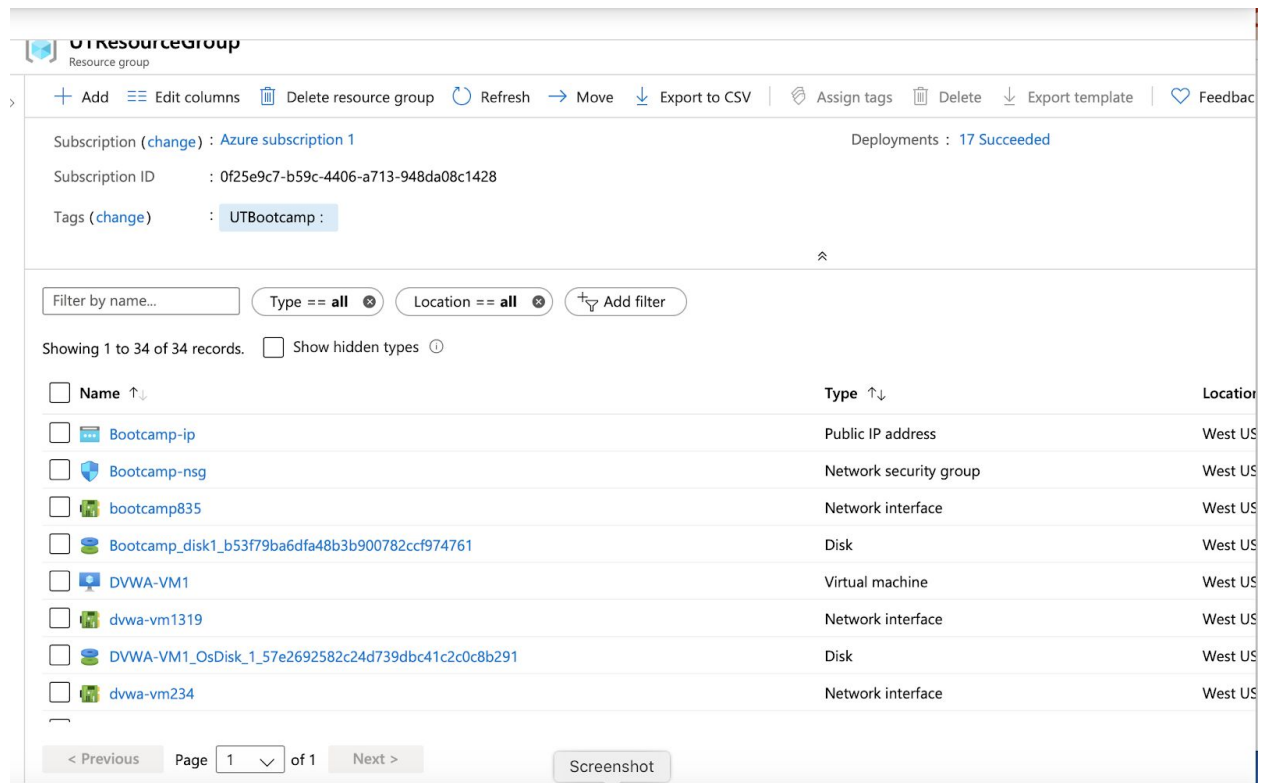


This is Project 1 of my Cybersecurity training

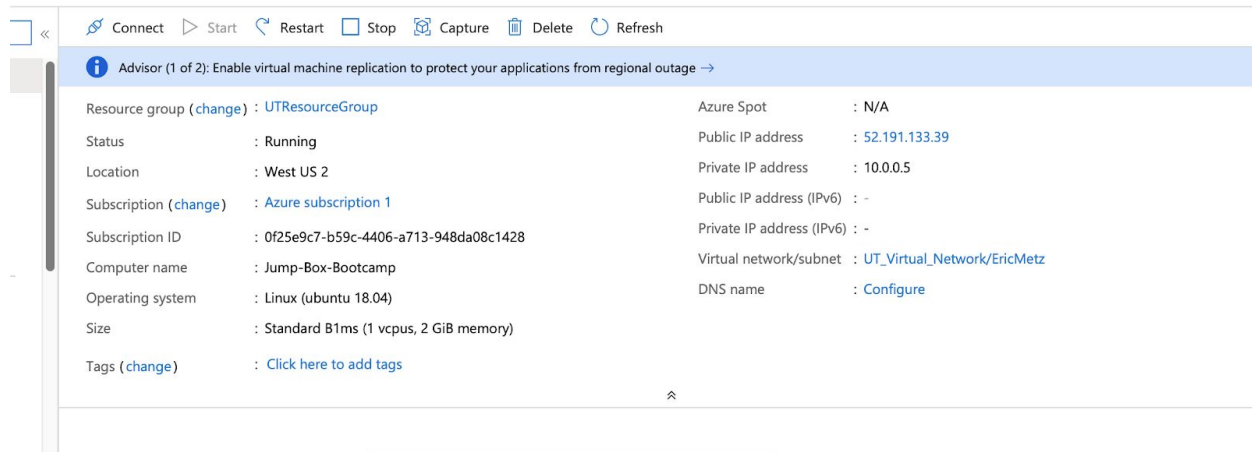
We started by creating an account in Azure and created several Virtual Machines. We then install Docker and a few containers running Ubuntu to act as DVWA web servers. Then to monitor our virtual network we created an ELK stack server so we can query the data and create metrics.

First step after getting into Azure is to create a Resource Group to put our network into.

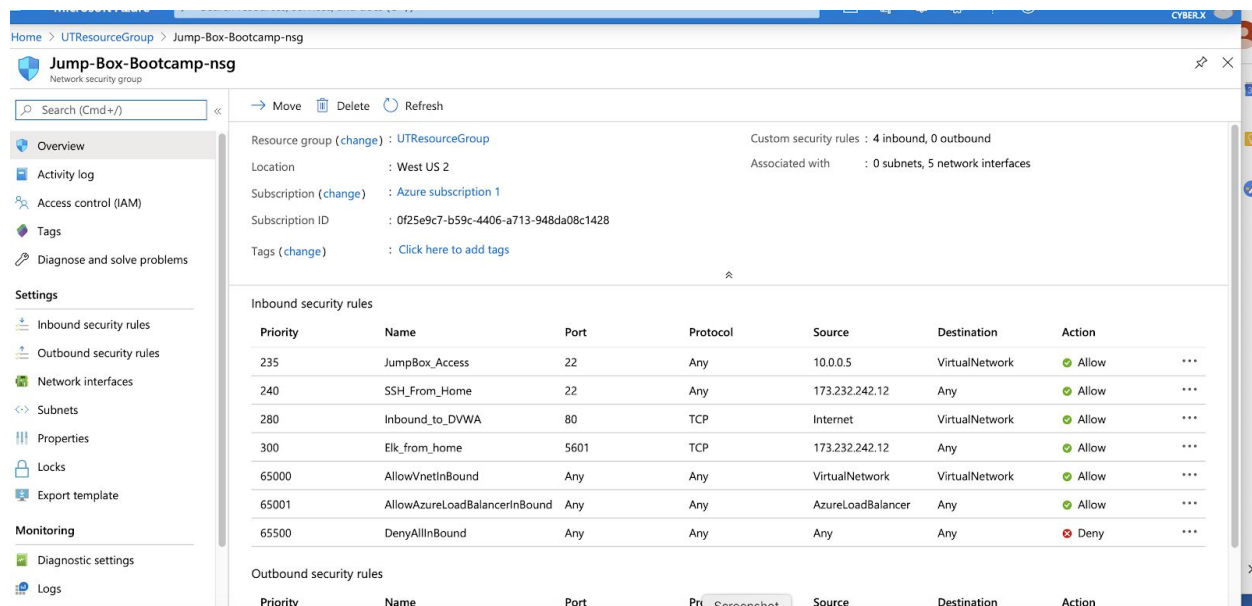


Now we can build our virtual network and add VMs, security rules, firewalls, and a load balancer.

The first VM is our jump-box. This VM has a public facing IP, where we will then jump to other VMs and containers in the network.



We need to configure our security group to allow SSH into the jump-box from only our home IP. This is the first layer of security in the network .



We then created two more VMs calling them DVWA 1 & 2.

To be able to log into both VMs independently and share data between them we had to generate and share a public RSA key between all the VMs and restrict sharing data to only within the virtual network by using the private IP addresses of the VMs.

