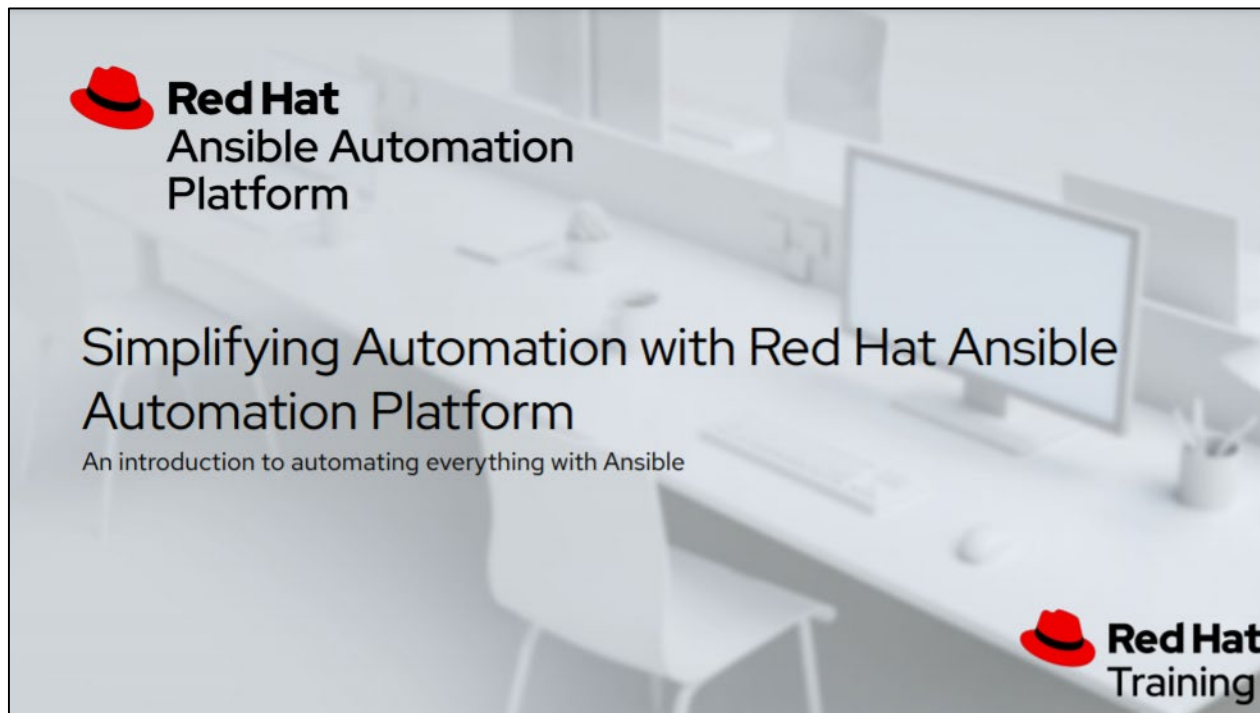




Simplifying Automation with Red Hat Ansible

Red Hat Training Workshop





Red Hat Ansible Automation Platform

Simplifying Automation with Red Hat Ansible Automation Platform

An introduction to automating everything with Ansible



Red Hat
Training

Housekeeping

- Timing
- Breaks
- Takeaways

What you will learn

- Introduction to Ansible Automation
- How it works
- Understanding modules, tasks & playbooks
- How to execute Ansible commands
- Using variables & templates
- Tower - where it fits in
- Basic usage of Tower
- Learn major Tower features: RBAC, workflows and so on

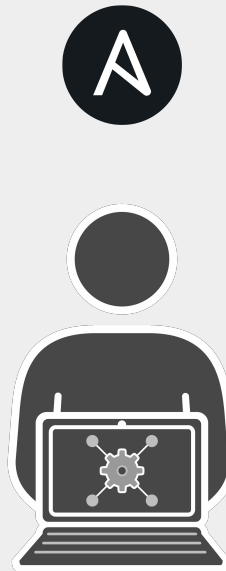
Introduction

Topics Covered:

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



Red Hat
Ansible Automation
Platform



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

Teams are automating...



Lines Of Business



Network



Security



Operations



Developers



Infrastructure

Ad-hoc Automation is happening in silos



Developers

→ Ansible used in silo



Security

→ DIY scripting automation



Infrastructure

→ Open source config management tool



Network

→ Proprietary vendor supplied automation

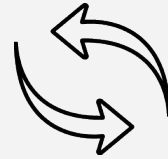
Is organic automation enough?

Why Ansible?



Simple

Human readable automation
No special coding skills needed
Tasks executed in order
Usable by every team
Get productive quickly



Powerful

App deployment
Configuration management
Workflow orchestration
Network automation
Orchestrate the app lifecycle



Agentless

Agentless architecture
Uses OpenSSH & WinRM
No agents to exploit or update
Get started immediately
More efficient & more secure

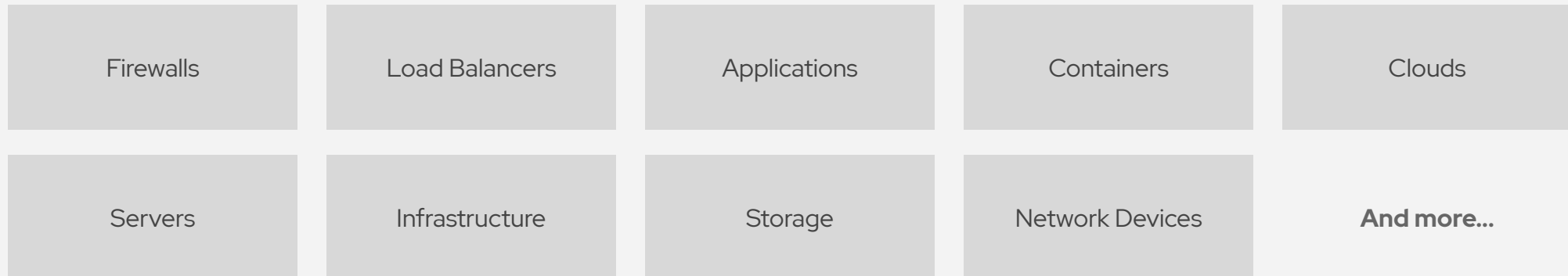
What can I do using Ansible?

Automate the deployment and management of your entire IT footprint.

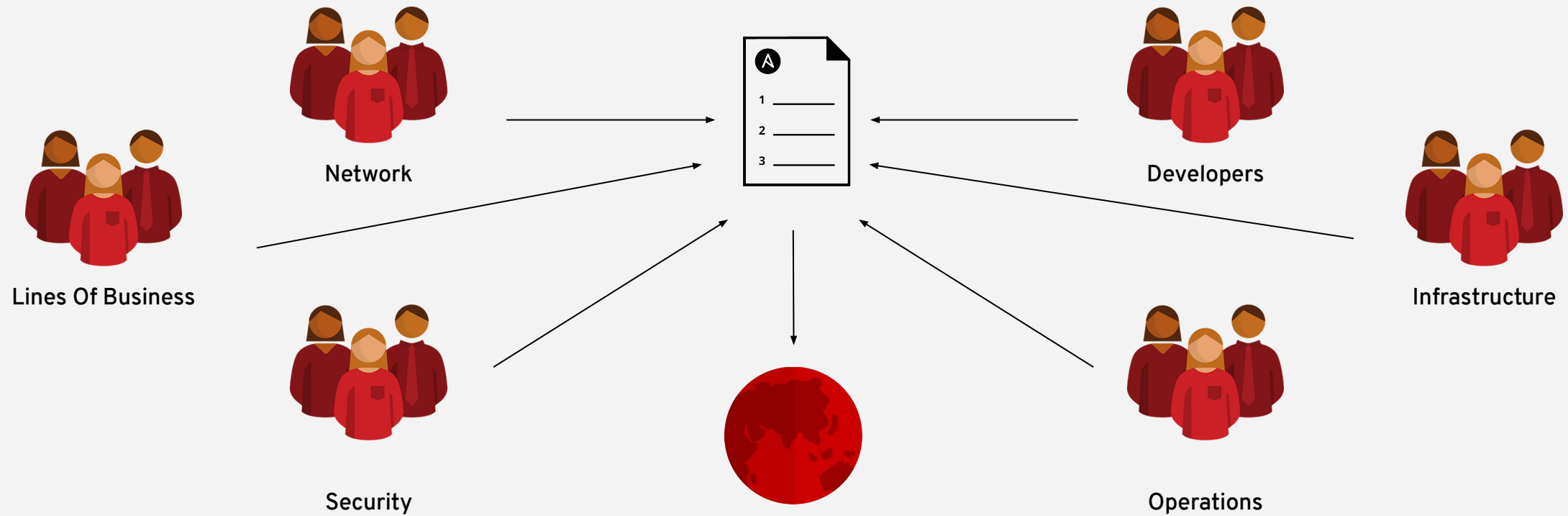
Do this...



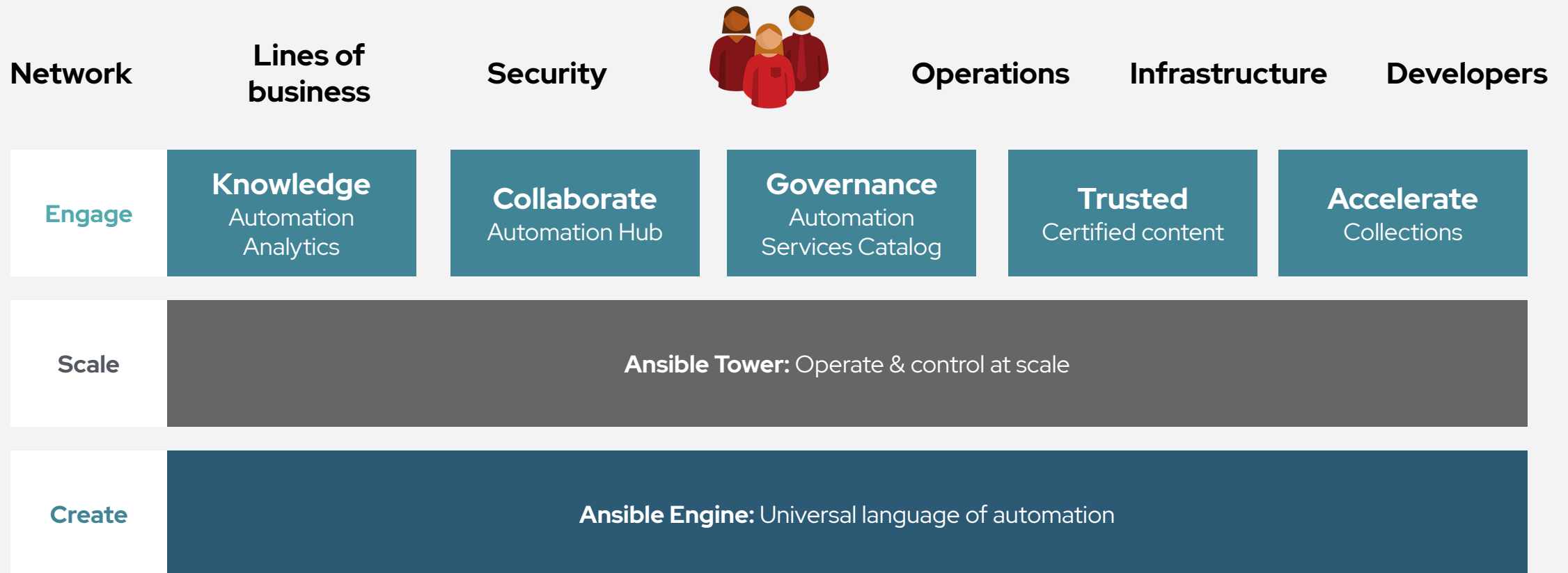
On these...



