ANSIBLE

AUTOMATION FOR EVERYONE

Josh Preston
Solutions Architect
Red Hat, Inc.

OPENSHIFT ROADSHOW 2016

JUNE 8, 2016
COLUMBUS
HILTON @ POLARIS

JUNE 9, 2016
CINCINNATI
RESIDENCE INN
DOWNTOWN

RED HAT SUMMIT

June 27 to 30, 2016
San Francisco, CA

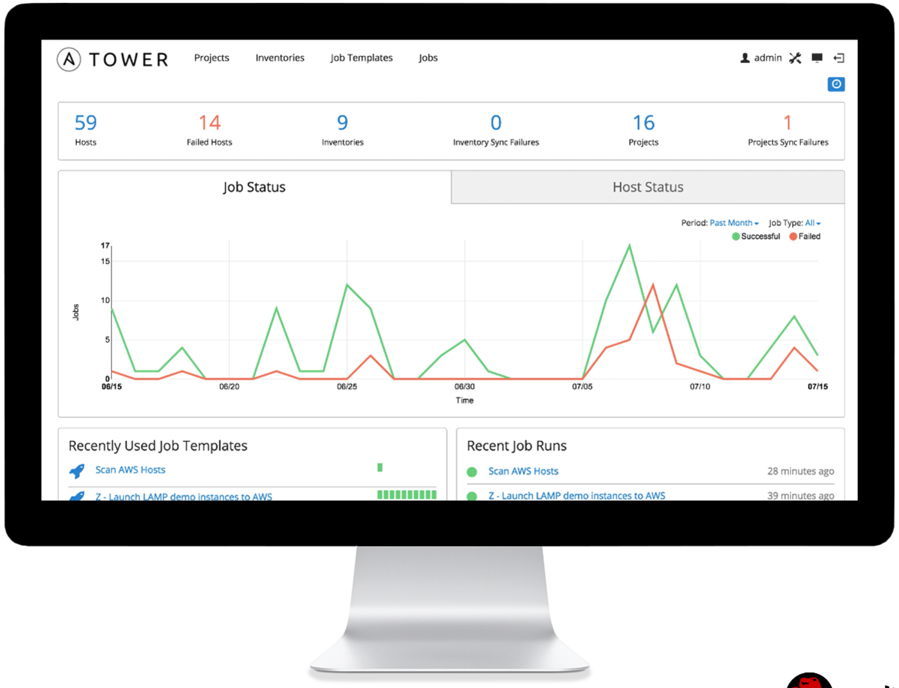
Register TODAY!

WHAT IS ANSIBLE?

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

It’s an **automation engine** that runs Ansible Playbooks.

Ansible Tower is an **enterprise framework** for controlling, securing and managing your Ansible automation with a **UI and restful API.**



59 Hosts | 14 Failed Hosts | 9 Inventories | 0 Inventory Sync Failures | 16 Projects | 1 Projects Sync Failures

Recently Used Job Templates

- Scan AWS Hosts
- Z - Launch LAMP demo instances to AWS

Recent Job Runs

- ● Scan AWS Hosts — 28 minutes ago
- ● Z - Launch LAMP demo instances to AWS — 39 minutes ago

