



**RED HAT**<sup>®</sup>  
**ANSIBLE**<sup>®</sup>  
Tower

# AUTOMATION ACROSS THE ENTERPRISE

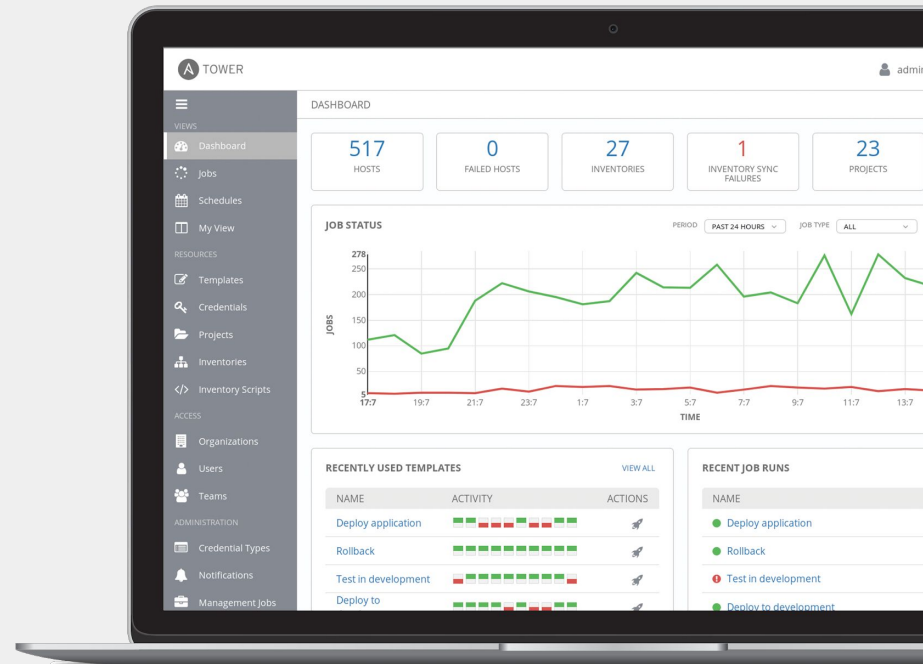
# WHAT WILL YOU LEARN?

- What is Ansible Tower
- How Ansible Tower Works
- Installing Ansible Tower
- Key Features

# 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





## RBAC

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

## PUSH BUTTON

An intuitive user interface experience makes it easy for novice users to execute playbooks you allow them access to.

## RESTful API

With an API first mentality every feature and function of Tower can be API driven. Allow seamless integration with other tools like ServiceNow and Infoblox.

## WORKFLOWS

Ansible Tower's multi-playbook workflows chain any number of playbooks, regardless of whether they use different inventories, run as different users, run at once or utilize different credentials.

## ENTERPRISE INTEGRATIONS

Integrate with enterprise authentication like TACACS+, RADIUS, Azure AD. Setup token authentication with OAuth 2. Setup notifications with PagerDuty, Slack and Twilio.

## CENTRALIZED LOGGING

All automation activity is securely logged. Who ran it, how they customized it, what it did, where it happened - all securely stored and viewable later, or exported through Ansible Tower's API.



ADMINS



USERS



ANSIBLE PLAYBOOKS



ANSIBLE CLI & CI SYSTEMS

<b>ANSIBLE TOWER</b>	ROLE-BASED ACCESS CONTROL	KNOWLEDGE & VISIBILITY	SCHEDULED & CENTRALIZED JOBS
	SIMPLE USER INTERFACE		TOWER API

<b>ANSIBLE ENGINE</b>	OPEN SOURCE MODULE LIBRARY	
	PLUGINS	PYTHON CODEBASE

TRANSPORT  
SSH, WINRM, ETC.

<b>AUTOMATE YOUR ENTERPRISE</b>	<b>INFRASTRUCTURE</b> LINUX, WINDOWS, UNIX ...	<b>NETWORKS</b> ARISTA, CISCO, JUNIPER ...	<b>CONTAINERS</b> DOCKER, LXC ...	<b>CLOUD</b> AWS, GOOGLE CLOUD, AZURE ...	<b>SERVICES</b> DATABASES, LOGGING, SOURCE CONTROL MANAGEMENT...
---------------------------------	---	---	--------------------------------------	--	---

USE CASES



PROVISIONING



CONFIGURATION MANAGEMENT



APP DEPLOYMENT



CONTINUOUS DELIVERY



SECURITY & COMPLIANCE



ORCHESTRATION

# INSTALLING ANSIBLE TOWER

```
# the most common and preferred way of
# installation for Red Hat Enterprise Linux
$ wget https://bit.ly/ansibletower

# bundled installer can be downloaded for
# Red Hat Enterprise Linux
$ wget https://bit.ly/ansibletowerbundle

# looking for a specific version? navigate to
# http://releases.ansible.com/ansible-tower
# to see all the versions available for download
```

