#ANSIBLEFEST2019

Managing Meaningful Inventories

Will Tome
Sr. Solutions Architect

Alan Rominger
Sr. Software Engineer

Red Hat | ANSIBLE

Share your automation story

1. How did you get started with Ansible?

2. How long have you been using it?

3. What's your favorite thing to do when you Ansible?

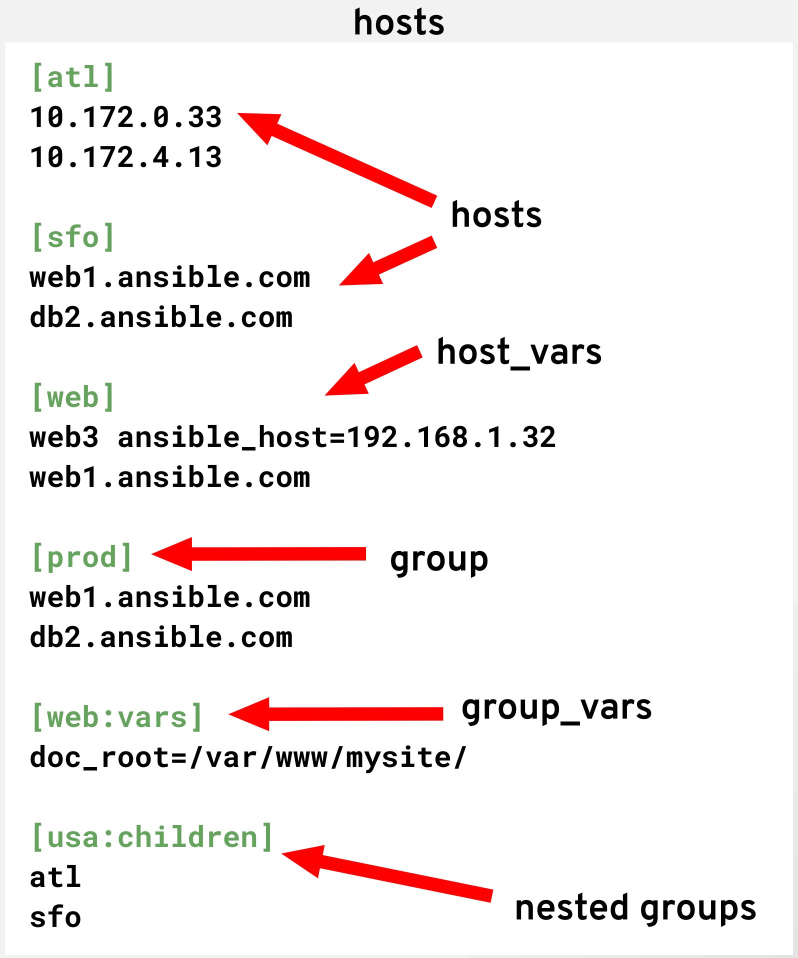
What is a meaningful inventory?

A meaningful inventory is...

- Up to date
- Aligned with process
- Logical
- Flexible
- Inclusive

Anatomy of an Inventory file

- Host name can be any of: IP address, Hostname, Alias
- Groups
    a. What - An application, stack or microservice.
    b. Where - A datacenter or region, to talk to local DNS, storage, etc.
    c. When - The dev stage, to avoid testing on production resources.
- Groups can be nested and define variables



```
[atl]
10.172.0.33
10.172.4.13

[sfo]
web1.ansible.com
db2.ansible.com

[web]
web3 ansible_host=192.168.1.32
web1.ansible.com

[prod]
web1.ansible.com
db2.ansible.com

[web:vars]
doc_root=/var/www/mysite/

[usa:children]
atl
sfo
```

https://github.com/willtome/managing-meaningful-inventories/

Working with Patterns

What hosts apply to a configuration or process?

```
---
- name: my really cool playbook
  hosts: all

  ...
```

```
---
- name: ONE OR MORE GROUPS
  hosts: sfo:atl

  ...
```

```
---
- name: EXCLUDE A GROUP
  hosts: all:!atl

  ...
```

```
---
- name: INTERSECTION
  hosts: atl:&prod

  ...
```

```
---
- name: VARIABLES
  hosts: “{{ HOSTS }}”

  ...
```

```
---
- name: WILDCARD
  hosts: *.com

  ...
```

```
---
- name: COMBINATIONS
  hosts: “atl:sfo:!{{excluded}}”

  ...
```

Using host_vars and group_vars Folders

- Architecture is “vars plugin”
  - host_group_vars
- Subdirectory in playbook or inventory folders
- Defines variables for the host or group
- Irrelevant of inventory type
- To set something for all hosts, use “all.yml” group vars

```
group_vars/
├── sfo.yml
├── atl.yml
└── prod/
      └── app.yml
      └── vault.yml
host_vars/
├── web3.yml
└── web1.ansible.com/
      └── network.yml
      └── vhosts.yml
```

Where does your inventory live?

servicenow
Infoblox
netbox
Red Hat
Azure
aws
Google Cloud Platform
vmware
MySQL
X
A
?

