redhat.

Ansible and Ansible Tower by Red Hat

Automation technology you can use everywhere

Jacek Skórzyński
Senior Solution Architect
Red Hat CEE

jacek@redhat.com

RED HAT MANAGEMENT

SATELLITE

BUILD A TRUSTED & SECURE RED HAT ENVIRONMENT

Manage the Red Hat Lifecycle
Provision & Configure at Scale
Standardize Your Environment

CLOUDFORMS

DELIVER SERVICES ACROSS YOUR HYBRID CLOUD

Hybrid Cloud Management
Self-Service Provisioning
Policy-driven Compliance

ANSIBLE

AUTOMATE YOUR IT PROCESSES & DEPLOYMENTS

Simple & powerful language
No agents to install
Scale with Ansible Tower

INSIGHTS

PREVENT CRITICAL ISSUES BEFORE THEY OCCUR

Continuous Insights
Verified Knowledge
Proactive Resolution

Ansible by Red Hat

SIMPLE

- Human readable automation
- No special coding skills needed
- Tasks executed in order
- Get productive quickly

POWERFUL

- App deployment
- Configuration management
- Workflow orchestration
- Orchestrate the app lifecycle

AGENTLESS

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

The Ansible Way

CROSS PLATFORM

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

HUMAN READABLE

Perfectly describe and document every aspect of your application environment.

PERFECT DESCRIPTION OF

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

VERSION CONTROLLED

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

DYNAMIC INVENTORIES

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

ORCHESTRATION PLAYS WELL WITH

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

What Can I Do With ANSIBLE?

Do this...

- Orchestration
- Configuration Management
- Application Deployment
- Provisioning
- Continuous Delivery
- Security and Compliance

On these...

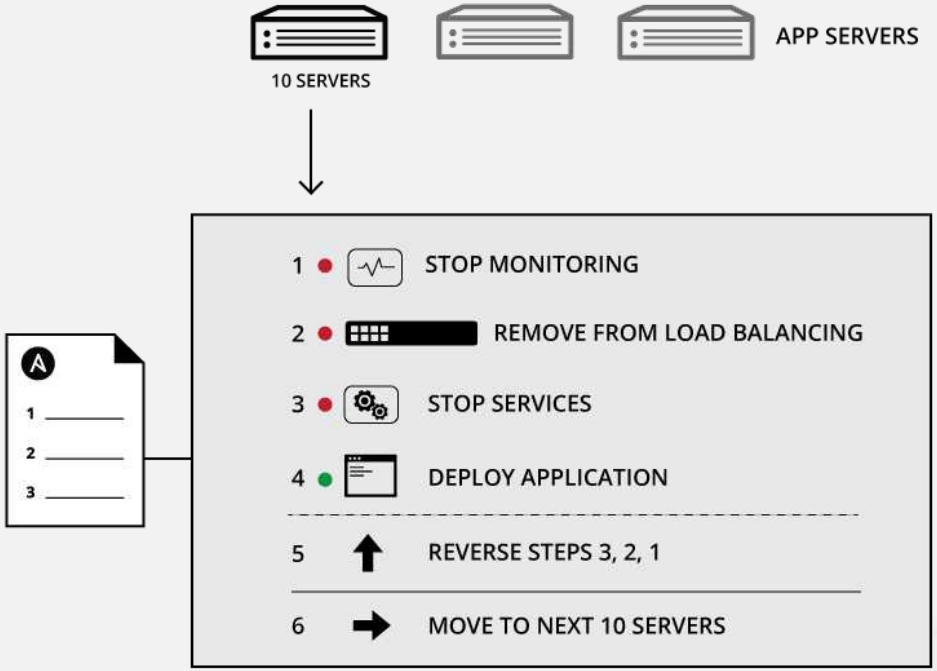
- Firewalls
- Load Balancers
- Applications
- Containers
- Clouds
- Servers
- Infrastructure
- Storage
- Network Devices
- **And more...**

Why Is Automation Important?

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

Ansible can do it all:

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

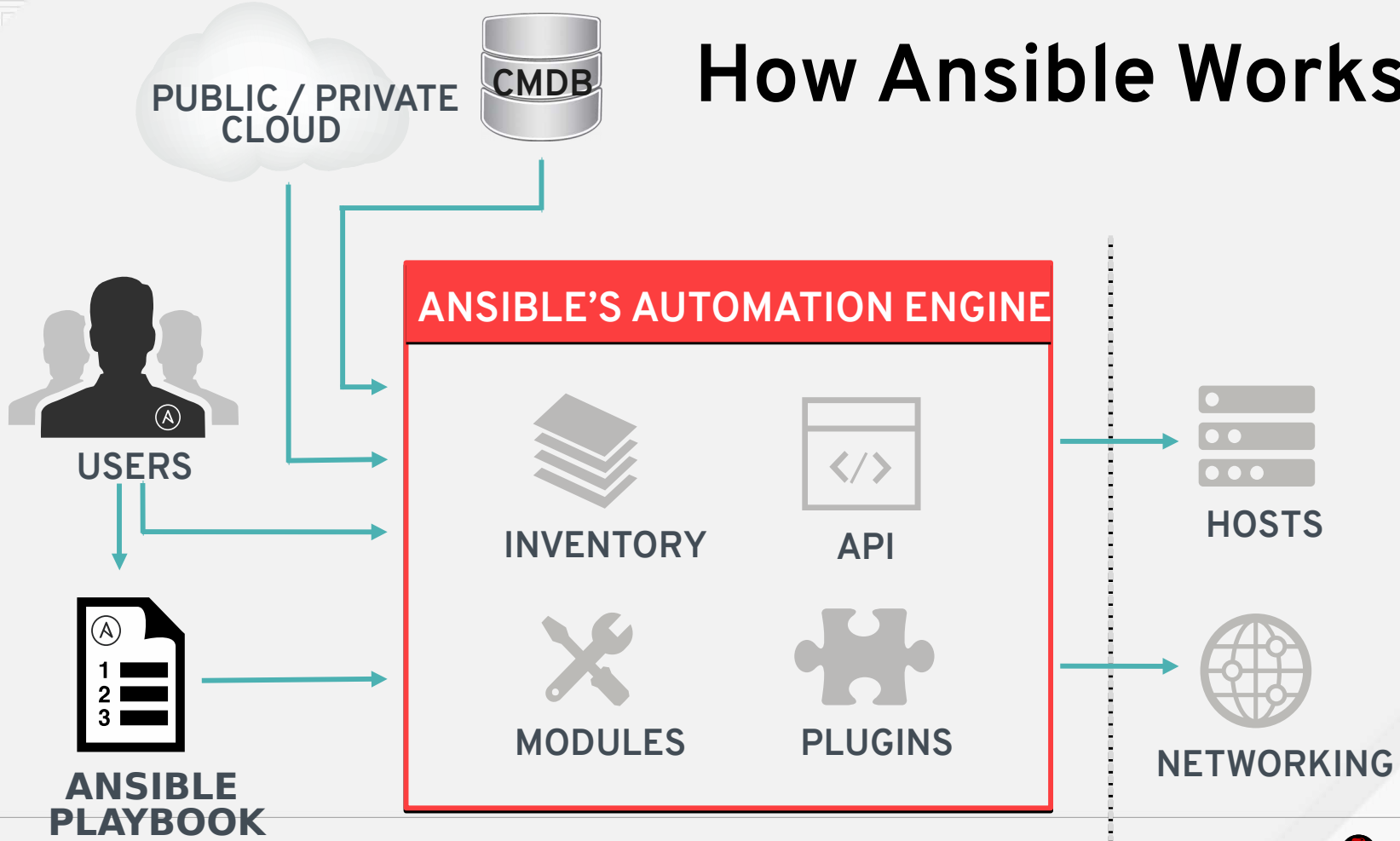


ANSIBLE

Quick Intro

redhat.

How Ansible Works



redhat.

Ansible Inventory File

Default: /etc/ansible/hosts

```
192.168.122.100
172.16.0.[100:120]

[webservers]
web[001:100]

[webservers:vars]
remote_user=webadmin

[dbservers]
db1
db2
db3

[prod:children]
webservers
dbservers
```

Running Ansible from Command Line

Ad-hoc commands:

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

Available modules:

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

Running playbooks:

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

Ansible Facts

```
[wfurmank@wfurmank ansible]$ ansible localhost -m setup
localhost | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "192.168.122.1"
        ],
        "ansible_all_ipv6_addresses": [],
        "ansible_architecture": "x86_64",
```

```
"ansible_virbr0": {
    "active": false,
    "device": "virbr0",
    "id": "8000.525400ec630c",
    "interfaces": [
        "virbr0-nic"
    ],
    "ipv4": {
        "address": "192.168.122.1",
        "broadcast": "192.168.122.255",
        "netmask": "255.255.255.0",
        "network": "192.168.122.0"
```

```
"ansible_date_time": {
    "date": "2016-10-31",
    "day": "31",
    "epoch": "1477926817",
    "hour": "16",
    "iso8601": "2016-10-31T15:13:37Z",
    "iso8601_basic": "20161031T161337117677"
```

```
"ansible_distribution": "RedHat",
"ansible_distribution_major_version": "7",
"ansible_distribution_release": "Maipo",
"ansible_distribution_version": "7.2",
```

```
    "time": "16:13:37",
    "tz": "CET",
    "tz_offset": "+0100",
    "weekday": "Monday",
    "weekday_number": "1",
    "weeknumber": "44",
    "year": "2016"
```

Usage examples:

{{ ansible_virbr0.ipv4.address }}

{{ ansible_date_time.date }}

{{ ansible_architecture }}

Ansible Playbook

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

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

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

```
playbook.yml
---
- name: play 1
  hosts, configuration parameters, variables

  tasks:
   - install sw component
   - modify config file
     notify:
       - restart_a_service
  - ...

  handlers:
   - restart_a_service
     service:
       name: a_service
       state: restarted
   - ...

- name: play 2
hosts, configuration parameters, variables

...
```

Ansible Modules

Docs » Module Index

Module Index

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

service - Manage services.

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

Synopsis

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

Options

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

redhat

Ansible Galaxy

GALAXY

ABOUT | EXPLORE | BROWSE ROLES | BROWSE AUTHORS | MY ROLES | JSKORZYN

BROWSE ROLES

Keyword | Search roles | SORT Relevance

mysql 1530
ansible role for mysql

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Fedora, Ubuntu
Tags	database, sql
Last Commit	NA
Last Import	NA

Watch 21 | Star 117

nginx 1323
ansible role nginx

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Fedora, Ubuntu
Tags	web
Last Commit	NA
Last Import	NA

Watch 17 | Star 89

network_interface 618
role for system network configuration

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Fedora, Ubuntu
Tags	development, networking, system
Last Commit	NA
Last Import	NA

Watch 12 | Star 55

ntp 9880
ansible role ntp

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Fedora, Ubuntu
Tags	development
Last Commit	NA
Last Import	NA

Watch 8 | Star 23

memcached 601
ansible role memcached

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Fedora, Ubuntu
Tags	web
Last Commit	NA
Last Import	NA

Watch 5 | Star 10

redis 177
ansible role for configuring redid

Type	Ansible
Author	bennojoy
Platforms	Enterprise_Linux, Ubuntu
Tags	web
Last Commit	NA
Last Import	NA

Watch 0 | Star 0

POPULAR TAGS

system	4139
development	2158
web	1838
monitoring	846
networking	746
database	717
cloud	630
packaging	610
ubuntu	341
docker	309

