Red Hat
Ansible
Automation

Automation for everyone

Ansible technical introduction and overview

Scott C. Danielson
Sr. Solution Architect
sdaniels@redhat.com

Red Hat

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

ACCELERATE

INTEGRATE

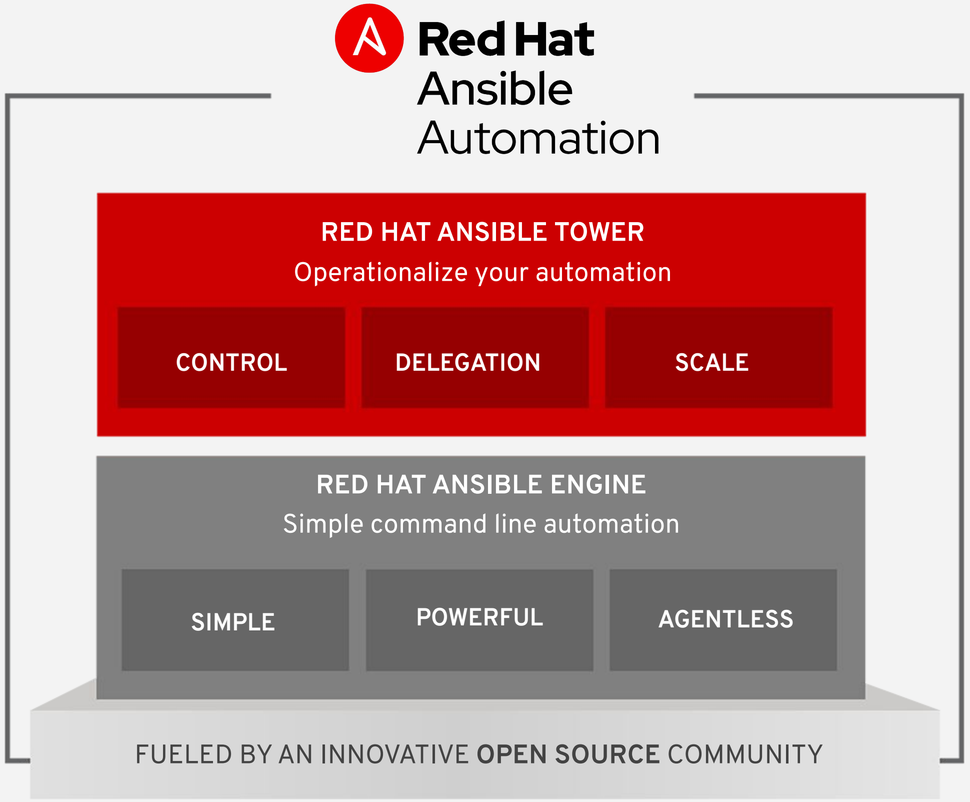
COLLABORATE

What is Ansible Automation?

Ansible Automation is the enterprise **framework** for automating across IT operations.

Ansible Engine runs Ansible Playbooks, the automation **language** that can perfectly describe an IT application infrastructure.

Ansible Tower allows you **scale** IT automation, manage complex deployments and speed productivity.


Red Hat Ansible Automation
RED HAT ANSIBLE TOWER
Operationalize your automation
CONTROL
DELEGATION
SCALE
RED HAT ANSIBLE ENGINE
Simple command line automation
SIMPLE
POWERFUL
AGENTLESS
FUELED BY AN INNOVATIVE OPEN SOURCE COMMUNITY

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

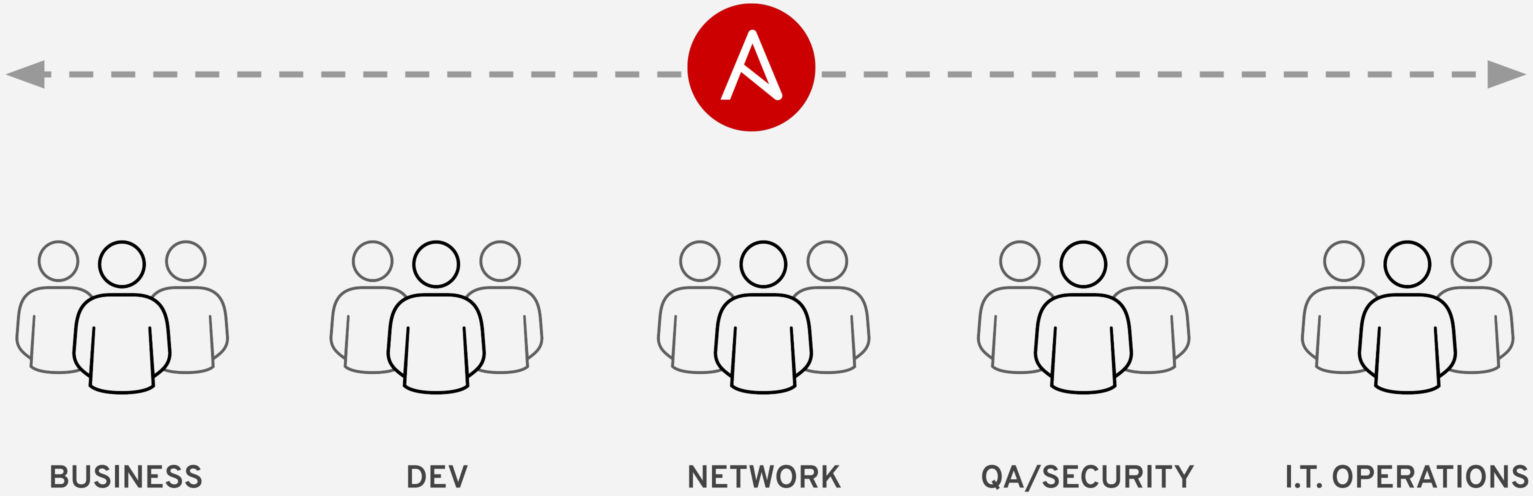
No agents to exploit or update

Get started immediately

More efficient & more secure

Red Hat

Ansible Automation works across teams



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...

Ansible automates technologies you use

Time to automate is measured in minutes

Cloud

AWS
Azure
Digital Ocean
Google
OpenStack
Rackspace
+more

Operating Systems

Rhel And Linux
Unix
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

Arista
A10
Cumulus
Bigswitch
Cisco
Cumulus
Dell
F5
Juniper
Palo Alto
OpenSwitch
+more

Devops

Jira
GitHub
Vagrant
Jenkins
Bamboo
Atlassian
Subversion
Slack
Hipchat
+more

Monitoring

Dynatrace
Airbrake
BigPanda
Datadog
LogicMonitor
Nagios
New Relic
PagerDuty
Sensu
StackDriver
Zabbix
+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%

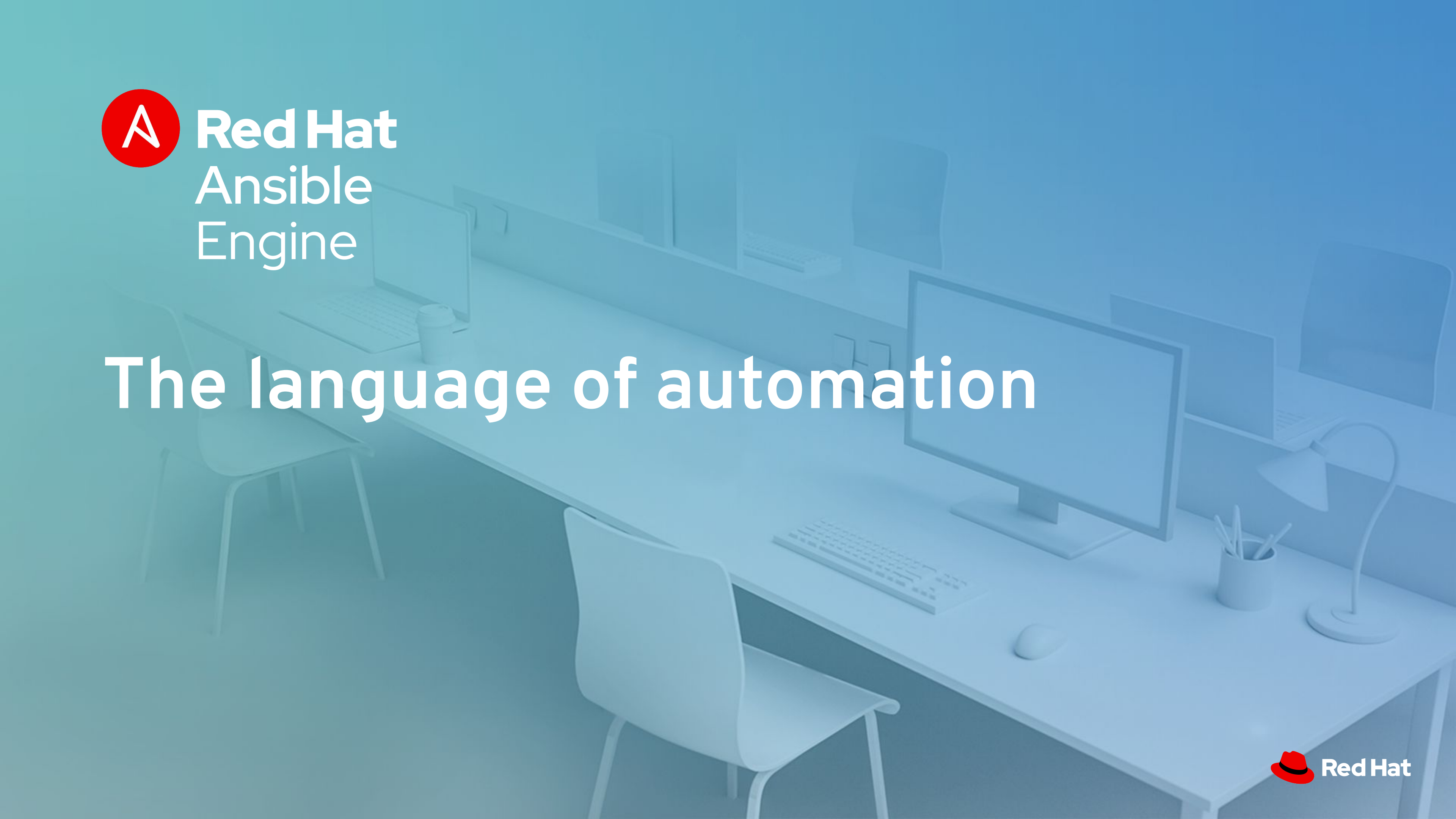
ROI on Ansible Tower

< 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
Engine

The language of automation

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

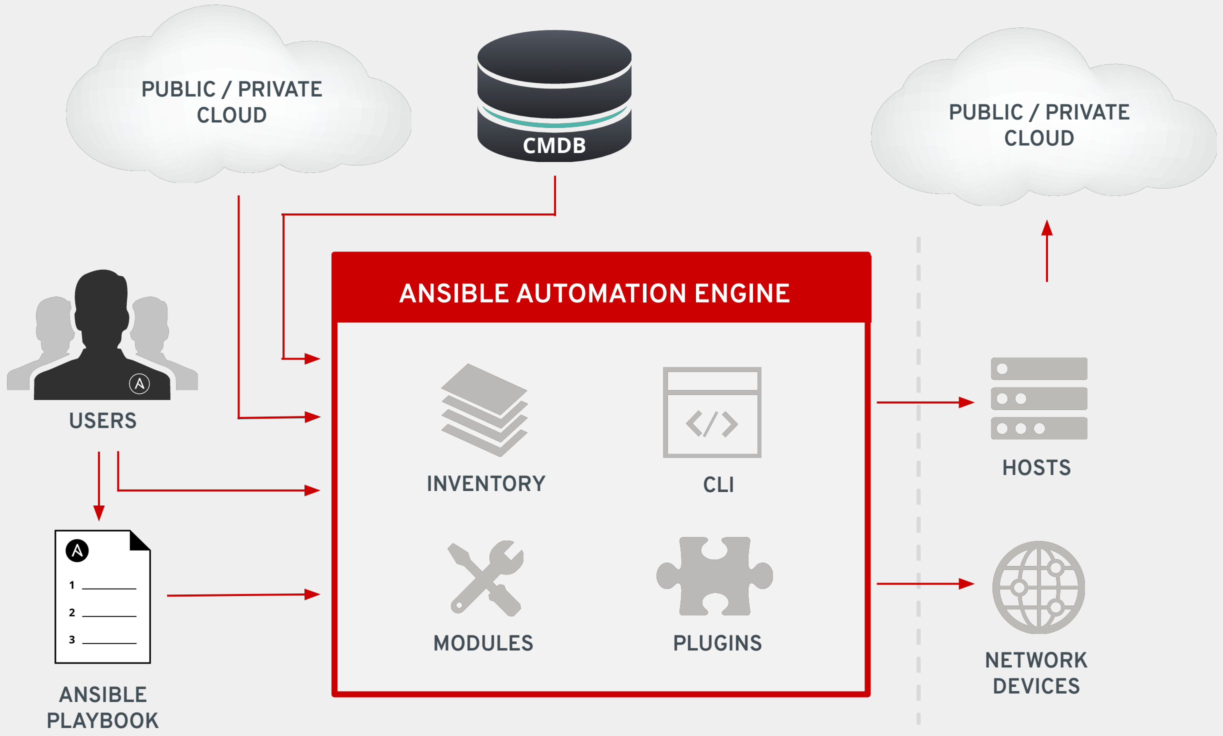
Capture all the servers 100% of the time, regardless of infrastructure, location, etc.

