Automation with Ansible

💡 Systems Administration 1 featured a very short introduction to Automation of Linux Administration using Ansible. We will extend this in Systems Administration 2, to include more advanced topics as well as the inclusion of Windows. We will control our heterogeneous server environment using Ansible on Ubuntu.

Your vSphere Environment

Please Power down and say goodbye to web01, nmon, and docker01. These systems will be deleted by tomorrow to make room for several new VMs. You keep ad01, fw01, mgmt01 and wks01.

controller (ubuntu), ansible1(centos) and ansible2(rocky)

Networking

- controller 10.0.5.90
- ansible1 10.0.5.91
- ansible2 10.0.5.92

Linux Accounts

Create the following Linux accounts:

- On controller, create a named sudo user (your name), & another sudo user named 'deployer'
- On ansible1 and ansible2, create a sudo user named 'deployer'
- All deployer passwords should be the same

Regular Setup

- Not domain joined

Deliverable 1. A screenshot similar to the one below showing an SSH session from mgmt01 to controller and within that session a DNS lookup for controller against ad01, pinging ansible1, ansible2 and champlain.edu

```
mgmt01-hermione.granger

hermi 1 contoller-hermione: ~
hermione@contoller-hermione:~$ nslookup controller-hermione ad01-hemione 2
Server:         ad01-hemione
Address:        10.0.5.5#53

Name:   controller-hermione.hermione.local
Address: 10.0.5.90

hermione@contoller-hermione:~$ ping -cl ansible1-hermione 3
PING ansible1-hermione.hermione.local (10.0.5.91) 56(84) bytes of data.
64 bytes from ansible1-hermione.hermione.local (10.0.5.91): icmp_seq=1 ttl=64 time=0.644 ms

--- ansible1-hermione.hermione.local ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.644/0.644/0.644/0.000 ms
hermione@contoller-hermione:~$ ping -cl ansible2-hermione 4
PING ansible2-hermione.hermione.local (10.0.5.92) 56(84) bytes of data.
64 bytes from ansible2-hermione.hermione.local (10.0.5.92): icmp_seq=1 ttl=64 time=0.556 ms

--- ansible2-hermione.hermione.local ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.556/0.556/0.556/0.000 ms
hermione@contoller-hermione:~$ ping -cl champlain.edu 5
PING champlain.edu (208.115.107.132) 56(84) bytes of data.
64 bytes from 208-115-107-132-reverse.wowrack.com (208.115.107.132): icmp_seq=1 ttl=48 time=71.1 ms

--- champlain.edu ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 71.092/71.092/71.092/0.000 ms
hermione@contoller-hermione:~$
```

Deliverable 2. Within your ssh login as a named sudo user, use sudo su - deployer to switch to the deployer user. Provide a screenshot similar to the one below.

```
root@contoller-hermione: ~
hermione@contoller-hermione:~$ sudo su - deployer 1
[sudo] password for hermione:
deployer@contoller-hermione:~$ sudo -i 2
[sudo] password for deployer:
root@contoller-hermione:~#
```

Installing Ansible

```
sudo apt install ansible sshpass python3-paramiko
```

Deliverable 3. Provide a screenshot similar to the one below, indicating a successful ansible installation:

```
root@contoller-hermione: ~
root@contoller-hermione:~# ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.5 (default, Jul 28 2020, 12:59:40) [GCC 9.3.0]
root@contoller-hermione:~#
```

Create /etc/sudoers.d/sys265 on all Linux systems.