redhat

ANSIBLE TOWER

ANSIBLE TOWER by Red Hat®

TOWER EMPOWERS TEAMS TO AUTOMATE

CONTROL
Scheduled and centralized jobs

KNOWLEDGE
Visibility and compliance

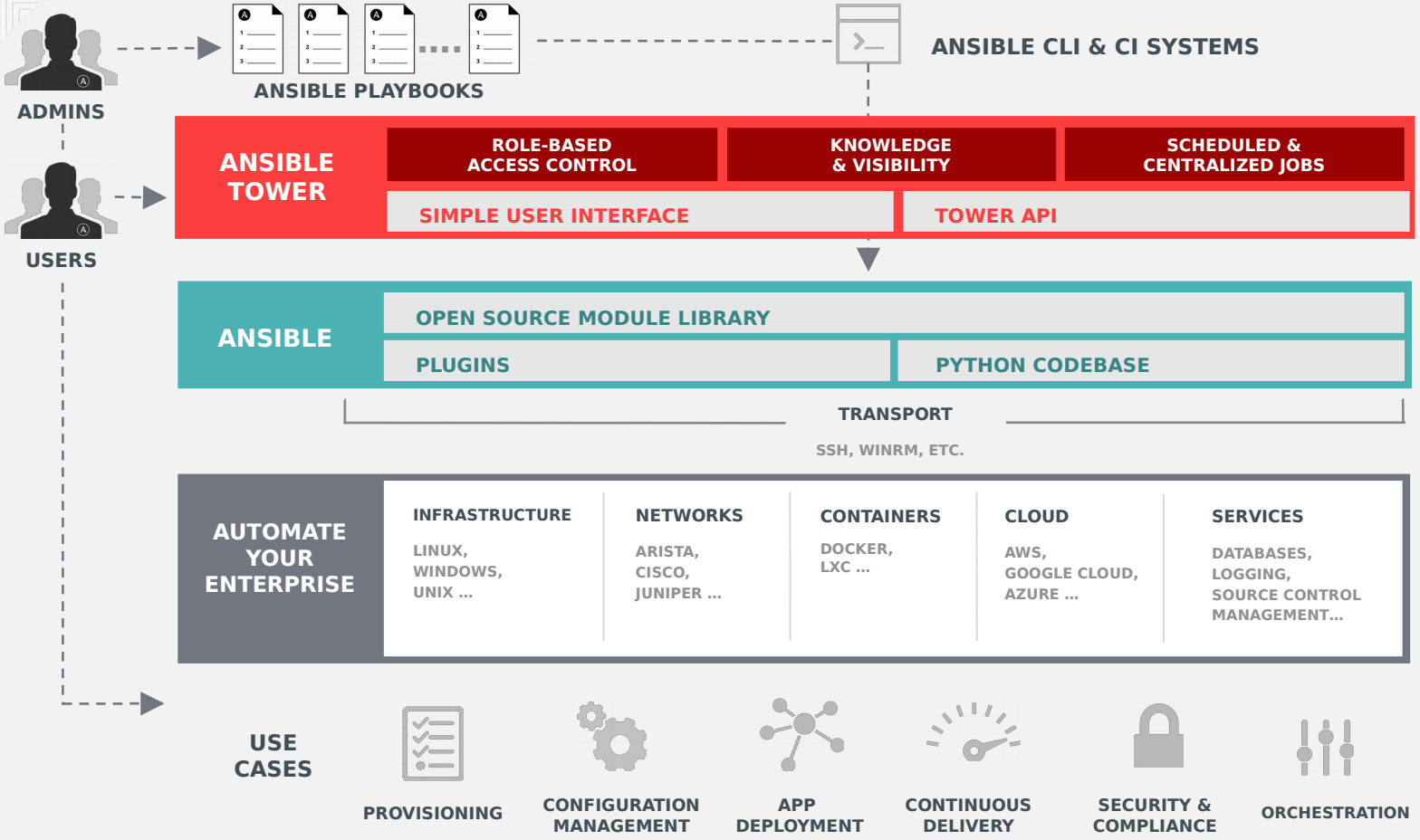
DELEGATION
Role-based access and self-service

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


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
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

redhat.

Ansible Tower

ACCESS CONTROL
Role-based access control & LDAP integration

DELEGATION OF CREDENTIALS
Delegate credentials without giving away secrets

INVENTORY MANAGEMENT
Graphically manage your internal & cloud resources

API & CLI
Documented RESTful API and Tower CLI to integrate Tower into your tools

PUSH-BUTTON LAUNCH
Launch automation jobs with a button

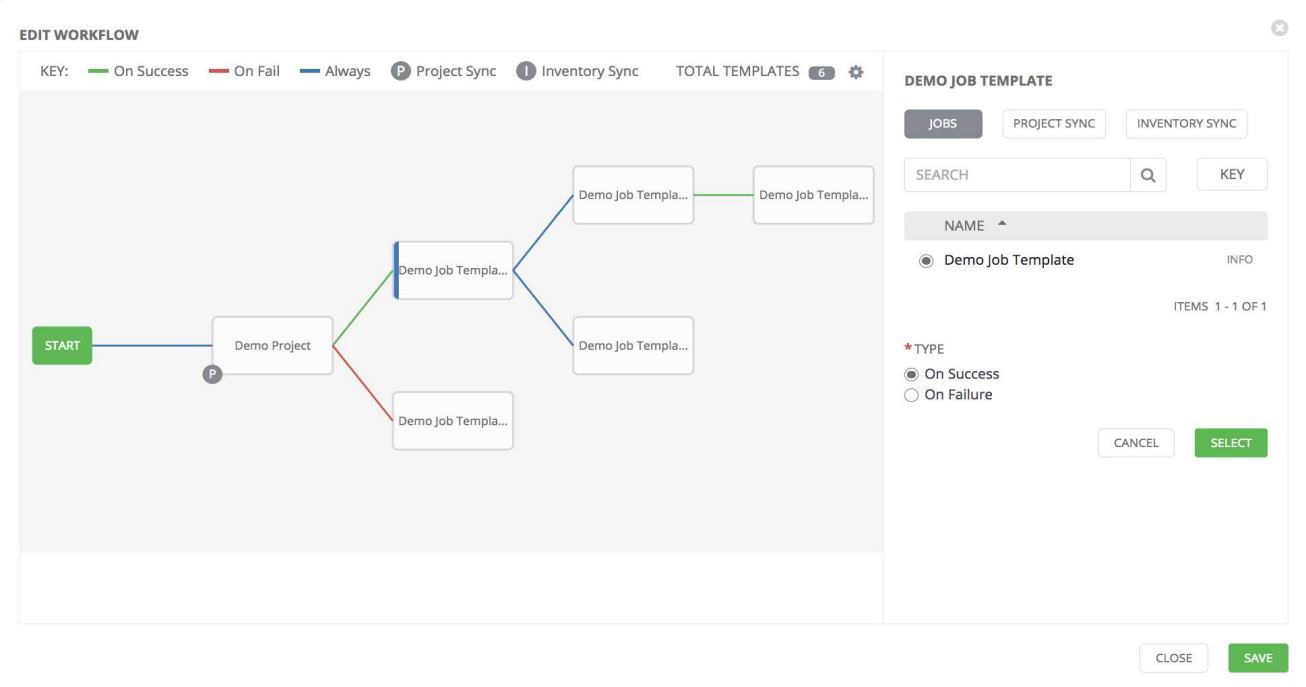
AUDITING
See a full Ansible job history with drill-in details

SCHEDULING
Schedule automation jobs (great for periodic remediation)

Ansible Tower Workflows

MULTI-PLAYBOOK WORKFLOWS

Tower's multi-Playbook workflows chains any number of Playbooks together to create a single workflow. Different Jobs can be run depending on success or failure of the prior Playbook.