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

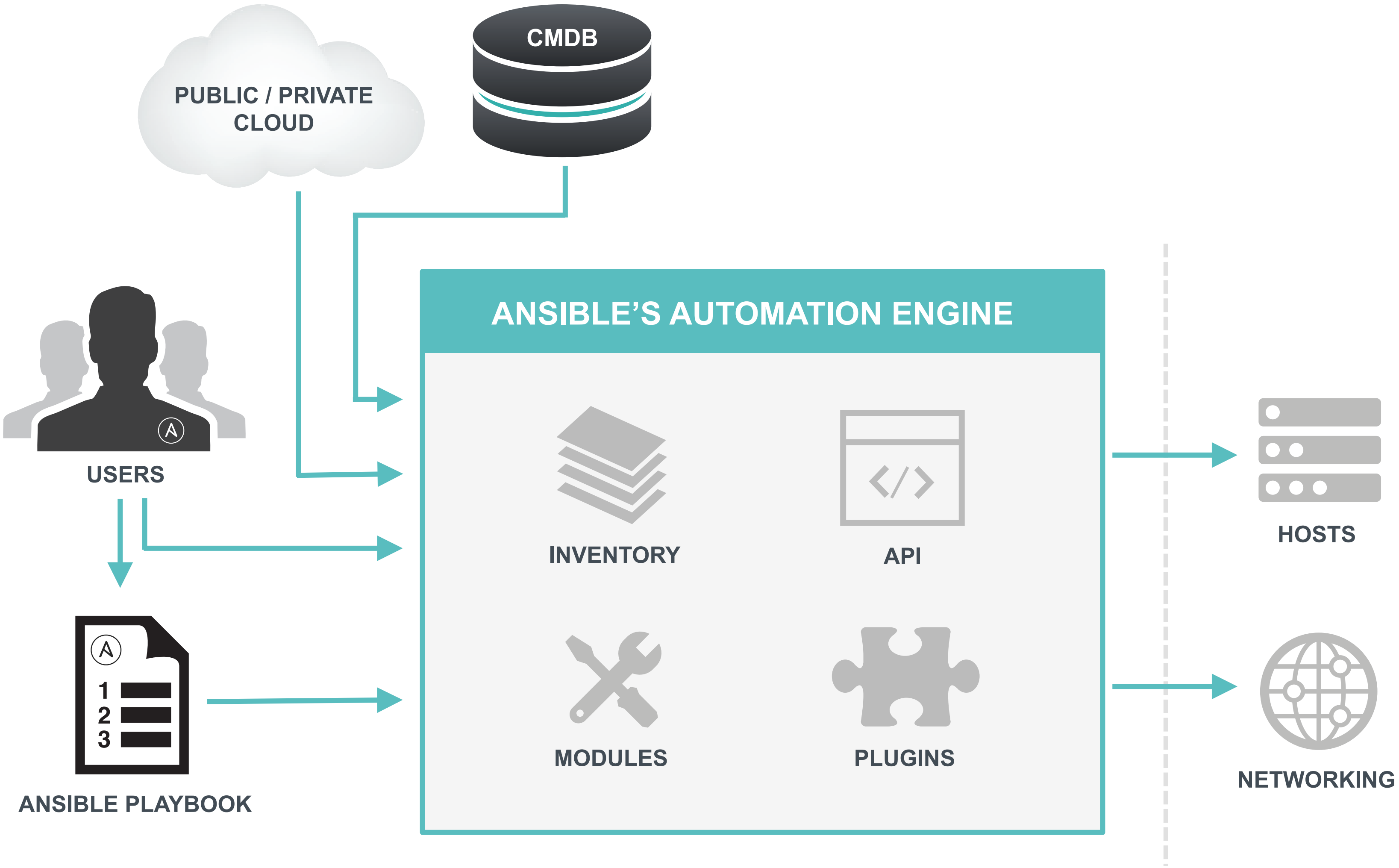
Agentless architecture

Uses OpenSSH & WinRM

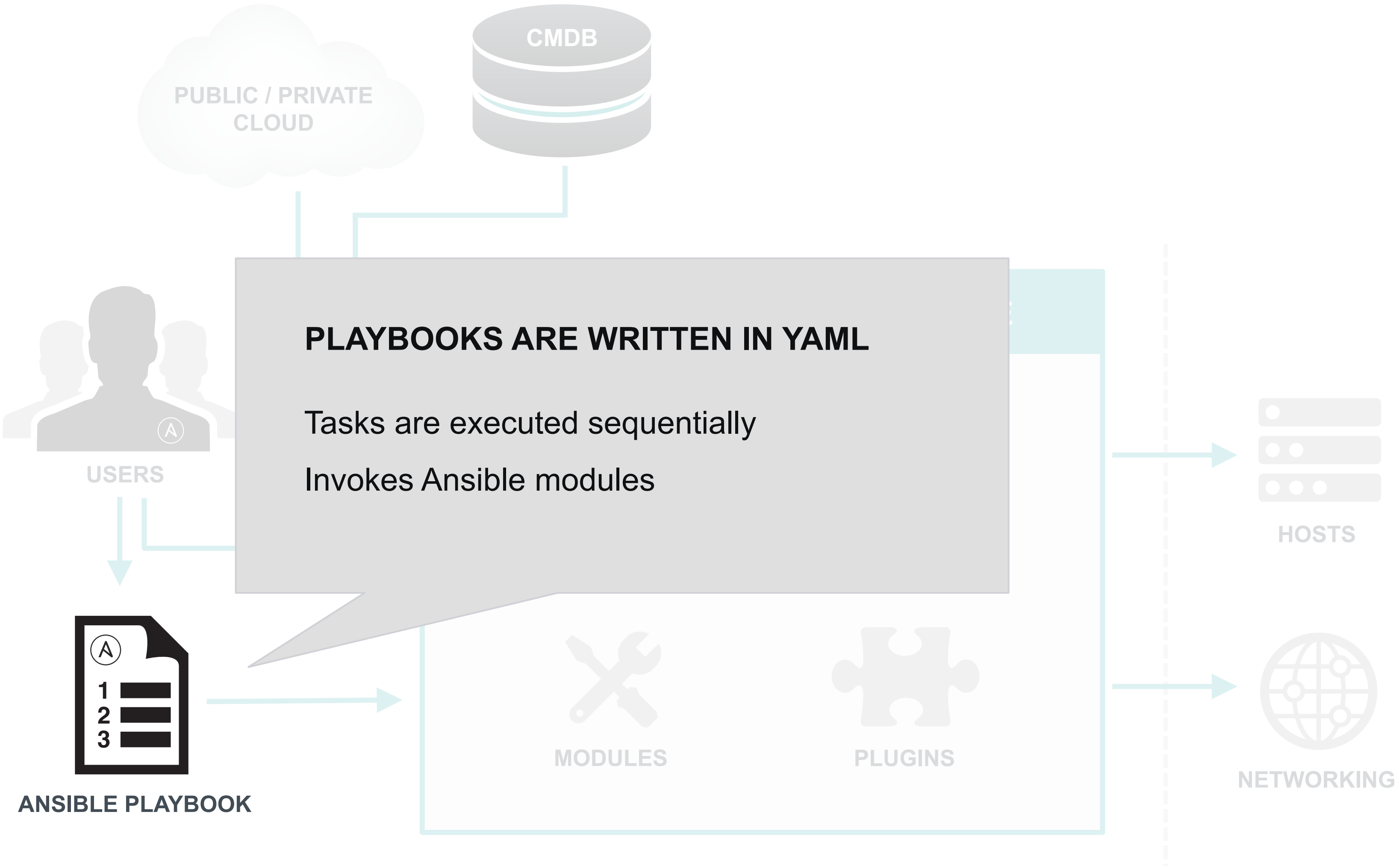
No agents to exploit or update

More efficient & more secure

HOW ANSIBLE WORKS



HOW ANSIBLE WORKS

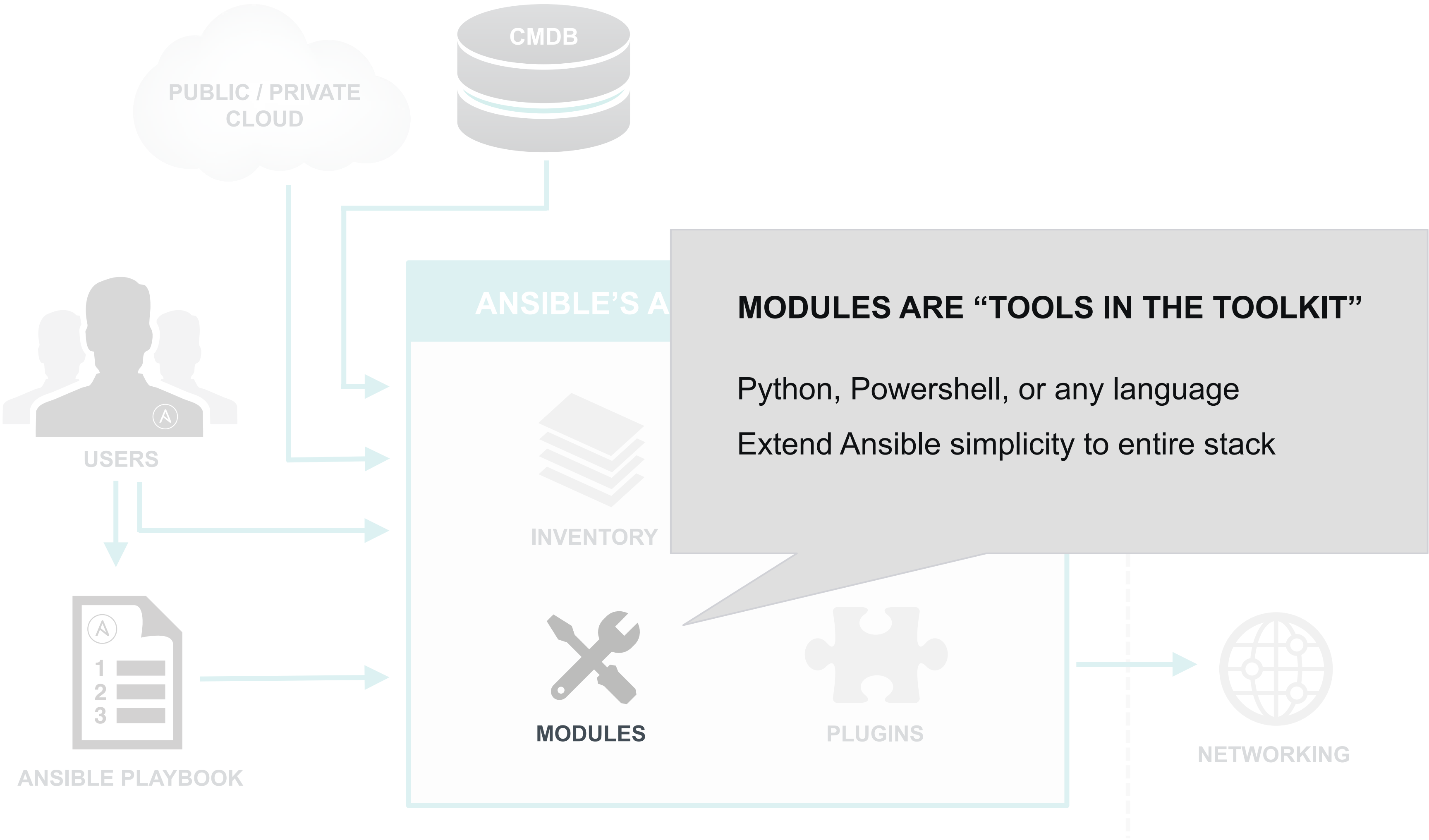


PLAYBOOKS ARE WRITTEN IN YAML

Tasks are executed sequentially

