Testing and Debugging Ansible Automation

Debugging and Testing Playbooks

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Using Command Line Options

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Objective

Test playbooks from the ansible-playbook command line.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Syntax Check

You can check the syntax of a playbook without executing it with the following command :

```
$ ansible-playbook playbook_name.yml --syntax-check
```

If the syntax check was successful you will only see the playbook name.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Validating a Playbook

- You can use the **-C** option (or **--check**) to perform a dry run of the playbook execution.
- This causes Ansible to report what changes would have occurred if the playbook were executed, but does not make any actual changes to managed hosts.

```
[student@workstation ~]$ ansible-playbook -C webserver.yml

PLAY [play to setup web server] ************************************************

TASK [Gathering Facts] *********************************************************
ok: [servera.lab.example.com]

TASK [latest httpd version installed] ******************************************
changed: [servera.lab.example.com]

PLAY RECAP *********************************************************************
servera.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Check Mode: "Dry Run"

- **`ansible-playbook --check`** will not make any changes on managed hosts.

- Not all modules fully support check mode:
  - Tasks using modules that support **check mode** will report what changes they would have made.
  - Tasks using modules that do not support **check mode** will also take no action, but will not report what changes they might have made.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Check Mode: "Dry Run"

- You can configure a task to never run in check mode: set **`check_mode: no`** on the task
- This is useful when a module needs to run in normal mode for other modules to work
- The second task in the example at right will break in check mode if **check_mode: no** is not set on the first task, because in that case the variable **date_result** will be undefined

```
 - name: check_mode example
   command:
     cmd: /usr/bin/date +%s
   register: date_result
   check_mode: no

- name: check date_result
  debug:
    var: date_result.stdout
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Check Mode: Checking State

- You can configure a task to *always* run in check mode: set **`check_mode: yes`** on the task
- In this case, you run the play normally and that task still runs in check mode
- If the task runs and it would normally change the remote system, the task will report CHANGED but not change anything
- You can register a variable and react to the result without changing the managed host
- In the example at right, if the first task reports it would change a host to "correct" its configuration if it were not in check mode, the second task will fail on that host

```
- name: check_mode example
   yum:
      name: httpd
      state: absent
   register: httpd_pkg_result
   check_mode: yes

- name: Fail if httpd is not absent
  fail:
  when: httpd_pkg_result['changed']
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Verbosity

Control the level of output Ansible produces as the playbook executes with the **`-v`** or **`--verbose`** flags. The **`-v`** flag can be stacked to add verbosity like **`-vvv`**. Up to 5 v's can be used.

Verbosity Info	
-v	Shows config file used and task args.
-vv	Shows Ansible version info and meta data about handlers.
-vvv	Shows connection info and pretty prints json output.
-vvvv	Shows deeper connection info.
-vvvvv	Full verbosity. Includes winrm connection debugging.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Using Test Modules

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Objective

Investigate issues by using modules.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Debug Module

The debug module prints a message or a variable to the Ansible output. The debug module is useful for debugging variables or expressions.

Parameter	Description
msg	A customized message to be printed. Can contain variables.
var	A variable to print. No need for `{{ }}`. Mutually exclusive with **`msg`**.
verbosity	A number to control which verbosity level the debug module runs on. If set to 3, it will only execute the task when the playbook was ran with **`-vvv`** or above.

```
- debug:
    msg: System {{ inventory_hostname }} has uuid {{ ansible_product_uuid }}
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Assert Module

The assert module checks that the given expressions are true with an optional custom message. The assert module is useful for validating your variables and conditions are what you expect.

Parameter	Description
that	A list of string expressions. Same format as **`when`** statements
quiet	A boolean to avoid verbose output.
success_msg	A custom message used when the asserted expressions pass.
fail_msg	A custom message used when the asserted expressions fail.

```
- name: After version 2.7 both 'msg' and 'fail_msg' can customize failing assertion message
  assert:
    that:
      - my_param <= 100
      - my_param >= 0
    fail_msg: "'my_param' must be between 0 and 100"
    success_msg: "'my_param' is between 0 and 100"
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Fail Module

The fail module forces the Ansible playbook to fail with a custom message. The fail module is useful when testing certain conditions with **`when`**.

Parameter	Description
msg	A customized message to be printed for failing the playbook.