We balance the load on our network by placing our VMs within a load balancer and assigning the VMs into availability sets so they can be balanced accordingly.

Home > Red-LB

Red-LB

Load balancer

Move

Delete

Refresh

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Frontend IP configuration

Backend pools

Health probes

Resource group (change) : [UTResourceGroup](#)

Location : West US 2

Subscription (change) : [Azure subscription 1](#)

Subscription ID : 0f25e9c7-b59c-4406-a713-948da08c1428

SKU : Basic

Tags (change) : [Click here to add tags](#)

Backend pool : RedTeamPool (2 virtual mach

Health probe : RedTeamProbe (Tcp:80)

Load balancing rule : PenTestLBR (Tcp/80)

NAT rules : 0 inbound

Public IP address : [52.148.171.184 \(Red-LB\)](#)

Configure high availability and scalability for your applications

Create highly-available and scalable applications in minutes by using built-in load balancing for cloud services and virtual ma

Load Balancer supports TCP/UDP-based protocols and protocols used for real-time voice and video messaging applications.

Home > RedTeamAS

RedTeamAS

Availability set

Delete

Refresh

Overview

Activity log

Access control (IAM)

Tags

Settings

Configuration

Virtual machines

Properties

Locks

Resource group (change) : [UTResourceGroup](#)

Location : West US 2

Subscription (change) : [Azure subscription 1](#)

Subscription ID : 0f25e9c7-b59c-4406-a713-948da08c1428

Fault domains : 2

Update domains : 5

Virtual machines : 2

Managed : Yes

Search virtual machines

Name	↑↓	Status	↑↓	Fault Domain	↑↓	Upda
DVWA-VM1		Running		0		0
DVWA-VM2		Running		1		1

Now we need to install Docker and get some containers going on these VMs. For the sake of this project we are only putting Ubuntu DVWA web servers on these machines. To automate the process we are creating YAML playbooks to install the web server on all containers at the same time.

```

RedAdmin@Jump-Box-Bootcamp:~$ sudo docker container list -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
9ec4f2f92ba0       cyberxsecurity/ansible  "/bin/bash"        7 days ago         Exited (0) 13 hours ago                  elastic
_jennings
18586282e41f       cyberxsecurity/ubuntu:bionic  "bash"             8 days ago         Exited (0) 8 days ago                  unruffl
ed_haibt
RedAdmin@Jump-Box-Bootcamp:~$ sudo docker start elastic_jenkins
Error response from daemon: No such container: elastic_jenkins
Error: failed to start containers: elastic_jenkins
RedAdmin@Jump-Box-Bootcamp:~$ sudo docker start elastic_jennings
elastic_jennings
RedAdmin@Jump-Box-Bootcamp:~$ sudo docker attach elastic_jennings
root@9ec4f2f92ba0:~# cd /etc/ansible
root@9ec4f2f92ba0:/etc/ansible#

```

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

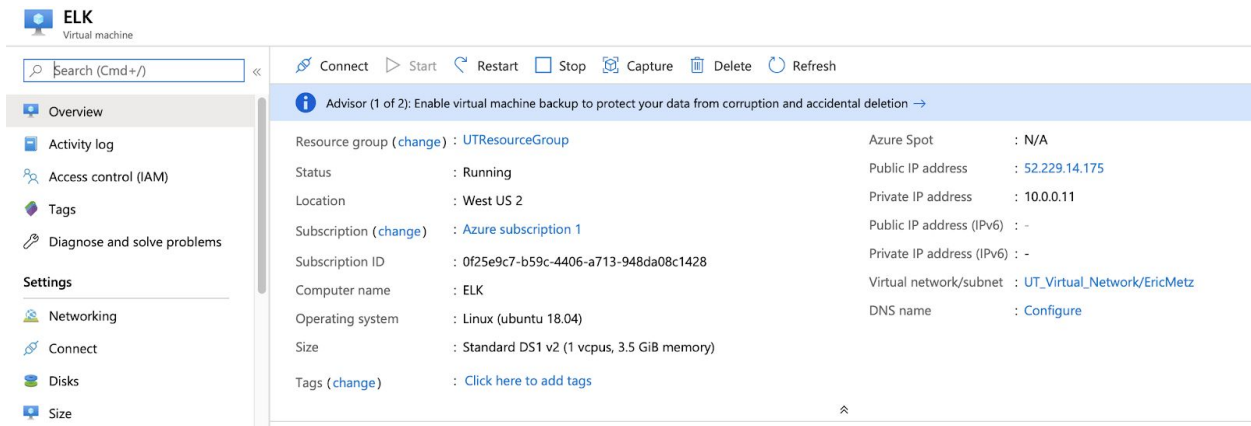
- name: Install pip
  apt:
    force_apt_get: yes
    name: python-pip
    state: present

