
ansible-runner-role Documentation

Release 1.2.1

Vladimir Botka

Jul 07, 2020

TABLE OF CONTENTS:

1	User's guide	3
1.1	Introduction	3
1.2	Installation	4
1.3	Playbook	4
1.4	Debug	5
1.5	Tags	5
1.6	Tasks	5
1.7	Variables	15
1.8	Default variables	15
1.9	Best practice	16
2	Annotated source code	17
2.1	Tasks	17
3	Copyright	25
4	Legal Notice	27
5	Indices and tables	29

This [role](#) and the documentation is work in progress. Please feel free to [share your feedback and report issues](#). Contributions are welcome.

USER'S GUIDE

Table of Contents

- *User's guide*
 - *Introduction*
 - *Installation*
 - *Playbook*
 - *Debug*
 - *Tags*
 - *Tasks*
 - * *Ansible Runner packages*
 - *Synopsis*
 - *Examples*
 - *Variables*
 - *Default variables*
 - *Best practice*

1.1 Introduction

Run this role to install and configure Ansible Runner. Optionally configure cron to periodically run Ansible playbooks.

- Ansible role: `ansible_runner`
- Supported systems:
 - FreeBSD Supported Production Releases
 - Ubuntu Supported Releases
- (Requirements in future releases: `ansible_lib`)

Note:

- The utility *ansible-runner* is not part of standard Ansible installation. See [Installing Ansible Runner](#)

See also:

- [Ansible Runner documentation](#)
- [Ansible Runner source code](#)
- [REST API ansible-runner-service](#)

1.2 Installation

The most convenient way how to install an Ansible role is to use Ansible Galaxy CLI `ansible-galaxy`. The utility comes with the standard Ansible package and provides the user with a simple interface to the Ansible Galaxy's services. For example, take a look at the current status of the role

```
shell> ansible-galaxy info vbotka.ansible_runner
```

and install it

```
shell> ansible-galaxy install vbotka.ansible_runner
```

Install the library of tasks (for future releases)

```
shell> ansible-galaxy install vbotka.ansible_lib
```

See also:

- To install specific versions from various sources see [Installing content](#).
- Take a look at other roles `shell> ansible-galaxy search --author=vbotka`

1.3 Playbook

Below is a simple playbook that calls this role at a single host `srv.example.com` (2)

```
1 shell> cat ansible-runner.yml
2 - hosts: srv.example.com
3   gather_facts: true
4   connection: ssh
5   remote_user: admin
6   become: yes
7   become_user: root
8   become_method: sudo
9   roles:
10    - vbotka.ansible_runner
```

Note: `gather_facts: true` (3) must be set to gather facts needed to evaluate OS-specific options of the role. For example to install packages the variable `ansible_os_family` is needed to select the appropriate Ansible module.

See also:

- For details see [Connection Plugins](#) (4-5)

- See also [Understanding Privilege Escalation \(6-8\)](#)

1.4 Debug

To see additional debug information enable debug output in the configuration

```
ar_debug: true
```

, or set the extra variable in the command

```
shell> ansible-playbook ansible-runner.yml -e 'ar_debug=true'
```

Note: The debug output of this role is optimized for the **yaml** callback plugin. Set this plugin for example in the environment `shell> export ANSIBLE_STDOUT_CALLBACK=yaml`.

See also:

- [Playbook Debugger](#)

1.5 Tags

The tags provide the user with a very useful tool to run selected tasks of the role. To see what tags are available list the tags of the role with the command

```
shell> ansible-playbook ansible-runner.yml --list-tags

playbook: ansible-runner.yml

play #1 (srv.example.com): srv.example.com TAGS: []
    TASK TAGS: [always, ar_config, ar_debug, ar_links, ar_packages, ar_vars]
```

For example, display the list of the variables and their values with the tag `ar_debug` (when the debug is enabled `ar_debug: true`)

```
shell> ansible-playbook ansible-runner.yml -t ar_debug
```

See what packages will be installed

```
shell> ansible-playbook ansible-runner.yml -t ar_packages --check
```

Install packages and exit the play

```
shell> ansible-playbook ansible-runner.yml -t ar_packages
```