Why use Dynamic Inventory?

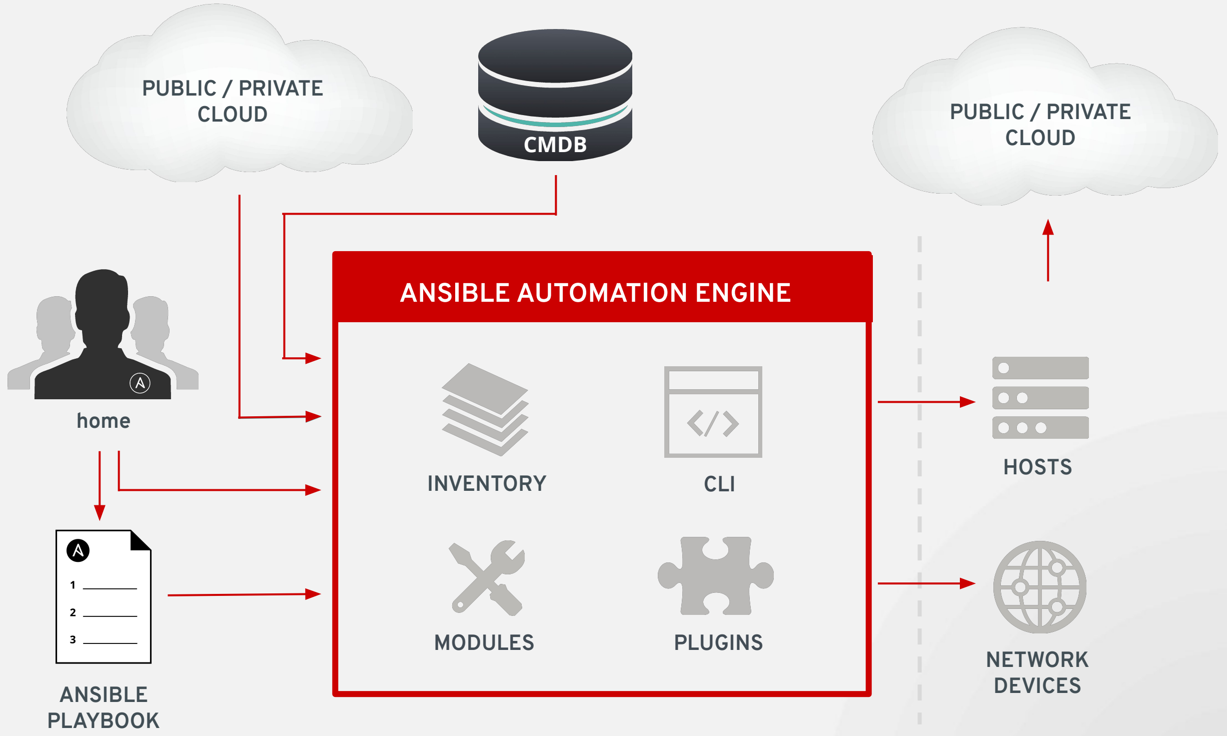
Don’t reinvent the wheel

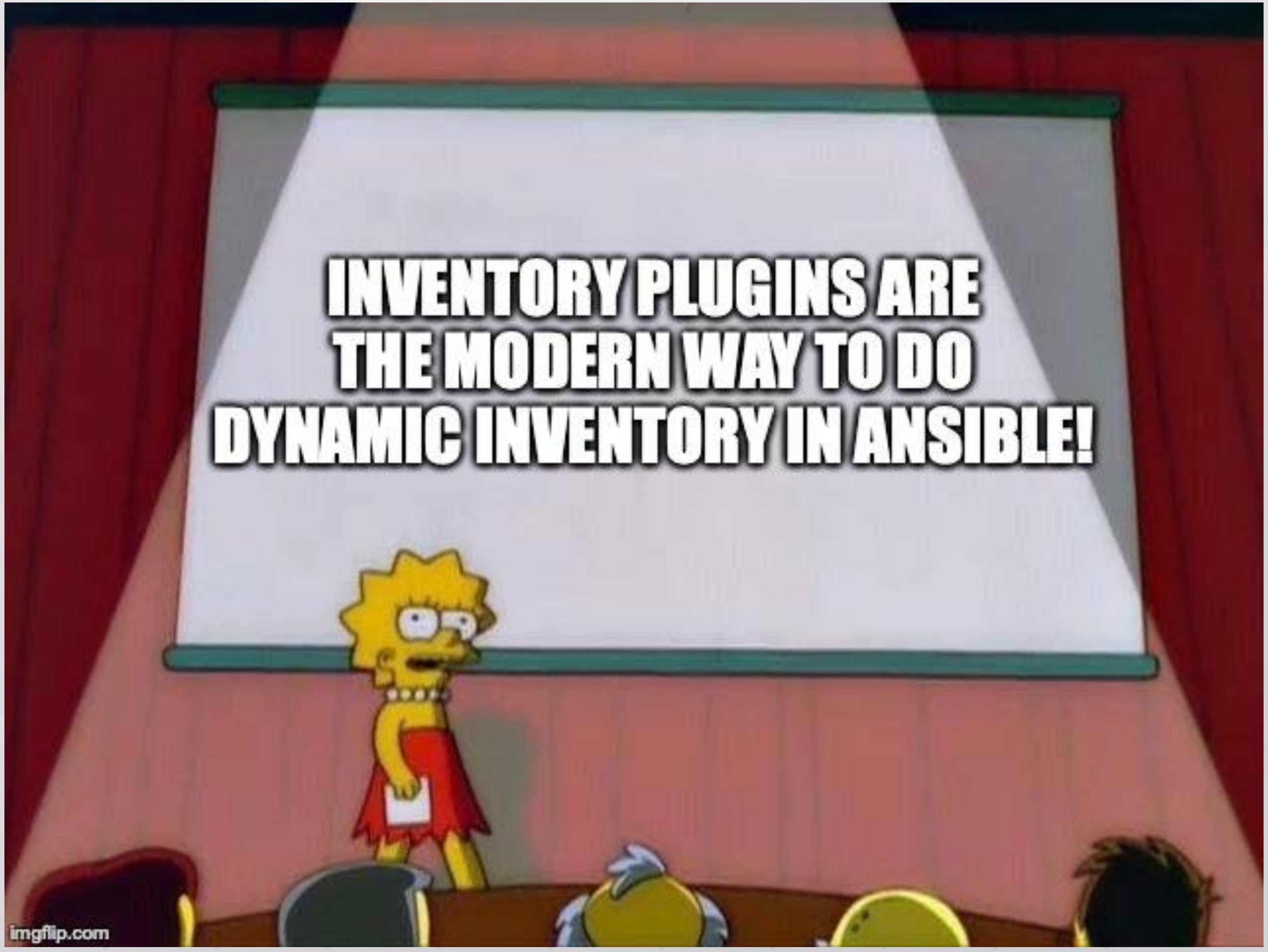
Keep up with change

Capture all systems

Integrate with everything

Extend Ansible Core




INVENTORY PLUGINS ARE THE MODERN WAY TO DO DYNAMIC INVENTORY IN ANSIBLE!
imgflip.com

Dynamic Inventory Example - azure_rm (Azure Resource Manager)

azure_rm.yml *or* **foo.azure_rm.yml**

```
plugin: azure_rm
auth_source: env
```

Specify credentials by environment variables

<u>Pre-requisites</u>

# install dependency
`pip install azure`

set_private_env.sh

```
#!/bin/sh
export AZURE_SUBSCRIPTION_ID=<private>
export AZURE_CLIENT_ID=<private>
export AZURE_SECRET=<private>
export AZURE_TENANT=<private>
```

https://github.com/willtome/managing-meaningful-inventories/

Debugging Inventory

# Check inventory plugin docs locally

```
ansible-doc -t inventory azure_rm
```

# See inventory JSON representation

```
ansible-inventory -i azure_rm.yml --list
```

# Run playbook

```
ansible-playbook -i azure_rm.yml debug.yml
```

# Multiple inventories

```
ansible-playbook -i <1> -i <2> debug.yml
ansible-playbook -i <dir> debug.yml
```

```
$ ansible-doc -t inventory -l
advanced_host_list  Parses a 'host list' with ranges
auto                Loads and executes an inventory plugin
aws_ec2             EC2 inventory source
aws_rds             rds instance source
azure_rm            Azure Resource Manager inventory plugin
cloudscale          cloudscale.ch inventory source
constructed         Uses Jinja2 to construct vars and groups
docker_machine      Docker Machine inventory source
...
```

Inventory plugins that Ansible “sees”
See `ansible.cfg` env
`ANSIBLE_INVENTORY_PLUGINS`

How ansible processes inventory?