- 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
```

After we have the containers running and have our YAML code executed we need to get an ELK stack server up and running so we can monitor the traffic and create logs and metrics.

The ELK server is much like the Jump-Box in that we need to be able to look at it publicly in a browser. It has a public IP and gets traffic from multiple ports. It resides behind our secure firewall and can only be accessed from an IP we designate.



After this server is up and running we need to install the ELK stack on it. Again with YAML code

```
GNU nano 2.9.3
install-elk.yml

---
- name: Config elk VM with Docker
  hosts: elkservers
  remote_user: ansible
  become: true
  tasks:

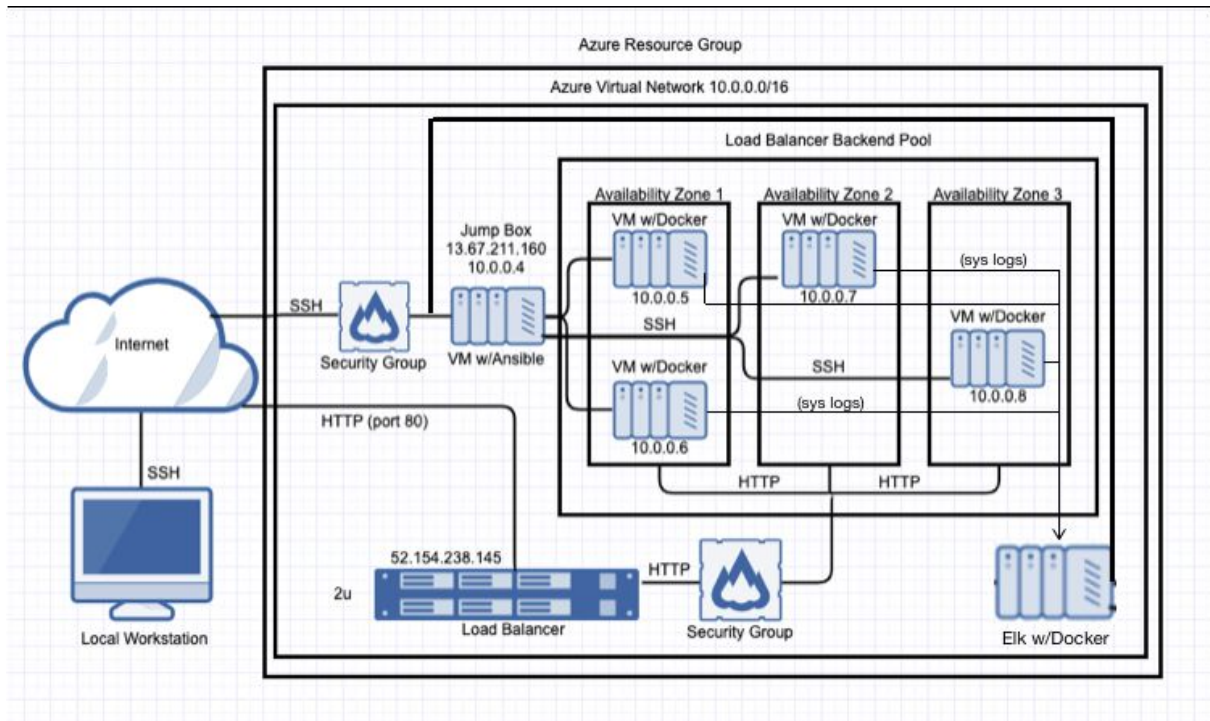
- name: Configure Elk VM with Docker
  hosts: elkservers
  remote_user: ansible
  become: true
  tasks:

- name: Install docker.io
  apt:
    force_apt_get:yes
    name: docker.io
    state: present

- name: Install pip
  apt:
    force_apt_get: yes
    name: python-pip
    state: present

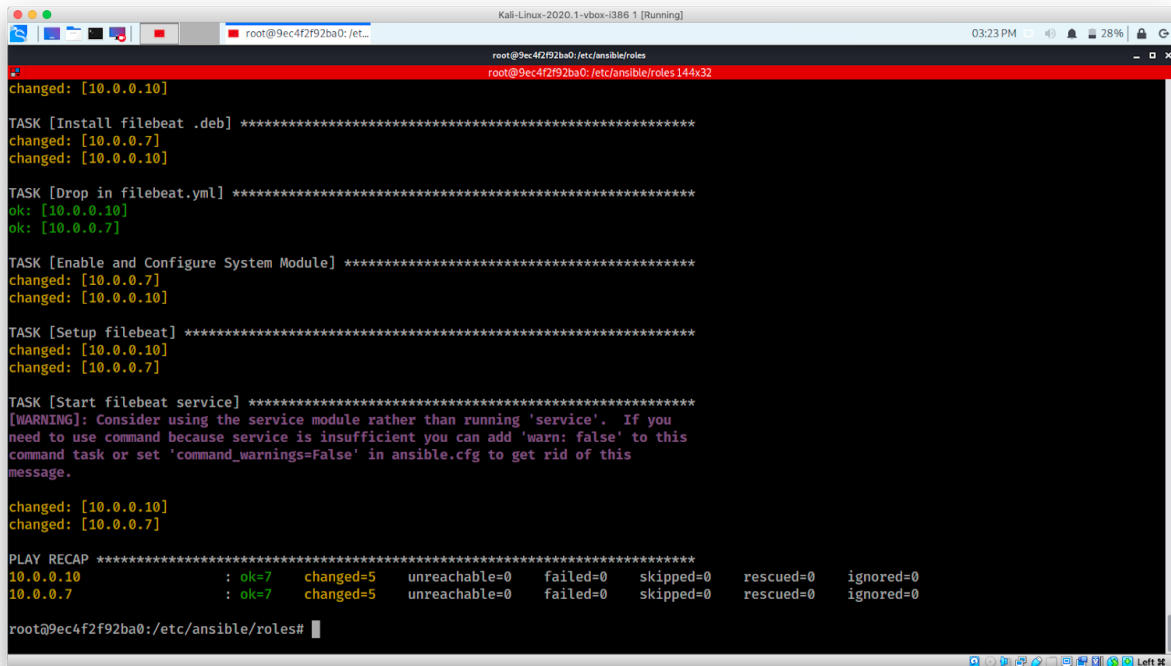
- name: Install Docker python module
  Read 43 lines
  Get Help  Write Out  Where Is  Cut Text  Justify  Cur Pos  M-U  Undo  M-A  Mark Text  M-J  To Bracket
  Exit      Read File  Replace  Uncut Text  To Spell  Go To Line  M-E  Redo  M-6  Copy Text  M-W  WhereIs Next
```

When everything is setup it should be mapped out like this.



No let's capture some log data with Filebeat.

This is open source programming so we can go to our ELK server web page and grab the config file and the YAML execution.



```
Kali-Linux-2020.1-vbox-i386 1 [Running]
root@9ec4f2f92ba0:/etc/ansible/roles
root@9ec4f2f92ba0:/etc/ansible/roles 144x32
changed: [10.0.0.10]

TASK [Install filebeat .deb] *****
changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [Drop in filebeat.yml] *****
ok: [10.0.0.10]
ok: [10.0.0.7]

TASK [Enable and Configure System Module] *****
changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [Setup filebeat] *****
changed: [10.0.0.10]
changed: [10.0.0.7]

