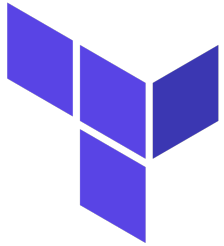

AEROSPIKE
SUMMIT '19
AEROSPIKE
Aerospike Automation
Shakthi Kannan
Senior DevOps Engineer
Aerospike

Tools

CFEngine



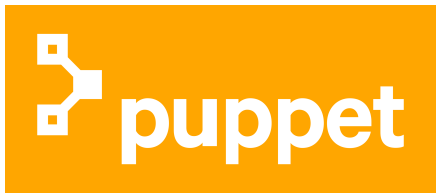
HashiCorp Terraform

CHEF

openstack®

Rex

SALTSTACK



RUDDER

BCFG2

cdist

Pivotal Cloud Foundry®

AEROSPIKE

AEROSPIKE SUMMIT '19

Strategy

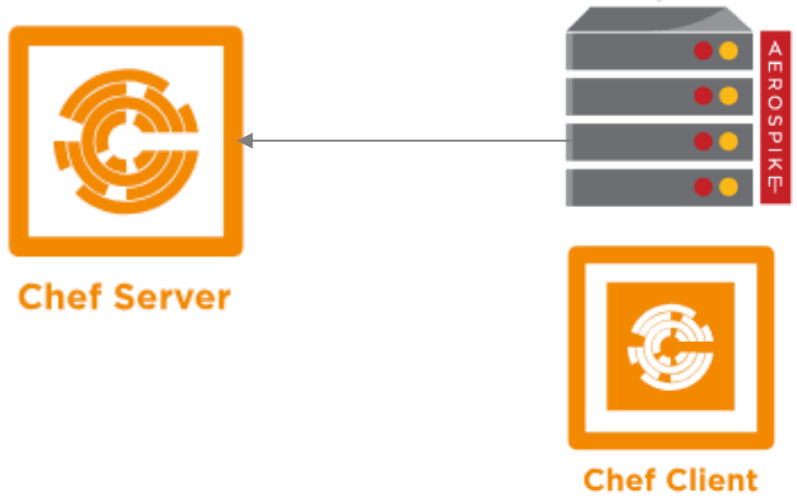
Deployment Models

- Control
  - Synchronous
  - Asynchronous
- Ease of installation
- Security
  - SSH keys on server
  - User access/roles
- Scalability
- Simplicity
- Configuration Management Language
- Scope
  - Full Automation
  - Manual
- Hybrid
- Workflow

Push

Control node

Pull



Inventory Management

Project Layout

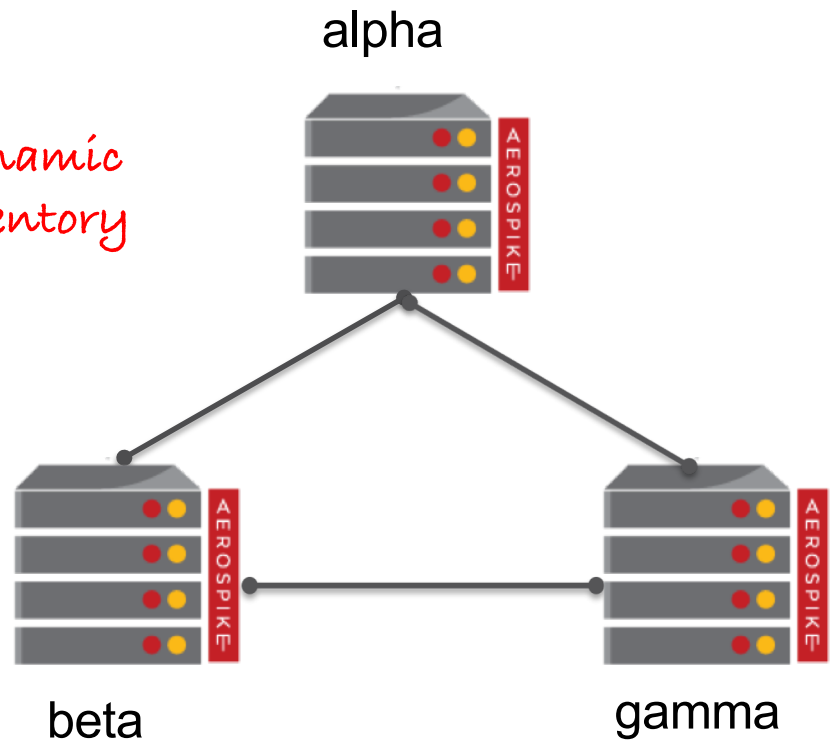
```
README.md
requirements.txt
files/scripts/…
     /templates/…
inventory/alpha/inventory/…
          /alpha/inventory/ec2.py
          /alpha/group_vars/all/all.yml
          /alpha/group_vars/all/…
          /alpha/host_vars/…
          /beta/…
          /gamma/…
playbooks/admin/…
          /provision/…
          /configuration/…
          /meta/…
ansible.cfg
```

