ANSIBLE

AUTOMATION FOR EVERYONE

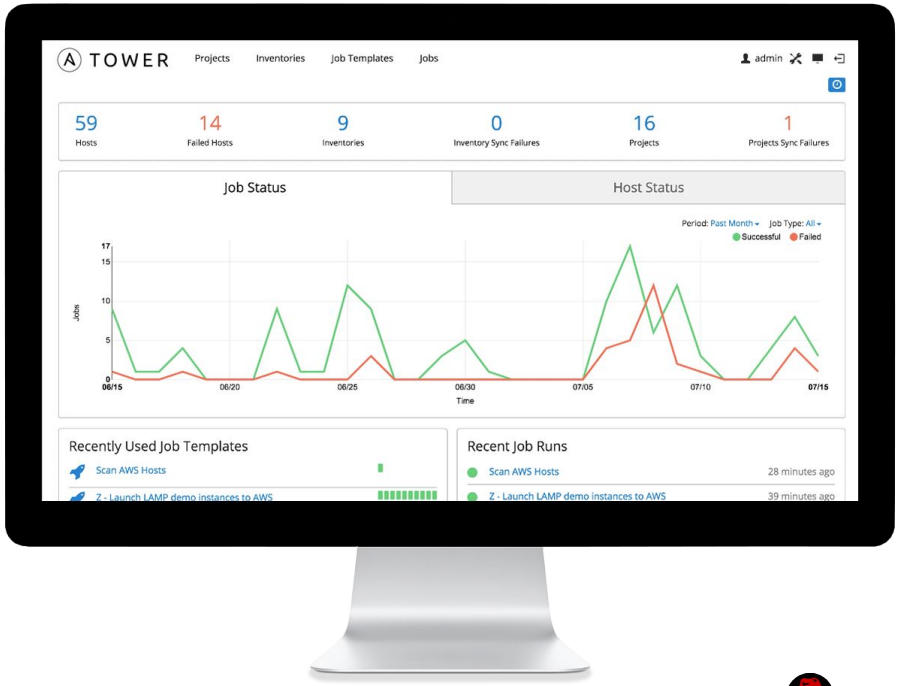
redhat

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.**



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

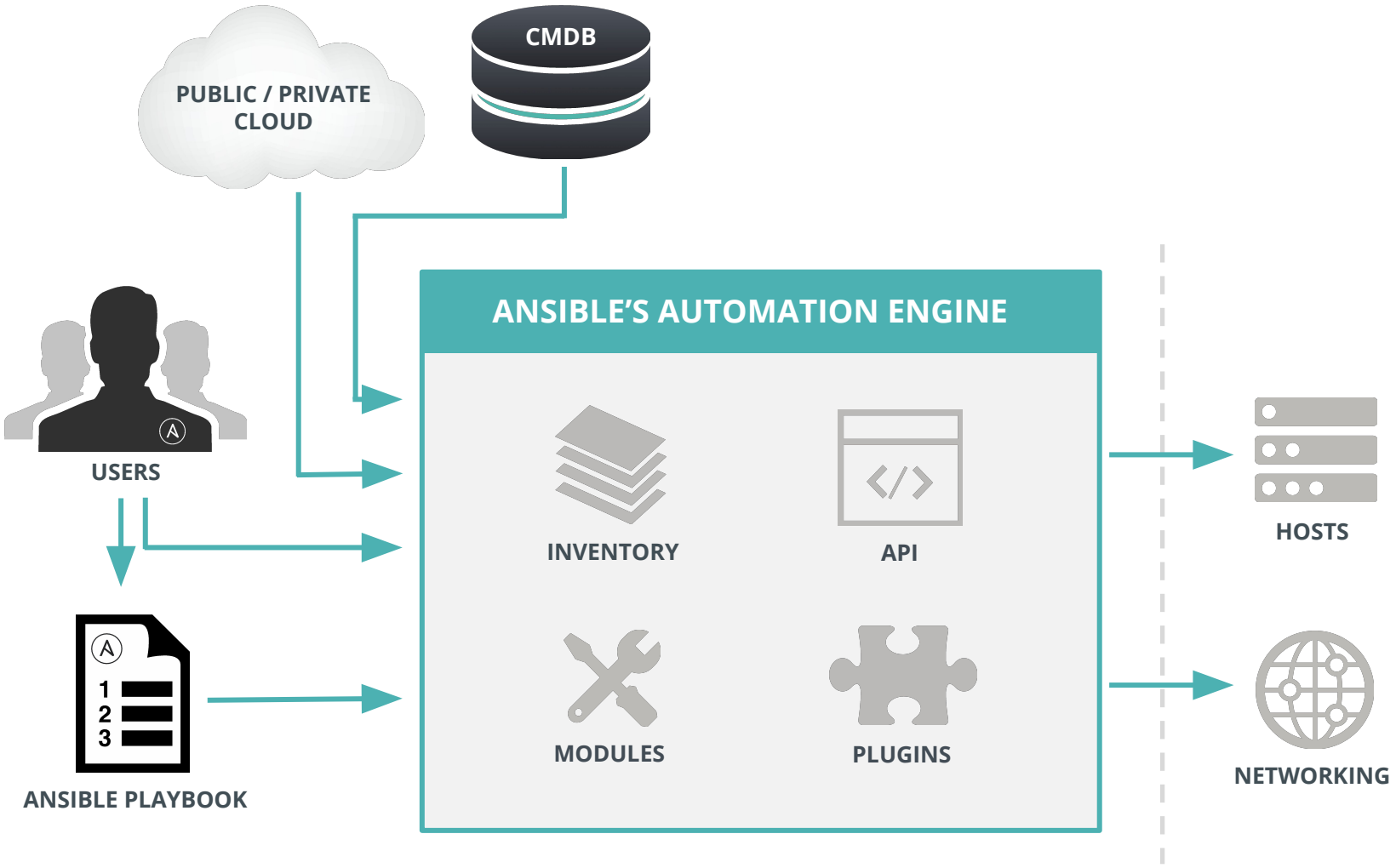
Agentless architecture

