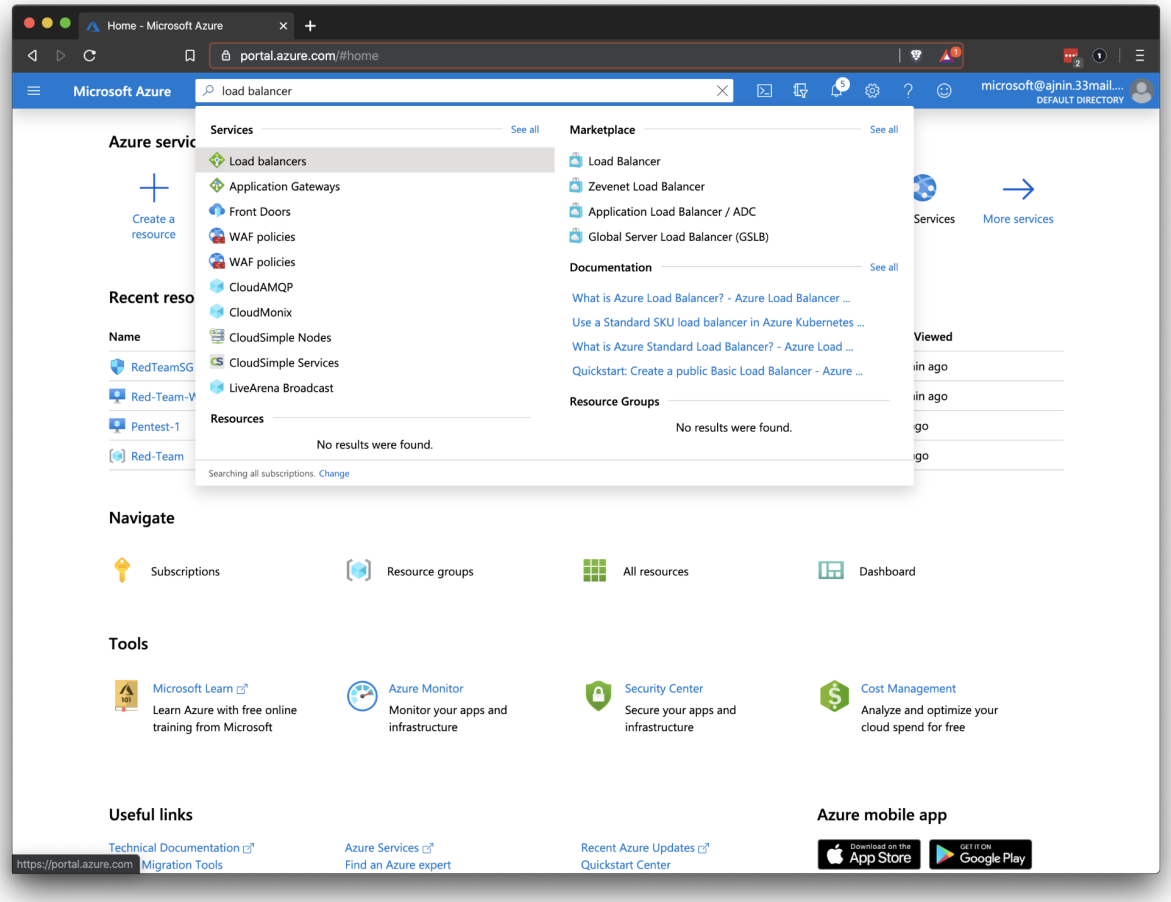
6. `</head>`

Setting up the Load Balancer

To complete this activity, you had to install a load balancer in front of the VM to distribute the traffic among more than one VM.

---

1. Create a new load balancer and assign it a static IP address.
    - Start from the homepage and search for "load balancer."



- Click + Add to create a new load balancer.
    - It should have a static public IP address.
    - Click Create to create the load balancer.

Create load balancer

Basics Tags Review + create

Azure load balancer is a layer 4 load balancer that distributes incoming traffic among healthy virtual machine instances. Load balancers uses a hash-based distribution algorithm. By default, it uses a 5-tuple (source IP, source port, destination IP, destination port, protocol type) hash to map traffic to available servers. Load balancers can either be internet-facing where it is accessible via public IP addresses, or internal where it is only accessible from a virtual network. Azure load balancers also support Network Address Translation (NAT) to route traffic between public and private IP addresses. Learn more.

