Red Hat
Ansible Automation
Platform

Ansible Windows Workshop

Microlearning Edition

Red Hat

Day 1

- Administration
- Know each other
- What you will learn
- Intro to Ansible
- Exercise 1

Red Hat
Ansible Automation
Platform

Red Hat

Administration

- Explain the format
- Clarify group rules (if any)
- Resources (exercises, decks, scripts, etc.)
- Feedback form
- Lab

Know each other

- Introduce yourself
- Your experience with Automation and Ansible
- Your goals & expectations
- Your use case
- Your position

What you will learn

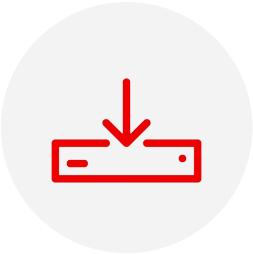
- Introduction to Ansible Automation
- How Ansible works for Windows Automation
- Understanding Ansible modules and playbooks
- Using Ansible Tower to scale automation to the enterprise
- Reusing automation with Ansible Roles

Red Hat

INTRODUCTION TO ANSIBLE

Red Hat

Growth by the numbers

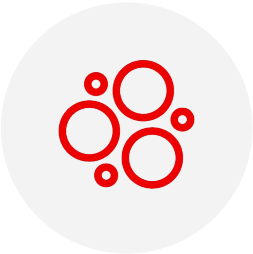


2M

downloads per month

4K

modules



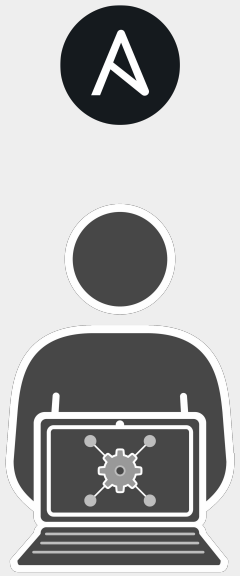
7th

of 96M projects on GitHub by contributors

4M+

systems managed by Red Hat

Source:



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

Red Hat

Teams are automating...

Lines Of Business

Network

Security

Operations

Developers

Infrastructure

Why Ansible?

Simple

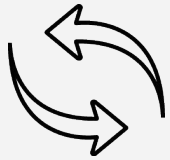
Human readable automation

No special coding skills needed

Tasks executed in order

Usable by every team

Get productive quickly



Powerful

App deployment

Configuration management

Workflow orchestration

Network automation

Orchestrate the app lifecycle

Agentless

Agentless architecture

Uses OpenSSH & WinRM

No agents to exploit or update

Get started immediately

More efficient & more secure

Red Hat

Red Hat Ansible Automation Platform

Network | Lines of business | Security

Operations | Infrastructure | Developers

Engage — **Ansible Hosted Services:** Engage users with an automation focused experience

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

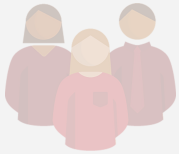
Create — **Ansible Engine:** Universal language of automation

Fueled by an open source community

Red Hat

Red Hat Ansible Automation Platform

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



Engage

Ansible SaaS: Engage users with an automation focused experience

Scale

Control
Web UI and API

Delegation
Role Based Access Controls

Scale
Scalable Execution Capacity

Create

Ansible Engine: Universal language of automation

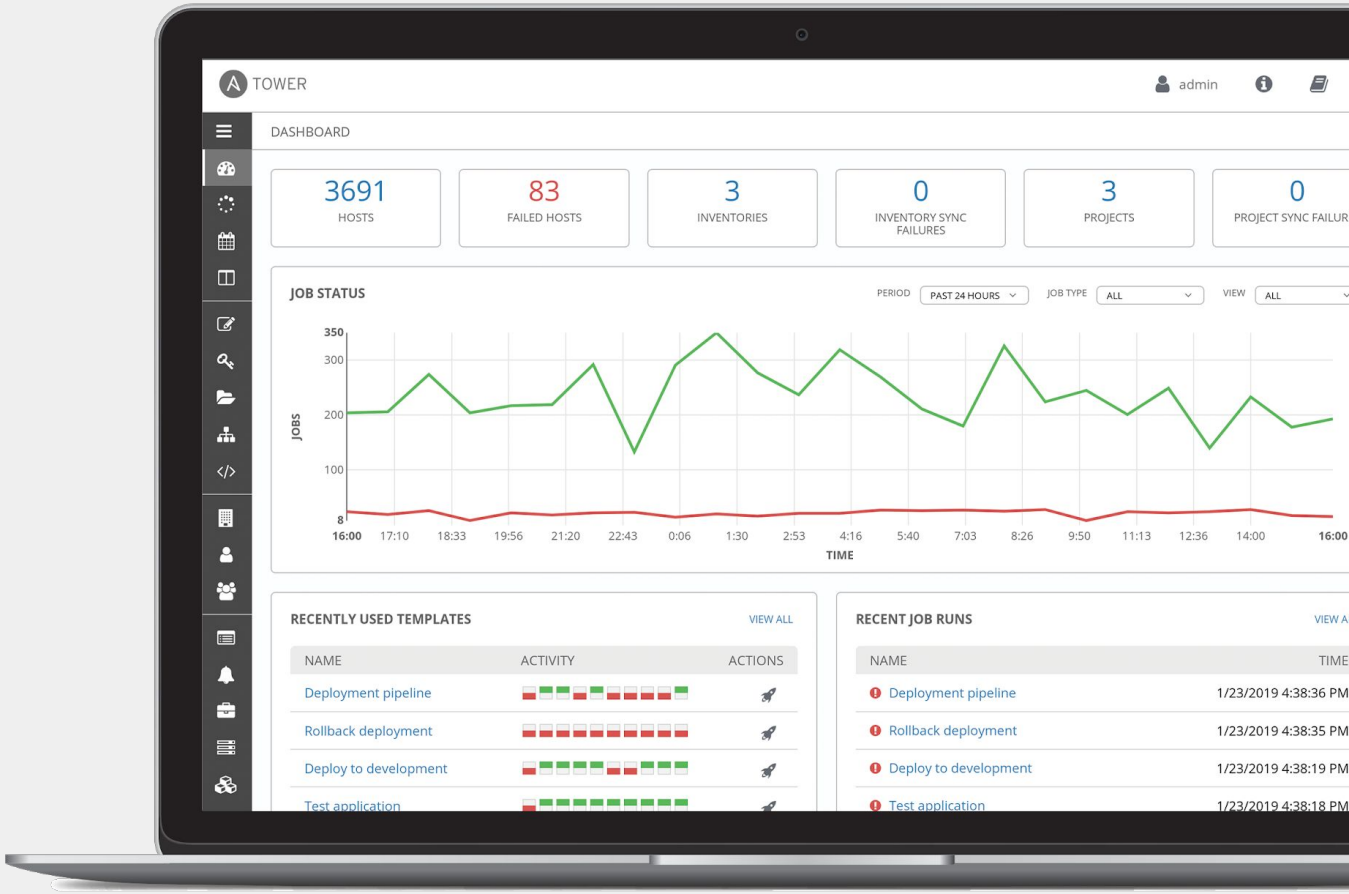
Fueled by an open source community

Red Hat

What is Ansible Tower?