> 💣 Although it is not uncommon to update /etc/sudoers directly, it is far easier to script the addition of a file to /etc/sudoers.d. The following line allows the deployer sudo user to elevate without a password.

```
hermione@ansible1-hermione:~
  GNU nano 2.3.1                    File: /etc/sudoers.d/sys265

deployer        ALL=(ALL)       NOPASSWD: ALL
```

As the deployer user on controller, create an RSA keypair <u>with</u> a passphrase protected private key and using ssh-copy-id, add deployer@controller's public key to the deployer accounts on ansible1 and ansible2.

```
deployer@ansible2-hermione:~
deployer@controller-hermione:~$ eval $(ssh-agent)
Agent pid 19338
deployer@controller-hermione:~$ ssh-add -t 14400
Enter passphrase for /home/deployer/.ssh/id_rsa:
Identity added: /home/deployer/.ssh/id_rsa (/home/deployer/.ssh/id_rsa)
Lifetime set to 14400 seconds
deployer@controller-hermione:~$ ssh deployer@ansible2-hermione
Last login: Sun Feb 23 06:48:09 2020 from controller-hermione.hermione.local
[deployer@ansible2-hermione ~]$
```

> 💡 ssh-agent allows you to decrypt your private key, in this case for 4 hours so that you only have to type your passphrase once every four hours. The eval command will test to see if ssh-agent is running, and if not, it will run it.

Deliverable 4. Demonstrate passwordless ssh with rsa authentication to both ansible1 and ansible2 from the controller. Provide a screenshot similar to the one below that shows passwordless authentication and then passwordless elevation to root on each system.

```
deployer@contoller-hermione: ~
deployer@contoller-hermione:~$ ssh ansible1-hermione
Last login: Thu Mar  4 15:53:25 2021 from mgmt1-hermione.hermione.local
[deployer@ansible1-hermione ~]$ sudo -i
[root@ansible1-hermione ~]# exit
logout
[deployer@ansible1-hermione ~]$ exit
logout
Connection to ansible1-hermione closed.
deployer@contoller-hermione:~$ ssh ansible2-hermione
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Thu Mar  4 15:54:32 2021 from 10.0.5.10
[deployer@ansible2-hermione ~]$ sudo -i
[root@ansible2-hermione ~]# exit
logout
[deployer@ansible2-hermione ~]$ exit
logout
Connection to ansible2-hermione closed.
deployer@contoller-hermione:~$
```

First run

Setup the following directory hierarchy and inventory file on controller-yourname. The assumption is that ansible1-yourname and ansible2-yourname resolve via DNS. Run the first ansible ping.

```
deployer@contoller-hermione: ~/ansible
deployer@contoller-hermione:~$ pwd
/home/deployer
deployer@contoller-hermione:~$ mkdir -p ansible/roles
deployer@contoller-hermione:~$ cd ansible/
deployer@contoller-hermione:~/ansible$ echo ansible1-hermione >> inventory.txt
deployer@contoller-hermione:~/ansible$ echo ansible2-hermione >> inventory.txt
deployer@contoller-hermione:~/ansible$ cat inventory.txt
ansible1-hermione
ansible2-hermione
deployer@contoller-hermione:~/ansible$ ansible all -m ping -i inventory.txt
ansible1-hermione | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
ansible2-hermione | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
deployer@contoller-hermione:~/ansible$
```

Try a few ad-hoc operating system commands similar to the use of `id` below.

```
deployer@contoller-hermione:~/ansible$ ansible all -a id -i inventory.txt
ansible1-hermione | CHANGED | rc=0 >>
uid=1001(deployer) gid=1001(deployer) groups=1001(deployer),10(wheel) context=un
confined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
ansible2-hermione | CHANGED | rc=0 >>
uid=1000(deployer) gid=1000(deployer) groups=1000(deployer),10(wheel) context=un
confined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
deployer@contoller-hermione:~/ansible$
```

`Deliverable 5. Provide a screenshot of one of your executed commands (not id)`

Update your inventory to categorize your ansible2 host by type. Then test ping against just the hosts under the [webmin] tag

```
deployer@contoller-hermione:~/ansible$ cat inventory.txt
ansible1-hermione
[webmin]
ansible2-hermione
deployer@contoller-hermione:~/ansible$ ansible webmin -m ping -i inventory.txt
ansible2-hermione | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
deployer@contoller-hermione:~/ansible$
```

webmin playbook installation

Ansible galaxy is similar to docker hub and contains a rich set of Ansible scripts. We are going to use a relatively simple script that installs an administration tool on our centos server.

```
deployer@controller-hermione: ~/ansible
deployer@controller-hermione:~/ansible$ ansible-galaxy install semuadmin.webmin -p roles/
- downloading role 'webmin', owned by semuadmin
- downloading role from https://github.com/semuconsulting/ansible_webmin_role/archive/master.tar.gz
- extracting semuadmin.webmin to /home/deployer/ansible/roles/semuadmin.webmin
- semuadmin.webmin (master) was installed successfully
deployer@controller-hermione:~/ansible$ ls roles/
semuadmin.webmin
deployer@controller-hermione:~/ansible$
```