Project details

Subscription * Azure subscription 1

Resource group * RedTeam

Create new

Instance details

Name * Red-Team-LB

Region * (US) East US

Type * Internal Public

SKU * Basic Standard

Public IP address

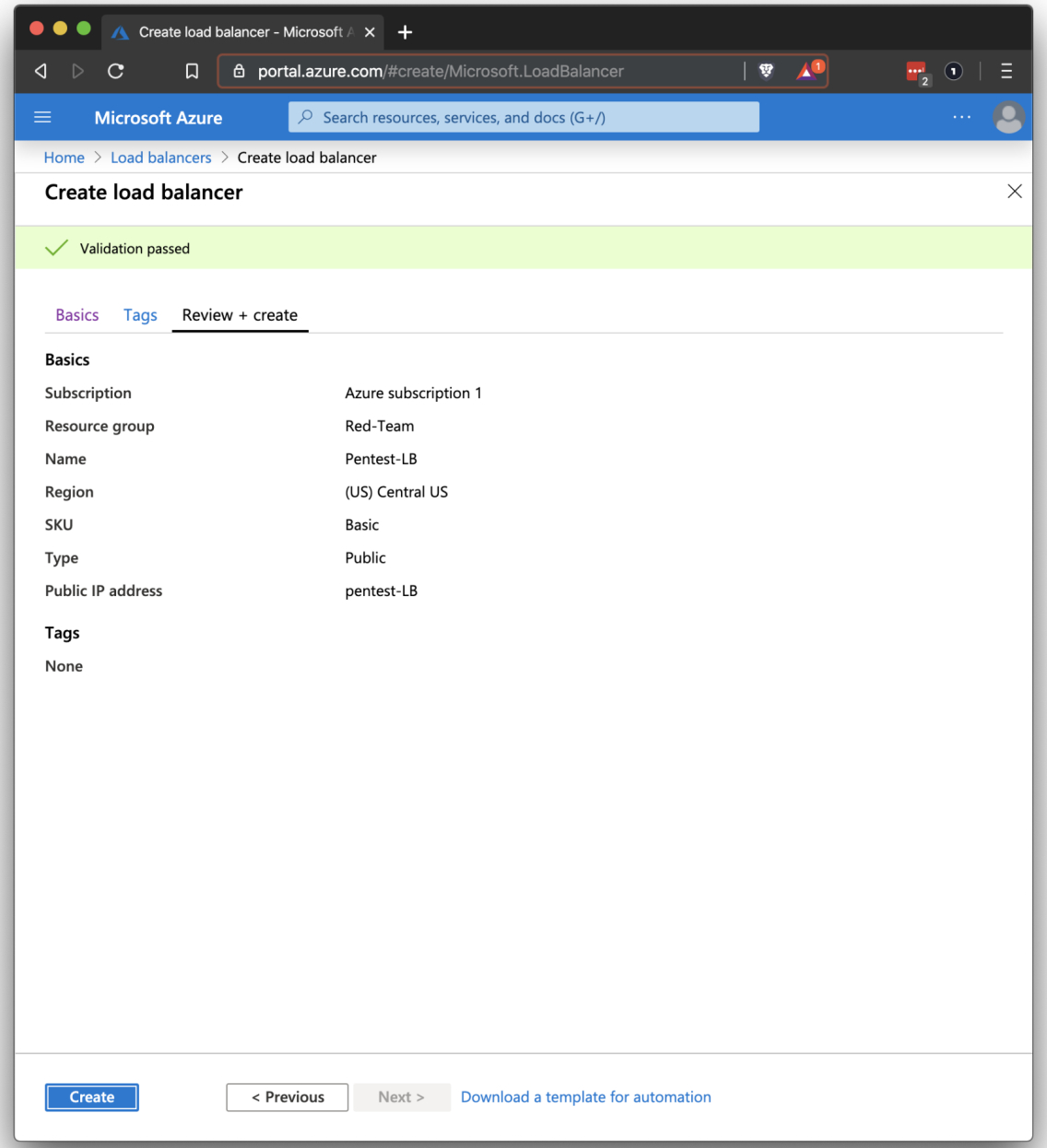
Public IP address * Create new Use existing