1.6 Tasks

Test single tasks at single remote host `test_01`. Create a playbook

```
shell> cat ansible.yml
- hosts: test_01
  become: true
  roles:
    - vbotka.ansible_runner
```

Customize configuration in `host_vars/test_01/ar-*.yaml` and check the syntax

```
shell> ansible-playbook ansible-runner.yml --syntax-check
```

Then dry-run the selected task and see what will be changed. Replace `<tag>` with valid tag.

```
shell> ansible-playbook ansible-runner.yml -t <tag> --check --diff
```

When all seems to be ready run the command. Run the command twice and make sure the playbook and the configuration is idempotent

```
shell> ansible-playbook ansible-runner.yml -t <tag>
```

1.6.1 Ansible Runner packages

Synopsis

`ansible-runner` can be installed by `pip` or from distribution's packages, and ports.

See also:

- Annotated Source code [packages.yml](#)

Examples

Example 1: Install ansible-runner in Ubuntu by pip for admin

Create a playbook

```
shell> ansible-runner.yml
- hosts: test_01
  become: true
  roles:
    - vbotka.ansible_runner
```

Create `host_vars/test_01/ansible-runner.yml`

```
shell> cat host_vars/test_01/ansible-runner.yml
ar_install: false
ar_pip_install: true
ar_debug: false
ar_owner: admin
```

Install ansible-runner

```
shell> ansible-playbook ansible-runner.yml -e "ar_install=true"
...
```

(continues on next page)

(continued from previous page)

```
TASK [vbotka.ansible_runner : packages: Install Ansible Runner pip packages for_
↪admin]
ok: [test_01] => (item={'name': 'ansible-runner'})
```

Show ansible-runner package installed by pip for admin

```
shell> whoami
admin

shell> pip list | grep ansible-runner
ansible-runner      1.4.6
```

Example 2: Install ansible-runner in FreeBSD from the port

Create a playbook

```
shell> ansible-runner.yml
- hosts: test_01
  become: true
  roles:
    - vbotka.ansible_runner
```

Create *host_vars/test_01/ansible-runner.yml*

```
shell> cat host_vars/test_01/ansible-runner.yml
<TBD>
```

Install ansible-runner

```
shell> ansible-playbook ansible-runner.yml -e "ar_install=true"
<TBD>
...
TASK [vbotka.ansible_runner : packages: Install Ansible Runner pip packages for_
↪admin]
ok: [test_01] => (item={'name': 'ansible-runner'})
```

Show ansible-runner package installed by pip for admin

```
<TBD>
shell> whoami

shell> which ansible-runner
ansible-runner      1.4.6
```

Example 3: Run Ansible playbooks in cron

ssh-agent is needed to provide the ssh connection plugin with the password to the private key, when used. The script below is executed by the command interpreter for login shells

```
1 cntrlr> cat /home/admin/.profile
2 if [ -n "$BASH_VERSION" ]; then
3     if [ -f "$HOME/.bashrc" ]; then
```

(continues on next page)

(continued from previous page)

```

4      . "$HOME/.bashrc"
5  fi
6  if [ -f "$HOME/.bashrc_ssh" ]; then
7      . "$HOME/.bashrc_ssh"
8  fi
9  fi
10 if [ -d "$HOME/bin" ] ; then
11     PATH="$HOME/bin:$PATH"
12 fi

```

and will start ssh-agent on login and prepare *SSH_ENV* (5)

```

1 cntrlr> cat /home/admin/.bashrc_ssh
2 SSH_ENV="$HOME/.ssh/environment"
3 function start_agent {
4     echo "Initialising new SSH agent..."
5     /usr/bin/ssh-agent | sed 's/^echo/#echo/' > "${SSH_ENV}"
6     echo succeeded
7     chmod 600 "${SSH_ENV}"
8     . "${SSH_ENV}" > /dev/null
9     /usr/bin/ssh-add;
10 }
11 if [ -f "${SSH_ENV}" ]; then
12     . "${SSH_ENV}" > /dev/null
13     #ps ${SSH_AGENT_PID} doesn't work under cywgin
14     ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
15         start_agent;
16     }
17 else
18     start_agent;
19 fi