When automation crosses teams, you need an automation platform



Red Hat Ansible Automation Platform



Fueled by an open source community

Ansible automates technologies you use

Time to automate is measured in minutes

Cloud

AWS
Azure
Digital Ocean
Google
OpenStack
Rackspace
+more

Operating Systems

RHEL
Linux
Windows
+more

Virt & Container

Docker
VMware
RHV
OpenStack
OpenShift
+more

Storage

Netapp
Red Hat Storage
Infinidat
+more

Windows

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

Network

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

Security

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

Monitoring

Dynatrace
Datadog
LogicMonitor
New Relic
Sensu
+more

Devops

Jira
GitHub
Vagrant
Jenkins
Slack
+more

Section 1

Ansible Engine



Red Hat
Ansible Automation
Platform

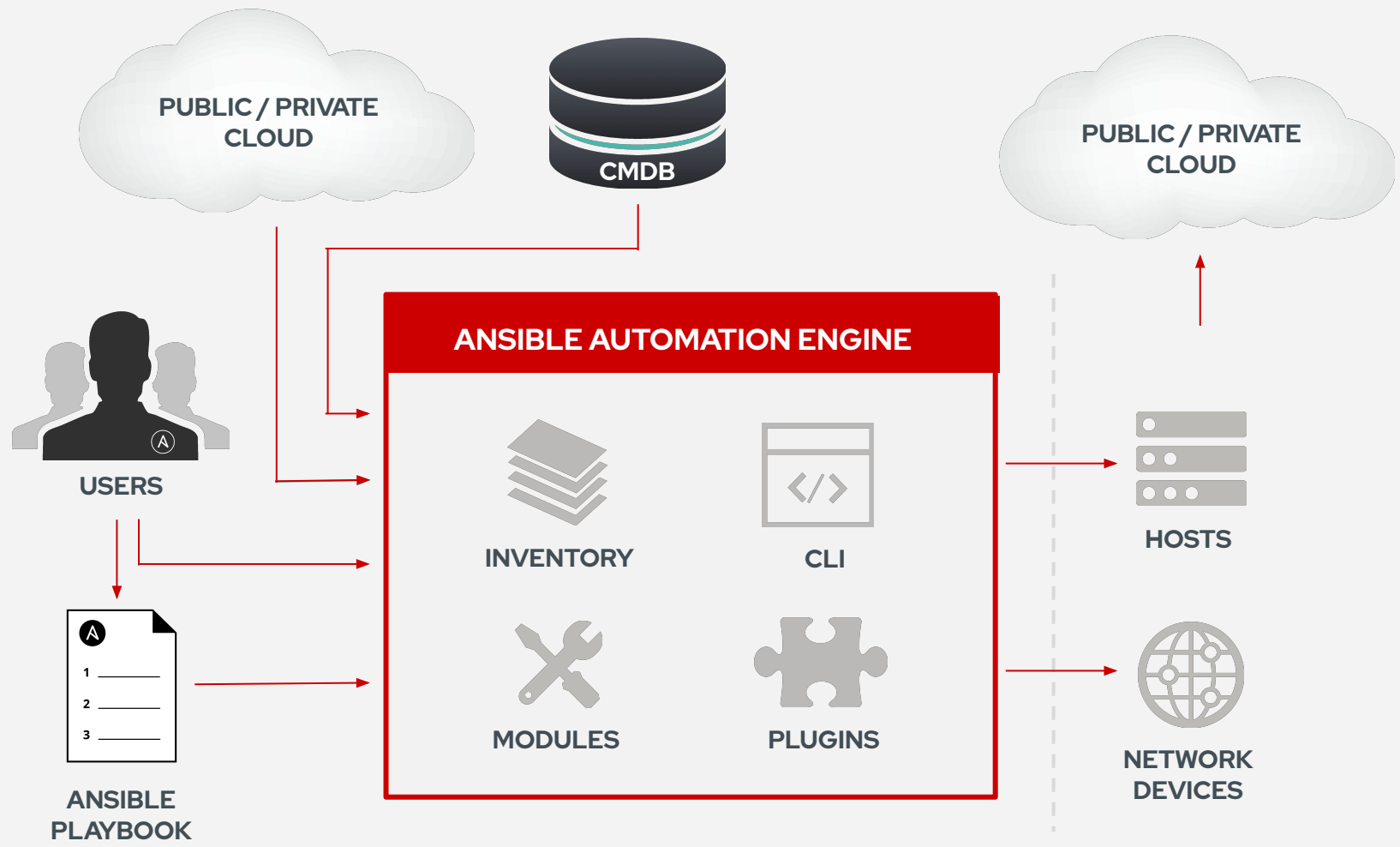
Section 1.1

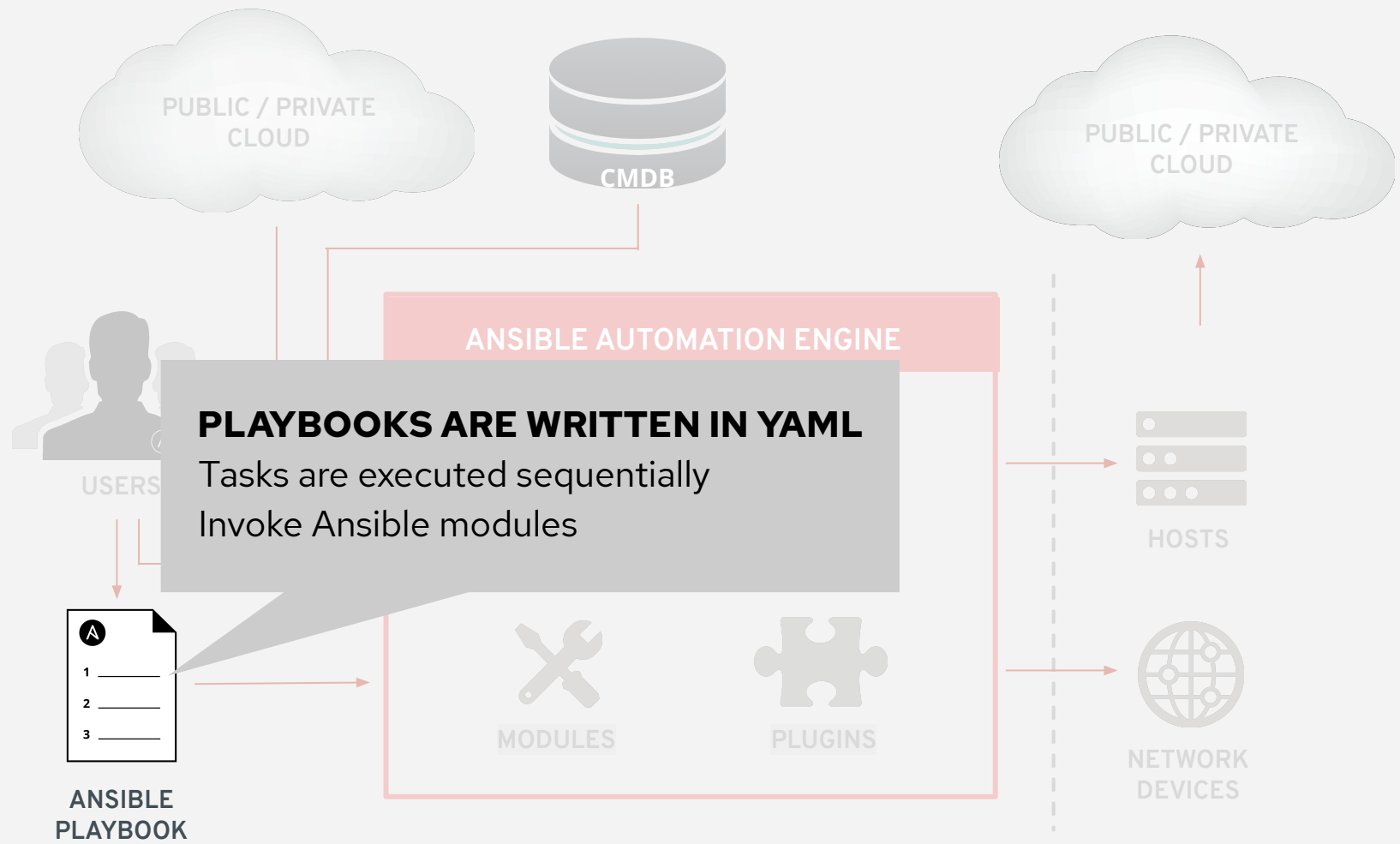
Topics Covered:

- Understanding the Ansible Infrastructure
- Check the prerequisites



Red Hat
Ansible Automation
Platform



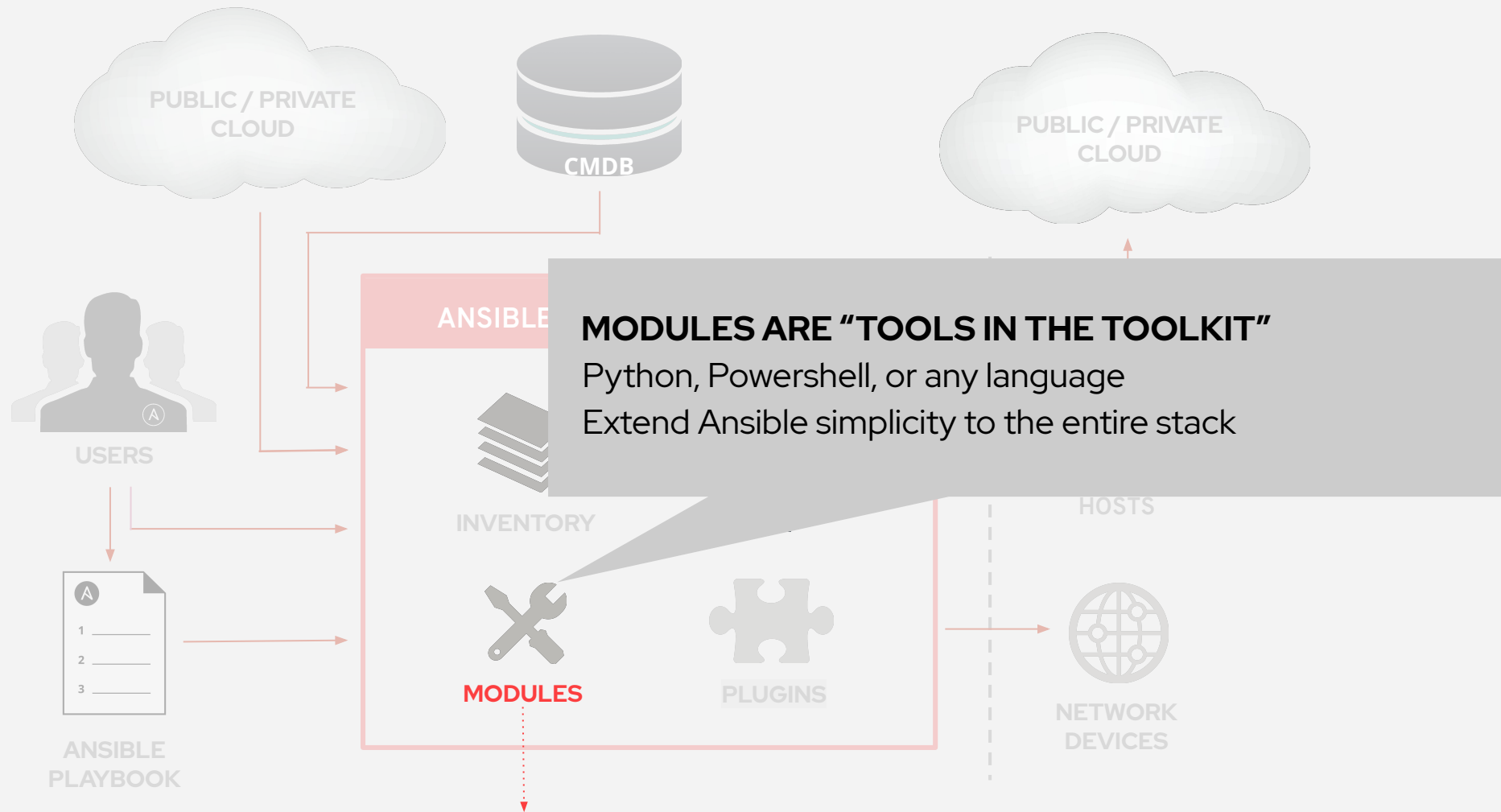


```
---
- name: install and start apache
  hosts: web
  become: yes

  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/

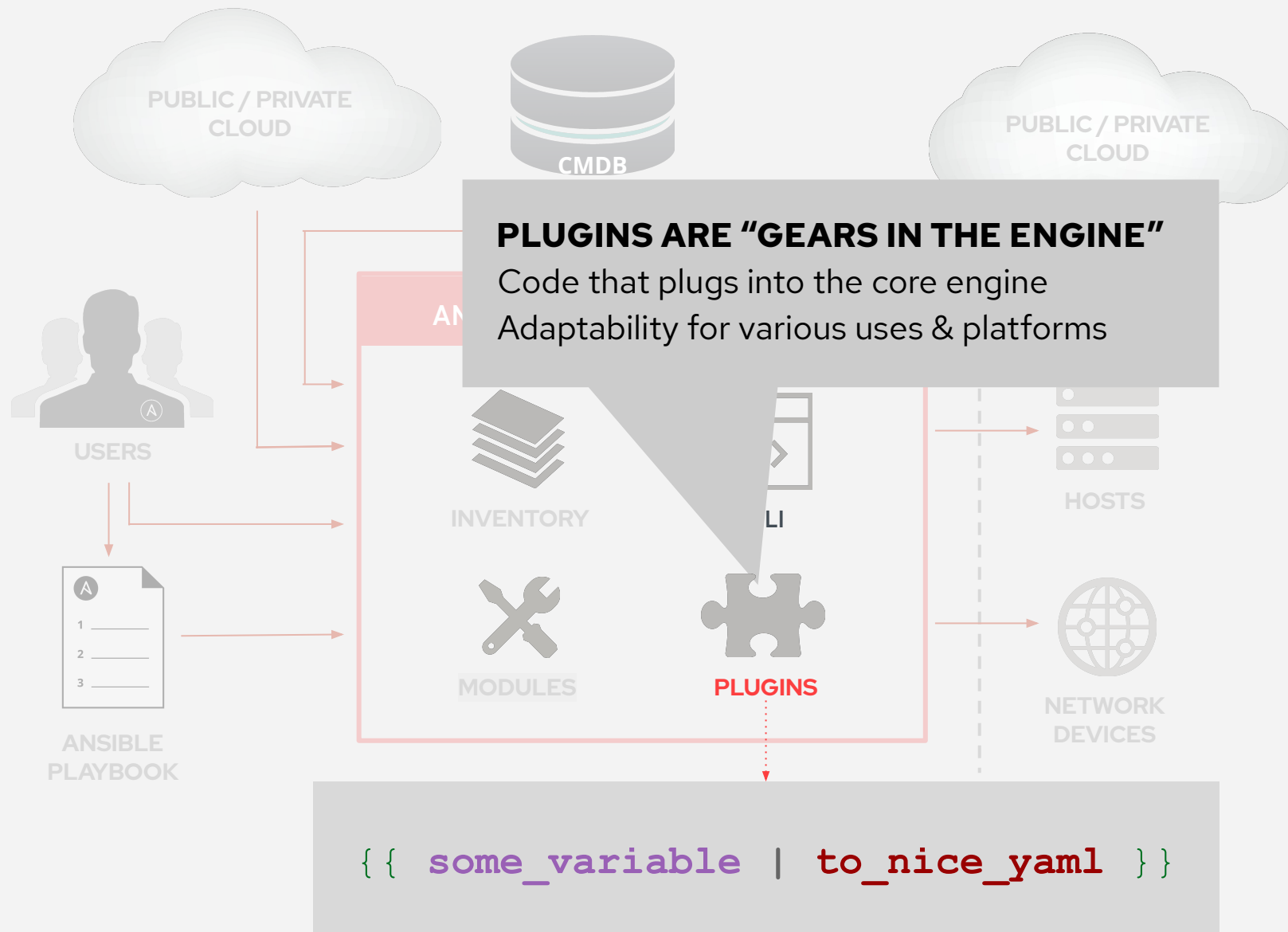
    - name: httpd is started
      service:
        name: httpd
        state: started
```

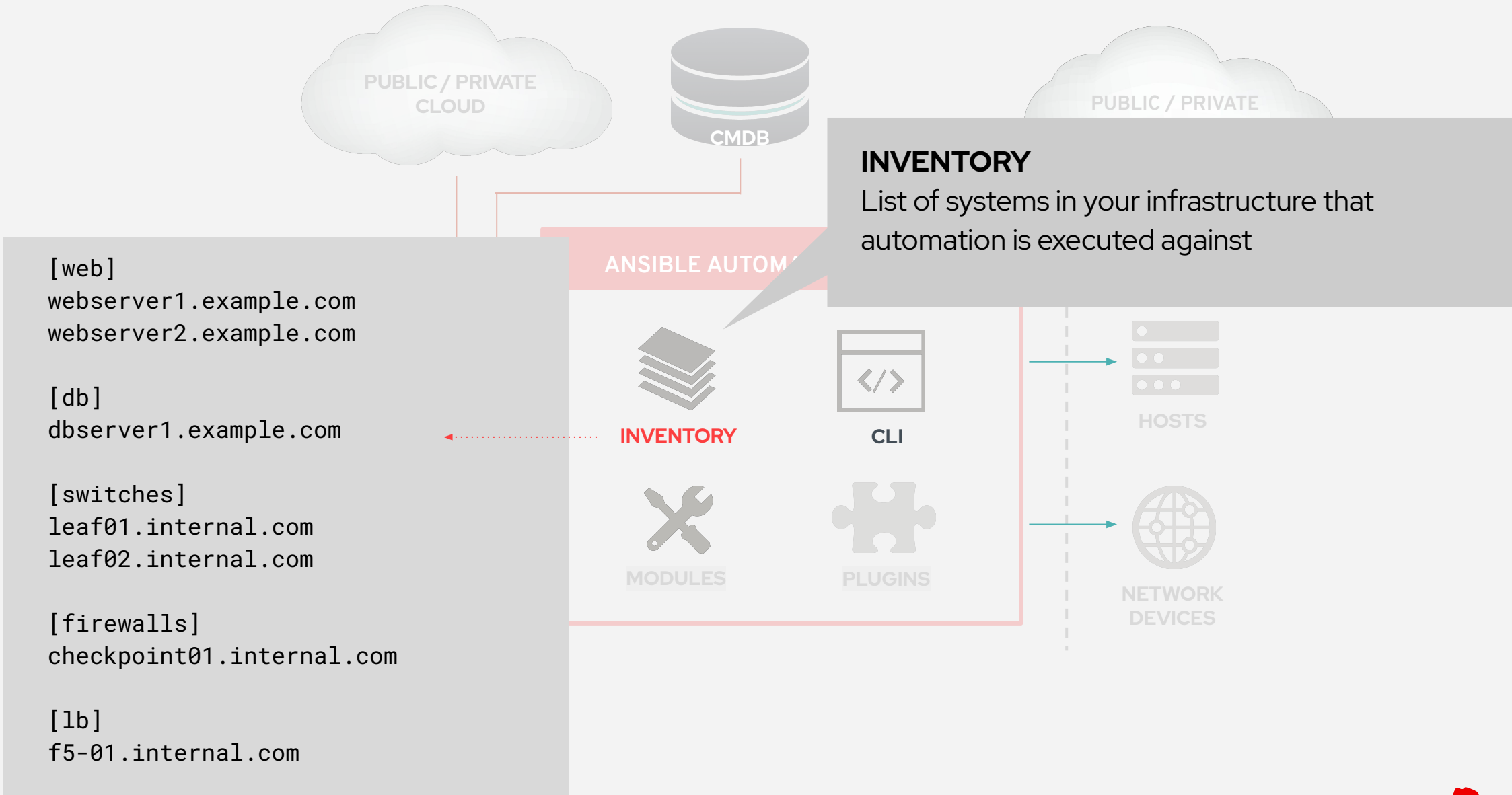


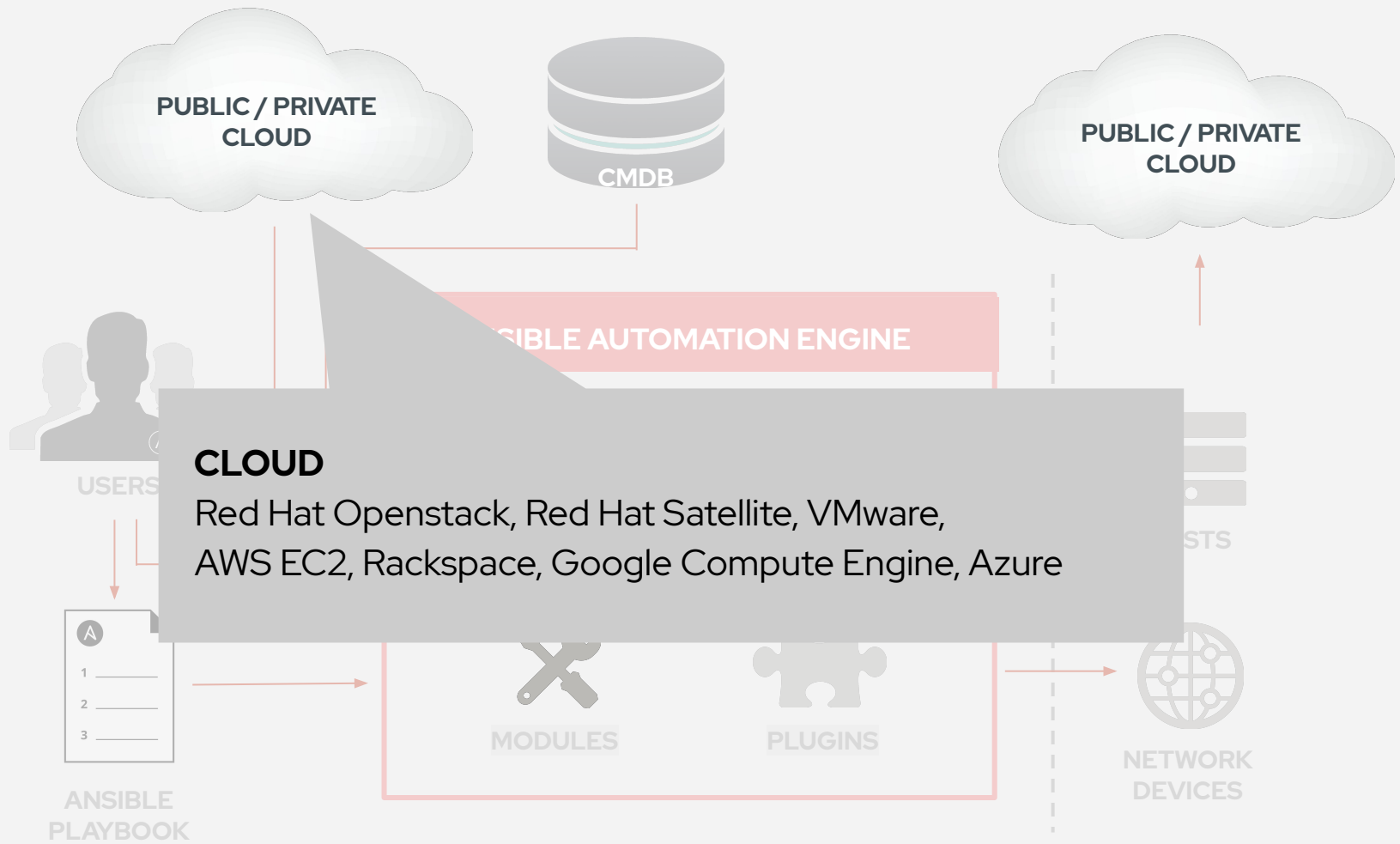
```

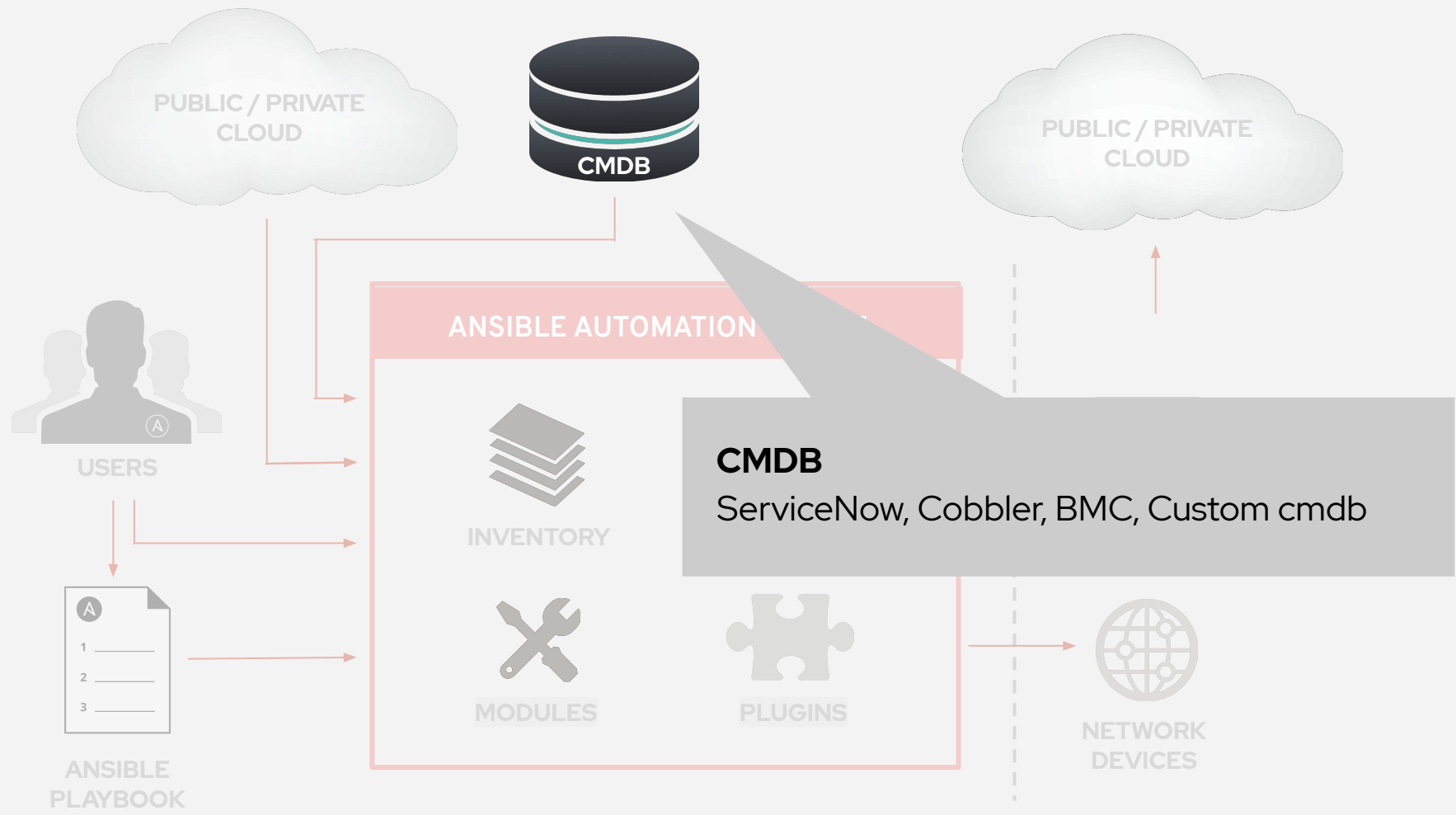
- name: latest index.html file is present
  template:
    src: files/index.html
    dest: /var/www/html/

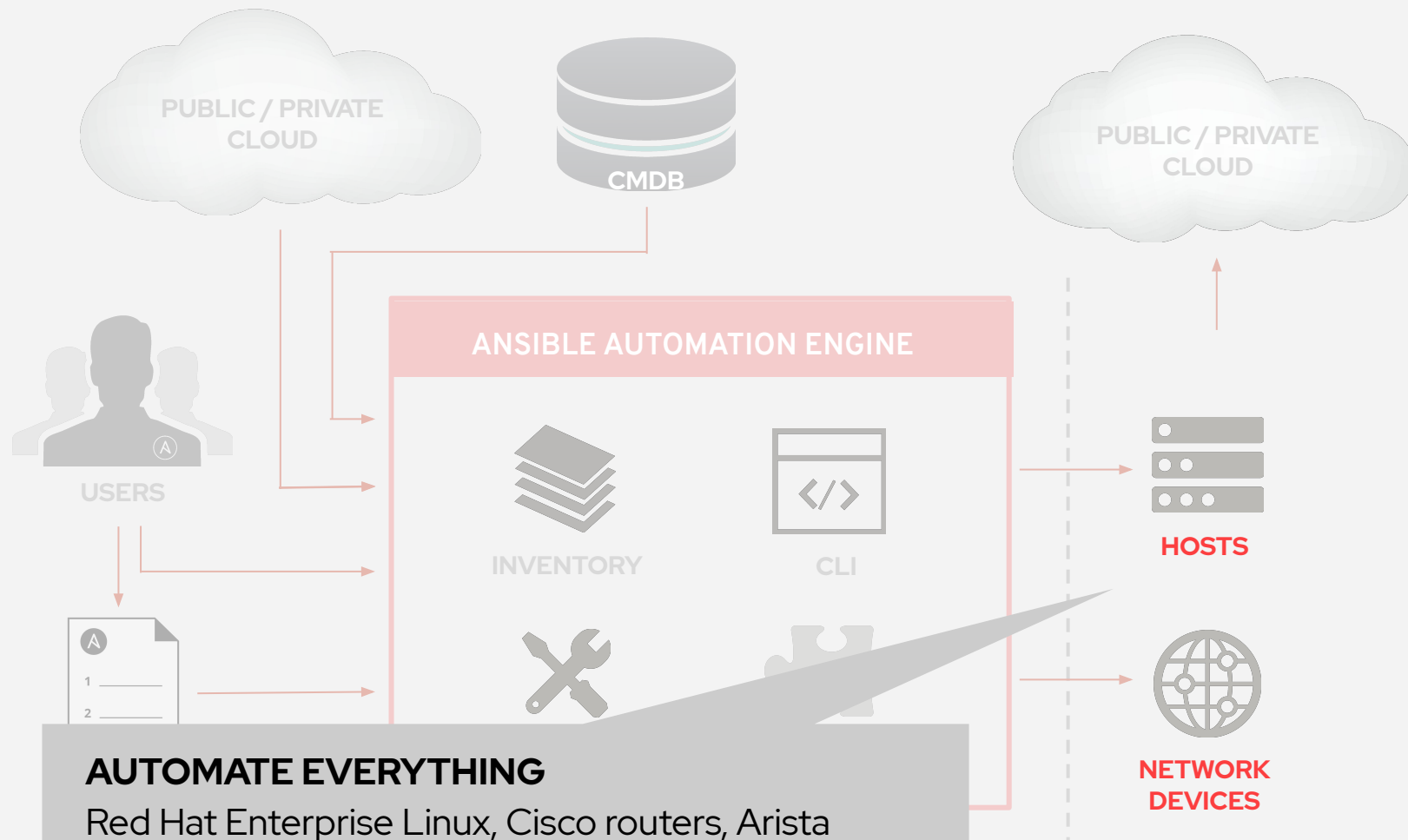
```











AUTOMATE EVERYTHING

Red Hat Enterprise Linux, Cisco routers, Arista switches, Juniper routers, Windows hosts, Check Point firewalls, NetApp storage, F5 load balancers and more

LINUX AUTOMATION

150+
Linux Modules

**AUTOMATE EVERYTHING
LINUX**

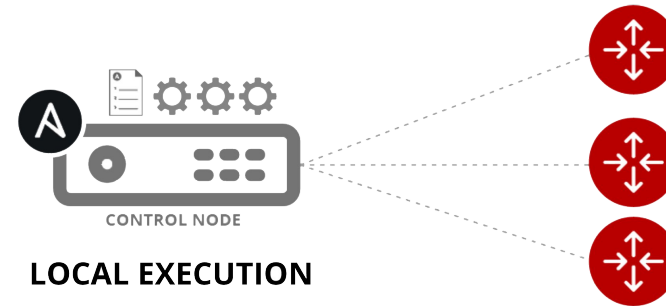
**Red Hat Enterprise Linux, BSD,
Debian, Ubuntu and many more!**

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

ansible.com/get-started

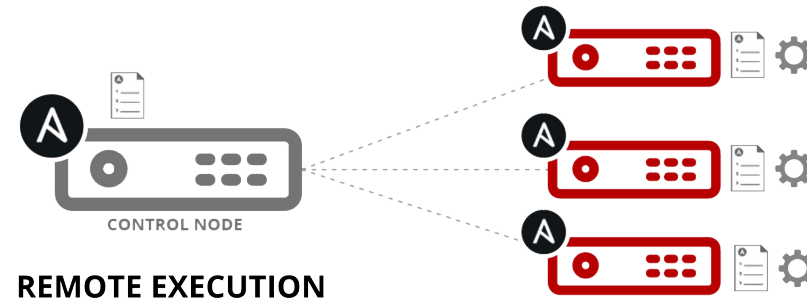
How Ansible Automation works

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**



Red Hat Ansible Automation Platform

Demo Time

Exercise 1.1

Section 1.2

Topics Covered:

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



Red Hat
Ansible Automation
Platform

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 - Basic

```
# 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 - Basic

[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
```

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

The Ansible Configuration

Configuration files will be searched for in the following order:

- **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)

First Ad-Hoc Command: ping

- Single Ansible command to perform a task quickly directly on command line
- Most basic operation that can be performed
- 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 "web1"
```

```
[user@ansible] $ ansible web1 -m setup
```




Red Hat Ansible Automation Platform

Demo Time

Exercise 1.2

Section 1.3

Topics Covered:

- Playbooks basics
- Running a playbook



Red Hat
Ansible Automation
Platform

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

Exercise 1.3

Section 1.4

Topics Covered:

- Working with variables
- What are facts?



Red Hat
Ansible Automation
Platform

An Ansible Playbook Variable Example

```
---
- name: variable playbook test
  hosts: localhost

  vars:
    var_one: awesome
    var_two: ansible is
    var_three: "{{ var_two }} {{ var_one }}"

  tasks:

    - name: print out var_three
      debug:
        msg: "{{var_three}}"
```

An Ansible Playbook Variable Example

```
---
- name: variable playbook test
  hosts: localhost

  vars:
    var_one: awesome
    var_two: ansible is
    var_three: "{{ var_two }} {{ var_one }}"

  tasks:

    - name: print out var_three
      debug:
        msg: "{{var_three}}"
```

ansible is awesome

Facts

- Just like variables, really...
- ...but: coming from the host itself!
- Check them out with the setup module

```
"ansible_facts": {  
  "ansible_default_ipv4": {  
    "address": "10.41.17.37",  
    "macaddress": "00:69:08:3b:a9:16",  
    "interface": "eth0",
```

```
...
```



Red Hat Ansible Automation Platform

Demo Time

Exercise 1.4

Section 1.5

Topics Covered:

- Conditionals
- Handlers
- Loops



Red Hat
Ansible Automation
Platform

Conditionals via VARS

```
vars:  
  my_mood: happy  
  
tasks:  
- name: conditional task, based on my_mood var  
  debug:  
    msg: "Come talk to me. I am {{ my_mood }}!"  
  when: my_mood == "happy"
```

Conditionals with variables

```
vars:  
  my_mood: happy  
  
tasks:  
- name: conditional task, based on my_mood var  
  debug:  
    msg: "Come talk to me. I am {{ my_mood }}!"  
  when: my_mood == "happy"
```

Alternatively

```
debug:  
  msg: "Feel free to interact. I am {{ my_mood }}"  
when: my_mood != "grumpy"
```


Conditionals with facts

```
tasks:  
- name: Install apache  
  apt:  
    name: apache2  
    state: latest  
  when: ansible_distribution == 'Debian' or ansible_distribution == 'Ubuntu'  
  
- name: Install httpd  
  yum:  
    name: httpd  
    state: latest  
  when: ansible_distribution == 'RedHat'
```

Using the previous task state

This is NOT a handler task, but has similar function

- **name: Ensure httpd package is present**
yum:
 - name:** httpd
 - state:** latest**register:** http_results

- **name: Restart httpd**
service:
 - name:** httpd
 - state:** restart**when:** httpd_results.changed

Handler Tasks

A handler task is run when a referring task result shows a change

tasks:

- **name: Ensure httpd package is present**

yum:

name: httpd

state: latest

notify: restart_httpd

handlers:

- **name: restart_httpd**

service:

name: httpd

state: restart

Handler Tasks

```
tasks:
- name: Ensure httpd package is present
  yum:
    name: httpd
    state: latest
    notify: restart_httpd

- name: Standardized index.html file
  copy:
    content: "This is my index.html file for {{ ansible_host }}"
    dest: /var/www/html/index.html
    notify: restart_httpd
```

If **either** task notifies a **changed** result, the handler will be notified **ONCE**.

```
TASK [Ensure httpd package is present] *****
ok: [web2]
ok: [web1] unchanged

TASK [Standardized index.html file] *****
changed: [web2]
changed: [web1] changed

NOTIFIED: [restart_httpd] *****
changed: [web2]
changed: [web1] handler runs once
```

Handler Tasks

```
tasks:  
- name: Ensure httpd package is present  
  yum:  
    name: httpd  
    state: latest  
    notify: restart_httpd  
  
- name: Standardized index.html file  
  copy:  
    content: "This is my index.html file for {{ ansible_host }}"  
    dest: /var/www/html/index.html  
    notify: restart_httpd
```

If **both** of these tasks notifies of a **changed** result, the handler will be notified **ONCE**.