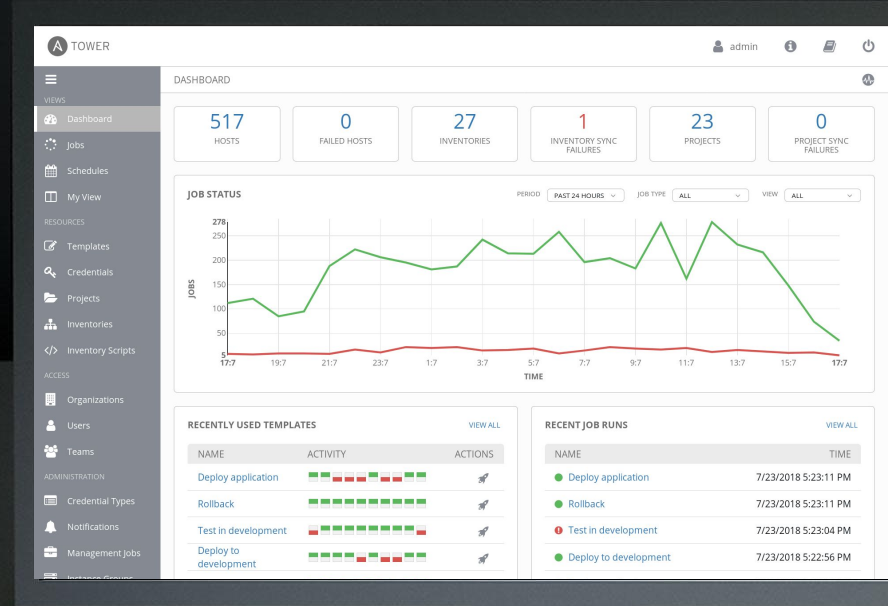
# SERVER REQUIREMENTS

- Red Hat Enterprise Linux (RHEL) 7 (and select derivatives), Ubuntu 14.04 64-bit, and Ubuntu 16.04 LTS 64-bit support required (kernel and runtime).
- A currently supported version of Mozilla Firefox or Google Chrome.
- 2 GB RAM minimum (4+ GB RAM highly recommended)
- 20 GB of dedicated hard disk space



RED HAT®  
ANSIBLE®  
Tower

# FEATURE OVERVIEW: TOWER CONCEPTS





# USER MANAGEMENT

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

# 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.

# 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

# PROJECTS

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

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

# JOB TEMPLATES

A job template is a definition and set of parameters for running an Ansible Playbook.

Job templates are useful to **execute** the same job many times and encourage the **reuse** of Ansible Playbook content and collaboration between teams.

# JOBS

A job is an instance of Ansible Tower launching an Ansible Playbook against an inventory of hosts.

- Job results can be easily viewed
- View the standard out for a more in-depth look

# ROLE BASED ACCESS CONTROL (RBAC)

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

# ROLE BASED ACCESS CONTROL (RBAC)

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



# DYNAMIC INVENTORY

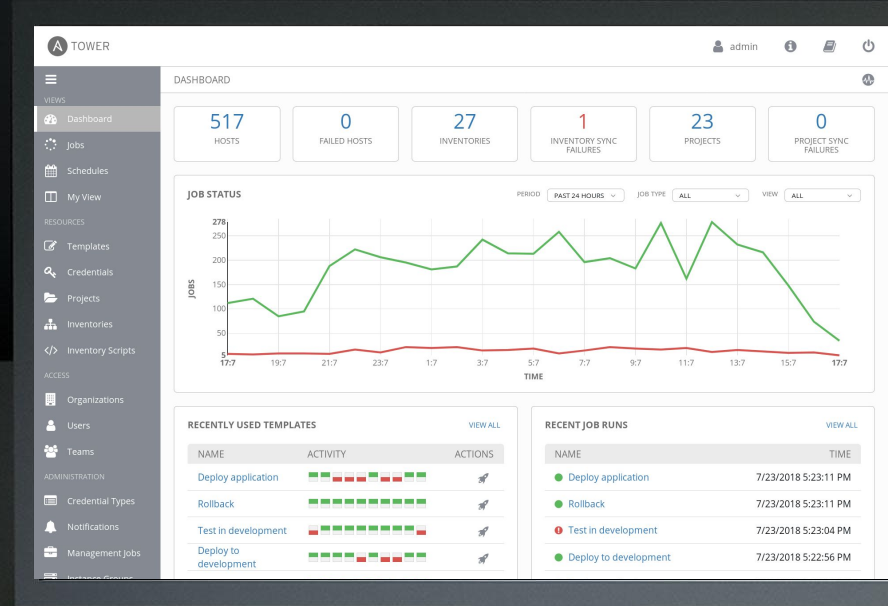
Dynamic inventory is a script that queries a service, like a cloud provider API or a management application. This data is formatted in an Ansible-specific JSON data structure and is used in lieu of static inventory files.