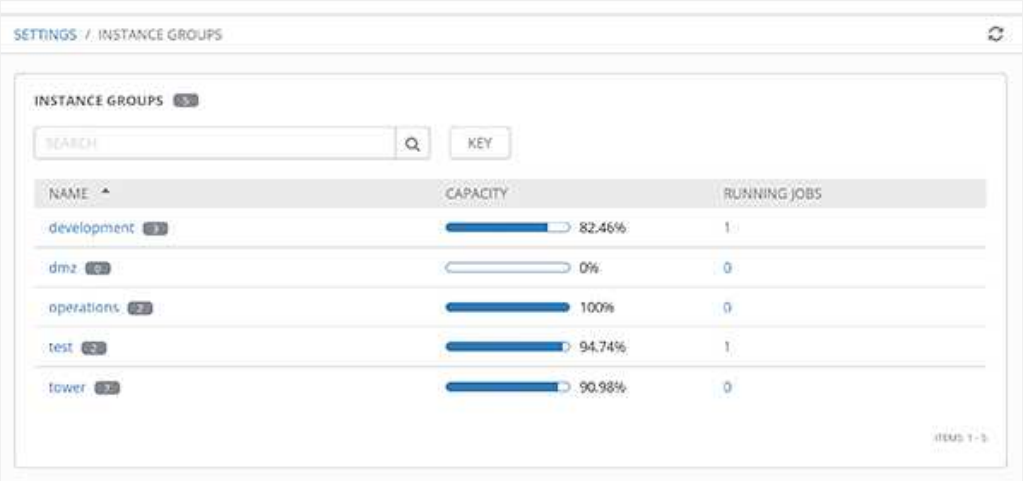


Ansible Tower

ENTERPRISE LOG INTEGRATION

- Log all Tower activity to central enterprise logging
- Cross-reference automation with events and application logs
- Use Tower's API to perform remediation if needed
- Support for:
  - Elastic
  - Splunk
  - Sumologic
  - Loggly
  - Custom (Via WebHook/RESTful API)

Scale-Out Clustering



SCALE-OUT CLUSTERING

Connect multiple Tower nodes into a Tower cluster to add redundancy and capacity to your automation platform.

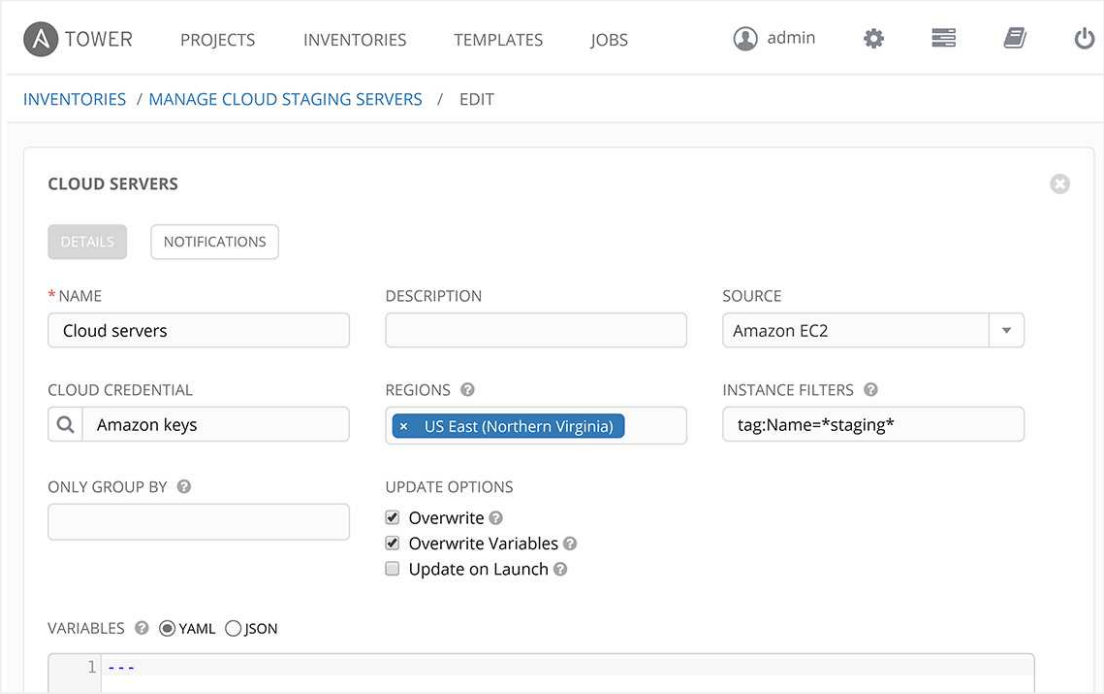
NEW! Add reserved capacity and capacity by organization, and deploy remote execution nodes for additional local capacity.

Manage And Track Your Inventory

MANAGE AND TRACK YOUR INVENTORY

Tower's **inventory syncing** and **provisioning callbacks** allow nodes to request configuration on demand, enabling autoscaling.

NEW! Smart Inventories allow you to organize and automate hosts across all your providers based on a powerful host fact query engine.

