Ansible Lab 02 Guide

Lindis Webb and Jason Smith

Syntax and usage section

A Command that is run by the user will following a $ and look like:

```
$ sudo apt-get update
```

Validate the environment

Log into the acs server and validate that ansible is installed..

```
$ ansible
Usage: ansible <host-pattern> [options]
Options:
 -a MODULE_ARGS, --args=MODULE_ARGS
                       module arguments
 --ask-become-pass     ask for privilege escalation password
 -k, --ask-pass        ask for connection password
 --ask-su-pass         ask for su password (deprecated, use become)
 -K, --ask-sudo-pass   ask for sudo password (deprecated, use become)
 --ask-vault-pass      ask for vault password
```

Check the installed version of ansible:

```
$ ansible --version
ansible 2.0.0.2
 config file = /home/vagrant/ansible.cfg
 configured module search path = Default w/o overrides
```

Create an ansible.cfg file in the root of your home directory. Point the inventory file to /home/ansible/inventory.ini

```
$ vim ~/ansible.cfg
```

```
[defaults]
inventory  = /home/ansible/inventory.ini
```

Create an inventory.ini file in the root of your home directory.

```
$ vim ~/inventory.ini
```

```
web03
```

Run an ad-hoc ansible command

```
$ ansible web03 -m ping
```

```
web03 | UNREACHABLE! => {
   "changed": false,
   "msg": "ERROR! SSH encountered an unknown error during the connection. We recommend
you re-run the command using -vvvv, which will enable SSH debugging output to help
diagnose the issue",
   "unreachable": true
}
```

The error occurs because ansible does not know what web03 is.

Rerun the ad-hoc ansible command with the verbose switched enabled.

```
$ ansible web03 -m ping -vvv
```

```
$ ping web03
```

Open the inventory.ini file and add a variable to web03 defining it's IP

```
$ vim ~/inventory.ini
```

```
web03  ansible_ssh_host=192.168.33.23
```

Run the ad-hoc ansible command, but add the user switch

```
$ ansible -u vagrant web03 -m ping
```

Run the ad-hoc ansible command, but add the password prompt switch

```
$ ansible -u vagrant web03 -m ping -k
```

This returns the following output, as discussed in the module, ansible primarily uses SSH to connect. When using Username and Password, they must be specified as a variable in an inventory file, the environment, or input from the user.

```
SSH password:
web03 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Add to web03's inventory.ini entry the following variables:

- ansible_ssh_user=vagrant
- ansible_ssh_pass=vagrant

Run the ad-hoc ansible command:

```
$ ansible web03 -m ping
```

This works without -u and -k because the username and password are now defined in the inventory.ini

CAUTION In priciple, usernames and passwords should not be store in plaintext

To avoid using username and passwords, Ansible can use ssh keys for making authorized connections. If you recall, when making the first connection, a "yes/no" dialogue was presented. Doing this for each server can become quite the task, given a thousand node environment. Run the following BUT don't accept it (don't press Y, instead exit - control+c):

```
$ ssh 192.168.33.22
```

```
The authenticity of host '192.168.33.23' can't be established
ECDSA key fingerprint is 38:76:a0:e3:d6:99:42:49:fa:42:dc:1d:0d:70:67:e2.
Are you sure you want to continue connecting (yes/no)?
```

Now run the ssh-keyscan program against the same target:

```
$ ssh-keyscan 192.168.33.22
# 192.168.33.23 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.33.23 ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBCLqK9EZT3rFH0HdhJ/PBf7KXqSLf6tZbD
Q1KYwNqk/Wfu7gyoBvmB5WOUXy7BbSLfQxw/6qnNK+uLrVVn15BDQ=
# 192.168.33.23 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.33.23 ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCuEhQd/CYFKYfBN3DgzfbRjJnuE9HSNdxKl/BXv4x9ShJzNdWb/CP+MV
VMO8b01MVCu+2R9iujJ/E+Z22+tVuo6kHyygl662fMFxkZ0ZwxZtaMgpiUjNqAPQPhP1cBeFIVV+q9hx/gqfmT
BD9pMn4lmW2gHzP/nuoYawde/qWFHBCd1VafPpOH0JinFzePoTvk5zB6p5g4I5V3iewnmhxy73ePHqzYoizaRk
IV1L64aEWsmG8dZXGMcieqpVP3Ef0OapRHMt2B9pKTpAt29TWKNUkS2wwdUQYaeZkxYpX4McvZtjiime9D0uA9
r+dg75jUTK7MWbUNjkRztWJbD+CH
vagrant@acs:~$
```

The output is the finger print that can be put into known hosts. Now run:

```
$ ssh-keyscan 192.168.33.22 >> ~/.ssh/known_hosts
```

Now running the "ssh 192.168.33.22" command will avoid the "yes/no" question and allow ansible to connect.

```
$ ssh 192.168.33.22
```

Run 'cat ~/.ssh/known_hosts' and you'll see the fingerprint for the server that was just added.

```
$ cat ~/.ssh/known_hosts
192.168.33.22 ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCq7/96EhYxORzmUdzNy5wifUSFhN9m71uf2NygIkizn2HDVtrpwX+78j
vV/6mbwjX+yd3Pi9NkTXAJ14Q3RcmzRd2GX0Y0f8O481Byi5RsnnQLCUPaXE3yZvdVDhrpSA4K6qxoj2Ek12G6
a++xTgsHLN80Uc3xse0XEiKznKaIufx3gqs3P46HJ5rH6R4BcwbBiapD7dDGJGY1oJs8dGdecPJ7D0OvwFA5dk
l2fF/x9FNtaVfjrhmnqkcLCRJrvyGmZjN5a6rwuTuZintrmpDnDcrh1bPhL/Ob0oA3RYXRE3WVYWPXcwU3n/H/
ZpCBk4n1ZcpmIFPQHfyrjtWgt7L9
192.168.33.22 ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBC6LBYfZZUvV4CGdcf2BhtqMdrphju8Nei
zDtnBlQQ4/DubgJFo/9jVOuqlv7QRzVIWIHW2fMR42w7PsM/Ls7Eg=
```