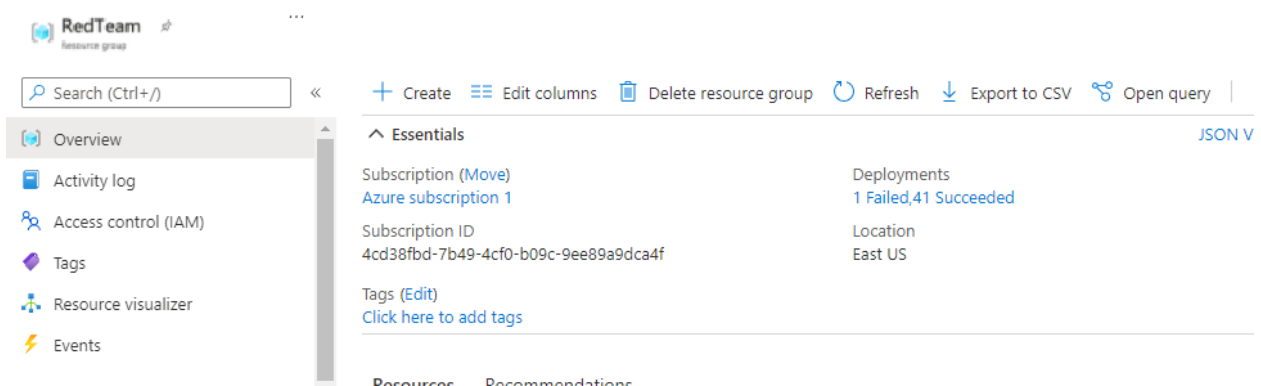


A description of deployment

1) Setting up the Cloud Environment

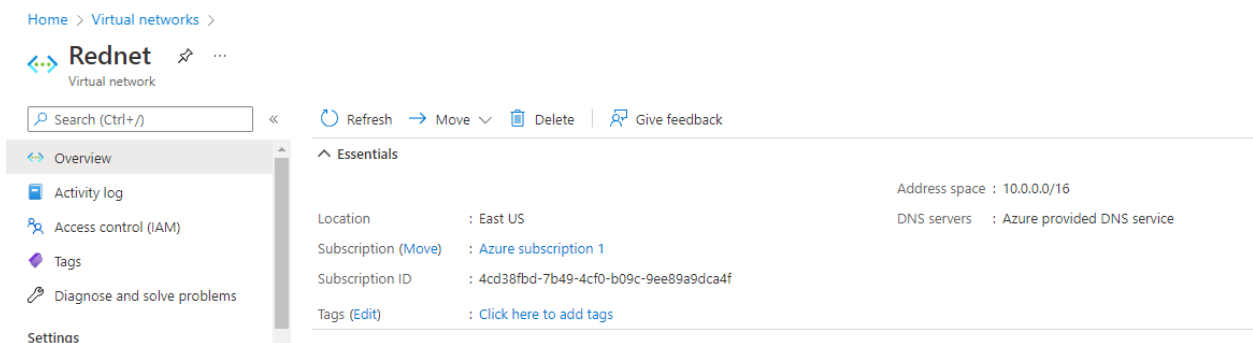
Create Resource Group

- Create a resource group and a region, I chose US East region:
 - Named as Resource Group RedTeam.






Create Vnet (Virtual Network)

- Create a new Virtual Network.
- Make sure to select the resource group we created previously as well as the same region:
 - Also, use the default IP and subnet settings.



Create a Network Security Group (NSG)

- Create a Network Security Group that is part of the Resource Group RedTeam.
- Named my NSG "Firewall".
- Make sure the NSG is in the same region as everything else we've created thus far.

 **Firewall**  

Network security group

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Inbound security rules

Outbound security rules

Network interfaces

Subnets

Properties

Locks

→ Move

🗑 Delete

🔄 Refresh

🗨 Give feedback

East US0 subnets, 10 network interfaces

Subscription (Move)
Azure subscription 1

Subscription ID
4cd38fbd-7b49-4cf0-b09c-9ee89a9dca4f

Tags (Edit)
[Click here to add tags](#)

Port == all

Protocol == all

Source == all

Destination == all

Action == all

Priority ↑↓	Name ↑↓	Port ↑↓	Protocol ↑↓	Source ↑↓
Inbound Security Rules				
100	ssh_incoming	22	TCP	216.209.173.22
110	docker_Ansib_ssh	22	TCP	10.0.0.7
120	Red_http_allow	80	TCP	216.209.0.0/16

2) Create the Virtual Machines

Create Jump Box VM

- Log in to Azure account.
- Click on the virtual machines box and then click new.
- Under the resource group, select the group RedTeam.
- Named the VM JumpT-Box.
- Use a Ubuntu server with at least 1GB of memory.
- Use a public SSH key from from local computer and give it a username
- Use "ssh-keygen" to create a public key that doesn't have one.
 - My username is sysadmin.