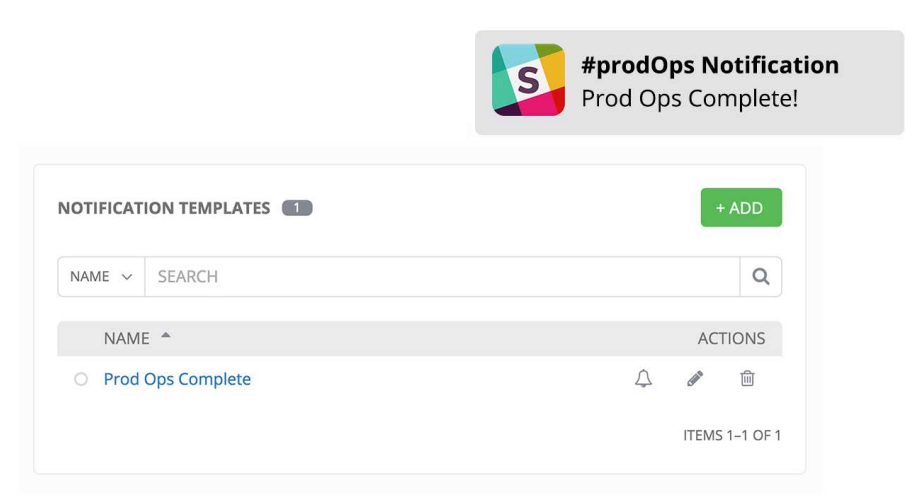


Integrated Notifications

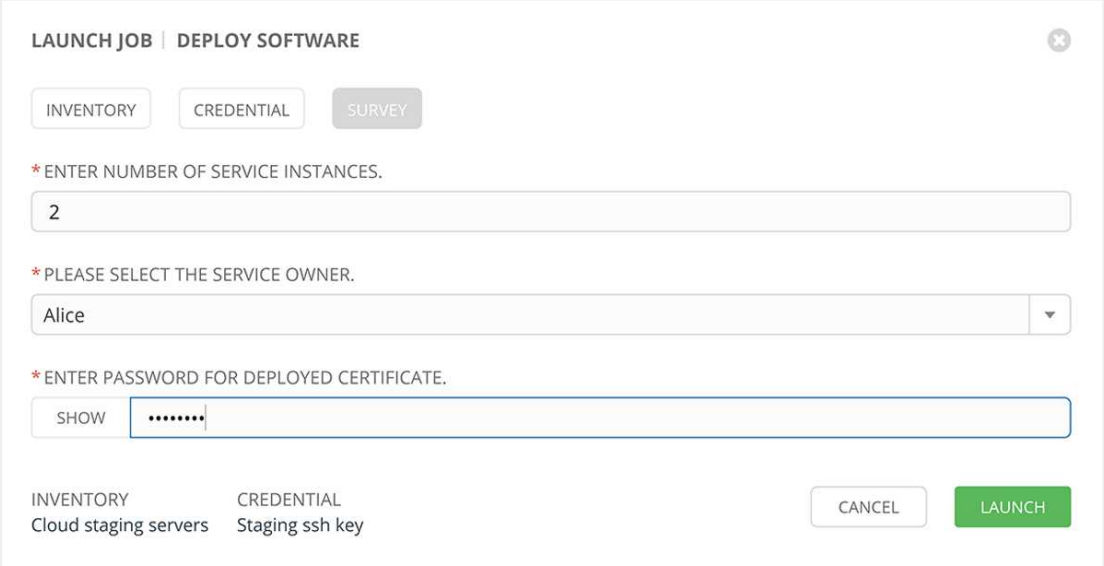
INTEGRATED NOTIFICATIONS

Stay informed of your automation status

via **integrated notifications**. Connect Slack, Hipchat, SMS, email and more.



Self-Service IT



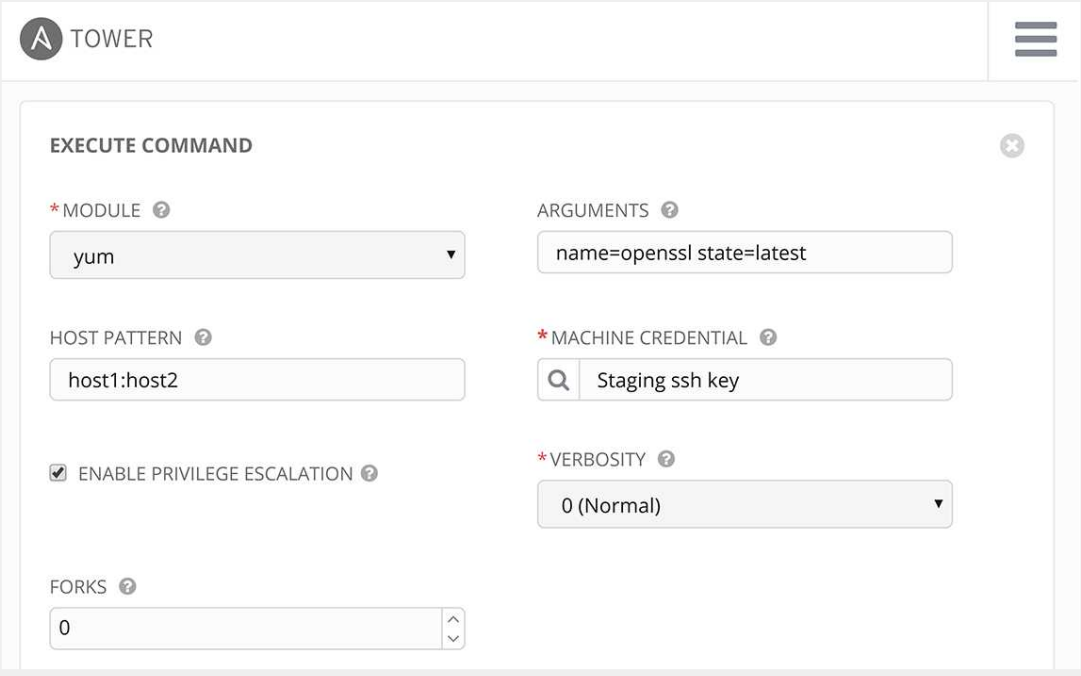
SELF-SERVICE IT

Tower lets you launch Playbooks with just a single click. It can prompt you for variables, let you choose from available secure credentials and monitor the resulting deployments.

Remote Command Execution

REMOTE COMMAND EXECUTION

Run simple tasks on any hosts with Tower's **remote command execution**. Add users or groups, reset passwords, restart a malfunctioning service or patch a critical security issue, quickly.



ANSIBLE EVERYWHERE !

redhat.

Ansible + Red Hat Virtualization

ANSIBLE
by Red Hat®

Red Hat Virtualization and Ansible 2.3 are integrated in order to provide streamlined configuration for:

- Virtual machines
- Virtual networks
- Virtual storage
- Configuration
- Updates

Ansible + Red Hat Virtualization

Provision Virtual Machines

Objective - Deploy application virtual infrastructure.

Ansible Playbook Flow:

- Download a RHEL 7 cloud image from Red Hat CDN
- Upload the image as a disk to RHV storage domain
- Attach disk to a temporary VM
- Create a template from the VM
- Create 8 vms per user based on convention username-rhel7-x
- Tag the VMs per their function
- Run the VMs with cloudinit
  - Configure root password
  - Configure user SSH key

Configure RHV / Build a RHV Data Center

Objective - Deploy a functional RHV Data Center, ready to support virtual machines

Ansible Playbook Flow :

- Create datacenter
- Create cluster/s
- Deploy hosts
- Configure storage domain
- Add users

