**Introduction**

This documentation is a detailed instruction on how to install and configure an Azure virtual network with five virtual machines (Jump Box and four Web (DVWA/ELK) servers using an Ansible container.

The Ansible playbooks and files can be found in the Ansible folder of the following GitHub repository

https://github.com/ttambio/elk-stack-project.git

See the following section for more information:

**Miscellaneous - GitHub Repository**

This documentation is broken down into the following sections:

* Azure Virtual Network Configuration
* ELK Stack Configuration
* Miscellaneous

**Azure Virtual Network Configuration**

* Azure Virtual Network Diagram
* Set-up Azure Account
* Create Resource Group
* Create Azure Virtual Network
* Create Security Group
  + Create Network Security Group (NSG)
  + Create Inbound Security Rule (Block)
* Create Jump Box Virtual Machine
  + Create SSH key
  + Create Virtual Machine
* Create Three Web Virtual Machines
* Create Inbound Security Rule (Client Port 22)
  + Find Client Public IP
  + Create Inbound Rule
  + Connect to Jump Box
* Create Containers
  + Install Docker
  + Create Inbound Security Rule (Jump Box Port 22)
* Create Container SSH Key
  + Start/Connect Container
  + Create SSH Key
  + Reset Web Virtual Machine Password
  + Connect to Web Containers

**Introduction (con’t)**

This documentation is broken down into the following sections: (con’t)

**Azure Virtual Network Configuration (con’t)**

* Update Configuration Files
  + Hosts File
  + Configuration File
  + Test Group Connectivity
* Create Ansible Playbook
  + Create Playbook
  + Execute Playbook
  + Test DVWA Container
* Configure Load Balancer
  + Create Load Balancer
  + Install Health Probe
  + Update Load Balancer Settings
* Set-up Security Configuration
  + Create Load Balancing Rule (Port 80)
  + Create Inbound Security Rule (Client Port80)
  + Remove Inbound Security Rule (Block)
  + Connect to DVWA Website

**ELK Stack Configuration**

* Create Second Virtual Network
  + Create Virtual Network
  + Create Peer Network Connection
* Create Fourth Web (ELK) Virtual Machine
  + Copy SSH Key
  + Create Virtual Machine
  + Connect to Virtual Machine
* Update Hosts File
* Create ELK Stack Playbook
  + Create Playbook
  + Execute Playbook
  + Launch Container
* Create Inbound Security Rule (Client Port 5601)
  + Create Inbound Security Rule
  + Connect to Kibana
* Filebeat
  + Create Filebeat Configuration File
  + Create Filebeat Playbook
  + Execute Filebeat Playbook

**Introduction (con’t)**

This documentation is broken down into the following sections: (con’t)

**ELK Stack Configuration (con’t)**

* Metricbeat
  + Create Metricbeat Configuration File
  + Create Metricbeat Playbook
* Kibana – Filebeat
* Kibana - Metricbeat

**Miscellaneous**

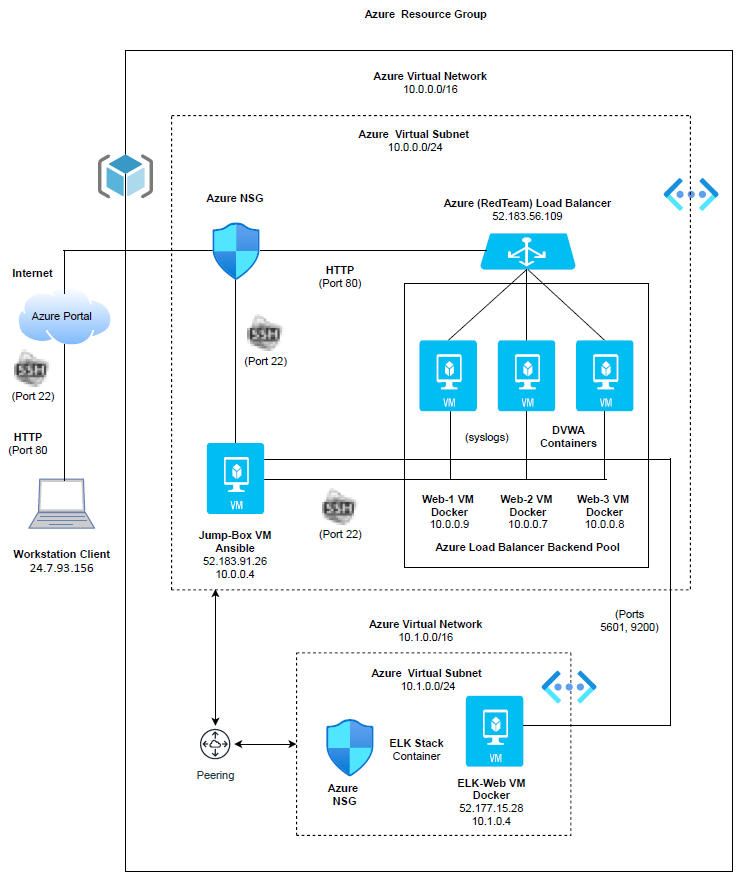
* GitHub Repository
* Locate IP Addresses
* SSH keys
* Machine Login
* Azure Portal - Screenshots
* Explore Kibana
* Pen Testing

**Azure Virtual Network Configuration**

**Azure Virtual Network Diagram**

This diagram is located in:

Diagrams/Azure\_Virtual\_Network - ELK\_Stack\_Diagram



**Set-up Azure Account**

An Azure account is needed in order to create this Azure Virtual Network.

For a free 30 day account, sign up at:

<https://azure.microsoft.com/en-us/free/>

After an account is created, log into:

<https://portal.azure.com>

**IMPORTANT:** Before logging out of your Azure account, always shut down all virtual machines!!!

**Create Resource Group**

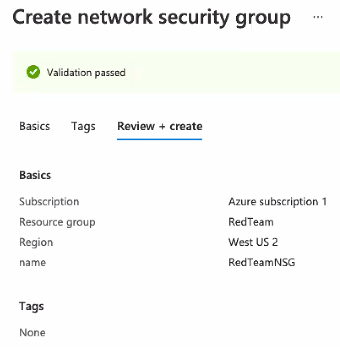
Create a new Resource Group.

From the Azure Portal homepage, search for ‘Resource groups’.

From the ‘Resource groups’ page, click ‘New’.

* Enter a resource group name.
  + Make a note of this name.
* Choose a region.
  + Make a note of this region you selected.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

Example of results:



**Create Azure Virtual Network**

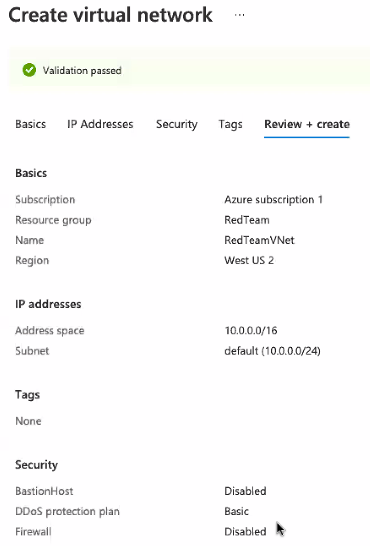
Create a new Virtual Network.

From the Azure Portal homepage, search for ‘Virtual networks’.

From the ‘Virtual networks’ page, click ‘New’.

* Select the resource group you created in ‘Create Resource Group’.
* Enter a virtual network name.
  + Make a note of this name.
* Choose the same region you selected when creating your network security group in ‘Create Resource Group’.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

Example of results:



**Create Security Group**

**Create Network Security Group (NSG):**

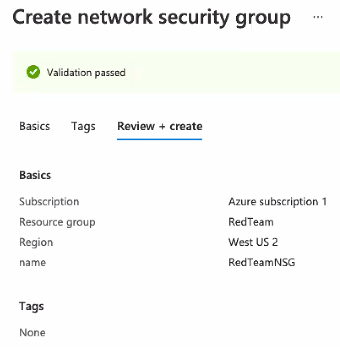
Create a new network security group (NSG).

From the Azure Portal homepage, search for ‘Network security groups’.

From the ‘Network security groups’ page, click ‘Add’.

* Select the resource group you created in ‘Create Resource Group’.
* Enter a network security group name.
* Choose the same region you selected when creating your network security group in ‘Create Resource Group’.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

Example of results:



**Create Security Group (con’t)**

**Create Inbound Security Rule (Block):**

Create a new inbound security rule to block all traffic coming into your subnet.

From the ‘Network security groups’ page, select the network security group you just recently created.

From the left pane, select ‘Inbound security rules’, then click ‘Add’.

Set the rule as:

Source: Select ‘Any’.

Source port ranges: \*

Destination: Select ‘Any’

Destination port ranges: \*

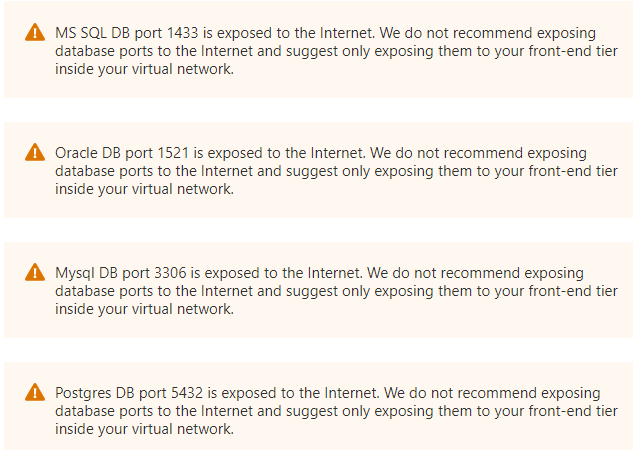
Protocol: Any

Action: Allow

Priority: This rule should have the highest number.

Name: Enter a name such as ‘DenyAll’.

NOTE: Ignore the following warning messages.



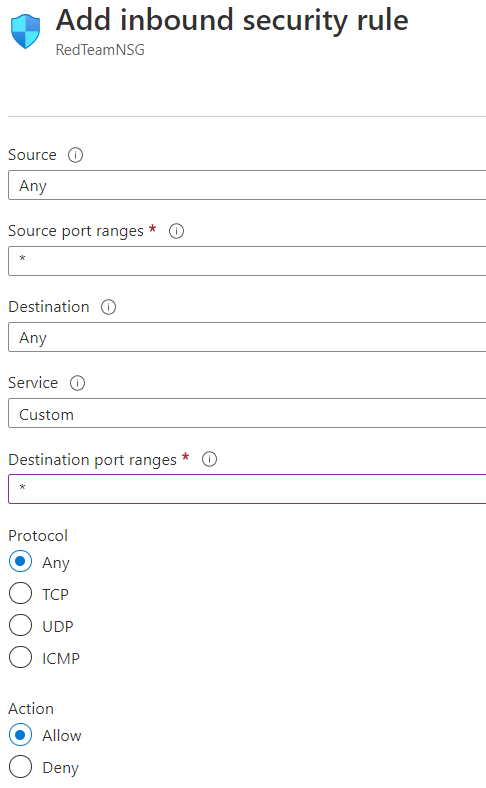
* Click ‘Add’.

See below for example.

**Create Security Group (con’t)**

**Create Inbound Security Rule (Block): (con’t)**

Example:





**Create Jump Box Virtual Machine**

**Create SSH key:**

Generate a new SSH public key to be able to access your cloud servers.

From Git Bash, enter the following command:

ssh-keygen

* + When asked ‘Enter file in which to save the key’, enter something like:

/c/Users/tricia/.ssh/Azure21\_rsa.pub

* + Do not enter a ‘passphrase’, just hit the ‘Enter’ key.

Copy your public SSH key:

$ cat /c/Users/tricia/.ssh/Azure21\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAEBgQD0eN7+tWU9wnNc1+6JMDMxlvz+H5A32vwHzC6A7UrJ4dTrM2jAN73dR6m61jYqDY+1D8OgKbGcvifHSJAnfLmkTipREjgCpvbe4FXCop35l1BIVSv73VXpd1DGipUBZ4ZJuI4RO+ThJOJJI8uhGeQGcdzhFbdU20T0l24ywylAtCjaS/pQJtbO/iD6OXbG4jKMUOnE/tojvAkD5n6AL4v0N+ZPUwt/KUons+ehO04O1W/rLUKSU8yCqukkMhMCq+wxE6ft5ltUmwrxCKY/667a27gB98dM0oREsdnzmZXpUkbEjSbnoqVgxCPfODnfHmbcItcyq5pdL4GO5hABOEWpnbWowCQw1B+BeCPBRhSBEGG/FaYHoCUWC/fkBXXZ7rIPxPdJNOoTM09wTqZklibneqII9376cs209xjiF7VmiNE7E= tricia@LAPTOP-GSGFQPM7

**Create Jump Box Virtual Machine (con’t)**

**Create Virtual Machine:**

Create a virtual machine for the Jump Box.

From the Azure Portal homepage, search for ‘Virtual machines’.

From the ‘Virtual machines’ page, click ‘Add’ and select ‘Virtual machine’.

**NOTE:** The public IP address for this Jump Box machine needs to be static!

Basic section:

* For ‘Resource group, select the resource group that you created in ‘Create Resource Group’.
* Enter a virtual machine name.
* Choose the same region you selected when creating your network security group in ‘Create Resource Group’.
* Image is ‘Ubuntu Server 18.04 LTS – Gen 1’.
* For ‘Size’, select ‘Standard\_B1s - 1 vcpu, 1 Gib’.
* ‘Authentication type’ should be ‘SSH public key’.
* Enter a username.
  + Make a note of this username.
* For the ‘SSH public key source’ field, select ‘Use existing public key’
* In the ‘SSH public key’ field, paste your SSH key that you recently generated.
* Leave other options to the default settings and continue to the ‘Network’ tab (scroll up to see tabs).

Networking section:

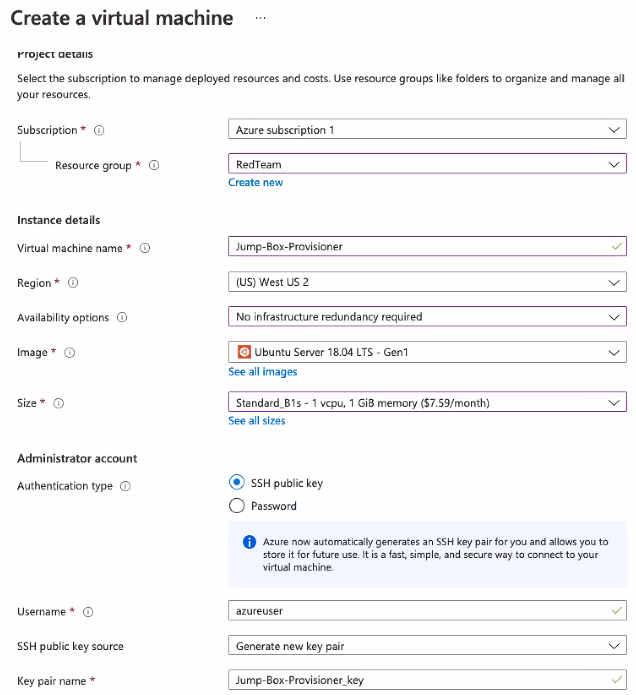
* Select the virtual network that you created in ‘Create Azure Virtual Network’.
* Leave the ‘Subnet’ field to the default settings.
* For the ‘Public IP’, click on ‘Create new.
  + From the pop-up window, leave the ‘Name’ field to the default name.
    - Click ‘OK’.
  + The ‘Assignment’ field should be ‘Static’.
* For the ‘NIC network security group’, select ‘Advanced’.
* Select the network security group that you created in ‘Create Network Security Group’.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

See below for examples.

**Create Jump Box Virtual Machine (con’t)**

**Create Virtual Machine: (con’t)**

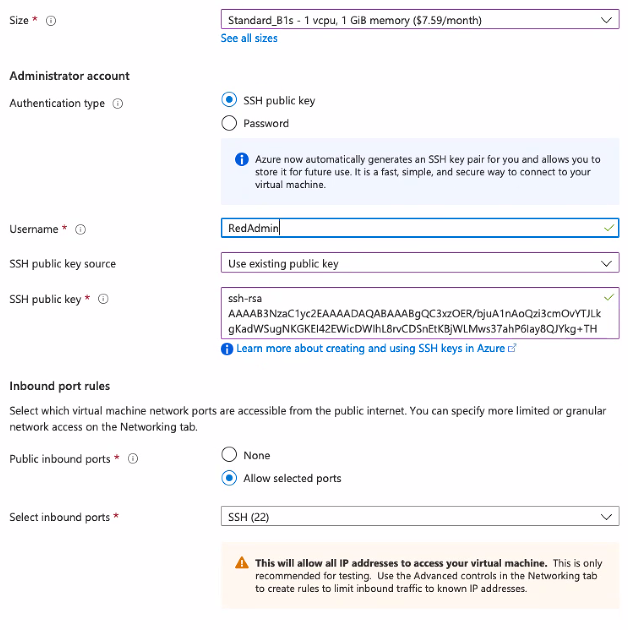
Example for the ‘Basic’ section:



**Create Jump Box Virtual Machine (con’t)**

**Create Virtual Machine: (con’t)**

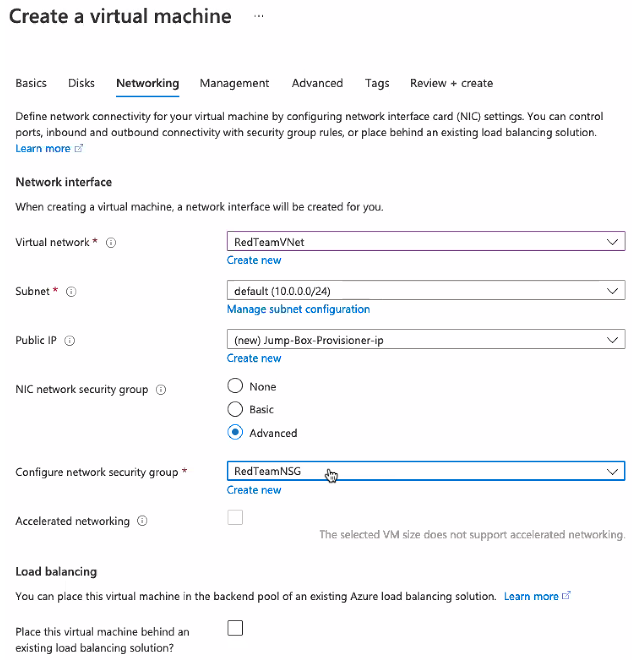
Example for the ‘Basic’ section: (con’t)



**Create Jump Box Virtual Machine (con’t)**

**Create Virtual Machine: (con’t)**

Example for the ‘Networking’ section:



**Create Three Web Virtual Machines**

Create a virtual machine called Web-1, Web-2 and Web-3.

From the Azure Portal homepage, search for ‘Virtual machines’.

From the ‘Virtual machines’ page, click ‘Add’ and select ‘Virtual machine’.