The screenshot shows the Azure portal interface for a virtual machine named 'JumpT-box'. The left sidebar contains navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Networking, Connect, Disks, Size, Security, Advisor recommendations, and Extensions & applications. The main area displays the VM's status as 'Stopped (deallocated)' and provides details such as location (East US), subscription (Azure subscription 1), and size (Standard DS1 v2). The 'Networking' section shows the public IP address as 52.149.134.141 and the private IP address as 10.0.0.7. The 'Properties' section lists the computer name as 'JumpT-box', health state as '-', operating system as 'Linux', publisher as 'Canonical', offer as 'UbuntuServer', and plan as '18_04-lts-gen2'.

Network Security Group Rules

- This is an overview of all of the inbound and outbound rules for the Firewall NSG:

Inbound Security Rules				
100	ssh_incoming	22	TCP	216.209.173.22
110	docker_Ansib_ssh	22	TCP	10.0.0.7
120	Red_http_allow	80	TCP	216.209.0.0/16
65000	AllowVnetInBound	Any	Any	VirtualNetwork
65001	AllowAzureLoadBalanc...	Any	Any	AzureLoadBalancer
65500	DenyAllInBound	Any	Any	Any
Outbound Security Rules				
65000	AllowVnetOutBound	Any	Any	VirtualNetwork
65001	AllowInternetOutBound	Any	Any	Any
65500	DenyAllOutBound	Any	Any	Any

Set up Docker.io on the Jump Box VM

- SSH into your Jump-Box VM, turn on your machine on Azure before that:

```
ssh sysadmin@[public IP]
```

- Once logged in, implement the following:
 - `sudo apt install docker.io`
 - `sudo docker pull cyberxsecurity/ansible`
 - **Launch the ansible container:** `docker run -ti cyberxsecurity/ansible:latest bash` to make sure it works
 - `type exit`

Config and Hosts File

- `cd` into the `/etc/ansible/` directory and `nano ansible.cfg` file
 - scroll to the `remote_user` section and update to include `sysadmin` instead of `root`. Save and exit.
- `nano /etc/ansible/hosts` file
 - Uncomment the `[webservers]` header
 - Under the header, add the internal IP address of the 3 VMs:
 - `10.0.0.11 ansible_python_interpreter=/usr/bin/python3`
 - `10.0.0.12 ansible_python_interpreter=/usr/bin/python3`
 - `10.0.0.13 ansible_python_interpreter=/usr/bin/python3`

Create 3 Virtual Machines

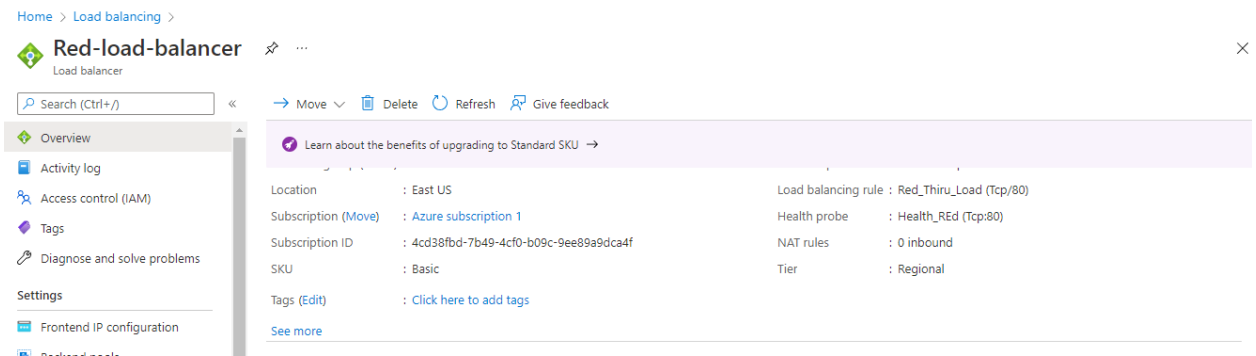
- We will create 3 additional virtual machines that will be web servers.
- We will name them Web-1, Web-2, and Web-3.
- Follow this criteria:
 - Allow no public IP address.
 - Create a new availability set, I called mine webset, set the 3 VMs to this.
 - Connect your VMs to the RedNet VNet and to the Firewall NSG.
- Use a public SSH key from the JumpT-Box VM docker container and give it a username .
 - Use "ssh-keygen" to create a public key if you don't have one.
 - username is sysadmin.
- To make sure it works:
 - SSH into the Jump Box VM
 - Start and attach your docker container: `sudo docker start "name of the container" && sudo docker attach "name of the container"`

