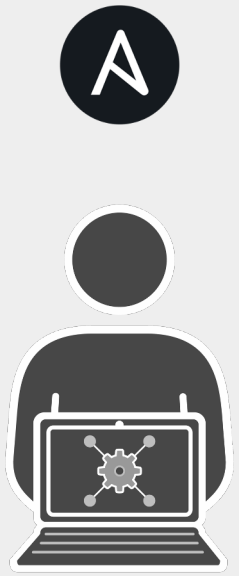
Red Hat
Ansible Automation
Platform

Automation for all

Ansible technical introduction and overview

Red Hat



Automation happens when one person meets a problem they never want to solve again

Red Hat

Teams are automating...

Lines Of Business

Network

Security

Operations

Developers

Infrastructure

Ad-hoc Automation is happening in silos

Developers → Ansible used in silo

Security → DIY scripting automation

Infrastructure → Open source config management tool

Network → Proprietary vendor supplied automation

Is organic automation enough?

Why Ansible?

Simple

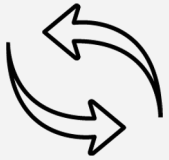
Human readable automation

No special coding skills needed

Tasks executed in order

Usable by every team

Get productive quickly



Powerful

App deployment

Configuration management

Workflow orchestration

Network automation

Orchestrate the app lifecycle

Agentless

Agentless architecture

Uses OpenSSH & WinRM

No agents to exploit or update

Get started immediately

More efficient & more secure

Red Hat

What can I do using Ansible?

Automate the deployment and management of your entire IT footprint.

Do this...

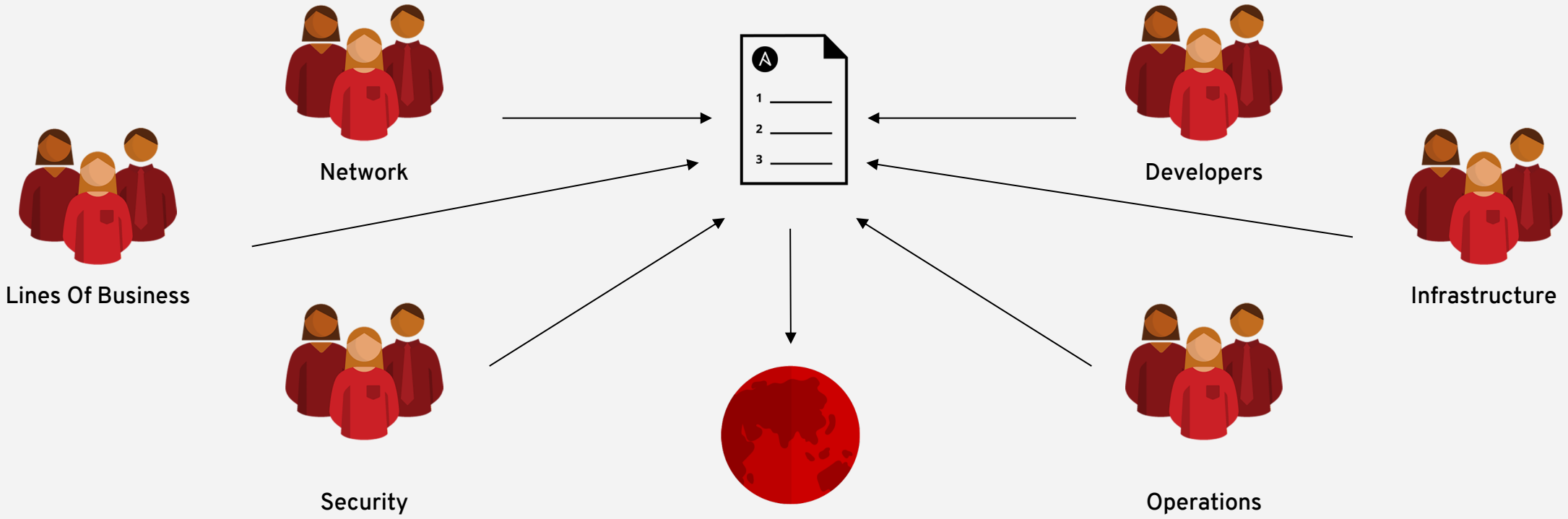
Orchestration	Configuration Management	Application Deployment	Provisioning	Continuous Delivery	Security and Compliance

On these...

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

Red Hat

When automation crosses teams, you need an automation platform



Ansible automates technologies you use

Time to automate is measured in minutes

Cloud

AWS
Azure
Digital Ocean
Google
OpenStack
Rackspace
+more

Operating Systems

RHEL
Linux
Windows
+more

Virt & Container

Docker
VMware
RHV
OpenStack
OpenShift
+more

Storage

Netapp
Red Hat Storage
Infinidat
+more

Windows

ACLs
Files
Packages
IIS
Regedits
Shares
Services
Configs
Users
Domains
+more

Network

A10
Arista
Aruba
Cumulus
Bigswitch
Cisco
Dell
Extreme
F5
Lenovo
MikroTik
Juniper
OpenSwitch
+more

Security

Checkpoint
Cisco
CyberArk
F5
Fortinet
Juniper
IBM
Palo Alto
Snort
+more

Monitoring

Dynatrace
Datadog
LogicMonitor
New Relic
Sensu
+more

Devops

Jira
GitHub
Vagrant
Jenkins
Slack
+more

Red Hat Ansible Tower

by the numbers:

94% Reduction in recovery time following a security incident

84% Savings by deploying workloads to generic systems appliances using Ansible Tower

67% Reduction in man hours required for customer deliveries

Financial summary:

146%

ROI on Ansible Tower

< 3 MONTHS

Payback on Ansible Tower

SOURCE: *"The Total Economic Impact™ Of Red Hat Ansible Tower, a June 2018 commissioned study conducted by Forrester Consulting on behalf of Red Hat."*
redhat.com/en/engage/total-economic-impact-ansible-tower-20180710

Red Hat

Red Hat
Ansible Automation
Platform

Red Hat Ansible Engine:

Universal language of automation

Red Hat

Red Hat Ansible Automation Platform

Network | Lines of business | Security

Operations | Infrastructure | Developers

Engage — **Ansible SaaS:** Engage users with an automation focused experience

Scale — **Ansible Tower:** Operate & control at scale

Create

Simple	**Powerful**	**Agentless**
Human readable automation	Thousands of integrations	No agents to exploit or update

Fueled by an open source community

Red Hat

Red Hat Ansible Engine

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 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.

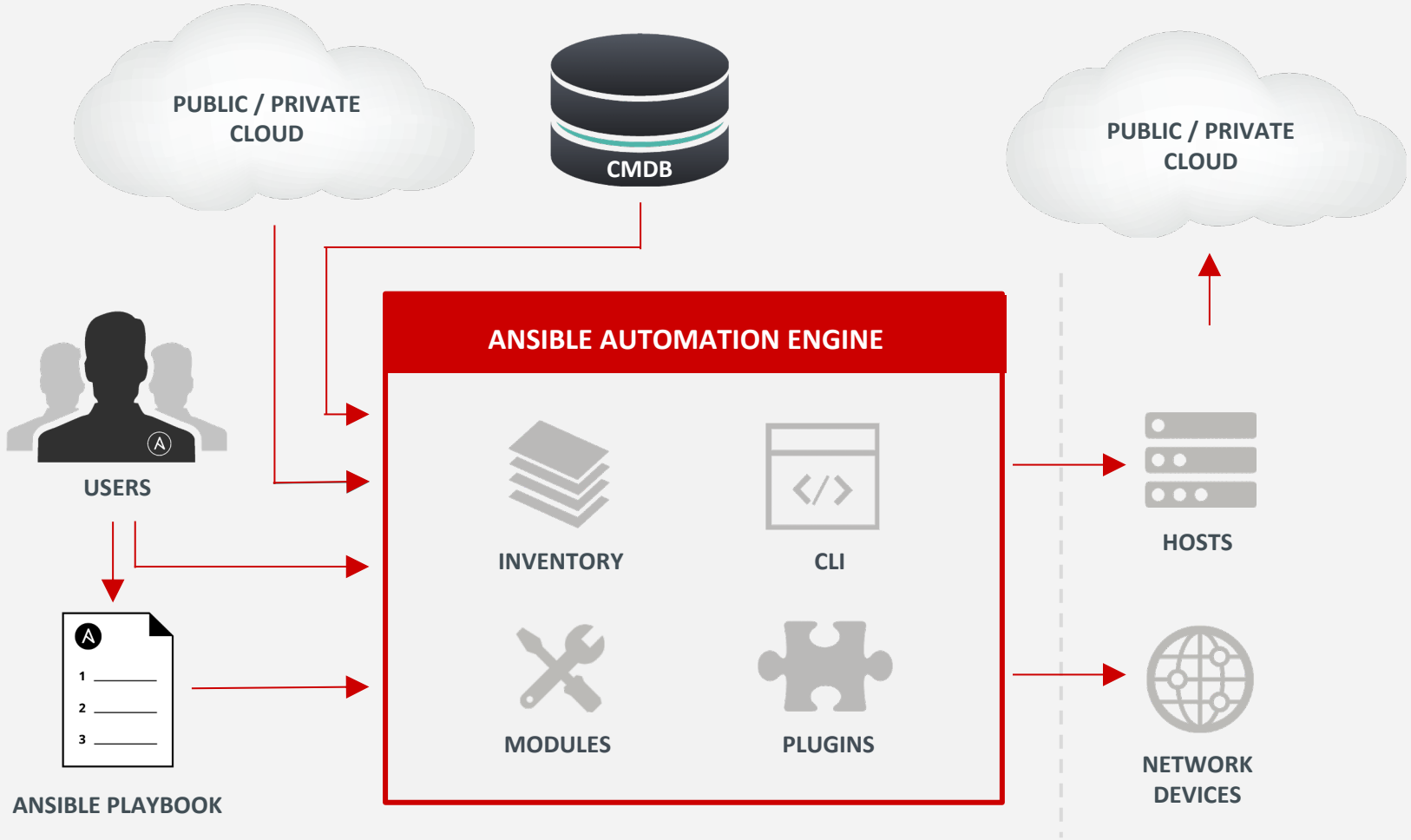
Dynamic inventories

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

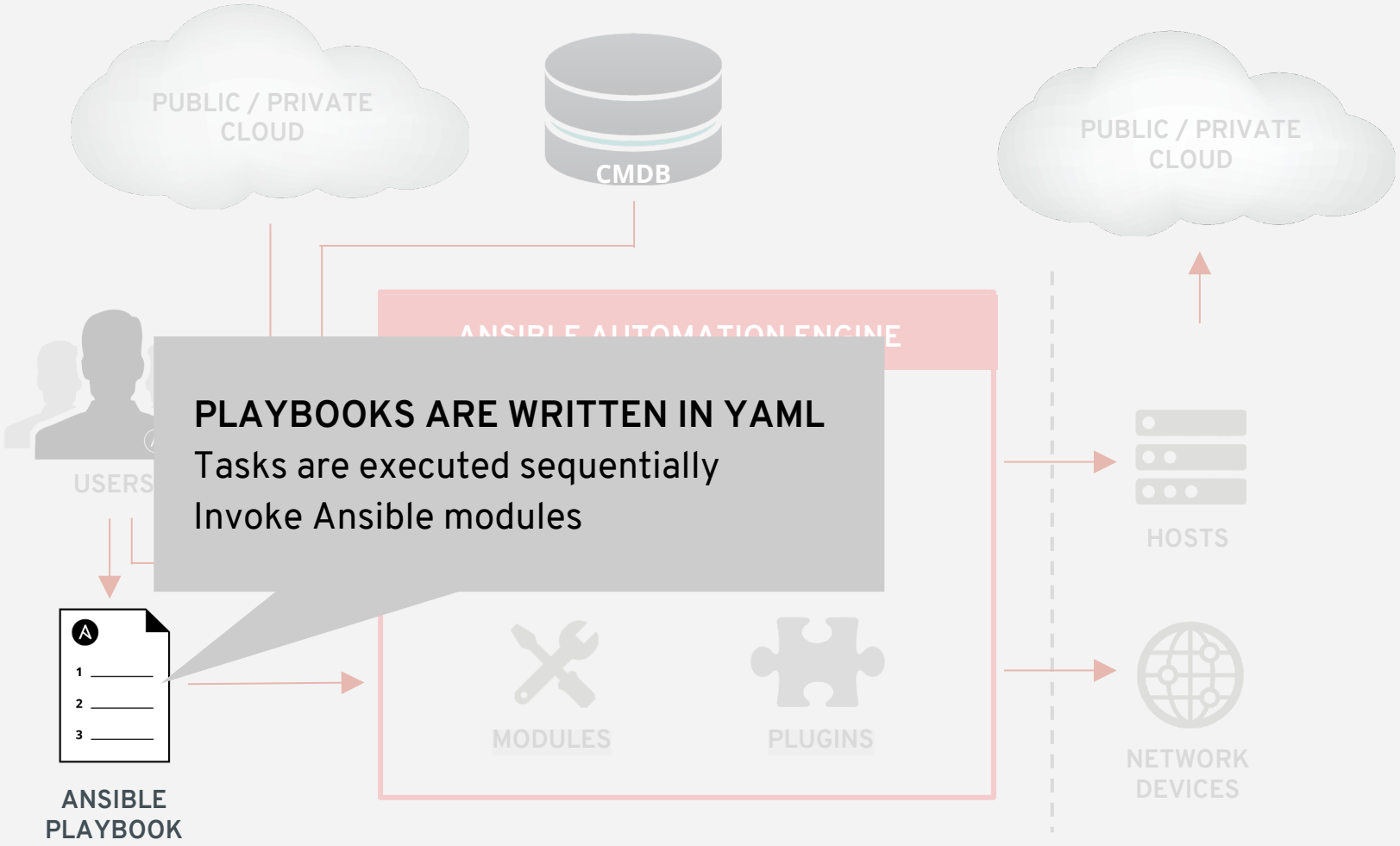
Orchestration plays well with others

Orchestration plays well with others: ServiceNow, Infoblox, AWS, Terraform, Cisco ACI and more

Red Hat

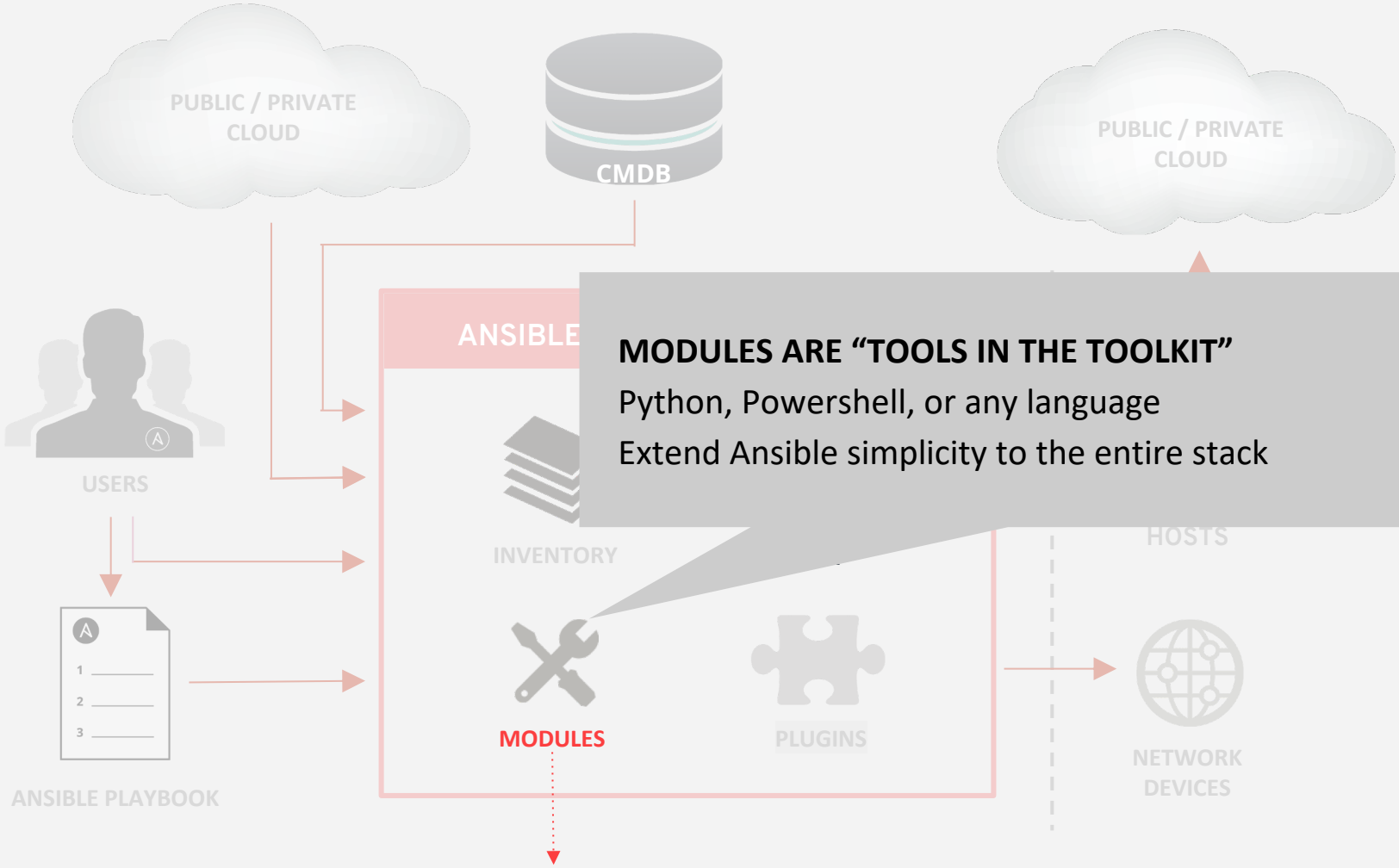

PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION ENGINE
USERS
INVENTORY
CLI
HOSTS
MODULES
PLUGINS
NETWORK DEVICES
1
2
3
ANSIBLE PLAYBOOK