Configure the inventory so that ansible2 is in the webmin group. Create a playbook called webmin.yml within the roles directory that has the displayed content. Recall that there are 2 OS families in play for ansible here: Redhat and Rocky (syntax hint). Don't use tabs, use two spaces for indentation.

```
deployer@contoller-hermione:~/ansible$ cat roles/webmin.yml
---
- name: webmin SYS265
  hosts: webmin
  become: true
  vars:
    install_utilities: false
    firewalld_enable: true
  roles:
  - semuadmin.webmin

  tasks:
  - name: add firewall rule
    firewalld:
      port: 10000/tcp
      permanent: true
      state: enabled
```

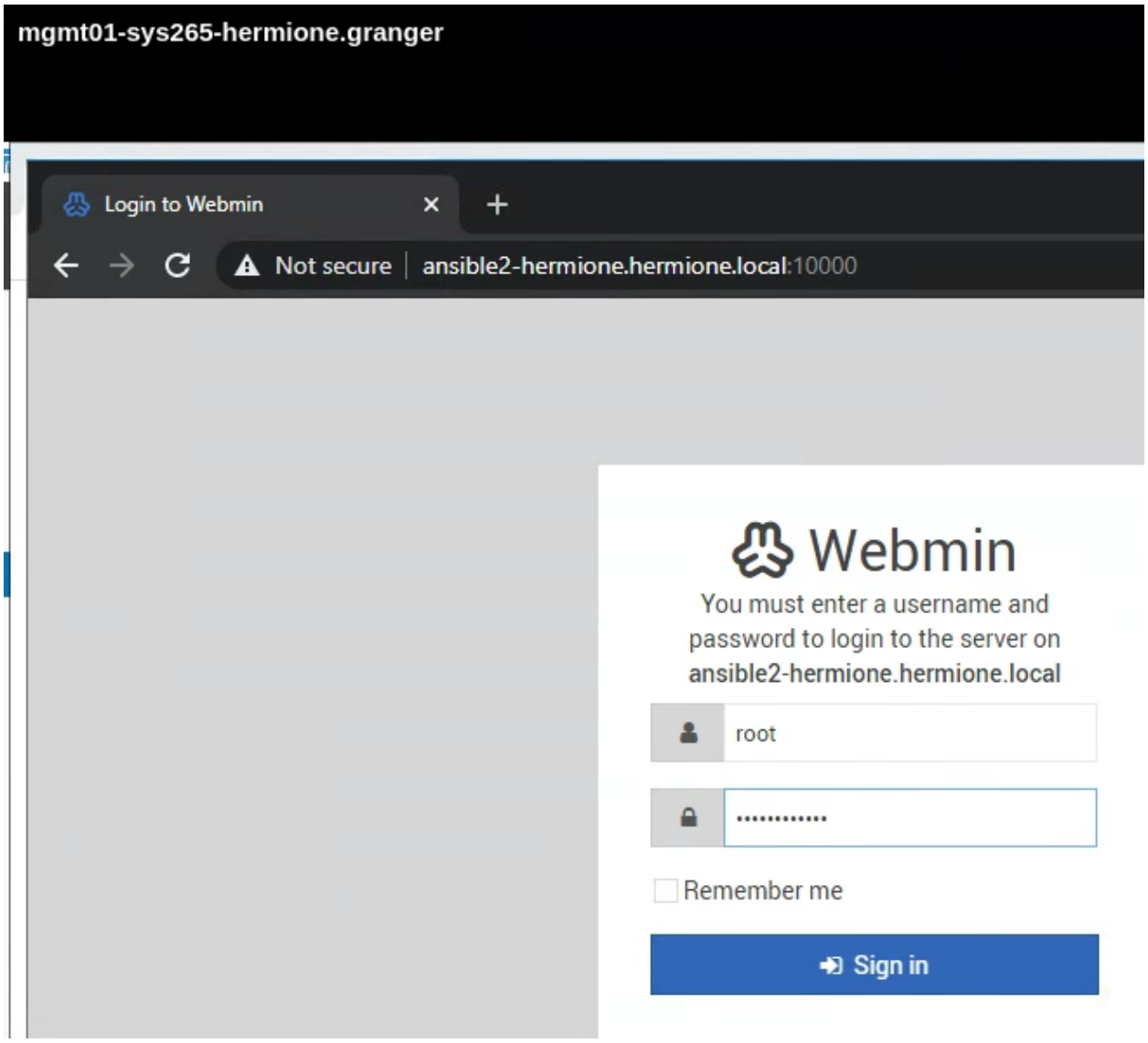
Supplementary task to deal with the role not properly dealing with the cent8 firewall

