ANSIBLE

Automation For Everyone

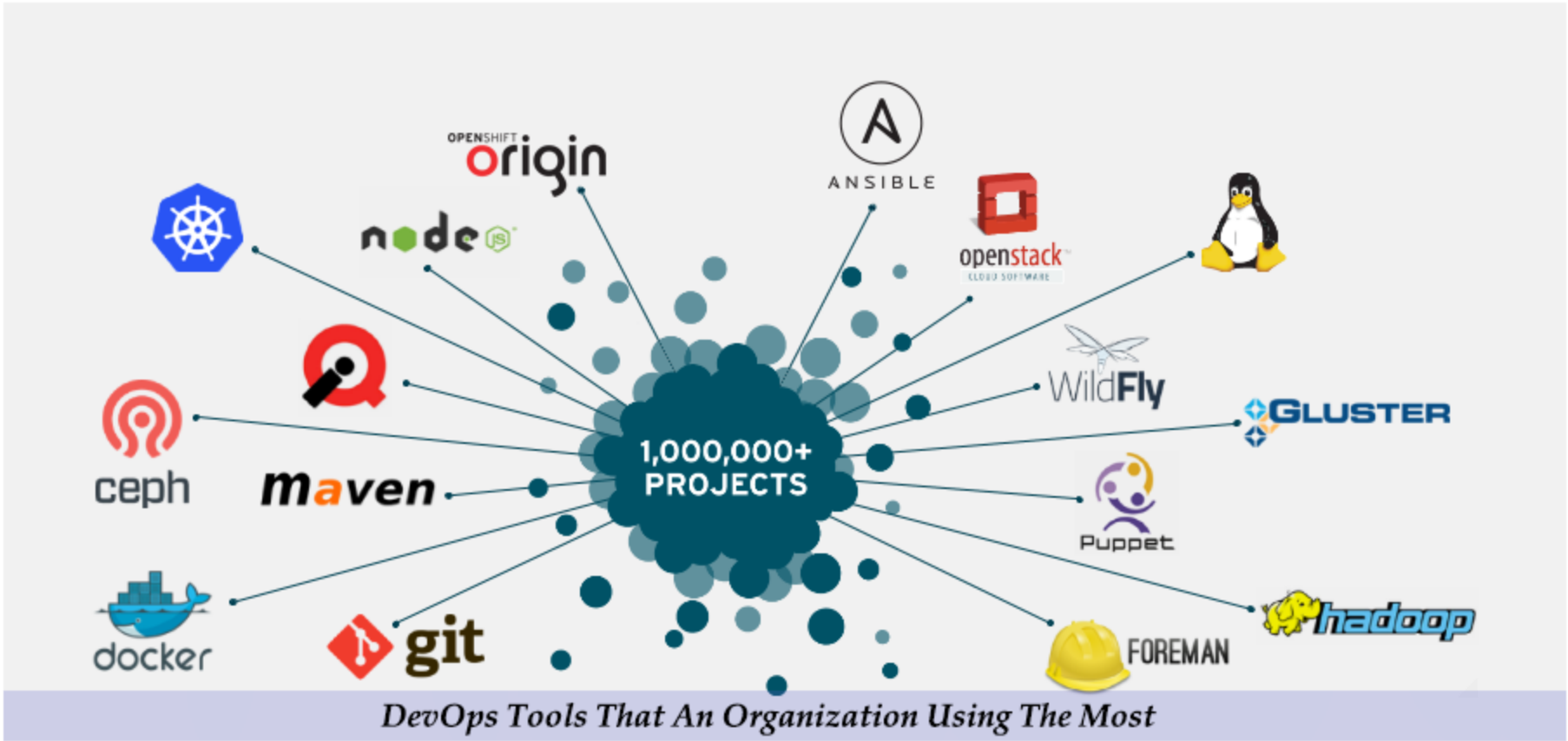
Ansible technical Introduction and overview

WHAT IS ANSIBLE?

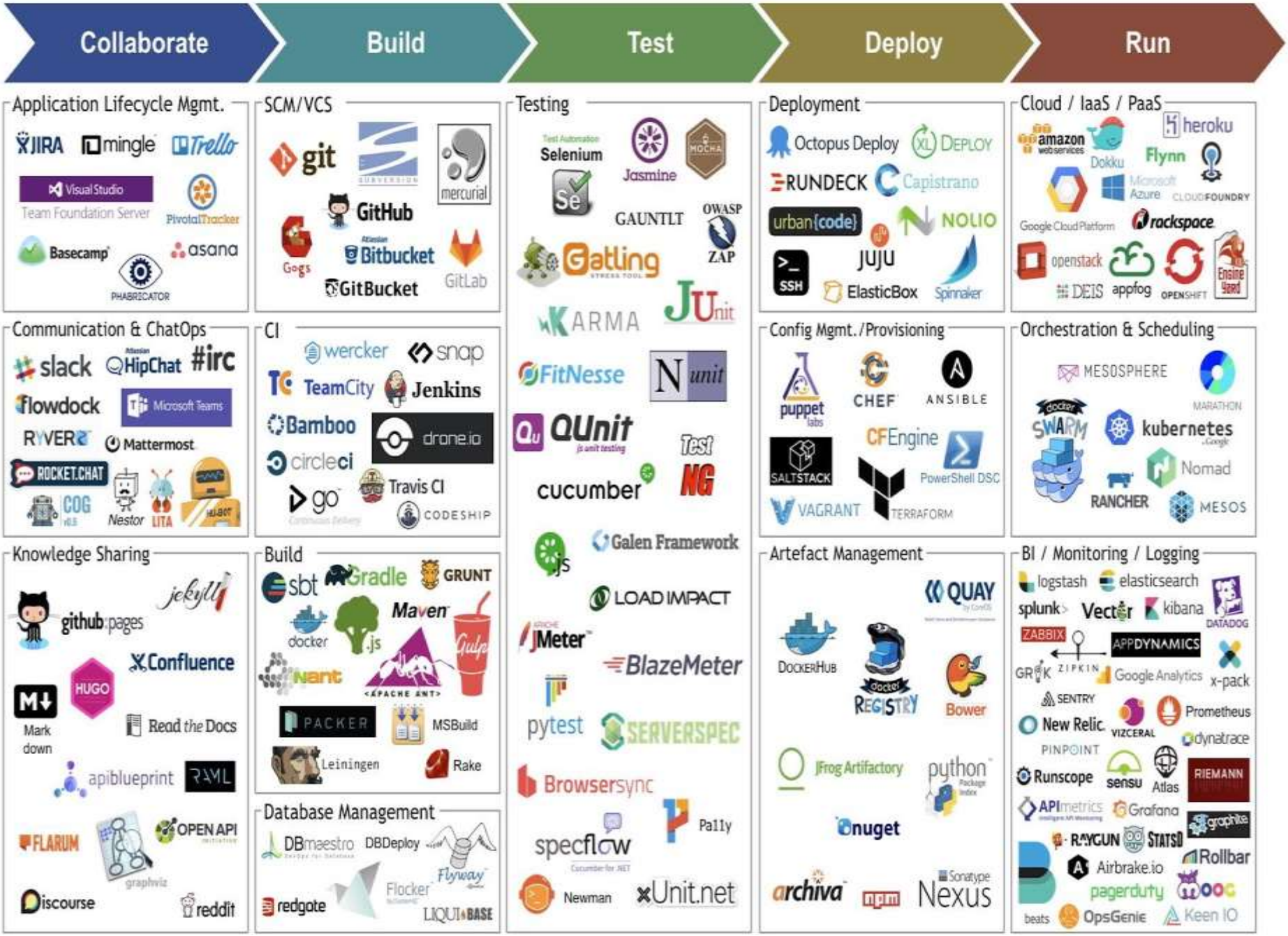
Ansible ?