Basic section:

* For ‘Resource group, select the resource group that you created in ‘Create Resource Group’.
* Enter a virtual machine name.
* Choose the same region you selected when creating your network security group in ‘Create Resource Group’.
* For the ‘Availability options’ field, select ‘Availability set’.
  + Click ‘Create new’.
  + A new window appears on the right side, enter a name.
  + NOTE: You will be using this same set for all three Web virtual machines.
* Image is ‘Ubuntu Server 18.04 LTS – Gen 1’.
* Select size that is Standard\_B1ms - 1 vpus, 2 Gib.
  + NOTE: This is a different size than the Jump Box!
* Authentication type should be ‘SSH public key’.
* Enter a username.
  + Make a note of this username.
  + Use this same name for all three Web virtual machines!
* For the ‘SSH public key source’ field, select ‘Use existing public key’
* In the ‘SSH public key’ field, paste your SSH key that you recently generated.
* Leave other options to the default settings and click ‘Next: Disks’.

Disks section:

* For the ‘OS disk type’, select ‘Premium SSD’.
* Leave other options to the default settings and click ‘Next: Networking’.

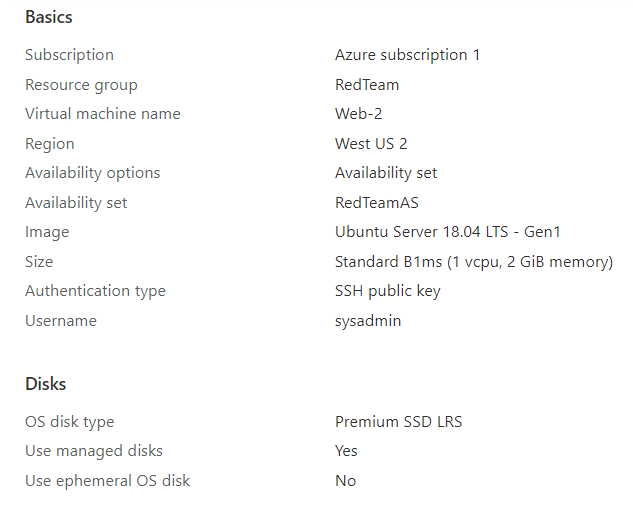
Networking section:

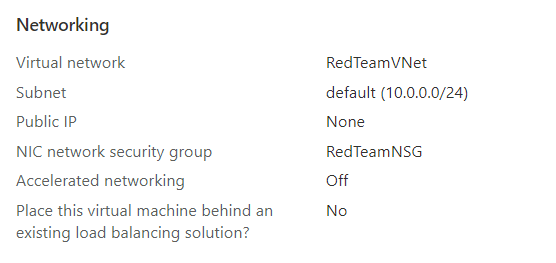
* Select the virtual network that you created in ‘Create Azure Virtual Network’.
* Leave the ‘Subnet’ field to the default settings.
* For the ‘Public IP’, select ‘None’.
* For the ‘NIC network security group’, select ‘Advanced’.
* Select the network security group that you created in ‘Create Network Security Group’.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

**Create Three Web Virtual Machines (con’t)**

Create a second Web virtual machine using the same steps you did for the first Web virtual machine (Web-1) but changing the virtual name to like ‘Web-2’.

Example of results for ‘Web-2’:





**Create Inbound Security Rule (Client Port 22)**

**Find Client Public IP:**

Locate the public IP address of your workstation (client).

This can be done by going to: https://whatismyipaddress.com/

* + Make a note of this address.

**Create Inbound Rule:**

Create a new inbound security rule to allow a SSH connection from the client’s public IP address.

From the Azure Portal homepage, search for ‘Network security groups’.

From the ‘Network security groups’ page, select the network security group you just recently created in ‘Create Network Security Group (NSG)’.

From the left pane, select ‘Inbound security rules’, then click ‘Add’.

Set the rule as:

Source: Select ‘IP Addresses’.

Source IP addresses: Client’s public IP address

Source port ranges: \*

Destination: Select ‘VirtualNetwork’

Destination port ranges: Enter ‘22’.

Protocol: Any

Action: Allow

Priority: Enter a number that is lower than the number for the

rule ‘DenyAll’ you recently created to deny all traffic.

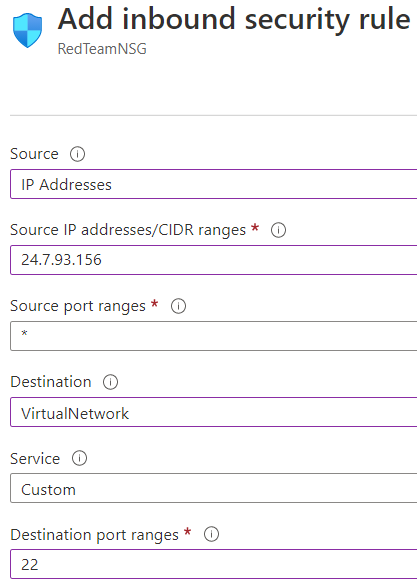
Name: Enter a name such as ‘SSH’ or ‘AllowSSH’.

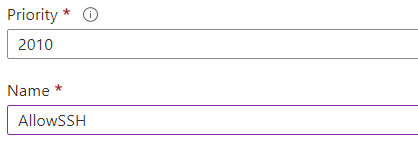
* Click ‘Add’.

Example of creating an inbound rule shown below.

**Create Inbound Security Rule (Client Port 22)**

**Create Inbound Rule: (con’t)**





**Create Inbound Security Rule (Client Port 22) (con’t)**

**Connect to Jump Box:**

From Git Bash, connect to the Jump Box using the following command:

ssh -i *public\_key jumpbox\_username@JumpBox\_publicIP*

* + To find the public IP address of the Jump Box:
    - From the ‘Virtual machine’ page on the Azure Portal, select the Jump Box.
    - From the ‘Overview’ section, the public IP address is on the right side.

Example:

ssh -i /c/Users/tricia/.ssh/Azure21\_rsa RedAdmin@52.183.91.26

The prompt should look like:



**Create Containers**

**Install Docker:**

NOTE: You do not need to setup a Docker account if you know the name of the image you want to download/pull.

Install Docker from the Jump Box.

From Git Bash, log into the Jump Box:

ssh -i *public\_key jumpbox\_username@JumpBox\_publicIP*

NOTE: Example shown in ‘Create Inbound Security Rule

(Client Port 22) - Connect to Jump Box’.

From the Jump Box, enter the following commands to install Docker:

sudo apt update

sudo apt install docker.io

Execute the following commands to pull and launch the container.

sudo docker pull cyberxsecurity/ansible

sudo docker run -ti cyberxsecurity/ansible

**Create Containers (con’t)**

**Create Inbound Security Rule (Jump Box Port 22):**

Create a new inbound security rule to allow a SSH connection from the Jump Box’s internal (private) IP address to the virtual network.

From the Azure Portal homepage, search for ‘Network security groups’.

From the ‘Network security groups’ page, select the network security group you just recently created in ‘Create Network Security Group (NSG)’.

From the left pane, select ‘Inbound security rules’, then click ‘Add’.

Set the rule as:

Source: Select ‘IP Addresses’.

Source IP addresses: Jump Box’s internal IP address

* + To locate the internal IP address of the Jump Box, enter the command ‘ifconfig’ from the Jump Box.



Source port ranges: \*

Destination: Select ‘VirtualNetwork’

Destination port ranges: Enter ‘22’.

Protocol: Any

Action: Allow

Priority: This number can be lower than the number for the

rule ‘SSH’ you recently created.

Name: Enter a name such as ‘JumpBoxAccess’.

* Click ‘Add’.

**Create Container SSH Key**

**Start/Connect Container:**

Start and connect to a container.

From Git Bash, log into the Jump Box:

To check if Ansible is running, enter the following command:

sudo docker run -ti cyberxsecurity/ansible bash

* + To exit, enter ‘exit’.

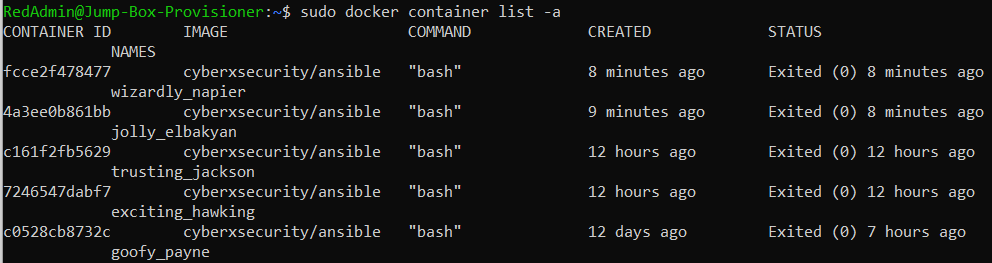
NOTE: This will create a new container!

**IMPORTANT:**

We only want one Ansible container executing.

To list all containers on the Jump Box, enter the following command:

sudo docker container list -a



If we have multiple containers running, we need to stop and delete all except one.

Select one container to use and delete the rest! The container names are the ones with the funny names.

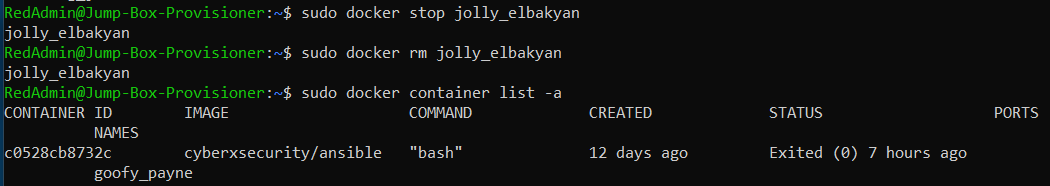
**Create Container SSH Key (con’t)**

**Start/Connect Container: (con’t)**

The command to stop and delete a container:

sudo docker stop jolly\_elbakyan

sudo docker rm jolly\_elbakyan

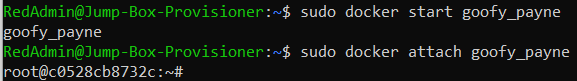


To start and connect to a container, enter the following command:

sudo docker start goofy\_payne

sudo docker attach goofy\_payne

* + The ‘start’ command basically means to engage.
  + The ‘attach’ command basically means to connect.



NOTE: If you can’t connect (‘attach’), try to restart the

container.

sudo docker restart goofy\_payne

**IMPORTANT:**

Whenever you need to start and connect to that one specific container, all you have to do is check to see if any other containers are running.

If yes, delete them. If no, just start and connect to that one specific container.

sudo docker container list -a

sudo docker start goofy\_payne

sudo docker attach goofy\_payne

**Create Container SSH Key (con’t)**

**Create SSH Key:**

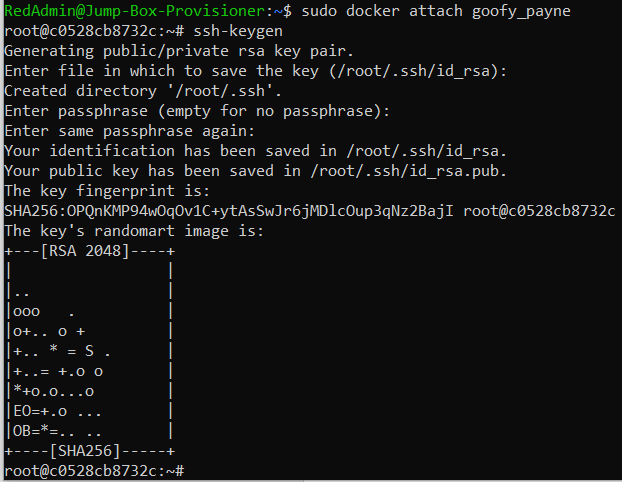
Generate a new SSH public key to be able to access the Web virtual machines from within a container that is executing inside of the Jump Box.

From Git Bash, log into the Jump Box, then start/attach to your container:

Enter the following command:

ssh-keygen

* + When asked ‘Enter file in which to save the key’, hit the ‘Enter’ key (this is the location we want).
  + Do not enter a ‘passphrase’, just hit the ‘Enter’ key.



**Create Container SSH Key (con’t)**

**Create SSH Key: (con’t)**

Copy the newly generated SSH key from ‘/root/.ssh’ (noted above):

root@c0528cb8732c:~# cat /root/.ssh/id\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDPBrR4CMmhByWz/3cAdxcUHUdaGMwa0Vn30xJRIuk00EOFSNK3uj+fFSy70yvOiW3FXQ73VB2Yz879sSBtyaJUUG2FRxy3v3/YLLl6ada7Y/ZTyA7xNxGFnBdhC+mhIXCNqS/JejuiMFDHZSfp5UhlPPC1MBXtMX/psUTc34Z4tQry8ap7RiGRl7hMzQOw4KR7pPTWKqkoLBD3GB8k0q5SWZQN2Lowix780vGdp0GJamBx8b9OiD9yc0nTjQPihe2ZsdJImayLtMO1dbAmmBG3Kxs2WwK3Q0gneiwIax2po9Dd6uLz/pu/mQ+s35N5c4/PluVr2NjIdM5 root@c0528cb8732c

**Reset Web Virtual Machine Password:**

Reset the password for all three Web virtual machines with the container’s public SSH key.

From the Azure Portal homepage, search for ‘Virtual machines’.

To reset the password for Web-1:

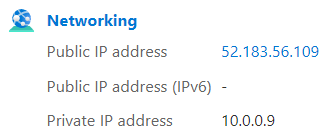
* From the ‘Virtual machines’ page, select Web-1.
* From the left pane, scroll down until you see ‘Reset password’.
  + Click on it.
* Replace the password with the container’s SSH key.
* Save changes.

Repeat these steps for Web-2 and Web-3.

In order to test the connection to these machines, we will need to use the internal (private) IP address.

To locate the internal IP address for Web-1:

* From the ‘Virtual machine’ page, select Web-1.
* From the ‘Overview’ section, the internal IP address is on the right side.



Repeat steps for Web-2 and Web-3.

**Create Container SSH Key (con’t)**

**Connect to Web Virtual Machines:**

Verify the connection to the ELK virtual machines within your container.

From Git Bash, log into the Jump Box, then start/attach to your container:

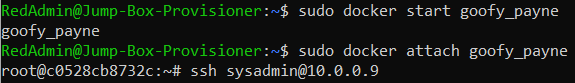
Enter the following command to connect to Web-1 using its internal IP address:

ssh *username*@*internal\_IP*

* + The username is the name created when the virtual machine was initially created.

Example:

ssh sysadmin@10.0.9



The prompt should look like:



Exit from Web-1 and verify the connections to Web-2 and Web-3 using its internal IP address.

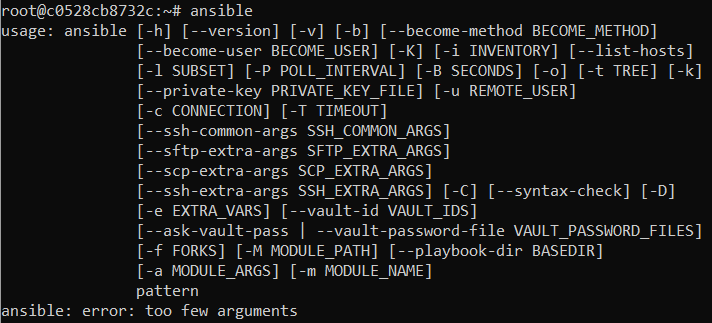
NOTE: Once you have exited from the Web virtual machines, you will be back to the container within the Jump Box.

**Update Configuration Files**

The Ansible configuration files will need to be modified to use the settings for the Web virtual machines.

From Git Bash, log into the Jump Box, then start/attach to your container.

First test if Ansible is installed and running by entering the ‘ansible’ command.



* + Ignore error message of ‘too few arguments’, we’re just testing if Ansible is running.

From the container prompt, change directory to ‘/etc/ansible’ and display its contents.

root@c0528cb8732c:~# cd /etc/ansible

root@c0528cb8732c:/etc/ansible# ls -l

total 28

-rw-r--r-- 1 root root 19985 Dec 4 2019 ansible.cfg

-rw-r--r-- 1 root root 1016 Dec 4 2019 hosts

drwxr-xr-x 2 root root 4096 Dec 4 2019 roles

The ‘hosts’ and ‘ansible.cfg’ file will need to be modified accordingly.

**Update Configuration Files (con’t)**

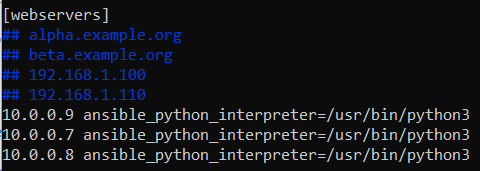
**Hosts File:**

Update the ‘hosts’ file.

Uncomment out the ‘[webservers]’.

Add the internal IP addresses for Web-1, Web-2 and Web-3, include the python version we want to execute our scripts.

Example:



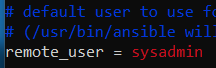
NOTE: The python version line will not be needed in the near future once python version 2 becomes obsolete.

**Configuration File:**

Update the ‘ansible.cfg’ file.

The ‘remote\_user’ field will need to be uncommented out and the user will need to be changed to the username of the Web virtual machines.

Example:



**Update Configuration Files (con’t)**

**Test Group Connectivity:**

To test our group connectivity, execute the following command:

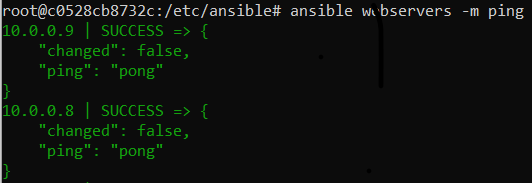
ansible webservers -m ping

* + ‘webserver’ is the group that the Web virtual machines is associated with.

A similar command to use to show all groups:

ansible all -m ping

Example:



NOTE: This is just an example. The IP address should reflect the internal IP address of your Web virtual machines.

* + The results of this command should say ‘SUCCESS’ as shown above.

**Create Ansible Playbook**

**Create Playbook:**

Create a YAML playbook for the configuration of the Web virtual machines with Docker.

From Git Bash, log into the Jump Box, then start/attach to your container.

Change directory to ‘/etc/ansible’ and create a playbook.

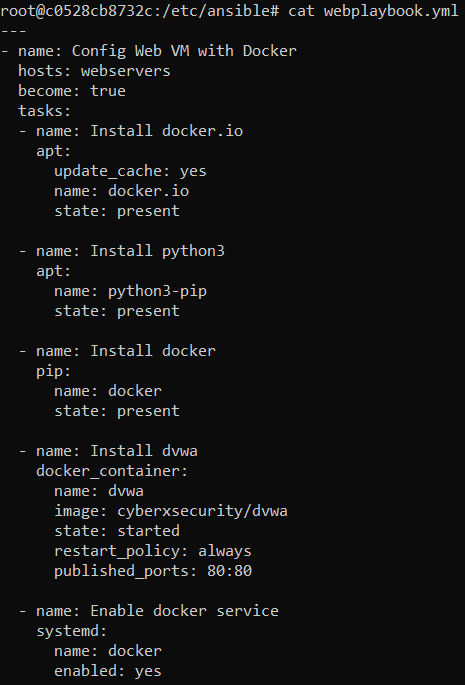
**Create Ansible Playbook (con’t)**

**Create Playbook:**

The playbook configuration should contain the following:

* Install ‘docker.io’ using ‘apt’.
* Install ‘python3’ using ‘apt’.
* Install ‘docker’ using ‘pip’.
* Install the ‘dvwa’ container using ‘docker-container’.
* Enable the docker service using ‘systemd’.

