Under the hood:

How ARA Records Ansible playbooks &

makes them easier to understand and troubleshoot

David Moreau-Simard (fosstodon.org/@rfc2549)

Config Management Camp 2023

ara

$ whoami

```yaml
---
- name: David Moreau-Simard
  hosts:
    - dmsimard:matrix.org
    - fosstodon.org/@rfc2549
  vars:
    location: Montreal, Canada (or rabbit holes)
    profile: sysadmin, dev/ops, CI/CD, SRE
  roles:
    - Ansible user since version 1.8 or so (2014?)
    - Created ara back in 2016 to make my life easier
    - Previously in the Ansible community team @ Red Hat
    - Part time open source contributor (between dev ops and dad ops)
```

ara

Ansible

README.rst

pypi v2.14.1 | docs latest | chat IRC | Azure Pipelines failed | code of conduct Ansible | mailing lists Ansible | license GPL v3.0
openssf best practices in progress 66%

Ansible

Ansible is a radically simple IT automation system. It handles configuration management, application deployment, cloud provisioning, ad-hoc task execution, network automation, and multi-node orchestration. Ansible makes complex changes like zero-downtime rolling updates with load balancers easy. More information on the Ansible website.

Design Principles

- Have an extremely simple setup process with a minimal learning curve.
- Manage machines quickly and in parallel.
- Avoid custom-agents and additional open ports, be agentless by leveraging the existing SSH daemon.
- Describe infrastructure in a language that is both machine and human friendly.
- Focus on security and easy auditability/review/rewriting of content.
- Manage new remote machines instantly, without bootstrapping any software.
- Allow module development in any dynamic language, not just Python.
- Be usable as non-root.
- Be the easiest IT automation system to use, ever.

https://github.com/ansible/ansible/blob/devel/README.rst

ara

Configuration management with Ansible

```yaml
- name: Setup nginx
  hosts: web01
  become: true
  handlers:
    - name: Restart nginx
      service:
        name: nginx
        state: restarted
  tasks:
    - name: Install nginx
      package:
        name: nginx
        state: present
    - name: Configure virtual host
      template:
        src: vhost.conf.j2
        dest: /etc/nginx/conf.d/service.conf
      notify:
        - Restart nginx
    - name: Ensure nginx is started
      service:
        name: nginx
        state: started
        enabled: true
```

ara

Running an Ansible playbook

```
$ ansible-playbook -i hosts playbook.yml
PLAY [Setup nginx] ***********************************************************************************************

TASK [Gathering Facts] *******************************************************************************************
ok: [web01]

TASK [Install nginx] *********************************************************************************************
changed: [web01]

TASK [Configure virtual host] ************************************************************************************
changed: [web01]

TASK [Ensure nginx is started] ***********************************************************************************
changed: [web01]

RUNNING HANDLER [Restart nginx] **********************************************************************************
changed: [web01]

PLAY RECAP *******************************************************************************************************
web01                      : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Running an Ansible playbook (again)

```
$ ansible-playbook -i hosts playbook.yml
PLAY [Setup nginx] ***********************************************************************************************

TASK [Gathering Facts] *******************************************************************************************
ok: [web01]

TASK [Install nginx] *********************************************************************************************
ok: [web01]

TASK [Configure virtual host] ************************************************************************************
ok: [web01]

TASK [Ensure nginx is started] ***********************************************************************************
ok: [web01]

PLAY RECAP *******************************************************************************************************
web01                      : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Running an Ansible playbook with verbosity