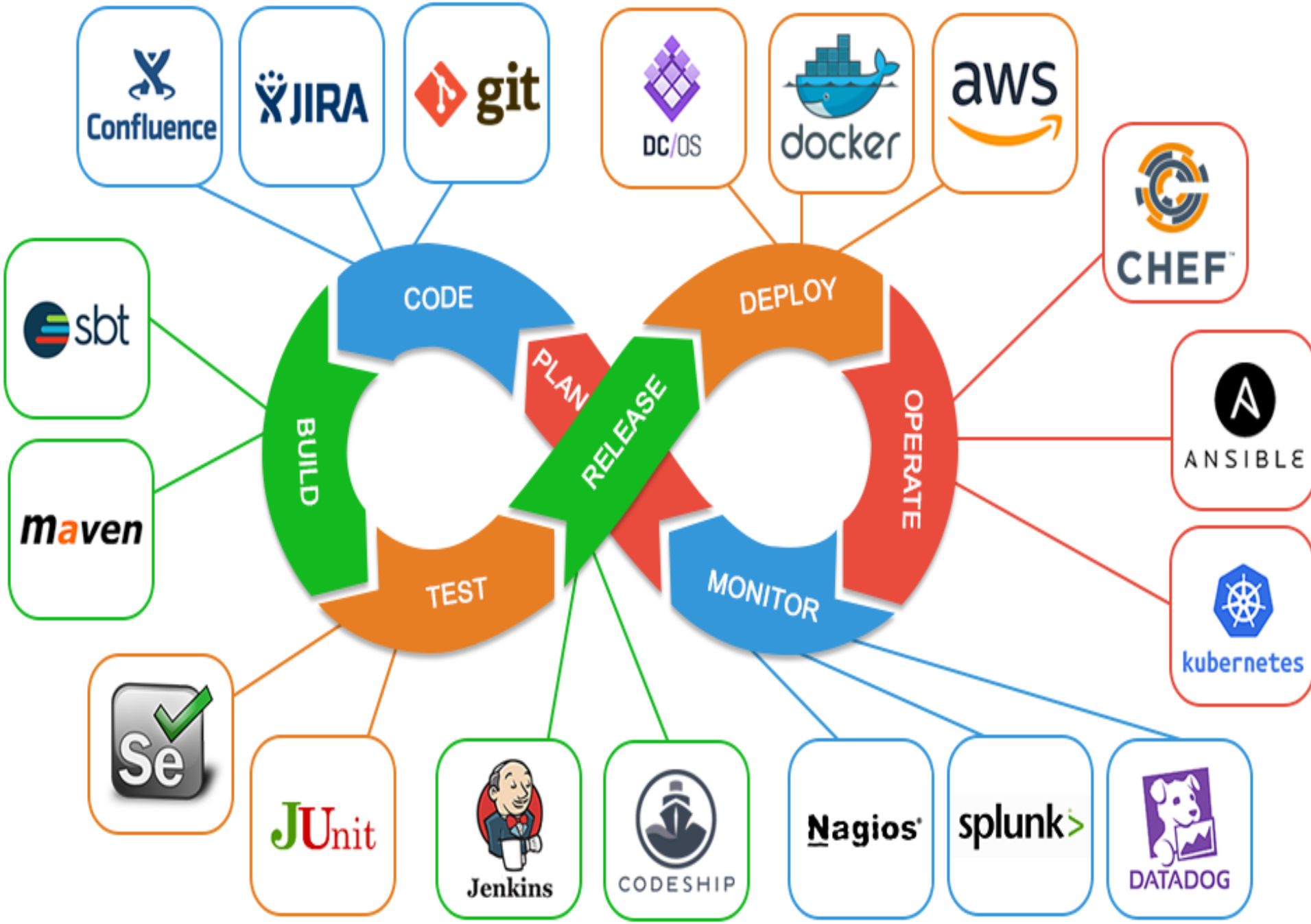
The term Ansible is a Science Fiction reference for a ficitonal communications device that can transfer information faster than the speed of light.