Execute the playbook (may take a moment):

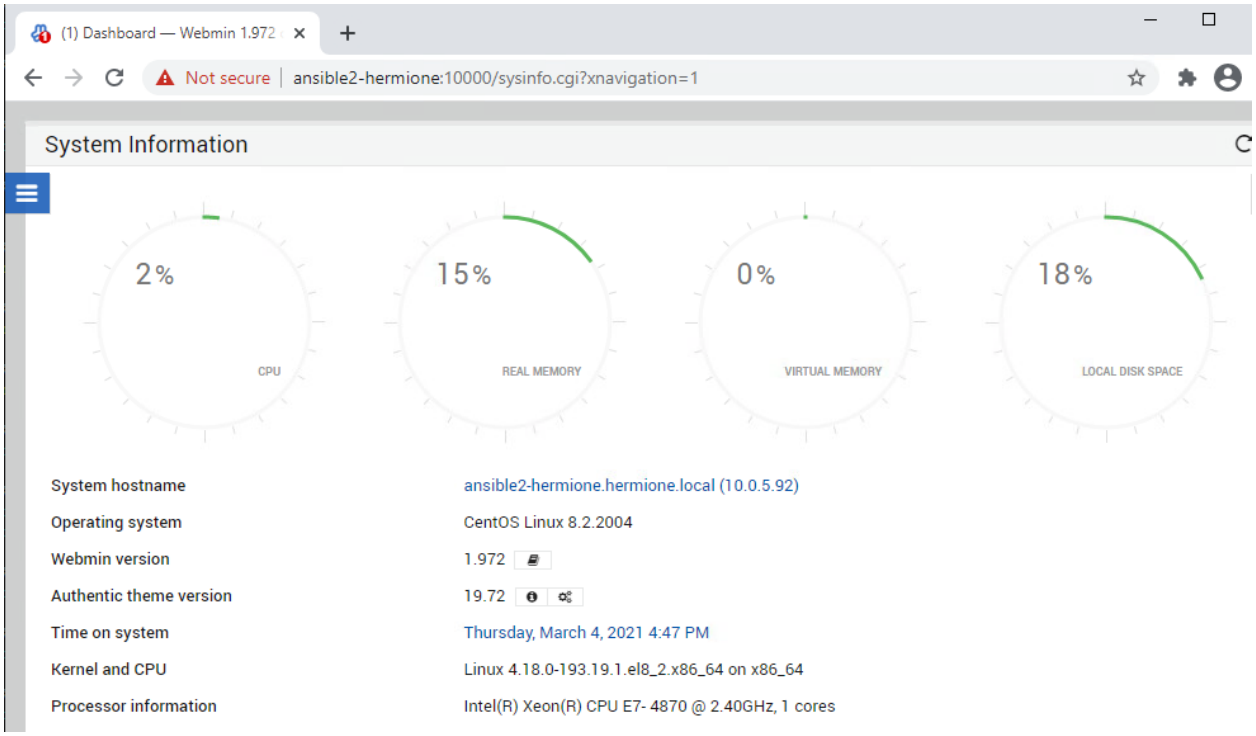
deployer@contoller-hermione: ~/ansible

```
deployer@contoller-hermione:~/ansible$ nano roles/webmin.yml
deployer@contoller-hermione:~/ansible$ ansible-playbook -i inventory.txt roles/webmin.yml

PLAY [webmin SYS265] ***********************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [ansible2-hermione]

TASK [semuadmin.webmin : Install firewalld service template.] ******************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Reload firewalld to register new service.] ************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Enable firewalld service.] ****************************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Add yum repository and gpg key for Redhat platforms.] ***********
ok: [ansible2-hermione]

TASK [semuadmin.webmin : Add a gpg key for Debian platforms.] ******************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Add apt repository for Debian platforms.] *************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Update apt cache for Debian platforms.] ***************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Install https transport for Debian platforms.] ********************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Install Webmin.] ***************************************************
ok: [ansible2-hermione]

TASK [semuadmin.webmin : Install supporting packages if required.] *************************
skipping: [ansible2-hermione]

TASK [semuadmin.webmin : Configure Webmin systemd service.] ********************************
changed: [ansible2-hermione]

TASK [semuadmin.webmin : Stop running instance before restarting under systemd.] *********
changed: [ansible2-hermione]

TASK [semuadmin.webmin : Enable webmin as systemd service.] ********************************
changed: [ansible2-hermione]

TASK [semuadmin.webmin : Reboot machine.] ***************************************************
changed: [ansible2-hermione]
```