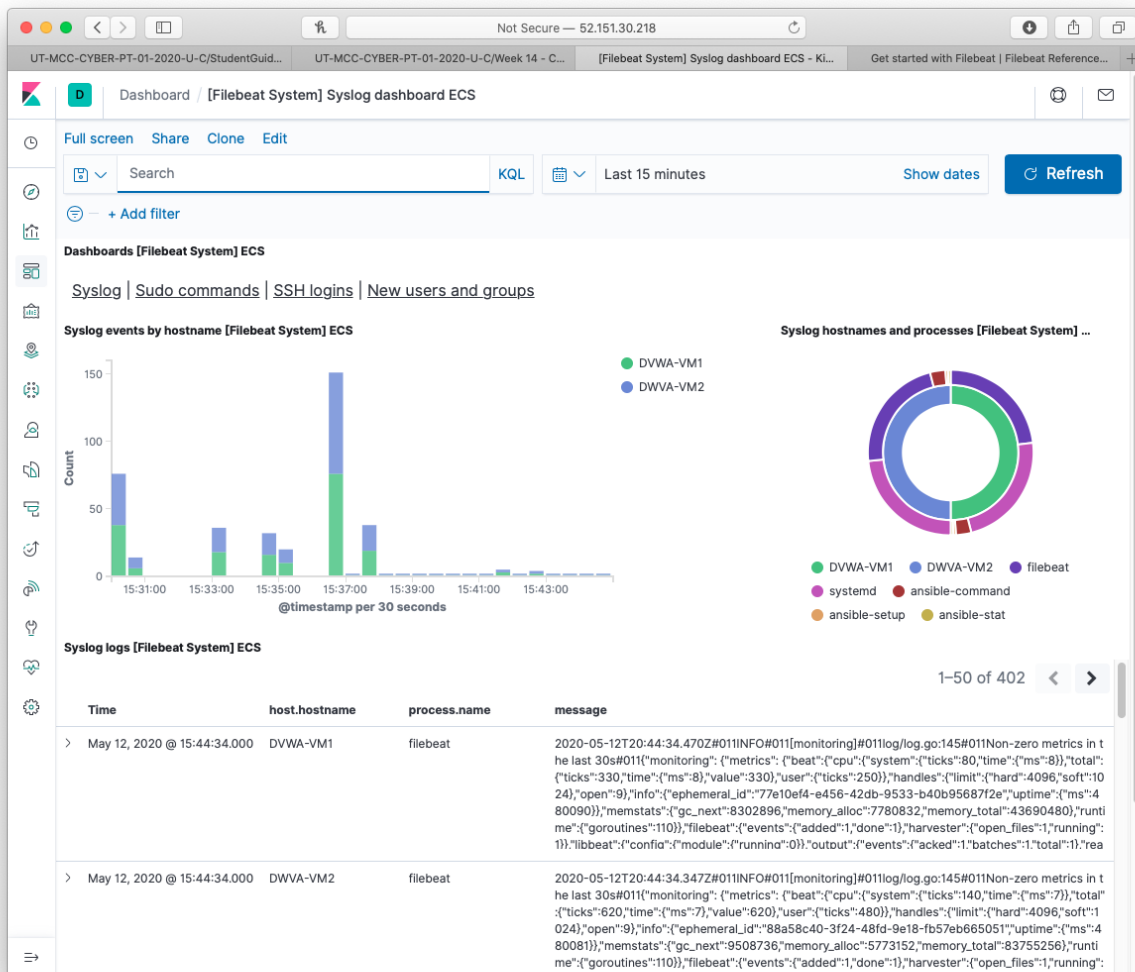
TASK [Start filebeat service] *****
[WARNING]: Consider using the service module rather than running 'service'. If you
need to use command because service is insufficient you can add 'warn: false' to this
command task or set 'command_warnings=False' in ansible.cfg to get rid of this
message.

changed: [10.0.0.10]
changed: [10.0.0.7]

PLAY RECAP *****
10.0.0.10      : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.0.7      : ok=7    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

root@9ec4f2f92ba0:/etc/ansible/roles#
```

And then check if there is log data flowing



Now to get some metrics on these logs we need to do the same process with MetricBeat.



