https://docs.ansible.com/

1. Installation

1. Install Ansible on Ubuntu 14.04

Step 1	Login as root user & Make sure your Ubuntu machine is up to date with latest packages.	
2	Install the software-properties-common package	**apt-get install software-properties-common**
3	Add the Ansible repository to your system	**apt-add-repository ppa:ansible/ansible**
4	Install Ansible	**apt-get update** **apt-get install ansible** The configuration files of Ansible are stored in **/etc/ansible/** Here you can see three things in the folder itself. **ansible.cfg -** This is the configuration file of Ansible, eg: you can change the ssh port of Ansible in this file **hosts** - Store all the machines hostname and IP address and you can specify the ssh port over here. **Roles** - The powerfull thing of ansible is stored in this file. It is Playbooks
5	Generate SSH key	**ssh-keygen**
6	Copy SSH key to node machine (client machine) and connect	**ssh-copy-id -i root@<ip address of remote host/node>** **Eg:-** ssh-copy-id -i root@192.168.43.226
7	Edit Host Inventory (Add IP address of remote host)	Edit Host Inventory (Add IP address of remote host) Go to host inventory file **vi /etc/ansible/hosts** Create a group (clients) and add IP address of remote host --- **[clients]** **192.168.43.226** **---**
8	Test Connection	**root@ansible-server:/etc/ansible# ansible -m ping 'clients'** 192.168.43.226 | SUCCESS => { "changed": false,

		"ping": "pong" } If you want to ping all the host in your host list you can simply use the command below: **root@ansible-server:/etc/ansible# ansible -m ping all**
9	Create a Playbook for ping	Go to below directory **/etc/ansible/roles** Creat Playbook file **ping.yml** **root@ansible-server:/etc/ansible/roles# vi ping.yml** --- **- hosts: clients** **tasks:** **- name: ping** **action: ping** Run Ansible PlayBook **“ping.yml”** **root@ansible-server:/# ansible-playbook etc/ansible/roles/ping.yml** **PLAY [clients] *** **TASK [Gathering Facts] ** **ok: [192.168.43.226]** **TASK [ping] ** **ok: [192.168.43.226]** **PLAY RECAP *** **192.168.43.226 : ok=2 changed=0 unreachable=0 failed=0**

2. Install Ansible on CentOS 7.5

Step 1	Login as root user and you need to ensure that all the packages are up to dated	**yum -y install updates**
2	Install the extra EPEL repositories from dl.fedoraproject.org	1. The command is as follows to download epel release for CentOS and RHEL 7.x using wget command: **wget https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm** 2. To install epel-release-7-5.noarch.rpm, type: **yum install epel-release-latest-7.noarch.rpm** **Note: -** Install Extra Packages for Enterprise Linux repository configuration (recommended) Just type the following yum command on a CentOS 7 or RHEL 7: **yum install epel-release**
3	Install Ansible	**yum install ansible** Check Ansible version **ansible --version**
4	Generate SSH key	**ssh-keygen** ### Save the file in default location (/root/.ssh/id_rsa), Just hit enter Enter file in which to save the key (/root/.ssh/id_rsa): Hit Enter /root/.ssh/id_rsa already exists. Overwrite (y/n)? y Enter passphrase (empty for no passphrase): <Empty> Enter same passphrase again : <Empty> ###
5	Copy SSH key to node machine and connect	**ssh-copy-id -i root@<ip address of remote host/node>** **Eg:-** ssh-copy-id -i root@192.168.43.226

6	Edit Host Inventory (Add ip address of remote host)	Edit Host Inventory (Add ip address of remote host) Go to host inventory file **vi /etc/ansible/hosts** ### Create a group (web Servers) ### **[web-servers]** **192.168.43.226**

7	Test Connection	**root@ansible-server:/etc/ansible# ansible -m ping 'web-servers'**
		192.168.43.226 | SUCCESS => {
		"changed": false,
		"ping": "pong"
		}
		
		If you want to ping all the host in your host list you can simply use the command below:
		root@ansible-server:/etc/ansible# ansible -m ping all

3. Setting up a Windows host

Step 1	Requirements: PowerShell version 3.0 and .NET Framework 4.0	Ansible requires PowerShell version 3.0 or newer and .NET Framework 4.0 or newer to function on older operating systems like Server 2008 and Windows 7. The base image does not meet this requirement. You can use the **Upgrade-PowerShell.ps1 (https://github.com/jborean93/ansible-windows/blob/master/scripts/Upgrade-PowerShell.ps1)** script to update these. Set execution policy unrestricted before running PS script:- **PS C:\Windows\system32> Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Force**
2	Setting up WinRM	**WinRM Memory Hotfix:** When running on **PowerShell v3.0**, there is a bug with the WinRM service that limits the amount of memory available to WinRM. Without this hotfix installed, Ansible will fail to execute certain commands on the Windows host. These hotfixes should installed as part of the system bootstapping or imaging process. The script **Install-WMF3Hotfix.ps1**