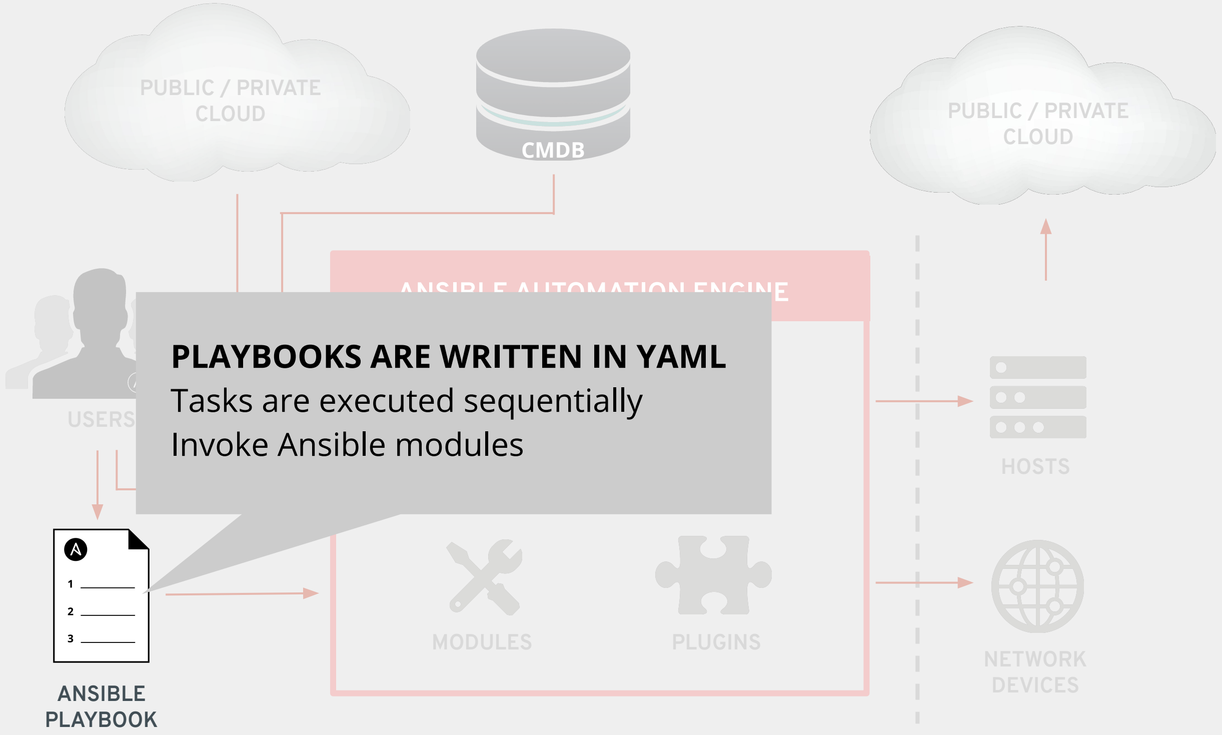
Orchestration plays well with others

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

Red Hat

```yaml
---
- 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
      copy:
        src: files/index.html
        dest: /var/www/html/

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

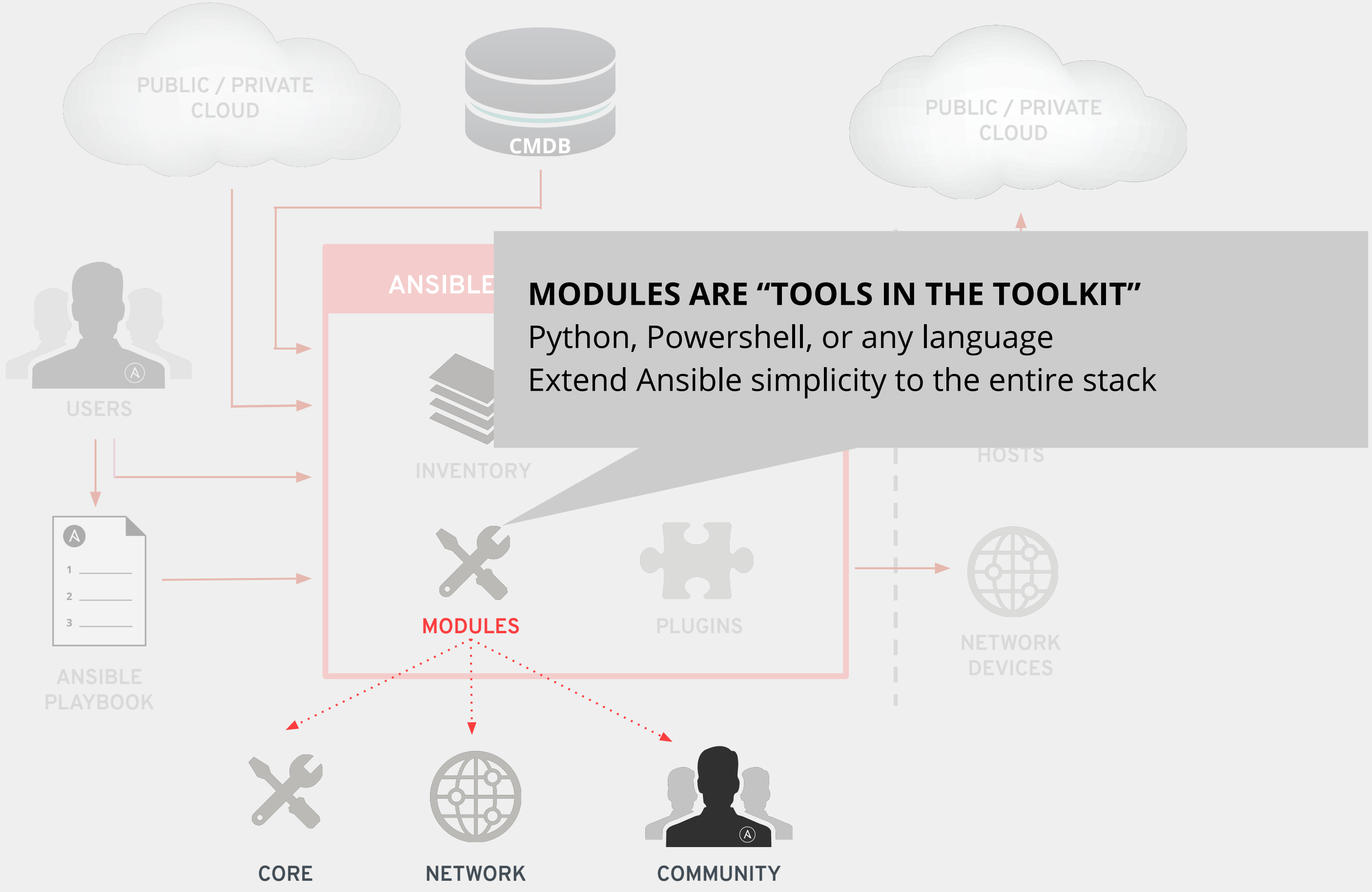

PUBLIC / PRIVATE CLOUD
CMDB
PUBLIC / PRIVATE CLOUD
USERS
ANSIBLE AUTOMATION ENGINE
INVENTORY
CLI
MODULES
PLUGINS
HOSTS
NETWORK DEVICES
1
2
3
ANSIBLE PLAYBOOK



PLAYBOOKS ARE WRITTEN IN YAML

Tasks are executed sequentially

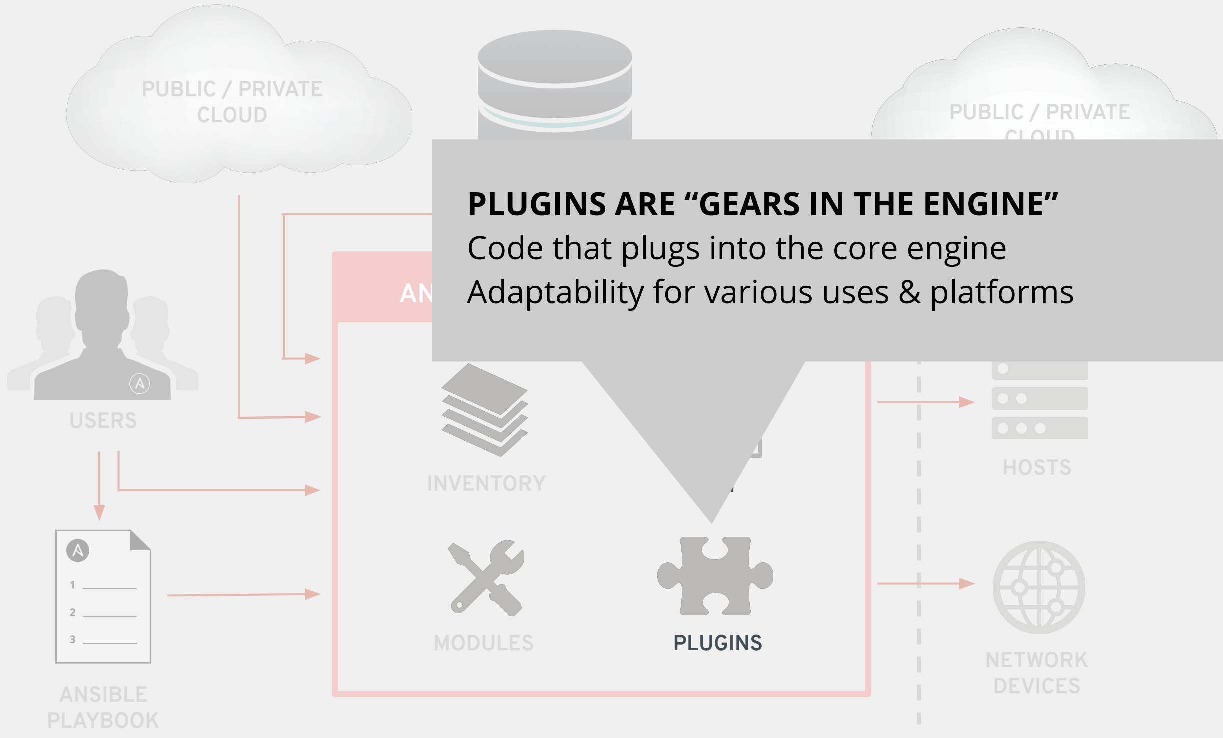
Invoke Ansible modules



MODULES ARE "TOOLS IN THE TOOLKIT"

Python, Powershell, or any language

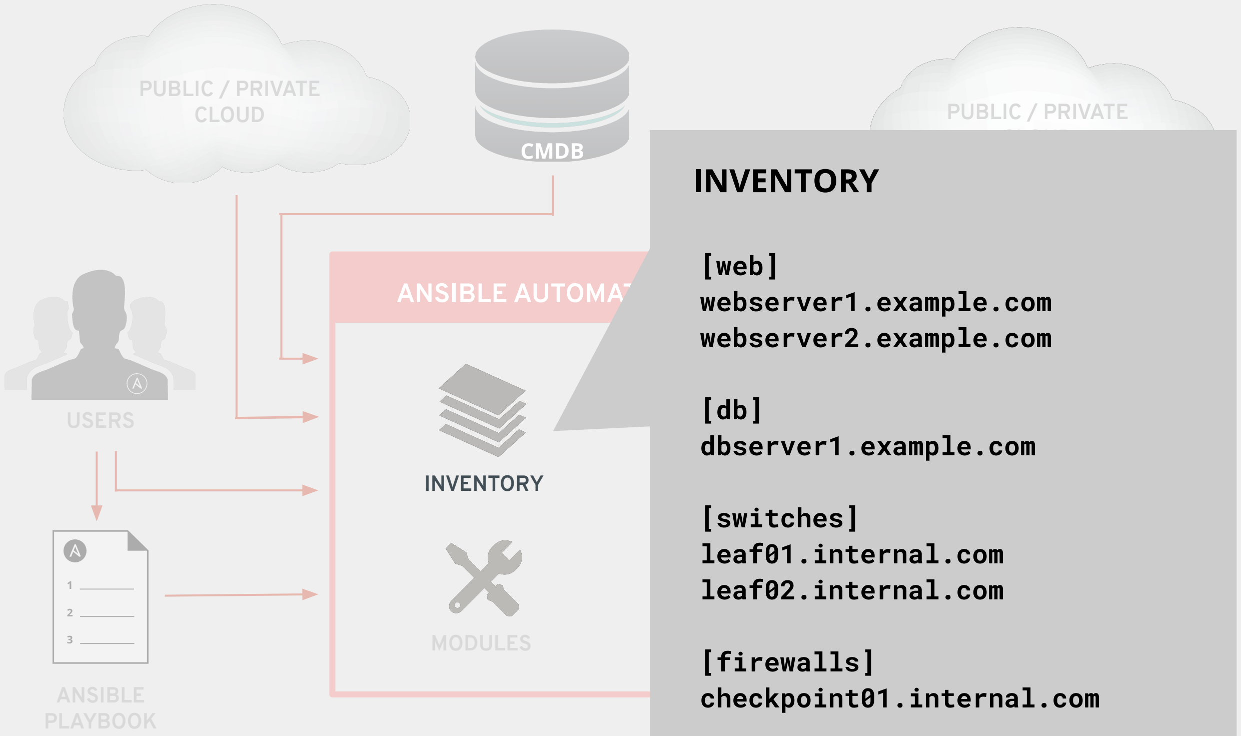
Extend Ansible simplicity to the entire stack



PLUGINS ARE “GEARS IN THE ENGINE”

Code that plugs into the core engine

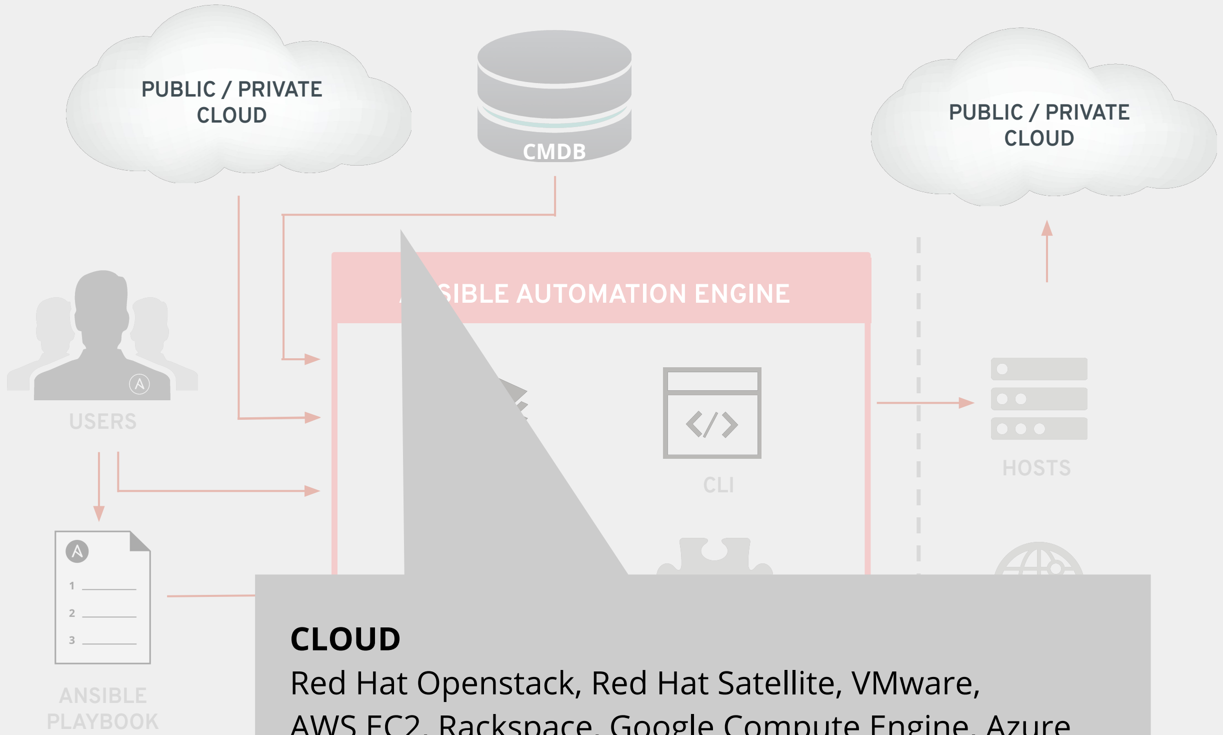
Adaptability for various uses & platforms



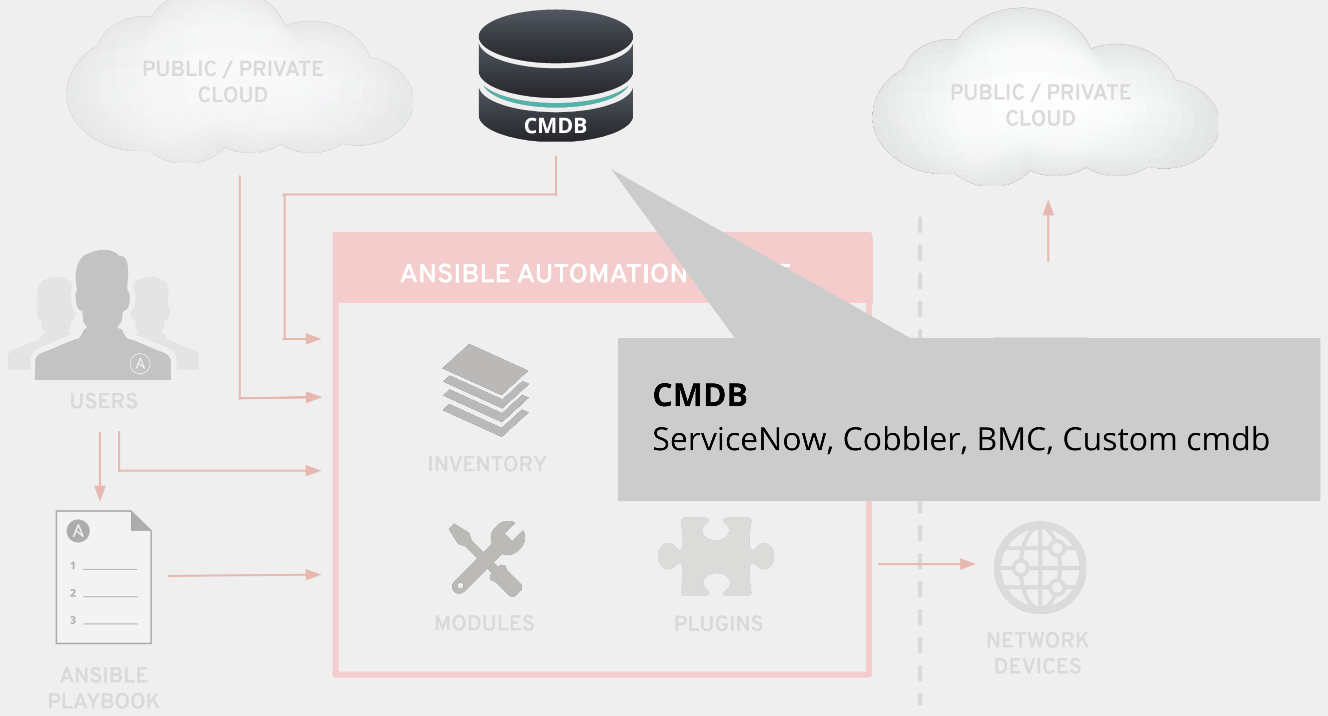
INVENTORY