The author Ursula LeGuin invented the concept in her 1966 book ‘Rocannon’s World’,



DevOps Tools That An Organization Using The Most


Collaborate
Build
Test
Deploy
Run
Application Lifecycle Mgmt.
JIRA
mingle
Trello
Visual Studio
Team Foundation Server
PivotalTracker
Basecamp
asana
PHABRICATOR
SCM/VCS
git
SUBVERSION
mercurial
GitHub
Bitbucket
Gogs
GitLab
GitBucket
Testing
Test Automation
Selenium
Jasmine
MOCHA
GAUNTLT
OWASP
ZAP
Gatling
STRESS TOOL
JUnit
KARMA
FitNesse
N unit
QUnit
js unit testing
Test
NG
cucumber
Galen Framework
js
LOAD IMPACT
JMeter
BlazeMeter
pytest
SERVERSPEC
Browsersync
Pally
specflow
Cucumber for .NET
Newman
xUnit.net
Deployment
Octopus Deploy
DEPLOY
RUNDECK
Capistrano
urban{code}
NOLIO
SSH
JUJU
ElasticBox
Spinnaker
Cloud / IaaS / PaaS
amazon web services
Dokku
heroku
Flynn
Microsoft Azure
CLOUDFOUNDRY
Google Cloud Platform
rackspace
openstack
DEIS
appfog
OPENSHIFT
Engine Yard
Communication & ChatOps
slack
HipChat
#irc
flowdock
Microsoft Teams
RIVER
Mattermost
ROCKET.CHAT
COG
Nestor
LITA
CI
wercker
snap
TeamCity
Jenkins
Bamboo
drone.io
circleci
go
Travis CI
CODESHIP
Config Mgmt. / Provisioning
puppet labs
CHEF
ANSIBLE
SALTSTACK
CFEngine
PowerShell DSC
VAGRANT
TERRAFORM
Orchestration & Scheduling
MESOSPHERE
MARATHON
SWARM
kubernetes
Nomad
RANCHER
MESOS
Knowledge Sharing
github pages
jekyll
Confluence
HUGO
Mark down
Read the Docs
apiblueprint
RAML
FLARUM
OPEN API
graphviz
Discourse
reddit
Build
sbt
Gradle
GRUNT
docker
Maven
.js
Gulp
nant
APACHE ANT
PACKER
MSBuild
Leiningen
Rake
Artefact Management
QUAY
DockerHub
docker REGISTRY
Bower
JFrog Artifactory
python Package Index
nuget
archiva
npm
Sonatype Nexus
BI / Monitoring / Logging
logstash
elasticsearch
splunk>
Vector
kibana
DATADOG
ZABBIX
APPDYNAMICS
GROK
ZIPKIN
Google Analytics
x-pack
SENTRY
New Relic.
VIZCERAL
Prometheus
dynatrace
PINPOINT
Runscope
sensu
Atlas
RIEMANN
APImetrics
Grafana
graphite
RAYGUN
STATSD
Rollbar
Airbrake.io
pagerduty
beats
OpsGenie
Keen IO
Database Management
DBmaestro
DBDeploy
Flyway
Flocker
redgate
LIQUIBASE


Confluence
JIRA
git
DC/OS
docker
aws
CHEF
sbt
maven
ANSIBLE
kubernetes
Se
JUnit
Jenkins
CODESHIP
Nagios
splunk>
DATADOG
CODE
PLAN
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR

redhat.

Ansible

Automation technology you can use everywhere

AGENDA:

1. Ansible Automation and Introduction
2. Demo

KEY TAKE AWAYS:

1. What is IT automation ?
2. What is Ansible
3. How to write ansible Playbook
4. What Ansible Tower ?

Ansible by Red Hat

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

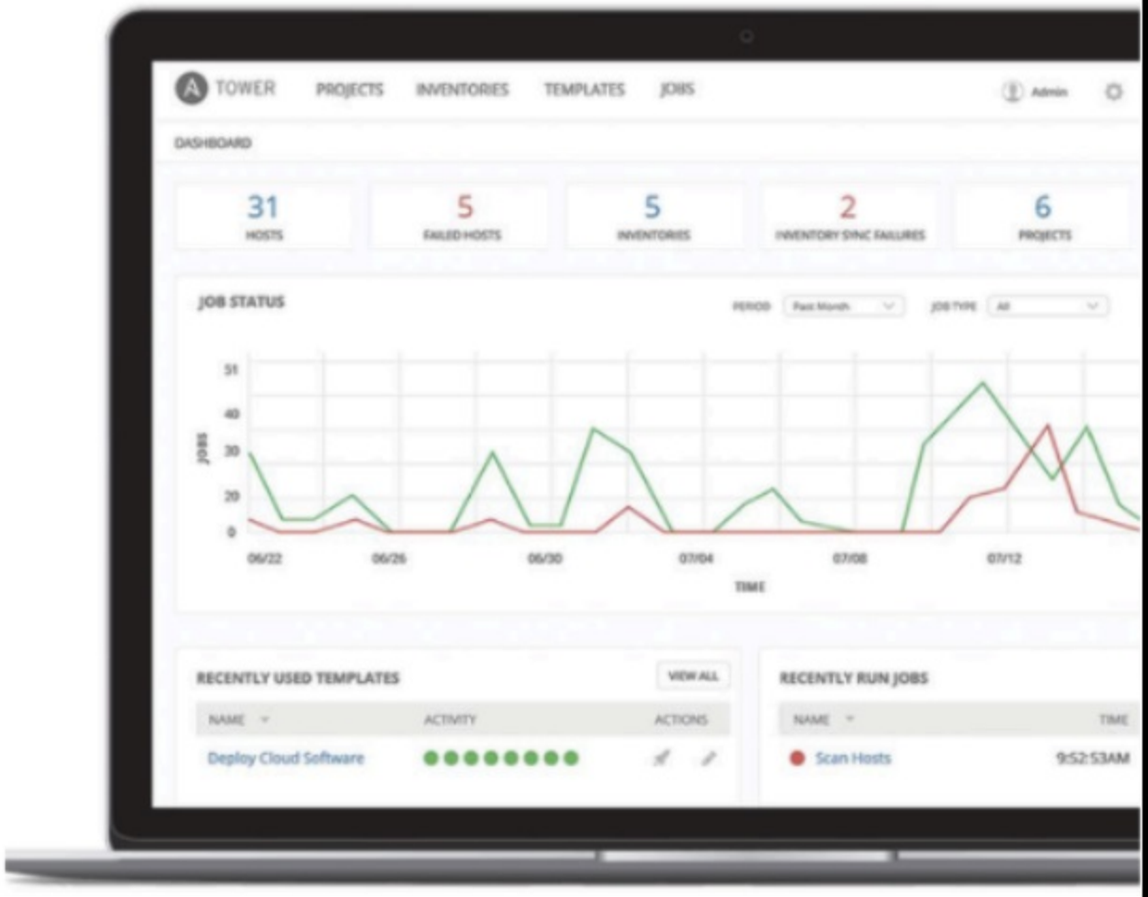
AGENTLESS

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

WHAT IS ANSIBLE ?