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

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



Red Hat

Exercise 1

- Log into your Ansible Tower instance
- Review & Configuring Ansible Tower
- What are ad-hoc commands
- Run from
  - Command line
  - Ansible Tower

Red Hat
Ansible Automation
Platform

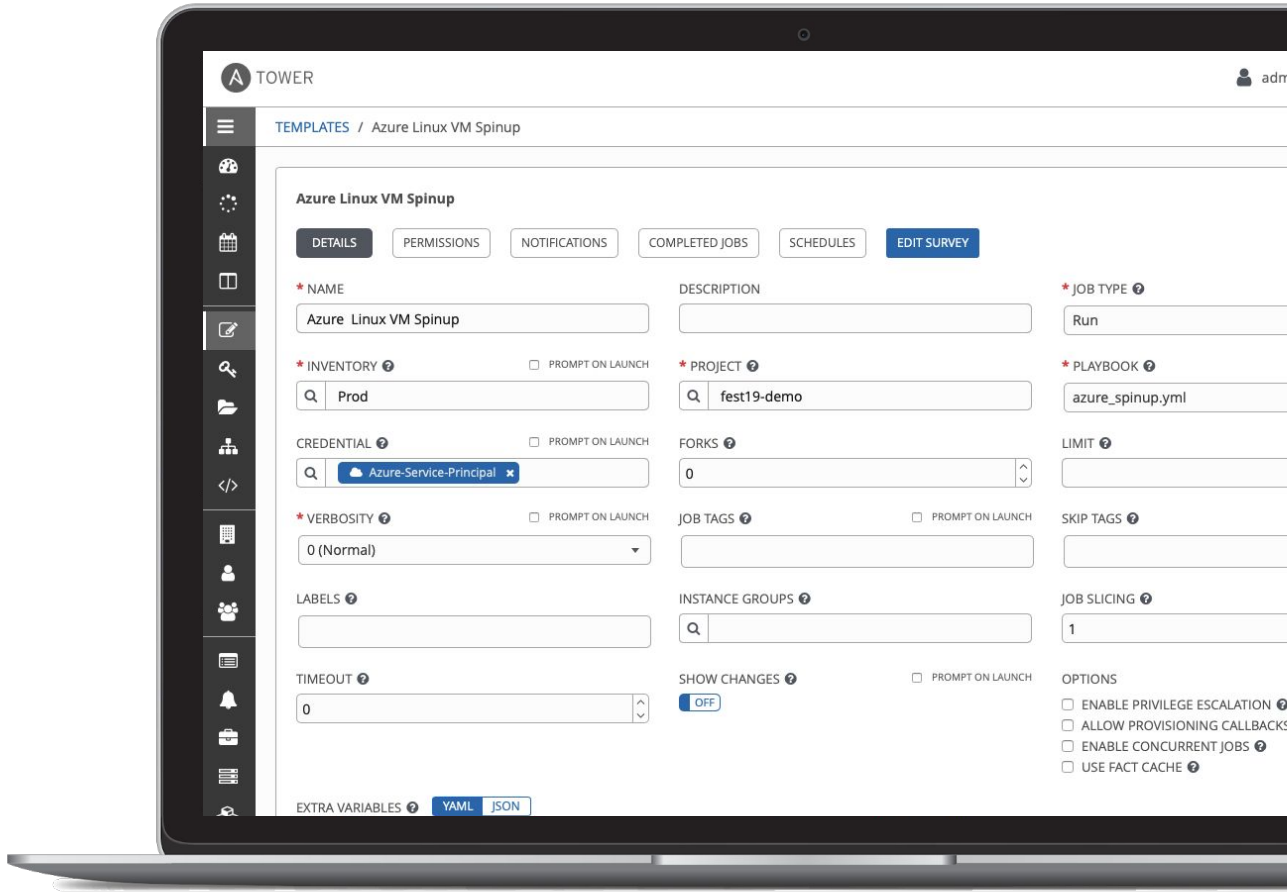
Job Templates

Everything in Ansible Tower revolves around the concept of a **Job Template**. A job template encapsulates all arguments that you would specify to run a playbook.

Job templates encourages the reuse of playbooks and collaboration between teams.

A **Job Template** requires:

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

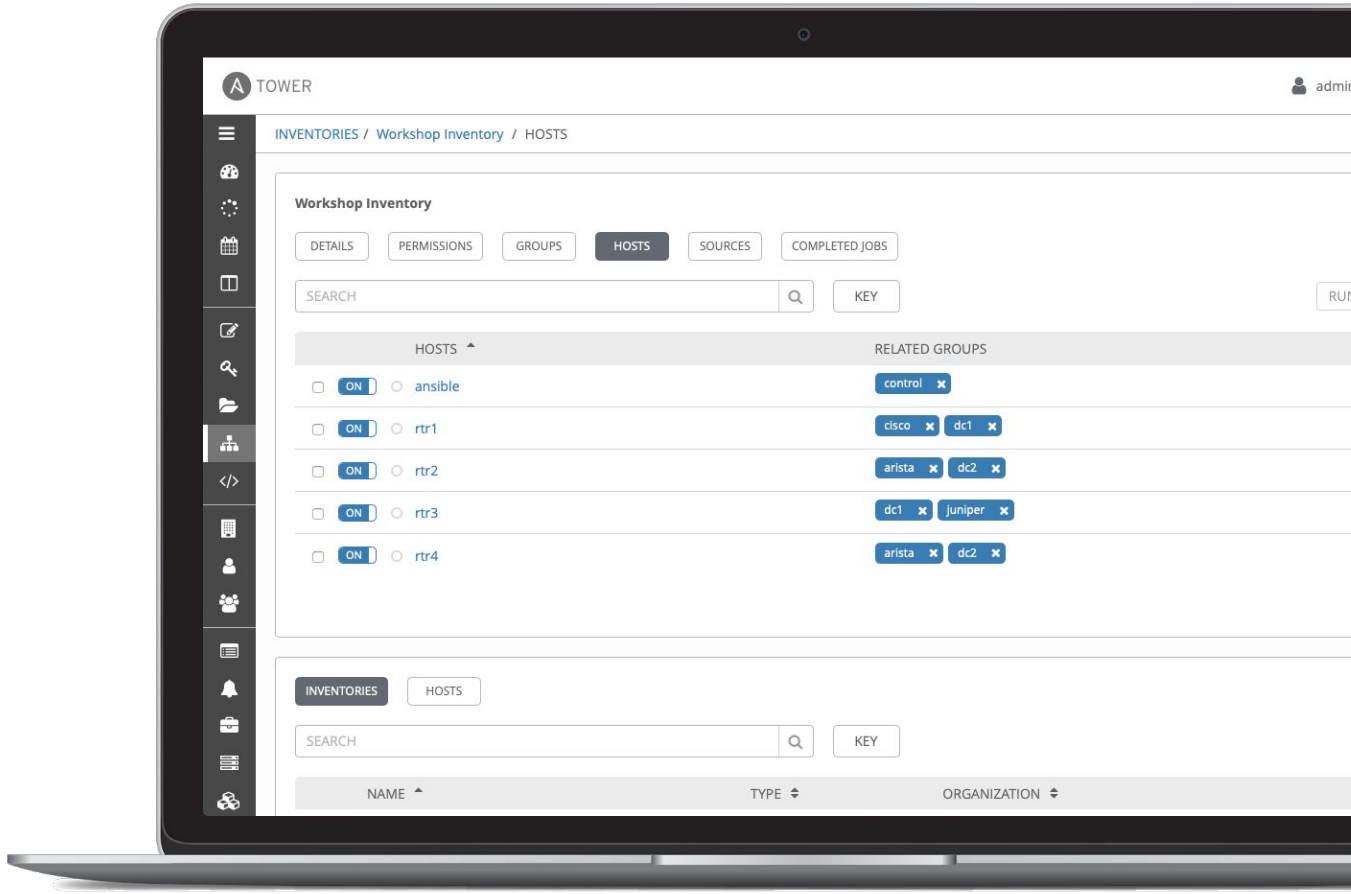


Red Hat

Inventory

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

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

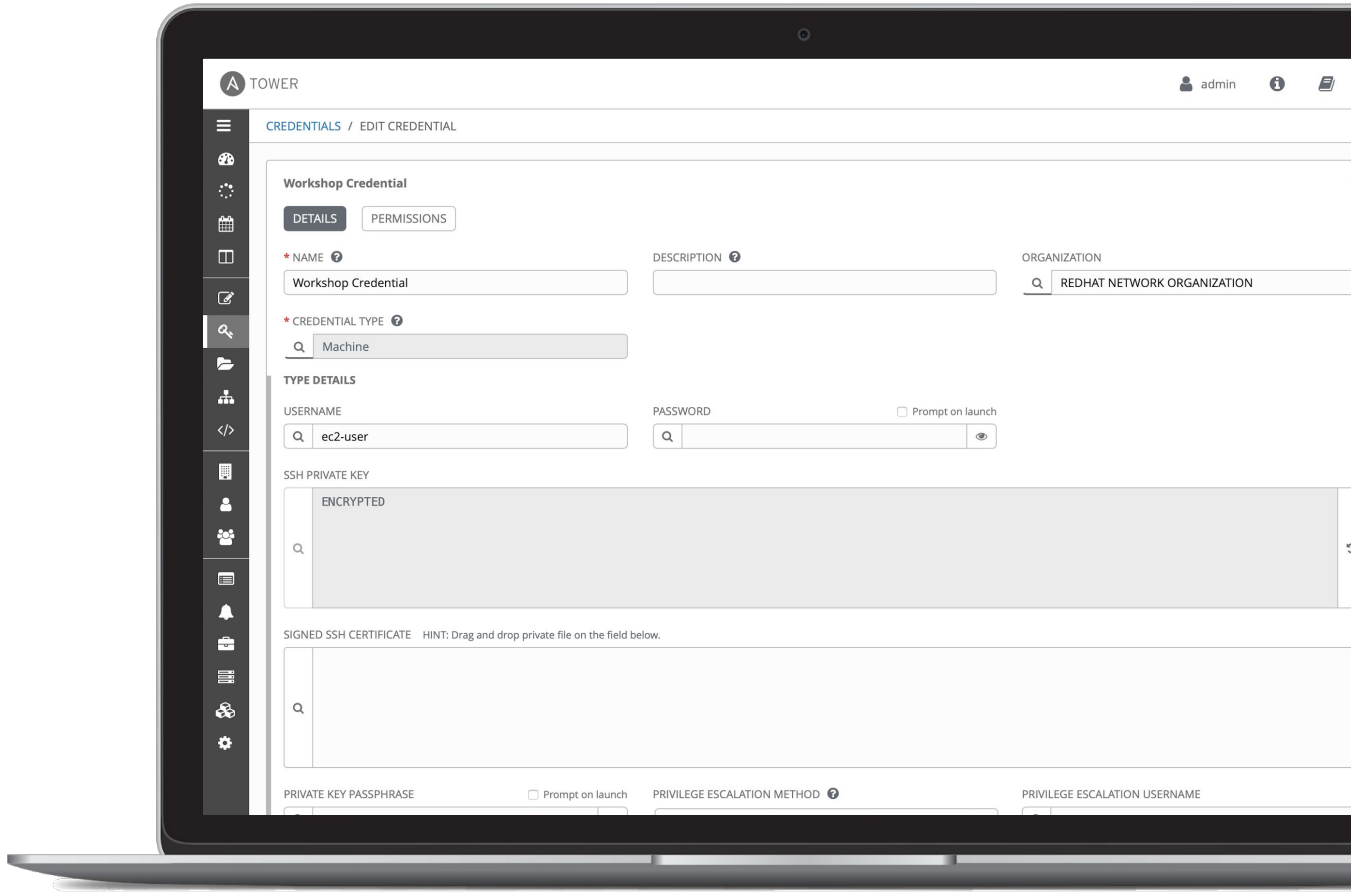


Red Hat

Credentials

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

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



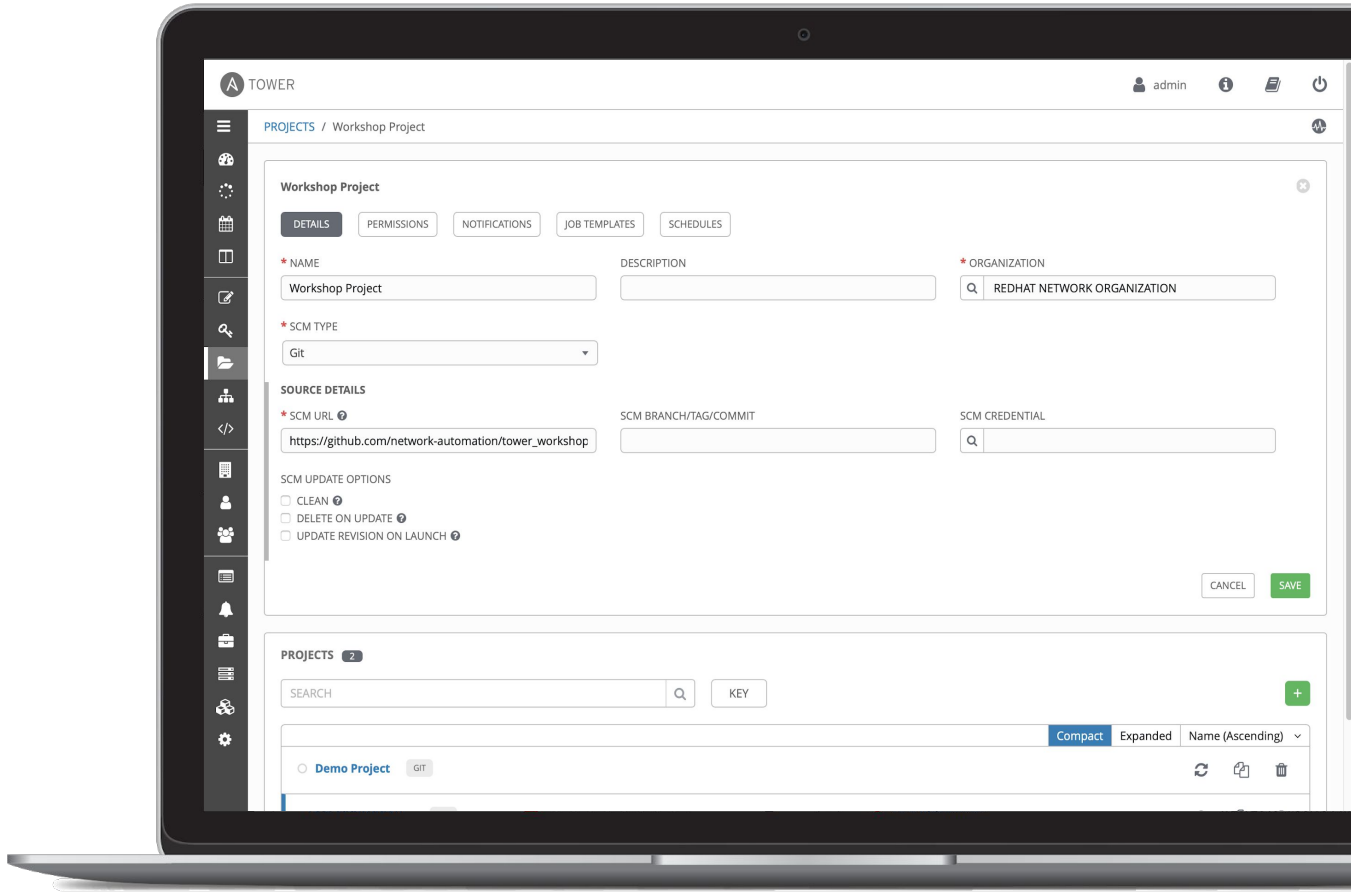
Centralized management of various credentials allows end users to leverage a secret without ever exposing that secret to them.

Red Hat

Project

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

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



Red Hat

Ad-hoc Commands

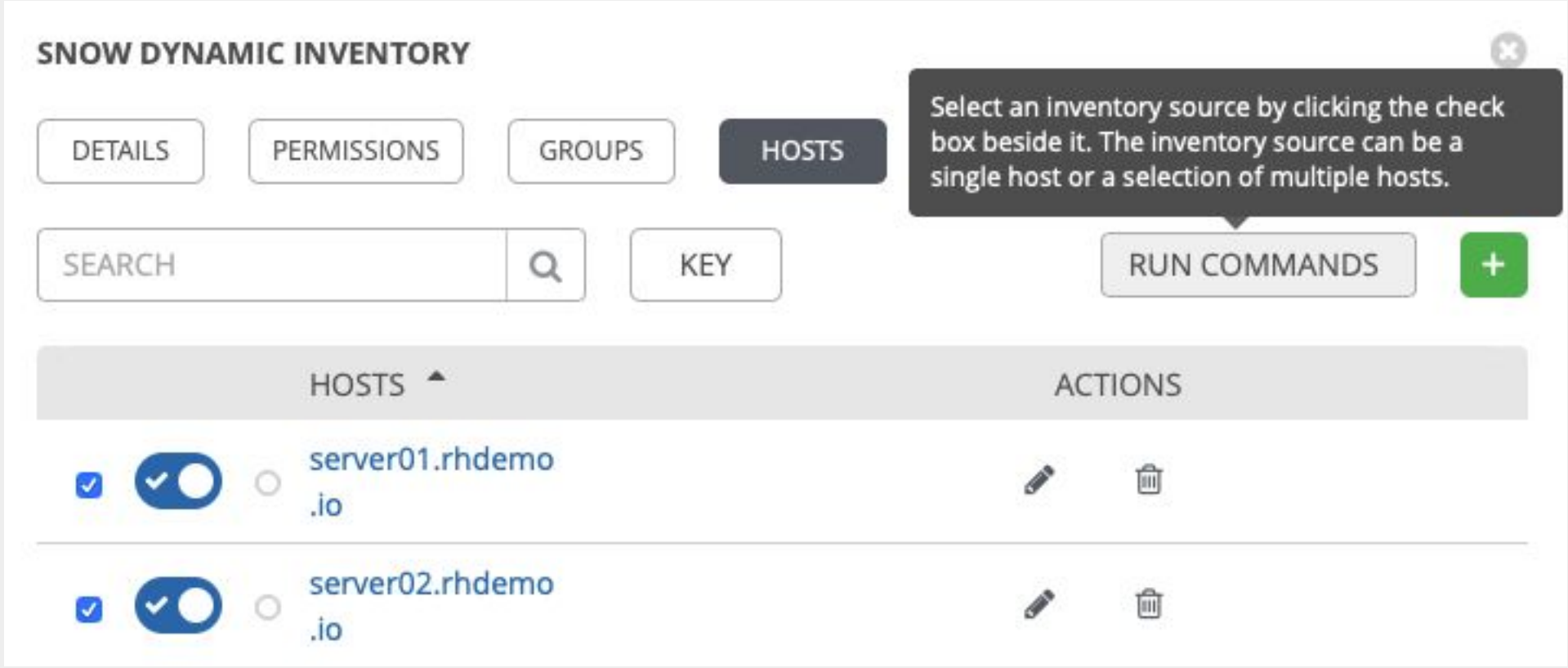
An ad-hoc command is a single Ansible task to perform quickly, but don't want to save for later.

```
# check all my inventory hosts are ready to be
# managed by Ansible
$ ansible all -m ping

# collect and display the discovered facts
# for the localhost
$ ansible localhost -m setup

# run the uptime command on all hosts in the
# web group
$ ansible web -m command -a "uptime"
```

Ad-hoc Commands from Tower



Red Hat

Bonus

- Review your lab environment
- Review ansible https://docs.ansible.com
- Execute more adhoc commands
- https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html
- More on Tower: https://www.youtube.com/watch?v=wZ_mh4-4HPY

Red Hat
Ansible Automation
Platform

Red Hat

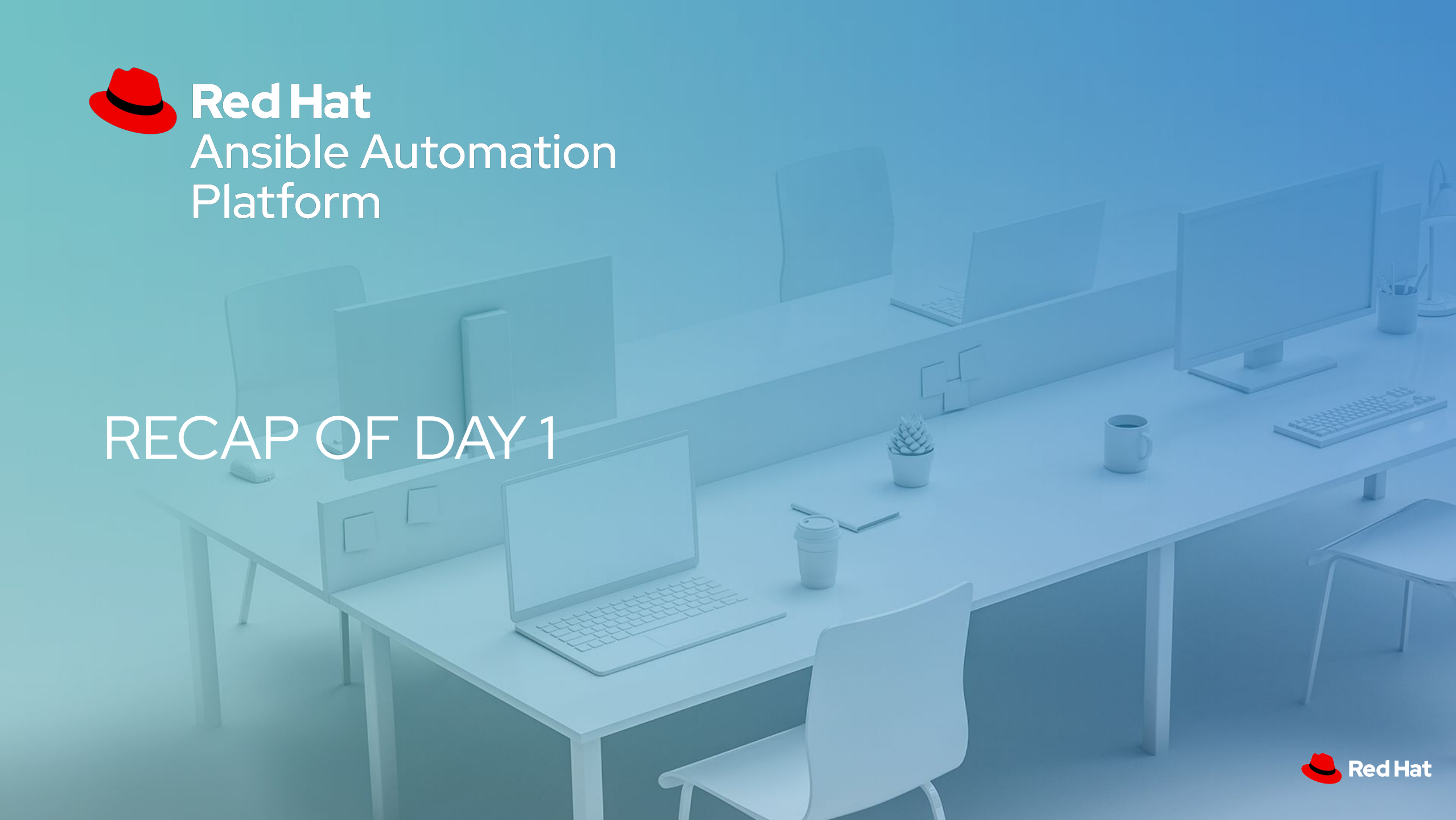
Day 2

Topics Covered:

- Recap of Day 1
- What is a module
- What is a playbook
- What are Windows Ansible modules
- Exercises 3, 4

Red Hat
Ansible Automation
Platform

Red Hat



Red Hat
Ansible Automation
Platform

RECAP OF DAY 1

Red Hat

What can I do using Ansible?

Automate the deployment and management of your entire IT footprint.

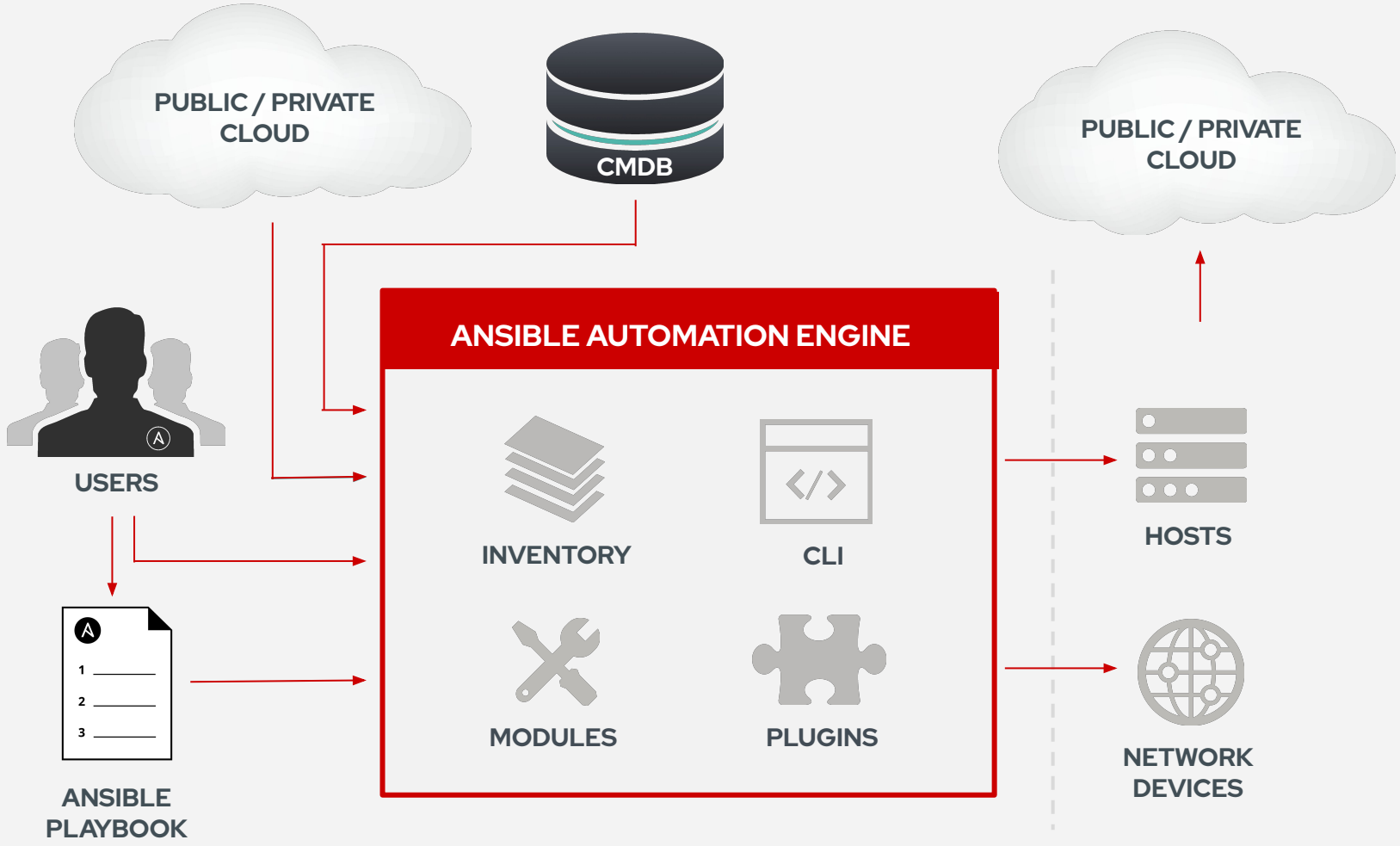
Do this...

Orchestration	Configuration Management	Application Deployment	Provisioning	Continuous Delivery	Security and Compliance

On these...

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

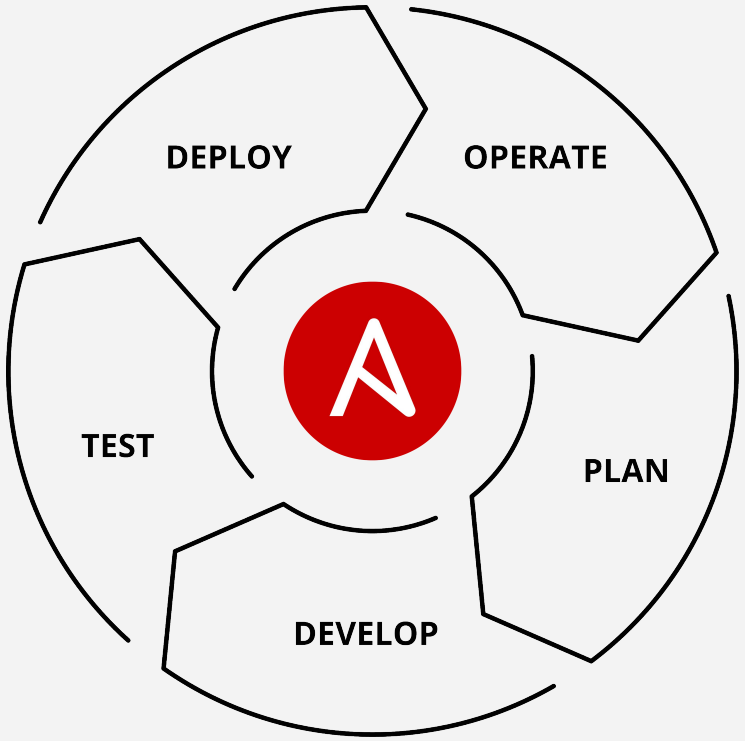
Red Hat


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

Red Hat

WHAT CAN I DO USING ANSIBLE FOR WINDOWS

Native Windows support uses PowerShell remoting to manage Windows in the same Ansible agentless way



- Install and uninstall MSIs
- Gather facts on Windows hosts
- Enable and disable Windows features
- Start, stop, and manage Windows Services
- Create and Manage local users and groups
- Manage Windows packages via Chocolatey package manager
- Manage and install Windows updates
- Fetch files from remote sites
- Push and execute any Powershell scripts

Red Hat

Windows modules

Ansible modules for Windows automation typically begin with win_*

win_copy - Copies files to remote locations on windows hosts

win_service - Manage and query Windows services

win_domain - Ensures the existence of a Windows domain

win_reboot - Reboot a windows machine

win_regedit - win_regedit – Add, change, or remove registry keys and values

win_ping - A windows version of the classic ping module

win_dsc - Invokes a PowerShell DSC configuration

win_acl - Set file/directory/registry permissions for a system user or group

Playbooks

Topics Covered:

- Variables
  - Facts
  - Precedence
- Tasks
  - Handlers

Red Hat
Ansible Automation
Platform

Variables

Ansible can work with metadata from various sources and manage their context in the form of variables.

- Command line parameters
- Plays and tasks
- Files
- Inventory
- Discovered facts
- Roles

Discovered facts

Facts are bits of information derived from examining a host systems that are stored as variables for later use in a play.

```
$ ansible localhost -m setup
localhost | success >> {
  "ansible_facts": {
      "ansible_default_ipv4": {
          "address": "192.168.1.37",
          "alias": "wlan0",
          "gateway": "192.168.1.1",
          "interface": "wlan0",
          "macaddress": "c4:85:08:3b:a9:16",
          "mtu": 1500,
          "netmask": "255.255.255.0",
          "network": "192.168.1.0",
          "type": "ether"
      },
```

Variable Precedence

The order in which the same variable from different sources will override each other.

1. command line values (eg “-u user”)
2. role defaults [1]
3. inventory file or script group vars [2]
4. **inventory group_vars/all** [3]
5. playbook group_vars/all [3]
6. **inventory group_vars/*** [3]
7. playbook group_vars/* [3]
8. inventory file or script host vars [2]
9. **inventory host_vars/*** [3]
10. playbook host_vars/* [3]
11. host facts / cached set_facts [4]
12. play vars
13. play vars_prompt
14. play vars_files
15. role vars (defined in role/vars/main.yml)
16. block vars (only for tasks in block)
17. task vars (only for the task)
18. include_vars
19. set_facts / registered vars
20. role (and include_role) params
21. include params
22. extra vars (**always win precedence**)

Tasks

Tasks are the application of a module to perform a specific unit of work.

- **win_file**: A directory should exist
- **win_package**: A package should be installed
- **win_service**: A service should be running
- **win_template**: Render a configuration file from a template
- **win_get_url**: Fetch an archive file from a URL
- **win_copy**: Copy a file from your repository or a remote source

Tasks

```
tasks:
- name: Ensure IIS Server is present
  win_feature:
    name: Web-Server
    state: present

- name: Ensure latest index.html file is present
  win_copy:
    src: files/index.html
    dest: c:\www\

- name: Restart IIS
  win_service:
    name: IIS Admin Service
    state: restarted
```

Handler Tasks

Handlers are special tasks that run at the end of a play if notified by another task when a change occurs.