Public IP address name * Red-Team-LB

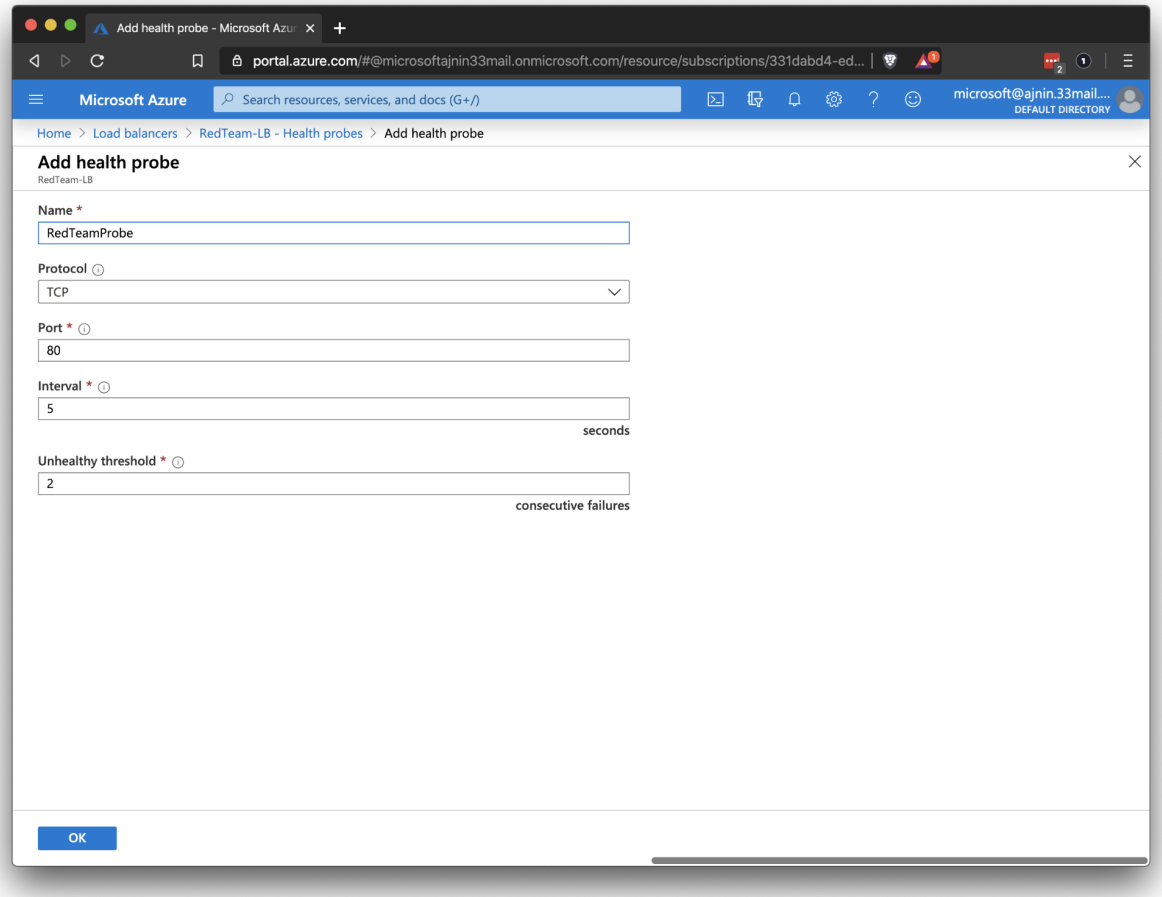
Public IP address SKU Basic

Assignment * Dynamic Static

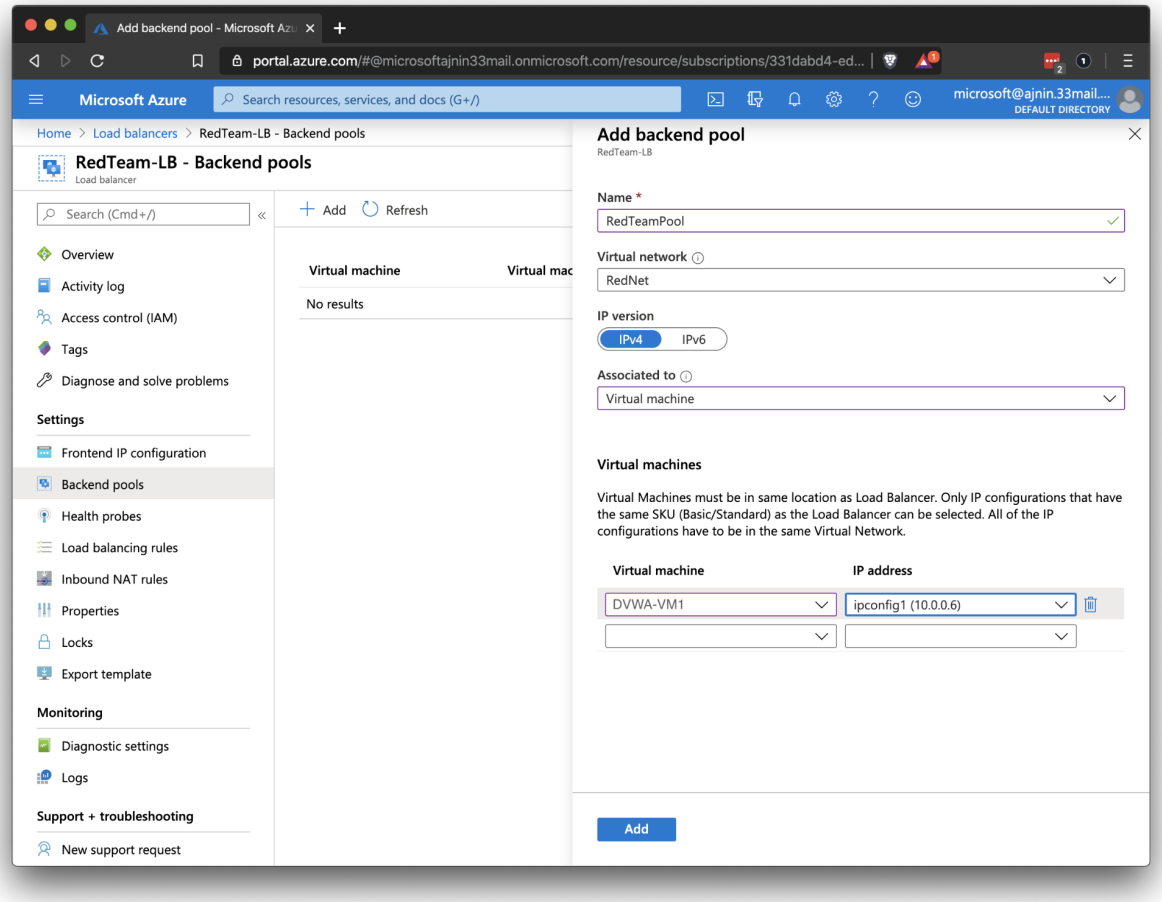
Add a public IPv6 address No Yes


Create load balancer - Microsoft A
portal.azure.com/#create/Microsoft.LoadBalancer
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Load balancers > Create load balancer
Create load balancer
Validation passed
Basics Tags Review + create
Basics
Subscription Azure subscription 1
Resource group Red-Team
Name Pentest-LB
Region (US) Central US
SKU Basic
Type Public
Public IP address pentest-LB
Tags
None
Create
< Previous
Next >
Download a template for automation