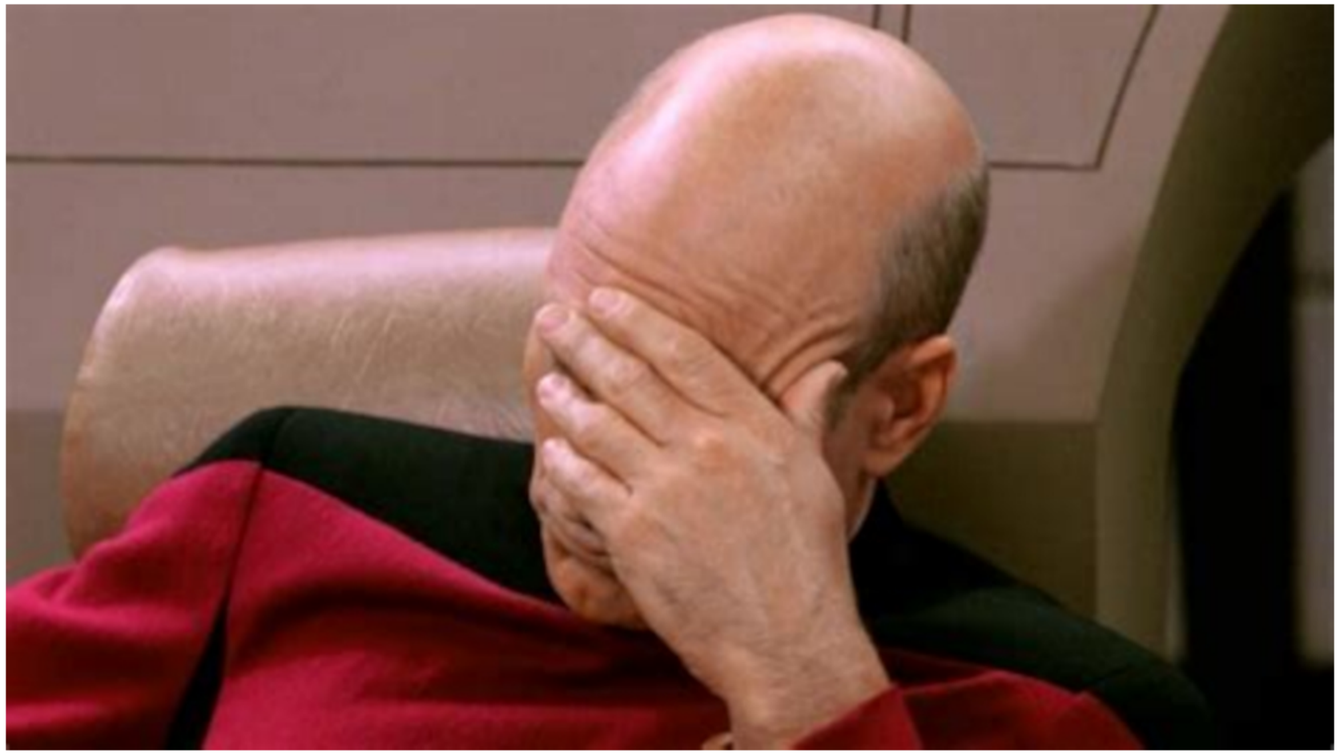
```
$ ansible-playbook -vv -i hosts playbook.yml --diff
PLAY [Setup nginx] *******************************************************************************************

TASK [Gathering Facts] ***************************************************************************************
task path: /root/ansible/playbook.yml:1
ok: [web01]

TASK [Install nginx] *****************************************************************************************
task path: /root/ansible/playbook.yml:10
changed: [web01] => {"changed": true, "msg": "", "rc": 0, "results": ["Installed: nginx-filesystem-1:1.22.1-2.el9.noarch", "Installed: nginx-core-1:1.22.1-2.el9.x86_64",

TASK [Configure virtual host] ********************************************************************************
task path: /root/ansible/playbook.yml:14
NOTIFIED HANDLER Restart nginx for web01
--- before
+++ after: /root/.ansible/tmp/ansible-local-16494np9ywx2w/tmprulqs3af/vhost.conf.j2
@@ -0,0 +1,5 @@
+server {
+    listen 80;
+    server_name web01.domain.tld;
+    return 301 https://ara.recordsansible.org$request_uri;
+}

changed: [web01] => {"changed": true, "checksum": "0d3a4953ac474f9849fabb52f606b7b0bf4076ca", "dest": "/etc/nginx/conf.d/service.conf", "gid": 0, "group": "root", "md5su

TASK [Ensure nginx is started] *******************************************************************************
```