		(https://github.com/jborean93/ansible-windows/blob/master/scripts/Install-WMF3Hotfix.ps1) can be used to install the hotfix on affected hosts. **WinRM Setup:** Once **Powershell** has been upgraded to at least **version 3.0**, the final step is for the **WinRM** service to be configured so that Ansible can connect to it. There are two main components of the WinRM service that governs how Ansible can interface with the Windows host: the listener and the service configuration settings. Details about each component can be read below, but the script **ConfigureRemotingForAnsible.ps1 (https://github.com/ansible/ansible/blob/devel/examples/scripts/ConfigureRemotingForAnsible.ps1)** can be used to set up the basics. This script sets up both HTTP and HTTPS listeners with a self-signed certificate and enables the Basic authentication option on the service.
3	Install **Pywinrm** on **Ansible Server (Ubuntu 14.04)**	Make sure you install the pywinrm-related library on the machine that Ansible is installed on. The simplest method is to run **pip install pywinrm** in your Terminal. **Install PIP (On Ubuntu 14.04)** root@ansible-server:/# **apt-get install python-pip** root@ansible-server:/# **pip install ansible** **Install PIP (On Cent OS)** root@ansible-server:/# **yum -y install python-pip** root@ansible-server:/# **pip install ansible** **Install pywinrm** root@ansible-server:/# **pip install pywinrm**
	Fix errors while installing **pywinrm** on **Ansible Server (Ubuntu 14.04)**	
	Error 1 - Cryptography requires setuptools 18.5 or newer	**Soln:** Install python-setuptools properly on Ubuntu 14.04 **pip install -U setuptools**

	Error 2 - [python] setup script exited with error: command 'x86_64-linux-gnu-gcc' failed with exit status 1	**Soln:** **apt-get install build-essential autoconf libtool pkg-config python-opengl python-imaging python-pyrex python-pyside.qtopengl idle-python2.7 qt4-dev-tools qt4-designer libqtgui4 libqtcore4 libqt4-xml libqt4-test libqt4-script libqt4-network libqt4-dbus python-qt4 python-qt4-gl libgle3 python-dev libssl-dev** **easy_install greenlet** **easy_install gevent**
	Error 3 - /usr/local/lib/python2.7/dist-packages/requests/__init__.py:91:RequestsDependencyWarning: urllib3 (1.7.1) or chardet (2.0.1) doesn't match a supported version! RequestsDependencyWarning) urllib3 (1.22) or chardet (2.2.1) doesn't match a supported version! RequestsDependencyWarning)	**Soln:** **pip uninstall requests** **pip install requests** **pip uninstall docopt** **pip install docopt**
	Error 4 - AttributeError: 'Session' object has no attribute 'merge_environment_settings'	**Soln:** **sudo pip install -U requests**
4	Update the inventory file on **Ansible Server (Ubuntu 14.04)**	Edit **/etc/ansible/hosts** and add: **[windows]** **172.20.4.127**
5	Update the Ansible Group Variables for Windows	Create a **group-vars** file for the **windows-host**, edit **/etc/ansible/group_vars/windows.yml** (create directory if it does not exist). And add details of Windows host **---** **ansible_user: 'administrator'** **ansible_password: 'password'** **ansible_port: '5986'** **ansible_connection: 'winrm'** **ansible_winrm_server_cert_validation: 'ignore'**

6	Test Connection	root@ansible-server:/# **ansible windows -m win_ping**
		172.20.4.127 | SUCCESS => {
		"changed": false,
		"ping": "pong"
		}

2. Commands:

3.1. Ping (For Linux Client)

Syntax:-

ansible -m ping <hostgroup/hosts>

Eg:-

```
root@ansible-server:/# ansible -m ping clients
172.20.4.130 | SUCCESS => {
   "changed": false,
   "ping": "pong"
}
172.20.4.112 | SUCCESS => {
   "changed": false,
   "ping": "pong"
}
```

3.2. Ping (For Windows Client)

Syntax:-

ansible -m win_ping <hostgroup/hosts>

Eg:-

```
root@ansible-server:/# ansible -m win_ping windows
172.20.4.127 | SUCCESS => {
   "changed": false,
   "ping": "pong"
}
```

