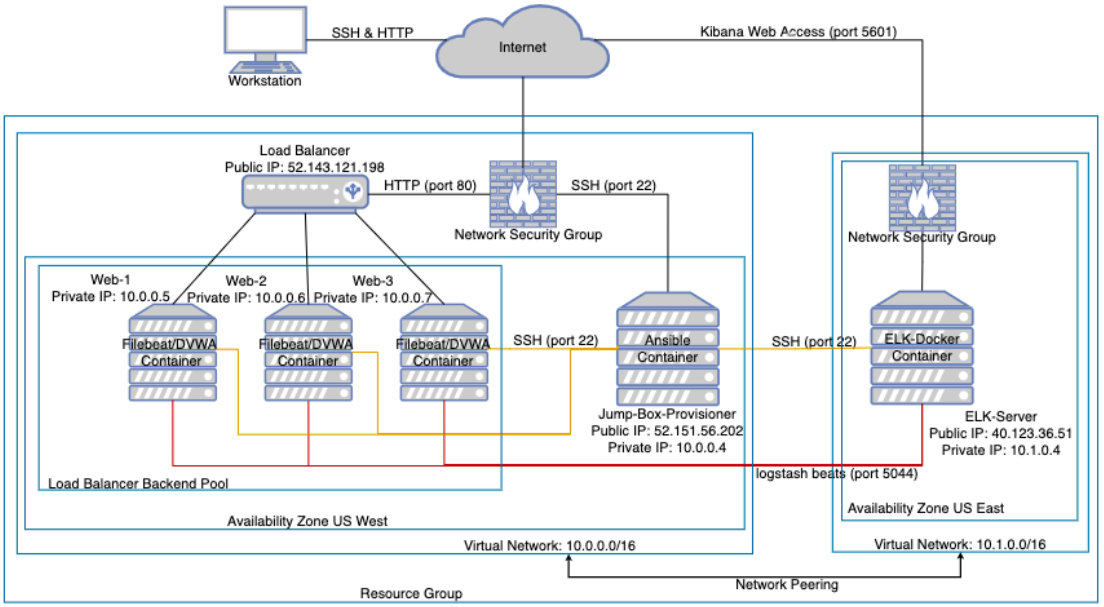
Instructions - Azure Cloud Environment

This document contains instructions to create the cloud infrastructure as depicted in the network diagram below:



1) Setting up the cloud Environment

Step 1: Create a Resource Group

- Create a resource group within a region.
- I created mine in US West 2 and named it as **Red-Team**.

Red-Team	Azure subscription 1	West US 2

Step 2: Create a Virtual Network

- Create a new Virtual Network.
- Make sure to create the Virtual Network under the previously created resource group and within the same region as the resource group.
- I named my Virtual Network as **RedTeamNet**.

RedTeamNet	Red-Team	West US 2

Step 3: Create a Network Security Group (NSG)

- Create a Network Security Group under the Virtual Network just created.
- Make sure the NSG is in the same region as everything else created thus far.
- I named my NSG as **NetSecGroup.**

NetSecGroup	Red-Team	West US 2	Azure subscription 1

2) Create the Virtual Machines

Step 1: 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 **Red-Team**.
- I named my VM as **Jump-Box-Provisioner.**
- Use an Ubuntu server with at least 1GB of memory.
- Use a public SSH key from your local computer. Use "ssh-keygen" to create a public key if you don't have one.
- Give it a username you will remember. My username is **sysadmin**.

Jump-Box-Provisioner	Azure subscription 1	Red-Team	West US 2	Stopped (deallocated)	Linux	Standard_B1s	52.151.56.202

Step 2: Network Security Group Rules

Below is an overview of all of the inbound rules for my **NetSecGroup** NSG:

Priority	Name	Port	Protocol	Source	Destination	Action
700	AllowJumpBoxAccess	22	Any	10.0.0.4	10.0.0.5,10.0.0.6,10.0.0....	Allow
750	AllowHTTP	80	Any	173.177.218.21	VirtualNetwork	Allow
1000	AllowSSH	22	Any	173.177.218.21	VirtualNetwork	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalanc...	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Step 3: Set up Docker.io on the Jump Box VM

- Start the VM from Azure portal and SSH into your Jump-Box-Provisioner VM. Run the following command: `ssh -i <path_to_private_key> <VMusername>@<VMpublicIP>`.
- Once logged in, implement the following:
  - `sudo apt install docker.io`

    - `sudo docker pull cyberxsecurity/ansible`
    - `sudo docker run -ti cybersecurity/ansible bash`
- Launch the ansible container and make sure it works.

