



## **Configuration Management & Provisioning**

# OUTLINE



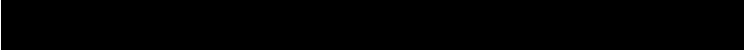
Goals : Participant will be able to use ansible for provisioning end-to-end

## Understanding Ansible

- What is ansible
- What can it do?
- Getting Ansible
- Ansible Setup/Architecture

## Start Using Ansible

- Inventory
- Ad-Hoc Commands
- Modules
- Playbook

Extras : 





## ANSIBLE ?

- Simple automation **language** that can describe an IT Application Infrastructure
- **Agentless** automation **engine** that runs playbooks.
- It can **configure** systems, **deploy** software, and **orchestrate** more advanced IT tasks such as **continuous deployments**.

# Why ansible?



ANSIBLE



## SIMPLE

Human readable automation  
No special coding skills needed  
Tasks executed in order  
**Get productive quickly**



## POWERFUL

App deployment  
Configuration management  
Workflow orchestration  
**Orchestrate the app lifecycle**



## AGENTLESS

Agentless architecture  
Uses OpenSSH & WinRM  
No agents to exploit or update  
**More efficient & more secure**



# What it do?



**CONFIG MANAGEMENT**



**APP DEPLOYMENT**



**PROVISIONING**



**CONTINUOUS DELIVERY**



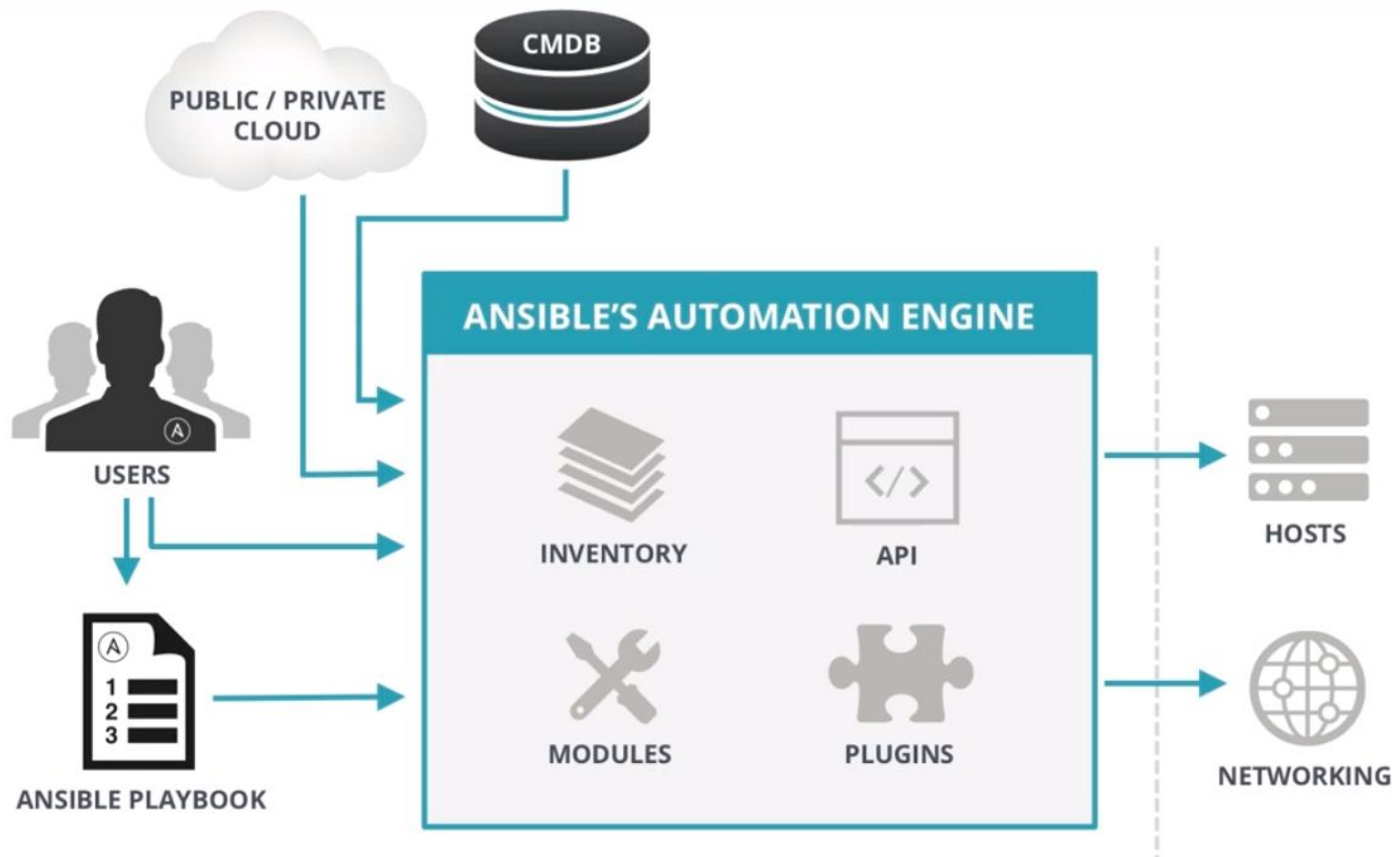
**SECURITY & COMPLIANCE**



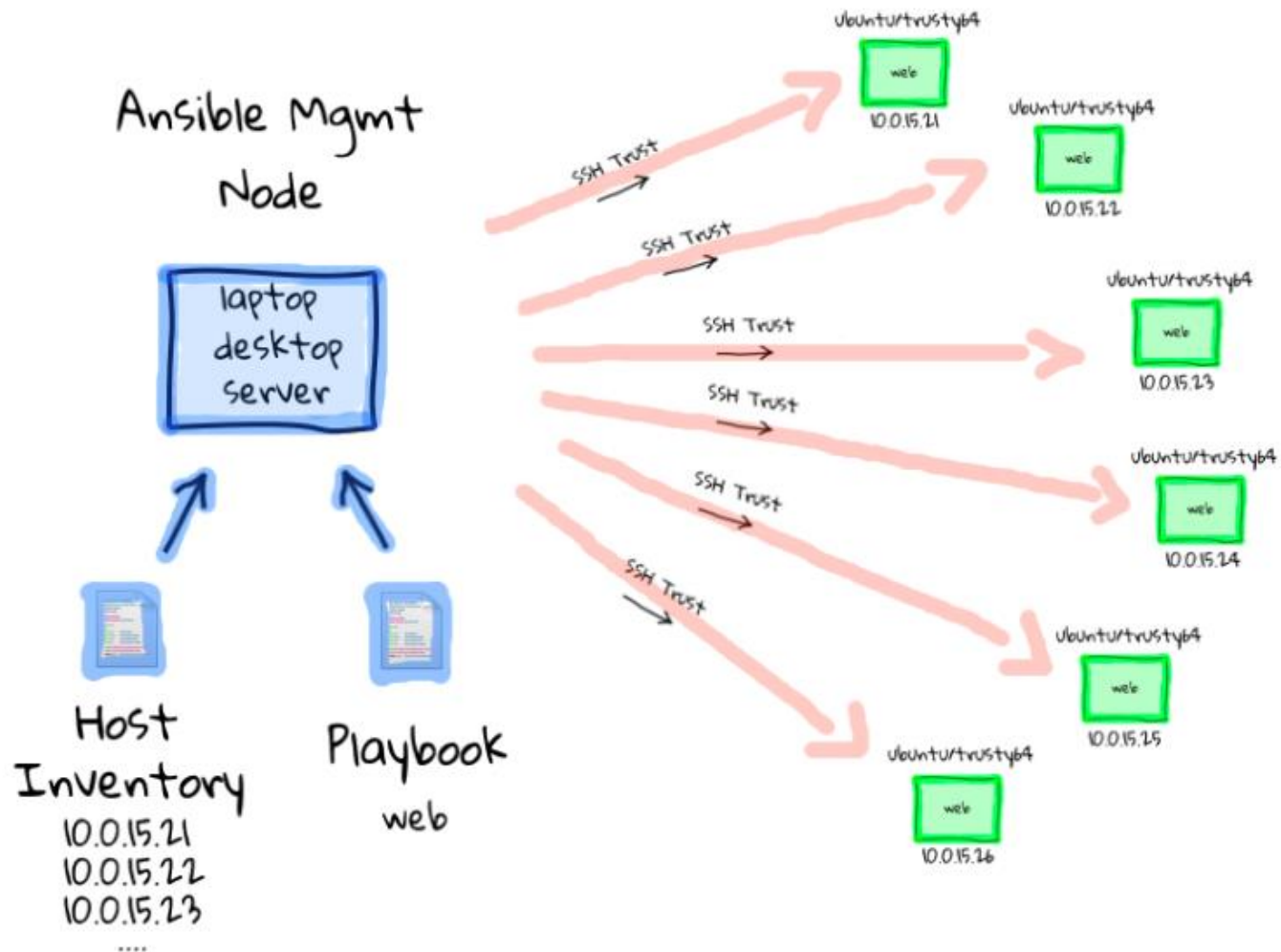
**ORCHESTRATION**



# Ansible Architecture (RedHat)



# Ansible Architecture (Noobs)





**Everyone was a noob at some point in life**



**And its ok**





# Installing ansible



THE ULTRA NOOB WAY (but recommended) :



Step 1 :

- Open up your terminal and install python2 **AND** python3

```
yum/apt-get/zypper/pacman install -y python2 python3
```

Step 2 :

- Install ansible engine

CentOS / Fedora / RHEL :

```
$ yum install -y ansible
```

Ubuntu :

```
$ sudo apt update
```

```
$ sudo apt install software-properties-common
```

```
$ sudo apt-add-repository --yes --update ppa:ansible/ansible
```

```
$ sudo apt install ansible
```



# Installing ansible



THE ULTRA NOOB WAY (but recommended) :

Step 3 :

Nothing, its done, now start working.