Red Hat


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
USERS
HOSTS
MODULES
PLUGINS
NETWORK DEVICES
ANSIBLE PLAYBOOK
PLAYBOOKS ARE WRITTEN IN YAML
Tasks are executed sequentially
Invoke Ansible modules

Red Hat

```
---
- name: install and start apache
  hosts: web
  become: yes
  vars:
       http_port: 80

  tasks:
    - name: httpd package is present
          yum:
            name: httpd
            state: latest

    - name: latest index.html file is present
      template:
        src: files/index.html
        dest: /var/www/html/

    - name: httpd is started
      service:
        name: httpd
        state: started
```



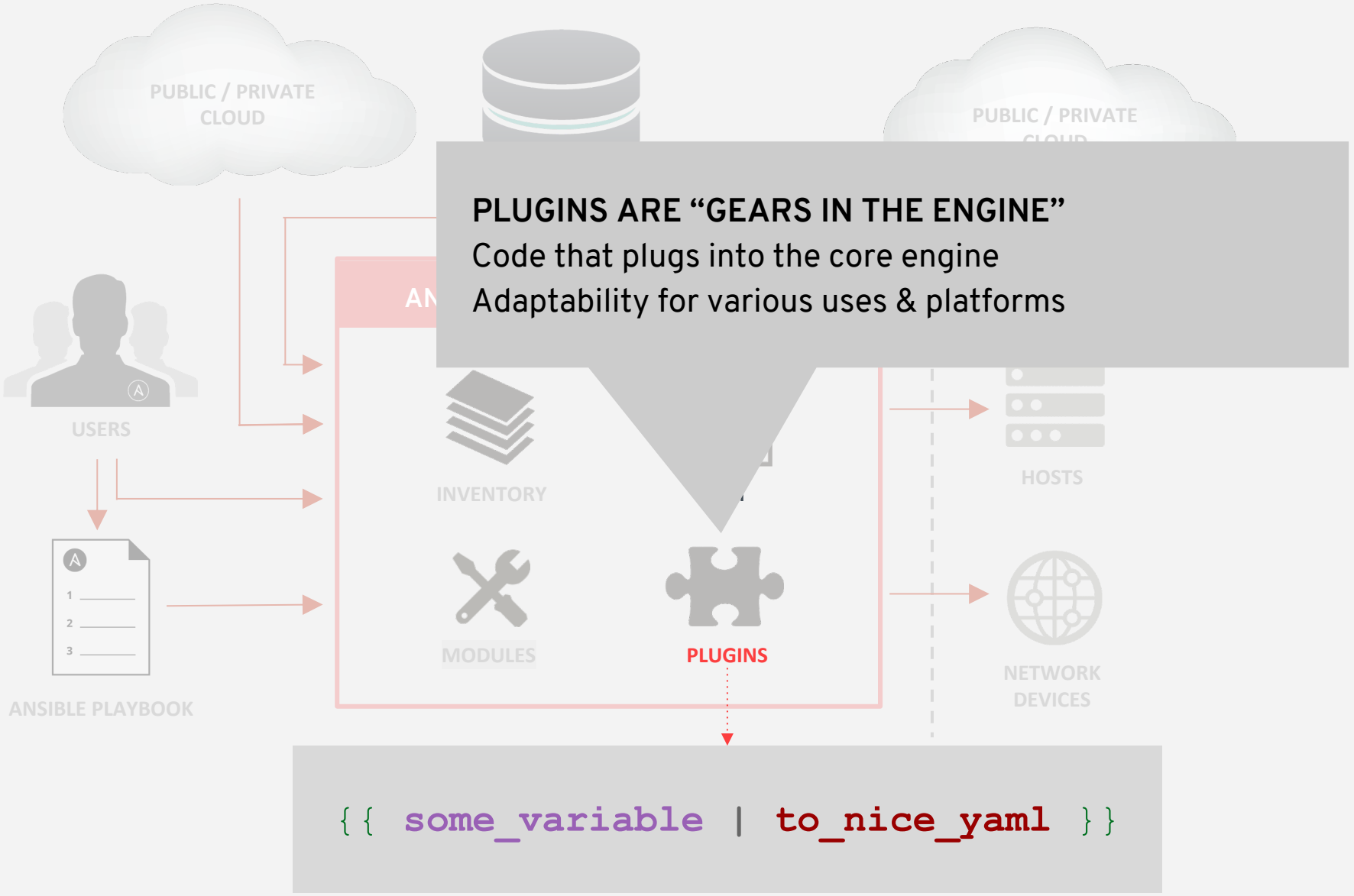
MODULES ARE "TOOLS IN THE TOOLKIT"

Python, Powershell, or any language

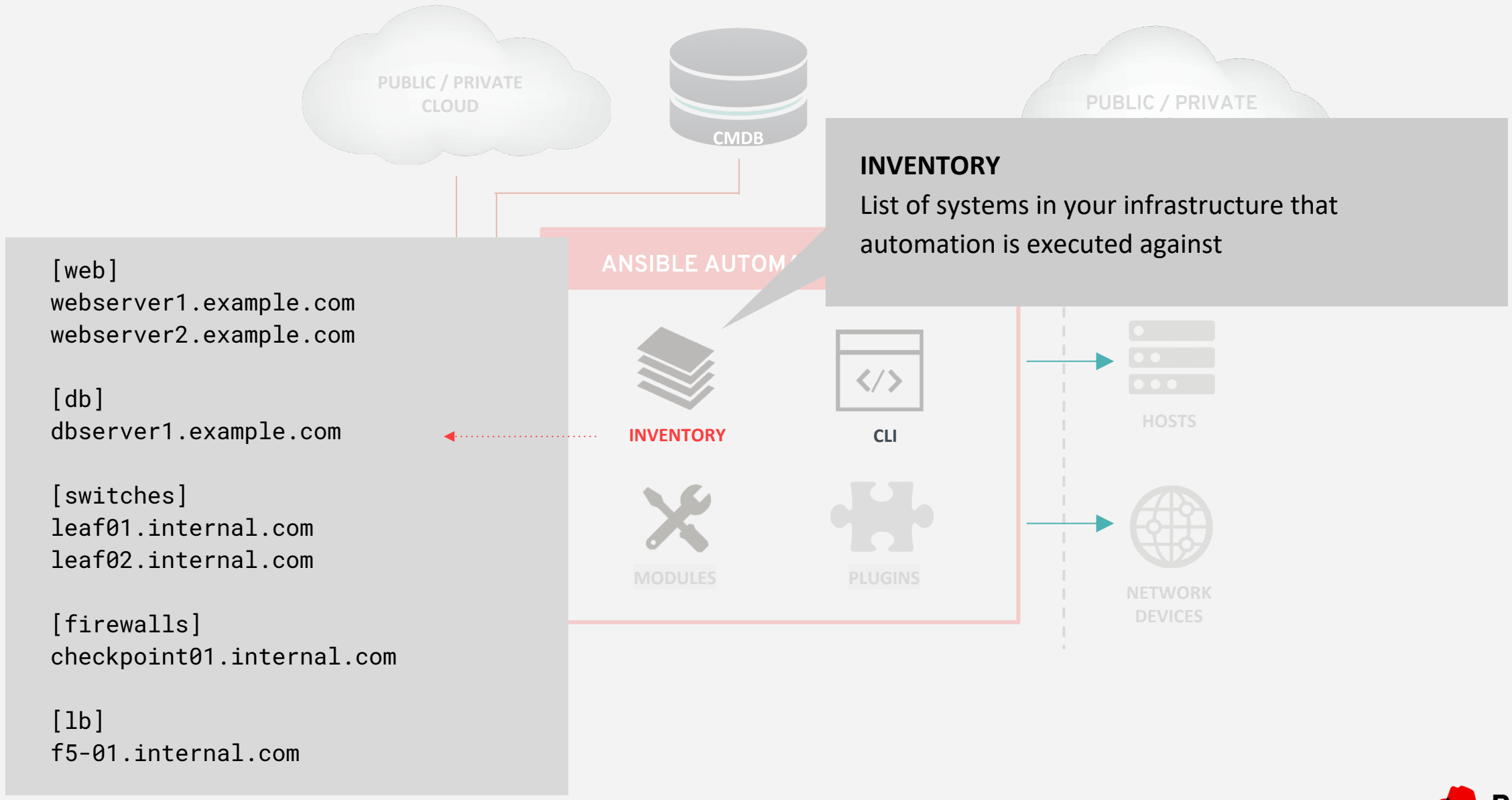
Extend Ansible simplicity to the entire stack

```
- name: latest index.html file is present
  template:
    src: files/index.html
    dest: /var/www/html/
```

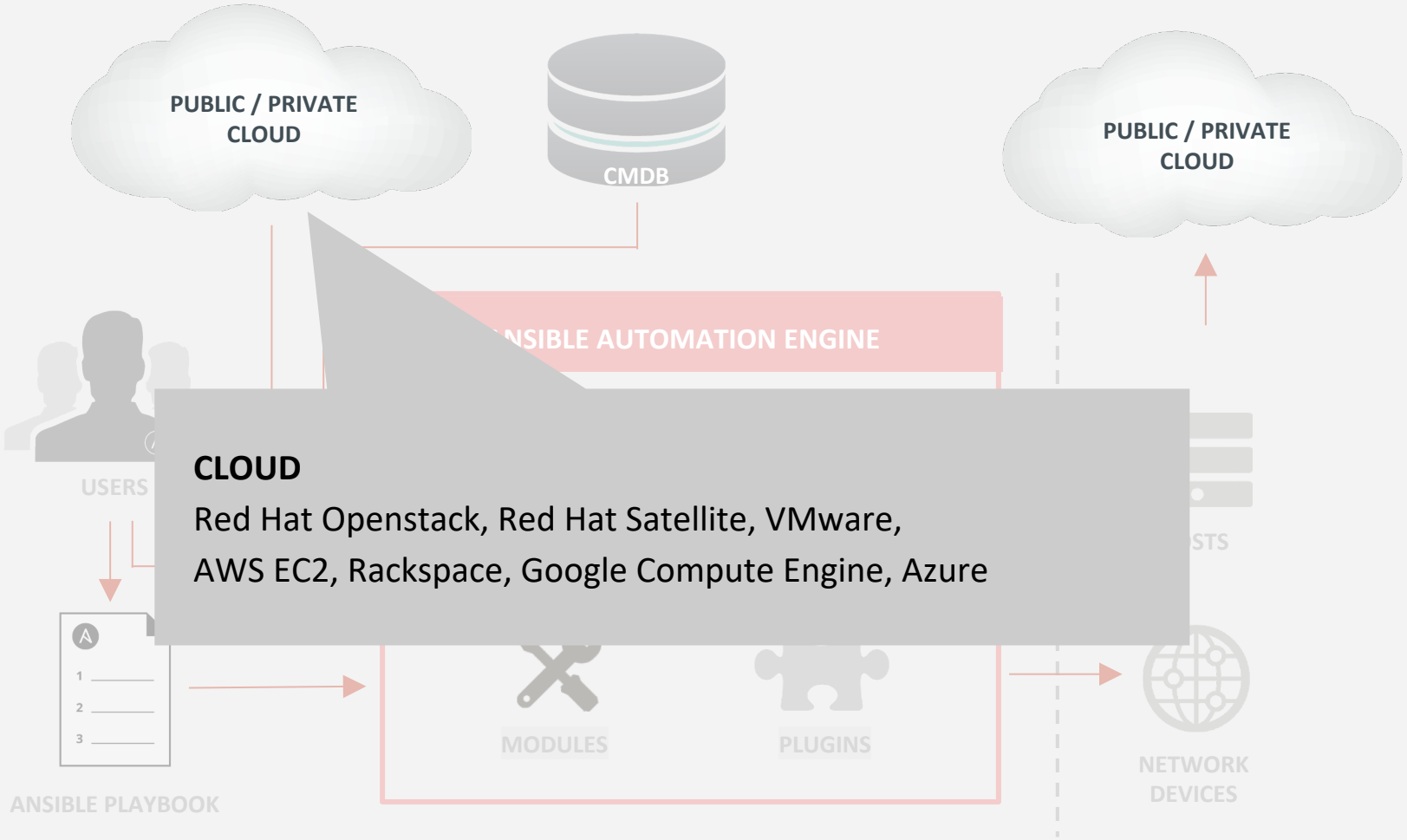
Red Hat


PUBLIC / PRIVATE CLOUD
PUBLIC / PRIVATE CLOUD
PLUGINS ARE “GEARS IN THE ENGINE”
Code that plugs into the core engine
Adaptability for various uses & platforms
USERS
INVENTORY
HOSTS
MODULES
PLUGINS
NETWORK DEVICES
ANSIBLE PLAYBOOK
{{ some_variable | to_nice_yaml }}

Red Hat


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE
INVENTORY
List of systems in your infrastructure that automation is executed against
[web]
webserver1.example.com
webserver2.example.com
[db]
dbserver1.example.com
[switches]
leaf01.internal.com
leaf02.internal.com
[firewalls]
checkpoint01.internal.com
[lb]
f5-01.internal.com
ANSIBLE AUTOM
INVENTORY
CLI
MODULES
PLUGINS
HOSTS
NETWORK DEVICES