Invokes Ansible modules

HOW ANSIBLE WORKS



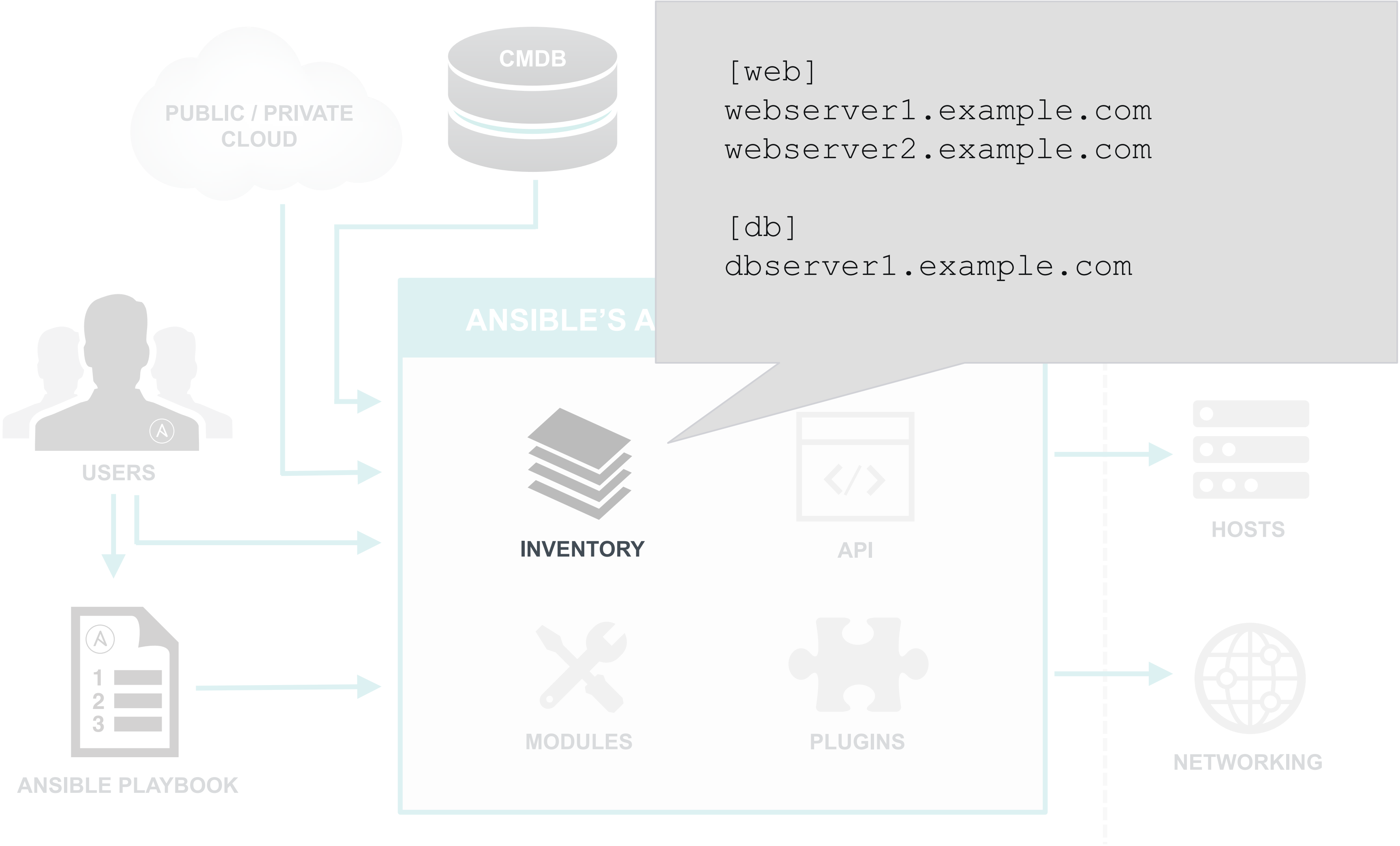
MODULES ARE "TOOLS IN THE TOOLKIT"

Python, Powershell, or any language

Extend Ansible simplicity to entire stack

HOW ANSIBLE WORKS

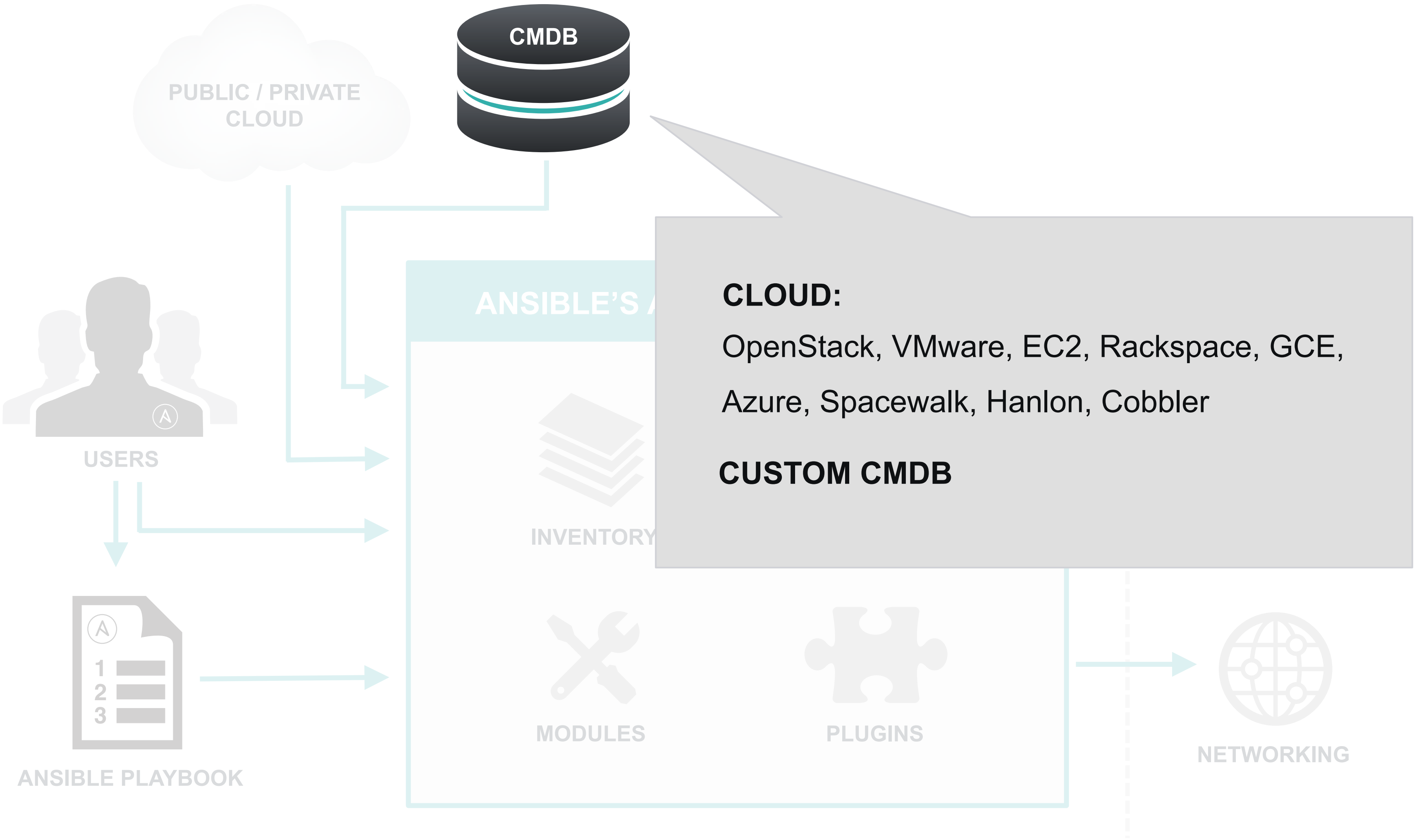
ANSIBLE



```
[web]
webserver1.example.com
webserver2.example.com

[db]
dbserver1.example.com
```

