
Ansible Tower Upgrade and Migration

Release Ansible Tower 3.1.3

Red Hat, Inc.

Jul 12, 2017

CONTENTS

1	Release Notes for Ansible Tower Version 3.1.3	2
1.1	Ansible Tower Version 3.1.3	2
1.2	Ansible Tower Version 3.1.2	2
1.3	Ansible Tower Version 3.1.1	4
1.4	Ansible Tower Version 3.1.0	4
2	Upgrading Ansible Tower	6
2.1	Upgrade Planning	6
2.2	Obtaining Ansible Tower	6
2.3	Setting up the Inventory File	7
2.4	The Setup Playbook	10
3	System Tracking Migration	11
4	Role-Based Access Controls	12
4.1	Enhanced and Simplified RBAC System	12
4.2	Specific Changes to Note	12
5	Job Template Changes	13
5.1	Prompt on Launch	13
5.2	Permissions/RBAC Notes	15
5.3	Surveys	15
6	Job Output View Changes	17
6.1	Details	17
6.2	Standard Out Pane	19
7	Using virtualenv with Ansible Tower	22
7.1	Modifying the virtualenv	22
8	Index	23
9	Copyright © 2017 Red Hat, Inc.	24
	Index	25

Thank you for your interest in Ansible Tower by Red Hat. Ansible Tower is a commercial offering that helps teams manage complex multi-tier deployments by adding control, knowledge, and delegation to Ansible-powered environments.

Note: You must upgrade your Ansible Tower 2.4.4 (or later) system to Ansible Tower 3.0 before you can upgrade to Ansible Tower 3.1.0.

The *Ansible Tower Upgrade and Migration Guide* discusses how to upgrade your Ansible Tower 2.4.4 (or later) system to the 3.0 version. The Ansible Tower 3.0 release introduced many updates and changes to Tower, including changes to the way upgrades run and a completely rewritten RBAC system. This guide covers these and other related changes that you should keep in mind as you plan to migrate your data and prepare for this upgrade.

Ansible Tower Version 3.1.3; May 01, 2017; <https://access.redhat.com/>

RELEASE NOTES FOR ANSIBLE TOWER VERSION 3.1.3

1.1 Ansible Tower Version 3.1.3

- Added Spanish translations for Ansible Tower
- Fixed a problem where survey password defaults of a certain length could prevent a job from launching
- Fixed truncation of long job run results causing an excessive number of blank lines
- Fixed a problem where running two different jobs in parallel from the same project could cause an error
- Fixed a problem where Splunk HTTP event collectors did not send job_event data
- Fixed a problem where email notifications could send survey password values in plain text in extra_vars
- Fixed handling of job output from certain modules
- Fixed a problem where setting “Required” for a Multiple Choice (Single Select) survey question could prevent changing the default
- Fixed a problem where provisioning callbacks configured to prompt for extra variables could return an error
- Fixed a problem where modifying team permissions when using a basic license would return an error
- Fixed a problem in LDAP configuration where adding values to the database caused users to lose permanent access to Tower
- Fixed an issue where a database restore could fail
- Fixed an issue where setup could fail if there was no firewall installed
- Fixed an issue where Insights projects could cause project syncs to fail
- Fixed an issue applying Tower configuration across cluster nodes
- Added the ability to disable SSL certificate verification for hosted Splunk logging
- Improved Ansible Tower to allow for passing extra_vars on ad_hoc commands
- Updated Ansible Tower so that it can process facts output from Ansible 2.3

1.2 Ansible Tower Version 3.1.2

- Added subpackaging for sever, UI, and setup packages
- Added support for Red Hat Insights project type
- Added support for explicitly specifying the host descriptor used for RabbitMQ config via *rabbitmq_host*