configuration files

dynamic inventory

separation of Concerns

- Ansible Way
- Simplicity
- Readability
- Declarative



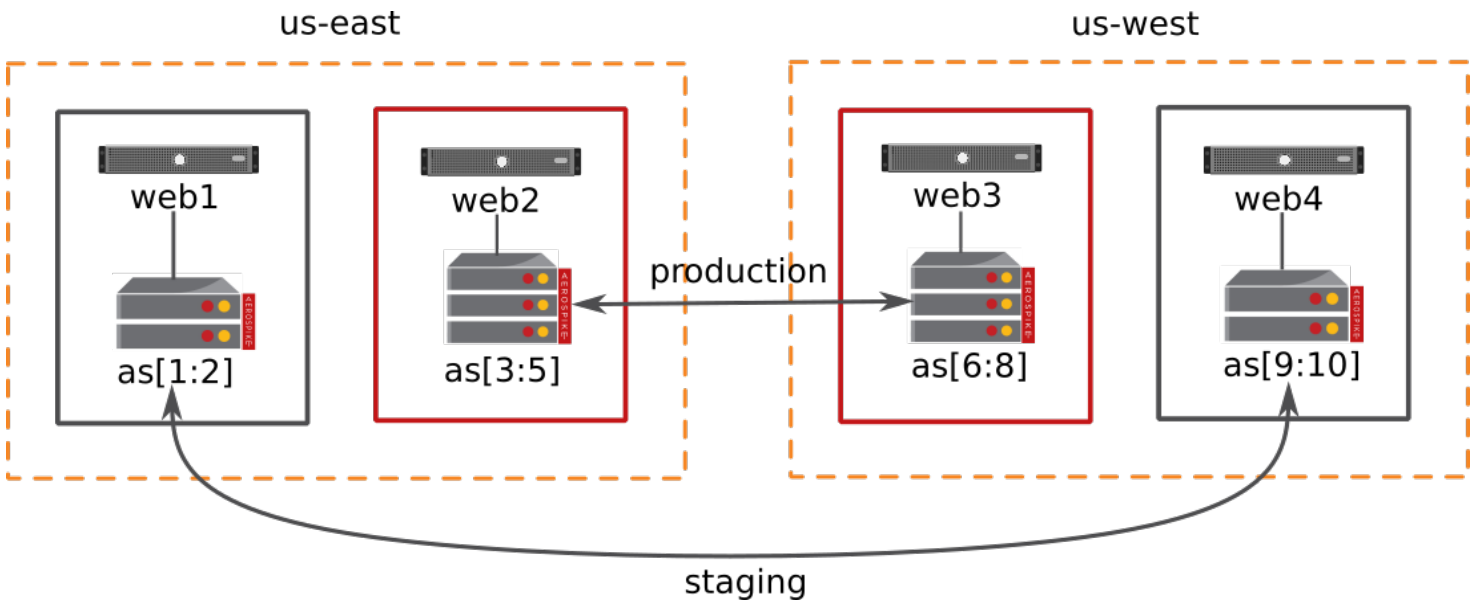
Inventory Groups

inventory/aws/inventory

```
WHAT          WHERE          WHEN

[db]          [east]         [staging]
as[1:10]      as[1:5]        as1
              web1           as2
              web2           as9
                             as10
                             web1
                             web4

[web]         [west]         [production]
web[1:4]      as[6:10]       as3
              web3           as4
              web4           as5
                             as6
                             as7
                             as8
                             web2
                             web3
```