HOW ANSIBLE WORKS



CLOUD:

OpenStack, VMware, EC2, Rackspace, GCE, Azure, Spacewalk, Hanlon, Cobbler

CUSTOM CMDB

PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

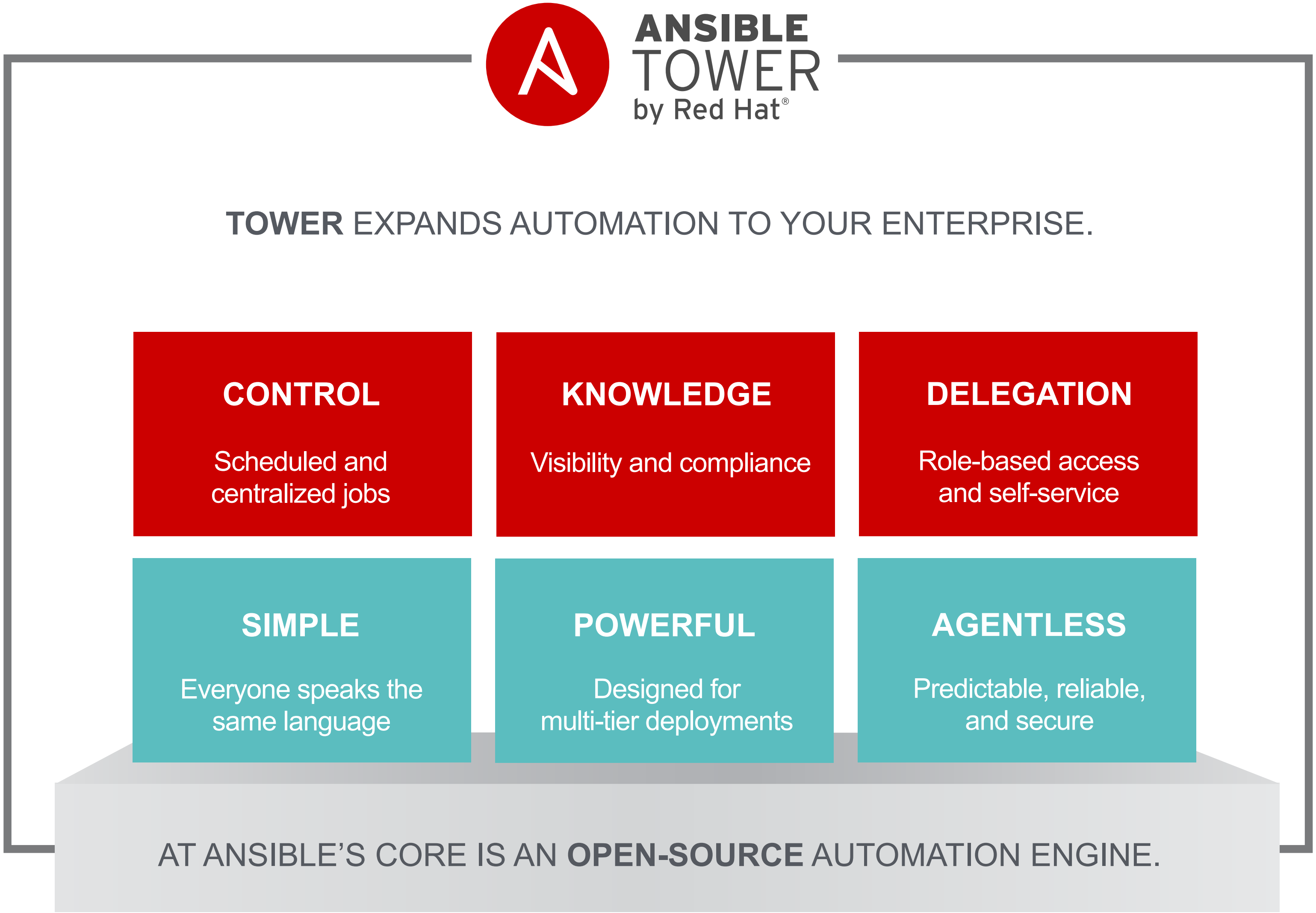
PLAYBOOK EXAMPLE

```
---
- name: install and start apache
  hosts: all
  vars:
    http_port: 80
    max_clients: 200
  remote_user: root

  tasks:
  - name: install httpd
    yum: pkg=httpd state=latest
  - name: write the apache config file
    template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  - name: start httpd
    service: name=httpd state=running
```

ANSIBLE TOWER

ANSIBLE



ANSIBLE TOWER by Red Hat®

TOWER EXPANDS AUTOMATION TO YOUR ENTERPRISE.