Some plugins are enabled by default

Set in `ansible.cfg` or `ANSIBLE_INVENTORY_ENABLED` env variable

ansible.cfg

```
[inventory]
enable_plugins = host_list, script, auto, yaml, ini, toml
```

`Auto` uses **inventory file** to load **inventory plugins** that are not enabled

aws_ec2.yml

```
plugin: aws_ec2
regions:
  - us-east-1
filters:
  tag:Environment: dev
```

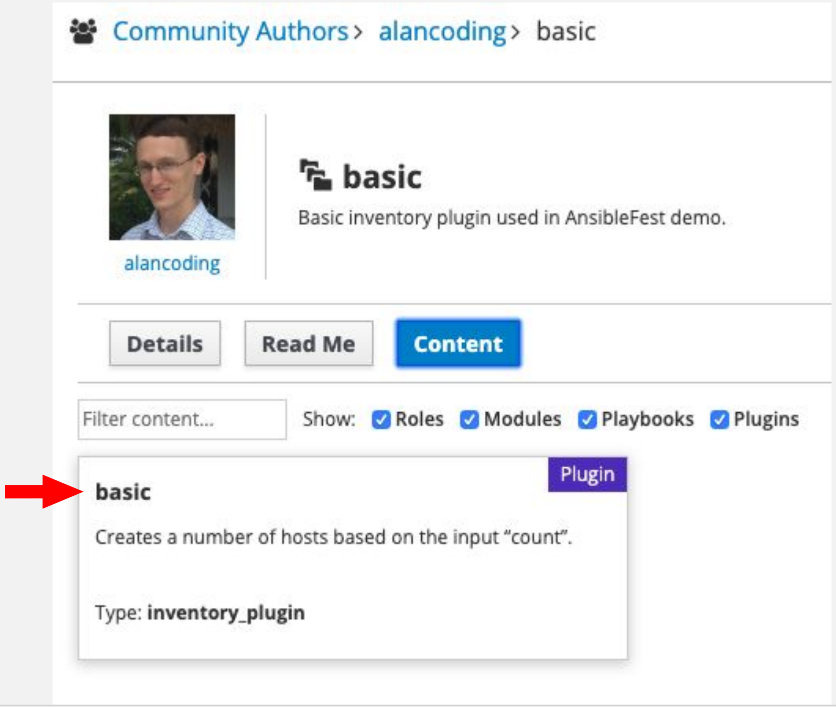
aws_ec2.py

```
class InventoryModule(BaseInventoryPlugin, Constructable, Cacheable):

    NAME = 'aws_ec2'

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

Writing Inventory Plugins

Overview of Developing a Custom Inventory Plugin

Demo plugin: “basic”
https://galaxy.ansible.com/alancoding/basic


Community Authors > alancoding > basic

basic

Basic inventory plugin used in AnsibleFest demo.

alancoding

basic

Creates a number of hosts based on the input “count”.

Type: **inventory_plugin**

- Python content
  - Add DOCUMENTATION
  - Inherit from base class
- Decide namespace (ex. basic)
  - Name file `basic.py`
  - Set `NAME = “basic”`
- Define `verify_file` method, if desired
- **Define `parse` method**
- Make it available for use

https://github.com/willtome/managing-meaningful-inventories/

```
DOCUMENTATION = r'''
    name: Inventory Plugin Basics
    plugin_type: inventory
    author:
      - First Last (@username)
    short_description: Used for instructive purposes.
    version_added: "2.10"
    description:
        - Demonstrates basics of a custom inventory plugin.
    options:
        count:
            description: The number of hosts to make.
            type: integer
            required: True
            default: 1
            required: False
            env:
                - name: HOST_COUNT
    requirements:
        - python >= 3.4
'''
```

Documentation

- This section is required
- Functional impacts:
  - Options documentation is used by get_option() method
  - Environment variables can be used for authentication parameters
  - Types will be enforced in Ansible 2.9

/github.com/willtome/managing-meaningful-inventories/

Imports

```
from ansible.module_utils.six.moves.urllib.parse import urljoin
from ansible.module_utils.urls import Request, ConnectionError, urllib_error

from ansible.errors import AnsibleParserError
from ansible.plugins.inventory import BaseInventoryPlugin

from base64 import b64encode
import json
```

Ansible module utils

Ansible Core libraries

Other dependencies

https://github.com/willtome/managing-meaningful-inventories/

Custom Inventory Plugin Interface

- Inherit from base class (must be InventoryModule) → Can add other mixins for added functionality
- In parse, call methods on inventory
  - add_group(group) ← Group must be created to add host
  - add_host(host, group=None, port=None)
  - add_child(group, entity) ← Group or Host
  - set_variable(entity, varname, value)

Group or Host

Class name must be "InventoryModule"

```
class InventoryModule(BaseInventoryPlugin, Constructable, Cacheable):

    NAME = 'my_custom_plugin'

    def parse(self, inventory, loader, path):
        super(InventoryModule, self).parse(inventory, loader, path)
        self._read_config_data(path)
```

The plugin name

Copy and paste these lines exactly, your logic comes after

https://github.com/willtome/managing-meaningful-inventories/

Use the Inventory Data Methods to Build Inventory

