



# Automation with Ansible: Past, Present & Future

Red Hat Virtual Training



**New York Virtual Training:**  
Automation with  
Ansible: Past,  
Present & Future

Thursday, January 27, 2022  
1:00pm ET; 10:00am PT

carahtsoft.<sup>®</sup>

For more information, contact Carahsoft or our reseller partners:  
[redhat@carahtsoft.com](mailto:redhat@carahtsoft.com) | 877-RHAT-GOV



# New York Virtual Training: Automation with Ansible: Past, Present & Future

Thursday, January 27, 2022  
1:00pm ET; 10:00am PT



**Red Hat**

carahsoft.<sup>®</sup>



## **Connor Murray**

Red Hat Account Manager

703-230-7583

[Connor.Murray@carahsoft.com](mailto:Connor.Murray@carahsoft.com)



**Red Hat**

**carahsoft.**



## About Carahsoft

Carahsoft is The Trusted Public Sector IT Solutions Provider®, supporting Federal, State and Local Government and Education and Healthcare organizations with IT products, services and training through our partners and contracts.



## Red Hat Ansible

*Simple, powerful, agentless IT automation*

Red Hat Ansible Automation Platform provides a foundation for building and operating IT automation, at scale.



**Red Hat**

**carahtsoft.**



## **Featured Expert:**



**Travis Michette**  
Principal Instructor,  
**Red Hat**



**Red Hat**  
Ansible Automation  
Platform

# **Ansible Automation Workshop**

## **Ansible: Past, Present, and Future**

**An overview of Ansible from Ansible Engine to Ansible Automation Platform 2.x**

**Travis Michette**  
**Principal Instructor**



**Red Hat**  
**Training**

# Housekeeping

- Timing
- Breaks
- Takeaways
- Materials: <https://red.ht/aap2x>
  - RHLS Subscribers - DO374EA



# What you will learn

- Introduction to Ansible Automation
- How it works
- Understanding modules, tasks & playbooks
- How to execute Ansible commands & Playbooks
- Evolution of Ansible
  - Ansible Playbooks and Ad-Hoc Commands
  - Ansible Roles
  - Ansible Collections
  - Ansible Execution Environments
- Ansible Content Navigator, Ansible Automation Hub, and Ansible Controller (High-Level Overview)

# Agenda

- Introduction
- Ansible Engine (Past)
- Ansible Automation Platform 1.x (Present)
- Break (10 min)
- Ansible Automation Platform 2x (Future)
- Ansible Automation Training



**Red Hat**  
Ansible Automation  
Platform

# Introduction

Topics Covered:

- What is the Ansible Automation Platform?
- What can it do?

# Why Ansible?



## Simple

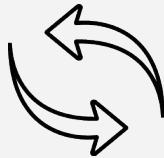
Human readable automation

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**

# What can I do using Ansible?

Automate the deployment and management of your entire IT footprint.

**Do this...**



**On these...**



# Ansible automates technologies you use

Time to automate is measured in minutes

| Cloud             | Virt & Container | Windows  | Network    | Security   | Monitoring   |
|-------------------|------------------|----------|------------|------------|--------------|
| AWS               | Docker           | ACLs     | A10        | Checkpoint | Dynatrace    |
| Azure             | VMware           | Files    | Arista     | Cisco      | Datadog      |
| Digital Ocean     | RHV              | Packages | Aruba      | CyberArk   | LogicMonitor |
| Google            | OpenStack        | IIS      | Cumulus    | F5         | New Relic    |
| OpenStack         | OpenShift        | Regedits | Bigswitch  | Fortinet   | Sensu        |
| Rackspace         | +more            | Shares   | Cisco      | Juniper    | +more        |
| +more             |                  | Services | Dell       | IBM        |              |
| Operating Systems | Storage          | Configs  | Extreme    | Palo Alto  | Devops       |
| RHEL              | Netapp           | Users    | F5         | Snort      | Jira         |
| Linux             | Red Hat Storage  | Domains  | Lenovo     | +more      | GitHub       |
| Windows           | Infinidat        | +more    | MikroTik   |            | Vagrant      |
| +more             | +more            |          | Juniper    |            | Jenkins      |
|                   |                  |          | OpenSwitch |            | Slack        |
|                   |                  |          | +more      |            | +more        |

# Section 1

# Ansible Engine

*Past*



**Red Hat**  
Ansible Automation  
Platform

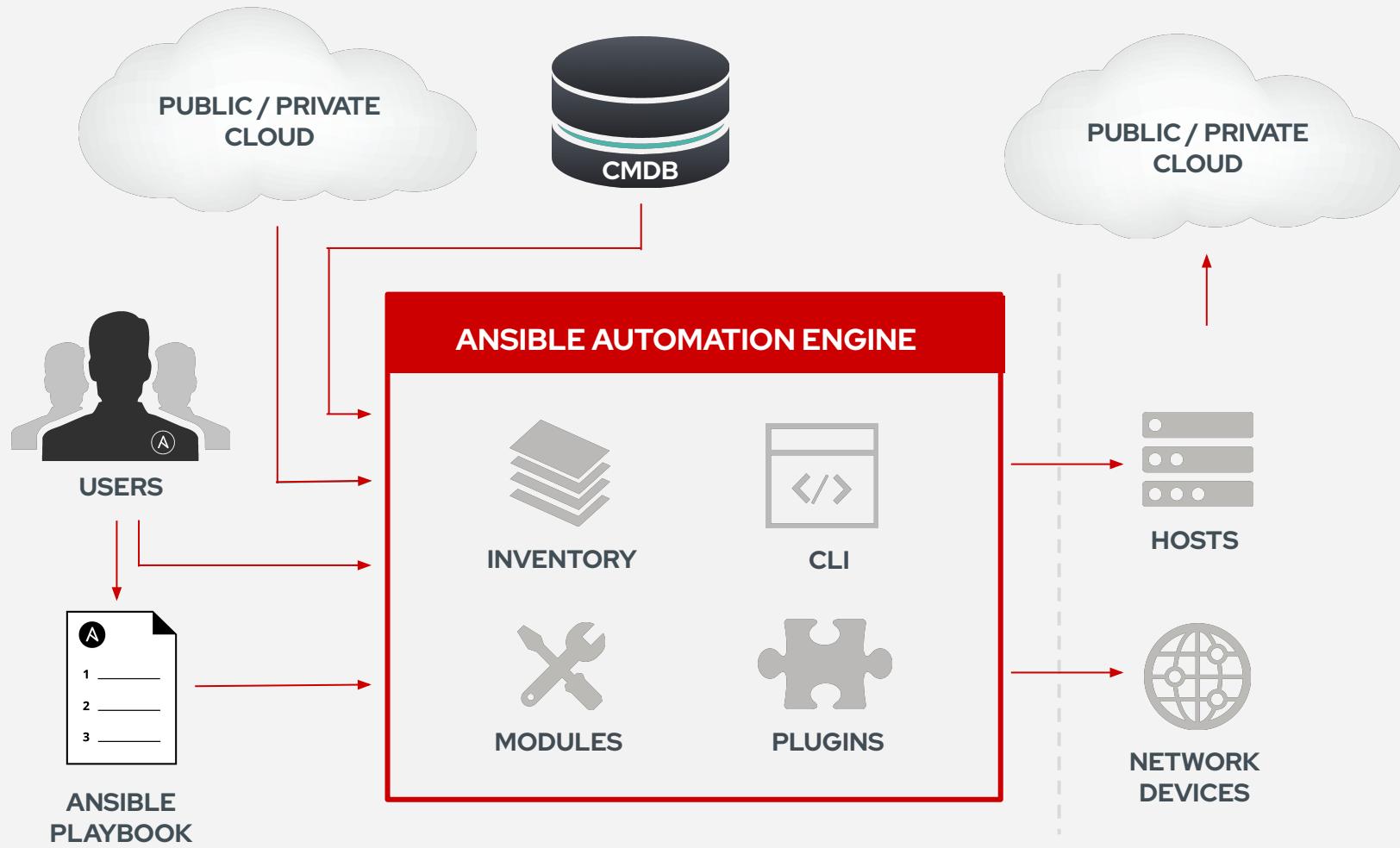


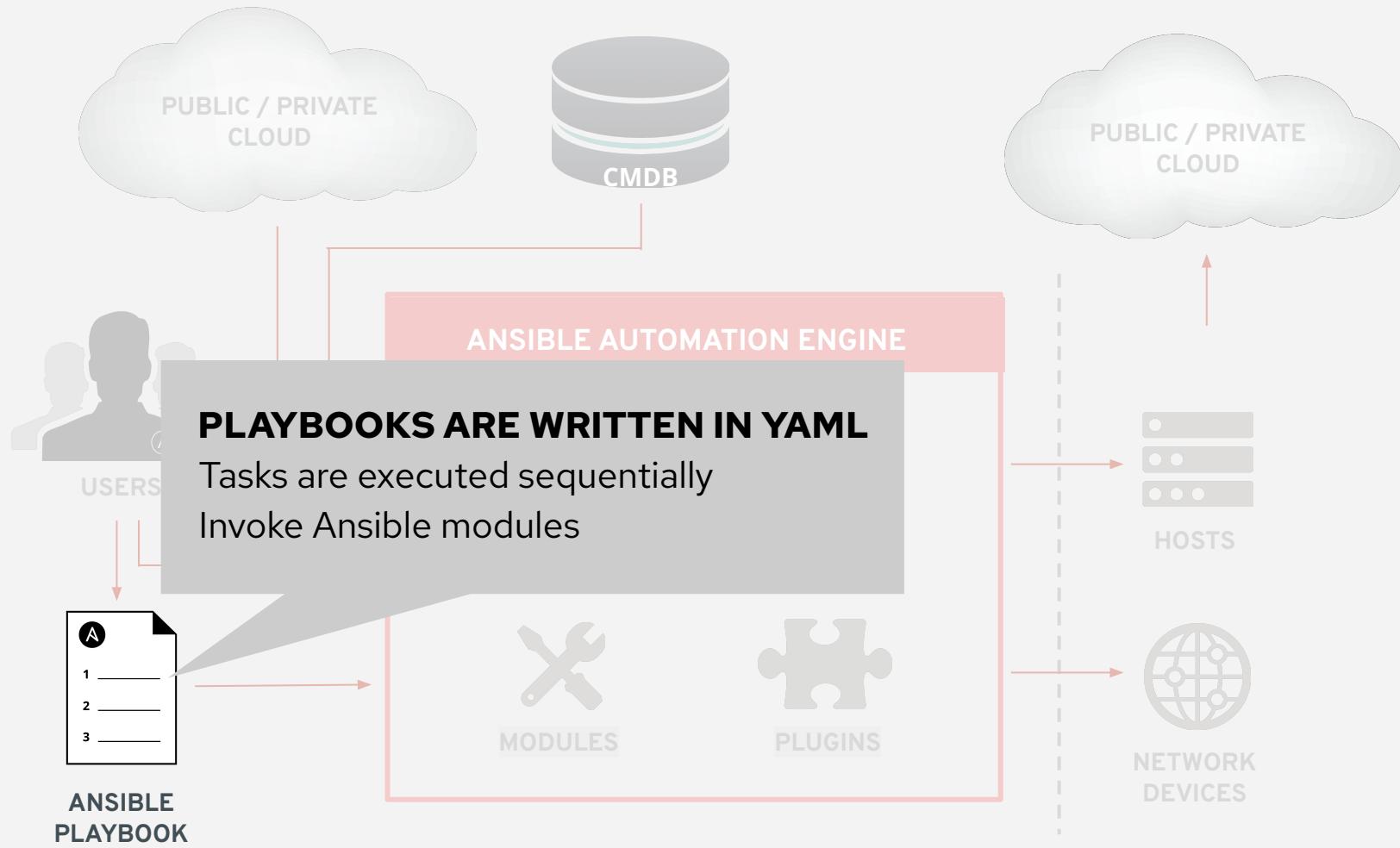
**Red Hat**  
Ansible Automation  
Platform

# Section 1.1

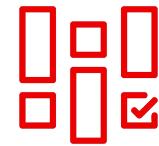
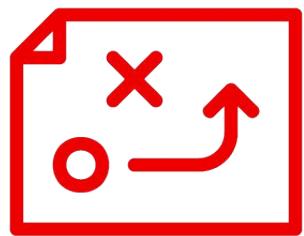
Topics Covered:

- Understanding the Ansible Infrastructure
- Ansible Tower (Enterprise Solutions)





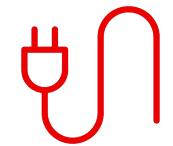
# What makes up an Ansible playbook?



Plays



Modules



Plugins

# Ansible plays

What am I automating?



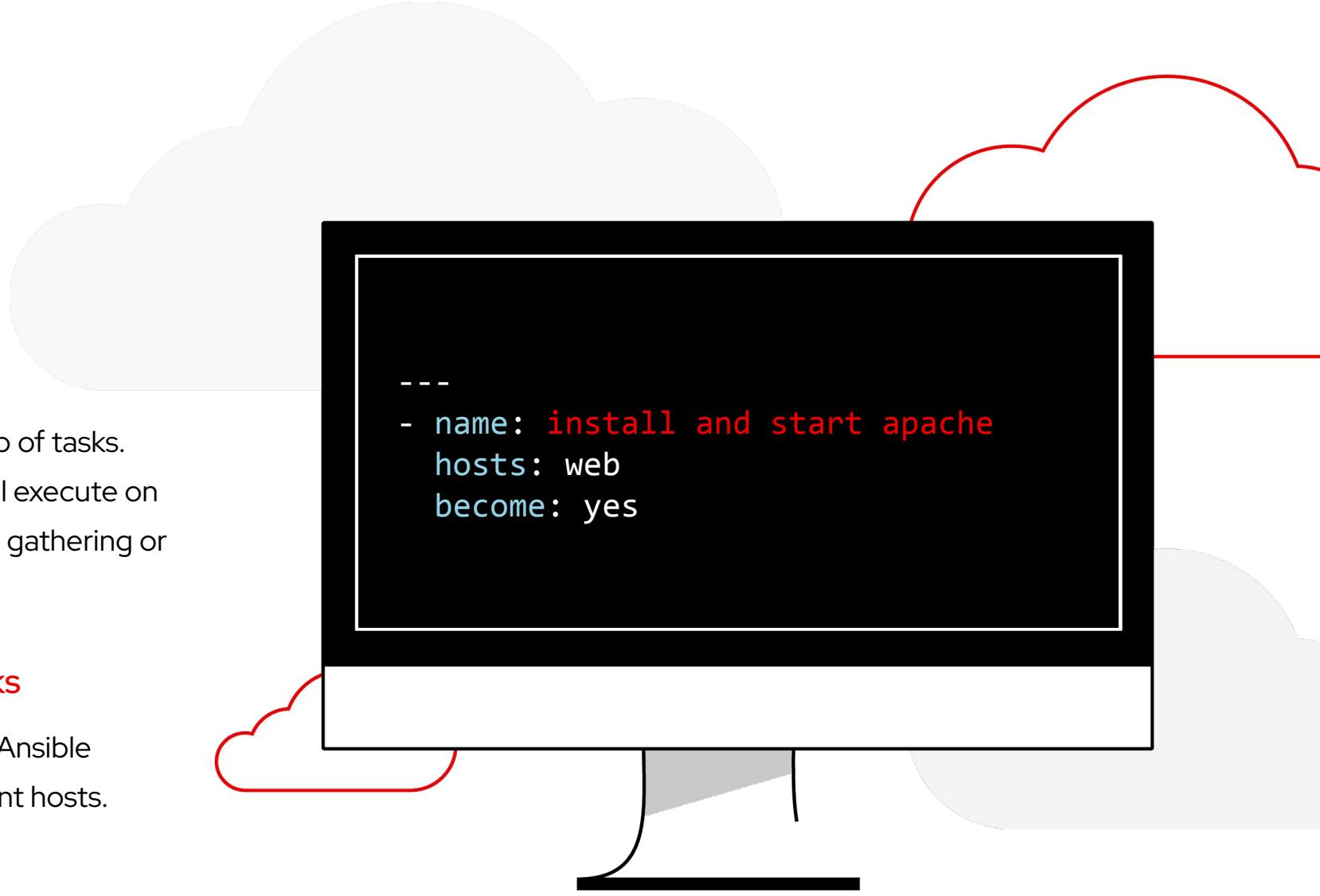
## What are they?

Top level specification for a group of tasks.  
Will tell that play which hosts it will execute on  
and control behavior such as fact gathering or  
privilege level.



## Building blocks for playbooks

Multiple plays can exist within an Ansible  
playbook that execute on different hosts.