CONTROL

Scheduled and centralized jobs

KNOWLEDGE

Visibility and compliance

DELEGATION

Role-based access and self-service

SIMPLE

Everyone speaks the same language

POWERFUL

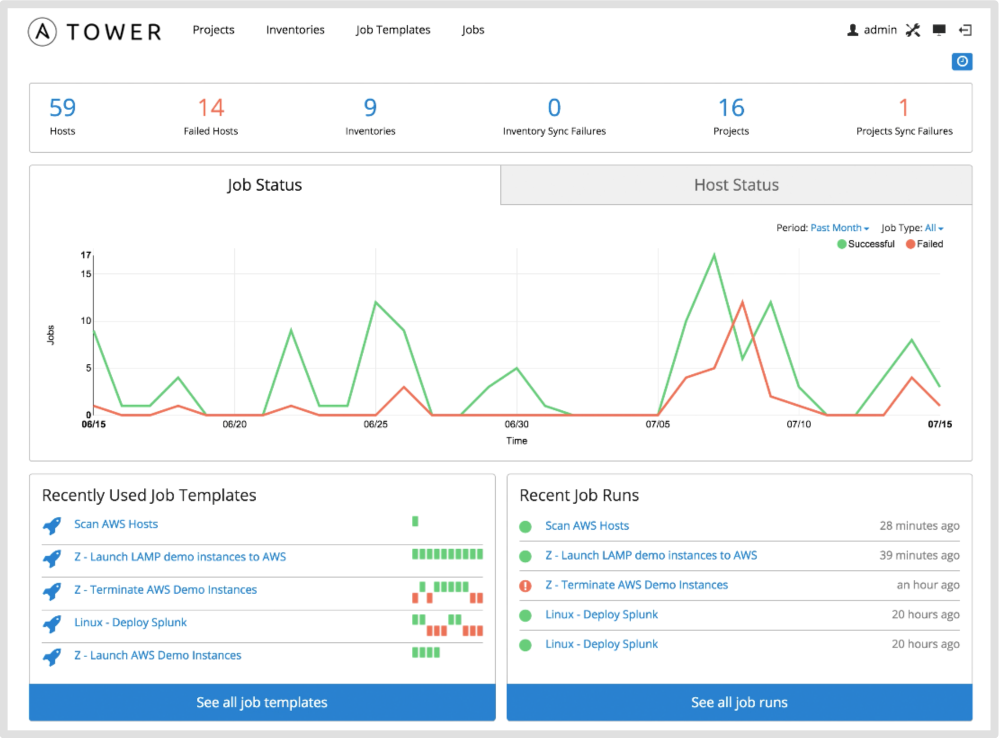
Designed for multi-tier deployments

AGENTLESS

Predictable, reliable, and secure

AT ANSIBLE'S CORE IS AN **OPEN-SOURCE** AUTOMATION ENGINE.

WHAT IS ANSIBLE TOWER?



59	14	9	0	16	1
Hosts	Failed Hosts	Inventories	Inventory Sync Failures	Projects	Projects Sync Failures

Recently Used Job Templates

- Scan AWS Hosts
- Z - Launch LAMP demo instances to AWS
- Z - Terminate AWS Demo Instances
- Linux - Deploy Splunk
- Z - Launch AWS Demo Instances

See all job templates

Recent Job Runs

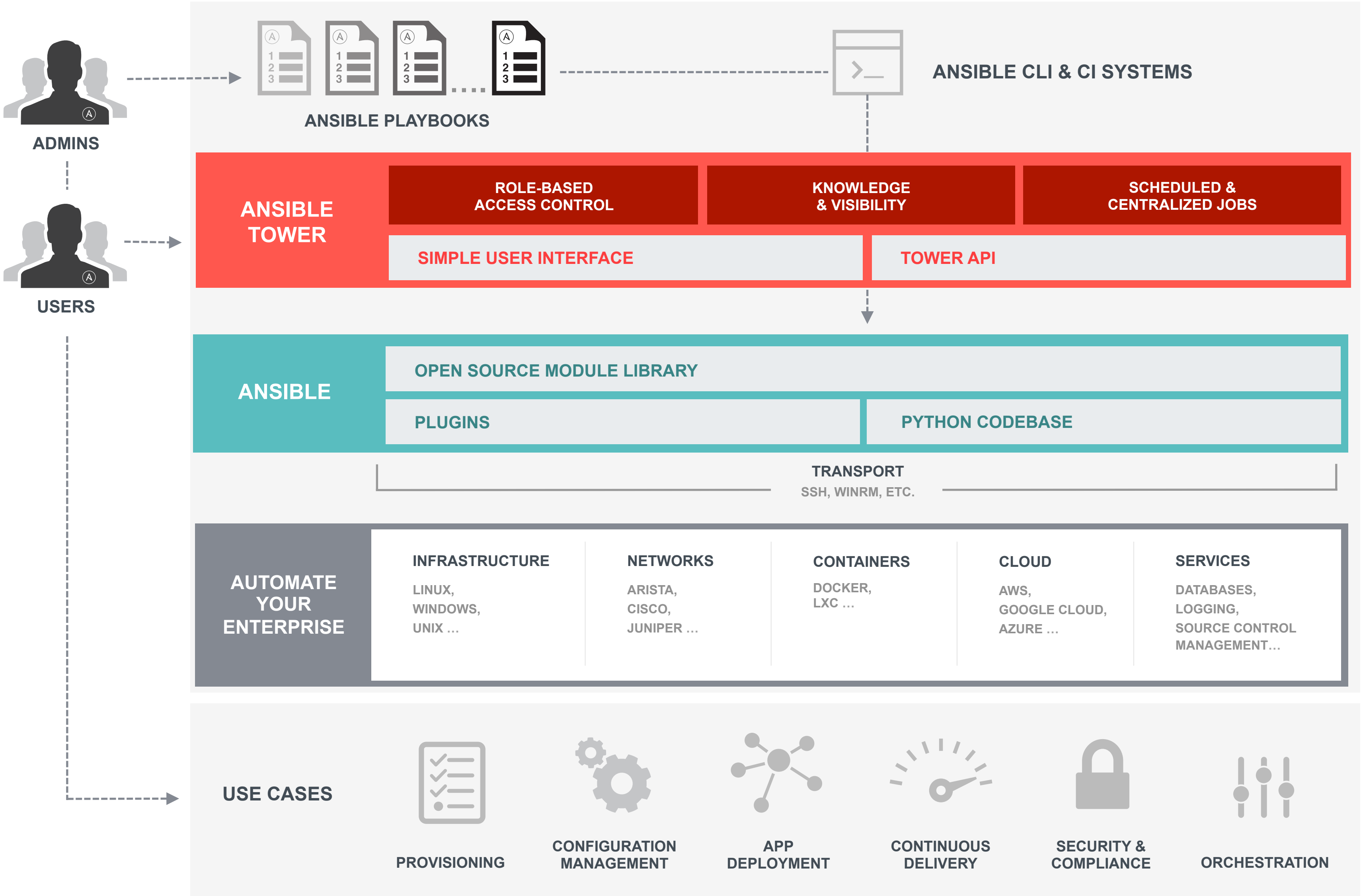
Scan AWS Hosts	28 minutes ago
Z - Launch LAMP demo instances to AWS	39 minutes ago
Z - Terminate AWS Demo Instances	an hour ago
Linux - Deploy Splunk	20 hours ago
Linux - Deploy Splunk	20 hours ago

See all job runs

Ansible tower is an **enterprise framework** for controlling, securing and managing your Ansible automation – with a **UI and restful API.**

- **Role-based access control** keeps environments secure, and teams efficient.
- Non-privileged users can **safely deploy** entire applications with **push-button deployment** access.
- All Ansible automations are **centrally logged,** ensuring **complete auditability and compliance**.