Step 4: Create Web Server Virtual Machines

- Create 3 additional Virtual Machines that will act as web servers. I named them as **Web-1**, **Web-2**, and **Web-3**.
- Follow this criteria:
    - Allow no public IP address.
    - Connect your VMs to the previously created Virtual Networks (**RedTeamNet**) and NSG (**NetSecGroup**).
- Generate a SSH key within the Jump-Box-Provisioner VM docker/ansible container.
    - Use "ssh-keygen" to create the keys.
    - Store the public keys in all three virtual machines.
    - Use the same usernames for all three VM web servers. My username is **sysadmin**.

Web-1	Azure subscription 1	Red-Team	West US 2	Stopped (deallocated)	Linux	Standard_B1ms
Web-2	Azure subscription 1	Red-Team	West US 2	Stopped (deallocated)	Linux	Standard_B1ms
Web-3	Azure subscription 1	Red-Team	West US 2	Stopped (deallocated)	Linux	Standard_B1s

Step 5: Config and Hosts File

- `cd /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:
        - `10.0.0.5 ansible_python_interpreter=/usr/bin/python3`
        - `10.0.0.6 ansible_python_interpreter=/usr/bin/python3`
        - `10.0.0.7 ansible_python_interpreter=/usr/bin/python3`

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

3) Load Balancer

- Create a new Load Balancer in Azure. I named mine **RedTeamLB.**
- Select a static IP address.

- Select the same Resource Group and region as we did for Virtual Machines.
- Once the Load Balancer is created, add a Health Probe.
- Create a Backend Pool and add the 3 VMs (Web-1, Web-2, Web-3) to it.

☐	RedTeamLB	Red-Team	West US 2

4) Logging into Jump-Box-Provisioner

- Login to Azure and start all of the Virtual Machines.
- Open your workstation terminal and ssh into the Jump-Box-Provisioner.
  `ssh -i <path_to_private_key> <VMusername>@<VMpublicIP>`

```
Last login: Mon Mar 29 10:03:38 on ttys000
[                                ~ % ssh -i /Users/        /.ssh/Azure21_rsa sysadmin@52.151.56.202
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1043-azure x86_64)

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

 System information disabled due to load higher than 1.0

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

16 packages can be updated.
6 of these updates are security updates.
To see these additional updates run: apt list --upgradable


Last login: Thu Mar 25 01:50:25 2021 from 173.177.218.21
sysadmin@Jump-Box-Provisioner:~$
```

5) Starting Docker

- Check to see which containers you have.

```
[sysadmin@Jump-Box-Provisioner:~$ sudo docker container list -a
CONTAINER ID        IMAGE                          COMMAND             CREATED             STATUS                      PORTS               NAMES
d11e7183ba7c        cyberxsecurity/ansible:latest  "bash"              13 days ago         Exited (0) 9 seconds ago                        nice_keller
sysadmin@Jump-Box-Provisioner:~$
```

- My container name is `nice_keller.`
- To start your container use the following commands:

```
sudo docker start nice_keller
sudo docker attach nice_keller
```

```
sysadmin@Jump-Box-Provisioner:~$ sudo docker start nice_keller
nice_keller
sysadmin@Jump-Box-Provisioner:~$ sudo docker attach nice_keller
root@d11e7183ba7c:~#
```

6) Logging into Web Servers

- Once in the ansible container execute the following commands to connect to the web servers. These are web-servers private IP addresses.

```
ssh sysadmin@10.0.0.5
ssh sysadmin@10.0.0.6
ssh sysadmin@10.0.0.7
```

7) Install ELK Stack

Step 1: Create a new VNet (Virtual Network)

- Create a new Virtual Network that is in the same resource group (**Red-Team**) but in a different region. I created one in US East 2 and named it as **ProjectvNet**.
- Create a peer connection between the two Virtual Networks (**RedTeamNet** and **ProjectvNet**).
- I created a new connection from **ProjectvNet** to **RedTeamNet** and called it: **Elk-to-Red**. I created another connection from **RedTeamNet** to **ProjectvNet** and called it: **Red-to-Elk**.

ProjectvNet	Red-Team	East US 2	Azure subscription 1

Step 2: Create a new Virtual Machine

- Create a new Ubuntu VM with a 4GB minimum RAM size.
- This VM should be in the same region as the new Virtual Network.
- The IP address of this Virtual Machine will be public.
- The Virtual Network will be **ProjectvNet**, and the resource group will be the one we have for the other VMs, which is Red-Team.
- Use the public key stored in the ansible container.
- I named my Virtual Machine as: **ELKserverVM**.

ELKserverVM	Azure subscription 1	Red-Team	East US 2	Stopped (deallocated)	Linux	Standard_B2s	40.123.36.51

Once the ELK Server VM is created, SSH into the VM to make sure it works:

```
root@d11e7183ba7c:~# ssh sysadmin@10.1.0.4
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1043-azure x86_64)

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

  System information as of Mon Mar 29 16:52:03 UTC 2021

  System load:  0.1               Processes:              129
  Usage of /:   17.8% of 28.90GB  Users logged in:        0
  Memory usage: 67%               IP address for eth0:    10.1.0.4
  Swap usage:   0%                IP address for docker0: 172.17.0.1

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability

9 packages can be updated.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


*** System restart required ***
Last login: Thu Mar 25 01:51:22 2021 from 10.0.0.4
sysadmin@ELKserverVM:~$
```

Step 3: Download and Configure the Container

We will add the ELK Stack server IP address in the ansible hosts file under the [elk] group.