> *If a package gets installed or updated, notify a service restart task that it needs to run.*

Handler Tasks

```
tasks:
- name: Ensure IIS Server is present
  win_feature:
    name: Web-Server
    state: present
  notify: Restart IIS

- name: Ensure latest index.html file is present
  win_copy:
    src: files/index.html
    dest: c:\www\

handlers:
- name: Restart IIS
  win_service:
    name: IIS Admin Service
    state: restarted
```

Plays and playbooks

Plays are ordered sets of tasks to execute against host selections from your inventory. A playbook is a file containing one or more plays.

```
---
- name: Ensure IIS is installed and started
  hosts: web
  become: yes
  vars:
    service_name: IIS Admin Service

  tasks:
  - name: Ensure IIS Server is present
    win_feature:
      name: Web-Server
      state: present

  - name: Ensure latest index.html file is present
    win_copy:
      src: files/index.html
      dest: c:\www\

  - name: Ensure IIS is started
    win_service:
      name: "{{ server_name }}"
      state: started
```

Meaningful names

```
---
- name: Ensure IIS is installed and started
  hosts: web
  become: yes
  vars:
    service_name: IIS Admin Service

  tasks:
  - name: Ensure IIS Server is present
    win_feature:
      name: Web-Server
      state: present

  - name: Ensure latest index.html file is present
    win_copy:
      src: files/index.html
      dest: c:\www\

  - name: Ensure IIS is started
    win_service:
      name: "{{ server_name }}"
      state: started
```

Red Hat
Ansible Automation
Platform

Exercise 3 & 4

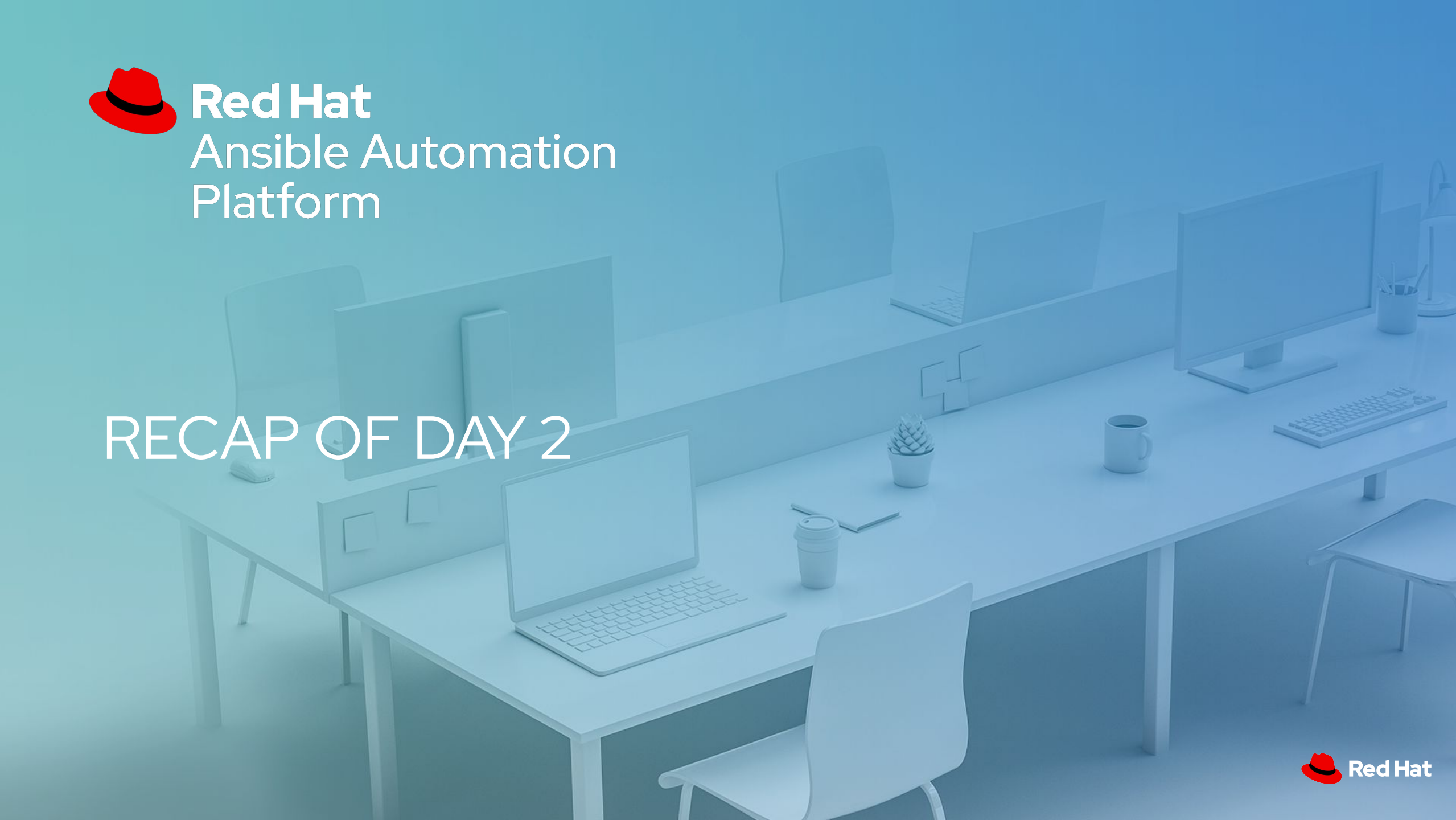
Day 3 : Advanced playbooks

Topics Covered:

- Templates
- Loops, Conditionals
- Tags, Blocks
- Exercise 5

Red Hat
Ansible Automation
Platform

Red Hat



Red Hat
Ansible Automation
Platform

RECAP OF DAY 2

Red Hat

Doing more with playbooks

Here are some more essential playbook features that you can apply:

- Templates
- Loops
- Conditionals
- Tags
- Blocks

Doing more with playbooks: **Templates**

Ansible embeds the Jinja2 templating engine that can be used to dynamically:

- Set and modify play variables
- Conditional logic
- Generate files such as configurations from variables

Doing more with playbooks: **Loops**

Loops can do one task on multiple things, such as create a lot of users, install a lot of packages, or repeat a polling step until a certain result is reached.

```
- name: Ensure IIS Server is present
  win_feature:
    name: "{{ item }}"
    state: present
  loop:
  - Web-Server
  - NET-Framework-Core
```

Doing more with playbooks: **Conditionals**

Ansible supports the conditional execution of a task based on the run-time evaluation of variable, fact, or previous task result.

```
- name: Ensure IIS Server is present
  win_feature:
    name: Web-Server
    state: present
  when: ansible_os_family == "Windows"
```