- Groups are generated based on host metadata
- Single source of truth saves time, avoids duplication and reduces human error
- Dynamic and static inventory sources can be used together

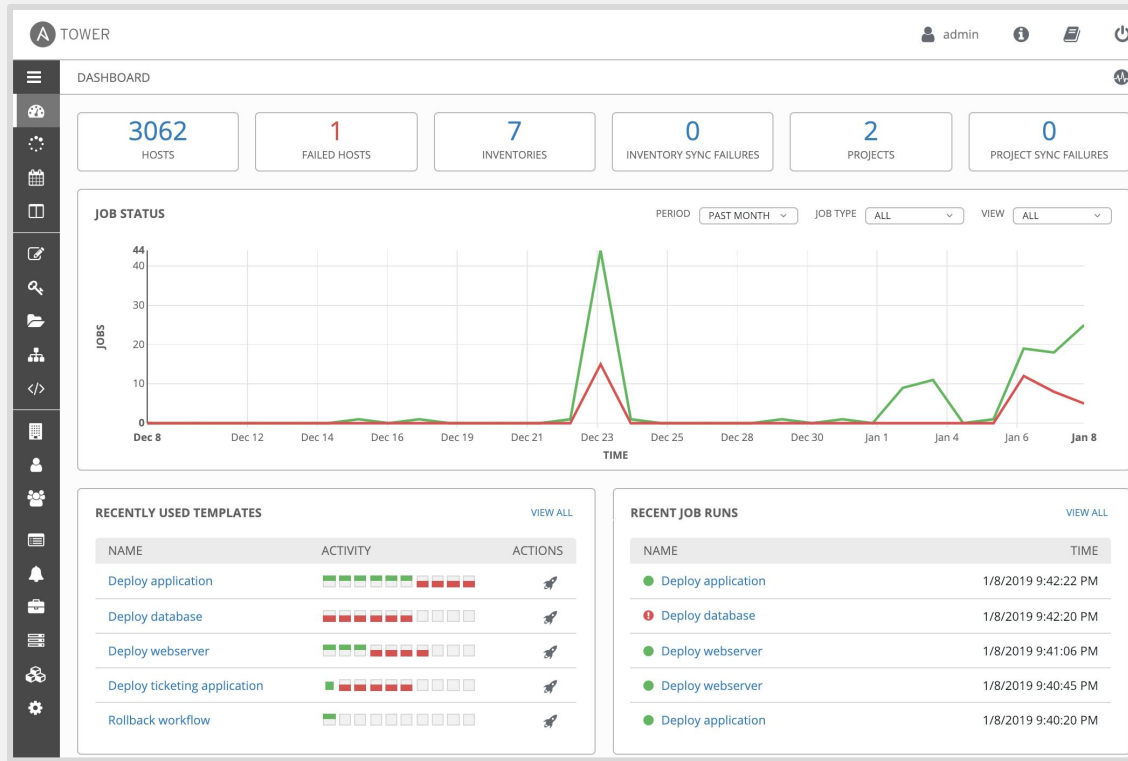


RED HAT®  
ANSIBLE®  
Tower

# FEATURE OVERVIEW: CONTROL



# ANSIBLE TOWER FEATURES: YOUR ANSIBLE DASHBOARD



# ANSIBLE TOWER FEATURES: **JOB STATUS UPDATE**

The screenshot displays the Ansible Tower web interface. The top navigation bar shows the 'TOWER' logo, the user 'admin', and various utility icons. The breadcrumb trail indicates the current location: 'JOBS / 184 - BACKUP NETWORK CONFIG'.