- Adjusted search on the Job Details screen to match the behavior across Tower
- Adjusted Tower logging to log asynchronously
- Fixed various and minor UI bugs
- Fixed a callback bug which was causing a *task_args* leak between job events
- Fixed an issue where jobs were not able to be sorted by descending ID
- Fixed an issue where, when working with Splunk, the log aggregator type shows as Logstash instead of Splunk
- Fixed an issue where, when a user has two groups in an inventory (one using a VMware script and one using a custom script), clicking *sync* on the custom script group caused the sync icon to link to the wrong inventory sync
- Fixed a problem where users were not able to put multi-line text in a Text Area-type field in a survey
- Fixed a problem where users who had admin access on Workflows, but were not Org level admins, could not add or remove job templates from Workflows
- Fixed a problem with job templates that include a multiple choice survey response, where, even when multiple selections are required, the job template ran with an empty array
- Fixed a problem where surveys were passing a variable as empty instead of null when they included text or a text area field that had a minimum length >0 and was not filled in
- Fixed a problem where Tower jobs hang and do not run when the Splunk server is unresponsive or unavailable
- Fixed a problem where users with admin level permissions on projects could not modify project details
- Fixed a problem in multiple choice survey inputs where, when selecting a string that had similar characters or words at the beginning or end of the string, a similar but smaller version of that string was rendered as the user's selection (even though the correct value was still passed to extra-vars on launch)
- Fixed an issue around Git project updates failing when the username was specified
- Fixed a problem where job templates from mercurial project updates failed to run
- Fixed a problem with provisioning callbacks where they failed with '400' responses when extra_vars were passed to the API through curl in the callback
- Fixed a problem where running the installer again anytime after successfully creating the rabbitmq user caused the installation program to fail
- Fixed an issue where Windows package scan jobs fail when targeting a Windows 2012R2 host
- Fixed an issue where users with admin access to Workflow Templates could not modify the workflow
- Fixed an issue where a warning was incorrectly displayed for the output of a canceled job
- Fixed an issue where Mercurial project revisions were not read correctly for Projects
- Fixed an issue where Tower upgrades would fail when applying *rabbitmq_user* in a cluster
- Fixed an issue where certain characters in a Project SCM URL would cause updates to fail
- Improved custom inventory scripts support by ensuring that newlines added to the script are not trimmed
- Relaxed the SELinux policy dependency to allow Tower to be installed on older Enterprise Linux 7 releases
- Updated Ansible Tower so that the host config key is marked as required when provisioning callbacks are selected
- Updated Ansible Tower so that PostgreSQL Server is no longer installed on Tower nodes not hosting the database
- Updated Ansible Tower so that Tower shows *extra_vars* for ad-hoc commands in the UI

1.3 Ansible Tower Version 3.1.1

- Added a preflight check for password and pre-3.1.0 active/passive (HA) inventory setups prior to installation
- Fixed a problem where, while running a clustered Tower deployment configuration, there were some instances where realtime job event data did not flow through the channel layer
- Fixed a problem with searching where an invalid search term was entered and the error dialog continued to persist
- Fixed a problem with Slack notifications where they were not emitted if only ‘Failure’ was selected
- Fixed a problem where logging out via Tower logout button caused subsequent login attempts to fail
- Fixed an issue where, when logging was enabled, a missing logging UUID setting would cause a startup error, making the system unresponsive

1.4 Ansible Tower Version 3.1.0

- Added support for configuring most aspects of Ansible Tower directly from the Tower user interface (and Tower API), rather than editing Tower configuration files
- Added support for “Scale-Out” Clusters, which replaces the HA/Redundancy method from prior Tower releases
- Added support for Workflows, a chain of job templates executed in order
- Added support for sending event and log messages to various logging services (Elastic, Splunk, Sumologic, Loggly, generic REST endpoint)
- Added support for a new Tower Search feature which supports GitHub-style “key:value” searching
- Added support for Ubuntu 16.04
- Added support for a New Project Sync Architecture, where projects are now checked out at job runtime
- Added support for setting timeouts on job runs
- Added support for internationalization and localization (French and Japanese)
- Added support for multi-playbook Workflows
- Added `/api/v1/settings` for Tower managed settings. This corresponds to the in-Tower configuration UI
- Added support for windows scan jobs
- Added support so that the SCM Revision used is now stored on Job
- Added support for API endpoints to now show `__search` filter fields for broader searching of objects
- Added support so that system jobs are now shown in `/api/v1/unified_jobs`
- Added support for the new Ansible `vmware_inventory` script
- Added support for Job stdout downloads, which may generate and cache on the fly
- Added support for `/api/v1/inventory_updates` and `/api/v1/project_updates` to view those specific job types
- Added support for `user_capabilities` API elements in various places to allow API consumers to know if their user can perform the referenced actions on the object
- Added support for `set_stats` for Workflow jobs to persist data between Workflow job runs, support added in ansible core also