```
TASK [Ensure httpd package is present] *****  
changed: [web2]  
changed: [web1] changed  
  
TASK [Standardized index.html file] *****  
changed: [web2]  
changed: [web1] changed  
  
NOTIFIED: [restart_httpd] *****  
changed: [web2]  
changed: [web1] handler runs once
```

Handler Tasks

```
tasks:
- name: Ensure httpd package is present
  yum:
    name: httpd
    state: latest
    notify: restart_httpd

- name: Standardized index.html file
  copy:
    content: "This is my index.html file for {{ ansible_host }}"
    dest: /var/www/html/index.html
    notify: restart_httpd
```

If **neither** task notifies a **changed** result, the handler **does not run**.

```
TASK [Ensure httpd package is present] *****
ok: [web2]
ok: [web1]    unchanged

TASK [Standardized index.html file] *****
ok: [web2]
ok: [web1]    unchanged

PLAY RECAP *****
web2      : ok=2   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
web1      : ok=2   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Variables & Loops

Great opportunity to use a loop

```
---
- name: Ensure users
  hosts: node1
  become: yes

  tasks:
    - name: Ensure user is present
      user:
        name: dev_user
        state: present

    - name: Ensure user is present
      user:
        name: qa_user
        state: present

    - name: Ensure user is present
      user:
        name: prod_user
        state: present
```

Variables & Loops

Using loops to simplify tasks

```
---  
- name: Ensure users  
  hosts: node1  
  become: yes  
  
  tasks:  
    - name: Ensure users are present  
      user:  
        name: "{{item}}"  
        state: present  
      loop:  
        - dev_user  
        - qa_user  
        - prod_user
```




Red Hat Ansible Automation Platform

Demo Time

Exercise 1.5

Section 1.6

Topics Covered:

- Templates



Red Hat
Ansible Automation
Platform

Variables & Templates

Using a system fact or declared variable to write a file

```
- name: Ensure apache is installed and started
hosts: web
become: yes
vars:
  http_port: 80
  http_docroot: /var/www/mysite.com

tasks:
  - name: Verify correct config file is present
    template:
      src: templates/httpd.conf.j2
      dest: /etc/httpd/conf/httpd.conf
```

Variables & Templates

Using a system fact or declared variable to write a file

```
- name: Ensure apache is installed and started
hosts: web
become: yes
vars:
  http_port: 80
  http_docroot: /var/www/mysite.com

tasks:
- name: Verify correct config file is present
  template:
    src: templates/httpd.conf.j2
    dest: /etc/httpd/conf/httpd.conf
```

```
## Excerpt from httpd.conf.j2

# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
# Listen 80    ## original line
Listen {{ http_port }}

# DocumentRoot: The directory out of which you will serve your
# documents.
# DocumentRoot "/var/www/html"
DocumentRoot {{ http_docroot }}
```



Red Hat Ansible Automation Platform

Demo Time

Exercise 1.6

Section 1.7

Topics Covered:

- What are roles?
- How they look like
- Galaxy



Red Hat
Ansible Automation
Platform

Role structure

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

```
user/  
├── defaults  
│   └── main.yml  
├── handlers  
│   └── main.yml  
├── meta  
│   └── main.yml  
├── README.md  
├── tasks  
│   └── main.yml  
├── templates  
├── tests  
│   ├── inventory  
│   └── test.yml  
└── vars  
    └── main.yml
```



Ansible Galaxy

**Sharing
Content**

Community

**Roles, and
more**



Red Hat Ansible Automation Platform

Demo Time

Exercise 1.7

Section 1.8

Topics Covered:

- A bonus lab – try it on your own, and when time permits



Red Hat
Ansible Automation
Platform



Red Hat Ansible Automation Platform

Demo Time

Exercise 1.8

Section 2

Ansible Tower



Red Hat
Ansible Automation
Platform

Section 2.1

Topics Covered:

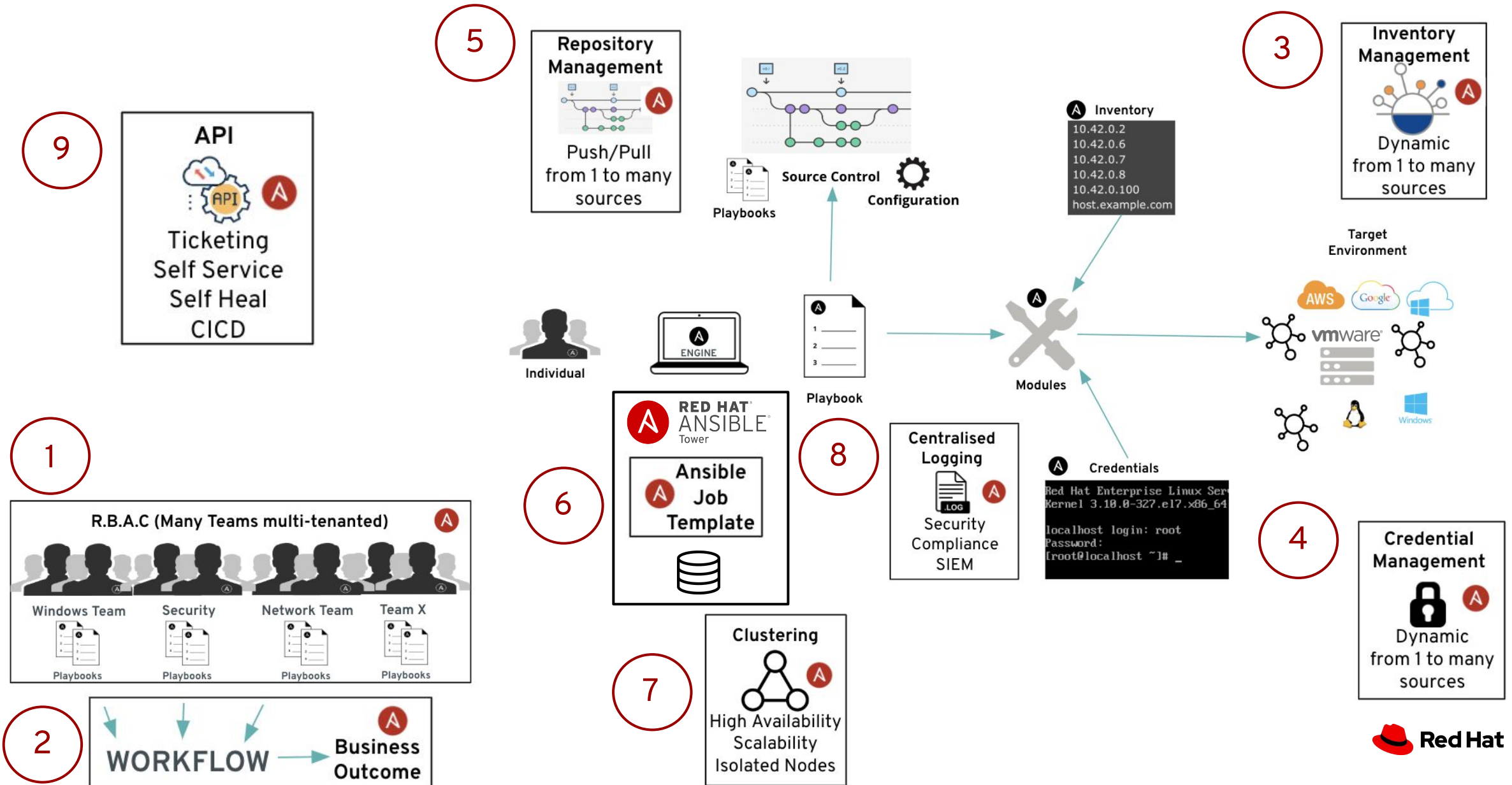
- Introduction to Tower



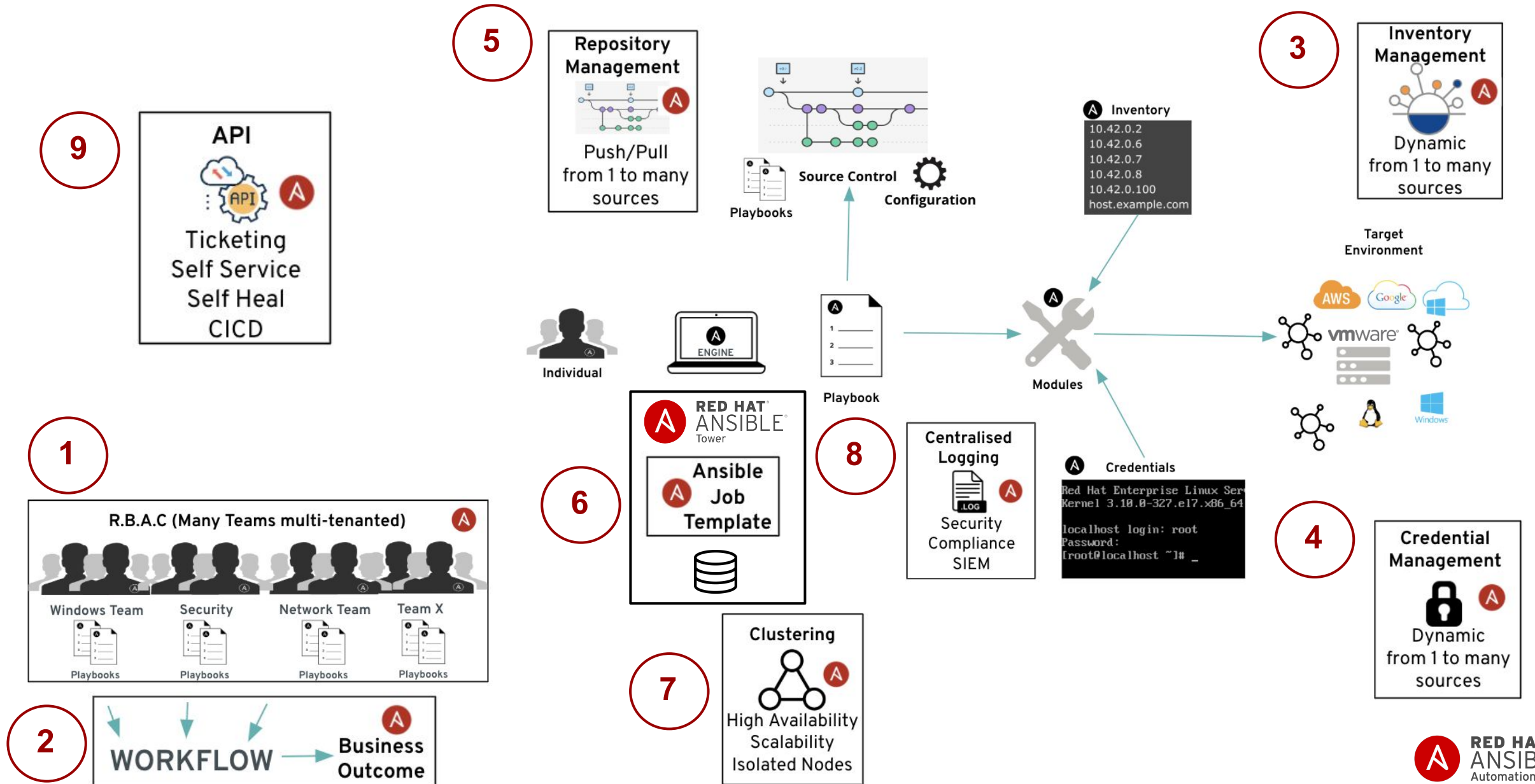
Red Hat
Ansible Automation
Platform

How Ansible Works - Ansible Tower

CONFIDENTIAL Designator



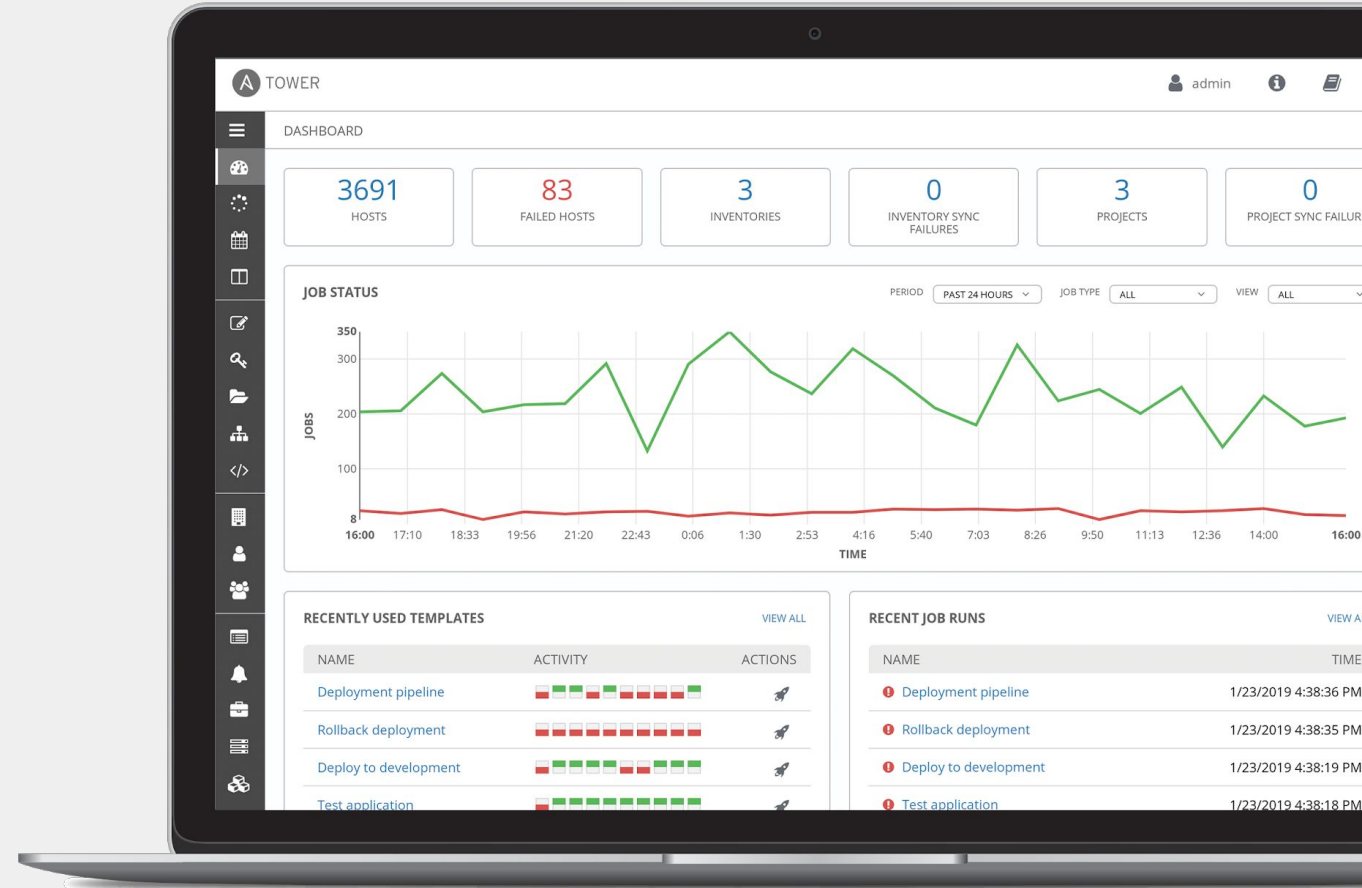
How Ansible Works - The Power of Ansible Tower



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.

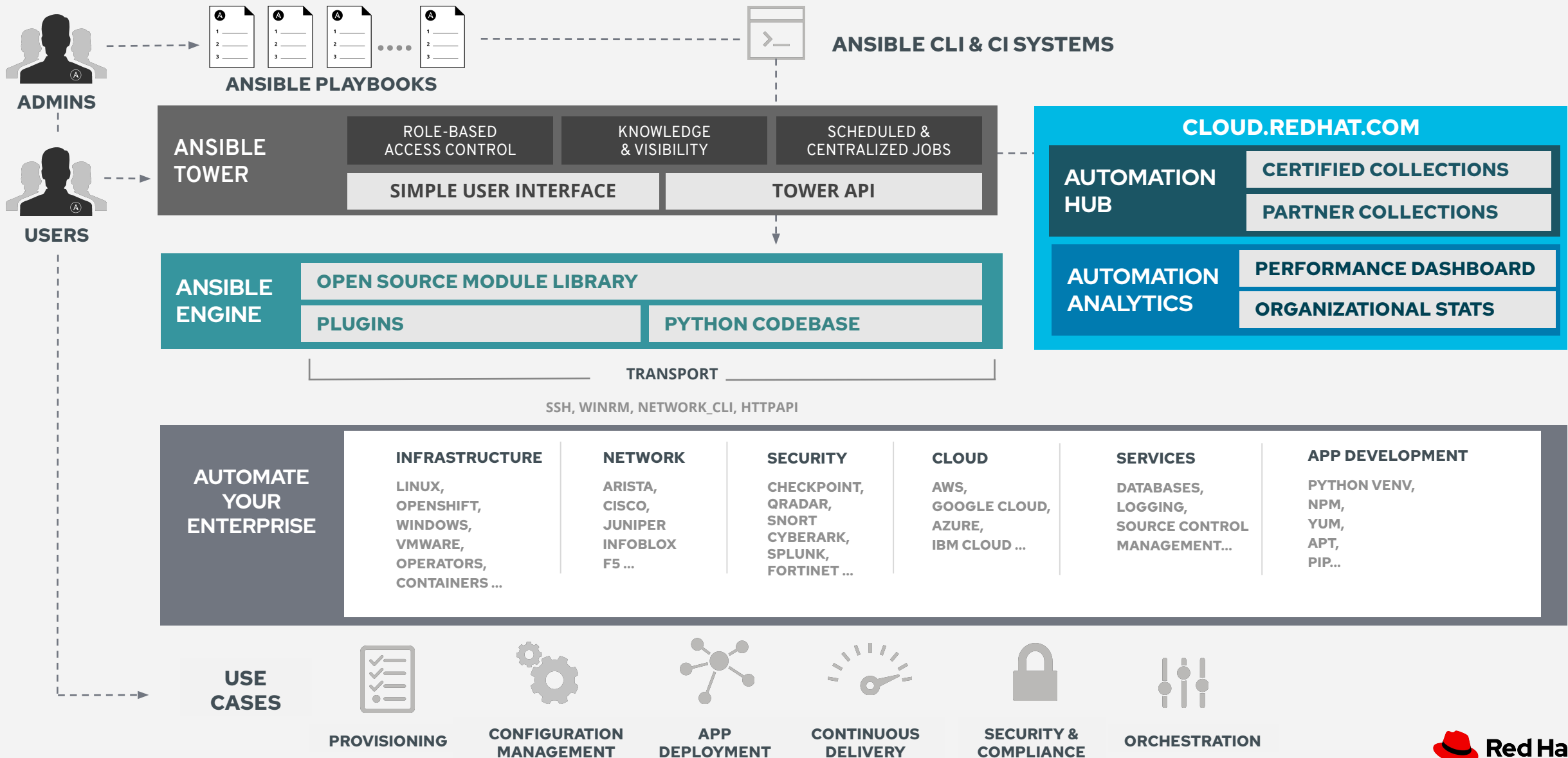
Centralized logging

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 Platform





Red Hat Ansible Automation Platform

Demo Time

Exercise 2.1

Section 2.2

Topics Covered:

- Inventories
- Credentials

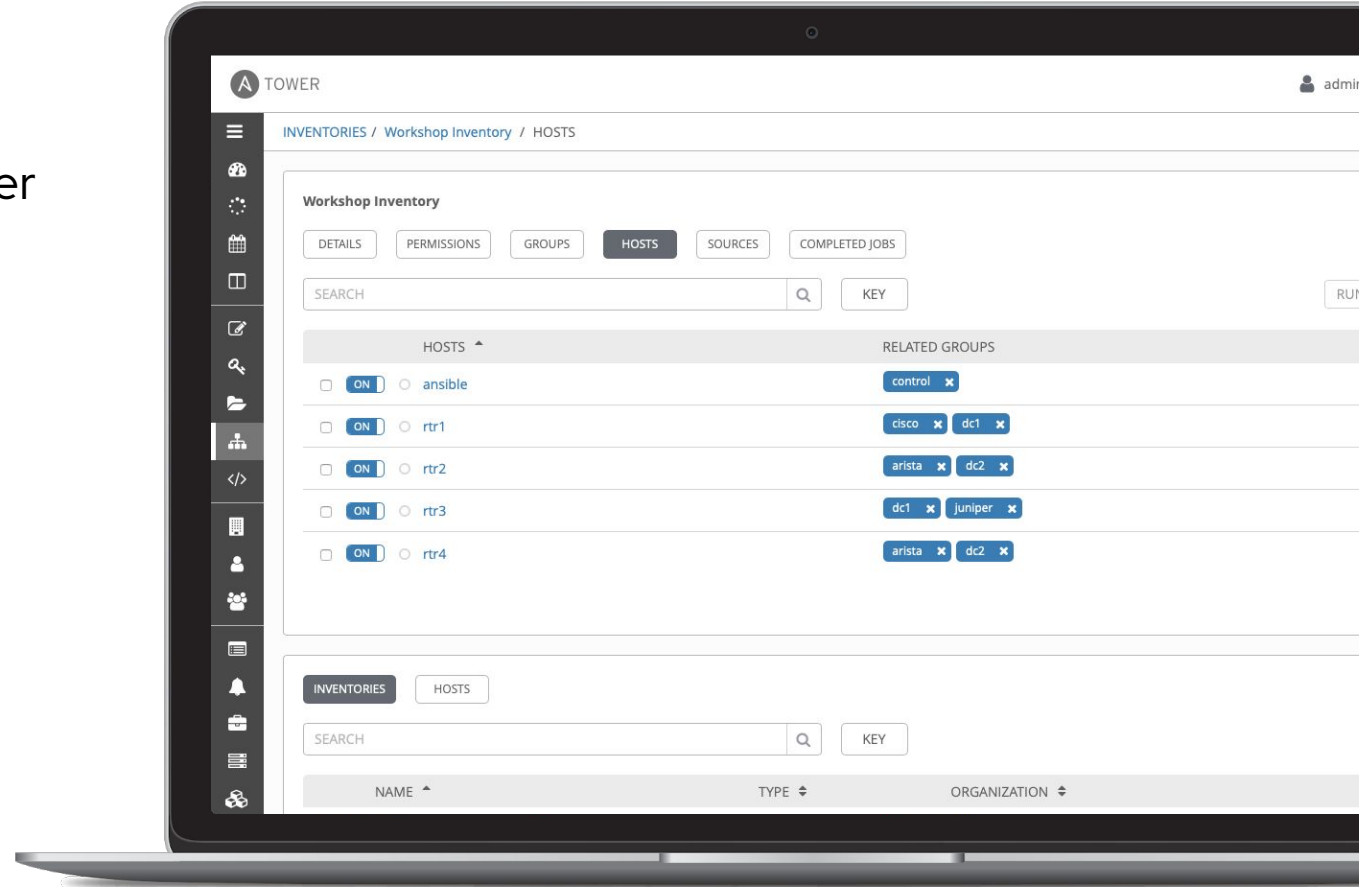


Red Hat
Ansible Automation
Platform

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

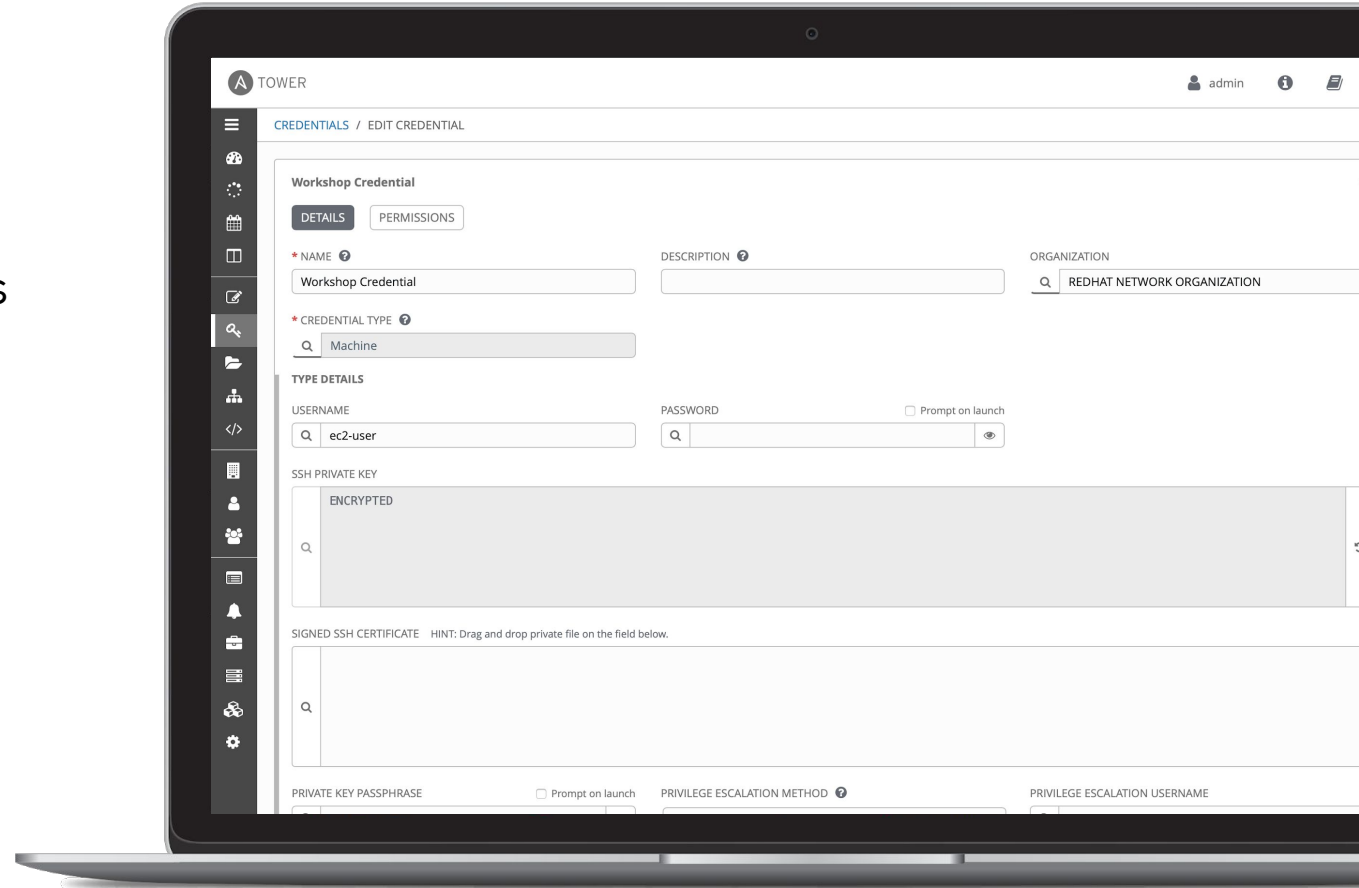


Credentials

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

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

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





Red Hat Ansible Automation Platform

Demo Time

Exercise 2.2

Section 2.3

Topics Covered:

- Projects
- Job Templates

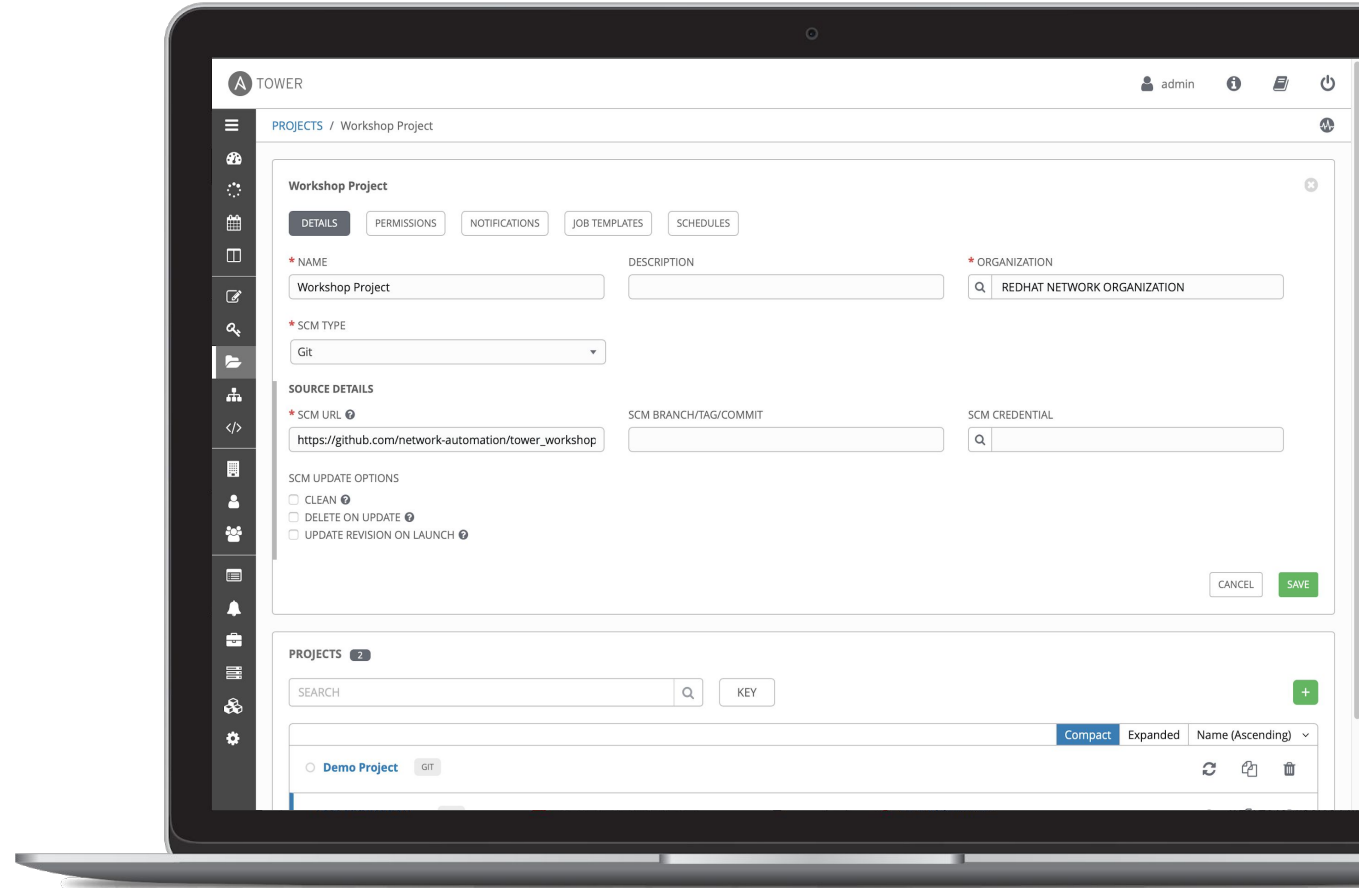


Red Hat
Ansible Automation
Platform

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.



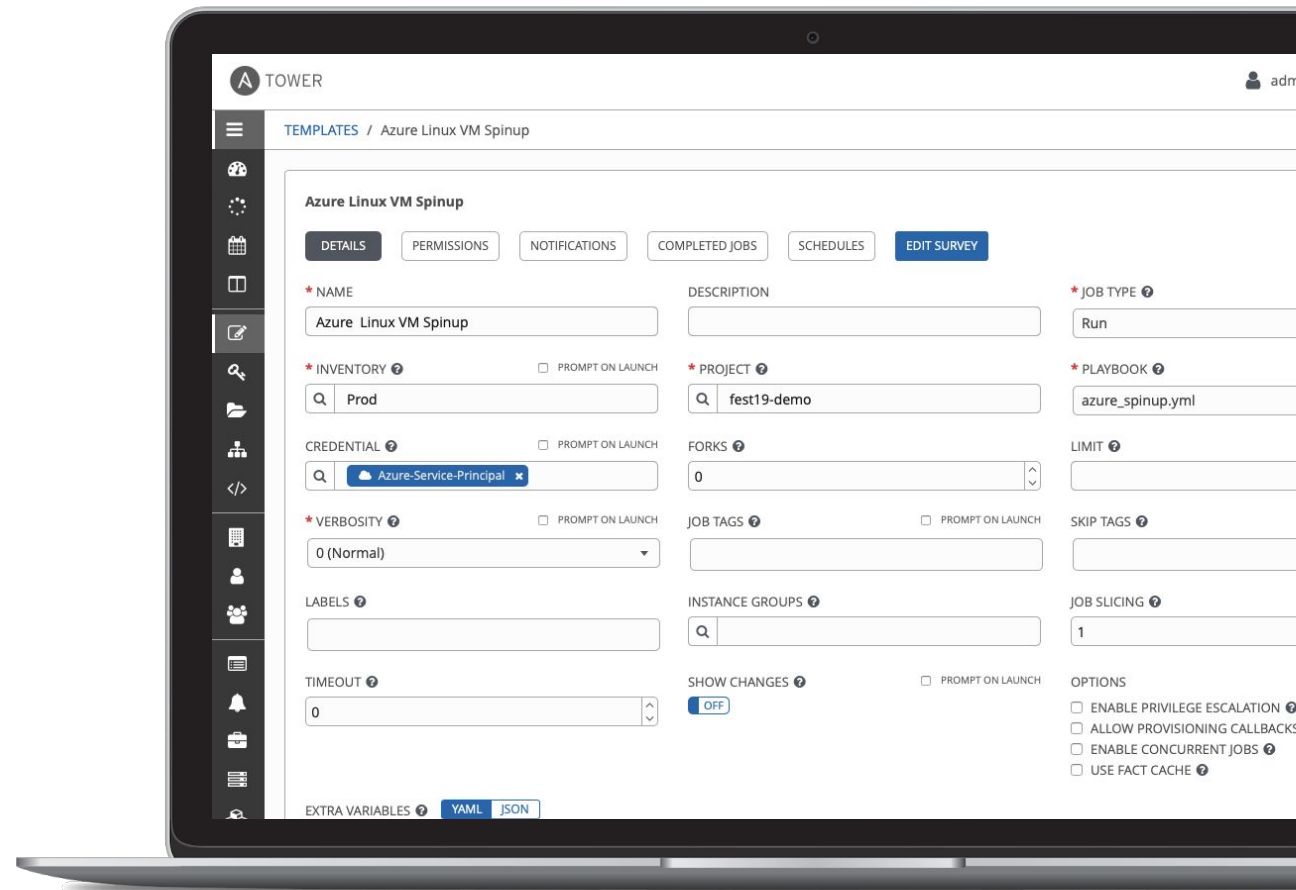
Job Templates

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



Expanding on Job Templates

Job Templates can be found and created by clicking the **Templates** button under the *RESOURCES* section on the left menu.



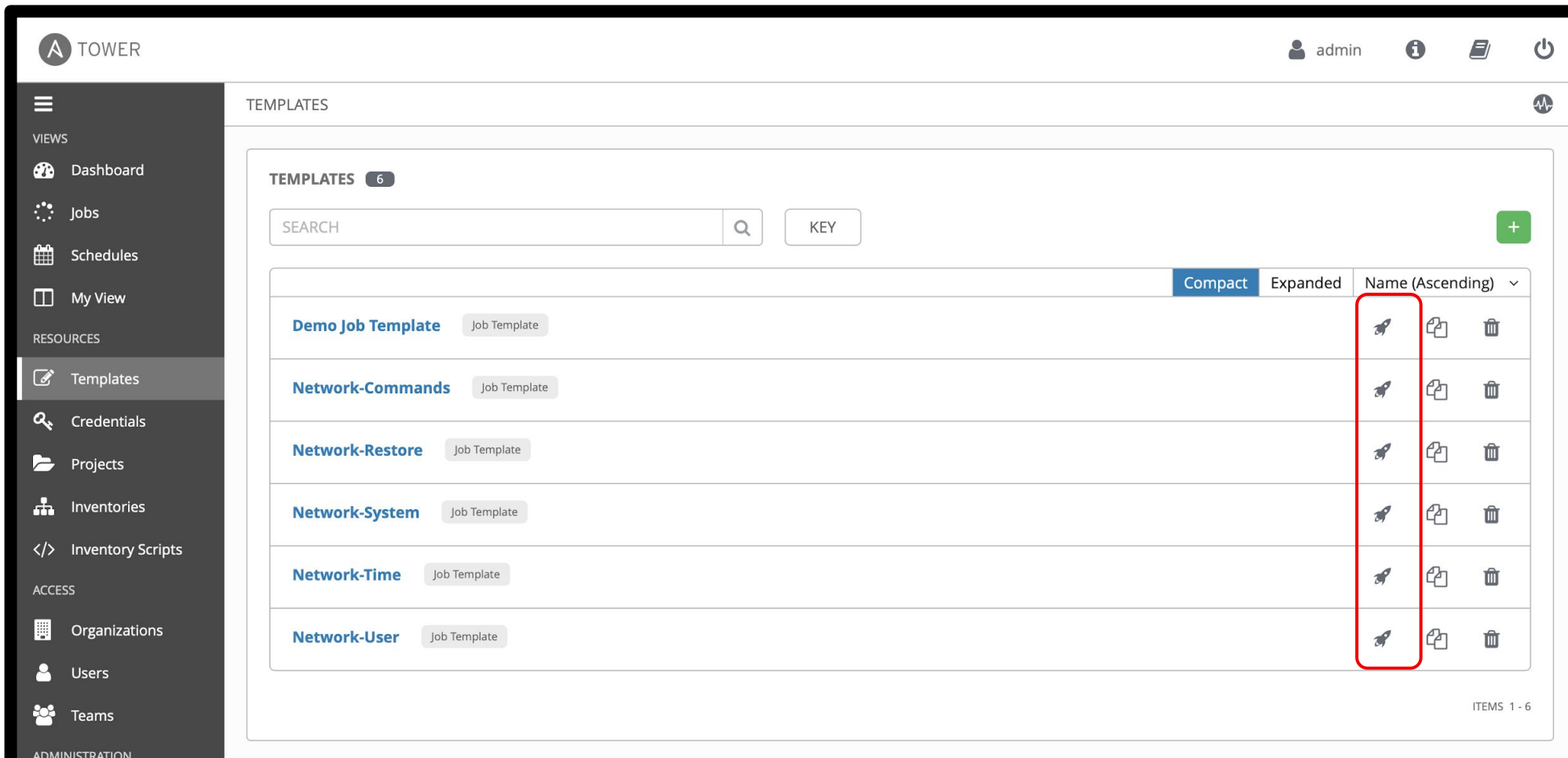
The screenshot shows the Tower web interface. The top navigation bar includes the 'TOWER' logo, a user profile for 'admin', and several utility icons. The left sidebar is divided into sections: 'VIEWS' (Dashboard, Jobs, Schedules, My View) and 'RESOURCES' (Templates, Credentials, Projects, Inventories, Inventory Scripts). The 'Templates' item is highlighted. The main content area is titled 'TEMPLATES' and shows a list of 6 job templates. The list is displayed in a table with columns for 'Name (Ascending)', 'Compact', and 'Expanded'. Each row represents a job template with its name, a 'Job Template' tag, and three action icons (rocket, document, trash).

TEMPLATES 6		SEARCH	KEY	
				Compact Expanded Name (Ascending) v
Demo Job Template	Job Template			🚀 📄 🗑️
Network-Commands	Job Template			🚀 📄 🗑️
Network-Restore	Job Template			🚀 📄 🗑️
Network-System	Job Template			🚀 📄 🗑️
Network-Time	Job Template			🚀 📄 🗑️
Network-User	Job Template			🚀 📄 🗑️



















ITEMS 1 - 6

Executing an existing Job Template

Job Templates can be launched by clicking the **rocketship button** for the corresponding Job Template 



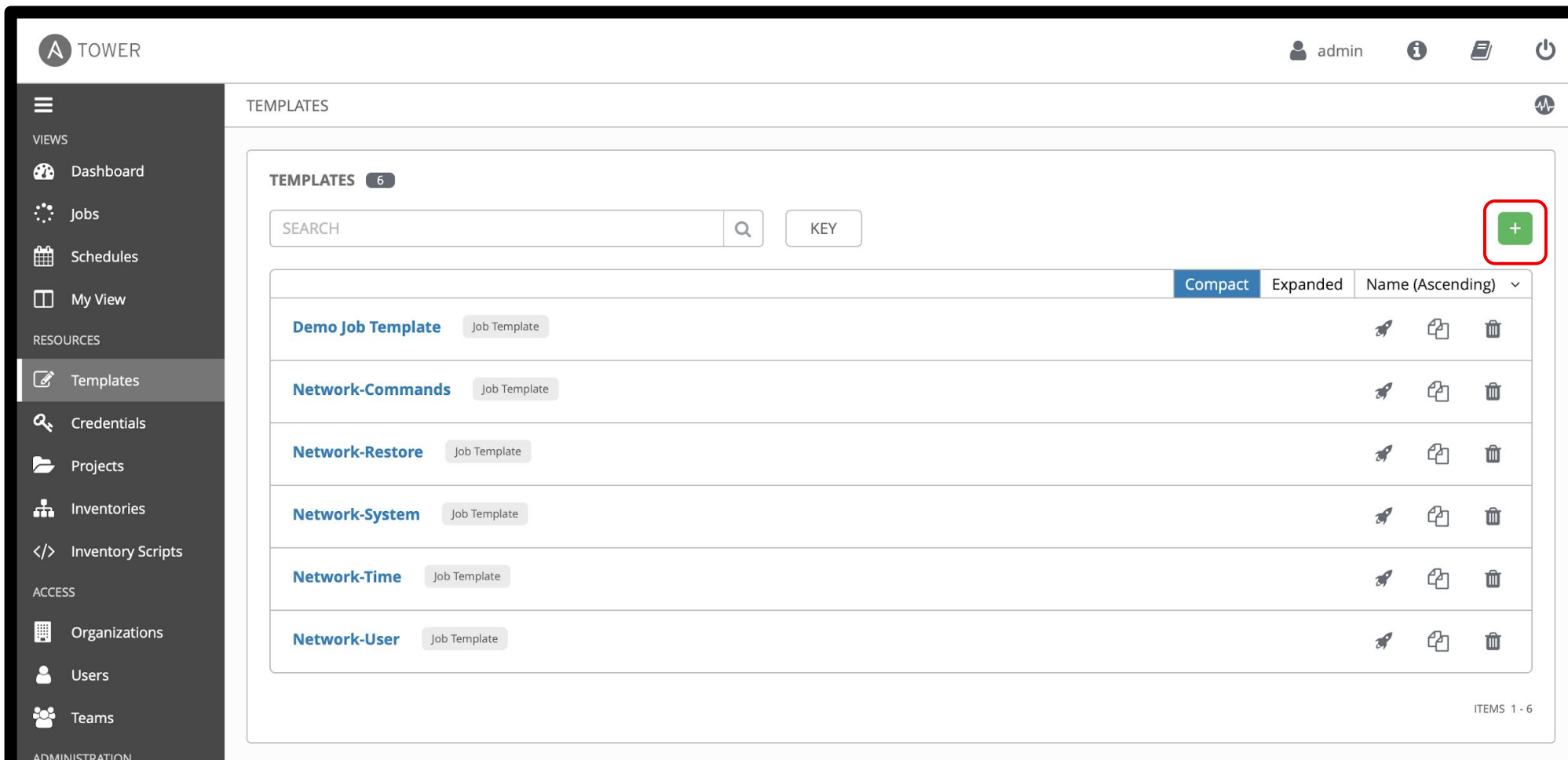
The screenshot displays the Tower web interface. The top navigation bar includes the 'TOWER' logo, user 'admin', and various utility icons. A left sidebar contains navigation options like 'Dashboard', 'Jobs', 'Schedules', and 'Templates'. The main content area is titled 'TEMPLATES' and shows a list of six job templates. Each row includes the template name, a 'Job Template' label, and a set of action icons: a rocketship icon, a copy icon, and a delete icon. The rocketship icon for the 'Demo Job Template' row is highlighted with a red rectangular box.

TEMPLATES 6		SEARCH	KEY	+
		Compact	Expanded	Name (Ascending) v
Demo Job Template	Job Template			
Network-Commands	Job Template			
Network-Restore	Job Template			
Network-System	Job Template			
Network-Time	Job Template			
Network-User	Job Template			

ITEMS 1 - 6

Creating a new Job Template (1/2)

New Job Templates can be created by clicking the **plus button**



The screenshot shows the Tower web interface. The top navigation bar includes the Tower logo, the user name 'admin', and icons for help, search, and power. The left sidebar contains navigation options: VIEWS (Dashboard, Jobs, Schedules, My View) and RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts). The main content area is titled 'TEMPLATES' and shows a list of 6 templates. A search bar and a 'KEY' button are at the top of the list. A green plus button is highlighted with a red box in the top right corner of the list area. The list of templates includes: Demo Job Template, Network-Commands, Network-Restore, Network-System, Network-Time, and Network-User. Each template row has a rocket icon, a copy icon, and a delete icon. The bottom right corner of the list area shows 'ITEMS 1 - 6'.

Creating a new Job Template (2/2)

This **New Job Template** window is where the inventory, project and credential are assigned. The red asterisk * means the field is required.

NEW JOB TEMPLATE

DETAILS PERMISSIONS COMPLETED JOBS SCHEDULES ADD SURVEY

* NAME	DESCRIPTION	* JOB TYPE ? <input type="checkbox"/> PROMPT ON LAUNCH
<input type="text"/>	<input type="text"/>	Run
* INVENTORY ? <input type="checkbox"/> PROMPT ON LAUNCH	* PROJECT ?	* PLAYBOOK ?
<input type="text"/>	<input type="text"/>	Choose a playbook
CREDENTIAL ? <input type="checkbox"/> PROMPT ON LAUNCH	FORKS ?	LIMIT ? <input type="checkbox"/> PROMPT ON LAUNCH
<input type="text"/>	0	<input type="text"/>
* VERBOSITY ? <input type="checkbox"/> PROMPT ON LAUNCH	JOB TAGS ? <input type="checkbox"/> PROMPT ON LAUNCH	SKIP TAGS ? <input type="checkbox"/> PROMPT ON LAUNCH
0 (Normal)	<input type="text"/>	<input type="text"/>
LABELS ?	INSTANCE GROUPS ?	JOB SLICING ?
<input type="text"/>	<input type="text"/>	1
TIMEOUT ?	SHOW CHANGES ? <input type="checkbox"/> PROMPT ON LAUNCH	OPTIONS
0	<input type="button" value="OFF"/>	<input type="checkbox"/> ENABLE PRIVILEGE ESCALATION ?
		<input type="checkbox"/> ALLOW PROVISIONING CALLBACKS ?



Red Hat Ansible Automation Platform

Demo Time

Exercise 2.3

Section 2.4

Topics Covered:

- Surveys

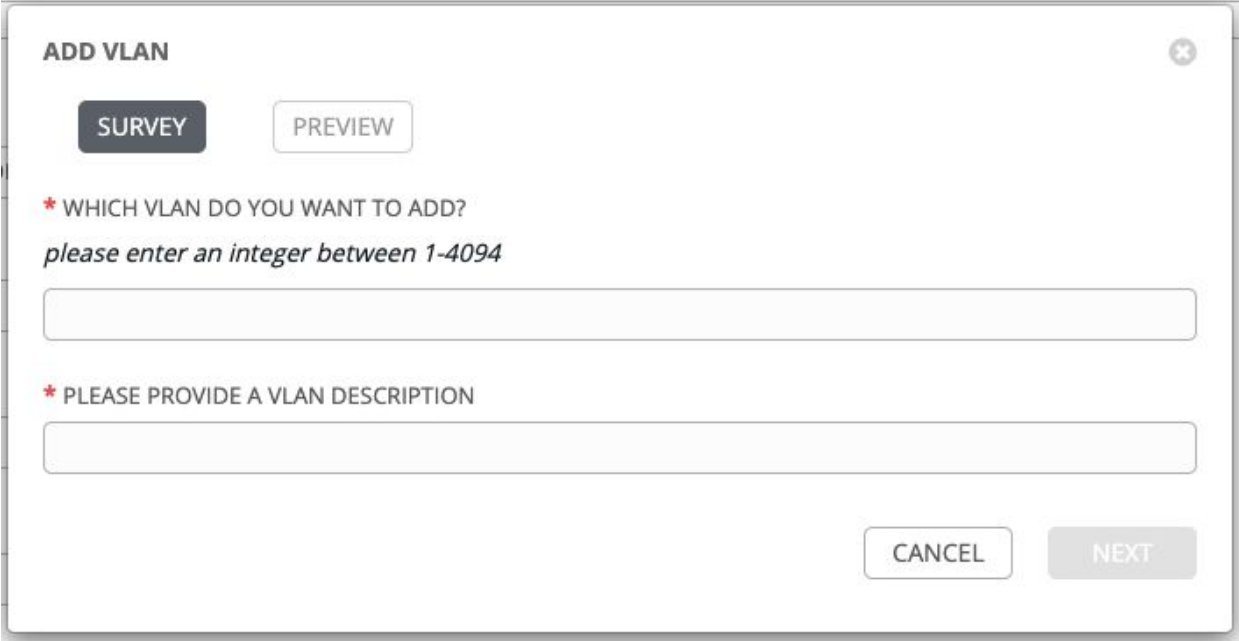


Red Hat
Ansible Automation
Platform

Surveys

Tower surveys allow you to configure how a job runs via a series of questions, making it simple to customize your jobs in a user-friendly way.

An Ansible Tower survey is a simple question-and-answer form that allows users to customize their job runs. Combine that with Tower's role-based access control, and you can build simple, easy self-service for your users.



The screenshot shows a web form titled "ADD VLAN" with a close button in the top right corner. Below the title are two buttons: "SURVEY" (highlighted in dark grey) and "PREVIEW" (light grey). The form contains two required fields, each marked with a red asterisk:

- The first field is labeled "* WHICH VLAN DO YOU WANT TO ADD?" and includes a hint: "please enter an integer between 1-4094". Below the label is an empty text input box.
- The second field is labeled "* PLEASE PROVIDE A VLAN DESCRIPTION" and has an empty text input box below it.

At the bottom right of the form are two buttons: "CANCEL" (light grey) and "NEXT" (dark grey).

Creating a Survey (1/2)

Once a Job Template is saved, the **Add Survey Button** will appear

Click the button to open the Add Survey window.

ADD SURVEY

The screenshot displays the Tower web interface. The top navigation bar includes the 'TOWER' logo, user 'admin', and various utility icons. The left sidebar contains navigation options: VIEWS (Dashboard, Jobs, Schedules, My View) and RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts). The main content area is titled 'TEMPLATES / Configure Banner' and features a 'Configure Banner' modal window. This modal has tabs for 'DETAILS', 'PERMISSIONS', 'NOTIFICATIONS', 'COMPLETED JOBS', 'SCHEDULES', and 'EDIT SURVEY'. The 'EDIT SURVEY' tab is highlighted with a red box. The configuration fields include: NAME (Configure Banner), DESCRIPTION, JOB TYPE (Run), INVENTORY (Workshop Inventory), PROJECT (Workshop Project), PLAYBOOK (network_banner.yml), CREDENTIAL (Workshop Credential), FORKS (0), LIMIT, VERBOSITY (0 (Normal)), JOB TAGS, SKIP TAGS, LABELS, INSTANCE GROUPS, and JOB SLICING. Each field has a search icon and a 'PROMPT ON LAUNCH' checkbox.

Creating a Survey (2/2)

The Add Survey window allows the Job Template to prompt users for one or more questions. The answers provided become variables for use in the Ansible Playbook.

The screenshot displays the 'CONFIGURE BANNER | SURVEY' window with a 'SURVEY' toggle set to 'ON'. The window is divided into two main sections: 'EDIT SURVEY PROMPT' and 'PREVIEW'.

EDIT SURVEY PROMPT

- * PROMPT**: A text input field containing 'Please enter the banner text'.
- DESCRIPTION**: A text input field containing 'Please type into the text field the desired banner'.
- * ANSWER VARIABLE NAME**: A text input field containing 'net_banner'.
- * ANSWER TYPE**: A dropdown menu set to 'Textarea'.
- MINIMUM LENGTH**: A numeric input field set to '0'.
- MAXIMUM LENGTH**: A numeric input field set to '4096'.
- DEFAULT ANSWER**: An empty text input field.
- REQUIRED**

PREVIEW

- * PLEASE ENTER THE BANNER TEXT**: A heading for the preview section.
- Please type into the text field the desired banner*: A line of italicized text.
- A large, empty, light-gray rectangular area representing the banner text input field.
- Icons for editing (pencil) and deleting (trash) are visible to the right of the preview area.

At the bottom of the window, there are buttons for 'CLEAR', 'UPDATE', 'CANCEL', and 'OK'.

Creating a Survey (2/2)

The Add Survey window allows the Job Template to prompt users for one or more questions. The answers provided become variables for use in the Ansible Playbook.

The screenshot displays the 'CONFIGURE BANNER | SURVEY' window with a toggle switch set to 'ON'. The window is divided into two main sections: 'EDIT SURVEY PROMPT' and 'PREVIEW'.

EDIT SURVEY PROMPT

- * PROMPT**: A text input field containing 'Please enter the banner text'.
- DESCRIPTION**: A text input field containing 'Please type into the text field the desired banner'.
- * ANSWER VARIABLE NAME**: A text input field containing 'net_banner'.
- * ANSWER TYPE**: A dropdown menu set to 'Textarea'.
- MINIMUM LENGTH**: A numeric input field set to '0'.
- MAXIMUM LENGTH**: A numeric input field set to '4096'.
- DEFAULT ANSWER**: An empty text input field.
- REQUIRED**

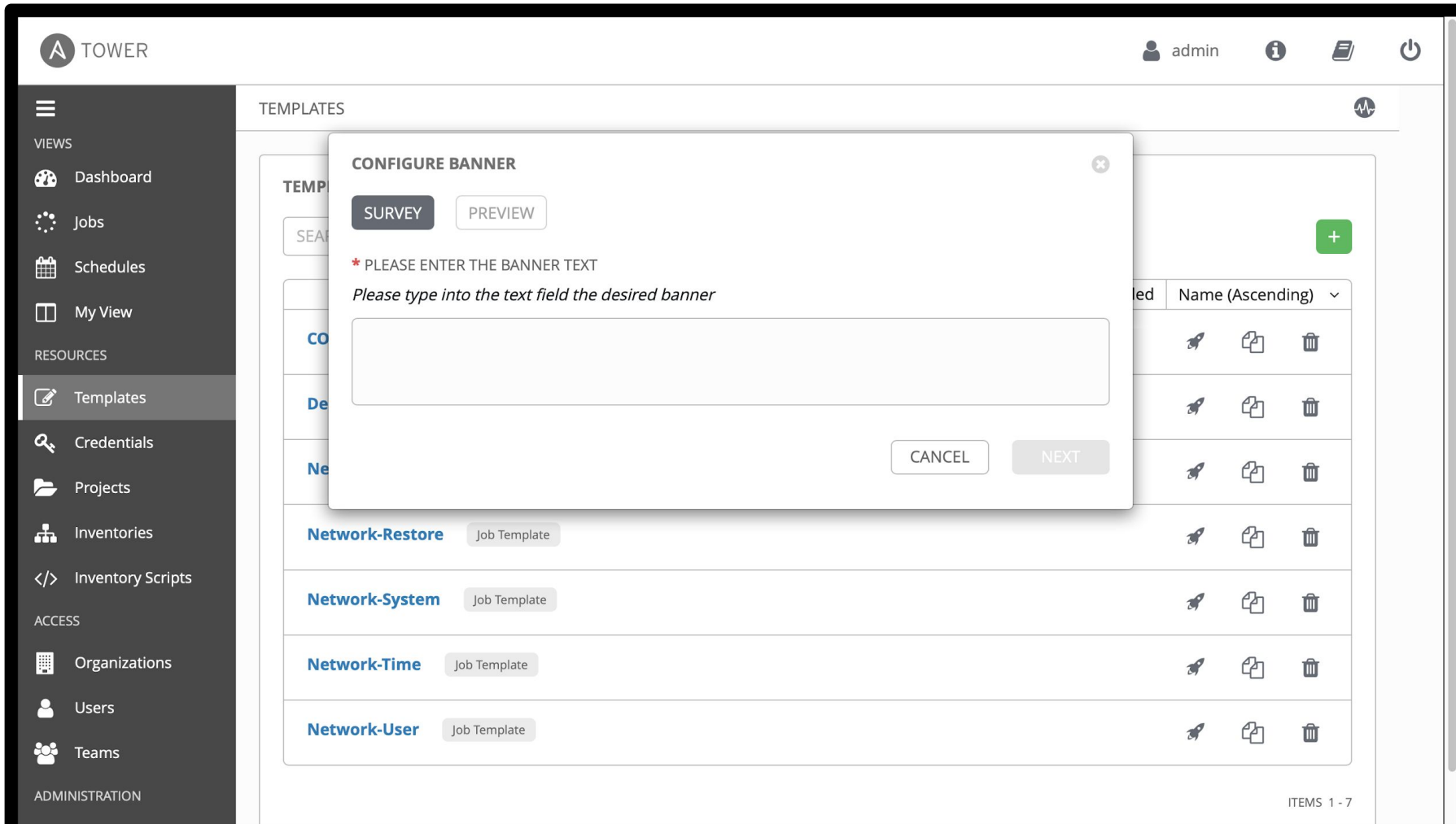
PREVIEW

- * PLEASE ENTER THE BANNER TEXT**: A heading for the preview section.
- Please type into the text field the desired banner*: A note indicating the expected content.
- A large, empty text area for previewing the banner text.
- Icons for editing (pencil) and deleting (trash) are visible next to the preview area.

At the bottom of the window, there are buttons for 'CLEAR', 'UPDATE', 'CANCEL', and 'OK'.

Using a Survey

When launching a job, the user will now be prompted with the Survey. The user can be required to fill out the Survey before the Job Template will execute.



The screenshot displays the Tower web interface. The top navigation bar includes the 'TOWER' logo, a user profile for 'admin', and several utility icons. A left-hand sidebar contains navigation categories: VIEWS (Dashboard, Jobs, Schedules, My View), RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts), ACCESS (Organizations, Users, Teams), and ADMINISTRATION. The main content area is titled 'TEMPLATES' and shows a list of job templates. A modal dialog box titled 'CONFIGURE BANNER' is open, featuring a 'SURVEY' button and a 'PREVIEW' button. The dialog contains the text: '* PLEASE ENTER THE BANNER TEXT' and 'Please type into the text field the desired banner', followed by a large text input field. At the bottom of the dialog are 'CANCEL' and 'NEXT' buttons. The background list of templates includes 'Network-Restore', 'Network-System', 'Network-Time', and 'Network-User', each with a 'Job Template' tag and action icons (run, copy, delete). The bottom right corner of the interface shows 'ITEMS 1 - 7'.



Red Hat Ansible Automation Platform

Demo Time

Exercise 2.4

Section 2.5

Topics Covered:

- Role based access control



Red Hat
Ansible Automation
Platform

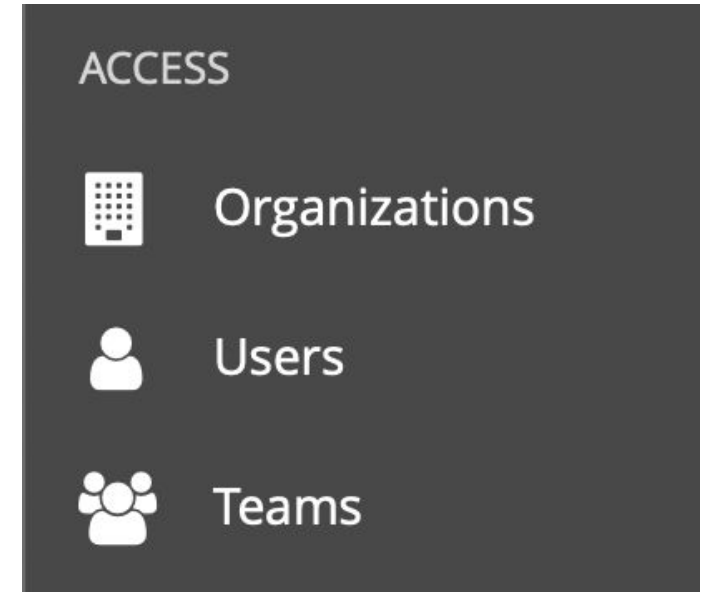
Role Based Access Control (RBAC)

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



in the left menu

The screenshot shows the Tower web interface. The top left corner has the "TOWER" logo. The top right corner shows the user "admin" and several utility icons. The left sidebar is a dark grey menu with categories: VIEWS (Dashboard, Jobs, Schedules, My View), RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts), ACCESS (Organizations, Users, Teams), and ADMINISTRATION. The "Organizations" menu item is highlighted. The main content area is titled "ORGANIZATIONS" and features a search bar, a "KEY" button, and a green "+" button. Below this are three organization cards, each with a title, edit/delete icons, and a grid of resource counts:

- Default** (3 items): 0 USERS, 0 TEAMS, 1 INVENTORIES, 1 PROJECTS, 1 JOB TEMPLATES, 0 ADMINS.
- REDHAT COMPUTE ORGANIZATION** (2 items): 0 USERS, 2 TEAMS, 0 INVENTORIES, 0 PROJECTS, 0 JOB TEMPLATES, 0 ADMINS.
- REDHAT NETWORK ORGANIZATION** (1 item): 2 USERS, 2 TEAMS, 1 INVENTORIES, 1 PROJECTS, 6 JOB TEMPLATES, 1 ADMINS.

At the bottom right of the main content area, it says "ITEMS 1 - 3".

Viewing Teams

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



in the left menu

The screenshot shows the Tower web interface. The top navigation bar includes the "TOWER" logo, a user profile for "admin", and icons for information, a document, and a power button. The left sidebar menu is expanded to show the "Teams" option, which is highlighted. The main content area is titled "TEAMS" and displays a table of team information. The table has columns for "NAME", "ORGANIZATION", and "ACTIONS". There are four teams listed: "Compute T1", "Compute T2", "Netadmin", and "Netops". Each team is associated with either "REDHAT COMPUTE ORGANIZATION" or "REDHAT NETWORK ORGANIZATION". The "ACTIONS" column contains edit and delete icons for each team. A search bar and a "KEY" input field are located above the table. A green "+" button is in the top right corner of the table area. The bottom right of the table area shows "ITEMS 1 - 4".

NAME	ORGANIZATION	ACTIONS
Compute T1	REDHAT COMPUTE ORGANIZATION	
Compute T2	REDHAT COMPUTE ORGANIZATION	
Netadmin	REDHAT NETWORK ORGANIZATION	
Netops	REDHAT NETWORK ORGANIZATION	

Viewing Users

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



in the left menu

The screenshot shows the Tower web interface. The top left corner has the "TOWER" logo. The top right corner shows the user "admin" and several utility icons. A dark grey sidebar on the left contains a menu with categories: VIEWS (Dashboard, Jobs, Schedules, My View), RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts), and ACCESS (Organizations, Users, Teams). The "Users" menu item is highlighted. The main content area is titled "USERS" and shows a list of 8 users. At the top of the list is a search bar with the text "SEARCH" and a magnifying glass icon, and a "KEY" button. A green "+" button is in the top right of the list area. The table below has columns for USERNAME, FIRST NAME, LAST NAME, and ACTIONS. The users listed are: admin, bbelcher (Bob Belcher), gbelcher (Gene Belcher), lbelcher (Louise Belcher), libelcher (Linda Belcher), network-admin (Larry Niven), network-operator (Issac Assimov), and tbelcher (Tina Belcher). Each user row has edit and delete icons in the ACTIONS column. At the bottom right of the list area, it says "ITEMS 1 - 8".

USERNAME	FIRST NAME	LAST NAME	ACTIONS
admin			
bbelcher	Bob	Belcher	
gbelcher	Gene	Belcher	
lbelcher	Louise	Belcher	
libelcher	Linda	Belcher	
network-admin	Larry	Niven	
network-operator	Issac	Assimov	
tbelcher	Tina	Belcher	



Red Hat Ansible Automation Platform

Demo Time

Exercise 2.5

Section 2.6

Topics Covered:

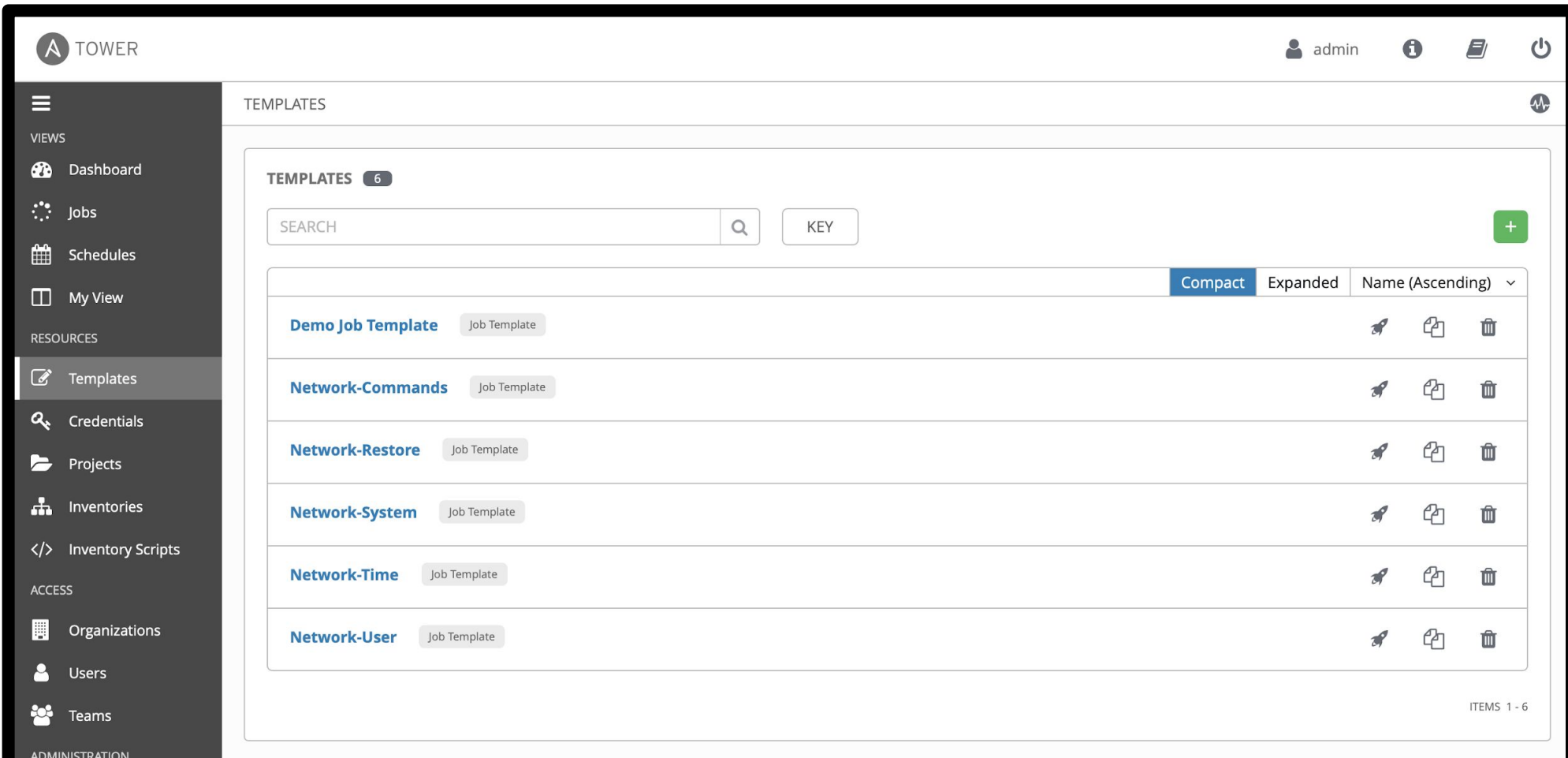
- Workflows





















Red Hat
Ansible Automation
Platform

Workflows

Workflows can be found alongside Job Templates by clicking the **Templates**  button under the *RESOURCES* section on the left menu.



The screenshot displays the Tower web interface. The top navigation bar includes the Tower logo, user name 'admin', and system status icons. The left sidebar is categorized into VIEWS, RESOURCES, and ADMINISTRATION. The 'RESOURCES' section has 'Templates' selected. The main content area is titled 'TEMPLATES' and shows a list of 6 job templates. The list is displayed in 'Compact' view and is sorted by 'Name (Ascending)'. Each template entry includes a name, a 'Job Template' label, and three action icons: a rocket (launch), a document (copy), and a trash can (delete).

TEMPLATES 6		SEARCH	KEY	
				Compact Expanded Name (Ascending) v
Demo Job Template	Job Template			  
Network-Commands	Job Template			  
Network-Restore	Job Template			  
Network-System	Job Template			  
Network-Time	Job Template			  
Network-User	Job Template			  

ITEMS 1 - 6

Adding a new Workflow Template

To add a new **Workflow** click on the green + button



This time select the **Workflow Template**

The screenshot shows the Tower web interface. The top navigation bar includes the Tower logo, the user name 'admin', and several utility icons. The left sidebar contains a menu with categories: VIEWS (Dashboard, Jobs, Schedules, My View), RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts), and ACCESS (Organizations, Users). The main content area is titled 'TEMPLATES' and shows a list of templates. A search bar and a 'KEY' button are at the top of the list. A dropdown menu is open, showing 'Job Template' and 'Workflow Template' options. The 'Workflow Template' option is highlighted. The list of templates includes: Backup network configurations (Job Template), Configure Banner (Job Template), Demo Job Template (Job Template), Network-Commands (Job Template), Network-Restore (Job Template), and Network-System (Job Template). Each template row has a 'Job Template' label and a set of icons for actions like run, copy, and delete.

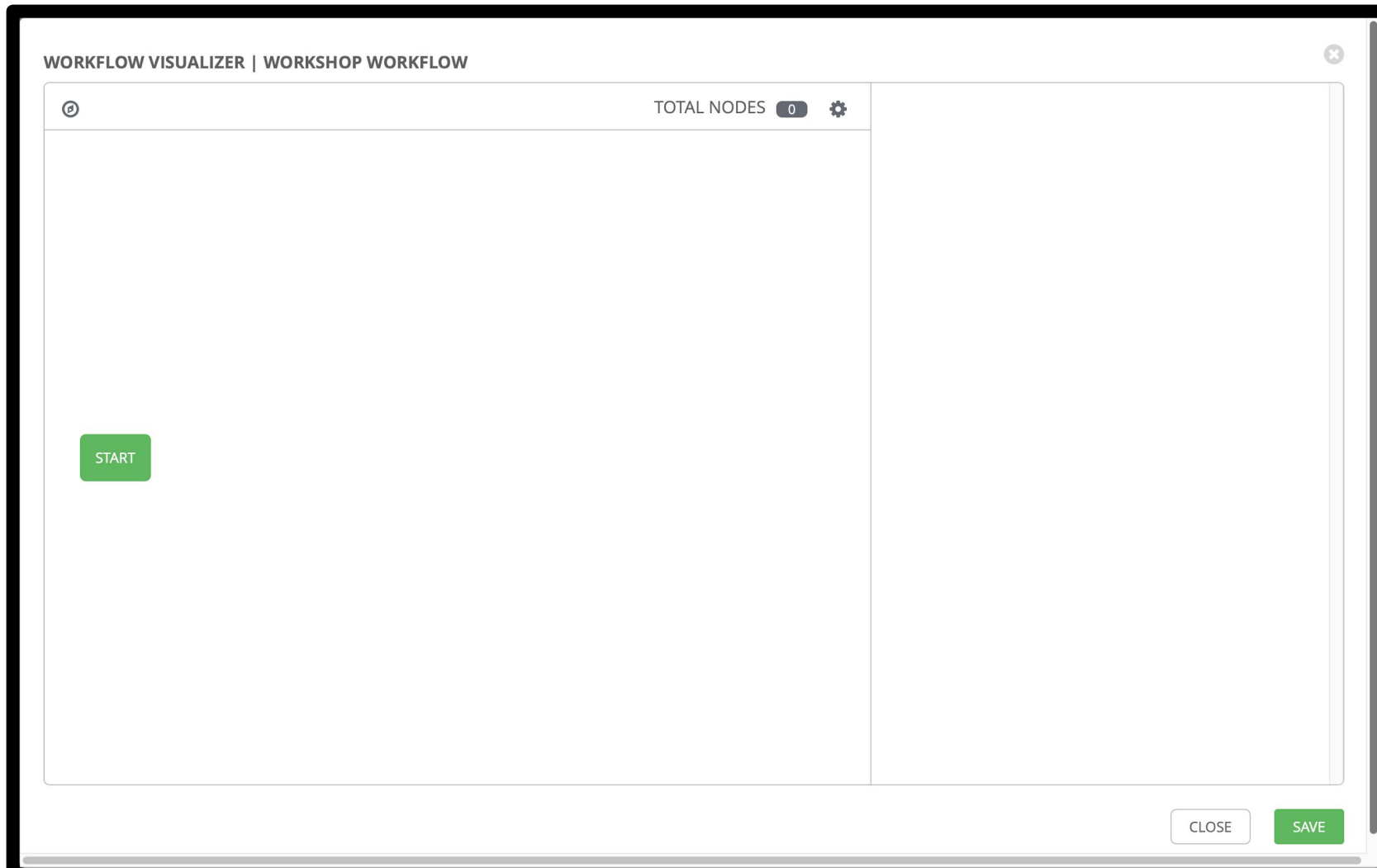
Creating the Workflow

Fill out the required parameters and click **SAVE**. As soon as the Workflow Template is saved the WORKFLOW VISUALIZER will open.

The screenshot displays the Tower web interface for configuring a 'WORKSHOP WORKFLOW'. The top navigation bar shows the 'TOWER' logo, the user 'admin', and various utility icons. The left sidebar contains navigation options under 'VIEWS', 'RESOURCES', and 'ACCESS'. The main content area is titled 'TEMPLATES / WORKSHOP WORKFLOW' and features a tabbed interface with buttons for 'DETAILS', 'PERMISSIONS', 'NOTIFICATIONS', 'COMPLETED JOBS', 'SCHEDULES', and 'ADD SURVEY'. The 'WORKFLOW VISUALIZER' button is highlighted with a red box. Below the tabs, there are input fields for 'NAME' (WORKSHOP WORKFLOW), 'DESCRIPTION', 'ORGANIZATION' (Default), 'INVENTORY' (Workshop Inventory), 'LABELS', and 'OPTIONS' (ENABLE CONCURRENT JOBS). There is also a section for 'EXTRA VARIABLES' with tabs for 'YAML' and 'JSON'.

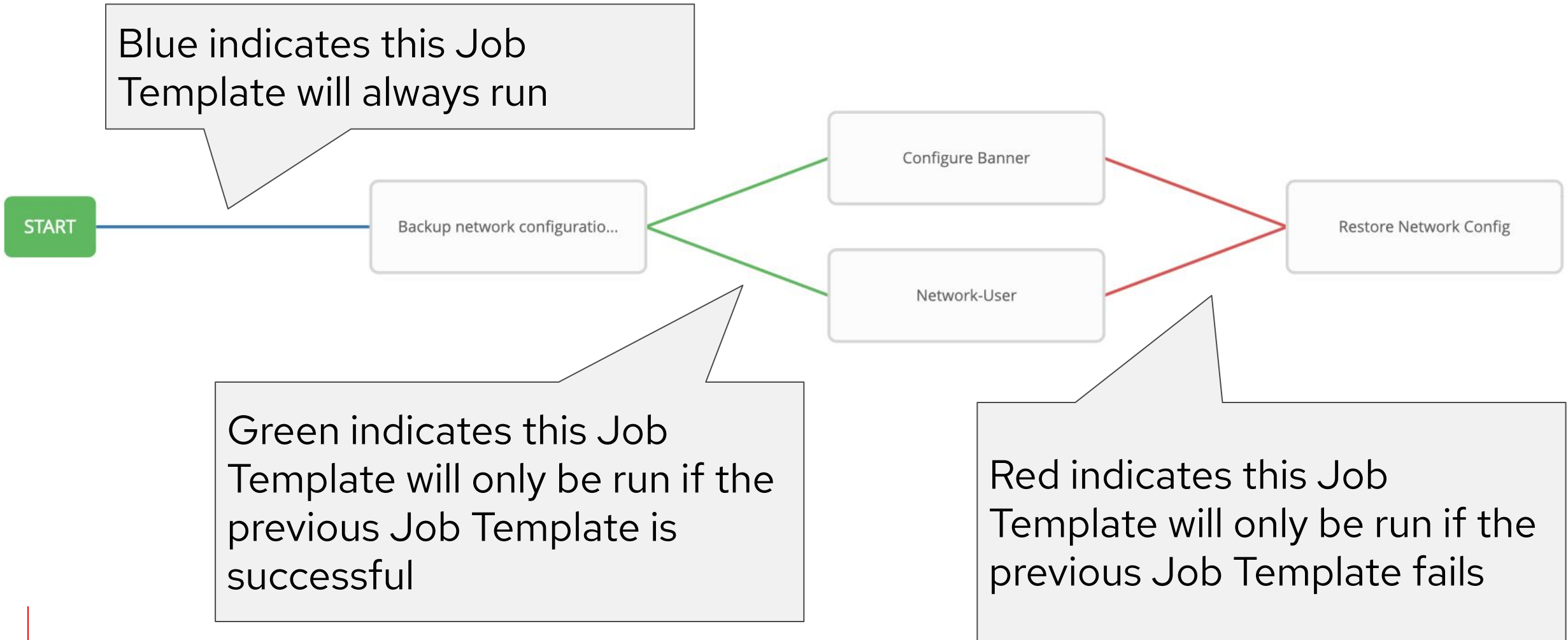
Workflow Visualizer

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

Exercise 2.6

Section 2.7

Topics Covered:

- Wrap-up



Red Hat
Ansible Automation
Platform



Red Hat Ansible Automation Platform

Demo Time
Exercise 2.7

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



34%

more efficient IT infrastructure teams



59%

faster to deploy new IT resources



76%

faster to full productivity, new hires already trained

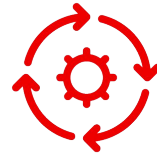
Improve productivity with training in Ansible automation

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.



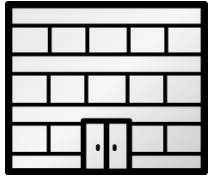
59% faster

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



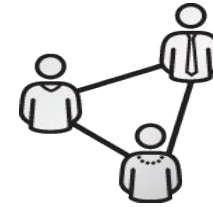
Onsite Training

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

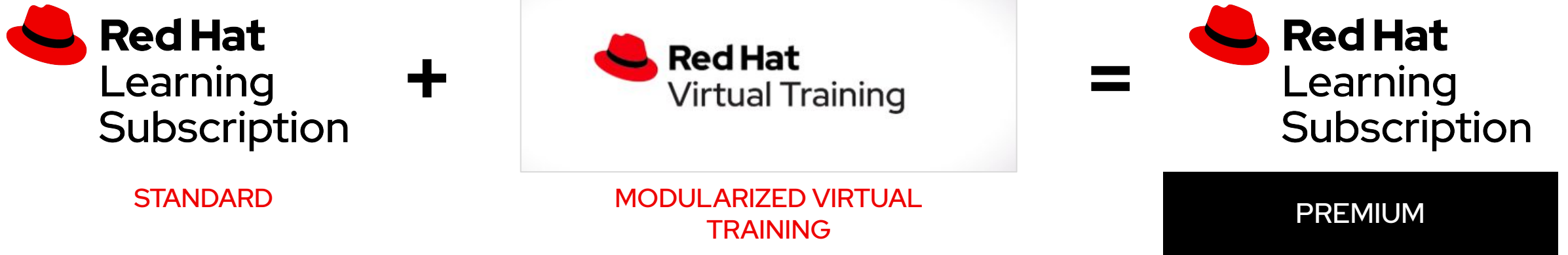
- 24x7 access globally, available offline
- Self-paced, unlimited access to Red Hat courses
- Access to content currently in development
- Updated content pushed as early releases
- Content spanning the entire Red Hat product portfolio
- Early access to completed chapters of courses



Red Hat Learning Subscription

Red Hat Learning Subscription Evolution

Introducing a Premium subscription tier



Ansible Curriculum

DO007	Ansible Essentials: Simplicity in Automation Technical Overview (FREE!)	
-------	---	--

Network Admins

DO457	Ansible for Network Automation	
-------	--------------------------------	--

Linux Admins

RH294	Red Hat System Administration III: Linux Automation	
-------	---	--

EX294	Red Hat Certified Engineer exam	
-------	---------------------------------	--

RH358	Red Hat Services Management and Automation	
-------	--	--

Windows Admins

DO417	Microsoft Windows Automation with Red Hat Ansible	
-------	---	--


DevOps Engineers


DO447	Advanced Automation: Ansible Best Practices	
-------	---	--


EX447	Red Hat Certified Specialist in Ansible Automation Exam	
-------	---	--




Thank you

 [linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)

 [youtube.com/AnsibleAutomation](https://www.youtube.com/AnsibleAutomation)

 [facebook.com/ansibleautomation](https://www.facebook.com/ansibleautomation)

 twitter.com/ansible

 github.com/ansible



Thank you for viewing this Red Hat workshop! 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:

carah.io/redhatsled



For additional Red Hat solutions:

carah.io/RedHatPortfolio



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

carah.io/RedHatContracts



For upcoming events:

carah.io/RedHatEvents



For additional Open Source solutions:

carah.io/OpenSourceSolutions



To set up a meeting:

redhat@carahsoft.com or 877-RHAT-GOV