```
[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
```

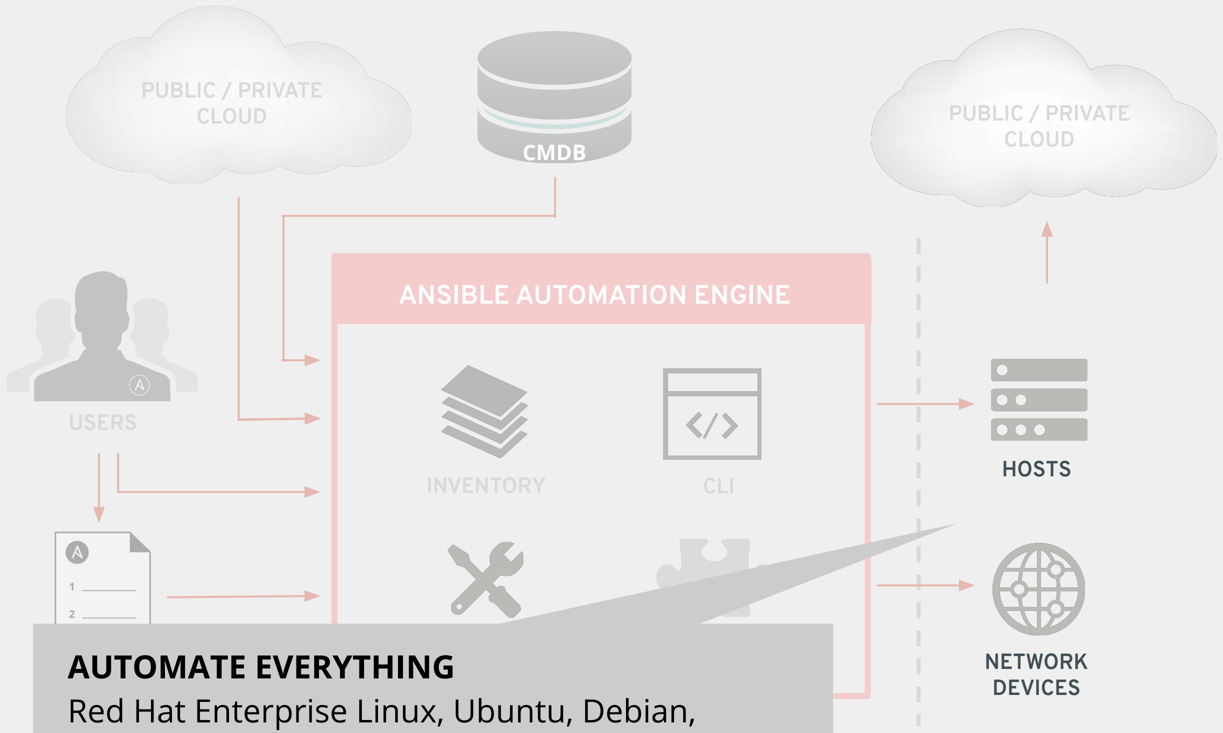
Red Hat



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



CMDB
ServiceNow, Cobbler, BMC, Custom cmdb



AUTOMATE EVERYTHING
Red Hat Enterprise Linux, Ubuntu, Debian,
Cisco routers, Arista switches, Juniper routers,
Windows hosts, Checkpoint firewalls and more

Playbook examples:

GITHUB
github.com/ansible/ansible-examples

LAMP + HAPROXY + NAGIOS
github.com/ansible/ansible-examples/tree/master/lamp_haproxy

WINDOWS
github.com/ansible/ansible-examples/tree/master/windows

SECURITY COMPLIANCE
github.com/ansible/ansible-lockdown

NETWORK AUTOMATION
ansible.com/linklight
github.com/network-automation

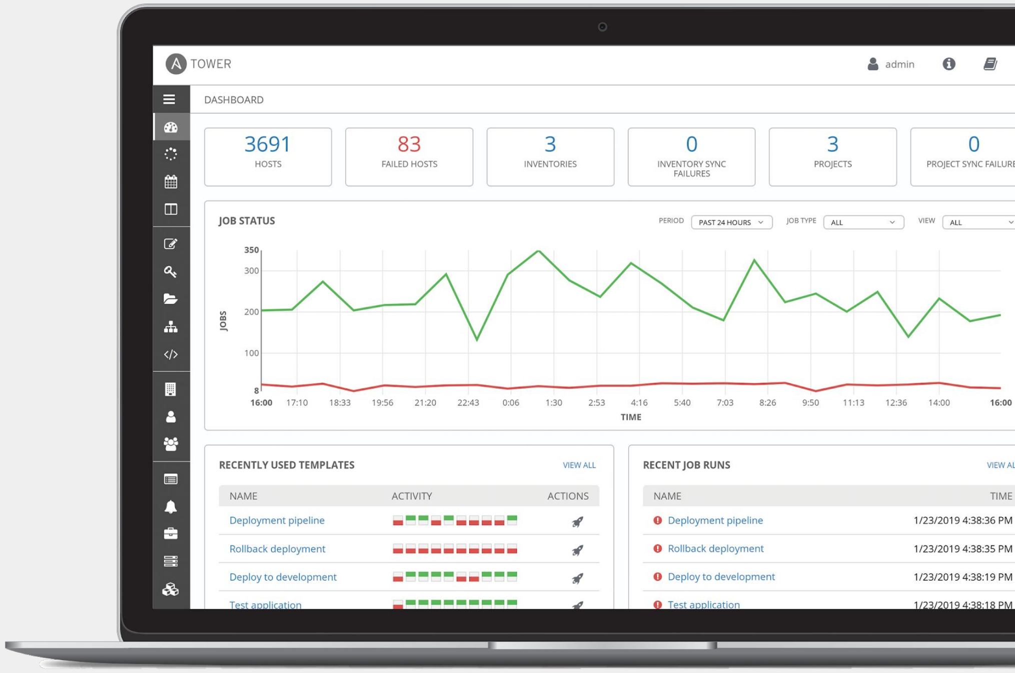
Red Hat
Ansible
Tower

Automation across the enterprise

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



3691	83	3	0	3	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURE

RECENTLY USED TEMPLATES

NAME	ACTIVITY	ACTIONS
Deployment pipeline		
Rollback deployment		
Deploy to development		
Test application		

RECENT JOB RUNS

NAME	TIME
Deployment pipeline	1/23/2019 4:38:36 PM
Rollback deployment	1/23/2019 4:38:35 PM
Deploy to development	1/23/2019 4:38:19 PM
Test application	1/23/2019 4:38:18 PM

Red Hat

Red Hat Ansible Tower

RBAC

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

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.

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.

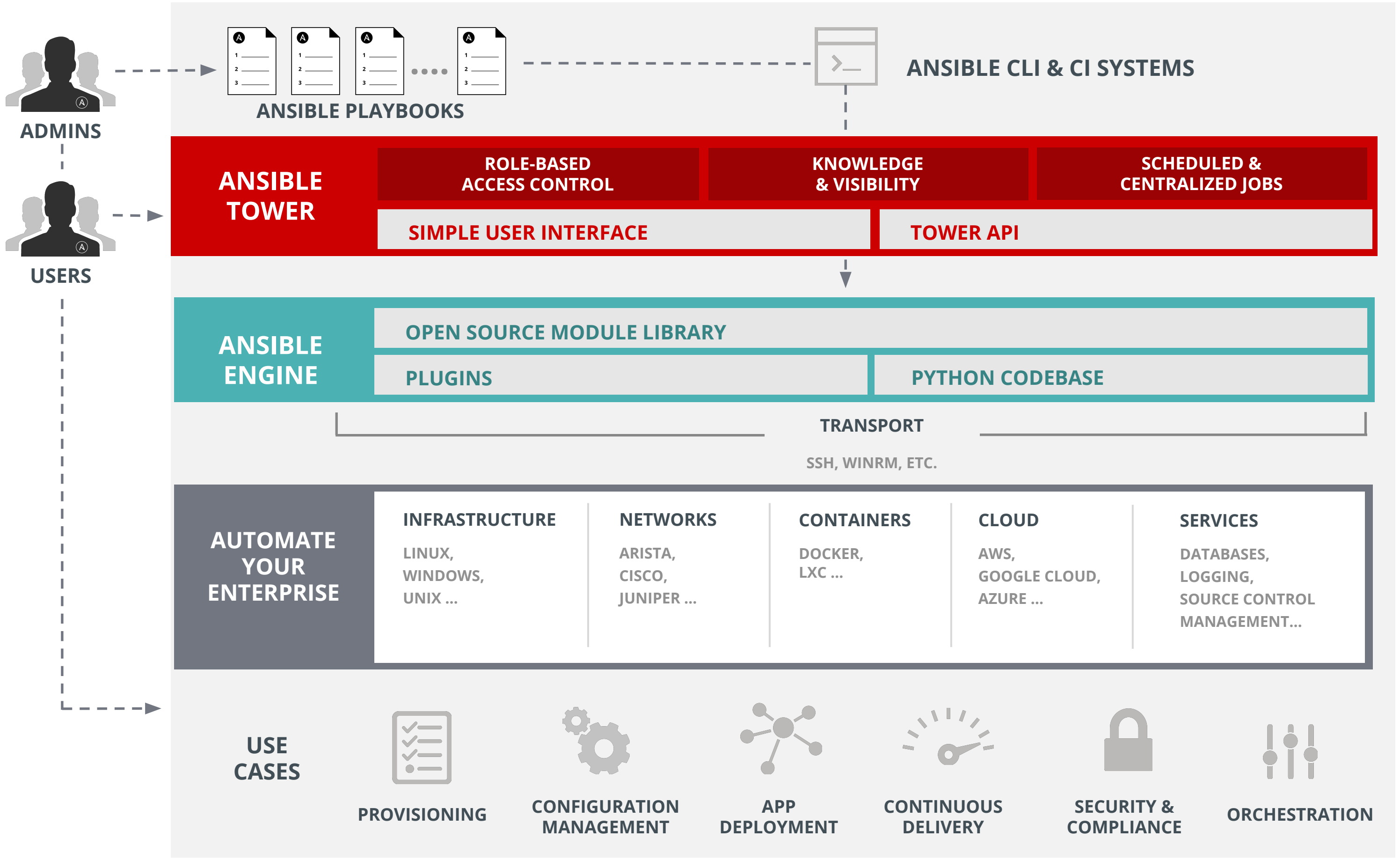
Enterprise integrations

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.

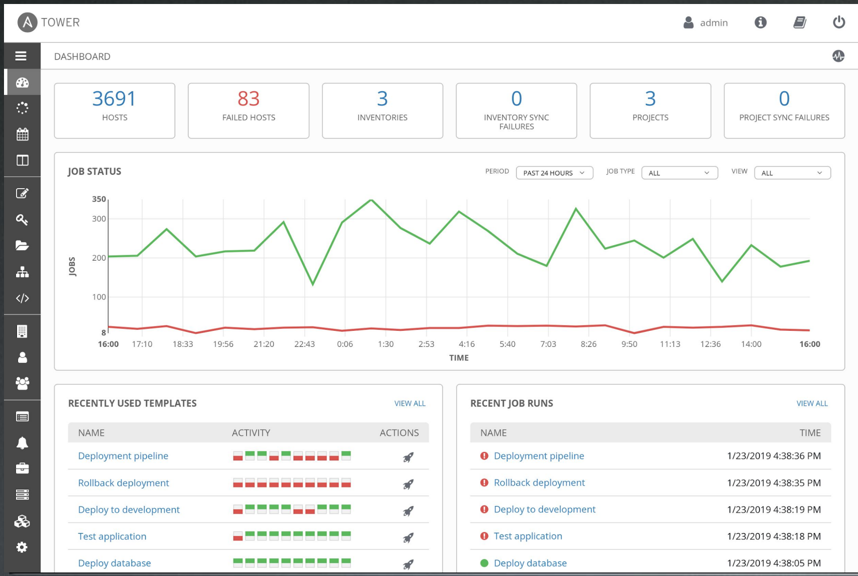
Red Hat


ADMINS
ANSIBLE PLAYBOOKS
ANSIBLE CLI & CI SYSTEMS
USERS
ANSIBLE TOWER
ROLE-BASED ACCESS CONTROL
KNOWLEDGE & VISIBILITY
SCHEDULED & CENTRALIZED JOBS
SIMPLE USER INTERFACE
TOWER API
ANSIBLE ENGINE
OPEN SOURCE MODULE LIBRARY
PLUGINS
PYTHON CODEBASE
TRANSPORT
SSH, WINRM, ETC.
AUTOMATE YOUR ENTERPRISE
INFRASTRUCTURE
LINUX, WINDOWS, UNIX ...
NETWORKS
ARISTA, CISCO, JUNIPER ...
CONTAINERS
DOCKER, LXC ...
CLOUD
AWS, GOOGLE CLOUD, AZURE ...
SERVICES
DATABASES, LOGGING, SOURCE CONTROL MANAGEMENT...
USE CASES
PROVISIONING
CONFIGURATION MANAGEMENT
APP DEPLOYMENT
CONTINUOUS DELIVERY
SECURITY & COMPLIANCE
ORCHESTRATION

Red Hat Ansible Tower

FEATURE OVERVIEW:

Control



3691	83	3	0	3	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

RECENTLY USED TEMPLATES — VIEW ALL

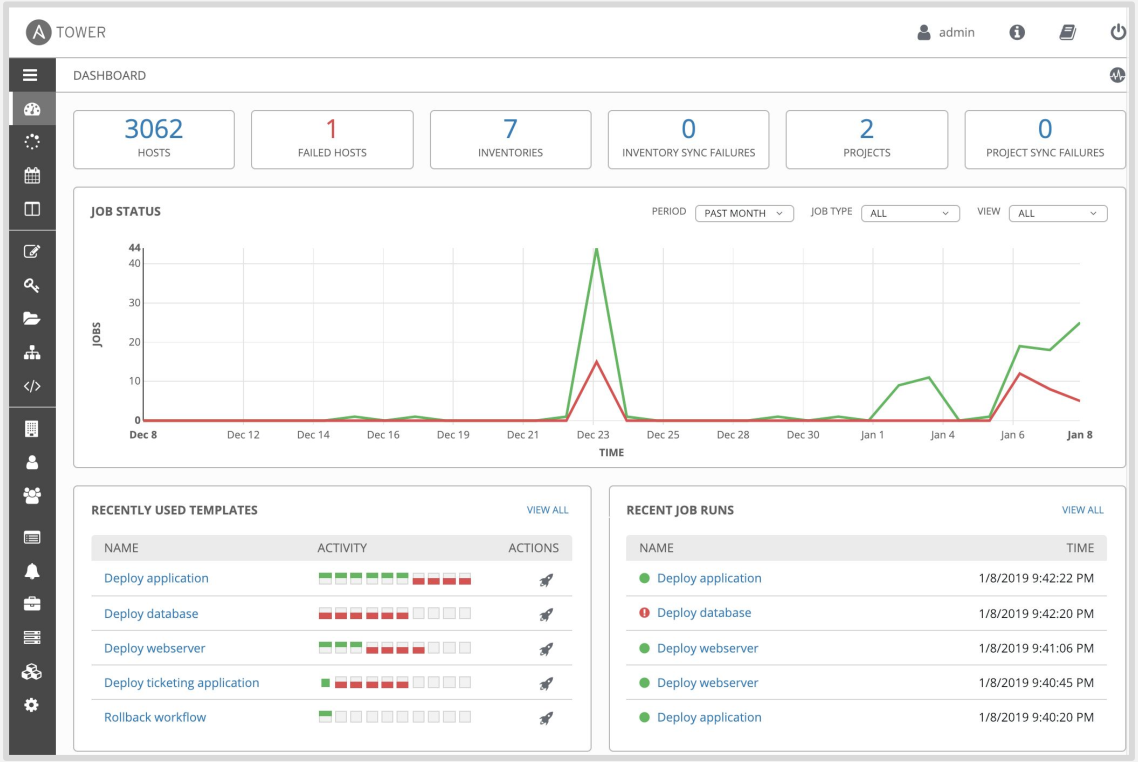
NAME	ACTIVITY	ACTIONS
Deployment pipeline		
Rollback deployment		
Deploy to development		
Test application		
Deploy database		

RECENT JOB RUNS — VIEW ALL

NAME	TIME
Deployment pipeline	1/23/2019 4:38:36 PM
Rollback deployment	1/23/2019 4:38:35 PM
Deploy to development	1/23/2019 4:38:19 PM
Test application	1/23/2019 4:38:18 PM
Deploy database	1/23/2019 4:38:05 PM

Red Hat

ANSIBLE TOWER FEATURES: YOUR ANSIBLE DASHBOARD



3062	1	7	0	2	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

PERIOD: PAST MONTH — JOB TYPE: ALL — VIEW: ALL

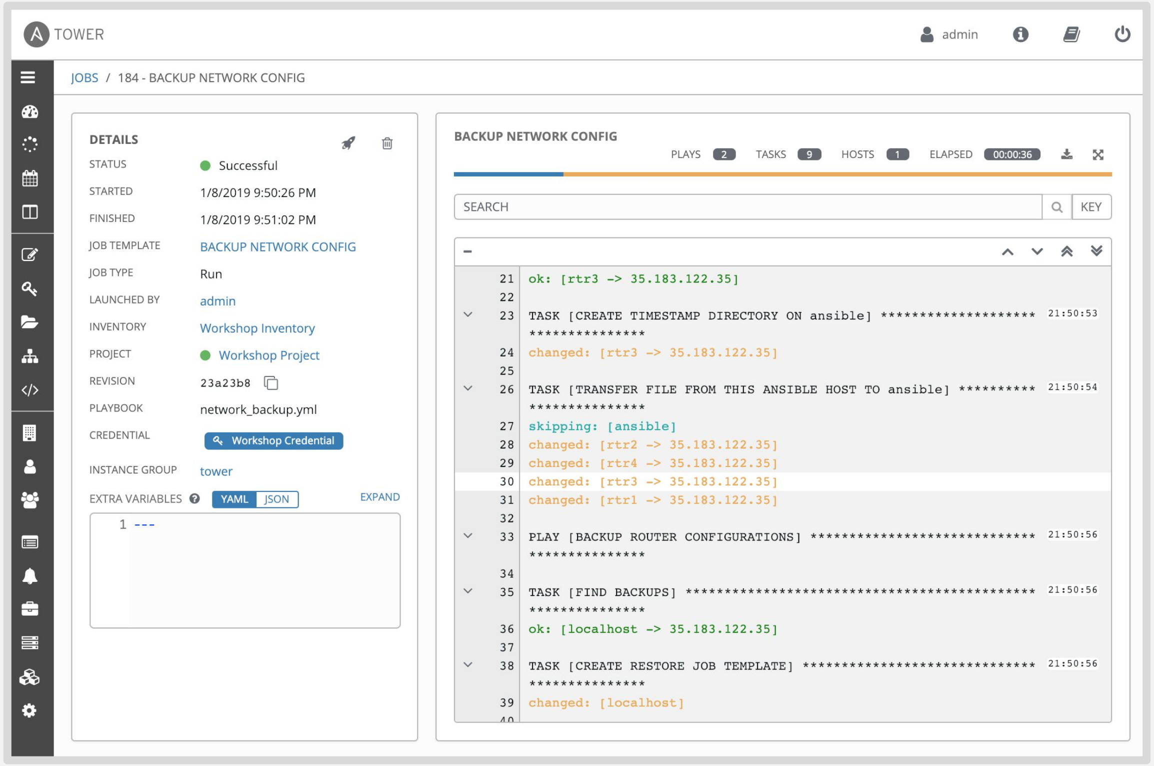
RECENTLY USED TEMPLATES — VIEW ALL

NAME	ACTIVITY	ACTIONS
Deploy application	🟩🟩🟩🟩🟩🟩🟥🟥🟥🟥	
Deploy database	🟥🟥🟥🟥🟥🟥⬜⬜⬜⬜	
Deploy webserver	🟩🟩🟩🟥🟥🟥🟥⬜⬜⬜	
Deploy ticketing application	🟩🟥🟥🟥🟥🟥⬜⬜⬜⬜	
Rollback workflow	🟩⬜⬜⬜⬜⬜⬜⬜⬜⬜	

RECENT JOB RUNS — VIEW ALL

NAME	TIME
● Deploy application	1/8/2019 9:42:22 PM
! Deploy database	1/8/2019 9:42:20 PM
● Deploy webserver	1/8/2019 9:41:06 PM
● Deploy webserver	1/8/2019 9:40:45 PM
● Deploy application	1/8/2019 9:40:20 PM

ANSIBLE TOWER FEATURES: JOB STATUS UPDATE



DETAILS

STATUS	● Successful
STARTED	1/8/2019 9:50:26 PM
FINISHED	1/8/2019 9:51:02 PM
JOB TEMPLATE	BACKUP NETWORK CONFIG
JOB TYPE	Run
LAUNCHED BY	admin
INVENTORY	Workshop Inventory
PROJECT	● Workshop Project
REVISION	23a23b8
PLAYBOOK	network_backup.yml
CREDENTIAL	Workshop Credential
INSTANCE GROUP	tower
EXTRA VARIABLES	YAML JSON