- Once in the container, SSH into each VM to make sure they work:

- `ssh sysadmin@10.0.0.12`
- `ssh sysadmin@10.0.0.13`
- `ssh sysadmin@10.0.0.14`

3) Load Balancer

- Create a new Load Balancer in Azure.
- Select static IP address and select the same Resource Group and region.
- Select create new public IP address.
- I named my Load Balancer: Red-Team-LB
- Create a new Backend Pool and add the 3 VMs to it.



4) Logging into Jump Box Provisioner

- Log in to Azure and turn on your Jump Box Provisioner virtual machine.
- Open your personal computer terminal

```
Thiru@DESKTOP-4MI3KGM MINGW64 ~  
$ ssh -i thiru sysadmin@52.149.134.141  
Enter passphrase for key 'thiru':  
Enter passphrase for key 'thiru':  
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1064-azure x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Thu Dec 16 15:41:11 UTC 2021  
  
System load:  0.0                Processes:            111  
Usage of /:   10.6% of 28.90GB   Users logged in:     0  
Memory usage: 8%                IP address for eth0:  10.0.0.7  
Swap usage:   0%                IP address for docker0: 172.17.0.1  
  
* Super-optimized for small spaces - read how we shrank the memory  
  footprint of MicroK8s to make it the smallest full K8s around.  
  
  https://ubuntu.com/blog/microk8s-memory-optimisation  
  
3 updates can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
New release '20.04.3 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Wed Dec 15 19:39:36 2021 from 216.209.173.22  
sysadmin@JumpT-box:~$
```

5) Starting Docker

- Check to see which containers you have:

```
sysadmin@JumpT-box:~$ sudo docker container list -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
F94e263ae442	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (137) 20 hours ago		festive_hofstadter
4603f1cf89b5	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (0) 11 days ago		confident_hopper
9238e469483f	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (0) 11 days ago		admiring_chatterjee
1659993e1552	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (2) 11 days ago		competent_chandrasekhar
13060f548160	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (130) 11 days ago		serene_boyd
430c77bef5f1	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (0) 11 days ago		wonderful_volhard
5ca0e65e0983	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (0) 11 days ago		jolly_jackson
93306b6561e5	cyberxsecurity/ansible	"/bin/sh -c /bin/bash"	11 days ago	Exited (137) 11 days ago		hungry_bohr
e532ddcd0dc	cyberxsecurity/ubuntu:bionic	"bash"	11 days ago	Exited (0) 11 days ago		goofy_bardeen

- my container is: festive_hofstadter
- To start your container, use the following commands:

```
sysadmin@JumpT-box:~$ sudo docker attach festive_hofstadter
You cannot attach to a stopped container, start it first
sysadmin@JumpT-box:~$ sudo docker start festive_hofstadter
festive_hofstadter
sysadmin@JumpT-box:~$ sudo docker attach festive_hofstadter
```

6) Install ELK Stack

Create a new VNet (Virtual Network)

- Create a new VNet that is in the same resource group (RedTeam) but in a different region.
- I created one and named it Canadacentral:

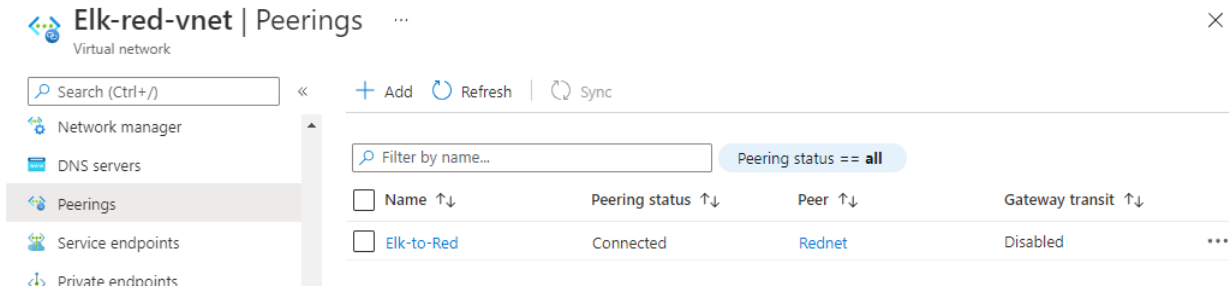
The screenshot shows the Azure portal interface for a Virtual Network named 'Elk-red-vnet'. The left sidebar contains navigation options: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Address space, Connected devices, Subnets, DDoS protection, and Firewall. The main content area is titled 'Essentials' and displays the following information:

- Location:** Canada Central
- Subscription (Move):** Azure subscription 1
- Subscription ID:** 4cd38fbd-7b49-4cf0-b09c-9ee89a9dca4f
- Tags (Edit):** [Click here to add tags](#)
- Address space:** 10.1.0.0/16
- DNS servers:** Azure provided DNS service

Below this information, there is a section for 'Connected devices' with a search bar. A table lists the connected devices:

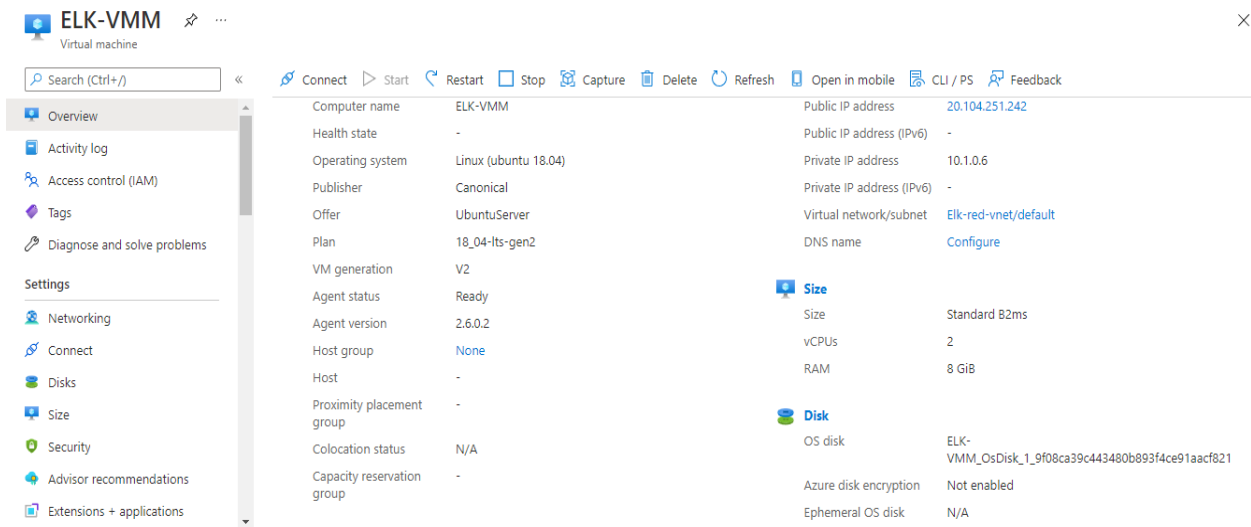
Device ↑↓	Type ↑↓	IP Address ↑↓	Subnet ↑↓
elk-vm75	Network interface	10.1.0.5	default
elk-vm138	Network interface	10.1.0.6	default

- Create a peer connection between the two VNets (RedNet and Elk-red-net)
- In my Elk-red-vnet page, I clicked on Peerings and added a new Peering with the following settings:
 - I created a new connection from EastNet to RedNet and called it: Elk-to-Red, connecting to my RedNet VNet.
 - I created another connection from RedNet to EastNet and called it: Red-to-Elk.



Create a new Virtual Machine

- Create a new Ubuntu VM with a 8GB minimum RAM size.
- The IP address will be public.
- The VNet will be Elk-red-vnet, and the resource group will be the one we have for the other VMs, which is RedTeam.
- I used the public key from the ansible container and my username as sysadmin.
- I named my VM: ELK.



- Once the VM is created, I ssh'd into the VM to make sure it works:

```
root@f94e263ae442:~# ssh -i thiruwx sysadmin@10.1.0.6
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1064-azure x86_64)