Example:



**Create Ansible Playbook (con’t)**

**Execute Playbook:**

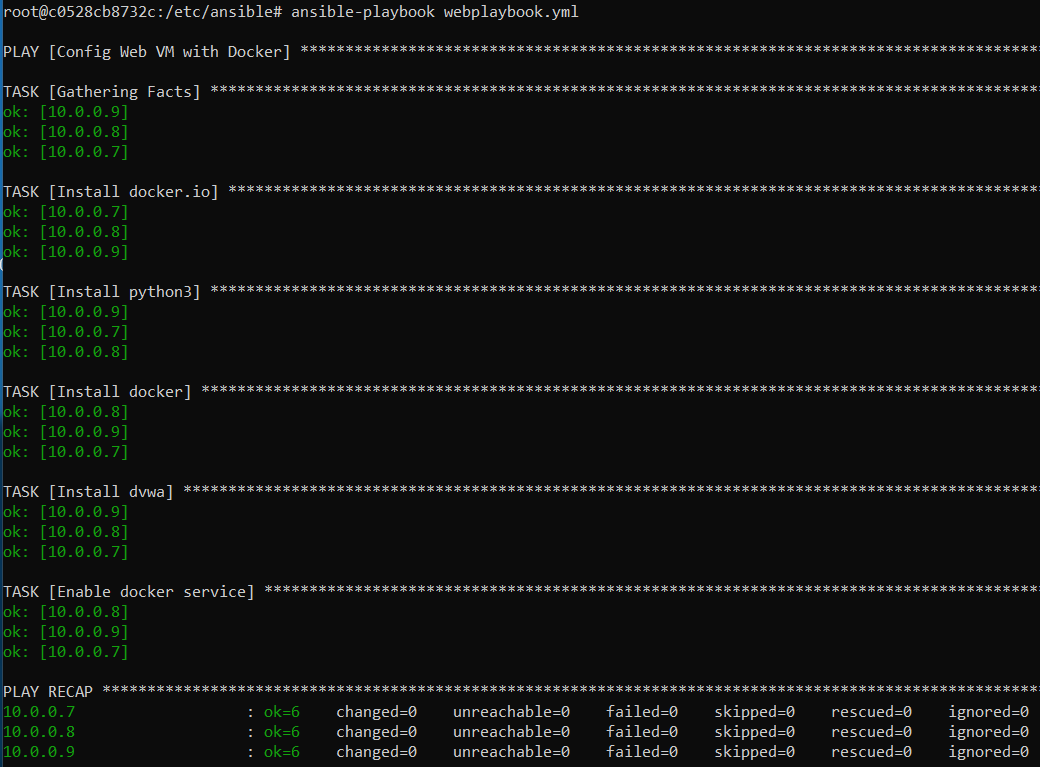
To execute a playbook, enter the following command from within your container:

ansible-playbook *playbook\_name*

Example of command:

ansible-playbook webplaybook.yml

Example of results:



**Create Ansible Playbook (con’t)**

**Test DVWA Container:**

Test that the DVWA container is running within your Web virtual machines.

From within your container, log into one of your Web virtual machines and enter the following command:

curl localhost/setup.php

Example:



NOTE: The follow command will also work without logging into the

Web virtual machines.

curl 10.0.0.9/setup.php

The results should display HTML code.



**Configure Load Balancer**

**Create Load Balancer:**

Configure a load balancer to distribute network traffic between the Web virtual machines.

**NOTE:** This load balancer needs to be configured to have a static public IP address.

From the Azure Portal homepage, search for ‘Load balancers’.

From the ‘Load balancers’ page, click ‘New’.

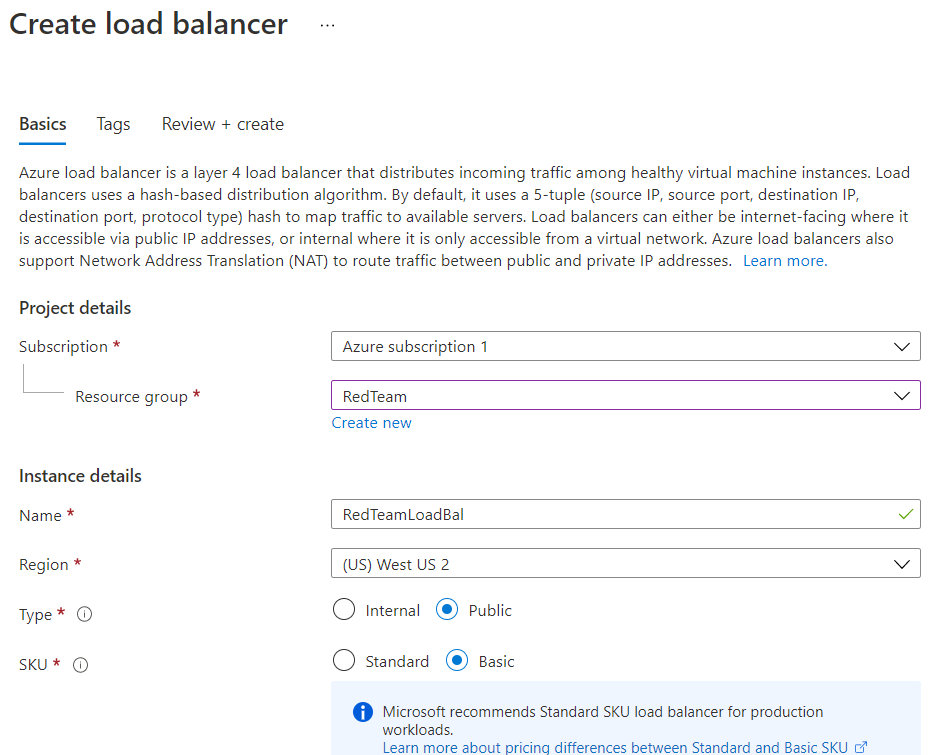
* Select the resource group you created in ‘Create Resource Group’.
* Enter a name.
* Choose the same region you selected when creating your network security group and virtual machines.
* ‘Type’ should be ‘Public’.
* ‘SKU’ should be ‘Basic’.
* For the ‘Public IP address’ field, select ‘Create new’.
* Enter a name for the ‘Public IP address name’.
* For the ‘IP address assignment’ field, select ‘Static’.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

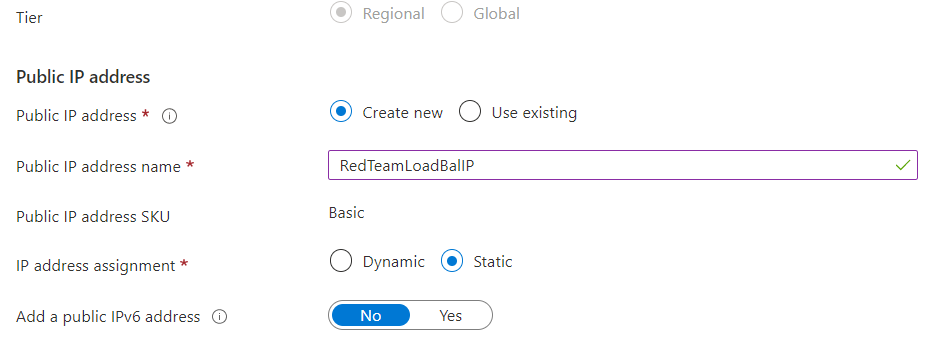
See below for example.

**Configure Load Balancer (con’t)**

**Create Load Balancer: (con’t)**

Example:

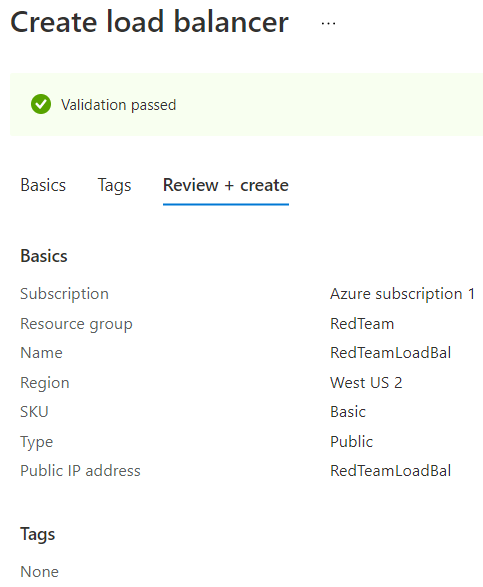




**Configure Load Balancer (con’t)**

**Create Load Balancer: (con’t)**

Example:



**Configure Load Balancer (con’t)**

**Create Load Balancer: (con’t)**

**Install Health Probe:**

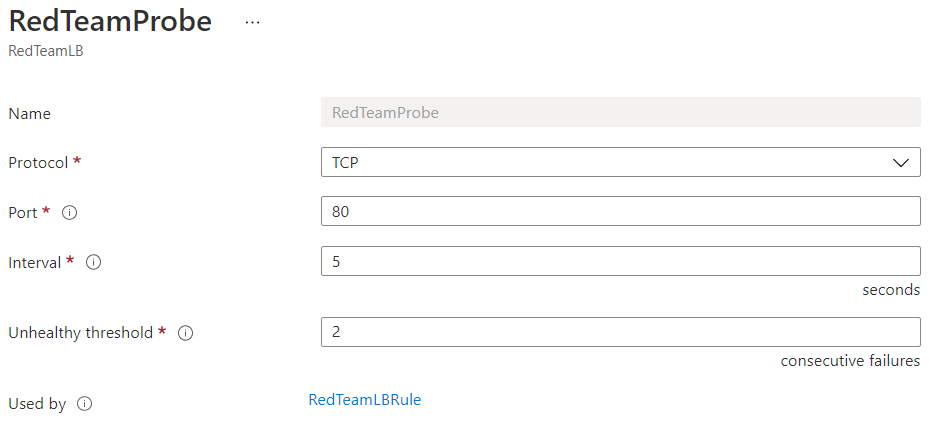
Install a health probe to check the status of your Web virtual machines which will determine which virtual machine is able to receive network traffic.

From the Azure Portal homepage, search for ‘Load balancers’.

From the ‘Load balancers’ page:

* Select the load balancer you just created.
* From the left pane, select ‘Health probes’, then click ‘Add’.
* Enter a name.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

Example:



**Configure Load Balancer (con’t)**

**Create Load Balancer: (con’t)**

**Update Load Balancer Settings:**

Add your Web virtual machines to the backend pool.

From the Azure Portal homepage, search for ‘Load balancers’.

From the ‘Load balancers’ page:

* Select the load balancer you just created.
* From the left pane, select ‘Backend pools’, then click ‘Add’.
* Enter a name.
* From the ‘Associated to’, select ‘Virtual machines’.
* ‘IP Version’ is ‘IPv4’.

In the ‘Virtual machines’ section:

* + Click ‘Add’.
  + A window appears, select your Web virtual machines.
  + Click ‘Add’.
* Then click ‘Add’ again.

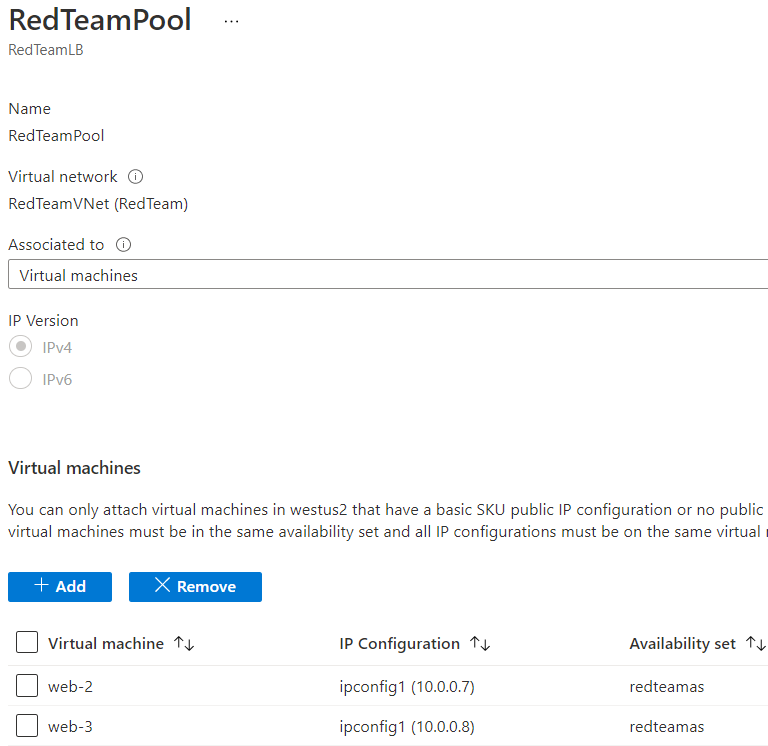
See below for an example.

**Configure Load Balancer (con’t)**

**Create Load Balancer: (con’t)**

**Update Load Balancer Settings: (con’t)**

Example:



**Set-up Security Configuration**

**Create Load Balancing Rule (Port 80):**

Create a load balancing rule to forward traffic on Port 80 from the load balancer to our virtual network.

From the Azure Portal homepage, search for ‘Load balancers’.

From the ‘Load balancers’ page:

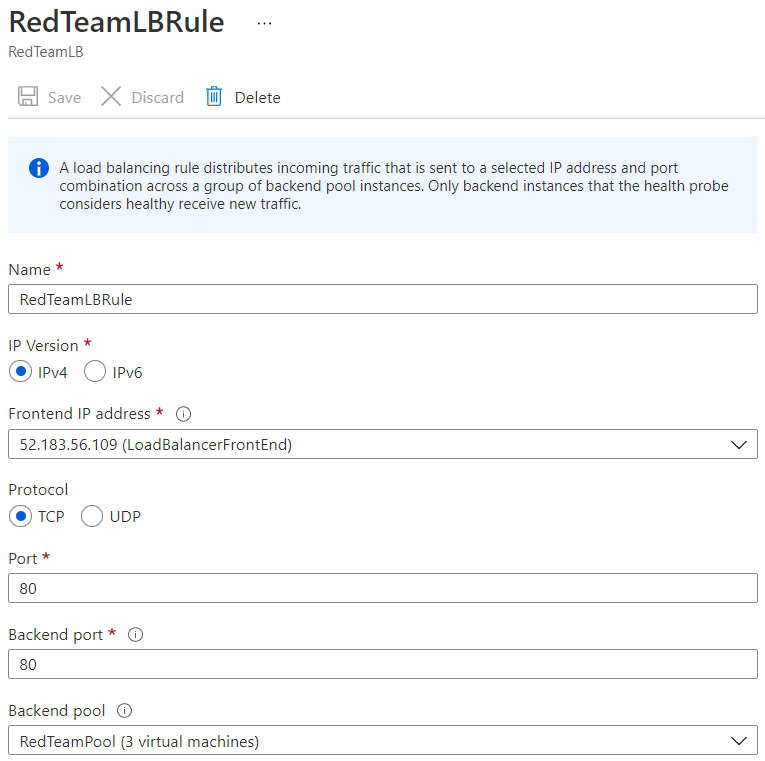
* Select the load balancer you just created.
* From the left pane, select ‘Load balancing rules’, then click ‘Add’.
* Enter a name.
* ‘IP Version’ is ‘IPv4’.
* Leave default setting for ‘Frontend IP address’.
  + It’s automatically assigned a static IP address.
* ‘Protocol’ is ‘TCP’.
* For ‘Port’ enter 80.
* For ‘Backend port’ enter 80.
* Leave other options to the default settings.
* Click ‘OK’.

See below for an example.

**Set-up Security Configuration (con’t)**

**Create Load Balancing Rule (Port 80): (con’t)**

Example:





**Set-up Security Configuration (con’t)**

**Create Inbound Security Rule (Client Port80):**

**Find Client Public IP:**

Locate the public IP address of your workstation (client).

This can be done by going to: https://whatismyipaddress.com/

* + Make a note of this address.

**Create Inbound Rule:**

Create a new inbound security rule to allow a SSH connection from the client’s public IP address.

From the Azure Portal homepage, search for ‘Network security groups’.

From the ‘Network security groups’ page, select the network security group you just recently created in ‘Create Network Security Group (NSG)’.

From the left pane, select ‘Inbound security rules’, then click ‘Add’.

Set the rule as:

Source: Select ‘IP Addresses’.

Source IP addresses: Client’s public IP address

Source port ranges: \*

Destination: Select ‘VirtualNetwork’

Destination port ranges: Enter ‘80’.

Protocol: Any

Action: Allow

Priority: Enter a number that is lower than the number for the

rule ‘DenyAll’.

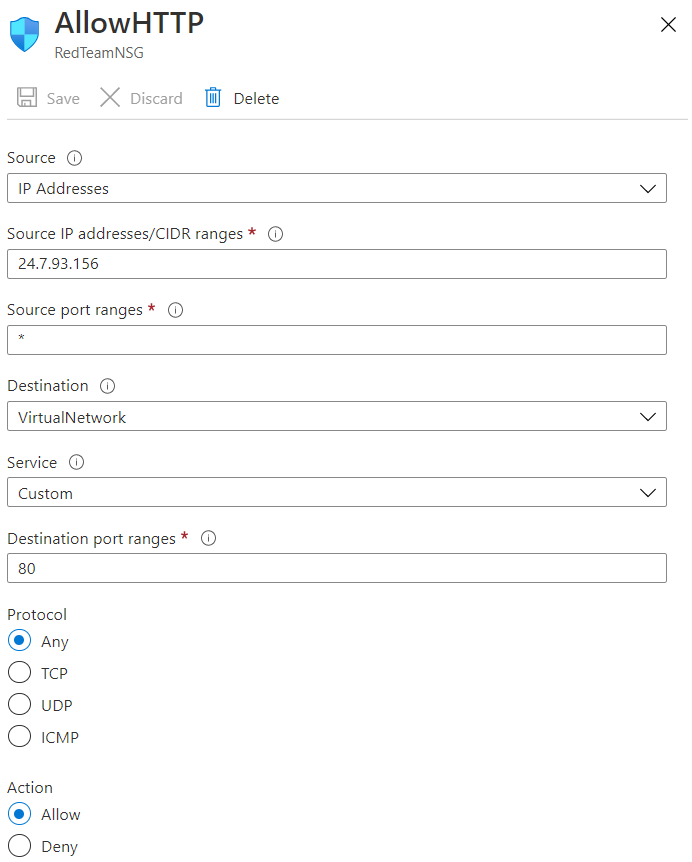
Name: Enter a name such as ‘AllowHTTP’.

* Click ‘Add’.