Dynamic inventory
- Reduces human error
- Use existing scripts
- Roll your own

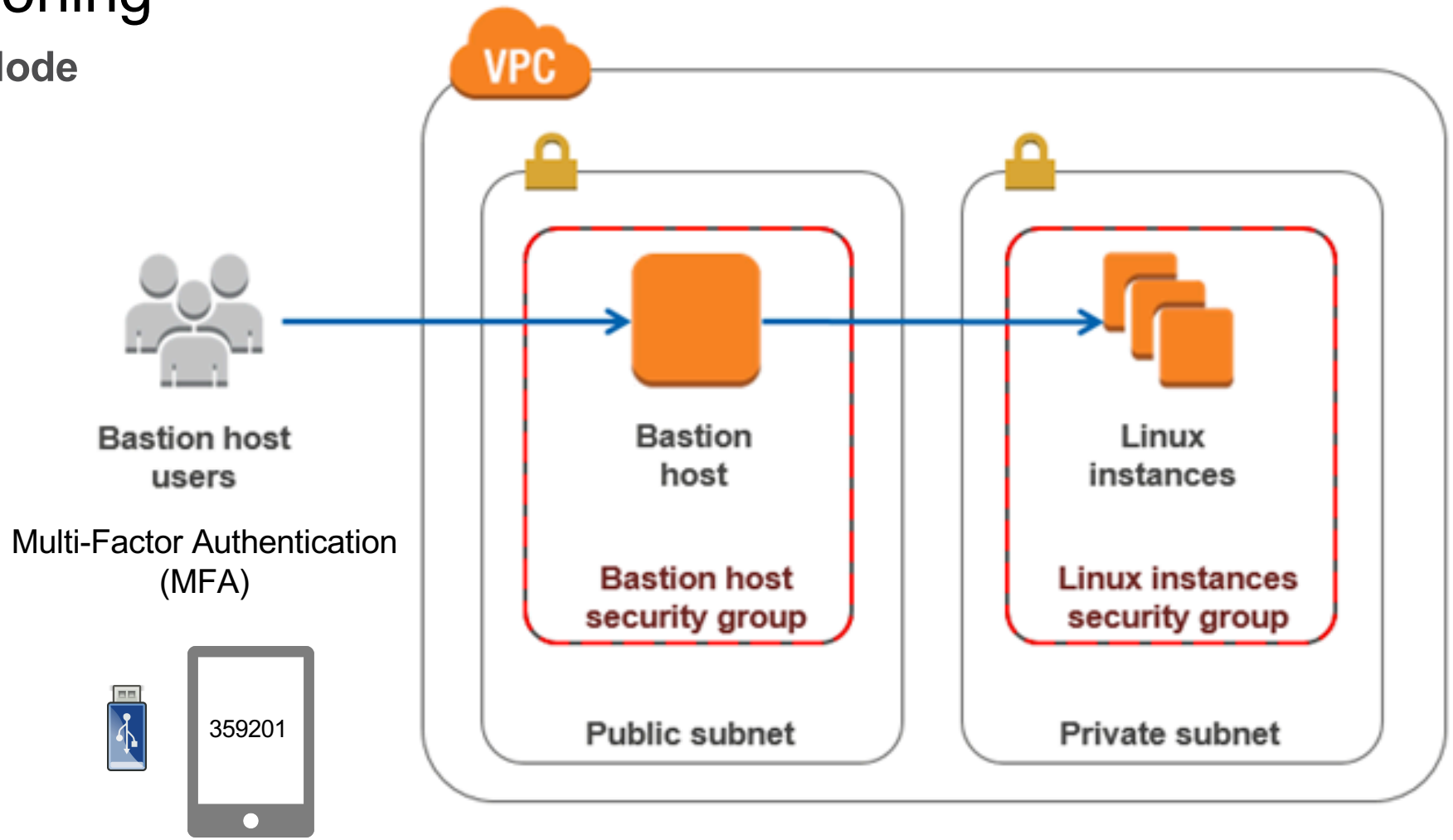
ANSIBLE



Source: https://docs.ansible.com/ansible/latest/user_guide/intro_inventory.html

Provisioning

Bastion Node



Source: https://aws.amazon.com/blogs/security/how-to-record-ssh-sessions-established-through-a-bastion-host/

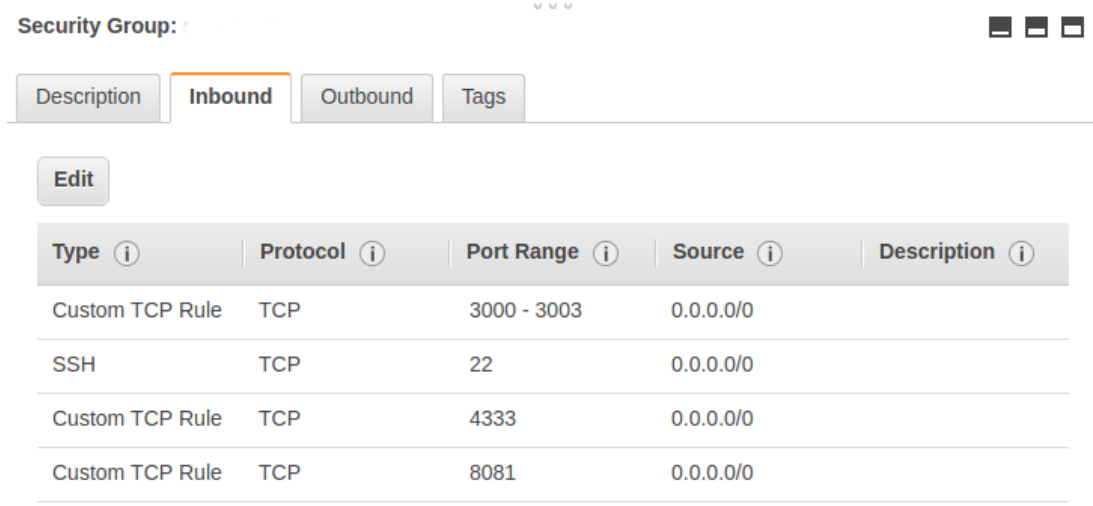
Provisioning

playbooks/provision/aerospike-aws-server.yml

```yaml
---
- name: Provision Aerospike nodes
  hosts: localhost
  gather_facts: true
  tags: [server]

  tasks:
    - name: Create Aerospike security group
      ec2_group:
        name: aerospike-server
        description: sg for Aerospike server
        region: us-east-1
        state: present
        rules:
          - proto: tcp
            from_port: 8081
            to_port: 8081
            cidr_ip: 0.0.0.0/0
          - proto: tcp
            from_port: 22
            to_port: 22
            cidr_ip: 0.0.0.0/0
          - proto: tcp
            from_port: 3000
            to_port: 3003
            cidr_ip: 0.0.0.0/0
      register: sg_aerospike
```