- *Ansible is an open source configuration management and Orchestration utility.*

- *It can automate and standardize the configuration of remote hosts and virtual machines.*



The Ansible Way

CROSS PLATFORM

Agentless support for all major OS variants, physical, virtual, cloud and network devices.

HUMAN READABLE

Perfectly describe and document every aspect of your application environment.

PERFECT DESCRIPTION OF

Every change can be made by Playbooks, ensuring everyone is on the same page.

VERSION CONTROLLED

Playbooks are plain-text. Treat them like code in your existing version control.

DYNAMIC INVENTORIES

Capture all the servers 100% of the time, regardless of infrastructure, location, etc.

ORCHESTRATION PLAYS WELL WITH

Every change can be made by Playbooks, ensuring everyone is on the same page.

What Can I Do With ANSIBLE?

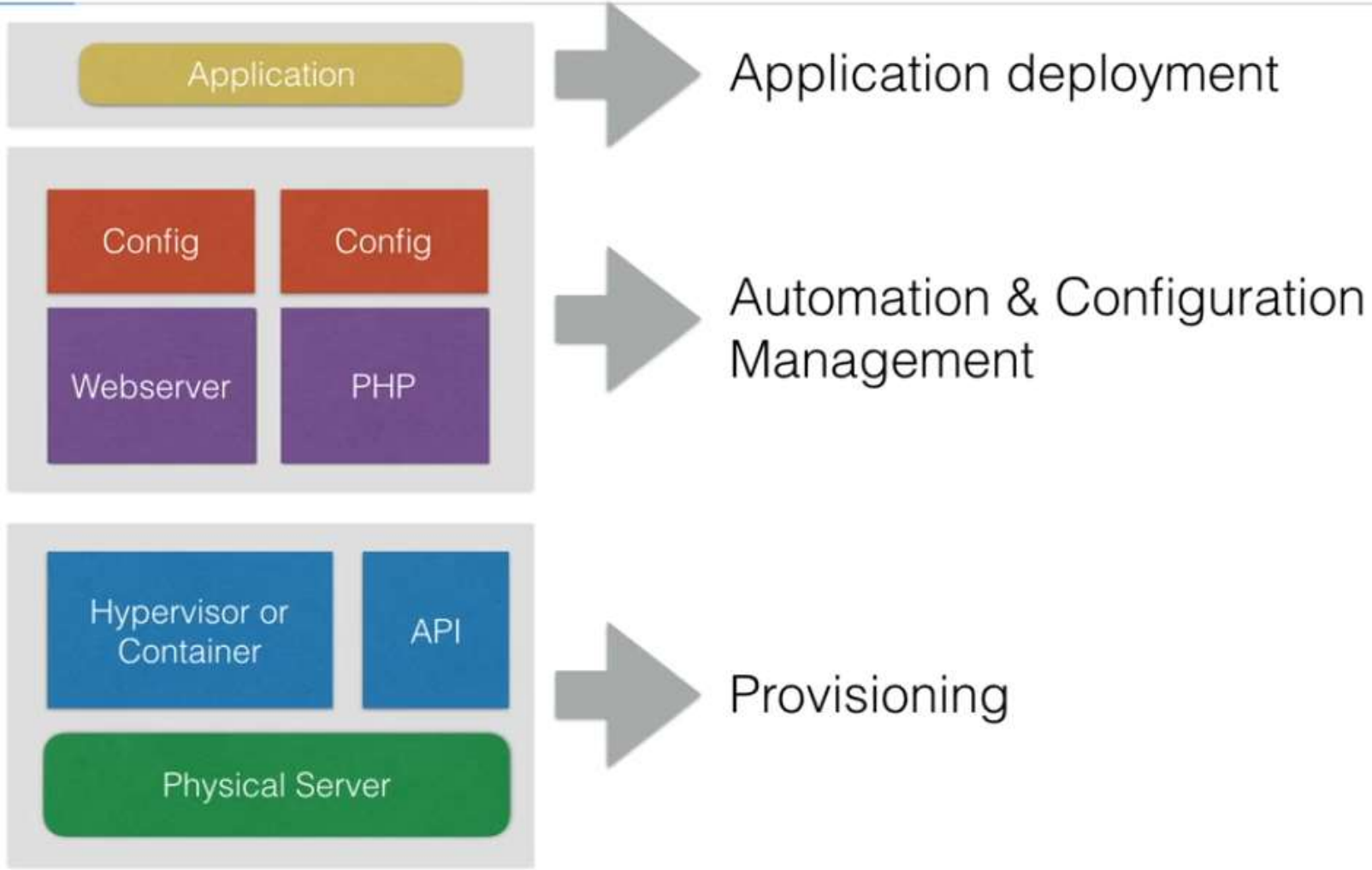
Do this...

Orchestration	Configuration Management	Application Deployment	Provisioning	Continuous Delivery	Security and Compliance

On these...

Firewalls	Load Balancers	Applications	Containers	Clouds
Servers	Infrastructure	Storage	Network Devices	**And more...**