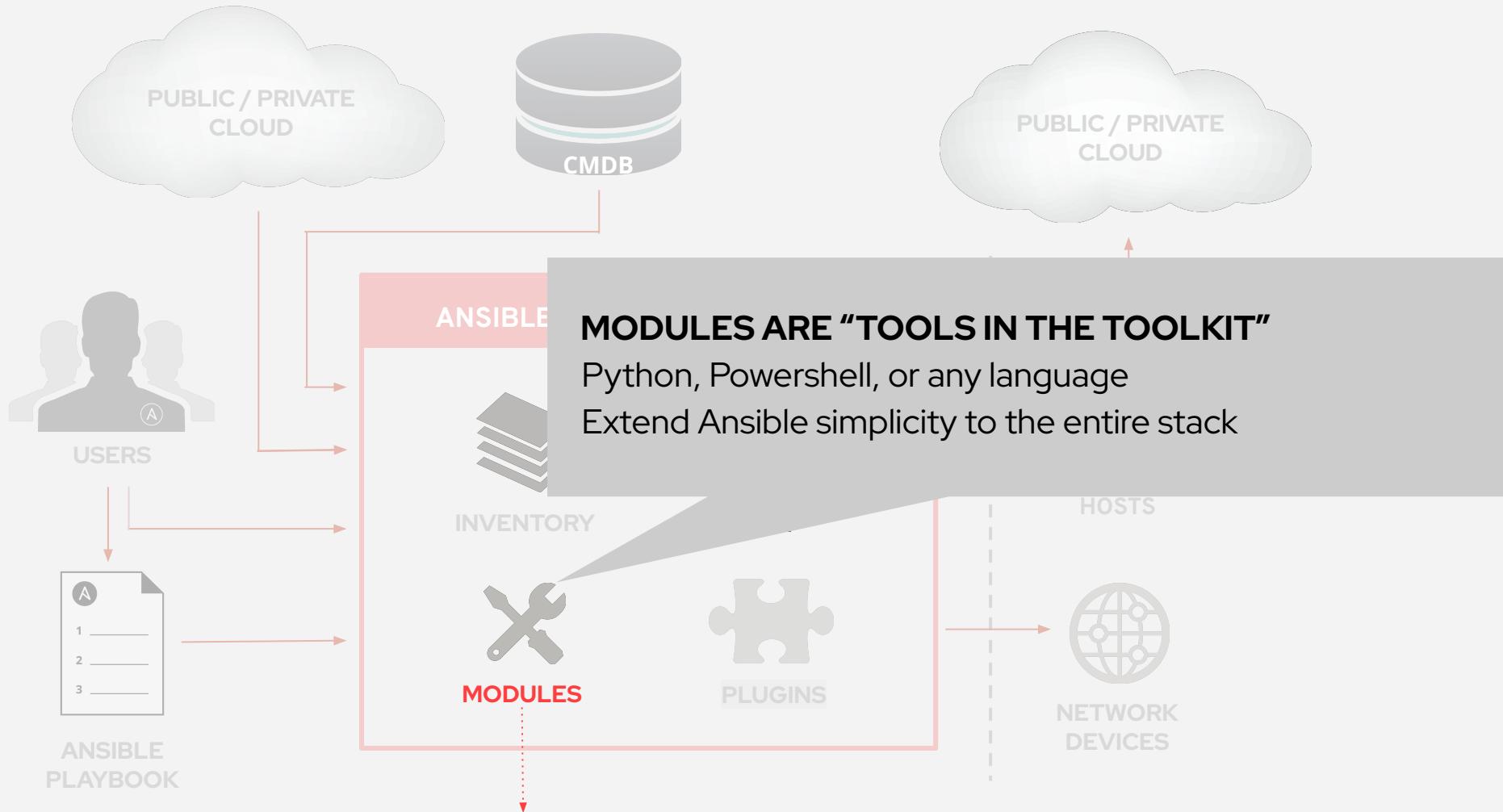


```
---
```

- **name: install and start apache**  
**hosts:** web  
**become:** yes

**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



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

# Ansible modules

The “tools in the toolkit”



## What are they?

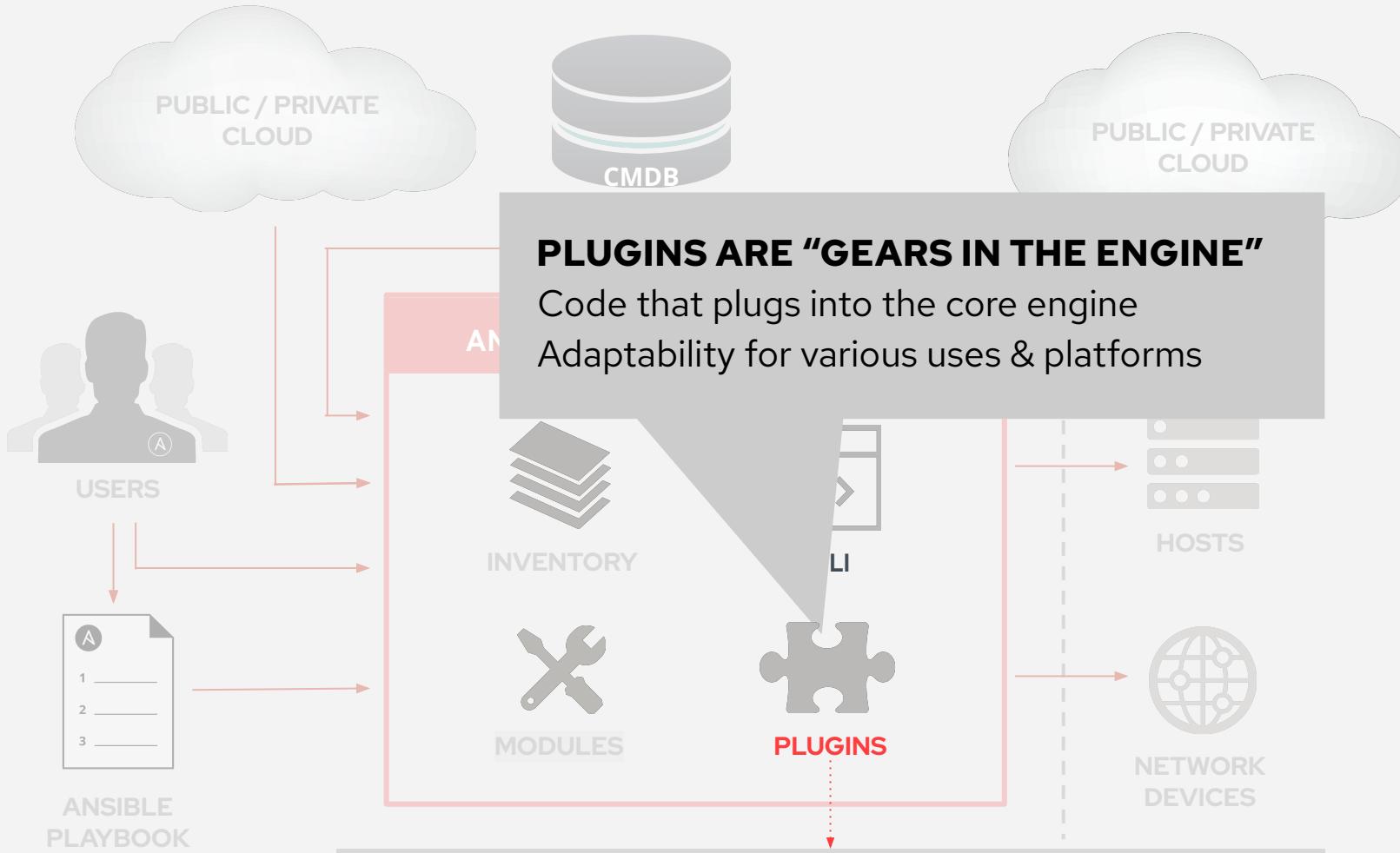
Parametrized components with internal logic,  
representing a single step to be done.  
The modules “do” things in Ansible.



## Language

Usually Python, or Powershell for Windows  
setups. But can be of any language.





```
{ { some_variable | to_nice_yaml } }
```

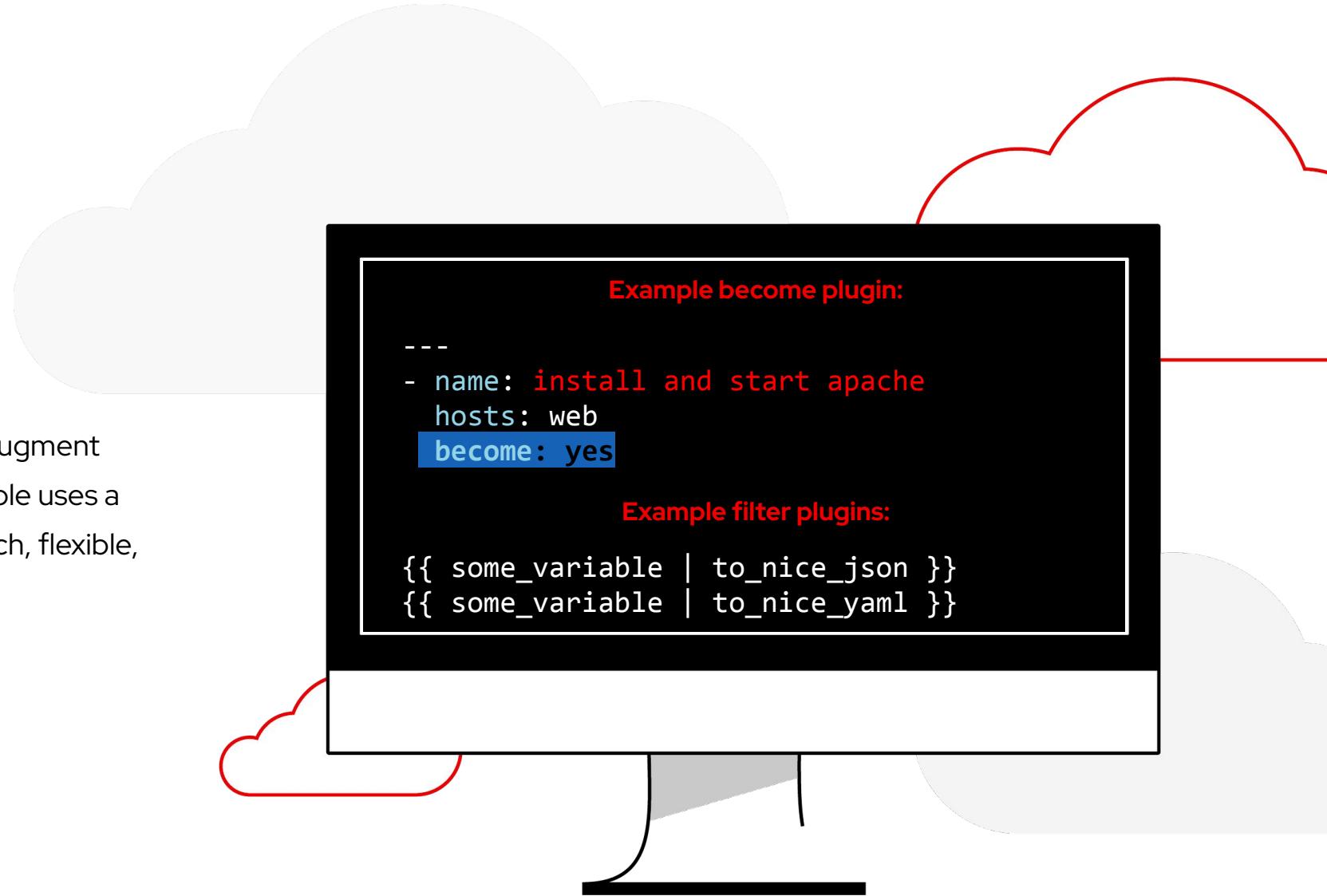
# Ansible plugins

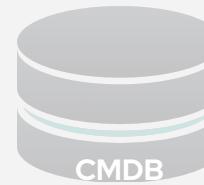
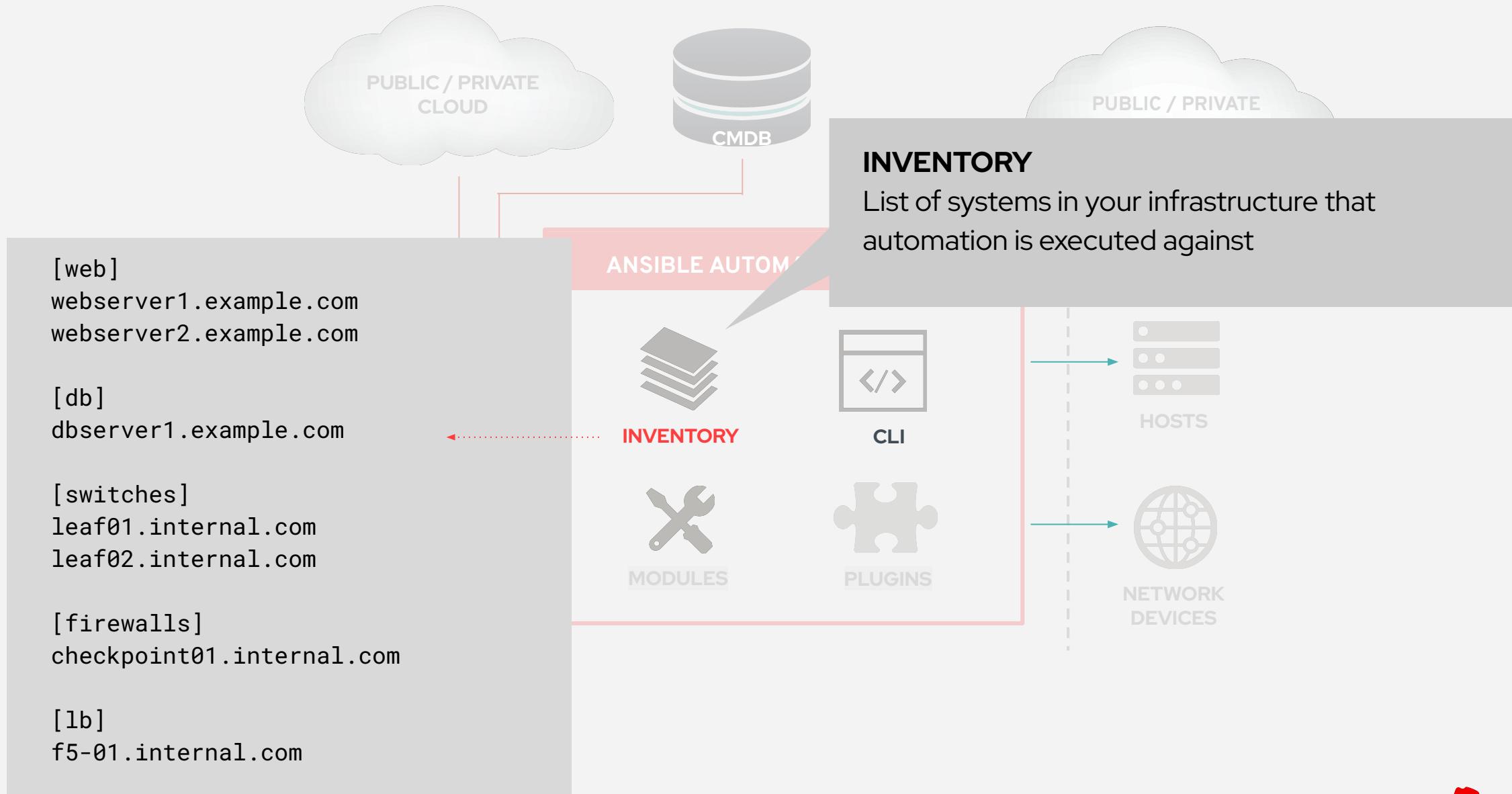
The “extra bits”



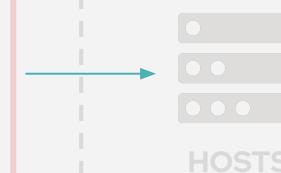
## What are they?

Plugins are pieces of code that augment Ansible's core functionality. Ansible uses a plugin architecture to enable a rich, flexible, and expandable feature set.