2. Add a health probe to regularly check all the VMs and make sure they are able to receive traffic.



3. Create a backend pool and add BOTH of your VM's to it.

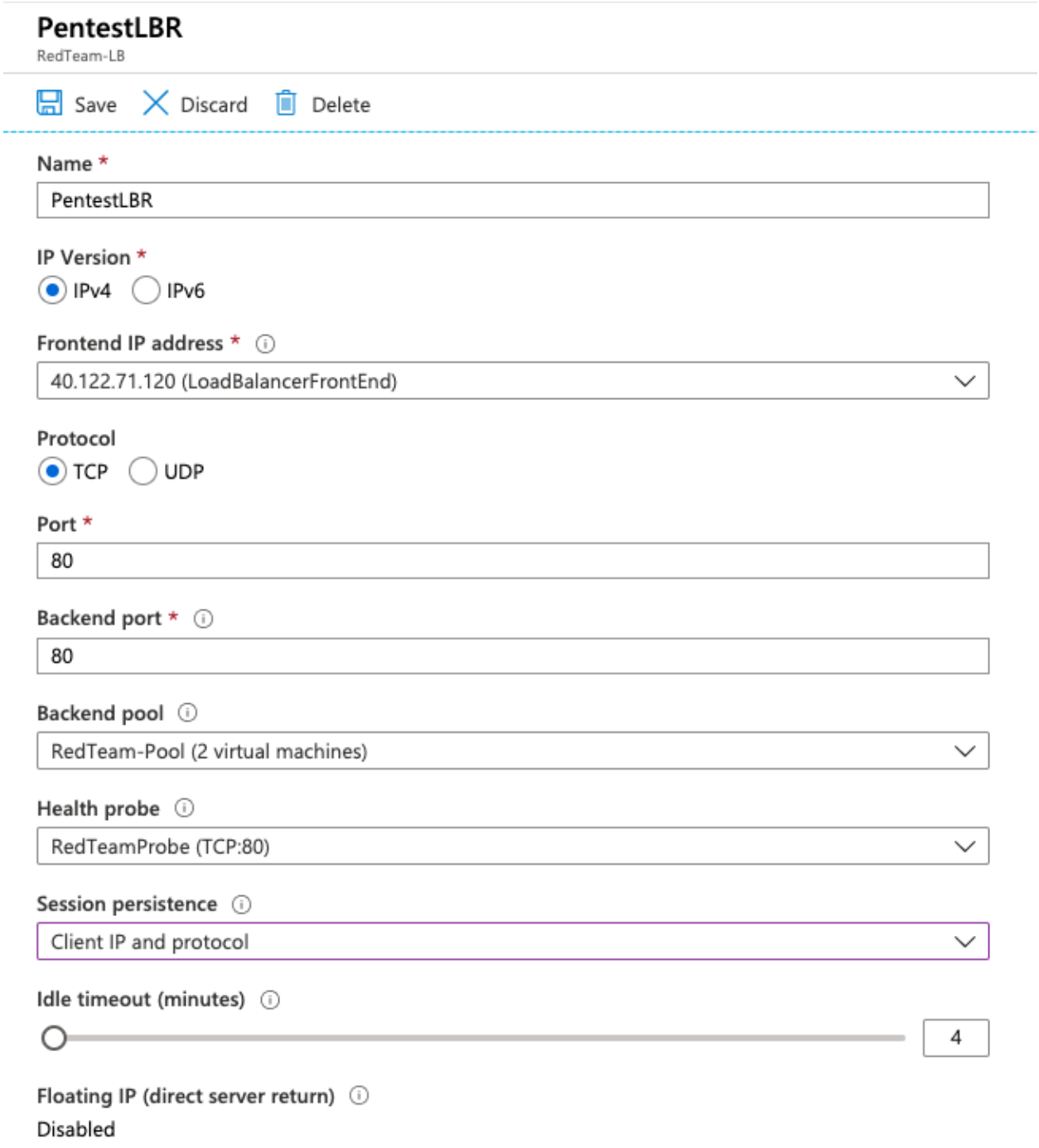


Setting up the NSG to expose port 80

To complete this activity, you had to configure the load balancer and security group to work together to expose port `80` of the VM to the internet.

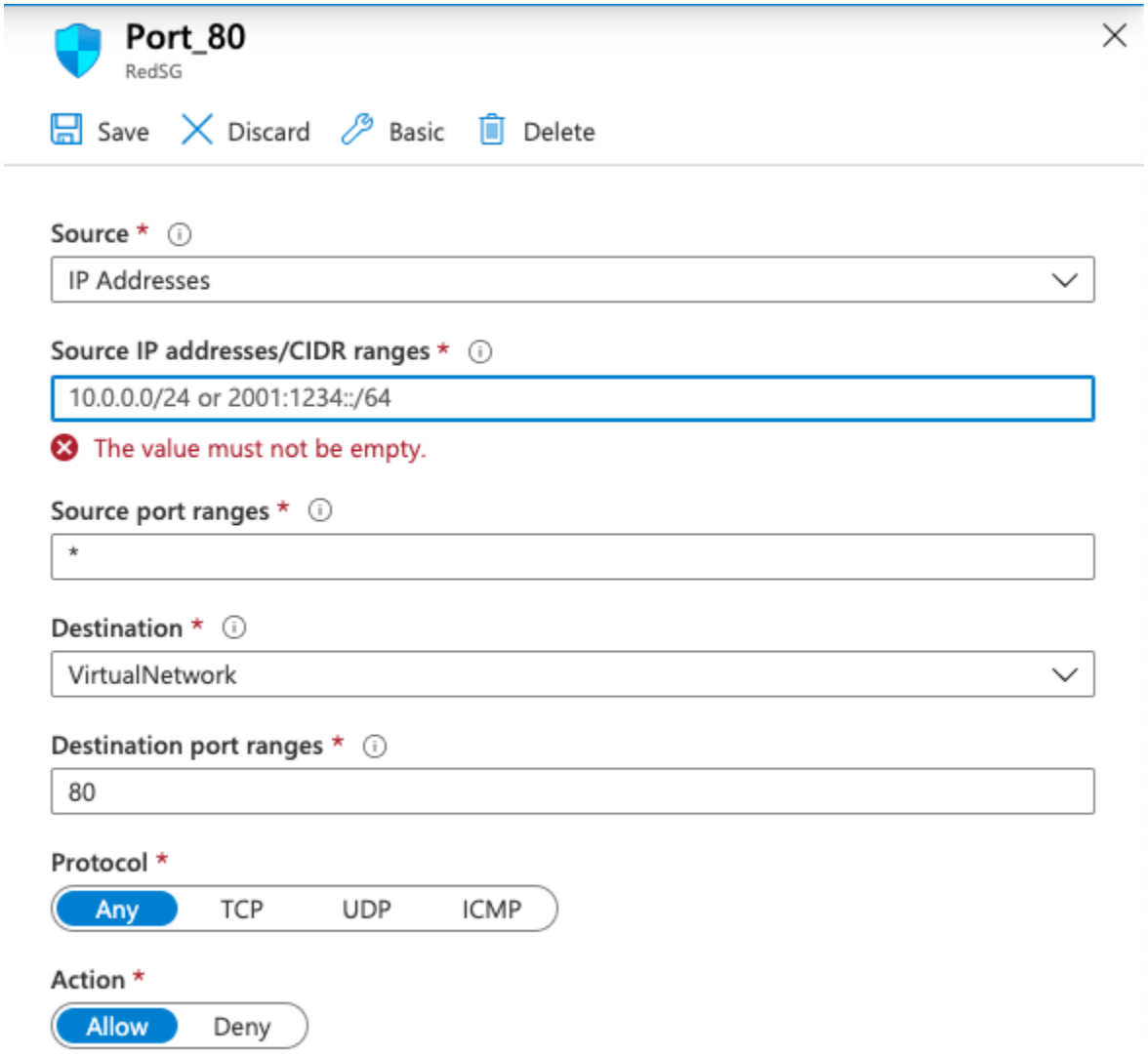
1. Create a load balancing rule to forward port `80` from the load balancer to your Red Team VNet.
    - Name: Give the rule an appropriate name that you will recognize later.
    - IP Version: This should stay on IPv4.
    - Frontend IP address: There should only be one option here.
    - Protocol: Protocol is TCP for standard website traffic.
    - Port: Port is `80`.
    - Backend port: Backend port is also `80`.