```
- fail:
    msg: The system may not be provisioned according to the CMDB status.
  when: cmdb_status != "to-be-staged"
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Meta tasks

Meta tasks are special tasks that can influence Ansible internal execution or state. Meta tasks can be used anywhere within the playbook. Meta is not actually a module or an action plugin, so it cannot be overwritten or used in loops.

Useful tasks for debugging are **`clear_facts`** and **`clear_host_errors`**. These two in particular can help debugging by allowing you to create multiple plays in a single playbook and reset the state between each play.

Choices*	Description
clear_facts	Removes the gathered facts for the hosts specified in the play's list of hosts, including removing the facts from the fact cache.
clear_host_errors	Clears the failed state from hosts specified in the play's list of hosts.

* Not all choices described in this table.

```
- name: Clear gathered facts from all currently targeted hosts
  meta: clear_facts
```

```
- meta: clear_host_errors
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Using the Playbook Debugger

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Objective

Investigate issues using the playbook debugger.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Playbook Debugger

- Ansible includes a debugger as part of the strategy plugins.
- You can check or set the value of variables, update module arguments, and re-run a task with new variables and arguments to help resolve the cause of a failure.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Debugger Config and Environment Variable

The debugger can be enabled in your Ansible configuration file (`ansible.cfg`):

```
[defaults]
enable_task_debugger = True
```

The debugger can also be set as an environment variable:

```
ANSIBLE_ENABLE_TASK_DEBUGGER=True ansible-playbook -i hosts site.yml
```

When using either method, any failed or unreachable task will start the debugger, unless disabled explicitly in the playbook.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Playbook Debugger Keyword

The **`debugger`** keyword can be used on any block where you provide a **name** attribute, such as a play, role, block, or task. The following options override any global configuration to enable or disable the debugger.

Choices	Description
always	Always invoke the debugger, regardless of the outcome.
never	Never invoke the debugger, regardless of the outcome.
on_failed	Only invoke the debugger if the task fails.
on_unreachable	Only invoke the debugger if the host was unreachable.
on_skipped	Only invoke the debugger if the task is skipped.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Debugger Keyword Examples

On a task

```
- name: Execute a command
  command: false
  debugger: on_failed
```

On a play

```
- name: Play
  hosts: all
  debugger: on_skipped
  tasks:
    - name: Execute a command
      command: true
      when: False
```

The more specific one wins

```
- name: Play
  hosts: all
  debugger: never
  tasks:
    - name: Execute a command
      command: false
      debugger: on_failed
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Debugger Commands

Commands	Description
p(print) task/task_vars/host/result	Print values used to execute a module.
task.args[key] = value	Update module's arguments.
task_vars[key] = value	Update the vars used in a task.
u(pdate_task)	Recreates the task from the original task data structure and templates with updated **`task_vars`**.
r(edo)	Run the task again.
c(ontinue)	Continue the playbook execution.
q(uit)	Quit from the debugger. The playbook execution is aborted.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Debugger Example

Sample playbook:

```
- hosts: test
  strategy: debug
  gather_facts: yes
  vars:
    pkg_name: not_exist
  tasks:
    - name: install package
      apt: name={{ pkg_name }}
```

Update the module's argument:

```
[192.0.2.10] TASK: install package (debug)> p task.args
{u'name': u'{{ pkg_name }}'}
[192.0.2.10] TASK: install package (debug)> task.args['name'] = 'bash'
[192.0.2.10] TASK: install package (debug)> p task.args
{u'name': 'bash'}
[192.0.2.10] TASK: install package (debug)> redo
```

Update the task's variable:

```
[192.0.2.10] TASK: install package (debug)> p task_vars['pkg_name']
u'not_exist'
[192.0.2.10] TASK: install package (debug)> task_vars['pkg_name'] = 'bash'
[192.0.2.10] TASK: install package (debug)> p task_vars['pkg_name']
'bash'
[192.0.2.10] TASK: install package (debug)> update_task
[192.0.2.10] TASK: install package (debug)> redo
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Old Syntax: Debug Strategy

- Old playbooks might start the debugger by calling the debug strategy:

```
- hosts: test
  strategy: debug
  tasks:
  ...
```

- When using the debug strategy, only tasks that fail trigger the debugger.
- The same actions work for both the playbook debugger and the debug strategy.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Installing and Using ansible-lint

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Objective

Validate playbooks using ansible-lint.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Installing ansible-lint

To install ansible-lint, run:

```
$ pip install ansible-lint
```

To install from source, run:

```
$ pip install git+https://github.com/ansible/ansible-lint.git
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Configuring ansible-lint

Ansible-lint supports local configuration via the `.ansible-lint` configuration file.