- Added support for Tower callbacks so that they can now resolve `ansible_host` as well as `ansible_ssh_host`
- Added support for Tower callbacks so that they now filter out `ansible_` variables on POST
- Added support for notifications so that they are emitted on jobs marked as failed by the dead job detector
- Added eu-west-2 and ca-central-1 to the list of supported EC2 regions
- Added support for `format=ansi_download` when downloading stdout
- Deprecated support for Rackspace inventories
- Fixed an issue where manual projects could be launched/updated
- Fixed various unicode issues
- Fixed various issues dealing with self signed certificates value.
- Fixed Jobs so that they now show `$encrypted` for these variables, where they previously did not
- Improved performance for viewing job and job template lists
- Improved Tower virtualenv so that it is purged on upgrade
- Improved setup playbook so that it is more tolerant of various iptables/firewalld configurations
- Improved the optimization of PostgreSQL installation to improve overall performance
- Improved database migrations through consolidation to make upgrades/installs faster
- Improved hardening for web server configuration (SSL, HSTS)
- Removed zeromq as a communications channel between dependent services in favor of rabbitmq
- Removed `/api/v1/jobs/n/job_plays` and `/api/v1/jobs/n/job_tasks`
- Removed proot in favor of bubblewrap for process isolation
- Removed the ability to make POST requests on the `/api/v1/jobs/` endpoint
- Removed `has_schedules` from various endpoints, as it was never populated
- Removed support for Red Hat Enterprise Linux 6/CentOS 6 and Ubuntu 12.04
- Updated surveys so that a blank value for a survey question default value now passes an empty string as a value
- Updated surveys so that previously existing surveys with blank default question values now pass empty strings as an extra variable
- Updated Websockets, moving them from socket.io to django channels and are now served under port 443/80 along with the regular web service. Port 8080 is no longer needed.
- Updated Job results so that they are now driven by job events and thus provides clickable context
- Updated Tower so that it now uses the system time zone by default
- Updated Tower requirements for Ansible–Tower now requires Ansible 2.1 or later
- Updated Ansible inventory plugins to the latest versions
- Updated Web server to NGINX from Apache
- Updated survey passwords so that they are now encrypted when stored in the database
- Updated `request_tower_configuration.sh`

UPGRADING ANSIBLE TOWER

Topics:

- *Upgrade Planning*
- *Obtaining Ansible Tower*
- *Setting up the Inventory File*
- *The Setup Playbook*

Note: You must upgrade your Ansible Tower 2.4.4 (or later) system to Ansible Tower 3.0 before you can upgrade to Ansible Tower 3.1.0.

2.1 Upgrade Planning

This section covers changes that you should keep in mind as you attempt to upgrade your Ansible Tower Instance

- If you are not yet using a 2.4.x version of Ansible Tower, **do not** attempt to upgrade directly to Ansible Tower 3.0.x or 3.1.0. You must start with a system which has a version of Tower 2.4.x installed or the upgrade will fail.
- If you are not using using a 3.0.x version of Ansible Tower; you must upgrade to 3.0.x before you can upgrade to 3.1.0.
- Ansible Tower 3.0 simplified installation and removed the need to run `./configure/` as part of the initial setup.
- The file `tower_setup_conf.yml` is no longer used. Instead, you should now edit the **inventory** file in the `/ansible-tower-setup-<tower_version>/` directory.
- Earlier version of Tower used MongoDB when setting up an initial database; please note that Ansible Tower 3.0 has replaced the use of MongoDB with PostgreSQL.

2.2 Obtaining Ansible Tower

Download and then extract the Ansible Tower installation/upgrade tool: <http://releases.ansible.com/ansible-tower/setup/>


```
root@localhost:~$ tar xvzf ansible-tower-setup-latest.tar.gz
root@localhost:~$ cd ansible-tower-setup-<tower_version>
```

To install or upgrade, start by editing the inventory file in the `ansible-tower-setup-<tower_version>` directory, replacing `<tower_version>` with the version number, such as `2.4.5` or `3.0.0` directory.

2.3 Setting up the Inventory File

As you edit your inventory file, there are a few things you must keep in mind:

- The contents of the inventory file should be defined in `./inventory`, next to the `./setup.sh` installer playbook.
- For **installations and upgrades**: If you need to make use of external databases, you must ensure the database sections of your inventory file are properly setup. Edit this file and add your external database information before running the setup script.
- For **redundant installations**: If you are creating a redundant setup, you must replace `localhost` with the hostname or IP address of all instances. All nodes/instances must be able to reach any others using this hostname or address. In other words, you cannot use the `localhost ansible_connection: local` on one of the nodes *AND* all of the nodes should use the same format for the host names.

Therefore, this will *not* work:

```
[tower]
localhost ansible_connection: local
hostA
hostB.example.com
172.27.0.4
```

Instead, use these formats:

```
[tower]
hostA
hostB
hostC
```

OR

```
hostA.example.com
hostB.example.com
hostC.example.com
```

OR

```
[tower]
172.27.0.2
172.27.0.3
172.27.0.4
```

- For **installations**: When performing an installation, you must supply any necessary passwords in the inventory file.

Note: Changes made to the installation process now require that you fill out the all of the password fields in the inventory file. If you need to know where to find the values for these they should be:

```
admin_password='' ← Tower local admin password
pg_password='' ← Found in /etc/tower/conf.d/postgres.py
rabbitmq_password='' ← create a new password here (alpha-numeric with no special characters)
```

Example Inventory file

- For **provisioning new nodes**: When provisioning new nodes add the nodes to the inventory file with all current nodes, make sure all passwords are included in the inventory file.
- For **upgrades**: When upgrading, be sure to compare your inventory file to the current release version. It is recommended that you keep the passwords in here even when performing an upgrade.