What verbosity actually looks like

```
TASK [Ensure nginx is started] ***************************************************************************
task path: /root/ansible/playbook.yml:20
changed: [web01] => {"changed": true, "enabled": true, "name": "nginx", "state": "started", "status": {"ActiveEnterTimestampMonotonic": "0", "ActiveExitTimesta
mpMonotonic": "0", "ActiveState": "inactive", "After": "basic.target -.mount sysinit.target tmp.mount remote-fs.target nss-lookup.target network-online.target
systemd-journald.socket system.slice systemd-tmpfiles-setup.service", "AllowIsolate": "no", "AssertResult": "no", "AssertTimestampMonotonic": "0", "Before": "s
hutdown.target", "BlockIOAccounting": "no", "BlockIOWeight": "[not set]", "CPUAccounting": "yes", "CPUAffinityFromNUMA": "no", "CPUQuotaPerSecUSec": "infinity"
, "CPUQuotaPeriodUSec": "infinity", "CPUSchedulingPolicy": "0", "CPUSchedulingPriority": "0", "CPUSchedulingResetOnFork": "no", "CPUShares": "[not set]", "CPUU
sageNSec": "[not set]", "CPUWeight": "[not set]", "CacheDirectoryMode": "0755", "CanFreeze": "yes", "CanIsolate": "no", "CanReload": "yes", "CanStart": "yes",
"CanStop": "yes", "CapabilityBoundingSet": "cap_chown cap_dac_override cap_dac_read_search cap_fowner cap_fsetid cap_kill cap_setgid cap_setuid cap_setpcap cap
_linux_immutable cap_net_bind_service cap_net_broadcast cap_net_admin cap_net_raw cap_ipc_lock cap_ipc_owner cap_sys_module cap_sys_rawio cap_sys_chroot cap_sy
s_ptrace cap_sys_pacct cap_sys_admin cap_sys_boot cap_sys_nice cap_sys_resource cap_sys_time cap_sys_tty_config cap_mknod cap_lease cap_audit_write cap_audit_c
ontrol cap_setfcap cap_mac_override cap_mac_admin cap_syslog cap_wake_alarm cap_block_suspend cap_audit_read cap_perfmon cap_bpf cap_checkpoint_restore", "Clea
nResult": "success", "CollectMode": "inactive", "ConditionResult": "no", "ConditionTimestampMonotonic": "0", "ConfigurationDirectoryMode": "0755", "Conflicts":
 "shutdown.target", "ControlPID": "0", "CoredumpFilter": "0x33", "DefaultDependencies": "yes", "DefaultMemoryLow": "0", "DefaultMemoryMin": "0", "Delegate": "n
o", "Description": "The nginx HTTP and reverse proxy server", "DevicePolicy": "auto", "DynamicUser": "no", "ExecMainCode": "0", "ExecMainExitTimestampMonotonic
": "0", "ExecMainPID": "0", "ExecMainStartTimestampMonotonic": "0", "ExecMainStatus": "0", "ExecReload": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/nginx -s re
load ; ignore_errors=no ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExecReloadEx": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/
nginx -s reload ; flags= ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExecStart": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/ng
inx ; ignore_errors=no ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExecStartEx": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/ng
inx ; flags= ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExecStartPre": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/nginx -t ;
ignore_errors=no ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExecStartPreEx": "{ path=/usr/sbin/nginx ; argv[]=/usr/sbin/nginx
 -t ; flags= ; start_time=[n/a] ; stop_time=[n/a] ; pid=0 ; code=(null) ; status=0/0 }", "ExitType": "main", "FailureAction": "none", "FileDescriptorStoreMax":
 "0", "FinalKillSignal": "9", "FragmentPath": "/usr/lib/systemd/system/nginx.service", "FreezerState": "running", "GID": "[not set]", "GuessMainPID": "yes", "I
OAccounting": "no", "IOReadBytes": "18446744073709551615", "IOReadOperations": "18446744073709551615", "IOSchedulingClass": "2", "IOSchedulingPriority": "4", "
IOWeight": "[not set]", "IOWriteBytes": "18446744073709551615", "IOWriteOperations": "18446744073709551615", "IPAccounting": "no", "IPEgressBytes": "[no data]"
, "IPEgressPackets": "[no data]", "IPIngressBytes": "[no data]", "IPIngressPackets": "[no data]", "Id": "nginx.service", "IgnoreOnIsolate": "no", "IgnoreSIGPIP
E": "yes", "InactiveEnterTimestampMonotonic": "0", "InactiveExitTimestampMonotonic": "0", "JobRunningTimeoutUSec": "infinity", "JobTimeoutAction": "none", "Job
TimeoutUSec": "infinity", "KeyringMode": "private", "KillMode": "mixed", "KillSignal": "3", "LimitAS": "infinity", "LimitASSoft": "infinity", "LimitCORE": "inf
inity", "LimitCORESoft": "0", "LimitCPU": "infinity", "LimitCPUSoft": "infinity", "LimitDATA": "infinity", "LimitDATASoft": "infinity", "LimitFSIZE": "infinity
", "LimitFSIZESoft": "infinity", "LimitLOCKS": "infinity", "LimitLOCKSSoft": "infinity", "LimitMEMLOCK": "65536", "LimitMEMLOCKSoft": "65536", "LimitMSGQUEUE":
 "819200", "LimitMSGQUEUESoft": "819200", "LimitNICE": "0", "LimitNICESoft": "0", "LimitNOFILE": "524288", "LimitNOFILESoft": "1024", "LimitNPROC": "14626", "L
imitNPROCSoft": "14626", "LimitRSS": "infinity", "LimitRSSSoft": "infinity", "LimitRTPRIO": "0", "LimitRTPRIOSoft": "0", "LimitRTTIME": "infinity", "LimitRTTIM
ESoft": "infinity", "LimitSIGPENDING": "14626", "LimitSIGPENDINGSoft": "14626", "LimitSTACK": "infinity", "LimitSTACKSoft": "8388608", "LoadState": "loaded", "
LockPersonality": "no", "LogLevelMax": "-1", "LogRateLimitBurst": "0", "LogRateLimitIntervalUSec": "0", "LogsDirectoryMode": "0755", "MainPID": "0", "ManagedOO
MMemoryPressure": "auto", "ManagedOOMMemoryPressureLimit": "0", "ManagedOOMPreference": "none", "ManagedOOMSwap": "auto", "MemoryAccounting": "yes", "MemoryAva
ilable": "infinity", "MemoryCurrent": "[not set]", "MemoryDenyWriteExecute": "no", "MemoryHigh": "infinity", "MemoryLimit": "infinity", "MemoryLow": "0", "Memo
```

ara

What verbosity actually looks like



ara

This output comes from the default Ansible stdout callback

ansible / ansible Public

Notifications | Fork 22.9k | Star 56.1k

Code | Issues 667 | Pull requests 345 | Actions | Projects 14 | Security | Insights

devel | ansible / lib / ansible / plugins / callback / default.py / Jump to | Go to file

mkrizek Properly send a skipped message for empty templ... Latest commit 058a69e on Jun 2, 2022 History