- Backend pool and Health probe: Select your backend pool and your health probe.
- Session persistence: This should be changed to Client IP and protocol.
    - Remember, these servers will be used by the Red Team to practice attacking machines. If the session changes to another server in the middle of their attack, it could stop them from successfully completing their training.
- Idle timeout: This can remain the default (4 minutes).
- Floating IP: This can remain the default (Disabled).



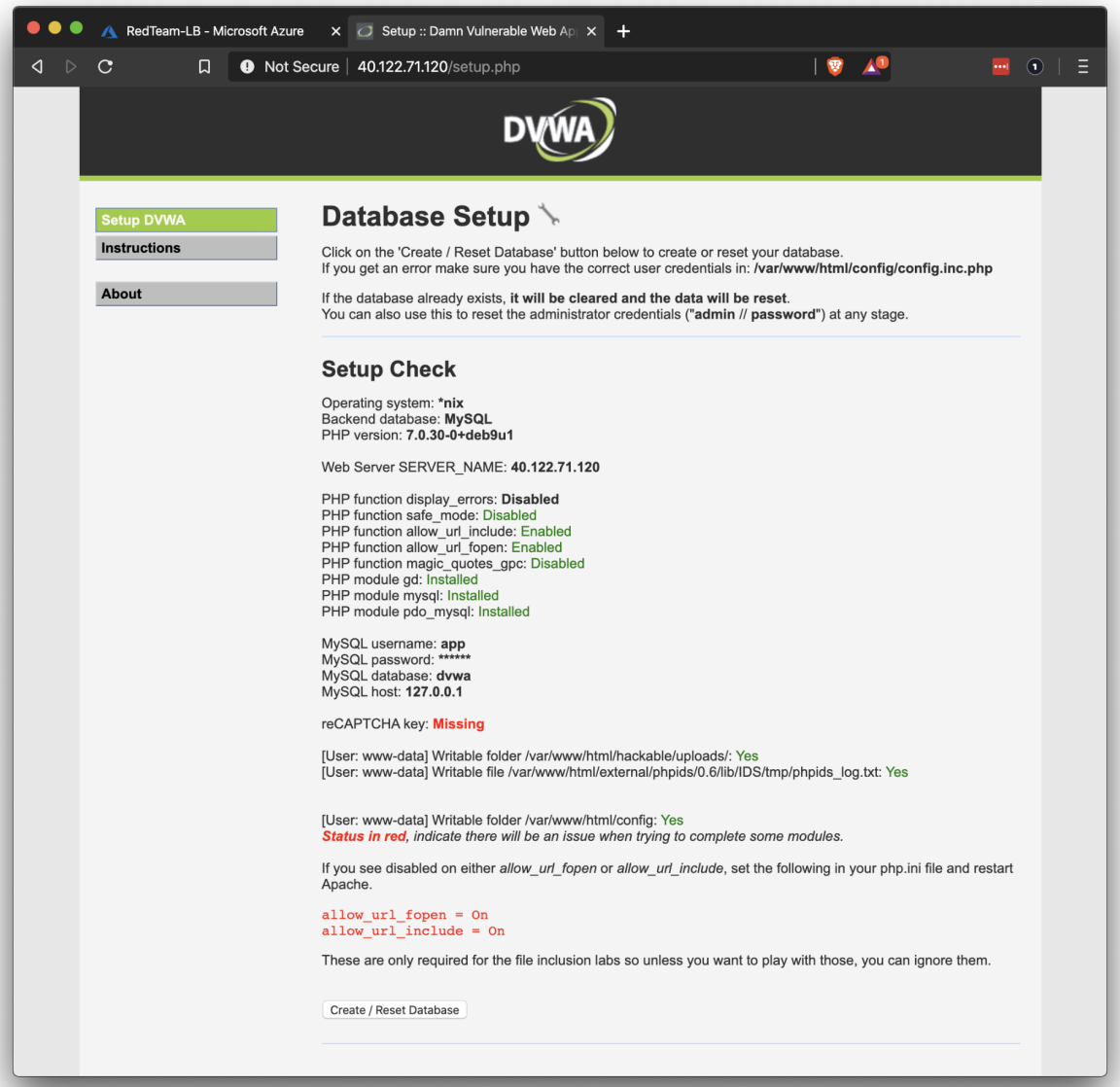
2. Create a new security group rule to allow port `80` traffic from the internet to your internal VNet.
    - Source: Change this your external IPv4 address.
    - Source port ranges: We want to allow Any source port, because they are chosen at random by the source computer.
    - Destination: We want the traffic to reach our VirtualNetwork.