Example Single Node Inventory File

```
[tower]
localhost ansible_connection=local

[database]

[all:vars]
admin_password='password'

pg_host=''
pg_port=''

pg_database='awx'
pg_username='awx'
pg_password='password'

rabbitmq_port=5672
rabbitmq_vhost=tower
rabbitmq_username=tower
rabbitmq_password='password'
rabbitmq_cookie=rabbitmqcookie

# Needs to be true for fqdns and ip addresses
rabbitmq_use_long_name=false
# Needs to remain false if you are using localhost
```

Example Multi Node Cluster Inventory File

```
[tower]
clusternode1.example.com
clusternode2.example.com
clusternode3.example.com

[database]
dbnode.example.com

[all:vars]
ansible_become=true

admin_password='password'

pg_host='dbnode.example.com'
pg_port='5432'
```

```
pg_database='tower'
pg_username='tower'
pg_password='password'

rabbitmq_port=5672
rabbitmq_vhost=tower
rabbitmq_username=tower
rabbitmq_password=tower
rabbitmq_cookie=rabbitmqcookie

# Needs to be true for fqdns and ip addresses
rabbitmq_use_long_name=true
```

Example Inventory file for an external existing database

```
[tower]
node.example.com ansible_connection=local

[database]

[all:vars]
admin_password='password'
pg_password='password'
rabbitmq_password='password'

pg_host='database.example.com'
pg_port='5432'

pg_database='awx'
pg_username='awx'
pg_password='password'
```

Example Inventory file for external database which needs installation

```
[tower]
node.example.com ansible_connection=local

[database]
database.example.com

[all:vars]
admin_password='password'
pg_password='password'
rabbitmq_password='password'

pg_host='database.example.com'
pg_port='5432'

pg_database='awx'
pg_username='awx'
pg_password='password'
```

Once any necessary changes have been made, you are ready to run `./setup.sh`.

Note: Root access to the remote machines is required. With Ansible, this can be achieved in different ways:

- `ansible_user=root ansible_ssh_password="your_password_here"` inventory host or group variables
 - `ansible_user=root ansible_ssh_private_key_file="path_to_your_keyfile.pem"` inventory host or group variables
 - `ANSIBLE_BECOME_METHOD='sudo' ANSIBLE_BECOME=True ./setup.sh`
 - `ANSIBLE_SUDO=True ./setup.sh`
-

2.4 The Setup Playbook

Note: Ansible Tower 3.0 simplifies installation and removes the need to run `./configure/` as part of the installation setup. Users of older versions should follow the instructions available in the v.2.4.5 (or earlier) releases of the Tower Documentation available at: <http://docs.ansible.com/>

The Tower setup playbook script uses the `inventory` file and is invoked as `./setup.sh` from the path where you unpacked the Tower installer tarball.

```
root@localhost:~$ ./setup.sh
```

The setup script takes the following arguments:

- `-h` – Show this help message and exit
- `-i INVENTORY_FILE` – Path to Ansible inventory file (default: `inventory`)
- `-e EXTRA_VARS` – Set additional Ansible variables as `key=value` or YAML/JSON (i.e. `-e bundle_install=false` forces an online installation)
- `-b` – Perform a database backup in lieu of installing
- `-r` – Perform a database restore in lieu of installing (a default restore path is used unless `EXTRA_VARS` are provided with a non-default path, as shown in the code example below)

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

Note: Please note that a issue was discovered in Tower 3.0.0 and 3.0.1 that prevented proper system backups and restorations.

If you need to back up or restore your Tower v3.0.0 or v3.0.1 installation, use the v3.0.2 installer to do so.

SYSTEM TRACKING MIGRATION

Once your system has been upgraded to Ansible Tower 3.x, you will find that your system tracking data has been migrated from MongoDB to PostgreSQL. If you want to delete the old data in MongoDB, you can do so manually. First, connect to your mongo database using the mongo command line client, then run the following commands:

```
$ use system_tracking
$ db.runCommand( { dropDatabase: 1 } )
```

At this point, you can also remove the MongoDB packages.

ROLE-BASED ACCESS CONTROLS

Ansible Tower 3.0 has changed significantly around the way that the Role-Based Access Control (RBAC) system works. For the latest RBAC documentation, refer to the [Role-Based Access Controls](#) section in the Tower User Guide.

4.1 Enhanced and Simplified RBAC System

Based on user feedback, Ansible Tower both expands and simplifies its role-based access control. No longer is job template visibility configured via a combination of permissions on inventory, projects, and credentials. If you want to give any user or team permissions to use a job template, just assign permissions directly on the job template. Similarly, credentials are now full objects in Tower's RBAC system, and can be assigned to multiple users and/or teams for use.