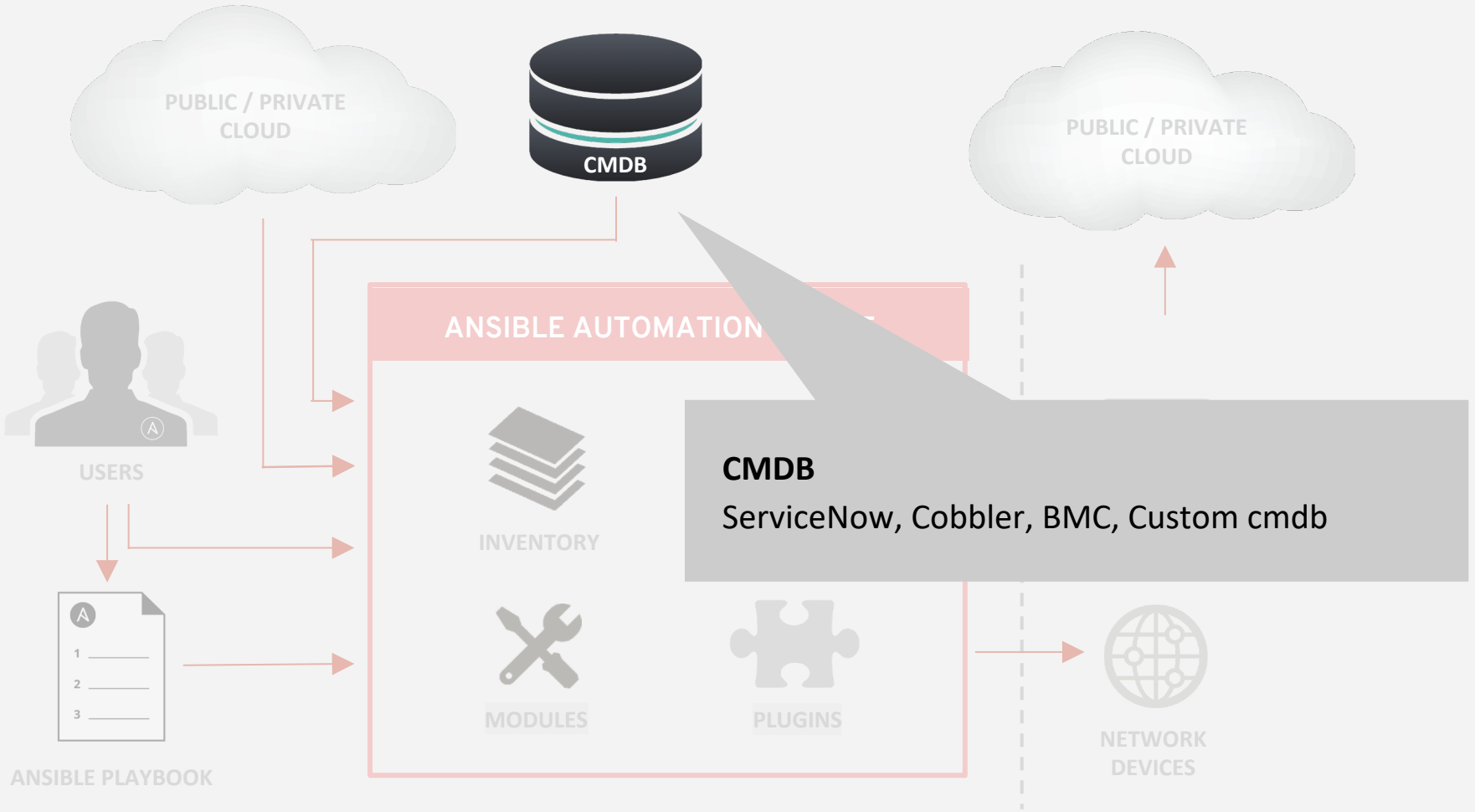
Red Hat



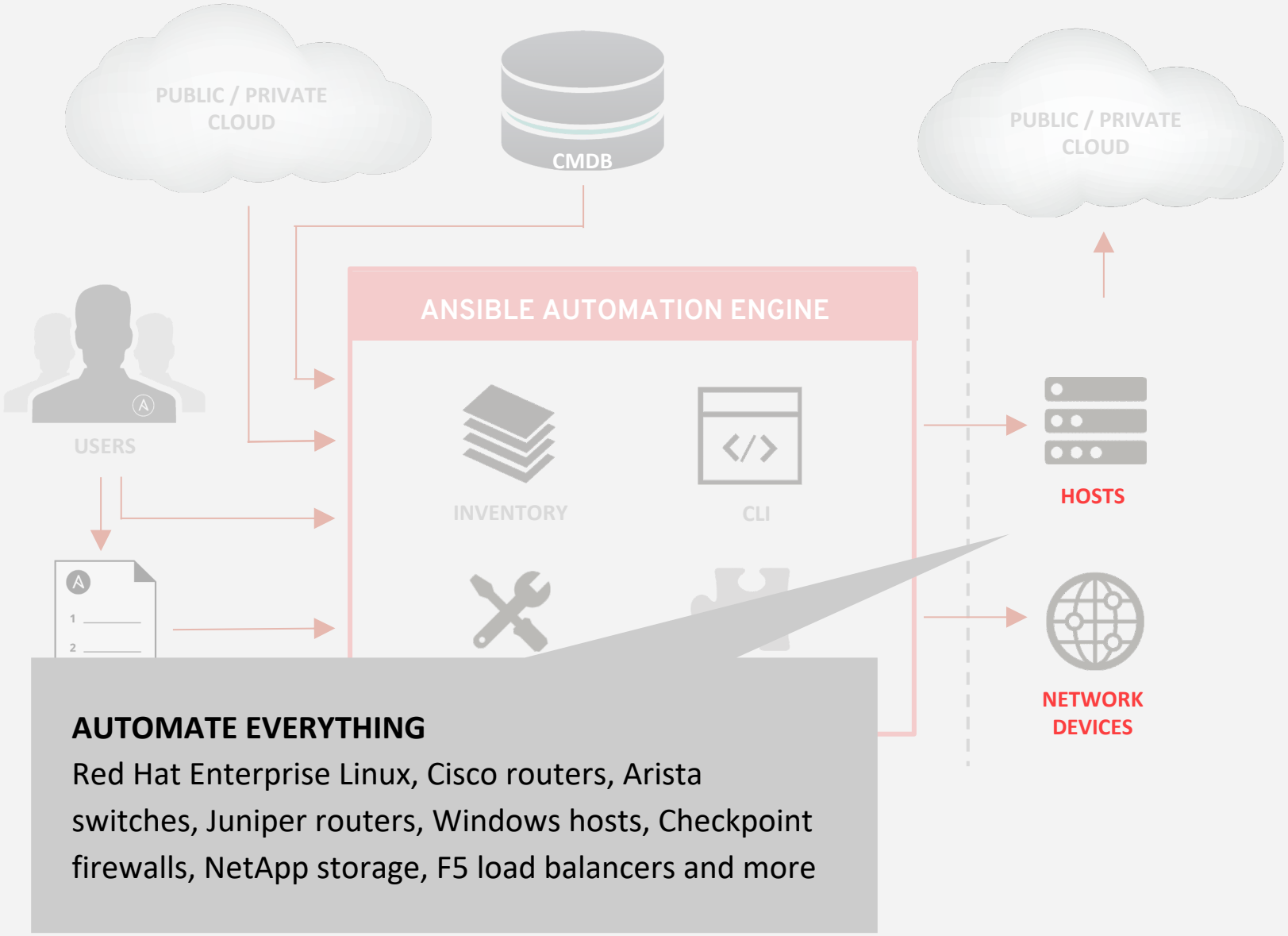
CLOUD

Red Hat Openstack, Red Hat Satellite, VMware,
AWS EC2, Rackspace, Google Compute Engine, Azure

Red Hat


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION
USERS
INVENTORY
CMDB
ServiceNow, Cobbler, BMC, Custom cmdb
MODULES
PLUGINS
NETWORK DEVICES
ANSIBLE PLAYBOOK

Red Hat


PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
ANSIBLE AUTOMATION ENGINE
USERS
INVENTORY
CLI
HOSTS
NETWORK DEVICES
AUTOMATE EVERYTHING
Red Hat Enterprise Linux, Cisco routers, Arista switches, Juniper routers, Windows hosts, Checkpoint firewalls, NetApp storage, F5 load balancers and more

Red Hat

Ansible Playbook examples:

GITHUB EXAMPLES
github.com/ansible/ansible-examples
github.com/ansible/workshops

LAMP + HAPROXY + NAGIOS
bit.ly/lamp_haproxy

WINDOWS
bit.ly/ansible_windows

COMPLIANCE
bit.ly/ansible_compliance

NETWORK
github.com/network-automation

SECURITY
github.com/ansible-security/

Red Hat
Ansible Automation
Platform

Red Hat Ansible Tower:

Operate and control at scale

Red Hat

Red Hat Ansible Automation Platform

Network | Lines of business | Security | Operations | Infrastructure | Developers

Engage

Ansible SaaS: Engage users with an automation focused experience

Scale

Control
Web UI and API

Delegation
Role Based Access Controls

Scale
Scalable Execution Capacity

Create

Ansible Engine: Universal language of automation

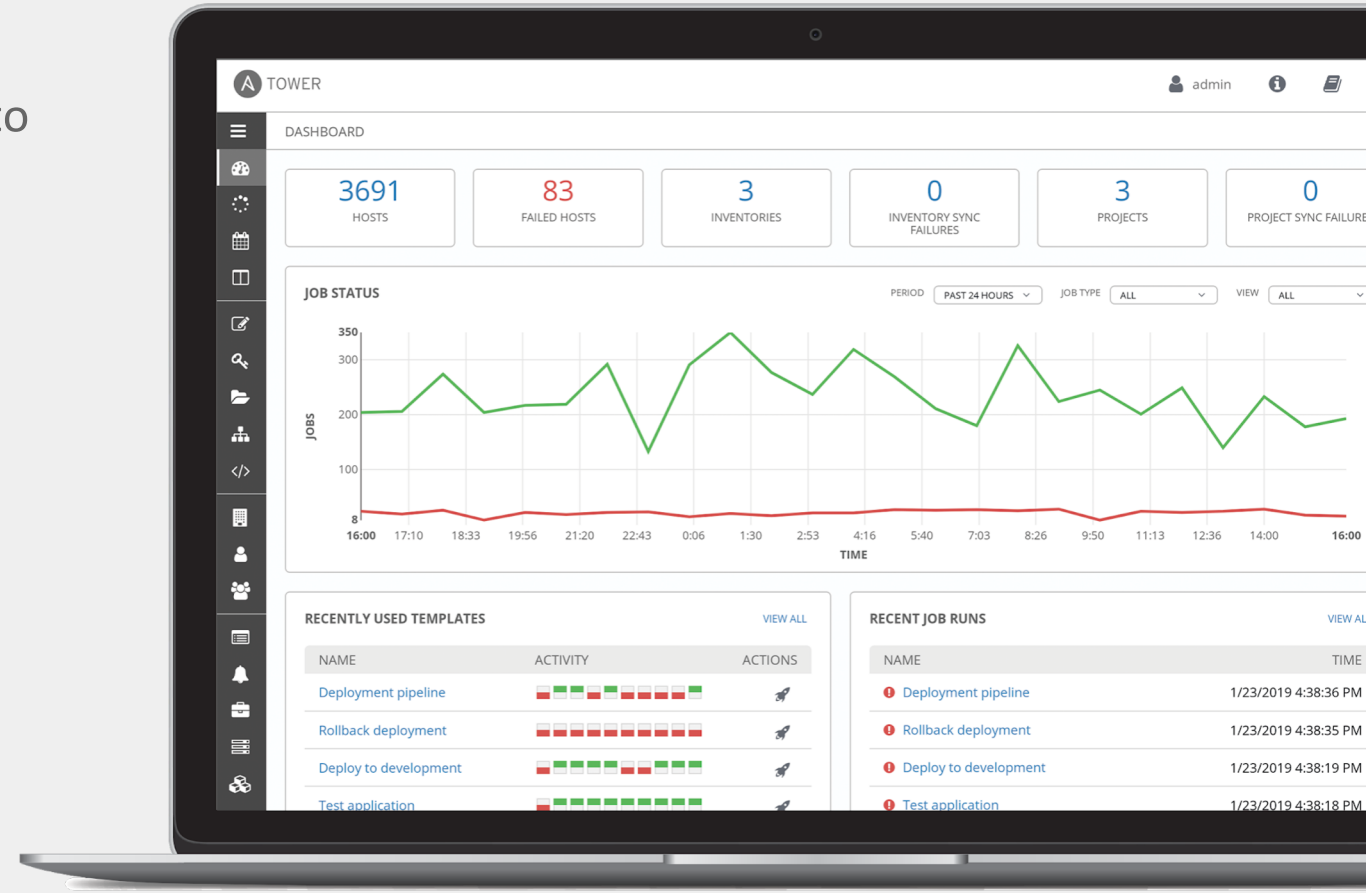
Fueled by an open source community

Red Hat

What is Ansible Tower?

Ansible Tower is a UI and RESTful API allowing you to scale IT automation, manage complex deployments and speed productivity.

- Role-based access control
- Deploy entire applications with push-button deployment access
- All automations are centrally logged
- Powerful workflows match your IT processes



Red Hat

Red Hat Ansible Tower

Push button

An intuitive user interface experience makes it easy for novice users to execute playbooks you allow them access to.

RESTful API

With an API first mentality every feature and function of Tower can be API driven. Allow seamless integration with other tools like ServiceNow and Infoblox.

RBAC

Allow restricting playbook access to authorized users. One team can use playbooks in check mode (read-only) while others have full administrative abilities.

Enterprise integrations

Integrate with enterprise authentication like TACACS+, RADIUS, Azure AD. Setup token authentication with OAuth 2. Setup notifications with PagerDuty, Slack and Twilio.

Centralized logging

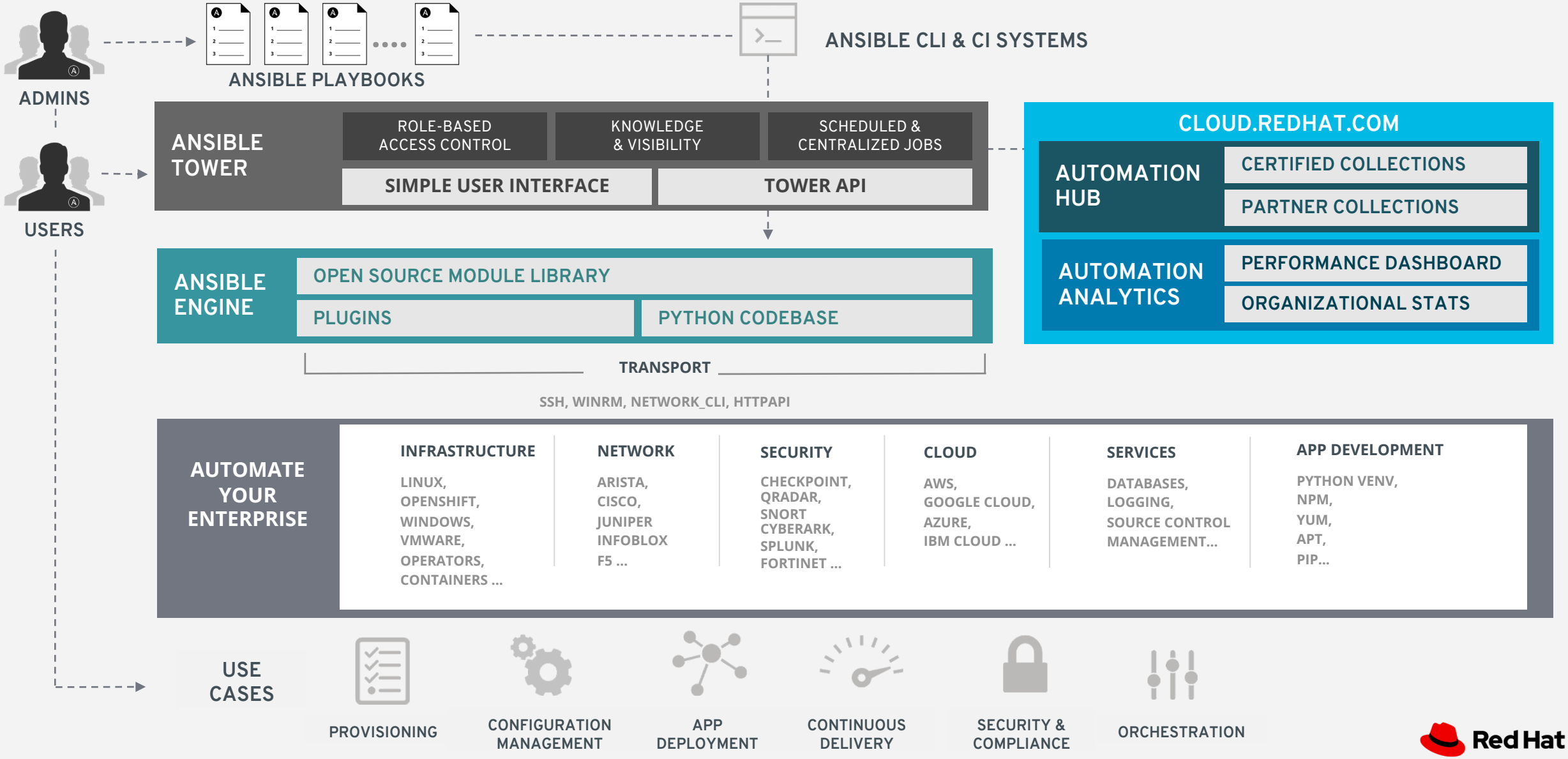
All automation activity is securely logged. Who ran it, how they customized it, what it did, where it happened - all securely stored and viewable later, or exported through Ansible Tower's API.

Workflows

Ansible Tower's multi-playbook workflows chain any number of playbooks, regardless of whether they use different inventories, run as different users, run at once or utilize different credentials.