24 contributors +12

409 lines (325 sloc) | 16.5 KB | Raw | Blame

```
# (c) 2012-2014, Michael DeHaan <michael.dehaan@gmail.com>
# (c) 2017 Ansible Project
# GNU General Public License v3.0+ (see COPYING or https://www.gnu.org/licenses/gpl-3.0.txt)

from __future__ import (absolute_import, division, print_function)
__metaclass__ = type

DOCUMENTATION = '''
    name: default
    type: stdout
    short_description: default Ansible screen output
    version_added: historical
    description:
        - This is the default output callback for ansible-playbook.
    extends_documentation_fragment:
```

https://github.com/ansible/ansible/blob/devel/lib/ansible/plugins/callback/default.py

ara

There's different kinds of callbacks

```python
class CallbackModule(CallbackBase):
    """
    Saves data from an Ansible run into a database
    """

    CALLBACK_VERSION = 2.0
    CALLBACK_TYPE = "awesome"
    CALLBACK_NAME = "ara_default"

    def __init__(self):
        super().__init__()
```

https://github.com/ansible-community/ara/blob/master/ara/plugins/callback/ara_default.py

See examples: https://rndmh3ro.github.io/

ara

Ansible callbacks provide hooks to react on certain events

```
> grep v2_ lib/ansible/plugins/callback/__init__.py
def v2_on_any(self, *args, **kwargs):
def v2_runner_on_failed(self, result, ignore_errors=False):
def v2_runner_on_ok(self, result):
def v2_runner_on_skipped(self, result):
def v2_runner_on_unreachable(self, result):
def v2_runner_on_async_poll(self, result):
def v2_runner_on_async_ok(self, result):
def v2_runner_on_async_failed(self, result):
def v2_playbook_on_start(self, playbook):
def v2_playbook_on_notify(self, handler, host):
def v2_playbook_on_no_hosts_matched(self):
def v2_playbook_on_no_hosts_remaining(self):
def v2_playbook_on_task_start(self, task, is_conditional):
def v2_playbook_on_cleanup_task_start(self, task):
def v2_playbook_on_handler_task_start(self, task):
def v2_playbook_on_vars_prompt(self, varname, private=True, prompt=None, encrypt=None, confirm=False, salt_size=None, salt=None, default=None, unsafe=None):
def v2_playbook_on_import_for_host(self, result, imported_file):
def v2_playbook_on_not_import_for_host(self, result, missing_file):
def v2_playbook_on_play_start(self, play):
def v2_playbook_on_stats(self, stats):
def v2_on_file_diff(self, result):
def v2_playbook_on_include(self, included_file):
def v2_runner_item_on_ok(self, result):
def v2_runner_item_on_failed(self, result):
def v2_runner_item_on_skipped(self, result):
```

ara

ara uses these hooks to record results

At first there was no API 😱

```
    def get_or_create_host(self, hostname):
        try:
            host = (models.Host.query
                    .filter_by(name=hostname)
                    .filter_by(playbook_id=self.playbook.id)
                    .one())
        except models.NoResultFound:
            host = models.Host(name=hostname, playbook=self.playbook)
            db.session.add(host)
            db.session.commit()

        return host
```

https://github.com/ansible-community/ara/blob/stable/0.x/ara/plugins/callbacks/log_ara.py

ara

It worked... but:

- Built on the foundation of a weekend proof of concept
- Not very flexible, requires the callback to have access to the database
- Need knowledge of the database model to integrate in tools and scripts
- Duplicated code across callback, CLI and web interface
- Tried to add an API on top of the existing Flask backend
- Ended up re-writing everything to django & django-rest-framework

ara

Now there's an API 🎉

Django REST framework

Index / Api Index / Api Root

Api Root

GET

The default basic root view for DefaultRouter

```
GET /api/v1/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "labels": "https://demo.recordsansible.org/api/v1/labels",
    "playbooks": "https://demo.recordsansible.org/api/v1/playbooks",
    "plays": "https://demo.recordsansible.org/api/v1/plays",
    "tasks": "https://demo.recordsansible.org/api/v1/tasks",
    "hosts": "https://demo.recordsansible.org/api/v1/hosts",
    "latesthosts": "https://demo.recordsansible.org/api/v1/latesthosts",
    "results": "https://demo.recordsansible.org/api/v1/results",
    "files": "https://demo.recordsansible.org/api/v1/files",
    "records": "https://demo.recordsansible.org/api/v1/records"
}
```

https://demo.recordsansible.org/api/v1/

ara

So the callback can use it

```
680     def _get_or_create_host(self, host):
681         """Note: The get_or_create is handled through the serializer of the API server."""
682         # We might want to record results against the fqdn instead of localhost
683         # so that we can differentiate between different actual hosts
684         if self.localhost_as_hostname and host in ["localhost", "127.0.0.1"]:
685             host = self.localhost_hostname
686
687         # Ansible inventory hostnames can be longer than 255 characters
688         # Ansible doesn't mind and this is OK in AWX with postgresql but is an issue with mysql in ara
689         # https://github.com/ansible-community/ara/issues/265
690         if len(host) >= 255:
691             # Only warn about this once so we don't print a warning on every task
692             if host not in self.warned_about_host_length:
693                 self.log.warn("Truncating hostname before recording: it's longer than 255 characters (%s)" % ho
694                 self.warned_about_host_length.append(host)
695             host = host[:254]
696
697         if host not in self.host_cache:
698             self.log.debug("Host not in cache, getting or creating: %s" % host)
699             self.host_cache[host] = self.client.post("/api/v1/hosts", name=host, playbook=self.playbook["id"])
700         return self.host_cache[host]
```