```
def parse(self, inventory, loader, path, cache=True):
    super(InventoryModule, self).parse(inventory, loader, path)
    self._read_config_data(path)

    root_group_name = self.inventory.add_group('root-group')

    for i in range(self.get_option('count')):
        group_name = self.inventory.add_group('group_{}'.format(count))
        self.inventory.add_child(root_group_name, group_name)
        host_name = self.inventory.add_host('host_{}'.format(count))
        self.inventory.add_child(group_name, host_name)

    self.inventory.set_variable(
        root_group_name, 'hashed_password', _hash_password(self.get_option('password'))
    )
```

Copy and paste these lines exactly, your logic comes after

Group must be present to add hosts

get_options requires correct DOCUMENTATION

Parse Logic

https://github.com/willtome/managing-meaningful-inventories/

Share your Custom Inventory Plugin

- ☹️ Put it in a share folder
  - `/usr/share/ansible/plugins/inventory`
- 😐 Point to it in your Ansible config
- 🙂 Use relative path (Ansible 2.8+)
  - Next to playbook (`inventory_plugins/`)
- 😃 Add to a collection (Ansible 2.9+)
  - `stuff/plugins/inventory/foo.py`
  - Upload to Ansible Galaxy

PUT IN GLOBAL SHARE FOLDER
USE ANSIBLE CONFIG
SOURCE CONTROL IN INVENTORY_PLUGINS
DISTRIBUTE AS PART OF COLLECTION
imgflip.com

Troubleshooting Inventory

Troubleshoot Inventory Loading

```
ansible-playbook -i sqlite.yml debug.yml
```

Custom inventory plugin

Project

```
.
├── sqlite.yml
├── hosts.db
└── sqlite.py
```

sqlite.yml

```yaml
plugin: sqlite
db_path: hosts.db
db_table: hosts
```

sqlite.py

```python
from ansible.errors import AnsibleError, AnsibleParserError
from ansible.plugins.inventory import BaseFileInventoryPlugin
import sqlite3
import os

class InventoryModule(BaseFileInventoryPlugin):

   NAME = 'sqlite'

   def verify_file(self, path):
     super(InventoryModule, self).verify_file(path)
     return path.endswith(('sqlite.yml', 'sqlite.yaml'))
```

https://github.com/willtome/managing-meaningful-inventories/

Troubleshoot Inventory Loading

```
[WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with auto plugin: inventory config
'/home/wtome/inventory/sqlite.yml' specifies
unknown plugin 'sqlite'


 [WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with yaml plugin: Plugin configuration
YAML file, not YAML inventory


 [WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with ini plugin: Invalid host pattern
'---' supplied, '---' is normally a sign this is a
YAML file.


 [WARNING]: Unable to parse /home/wtome/inventory/sqlite.yml as an inventory source
```

Troubleshoot Inventory Loading

```
[WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with auto plugin: inventory config
'/home/wtome/inventory/sqlite.yml' specifies
unknown plugin 'sqlite'
```

Auto plugin
is being used

The plugin (.py)
cannot be found

```
 [WAR                    se /home/wtome/inventory/sq                           in: Plugin configuration
YAML                     ry

 [WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with ini plugin: Invalid host pattern
```

1) **Plugin (.py) is not in the correct location**
   (`./inventory_plugins:~/.ansible/plugins/inventory:/usr/share/ansible/plugins/inventory`)

2) **Plugin name (sqlite) does not match NAME in plugin (.py)**

Troubleshoot Inventory Loading

```
[WARNING]:  * Failed to pars                    ory/sqlite.yml wi                    config
'/home/wtome/inventory/sqlit
unknown plugin 'sqlite'
```

Invalid format for yaml plugin

Ansible is trying the yaml plugin

```
[WARNING]:  * Failed to parse /h    e/wtome/inventory/sqlite.yml with yaml plugin: Plugin configuration
YAML file, not YAML inventory
```

```
[WARNING]:  * Failed to parse /home/wtome/inventory/sqlite.yml with ini plugin: Invalid host pattern
```

1) Invalid YAML syntax
2) Intended failure because you weren't using the YAML plugin anyways

Troubleshoot Inventory Loading

[WARNING]: * Failed to parse /home/wtome/inventory/sqlite.yml with auto plugin: inventory config

1) Invalid INI syntax

2) Intended failure because you weren’t using the INI plugin anyways

YAML file, not YAML inventory

[WARNING]: * Failed to parse /home/wtome/inventory/sqlite.yml with ini plugin: Invalid host pattern '---' supplied, '---' is normally a sign this is a YAML file.

Ansible is trying the ini plugin

[WARNING]: Unable to parse /home/wtome/inventory/s source

Troubleshoot Inventory Loading

[WARNING]: * Failed to parse /home/wtome/inventory/sqlite.yml with auto plugin: inventory config '/home/wtome/inventory/sqlite.yml' specifies
unknown plugin 'sqlite'

1) All enabled plugins failed to produce a valid inventory

2) Playbook will continue but only `localhost` will be available

[WARNING]: * Failed to parse /home/wtome/inventory/sqlite.yml with ini plugin: invalid host pattern
'---' supplied, '---' is normally a sign this is a
YAML file.

[WARNING]: Unable to parse /home/wtome/inventory/sqlite.yml as an inventory source

Troubleshoot Inventory Loading

Relative path solution for custom inventory plugins:

```
.
├── hosts.db
├── sqlite.py
└── sqlite.yml
```

```
.
├── hosts.db
├── inventory_plugins
│   └── sqlite.py
└── sqlite.yml
```

This make the sqlite inventory plugin available.

WARNINGS:
Folder inventory_plugins is relative to playbook, so commands

- ansible-config
- ansible-inventory

Will not identify your plugin without additional work (`--playbook-dir=./`)

Advanced Inventory Options

Common / Shared Inventory Plugin Functionality

"Constructed" Functionality

- Compose
  - Set hostvars based on jinja2 expressions of other hostvars
- Conditional groups (sometimes just "groups")
  - Assign hosts to groups based on True/False evaluation of hostvars
- Keyed Groups
  - Create groups based on the value of hostvars (can combine jinja2)
- Filters
  - Limit the hosts returned, specific to each API
- Local Cache
  - Avoids slow API calls between multiple playbook runs

Constructed dev example (foreman): https://github.com/ansible/ansible/pull/62542/files

Simple AWS EC2 “Constructed” Example

- **`regions`** filters the returned hosts
- **`compose`** creates hostvars by templating other hostvars
- **`groups`** (“conditional groups” elsewhere) groups hosts by boolean expression
- **`keyed_groups`** creates groups with names from templating hostvars

example.aws_ec2.yml

```yaml
plugin: aws_ec2
regions:
 - us-east-1
compose:  # Set individual hostvars
 ec2_state: state.name
groups:
 ec2: true  # conditional groups
 platform_undefined: platform is not defined
keyed_groups:  # Create groups for each region
 - key: placement.region
   prefix: aws_region
```

https://github.com/willtome/managing-meaningful-inventories/

Keyed Groups AWS Regions / Zones

keyed.aws_ec2.yml

```yaml
plugin: aws_ec2
keyed_groups:
  - key: placement.region
    parent_group: regions
    prefix: ''
    separator: ''
  - key: placement.availability_zone
    separator: ''
    parent_group: '{{placement.region}}'
```

Hostvars of an ec2 machine:

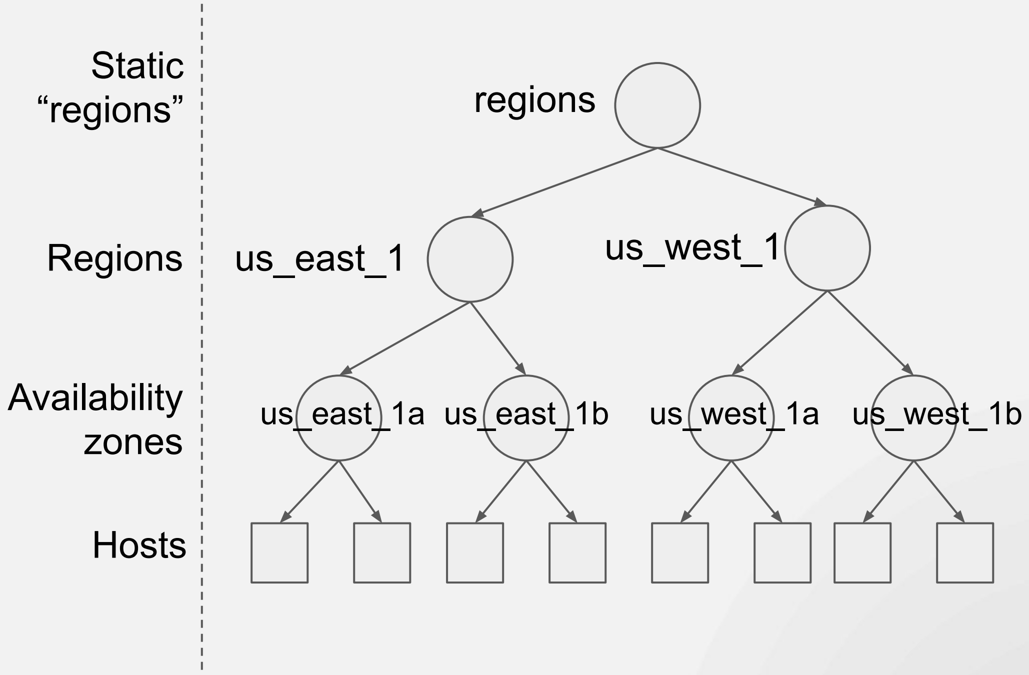
```
"http://ipv4-address.compute-1.amazonaws.com":
{
    "...",
    "placement": {
        "availability_zone": "us-east-2c",
        "group_name": "",
        "region": "us-east-2",
        "tenancy": "default"
    },
    "..."
},
```

https://github.com/willtome/managing-meaningful-inventories/

Keyed Groups AWS Regions / Zones

keyed.aws_ec2.yml