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

System information as of Thu Dec 16 15:57:32 UTC 2021

System load:  0.0               Processes:            120
Usage of /:   16.8% of 28.90GB   Users logged in:     0
Memory usage: 34%              IP address for eth0:  10.1.0.6
Swap usage:   0%               IP address for docker0: 172.17.0.1

* Super-optimized for small spaces - read how we shrank the memory
  footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

3 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Dec 15 19:41:39 2021 from 10.0.0.7
sysadmin@ELK-VMM:~$
```

Download and Configure the Container

- We will add the new VM IP address to the hosts file in the ansible container
- Edited the hosts file and added the elk IP address under the [elk] group:

```
[webservers]
#alpha.example.org
#beta.example.org
#192.168.1.100
#192.168.1.110
10.0.0.12 ansible_python_interpreter=/usr/bin/python3
10.0.0.11 ansible_python_interpreter=/usr/bin/python3
10.0.0.13 ansible_python_interpreter=/usr/bin/python3

[elk]
10.1.0.6 ansible_python_interpreter=/usr/bin/python3
```

- Created a playbook that will configure the ELK server, named it "install-elk.yml":

Launched and Exposed the Container

- Ran the yaml playbook, made sure it worked:

```

root@f94e263ae442:~# ansible-playbook pentest.yml
[WARNING]: ansible.utils.display.initialize_locale has not been called, this may result in incorrectly calculated text widths that can cause Display to print incorrect line lengths

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

TASK [Gathering Facts] *****
ok: [10.0.0.13]
ok: [10.0.0.11]
ok: [10.0.0.12]

TASK [docker.io] *****
ok: [10.0.0.12]
ok: [10.0.0.11]
changed: [10.0.0.13]

TASK [Install pip3] *****
ok: [10.0.0.12]
ok: [10.0.0.11]
changed: [10.0.0.13]

TASK [Install Docker python module] *****
ok: [10.0.0.12]
ok: [10.0.0.11]
changed: [10.0.0.13]

TASK [download and launch a docker web container] *****
[DEPRECATION WARNING]: The container_default_behavior option will change its default value from "compatibility" to "no_defaults" in community.docker 2.0.0. To remove this warning, please specify an explicit value for it now. This feature will be removed from community.docker in version 2.0.0. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
ok: [10.0.0.11]
ok: [10.0.0.12]
changed: [10.0.0.13]

TASK [Enable docker service] *****
ok: [10.0.0.12]
ok: [10.0.0.11]
ok: [10.0.0.13]

PLAY RECAP *****
10.0.0.11 : ok=6 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
10.0.0.12 : ok=6 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
10.0.0.13 : ok=6 changed=4 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

root@f94e263ae442:~#

```

- As we can see from above, the container was successfully created, with the image "sebp/elk:761".