A new 'Auditor' type has been introduced in Tower as well, who can see all aspects of the systems automation, but has no permission to run or change automation, for those that need a system-level auditor. (This may also be useful for a service account that scrapes automation information from Tower's API.)

4.2 Specific Changes to Note

There are a few changes you should keep in mind as you work with the RBAC system as redesigned for Ansible Tower:

- You no longer set the "team" or "user" for a credential. Instead, you use Tower's RBAC system to grant ownership, auditor, or usage roles.
- Deletion of job run data is now restricted to system and organization administrators.
- Projects no longer have multiple organizations. You *must* provide an organization when creating a new project through the API:

```
- projects/:id/organizations --> removed
```

- New Auditor type in Tower has been added which can see all aspects of the systems automation but does not have permission to run or change things.

JOB TEMPLATE CHANGES

Job templates have been updated in Tower to allow you more flexibility when creating and working with them.

5.1 Prompt on Launch

In prior versions of Ansible Tower, you could set “Prompt on Launch” against Extra Variables that you want to potentially pass through the job template. Starting with version 3.0, Ansible Tower allows you to prompt for an inventory selection, job type, and more.

Selecting “Prompt on Launch” means that even if a value is supplied at the time of the job template creation, the user launching the job will be prompted to supply new information or confirm what was entered in the job template originally.

The following job template settings allow for prompting at the time of launch:

- Job Type (run or check type jobs only, as scan jobs cannot be changed at the time of launch)
- Inventory
- Machine credential
- Limit
- Job Tags
- Extra variables

NEW JOB TEMPLATE

DETAILS | COMPLETED JOBS | PERMISSIONS | NOTIFICATIONS

*NAME DESCRIPTION *JOB TYPE
 Prompt on launch

*INVENTORY *PROJECT *PLAYBOOK
 Prompt on launch

*MACHINE CREDENTIAL CLOUD CREDENTIAL NETWORK CREDENTIAL
 Prompt on launch

FORKS LIMIT *VERBOSITY
 Prompt on launch

JOB TAGS OPTIONS
 Enable Privilege Escalation
 Allow Provisioning Callbacks

LABELS

EXTRA VARIABLES YAML JSON

 Prompt on launch

As you work with migrating your Tower 2.4.5 job templates to 3.x, please keep in mind the following:

- All “Prompt on Launch” fields are set to *False* by default after migrating to 3.x (new job templates created also have all “Prompt on Launch” fields set to *False* by default).
 - With one exception for those upgrading from 2.4.5 to 3.x: if a credential used in Tower 2.4.5 was null, the credential will be prompted for in 3.x.
- If you have Job Templates with a null credential, in the migration from 2.4.5 to 3.x, “ask_credential_on_launch” is set to *True*.
 - Note that there was no way to set a default credential in 2.4.5. However, in 3.x, you can set a default credential and select to prompt the user at launch time to confirm the default credential or change it to something new.
- All other “ask_xx_on_launch” prompts are set to *False*.
- Starting with Tower 3.x, if “ask_variables_on_launch” is set to *False*, extra variables passed at launch time (via UI or API) that are not part of an enabled survey are ignored.
- While there are no changes to how “ask_variables_on_launch” behaves, keep in mind that these variables combine with survey answers.

5.2 Permissions/RBAC Notes

Job template visibility is no longer configured via a combination of permissions on inventory, projects, and credentials. Admins who want to give any user or team permissions to use a job template can quickly assign permissions directly on the job template. Similarly, credentials are now full objects in Tower’s RBAC system, and can be assigned to multiple users and/or teams for use.

If a job template a user has been granted execution capabilities on does not specify an inventory or credential, the user will be prompted at run-time to select among the inventory and credentials in the organization they own or have been granted usage capabilities.

Users that are job template administrators can make changes to job templates; however, to make changes to the inventory, project, playbook, or credentials used in the job template, the user must also have the “Use” role for the project, inventory, and all credentials currently being used or being set.

5.3 Surveys

In prior versions of Ansible Tower, you had to select a checkbox to “Enable Survey” on the Job Template before a button appeared allowing you to “Create Survey”.

Enabling and creating surveys is much simpler in Ansible Tower.

At the bottom of each job template is a button () which opens a new dialog where you can enter your survey questions and responses.

Use the **ON/OFF** toggle button to quickly activate or deactivate this survey prompt.

Once you have entered the question information, click **Add** to add the survey prompt.

A stylized preview of the survey is presented, along with a **New Question** button. Click this button to add additional questions.

For any question, you can click on the **Edit** button to edit the question, the **Delete** button to delete the question, and click on the Up and Down arrow buttons to rearrange the order of the questions. Click **Save** to save the survey.

