Custom Template Filters for a Better World

Adam Johnson - me@adamj.eu

25th March 2015

Me

- DevOps/DBA at YPlan
- Ansible all the things!

Problem: Anonymous EC2 Instances

- No name, no tags, no CPU or network usage

Sneaky Amazon?

- Revenue boosting?

Naughty Jenkins!

- Stopping a build = stopping Ansible on Jenkins finishing the two API calls to launch and tag the instance

Solution

- Already had **clean_resources** playbook run every 15 minutes to clean away cruft
- Just needed to add:

```
- name: delete anonymous instances
  ec2_sql:
    sql: DELETE FROM ec2_instances
         WHERE length(tags) = 0 AND age > 30 minutes
```

- D'oh, **ec2_sql** doesn't exist!

Step 0

- Add the actual play using **ec2** module:

```
- name: delete anonymous instances
  hosts: ec2
  tasks:
    - when: "True"  # length(tags) = 0 AND
                    # age > 30 minutes
      local_action: ec2 state=absent
                        instance_ids={{ ec2_id }}
```

- Will delete **all** instances as-is!
- Need to fill in ‘**when**’

Step 1 - length(tags) = 0

- Host variables:

```
"ec2-54-75-123-123.eu-west-1.compute.amazonaws.com": {
    "ec2_architecture": "x86_64",
    "ec2_client_token": "",
    ...
    "ec2_tag_Name": "my_fancy_machine",
    "ec2_tag_ansible_role": "web",
    ...
}
```

- All tags start **ec2_tag_**

Step 1 - length(tags) = 0

- Drafted python function:

```
In [1]: def filter_prefix(items, prefix):
   ...:     return [x for x in items
                    if x.startswith(prefix)]

In [2]: var_names = ['ec2_architecture',
   ...:              'ec2_tag_Name',
   ...:              'ec2_tag_ansible_role']

In [3]: filter_prefix(var_names, 'ec2_tag_')
Out[3]: ['ec2_tag_Name', 'ec2_tag_ansible_role']
```

Step 1 - length(tags) = 0

- Create **filter_plugins/my_plugins.py** relative to playbook:

```python
def filter_prefix(items, prefix):
    return [x for x in items if x.startswith(prefix)]

class FilterModule(object):
    def filters(self):
        return {
            'filter_prefix': filter_prefix,
        }
```

Step 1 - length(tags) = 0

- Add in to play:

```
- name: delete anonymous instances
  hosts: ec2
  tasks:
    - when: >
        hostvars[inventory_hostname].keys()
          | filter_prefix('ec2_tag_')
          | length == 0

      local_action: ec2 state=absent
                        instance_ids={{ ec2_id }}
```

- *Halfway!*

Step 2 - age > 30 minutes

- Also in host variables:

```
"ec2-54-75-123-123.eu-west-1.compute.amazonaws.com": {
    ...
    "ec2_launch_time": "2014-11-15T11:25:57.000Z",
    ...
}
```

- It's a string since JSON has no standard datetime format.

Step 2 - age > 30 minutes

- Coding by Stack Overflow:

```python
def aws_age_seconds(ec2_launch_time):
    # Strip trailing subsecond part
    launch_time = ec2_launch_time[:-len('.000Z')]
    # Turn into datetime
    time_format = "%Y-%m-%dT%H:%M:%S"
    time_tuple = strptime(launch_time, time_format)
    time_dt = datetime(*time_tuple[:6])
    # Return difference in seconds
    diff = datetime.utcnow() - time_dt
    return seconds_diff.total_seconds()
```

- Add again to filter module

Step 2 - age > 30 minutes

- Final play:

```
- name: delete anonymous instances
  hosts: ec2
  tasks:
    - when: >
        hostvars[inventory_hostname].keys()
          | filter_prefix('ec2_tag_')
          | length == 0
        and ec2_launch_time|aws_age_seconds > 1800
      local_action: ec2 state=absent
                        instance_ids={{ ec2_id }}
```

Problem: **SOLVED**

- Bye bye anonymous instances!

Thank you

- me@adamj.eu