Provisioning & Configuration Management

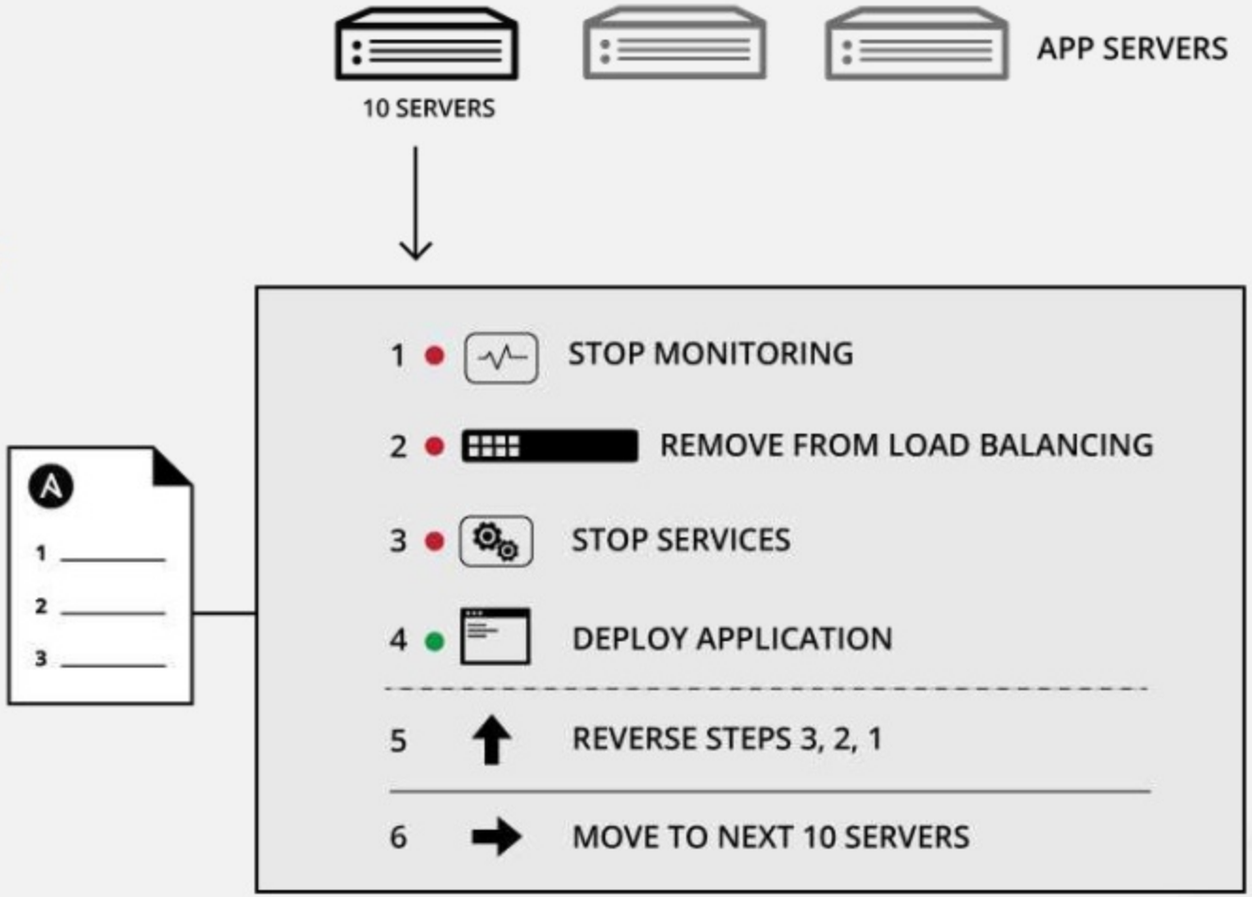


Why Is Automation Important?

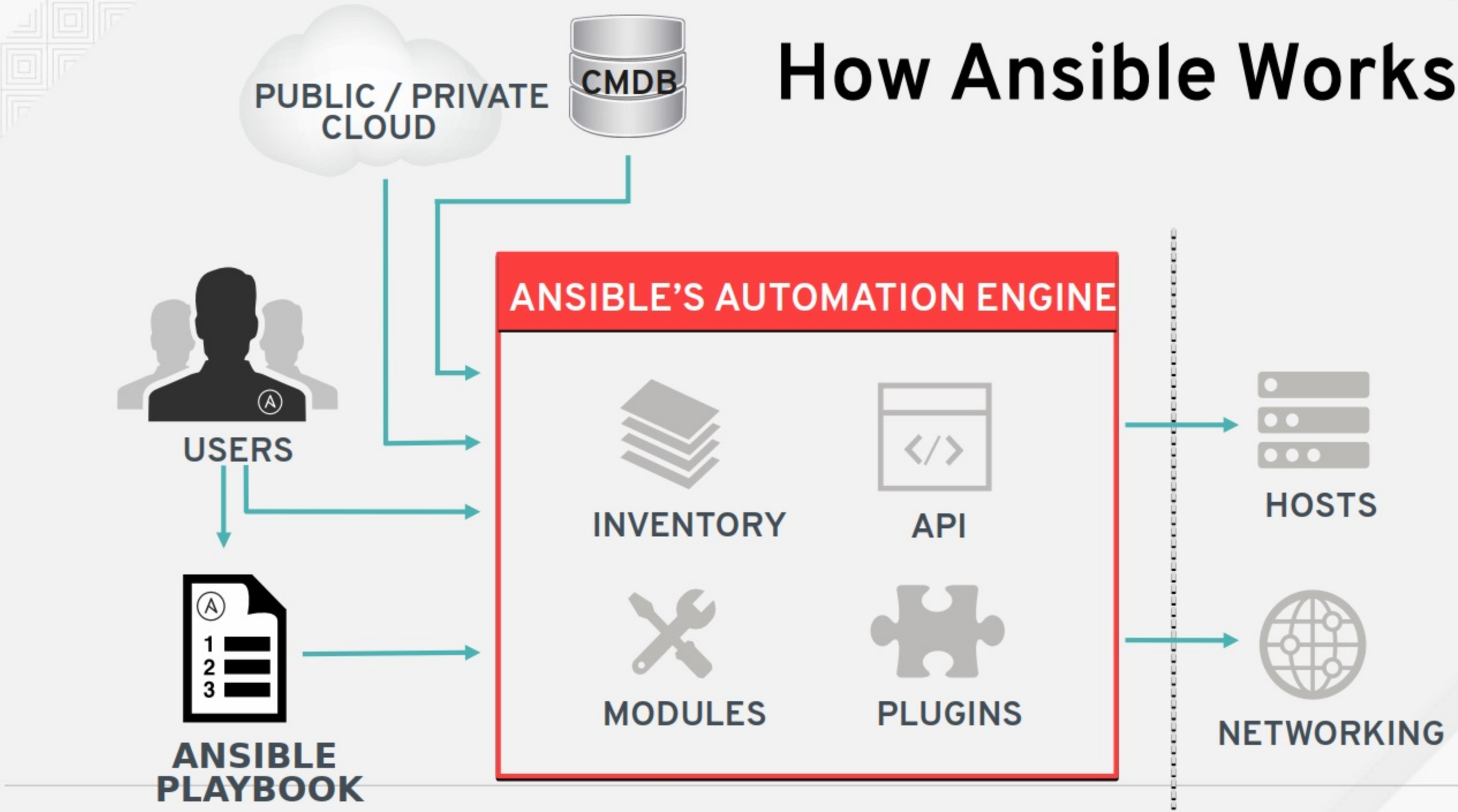
Your applications and systems **are more than just collections of configurations**. They're a finely tuned and **ordered list** of tasks and processes that result in **your working application**.