NEW JOB TEMPLATE | SURVEY ON

ADD SURVEY PROMPT

* PROMPT

DESCRIPTION

* ANSWER VARIABLE NAME ⓘ

* ANSWER TYPE

Choose an answer type ▼

REQUIRED

CANCEL ADD

PREVIEW

* WHICH GROUP(S) SHOULD INCLUDE THIS USER?

Enter groups, one per line.

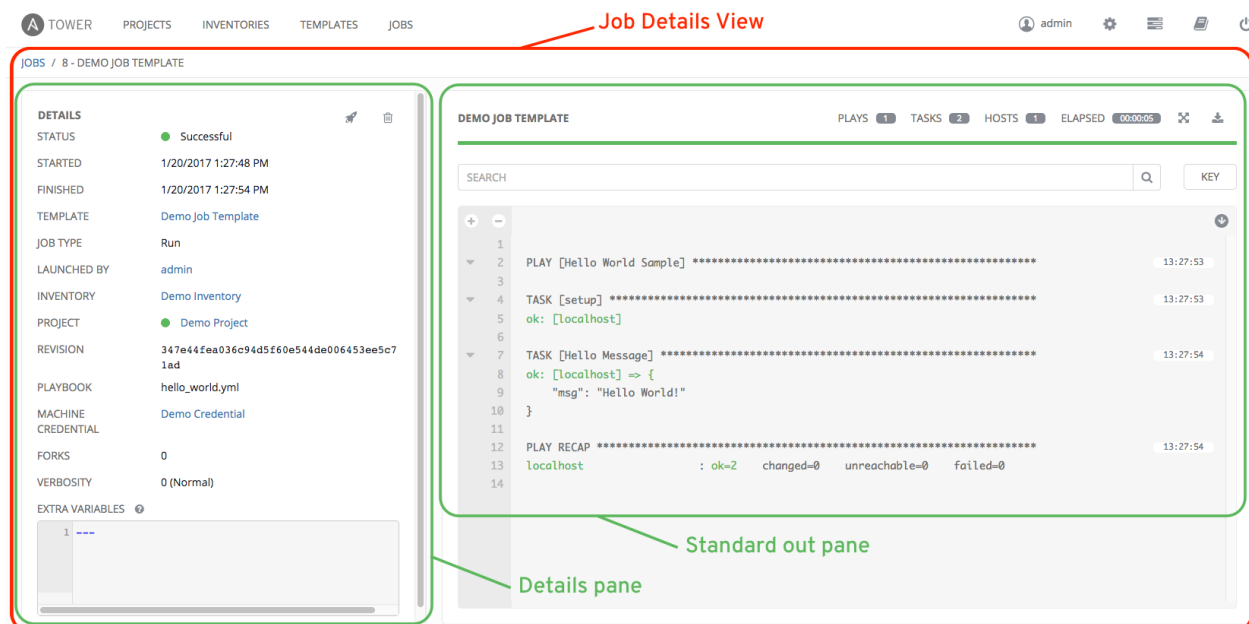
⋮ ✎ 🗑

CANCEL SAVE

JOB OUTPUT VIEW CHANGES

With the update of the overall Tower user interface, it is worth noting the changes to how job results are displayed.

Job results for inventory syncs and SCM updates only show the Results and Standard Out of the job recently Run. Job results for playbook runs consist of Results, Standard Out, Details, and the Event Summary.



Copyright © 2017 Red Hat, Inc.



For more details regarding Job Results, refer to [Jobs](#) in the *Ansible Tower User Guide*.


6.1 Details

The **Details** area shows the basic status of the job (*Running*, *Pending*, *Successful*, or *Failed*), its start and end times, which template was used, how long the job run took, who launched it, and more. The buttons in the top right of the Details view allow you to relaunch or delete the job.

By clicking on these detail entries, where appropriate, you can view the corresponding job templates, projects, and other Tower objects.



DETAILS

STATUS	● Successful
STARTED	1/20/2017 1:27:48 PM
FINISHED	1/20/2017 1:27:54 PM
TEMPLATE	Demo Job Template
JOB TYPE	Run
LAUNCHED BY	admin
INVENTORY	Demo Inventory
PROJECT	● Demo Project
REVISION	347e44fea036c94d5f60e544de006453ee5c71ad
PLAYBOOK	hello_world.yml
MACHINE CREDENTIAL	Demo Credential
FORKS	0
VERBOSITY	0 (Normal)
EXTRA VARIABLES 	

1	
---	--

6.2 Standard Out Pane

The **Standard Out** pane shows the full results of running the Ansible playbook. This shows the same information you would see if you ran it through the Ansible command line, and can be useful for debugging. You can view the event summary, host status, and the host events. The icons at the top right corner of the Standard Out pane allow you to toggle the output as a main view () or to download the output ().