Ansible + Red Hat Virtualization

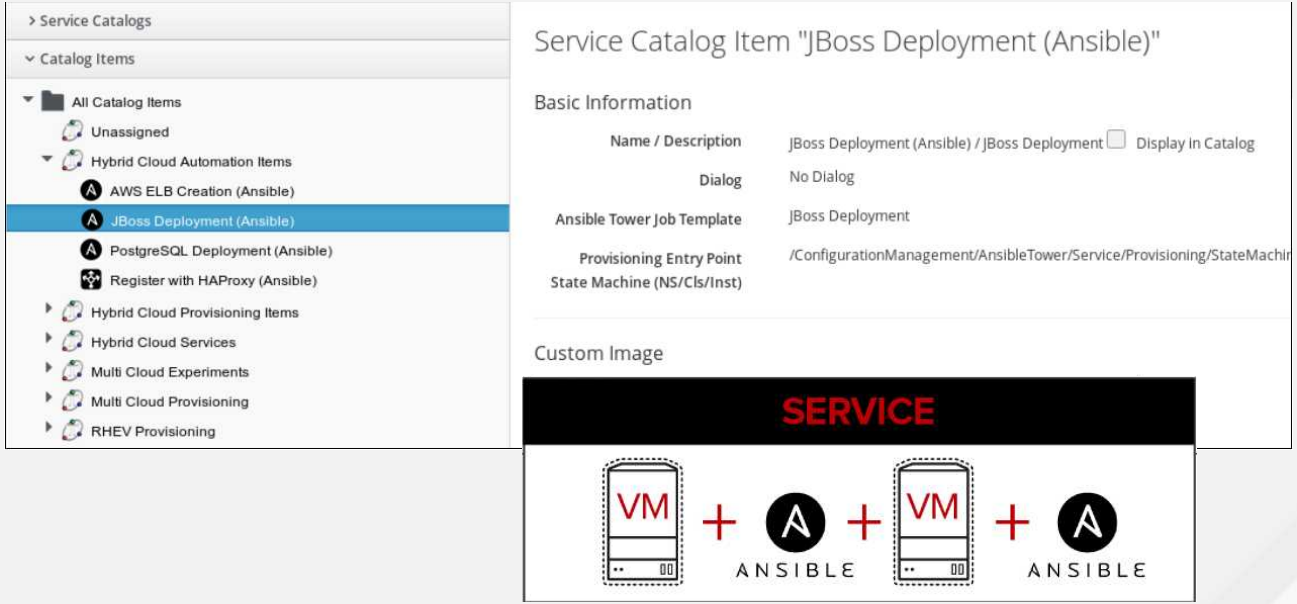
```
# Run VM with cloud init:
ovirt_vms:
    name: rhel7
    template: rhel7
    cluster: Default
    memory: 1GiB
    high_availability: true
    cloud_init:
      nic_boot_protocol: static
      nic_ip_address: 10.34.60.86
      nic_netmask: 255.255.252.0
      nic_gateway: 10.34.63.254
      nic_name: eth1
      nic_on_boot: true
      host_name: example.com
      custom_script: |
        write_files:
         - content: |
             Hello, world!
           path: /tmp/greeting.txt
           permissions: '0644'
      user_name: root
      root_password: super_password
```

```
# Add data iSCSI storage domain:
- ovirt_storage_domains:
    name: data_iscsi
    host: myhost
    data_center: mydatacenter
    iscsi:
      target: iqn.2016-08-
09.domain-01:nickname
      lun_id: 1IET_000d0002
      address: 10.34.63.204
```

```
# Upload local image to disk and
attach it to vm:
# Since Ansible 2.3
- ovirt_disks:
    name: mydisk
    vm_name: myvm
    interface: virtio
    size: 10GiB
    format: cow
    image_path:
/path/to/mydisk.qcow2
    storage_domain: data
```

Ansible + CloudForms

- Automatically deploys and configures requested services on any infrastructure platform.
- Automation steps can be codified in Ansible playbooks or natively in CloudForms.
- Integration to external IT systems allows CloudForms to automate all process steps.



Ansible + CloudForms

Ansible Tower to Ensure Compliance

Objective - Customer has an Ansible playbook that configures security parameters for production systems running in the DMZ.

Ansible Tower and CloudForms:

- Create an Ansible Tower job as control action
- Assign action to CloudForms Compliance Policy

Result - All virtual machines analyzed in the DMZ will have Ansible playbook applied

Ansible Tower in for Application Deployment

Objective - Customer has an Ansible playbook that deploys JBoss in a development environment.

Ansible Tower and CloudForms:

- Create an Ansible Tower job
- Create a CloudForms Service Catalog item related to the Ansible Tower job

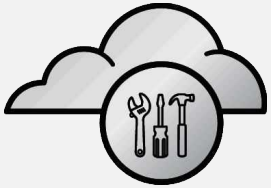
Result - CloudForms service catalog can provision JBoss into development environments utilizing the Ansible playbook.