https://github.com/ansible-community/ara/blob/master/ara/plugins/callback/ara_default.py

ara

The CLI too

```
117         client = get_client(
118             client=args.client,
119             endpoint=args.server,
120             timeout=args.timeout,
121             username=args.username,
122             password=args.password,
123             cert=args.ssl_cert,
124             key=args.ssl_key,
125             verify=verify,
126             run_sql_migrations=False,
127         )
128         query = {}
129         if args.label is not None:
130             query["label"] = args.label
131
132         if args.ansible_version is not None:
133             query["ansible_version"] = args.ansible_version
```

https://github.com/ansible-community/ara/blob/master/ara/cli/playbook.py

ara

The CLI too

```
147         if args.name is not None:
148             query["name"] = args.name
149
150         if args.path is not None:
151             query["path"] = args.path
152
153         if args.status is not None:
154             query["status"] = args.status
155
156         if args.user is not None:
157             query["user"] = args.user
158
159         query["order"] = args.order
160         query["limit"] = args.limit
161
162         playbooks = client.get("/api/v1/playbooks", **query)
```

https://github.com/ansible-community/ara/blob/master/ara/cli/playbook.py

ara

... and anything else ¯\(ツ)/¯

Ansible action and lookup plugins

```yaml
- name: Test playbook
  hosts: localhost
  tasks:
    - name: Get the currently running playbook
      ara_playbook:
      register: query

    - name: Retrieve playbook id
      set_fact:
        playbook_id: "{{ query.playbook.id | string }}"
    # With the playbook id we can create a link to the playbook report
    - name: Recover base url from ara
      set_fact:
        api_base_url: "{{ lookup('ara_api', '/api/') }}"

    - name: Print link to playbook report
      vars:
        ui_base_url: "{{ api_base_url.api[0] | regex_replace('/api/v1/', '') }}"
      debug:
        msg: "{{ ui_base_url }}/playbooks/{{ playbook_id }}.html"
```

ara

... and anything else ¯\(ツ)/¯

```yaml
- name: Assert playbook properties
  hosts: localhost
  gather_facts: yes
  vars:
    ara_playbook_name: ARA self tests
    ara_playbook_labels:
      - lookup-tests
  tasks:
    - name: Retrieve the current playbook so we can get the ID
      ara_playbook:
      register: playbook_query

    - name: Save the playbook id so we can re-use it easily
      set_fact:
        playbook_id: "{{ playbook_query.playbook.id | string }}"

    - name: Recover data from ARA
      set_fact:
        playbook: "{{ lookup('ara_api', '/api/v1/playbooks/' + playbook_id) }}"
        tasks: "{{ lookup('ara_api', '/api/v1/tasks?playbook=' + playbook_id) }}"
        results: "{{ lookup('ara_api', '/api/v1/results?playbook=' + playbook_id) }}"
        hosts: "{{ lookup('ara_api', '/api/v1/hosts?playbook=' + playbook_id) }}"

    - name: Print versions we're testing with
      debug:
        msg: >-
          Integration test playbook running under ansible-core {{ ansible_version.full }},
          ara {{ playbook.client_version }} (client) and {{ playbook.server_version }} (server)

    - name: Assert playbook properties
      assert:
        that:
          - playbook.name == 'ARA self tests'
          - "playbook.labels | selectattr('name', 'search', 'lookup-tests') | list | length == 1"
          - playbook.ansible_version == ansible_version.full
          - playbook_dir in playbook.path
          - "'tests/integration/lookups.yaml' in playbook.path"
```

https://github.com/ansible-community/ara/blob/master/tests/integration/lookups.yaml

ara

... and anything else ¯\(ツ)/¯

```yaml
- name: "Get the time {{ minutes }} minutes ago"
  command: date +{{ format }} -d "{{ minutes }} minutes ago"
  changed_when: false
  register: timestamp

- name: Find out if there's been playbooks run since that timestamp
  set_fact:
    playbooks: "{{ lookup('ara_api', '/api/v1/playbooks?started_after=' + timestamp.stdout) }}"

- name: Send Slack notifications
  vars:
    # The lookup plugin provides the url to the API endpoint
    _api_base_url: "{{ lookup('ara_api', '/api/') }}"
    # From which we can deduce the url to the reporting interface
    _ui_base_url: "{{ _api_base_url.api[0] | regex_replace('/api/v1/', '') }}"
    # From which we can build the playbook report URL
    _playbook_url: "{{ _ui_base_url }}/playbooks/{{ item.id }}.html"
    _results: "{{ item['items']['results'] }}"
    _hosts: "{{ item['items']['hosts'] }}"
    _duration: "{{ 'in ' + item.duration if item.duration }}"
  slack:
    token: "{{ token }}"
    channel: "{{ channel }}"
    msg: |
      [{{ item.status }}] {{ _playbook_url }} | {{ item.path }}: {{ _results }} results on {{ _hosts }}
  loop: "{{ playbooks.results | selectattr('name', 'equalto', ara_playbook_name) | list }}"
```

https://github.com/ansible-community/ara/blob/master/contrib/playbooks/slack-notifications.yaml

ara

Thanks to built-in API clients

```python
#!/usr/bin/env python3
# Import the client
from ara.clients.http import AraHttpClient

# Instanciate the HTTP client with an endpoint where an API server is listening
client = AraHttpClient(endpoint="https://demo.recordsansible.org")

# Get a list of failed playbooks
# /api/v1/playbooks?status=failed
playbooks = client.get("/api/v1/playbooks", status="failed")
```

Even "offline" (by default!)