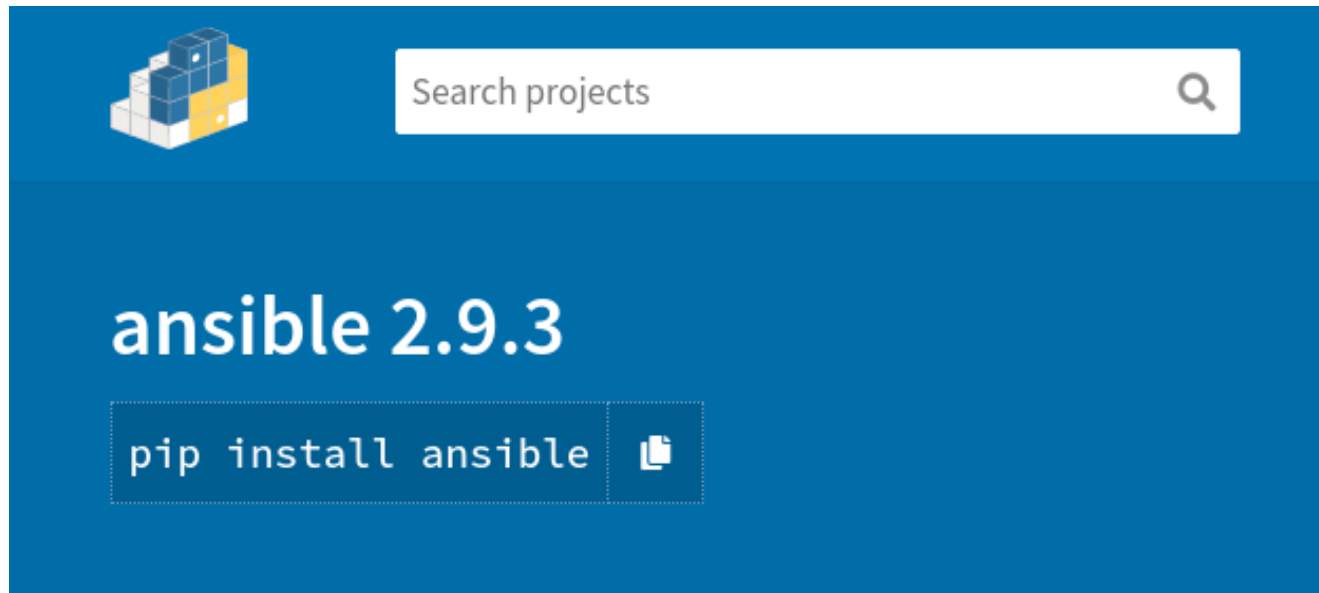
# Installing ansible



THE EZ WAY (ultimate noob):



Ansible is a python project. Its available via pip, and you can look it up in Pypi.org



# A little caveat



- Installing via package manager is recommended because it lets your distribution manage the necessary files.
- Installing via pip means you're going to have to manage your files on your own (and its super simple trust me).
- Me personally, prefer using the pip version because its going to get faster updates (this includes 450+ modules that ships with it)
- [https://docs.ansible.com/ansible/latest/modules/list\\_of\\_all\\_modules.html](https://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html)
- Ansible is a fast-moving project, using faster upstream for your system going to solve most of your problems with it (usually)

The screenshot shows the GitHub repository for Ansible. At the top, it says 'ansible / ansible'. To the right are statistics: 'Used by 9.8k', 'Watch 2k', 'Unstar 41.4k', and 'Fork 18k'. Below this is a navigation bar with links to 'Code', 'Issues 4,289', 'Pull requests 2,020', 'Actions', 'Projects 26', 'Security', and 'Insights'. The main content area has a description: 'Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. Avoid writing scripts or custom code to deploy and update your applications — automate in a language that approaches plain English, using SSH, with no agents to install on remote systems.' It includes two links: <https://docs.ansible.com/ansible/> and <https://www.ansible.com/>. Below the description are two tags: 'python' and 'ansible'. At the bottom, there is a bar with more statistics: '48,871 commits', '43 branches', '0 packages', '298 releases', '4,854 contributors', and 'GPL-3.0'.



# Ansible File Structure



```
production          # inventory file for production servers
staging             # inventory file for staging environment

group_vars/
  group1.yml        # here we assign variables to particular groups
  group2.yml
host_vars/
  hostname1.yml     # here we assign variables to particular systems
  hostname2.yml

library/            # if any custom modules, put them here (optional)
module_utils/       # if any custom module_utils to support modules, put them here (optional)
filter_plugins/     # if any custom filter plugins, put them here (optional)

site.yml            # master playbook
webservers.yml      # playbook for webserver tier
dbservers.yml       # playbook for dbserver tier

roles/
  common/           # this hierarchy represents a "role"
    tasks/          #
      main.yml       # <-- tasks file can include smaller files if warranted
    handlers/       #
      main.yml       # <-- handlers file
    templates/      # <-- files for use with the template resource
      ntp.conf.j2    # <----- templates end in .j2
    files/          #
      bar.txt        # <-- files for use with the copy resource
      foo.sh         # <-- script files for use with the script resource
    vars/           #
      main.yml       # <-- variables associated with this role
    defaults/       #
      main.yml       # <-- default lower priority variables for this role
    meta/           #
      main.yml       # <-- role dependencies
    library/        # roles can also include custom modules
    module_utils/   # roles can also include custom module_utils
    lookup_plugins/ # or other types of plugins, like lookup in this case

webtier/            # same kind of structure as "common" was above, done for the webtier role
monitoring/         # ""
fooapp/             # ""
```