Always name tasks

Vertical reading is easier



- Workflow
- Style Guide
- Consistency
  - Tagging
  - Whitespace
  - Naming
  - Layout
- Version Control
- Iterate

ANSIBLE

AEROSPIKE SUMMIT '19

Provisioning

playbooks/provision/aerospike-aws-server.yml

```
…
- name: Launch instance
  ec2:
    instance_type: t2.micro
    key_name: aerospike@phoenix
    image: ami-bc27135e
    region: us-east-1
    group: aerospike-server,default
    vpc_subnet_id: subnet-9ef744c5   # us-east-1a
    instance_tags:
      Name: alpha
    wait: yes
    assign_public_ip: yes
  register: ec2_info

- name: Add new instance to host group
  add_host: hostname={{ item.public_ip }} groupname=nodes
  with_items: '{{ec2_info.instances}}'

- name: Wait for SSH to come up
  wait_for: host={{ item.public_ip }} port=22 delay=60 timeout=320 state=started
  with_items: '{{ec2_info.instances}}'
```

native YAML

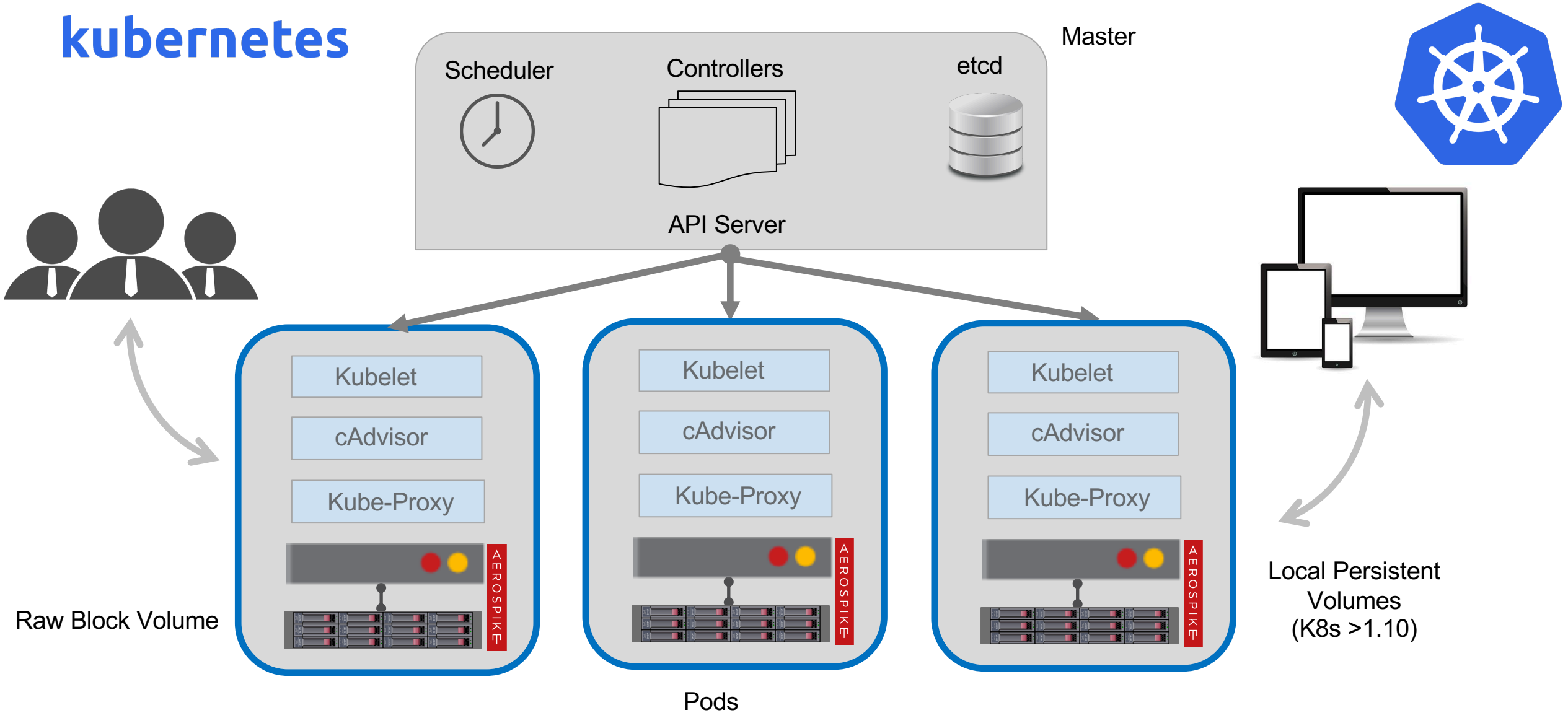
variable substitution

validation steps

- Avoid complexity
- Focused playbooks
- Simple
- Multiple plays
- Do one thing
- Syntax highlighting
- Smoke tests