```
1 ---
```

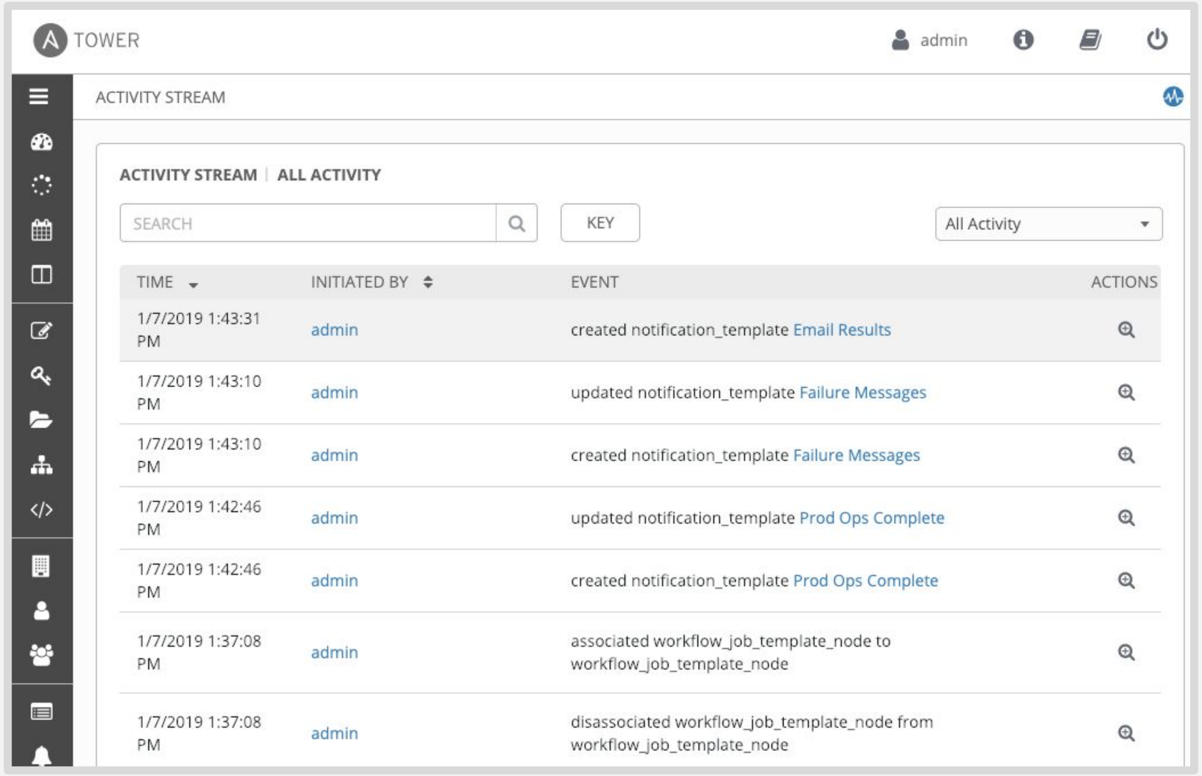
BACKUP NETWORK CONFIG

PLAYS 2 TASKS 9 HOSTS 1 ELAPSED 00:00:36

```
ok: [rtr3 -> 35.183.122.35]

TASK [CREATE TIMESTAMP DIRECTORY ON ansible] ******************** 21:50:53
**************
changed: [rtr3 -> 35.183.122.35]

TASK [TRANSFER FILE FROM THIS ANSIBLE HOST TO ansible] ********** 21:50:54
**************
skipping: [ansible]
changed: [rtr2 -> 35.183.122.35]
changed: [rtr4 -> 35.183.122.35]
changed: [rtr3 -> 35.183.122.35]
changed: [rtr1 -> 35.183.122.35]

PLAY [BACKUP ROUTER CONFIGURATIONS] ***************************** 21:50:56
**************

TASK [FIND BACKUPS] ********************************************* 21:50:56
**************
ok: [localhost -> 35.183.122.35]

TASK [CREATE RESTORE JOB TEMPLATE] ****************************** 21:50:56
**************
changed: [localhost]
```

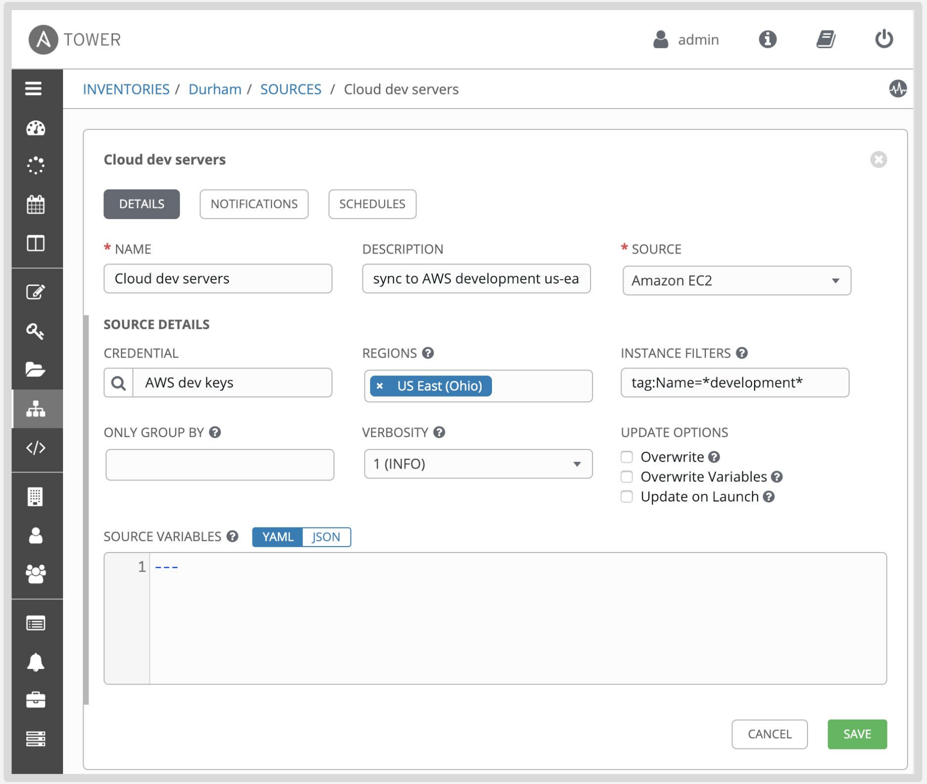
Red Hat

ANSIBLE TOWER FEATURES: ACTIVITY STREAM



TIME	INITIATED BY	EVENT	ACTIONS
1/7/2019 1:43:31 PM	admin	created notification_template Email Results	
1/7/2019 1:43:10 PM	admin	updated notification_template Failure Messages	
1/7/2019 1:43:10 PM	admin	created notification_template Failure Messages	
1/7/2019 1:42:46 PM	admin	updated notification_template Prod Ops Complete	
1/7/2019 1:42:46 PM	admin	created notification_template Prod Ops Complete	
1/7/2019 1:37:08 PM	admin	associated workflow_job_template_node to workflow_job_template_node	
1/7/2019 1:37:08 PM	admin	disassociated workflow_job_template_node from workflow_job_template_node	

ANSIBLE TOWER FEATURES: MANAGE AND TRACK YOUR INVENTORY



Cloud dev servers

* NAME: Cloud dev servers	DESCRIPTION: sync to AWS development us-ea	* SOURCE: Amazon EC2

SOURCE DETAILS

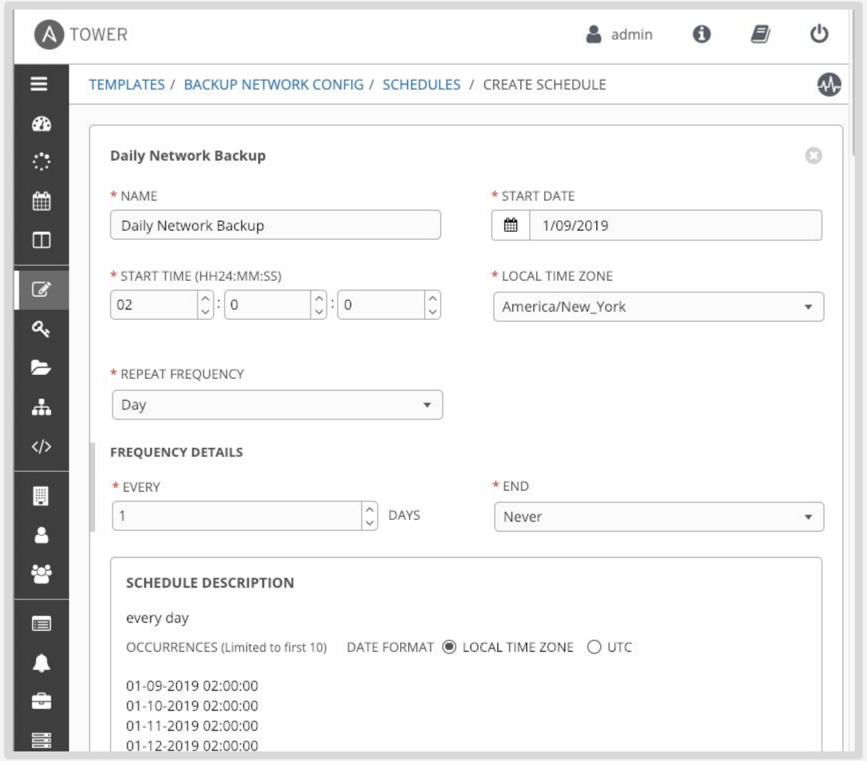
CREDENTIAL: AWS dev keys	REGIONS: US East (Ohio)	INSTANCE FILTERS: tag:Name=*development*
ONLY GROUP BY:	VERBOSITY: 1 (INFO)	UPDATE OPTIONS: ☐ Overwrite ☐ Overwrite Variables ☐ Update on Launch

SOURCE VARIABLES [✓ YAML] JSON