6.2.1 Events Summary

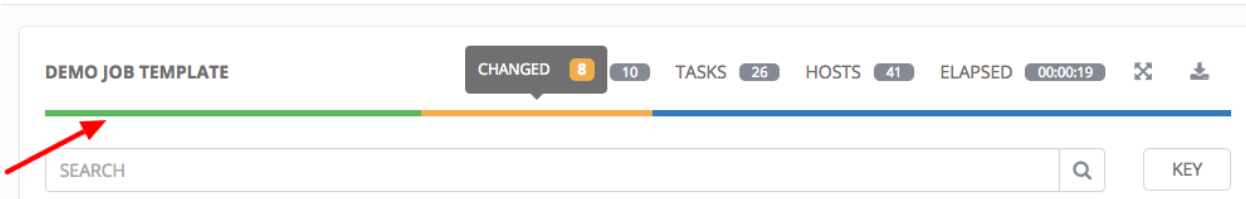
The events summary captures a tally of events that were run as part of this playbook:

- the number of plays
- the number of tasks
- the number of hosts
- the elapsed time to run the job template





6.2.2 Host Status Bar

The host status bar runs across the top of the **Standard Out** pane. Hover over a section of the host status bar and the number of hosts associated with that particular status displays.



6.2.3 Standard output view

The standard output view displays all the events that occur on a particular job. By default, all rows are expanded so that all the details are displayed. Use the collapse-all button () to switch to a view that only contains the headers for plays and tasks. Click the () button to view all lines of the standard output.

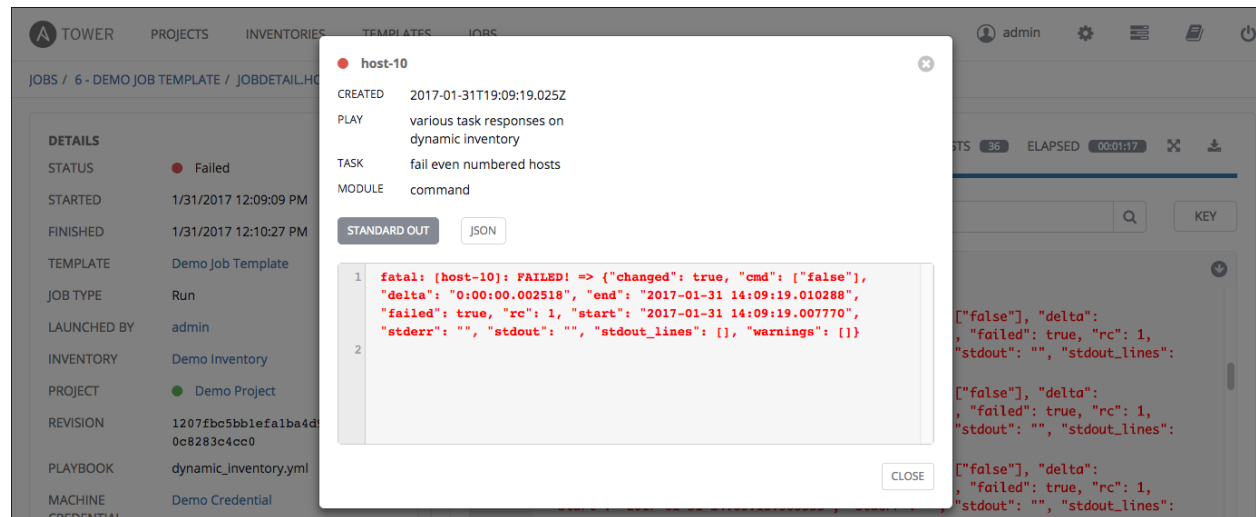
Alternatively, you can display all the details of a specific play or task by clicking on the arrow icons next to them. Click an arrow from sideways to downward to expand the the lines associated with that play or task. Click the arrow back to the sideways position to collapse and hide the lines.