ANSIBLE

kubernetes



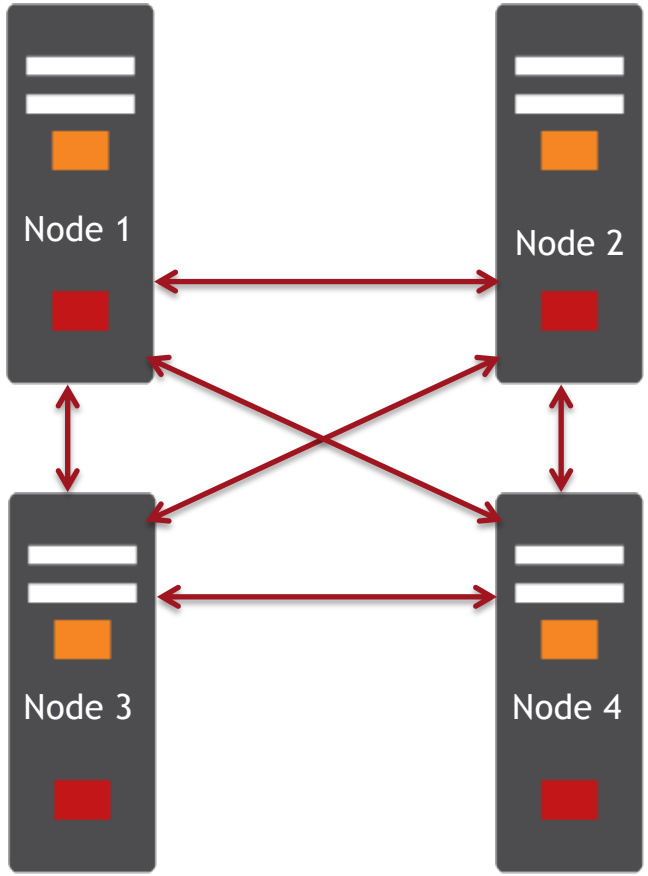
Source: https://www.aerospike.com/docs/deploy_guides/kubernetes/index.html
https://gist.github.com/clusc/1ae75f3a260150ec0f9b0a0c8041eb32

Configuration

aerospike-mesh-cluster.yml

```
---
- name: Create mesh cluster
  hosts: nodes
  gather_facts: true
  become: yes
  become_method: sudo
  tags: [server]

  vars:
    mesh_file: "{{ mesh_config | default('aerospike-mesh') }}"

  tasks:
    - name: Stop aerospike server
      command: service aerospike stop

    - name: Copy Aerospike configuration file
      template:
        src: "../../files/templates/{{ mesh_file }}.conf.j2"
        dest: /etc/aerospike/aerospike.conf
        owner: root
        group: root
        mode: 0644

    - name: Start aerospike server
      command: service aerospike start
```

privilege escalation



Configuration

files/templates/aerospike-aws-mesh.conf.j2

```
network {
        service { …
        }

        heartbeat {
                mode mesh
                address {{ hostvars[inventory_hostname].ansible_default_ipv4.address }}

                port 3002
{% for item in groups['nodes'] %}
{% if item != inventory_hostname %}
                mesh-seed-address-port {{ hostvars[item].ansible_default_ipv4.address }} 3002
{% endif %}
{% endfor %}

                interval 150
                timeout 10
        }

        fabric {
                port 3001
        }

        info {
                port 3003
        }
}
```

Templates

- Simple
- Variable Substitution
- Conditionals
- Control Structures
- Iterations