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

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

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

```yaml
---
- name: Configure Elk VM with Docker
  hosts: elk
  remote_user: azadmin
  become: true
  tasks:

    # Use apt module
    - name: Install docker.io
      apt:
        update_cache: yes
        name: docker.io
        state: present

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

      # Use pip module
    - name: Install Docker python module
      pip:
        name: docker
        state: present

      # Use sysctl module
    - name: Use more memory
      sysctl:
        name: vm.max_map_count
        value: "262144"
        state: present
        reload: yes

      # Use docker_container module
    - name: download and launch a docker elk container
      docker_container:
        name: elk
        image: sebp/elk:761
        state: started
        restart_policy: always
        published_ports:
          - 5601:5601
          - 9200:9200
          - 5044:5044

      # Use systemd module
    - name: Enable Docker on Boot
      systemd:
        name: docker
        enabled: yes
```

Step 4: Launch and Expose the Container

- Run the yml playbook and make sure it works.

```
root@d11e7183ba7c:/etc/ansible# ansible-playbook elk-playbook.yml
```

- SSH into the ELK server and run “docker ps”.

```
sysadmin@ELKserverVM:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                                                                                    NAMES
35dd45b18fc5        sebp/elk:761        "/usr/local/bin/star…"   18 minutes ago      Up 18 minutes       0.0.0.0:5044->5044/tcp, 0.0.0.0:5601->5601/tcp, 0.0.0.0:9200->9200/tcp, 9300/tcp   elk
sysadmin@ELKserverVM:~$
```

Step 5: Identity and Access Management

- Restrict access to the ELK VM through the ELK NSG.
- Add an inbound rule that will only allow access from our computer to the ELK server on port 5601.
- Add another security rule that will restrict all access to the ELK server, with a higher priority number.
- Finally, verify that you can log into the server via web browser at http://[ELK-public-IP]:5601/app/kibana.

8) Install Filebeat

Step 1: Install Filebeat

Create Filebeat Configuration file

- Create a directory named files in /etc/ansible inside ansible container
- Download filebeat-config.yml
- Add the ELK server IP address in filebeat-config.yml file as specified below.

```
output.elasticsearch:
  hosts: ["10.1.0.4:9200"]
  username: "elastic"
  password: "changeme"

setup.kibana:
  host: "10.1.0.4:5601"
```

Create Filebeat Installation Playbook

- Create the filebeat-playbook.yml under /etc/ansible/roles.
- Once completed, run the playbook.

```yaml
---
- name: installing and launching filebeat
  hosts: webservers
  become: yes
  tasks:

    # Use command module
  - name: download filebeat deb
    command: curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-amd64.deb

    # Use command module
  - name: install filebeat deb
    command: dpkg -i filebeat-7.4.0-amd64.deb

    # Use copy module
  - name: drop in filebeat.yml
    copy:
      src: /etc/ansible/files/filebeat-config.yml
      dest: /etc/filebeat/filebeat.yml

    # Use command module
  - name: Enable and Configure System Module
    command: filebeat modules enable system

    # Use command module
  - name: setup filebeat
    command: filebeat setup

    # Use command module
  - name: start filebeat service
    command: service filebeat start

    # Use systemd module
  - name: enable service filebeat on boot
    systemd:
      name: filebeat
      enabled: yes
```

9) Install Metricbeat

Similarly, create a config file and a .yml playbook for Metricbeat.

```
output.elasticsearch:
  hosts: ["10.1.0.4:9200"]
  username: "elastic"
  password: "changeme"

setup.kibana:
  host: "10.1.0.4:5601"
```

```yaml
---
- name: Install metric beat
  hosts: webservers
  become: true
  tasks:

    # Use command module
  - name: Download metricbeat
    command: curl -L -O https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-7.4.0-amd64.deb

    # Use command module
  - name: install metricbeat
    command: dpkg -i metricbeat-7.4.0-amd64.deb

    # Use copy module
  - name: drop in metricbeat config
    copy:
      src: /etc/ansible/files/metricbeat-config.yml
      dest: /etc/metricbeat/metricbeat.yml

    # Use command module
  - name: enable and configure docker module for metric beat
    command: metricbeat modules enable docker

    # Use command module
  - name: setup metric beat
    command: metricbeat setup

    # Use command module
  - name: start metric beat
    command: service metricbeat start

    # Use systemd module
  - name: enable service metricbeat on boot
    systemd:
      name: metricbeat
      enabled: yes
```