Doing more with playbooks: **Tags**

Tags are useful to be able to run a subset of a playbook on-demand.

```
- name: Ensure IIS Server is present
  win_feature:
    name: "{{ item }}"
    state: present
  with_items:
  - Web-Server
  - NET-Framework-Core
  tags:
    - packages

- name: Copy web.config template to Server
  win_template:
    src: templates/web.config.j2
    dest: C:\inetpub\wwwroot\web.config
  tags:
    - configuration
```

Red Hat

Doing more with playbooks: **Blocks**

Blocks cut down on repetitive task directives, allow for logical grouping of tasks and even in play error handling.

```
- block:
  - name: Ensure IIS Server is present
    win_feature:
      name: "{{ item }}"
      state: present
    with_items:
    - Web-Server

  - name: Copy web.config template to Server
    win_template:
      src: templates/web.config.j2
      dest: C:\inetpub\wwwroot\web.config

  when: ansible_os_family == "Windows"
```

Exercise 5

- Practical Playbook Development

Red Hat
Ansible Automation
Platform

Red Hat

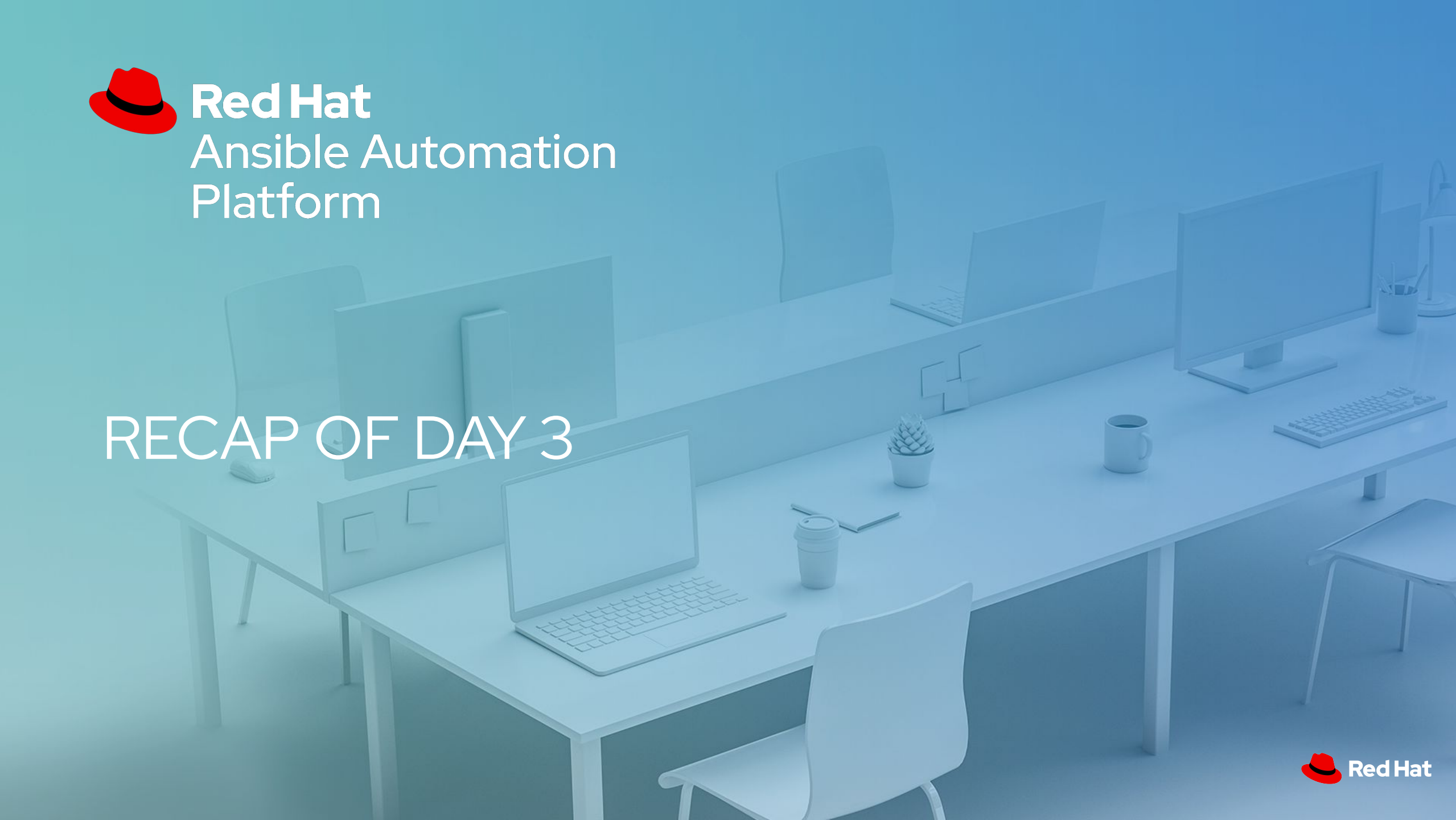
Day 4: Sharing automation

Topics Covered:

- Roles
- Galaxy
- Exercise 6

Red Hat
Ansible Automation
Platform

Red Hat



Red Hat
Ansible Automation
Platform

RECAP OF DAY 3

Red Hat

Roles

Roles are a packages of closely related Ansible content that can be shared more easily than plays alone.

- Improves readability and maintainability of complex plays
- Eases sharing, reuse and standardization of automation processes
- Enables Ansible content to exist independently of playbooks, projects -- even organizations
- Provides functional conveniences such as file path resolution and default values

Roles

Project with Embedded Roles Example

```
site.yml
roles/
   common/
     files/
     templates/
     tasks/
     handlers/
     vars/
     defaults/
     meta/
```

```
   iis/
     files/
     templates/
     tasks/
     handlers/
     vars/
     defaults/
     meta/
```

Roles

Project with Embedded Roles Example

```
# site.yml
---
- name: Execute common and iis role
  hosts: web
  roles:
    - common
    - iis
```

Roles

http://galaxy.ansible.com

Ansible Galaxy is a hub for finding, reusing and sharing Ansible content.

Jump-start your automation project with content contributed and reviewed by the Ansible community.

Exercise 6

- A Playbook Using Roles

Red Hat
Ansible Automation
Platform

Red Hat

Day 5: Patching Windows

- Recap of the workshop
- Patching Windows (Exercise 7)
- Survey
- Next Steps:
  - Learning Subscription
  - Trial deployment

Red Hat
Ansible Automation Platform

Next Steps

GET STARTED

ansible.com/get-started

ansible.com/tower-trial

JOIN THE COMMUNITY

ansible.com/community

WORKSHOPS & TRAINING

ansible.com/workshops

Red Hat Training

SHARE YOUR STORY

Follow us @Ansible

Friend us on Facebook

Red Hat

Thank you


linkedin.com/company/red-hat


youtube.com/AnsibleAutomation


facebook.com/ansibleautomation


twitter.com/ansible


github.com/ansible

Red Hat