Uses OpenSSH & WinRM

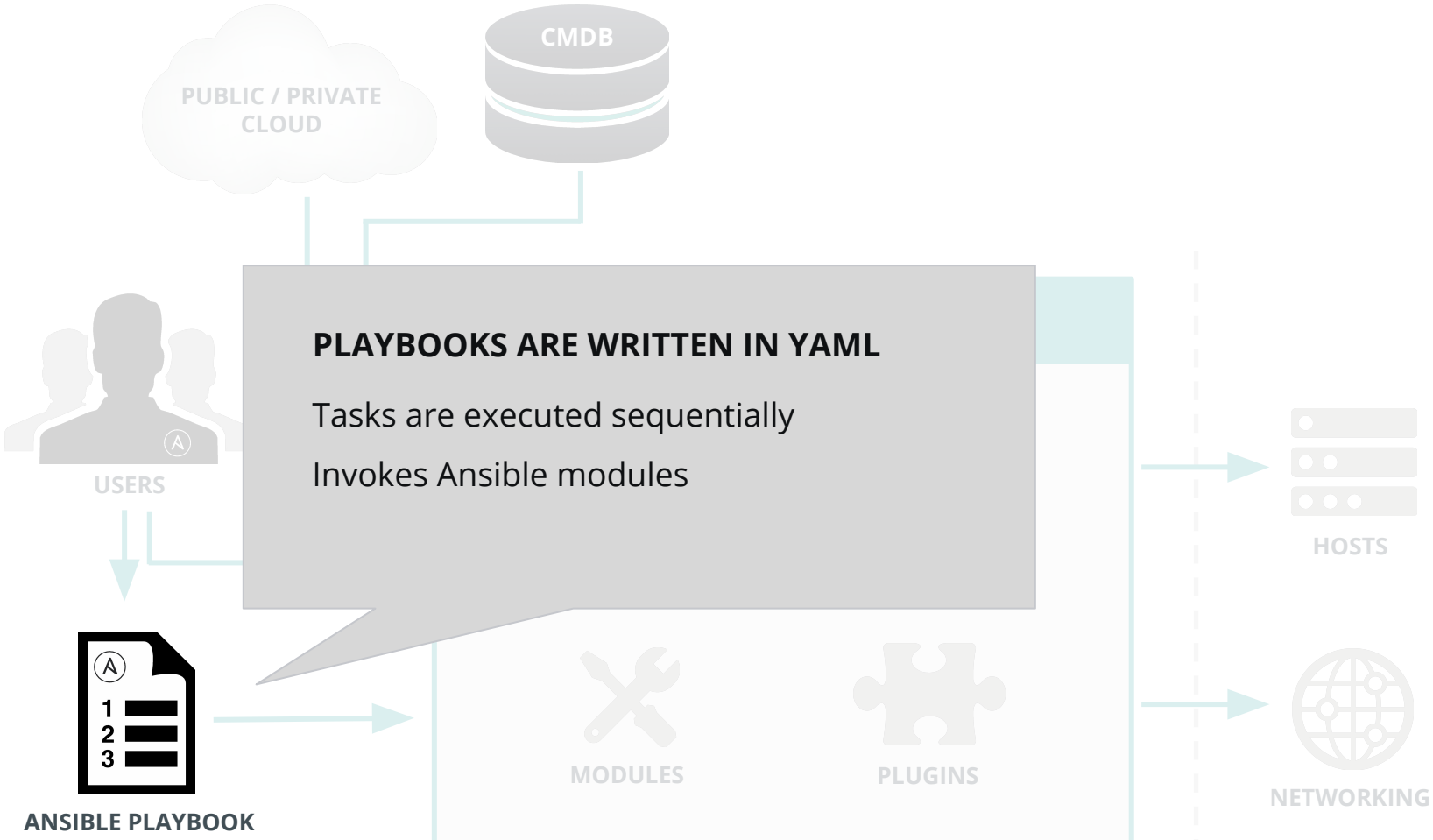
No agents to exploit or update

More efficient & more secure

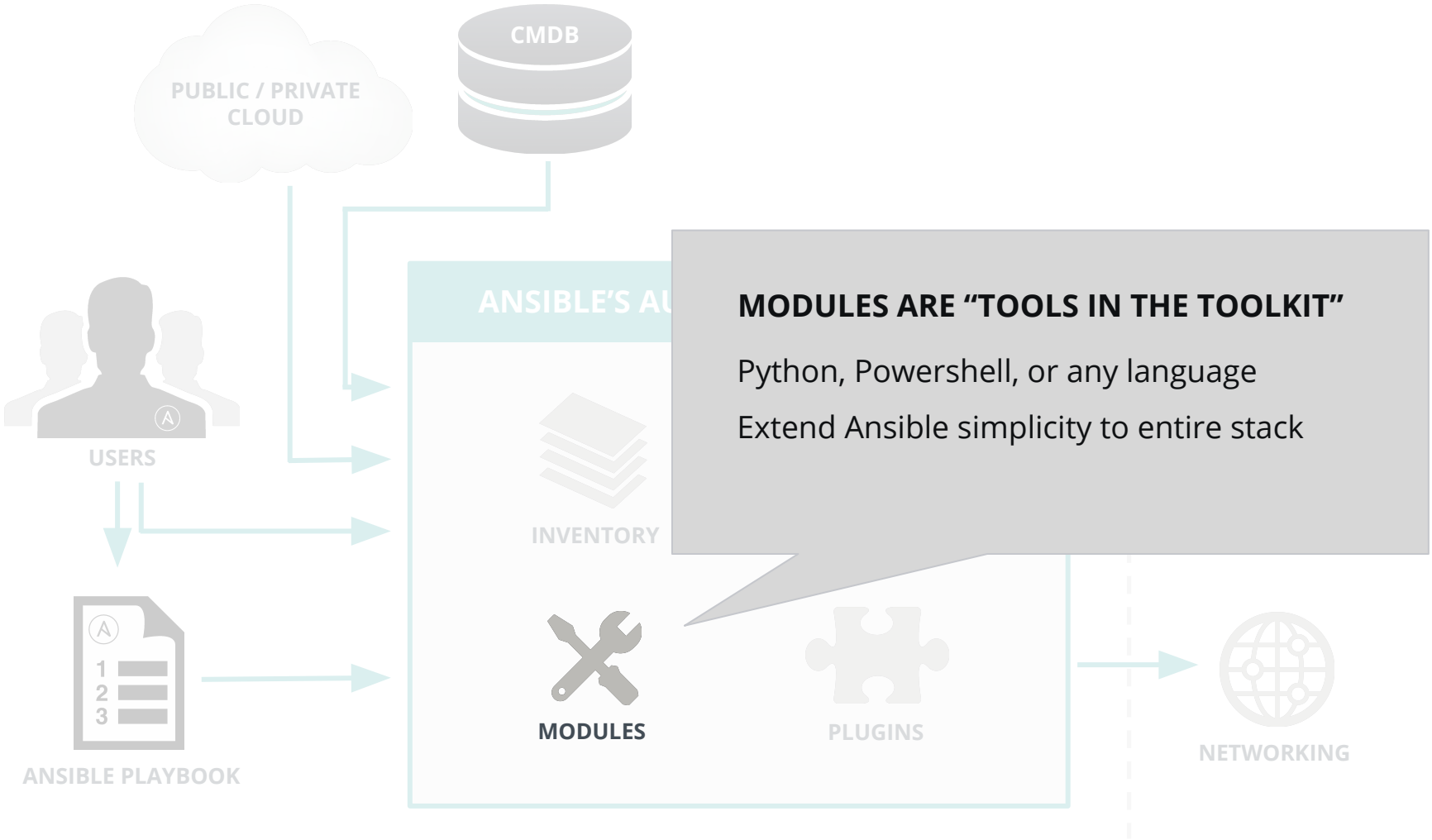
HOW ANSIBLE WORKS



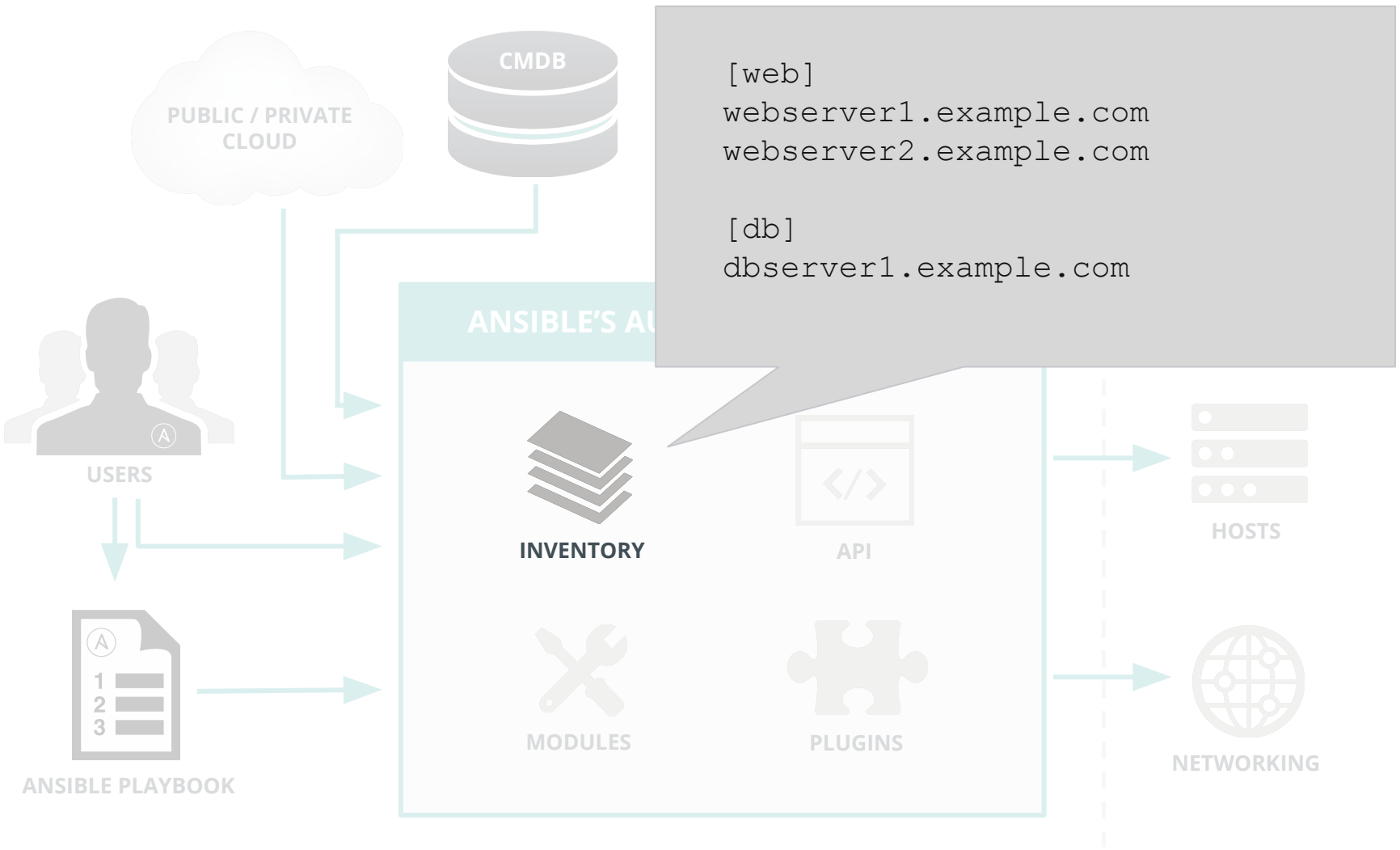
HOW ANSIBLE WORKS



HOW ANSIBLE WORKS



HOW ANSIBLE WORKS