Ansible can do it all:

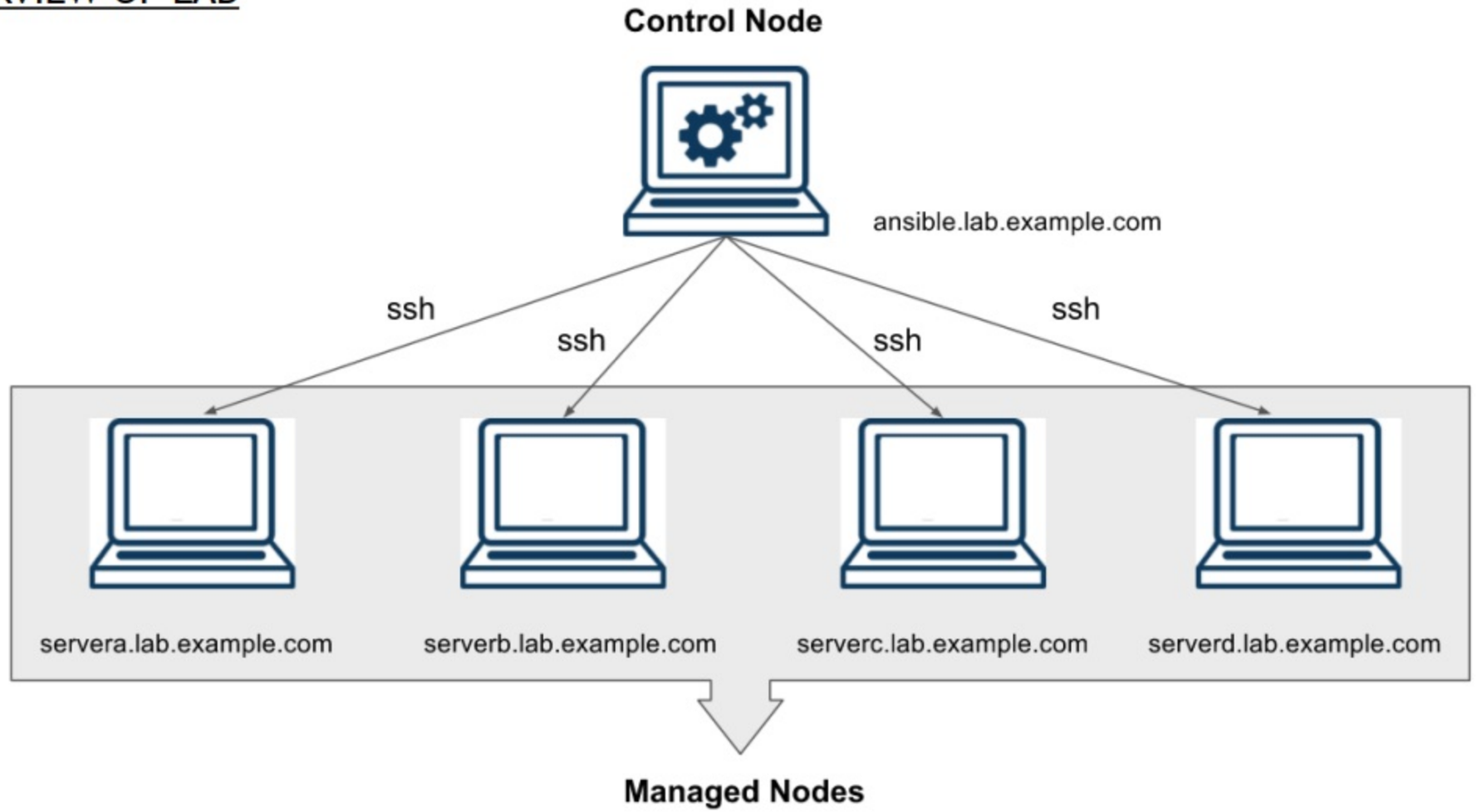
- Provisioning
- App Deployment
- Configuration Management
- Multi-tier Orchestration