3.3. Ansible with Linux Shell

Eg:-1

```
root@ansible-server:/# ansible -m shell -a 'hostname' clients
172.20.4.130 | CHANGED | rc=0 >>
client-02

172.20.4.112 | CHANGED | rc=0 >>
client-01
```

```
root@ansible-server:/# ansible -m shell -a 'uname -a' clients

172.20.4.112 | CHANGED | rc=0 >>
Linux client-01 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux

172.20.4.130 | CHANGED | rc=0 >>
Linux client-02 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
```

Eg: - 2
If we only wanted to run playbook on 172.20.4.106, we could type this

```
root@ansible-server:/etc/ansible/roles# ansible -m shell -a "hostname" 172.20.4.106
172.20.4.106 | CHANGED | rc=0 >>
nagios-server
```

3.4. Ansible with Windows Command Prompt & PowerShell

Syntax:-

ansible -m win_command -a “cmd” <hostgroup/hosts>

Eg:-1

```
root@ansible-server:/# ansible -m  win_command -a 'hostname' windows_domain
172.20.4.146 | CHANGED | rc=0 >>
KWS1A064
```

PowerShell Command

```
root@ansible-server:/# ansible -m  win_shell -a 'hostname' windows_domain
172.20.4.127 | CHANGED | rc=0 >>
DESKTOP-OMB0O4F
```

Eg: - 2
If we only wanted to run playbook on 172.20.4.127, we could type this

```
root@ansible-server:/etc/ansible/roles# ansible -m win_command -a "hostname" 172.20.4.127
172.20.4.127 | CHANGED | rc=0 >>
DESKTOP-OMB0O4F
```

3. PlayBook:

hosts	Host or Group of hosts
tasks	Create task
- name	Name of the task
yum	This module allows us to specify a package and the state that it should be in, which is "installed" in our case.
update-cache=true	Tells our remote machine to update its package cache (apt-get update) prior to installing the software
notify	The "notify" item contains a list with one item, which is called "start nginx". This is not an internal Ansible command, it is a reference to handler, which can perform certain functions when it is called from within a task.
handlers	Handlers are just like tasks, but they only run when they have been told by a task that changes have occurred on the client system. For instance, we have a handler here that starts the Nginx service after the package is installed. The handler is not called unless the "Installs nginx web server" task results in changes to the system, meaning that the package had to be installed and wasn't already there. Eg:- --- - hosts: clients tasks: - name: Installs nginx web server apt: pkg=nginx state=installed update_cache=true notify: - start nginx handlers: - name: start nginx service: name=nginx state=started

3.1. Running an Ansible PlayBook

root@ansible-server:/etc/ansible/roles# **ansible-playbook ping.yml**

PLAY [clients] **
TASK [Gathering Facts] **
ok: [172.20.4.130]

```
ok: [172.20.4.112]
TASK [ping] ********************************************************************
ok: [172.20.4.130]
ok: [172.20.4.112]
PLAY RECAP *********************************************************************
172.20.4.112               : ok=2    changed=0    unreachable=0    failed=0
172.20.4.130               : ok=2    changed=0    unreachable=0    failed=0
```

3.2. If we only wanted to run playbook on 172.20.4.112, we could type this.

root@ansible-server:/etc/ansible/roles# **ansible-playbook -l 172.20.4.112 ping.yml**

```
PLAY [clients] *****************************************************************
TASK [Gathering Facts] *********************************************************
ok: [172.20.4.112]
TASK [ping] ********************************************************************
ok: [172.20.4.112]
PLAY RECAP *********************************************************************
172.20.4.112               : ok=2    changed=0    unreachable=0    failed=0
```

3.3 Linux PlayBooks

Task	**PlayBook**	
	CentOS	Ubuntu
Ping	--- - hosts: clients Tasks:	--- - hosts: clients Tasks:
	- name: ping action: ping	- name: ping action: ping
Update packages	- name: Update packages yum: update_cache: yes	- name: Update packages apt: update_cache: yes