```




sysadmin@ELK-VMM: ~
sysadmin@ELK-VMM:~$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED    STATUS    PORTS
9ced8c142b34   sebp/elk:761   "/usr/local/bin/star_..." 4 days ago Up 3 minutes 0.0.0.0:5044->5044/tcp, 0.0.0.0:5601->5601/tcp, 0.0.0.0:9200->9200/tcp, 9300/tcp
sysadmin@ELK-VMM:~$

```

Identity and Access Management

- We are going to restrict access to the ELK-VMM through the ELK Network Security Group:
- Once in the NSG for ELK, we are going to add an inbound rule that will allow access from our computer to the ELK server on port 5601:

 **Kibana** ELK-VMM-nsg ×

 Save  Discard  Delete

Source ⓘ

IP Addresses ▼

Source IP addresses/CIDR ranges * ⓘ

216.209.0.0/16

Source port ranges * ⓘ

*

Destination ⓘ

Any ▼

Service ⓘ

Custom ▼

Destination port ranges * ⓘ

5601

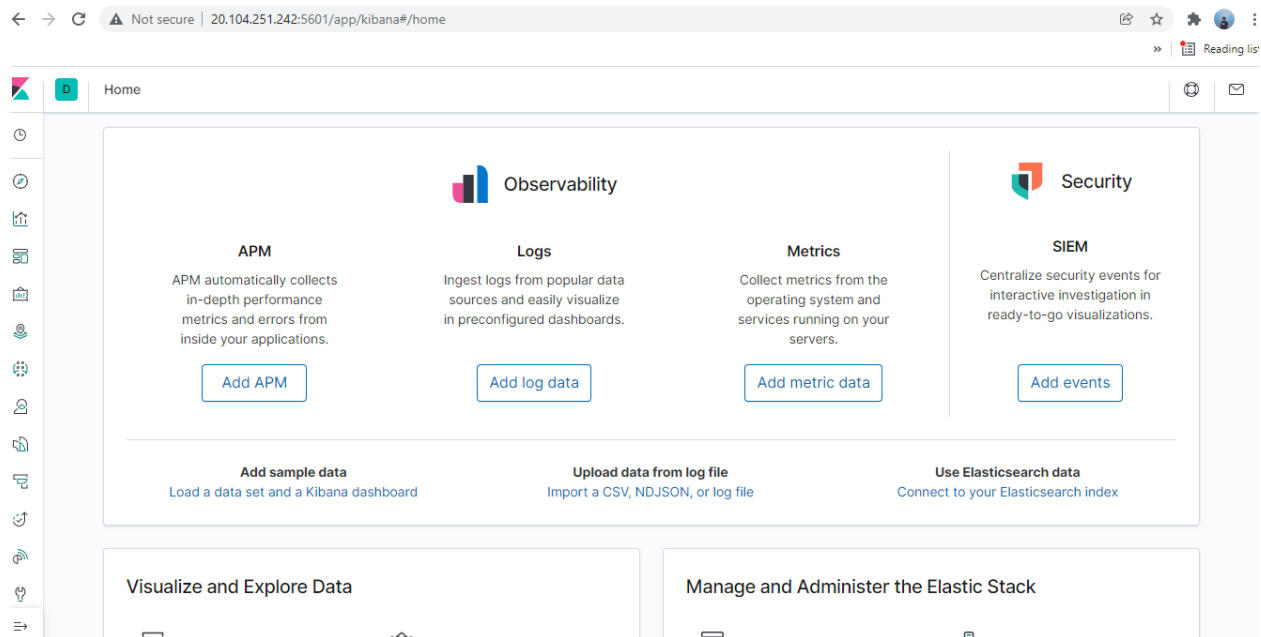
Protocol

☒ Any

☐ TCP

☐ UDP

- Finally, we will verify that we can log into the server by accessing on our browser, [ELK-public-IP]:5601/app/kibana:
 - Note: the public IP will always change every time we restart it.



7) Install Filebeat

- We can use Filebeat to collect, parse, and visualize ELK logs in a single command. This will help us better track our organizational goals.

Install Filebeat on the container

- First, we will start the virtual machines (including the ELK server) on Azure.
- Then, we will access the kibana page and make sure it works.
- We will start the container within the jump box vm:

- I jumped back to the kibana page and found the DEB page for creating a system log and will use this guide to create our filebeat playbook.

Getting Started

macOS **DEB** RPM

1 Download and install Filebeat

First time using Filebeat? See the [Getting Started Guide](#).

Copy snippet

```
curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.6.1-and64.deb
sudo dpkg -i filebeat-7.6.1-and64.deb
```

Looking for the 32-bit packages? See the [Download page](#).

Create Filebeat configuration file

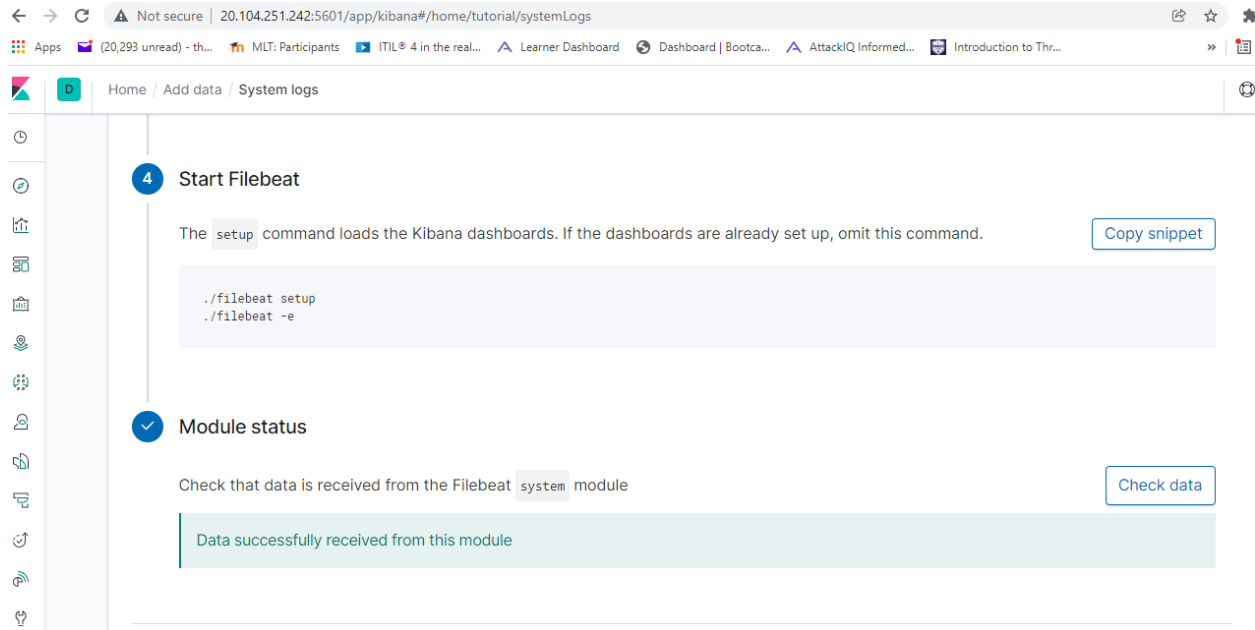
- Once the folder is created, I will download the filebeat-config.yml:
- I opened the filebeat-config.yml file and added the ELK server private IP address in two areas:

```
output.elasticsearch:
  hosts: ["10.1.0.6:9200"]
  username: "elastic"
  password: "changeme"
```

```
setup.kibana:
  host: "10.1.0.6:5601"
```

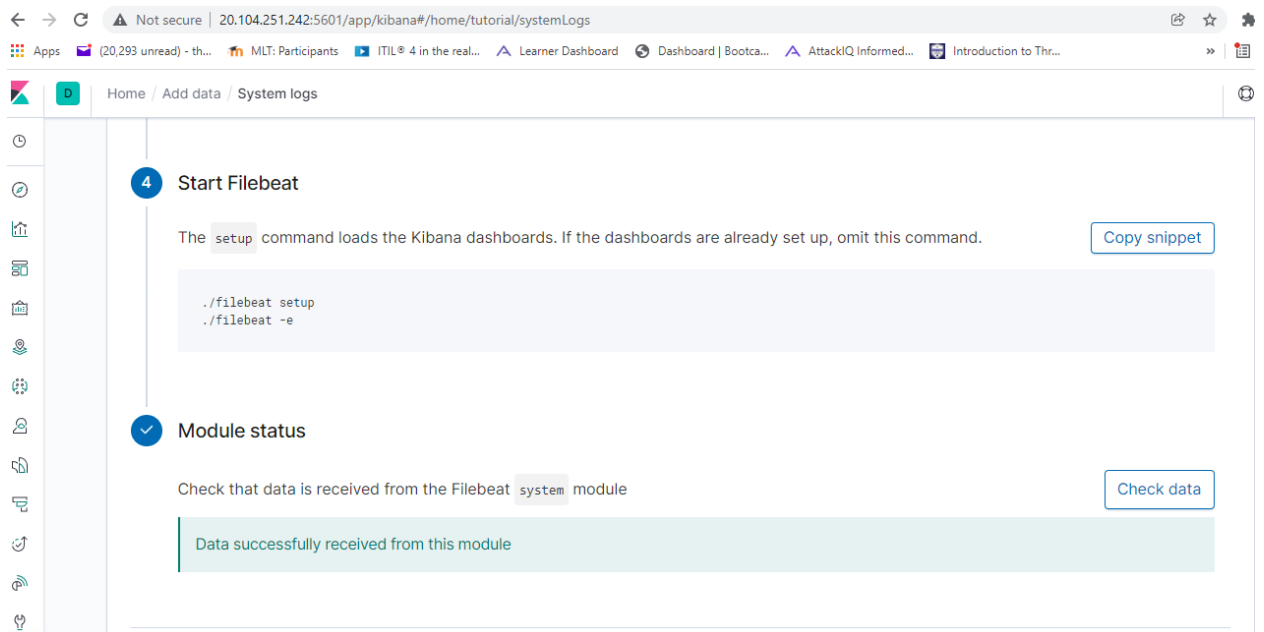
Create Filebeat Installation Playbook

- I created the filebeat-playbook.yml under /etc/ansible/files.
- Once created, using the DEB page, I added the needed commands:

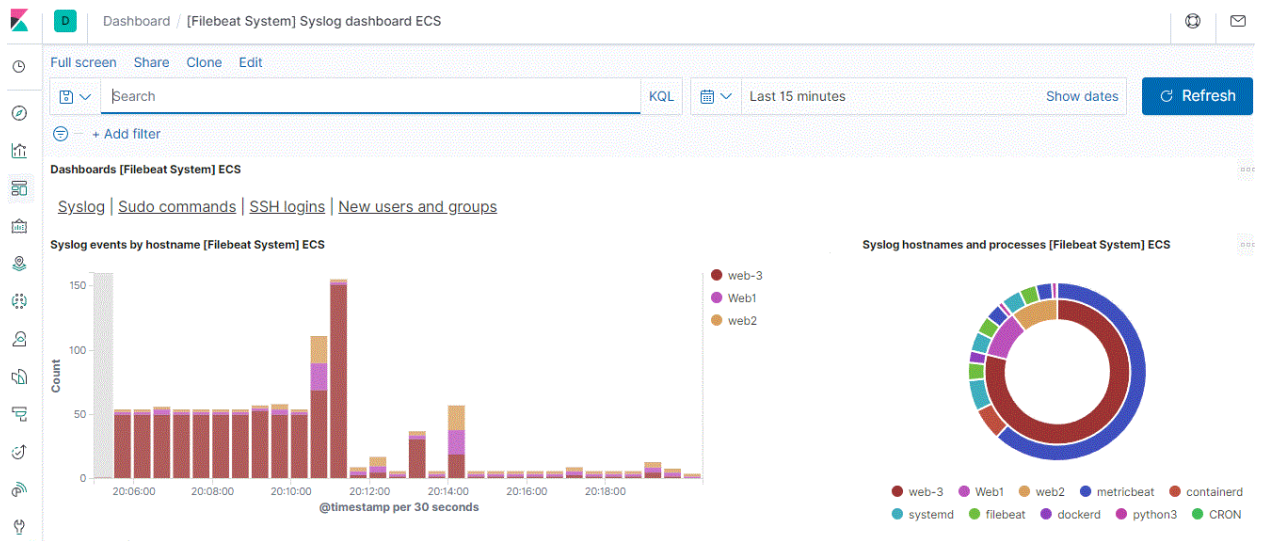


Verify Installation and Playbook

- Back on the kibana page, I will check the data on the instruction page:



- Finally, I checked the dashboard to make sure things are running:



Create a play to install Metricbeat

- Using the same steps for Filebeat, I created a config file for Metricbeat and added ELK's private IP in two areas:

```
output.elasticsearch:
  hosts: ["10.1.0.6:9200"]
  username: "elastic"
  password: "changeme"
```

```
setup.kibana:
  host: "10.1.0.6:5601"
```

- Under the files folder, I will create the metricbeat-playbook.yml:

- Once both playbooks were created, I ran the metribeat-playbook.yml:

```
[WARNING]: ansible.utils.display.initialize_locale has not been called, this may result in incorrectly calculated text widths that can cause
PLAY [Install metric beat] *****
TASK [Gathering Facts] *****
ok: [10.0.0.13]
ok: [10.0.0.12]
ok: [10.0.0.11]
TASK [Download metricbeat] *****
changed: [10.0.0.13]
changed: [10.0.0.12]
changed: [10.0.0.11]
TASK [install metricbeat] *****
changed: [10.0.0.11]
changed: [10.0.0.13]
changed: [10.0.0.12]
TASK [drop in metricbeat config] *****
changed: [10.0.0.13]
changed: [10.0.0.11]
changed: [10.0.0.12]
TASK [enable and configure docker module for metric beat] *****
changed: [10.0.0.13]
changed: [10.0.0.12]
changed: [10.0.0.11]
TASK [setup metric beat] *****
changed: [10.0.0.12]
changed: [10.0.0.13]
changed: [10.0.0.11]
TASK [start metric beat] *****
changed: [10.0.0.13]
changed: [10.0.0.12]
changed: [10.0.0.11]
TASK [enable service metricbeat on boot] *****
changed: [10.0.0.13]
changed: [10.0.0.12]
changed: [10.0.0.11]
PLAY RECAP *****
10.0.0.11      : ok=8    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.0.12      : ok=8    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.0.13      : ok=8    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

- Now that the playbook ran successfully, I will check the data on the kibana page:

✓

Module status

Check that data is received from the Metricbeat `docker` module

Check data

Data successfully received from this module

- Finally, we will check the dashboard, to make sure it works:

