**Container and Provisioner Demo: Docker and Ansible**

During our last activity, we downloaded a Docker container that runs Ansible. It has everything we need to run Ansible and configure other servers. During the next few activities, we will be using the Ansible container on our jump box to connect to another VM and make configuration changes.

In the following walkthrough, we will explore Docker more in-depth and use our Ansible container to connect to a new VM. This means we need to create an SSH key pair on the Ansible container and reset the SSH keys on our Web-VM's to use the SSH id\_rsa.pub file from our Ansible container.

* Run ssh admin@jump-box-ip

Once you are connected, there are a few Docker commands we will need in order to get the container up and running.

* If we began the with the same command we used to create our container, we would end up creating another new container.
* We want to make sure we are only running one container and that it is the same container every time we use Ansible.

**IMPORTANT NOTE** The difference between docker run and docker start is very important! docker run will create a *new* container from the container image and that new container will have default settings. None of your previous Ansible changes will be there. Because of this, docker run should only be used on the first time the Ansible container is created/started. docker start should be used *every* time thereafter, or problems with the activities will arise because of missing files on the Ansible container. Begin by connecting to your jump box via SSH.

List all the containers created on the system:

* Run docker container list -a
* Your output should resemble the following:
* root@Red-Team-Web-VM-1:/home/RedAdmin# docker container list -a
* CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
* Exited (0) 2 minutes ago hardcore\_brown

a0d78be636f7 cyberxsecurity/ansible:latest "bash" 3 days ago

* In this example, the container name is hardcore\_brown, but your container will have a different name.
* This name will be automatically created. These names are generated by randomly pairing a strong adjective with the last name of a notable computer programmer.
* Your unique container name will be used when referring to a container with Docker commands.

Type sudo docker start hardcore\_brown

* sudo docker: Uses sudo to run Docker with root.
* start: Initiates the container that follows.
* container\_name: The unique name for your container.

Run the command and note that nothing seems to have happened.

$ sudo docker start container\_name

$ hardcore\_brown

List all running containers:

* Run sudo docker ps
* You should see your container in the list.
* $ sudo docker ps
* CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

21a0f55d4e30 775349758637 "bash" 6 weeks ago Up 3 seconds hardcore\_brown

Activate a shell on your container:

* Run sudo docker attach container\_name
* Your prompt has changed and you are now connected to your Ansible Docker container.
* $ sudo docker attach hardcore\_brown

root@23b86e1d62ad:~#

During the activity you will need to create an SSH key from inside the Ansible container and use that SSH key to create a new VM. Since you have completed these steps during the previous activity, we will skim this subject here.

Run ssh-keygen to create an ssh key inside the ansible container. - Hit enter 3 times to accept all default key settings (located in /root/.ssh/id\_rsa.pub and no password)

Reset the password for one of your Web-VM's inside the azure portal and copy over your new ssh key.

After we have a VM running with the SSH key installed, we want to configure Ansible to connect to that VM.

First, we need to verify that our ssh connection is working and accept the key.

Run ssh <username>@<webVM-IP> to connect to the Web-VM and accept the key.

root@6160a9be360e:/etc/ansible# ssh sysadmin@10.0.0.5

The authenticity of host '10.0.0.5 (10.0.0.5)' can't be established.

ECDSA key fingerprint is SHA256:/jlvMP2ZRQu3fdXXPc65t8Z8fjuDMIMxhXCsaihHcJU.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '10.0.0.5' (ECDSA) to the list of known hosts.

Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 5.3.0-1020-azure x86\_64)

Run exit to close the ssh connection and return to your ansible container.

* Run ansible. Ansible should now be installed and running.
* root@9bd16493749f:~# ansible
* usage: ansible [-h] [--version] [-v] [-b] [--become-method BECOME\_METHOD]
* [--become-user BECOME\_USER] [-K] [-i INVENTORY] [--list-hosts]
* [-l SUBSET] [-P POLL\_INTERVAL] [-B SECONDS] [-o] [-t TREE] [-k]
* [--private-key PRIVATE\_KEY\_FILE] [-u REMOTE\_USER]
* [-c CONNECTION] [-T TIMEOUT]
* [--ssh-common-args SSH\_COMMON\_ARGS]
* [--sftp-extra-args SFTP\_EXTRA\_ARGS]
* [--scp-extra-args SCP\_EXTRA\_ARGS]
* [--ssh-extra-args SSH\_EXTRA\_ARGS] [-C] [--syntax-check] [-D]
* [-e EXTRA\_VARS] [--vault-id VAULT\_IDS]
* [--ask-vault-pass | --vault-password-file VAULT\_PASSWORD\_FILES]
* [-f FORKS] [-M MODULE\_PATH] [--playbook-dir BASEDIR]
* [-a MODULE\_ARGS] [-m MODULE\_NAME]
* pattern

ansible: error: too few arguments

We must make two changes to the configuration files to let Ansible make connections:

* Ansible needs to know which administrative username it should use when making SSH connections. This will be the admin username you used when you created your Web-VM's.
* Ansible needs to know the IP address of the VM you want it to connect to.

Each of these settings is located in a different file, but *all* Ansible configurations live in /etc/ansible.

* Run cd /etc/ansible and then ls to show all the files:
* root@9bd16493749f:~# cd /etc/ansible/
* root@9bd16493749f:/etc/ansible# ls
* ansible.cfg hosts roles

root@9bd16493749f:/etc/ansible#

* ansible.cfg: The file with the setting for the admin name that should be used.
* hosts: The file with all of the IP addresses that should be used.

Use Nano to open the ansible.cfg file:

* This setting is called the remote\_user. We will need to make sure that this user matches the admin username we use when we create the new VM.

Scroll down or search for the remote\_user option.

* We need to uncomment the remote\_user line and replace root with the admin username you used when creating the Web-VM's.
* The file should look like this:
* # What flags to pass to sudo
* # WARNING: leaving out the defaults might create unexpected behaviors
* #sudo\_flags = -H -S -n
* # SSH timeout
* #timeout = 10
* # default user to use for playbooks if user is not specified
* # (/usr/bin/ansible will use current user as default)
* remote\_user = YOUR\_USER\_NAME
* # logging is off by default unless this path is defined
* # if so defined, consider logrotate
* #log\_path = /var/log/ansible.log
* # default module name for /usr/bin/ansible

#module\_name = command

Exit the file.

Run nano hosts to open the hosts file with Nano.

* This file must contain the IP address for any machines that Ansible connects to.
* Machines can be grouped together under headers using brackets:
  + [webservers] or [databases] or [workstations]headers can all hold different groups of IP addresses, which Ansible can run configurations on individually or together.

For now, we will only have one web server, so we can add our IP to the provided web server header.

* Uncomment the [webservers] header line.
* Add a random IP address under the [webservers] header as an example.
* The file should resemble the following:
* # This is the default ansible 'hosts' file.
* #
* # It should live in /etc/ansible/hosts
* #
* # - Comments begin with the '#' character
* # - Blank lines are ignored
* # - Groups of hosts are delimited by [header] elements
* # - You can enter hostnames or ip addresses
* # - A hostname/ip can be a member of multiple groups
* # Ex 1: Ungrouped hosts, specify before any group headers.
* ## green.example.com
* ## blue.example.com
* ## 192.168.100.1
* ## 192.168.100.10
* # Ex 2: A collection of hosts belonging to the 'webservers' group
* [webservers]
* ## alpha.example.org
* ## beta.example.org
* ## 192.168.1.100
* ## 192.168.1.110

10.0.0.4

The way Ansible works is, it creates a python script and then runs that script on the target machine using *that machine's installation of Python*. Typically, Ansible may have issues determining which python to use on the target machine, but we will solve this by forcing ansible to use python 3 on each machine we configure.

Add the line: `ansible\_python\_interpreter=/usr/bin/python3` besides each IP address.

```bash

[webservers]

## alpha.example.org

## beta.example.org

## 192.168.1.100

## 192.168.1.110

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

```

Without this line, Ansible will still connect, but you will see this Warning:

```bash

[DEPRECATION WARNING]: Distribution Ubuntu 18.04 on host 10.0.0.5 should use

/usr/bin/python3, but is using /usr/bin/python for backward compatibility with

prior Ansible releases. A future Ansible release will default to using the

discovered platform python for this host. See https://docs.ansible.com/ansible/

2.9/reference\_appendices/interpreter\_discovery.html for more information. This

feature will be removed in version 2.12. Deprecation warnings can be disabled

by setting deprecation\_warnings=False in ansible.cfg.

```

Exit the file and then exit the VM. These two settings are all that's needed to create connections from Ansible to a virtual machine.

Ansible has a ping command that will verify these connections.

* Run ansible webservers -m ping
* We are specifying the [webservers] group that we just created.
* We *could* target any group in the hosts file or we can target *all* the machines in any group by using ansible all -m ping.
  + ansible: Used to run one-off Ansible commands.
  + -m: Specifies the Ansible module to run.
  + ping: Module that checks the connection with each of the machines given.
  + all or webservers: The group of machines you want to run the ping module on.
    - all will run it on all the hosts listed in the hosts file.
    - In this case, there is only one host, so either group will work.
* Output of a successful ping command should resemble:
* root@1f08425a2967:~# ansible all -m ping
* 10.0.0.6 | SUCCESS => {
* "changed": false,
* "ping": "pong"

}