# Ansible File Structure (Alternative)



```
inventories/  
  production/  
    hosts          # inventory file for production servers  
    group_vars/  
      group1.yml   # here we assign variables to particular groups  
      group2.yml  
    host_vars/  
      hostname1.yml # here we assign variables to particular systems  
      hostname2.yml  
  
  staging/  
    hosts          # inventory file for staging environment  
    group_vars/  
      group1.yml   # here we assign variables to particular groups  
      group2.yml  
    host_vars/  
      stagehost1.yml # here we assign variables to particular systems  
      stagehost2.yml  
  
library/  
module_utils/  
filter_plugins/  
  
site.yml  
webservers.yml  
dbservers.yml  
  
roles/  
  common/  
  webtier/  
  monitoring/  
  fooapp/
```



# Ansible File Structure (KISS)



```
hardika.gutama in ~/Workspace/project/ansiblefuture on master λ tree .
.
├── ansible.cfg
├── hosts.ini
├── playbooks
│   ├── files
│   │   └── index.html
│   └── systemupdate.yaml
└── README.md

2 directories, 5 files
```

<https://github.com/hrdkgtm/ansiblefuture>



# Setting Up (post-install?)



Ansible will run via SSH, this requires you to setup the users in the target system. Using 1 deployment user is preferable in this case.

Steps to do :

1. Create the 'ansible' user for your target system.
2. Allow the user in /etc/sudoers (use `visudo` u scrub)
3. Delegate the SSH Key file to target system.





# Start Using Ansible : The Inventory



- Inventory is a list of hosts that you give to ansible to tell it where it should run. Basically, list of targets.
- One of the features of ansible is the ability to define the Inventory in a logical way. This means you can Group them however you want to group them
- You can even put them in 2 groups because ansible doesn't limit anyone (im referring a server) to be in multiple groups at once.
- This definition of hosts can be described in a .INI format or .YAML format.
- Choose whatever you like and can get you run faster. Don't listen if anyone says .INI or .YAML sucks. (i think .ini sucks tho)



# Use Case



- Now imagine something happened to your infrastructure, and somehow it requires you to restart a service in systemd.
- You probably will type this into your terminal :

```
$ ssh brokenserver-1  
$ sudo systemctl restart brokenservice
```
- It turned out that you have 500+ different servers, with different names, that spreads around different clouds, datacenters, somebody else's laptop that you hacked into in 2014 and you still have access to it.



# Use Case



It turned out that you have 500+ different servers, with different names, that spreads around different clouds, datacenters, somebody else's laptop that you hacked into in 2014 and you still have access to it, or whatever.



# Use Case



```
$ ssh brokerserver-1
$ sudo systemctl restart brokerservice
$ ssh brokerserver-2
$ sudo systemctl restart brokerservice
$ ssh brokerserver-3
$ sudo systemctl restart brokerservice
$ ssh brokendb-1
$ sudo systemctl restart brokerservice
$ ssh brokendb-2
$ sudo systemctl restart brokerservice
$ ssh brokendb-3
$ sudo systemctl restart brokerservice
$ ssh brokerserver-3 # I mistyped on the server
$ sudo systemctl restart brokerservice
$ ssh brokenkafka-1
$ sudo systemctl restart brokerservice
$ ssh brokenkafka-2
$ sudo systemctl restart brokerservice
$ ssh brokenkafka-3
$ sudo systemctl restart brokerservice
$ ssh justinlaptop
$ sudo systemctl restart brokerservice
$ ssh justinsmarttv
$ sudo systemctl restart brokerservice
$ ssh thepiratebay-1
$ sudo systemctl restart brokerservice
$ ssh thepiratebay-2
$ sudo systemctl restart brokerservice
$ ssh nfsbroke-1
$ sudo systemctl restart brokerservice
$ ssh nfsnotbroken-15
$ sudo systemctl restart brokerservice
$ sudo rm rf /
```



# Start Using Ansible : Ad-hoc Commands



- Ofcourse not, use ansible :

```
$ ansible -m service -a 'name=brokenservice state=restarted' all
```