```
---
```

Red Hat

ANSIBLE TOWER FEATURES: SCHEDULE JOBS



TEMPLATES / BACKUP NETWORK CONFIG / SCHEDULES / CREATE SCHEDULE

Daily Network Backup

* NAME	Daily Network Backup
* START DATE	1/09/2019
* START TIME (HH24:MM:SS)	02 : 0 : 0
* LOCAL TIME ZONE	America/New_York
* REPEAT FREQUENCY	Day

FREQUENCY DETAILS

* EVERY	1 DAYS
* END	Never

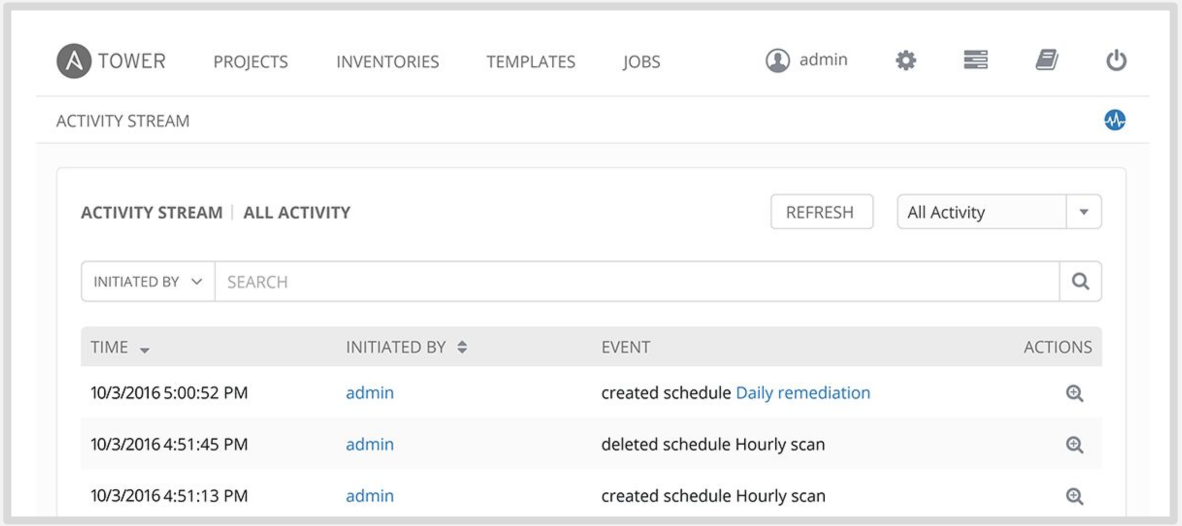
SCHEDULE DESCRIPTION

every day

OCCURRENCES (Limited to first 10) DATE FORMAT [✓ LOCAL TIME ZONE] UTC

01-09-2019 02:00:00
01-10-2019 02:00:00
01-11-2019 02:00:00
01-12-2019 02:00:00

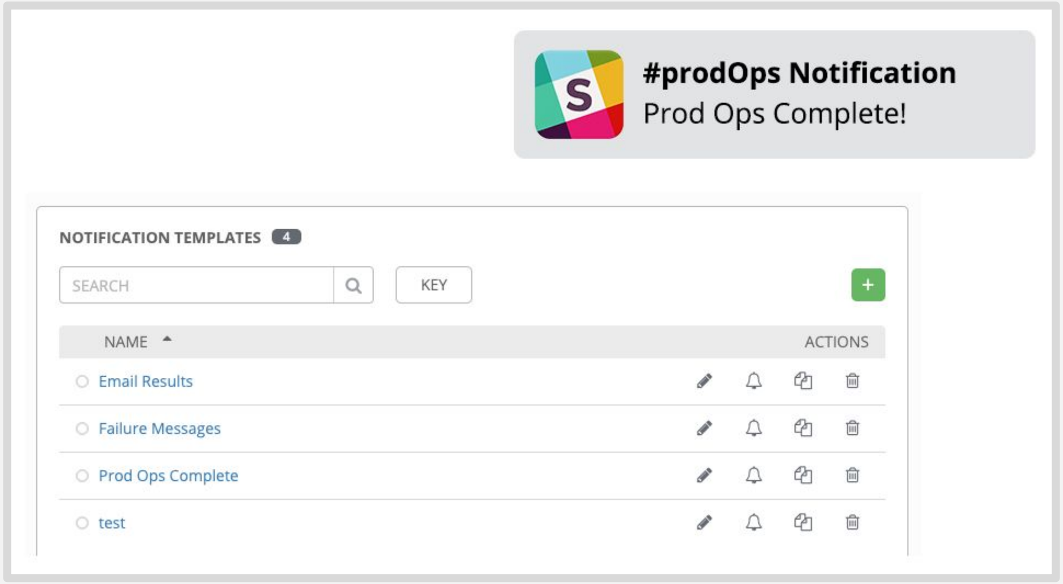
ANSIBLE TOWER FEATURES: EXTERNAL LOGGING



ACTIVITY STREAM | ALL ACTIVITY

TIME	INITIATED BY	EVENT	ACTIONS
10/3/2016 5:00:52 PM	admin	created schedule Daily remediation	
10/3/2016 4:51:45 PM	admin	deleted schedule Hourly scan	
10/3/2016 4:51:13 PM	admin	created schedule Hourly scan	

ANSIBLE TOWER FEATURES: INTEGRATED NOTIFICATIONS



#prodOps Notification
Prod Ops Complete!

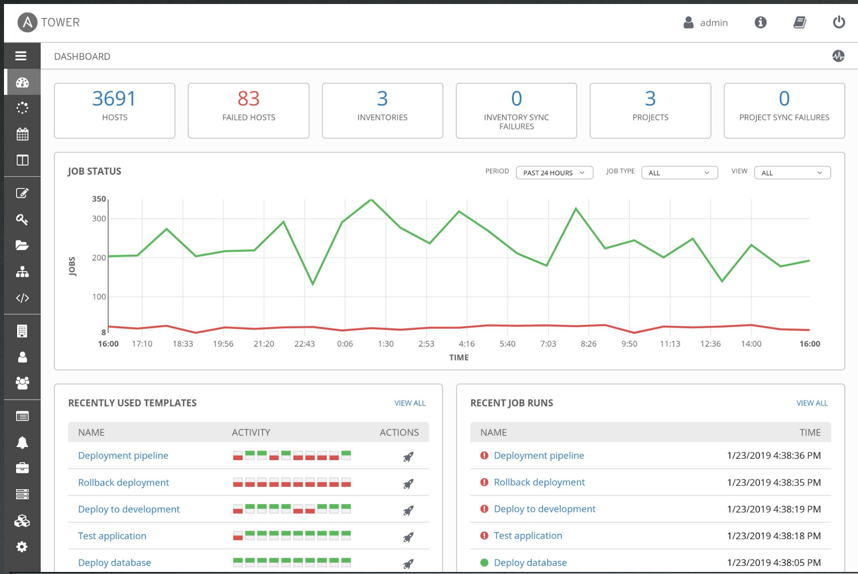
NOTIFICATION TEMPLATES 4

NAME	ACTIONS
Email Results	
Failure Messages	
Prod Ops Complete	
test	

Red Hat Ansible Tower

FEATURE OVERVIEW:

Delegation



TOWER admin

DASHBOARD

3691	83	3	0	3	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

JOB STATUS

PERIOD: PAST 24 HOURS JOB TYPE: ALL VIEW: ALL

RECENTLY USED TEMPLATES VIEW ALL

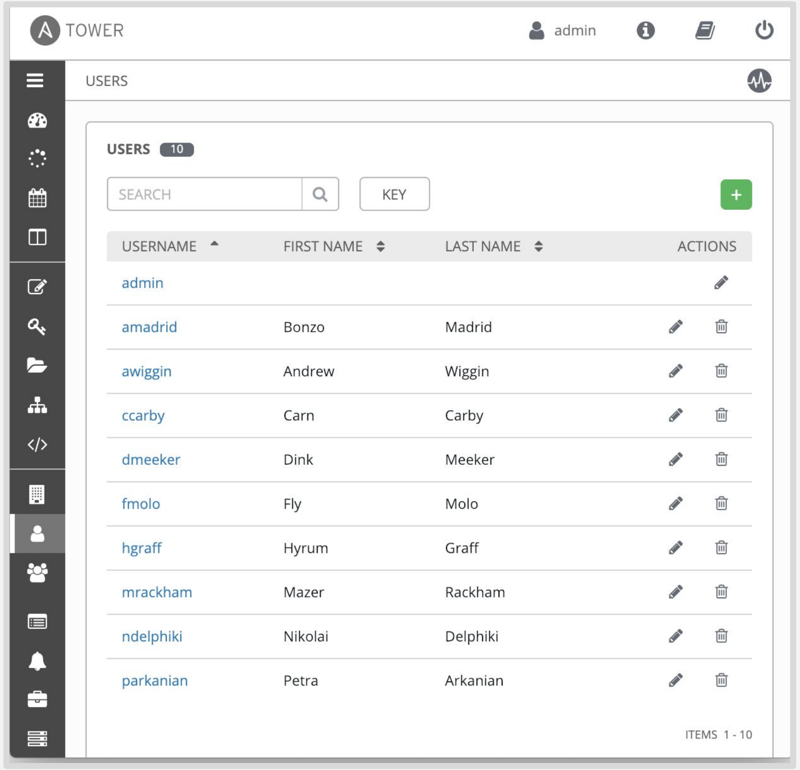
NAME	ACTIVITY	ACTIONS
Deployment pipeline	🟥🟩🟩🟥🟩🟥🟥🟥🟥🟩	
Rollback deployment	🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥	
Deploy to development	🟥🟩🟩🟩🟩🟥🟥🟩🟩🟩	
Test application	🟥🟩🟩🟩🟩🟩🟩🟩🟩🟩	
Deploy database	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩	

RECENT JOB RUNS VIEW ALL

NAME	TIME
Deployment pipeline	1/23/2019 4:38:36 PM
Rollback deployment	1/23/2019 4:38:35 PM
Deploy to development	1/23/2019 4:38:19 PM
Test application	1/23/2019 4:38:18 PM
● Deploy database	1/23/2019 4:38:05 PM

ANSIBLE TOWER FEATURES: ROLE BASED ACCESS CONTROL

USERS

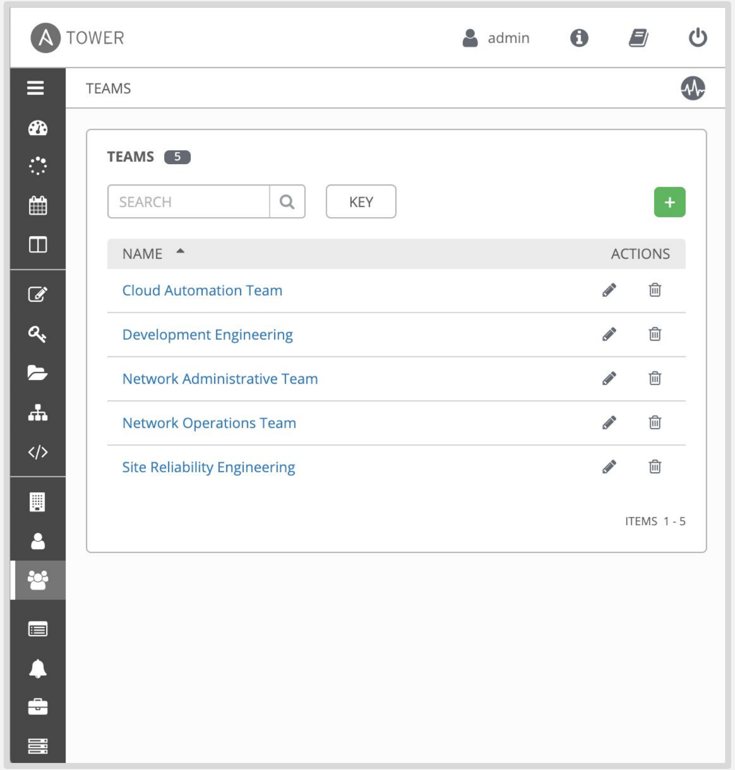


USERS 10

USERNAME	FIRST NAME	LAST NAME	ACTIONS
admin			
amadrid	Bonzo	Madrid	
awiggin	Andrew	Wiggin	
ccarby	Carn	Carby	
dmeeker	Dink	Meeker	
fmolo	Fly	Molo	
hgraff	Hyrum	Graff	
mrackham	Mazer	Rackham	
ndelphiki	Nikolai	Delphiki	
parkanian	Petra	Arkanian	

ITEMS 1 - 10

TEAMS

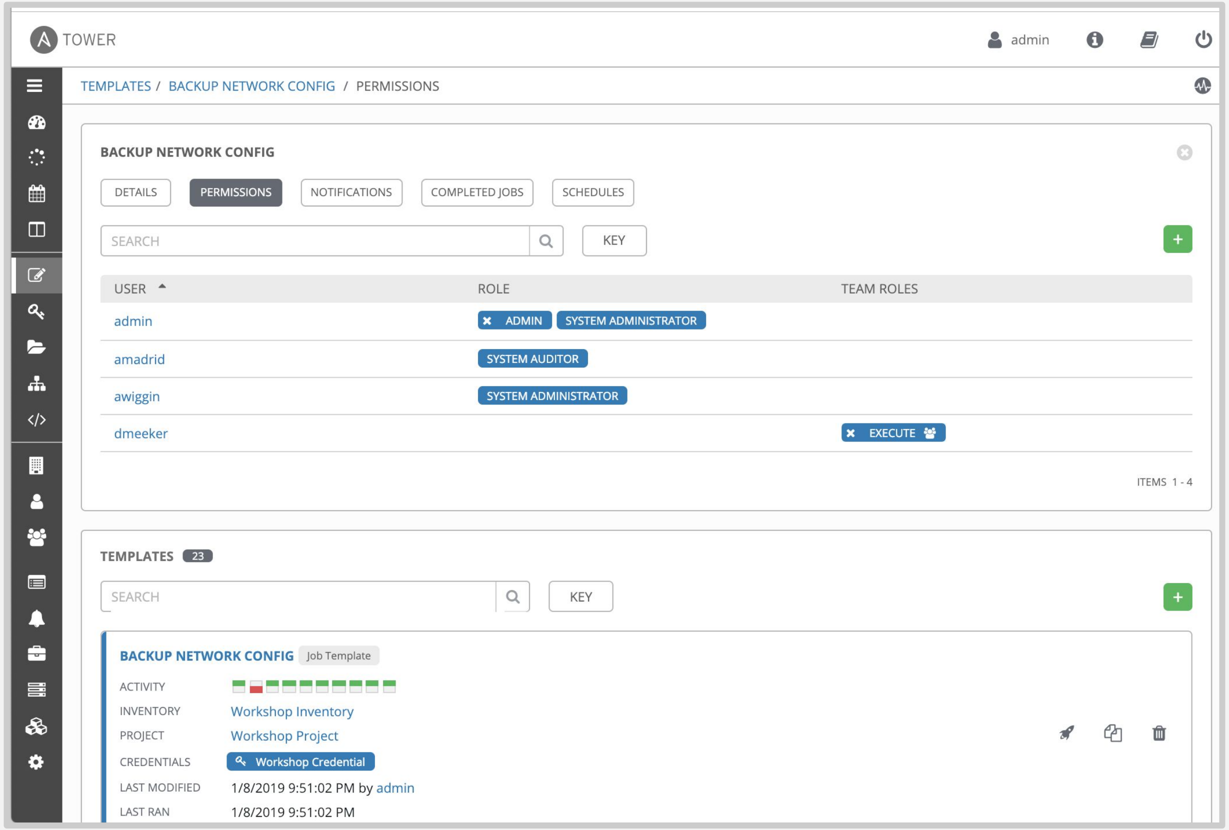


TEAMS 5

NAME	ACTIONS
Cloud Automation Team	
Development Engineering	
Network Administrative Team	
Network Operations Team	
Site Reliability Engineering	

ITEMS 1 - 5

ANSIBLE TOWER FEATURES: ROLE BASED ACCESS CONTROL



TOWER admin

TEMPLATES / BACKUP NETWORK CONFIG / PERMISSIONS

BACKUP NETWORK CONFIG

DETAILS | PERMISSIONS | NOTIFICATIONS | COMPLETED JOBS | SCHEDULES

SEARCH KEY

USER	ROLE	TEAM ROLES
admin	ADMIN, SYSTEM ADMINISTRATOR	
amadrid	SYSTEM AUDITOR	
awiggin	SYSTEM ADMINISTRATOR	
dmeeker		EXECUTE

ITEMS 1 - 4

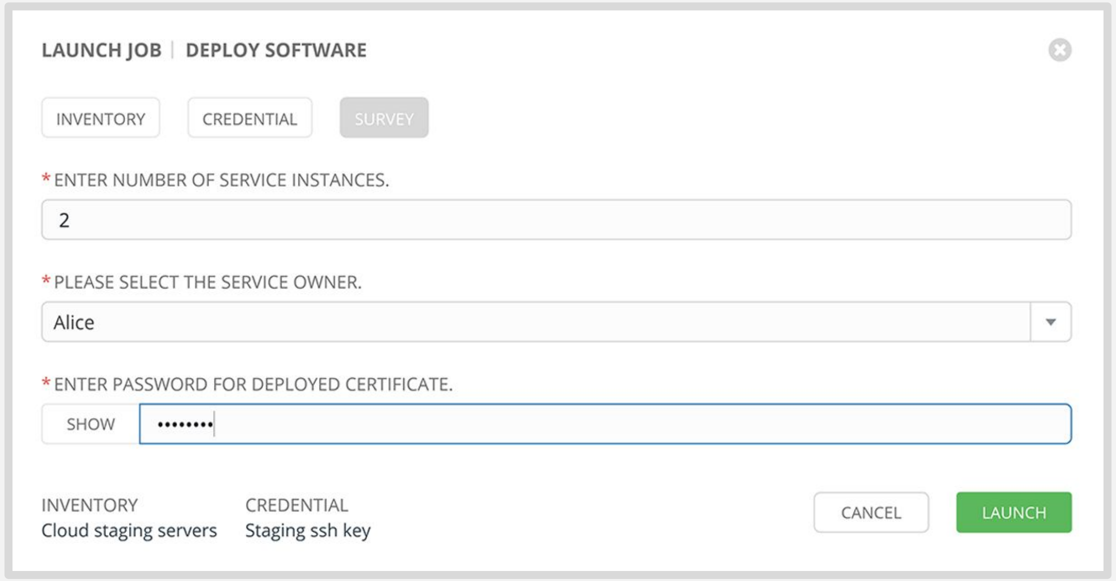
TEMPLATES 23

SEARCH KEY

BACKUP NETWORK CONFIG Job Template

ACTIVITY	🟩🟥🟩🟩🟩🟩🟩🟩🟩🟩
INVENTORY	Workshop Inventory
PROJECT	Workshop Project
CREDENTIALS	Workshop Credential
LAST MODIFIED	1/8/2019 9:51:02 PM by admin
LAST RAN	1/8/2019 9:51:02 PM

ANSIBLE TOWER FEATURES: SELF-SERVICE I.T.


LAUNCH JOB | DEPLOY SOFTWARE

INVENTORY CREDENTIAL SURVEY

* ENTER NUMBER OF SERVICE INSTANCES.

2

* PLEASE SELECT THE SERVICE OWNER.

Alice

* ENTER PASSWORD FOR DEPLOYED CERTIFICATE.

SHOW ••••••••

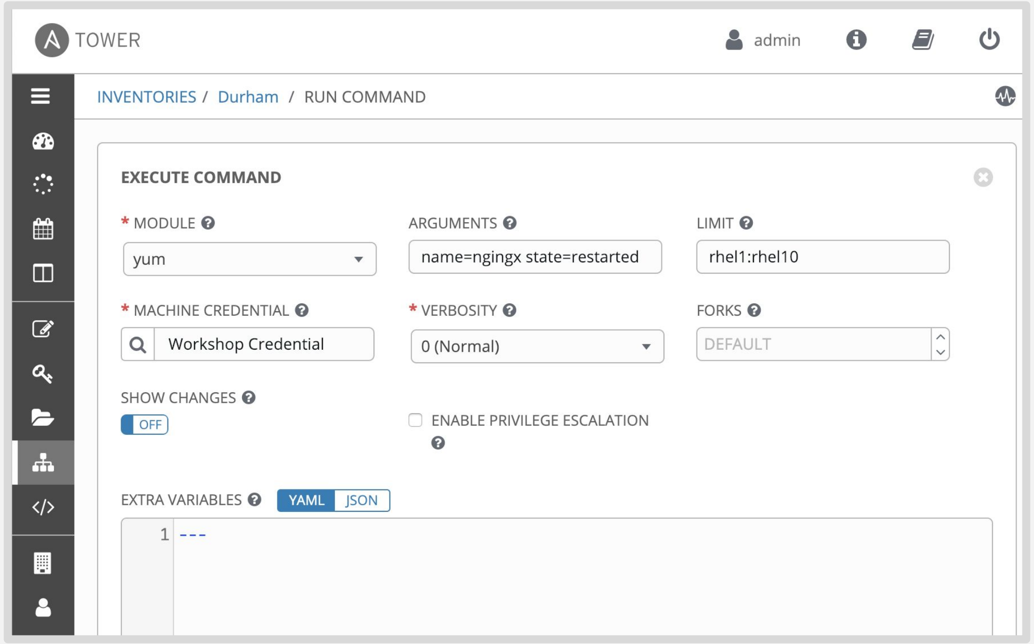
INVENTORY
Cloud staging servers

CREDENTIAL
Staging ssh key

CANCEL LAUNCH

Red Hat

ANSIBLE TOWER FEATURES: REMOTE COMMAND EXECUTION



* MODULE: yum

ARGUMENTS: name=nginx state=restarted

LIMIT: rhel1:rhel10

* MACHINE CREDENTIAL: Workshop Credential

* VERBOSITY: 0 (Normal)

FORKS: DEFAULT

SHOW CHANGES: OFF

ENABLE PRIVILEGE ESCALATION

EXTRA VARIABLES: YAML JSON

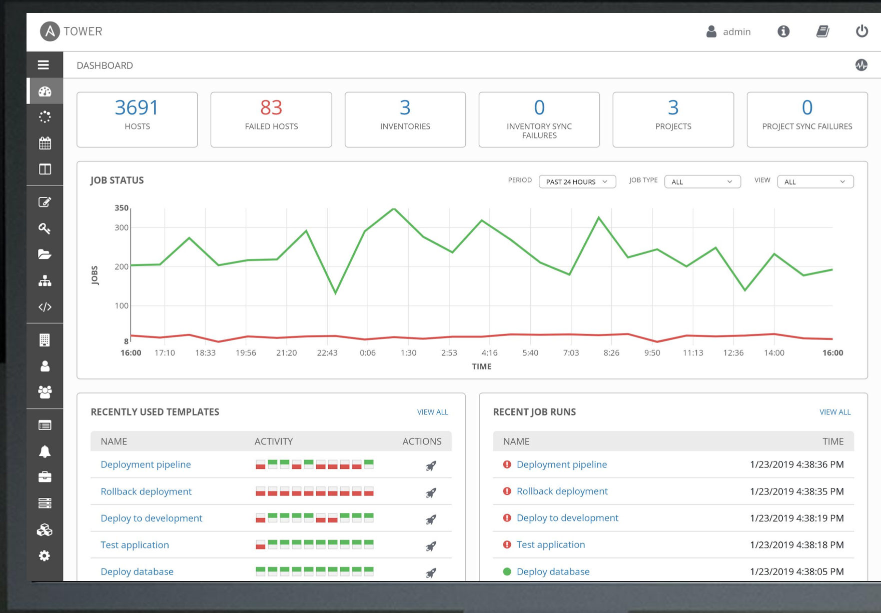
```
---
```

Red Hat

Red Hat Ansible Tower

FEATURE OVERVIEW:

Scale



3691	83	3	0	3	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

JOB STATUS

PERIOD: PAST 24 HOURS · JOB TYPE: ALL · VIEW: ALL

RECENTLY USED TEMPLATES — VIEW ALL

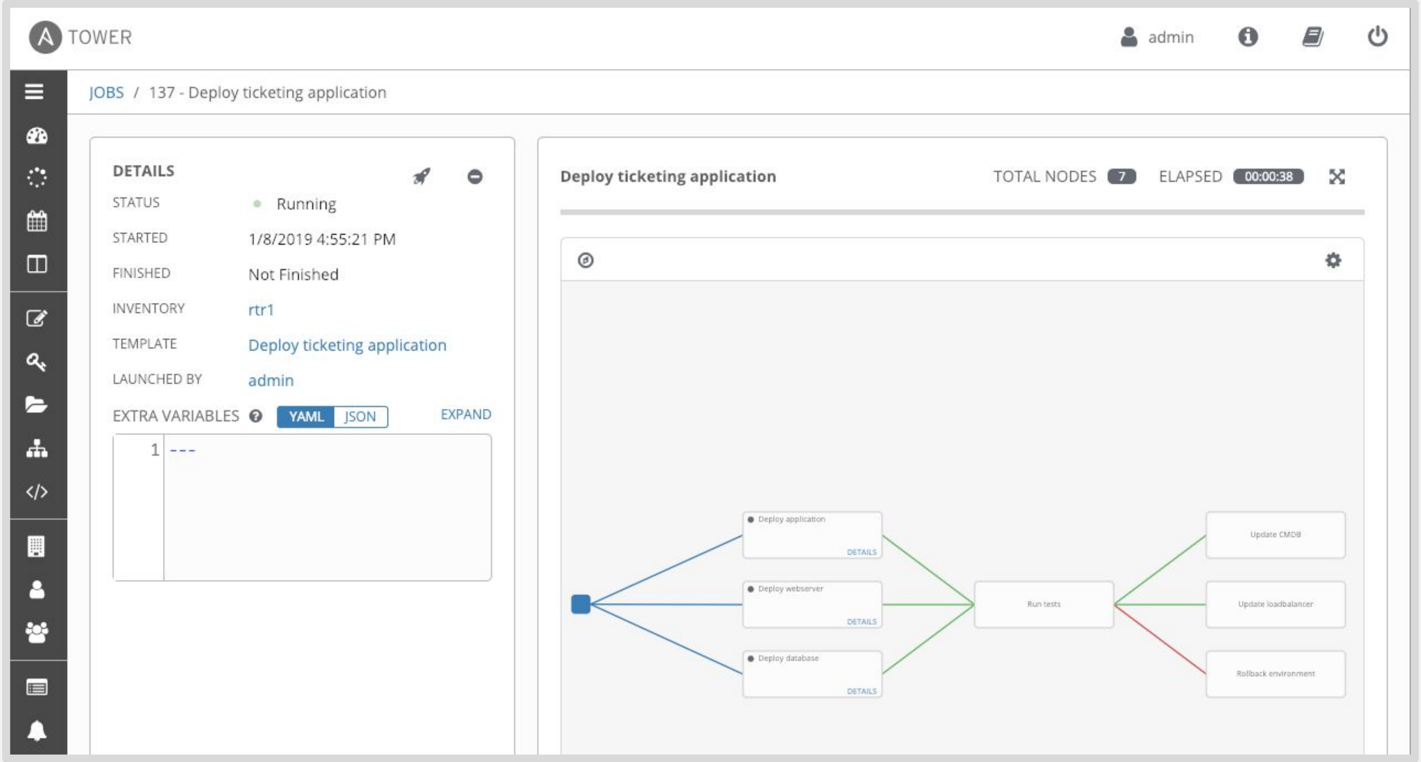
NAME	ACTIVITY	ACTIONS
Deployment pipeline	🟥🟩🟩🟥🟩🟥🟥🟥🟥🟩	
Rollback deployment	🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥	
Deploy to development	🟥🟩🟩🟩🟩🟥🟥🟩🟩🟩	
Test application	🟥🟩🟩🟩🟩🟩🟩🟩🟩🟩	
Deploy database	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩	

RECENT JOB RUNS — VIEW ALL

NAME	TIME
Deployment pipeline	1/23/2019 4:38:36 PM
Rollback deployment	1/23/2019 4:38:35 PM
Deploy to development	1/23/2019 4:38:19 PM
Test application	1/23/2019 4:38:18 PM
Deploy database	1/23/2019 4:38:05 PM

Red Hat

ANSIBLE TOWER FEATURES: CREATE AUTOMATION WORKFLOWS



DETAILS

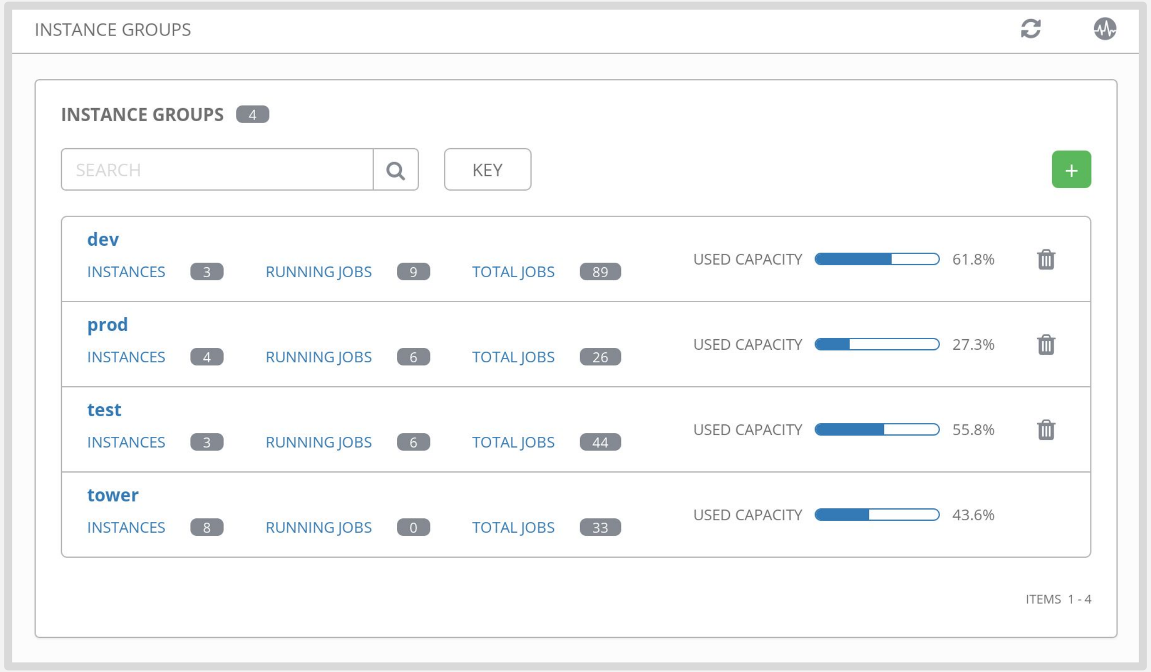
STATUS	● Running
STARTED	1/8/2019 4:55:21 PM
FINISHED	Not Finished
INVENTORY	rtr1
TEMPLATE	Deploy ticketing application
LAUNCHED BY	admin

EXTRA VARIABLES [✓ YAML] JSON EXPAND