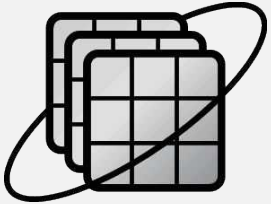
Ansible + CloudForms

Ansible Inside

Simple & Easy Customizations

Off the Shelf, Huge Community, Infinitely Extensible

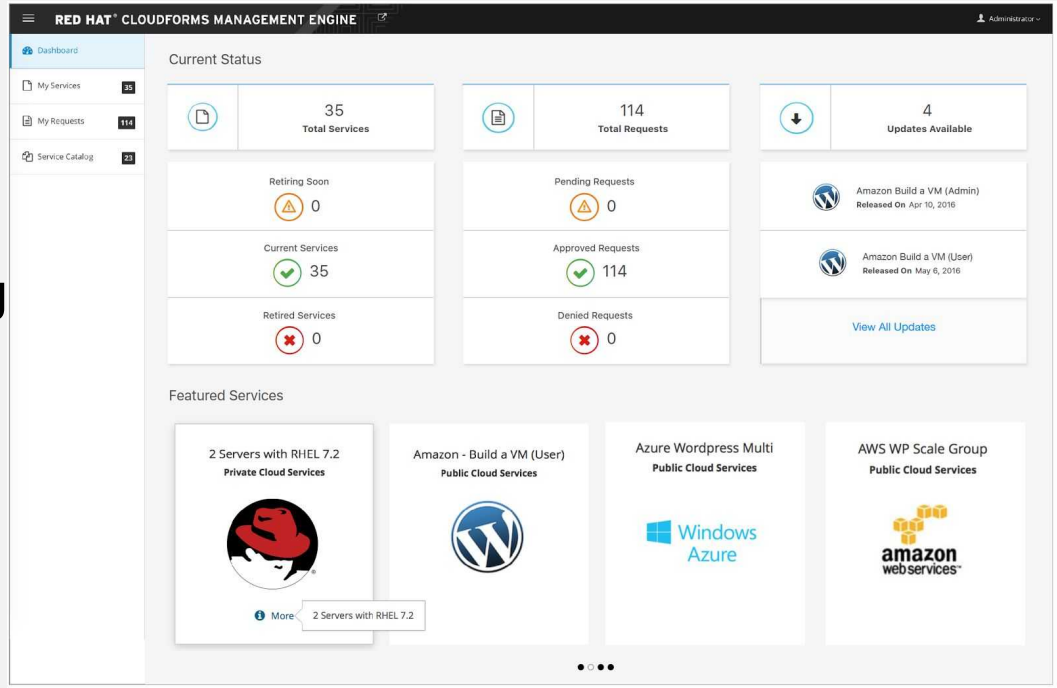




Ansible + CloudForms

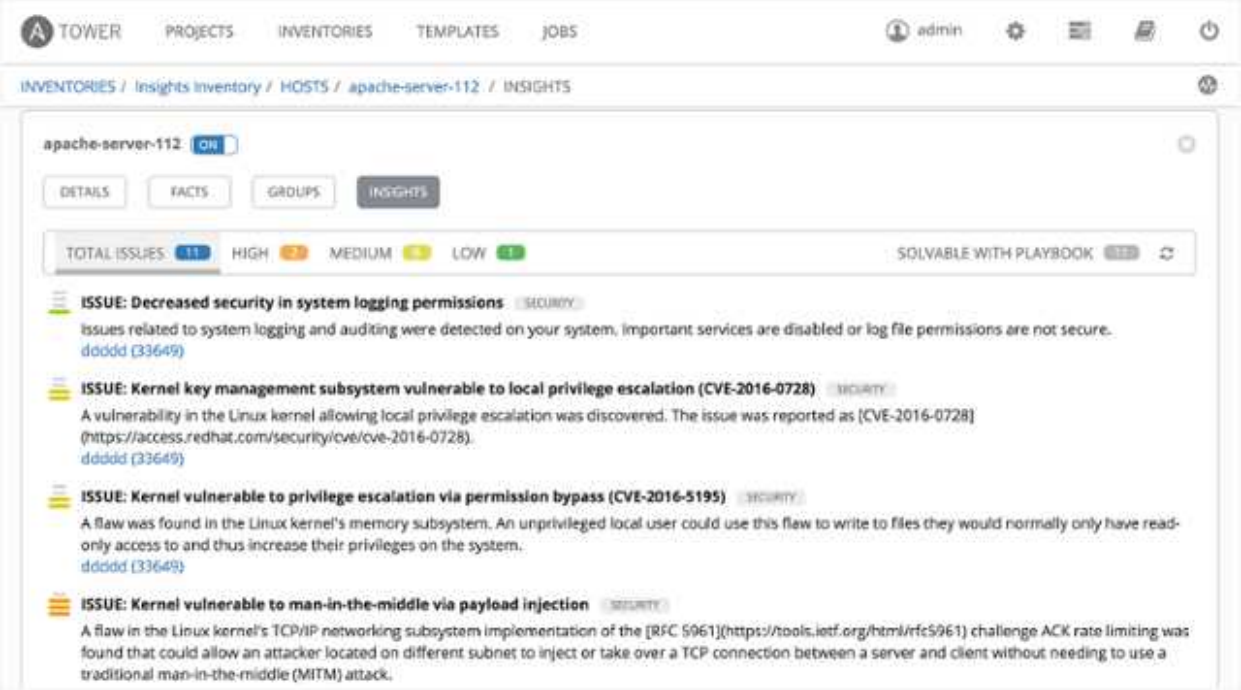
Playbooks as a Service

- nTier Applications
- Compute, Storage, Networking
- Cloud, Virtual or Physical
- Configuration Management



Ansible Tower + Red Hat Insights

NEW! See alerts from Red Hat Insights directly from Tower, and use Insights-provided Playbook Remediation to fix issues in your infrastructure.



redhat.

NETWORK AUTOMATION

redhat.

Network Automation with Ansible

ANSIBLE NETWORK AUTOMATION

Use Ansible to manage, validate, and continuously track heterogeneous network device configurations and deployments.

Network modules are included as part of the Ansible distribution.

33+
Networking platforms

460+
Networking Modules

ansible.com/networking

Network Automation with Ansible

NETWORK INTEGRATIONS

Ansible includes over 250 network modules to support a wide variety of network device vendors, including:

A10	AVI Networks	ARISTA	CISCO	cumulus networks	DELL EMC
	MORE INFO	MORE INFO	MORE INFO	MORE INFO	MORE INFO
f5	Hewlett Packard Enterprise	JUNIPER NETWORKS	Lenovo	NOKIA	paloalto NETWORKS
MORE INFO		MORE INFO	MORE INFO	MORE INFO	
Pluribus NETWORKS					
MORE INFO					

Ansible Network Modules

Nxos

- nxos_aaa_server - Manages AAA server global configuration.
- nxos_aaa_server_host - Manages AAA server host-specific configuration.
- nxos_acl - Manages access list entries for ACLs.
- nxos_acl_interface - Manages applying ACLs to interfaces.
- nxos_bgp - Manages BGP configuration.
- nxos_bgp_af - Manages BGP Address-family configuration.
- nxos_bgp_neighbor - Manages BGP neighbors configurations.
- nxos_bgp_neighbor_af - Manages BGP address-family's neighbors configuratior
- nxos_command - Run arbitrary command on Cisco NXOS devices
- nxos_config - Manage Cisco NXOS configuration sections
- nxos_evpn_global - Handles the EVPN control plane for VXLAN.
- nxos_evpn_vni - Manages Cisco EVPN VXLAN Network Identifier (VNI).
- nxos_facts - Gets facts about NX-OS switches
- nxos_feature - Manage features in NX-OS switches.
- nxos_file_copy - Copy a file to a remote NXOS device over SCP.
- nxos_gir - Trigger a graceful removal or insertion (GIR) of the switch.
- nxos_gir_profile_management - Create a maintenance-mode or normal-mode p
- nxos_hsrp - Manages HSRP configuration on NX-OS switches.
- nxos_igmp - Manages IGMP global configuration.
- nxos_igmp_interface - Manages IGMP interface configuration.
- nxos_igmp_snooping - Manages IGMP snooping global configuration.
- nxos_install_os - Set boot options like boot image and kickstart image.
- nxos_interface - Manages physical attributes of interfaces.
- nxos_interface_ospf - Manages configuration of an OSPF interface instance.
- nxos_ip_interface - Manages L3 attributes for IPv4 and IPv6 interfaces.
- nxos_mtu (D) - Manages MTU settings on Nexus switch.
- nxos_ntp - Manages core NTP configuration.
- nxos_ntp_auth - Manages NTP authentication.
- nxos_ntp_options - Manages NTP options.

Cloudengine

- ce_aaa_server - Manages AAA server global configuration on HUAWEI CloudEngine switches.
- ce_aaa_server_host - Manages AAA server host configuration on HUAWEI
- ce_acl - Manages base ACL configuration on HUAWEI CloudEngine switch
- ce_acl_advance - Manages advanced ACL configuration on HUAWEI Clou
- ce_acl_interface - Manages applying ACLs to interfaces on HUAWEI Cloud
- ce_bgp - Manages BGP configuration on HUAWEI CloudEngine switches.
- ce_bgp_af - Manages BGP Address-family configuration on HUAWEI Clou
- ce_bgp_neighbor - Manages BGP peer configuration on HUAWEI CloudEn
- ce_bgp_neighbor_af - Manages BGP neighbor Address-family configuratic
- ce_command - Run arbitrary command on HUAWEI CloudEngine devices.
- ce_config - Manage Huawei CloudEngine configuration sections.
- ce_dldp - Manages global DLDP configuration on HUAWEI CloudEngine sv
- ce_dldp_interface - Manages interface DLDP configuration on HUAWEI C
- ce_eth_trunk - Manages Eth-Trunk interfaces on HUAWEI CloudEngine sw
- ce_evpn_bd_vni - Manages EVPN VXLAN Network Identifier (VNI) on HU
- ce_evpn_bgp - Manages BGP EVPN configuration on HUAWEI CloudEngin
- ce_evpn_bgp_rr - Manages RR for the VXLAN Network on HUAWEI Cloud
- ce_evpn_global - Manages global configuration of EVPN on HUAWEI Clou
- ce_facts - Gets facts about HUAWEI CloudEngine switches.
- ce_file_copy - Copy a file to a remote cloudengine device over SCP on HUA
- ce_info_center_debug - Manages information center debug configuration
- ce_info_center_global - Manages outputting logs on HUAWEI CloudEngine
- ce_info_center_log - Manages information center log configuration on HUA
- ce_info_center_trap - Manages information center trap configuration on H
- ce_interface - Manages physical attributes of interfaces on HUAWEI Clouc
- ce_interface_ospf - Manages configuration of an OSPF interface instanceo

F5

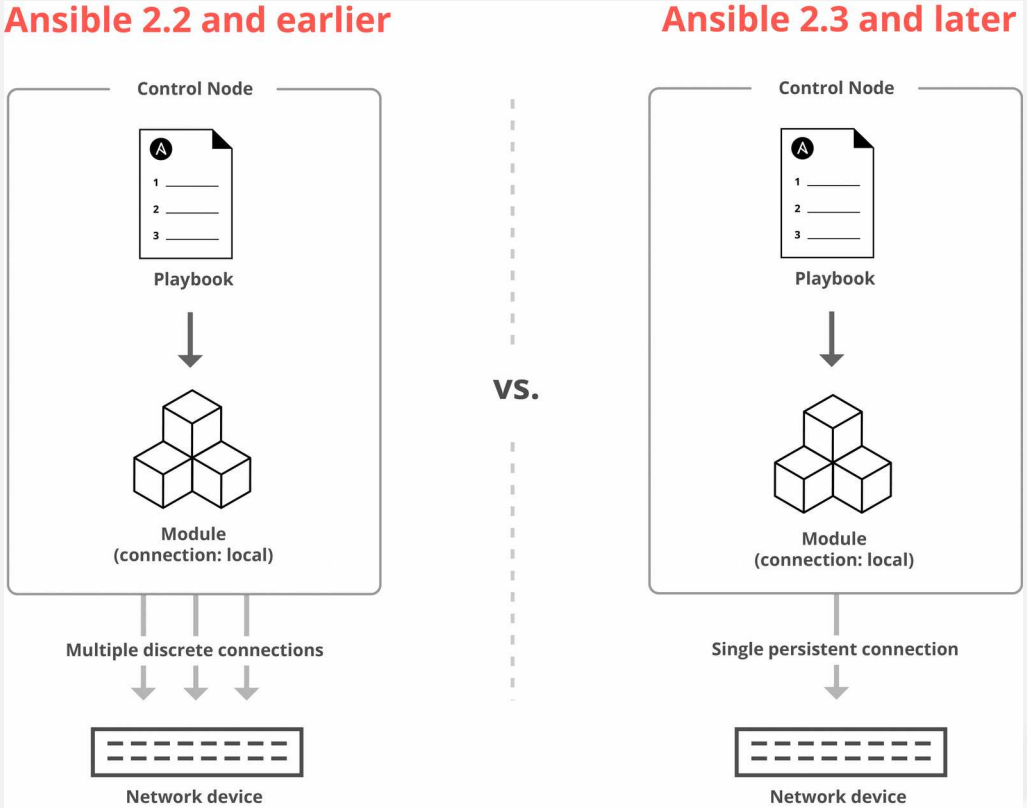
- bigip_command - Run arbitrary command on F5 devices.
- bigip_device_dns - Manage BIG-IP device DNS settings
- bigip_device_ntp - Manage NTP servers on a BIG-IP
- bigip_device_sshd - Manage the SSHD settings of a BIG-IP
- bigip_facts - Collect facts from F5 BIG-IP devices
- bigip_gtm_datacenter - Manage Datacenter configuration in BIG-IP
- bigip_gtm_facts - Collect facts from F5 BIG-IP GTM devices.
- bigip_gtm_virtual_server - Manages F5 BIG-IP GTM virtual servers
- bigip_gtm_wide_ip - Manages F5 BIG-IP GTM wide ip
- bigip_hostname - Manage the hostname of a BIG-IP.
- bigip_irule - Manage iRules across different modules on a BIG-IP.
- bigip_monitor_http - Manages F5 BIG-IP LTM http monitors
- bigip_monitor_tcp - Manages F5 BIG-IP LTM tcp monitors
- bigip_node - Manages F5 BIG-IP LTM nodes
- bigip_pool - Manages F5 BIG-IP LTM pools
- bigip_pool_member - Manages F5 BIG-IP LTM pool members
- bigip_routedomain - Manage route domains on a BIG-IP
- bigip_selfip - Manage Self-IPs on a BIG-IP system
- bigip_snat_pool - Manage SNAT pools on a BIG-IP.
- bigip_snmp_trap - Manipulate SNMP trap information on a BIG-IP.
- bigip_ssl_certificate - Import/Delete certificates from BIG-IP
- bigip_sys_db - Manage BIG-IP system database variables
- bigip_sys_global - Manage BIG-IP global settings.
- bigip_user - Manage user accounts and user attributes on a BIG-IP.
- bigip_virtual_server - Manages F5 BIG-IP LTM virtual servers
- bigip_vlan - Manage VLANs on a BIG-IP system

redhat

Ansible – Networking

New Connection Plugins:

- Network_cli
  Designed to work with traditional network devices that require connectivity to a device CLI in order to configure resources.
- Netconf
  Designed to work primarily with netork devices using the netconf protocol.



NEW OFFERING

RED HAT® ANSIBLE® Automation

This is the Ansible Product Family. It is used to generally refer to Ansible products.

RED HAT® ANSIBLE® Engine

Introducing Red Hat Ansible Engine. Simple, powerful, agentless. Includes full support for execution engine, core modules, and the Red Hat Subscription.

RED HAT® ANSIBLE® Tower

Control, knowledge, delegation for operationalizing enterprise automation environments for teams.

Trainings and Materials

Ansible Courses:

- DO407 - Automation with Ansible
- EX407 - Red Hat Certificate of Expertise in Ansible Automation

Materials:

- Tower Trial - ansible.com/tower-trial
- Getting started – ansible.com/getting-started
- More? - ansible.com/whitepapers

Thank you


plus.google.com/+RedHat


facebook.com/redhatinc


linkedin.com/company/red-hat


twitter.com/RedHatNews


youtube.com/user/RedHatVideos