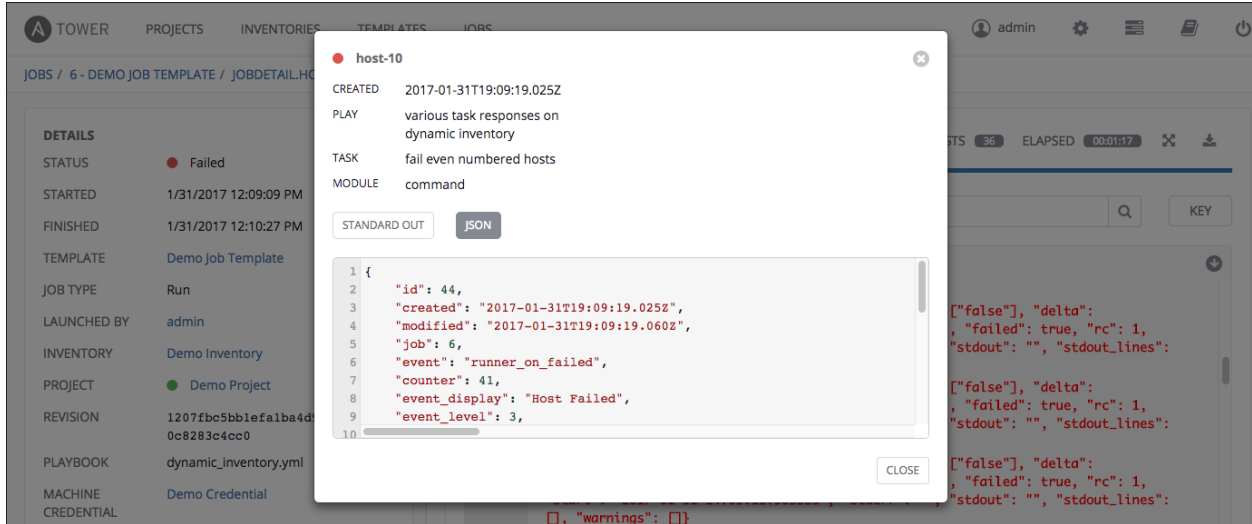
Click on a line of an event from the **Standard Out** pane and a **Host Events** dialog displays in a separate window. This window shows the host that was affected by that particular event.

6.2.4 Host Events

The **Host Events** dialog shows information about the host affected by the selected event and its associated play and task:



To view the results in JSON format, click on the **JSON** tab.



USING VIRTUALENV WITH ANSIBLE TOWER

Ansible Tower 3.0 uses *virtualenv*. Virtualenv creates isolated Python environments to avoid problems caused by conflicting dependencies and differing versions. Virtualenv works by simply creating a folder which contains all of the necessary executables and dependencies for a specific version of Python. Ansible Tower creates two virtualenvs during installation—one is used to run Tower, while the other is used to run Ansible. This allows Tower to run in a stable environment, while allowing you to add or update modules to your Ansible Python environment as necessary to run your playbooks.

Note: For more information on virtualenv, see [Virtual Environments](#)

7.1 Modifying the virtualenv

Modifying the virtualenv used by Tower is unsupported and not recommended. Instead, you can add modules to the virtualenv that Tower uses to run Ansible.

To do so, activate the Ansible virtualenv:

```
. /var/lib/awx/venv/ansible/bin/activate
```

...and then install whatever you need using `pip`:

```
pip install mypackagename
```


- genindex

COPYRIGHT © 2017 RED HAT, INC.

Ansible, Ansible Tower, Red Hat, and Red Hat Enterprise Linux are trademarks of Red Hat, Inc., registered in the United States and other countries.

If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original version.

Third Party Rights

Ubuntu and Canonical are registered trademarks of Canonical Ltd.

The CentOS Project is copyright protected. The CentOS Marks are trademarks of Red Hat, Inc. (“Red Hat”).

Microsoft, Windows, Windows Azure, and Internet Explore are trademarks of Microsoft, Inc.

VMware is a registered trademark or trademark of VMware, Inc.

Rackspace trademarks, service marks, logos and domain names are either common-law trademarks/service marks or registered trademarks/service marks of Rackspace US, Inc., or its subsidiaries, and are protected by trademark and other laws in the United States and other countries.

Amazon Web Services”, “AWS”, “Amazon EC2”, and “EC2”, are trademarks of Amazon Web Services, Inc. or its affiliates.

OpenStack™ and OpenStack logo are trademarks of OpenStack, LLC.

Chrome™ and Google Compute Engine™ service registered trademarks of Google Inc.

Safari® is a registered trademark of Apple, Inc.

Firefox® is a registered trademark of the Mozilla Foundation.

All other trademarks are the property of their respective owners.

INDEX

A

Ansible, executing in a virtual environment, 22

I

installation script

 inventory file setup, 7

 playbook setup, 10

inventory file setup, 7

J

job output, stdout, 17

jobs

 event summary, 19

 events summary, 19

 host events, 20

 host status bar, 19

 host summary, 19

 job summary, 19

M

migration, 11

migration considerations, 11

MongoDB data removal, 11

P

permissions, 12, 13

playbook setup, 10

 setup.sh, 10

PostgreSQL data migration, 11

R

RBAC, 12, 13

release notes, v3.0, 2

roles, 12

S

singleton roles, 12

system tracking data, 11

system-wide roles, 12

U

upgrade, 6

upgrade considerations, 6

V

virtual environment, 22