- Destination port ranges: We only want to allow port `80`.
- Protocol: Set the standard web protocol of TCP or Any.
- Action: Set to Allow traffic.
- Name: Choose an appropriate name that you can recognize later.



3. Remove the security group rule that blocks *all* traffic on your vnet to allow traffic from your load balancer through.
    - Remember that when we created this rule we were blocking traffic from the allow rules that were already in place. One of those rules allows traffic from load balancers.
    - Removing your default deny all rule will allow traffic through.
4. Verify that you can reach the DVWA app from your browser over the internet.

- Open a web browser and enter the front-end IP address for your load balancer with `/setup.php` added to the IP address.
  - For example: `http://40.122.71.120/setup.php`



Note: With the stated configuration, you will not be able to access these machines from another location unless the security Group rule is changed.

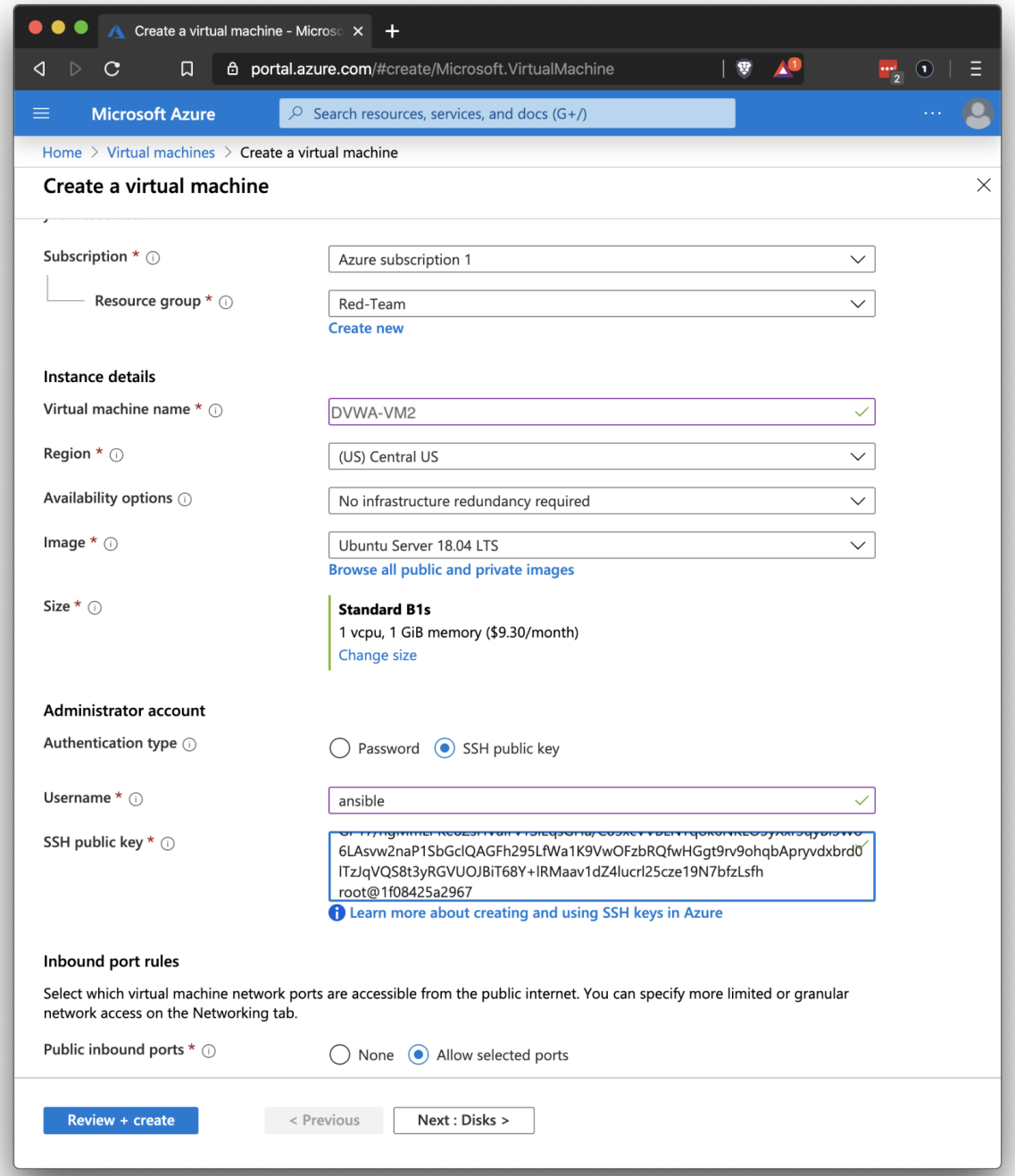
Setting up Redundancy

To complete this activity, you had to create a copy of your VM using your Ansible playbook for the configuration, and then place the VM in the backend pool for the load balancer.