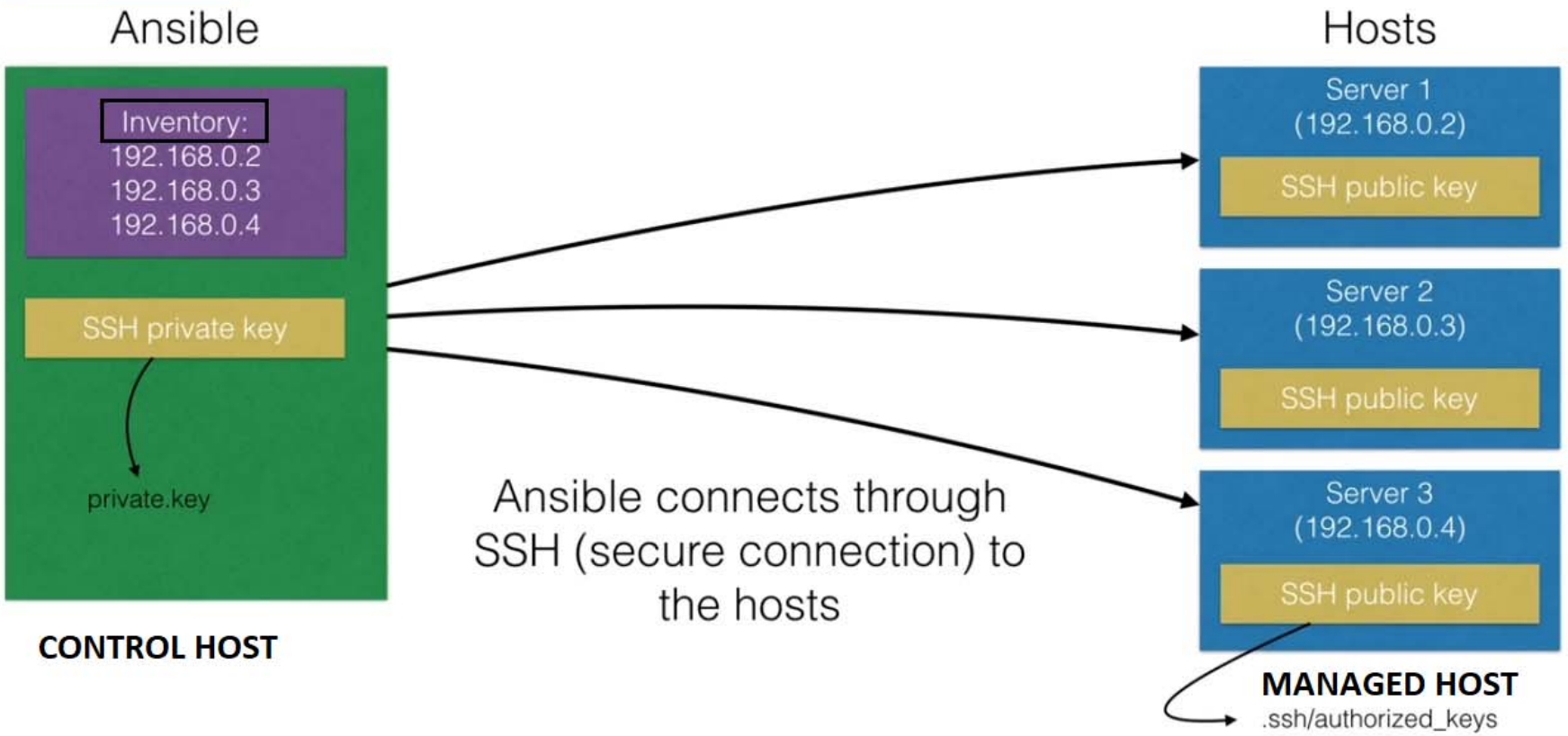
How Ansible Works



OVERVIEW OF LAB



Ansible


Ansible
Hosts
Inventory:
192.168.0.2
192.168.0.3
192.168.0.4
SSH private key
private.key
CONTROL HOST
Ansible connects through SSH (secure connection) to the hosts
Server 1 (192.168.0.2)
SSH public key
Server 2 (192.168.0.3)
SSH public key
Server 3 (192.168.0.4)
SSH public key
MANAGED HOST
.ssh/authorized_keys

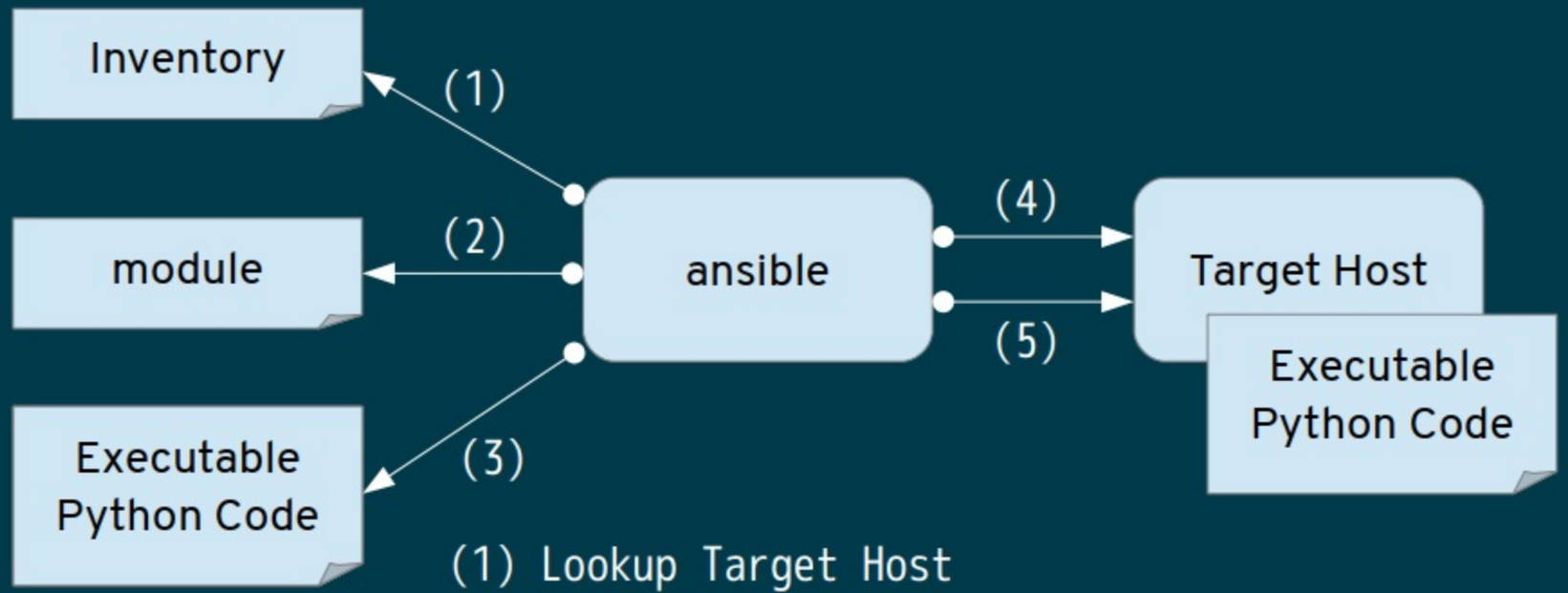
Ansible Core

Ansible Core is command-line IT automation Tool and libraries

Introduce following components of Ansible Core:

1. Command Line Tools
2. Playbooks
3. Inventory
4. Modules
5. Plugins

COMMAND MECHANISM



(1) Lookup Target Host
(2) Read Module
(3) Generate executable code from Module
(4) Copy Executable python code to via SCP
(5) Execute python code on Target Host

PREREQUISITES AND REQUIREMENTS

Hardware Requirements:

- 2 CPUs minimum
- 2 GB RAM minimum (4+ GB RAM recommended)
- 2 GB RAM (recommended)
- 4 GB RAM (recommended)
- 20 GB of dedicated hard disk space for Tower service nodes

Software Requirements:

Supported Operating Systems:

Red Hat Enterprise Linux 7.2 or later 64-bit

CentOS 7.2 or later 64-bit

Ubuntu 14.04 LTS 64-bit

Ubuntu 16.04 LTS 64-bit

Control Node Requirements:

Python 2 (versions 2.6 or 2.7) or Python 3 (versions 3.5 and higher) .

Managed Node Requirements:

Python 2.6 or later.