Login to webmin as root@ansible2

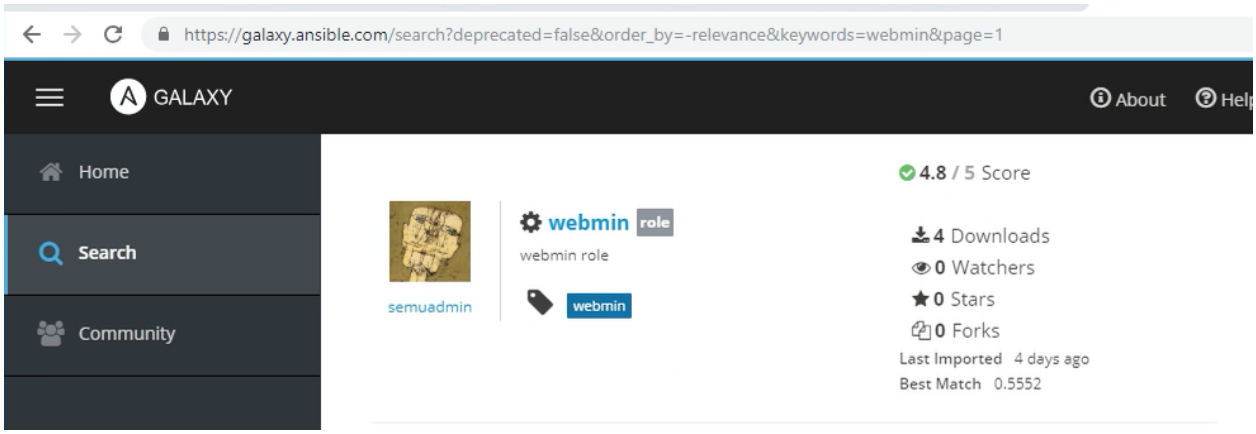


Deliverable 6. Provide a screenshot that shows some aspect of Webmin's logged-in interface like the one shown below:



Ansible Galaxy

Head over to galaxy.ansible.com and spend some time looking for roles that are built for CentOS, Redhat or EL version 7.