```
Kali-Linux-2020.1-vbox-i386 1 [Running]
root@9ec4f2f92ba0:/et...
root@9ec4f2f92ba0:/etc/ansible/roles
root@9ec4f2f92ba0:/etc/ansible/roles 144x32

changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [install metricbeat] *****
changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [drop in metricbeat config] *****
changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [enable and configure docker module for metric beat] *****
changed: [10.0.0.10]
changed: [10.0.0.7]

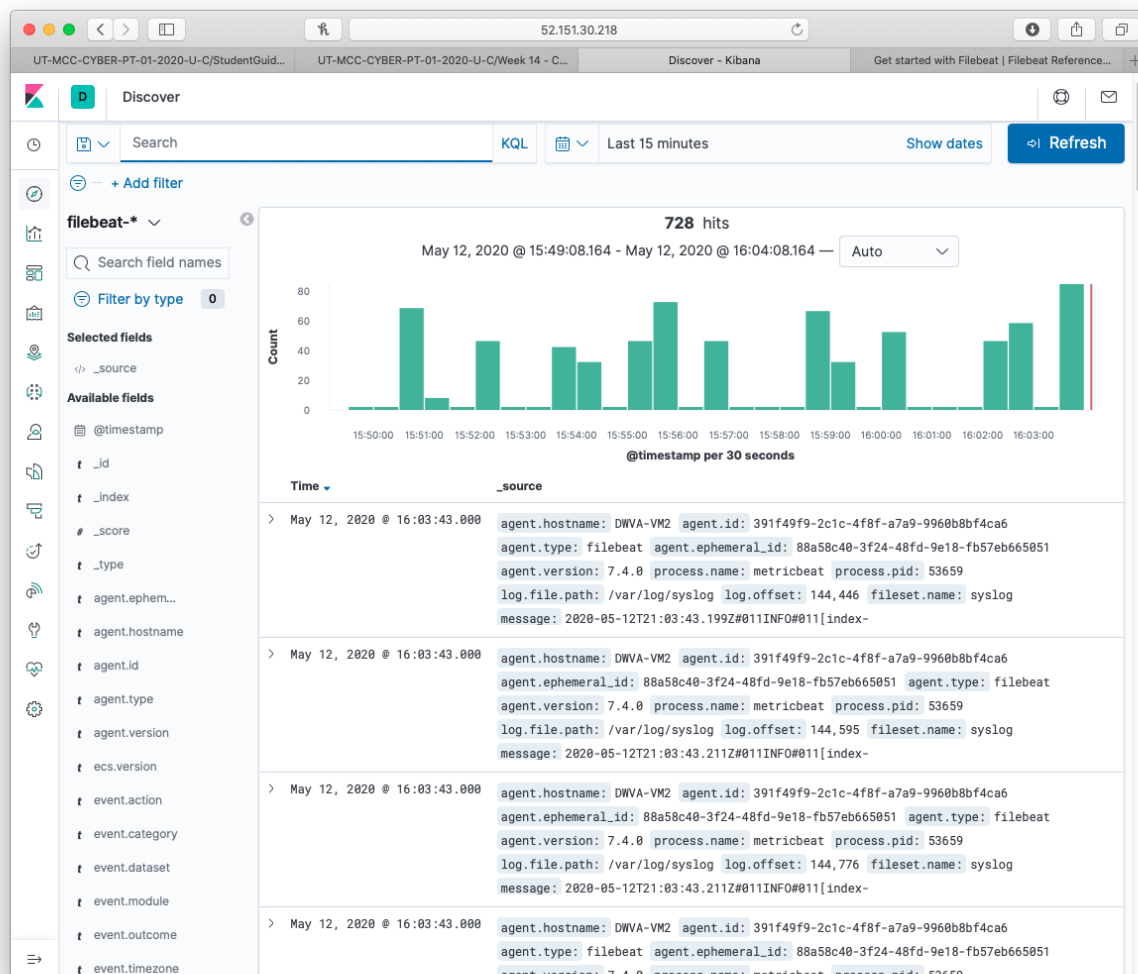
TASK [setup metric beat] *****
changed: [10.0.0.7]
changed: [10.0.0.10]

TASK [start metric beat] *****
[WARNING]: Consider using the service module rather than running 'service'. If you need to use command because service is insufficient you can
add 'warn: false' to this command task or set 'command_warnings=False' in ansible.cfg to get rid of this message.
changed: [10.0.0.7]
changed: [10.0.0.10]

PLAY RECAP *****
10.0.0.10      : ok=7    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.0.7      : ok=7    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

root@9ec4f2f92ba0:/etc/ansible/roles#
```





We now have a functioning cloud based network that is robustly secure and software installed that we can configure to monitor the website traffic