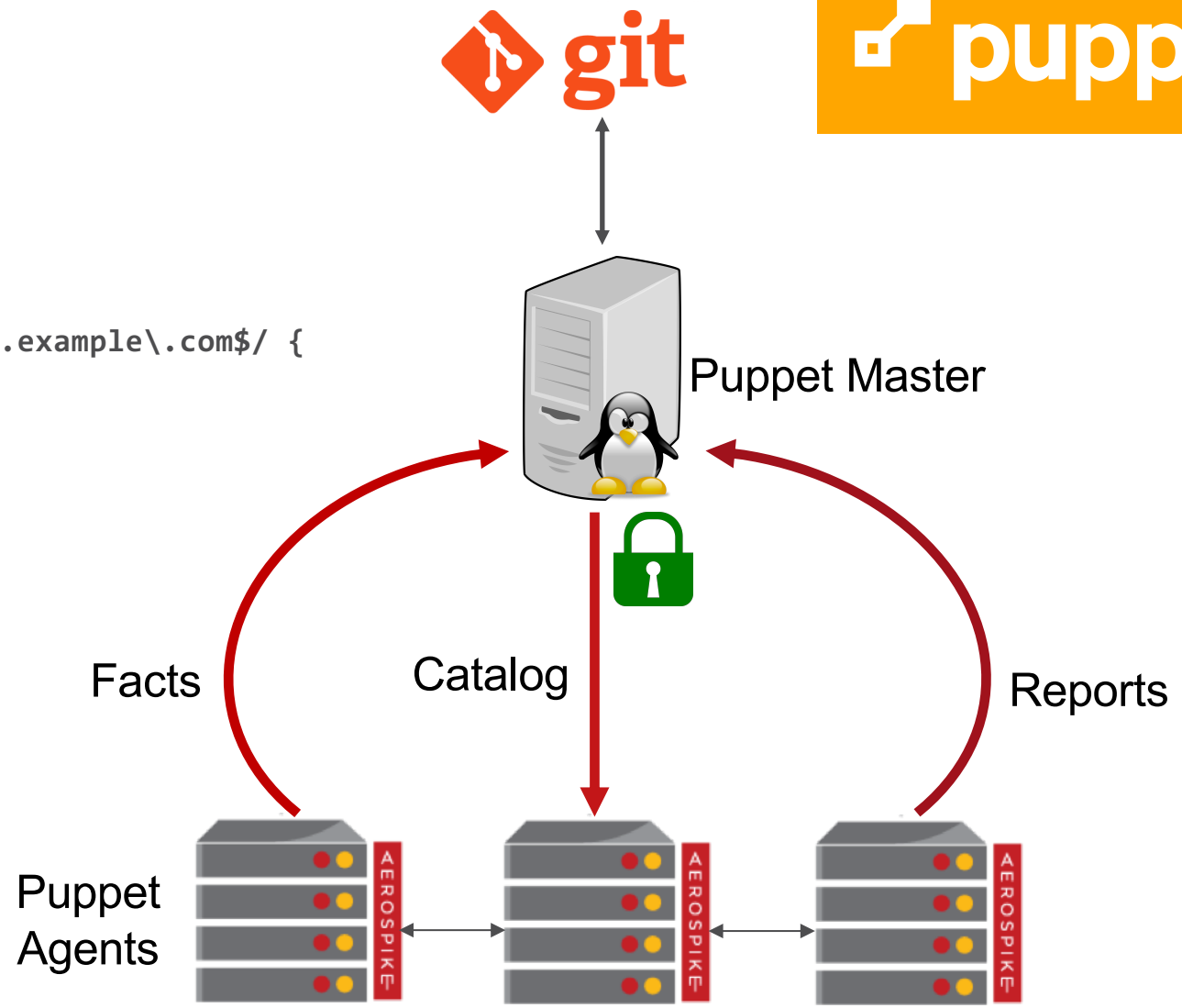
Configuration

aerospike.pp

```
class aerospike_common {
  $environment = 'production'
  include aerospike
  include sudo
  include ...
}

node /^as0(1|2|3|4|5)\.userprofile\.oregon\.example\.com$/ {
  include aerospike_common
  …
}
```

init.pp

```
file {
  "/etc/aerospike":
    ensure => directory,
    owner => root,
    group => root,
    mode => '0755';

  "/var/log/aerospike":
    ensure => directory,
    owner => aerospike,
    group => aerospike,
    mode => '0755';
}
```



Plugins

playbooks/admin/filter_plugins/get_instance_info.py

```
from jinja2.utils import soft_unicode

class FilterModule(object):

    def filters(self):
        return {
            'get_names_and_ips': get_names_and_ips,
        }

def get_names_and_ips(dict):
    names_and_ips = {}
    ip_list=[]

    for key, value in dict['_meta']['hostvars'].items():
        for k1, v1 in value.items():
            if k1 == "ec2_tag_Purpose" and v1 == ”Aerospike":
                k = value['ec2_tag_Name']
                v = value['ec2_ip_address']
                names_and_ips[k] = v
                ip_list.append(v)

    return names_and_ips
```

Plugins included in Ansible

- Action Plugins
- Cache Plugins
- Callback Plugins
- Connection Plugins
- Inventory Plugins
- Lookup Plugins
- Shell Plugins
- Strategy Plugins
- Vars Plugins
- Filters

Invocation

```
- set_fact:
    names_and_ips: "{{ ec2_cache_stdout | get_names_and_ips() }}"

- debug: msg="{{ names_and_ips }}"
```


Scripting

Collectinfo

```
---
- name: Collectinfo
  hosts: machines
  become: yes
  become_method: sudo
  gather_facts: true
  tags: [collectinfo]

  tasks:
    - name: Run collectinfo
      command: asadm -e "collectinfo"
```