```

Example of *.ssh/environment* created by *ssh-agent*

```

cntrlr> cat /home/admin/.ssh/environment
SSH_AUTH_SOCK=/tmp/ssh-8fUkZ7qOzVPs/agent.5214; export SSH_AUTH_SOCK;
SSH_AGENT_PID=5216; export SSH_AGENT_PID;
#echo Agent pid 5216;

```

See also:

- Start ssh-agent on login

Wrapper of *ansible-runner* will source *.ssh/environment* (9) and run the *playbook* from the *project* (10)

```

1 cntrlr> cat /home/admin/bin/arwrapper.sh
2 #!/bin/bash
3 runner=$HOME/bin/ansible-runner
4 project=$HOME/.ansible/runner/$2
5 playbook=${3:-all.yml}
6 case "$1" in
7     run)
8         echo $(date '+%Y-%m-%d %H:%M:%S') $0
9         source $HOME/.ssh/environment
10        $runner run $project -p $playbook
11        exit 0
12    ;;

```

(continues on next page)

(continued from previous page)

```

13     clean)
14         rm -rf $project/artifacts
15         exit 0
16         ;;
17     *)
18         printf "$0: run|clean project [playbook]\n"
19         exit 1
20         ;;
21 esac

```

The script below will use *arwrapper.sh* (9) to run the playbook *pb-01.yml* (7) in the projects *test_01*, *test_02*, and *test_03* (6). If the command (9) succeeds the script will print *[OK]* report (14). If you don't want to receive email on success remove this line and optionally enable the cleaning of the artifacts (13)

```

1  cntrlr> cat /home/admin/bin/ansible-cron-test.sh
2  #!/bin/bash
3  marker=$(printf "%75s" | sed "s/ /.g")
4  rc=0
5  cmd=$HOME/bin/arwrapper.sh
6  projects="test_01 test_02 test_03"
7  playbook=pb-01.yml
8  for project in ${projects[@]}; do
9      if ! out=$( $cmd run $project $playbook ); then
10         rc=1
11         printf "[ERR] $out\n$marker\n"
12     else
13         # $cmd clean $project
14         printf "[OK] $project $playbook PASSED\n"
15     fi
16 done
17 exit $rc

```

Schedule the script in *cron*

```

cntrlr> whoami
admin
cntrlr> crontab -l
MAILTO=admin
#Ansible: Ansible runner daily test
50 20 * * * $HOME/bin/ansible-cron-test.sh

```

In our case the */etc/aliases* redirect the emails for *root* to the user *admin*. Cron will report the result of the script *ansible-cron-test.sh*. If you want to receive email on a failure only remove the *[OK]* report from the script and optionally clean the *artifacts*. The *artifacts* will be available for a review if the script fails

```

Date: Tue,  7 Jul 2020 20:50:06 +0200 (CEST)
From: Cron Daemon <root@cntrlr.example.com>
To: admin@cntrlr.example.com
Subject: Cron <admin@cntrlr> $HOME/bin/ansible-cron-test.sh

[OK] test_01 pb-01.yml PASSED
[OK] test_02 pb-01.yml PASSED
[OK] test_03 pb-01.yml PASSED

```

Example of the project's directory without the artifacts. The artifacts will be created by *ansible-runner*

```
cntrlr> tree /home/vlado/.ansible/runner/test_01
/home/admin/.ansible/runner/test_01
├── env
├── inventory
│   └── hosts
├── project
│   ├── ansible.cfg
│   ├── group_vars
│   ├── host_vars
│   └── pb-01.yml
```

See also:

- [Runner Input Directory Hierarchy](#)

Example of the project's artifacts

```
cntrl> tree /home/admin/.ansible/runner/test_01/artifacts/
/home/admin/.ansible/runner/test_01/artifacts
├── aaa5d36e-e8d4-432a-ab52-b69062c85311
│   ├── command
│   ├── fact_cache
│   ├── job_events
│   │   ├── 1-2b5c9412-f0c4-45dc-a425-5c8c29e37ec0.json
│   │   ├── 2-5ce0c5a2-1f02-cdab-8869-00000000001f.json
│   │   ├── 3-5ce0c5a2-1f02-cdab-8869-000000000021.json
│   │   ├── 4-28749e27-409a-46c4-9551-7ce80c02be83.json
│   │   ├── 5-997d90c1-6357-45c6-8df9-437c2940c74e.json
│   │   └── 6-6e41cf27-8c1e-4266-9ffb-8a54375bd4cc.json
│   ├── rc
│   ├── status
│   └── stdout
```

See also:

- [Runner Artifacts Directory Hierarchy](#)
- `ansible_lib: al_runner_events`

Example of a playbook used in the test

```
cntrlr> cat /home/admin/.ansible/runner/test_01/project/pb-01.yml
- hosts: test_01
  remote_user: admin
  gather_facts: no
  tasks:
    - debug:
        msg: TEST
```

Example 1: Install ansible-runner in Ubuntu by pip for admin

Create a playbook

```
shell> ansible-runner.yml
- hosts: test_01
  become: true
  roles:
    - vbotka.ansible_runner
```

Create *host_vars/test_01/ansible-runner.yml*

```
shell> cat host_vars/test_01/ansible-runner.yml
ar_install: false
ar_pip_install: true
ar_debug: false
ar_owner: admin
```

Install ansible-runner

```
shell> ansible-playbook ansible-runner.yml -e "ar_install=true"
...
TASK [vbotka.ansible_runner : packages: Install Ansible Runner pip packages for_
↪admin]
ok: [test_01] => (item={'name': 'ansible-runner'})
```

Show ansible-runner package installed by pip for admin

```
shell> whoami
admin

shell> pip list | grep ansible-runner
ansible-runner          1.4.6
```

Example 2: Install ansible-runner in FreeBSD from the port

Create a playbook

```
shell> ansible-runner.yml
- hosts: test_01
  become: true
  roles:
    - vbotka.ansible_runner
```

Create *host_vars/test_01/ansible-runner.yml*

```
shell> cat host_vars/test_01/ansible-runner.yml
<TBD>
```

Install ansible-runner

```
shell> ansible-playbook ansible-runner.yml -e "ar_install=true"
<TBD>
...
TASK [vbotka.ansible_runner : packages: Install Ansible Runner pip packages for_
↪admin]
ok: [test_01] => (item={'name': 'ansible-runner'})
```

Show ansible-runner package installed by pip for admin

```
<TBD>
shell> whoami

shell> which ansible-runner
ansible-runner          1.4.6
```

Example 3: Run Ansible playbooks in cron

ssh-agent is needed to provide the ssh connection plugin with the password to the private key, when used. The script below is executed by the command interpreter for login shells

```
1 cntlr> cat /home/admin/.profile
2 if [ -n "$BASH_VERSION" ]; then
3     if [ -f "$HOME/.bashrc" ]; then
4         . "$HOME/.bashrc"
5     fi
6     if [ -f "$HOME/.bashrc_ssh" ]; then
7         . "$HOME/.bashrc_ssh"
8     fi
9 fi
10 if [ -d "$HOME/bin" ] ; then
11     PATH="$HOME/bin:$PATH"
12 fi
```

and will start ssh-agent on login and prepare *SSH_ENV* (5)

```
1 cntlr> cat /home/admin/.bashrc_ssh
2 SSH_ENV="$HOME/.ssh/environment"
3 function start_agent {
4     echo "Initialising new SSH agent..."
5     /usr/bin/ssh-agent | sed 's/^echo/#echo/' > "${SSH_ENV}"
6     echo succeeded
7     chmod 600 "${SSH_ENV}"
8     . "${SSH_ENV}" > /dev/null
9     /usr/bin/ssh-add;
10 }
11 if [ -f "${SSH_ENV}" ]; then
12     . "${SSH_ENV}" > /dev/null
13     #ps ${SSH_AGENT_PID} doesn't work under cywgin
14     ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
15         start_agent;
16     }
17 else
18     start_agent;
19 fi
```

Example of *.ssh/environment* created by *ssh-agent*

```
cntlr> cat /home/admin/.ssh/environment
SSH_AUTH_SOCK=/tmp/ssh-8fUkZ7qOzVPs/agent.5214; export SSH_AUTH_SOCK;
SSH_AGENT_PID=5216; export SSH_AGENT_PID;
#echo Agent pid 5216;
```

See also:

- Start ssh-agent on login

Wrapper of *ansible-runner* will source *.ssh/environment* (9) and run the *playbook* from the *project* (10)

```
1 cntlr> cat /home/admin/bin/arwrapper.sh
2 #!/bin/bash
3 runner=$HOME/bin/ansible-runner
4 project=$HOME/.ansible/runner/$2
5 playbook=${3:-all.yml}
```

(continues on next page)

(continued from previous page)

```

6 case "$1" in
7     run)
8         echo $(date '+%Y-%m-%d %H:%M:%S') $0
9         source $HOME/.ssh/environment
10        $runner run $project -p $playbook
11        exit 0
12        ;;
13    clean)
14        rm -rf $project/artifacts
15        exit 0
16        ;;
17    *)
18        printf "$0: run|clean project [playbook]\n"
19        exit 1
20        ;;
21 esac

```

The script below will use *arwrapper.sh* (9) to run the playbook *pb-01.yml* (7) in the projects *test_01*, *test_02*, and *test_03* (6). If the command (9) succeeds the script will print *[OK]* report (14). If you don't want to receive email on success remove this line and optionally enable the cleaning of the artifacts (13)

```

1 cntrlr> cat /home/admin/bin/ansible-cron-test.sh
2 #!/bin/bash
3 marker=$(printf "%75s" | sed "s/ /. /g")
4 rc=0
5 cmd=$HOME/bin/arwrapper.sh
6 projects="test_01 test_02 test_03"
7 playbook=pb-01.yml
8 for project in ${projects[@]}; do
9     if ! out=$( $cmd run $project $playbook ); then
10         rc=1
11         printf "[ERR] $out\n$marker\n"
12     else
13         # $cmd clean $project
14         printf "[OK] $project $playbook PASSED\n"
15     fi
16 done
17 exit $rc

```

Schedule the script in *cron*

```

cntrlr> whoami
admin
cntrlr> crontab -l
MAILTO=admin
#Ansible: Ansible runner daily test
50 20 * * * $HOME/bin/ansible-cron-test.sh

```

In our case the */etc/aliases* redirect the emails for *root* to the user *admin*. Cron will report the result of the script *ansible-cron-test.sh*. If you want to receive email on a failure only remove the *[OK]* report from the script and optionally clean the *artifacts*. The *artifacts* will be available for a review if the script fails

```

Date: Tue,  7 Jul 2020 20:50:06 +0200 (CEST)
From: Cron Daemon <root@cntrlr.example.com>
To: admin@cntrlr.example.com
Subject: Cron <admin@cntrlr> $HOME/bin/ansible-cron-test.sh

```

(continues on next page)

(continued from previous page)

```
[OK] test_01 pb-01.yml PASSED
[OK] test_02 pb-01.yml PASSED
[OK] test_03 pb-01.yml PASSED
```

Example of the project's directory without the artifacts. The artifacts will be created by *ansible-runner*

```
cntrlr> tree /home/vlado/.ansible/runner/test_01
/home/admin/.ansible/runner/test_01
├── env
├── inventory
│   └── hosts
├── project
│   ├── ansible.cfg
│   ├── group_vars
│   ├── host_vars
│   └── pb-01.yml
```

See also:

- [Runner Input Directory Hierarchy](#)

Example of the project's artifacts

```
cntrl> tree /home/admin/.ansible/runner/test_01/artifacts/
/home/admin/.ansible/runner/test_01/artifacts
├── aaa5d36e-e8d4-432a-ab52-b69062c85311
│   ├── command
│   ├── fact_cache
│   ├── job_events
│   │   ├── 1-2b5c9412-f0c4-45dc-a425-5c8c29e37ec0.json
│   │   ├── 2-5ce0c5a2-1f02-cdab-8869-00000000001f.json
│   │   ├── 3-5ce0c5a2-1f02-cdab-8869-000000000021.json
│   │   ├── 4-28749e27-409a-46c4-9551-7ce80c02be83.json
│   │   ├── 5-997d90c1-6357-45c6-8df9-437c2940c74e.json
│   │   └── 6-6e41cf27-8c1e-4266-9ffb-8a54375bd4cc.json
│   ├── rc
│   ├── status
│   └── stdout
```

See also:

- [Runner Artifacts Directory Hierarchy](#)
- `ansible_lib: al_runner_events`

Example of a playbook used in the test

```
cntrlr> cat /home/admin/.ansible/runner/test_01/project/pb-01.yml
- hosts: test_01
  remote_user: admin
  gather_facts: no
  tasks:
    - debug:
        msg: TEST
```

1.7 Variables

In this chapter we describe role's default variables stored in the directory **defaults**.

See also:

- Ansible variable precedence: Where should I put a variable?

1.8 Default variables

<TBD>

[defaults/main.yml]

```

1  ---
2  # defaults ansible_runner
3
4  ar_install: true
5  ar_debug: false
6  ar_backup_conf: false
7
8  # Install distro packages or pip
9  # false - distro packages, true - pip
10 # ar_pip_install: false # OS specific variable see vars/defaults
11
12 # FreeBSD
13 freebsd_install_retries: 10
14 freebsd_install_delay: 5
15 freebsd_install_method: "packages"
16 # freebsd_install_method: "ports"
17 freebsd_use_packages: true
18
19 # Linux
20 linux_install_retries: 10
21 linux_install_delay: 5
22
23 # Python
24 pip_install_retries: 10
25 pip_install_delay: 5
26 # pip_extraargs: "" # Optional
27
28 # pip package dependent
29 # ar_pip_executable: "pip3" # OS specific variable see vars/defaults
30 # ar_pip_requirements: [] # Optional
31
32 # Configuration
33 ar_config: []
34
35 # Links
36 # ar_links: [] # OS specific variable see vars/defaults
37
38 # EOF
39 ...

```

1.9 Best practice

Display the variables for debug if needed. Then disable this task `ar_debug: false` to speedup the playbook

```
shell> ansible-playbook ansible-runner.yml -t ar_debug
```

Install packages Then disable this task `ar_install: false` to speedup the playbook

```
shell> ansible-playbook ansible-runner.yml -t ar_packages \
-e 'ar_install=true'
```

The role and the configuration data in the examples are idempotent. Once the installation and configuration have passed there should be no changes reported by *ansible-playbook* when running the playbook repeatedly. Disable debug, and install to speedup the playbook

```
shell> ansible-playbook ansible-runner.yml
```

ANNOTATED SOURCE CODE

Table of Contents

- *Annotated source code*
 - *Tasks*
 - * *main.yml*
 - * *config.yml*
 - * *debug.yml*
 - * *links.yml*
 - * *packages.yml*
 - * *vars.yml*

2.1 Tasks

2.1.1 main.yml

Synopsis: Tasks of the playbook.

Description of the task.

[main.yml]

```
1 ---
2 # tasks for ansible_runner
3
4 - import_tasks: vars.yml
5   tags: [ar_vars, always]
6
7 - import_tasks: debug.yml
8   when: ar_debug|bool
9   tags: [ar_debug, always]
10
11 - import_tasks: packages.yml
12   when: ar_install|bool
13   tags: ar_packages
14
```

(continues on next page)

(continued from previous page)

```

15 - import_tasks: links.yml
16   tags: ar_links
17
18 - import_tasks: config.yml
19   tags: ar_config
20
21 # EOF
22 ...

```

2.1.2 config.yml

Synopsis: Configure config.

Description of the task.

[config.yml]

```

1 ---
2
3 - name: "config: configure ansible-runner"
4   debug:
5     msg: No config
6     when: ar_debug|bool
7
8 # EOF
9 ...

```

2.1.3 debug.yml

Synopsis: Configure debug.

Description of the task.

[debug.yml]

```

1 ---
2
3 - name: "debug: Ansible Runner"
4   vars:
5     msg: |
6       ansible_os_family [{{ ansible_os_family }}]
7       ansible_distribution [{{ ansible_distribution }}]
8       ansible_distribution_major_version [{{ ansible_distribution_major_version }}]
9       ansible_distribution_version [{{ ansible_distribution_version }}]
10      ansible_distribution_release [{{ ansible_distribution_release }}]
11      ansible_python_version [{{ ansible_python_version }}]
12
13      ar_install [{{ ar_install }}]
14
15      freebsd_install_method [{{ freebsd_install_method }}]
16      freebsd_use_packages [{{ freebsd_use_packages }}]
17      freebsd_install_retries [{{ freebsd_install_retries }}]
18      freebsd_install_delay [{{ freebsd_install_delay }}]
19
20      linux_install_retries [{{ linux_install_retries }}]

```

(continues on next page)

(continued from previous page)

```

21     linux_install_delay [{{ linux_install_delay }}]
22
23     pip_install_retries [{{ pip_install_retries }}]
24     pip_install_delay [{{ pip_install_delay }}]
25     pip_extraargs [{{ pip_extraargs|default('UNDEFINED') }}]
26
27     ar_pip_install [{{ ar_pip_install }}]
28     ar_pip_executable [{{ ar_pip_executable }}]
29     ar_pip_requirements [{{ pip_requirements|default('UNDEFINED') }}]
30
31     ansible_user_id [{{ ansible_user_id }}]
32
33     ar_owner [{{ ar_owner }}]
34     ar_backup_conf [{{ ar_backup_conf }}]
35
36     ar_packages
37     {{ ar_packages|to_nice_yaml }}
38     ar_links
39     {{ ar_links|to_nice_yaml }}
40
41     ar_config
42     {{ ar_config|to_nice_yaml }}
43
44     debug:
45         msg: "{{ msg.split('\n') }}"
46
47     # EOF
48     ...

```

2.1.4 links.yml

Synopsis: Configure links.

Description of the task.

[links.yml]

```

1  ---
2
3  - name: "links: Create directories for links"
4    file:
5      state: "directory"
6      dest: "{{ item.dest|dirname }}"
7      loop: "{{ ar_links }}"
8
9  - name: "links: Create links"
10   file:
11     state: "link"
12     src: "{{ item.src }}"
13     dest: "{{ item.dest }}"
14     force: "{{ item.force|default(false) }}"
15     loop: "{{ ar_links }}"
16
17   # EOF
18   ...

```

2.1.5 packages.yml

Synopsis: Configure packages.

Description of the task.

[packages.yml]

```

1  ---
2
3  # packages -----
4
5  # FreeBSD
6  - name: "packages: Install Ansible Runner FreeBSD packages"
7    block:
8      - name: "packages: Install Ansible Runner packages FreeBSD"
9        pkgng:
10          name: "{{ item.name }}"
11          loop: "{{ ar_packages }}"
12          register: result
13          until: result is succeeded
14          retries: "{{ freebsd_install_retries }}"
15          delay: "{{ freebsd_install_delay }}"
16      - name: "packages: Debug FreeBSD packages"
17        when: ar_debug|bool
18        debug:
19          var: result
20    when:
21      - not ar_pip_install
22      - ansible_os_family == "FreeBSD"
23      - freebsd_install_method|lower == "packages"
24
25  - name: "packages: Install FreeBSD ports"
26    block:
27      - name: "packages: Install Ansible Runner ports FreeBSD"
28        portinstall:
29          name: "{{ item.name }}"
30          use_packages: "{{ freebsd_use_packages }}"
31          loop: "{{ ar_packages }}"
32          register: result
33          until: result is succeeded
34          retries: "{{ freebsd_install_retries }}"
35          delay: "{{ freebsd_install_delay }}"
36      - name: "packages: Debug FreeBSD ports"
37        when: ar_debug|bool
38        debug:
39          var: result
40    when:
41      - not ar_pip_install
42      - ansible_os_family == "FreeBSD"
43      - freebsd_install_method|lower == "ports"
44
45  # Linux
46  - name: "packages: Install Ansible Runner packages Linux"
47    block:
48      - name: "packages: Install Ansible Runner packages Linux"
49        package:
50          name: "{{ item.name }}"
51          loop: "{{ ar_packages }}"

```

(continues on next page)

(continued from previous page)

```

52     register: result
53     until: result is succeeded
54     retries: "{{ linux_install_retries }}"
55     delay: "{{ linux_install_delay }}"
56 - name: "packages: Debug Linux"
57   when: ar_debug|bool
58   debug:
59     var: result
60 when:
61   - not ar_pip_install
62   - ansible_os_family == "RedHat" or ansible_os_family == "Debian"
63
64 # pip -----
65 - name: "packages: Test {{ ar_pip_executable }} exists"
66   when: ar_pip_install
67   block:
68     - name: "packages: Stat {{ ar_pip_executable }}"
69       stat:
70         path: "{{ ar_pip_executable }}"
71       register: result
72     - name: "packages: Not exists {{ ar_pip_executable }}"
73       fail:
74         msg: "[ERROR] {{ ar_pip_executable }} does not exist."
75       when: not result.stat.exists
76
77 - name: "packages: Install Ansible Runner pip packages for {{ ar_owner }}"
78   when: ar_pip_install
79   become_user: "{{ ar_owner }}"
80   become: true
81   changed_when: false # Note 1.
82   pip:
83     name: "{{ item.name }}"
84     executable: "{{ ar_pip_executable }}"
85     version: "{{ item.version|default(omit) }}"
86     state: "{{ item.state|default(omit) }}"
87     extra_args: "{{ pip_extraargs|default(omit) }}"
88   loop: "{{ ar_packages }}"
89   register: result
90   until: result is succeeded
91   retries: "{{ pip_install_retries }}"
92   delay: "{{ pip_install_delay }}"
93
94 - name: "packages: Debug pip packages"
95   when:
96     - ar_pip_install
97     - ar_debug|bool
98   debug:
99     var: result
100
101 - name: "packages: Install Ansible Runner pip requirements for {{ ar_owner }}"
102   when:
103     - ar_pip_install
104     - ar_pip_requirements is defined
105   become_user: "{{ ar_owner }}"
106   become: true
107   changed_when: false # Note 1.
108   pip:

```

(continues on next page)

(continued from previous page)

```

109     requirements: "{{{ ar_pip_requirements }}}}"
110     executable: "{{{ ar_pip_executable }}}}"
111     extra_args: "{{{ pip_extraargs|default(omit) }}}}"
112     register: result
113     until: result is succeeded
114     retries: "{{{ pip_install_retries }}}}"
115     delay: "{{{ pip_install_delay }}}}"
116
117 - name: "packages: Debug pip requirements"
118   when:
119     - ar_pip_install
120     - ar_debug|bool
121   debug:
122     var: result
123
124 # Note 1.
125 # The pip module isn't always idempotent #28952
126 # https://github.com/ansible/ansible/issues/28952
127
128 # EOF
129 ...

```

2.1.6 vars.yml

Synopsis: Configure vars.

Description of the task.

[vars.yml]

```

1 ---
2
3 - name: "Declare ar_owner when undefined"
4   when: ar_owner is undefined
5   set_fact:
6     ar_owner: "{{{ ansible_user_id }}}}"
7
8 - name: "Default vars for {{{ ansible_os_family }}}
9           {{{ ansible_distribution }}}
10          {{{ ansible_distribution_release }}}}"
11   include_vars: "{{{ item }}}}"
12   with_first_found:
13     - files:
14         - "{{{ ansible_distribution }}}-{{{ ansible_distribution_release }}}.yaml"
15         - "{{{ ansible_distribution }}}.yaml"
16         - "{{{ ansible_os_family }}}.yaml"
17         - "defaults.yaml"
18         - "default.yaml"
19     paths: "{{{ role_path }}}/vars/defaults"
20
21 - name: "Custom vars for {{{ ansible_os_family }}}
22           {{{ ansible_distribution }}}
23           {{{ ansible_distribution_release }}}}"
24   include_vars: "{{{ item }}}}"
25   with_first_found:
26     - files:

```

(continues on next page)

(continued from previous page)

```
27     - "{{ ansible_distribution }}-{{ ansible_distribution_release }}.yaml"
28     - "{{ ansible_distribution }}.yaml"
29     - "{{ ansible_os_family }}.yaml"
30     - "defaults.yaml"
31     - "default.yaml"
32     paths: "{{ role_path }}/vars"
33
34     # EOF
35     . . .
```


COPYRIGHT

Redistribution and use in source (RST DocUtils) and ‘compiled’ forms (XML, HTML, PDF, PostScript, RTF and so forth) with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code (RST Docutils) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in compiled form (transformed to other DTDs, converted to PDF, PostScript, RTF and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Important: THIS DOCUMENTATION IS PROVIDED “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE PROVIDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

LEGAL NOTICE

All product names, logos, and brands are property of their respective owners.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`