**Set-up Security Configuration (con’t)**

**Create Inbound Security Rule (Client Port80):**

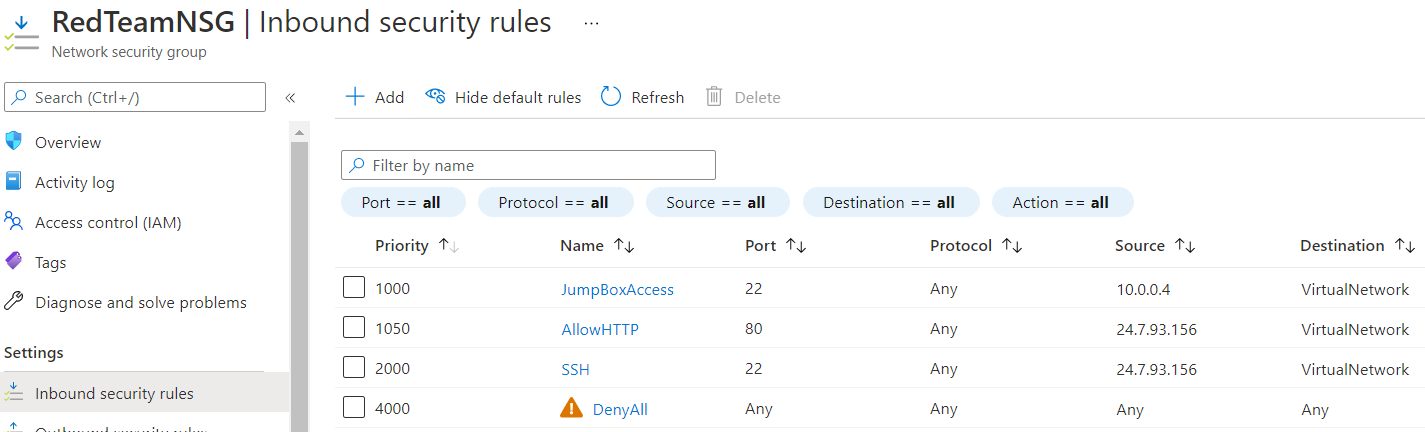
Example of creating an inbound rule for Port 80:



**Set-up Security Configuration (con’t)**

**Remove Inbound Security Rule (Block):**

At this point, you may delete the ‘DenyAll’ rule.



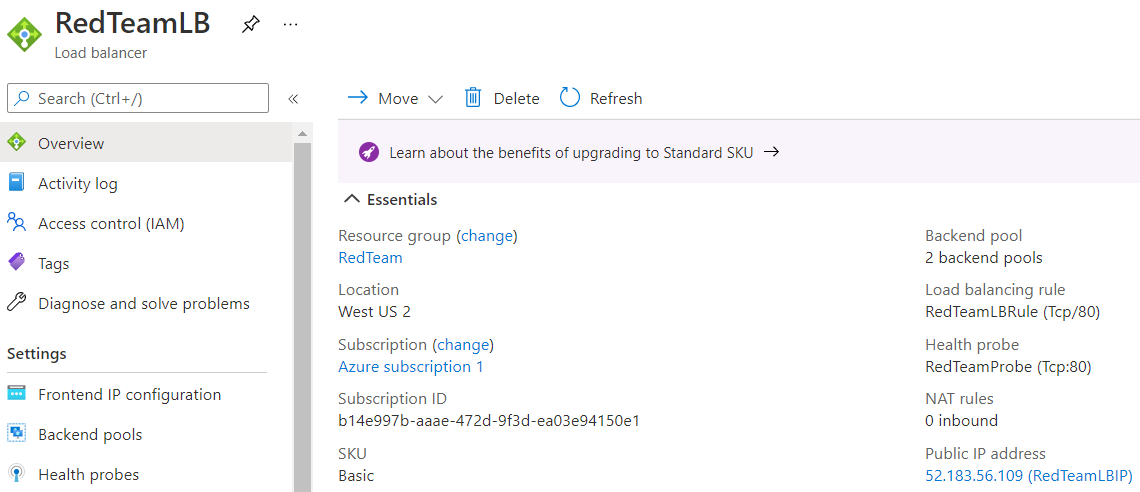
**Connect to DVWA Website:**

Verify your connection to the DVWA application through the Internet using the load balancer’s public IP address.

To locate the public IP address of our load balancer:

* From the ‘Load balancers’ page, select your load balancer.
* From the ‘Overview’ section, the publioc IP address is on the right side.

Example:



**Set-up Security Configuration (con’t)**

**Connect to DVWA Website: (con’t)**

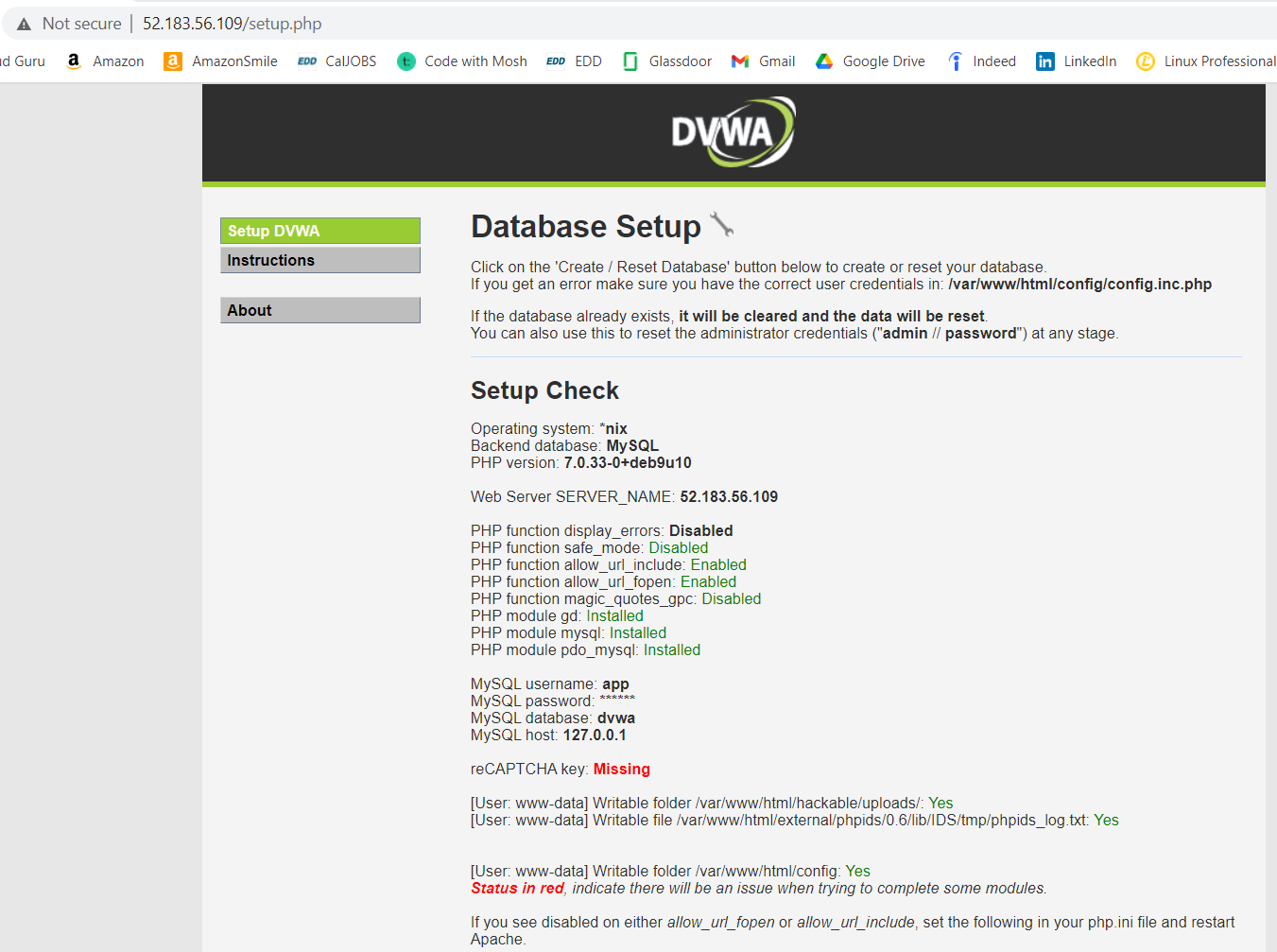
To connect, enter the following command from a browser:

http://*load\_bal\_publicIP*/setup.php

Example:

http://52.183.56.109/setup.php

Your browser should come up with the DVWA set-up page:



**ELK Stack Configuration**

The next sections are instructions to set-up our ELK Stack server.

**Create Second Virtual Network**

**Create Virtual Network:**

Create a new Virtual Network.

From the Azure Portal homepage, search for ‘Virtual networks’.

From the ‘Virtual networks’ page, click ‘New’.

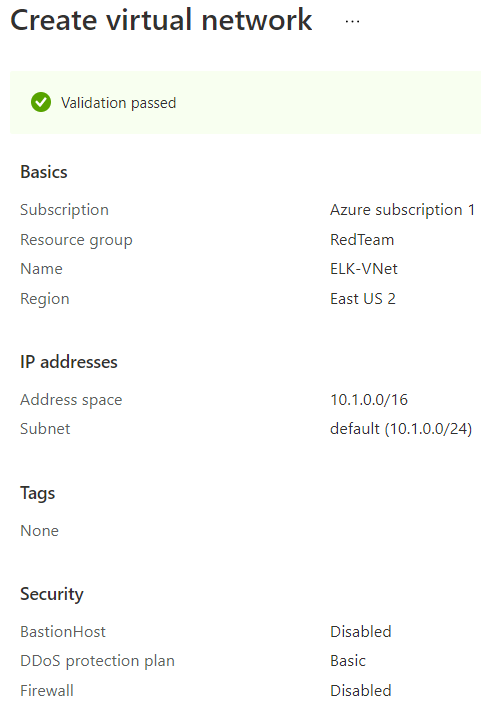
* For ‘Resource group, select the same resource group used by the Web virtual machines.
* Enter a name.
* Choose a different region from the region you selected when creating your first virtual network.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

See below for an example.

**Create Second Virtual Network (con’t)**

**Create Virtual Network: (con’t)**

Example:



**Create Second Virtual Network (con’t)**

**Create Peer Network Connection:**

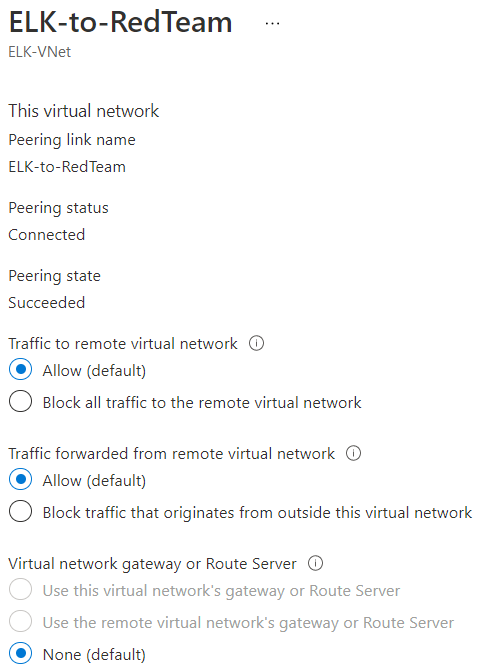
Create a peer network connection between your two virtual networks.

From the Azure Portal homepage, search for ‘Virtual networks’.

From the ‘Virtual networks’ page, select your new virtual network (ELK-VNet).

* From the left pane, under ‘Settings’, select ‘Peerings’ and click the ‘Add’ button to create a new Peering.
* Enter a name for ‘This virtual network’
  + Example: ELK-to-RedTeam
* Enter a name for ‘Remote virtual network’.
  + Example: RedTeam-to-ELK
* For ‘Virtual Network’, select your original virtual network (RedTeamVNet).

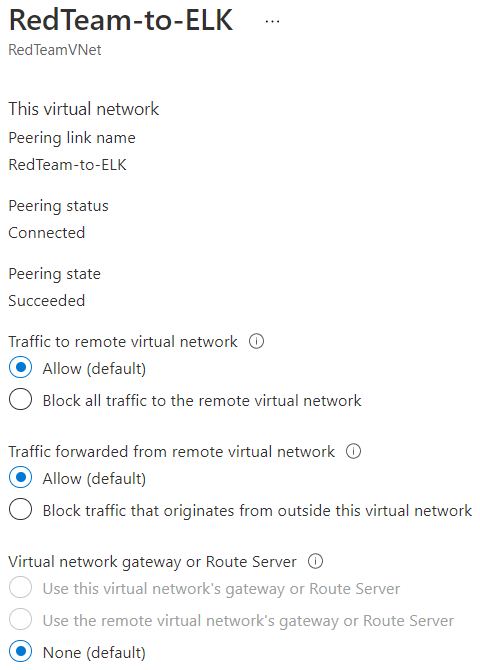
Example of the result of a peer connection between both virtual networks:



**Create Second Virtual Network (con’t)**

**Create Peer Network Connection: (con’t)**

Example of the result of a peer connection between both virtual networks: (con’t)



**Create Fourth Web (ELK) Virtual Machine**

**Copy SSH Key:**

Copy the SSH key from your Ansible container within your jump-box:

root@c0528cb8732c:~/.ssh# cat id\_rsa.pub

NOTE: This key was previously created in section ‘Create Container SSH Key - Create SSH Key’.

**Create Virtual Machine:**

Create a virtual machine for the ELK server.

From the Azure Portal homepage, search for ‘Virtual machines’.

From the ‘Virtual machines’ page, click ‘Add’ and select ‘Virtual machine’.

Basic section:

* For ‘Resource group, select the same resource group used by the Web virtual machines.
* Enter a virtual machine name.
* Choose the region you selected when creating your second (ELK) virtual network.
* For the ‘Availability options’ field, leave option to the default setting of ‘No existing availability sets …’
* Image is ‘Ubuntu Server 18.04 LTS – Gen 1’.
* Select size that is Standard\_B2s - 2 vcpus, 4 GiB
  + NOTE: The size has to be at least 4 GB!
* Authentication type should be ‘SSH public key’.
* Enter a username.
  + Use the same name used by the Web virtual machines!
* For the ‘SSH public key source’ field, select ‘Use existing public key’
* In the ‘SSH public key’ field, paste the container SSH key.
  + NOTE: This is the same SSH key used by the Web virtual machines.
* Leave other options to the default settings and click ‘Next: Disks’.

Disks section:

* For the ‘OS disk type’, select ‘Premium SSD’.
* Leave other options to the default settings and click ‘Next: Networking’.

**Create Fourth Web (ELK) Virtual Machine (con’t)**

**Create Virtual Machine: (con’t)**

Networking section:

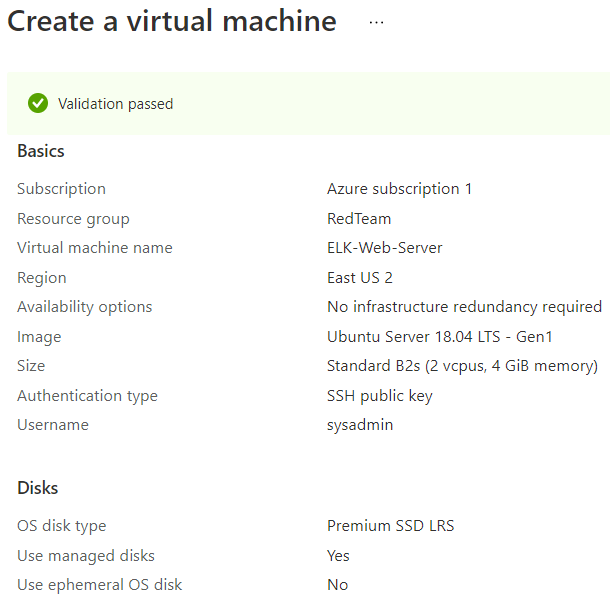
* Select the ELK virtual network.
* Leave the ‘Subnet’ field to the default settings.
* For the ‘Public IP’, click on ‘Create new.
  + From the pop-up window, leave the ‘Name’ field and other options to the default settings.
  + Click ‘OK’.
* For the ‘NIC network security group’, select ‘Advanced’.
* For the ‘Configure network security group’, leave the default name as is.
* Leave other options to the default settings.
* Click on ‘Review and Create’ and then ‘Create’.

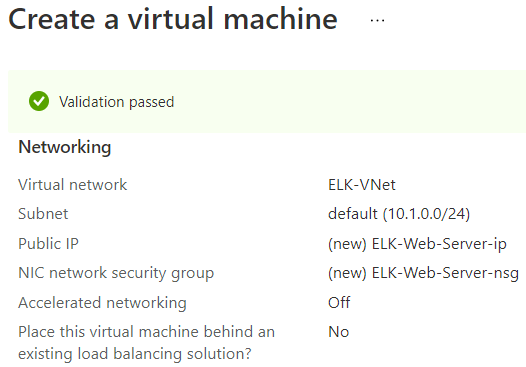
See below for an example.

**Create Fourth Web (ELK) Virtual Machine (con’t)**

**Create Virtual Machine: (con’t)**

Example:





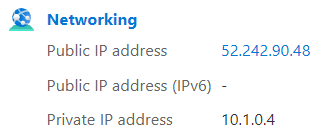
**Create Fourth Web (ELK) Virtual Machine (con’t)**

**Connect to Virtual Machine:**

In order to test the connection to this machine, you will need to use the internal (private) IP address.

To locate the internal IP address for this machine:

* From the ‘Virtual machine’ page, select the ELK virtual machine.
* From the ‘Overview’ section, the internal IP address is on the right side.



Verify the connection to the ELK virtual machine within your container.

From Git Bash, log into the Jump Box, then start/attach to your container:

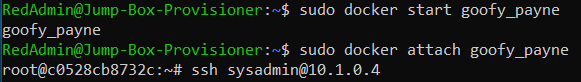
Enter the following command to connect to the ELK virtual machine using its internal IP address:

ssh *username*@*internal\_IP*

* + The username is the name created when the virtual machine was initially created.

Example:

ssh sysadmin@10.1.0.4



The prompt should look like:



**Update Hosts File**

Update the ‘hosts’ file.

From Git Bash, log into the Jump Box, then start/attach to your container.

From the container prompt, change directory to ‘/etc/ansible’ and display its contents.

root@c0528cb8732c:~# cd /etc/ansible

root@c0528cb8732c:/etc/ansible# ls -l

total 28

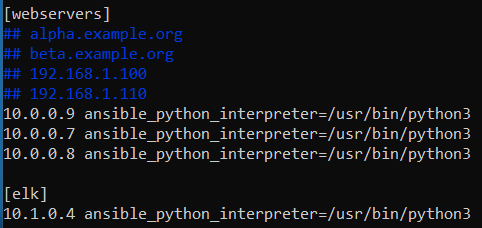
-rw-r--r-- 1 root root 19985 Dec 4 2019 ansible.cfg

-rw-r--r-- 1 root root 1016 Dec 4 2019 hosts

drwxr-xr-x 2 root root 4096 Dec 4 2019 roles

Edit the ‘hosts’ file to add the internal IP addresses for the ELK virtual machine with the python version we want to execute our scripts.