```yaml
plugin: aws_ec2
keyed_groups:
  - key: placement.region
    parent_group: regions
    prefix: ''
    separator: ''
  - key: placement.availability_zone
    separator: ''
    parent_group: '{{placement.region}}'
```



https://github.com/willtome/managing-meaningful-inventories/

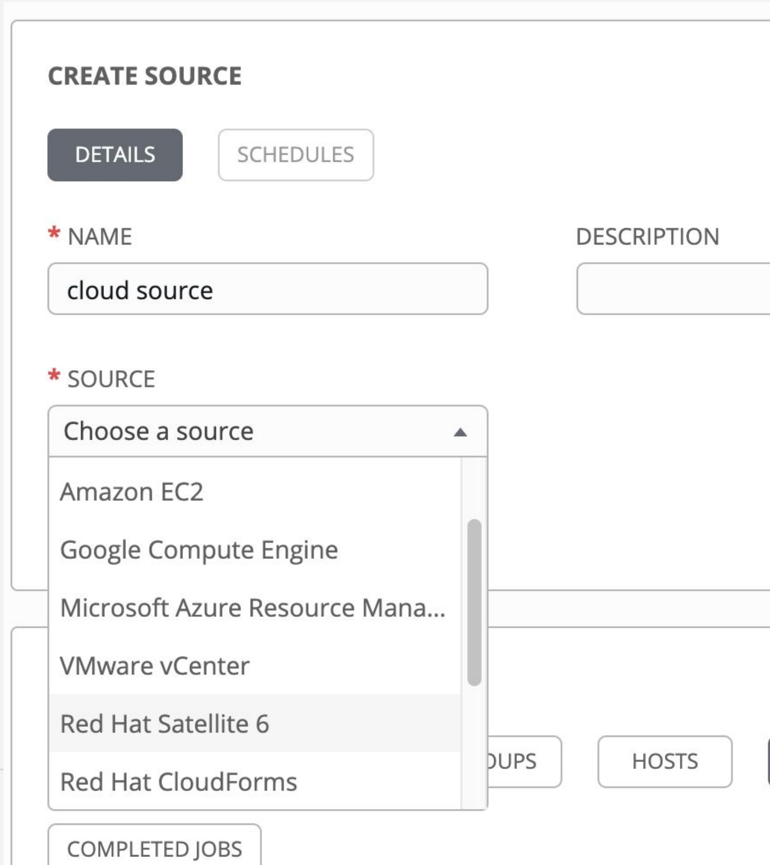
Inventory Plugins In Ansible Tower (use cases)

Sourcing Inventory in Ansible Tower

* venv = custom virtual environment (if needed)

What you need to get inventory from...

1. a Tower built-in type
   - UI
2. an Ansible inventory plugin not built-in
   - SCM inventory file + venv* + UI
3. a custom inventory plugin
   - SCM inventory file + venv* + UI + SCM inventory plugin (relative path)
2. a custom inventory script
   - UI



cloud source

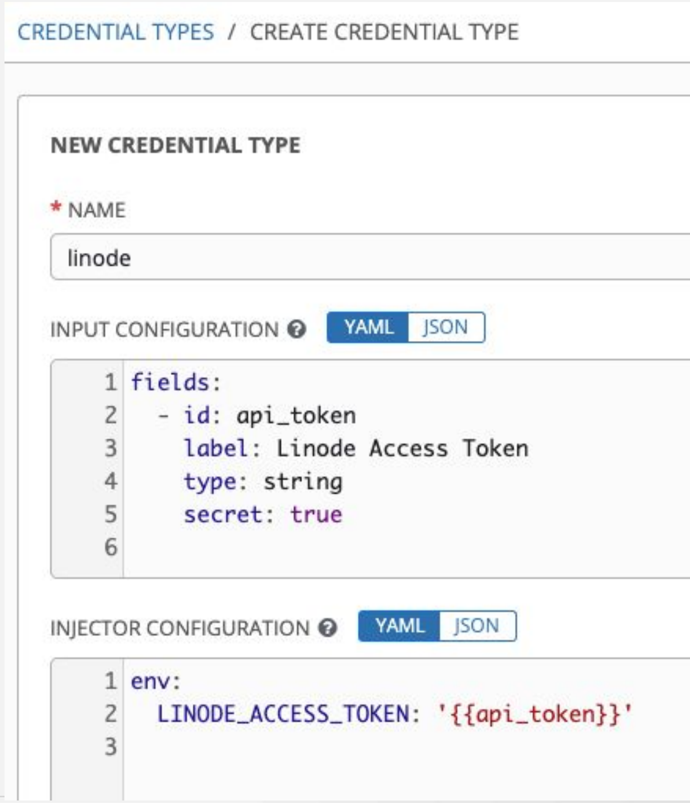
Linode Inventory Plugin (Shipped in Ansible but not Tower)

- Put inventory file in source control
  https://github.com/ansible/test-playbooks

  inventories/linode.yml