**Job Details Panel:**

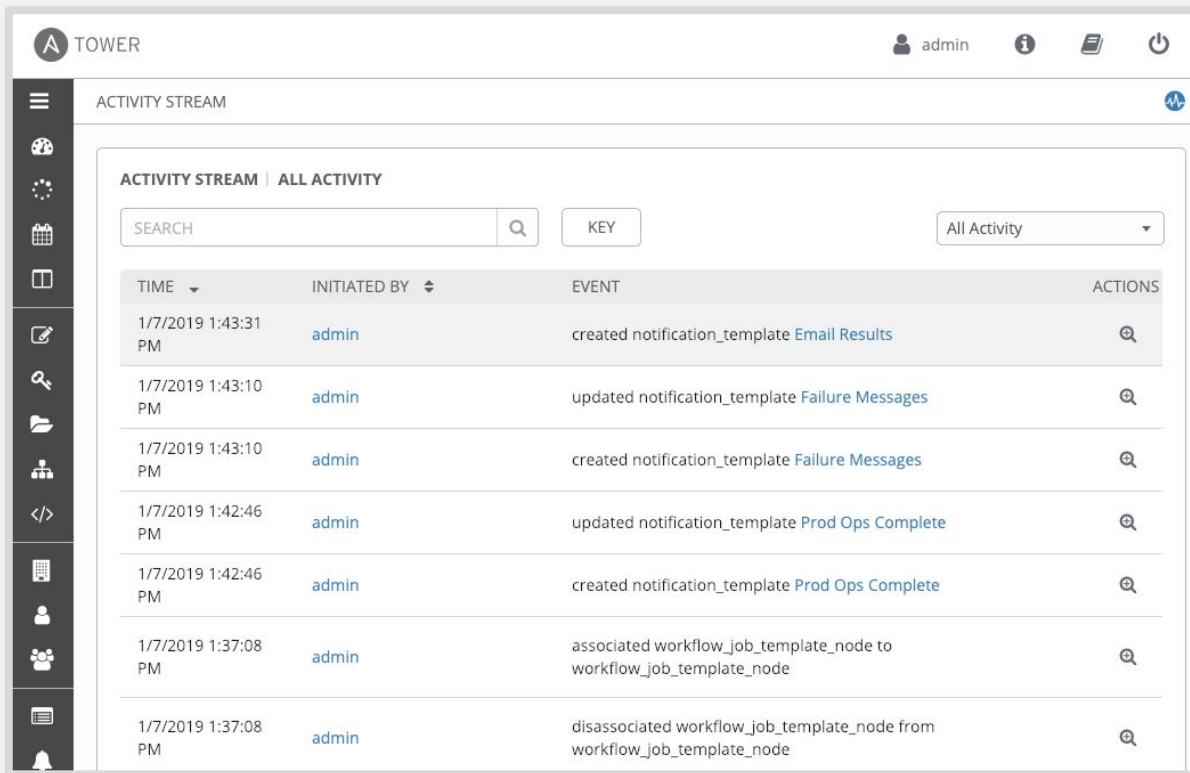
- STATUS:** Successful
- STARTED:** 1/8/2019 9:50:26 PM
- FINISHED:** 1/8/2019 9:51:02 PM
- JOB TEMPLATE:** BACKUP NETWORK CONFIG
- JOB TYPE:** Run
- LAUNCHED BY:** admin
- INVENTORY:** Workshop Inventory
- PROJECT:** Workshop Project
- REVISION:** 23a23b8
- PLAYBOOK:** network\_backup.yml
- CREDENTIAL:** Workshop Credential
- INSTANCE GROUP:** tower
- EXTRA VARIABLES:** 1 --- (with YAML and JSON tabs)

**Execution Log Panel:**

The log is titled 'BACKUP NETWORK CONFIG' and includes a progress bar showing 2 plays, 9 tasks, and 1 host. A search bar is available above the log content. The log entries are as follows:

```
21 ok: [rtr3 -> 35.183.122.35]
22
23 TASK [CREATE TIMESTAMP DIRECTORY ON ansible] ***** 21:50:53
*****
24 changed: [rtr3 -> 35.183.122.35]
25
26 TASK [TRANSFER FILE FROM THIS ANSIBLE HOST TO ansible] ***** 21:50:54
*****
27 skipping: [ansible]
28 changed: [rtr2 -> 35.183.122.35]
29 changed: [rtr4 -> 35.183.122.35]
30 changed: [rtr3 -> 35.183.122.35]
31 changed: [rtr1 -> 35.183.122.35]
32
33 PLAY [BACKUP ROUTER CONFIGURATIONS] ***** 21:50:56
*****
34
35 TASK [FIND BACKUPS] ***** 21:50:56
*****
36 ok: [localhost -> 35.183.122.35]
37
38 TASK [CREATE RESTORE JOB TEMPLATE] ***** 21:50:56
*****
39 changed: [localhost]
40
```

# ANSIBLE TOWER FEATURES: **ACTIVITY STREAM**



The screenshot displays the Ansible Tower web interface. At the top, the 'TOWER' logo is on the left, and the user 'admin' is logged in on the right. A sidebar on the left contains navigation icons. The main content area is titled 'ACTIVITY STREAM' and shows a list of events under the 'ALL ACTIVITY' tab. The list includes search and filter controls and a table of activity entries.

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

# ANSIBLE TOWER FEATURES: **MANAGE AND TRACK YOUR INVENTORY**

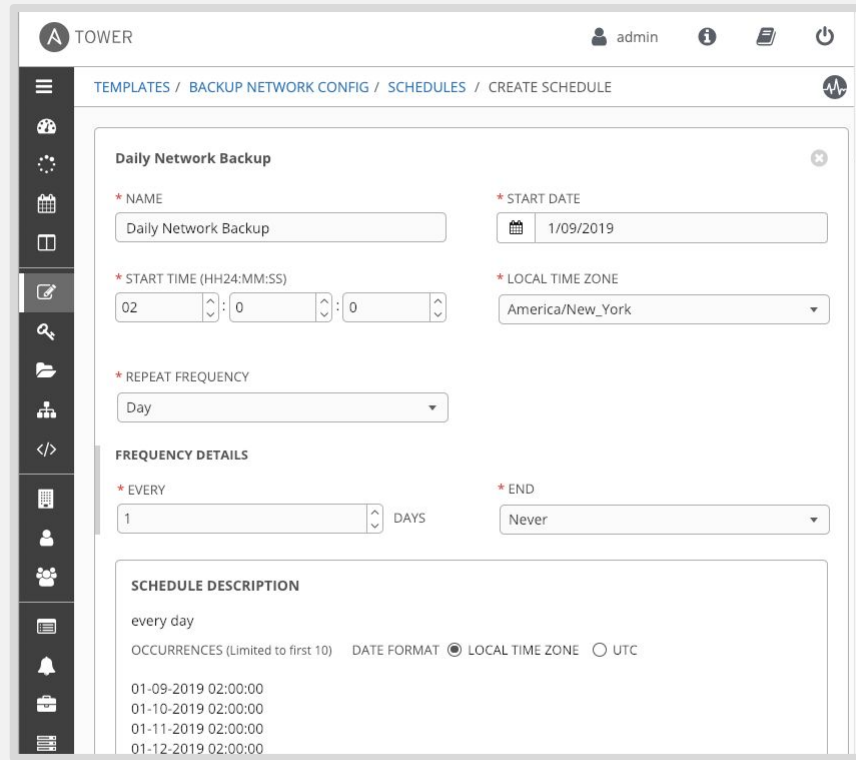
The screenshot displays the Ansible Tower web interface for configuring a source. The breadcrumb navigation shows: INVENTORIES / Durham / SOURCES / Cloud dev servers. The page title is 'Cloud dev servers'. There are three tabs: DETAILS (selected), NOTIFICATIONS, and SCHEDULES. The main configuration area includes:

- \* NAME:** Cloud dev servers
- DESCRIPTION:** sync to AWS development us-ea
- \* SOURCE:** Amazon EC2
- SOURCE DETAILS:**
  - CREDENTIAL:** AWS dev keys
  - REGIONS:** US East (Ohio)
  - INSTANCE FILTERS:** tag:Name=\*development\*
  - ONLY GROUP BY:** (empty)
  - VERBOSITY:** 1 (INFO)
  - UPDATE OPTIONS:**
    - Overwrite
    - Overwrite Variables
    - Update on Launch
- SOURCE VARIABLES:** (YAML/JSON tabs) with a text area containing:

```
1 ---
```

At the bottom right, there are 'CANCEL' and 'SAVE' buttons.

# ANSIBLE TOWER FEATURES: **SCHEDULE JOBS**



The screenshot displays the 'CREATE SCHEDULE' page in the Ansible Tower web interface. The page title is 'TEMPLATES / BACKUP NETWORK CONFIG / SCHEDULES / CREATE SCHEDULE'. The user is logged in as 'admin'. The form is for a schedule named 'Daily Network Backup'.

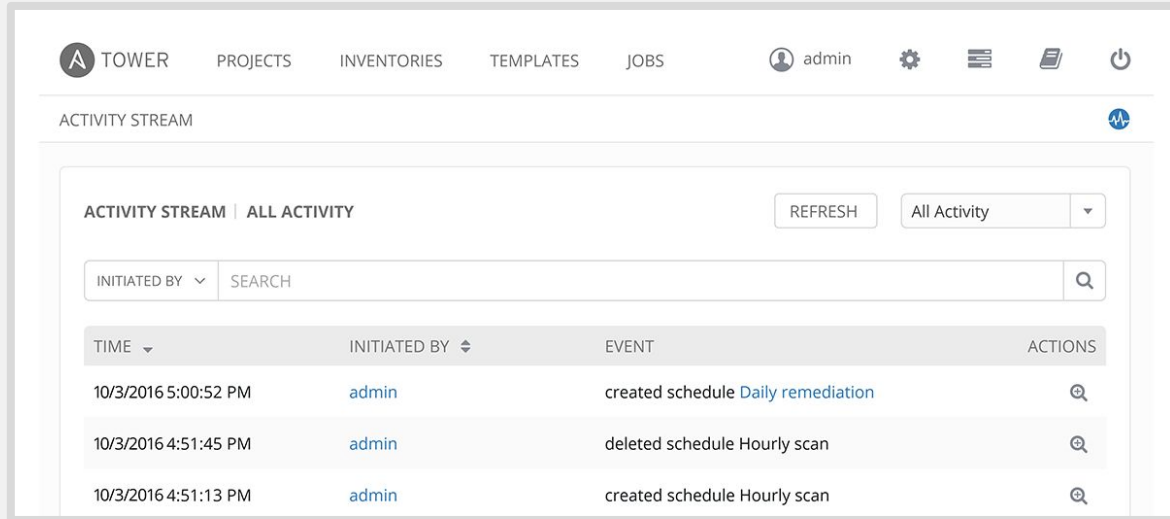
**Form Fields:**

- \* NAME:** Daily Network Backup
- \* START DATE:** 1/09/2019
- \* START TIME (HH24:MM:SS):** 02:00:00
- \* LOCAL TIME ZONE:** America/New\_York
- \* REPEAT FREQUENCY:** Day
- FREQUENCY DETAILS:**
  - \* EVERY:** 1 DAYS
  - \* END:** Never
- SCHEDULE DESCRIPTION:** every day

**OCCURRENCES (Limited to first 10):** DATE FORMAT  LOCAL TIME ZONE  UTC

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

# ANSIBLE TOWER FEATURES: **EXTERNAL LOGGING**

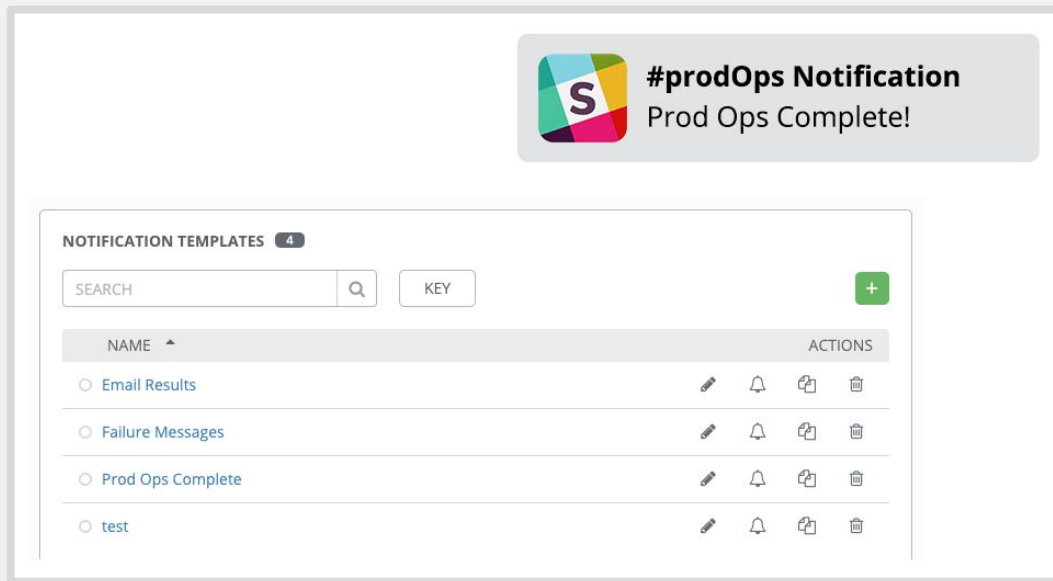


The screenshot displays the Ansible Tower web interface. At the top, there is a navigation bar with the 'TOWER' logo and menu items for 'PROJECTS', 'INVENTORIES', 'TEMPLATES', and 'JOBS'. The user 'admin' is logged in, and there are icons for settings, a list, a document, and a power button. Below the navigation bar is the 'ACTIVITY STREAM' section, which includes a 'REFRESH' button and a dropdown menu set to 'All Activity'. A search bar is present with a search icon. The main content is a table with the following data:

TIME	INITIATED BY	EVENT	ACTIONS
10/3/2016 5:00:52 PM	admin	created schedule <a href="#">Daily remediation</a>	
10/3/2016 4:51:45 PM	admin	deleted schedule Hourly scan	
10/3/2016 4:51:13 PM	admin	created schedule Hourly scan	



# ANSIBLE TOWER FEATURES: **INTEGRATED NOTIFICATIONS**



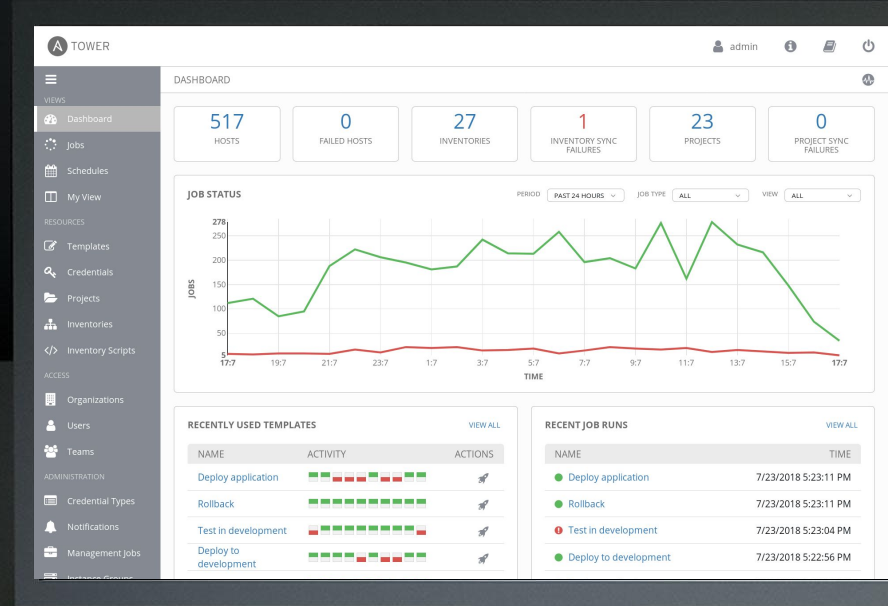
The screenshot displays the Ansible Tower notification interface. At the top right, a notification card features a colorful 'S' icon, the text "#prodOps Notification", and "Prod Ops Complete!". Below this is a section titled "NOTIFICATION TEMPLATES" with a count of 4. It includes a search bar, a "KEY" button, and a green "+" button. A table lists the templates with columns for "NAME" and "ACTIONS".