Example:



NOTE: The python version line will not be needed in the near future once python version 2 becomes obsolete.

**Create ELK Stack Playbook**

**Create Playbook:**

Create a YAML playbook for the configuration of the ELK virtual machines with Docker.

From Git Bash, log into the Jump Box, then start/attach to your container.

Change directory to ‘/etc/ansible’ and create a playbook.

The playbook configuration should contain the following:

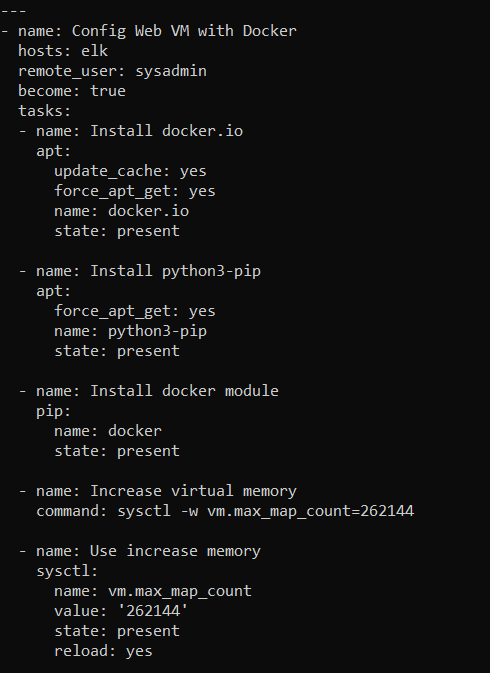
* Install ‘docker.io’ using ‘apt’.
* Install ‘python3’ using ‘apt’.
* Install ‘docker’ using ‘pip’.
* Increase virtual memory to 262144.
* Use increase memory.
* Install and launch a docker ELK container using ‘docker-container’.
* Enable the docker service using ‘systemd’.

See below for an example.

**Create ELK Stack Playbook (con’t)**

**Create Playbook: (con’t)**

Example:

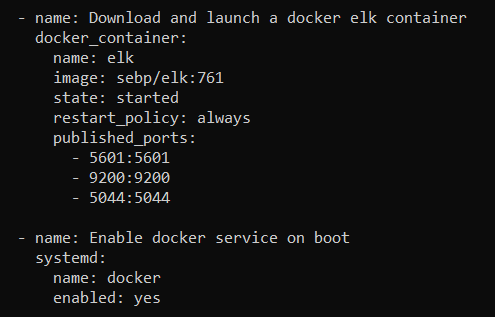


NOTE: Continues on next page!

**Create ELK Stack Playbook (con’t)**

**Create Playbook: (con’t)**

Example: (con’t)



**Execute Playbook:**

To execute a playbook, enter the following command from within your container:

ansible-playbook *playbook\_name*

Example of command:

ansible-playbook elkplaybook.yml

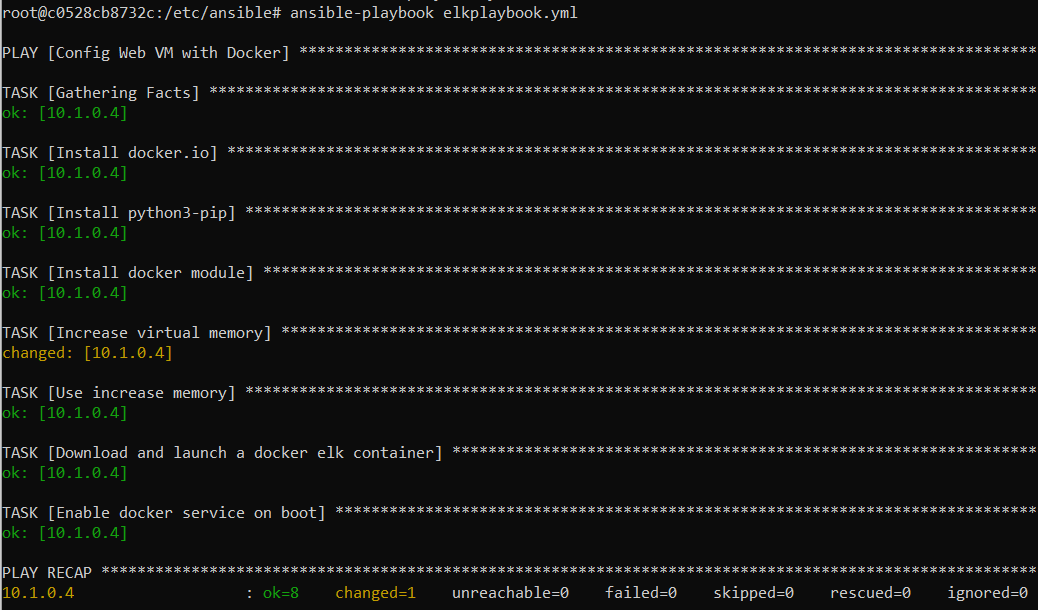
* Check if there are errors in your playbook and fix accordingly.

See below for an example.

**Create ELK Stack Playbook (con’t)**

**Execute Playbook: (con’t)**

Example of results:



**Launch Container:**

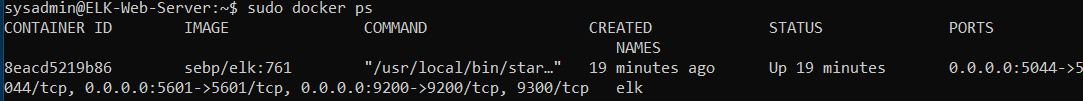
After the playbook completes without any errors, stay within your Ansible container, ssh into your ELK-Web-Server.

root@c0528cb8732c:/etc/ansible# ssh sysadmin@10.1.0.4

Check to see if container sebp/elk:761 is running by executing the following command:

sudo docker ps

Example:



**Create Inbound Security Rule (Port 5601)**

**Create Inbound Security Rule:**

Create a new inbound security rule to allow the connection from your workstation (client) public IP address to the internal IP address of the ELK virtual machine through port 5601.

From the Azure Portal homepage, search for ‘Network security groups’.

From the ‘Network security groups’ page, select the network security group you just recently created in ‘Create Network Security Group (NSG)’.

From the left pane, select ‘Inbound security rules’, then click ‘Add’.

Set the rule as:

Source: Select ‘IP Addresses’.

Source IP addresses: Client’s public IP address.

Source port ranges: \*

Destination: Select ‘IP Addresses’.

Destination: Enter ’10.1.0.4’.

* + This is the internal IP address for the ELK machine.

Destination port ranges: Enter ‘5601’.

Protocol: Any

Action: Allow

Priority: Enter a number that is lower than the numbers for the default rules listed.

Name: Enter a name such as ‘AllowPort5601’.

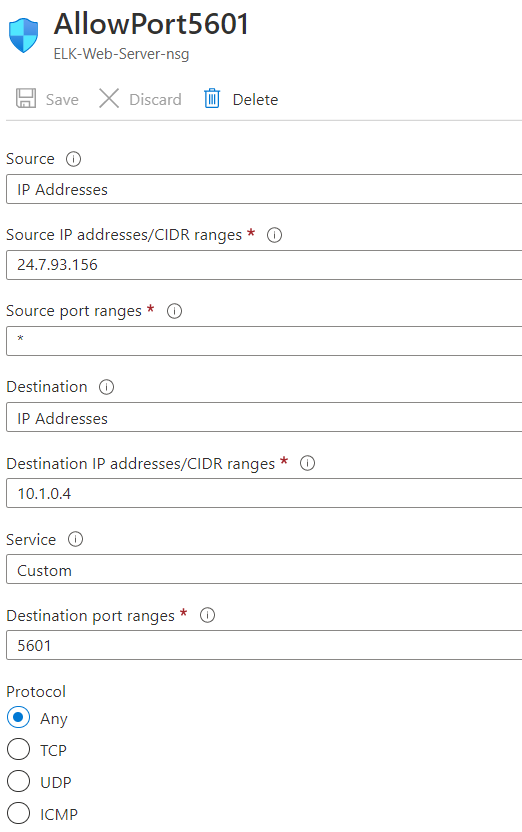
* Click ‘Add’.

See below for an example.

**Create Inbound Security Rule (Port 5601)**

**Create Inbound Security Rule: (con’t)**

Example of creating an inbound rule for Port 5601:



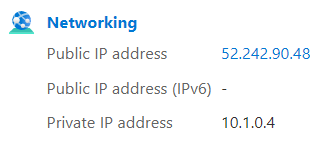
**Create Inbound Security Rule (Port 5601) (con’t)**

**Connect to Kibana:**

From a browser, verify that Kibana can load the ELK Stack server using its public IP address.

To locate the public and private IP addresses for your ELK virtual machine:

* From the ‘Virtual machine’ page on the Azure Portal, select the ELK machine.
* From the ‘Overview’ section, the public IP address is on the right side.



To connect, enter the following command from a browser:

http://*ELK\_publicIP* /app/kibana

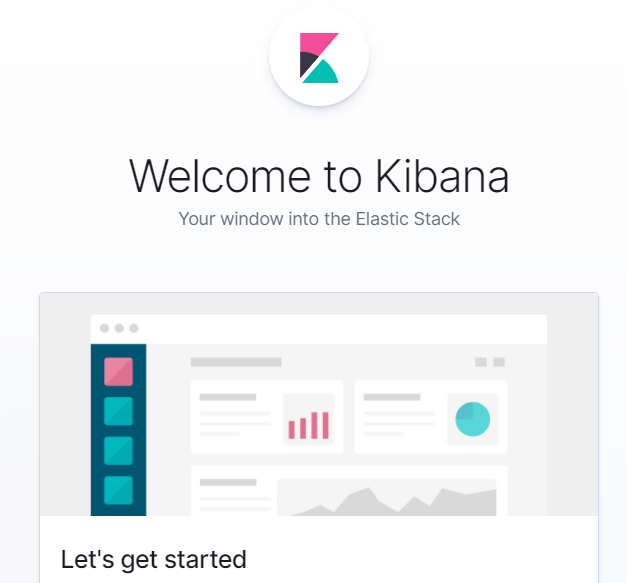
Example:

http://52.242.90.48/app/kibana

**Create Inbound Security Rule (Port 5601) (con’t)**

**Connect to Kibana: (con’t)**

Your browser should come up with the following page:



**Filebeat**

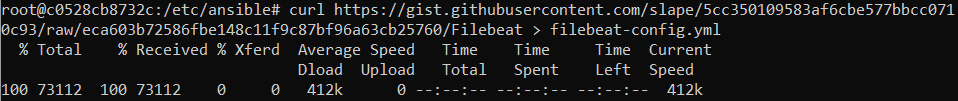
**Create Filebeat Configuration File:**

Create a Filebeat configuration file.

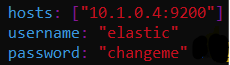
From Git Bash, log into the Jump Box, then start/attach to your container.

* Change directory to ‘/etc/ansible’.
* If it doesn’t exist, create a ‘files’ directory.
* Change to the ‘files’ directory.
* From the ‘files’ directory, copy the Filebeat configuration file from the following site into a file called ‘filebeat-config.yml’ using ‘curl’ into your Ansible container.

https://gist.githubusercontent.com/slape/5cc350109583af6cbe577bbcc0710c93/raw/eca603b72586fbe148c11f9c87bf96a63cb25760/Filebeat



* + - NOTE: The example was copied into the ‘/etc/ansible’ directory but was later moved to the ‘files’ directory.
* Edit the Filebeat configuration file accordingly.
  + On line #1106, replace the IP address with your ELK VM private IP address.
    - Leave the default port as ‘9200’
    - Leave the username/password defaulted to ‘elastic’/’changeme’.



* + On line #1806, replace the IP address with your ELK VM private IP address.
    - Leave the default port as ‘5601’.



**Filebeat (con’t)**

**Create Filebeat Playbook:**

Create a YAML playbook to install Filebeat on the DVWA container.

From Git Bash, log into the Jump Box, then start/attach to your container.

Change directory to ‘/etc/ansible’ and create a playbook.

The playbook configuration should contain the following:

* Download Filebeat deb file.

curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-amd64.deb

* Install Filebeat deb.

dpkg -i filebeat-7.4.0-amd64.deb

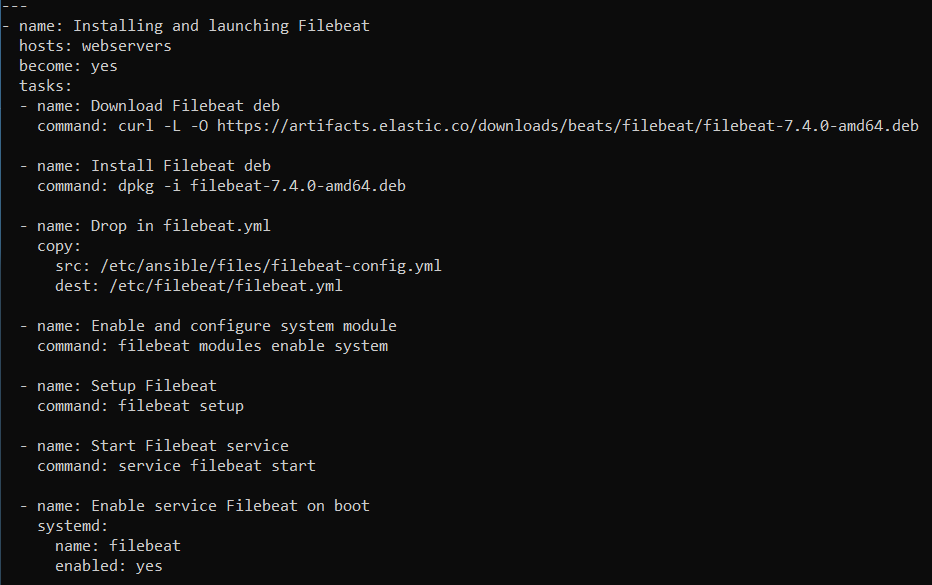
* Copy the filebeat.yml file.
* Enable and configure system module.
* Set-up Filebeat.
* Start Filebeat service.
* Enable the Filebeat service using ‘systemd’.

See below for an example.

**Filebeat (con’t)**

**Create Filebeat Playbook: (con’t)**

Example:



**Execute Filebeat Playbook:**

To execute a playbook, enter the following command from within your container:

ansible-playbook *playbook\_name*

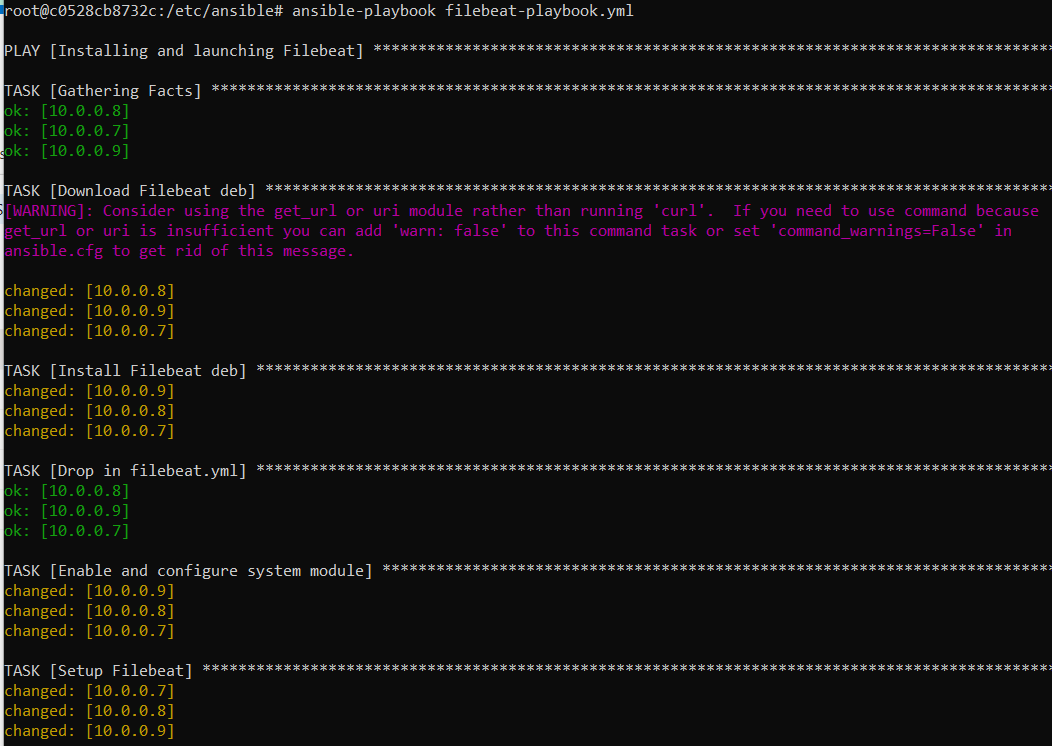
Example of command:

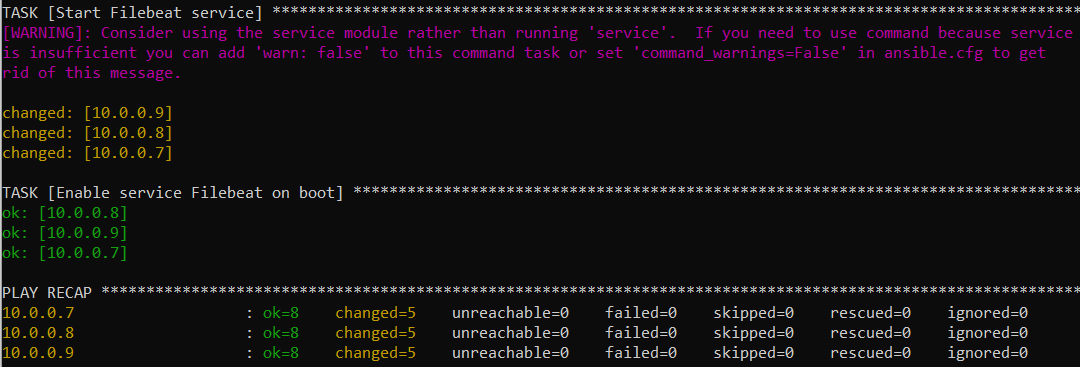
ansible-playbook filebeat-playbook.yml

**Filebeat (con’t)**

**Execute Filebeat Playbook: (con’t)**

Example of results:



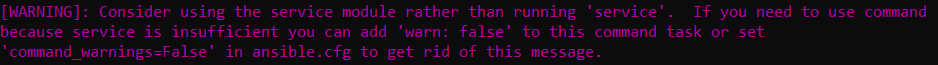


**Filebeat (con’t)**

**Execute Filebeat Playbook: (con’t)**

Example of results: (con’t)

**NOTE:** You will receive the following valid warning for both the ‘Download Filebeat deb’ and ‘Start Filebeat service’ tasks.