PLATFORM OVERVIEW



ADMINS

USERS

ANSIBLE PLAYBOOKS

ANSIBLE CLI & CI SYSTEMS

ANSIBLE TOWER

ROLE-BASED ACCESS CONTROL

KNOWLEDGE & VISIBILITY

SCHEDULED & CENTRALIZED JOBS

SIMPLE USER INTERFACE

TOWER API

ANSIBLE

OPEN SOURCE MODULE LIBRARY

PLUGINS

PYTHON CODEBASE

TRANSPORT

SSH, WINRM, ETC.

AUTOMATE YOUR ENTERPRISE

INFRASTRUCTURE	NETWORKS	CONTAINERS	CLOUD	SERVICES
LINUX, WINDOWS, UNIX …	ARISTA, CISCO, JUNIPER …	DOCKER, LXC …	AWS, GOOGLE CLOUD, AZURE …	DATABASES, LOGGING, SOURCE CONTROL MANAGEMENT…

USE CASES

PROVISIONING

CONFIGURATION MANAGEMENT

APP DEPLOYMENT

CONTINUOUS DELIVERY

SECURITY & COMPLIANCE

ORCHESTRATION

USE CASES

CONFIG MANAGEMENT

Centralizing configuration file management and deployment is a common use case for Ansible, and it's how many power users are first introduced to the Ansible automation platform.

APP DEPLOYMENT

When you define your application with Ansible, and manage the deployment with Tower, teams are able to effectively manage the entire application lifecycle from development to production.

PROVISIONING

Your apps have to live somewhere. If you're PXE booting and kickstarting bare-metal servers or VMs, or creating virtual or cloud instances from templates, Ansible and Ansible Tower help streamline the process.

CONTINUOUS DELIVERY

Creating a CI/CD pipeline requires buy-in from numerous teams. You can't do it without a simple automation platform that everyone in your organization can use. Ansible Playbooks keep your applications properly deployed (and managed) throughout their entire lifecycle.

SECURITY & COMPLIANCE

When you define your security policy in Ansible, scanning and remediation of site-wide security policy can be integrated into other automated processes and instead of being an afterthought, it'll be integral in everything that is deployed.

ORCHESTRATION

Configurations alone don't define your environment. You need to define how multiple configurations interact and ensure the disparate pieces can be managed as a whole. Out of complexity and chaos, Ansible brings order.

SELECTED ANSIBLE TOWER CUSTOMERS

ANSIBLE

SONOS
verizon✓
EA
allegiant
vmware

NASA
splunk>
Care.com
SONY
GoPro
JUNIPer NETWORKS

rackspace
NetApp
SAMSUNG
Bank of America
NEC
GRAINGER

EURONEXT
weight watchers
JPMORGAN CHASE & CO.
Google
J.CREW
U.S. DEPARTMENT OF HOMELAND SECURITY

DARDEN
bp
MARTHA STEWART
tbs
SAVE MART SUPERMARKETS
AMERICAN EAGLE OUTFITTERS

AUTOMATION = ACCELERATION

J.CREW

“With Ansible Tower, **we just click a button and deploy to production in 5 minutes**. It used to take us 5 hours with 6 people sitting in a room, making sure we didn’t do anything wrong (and we usually still had errors). We now deploy to production every other day instead of every 2 weeks, and nobody has to be up at 4am making sure it was done right.”

NEC

“By using Ansible Tower Surveys, we have created a self-service capability that allows our IT guys to provision new cloud customers quickly. Our first 3 customers each took 2 weeks to provision. With Ansible, **our next 500 customers took 10 minutes each to provision.”**

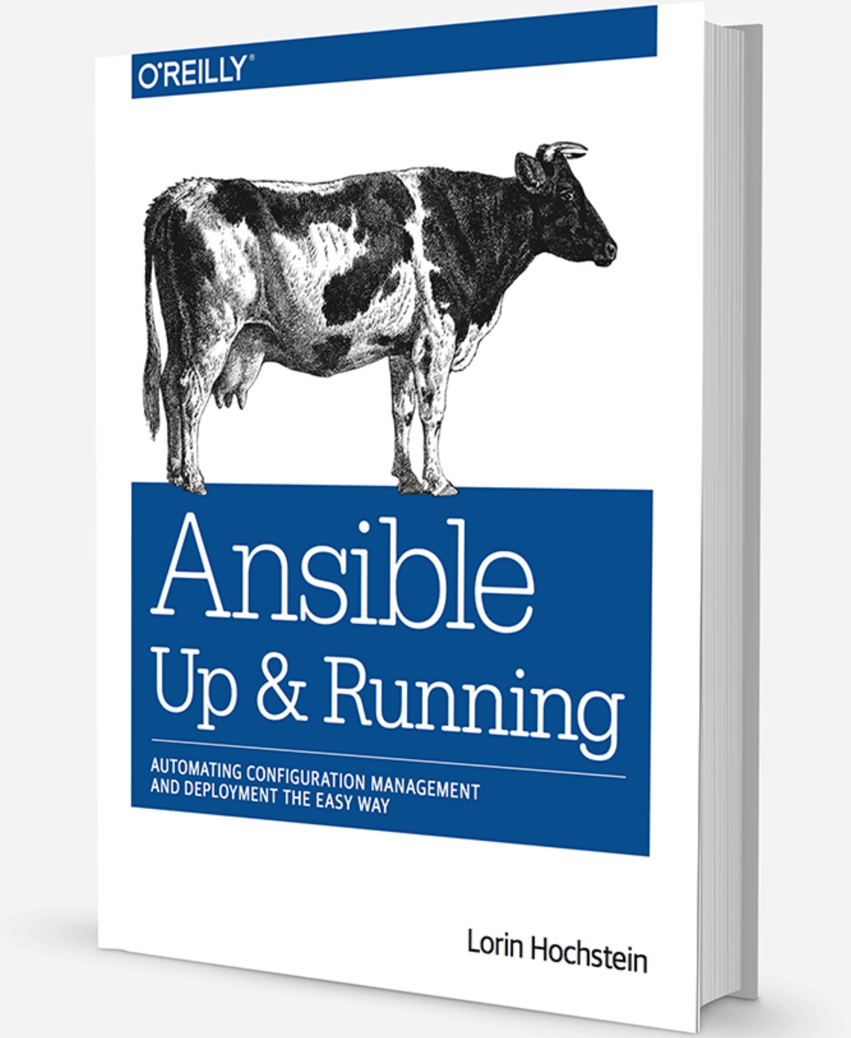
RIOT GAMES

“We use Ansible to deploy the network configurations to new datacenters for our games. Previously, it would take 4 days worth of work, onsite, to make sure that our networks were configured correctly. **With Ansible, I can now configure a datacenter in 4 minutes, remotely.**”

