ANSIBLE

ANSIBLE

A TECHNICAL INTRODUCTION

Prasad Deshpande

prdeshpa@redhat.com

AGENDA

- WHAT IS ANSIBLE?
- REQUIREMENTS
- ARCHITECTURE OVERVIEW
- ANSIBLE CORE COMPONENTS
- ADDITIONAL REFERENCES

ANSIBLE

IS INFRASTRUCTURE AS A CODE...

It’s a **simple automation language** that can perfectly describe an IT application infrastructure in Ansible Playbooks. It's also an **automation engine** that runs the Ansible Playbooks.

Ansible is an open source IT configuration management, deployment, and orchestration tool, based on Python.

Designed to be **minimal** in nature, **consistent**, **secure**, and **highly reliable**, with an extremely **low learning curve** for administrators, developers, and IT managers.

SIMPLE

Human readable automation

No special coding skills needed

Tasks executed in order

Get productive quickly

POWERFUL

Application Deployment

Continuous Delivery

Beyond just Servers

Orchestrate the App lifecycle

AGENTLESS

Agentless architecture

Uses OpenSSH and WinRM

No exploits or updates

More efficient and secure

REQUIREMENTS

- **CONTROL NODE**

① Ansible can be run from any machine with Python 2.6 or 2.7

- **MANAGED NODES**

① Linux/Unix

1.1 SSH

1.2 Python 2.4 or later

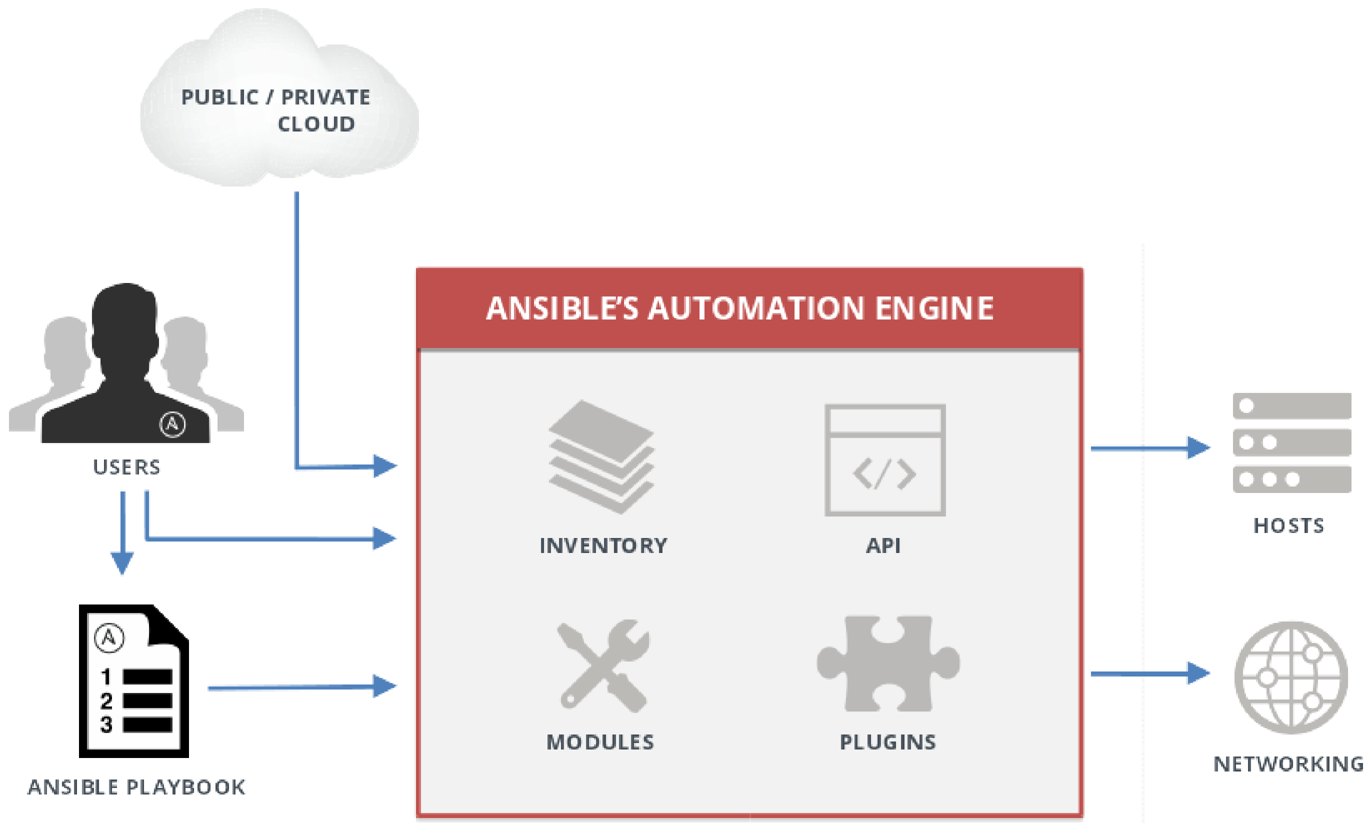
1.2.1 If running less than Python 2.5 on the remotes nodes, package "python-simplejson" is required. **Note**: RHEL 5.x has Python 2.4 only.

② Windows (Ansible 1.7+) - enable and configure PowerShell remoting 3.0+ (WinRM)

ANSIBLE

HOW ANSIBLE WORKS

- Ansible works by connecting from the **control node** to your **managed nodes** and **pushing out** small programs, called "Ansible modules" to them. These programs are written to be resource models of the desired state of the system.
- Ansible then executes these modules (**over SSH by default**) in the order they are specified in the playbook(s), and removes them when finished.



CORE COMPONENTS

- INVENTORIES
- MODULES
- PLAYBOOKS

INVENTORIES

Static - Defined in simple text files, a host can be member of more than one group, which is useful to identify the hosts role in the datacenter.

Dynamic - Generated for outside providers, some examples include pulling***** inventory from a cloud provider (OpenStack, AWS, etc)

```
[webservers]
web1.example.com
web2.example.com

[dbservers]
db[0:1].example.com

[appserver:children]
webservers
dbservers
```

Default location: /etc/ansible/hosts

MODULES

"**Ansible Modules**" are written to be resource models of the desired state of the system. Ansible then executes these modules (over SSH by default), and removes them when finished.

Modules allows us to manage from basic systems resources to sophisticated ones. For example, to manage users, packages, network*, files, services, as well provision cloud instances, create databases, and many more.

ad-hoc command to check module list:

```
# ansible-doc --list
```

PLAYBOOKS

A **playbook** consists of one or more **plays**, which map groups of hosts to well-defined tasks.

Plays also define the order in which tasks are configured. This allows us to orchestrate multitier deployments.

```
---
- name: My first playbook
 hosts: localhost
 tasks:
 - name: Install package httpd
  yum:
   name: httpd
   state: present

 - name: Restart httpd service
  service:
```

How to run a playbook:

```
# ansible-playbook <playbook_name>.yml
```

ADDITIONAL REFERENCES

GETTING STARTED

- Would you like to learn Ansible? It's easy to get started:

ansible.com/get-started

Questions?