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
```

What is missing?

SIMPLE

POWERFUL

AGENTLESS

CENTRAL

Central place for everyone

Overview of present and past

Schedule jobs

Have one common view

INTEGRATION

Simple, powerful API

Uses REST for quick adoption

No special agents or lib needed

Integrate with everything

ACCESS

Teams and users enable RBAC

Deposit credentials securely

Assign access to unprivileged

Separate access and execution

ANSIBLE TOWER

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

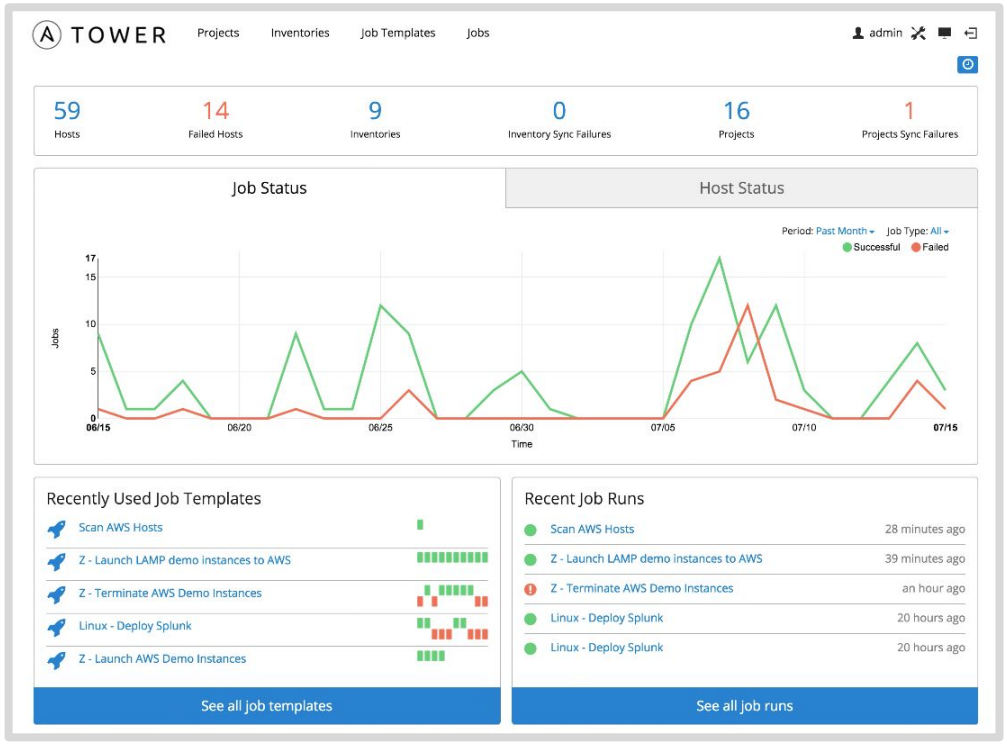
AGENTLESS

Predictable, reliable, and secure

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

WHAT IS ANSIBLE TOWER?

ANSIBLE



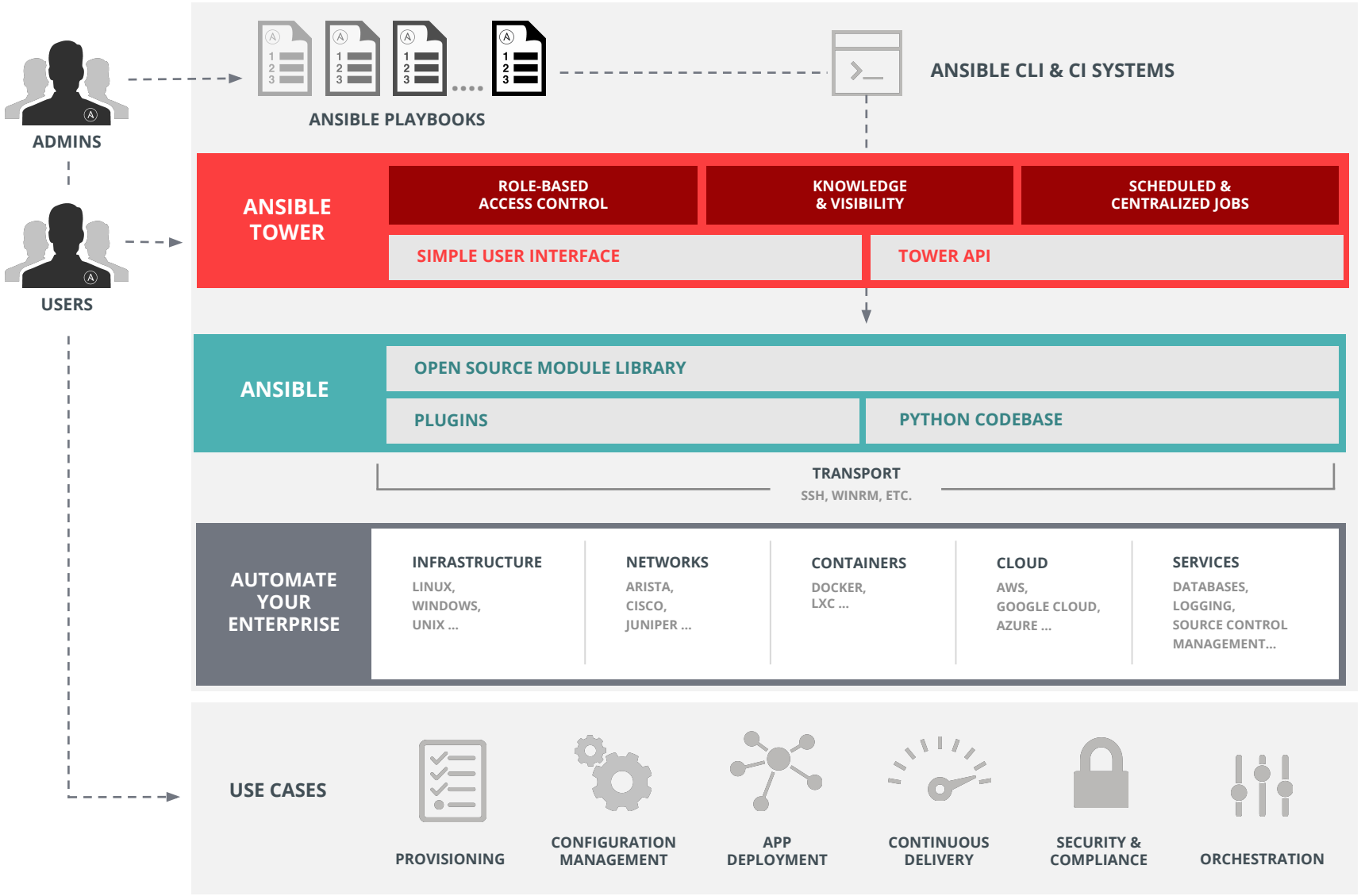
Ansible tower is an **enterprise framework** for