### **#SSH**

Secure Shell sets up an encrypted connection between two machines. Commands given on the first machine are executed on the second machine and output from the second machine is sent back to the first machine.

The result is the ability to control a remote machine using the command line while keeping all your actions private from any attacker or snooper.

#Generate a public key: ssh-keygen

Run cat ~/.ssh/id\_rsa.pub to display your id\_rsa.pub key:

Use your command line to SSH to the VM for administration. Windows users should use GitBash

The command to connect to JumpBox is ssh sysadmin@JumpBox-Public-IP

Command sudo -l: admin user has full sudo permissions without requiring a password.

#Creating a container

Sudo apt install docker.io Start by installing docker.io on your Jump box.

Run sudo systemctl status docker

Once Docker is installed, pull the container cyberxsecurity/ansible

Run docker run -ti cyberxsecurity/ansible:latest bash to start the container.

Run exit to quit

#launch a new VM from the Azure portal that could only be accessed using a new SSH key from the container running inside your jump box

Run docker images to view your image.

docker run -it cyberxsecurity/ansible /bin/bash to start your container and connect to it

Run cat .ssh/id\_rsa.pub to display your public key.

after your VM launches, test your connection using ssh from your jump box Ansible container.

#Locate the Ansible config file and hosts file.

nano /etc/ansible/hosts

Uncomment the [webservers] header line

[webservers]

## alpha.example.org

## beta.example.org

## 192.168.1.100

## 192.168.1.110

10.0.0.6 ansible\_python\_interpreter=/usr/bin/python3

10.0.0.7 ansible\_python\_interpreter=/usr/bin/python3

Open the file with nano /etc/ansible/ansible.cfg and scroll down to the remote\_user option

Uncomment the remote\_user line and replace root with your admin username using this format: remote\_user = sysadmin

#Test an Ansible connection using the appropriate Ansible command.

ansible -m ping all

results:

10.0.0.5 | SUCCESS => {

"changed": false,

"ping": "pong"

}

10.0.0.6 | SUCCESS => {

"changed": false,

"ping": "pong"

}

#Ansible playbook that installed Docker and configures a VM with the DVWA web application.

connect to the Ansible container in the box

docker container list -a

docker start [container\_name].

docker attach [container\_name]

Running your playbook should produce an output similar to the following

ansible-playbook /etc/ansible/playbookname.yml

sudo docker start your\_container\_name

sudo docker attach your\_container\_name

Create a YAML playbook file that you will use for your configuration

Nano /etc/ansible/DVWA.yml

Example: check ansible-playbook on ansibleVM-Web2.yml

To test that DVWA is running on the new VM, SSH to the web2 from your Ansible container(JumBox)

change the ansible hosts to your elkserver

#Test your Ansible configuration with the Ansible ping command.

Run ansible -m ping all

Run ansible-playbook your-playbook.yml

Run ssh ansible@10.0.0.7

Run curl localhost/setup.php in the browser.

#Elkserver commands:

sudo docker container list -a

Sudo docker container start [container name]

sudo docker container attach [container name]

to run the playbook command: root@0e52a6c7f577:/etc/ansible# ansible-playbook roles/filebeat-playbook.yml

This ELK web server runs on port 5601. Create an incoming rule for your security group that allows TCP traffic over port 5601 from your IP address.

Verify that you can load the ELK stack server from your browser at http://[elk PUBLIC IP address]:5601/app/kibana.

IP 10.1.0.4

## 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

[Elk]

10.1.0.4 ansible\_python\_interpreter=/usr/bin/python3

change the ansible hosts to your elkserver and run sudo docker ps in the command line to see if sebp/elk: 761 is installed.