```
---
```

Deploy ticketing application TOTAL NODES 7 ELAPSED 00:00:38

ANSIBLE TOWER FEATURES: SCALE OUT CLUSTERING


INSTANCE GROUPS

INSTANCE GROUPS 4

SEARCH KEY +

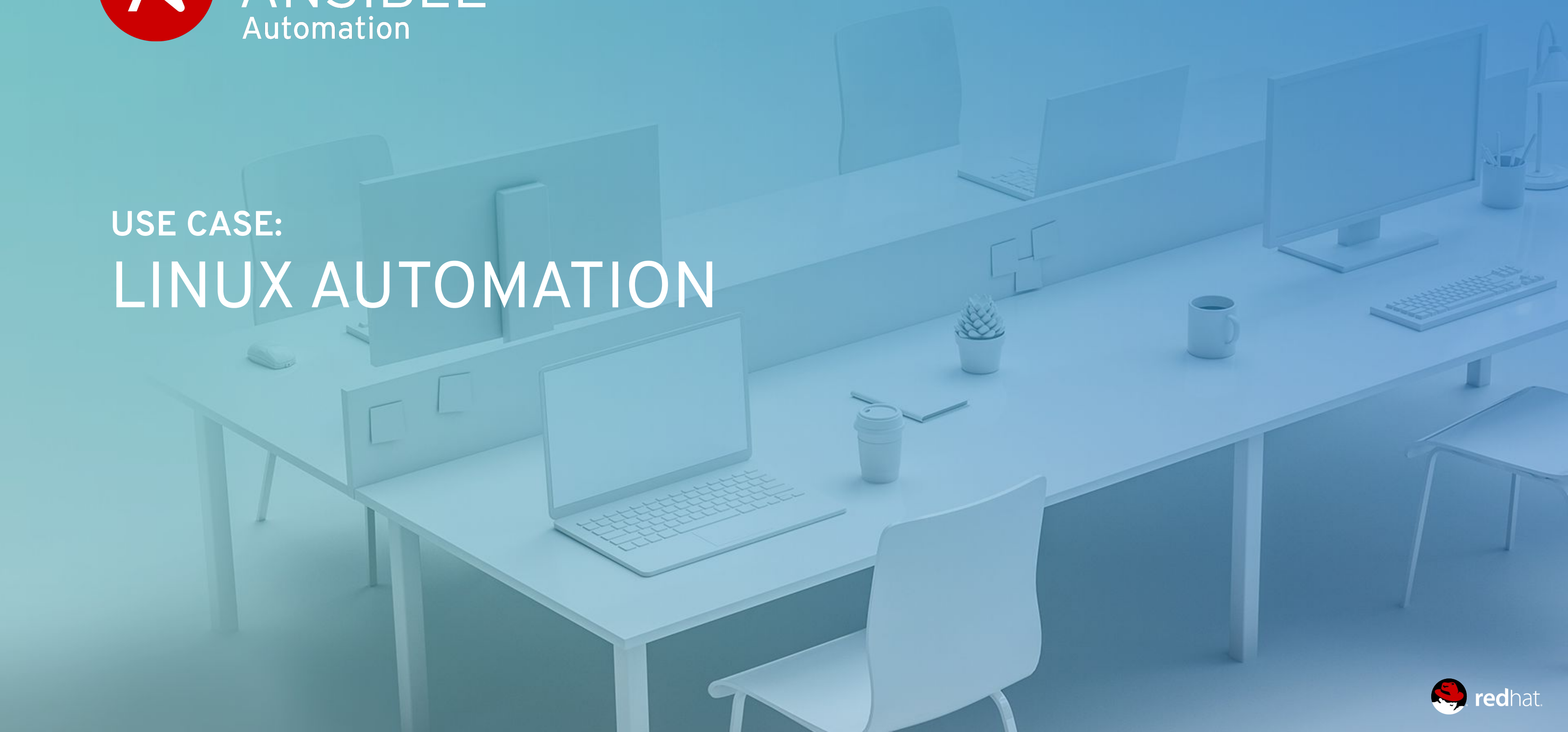
dev	INSTANCES 3	RUNNING JOBS 9	TOTAL JOBS 89	USED CAPACITY 61.8%
prod	INSTANCES 4	RUNNING JOBS 6	TOTAL JOBS 26	USED CAPACITY 27.3%
test	INSTANCES 3	RUNNING JOBS 6	TOTAL JOBS 44	USED CAPACITY 55.8%
tower	INSTANCES 8	RUNNING JOBS 0	TOTAL JOBS 33	USED CAPACITY 43.6%

ITEMS 1 - 4

RED HAT® ANSIBLE® Automation

USE CASE:

LINUX AUTOMATION



redhat.

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**

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

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

Red Hat

AUTOMATION FOR EVERYONE: **SYSTEM ADMINISTRATORS**

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

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

Red Hat

AUTOMATION FOR EVERYONE: **SYSTEM ADMINISTRATORS**

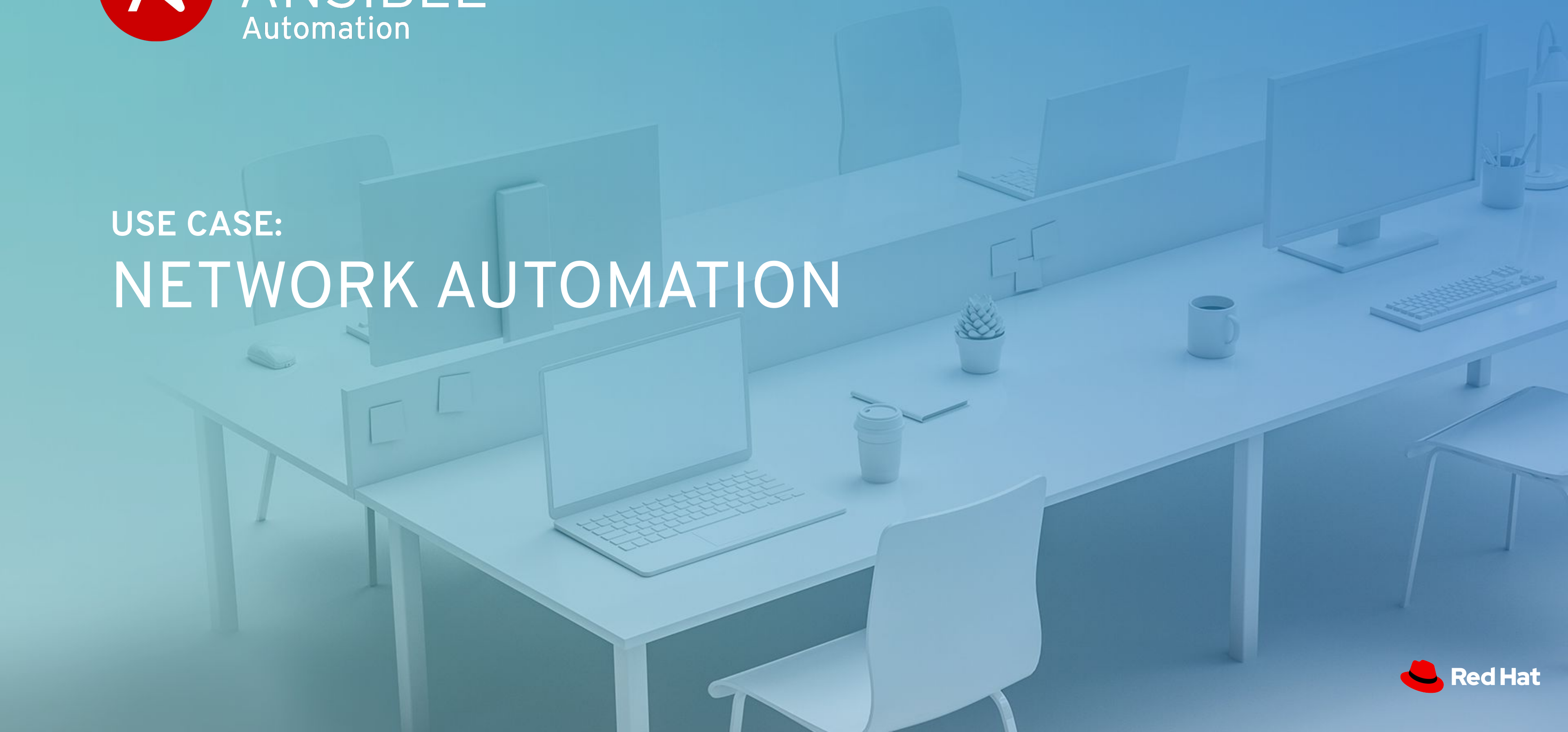
```yaml
---
- name: check services on rhel hosts
  hosts: rhel
  become: yes