Red Hat

Ansible Automation Platform



Red Hat

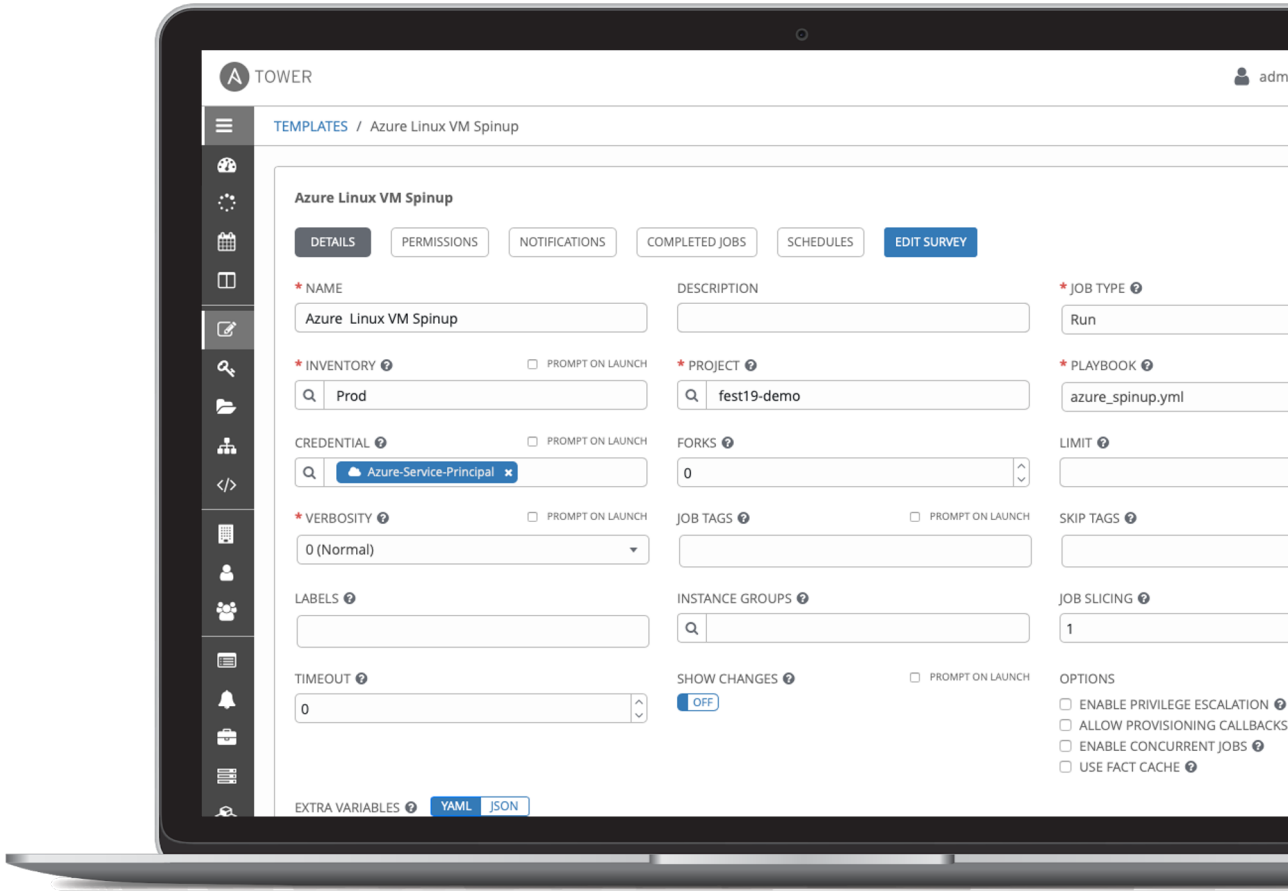
Job Templates

Everything in Ansible Tower revolves around the concept of a **Job Template**. Job Templates allow Ansible Playbooks to be controlled, delegated and scaled for an organization.

Job templates also encourage the reuse of Ansible Playbook content and collaboration between teams.

A **Job Template** requires:

- An **Inventory** to run the job against
- A **Credential** to login to devices.
- A **Project** which contains Ansible Playbooks

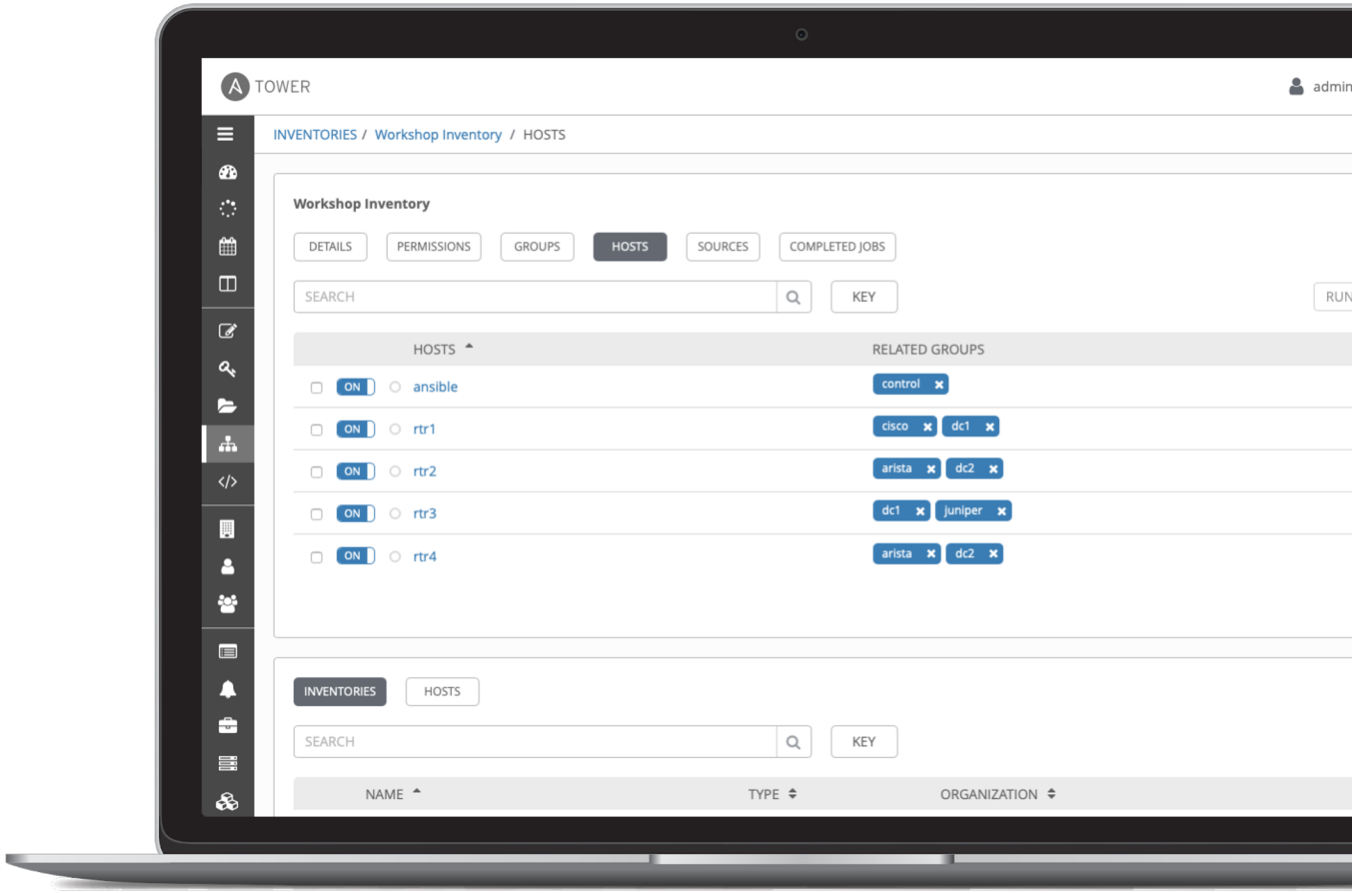


Red Hat

Inventory

Inventory is a collection of hosts (nodes) with associated data and groupings that Ansible Tower can connect to and manage.

- Hosts (nodes)
- Groups
- Inventory-specific data (variables)
- Static or dynamic sources



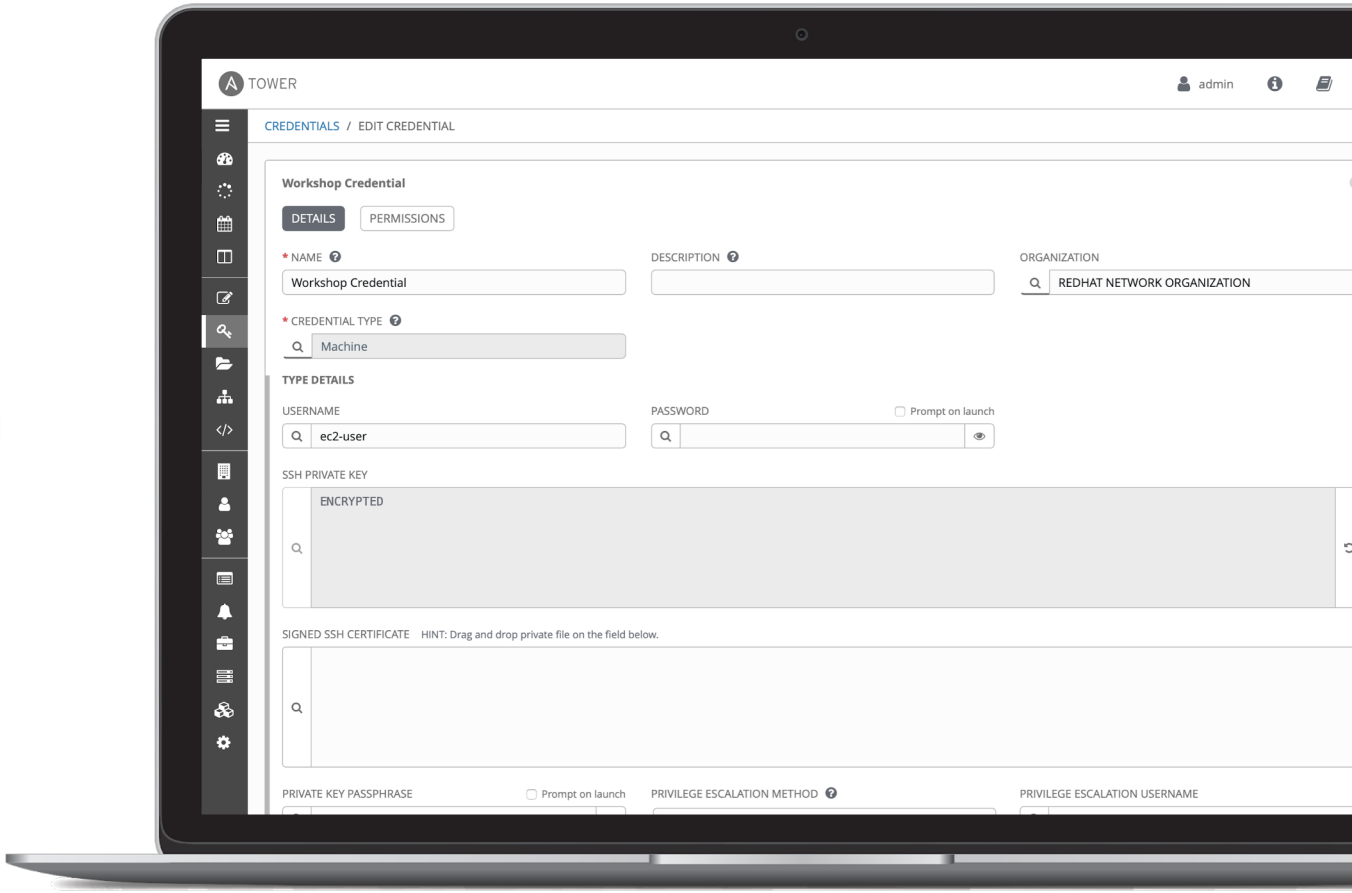
Red Hat

Credentials

Credentials are utilized by Ansible Tower for authentication with various external resources:

- Connecting to remote machines to run jobs
- Syncing with inventory sources
- Importing project content from version control systems
- Connecting to and managing network devices

Centralized management of various credentials allows end users to leverage a secret without ever exposing that secret to them.

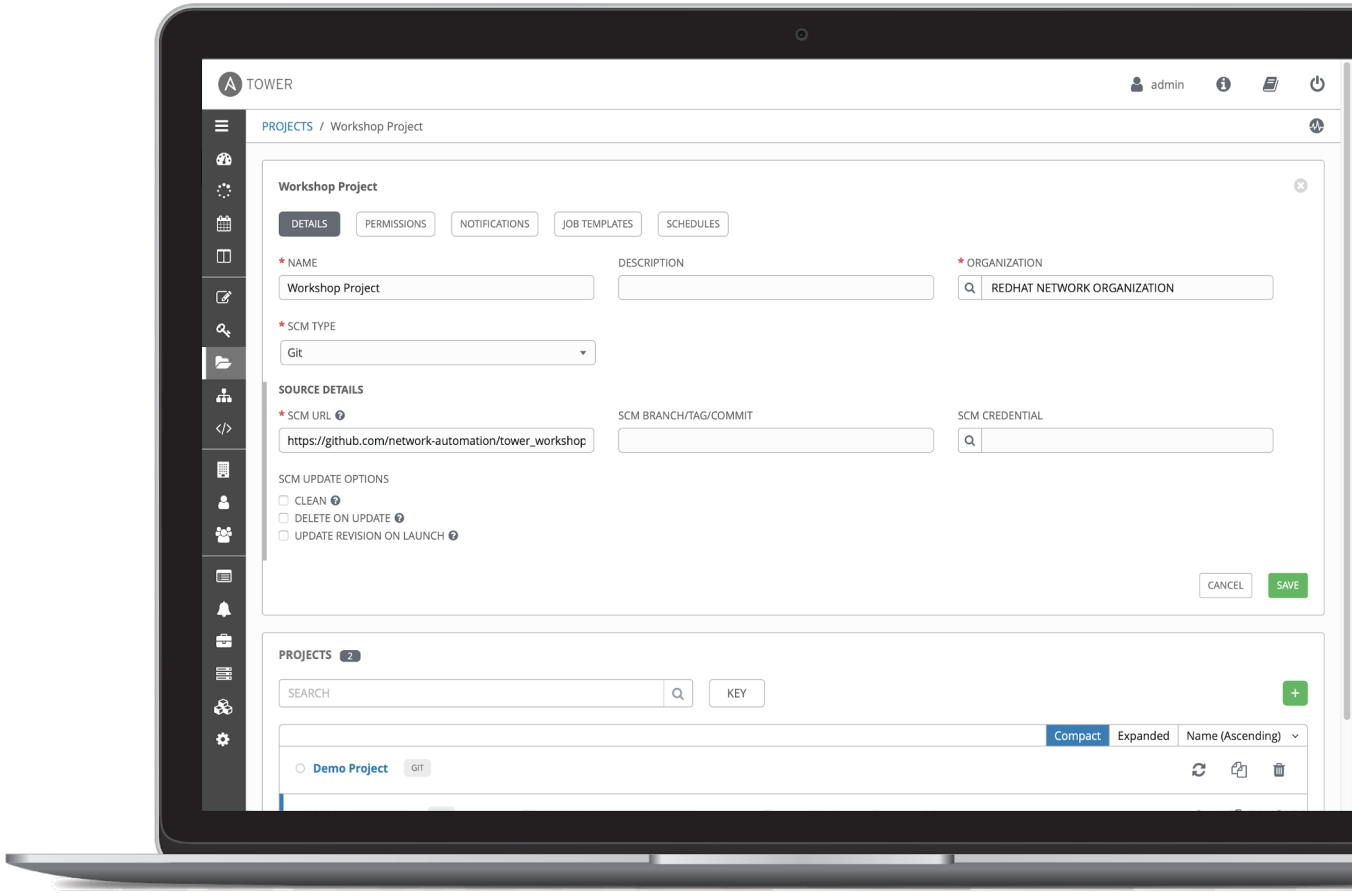


Red Hat

Project

A project is a logical collection of Ansible Playbooks, represented in Ansible Tower.

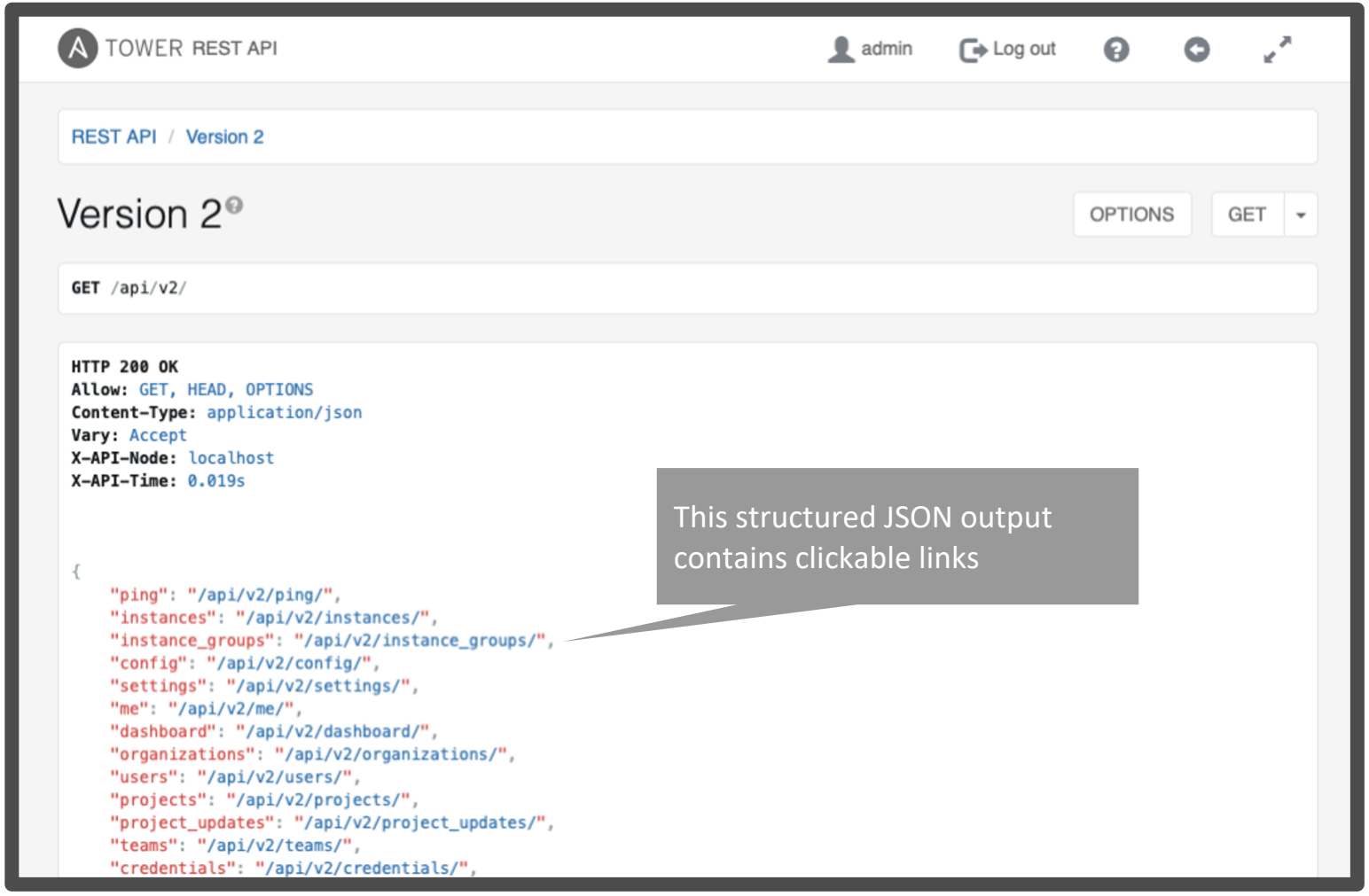
You can manage Ansible Playbooks and playbook directories by placing them in a source code management system supported by Ansible Tower, including Git, Subversion, and Mercurial.