Deliverable 7: Deploy a different role to ansible1. Provide a screenshot of your successful playbook execution

```
Deliverable 8:  Provide a screenshot of your new service
functionality from a remote client perspective.
```

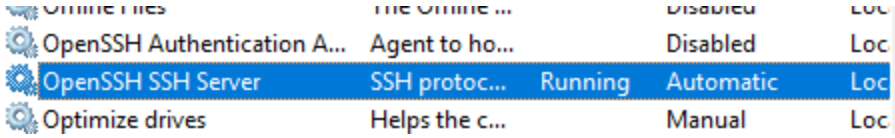
Windows Automation

> 💡 In Systems Administration 1, we explored the basics of Linux automation with Ansible. We will now see how Windows Administration can be achieved using the same framework.

Preparing MGMT01 for Ansible

> 💡 Beginning in Spring of 2021, we will start using SSH services on Windows Systems in lieu of WinRM, so these instructions may be different from ones you have seen before. SSH has been pre-installed on MGMT01 and on WKS01 but is not typically default.

Make sure OpenSSH is running on mgmt01



If for some reason, OpenSSH is not installed, one would install it in the following manner from an administrative powershell prompt:

```
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
Start-Service sshd
Set-Service -Name sshd -StartupType 'Automatic'
```

Set Powershell to be the Default Shell for SSH

If you get a normal command prompt when logging in over SSH, Run the following 2 commands to change the ssh shell to powershell (*Could Copy/Paste this, instead of typing)

```
Set-ItemProperty "HKLM:\Software\Microsoft\Powershell\1\ShellIds" -Name
ConsolePrompting -Value $true
New-ItemProperty -Path HKLM:\SOFTWARE\OpenSSH -Name DefaultShell -Value
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -PropertyType
String -Force
```

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Set-ItemProperty "HKLM:\Software\Microsoft\Powershell\1\ShellIds" -Name ConsolePrompting -Value $true
PS C:\Windows\system32> New-ItemProperty -Path HKLM:\SOFTWARE\OpenSSH -Name DefaultShell -Value "C:\Windows\System32\WindowsPowerShell\v1.0\p
owershell.exe" -PropertyType String -Force


DefaultShell : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
PSPath       : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\OpenSSH
PSParentPath : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE
PSChildName  : OpenSSH
PSDrive      : HKLM
PSProvider   : Microsoft.PowerShell.Core\Registry



PS C:\Windows\system32>
```

SSH into mgmt01

Deliverable 9. Provide a screenshot that shows a successful ssh login to a powershell prompt from controller to mgmt01 similar to the one below.

```
Administrator: c:\windows\system32\windowspowershell\v1.0\powershell.exe
deployer@contoller-hermione:~/ansible$ ssh hermione.granger-adm@hermione.local@mgmt1-hermione
The authenticity of host 'mgmt1-hermione (10.0.5.10)' can't be established.
ECDSA key fingerprint is SHA256:Or5HMMrAHT6mbs[illegible]5ZPsuFVZc4Y[illegible]6kw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'mgmt1-hermione,10.0.5.10' (ECDSA) to the list of known hosts.
hermione.granger-adm@hermione.local@mgmt1-hermione's password:
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\hermione.granger-adm>
```

Username Domain Host

Update your inventory file to add a new group called windows with mgmt01-yourname as the host in that group. Also include the variables associated with that group [windows:vars].

```
deployer@contoller-hermione:~/ansible$ cat inventory.txt
ansible1-hermione
[webmin]
ansible2-hermione
[windows]
mgmt1-hermione
[windows:vars]
ansible_shell_type=powershell
deployer@contoller-hermione:~/ansible$
```

1 2 3

Deliverable 10. Provide a screenshot similar to the one below that shows a successful win_ping from controller to mgmt01.