COMMUNITY

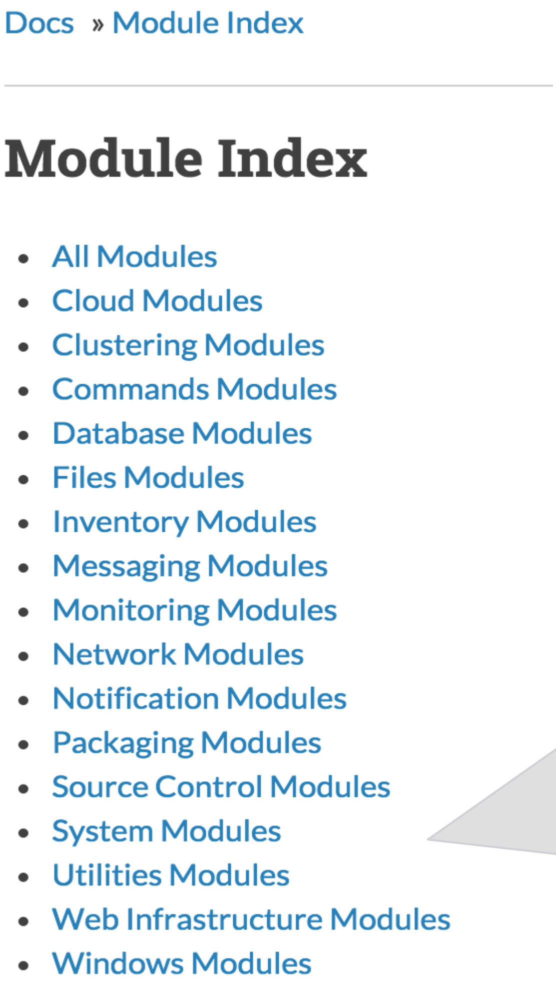
THE MOST POPULAR OPEN-SOURCE AUTOMATION COMMUNITY ON GITHUB

- 13,000+ stars & 4,000+ forks on GitHub
- 2000+ GitHub Contributors
- Over 450 modules shipped with Ansible
- New contributors added every day
- 1200+ users on IRC channel
- Top 10 open source projects in 2014
- World-wide meetups taking place every week
- Ansible Galaxy: over 18,000 subscribers
- 250,000+ downloads a month
- AnsibleFests in NYC, SF, London


O'REILLY®
Ansible
Up & Running
AUTOMATING CONFIGURATION MANAGEMENT
AND DEPLOYMENT THE EASY WAY
Lorin Hochstein

MODULES

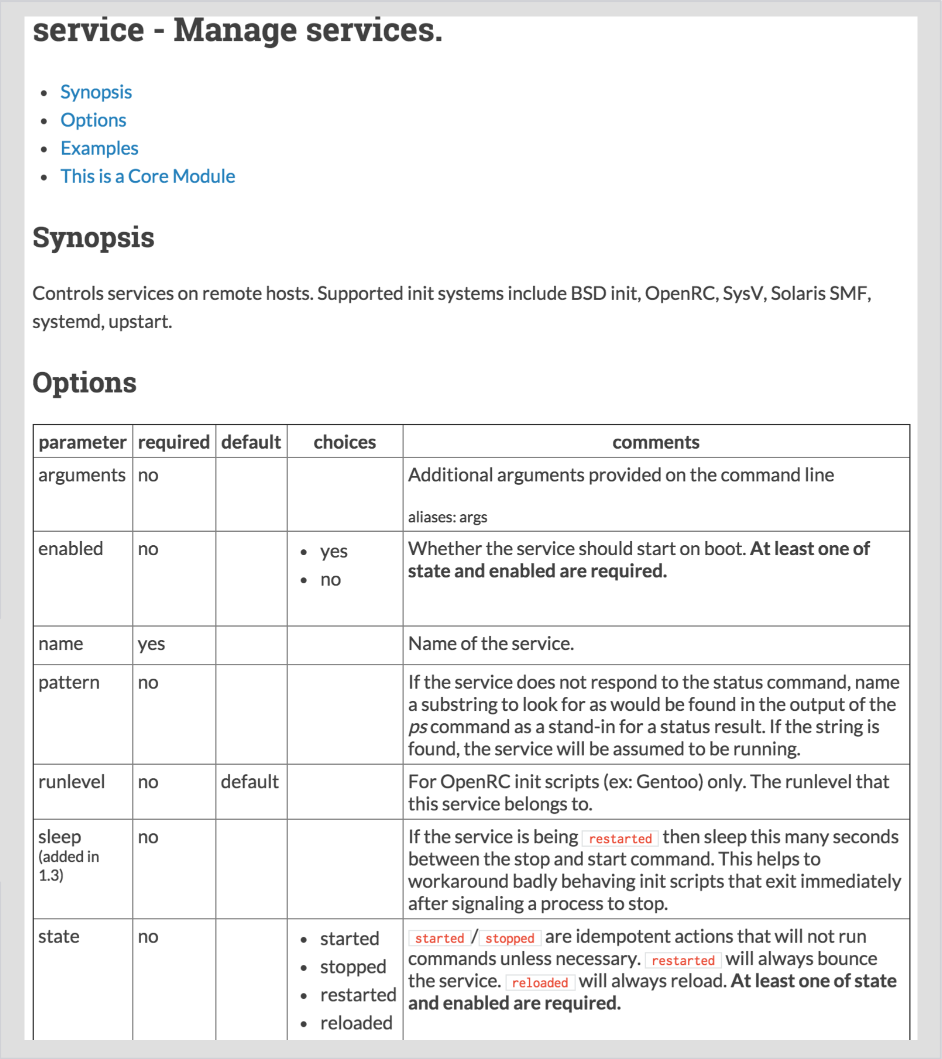
ANSIBLE



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.**

THE ANSIBLE WAY

CROSS PLATFORM – Linux, Windows, UNIX

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

HUMAN READABLE – YAML

Perfectly describe and document every aspect of your application environment

PERFECT DESCRIPTION OF APPLICATION

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.