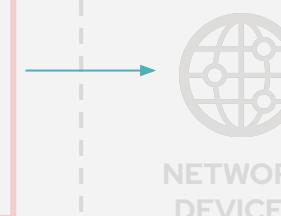




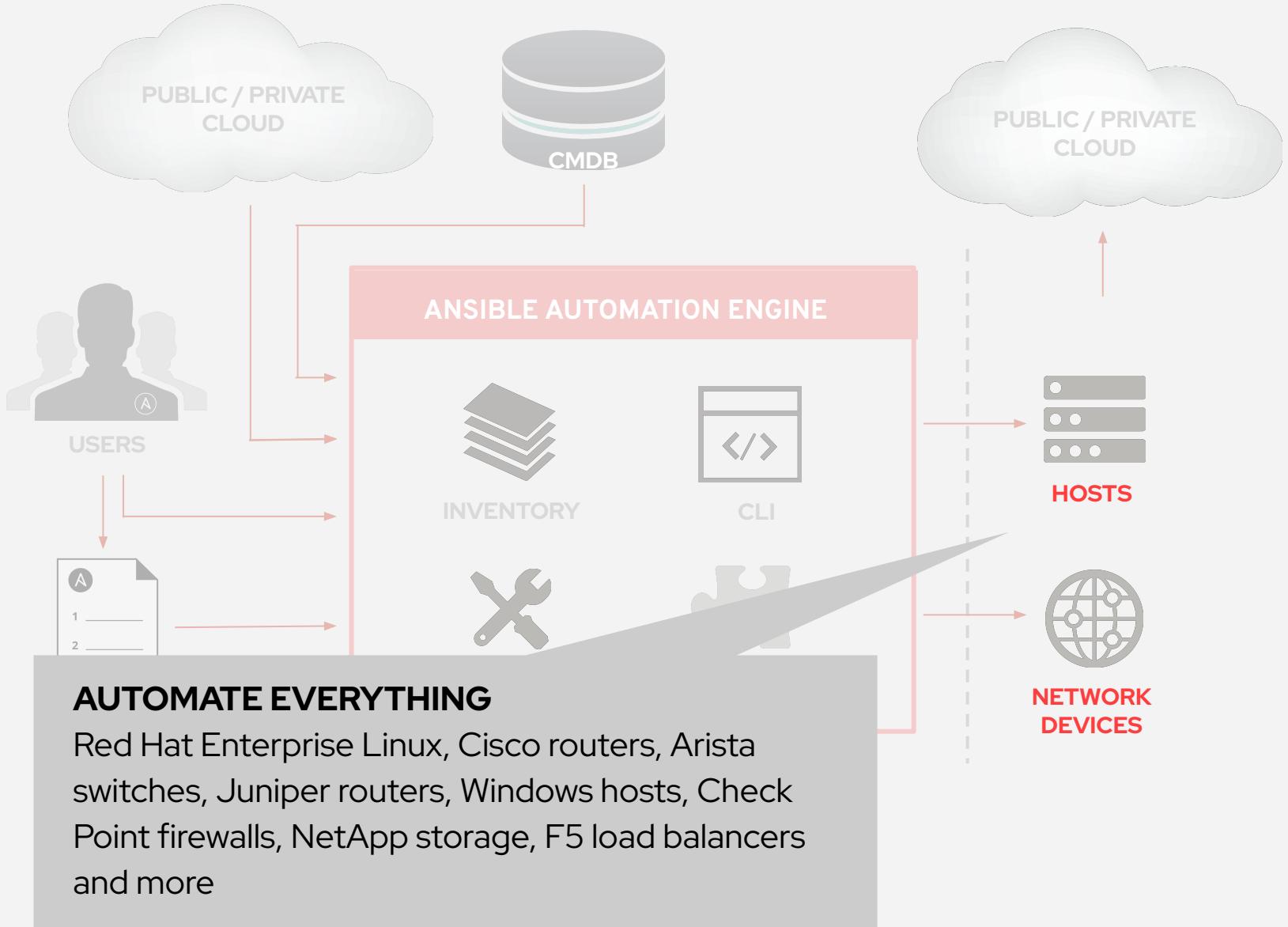
PUBLIC / PRIVATE



HOSTS



NETWORK  
DEVICES



# LINUX AUTOMATION

**150+**  
Linux Modules

**AUTOMATE EVERYTHING  
LINUX**

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

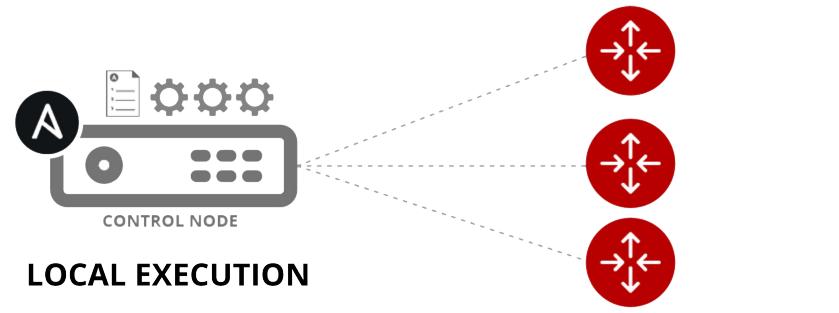
**ONLY REQUIREMENTS:**  
**Python 2 (2.6 or later)**  
**or Python 3 (3.5 or later)**

[ansible.com/get-started](http://ansible.com/get-started)



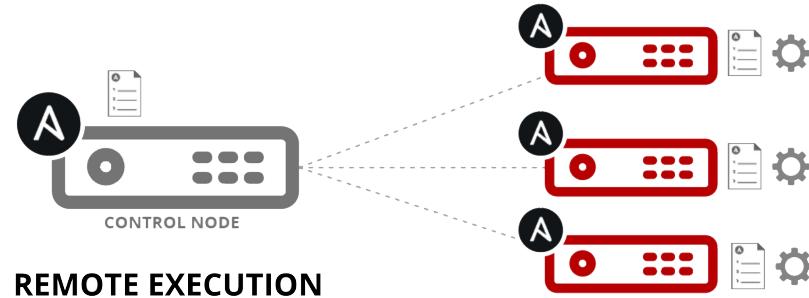
# How Ansible Automation works

*Module code is executed locally on the control node*



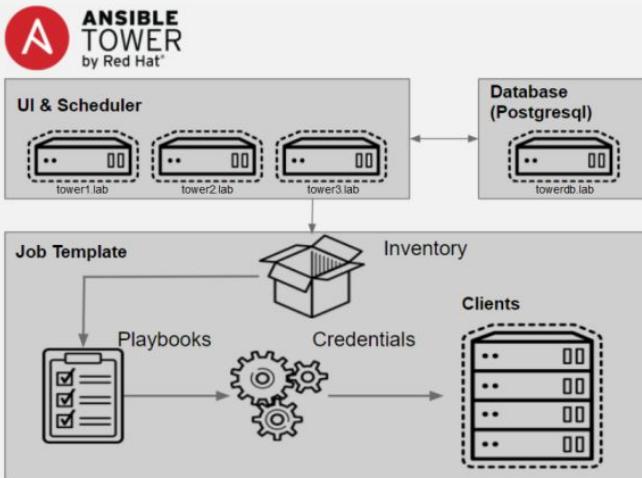
**NETWORKING  
DEVICES**

*Module code is copied to the managed node, executed, then removed*



**LINUX/WINDOWS  
HOSTS**

# Ansible Tower



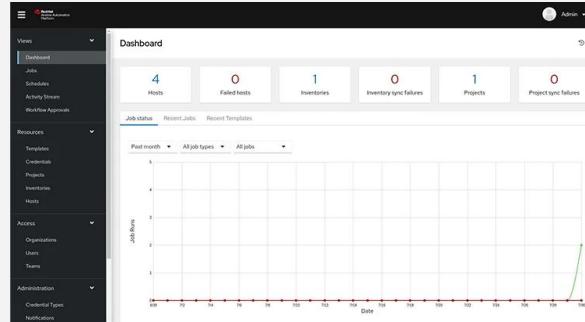
## Integrated

Manage Projects and Jobs

No CLI Administration skills needed

Automated

**Single Management Point**



## Simple

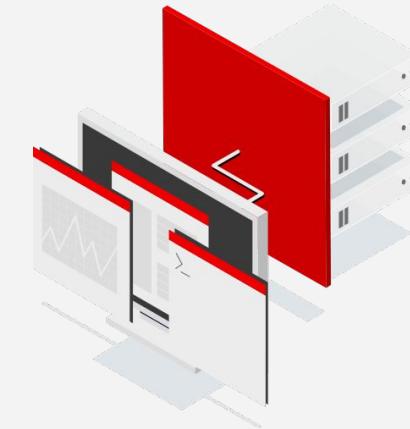
Environment Overview

Configuration management

Workflow orchestration

Logging and System Management

**Manage Access & Files**



## Streamlined

Web Interface / WebUI

Rest API

Plugins

User Management

Users / Roles / Credentials

**More Efficient & More Secure**



# 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 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.

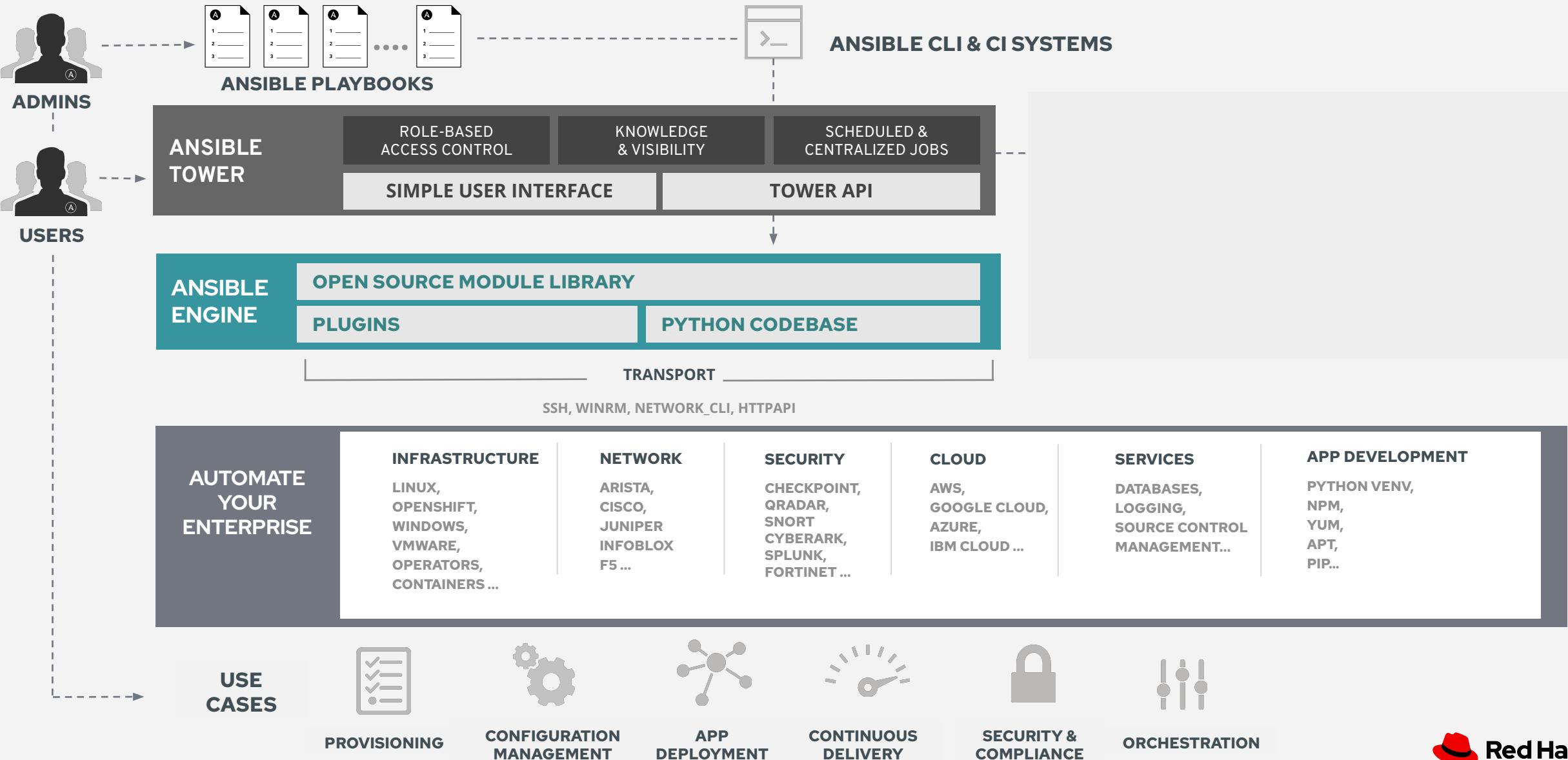
## Centralized logging

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.

# Ansible Automation Engine





**Red Hat**  
Ansible Automation  
Platform

# Section 1.2

## Topics Covered:

- Ansible inventories
- Main Ansible config file
- Modules and ad-hoc commands

# Inventory

- Ansible works against multiple systems in an **inventory**
- Inventory is usually file based
- Can have multiple groups
- Can have variables for each group or even host

# Understanding Inventory - (Simple Inventory)

```
# Static inventory example:  
[myservers]  
10.42.0.2  
10.42.0.6  
10.42.0.7  
10.42.0.8  
10.42.0.100  
host.example.com
```

# Understanding Inventory - Hosts

## [app1srv]

```
appserver01 ansible_host=10.42.0.2  
appserver02 ansible_host=10.42.0.3
```

## [web]

```
node-[1:30] ansible_host=10.42.0.[31:60]
```

## [web:vars]

```
apache_listen_port=8080  
apache_root_path=/var/www/mywebdocs/
```

## [all:vars]

```
ansible_user=kev  
ansible_ssh_private_key_file=/home/kev/.ssh/id_rsa
```

- Inventory can be written in short format and expanded using [x:y] syntax

## [web]

```
Node-1 ansible_host=10.42.0.31  
Node-2 ansible_host=10.42.0.32
```

...

```
Node-30 ansible_host=10.42.0.60
```

# Understanding Inventory - Variables

## [app1srv]

```
appserver01 ansible host=10.42.0.2  
appserver02 ansible_host=10.42.0.3
```

## [web]

```
node-[1:30] ansible_host=10.42.0.[31:60]
```

## [web:vars]

```
apache_listen_port=8080  
apache_root_path=/var/www/mywebdocs/
```

## [all:vars]

```
ansible_user=ender  
ansible_ssh_private_key_file=/home/ender/.ssh/id_rsa
```

# Understanding Inventory - Groups

**[nashville]**

bnaapp01

bnaapp02

**[atlanta]**

atlapp03

atlapp04

**[south:children]**

atlanta

nashville

hsvapp05

# Configuration File

- Basic configuration for Ansible
- Can be in multiple locations, with different precedence
- Here: `.ansible.cfg` in the home directory
- Configures where to find the inventory

# Ansible Configuration

Configuration files will be searched for in the following order (Highest Precedence to Lowest):

- **ANSIBLE\_CONFIG** (environment variable if set)
- **ansible.cfg** (in the current directory)
- **~/.ansible.cfg** (in the home directory)
- **/etc/ansible/ansible.cfg** (installed as Ansible default)

# The Ansible Configuration File: `ansible.cfg`

```
[user@ansible] $ cat ansible.cfg
```

```
[defaults]
inventory = inventory
remote_user = devops
```

# First Ad-Hoc Command: ping

- Single Ansible command to perform a task quickly directly on command line
- Most basic operation that can be performed
- Utilizes a single Ansible Module with options and arguments
- Here: an example Ansible ping - not to be confused with ICMP

```
$ ansible all -m ping
```

# Ad-Hoc Commands    ping

```
# Check connections (submarine ping, not ICMP)
[user@ansible] $ ansible all -m ping
```

```
web1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python":
"/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

# The Ansible Command

Some basics to keep you from getting stuck

--help (Display some basic and extensive options)

```
[user@ansible ~]$ ansible --help
```

```
Usage: ansible <host-pattern> [options]
```

```
Define and run a single task 'playbook' against a set of hosts
```

```
Options:
```

```
-a MODULE_ARGS, --args=MODULE_ARGS  
                      module arguments  
--ask-vault-pass      ask for vault password  
-B SECONDS, --background=SECONDS  
<<<snippet, output removed for brevity>>>
```

# Ad-Hoc Commands

Here are some common options you might use:

**-m MODULE\_NAME , --module-name=MODULE\_NAME**

Module name to execute the ad-hoc command

**-a MODULE\_ARGS , --args=MODULE\_ARGS**

Module arguments for the ad-hoc command

**-b , --become**

Run ad-hoc command with elevated rights such as sudo, the default method

**-e EXTRA\_VARS , --extra-vars=EXTRA\_VARS**

Set additional variables as key=value or YAML/JSON

# Ad-Hoc Commands

Here are some common options you might use:

```
# Check connections to all (submarine ping, not ICMP)  
[user@ansible] $ ansible all -m ping
```

```
# Run a command on all the hosts in the web group  
[user@ansible] $ ansible web -m command -a "uptime"
```

```
# Collect and display known facts for server "webl"  
[user@ansible] $ ansible webl -m setup
```



# **Red Hat** Ansible Automation Platform

## Demo Time

Ansible - Ad-Hoc Command to Test Environment (Ansible Ping)

Ansible - Ad-Hoc Command to Create User and Sudoers File



**Red Hat**



**Red Hat**  
Ansible Automation  
Platform

# Section 1.3

Topics Covered:

- Playbooks basics
- Running a playbook

# An Ansible Playbook

A play

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

  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
```

# An Ansible Playbook

A task

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

  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
```

# An Ansible Playbook

module



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

  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
```

# Running an Ansible Playbook:

The most important colors of Ansible

A task executed as expected, no change was made.

A task executed as expected, making a change

A task failed to execute successfully

# Running an Ansible Playbook

```
[user@ansible] $ ansible-playbook apache.yml

PLAY [webservers] ****
TASK [Gathering Facts] ****
ok: [web2]
ok: [web1]
ok: [web3]

TASK [Ensure httpd package is present] ****
changed: [web2]
changed: [web1]
changed: [web3]

TASK [Ensure latest index.html file is present] ****
changed: [web2]
changed: [web1]
changed: [web3]

TASK [Restart httpd] ****
changed: [web2]
changed: [web1]
changed: [web3]

PLAY RECAP ****
web2          : ok=1    changed=3  unreachable=0   failed=0
web1          : ok=1    changed=3  unreachable=0   failed=0
web3          : ok=1    changed=3  unreachable=0   failed=0
```



# Red Hat Ansible Automation Platform

## Demo Time

Ansible Engine - Running a Playbook to Create User and Sudoers File

Ansible Engine - Running a Playbook to Deploy Webserver (Failure - AAP)



Red Hat



**Red Hat**  
Ansible Automation  
Platform

# Section 1.4

Topics Covered:

- What are roles?
- What is the structure of a Role?
- Ansible Galaxy

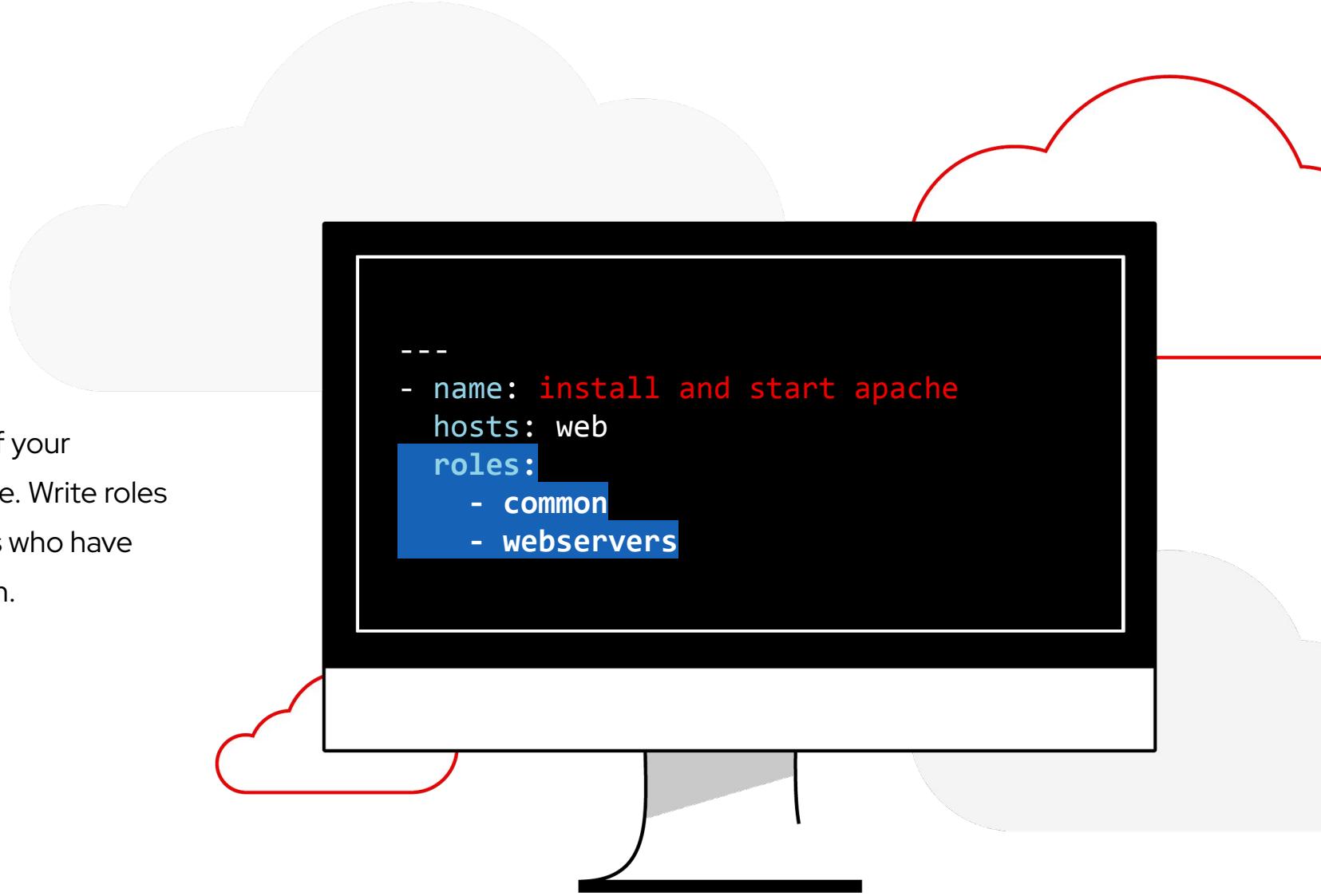
# Ansible roles

Reusable automation actions



## What are they?

Group your tasks and variables of your automation in a reusable structure. Write roles once, and share them with others who have similar challenges in front of them.



# Ansible Roles

- ❖ Ansible roles provide a way to reuse Ansible code generically and more effectively and have the following benefits:
  - Groups content for easy sharing of code with others
  - Make large projects manageable
  - Developed in parallel by different parties
  - Written generically and can be placed in version control
- ❖ Ansible roles can be shared via SCM or publicly through Ansible Galaxy

## Installing and Using a Role

```
$ ansible-galaxy install tmichett.deploy_packages
```

### Playbook using a Role

```
---
```

- **name: Install Packages**
- hosts: web**
- become: yes**
- roles:**
  - **tmichett.deploy\_packages**

## Creating an Ansible Role (beyond scope)

- ❖ Use the **ansible-galaxy init <RoleName>** command to create a Role
- ❖ Empty directories or unused directories can be deleted to clean up the Role
- ❖ Populate the various Role structures
  - Must have the following components (at minimum):
    - README.md
    - meta/main.yaml
    - tasks/main.yaml

# Role structure

- **Defaults:** default variables with lowest precedence (e.g. port)
- **Files:** contains files that are deployed
- **Handlers:** contains all handlers
- **Meta:** role metadata including dependencies to other roles.

*TIP: Used to construct some of the Ansible Galaxy automated documentation*

- **README:** contains the README for the role and used for Galaxy README
- **Tasks:** plays or tasks

*TIP: It's common to include tasks in main.yml with "when" (e.g. OS == xyz)*
- **Templates:** templates to deploy
- **Tests:** place for playbook tests
- **Vars:** variables (e.g. override port)

```
role_name/
  └── defaults
      └── main.yml
  └── files
  └── handlers
      └── main.yml
  └── meta
      └── main.yml
  └── README.md
  └── tasks
      └── main.yml
  └── templates
  └── tests
      └── inventory
          └── test.yml
  └── vars
```

# Ansible Galaxy

Sharing  
Content

Community

Roles, and  
more

v1 - Set config file to use on boot.

1. Write multiple configuration files.
  - For each environment/region.
2. Inspect metadata on boot and use the matching config file.

v1 - Set config file to use on boot.

1. Write multiple configuration files.
  - For each environment/region.
2. Inspect metadata on boot and use the matching config file.



# Red Hat Ansible Automation Platform

## Demo Time

Ansible Engine - Playbook with Roles (Warning - Uses Collections from Newer Ansible)



Red Hat



**Red Hat**  
Ansible Automation  
Platform

# Section 2

# Ansible Automation

# Platform 1.2

*Present*



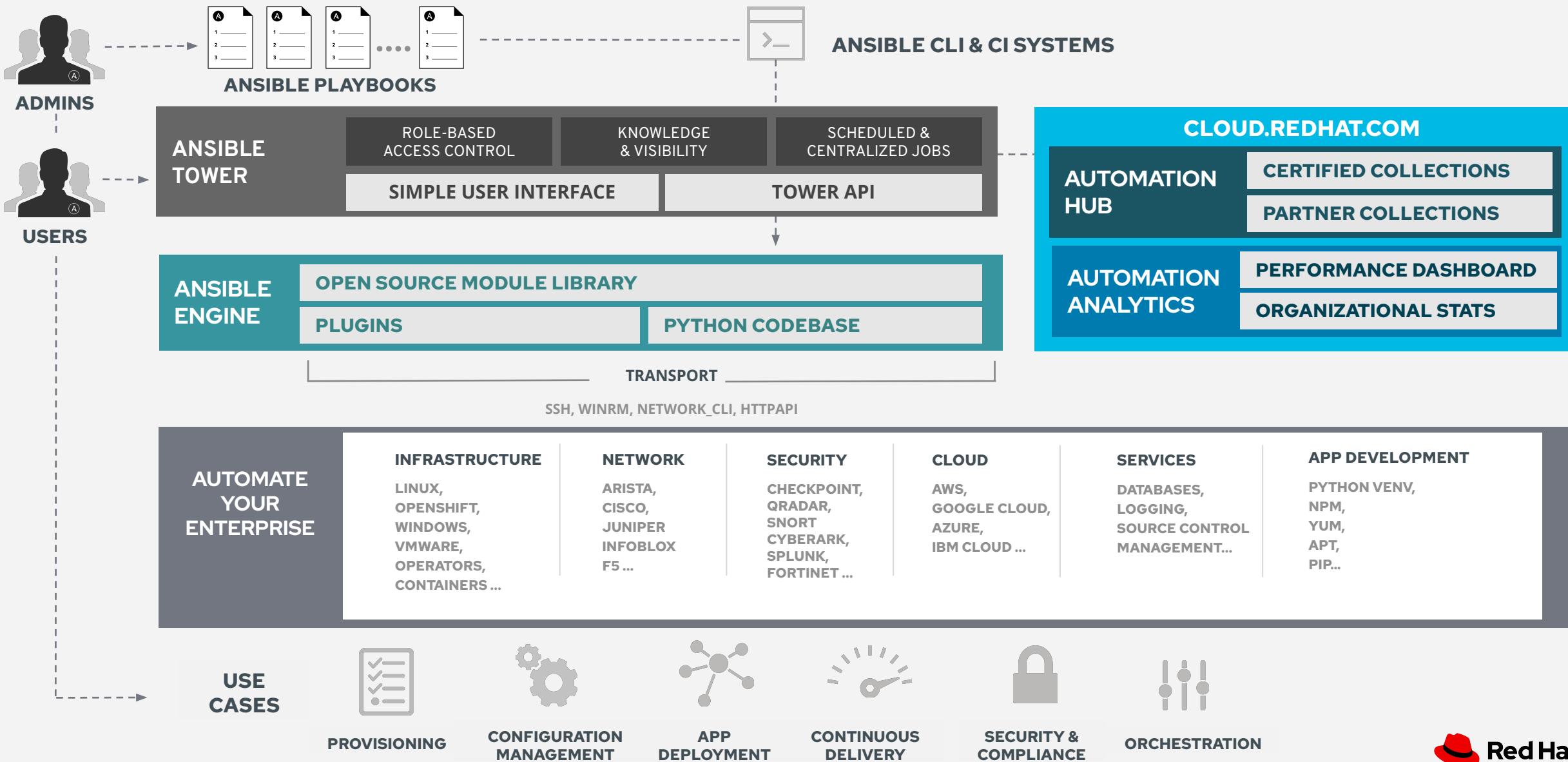
**Red Hat**  
Ansible Automation  
Platform

# Section 2.1

Topics Covered:

- Ansible Automation Hub
- Ansible Collections

# Ansible Automation Platform



# Ansible Automation Hub

## Trusted source

### Customer controlled

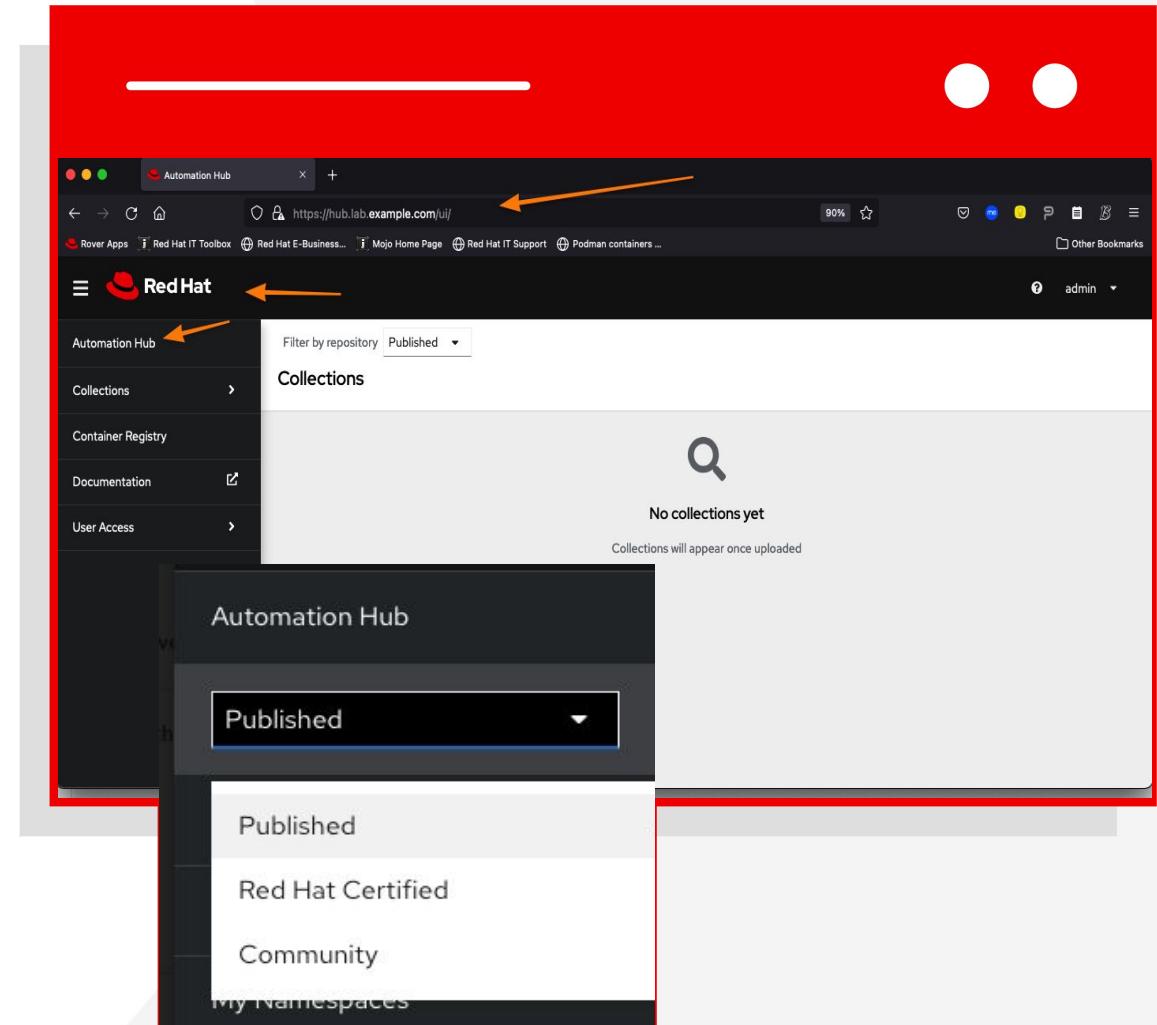
Deploying either on-prem or to a cloud, customers can run their own private instances of Automation Hub integrated into Red Hat Ansible Automation Platform.

### Private content

Manage the lifecycle and internal distribution of in-house Ansible content within.

### Customizable Content Catalog

Via sync from community (Galaxy) and supported (Automation Hub) sources, customers can supply internal users with approved content in one controlled location.



# Ansible Automation Hub

The screenshot shows the Ansible Automation Platform interface. At the top left is the Red Hat Hybrid Cloud Console logo. A navigation bar at the top has a dropdown labeled "All apps and services". On the right, there's a user profile for "Travis Michette". The main sidebar on the left is titled "Ansible Automation Platform" and includes links for Overview, Automation Hub (which is expanded to show Collections, Partners, and Repo Management), Automation Services (Catalog), Insights, Reports, Savings Planner, Automation Calculator, Organization Statistics, and Job Explorer. The "Collections" link in the Automation Hub section is highlighted with an orange arrow. The main content area is titled "Collections" and features a search bar with "Keywords" and "Filter by keywords" options. It displays three collections:

- cloud**  
Provided by Google Cloud  
The Google Cloud Platform collection.  
170 Modules 5 Roles 2 Plugins  
Tags: cloud, monitoring, gcsfuse, stackdriver, logging
- flashblade**  
Provided by Pure Storage  
Collection of modules to manage Pure Storage FlashBlades  
44 Modules 0 Roles 0 Plugins  
Tags: purestorage, flashblade, storage, object, nfs
- flasharray**  
Provided by Pure Storage  
Collection of modules to manage Pure Storage FlashArrays (including Cloud Block Store)  
51 Modules 0 Roles 0 Plugins  
Tags: purestorage, storage, flasharray, cloudblockstore

On the far right of the content area, there's a purple "Feedback" button. At the bottom right, there's a circular icon with a blue outline and a number "1" inside, next to a keyhole icon. The footer features the Red Hat logo.

# Ansible Automation Hub Collections

The screenshot shows the Red Hat Hybrid Cloud Console interface. On the left, the Ansible Automation Platform sidebar is visible with the 'Automation Hub' section highlighted. Two orange arrows point from the top of the sidebar towards the 'satellite' collection name on the main page. The main content area displays the 'satellite' collection details, including its version (v3.0.0), documentation links, and an 'Info' section describing it as Ansible Modules to manage Satellite installations. It lists dependencies (foreman, katello, satellite), specifies a GPL-3.0-or-later license, and provides installation instructions via ansible-galaxy collection install redhat.satellite. A note states that installing collections with ansible-galaxy is only supported in ansible 2.9+. A 'Download tarball' link is also present. The 'Install Version' dropdown shows 3.0.0 released 23 days ago (latest). Requirements specify Ansible >=2.9. The collection is identified as the 'Red Hat Satellite Ansible Collection' with a description of Ansible modules for interacting with the Satellite API.

Ansible Automation Platform

Overview

Automation Hub

Collections

Partners

Repo Management

Connect to Hub

Automation Services Catalog

Insights

Reports

Savings Planner

Automation Calculator

Organization Statistics

All apps and services

Partners > redhat > satellite

satellite

v3.0.0

Details Documentation Contents Import log

Docs site Website Issue tracker Repo

Info

Ansible Modules to manage Satellite installations

foreman katello satellite

License GPL-3.0-or-later

Installation ansible-galaxy collection install redhat.satellite

Note: Installing collections with ansible-galaxy is only supported in ansible 2.9+

[Download tarball](#)

Install Version 3.0.0 released 23 days ago (latest)

Requires Ansible >=2.9

**Red Hat Satellite Ansible Collection**

Ansible modules for interacting with the Satellite API.

# Ansible Galaxy Collections

The screenshot shows the Ansible Galaxy Collections search interface. The left sidebar includes links for Home, Search (highlighted by an orange arrow), Community, My Content, and My Imports. The main search interface features a search bar, filters for Type and Collection/Role, and a results summary showing 1217 results. The results list includes the 'ansible\_collection\_template' collection by sindhuparvat... (with 261 downloads) and the 'cluster' collection (with 533 downloads). An orange arrow points to the 'Type: Collection' filter in the results header.

Home

Search

Community

My Content

My Imports

GALAXY

About Help Documentation tmichett

Search

Search for...

Type Filter by Collection or Role... Best Match

1217 Results Active filters: Deprecated: False Type: Collection

Clear All Filters

Collections 1217

 **ansible\_collection\_template**  
The ansible\_collection\_template provides access to structured data from show commands  
sindhuparvat...  networking

 **cluster**  
Current Version: 1.0.2 uploaded 2 years ago  
533 Downloads

# Collections

Simplified and consistent content delivery



## What are they?

Collections are a data structure containing automation content:

- ▶ Modules
- ▶ Playbooks
- ▶ Roles
- ▶ Plugins
- ▶ Docs
- ▶ Tests



# Ansible Collections - Why?

- Ansible 2.9 introduced the concept of collections and provided mapping for Ansible modules that were moved into a collection namespace.
- Collections allowed development of Ansible core components to be separated from module and plug-in development.
- Upstream Ansible unbundled modules from Ansible core code beginning with Ansible Base 2.10/2.11.
- Never versions of Ansible require collections to be installed in order for modules to be available for Ansible
- Ansible 2.9 provides a mapping to the Fully Qualified Collection Name (FQCN)
  - [https://github.com/ansible/ansible/blob/devel/lib/ansible/config/ansible\\_builtin\\_runtime.yml](https://github.com/ansible/ansible/blob/devel/lib/ansible/config/ansible_builtin_runtime.yml)
- Playbooks should be developed using the FQCNs when referring to modules in tasks.
  - AAP requires older playbooks to be refactored to a degree to conform to new modules and component names
- Collections must be installed for modules to be available for Ansible playbooks
  - **ansible-galaxy collection install -r collections/requirements.yml -p collections/**

# Collections and Changes to Ansible Modules

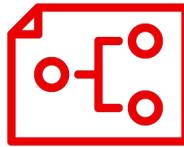
```
2 # GNU General Public License v3.0+ (see COPYING or https://www.gnu.org/licenses/gpl-3.0.txt)
3 plugin_routing:
4   connection:
5     # test entries
6     redirected_local:
7       redirect: ansible.builtin.local
8     buildah:
9       redirect: containers.podman.buildah
10    podman:
11      redirect: containers.podman.podman ←
12    aws_ssm:
13      redirect: community.aws.aws_ssm
14    chroot:
15      redirect: community.general.chroot
16    docker:
17      redirect: community.docker.docker
18    funcd:
19      redirect: community.general.funcd
20    iocage:
21      redirect: community.general.iocage
22    jail:
23      redirect: community.general.jail
24    kubectl:
25      redirect: kubernetes.core.kubectl
```

[https://github.com/ansible/ansible/blob/devel/lib/ansible/config/ansible\\_builtin\\_runtime.yml](https://github.com/ansible/ansible/blob/devel/lib/ansible/config/ansible_builtin_runtime.yml)



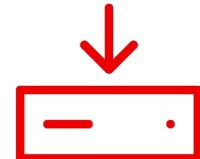
# Accessing collections

How to get them



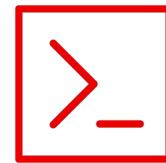
## Requirements file

Requirements file defines the required collections for a playbook



## Pull via controller

Automation controller pulls the collections from Automation Hub automatically



## Command line

CLI access is also possible via ansible-galaxy command

# Collections and Playbooks

## Older Playbooks

- **podman\_container** was a module that was able to be leveraged by the short module name in Ansible < 2.9.
- Ansible versions > 2.9 require that the FQCN be specified to that tasks can reference modules.
- It is possible to define collections at the top of a playbook similar to roles **(1)**.
  - This enables short **module** names to be used versus using the FQCN **(2)**.
- Not recommended as best practice.

1 [C]  
2 [C]

```
[user@ansible] $ cat playbook.yml

---
- name: Deploy HTTPD Server Demo
  hosts: localhost
  vars_files:
    - vars/registry_login.yml
  collections:
    - containers.podman

  tasks:

    ## Start and Run the HTTPD Container
    - name: Start the Webserver Container
      podman_container:
        name: Website_Demo
        image: quay.io/redhattraining/httpd-parent:2.4
        state: started
        restart: yes
        ports:
          - "7080:80"
        volume:
          - "/Webhosting:/var/www/html:z"
```

# Ansible Playbook with Collections

```
---  
- name: Playbook to Fully Setup and Configure a  
  Webserver  
  hosts: servera  
  tasks:  
    - name: Install Packages for Webserver  
      yum:  
        name:  
          - httpd  
          - firewalld  
        state: latest  
  
    - name: Create Content for Webserver  
      copy:  
        content: "I'm an awesome webserver\n"  
        dest: /var/www/html/index.html  
  
    - name: Firewall is Enabled  
      systemd:  
        name: firewalld  
        state: started  
        enabled: true
```

```
    - name: HTTP Service is Open on Firewall  
      ansible.posix.firewalld: (1)  
        service: http  
        state: enabled  
        permanent: true  
        immediate: yes  
  
    - name: httpd is started  
      systemd:  
        name: httpd  
        state: started  
        enabled: true
```

- **firewalld** was a module that was able to be leveraged by the short module name in Ansible <= 2.9, moved to the **Posix** collection using FQCN **(1)** above.



# **Red Hat** Ansible Automation Platform

## Demo Time

Ansible Automation Platform 1.2 - Ansible Collections to Deploy Webserver



**Red Hat**



**Red Hat**  
Ansible Automation  
Platform

# Break Time





**Red Hat**  
Ansible Automation  
Platform

# Section 3

# Ansible Automation

# Platform 2.x

*Future*

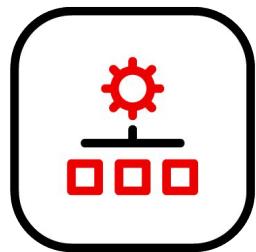
# New in Ansible Automation Platform 2.X

What changes?



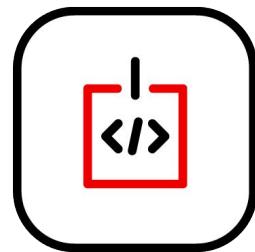
## Updated Private Automation Hub

Hosting of private content,  
container registry



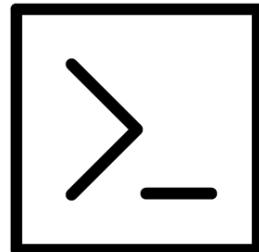
## Automation controller

Replaced Ansible Tower



## Automation execution environments

Replaced Ansible Engine



## `ansible-builder` and `ansible-navigator`

New tools for enterprise  
automation developers



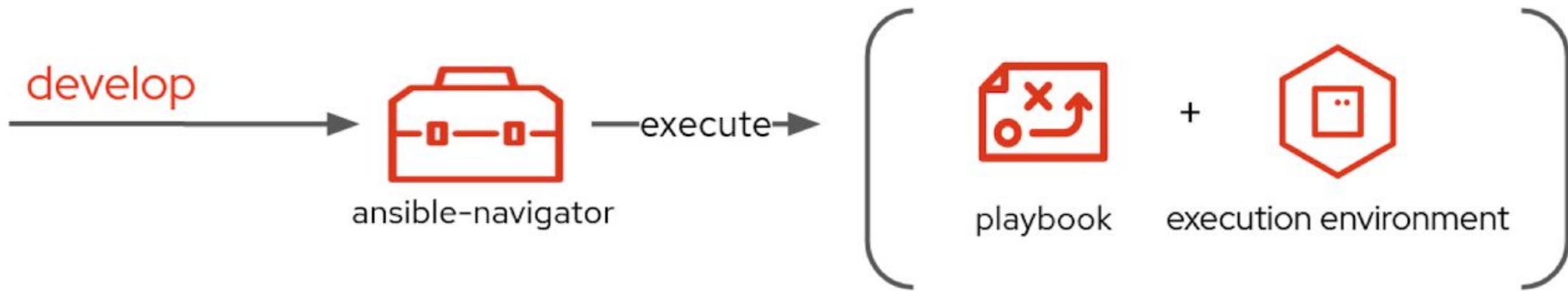
**Red Hat**  
Ansible Automation  
Platform

# Section 3.1

Topics Covered:

- Introduction to AAP 2.x Components
  - Ansible Content Navigator
  - Ansible Execution Environments

# Ansible Content Navigator



- ✓ Supported tooling
- ✓ Portable
- ✓ Scalable

# Ansible Content Navigator

| Ansible Command                | Automation Content Navigator Subcommands |
|--------------------------------|--|
| <code>ansible-config</code>    | <code>ansible-navigator config</code>    |
| <code>ansible-doc</code>       | <code>ansible-navigator doc</code>       |
| <code>ansible-inventory</code> | <code>ansible-navigator inventory</code> |
| <code>ansible-playbook</code>  | <code>ansible-navigator run</code>       |

```
# Running Navigator Interactively  
[user@ansible] $ ansible-navigator run Playbook.yml -m interactive
```

```
# Running Navigator Non-Interactively (Similar to ansible-playbook output)  
[user@ansible] $ ansible-navigator run Playbook.yml -m stdout
```

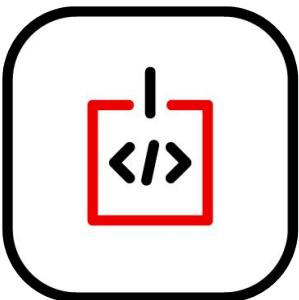
# ansible-navigator.yml

```
---  
ansible-navigator:  
  execution-environment: (1)  
  enabled: true  
  environment-variables:  
    set:  
      ANSIBLE_CONFIG: ansible.cfg (2)  
  image: hub.lab.example.com/ee-29-rhel8:latest (3)  
  logging:  
    level: critical  
    mode: stdout (4)
```

- **Execution Environment** - Configures Ansible Navigator to use an Execution Environment (EE). (1)
- Specifies where Ansible Navigator and the Ansible EE will receive Ansible configuration settings. (2)
  - Provides **ansible.cfg** file for the container runtime environment
- Specifies Ansible EE to use for Ansible Navigator. (3)
  - Defines container image and registry to be used for Ansible Navigator
- Specified Mode, in this case, we are using **STDOUT** so that the output will look like it does with the **ansible-playbook** command. (4)

# Automation Execution Environments

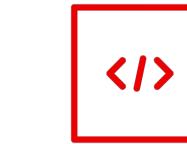
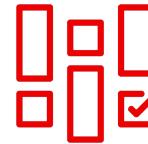
Components needed for automation, packaged in a cloud-native way



Execution  
Environments



Collections



Libraries



Ansible Core

# Ansible Execution Environments

## EE-29-RHEL8:LATEST (PRIMARY)

```
0 | Image information  
1 | General information  
2 | Ansible version and collections  
3 | Python packages  
4 | Operating system packages  
5 | Everything
```

## DESCRIPTION

```
Information collected from image inspection  
OS and python version information  
Information about ansible and ansible collections  
Information about python and python packages  
Information about operating system packages  
All image information
```

## EE-29-RHEL8:LATEST (PRIMARY) (OS AND PYTHON VERSION INFORMATION)

```
0 | ---  
1 | friendly:  
2 |   details: |-  
3 |     Red Hat Enterprise Linux release 8.5 (Ootpa)  
4 | os:
```

## EE-29-RHEL8:LATEST (PRIMARY) (INFORMATION ABOUT ANSIBLE AND ANSIBLE COLLECTIONS)

```
0 | ---  
1 | ansible:  
2 |   collections:  
3 |     details: {}  
4 |   errors:  
5 |     - |-  
6 |       usage: ansible-galaxy collection [-h] COLLECTION_ACTION ...  
7 |       ansible-galaxy collection: error: argument COLLECTION_ACTION: invalid choice:  
8 |   version:  
9 |     details: .9.2
```



# Ansible Execution Environments - SSH Keys

- **Execution Environment** - Leverages containers to run Ansible Playbooks

- **Contains**

- **Contains**
  - Ansible Core
  - Ansible Collections
  - Python Environment

- **Requires**

- **Requires**
  - Configuration Files
  - Inventory
  - SSH
    - SSH Keys must be provided through the SSH-Agent service

```
[student@workstation ~]$ eval $(ssh-agent)
```

```
[student@workstation ~]$ ssh-add ~/.ssh/lab_rsa
```



# **Red Hat**

## Ansible Automation Platform

## Demo Time

Ansible - Deploy Webserver with Ansible Content Navigator

Ansible - Ansible Content Navigator - Interactive Mode



**Red Hat**



**Red Hat**  
Ansible Automation  
Platform

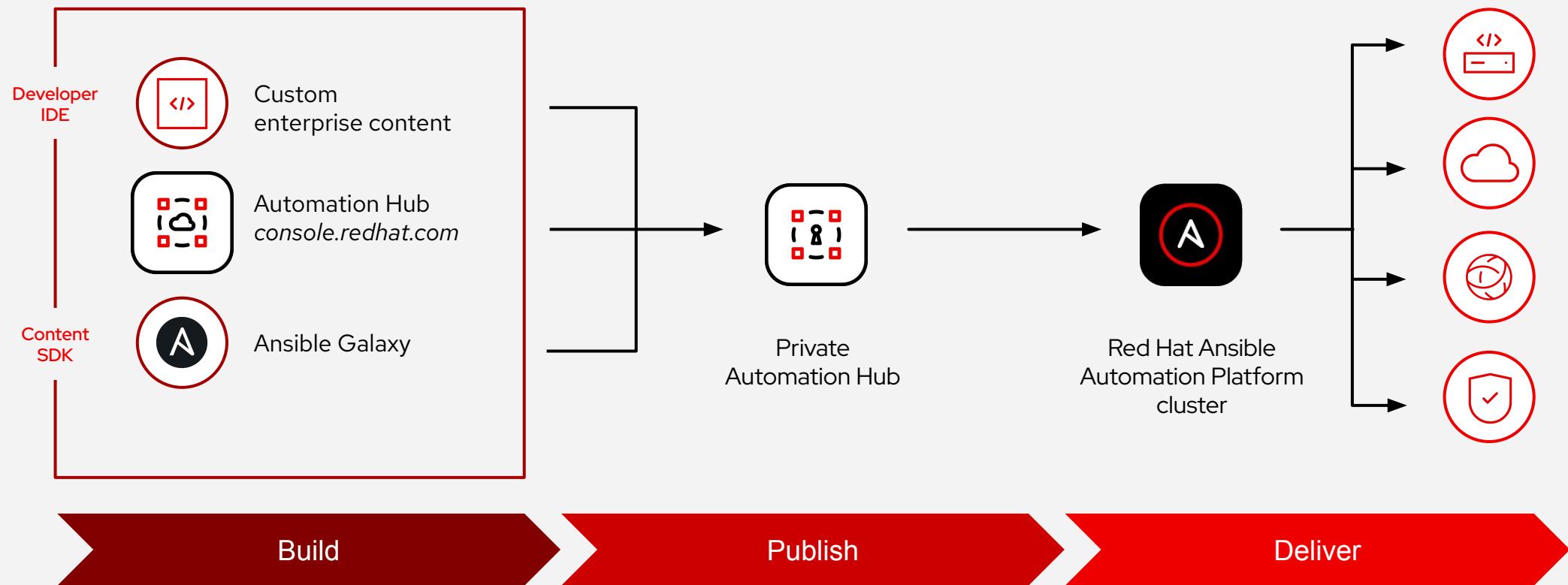
# Section 3.2

Topics Covered:

- Introduction to AAP 2.x - Ansible Automation Hub
  - Private Automation Hub
  - Custom Execution Environments

# Private Automation Hub

## Value of Private Automation Hub



# Automation Hub - Collections

The screenshot shows the Red Hat Automation Hub interface. The left sidebar has a 'Collections' section with 'Collections' selected, indicated by an orange arrow. A 'Filter by repository' dropdown is set to 'Red Hat Certified', also indicated by an orange arrow. The main area displays a grid of 10 collections, each with a 'Certified' badge:

| Certified Collection  | Description                     | Modules                        | Roles                           | Plugins                         |                                 |                                |
|---|---------------------------------|--------------------------------|---------------------------------|---------------------------------|---------------------------------|--------------------------------|
| <b>network</b><br>Provided by ansible<br>Ansible Network meta Collection to install all network             | 0 Modules, 1 Role, 0 Plugins    | 4 Modules, 0 Roles, 2 Plugins  | 11 Modules, 0 Roles, 11 Plugins | 78 Modules, 0 Roles, 3 Plugins  | 65 Modules, 11 Roles, 3 Plugins | 39 Modules, 0 Roles, 3 Plugins |
| <b>openshift</b><br>Provided by redhat<br>OpenShift Collection for Ansible.                                 |                                 |                                |                                 |                                 |                                 |                                |
| <b>posix</b><br>Provided by ansible<br>Ansible Collection targeting POSIX and POSIX-ish platforms.          |                                 |                                |                                 |                                 |                                 |                                |
| <b>controller</b><br>Provided by ansible<br>Ansible content that interacts with the AWX or Automation Pl... |                                 |                                |                                 |                                 |                                 |                                |
| <b>satellite</b><br>Provided by redhat<br>Ansible Modules to manage Satellite installations                 |                                 |                                |                                 |                                 |                                 |                                |
| <b>tower</b><br>Provided by ansible<br>Ansible content that interacts with the AWX or Ansible Tower...      |                                 |                                |                                 |                                 |                                 |                                |
| <b>rhel_system_roles</b><br>Provided by redhat<br>Red Hat Enterprise Linux System Roles Ansible Collection  | 12 Modules, 25 Roles, 0 Plugins | 4 Modules, 0 Roles, 41 Plugins | 56 Modules, 11 Roles, 4 Plugins | 26 Modules, 0 Roles, 36 Plugins |                                 |                                |
| <b>utils</b><br>Provided by ansible<br>Ansible Collection with utilities to ease the management, ma...      |                                 |                                |                                 |                                 |                                 |                                |
| <b>rhv</b><br>Provided by redhat<br>The oVirt Ansible Collection.   |                                 |                                |                                 |                                 |                                 |                                |
| <b>netcommon</b><br>Provided by ansible<br>Ansible Collection with common content to help automate the ...  |                                 |                                |                                 |                                 |                                 |                                |

# Automation Hub - Execution Environments

Red Hat

Automation Hub

Collections

Repository Management

API Token

Approval

Container Registry

Documentation

User Access

Container Registry

Container Registry

Container repository name

Filter by container repos...

Push container images

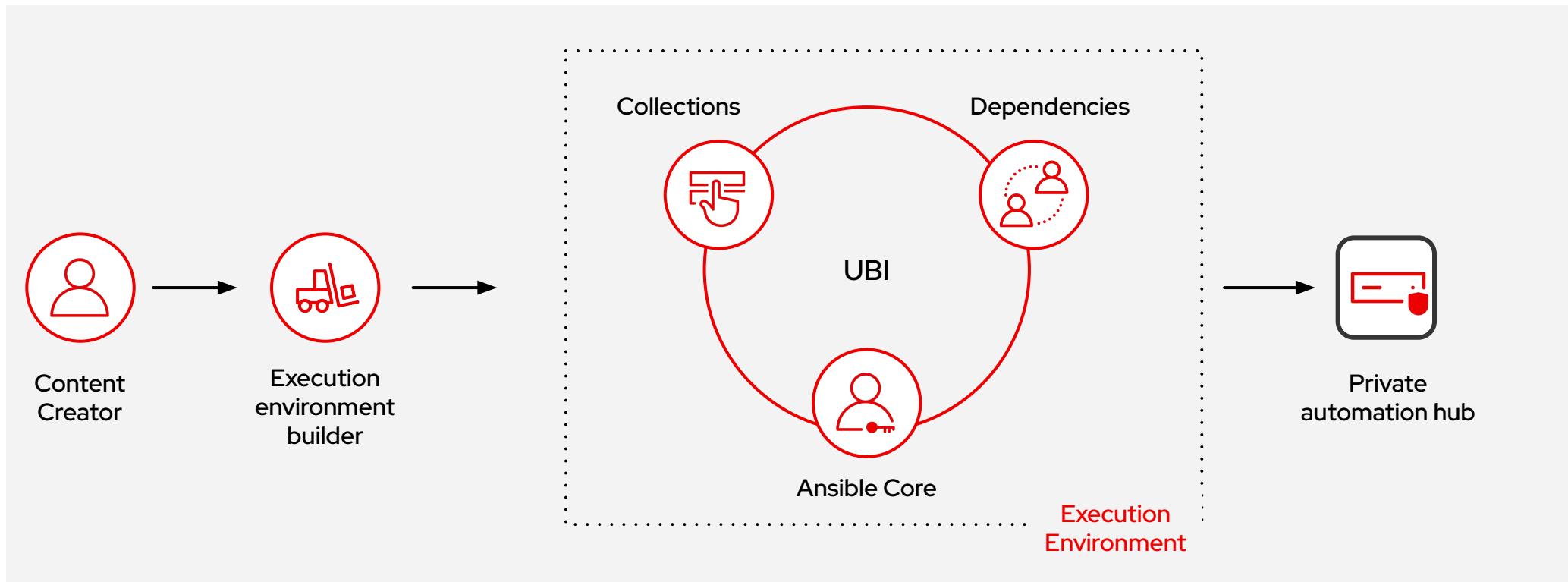
1 - 4 of 4

| Container repository name | Description | Created      | Last modified |
|---------------------------|-------------|--------------|---------------|
| ansible-builder-rhel8     |             | 2 months ago | 2 months ago  |
| ee-29-rhel8               |             | 2 months ago | 2 months ago  |
| ee-minimal-rhel8          |             | 2 months ago | 2 months ago  |
| ee-supported-rhel8        |             | 2 months ago | 2 months ago  |

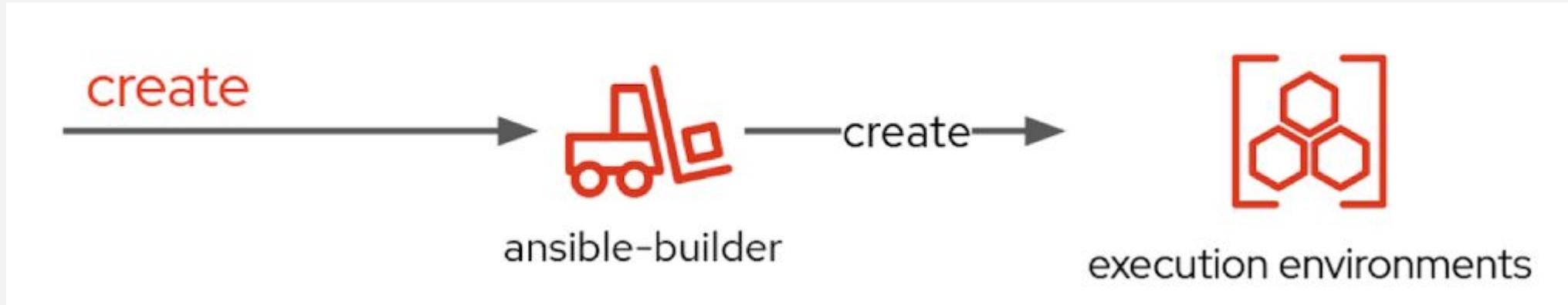
1 - 4 of 4

# Build, create, publish

Development cycle of an automation execution environment



# Ansible Execution Environments - Building/Customizing



```
# Running ansible-builder to Create Structure  
[user@ansible] $ ansible-builder create
```

```
# Running ansible-builder to Build Execution Environment  
[user@ansible] $ ansible-builder build -t ee-motd-minimal-demo:1.0
```

# Ansible Execution Environments - Building/Customizing

## execution-environment.yml (1)

```
---  
version: 1  
Build_arg_defaults:  
  EE_BASE_IMAGE: 'hub.lab.example.com/ee-minimal-rhel8:2.0' (1a)  
  EE_BUILDER_IMAGE: 'hub.lab.example.com/ansible-builder-rhel8:2.0' (1b)  
dependencies:  
  galaxy: requirements.yml (1c)(2)  
  python: requirements.txt (1d)(3)  
  system: bindep.txt (1e)(4)
```

## requirements.yml (1c)(2)

```
---  
collections:  
  - name: /build/exercise.motd.tar.gz (2a)  
    type: file
```

## requirements.txt (3)

```
# Python dependencies  
funmotd (3a)
```

## bindep.txt (4)

```
# System-level dependencies  
hostname (4a)
```

1. **execution-environment.yml** - Defines parameters and definitions for building the execution environment (EE) including the base image, and builder image along with all Ansible dependencies.
  - a. Defines base container image to be used for creating the EE
  - b. Defines the builder image to be used for the EE
  - c. Points to file containing the Collections and Roles to be installed and included in the EE
  - d. Points to file containing the required Python components to be installed and included in the EE
  - e. Points to file containing the system applications to be installed in the EE
2. **requirements.yml** - Defines the collections and roles to be used as part of the Ansible Execution Environment.
  - a. Listing of collections to be installed in the EE
3. **requirements.txt** - Defines the python dependencies and requirements needed by the Ansible Execution Environment and the included Ansible Collections.
  - a. List of Python tools to be installed in the EE
4. **bindep.txt** - Defines the system packages needed in the Ansible Execution Environment to run effectively and support the installed Ansible Collections and Python modules.
  - a. List of system packages needed installed in the EE

### IMPORTANT

Remember that Ansible Execution Environments are based on containers and container images. The **ansible-builder** command will build and create a new container image based on the **execution-environment.yml** file specifications.

```
# Building an Execution Environment with ansible-builder  
[student@workstation EE]$ ansible-builder build -t ee_aap_demo:latest
```



# Ansible Execution Environments - Publishing

```
# Using podman to Tag Image for Upload to Private Automation Hub  
[user@ansible] $ podman tag localhost/aap-demo:latest  
hub.lab.example.com/aap-demo:latest
```

```
# Using podman to Push Image to Private Automation Hub  
[user@ansible] $ podman push hub.lab.example.com/aap-demo:latest
```

```
# Using ansible-navigator to test image from Private Automation Hub  
[user@ansible] $ ansible-navigator run --pp always --eei  
hub.lab.example.com/aap-demo:latest -m stdout  
Custom_EE_Playbook.yml -b
```

# Ansible Execution Environments - Publishing Cont.

The screenshot shows the Red Hat Automation Hub interface, specifically the Container Registry section. The left sidebar includes options like Collections, Namespaces, Repository Management, API Token, Approval, and Container Registry, with an orange arrow pointing to the Container Registry item. The main area displays a table of container repositories with columns for Name, Description, Created, and Last modified. The table shows five entries:

| Container repository name | Description | Created       | Last modified |
|---------------------------|-------------|---------------|---------------|
| aap-demo                  |             | 3 minutes ago | 3 minutes ago |
| ansible-builder-rhel8     |             | 2 months ago  | 2 months ago  |
| ee-29-rhel8               |             | 2 months ago  | 2 months ago  |
| ee-minimal-rhel8          |             | 2 months ago  | 2 months ago  |
| ee-supported-rhel8        |             | 2 months ago  | 2 months ago  |



# **Red Hat** Ansible Automation Platform

## Demo Time

Ansible Automation Platform - Create Custom Execution Environment (EE)

Ansible Automation Platform - Run a Playbook with Custom Execution Environment

Ansible Automation Platform - Publish EE to Private Automation Hub



**Red Hat**



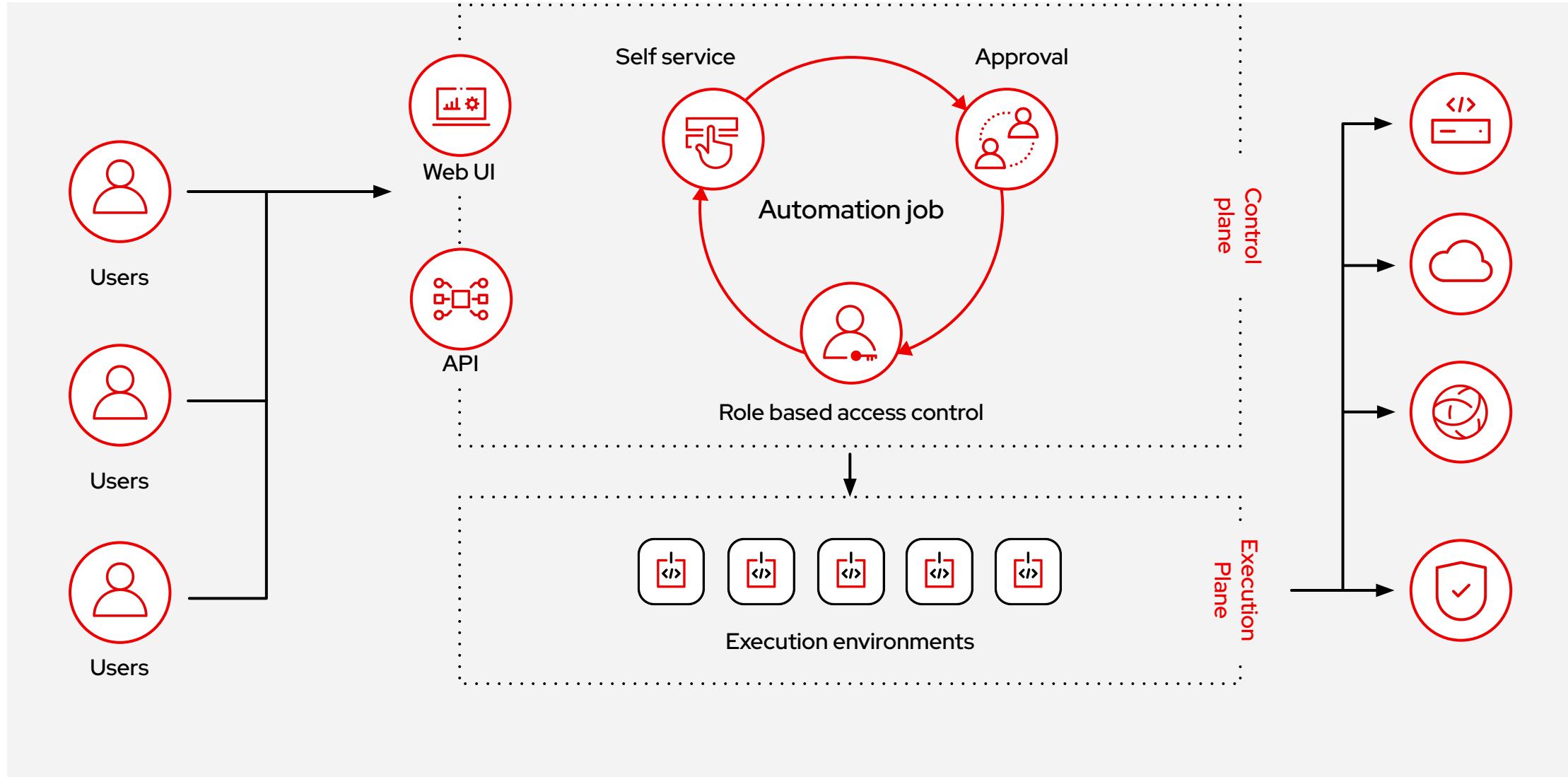
**Red Hat**  
Ansible Automation  
Platform

# Section 3.3

Topics Covered:

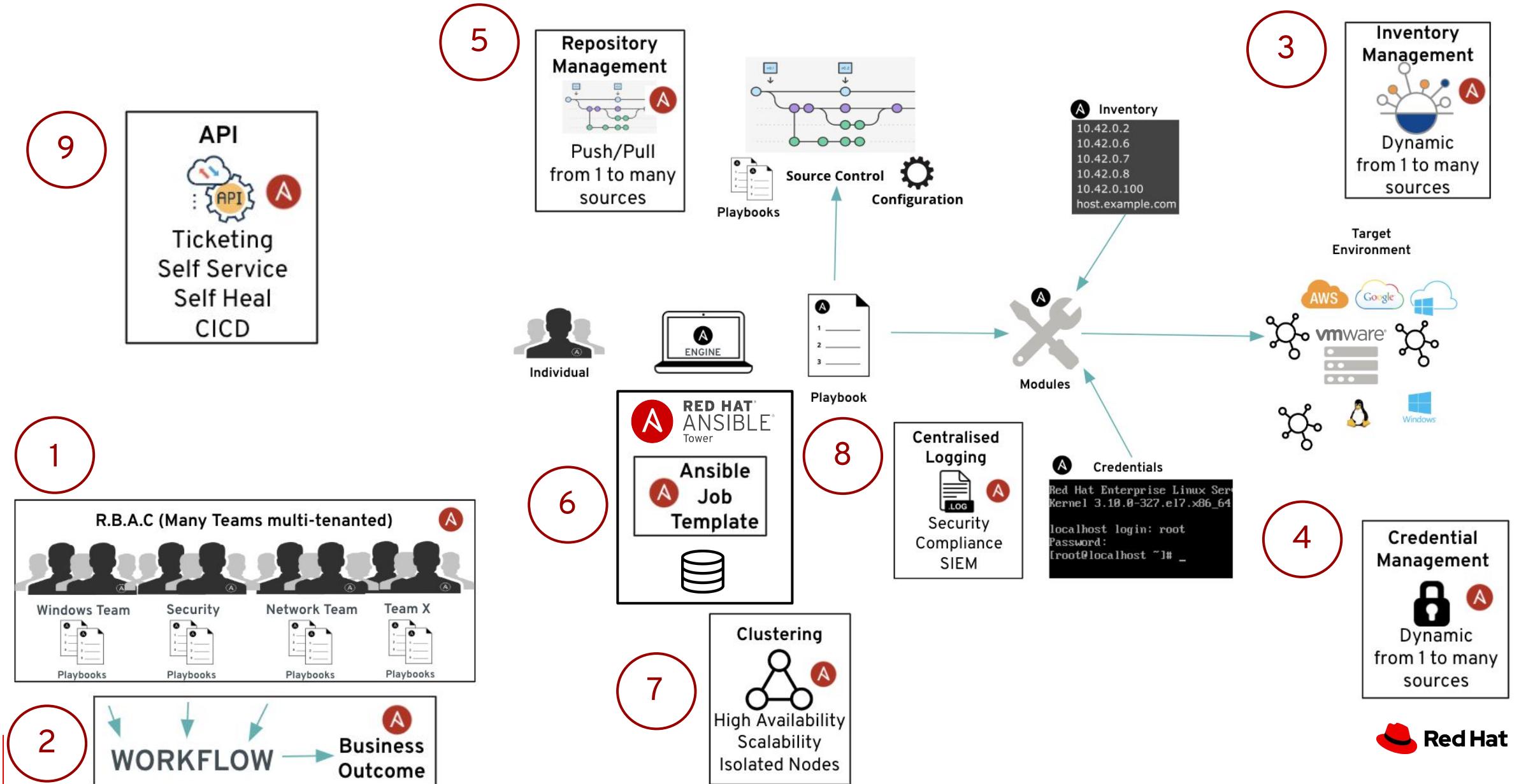
- Introduction to AAP 2.x - Ansible Controller
  - Organizations, Teams, and RBAC
  - Inventories and Credentials
  - Projects and Job Templates
  - Workflows

# Ansible Controller



# How Ansible Works - Ansible Controller

CONFIDENTIAL Designator



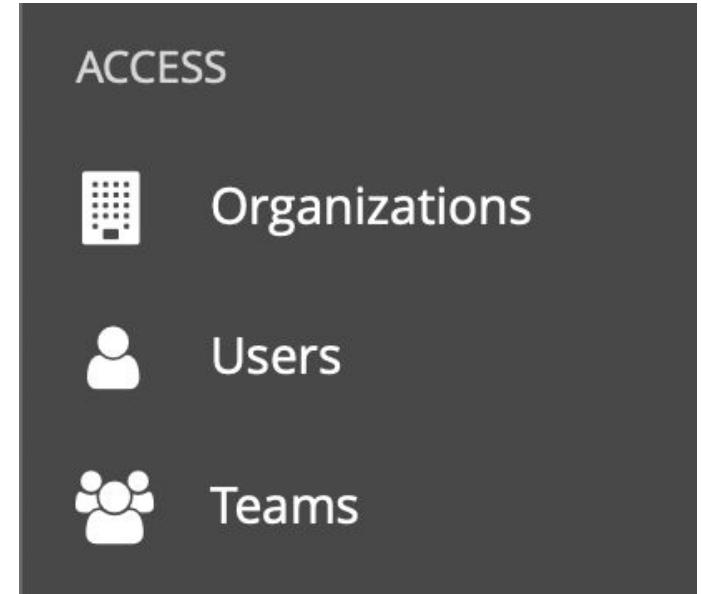
# Role Based Access Control (RBAC)

Role-Based Access Controls (RBAC) are built into Ansible Tower and allow administrators to delegate access to inventories, organizations, and more. These controls allow Ansible Tower to help you increase security and streamline management of your Ansible automation.



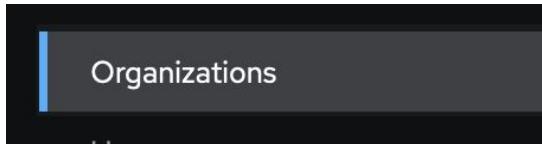
# User Management

- An **Organization** is a logical collection of users, teams, projects, inventories and more. All entities belong to an organization.
- A **User** is an account to access Ansible Tower and its services given the permissions granted to it.
- **Teams** provide a means to implement role-based access control schemes and delegate responsibilities across organizations.



# Viewing Organizations

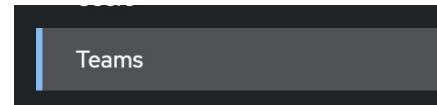
Clicking on the **Organizations** button will open up the Organizations window



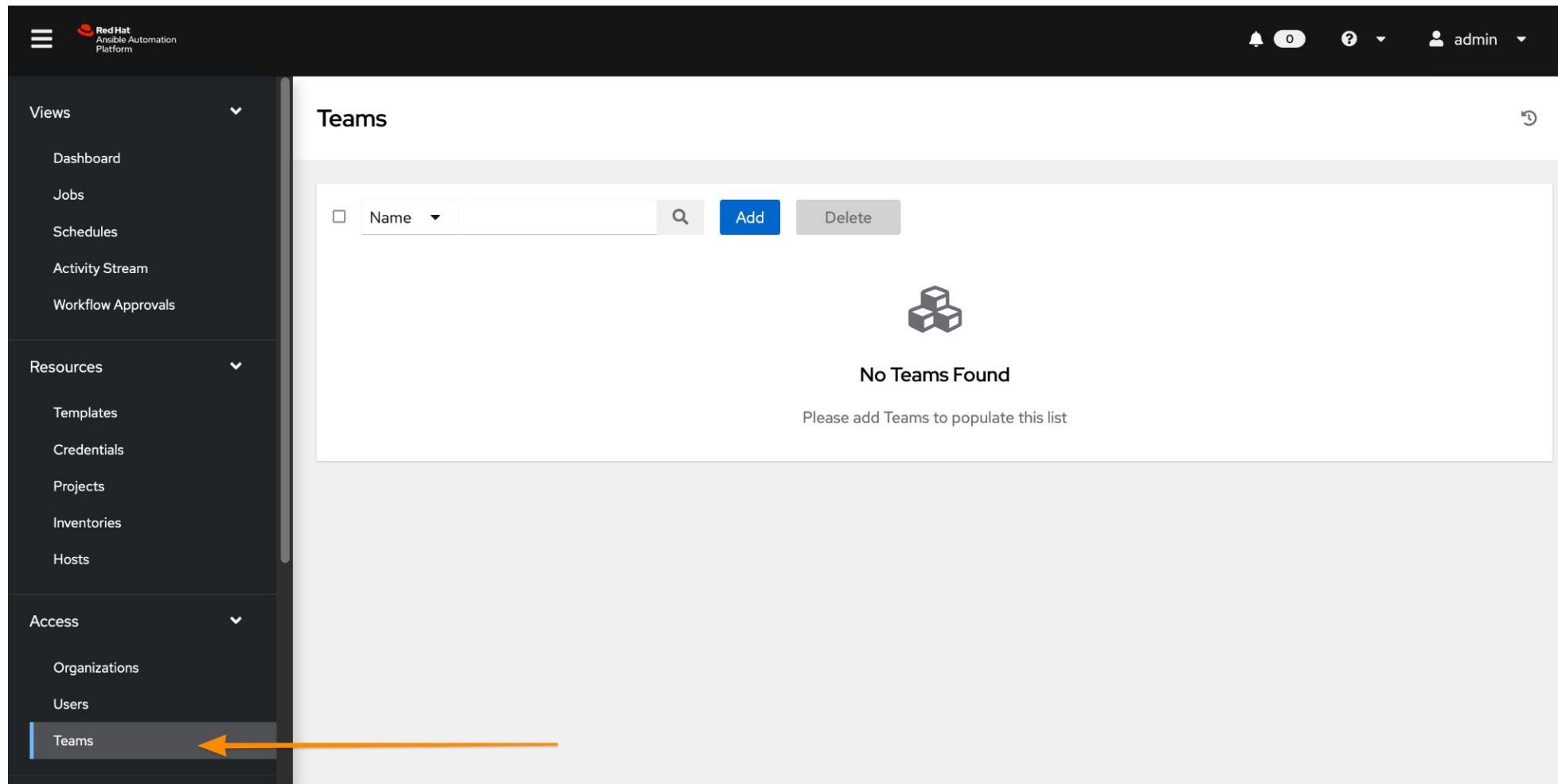
A screenshot of the 'Organizations' window. The window has a light grey header with the title 'Organizations'. Below the header is a search bar with a dropdown menu set to 'Name', a magnifying glass icon, and two buttons: 'Add' (blue) and 'Delete' (grey). To the right of the search bar is a page number '1-1 of 1' and navigation arrows. The main content area is a table with four columns: 'Name' (sorted by ascending order), 'Members', 'Teams', and 'Actions'. A single row is shown for the organization 'Default', which has 0 members and 0 teams. There is a blue pencil icon in the 'Actions' column for this row. At the bottom of the table is a page number '1-1 of 1 items' and navigation arrows. On the far left of the interface is a dark sidebar with three sections: 'Views', 'Resources', and 'Access'. The 'Access' section is expanded, showing 'Organizations' (which is highlighted with a blue background and has an orange arrow pointing to it from below), 'Users', and 'Teams'. The top of the interface features the Red Hat logo and the text 'Ansible Automation Platform'. The top right corner shows a notification bell with '0', a help icon, and a user profile for 'admin'.

# Viewing Teams

Clicking on the **Teams** button  
will open up the Teams window



in the left menu

A screenshot of the 'Teams' window. The title bar says 'Teams'. Below it is a search bar with a dropdown menu set to 'Name', a magnifying glass icon, and an 'Add' button. To the right of the search bar is a 'Delete' button. In the center of the window is a small icon of three stacked cubes. Below the icon, the text 'No Teams Found' is displayed, followed by the message 'Please add Teams to populate this list'. The left sidebar of the platform is visible on the left side of the window.

# Viewing Users

Clicking on the **Users** button in the left menu will open up the Users window

The screenshot shows the Red Hat Ansible Automation Platform web interface. The top navigation bar includes the Red Hat logo, a search bar, and user account information for 'admin'. The left sidebar has sections for 'Views' (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), 'Resources' (Templates, Credentials, Projects, Inventories, Hosts), and 'Access' (Organizations, **Users**, Teams). An orange arrow points to the 'Users' link in the Access section. The main content area is titled 'Users' and displays a table with two rows of user data. The columns are: Username, First Name, Last Name, Role, and Actions. The first user is 'admin' (System Administrator) and the second is 'travis' (Normal User).

| Username | First Name | Last Name | Role                 | Actions |
|----------|------------|-----------|----------------------|---------|
| admin    |            |           | System Administrator |         |
| travis   | Travis     | Michette  | Normal User          |         |



# **Red Hat**

## Ansible Automation Platform

### Demo Time

Creating an Organization with Users and Teams



**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

The screenshot shows the Red Hat Ansible Automation Platform interface on a tablet device. The top navigation bar includes the Red Hat logo, 'Ansible Automation Platform', a search bar, and user information for 'admin'. The left sidebar has a dark theme with white text, showing sections for 'Views' (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), 'Resources' (Templates, Credentials, Projects, Inventories, Hosts), 'Access' (Organizations, Users, Teams), and 'Administration' (Credential Types). The 'Inventories' section is currently selected and highlighted with a blue bar at the bottom of the sidebar. The main content area is titled 'Inventories' and displays a table with one row. The table columns are 'Name' (sorted by ascending Name), 'Status' (disabled), 'Type' (Inventory), 'Organization' (Default), and 'Actions' (edit and delete icons). The single row in the table is labeled 'Demo Inventory'.

| Name           | Status   | Type      | Organization | Actions |
|----------------|----------|-----------|--------------|---------|
| Demo Inventory | Disabled | Inventory | Default      |         |

# 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.

The screenshot shows the Ansible Tower web interface. The top navigation bar includes the Red Hat logo, 'Ansible Automation Platform', a search icon, a notification bell with 0 notifications, a help icon, and a user account for 'admin'. The left sidebar has sections for 'Views' (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), 'Resources' (Templates, Credentials, Projects, Inventories, Hosts), 'Access' (Organizations, Users, Teams), and 'Administration' (Credential Types). The main content area is titled 'Credentials' and displays a list of four entries:

| Name  | Type                                    | Actions |
|---|---|---------|
| Ansible Galaxy                                    | Ansible Galaxy/Automation Hub API Token |         |
| Default Execution Environment Registry Credential | Container Registry                      |         |
| Demo Credential                                   | Machine                                 |         |
| Private Hub Credential                            | Container Registry                      |         |

Pagination at the bottom indicates '1 - 4 of 4 items' and '1 of 1 page'.



# **Red Hat**

## Ansible Automation Platform

### Demo Time

Creating an Inventory and Credentials



**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.

The screenshot shows the Ansible Tower web interface on a laptop screen. The left sidebar has a dark theme with white text. It includes sections for Views (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), Resources (Templates, Credentials, Projects, Inventories, Hosts), Access (Organizations, Users, Teams), and Administration (Credential Types). The 'Projects' section is currently selected. The main content area is titled 'Projects' and displays a table with two rows:

| Name           | Status     | Type              | Revision | Actions |
|----------------|------------|-------------------|----------|---------|
| Demo Project   | Git        | Sync for revision |          |         |
| NYPD Webserver | Successful | Git               | d54594d  |         |

At the bottom of the table, it says '1 - 2 of 2 items' and '1 of 1 page'.

# 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

The screenshot shows the Ansible Tower web interface. On the left is a dark sidebar with navigation links: Views (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), Resources (Templates, Credentials, Projects, Inventories, Hosts), Access (Organizations, Users, Teams), and Administration (Credential Types). The main content area is titled 'Templates > NYPD Webserver Deploy Details'. The 'Details' tab is selected. The table displays the following information:

| Name            | NYPD Webserver Deploy                     | Job Type      | run                            | Organization          | NYPD                                     |
|-----------------|---|---------------|--------------------------------|-----------------------|--|
| Inventory       | NYPD Systems                              | Project       | NYPD Webserver                 | Execution Environment | Ansible Engine 2.9 execution environment |
| Playbook        | Future/NYPD/Website_Ansible_Past.yml      | Forks         | 0                              | Verbosity             | 0 (Normal)                               |
| Timeout         | 0   | Show Changes  | Off                            | Job Slicing           | 1  |
| Created         | 1/12/2022, 4:40:57 PM by admin            | Last Modified | 1/12/2022, 4:49:08 PM by admin |                       |  |
| Enabled Options | Privilege Escalation                      |               |                                |                       |  |
| Credentials     | SSH: NYPD Machine ...                     |               |                                |                       |  |
| Variables       | <a href="#">YAML</a> <a href="#">JSON</a> |               |                                |                       |  |



# **Red Hat**

## Ansible Automation Platform

### Demo Time

Creating a Project and Job Template



**Red Hat**

# Workflows

Recall that everything in Ansible Tower revolves around the concept of a Job Template. **Job Workflows** allow multiple Job Templates to be controlled, delegated and scaled for an organization.

Job workflows allow building Ansible pipelines to execute multiple job templates and other functions depending on if the running Job Template succeeds or fails.

A **Job Workflow** requires:

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

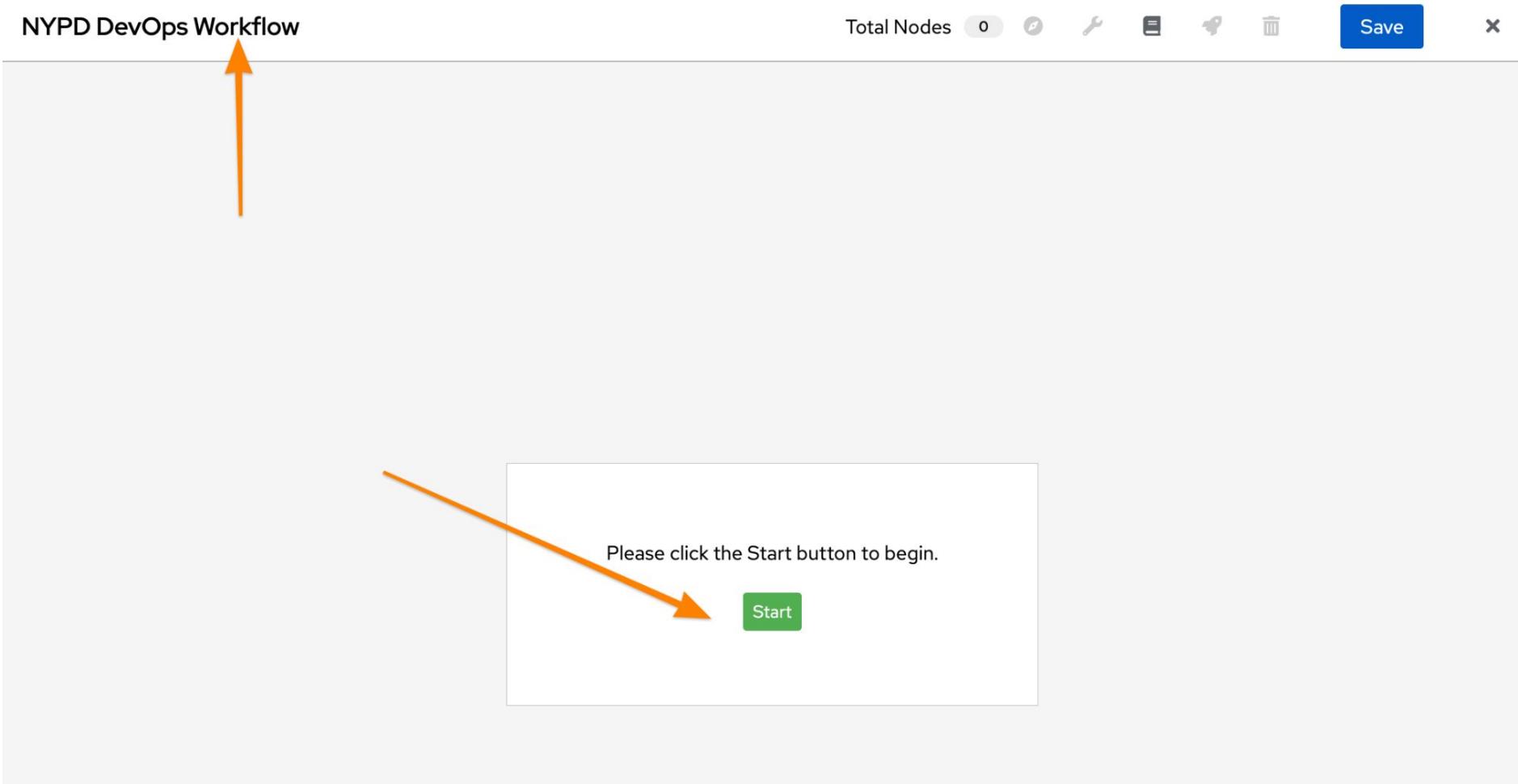
The screenshot shows the Ansible Tower web interface. The left sidebar has a dark theme with white text and includes sections for Views (Dashboard, Jobs, Schedules, Activity Stream, Workflow Approvals), Resources (Templates, Credentials, Projects, Inventories, Hosts), Access (Organizations, Users, Teams), and Administration (Credential Types). The main content area is titled 'Templates' and displays a table of five entries:

| Name                  | Type                  | Last Ran              | Actions |
|-----------------------|-----------------------|-----------------------|---------|
| Demo Job Template     | Job Template          |                       |         |
| NYPD DevOps Workflow  | Workflow Job Template | 1/13/2022, 1:29:40 PM |         |
| NYPD Dev Webserver    | Job Template          | 1/13/2022, 1:29:16 PM |         |
| NYPD Test Webserver   | Job Template          | 1/13/2022, 1:29:40 PM |         |
| NYPD Webserver Deploy | Job Template          | 1/12/2022, 4:51:27 PM |         |

At the bottom right of the table, it says '1 - 5 of 5 items' and 'of 1 page'.

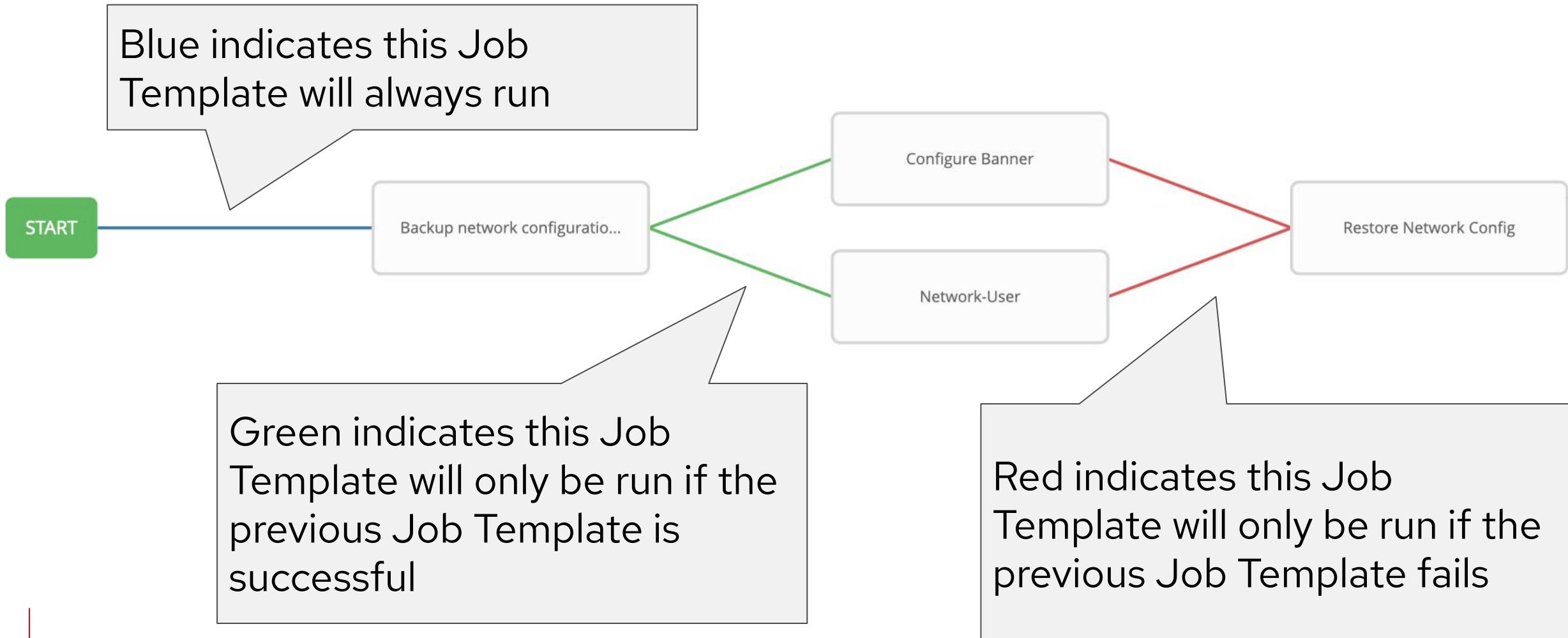
# Workflow Visualizer

The workflow visualizer will start as a blank canvas.



# Visualizing a Workflow

Workflows can branch out, or converge in.





# **Red Hat**

## Ansible Automation Platform

### Demo Time

Executing Multiple Playbooks and Projects with Workflows



**Red Hat**

# Section 4

# Ansible Automation



**Red Hat**  
Ansible Automation  
Platform

---

# Training at Red Hat

# Customer return on investment from training

## 365% 3-year ROI

—

IDC conducted a study to explore how Red Hat® training courses impacted the skills, performance, and productivity levels of customers. They found that training for impacted IT professionals and developers consistently increases both individual capability and the ultimate business value of the supported technology.

Other key findings include:

-  **44%**  
higher DevOps team productivity
-  **59%**  
faster to deploy new IT resources

-  **34%**  
more efficient IT infrastructure teams
-  **76%**  
faster to full productivity, new hires already trained

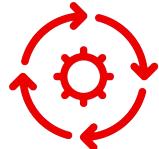
# Improve productivity with training in Ansible automation

Scale people, processes, and infrastructure



## Red Hat Ansible Automation Platform

A powerful foundation to build and operate automation across organizations. Prepare your teams with the right skills to make the most out of new technology investments.

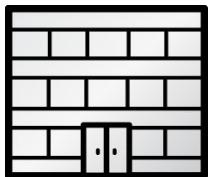


**59% faster**  
deployment of new IT resources

"Red Hat Training shows our DevOps team how to automate a repeatable task. They can write one playbook to execute a set of tasks that would have taken hours or days of time."

"With Red Hat Training it doesn't matter which engineer is engaged on a project. They are all using Ansible for automating tasks, allowing them collectively to be **five times as productive** ... This was not possible previously. As a result, they've definitely picked up the pace of productivity."

# WAYS TO TRAIN



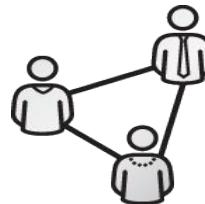
## Onsite Training

Private On-site training and exams delivered at your location or at one of our training centers



## Classroom Training

Training and test in a professional classroom environment led by Red Hat Certified Instructors



## Virtual Training

Live instructor-led online training with the same high-quality, hands-on labs you'd find in our classrooms



## Online Learning

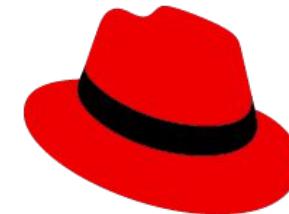
90 days of access to course content and up to 80 hours of hands on labs – all available online, at your pace, and your schedule.

# RED HAT LEARNING SUBSCRIPTION

A prescriptive, reliable, learning solution for rapid skills transformation on Red Hat technologies

## Simple, flexible, on-demand training

- 24x7 access globally, available offline
- Self-paced, unlimited access to Red Hat courses
- Access to content currently in development
- Updated content pushed as early releases
- Content spanning the entire Red Hat product portfolio
- Early access to completed chapters of courses



**Red Hat  
Learning  
Subscription**

# Red Hat Learning Subscription Evolution

Introducing a Premium subscription tier



STANDARD

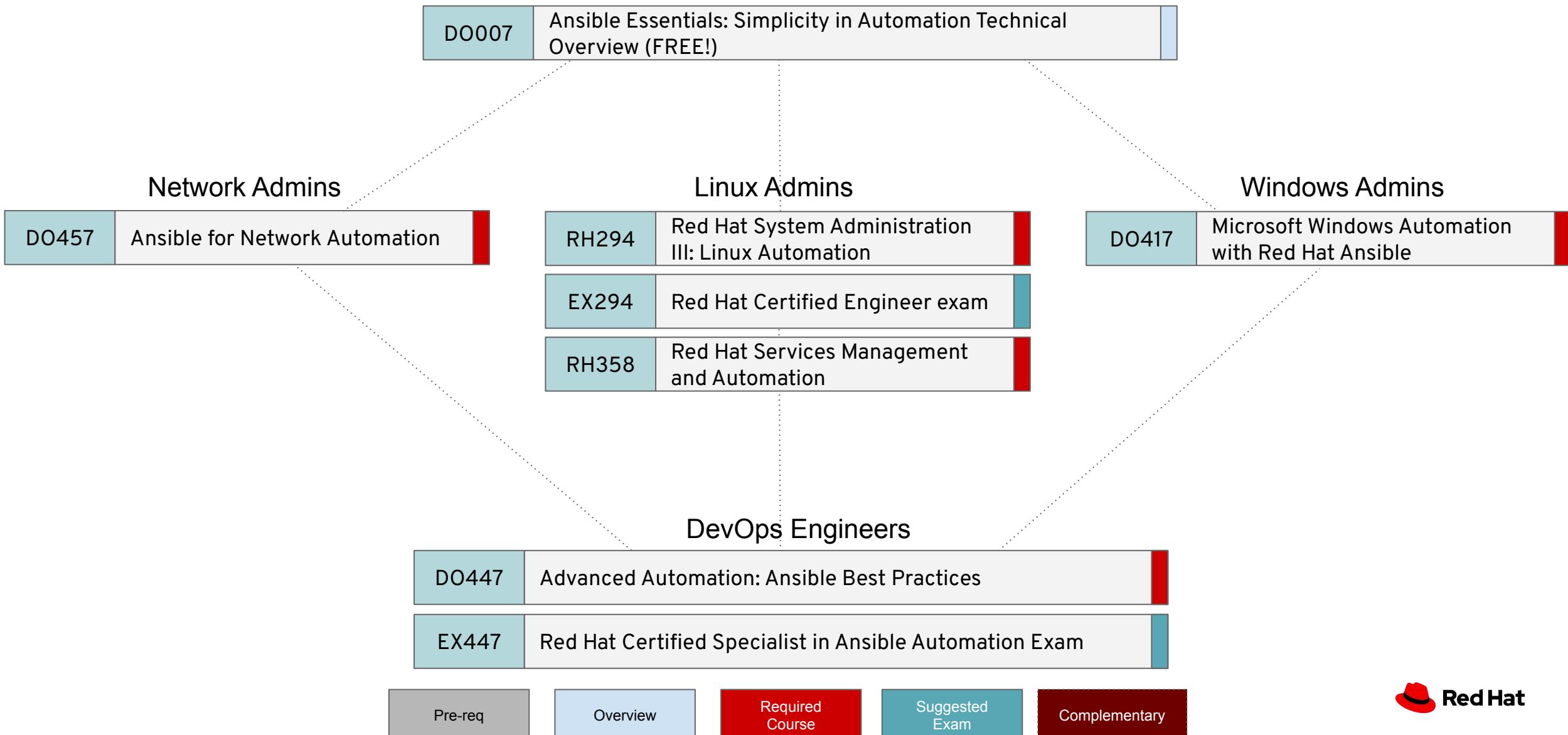


MODULARIZED VIRTUAL  
TRAINING

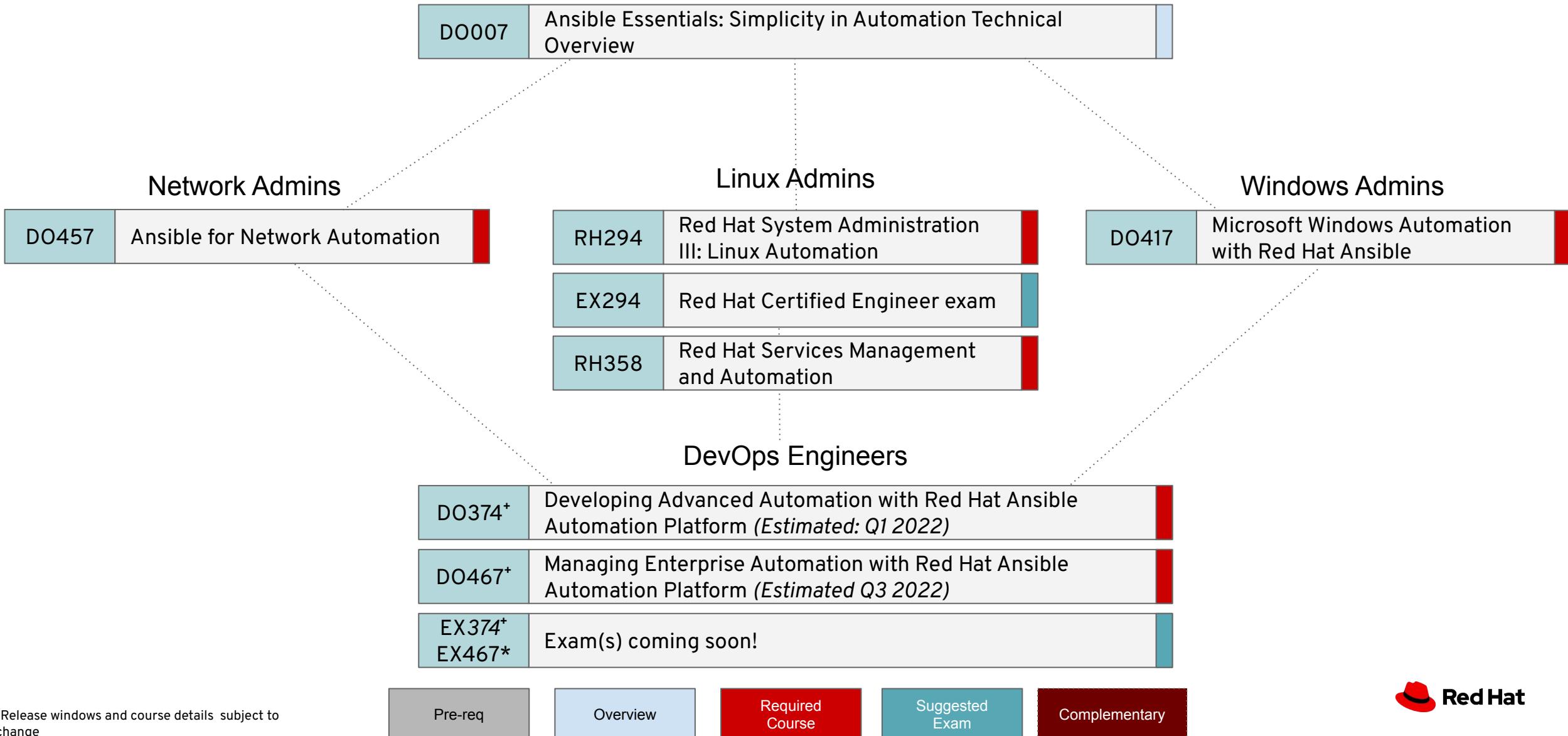


PREMIUM

# Ansible Curriculum (current as of January 1, 2022)



# Ansible Curriculum (Future as of Q2/Q3Y22)





**Red Hat**

carahsoft.<sup>®</sup>



**Thank you for attending!**

***Please contact us with questions.***

**Connor Murray**

**703-230-7583**

**877-RHAT-GOV (Team)**

**Connor.Murray@carahsoft.com**

**<https://carahtsled>**



Thank you for viewing this Red Hat virtual training! Carahsoft is the Master GSA and SLSA Dealer and Distributor for Red Hat Enterprise Open Source solutions available via GSA, SLSA, ITES-SW2, The Quilt and other contract vehicles.

To learn how to take the next step toward acquiring Red Hat's solutions, please check out the following resources and information:



For additional resources:

[carahtech.com/redhatsled](https://carahtech.com/redhatsled)



For additional Red Hat solutions:

[carahtech.com/RedHatPortfolio](https://carahtech.com/RedHatPortfolio)



To purchase, check out the contract vehicles available for procurement:

[carahtech.com/RedHatContracts](https://carahtech.com/RedHatContracts)



For upcoming events:

[carahtech.com/RedHatEvents](https://carahtech.com/RedHatEvents)



For additional Open Source solutions:

[carahtech.com/OpenSourceSolutions](https://carahtech.com/OpenSourceSolutions)



To set up a meeting:

[redhat@carahtech.com](mailto:redhat@carahtech.com) or 877-RHAT-GOV