NAME ^	ACTIONS
<input type="radio"/> Email Results	
<input type="radio"/> Failure Messages	
<input type="radio"/> Prod Ops Complete	
<input type="radio"/> test	



RED HAT®  
ANSIBLE®  
Tower

# FEATURE OVERVIEW: DELEGATION



# ANSIBLE TOWER FEATURES: **ROLE BASED ACCESS CONTROL**

## USERS

The screenshot shows the 'USERS' management page in Ansible Tower. At the top, the user 'admin' is logged in. The page title is 'USERS' with a sub-count of '10'. There is a search bar with a 'KEY' input field and a green '+' button. Below the search bar is a table with columns for 'USERNAME', 'FIRST NAME', 'LAST NAME', and 'ACTIONS'. The table lists ten users: admin, amadrid, awiggin, ccarby, dmeekeer, fmolo, hgraff, mrackham, ndelphiki, and parkanian. Each user row has edit and delete icons. A sidebar on the left contains navigation icons for various system functions. At the bottom right, it says 'ITEMS 1 - 10'.

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

## TEAMS

The screenshot shows the 'TEAMS' management page in Ansible Tower. At the top, the user 'admin' is logged in. The page title is 'TEAMS' with a sub-count of '5'. There is a search bar with a 'KEY' input field and a green '+' button. Below the search bar is a table with columns for 'NAME' and 'ACTIONS'. The table lists five teams: Cloud Automation Team, Development Engineering, Network Administrative Team, Network Operations Team, and Site Reliability Engineering. Each team row has edit and delete icons. A sidebar on the left contains navigation icons for various system functions. At the bottom right, it says 'ITEMS 1 - 5'.

NAME	ACTIONS
Cloud Automation Team	[edit] [delete]
Development Engineering	[edit] [delete]
Network Administrative Team	[edit] [delete]
Network Operations Team	[edit] [delete]
Site Reliability Engineering	[edit] [delete]

# ANSIBLE TOWER FEATURES: **ROLE BASED ACCESS CONTROL**

The screenshot displays the Ansible Tower web interface. At the top, the breadcrumb navigation shows 'TEMPLATES / BACKUP NETWORK CONFIG / PERMISSIONS'. The main content area is titled 'BACKUP NETWORK CONFIG' and includes tabs for 'DETAILS', 'PERMISSIONS' (which is active), 'NOTIFICATIONS', 'COMPLETED JOBS', and 'SCHEDULES'. Below the tabs is a search bar with a 'KEY' button and a green '+' icon. The main table lists users and their assigned roles:

USER	ROLE	TEAM ROLES
admin	<span>ADMIN</span> <span>SYSTEM ADMINISTRATOR</span>	
amadrid	<span>SYSTEM AUDITOR</span>	
awiggin	<span>SYSTEM ADMINISTRATOR</span>	
dmeeker		<span>EXECUTE</span>

At the bottom of the page, there is a 'TEMPLATES' section with a count of 23. It includes a search bar and a list of templates. The first template shown is 'BACKUP NETWORK CONFIG' (Job Template), which has a progress indicator (green and red bars) and the following details:

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

# ANSIBLE TOWER FEATURES: SELF-SERVICE I.T.

**LAUNCH JOB | DEPLOY SOFTWARE** ✕

\* ENTER NUMBER OF SERVICE INSTANCES.

\* PLEASE SELECT THE SERVICE OWNER.

 ▼

\* ENTER PASSWORD FOR DEPLOYED CERTIFICATE.

Cloud staging servers      Staging ssh key

# ANSIBLE TOWER FEATURES: **REMOTE COMMAND EXECUTION**

**TOWER** admin

INVENTORIES / Durham / RUN COMMAND

### EXECUTE COMMAND

\* **MODULE** ?

**ARGUMENTS** ?

**LIMIT** ?

\* **MACHINE CREDENTIAL** ?

\* **VERBOSITY** ?

**FORKS** ?

**SHOW CHANGES** ?

**ENABLE PRIVILEGE ESCALATION** ?

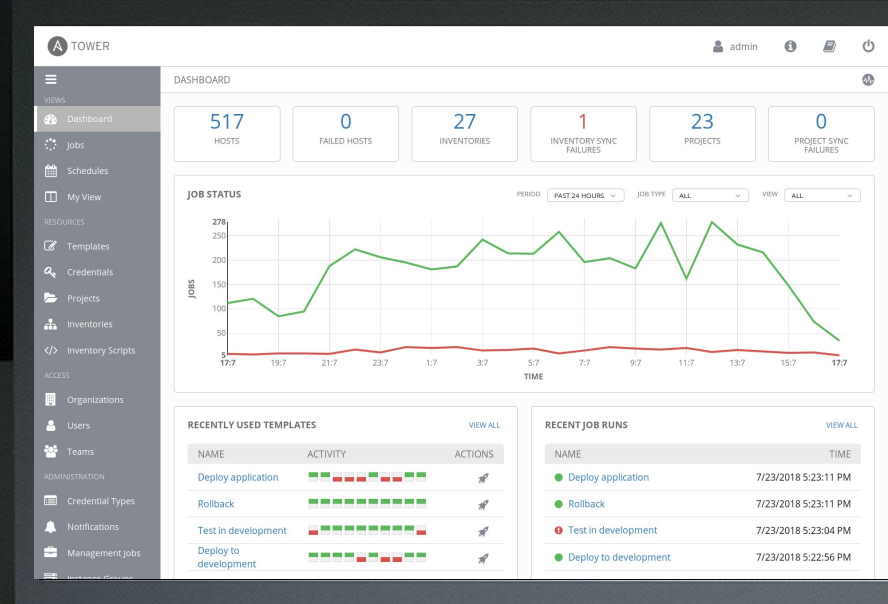
**EXTRA VARIABLES** ?

```
1 ---
```



**RED HAT**<sup>®</sup>  
**ANSIBLE**<sup>®</sup>  
Tower

## FEATURE OVERVIEW: SCALE



# ANSIBLE TOWER FEATURES: CREATE AUTOMATION WORKFLOWS

The screenshot displays the Ansible Tower web interface. At the top, the 'TOWER' logo is on the left, and the user 'admin' is logged in on the right. The main navigation bar shows 'JOBS / 137 - Deploy ticketing application'. The left sidebar contains various icons for navigation.

The central area is divided into two panels. The left panel, titled 'DETAILS', shows the job's status as 'Running'. It lists the following information:

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

Below this, there is a section for 'EXTRA VARIABLES' with tabs for 'YAML' and 'JSON', and an 'EXPAND' button. The content area shows a single variable: '1 ---'.

The right panel, titled 'Deploy ticketing application', shows the job's progress. It indicates 'TOTAL NODES 7' and 'ELAPSED 00:00:38'. Below this is a workflow diagram illustrating the sequence of tasks:

```
graph LR; Start(( )) --> DeployApp[Deploy application]; Start --> DeployWebserver[Deploy webserver]; Start --> DeployDatabase[Deploy database]; DeployApp --> RunTests[Run tests]; DeployWebserver --> RunTests; DeployDatabase --> RunTests; RunTests --> UpdateCMDB[Update CMDB]; RunTests --> UpdateLoadBalancer[Update loadbalancer]; RunTests --> Rollback[Rollback environment];
```



# ANSIBLE TOWER FEATURES: **SCALE OUT CLUSTERING**

INSTANCE GROUPS

INSTANCE GROUPS 4

SEARCH

<b>dev</b>	INSTANCES <input type="button" value="3"/>	RUNNING JOBS <input type="button" value="9"/>	TOTAL JOBS <input type="button" value="89"/>	USED CAPACITY <div style="display: inline-block; width: 61.8%; height: 10px; background: linear-gradient(to right, #007bff, #d9d9d9); border: 1px solid #d9d9d9;"></div> 61.8%	<input type="button" value="🗑"/>
<b>prod</b>	INSTANCES <input type="button" value="4"/>	RUNNING JOBS <input type="button" value="6"/>	TOTAL JOBS <input type="button" value="26"/>	USED CAPACITY <div style="display: inline-block; width: 27.3%; height: 10px; background: linear-gradient(to right, #007bff, #d9d9d9); border: 1px solid #d9d9d9;"></div> 27.3%	<input type="button" value="🗑"/>
<b>test</b>	INSTANCES <input type="button" value="3"/>	RUNNING JOBS <input type="button" value="6"/>	TOTAL JOBS <input type="button" value="44"/>	USED CAPACITY <div style="display: inline-block; width: 55.8%; height: 10px; background: linear-gradient(to right, #007bff, #d9d9d9); border: 1px solid #d9d9d9;"></div> 55.8%	<input type="button" value="🗑"/>
<b>tower</b>	INSTANCES <input type="button" value="8"/>	RUNNING JOBS <input type="button" value="0"/>	TOTAL JOBS <input type="button" value="33"/>	USED CAPACITY <div style="display: inline-block; width: 43.6%; height: 10px; background: linear-gradient(to right, #007bff, #d9d9d9); border: 1px solid #d9d9d9;"></div> 43.6%	

ITEMS 1 - 4

# NEXT STEPS

## GET STARTED

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

[ansible.com/tower-trial](https://ansible.com/tower-trial)

---

## WORKSHOPS & TRAINING

[ansible.com/workshops](https://ansible.com/workshops)

[Red Hat Training](#)

## JOIN THE COMMUNITY

[ansible.com/community](https://ansible.com/community)

---

## SHARE YOUR STORY

[Follow us @Ansible](#)

[Friend us on Facebook](#)