**Metricbeat**

**Create Metricbeat Configuration File:**

Create a Metricbeat configuration file.

From Git Bash, log into the Jump Box, then start/attach to your container.

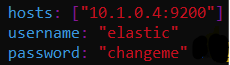
* Change directory to ‘/etc/ansible’.
* If it doesn’t exist, create a ‘files’ directory.
* Change to the ‘files’ directory.
* Copy the Metricbeat configuration file from the following site into a file called ‘metricbeat-config.yml’ using ‘curl’ into your Ansible container.

<https://gist.githubusercontent.com/slape/58541585cc1886d2e26cd8be557ce04c/raw/0ce2c7e744c54513616966affb5e9d96f5e12f73/metricbeat>

* Edit the Metricbeat configuration file accordingly.
  + On line #62, replace the IP address with your ELK VM private IP address.
    - Leave the default port as ‘5601’.



* + On line #95, replace the IP address with your ELK VM private IP address.
    - Leave the default port as ‘9200’
    - Leave the username/password defaulted to ‘elastic’/’changeme’.



* + Save your file.

**Metricbeat (con’t)**

**Create Metricbeat Playbook:**

Create a YAML playbook to install Metricbeat on the DVWA container.

From Git Bash, log into the Jump Box, then start/attach to your container.

Change directory to ‘/etc/ansible’ and create a playbook.

The playbook configuration should contain the following:

* Download Metricbeat deb file.

curl -L -O https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.4.0-amd64.deb

* Install Metricbeat deb.

dpkg -i metricbeat-7.4.0-amd64.deb

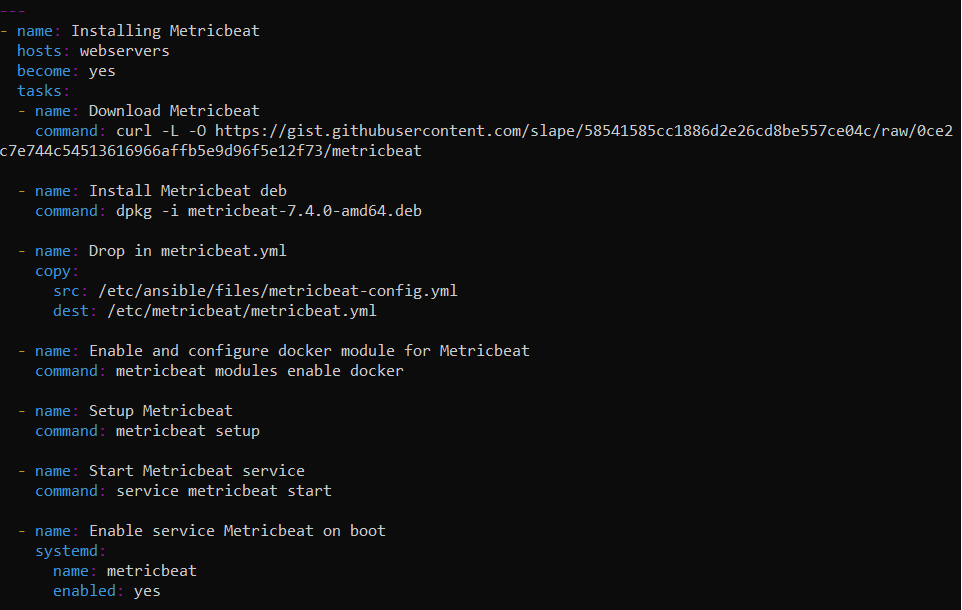
* Copy the metricbeat.yml file.
* Enable and configure Metricbeat docker module
* Set-up Metricbeat.
* Start Metricbeat service.
* Enable the Metricbeat service using ‘systemd’.

See below for an example.

**Metricbeat (con’t)**

**Create Metricbeat Playbook: (con’t)**

Example:



**Execute Metricbeat Playbook:**

To execute a playbook, enter the following command from within your container:

ansible-playbook *playbook\_name*

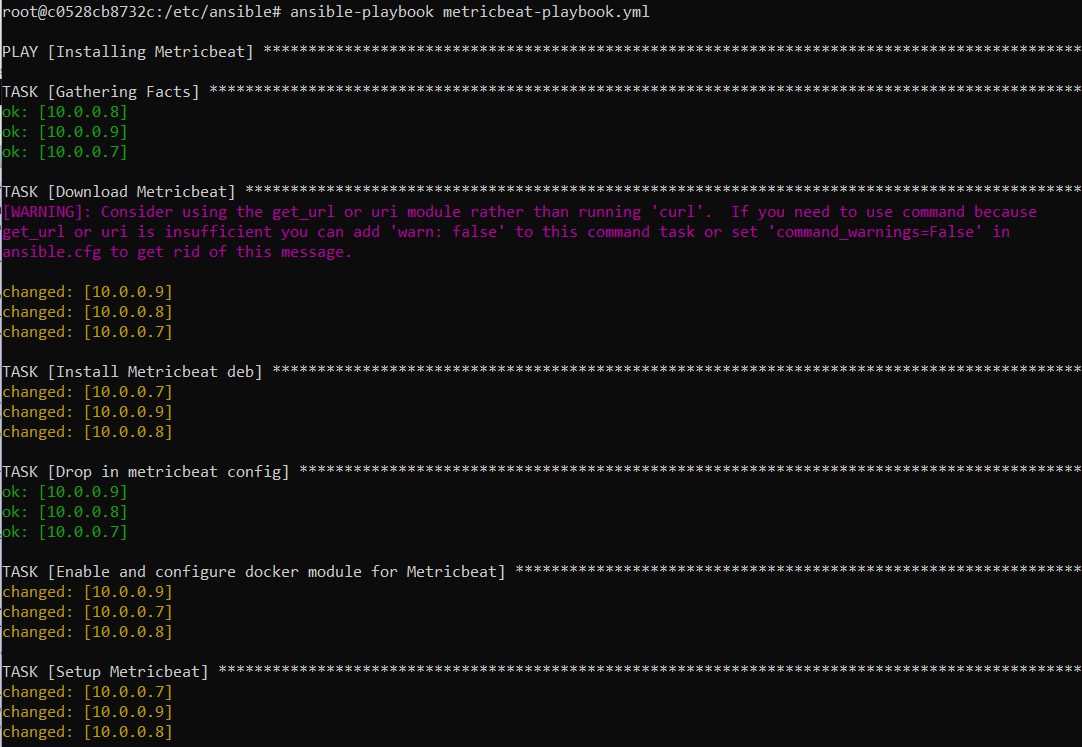
Example of command:

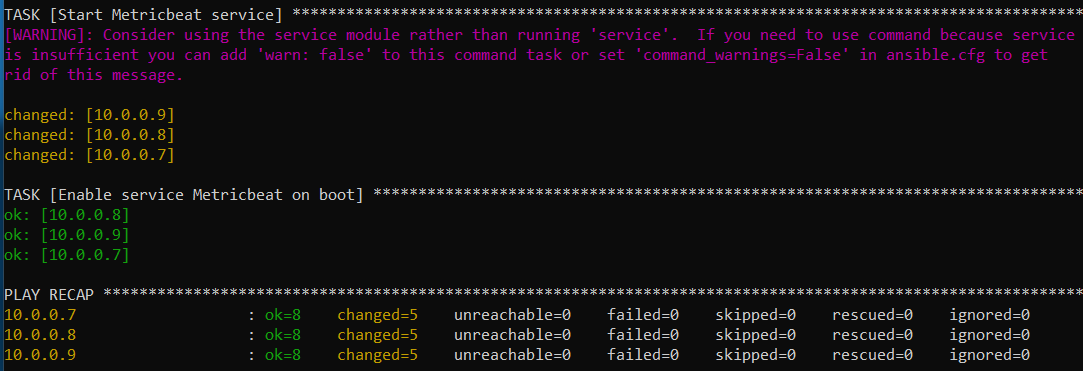
ansible-playbook metricbeat-playbook.yml

**Metricbeat (con’t)**

**Execute Metricbeat Playbook:**

Example of results:



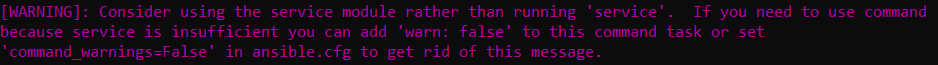


**Metricbeat (con’t)**

**Execute Metricbeat Playbook:**

Example of results: (con’t)

**NOTE:** You will receive the following valid warning for both the ‘Download Metricbeat’ and ‘Start Metricbeat service’ tasks.



**Kibana – Filebeat**

Verify that Filebeat is working from your ELK server’s GUI in Kibana by exploring system logs.

**NOTE:** Make sure your ELK server container is up and running.

From Git Bash, log into the Jump Box, then start/attach to your container.

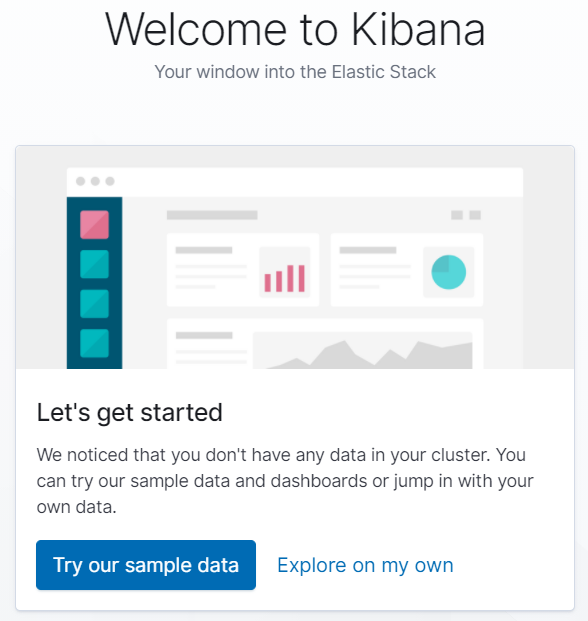
* Verify the connection to the ELK virtual machine within your container:

root@c0528cb8732c:~# ssh sysadmin@10.1.0.4

* From a browser, navigate to:

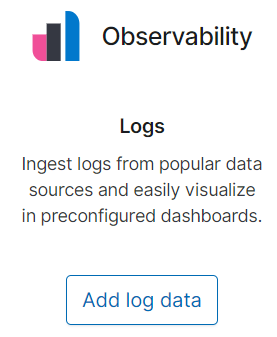
http://52.242.90.48:5601/app/kibana

* From the Kibana home page, click on ‘**Explore on my Own**’.

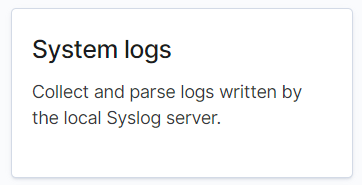


**Kibana – Filebeat (con’t)**

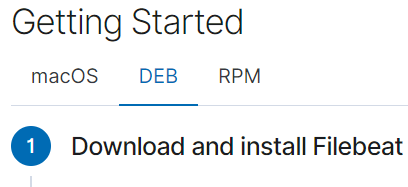
* From the ‘Observability’ section, click ‘Add Log Data’.



* Select ‘System Logs’.



* Under ‘Getting Started’, click on the ‘DEB’ tab.



* Scroll to the bottom of the page and from the ‘**Module status**’ step, click ‘**Check Data**’.

You should see the following message if this step was successful.

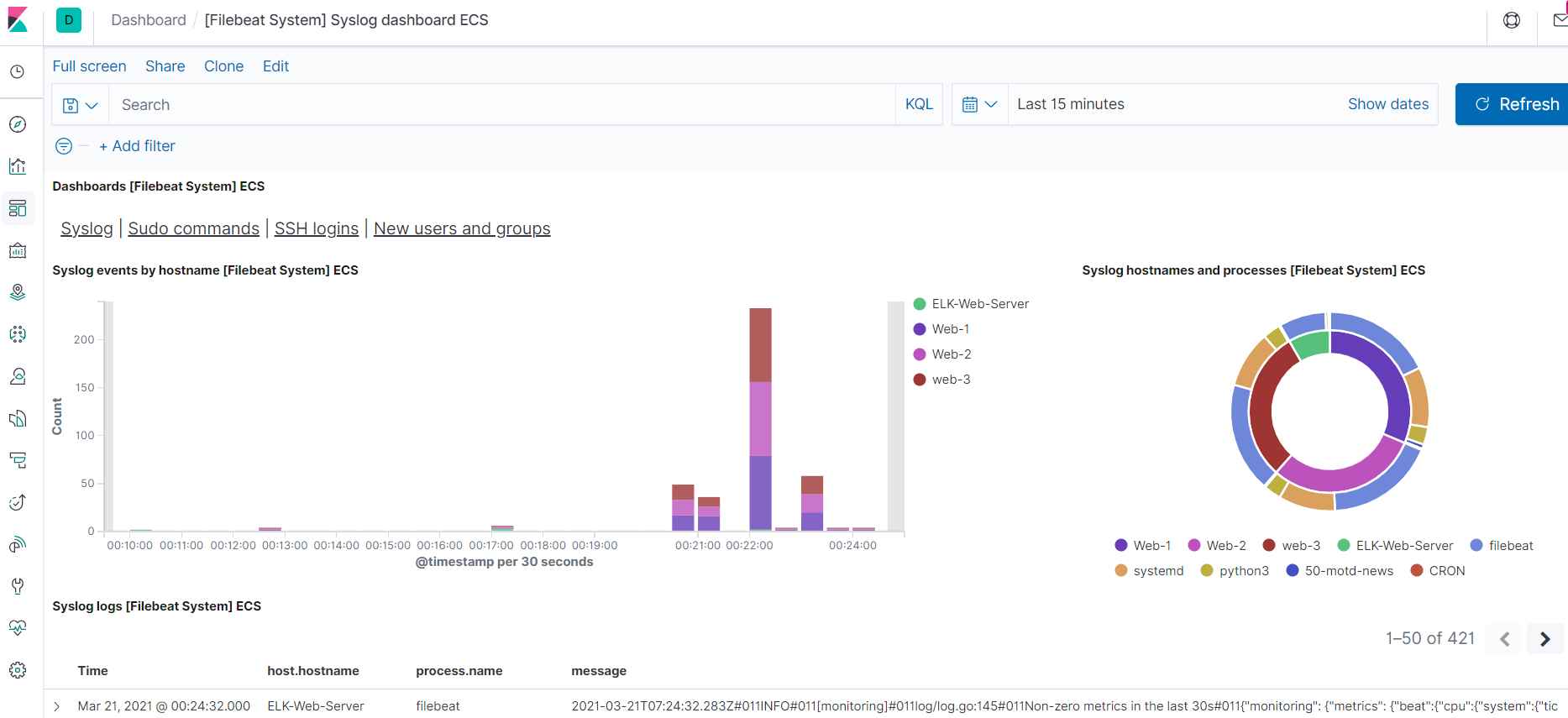


**Kibana – Filebeat (con’t)**

* To view the system logs in Kibana, click on the ‘System logs dashboard’ button.



Below is an example of the system logs being displayed on the Kibana dashboard:



**Kibana - Metricbeat**

Verify that Filebeat is working from your ELK server’s GUI in Kibana by exploring Docker metrics.

**NOTE:** Make sure your ELK server container is up and running.

From Git Bash, log into the Jump Box, then start/attach to your container.

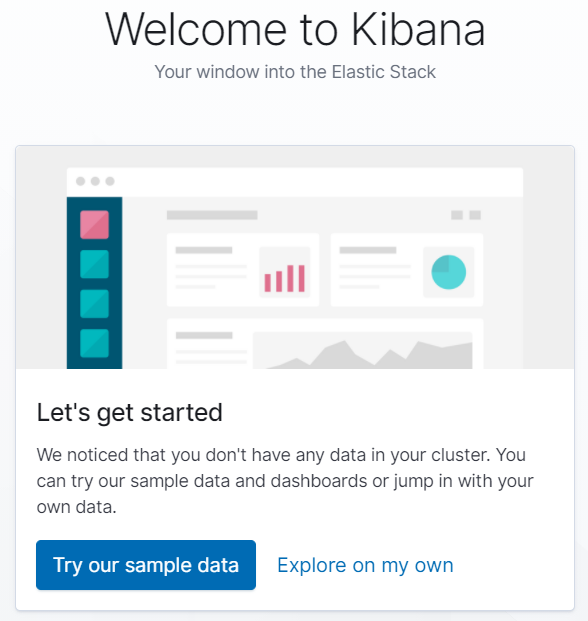
* Verify the connection to the ELK virtual machine within your container:

root@c0528cb8732c:~# ssh sysadmin@10.1.0.4

* From a browser, navigate to:

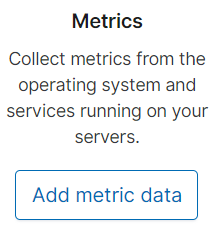
http://52.242.90.48:5601/app/kibana

* From the Kibana home page, click on ‘**Explore on my Own**’.

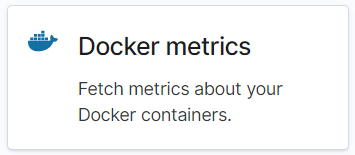


**Kibana – Metricbeat (con’t)**

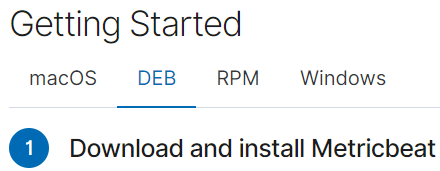
* From the ‘Observability’ section, click ‘Add Metric Data’.



* Select ‘Docker Metrics’.



* Under ‘Getting Started’, click on the ‘DEB’ tab.



* Scroll to the bottom of the page and from the ‘**Module status**’ step, click ‘**Check Data**’.
* You should see the following message if this step was successful.