Practice

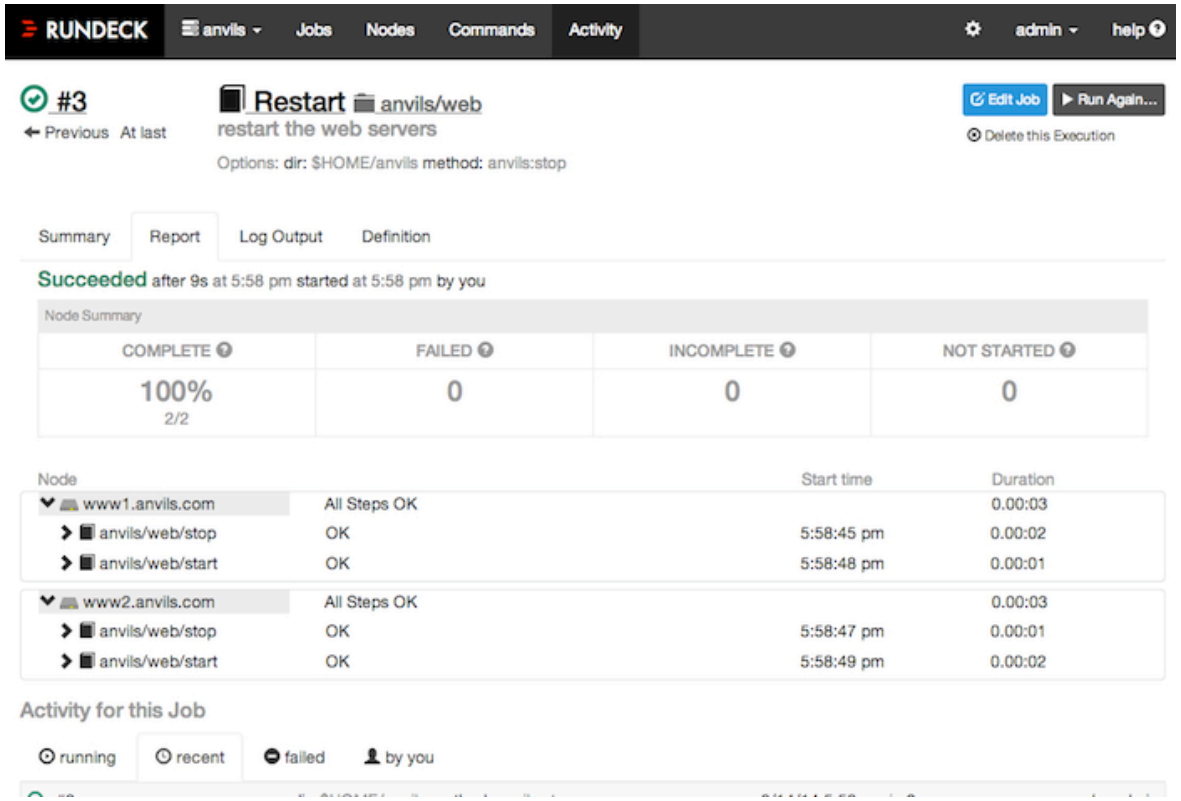
- Enforce the style (post-commit hook)
- Version control
- Iterate
  - Basic playbook and static inventory
  - Refactor
  - Modularize
- Break playbooks into roles
- Attempt pre-defined module
- Read documentation