Red Hat

RESTful API

Fully browsable API, everything within the Web UI can be accessed via the API for programmatic access

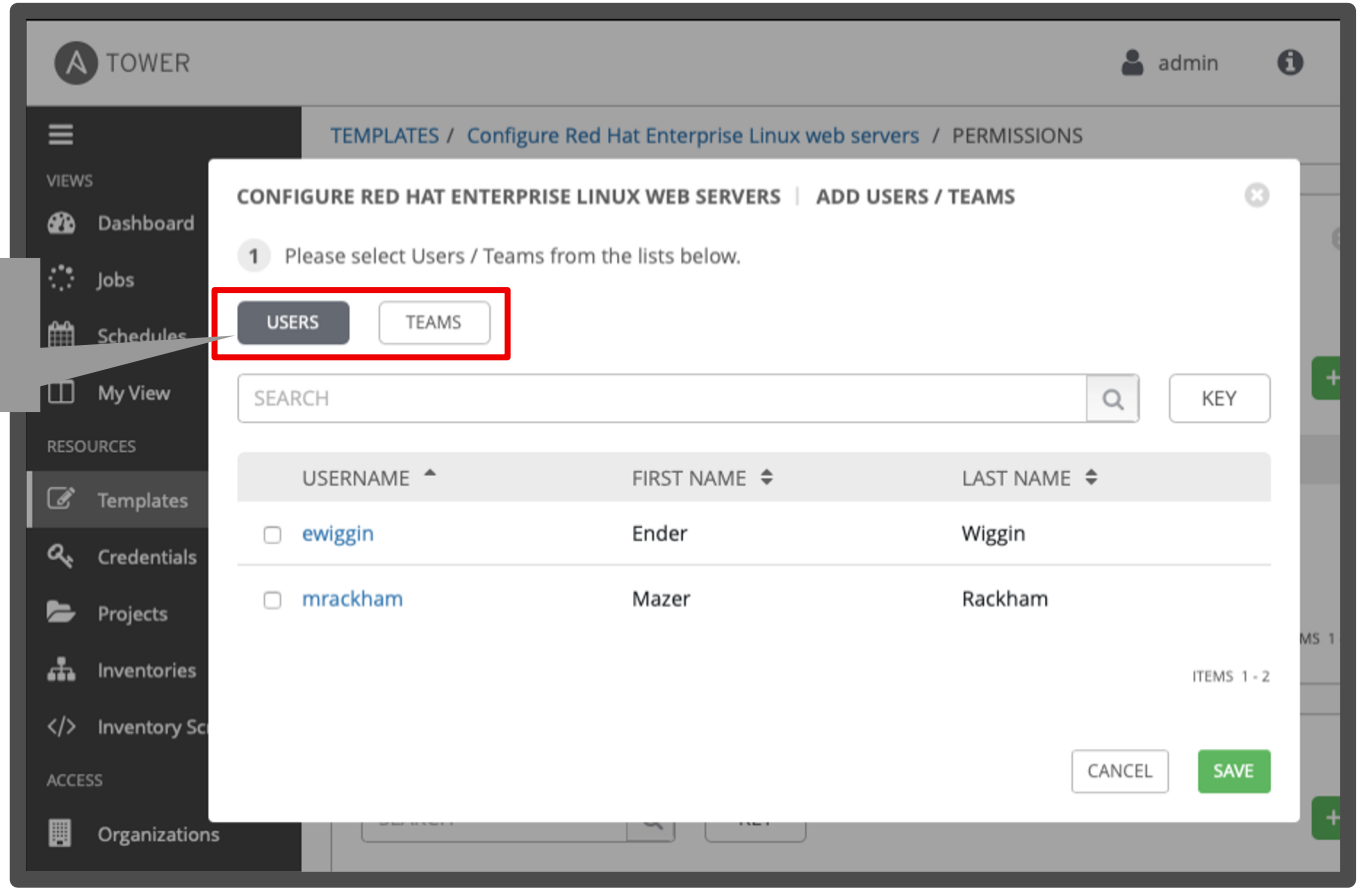


Red Hat

Role Based Access Control (RBAC)

Job Templates, Inventory, Credentials and Projects can be assigned to specific Users and Teams.

Clicking the USERS or TEAMS buttons shows available options

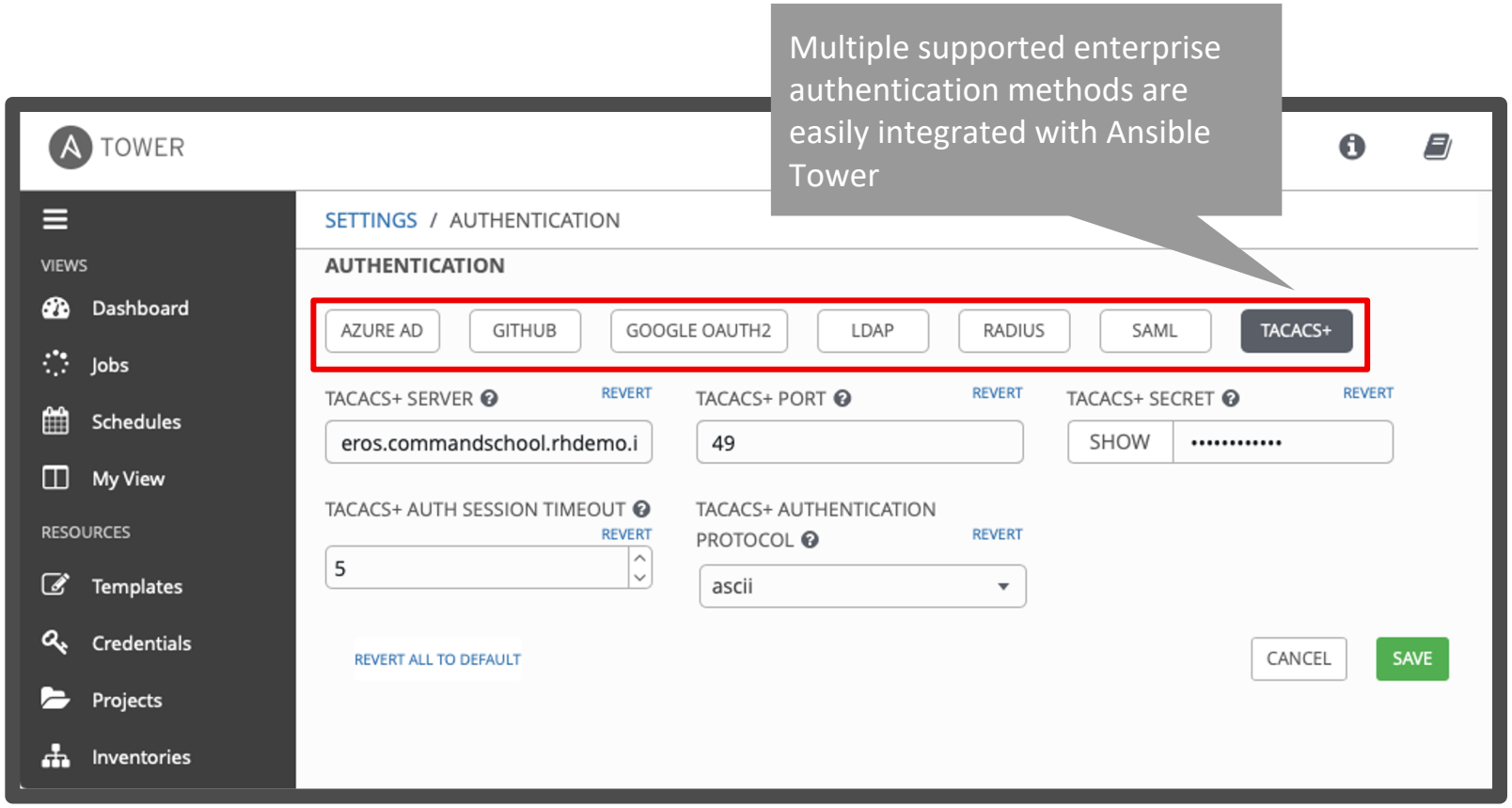


Red Hat

Enterprise Authentication

Use your existing enterprise authentication including:

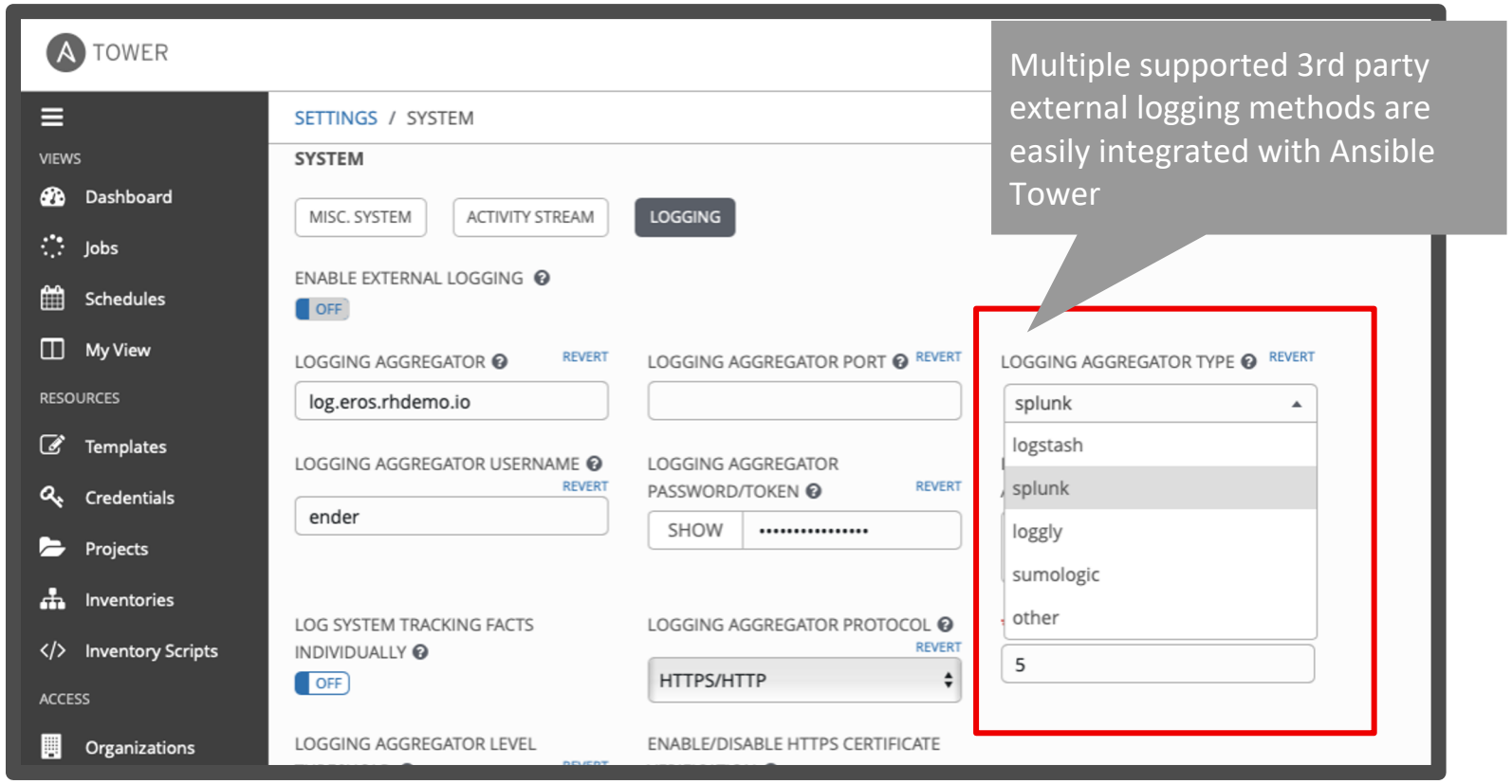
- Azure AD
- Github
- Google OAuth2
- LDAP
- Radius
- SAML
- TACACS+



Red Hat

Centralized Logging

Ansible Tower creates a centralized control point for Ansible Automation. If desired Ansible Tower can integrated with existing log aggregation services.



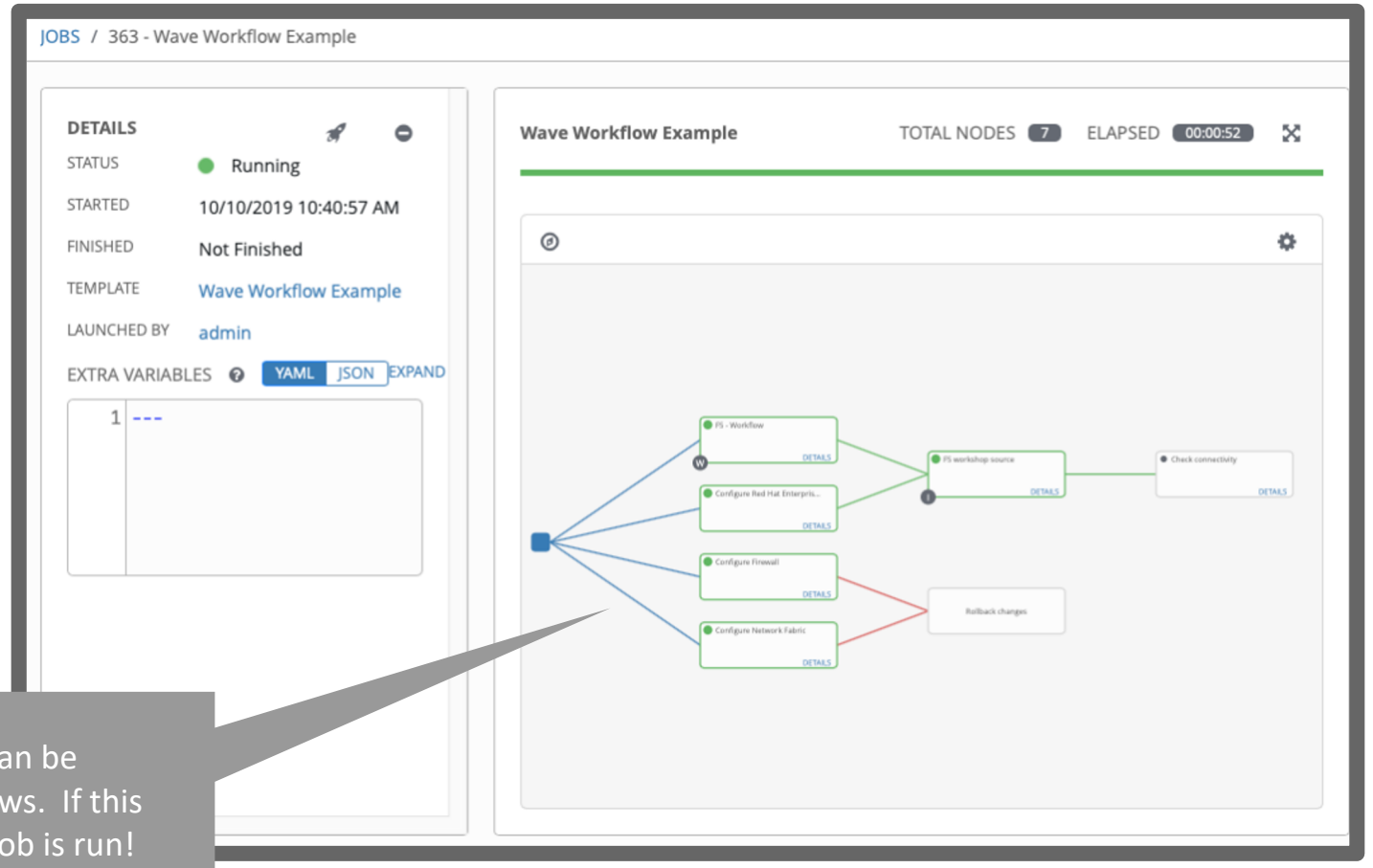
Red Hat

Workflows

Create powerful holistic automation using Ansible Workflows.