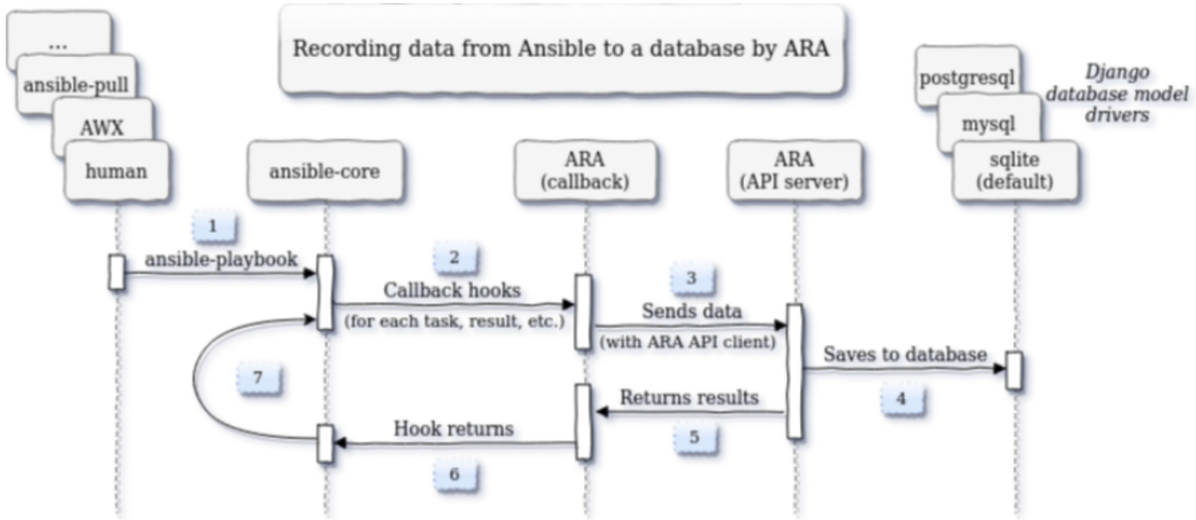
```python
#!/usr/bin/env python3
# Import the client
from ara.clients.offline import AraOfflineClient

# Instanciate the HTTP client with an endpoint where an API server is listening
client = AraOfflineClient()

# Get a list of failed playbooks
# /api/v1/playbooks?status=failed
playbooks = client.get("/api/v1/playbooks", status="failed")
```

Kind of offline

An ephemeral server made available in the background

```
> netstat -ntlp
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
# ...
tcp        0      0 127.0.0.1:34293         0.0.0.0:*               LISTEN      29659/python # <---
```

What it looks like



ara

README.md

ARA Records Ansible

ARA Records Ansible and makes it easier to understand and troubleshoot.

It's another recursive acronym and features simplicity as a core principle.