  ```
  plugin: linode
  ```

- Create linode custom credential type
- Create custom virtual environment
  - `pip install linode_api4`
- Create Inventory Source from project
  - + credential, venv, etc.
- Update inventory source


CREDENTIAL TYPES / CREATE CREDENTIAL TYPE

NEW CREDENTIAL TYPE

* NAME

linode

INPUT CONFIGURATION YAML JSON

```
fields:
  - id: api_token
    label: Linode Access Token
    type: string
    secret: true

```

INJECTOR CONFIGURATION YAML JSON

```
env:
  LINODE_ACCESS_TOKEN: '{{api_token}}'

```

Custom Inventory Plugins in Tower (Not Shipped with Ansible)

- Relative tree structure is advised (use symlinks if you have to)

```
# sqlite example from earlier

.
├── hosts.db
├── inventory_plugins
│   └── sqlite.py
└── sqlite.yml
```

- Add both to source control
  - Inventory plugin
  - Inventory file

- Collections coming soon in Ansible Tower
  - Add inventory file to source control
  - Add `collections/requirements.yml`

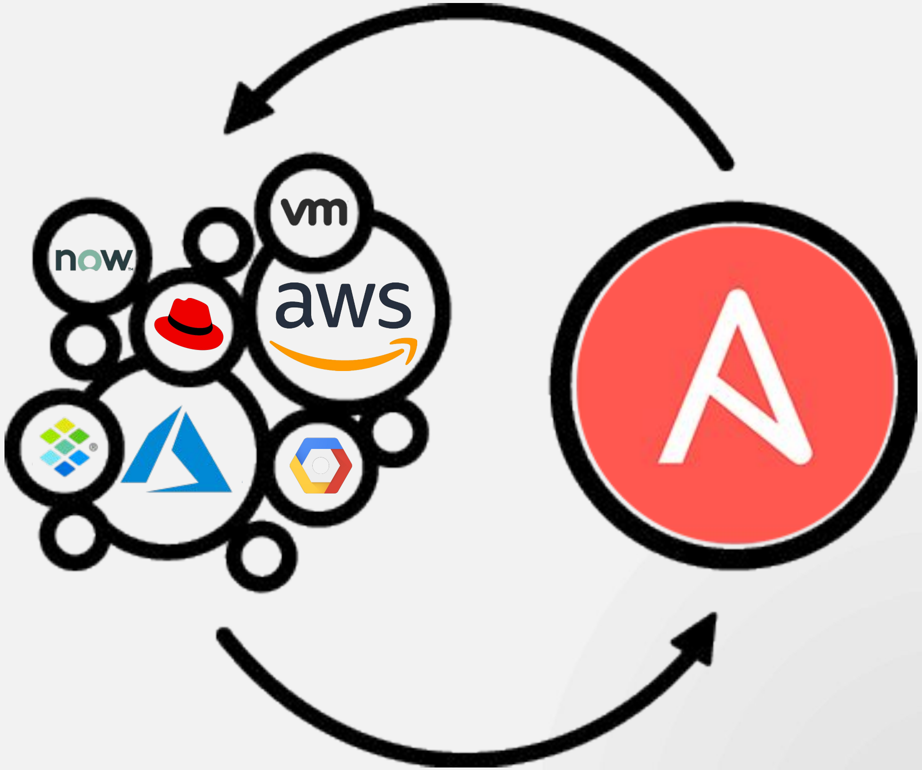
Drive your Automation → Drive your Business

Ansible Automation is glue. Inventory drives your automation which drives your business.

Inventory plugins give you more power than ever to integrate Ansible with your infrastructure and business systems.

Combine multiple SOT (sources of truth) to get complete data and a bigger picture.

Use Ansible as a feedback loop to keep all systems accurate and current.



#ANSIBLEFEST2019

THANK YOU

youtube.com/AnsibleAutomation

facebook.com/ansibleautomation

linkedin.com/company/Red-Hat

twitter.com/ansible

https://github.com/willtome/managing-meaningful-inventories/