Key-based authentication (ssh) for communication purpose with control node.

Running Ansible from Command Line

Ad-hoc commands:

```
ansible all -m command -a “uname -a”
ansible webservers -m service -a “name=httpd state=restart”
```

Available modules:

```
ansible-doc -l
ansible-doc yum
```

Running playbooks:

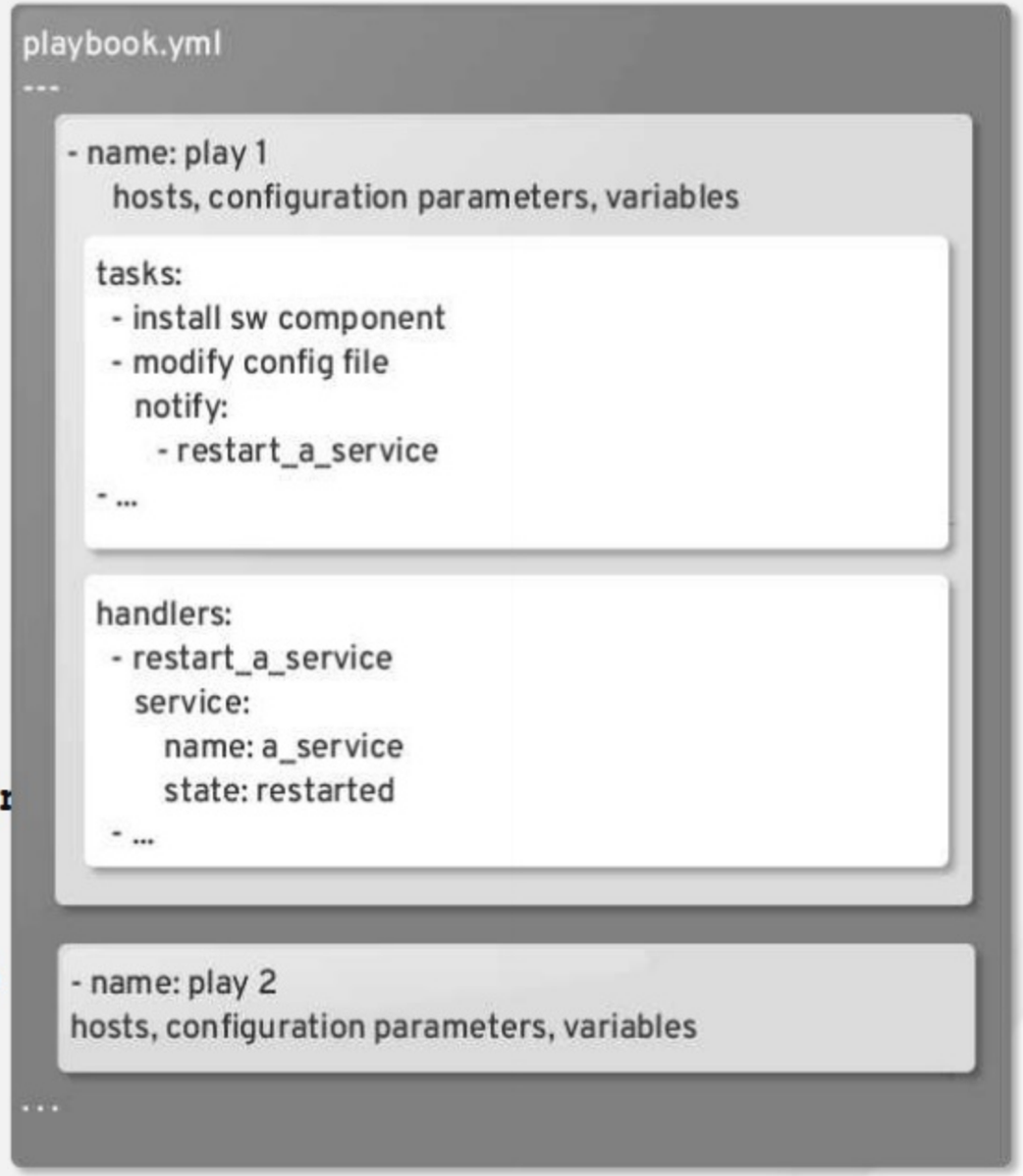
```
ansible-playbook –syntax-check playbook.yml
ansible-playbook playbook.yml -C
ansible-playbook playbook.yml
```

Ansible Playbook

```
---
- name: Update httpd config
  hosts: webservers
  vars:
    http_port: 80
   max_clients: 200
  remote_user: devops
  become: true

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.cor
    notify:
    - restart_httpd
  - name: start httpd
    service: name=httpd state=running enabled=true

  handlers:
  - restart_httpd
      service: name=httpd state=restarted
```



Ansible Modules

Docs » Module Index

Module Index

- All Modules
- Cloud Modules
- Clustering Modules
- Commands Modules
- Database Modules
- Files Modules
- Inventory Modules
- Messaging Modules
- Monitoring Modules
- Network Modules
- Notification Modules
- Packaging Modules
- Source Control Modules
- System Modules
- Utilities Modules
- Web Infrastructure Modules
- Windows Modules

service - Manage services.

- Synopsis
- Options
- Examples
- This is a Core Module

Synopsis

Controls services on remote hosts. Supported init systems include BSD init, OpenRC, SysV, Solaris SMF, systemd, upstart.

Options

parameter	required	default	choices	comments
arguments	no			Additional arguments provided on the command line aliases: args
enabled	no		• yes • no	Whether the service should start on boot. **At least one of state and enabled are required.**
name	yes			Name of the service.
pattern	no			If the service does not respond to the status command, name a substring to look for as would be found in the output of the *ps* command as a stand-in for a status result. If the string is found, the service will be assumed to be running.
runlevel	no	default		For OpenRC init scripts (ex: Gentoo) only. The runlevel that this service belongs to.
sleep (added in 1.3)	no			If the service is being `restarted` then sleep this many seconds between the stop and start command. This helps to workaround badly behaving init scripts that exit immediately after signaling a process to stop.
state	no		• started • stopped • restarted • reloaded	`started` / `stopped` are idempotent actions that will not run commands unless necessary. `restarted` will always bounce the service. `reloaded` will always reload. **At least one of state and enabled are required.**

Automation For Teams

redhat.

Automation with Ansible
DO407

Thank you