Orchestration can easily be configured by linking Job Templates.

Workflow approvals allow Workflows to pause and wait for human interaction



Conditional logic can be applied to workflows. If this job fails this next Job is run!

Red Hat

Webhooks - Enabling GitOps

Trigger Job Templates or Workflows straight via configurable webhooks

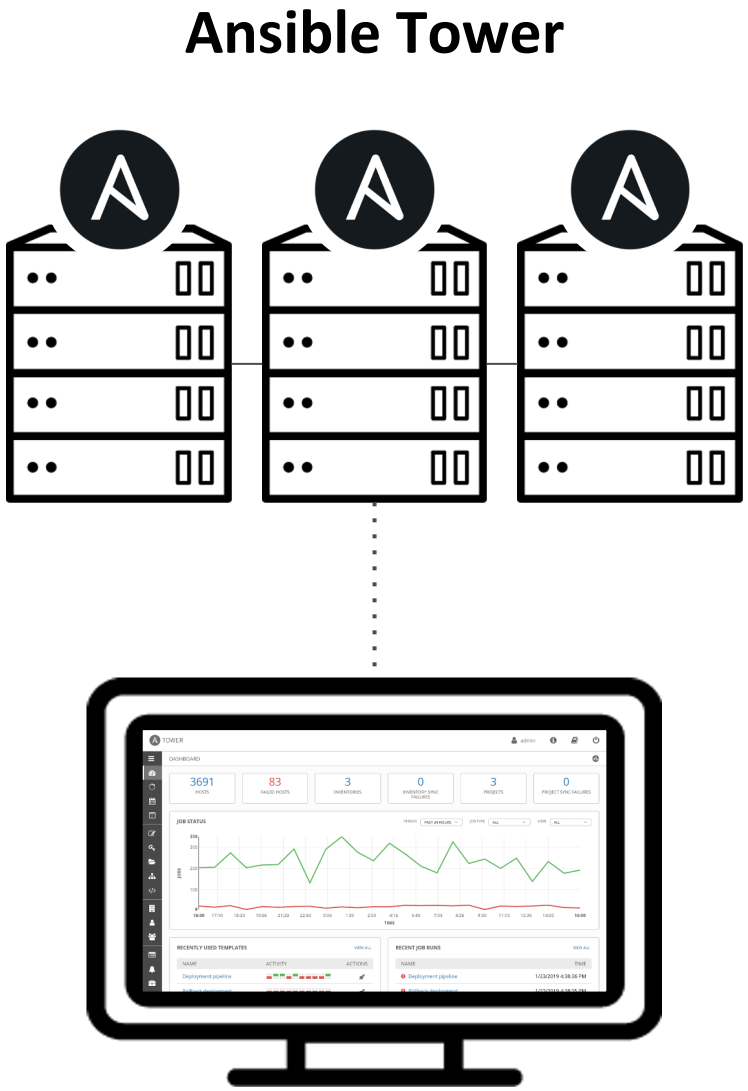
Automatically provision, update, configure, and apply based on pushes to your source control.

Red Hat

Scale

Ansible Tower clusters add redundancy and capacity, allowing you to scale Ansible automation across your enterprise.

- Unifying task execution across execution nodes
- Leverage Kubernetes and OpenShift to spin up execution capacity at runtime
- Expand execution to be able to pull jobs from a central Ansible Tower infrastructure



Red Hat

Red Hat
Ansible Automation
Platform

CLOUD.REDHAT.COM

Engage users with an automation focused experience

Red Hat

Red Hat Ansible Automation Platform

Network | Lines of business | Security

Operations | Infrastructure | Developers

Engage

Knowledge
Automation Analytics

Collaborate
Automation Hub

Trusted
Certified content

Accelerate
Collections

Scale

Ansible Tower: Operate & control at scale

Create

Ansible Engine: Universal language of automation

Fueled by an open source community

Red Hat

Automation Analytics: What is it?

SaaS (Software as a Service) on cloud.redhat.com

Analytics for all Ansible Tower clusters for an organization

Includes:

- visual dashboard
- health notifications
- organization statistics

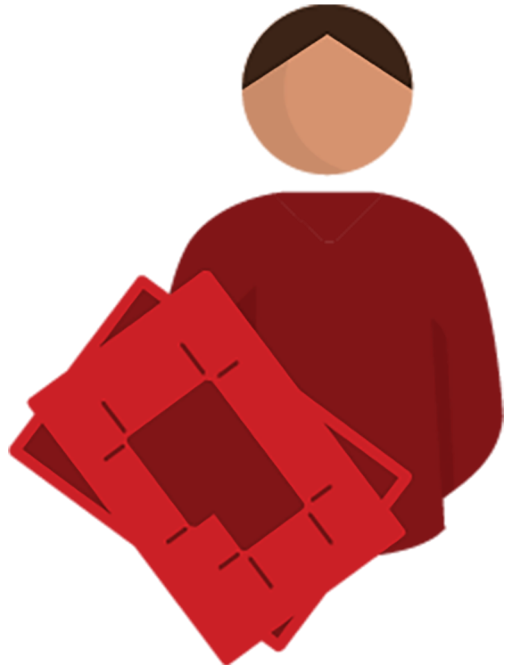
Red Hat

Automation Analytics: What does it provide?

Enables an Automation Center of Excellence

- View information about automation health, usage and performance across your enterprise.

- Gain information about automation in your enterprise:
  - Which organizations are using the most automation?
  - Utilization rates
  - Enterprise-wide success and failure rates for automation

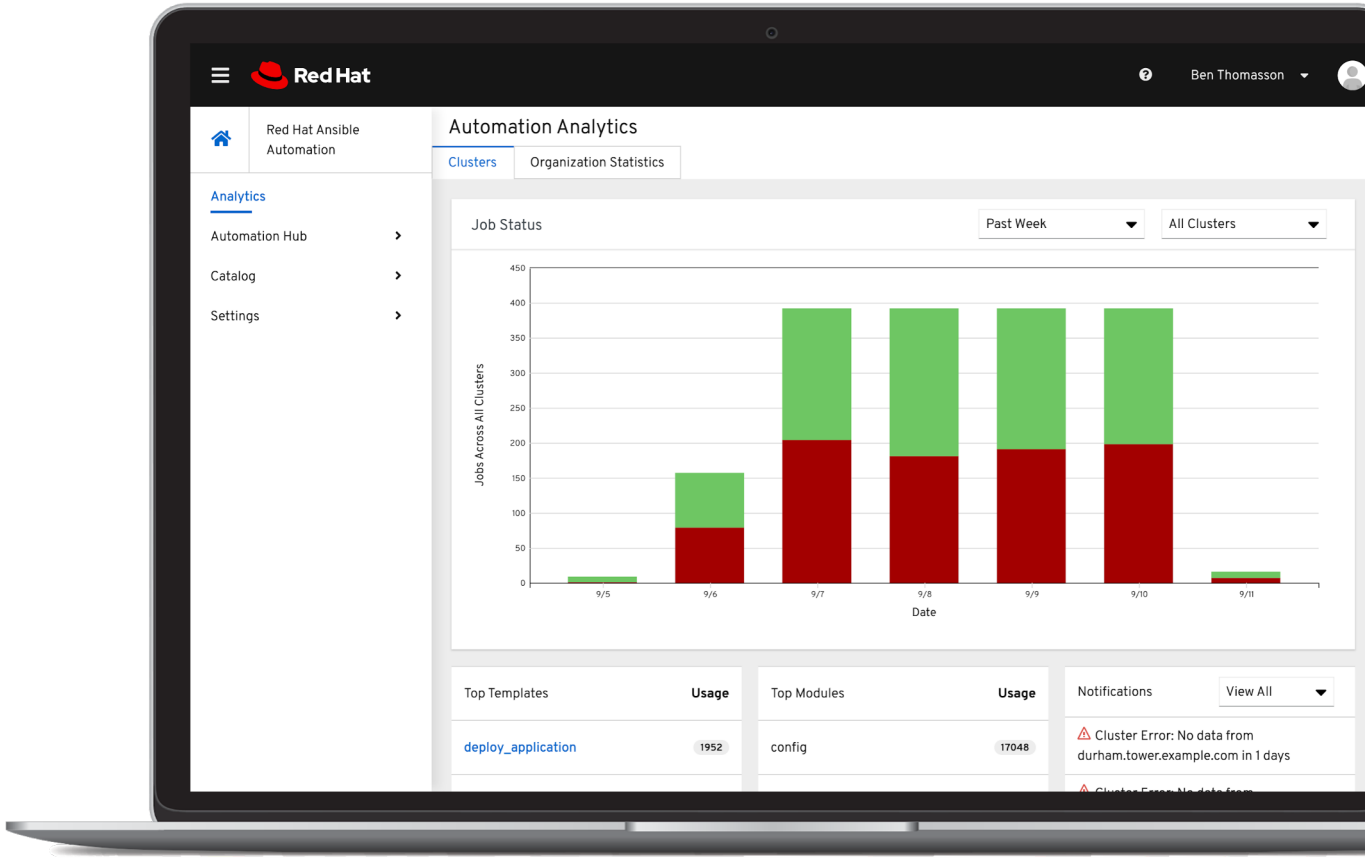


Red Hat

Analytics dashboard

Information across all clusters for an enterprise:

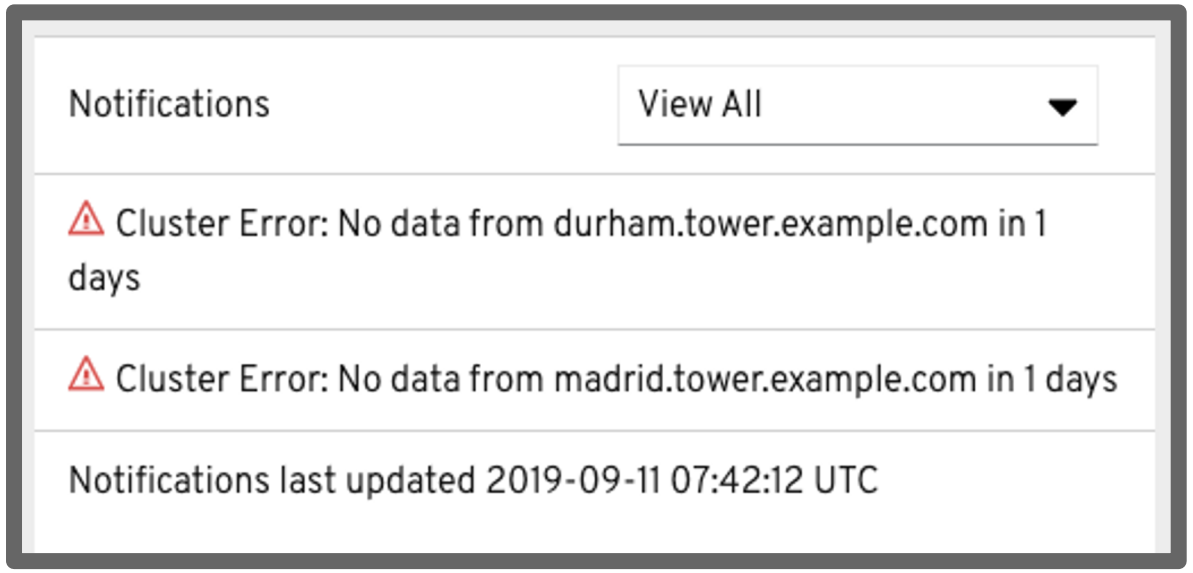
- Job Status graph
- Top Job Templates
- Top Modules



Red Hat

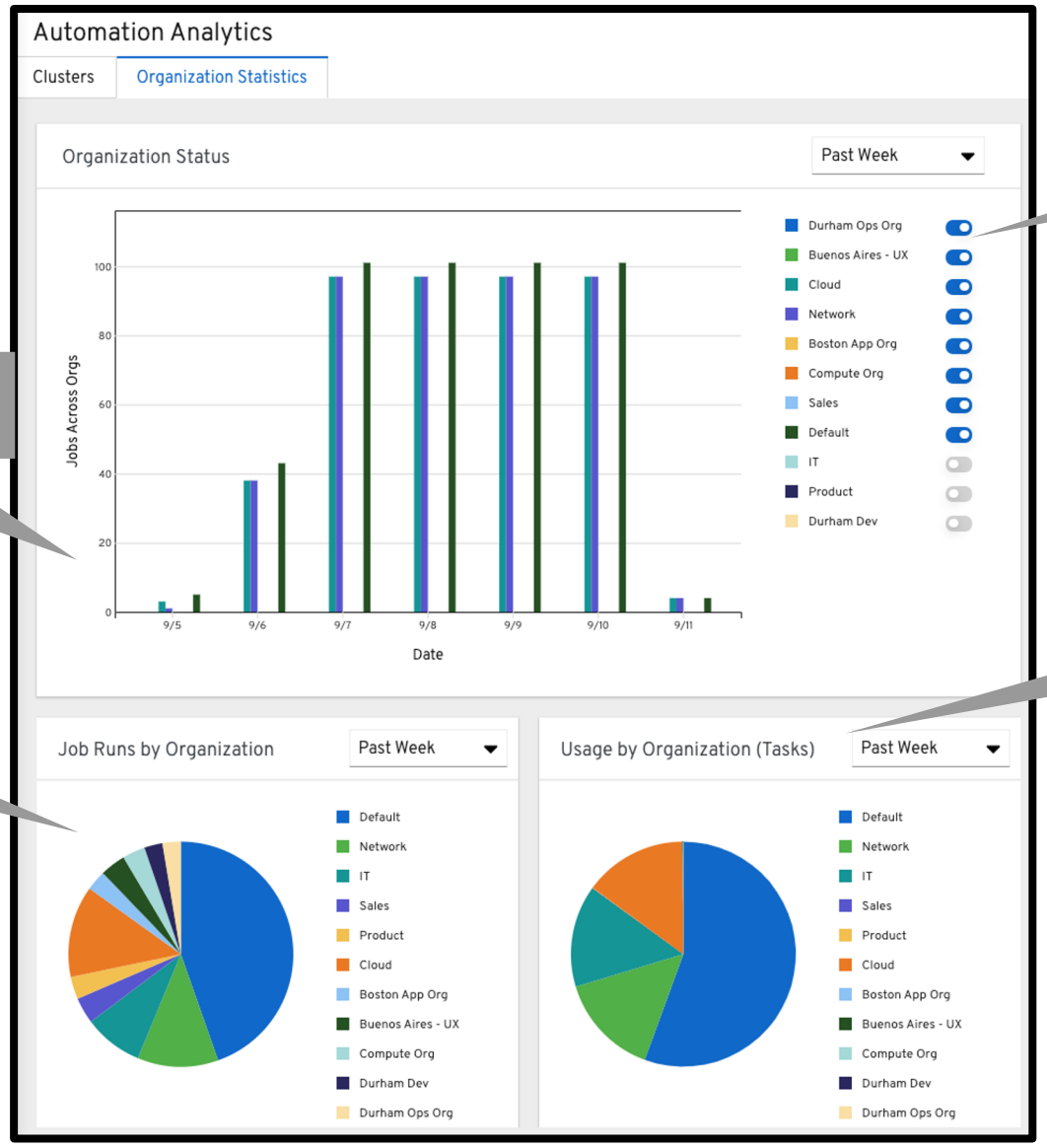
Health notifications

- Ansible Tower Cluster is down
- Node (within a cluster) is down
- Last time data was updated
- Near license count



Red Hat

Organizational statistics



Filter by Organization

Job Status by Organization

Usage by Organization

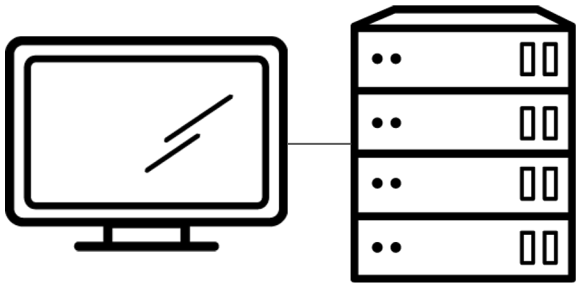
Job Runs by Organization

Red Hat

Dashboard comparison

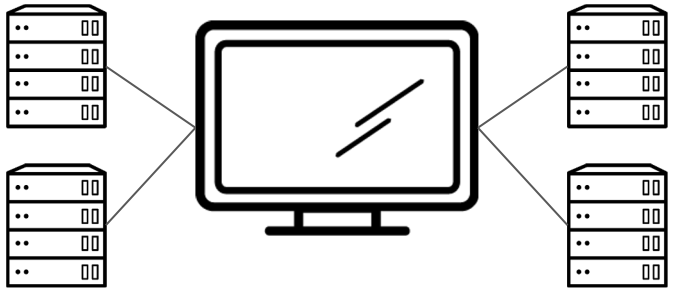
Ansible Tower

- Recent job templates
- No module data
- One cluster



Automation Analytics

- Top job templates
- Top modules
- All clusters
- Filter by cluster



Red Hat

Ansible Content Collections

Simplified and consistent content delivery

Provides quick benefit by lowering barriers to automation

Streamlines tech partners providing direct-to-user automation

Simplifies internal collaboration, distribution, versioning

Ability to distribute, share and consume content at your own pace

Red Hat

Ansible Content Collection example

Directory Layout

```
.
├── galaxy.yml
├── plugins
│   ├── action
│   │   └── ping.py
│   ├── module_utils
│   │   └── pingutils.py
│   └── modules
│       └── ping.py
└── roles
    ├── ping_bootstrap
    │   ├── defaults
    │   ├── filters
    │   ├── meta
    │   ├── tasks
    │   └── vars
    └── ping_deploy
        ├── defaults
        ├── meta
        └── tasks
```