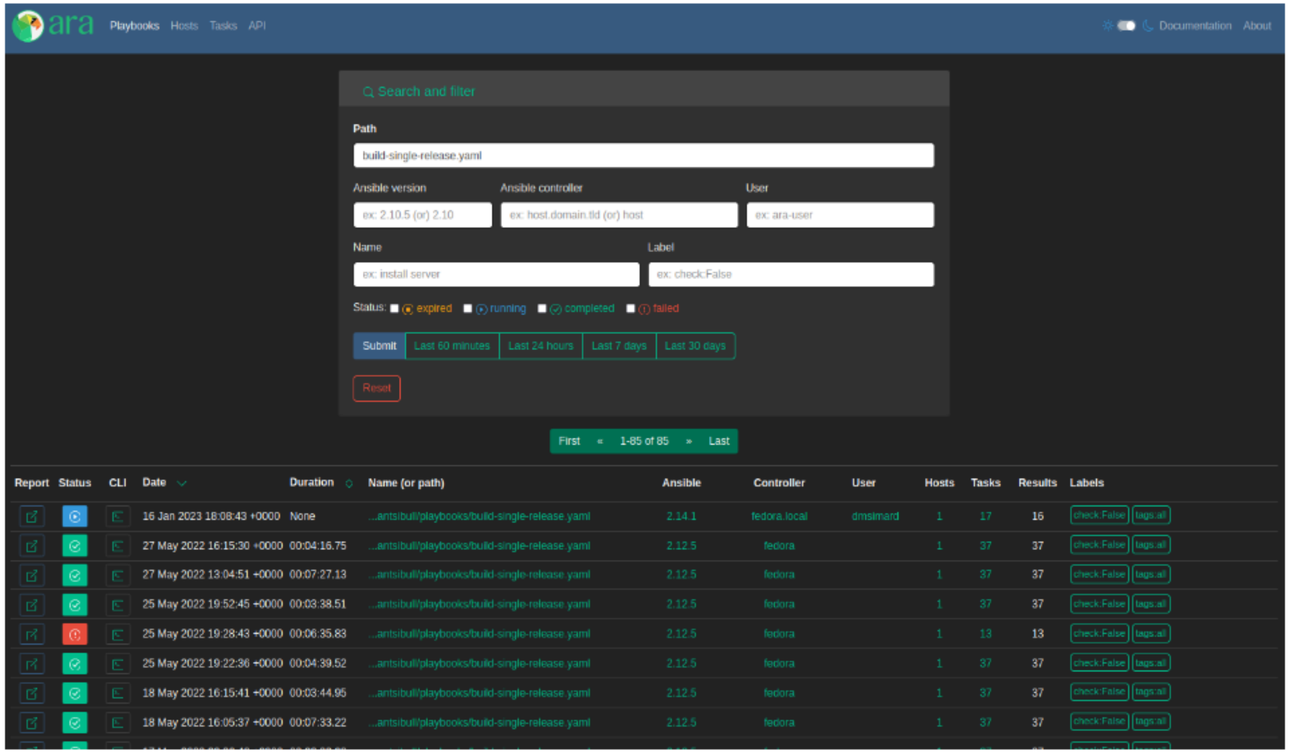
About ara

ara provides Ansible reporting by recording `ansible` and `ansible-playbook` commands wherever and however they are run:

- from a terminal, by hand or from a script
- from a laptop, desktop, server, VM or container
- for development, CI or production
- from most Linux distributions and even on Mac OS (as long as `python >= 3.6` is available)
- from tools that run playbooks such as AWX & Automation Controller (Tower), ansible-(pull|test|runner|navigator) and Molecule
- from CI/CD platforms such as Jenkins, GitHub Actions, GitLab CI, Rundeck and Zuul

https://github.com/ansible-community/ara

It records playbooks

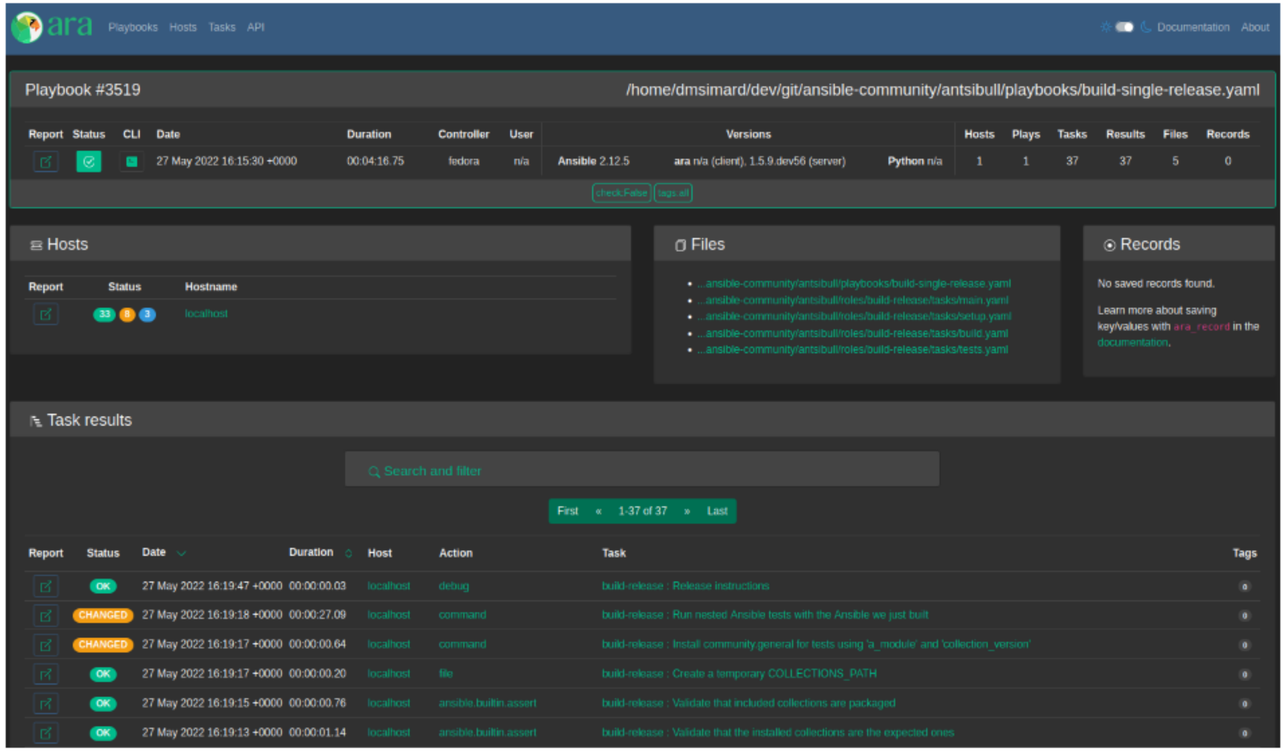


https://demo.recordsansible.org/?path=build-single-release.yaml

ara

It records hosts, tasks, results and files

(and a bunch of other things, too)



https://demo.recordsansible.org/playbooks/3519.html

ara

Getting started

```
# Install ansible (or ansible-core) with ara (including API server dependencies)
python3 -m pip install --user ansible "ara[server]"

# Configure Ansible to use the ara callback plugin
export ANSIBLE_CALLBACK_PLUGINS="$(python3 -m ara.setup.callback_plugins)"

# Run an Ansible playbook as usual
ansible-playbook hello-world.yml

# Use the CLI to see recorded playbooks
ara playbook list

# Start the development server at http://127.0.0.1:8000
# to query the API or browse recorded results
ara-manage runserver
```