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

Red Hat

RED HAT ANSIBLE Automation

USE CASE:

NETWORK AUTOMATION



Red Hat

ANSIBLE NETWORK AUTOMATION

50
Network Platforms

700+
Network Modules

12*
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?

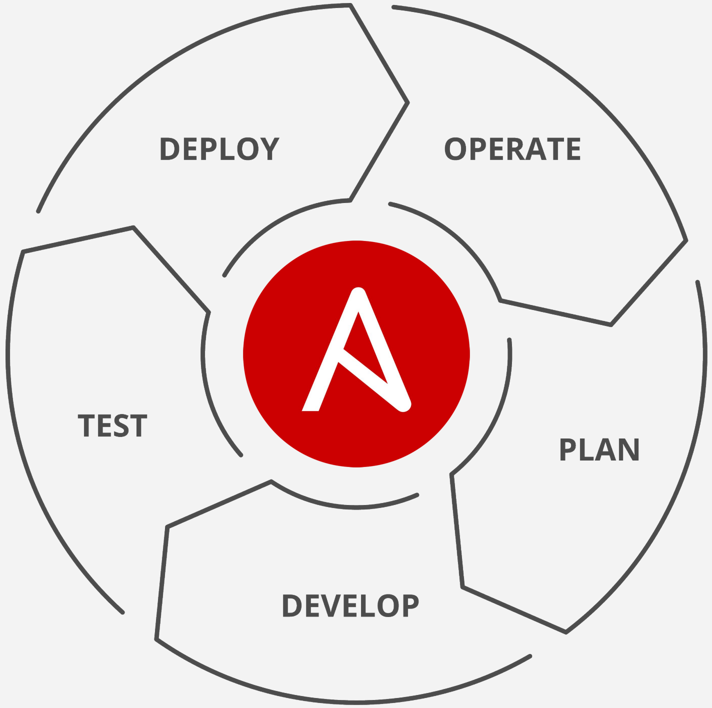
PLAN AND PROTOTYPE VIRTUALLY
Use tasks as reusable building blocks

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

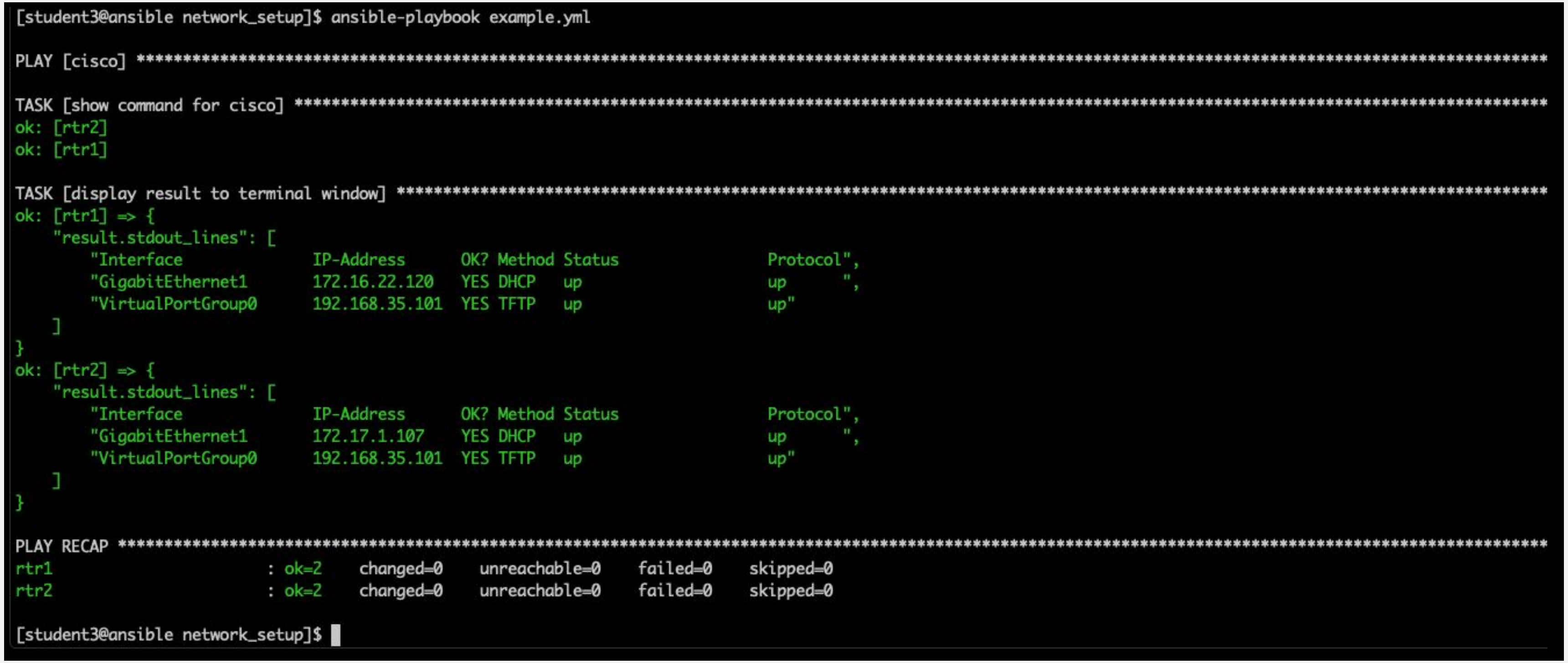
ENSURE AN ON-GOING STEADY-STATE



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**

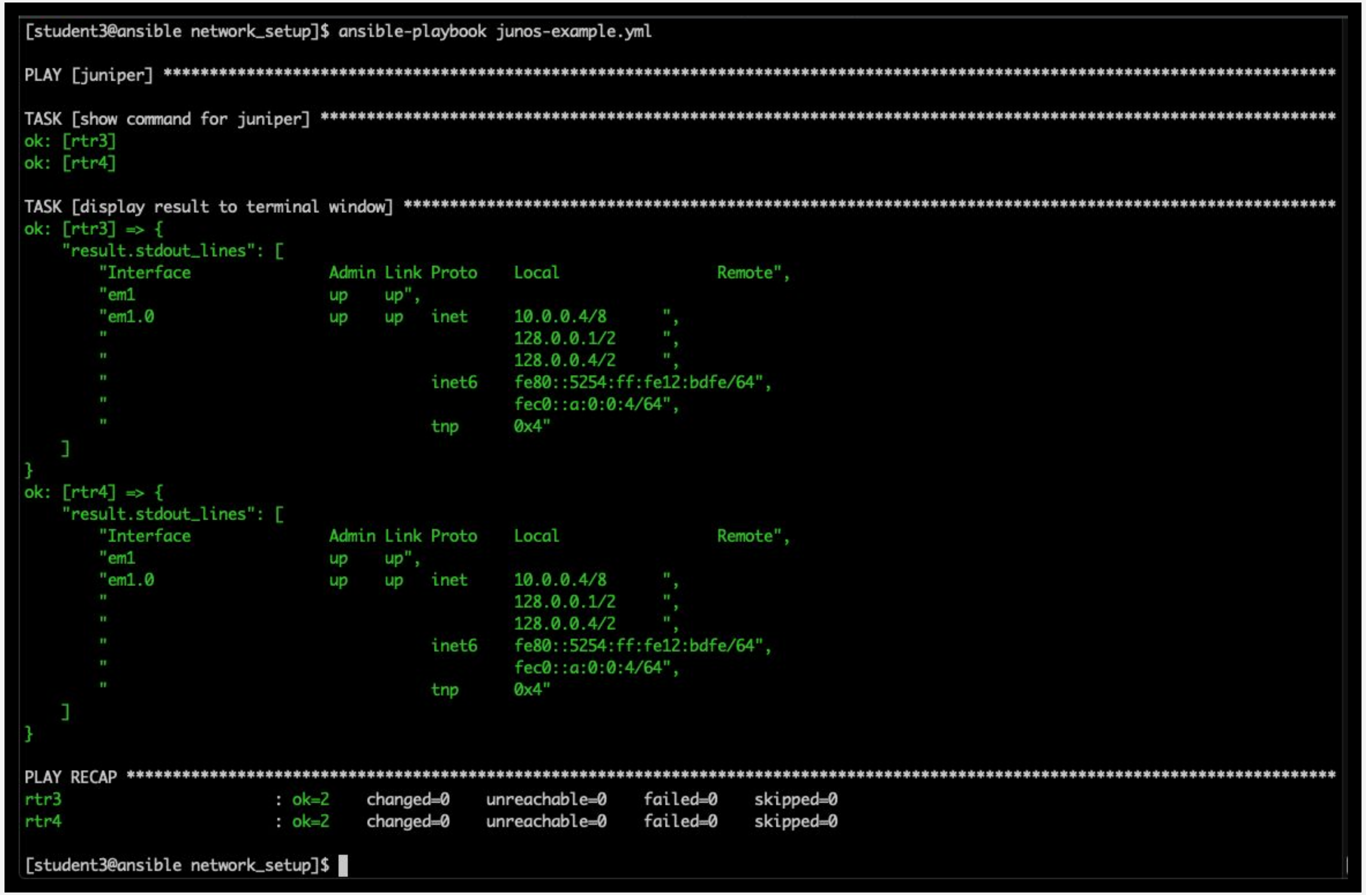


```
[student3@ansible network_setup]$ ansible-playbook example.yml

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]$
```