In a playbook

```yaml
hosts: somehosts
collections:
  - custom.pinger
  - redhat.open_ping

tasks:
  - custom.pinger.ping:

  - ansible.builtin.ping: # use only the ping packaged in core

  - ansible.legacy.ping: # use core or library(etc)/ping.py
    when: thing | custom.pinger.filter == 42

  - ping: # searches collections “path” otherwise…
          # still works, == ansible.legacy.ping:
```

Red Hat

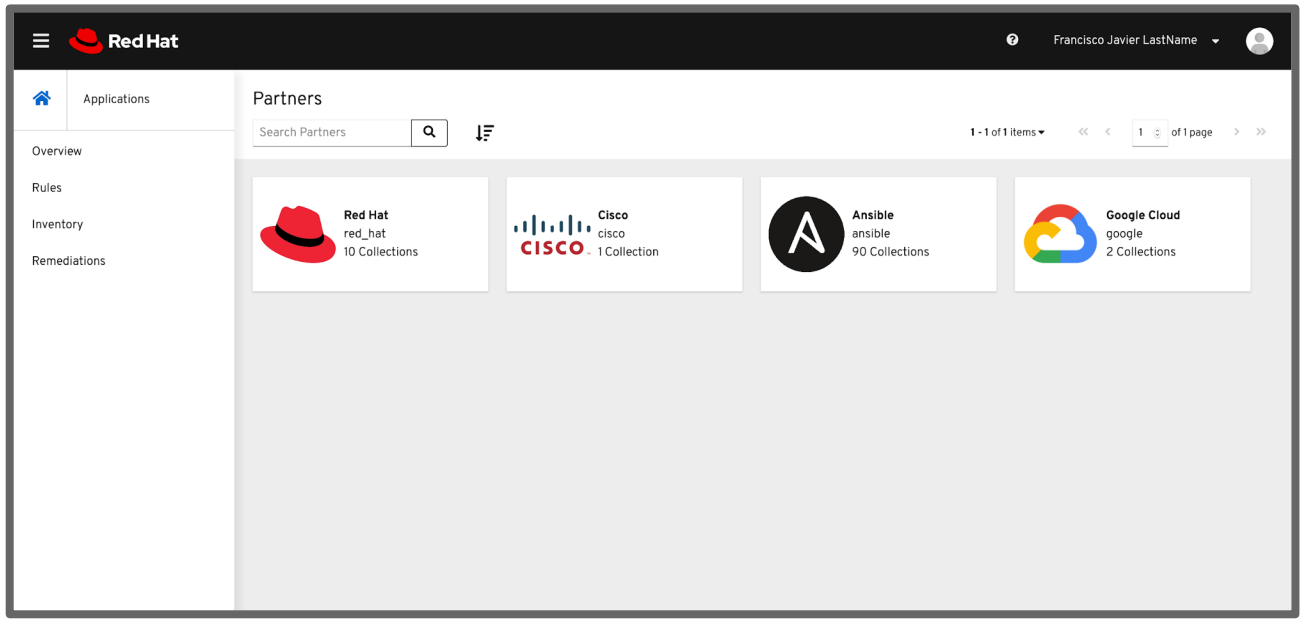
Automation Hub

Discover, publish, and manage Collections

Quickly discover available Red Hat and certified content through Collections.

Manage and test your organization's view of available content.*

Manage your locally available automation via on-premise.*



Red Hat

Next steps:

Get started

ansible.com/get-started

ansible.com/tower-trial

Join the community

ansible.com/community

Workshops and training

ansible.com/workshops

Red Hat Training

Share your story

Follow us @Ansible

Friend us on Facebook

Red Hat

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.


linkedin.com/company/red-hat


youtube.com/user/AnsibleAutomation


facebook.com/AnsibleAutomation


twitter.com/Ansible

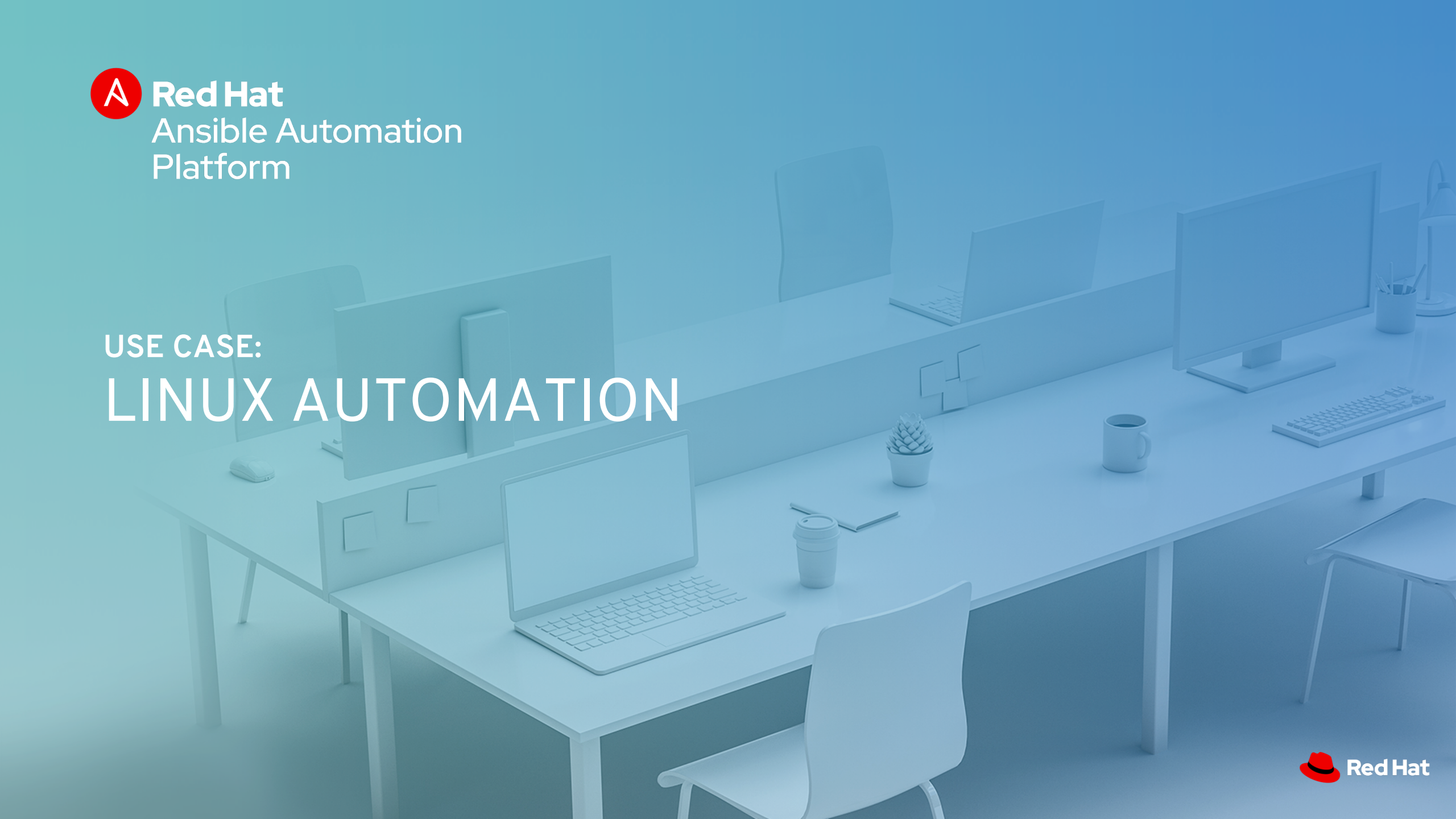
Red Hat
Ansible Automation
Platform

Strategic Use Cases

Red Hat
Ansible Automation
Platform

USE CASE:

LINUX AUTOMATION



Red Hat

LINUX AUTOMATION

150+

Linux Modules

AUTOMATE EVERYTHING LINUX

Red Hat Enterprise Linux, BSD, Debian, Ubuntu and many more!

ONLY REQUIREMENTS:
Python 2 (2.6 or later)
or Python 3 (3.5 or later)

ansible.com/get-started

Red Hat

AUTOMATION FOR EVERYONE: **SYSTEM ADMINISTRATORS**

```
---
- name: upgrade rhel packages
  hosts: rhel

  tasks:
    - name: upgrade all packages
      yum:
          name: '*'
          state: latest
```

Red Hat

AUTOMATION FOR EVERYONE: **SYSTEM ADMINISTRATORS**

```
---
- name: reboot rhel hosts
  hosts: rhel

  tasks:
    - name: reboot the machine
      reboot:
```

AUTOMATION FOR EVERYONE: **SYSTEM ADMINISTRATORS**

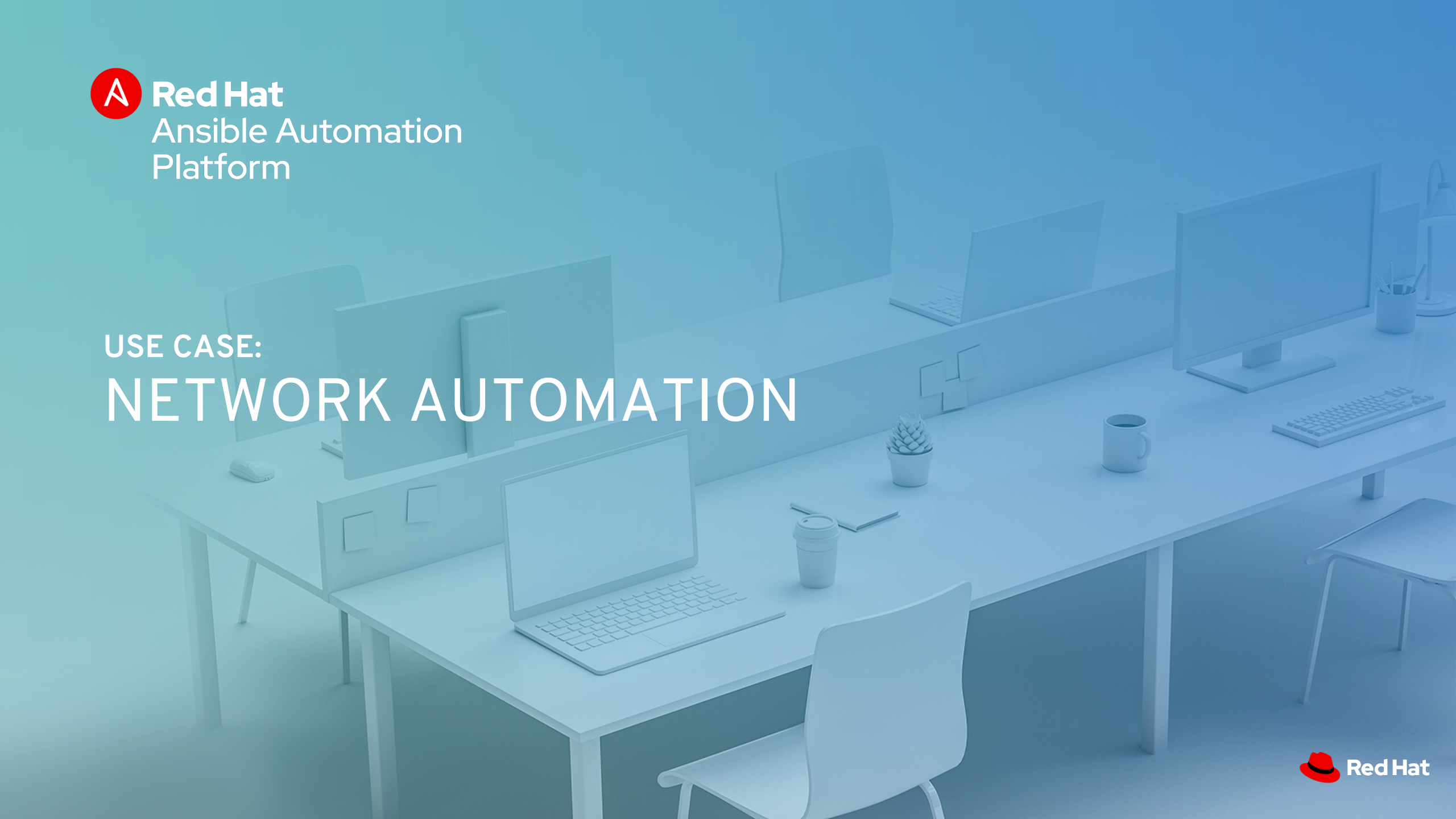
```
---
- name: check services on rhel hosts
  hosts: rhel
  become: yes

  tasks:
   - name: ensure nginx is started
     service:
     name: nginx
     state: started
```

Red Hat

Red Hat
Ansible Automation
Platform

USE CASE:

NETWORK AUTOMATION



Red Hat

ANSIBLE NETWORK AUTOMATION

65+
Network
Platforms

1000+
Network
Modules

15*
Galaxy
Network Roles

ansible.com/for/networks
galaxy.ansible.com/ansible-network

**Roles developed and maintained by Ansible Network Engineering*

Red Hat

WHY AUTOMATE YOUR NETWORK?

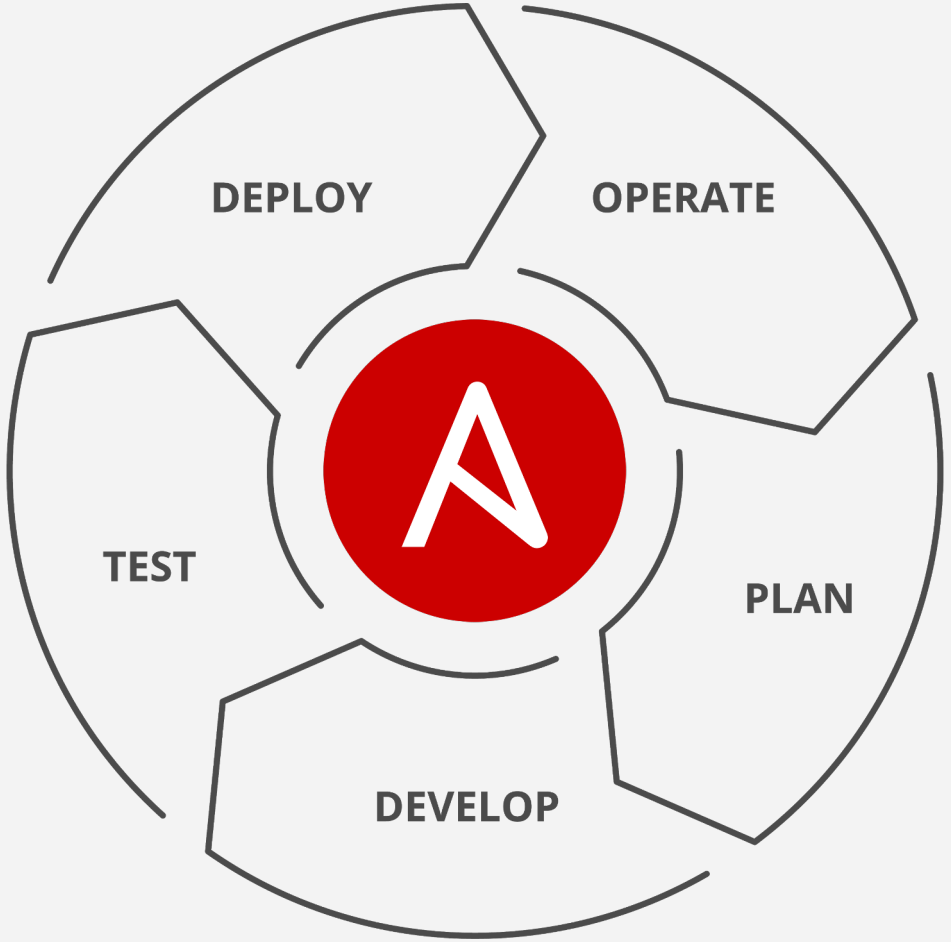
PLAN AND PROTOTYPE VIRTUALLY
Use tasks as reusable building blocks