The configuration file location can also be set with the configuration file CLI flag:

```
-c path/to/file
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Configuring ansible-lint

The following values are supported in the ansible-lint configuration file:

Configuration Section	Description
exclude_paths	Paths to directories or files to skip.
parseable	Parseable output in the format of pep8.
quiet	Quieter, although not silent output.
rulesdir	Specify one or more rules directories. These override the default rules unless **use_default_rules** is also set.
skip_list	Only check rules whose id/tags do not match these values.
tags	Only check rules whose id/tags match these values.
use_default_rules	Use default rules ['/path/to/ansible-lint/lib/ansiblelint/rules'] in addition to any extra rules directories specified with **rulesdir**.
verbosity	Set the verbosity level.

```
exclude_paths:
  - ./my/excluded/directory/
  - ./my/other/excluded/directory/
  - ./last/excluded/directory/
parseable: true
quiet: true
rulesdir:
  - ./rule/directory/
skip_list:
  - skip_this_tag
  - and_this_one_too
  - skip_this_id
  - '401'
tags:
  - run_this_tag
use_default_rules: true
verbosity: 1
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Running ansible-lint

The **`ansible-lint`** command accepts a list of Ansible playbook files or role directories.

Examples:

```
$ ansible-lint playbook.yml
$ ansible-lint geerlingguy.apache
$ ansible-lint roles/*
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Example Output

```
$ ansible-lint my-role/
[502] All tasks should be named
/home/user/ansible/my-role/tasks/main.yml:7
Task/Handler: stat __file__=/home/user/ansible/my-role/tasks/main.yml __line__=8 path={{path}}

[201] Trailing whitespace
/home/user/ansible/my-role/tasks/main.yml:9
    path: "{{path}}"

[206] Variables should have spaces before and after: {{ var_name }}
/home/user/ansible/my-role/tasks/main.yml:9
    path: "{{path}}"
```

The output specifies each linting rule that was broken, gives an example, and specifies the line the error occurred on.

If no errors are found, there will be no output.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Ansible-lint Rules

The default rules are located in `ansible-lint/lib/ansiblelint/rules`.

Custom rules can be set in the configuration file or via the CLI. The default rules can be used with custom rules by using both the **`-R`** flag to use the default rules and the **`-r`** flag to use custom rules.

```
$ ansible-lint -r ~/dir_of_custom_rules/ -R playbook.yml
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Excluding Ansible-lint Rules

To exclude rules from the available set of rules, use the **`-x SKIP_LIST`** option. You can specify tags and ids with the **`-x`** flag. Rules can also be skipped by configuring skip_list in the `.ansible-lint` configuration file.

```
$ ansible-lint -x formatting,502 playbook.yml
```

View the list of available tags and ids to use with ansible-lint with the following command:

```
$ ansible-lint -T
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Example Output Excluding Rules

```
$ ansible-lint my-role/ -x formatting
[502] All tasks should be named
/home/user/ansible/my-role/tasks/main.yml:7
Task/Handler: stat __file__=/home/user/ansible/my-role/tasks/main.yml __line__=8 path={{path}}
```


©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Conclusion

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Learn More about Red Hat Training and Certification

- Congratulations on completing this course! Want to learn more? Visit the Red Hat Training and Certification page to explore Red Hat courses and certifications.
- Join the Red Hat Learning Community to ask questions and access a collaborative learning environment that enables open source skill development.

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos


facebook.com/redhatinc


twitter.com/RedHat

©2020 Red Hat, Inc., licensed to Pluralsight, LLC. All trademarks, service marks, and logos used herein are the property of their respective owners.

Red Hat