```
deployer@contoller-hermione: ~/ansible
deployer@contoller-hermione:~/ansible$ ansible windows -i inventory.txt -m win_p
ing -u hermione.granger-adm@hermione.local --ask-pass
SSH password:
mgmt1-hermione | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
deployer@contoller-hermione:~/ansible$
```

Add wks1 to your inventory under the windows category and rerun the win_ping. You will likely get the following common error:

```
deployer@contoller-hermione:~/ansible$ ansible windows -i inventory.txt -m win_p
ing -u hermione.granger-adm@hermione.local --ask-pass
SSH password:
wks1-hermione | FAILED! => {
    "msg": "Using a SSH password instead of a key is not possible because Host K
ey checking is enabled and sshpass does not support this.  Please add this host'
s fingerprint to your known_hosts file to manage this host."
}
mgmt1-hermione | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
deployer@contoller-hermione:~/ansible$
```

You can fix this in one of two ways. The first would be to ssh into wks1 first and accept the key. The second would be to ignore unknown hosts and you would do so by adding the following file to the directory in which you are running your ansible commands:

```
deployer@contoller-hermione: ~/ansible
deployer@contoller-hermione:~/ansible$ pwd
/home/deployer/ansible
deployer@contoller-hermione:~/ansible$ cat ansible.cfg
[defaults]
host_key_checking = false
deployer@contoller-hermione:~/ansible$
```

Deliverable 11. Rerun the playbook with successful pings on wks1 and mgmt1 similar to the one below

```
deployer@contoller-hermione:~/ansible$ ansible windows -i inventory.txt -m win_p
ing -u hermione.granger-adm@hermione.local --ask-pass
SSH password:
mgmt1-hermione | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
wks1-hermione | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Software deployment using win_chocolatey

> 💣Bear in mind, the public chocolatey servers rate limit connections. If you get a failure, take a break, go for a 5 mile run, come back and try again. Don't just repeat over and over, as you will make them mad.

Construct a new playbook within the roles directory called windows_software.yml. This is a simple playbook that uses built-in ansible functionality as opposed to a downloaded role. The list of tasks below will use a module called win_chocolatey which is a package manager for Windows similar to apt-get or yum that is becoming more popular in enterprises.

Deliverable 12. Provide a screenshot showing the successful playbook run and software Installation

The Playbook

```
deployer@contoller-hermione:~/ansible$ cat roles/windows_software.yml
---
- name: install windows applications
  hosts: windows
  tasks:
    - name: Install Firefox and 7zip
      win_chocolatey:
        name:
        - firefox
        - 7zip
        state: present
deployer@contoller-hermione:~/ansible$
```

Installation

deployer@contoller-hermione: ~/ansible

```
deployer@contoller-hermione:~/ansible$ nano roles/windows_software.yml
deployer@contoller-hermione:~/ansible$ ansible-playbook -i inventory.txt roles/w
indows_software.yml -u hermione.granger-adm@hermione.local --ask-pass
SSH password:

PLAY [install windows applications] ******************************************

TASK [Gathering Facts] *******************************************************
ok: [mgmt1-hermione]
ok: [wks1-hermione]

TASK [Install Firefox and 7zip] **********************************************
[WARNING]: Chocolatey was missing from this system, so it was installed during
this task run.
changed: [wks1-hermione]
changed: [mgmt1-hermione]

PLAY RECAP *******************************************************************
mgmt1-hermione             : ok=2    changed=1    unreachable=0    failed=0    ski
nored=0
wks1-hermione              : ok=2    changed=1    unreachable=0    failed=0    ski
nored=0

deployer@contoller-hermione:~/ansible$
```

See if you can figure out how to add the Notepad++ for windows package to wks1 and mgmt01. Rerun your playbook.

Deliverable 13. Provide a screenshot from an ssh session to mgmt01 that displays installed packages similar to the one below, notepad++ should be there.

Administrator: c:\windows\system32\windowspowershell\v1.0\powershell.exe

```
PS C:\Users\hermione.granger-adm> C:\ProgramData\chocolatey\bin\choco.exe list --local-only
Chocolatey v0.10.15
7zip 19.0
7zip.install 19.0
chocolatey 0.10.15
chocolatey-core.extension 1.3.5.1
Firefox 86.0
notepadplusplus 7.9.3
notepadplusplus.install 7.9.3
7 packages installed.
PS C:\Users\hermione.granger-adm>
```

Deliverable 14. Link to your wiki. You should clearly document the commands used to install ansible on your controller, prepare linux and windows hosts for automation, as well as upload and link your various ansible specific configuration files and playbooks used in the course of this lab.