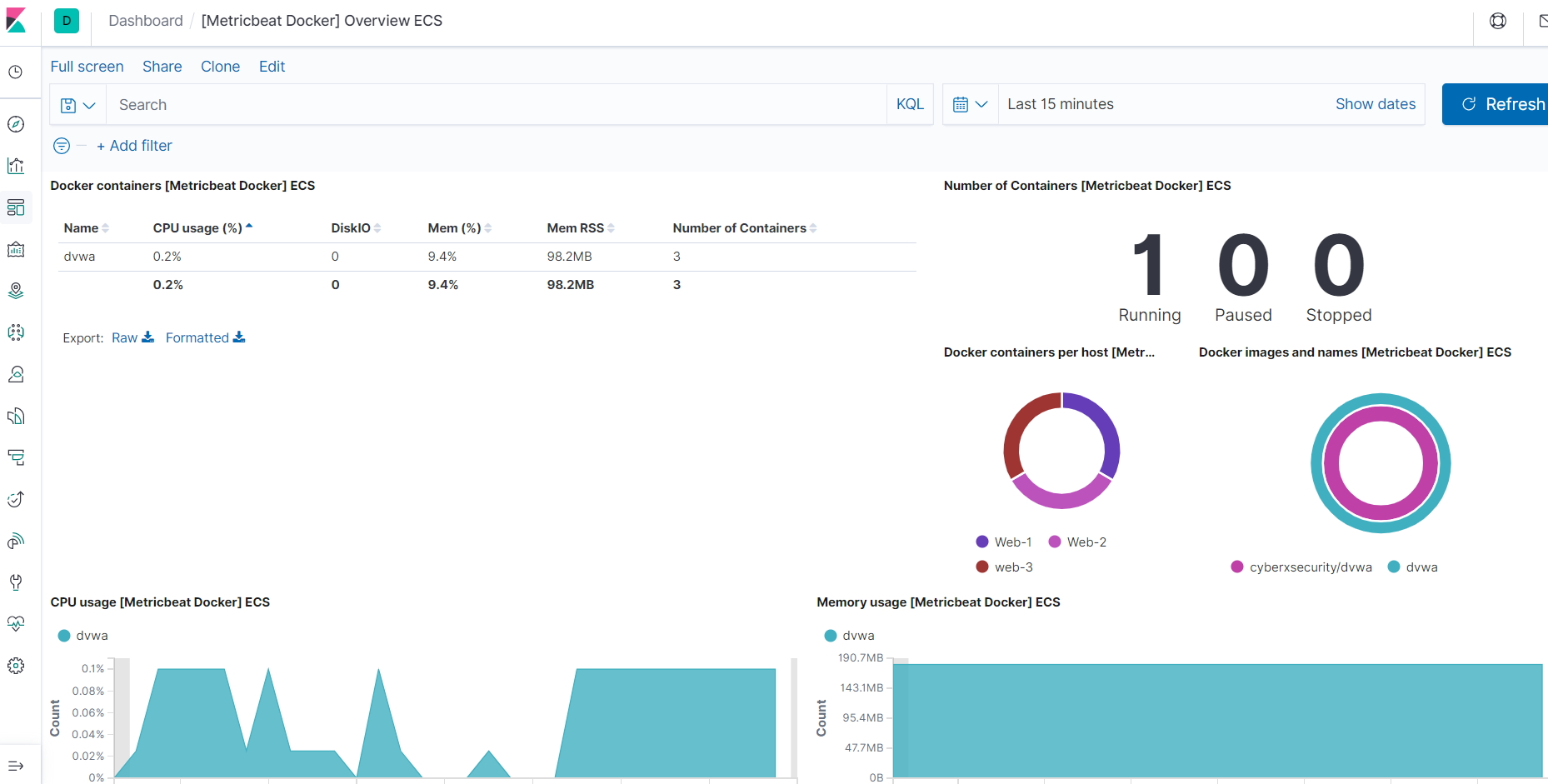


**Kibana – Metricbeat (con’t)**

* To view the Docker metrics in Kibana, click on the ‘Docker metrics dashboard’ button.



Below is an example of the Metricbeat Docker being displayed on the Kibana dashboard:



**Miscellaneous**

The next sections are information that provide some guidance with the Azure Virtual Network and ELK Stack configurations.

**GitHub Repository**

For our deployment, we used Git Bash to access the Jump Box.

From Git Bash, SSH into the Jump Box and follow the steps below:

1. Log into the Jump Box.
2. Start and attach to an Ansible container.
3. Change directory to ‘/etc/ansible’.
4. The Playbooks and files can be copied over by cloning them from Git or creating the playbooks manually.

NOTE: Make sure you are in your Ansible container on the Jump Box.

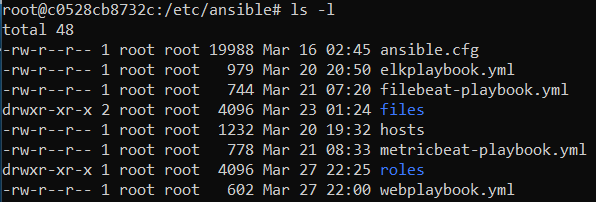
To clone from Git, follow the directions below:

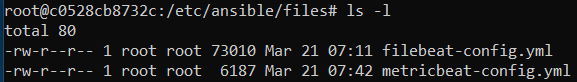
* + Change to a directory where you want to clone from Git.
    - Example: /root
  + Clone repository from Git.
    - git clone https://github.com/ttambio/elk-stack-project.git
  + Change directory to your Git repository.
    - Example: cd elk-stack-project
  + Copy Playbooks and files to the appropriate folders in ‘/etc/ansible’.
    - cp elk-stack-project/Ansible/hosts /etc/ansible
    - cp elk-stack-project/Ansible/\*.cfg /etc/ansible
    - cp elk-stack-project/Ansible/\*.yml /etc/ansible
    - cp elk-stack-project/Ansible/\*config.yml /etc/ansible/files

See below for example of the location of these files.

**GitHub Repository (con’t)**

The Playbooks and files should be located in the ‘/etc/ansible’ directory in the Ansible container of the Jump Box.





**Locate IP Addresses**

The following indicates the areas where to find the public/private addresses of the machines used.

* To locate the public IP address of your workstation (client):

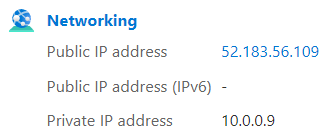
This can be done by going to: https://whatismyipaddress.com/

* To find the private IP address of the Jump Box:
  + From the ‘Virtual machine’ page on the Azure Portal, select the Jump Box.
    - From the ‘Overview’ section, the private IP address is on the right side.
  + From the Jumpbox, enter the command ‘ifconfig’.

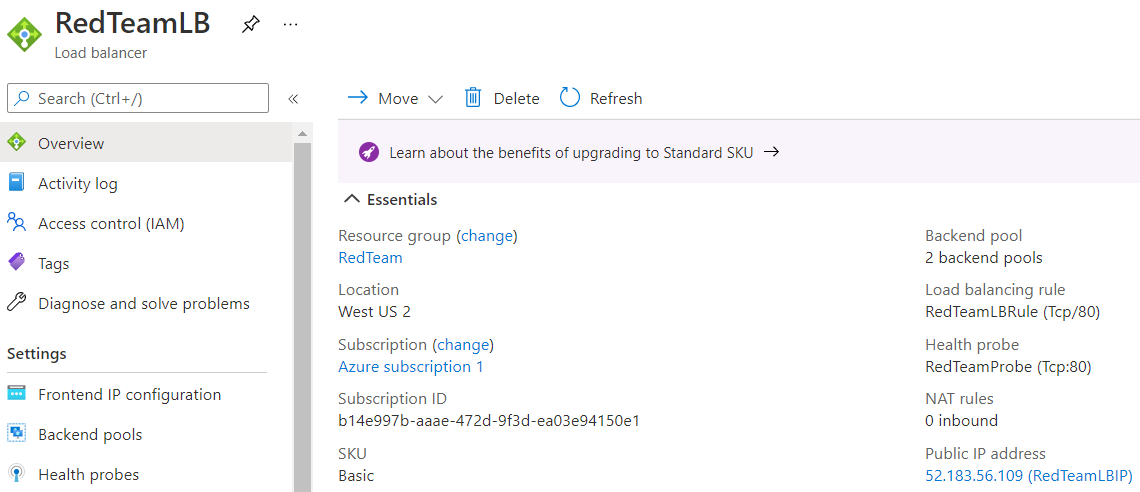


**Locate IP Addresses (con’t)**

* To locate the private IP address for virtual machine Web-1:
  + From the Virtual machine’ page on the Azure Portal, select Web-1.
  + From the ‘Overview’ section, the private IP address is on the right side.

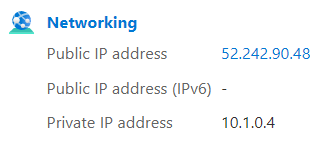


* To locate the public IP address of our load balancer:
  + From the ‘Load balancers’ page on the Azure Portal, select your load balancer.
  + From the ‘Overview’ section, the public IP address is on the right side.



**Locate IP Addresses (con’t)**

* To locate the public and private IP addresses for your ELK virtual machine:
  + From the ‘Virtual machine’ page on the Azure Portal, select the ELK machine.
  + From the ‘Overview’ section, the public IP address is on the right side.



**SSH keys**

The following are the paths to the SSH keys used in the configuration of our Azure Network and ELK Stack.

Client (workstation) public ssh key:

/c/Users/*client\_name*/.ssh/Azure21\_rsa.pub

NOTE: Public ssh key was manually named to ‘Azure\_rsa.pub’.

Ansible container public ssh key:

/root/.ssh/id\_rsa.pub

NOTE: This ssh key is within an Ansible container.

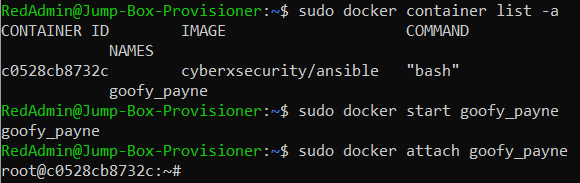
**Machine Login**

To log into the Jump Box and then into an Ansible container.

To log into the jumpbox from your workstation in Git Bash, enter the following command:

ssh -i /c/Users/*pc\_name*/.ssh/Azure21\_rsa RedAdmin@52.183.91.26

The prompt will display as shown below, display container list, then start/attach to your Ansible container:



To log or test connection to your Web and ELK virtual machines, enter the following command from within your Ansible container (example Web-1):



The following screenshots shows the cursor prompt for our virtual machines once we are logged in accordingly.

Web-1:



Web-2:



Web-3:



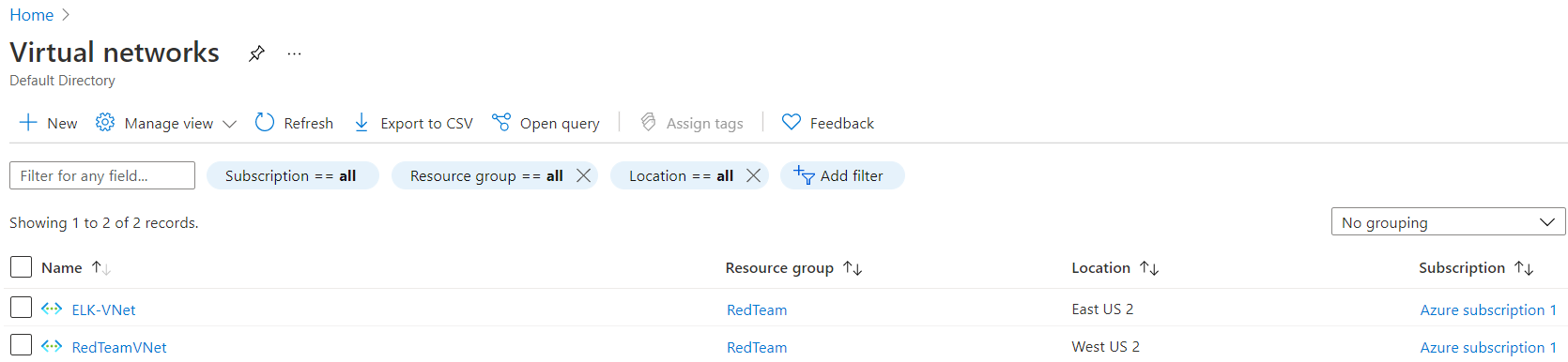
ELK-Web-Server:



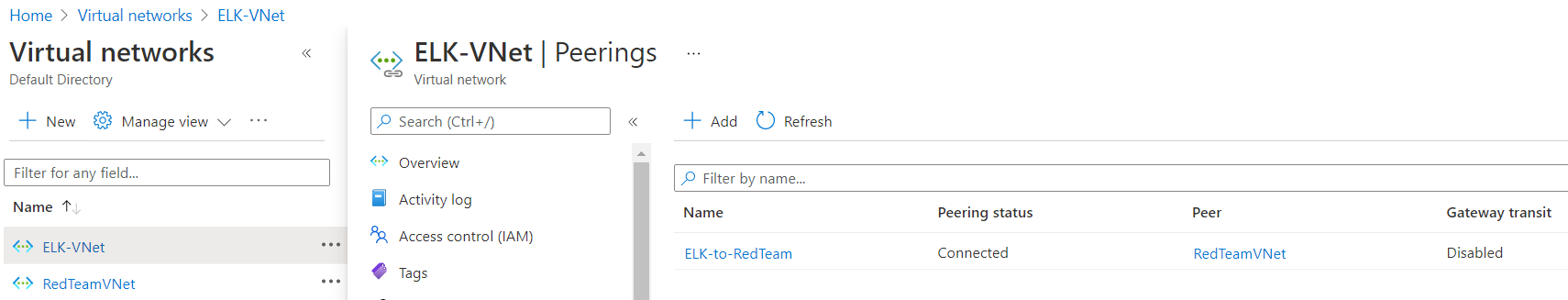
**Azure Portal - Screenshots**

The following screenshots are the machines/rules/connections created for the Azure Network and ELK Stack.

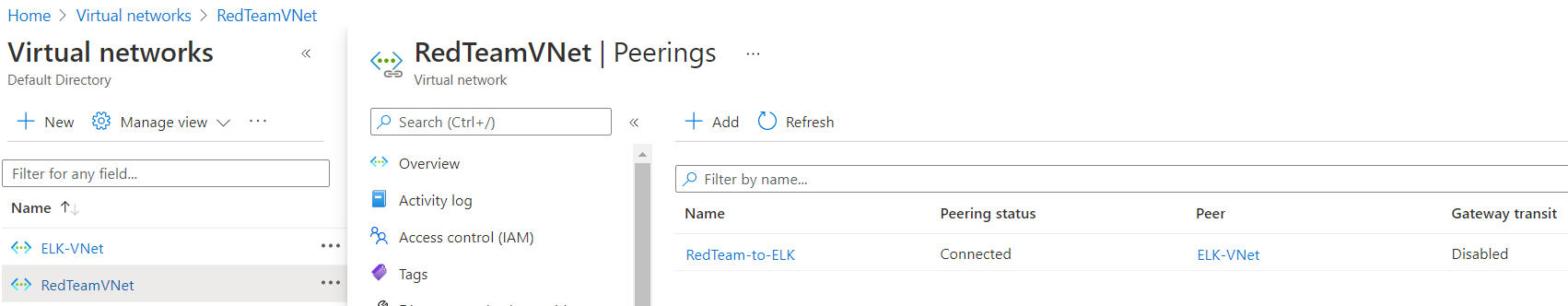
‘Virtual networks’ page:



ELK-VNet – Peerings:

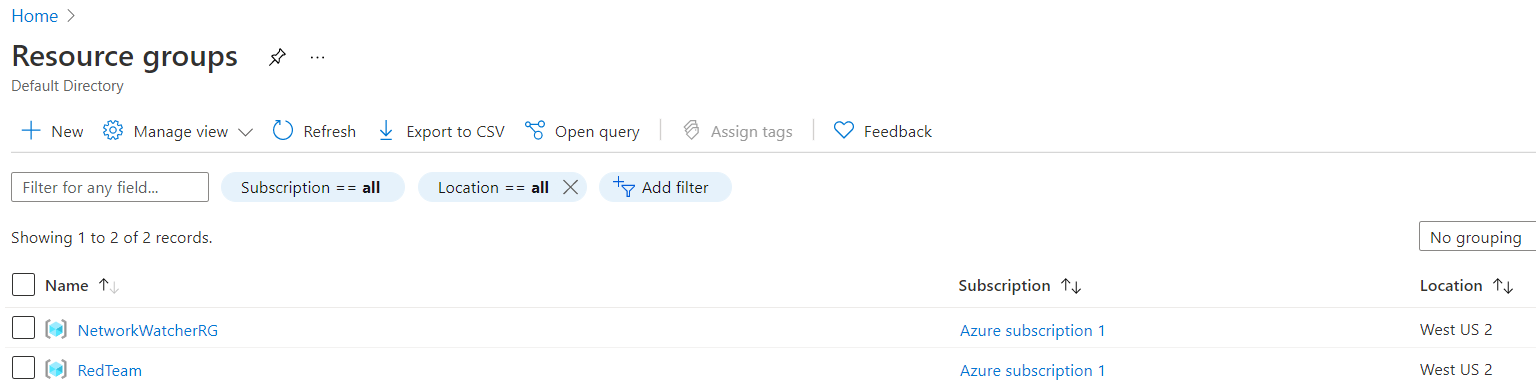


RedTeamVNet – Peerings:

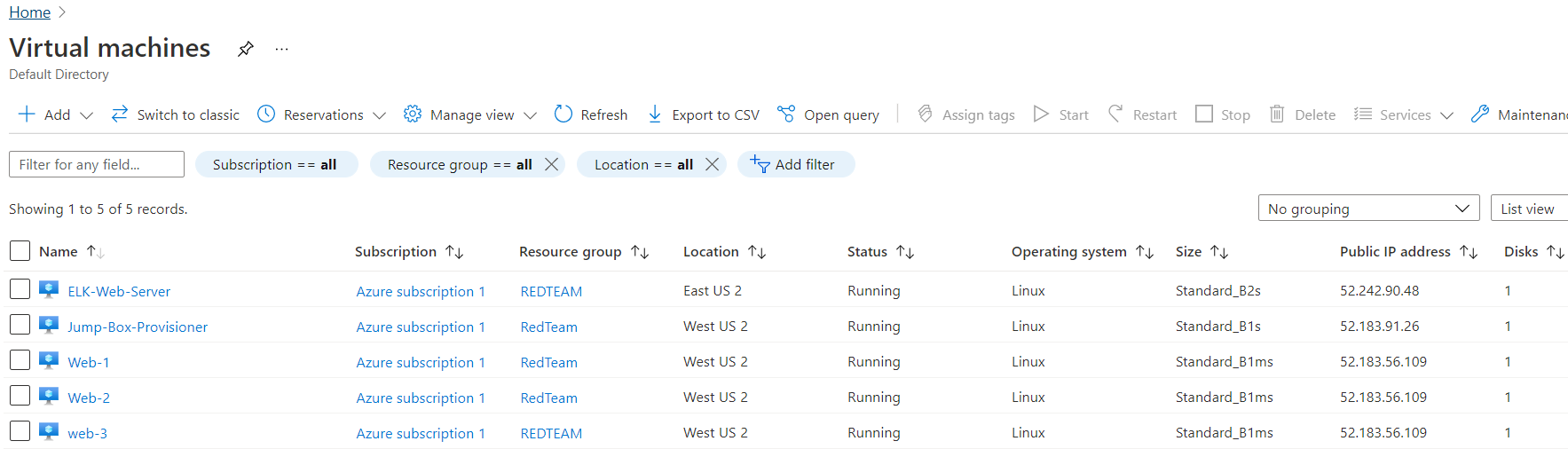


**Azure Portal – Screenshots (con’t)**

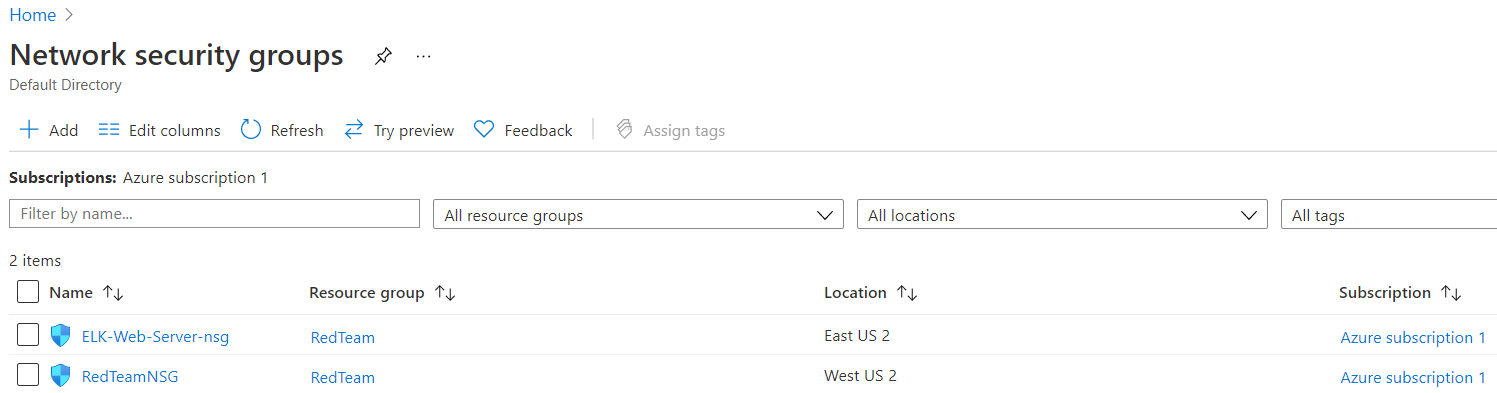
‘Resource groups’ page:



‘Virtual machine’ page.