USE YOUR CURRENT DEVELOPMENT PRACTICES
Agile, DevOps, Waterfall

GO BEYOND THE “PING” TEST
Integrate with formal testing platforms

BE CONFIDENT DURING DEPLOYMENT
Validate changes were successful

ENSURE AN ON-GOING STEADY-STATE



Red Hat

AUTOMATION FOR EVERYONE: **NETWORK ENGINEERS**

```
---
- hosts: cisco
  gather_facts: false
  connection: network_cli

  tasks:
      - name: show command for cisco
        cli_command:
        command: show ip int br
        register: result

      - name: display result to terminal window
        debug:
        var: result.stdout_lines
```

AUTOMATION FOR EVERYONE: **PLAYBOOK RESULTS**

```
[student3@ansible network_setup]$ ansible-playbook example.yml

PLAY [cisco] ***********************************************************************************************************************************************************

TASK [show command for cisco] ******************************************************************************************************************************************
ok: [rtr2]
ok: [rtr1]

TASK [display result to terminal window] *******************************************************************************************************************************
ok: [rtr1] => {
    "result.stdout_lines": [
        "Interface              IP-Address      OK? Method Status                Protocol",
        "GigabitEthernet1       172.16.22.120   YES DHCP   up                    up      ",
        "VirtualPortGroup0      192.168.35.101  YES TFTP   up                    up"
    ]
}
ok: [rtr2] => {
    "result.stdout_lines": [
        "Interface              IP-Address      OK? Method Status                Protocol",
        "GigabitEthernet1       172.17.1.107    YES DHCP   up                    up      ",
        "VirtualPortGroup0      192.168.35.101  YES TFTP   up                    up"
    ]
}

PLAY RECAP *************************************************************************************************************************************************************
rtr1                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0
rtr2                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0

[student3@ansible network_setup]$
```

Red Hat

AUTOMATION FOR EVERYONE: **NETWORK ENGINEERS**

```
---
- hosts: juniper
  gather_facts: false
  connection: network_cli

  tasks:
      - name: show command for juniper
        cli_command:
        command: show interfaces terse em1
        register: result

      - name: display result to terminal window
        debug:
        var: result.stdout_lines
```

Red Hat

AUTOMATION FOR EVERYONE: **PLAYBOOK RESULTS**

```
[student3@ansible network_setup]$ ansible-playbook junos-example.yml

PLAY [juniper] *****************************************************************************************************************************************

TASK [show command for juniper] ************************************************************************************************************************
ok: [rtr3]
ok: [rtr4]

TASK [display result to terminal window] ***************************************************************************************************************
ok: [rtr3] => {
    "result.stdout_lines": [
        "Interface               Admin Link Proto    Local                 Remote",
        "em1                     up    up",
        "em1.0                   up    up   inet     10.0.0.4/8      ",
        "                                   128.0.0.1/2     ",
        "                                   128.0.0.4/2     ",
        "                                inet6    fe80::5254:ff:fe12:bdfe/64",
        "                                         fec0::a:0:0:4/64",
        "                                tnp      0x4"
    ]
}
ok: [rtr4] => {
    "result.stdout_lines": [
        "Interface               Admin Link Proto    Local                 Remote",
        "em1                     up    up",
        "em1.0                   up    up   inet     10.0.0.4/8      ",
        "                                   128.0.0.1/2     ",
        "                                   128.0.0.4/2     ",
        "                                inet6    fe80::5254:ff:fe12:bdfe/64",
        "                                         fec0::a:0:0:4/64",
        "                                tnp      0x4"
    ]
}

PLAY RECAP *********************************************************************************************************************************************
rtr3                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0
rtr4                       : ok=2    changed=0    unreachable=0    failed=0    skipped=0

[student3@ansible network_setup]$
```

Red Hat

Red Hat
Ansible Automation
Platform

USE CASE:

WINDOWS AUTOMATION

Red Hat

WINDOWS AUTOMATION

90+

Windows Modules

1,300+

Powershell DSC resources

ansible.com/windows

Red Hat

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

```
---
- name: windows playbook
  hosts: new_servers


  tasks:
  - name: ensure local admin account exists
    win_user:
      name: localadmin
      password: '{{ local_admin_password }}'
      groups: Administrators
```

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

```
---
- name: windows playbook
  hosts: windows_machines


  tasks:
  - name: ensure common tools are installed
    win_chocolatey:
      name: '{{ item }}'
    loop: ['sysinternals', 'googlechrome']
```

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

```
---
- name: update and reboot
  hosts: windows_servers
  tasks:
  - name: ensure common OS updates are current
    win_updates:
    register: update_result

  - name: reboot and wait for host if updates change require it
    win_reboot:
    when: update_result.reboot_required
```

Red Hat

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

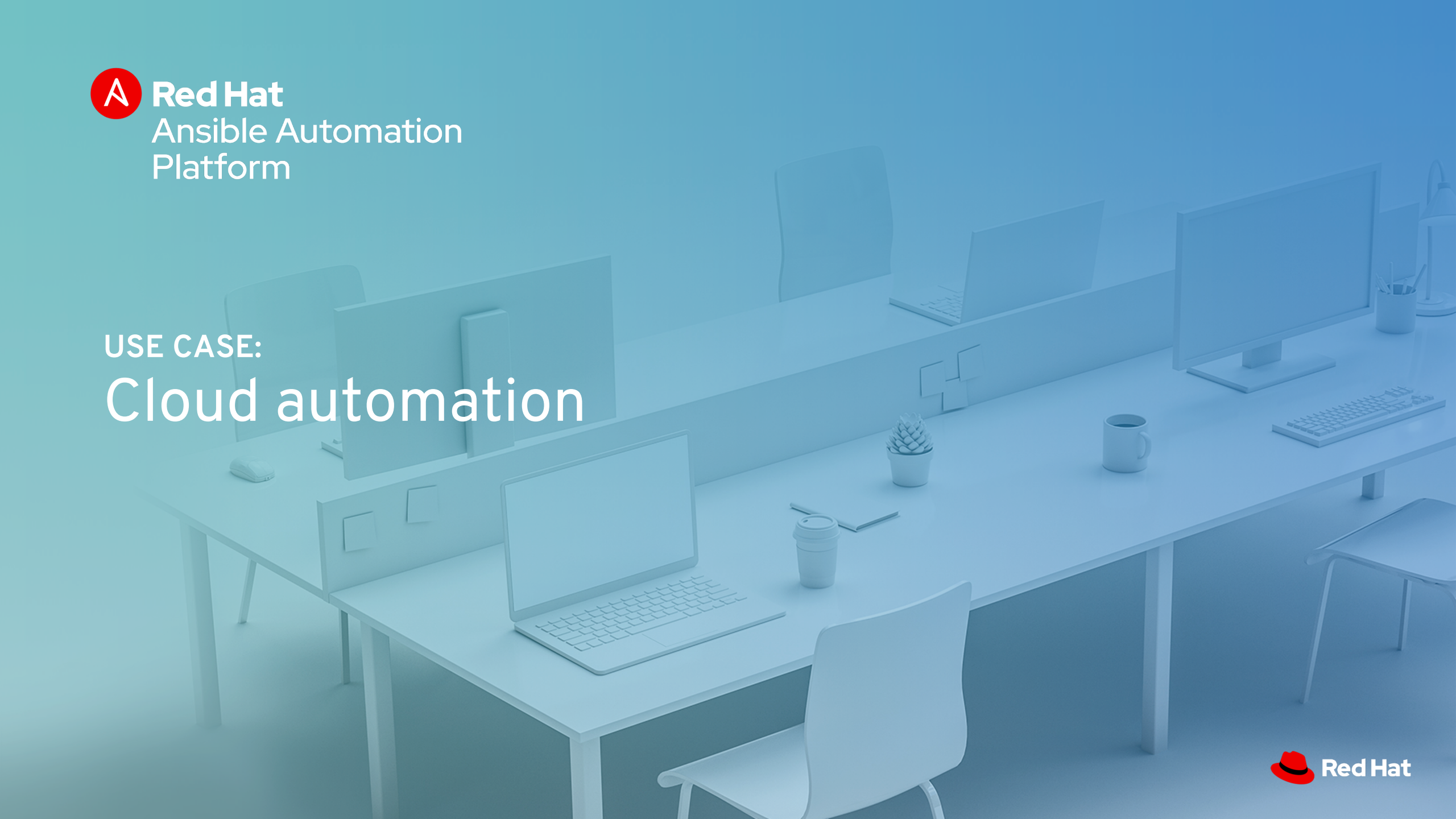
```
---
- name: update domain and reboot
  hosts: windows_servers
  tasks:
  - name: ensure domain membership
    win_domain_membership:
      dns_domain_name: contoso.corp
      domain_admin_user: '{{ domain_admin_username }}'
      domain_admin_password: '{{ domain_admin_password }}'
      state: domain
    register: domain_result

  - name: reboot and wait for host if domain change require it
    win_reboot:
    when: domain_result.reboot_required
```

Red Hat

Red Hat
Ansible Automation
Platform

USE CASE:

Cloud automation



Red Hat

CLOUD AUTOMATION

800+

Cloud Modules

30+

Cloud Platforms

ansible.com/cloud

Red Hat

PLAYBOOK EXAMPLE: **AWS**

```
---
- name: aws playbook
  hosts: localhost
  connection: local

  tasks:
       - name: create AWS VPC ansible-vpc
         ec2_vpc_net:
         name: "ansible-vpc"
         cidr_block: "192.168.0.0/24"
         tags:
           demo: the demo vpc
       register: create_vpc
```

PLAYBOOK EXAMPLE: **AZURE**

```
---
- name: azure playbook
  hosts: localhost
  connection: local

  tasks:
       - name: create virtual network
      azure_rm_virtualnetwork:
        resource_group: myResourceGroup
        name: myVnet
        address_prefixes: "10.0.0.0/16"
```

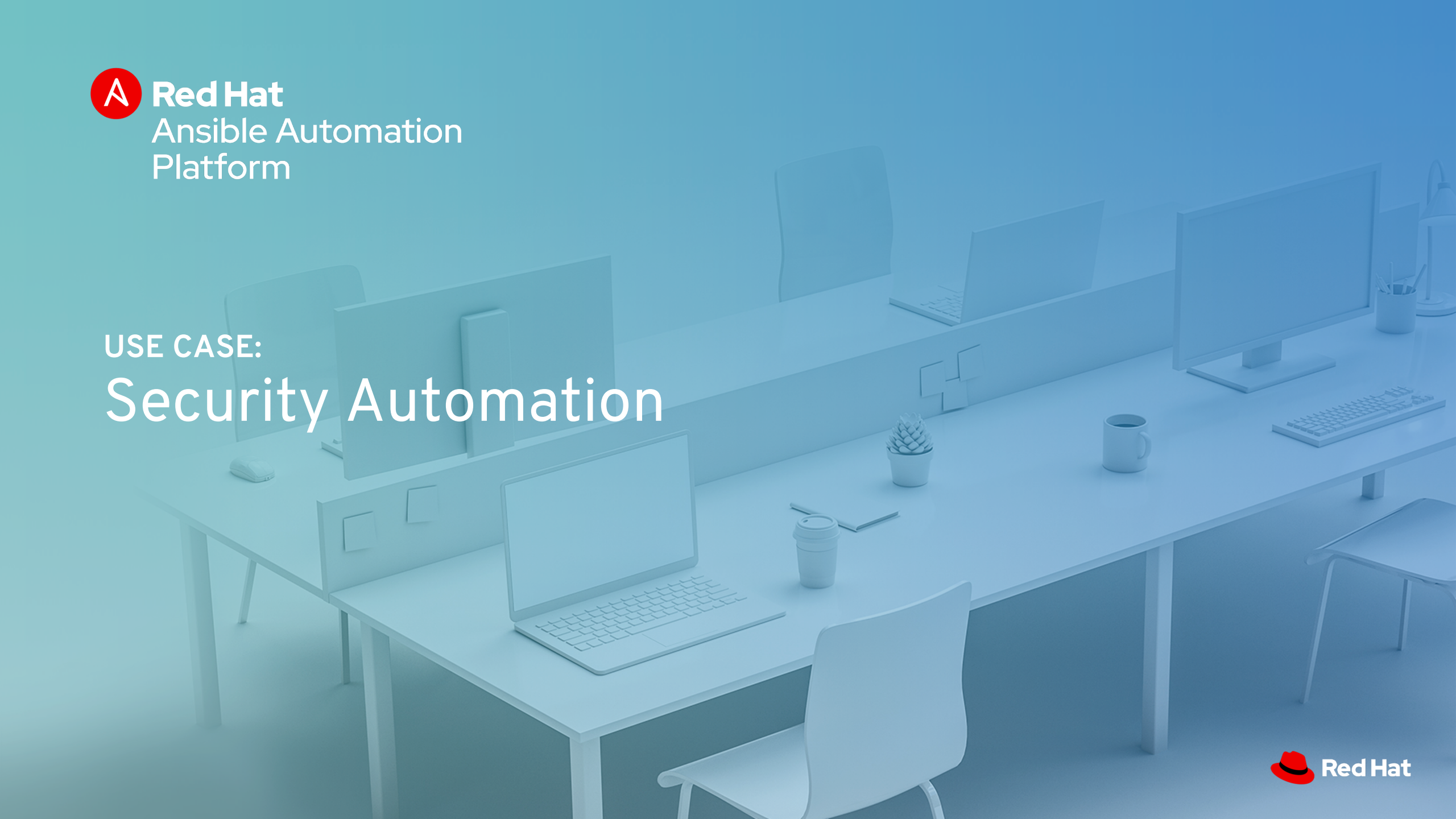
PLAYBOOK EXAMPLE: **RED HAT OPENSTACK**

```
---
- name: openstack playbook
  hosts: localhost
  connection: local

  tasks:
    - name: launch an instance
        os_server:
          name: vm1
          cloud: mordred
          region_name: ams01
          image: Red Hat Enterprise Linux 7.4
          flavor_ram: 4096
```

Red Hat
Ansible Automation
Platform

USE CASE:

Security Automation



Red Hat

What Is It?

Ansible Security Automation is our expansion deeper into the security use case. The goal is to provide a more efficient, streamlined way for security teams to automate their various processes for the identification, search, and response to security events. This is more complex and higher-value than the application of a security baseline (PCI, STIG, CIS) to a server.

Ansible Security Automation is a supported set of Ansible modules, roles and playbooks designed to unify the security response to cyberattacks.

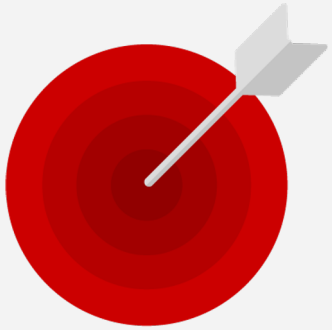
What Does It Do?

Triage Of Suspicious Activities

Enabling programmatic access to log configurations such as destination, verbosity, etc.

Threat Hunting

Automating alerts, correlation searches and signature manipulation

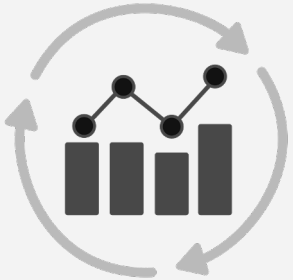


Incident Response

Creating new security policies to whitelist, blacklist or quarantine a machine

Red Hat

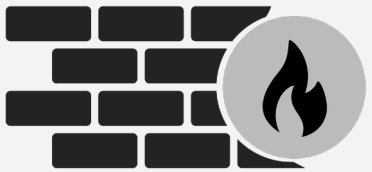
Partners - Security ISVs



Security Information & Events Management

splunk>

IBM



Enterprise Firewalls

Check Point SOFTWARE TECHNOLOGIES LTD

CISCO

f5

FORTINET

Intrusion Detection & Prevention Systems

SNORT®

Check Point SOFTWARE TECHNOLOGIES LTD

FORTINET

Privileged Access Management

CYBERARK

Syncope™

Red Hat

AUTOMATION FOR EVERYONE: **SECURITY OPERATIONS**

```
---
- name: Create access rule in Checkpoint
  hosts: checkpoint
  connection: httpapi

  tasks:
    - name: create access rule
      checkpoint_access_rule:
          layer: Network
          name: "Drop attacker"
          position: top
          source: attacker
          destination: Any
          action: Drop
```

Red Hat

AUTOMATION FOR EVERYONE: **SECURITY OPERATIONS**

```
---
- name: Change QRadar rule state
  hosts: qradar

  tasks:
    - name: get info about qradar rule
      qradar_rule_info:
        name: "Potential DDoS Against Single Host (TCP)"
      register: rule_info

    - name: disable rule by id
      qradar_rule:
        state: disabled
        id: "{{ rule_info.rules[0]['id'] }}"
```

Red Hat

AUTOMATION FOR EVERYONE: **SECURITY OPERATIONS**

```
---
- name: Add Snort rule
  hosts: snort

  tasks:
    - name: Add snort password attack rule
      include_role:
        name: "ansible_security.ids_rule"
      vars:
        rule: "alert tcp any 443 -> 192.168.12.0/24 any"
        state: present
        rules_file: /etc/snort/rules/grab_everything_http.rules
```