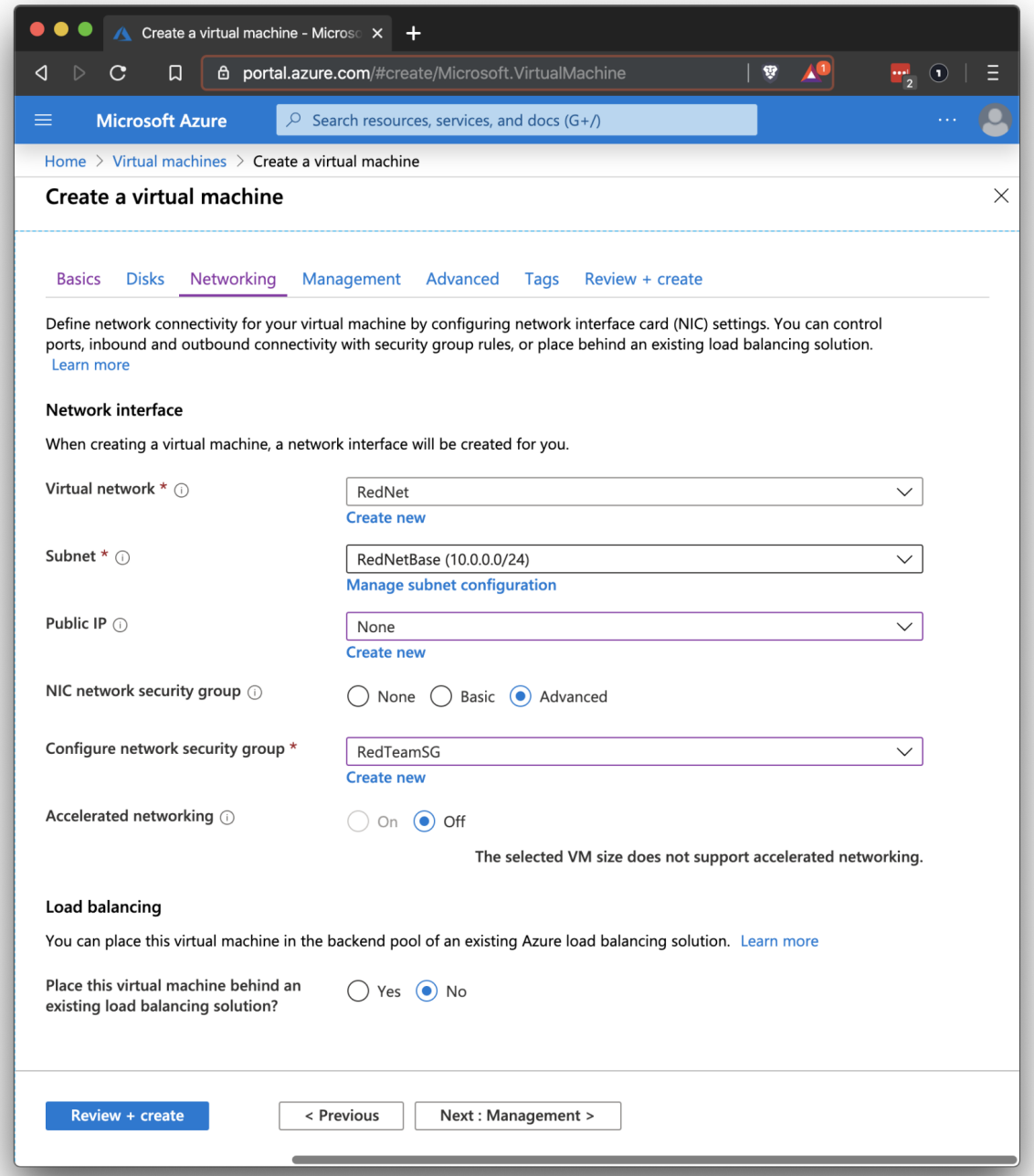
1. Launch a new VM in the Azure portal.
    - Name this VM: `Web-3`
    - Be sure to use the same admin name and SSH key from your Ansible container that you used for the current DVWA machine.
    - You may need to start your Ansible container on your jump box to get the key.
        - Run `sudo docker container list -a` to see a list of all the containers (you should only have one), and note the unique name of your container.
        - Run the following commands to start your container and get the key:

```
$ sudo docker start your_container_name
your_container_name
$ sudo docker attach your_container_name
$ cat .ssh/id_rsa.pub
```

        - `ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDdFS0nrcNG91P3HV60pPCDE0YCKNeS5Kr8edGxCeXUT1SP09Eyxxpi6LPZbL0Nkn8JNtdaxN9qyWG4Xpuh+rzCl9QnnGsdge76muzwl6awVUvRn0IAjM/e3RCKt0e1xSRiGaUY1ch41NY1Dih/MjxPunC2BykSGP17/hgMmLPKe8ZsHVaiFv1SiEqsGHa/`
            - Copy the key into your configuration.



- For your Availability set, set RedTeamAS.
- Do not give your new VM an external IP address.
- Do not assign a load balancer.



2. Once your machine is set up, connect to the Ansible container on your jump box and test the Ansible connection using SSH.