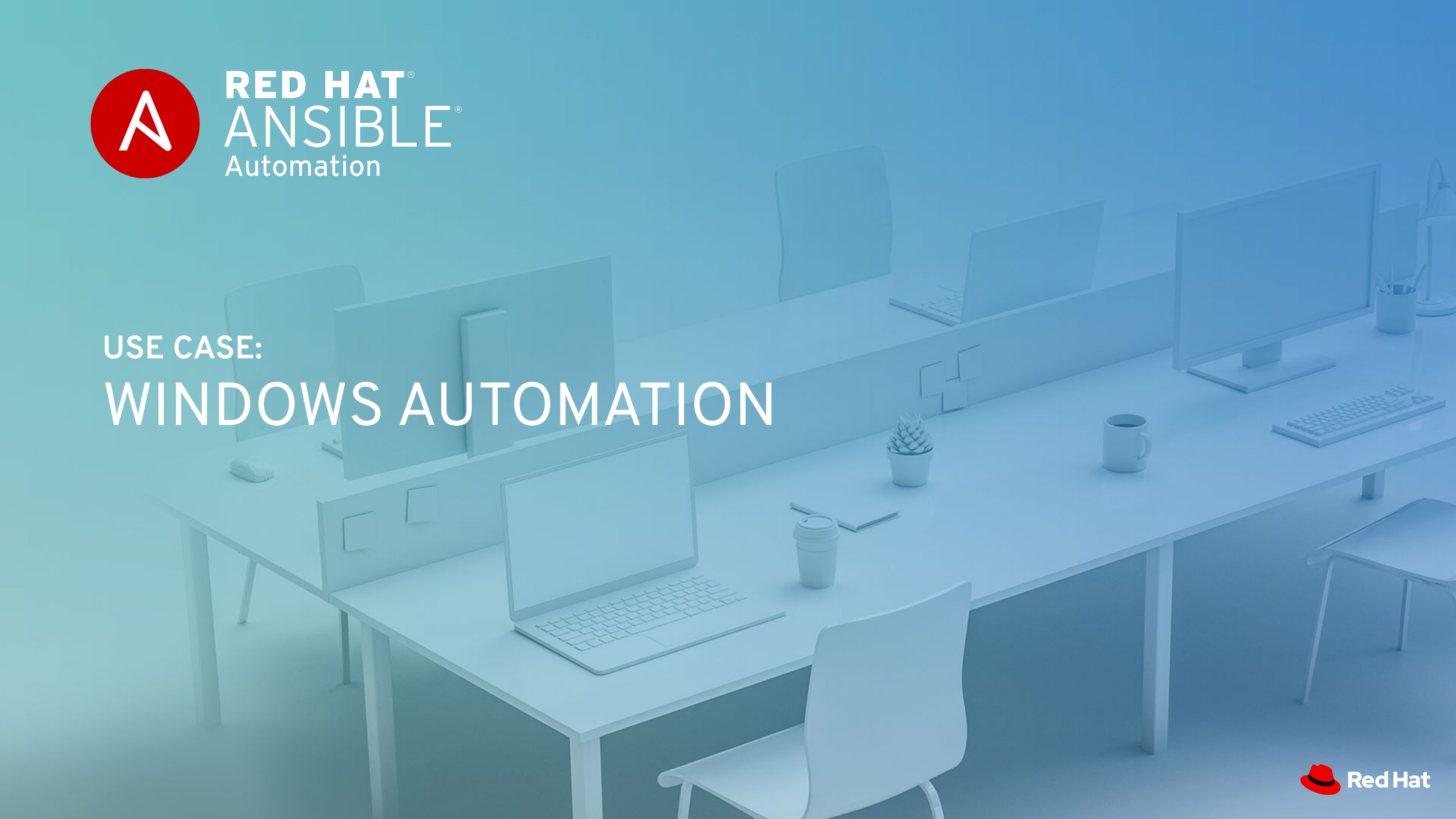
AUTOMATION FOR EVERYONE: **NETWORK ENGINEERS**

```yaml
---
- 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
```

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® ANSIBLE® Automation

USE CASE:

WINDOWS AUTOMATION

Red Hat

WINDOWS AUTOMATION

90+
Windows Modules

1,300+
Powershell DSC (Desired State Config) 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
```

Red Hat

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

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

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

Red Hat

AUTOMATION FOR EVERYONE: **WINDOWS ADMINS**

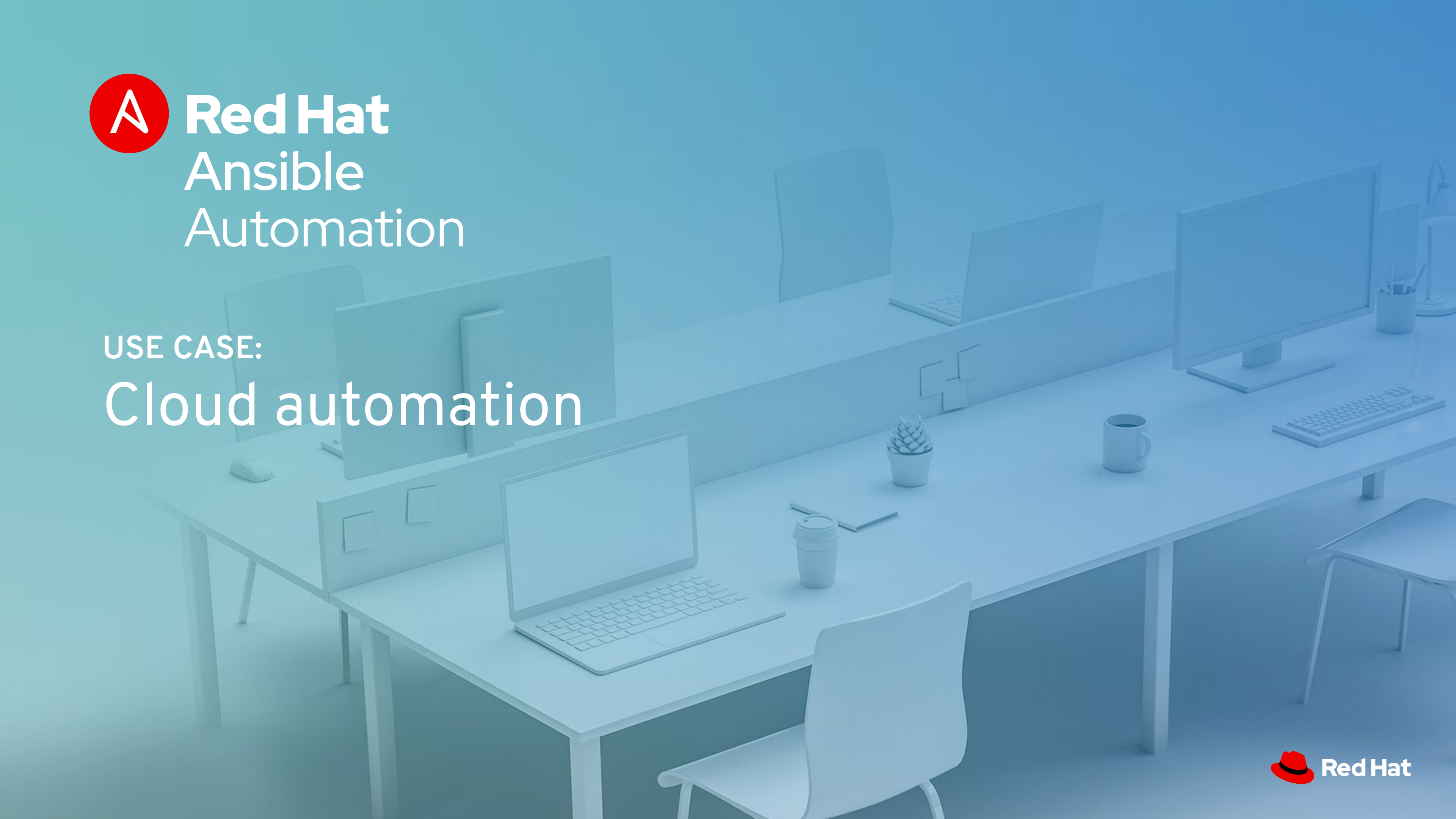
```
---
- name: update and reboot
  hosts: windows_servers
  tasks:
  - name: ensure common OS updates are current
    win_updates:
    register: update_result

  - 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**

```yaml
---
- 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

USE CASE:

Cloud automation

Red Hat

CLOUD AUTOMATION

800+

Cloud Modules

30+

Cloud Platforms

ansible.com/cloud

Red Hat

PLAYBOOK EXAMPLE: **AWS**

```yaml
---
- 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**

```yaml
---
- 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"
```

Red Hat

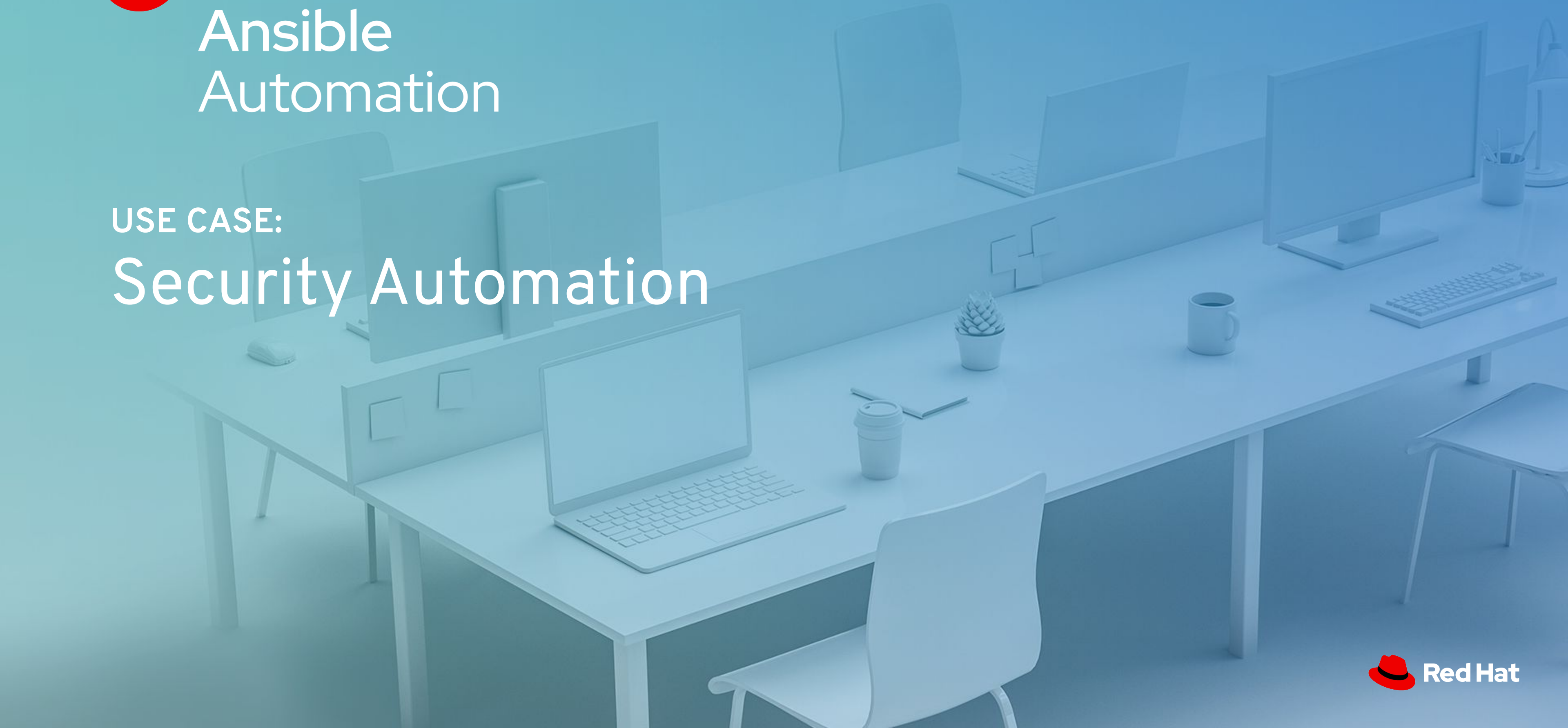
PLAYBOOK EXAMPLE: **RED HAT OPENSTACK**

```
---
- 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

USE CASE:

Security Automation



Red Hat

What is it?

Ansible Security Automation is a supported set of Ansible modules, roles and playbooks designed to unify the security response to cyberattacks in a new way - by orchestrating the activity of multiple classes of security solutions that wouldn’t normally integrate with each other.

What does it do?

Through Ansible Security Automation, IT organizations can address multiple popular use cases:

- For **detection and triage of suspicious activities**, for example, Ansible can automatically enable logging or increase the log verbosity across enterprise firewalls and IDS to enrich the alerts received by a SIEM for an easier triage.
- For **threat hunting,** for example, Ansible can automatically create new IDS rules to investigate the origin of a firewall rule violation, and whitelist those IP addresses recognized as non threats.
- For **incident response,** for example, Ansible can automatically validate a threat by verifying an IDS rule, trigger a remediation from the SIEM solution, and create new enterprise firewall rules to blacklist the source of an attack.

At launch, Red Hat's Ansible security automation platform provides support for:

- **Check Point** – Next Generation Firewall (NGFW);
- **Splunk** – Splunk Security Enterprise (SE);
- **Snort**

Who is it for?

Ansible Security Automation extends the Ansible agentless, modular and easy to use enterprise automation platform to support the following industry constituencies:

- **End-user organizations' security teams** in charge of Security Operations Centres (SOCs)
- **Managed security service providers (MSSPs)** responsible for the governance of thousands of enterprise security solutions across their whole customer base
- **Security ISVs** offering security orchestration and automation (SOAR) solutions currently using custom-made automation frameworks

AUTOMATION FOR EVERYONE: **SECURITY OPERATIONS**

```
---
- name: checkpoint playbook
  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**

```yaml
---
- name: checkpoint playbook
  hosts: checkpoint
  connection: httpapi

  tasks:
    - name: delete access rule
      checkpoint_access_rule:
        layer: Network
        name: "Drop attacker"
        state: absent
```

Red Hat

NEXT STEPS

GET STARTED

ansible.com/get-started

ansible.com/tower-trial

JOIN THE COMMUNITY

ansible.com/community

WORKSHOPS & TRAINING

ansible.com/workshops

Red Hat Training

SHARE YOUR STORY

Follow us @Ansible

Friend us on Facebook