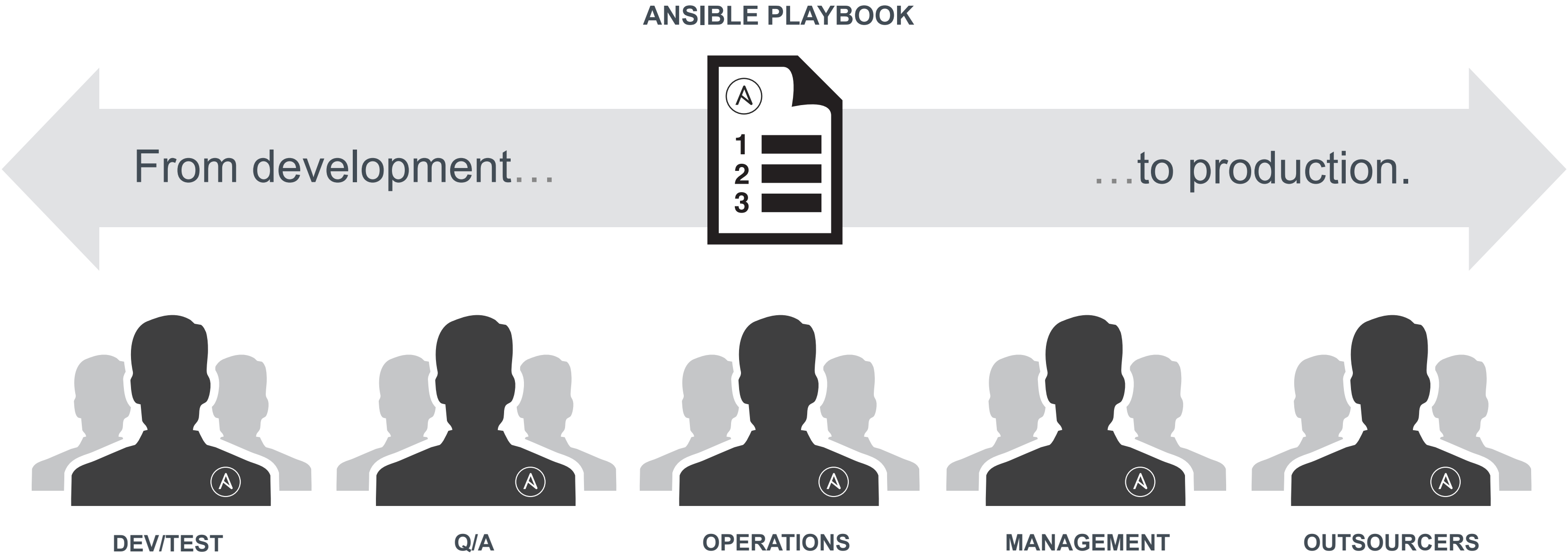
DYNAMIC INVENTORIES

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

ORCHESTRATION THAT PLAYS WELL WITH OTHERS – HP SA, Puppet, Jenkins, RHNSS, etc.

Homogenize existing environments by leveraging current toolsets and update mechanisms.

ANSIBLE: THE LANGUAGE OF DEVOPS

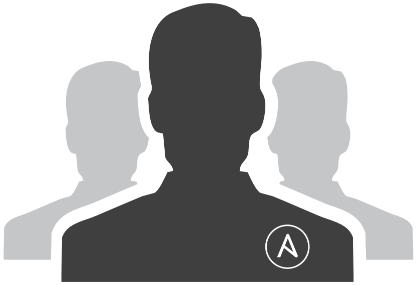


COMMUNICATION IS THE KEY TO DEVOPS.

Ansible is the first **automation language** that can be read and written across IT.

Ansible is the only **automation engine** that can automate the entire **application lifecycle** and **continuous delivery** pipeline.

ANSIBLE: ENTERPRISE IMPACT

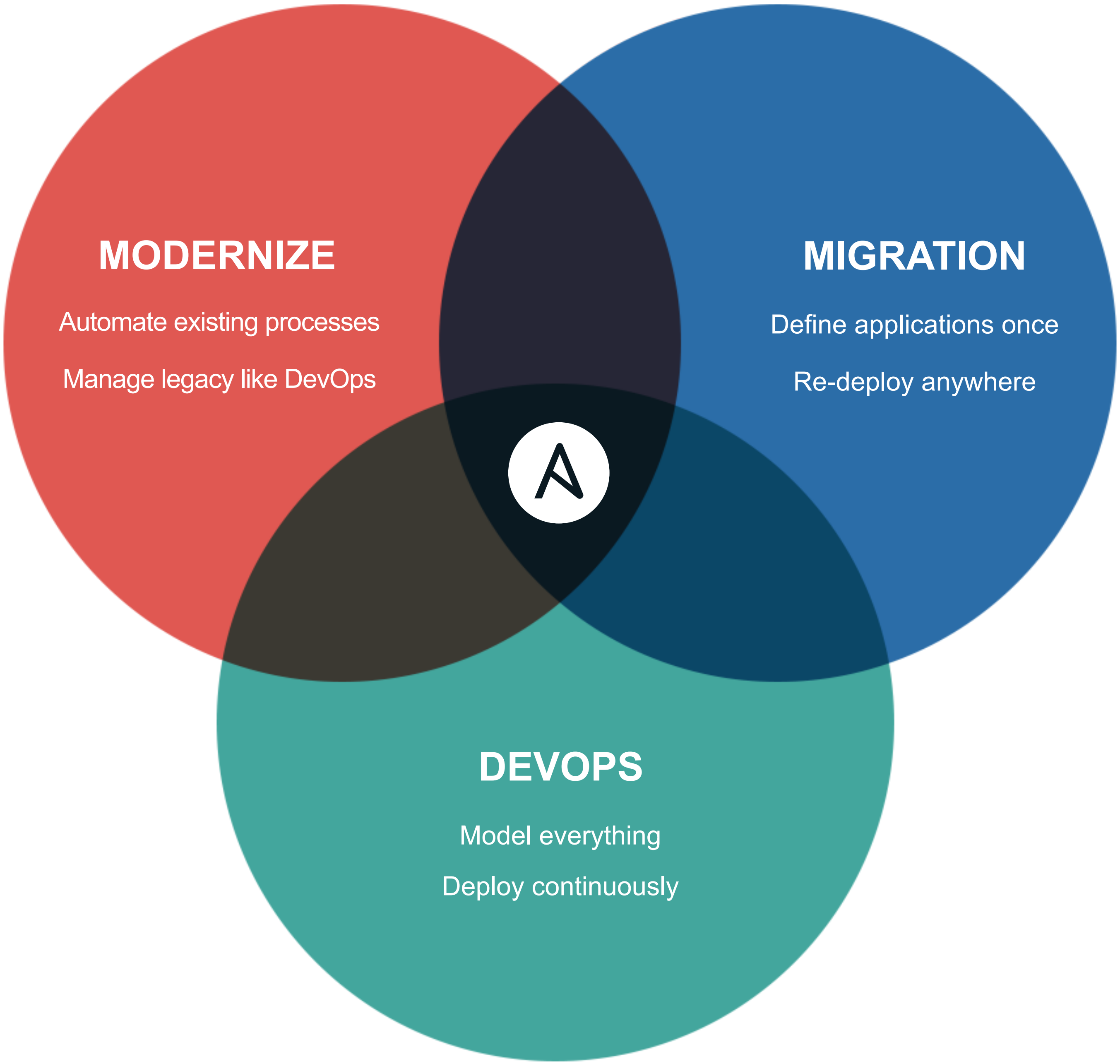


TEAM IMPACT

- Save time and be more productive
- Eliminate repetitive tasks
- Fewer mistakes & errors
- Improve collaboration and job satisfaction

ENTERPRISE IMPACT

- Overcome complexity
- More resources for innovation
- Increase accountability and compliance
- A culture of success



MODERNIZE

Automate existing processes

Manage legacy like DevOps

MIGRATION

Define applications once

Re-deploy anywhere

DEVOPS

Model everything

Deploy continuously

PLAYBOOK EXAMPLES

LAMP + HA Proxy + Nagios:

https://github.com/ansible/ansible-examples/tree/master/lamp_haproxy

JBoss Application Server:

https://github.com/ansible/ansible-examples/tree/master/jboss-standalone

RHEL DISA STIG Compliance:

http://www.ansible.com/security-stig

Many more examples at:

- http://galaxy.ansible.com
- https://github.com/ansible/ansible-examples

ANSIBLE

GETTING STARTED

Have you used Ansible already? Try Tower for free:
ansible.com/tower-trial

Would you like to learn Ansible? It's easy to get started:
ansible.com/get-started

Want to learn more?
ansible.com/whitepapers