Update packages with conditions		- name: Update packages apt: update_cache: yes when: ansible_os_family == 'Debian'
Install Apache	- name: install the latest version of Apache yum: name: httpd state: present	- name: Install apache httpd apt: name: apache2 state: present
Install the latest version of Apache	- name: install the latest version of Apache yum: name: httpd state: latest	- name: Install apache httpd apt: name: apache2 state: latest
Uninstall Apache	- name: remove the Apache package yum: name: httpd state: absent	- name: remove the Apache package apt: name: apache2 state: absent
Install a list of packages	- name: Install a list of packages yum: name: "{{ packages }}" vars: packages: - httpd - httpd-tools	- name: Install a list of packages apt: name: "{{ packages }}" vars: packages: - apache2 - apache2-tools
	- name: Install a list of packages yum: name: - nginx - postgresql - postgresql-server state: present	
Install one specific version of Apache	- name: install one specific version of Apache yum: name: httpd-2.2.29-1.4.amzn1 state: present	- name: install one specific version of Apache apt: name: apache2=2.2.20-1ubuntu1 state: present

Upgrade all packages to the latest version	- name: upgrade all packages yum: name: '*' state: latest	- name: upgrade all packages apt: name: "*" state: latest
Update all packages to the latest version		- name: Update all packages to the latest version apt: upgrade: dist
Check if a reboot is required		- name: Check if a reboot is required register: file stat: path=/var/run/reboot-required get_md5=no - name: Reboot the server command: /sbin/reboot when: file.stat.exists == true
Remove useless packages from the cache		- name: Remove useless packages from the cache apt: autoclean: yes
Remove dependencies that are no longer required		- name: Remove dependencies that are no longer required apt: autoremove: yes
Create File **Note:** - [state = touch -> Ansible check such file exists. If it exists, ansible will do nothing, but if it doesn't, it will create it.]	- name: Create file file: path: /home/ashir/note.txt state: touch	- name: Create file file: path: /home/ashir/note.txt state: touch
Copy file from Ansible server to client machines	- name: Copy test files copy: src=/home/ashir/test1/ dest=/home/ashir/ mode=0644	- name: Copy test files copy: src=/home/ashir/test1/ dest=/home/ashir/ mode=0644

Remove file	- name: Remove files command: rm /home/ashir/test_file1.txt	
Update file	- name: remove file command: rm /home/ashir/note.txt - name: update file copy: src=/home/ashir/test1/note.txt dest=/home/ashir/ mode=0644	- name: remove file command: rm /home/ashir/note.txt - name: update file copy: src=/home/ashir/test1/note.txt dest=/home/ashir/ mode=0644

Copy files with conditions

```
- name: Upload default index.php for host
  copy: src=static_files/index.php dest=/usr/share/nginx/www/ mode=0644
  register: php
  ignore_errors: True

- name: Remove index.html for host
  command: rm /usr/share/nginx/www/index.html
  when: php|success

- name: Upload default index.html for host
  copy: src=static_files/index.html dest=/usr/share/nginx/www/ mode=0644
  when: php|failed
```

3.4 Windows PlayBooks

Task	PlayBook
Hostname	--- - name: Windows hosts: windows_domain tasks: - name: Hostname win_command: hostname

To get PlayBook output to display	register: output - debug: var: output.stdout_lines
PlayBook with PowerShell (win_shell) commands	--- - name: Windows_PS hosts: windows_domain tasks: - name: Username **win_shell**: '(Get-WmiObject -Class win32_process | Where-Object name -Match explorer).getowner().user' register: username - debug: msg: '{{ username.stdout_lines }}' - name: Memory Usage (%) win_shell: '[math]::Round(((((gwmi -Class win32_operatingsystem).TotalVisibleMemorySize) - ((gwmi -Class win32_operatingsystem).FreePhysicalMemory))/ ((gwmi -Class win32_operatingsystem).TotalVisibleMemorySize))*100, 2)' register: mem_usage - debug: msg: '{{ mem_usage.stdout_lines }}'
Check Windows Version	--- - name: Windows File Version hosts: windows_domain tasks: - name: Get version **win_file_version:** path: C:\Windows\System32\cmd.exe register: version - debug: msg: '{{ version }}'

Rebooting automatically is also possible with a small playbook	- hosts: windows tasks: - name: apply critical and security windows updates win_updates: category_names: - SecurityUpdates - CriticalUpdates register: wuout - name: reboot if required win_reboot: when: wuout.reboot_required
Run a PowerShell script	- name: Run Powershell Scripts hosts: windows_domain tasks: - name: run a powershell script script: /etc/ansible/ps_scripts/System_Report.ps1 register: out - debug: var=out.stdout_lines

Ansible and Docker

1. **Ansible**

Ansible is a configuration management tool. It makes IT automation simple as it ends repetitive tasks and enables faster application deployments.
It automates configuration management, application deployment, and a number of other IT requirements. Using Ansible we can create containers on Docker.

2. **Docker**

Docker is a software container technology platform that an environment to create, deploy, run, and manage applications within the containers. This containers are isolated and separated from host OS.