ara

Getting started (with a server)

```
# Create a directory for a volume to store settings and a sqlite database
mkdir -p ~/.ara/server

# Start an API server with docker from the image on DockerHub:
docker run --name api-server --detach --tty \
  --volume ~/.ara/server:/opt/ara -p 8000:8000 \
  docker.io/recordsansible/ara-api:latest

# or with podman from the image on quay.io:
podman run --name api-server --detach --tty \
  --volume ~/.ara/server:/opt/ara -p 8000:8000 \
  quay.io/recordsansible/ara-api:latest
```

ara

Getting started (with a server)

```
# Install ansible (or ansible-core) with ara (excluding API server dependencies)
python3 -m pip install --user ansible ara

# Configure Ansible to use the ara callback plugin
export ANSIBLE_CALLBACK_PLUGINS="$(python3 -m ara.setup.callback_plugins)"

# Set up the ara callback to know where the API server is located
export ARA_API_CLIENT="http"
export ARA_API_SERVER="http://127.0.0.1:8000"

# Run an Ansible playbook as usual
ansible-playbook playbook.yaml

# Use the CLI to see recorded playbooks
ara playbook list

# Browse http://127.0.0.1:8000 (running from the container)
# to view the reporting interface
```

ara

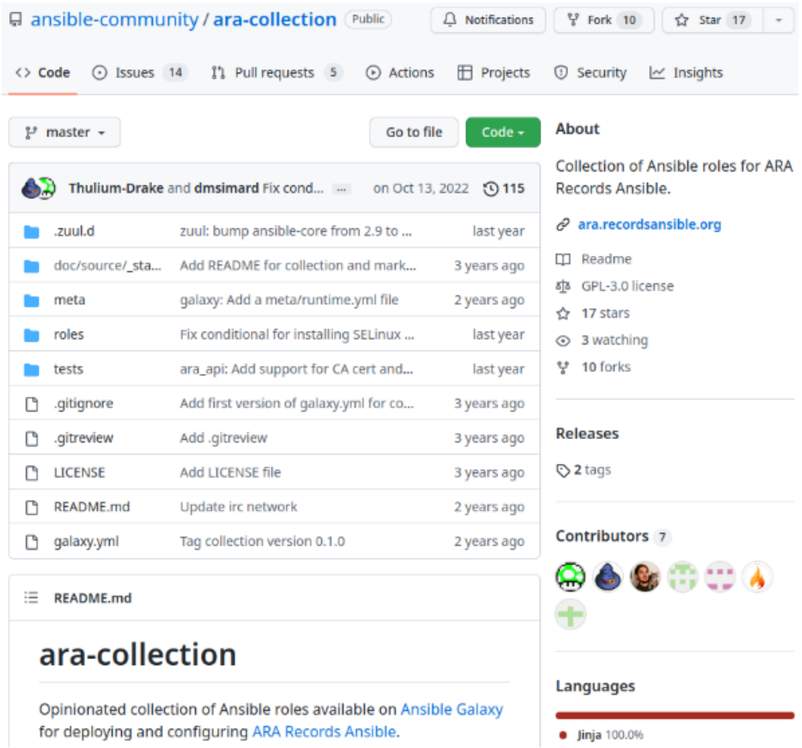
What about the developer side ?

Demo !

```
tox -e ansible-integration --notest
source .tox/ansible-integration/bin/activate
export ANSIBLE_CALLBACK_PLUGINS=$(python3 -m ara.setup.callback_plugins)
export ANSIBLE_ACTION_PLUGINS=$(python3 -m ara.setup.action_plugins)
export ANSIBLE_LOOKUP_PLUGINS=$(python3 -m ara.setup.lookup_plugins)
ansible-playbook tests/integration/smoke.yaml
ara-manage runserver
```

ara

There's a collection, too



https://github.com/ansible-community/ara-collection

https://galaxy.ansible.com/recordsansible/ara

The project could use your help !

- https://github.com/ansible-community/ara
- https://github.com/ansible-community/ara-collection
- https://github.com/ansible-community/ara-infra

https://ara.readthedocs.io/en/latest/contributing.html

ara

Other interesting approaches

- https://github.com/okapia/ansible-json-monitor
- https://github.com/jseguillon/caradoc
- https://github.com/fboender/ansible-cmdb

ara

Want to know more ?

- 2023-02-08, 09:00–18:00, B.1.017: Ansible Contributor Summit
- FOSDEM 2022: Simple (but useful) Ansible reporting with ara

ara

Thank You!

Any questions?

Come chat or stay up to date

- https://ara.recordsansible.org/blog/
- https://ara.recordsansible.org/community/
  - #ara (libera.chat) bridged to #ara:libera.chat (Matrix)
  - ~~https://twitter.com/@RecordsAnsible~~
  - https://fosstodon.org/@ara

Reach out:

- David Moreau-Simard
- Matrix: dmsimard:matrix.org
- Mastodon: fosstodon.org/@rfc2549

ara