controlling, securing and managing your Ansible automation – with a **UI and restful API.**

- **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**.

redhat.

PLATFORM OVERVIEW

ANSIBLE



redhat.

USE CASES

ANSIBLE

CONFIG MANAGEMENT

APP DEPLOYMENT

PROVISIONING

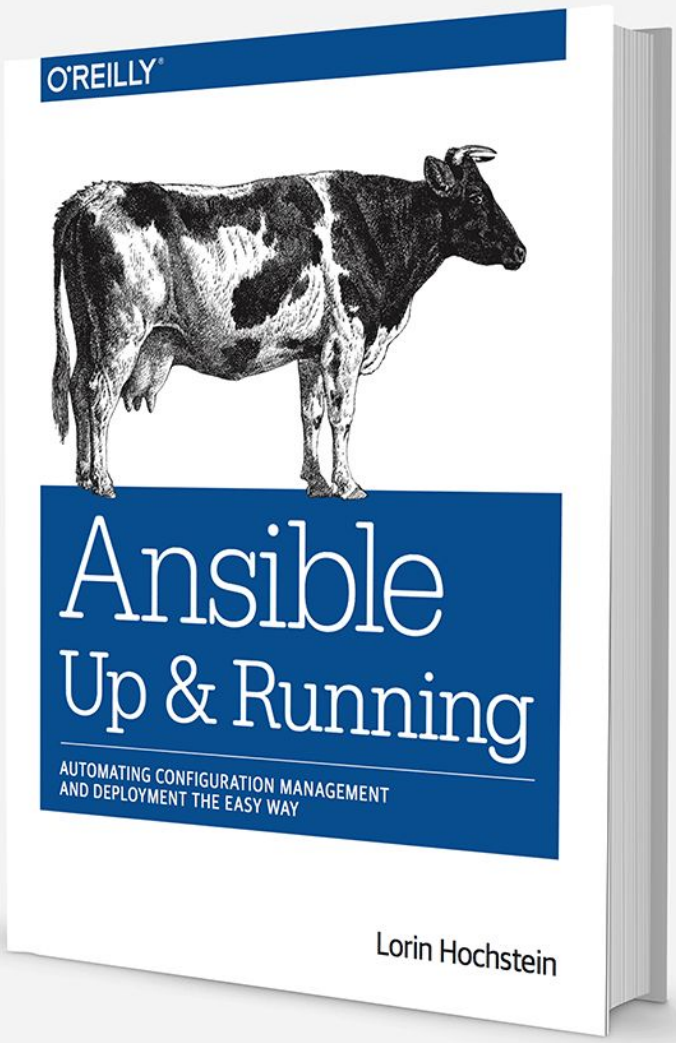
CONTINUOUS DELIVERY

SECURITY & COMPLIANCE

ORCHESTRATION

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



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.**

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

redhat