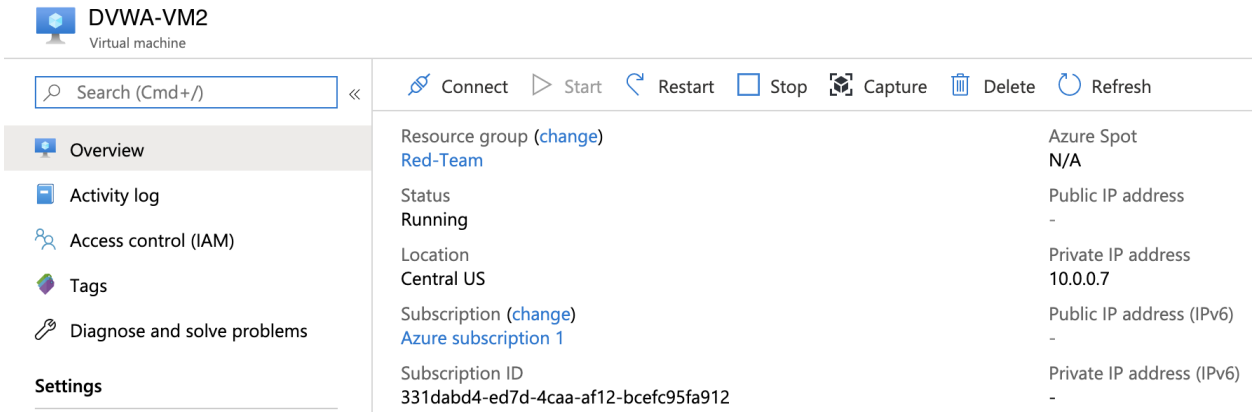
```
$ ssh ansible@10.0.0.7
The authenticity of host '10.0.0.7 (10.0.0.7)' can't be established.
ECDSA key fingerprint is
SHA256:Jes0kNsSifAVf/TEcfPxhP4/p2fmS7WGk2O8xo8vC64.
Are you sure you want to continue connecting (yes/no)? yes
```

```
Warning: Permanently added '10.0.0.7' (ECDSA) to the list of known hosts.

Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 5.0.0-1027-azure x86_64)
```

- Run `exit` to return to your Ansible container.

3. Add the internal IP address of the new VM to your Ansible configuration.
   - Get the internal IP from the VM details page in Azure:



   - On your Ansible container, run `nano /etc/ansible/hosts`.
   - Add the new IP address under the IP of the other VM.

```
# Ex 2: A collection of hosts belonging to the 'webservers' group

[webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
10.0.0.6 ansible_python_interpreter=/usr/bin/python3
10.0.0.7 ansible_python_interpreter=/usr/bin/python3
      10.0.0.8 ansible_python_interpreter=/usr/bin/python3
# If you have multiple hosts following a pattern you can specify
```

4. `# them like this:`
   - Save and exit the hosts file.
5. Test your Ansible configuration with the Ansible `ping` command.
   - Run `ansible -m ping all` (Ignore `[DEPRECATION WARNING]`.)

```
root@1f08425a2967:~# ansible -m  ping all

10.0.0.6 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
```

```
    "changed": false,
    "ping": "pong"
}

10.0.0.7 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
```

6. }
7. Run your Ansible playbook to configure your new machine.
   Hint: If you run your playbook, it will run on both machines. Ansible will recognize your original VM and check its settings. It should only make changes to the new VM.
   - Run `ansible-playbook your-playbook.yml`

```
root@1f08425a2967:~# ansible-playbook /etc/ansible/pentest.yml

PLAY [Config Web VM with Docker]
***************************************************

TASK [Gathering Facts]
*************************************************************
ok: [10.0.0.7]
ok: [10.0.0.6]

TASK [docker.io]
*********************************************************************
ok: [10.0.0.6]
[WARNING]: Updating cache and auto-installing missing dependency: python-apt

changed: [10.0.0.7]

TASK [Install pip]
*************************************************************
ok: [10.0.0.6]
changed: [10.0.0.7]

TASK [Install Docker python module]
***********************************************
ok: [10.0.0.6]
changed: [10.0.0.7]

TASK [download and launch a docker web container]
*********************************
changed: [10.0.0.6]
changed: [10.0.0.7]
```

```
PLAY RECAP
*********************************************************************
10.0.0.6                   : ok=5    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

8. `10.0.0.7                   : ok=5    changed=4    unreachable=0`
   `failed=0    skipped=0    rescued=0    ignored=0`
9. When the Ansible playbook is finished running, SSH to your new VM and test the DVWA app using `curl`.
   - Run `ssh ansible@10.0.0.7`
   - Run `curl localhost/setup.php`
   - Your output should look like the following:

```
root@1f08425a2967:~# ssh ansible@10.0.0.7
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 5.0.0-1027-azure x86_64)

* Documentation:  https://help.ubuntu.com
* Management:     https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of Fri Jan 10 21:01:52 UTC 2020

System load:  0.24              Processes:              122
Usage of /:   9.9% of 28.90GB   Users logged in:        0
Memory usage: 57%               IP address for eth0:    10.0.0.7
Swap usage:   0%                IP address for docker0: 172.17.0.1


19 packages can be updated.
16 updates are security updates.


Last login: Fri Jan 10 20:57:26 2020 from 10.0.0.4
ansible@Pentest-2:~$ curl localhost/setup.php

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

        <title>Setup :: Damn Vulnerable Web Application (DVWA) v1.10
*Development*</title>

        <link rel="stylesheet" type="text/css" href="dvwa/css/main.css" />
```

```
    <link rel="icon" type="\image/ico" href="favicon.ico" />

    <script type="text/javascript" src="dvwa/js/dvwaPage.js"></script>

</head>
#Truncated
```

END

Congratulations! You have created a highly available web server for XCorp's Red Team to use for testing and training.