# Start Using Ansible : Ad-hoc Commands



- Ansible comes with a feature to run individual modules via CLI / Interactive shell. Without a playbook.
- You can call specific modules
- Try it out :  

```
$ ansible -m ping all
```



# Start Using Ansible : Modules



- Modules is a thing (?) that is used by ansible to know what kind of tool it needs to run and have different tunables inside of it
- There are currently 450+ modules already listed in the Ansible Module Index
- You can develop your own personalized module if you need it.
- [https://docs.ansible.com/ansible/latest/modules/list\\_of\\_all\\_modules.html](https://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html)



# Start Using Ansible : Playbooks



- Playbooks is a declarative way for you to tell ansible what to do
- Executed Sequentially (top to bottom)
- YAML format
- Extreme ez





# Start Using Ansible : Variables



- You probably know what a variable is.
- The variables in ansible can be defined in a lot of different places and it has its own priority. (Google : Ansible Variable Precedence)
- These variables later can be loaded into a your playbook.
- Can be used for personalization of the services.



# TASK 1 : The Webserver



- I have 50 Apache servers running in production. These servers are currently failing and we have a plan to move to NGINX because someone in stackoverflow said apache sucks.
- Make me an ansible playbook to install nginx.



# TASK 2 : Change Request



- Some security dude told us that running on port **8080** is unsecure because its too obvious. We need to change it to **40080**.
- Also, we want to run on a different nginx folder. The security dude told us too that the default folder is going to be vulnerable for script kiddies that breaks into our server. Folder = **/opt/html/index.html**
- And then, we have this new and nice PHP code that we want to ship. You will need to add a step to make the code works with nginx whenever we call it.

Challenge :

- Oh, and can you also add a page, that can show us the IP of the webserver we are accessing later?

Kirim email : [hardika.gutama@gdn-commerce.com](mailto:hardika.gutama@gdn-commerce.com)

Subject : [Bibli-Future] Nama Lengkap



# Extras : Roles



- Roles are ways of automatically loading certain vars\_files, tasks, and handlers based on a known file structure. Grouping content by roles also allows easy sharing of roles with other users.
- Start from a playbook. Split it into tasks, and make a role.
- You can store a general usage steps into a role.
- Example : You want a role for installing nginx and installing base packages, and then you can call it from a playbook



# Extras : Ansible Galaxy



- Don't try to reinvent the wheel. Your time is precious and you should respect it
- <https://galaxy.ansible.com/>



# Extras : Ansible Vault



- allows you to keep sensitive data such as passwords or keys in encrypted files, rather than as plaintext in playbooks or roles. These vault files can then be distributed or placed in source control (git).
- [https://docs.ansible.com/ansible/latest/user\\_guide/vault.html](https://docs.ansible.com/ansible/latest/user_guide/vault.html)





**The Best Linux Blog In the Unixverse** @nixcraft · Jan 12

Explain DevOps to me like I am Five.



133



78



535



**The Red Pen, Move Zig for Great Justice**

@TheRedPen3

Replying to @nixcraft

When a development team, a QA team, and an operations team love each other very much...

...and then you automate it.



# References



- <https://www.ansible.com/resources/get-started>
- <https://docs.ansible.com/ansible/latest/index.html>
- [https://docs.ansible.com/ansible/latest/user\\_guide/intro\\_inventory.html](https://docs.ansible.com/ansible/latest/user_guide/intro_inventory.html)
- [https://docs.ansible.com/ansible/latest/modules/list\\_of\\_all\\_modules.html](https://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html)
- [https://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_variables.html](https://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html)
- <https://www.ansible.com/overview/how-ansible-works>
- [https://docs.ansible.com/ansible/latest/user\\_guide/vault.html](https://docs.ansible.com/ansible/latest/user_guide/vault.html)
- [https://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_reuse\\_roles.html](https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html)
- [https://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_best\\_practices.html](https://docs.ansible.com/ansible/latest/user_guide/playbooks_best_practices.html)





# Ansible File Structure



**BUT YOU SAID IT WAS SIMPLE?**