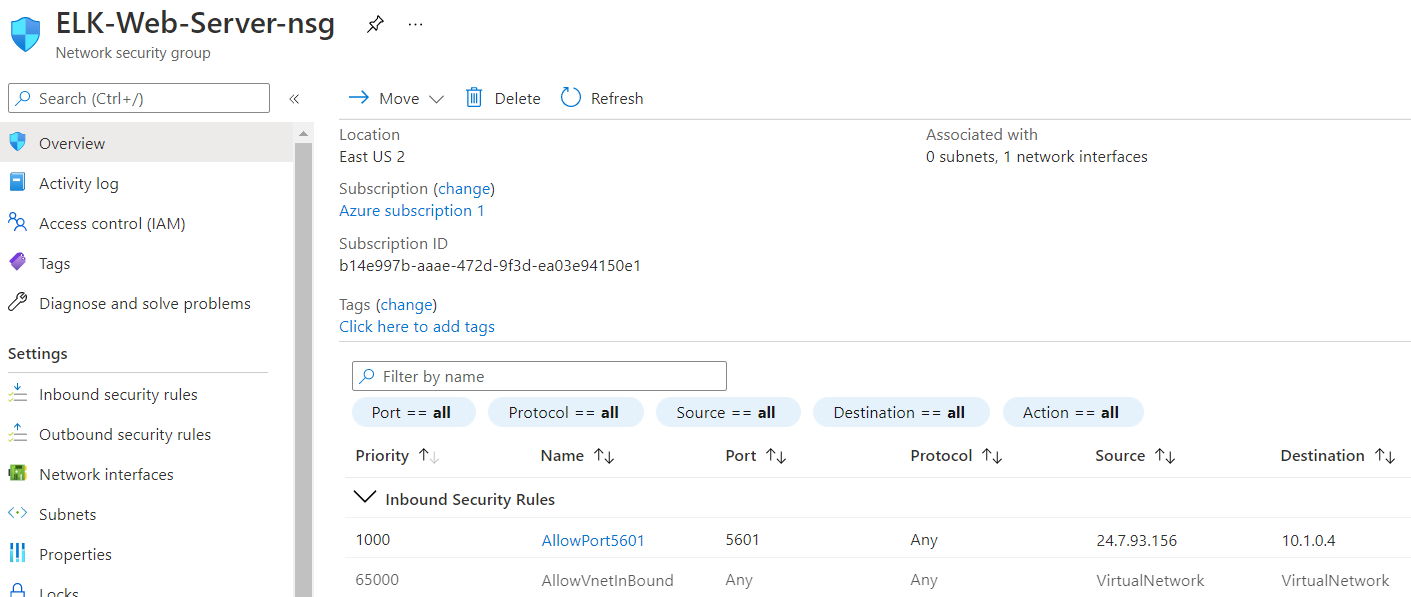
‘Network security group’ page:



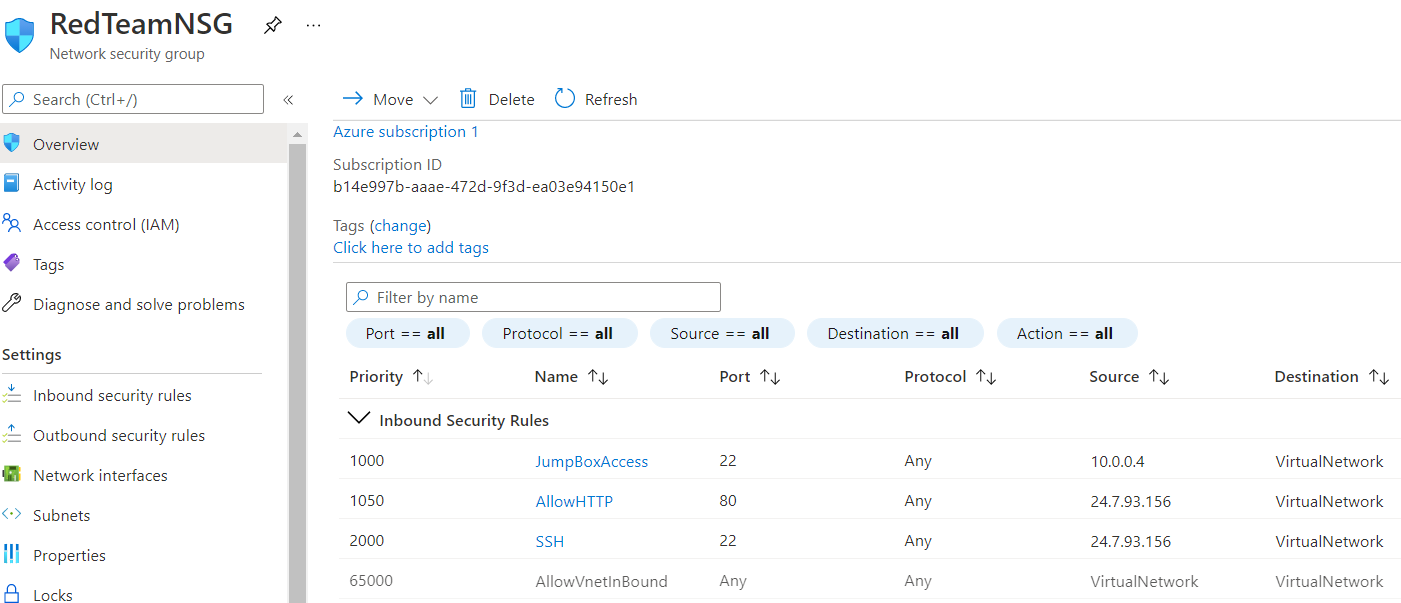
**Azure Portal – Screenshots (con’t)**

‘Network security group’ page: (con’t)

ELK-Web-Server-nsg - Network security group:

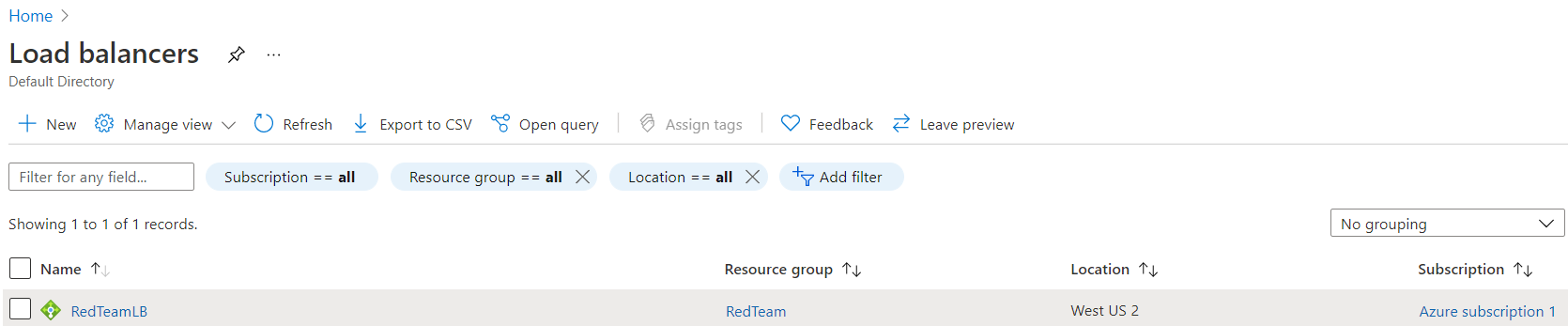


RedTeamNSG - Network security group:

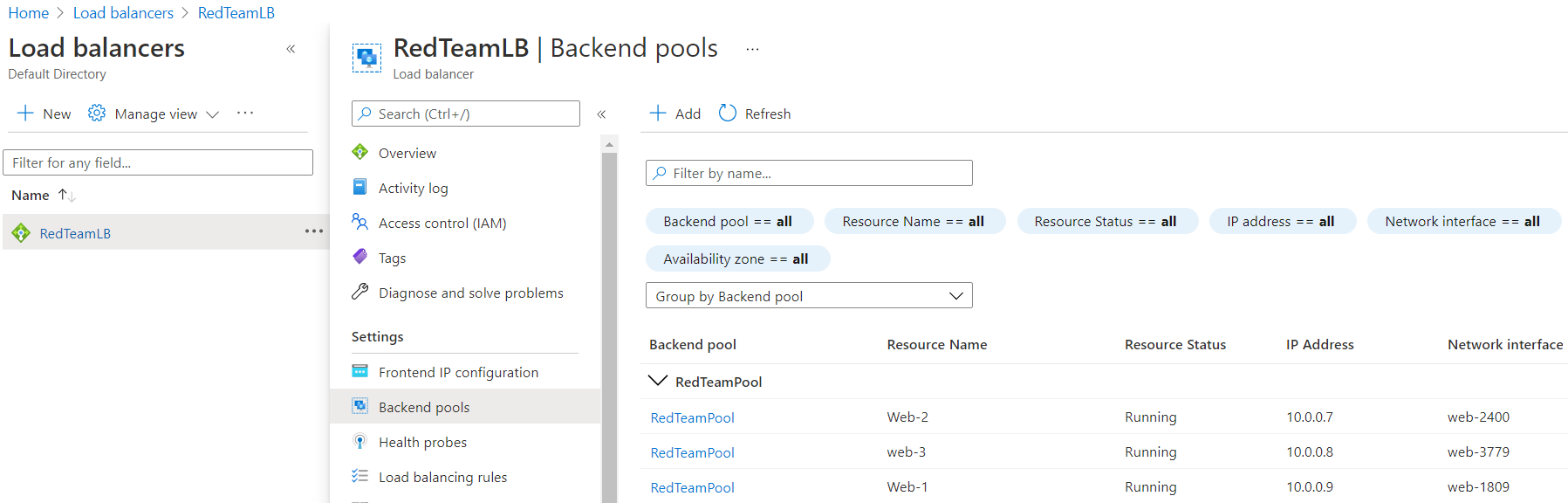


**Azure Portal – Screenshots (con’t)**

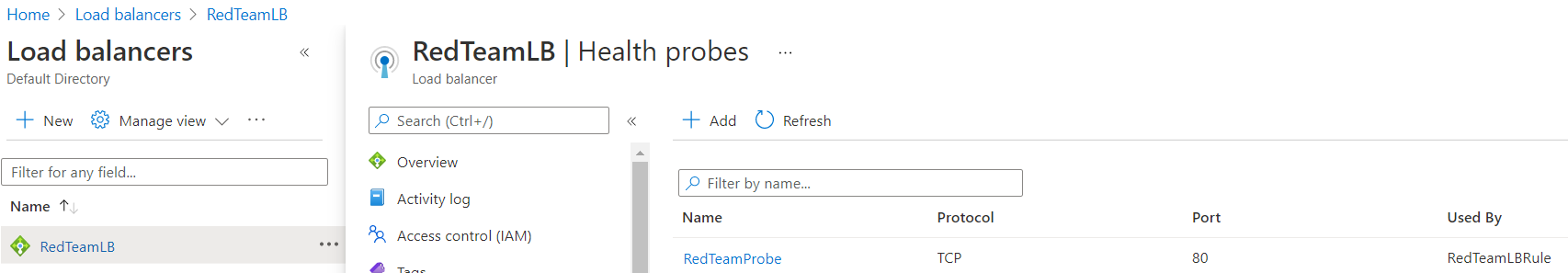
‘Load balancers’ page:



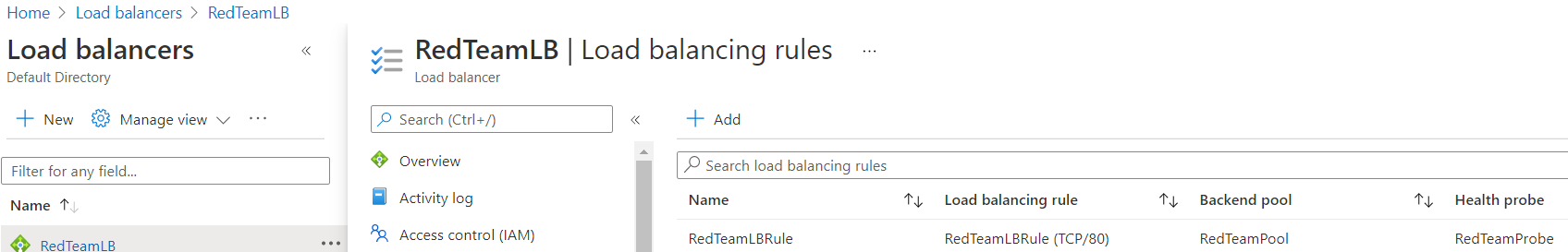
RedTeamLB – Backend pools:



RedTeamLB – Health probes

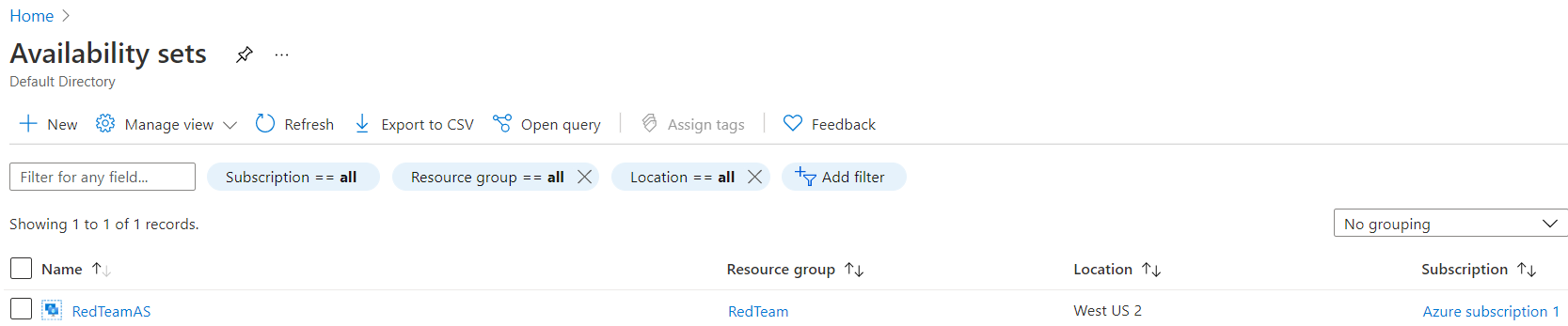


RedTeamLB – Load balancing rules

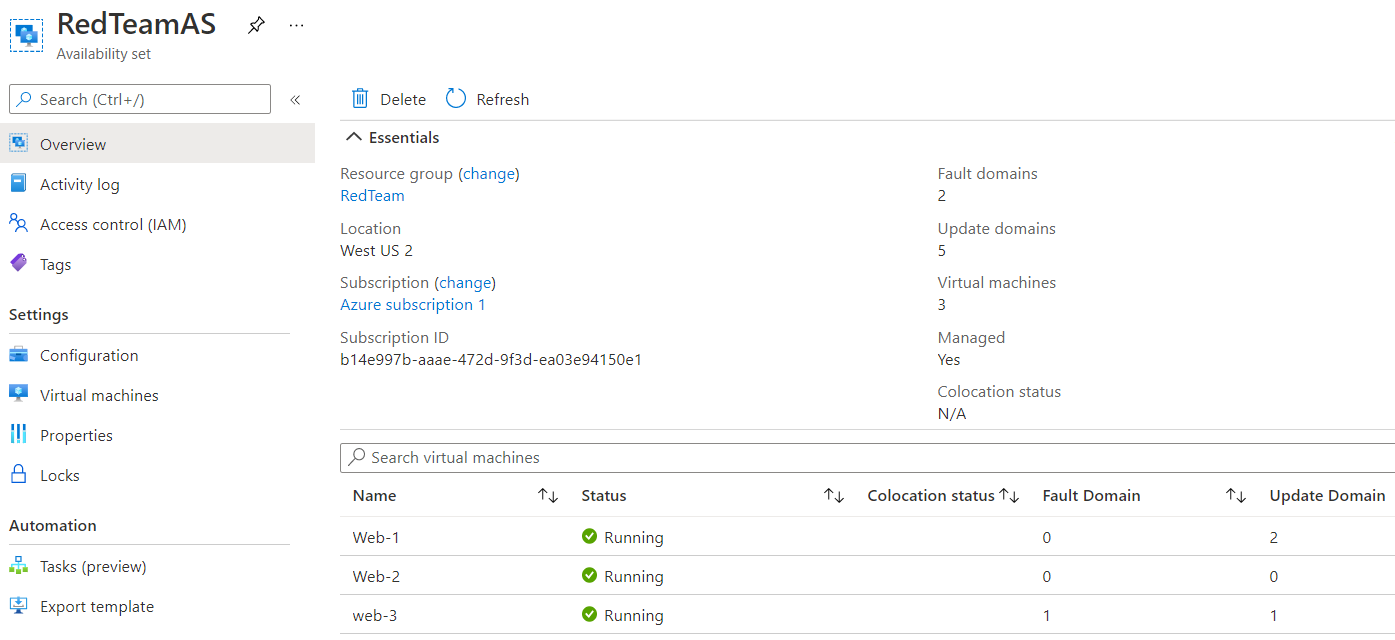


**Azure Portal – Screenshots (con’t)**

‘Availability sets’ page:



RedTeamAS – Availability set:



**Explore Kibana**

The following document contains an activity to view system logs in Kibana:

Documents/Exploring Kibana

**Pen Testing**

The following document contains an activity to perform different scenarios of penetration testing in Kibana:

Documents/Kibana - Continued