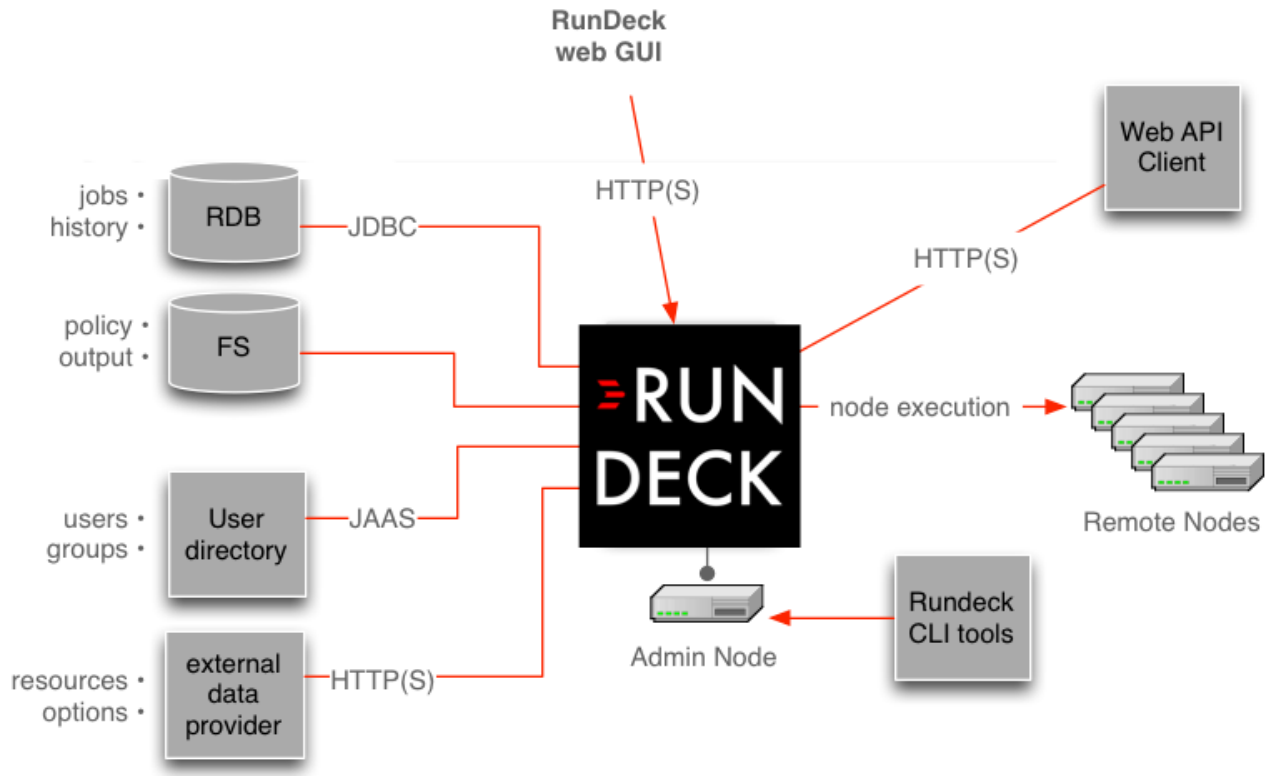
Legend
ok
changed
failed

Output

```
PLAY [Collectinfo] *************************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [as1]

TASK [Run collectinfo] *********************************************************************
changed: [as1] => {"changed": true, "cmd": ["asadm", "-e", "collectinfo"], "delta": "0:00:30.555326", "end": "2019-
03-29 10:38:29.022624", "rc": 0, "start": "2019-03-29 10:37:58.467298", "stderr": "\u001b[2J\u001b[H",
"stderr_lines": …

PLAY RECAP *********************************************************************************
as1                : ok=2    changed=1    unreachable=0    failed=0
```

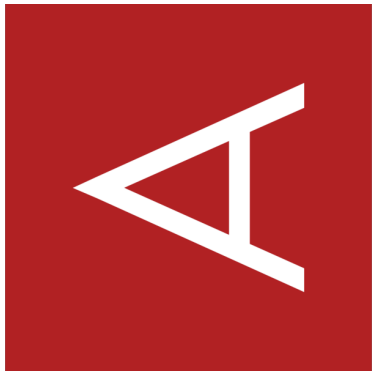

Rundeck





Source: https://docs.rundeck.com/docs/administration/overview/index.html

Upgrades



For each node:

1. Download Aerospike package
2. (Optional) Stop the Aerospike service

```
# For Debian 9
$ sudo systemctl stop aerospike
```

3. Upgrade to the latest package

```
# For Debian 9
$ tar xzf aerospike-server-<EDITION>-<VERSION>-debian9.tgz
$ cd aerospike-server-<EDITION>-<VERSION>-debian9
$ sudo dpkg -i aerospike-server-<EDITION>-<VERSION>.debian9.x86_64.deb
$ sudo dpkg -i aerospike-tools-<EDITION>-<VERSION>.debian9.x86_64.deb
```

4. Start the Aerospike service

```
$ sudo systemctl start aerospike
```

Verification

- As of version 4.3, use `cluster-stable` info command to determine if cluster is stable.
- For versions before 4.3, check:
  - `migrate_allowed` is true
  - `migrate_partitions_remaining` is 0
- Wait for migrations between nodes to complete before upgrading next node.
- `asadm –e “info network”`
- “`service ready: soon there will be cake!`” in Aerospike log.

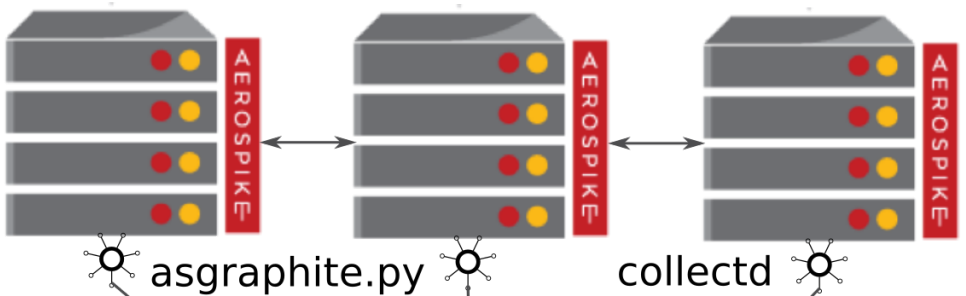
Monitoring

aerospike_nagios.py

Nagios®

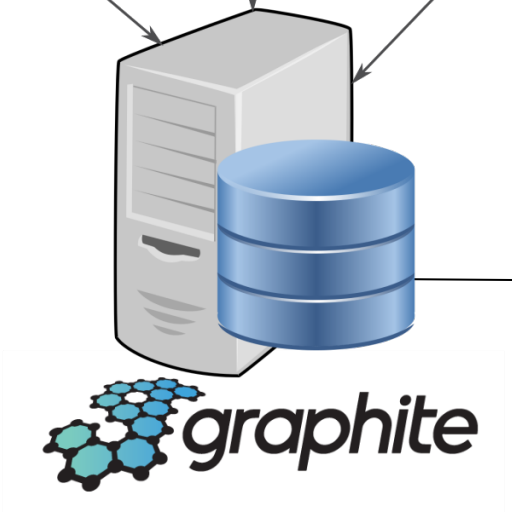
aerospike_discovery.py

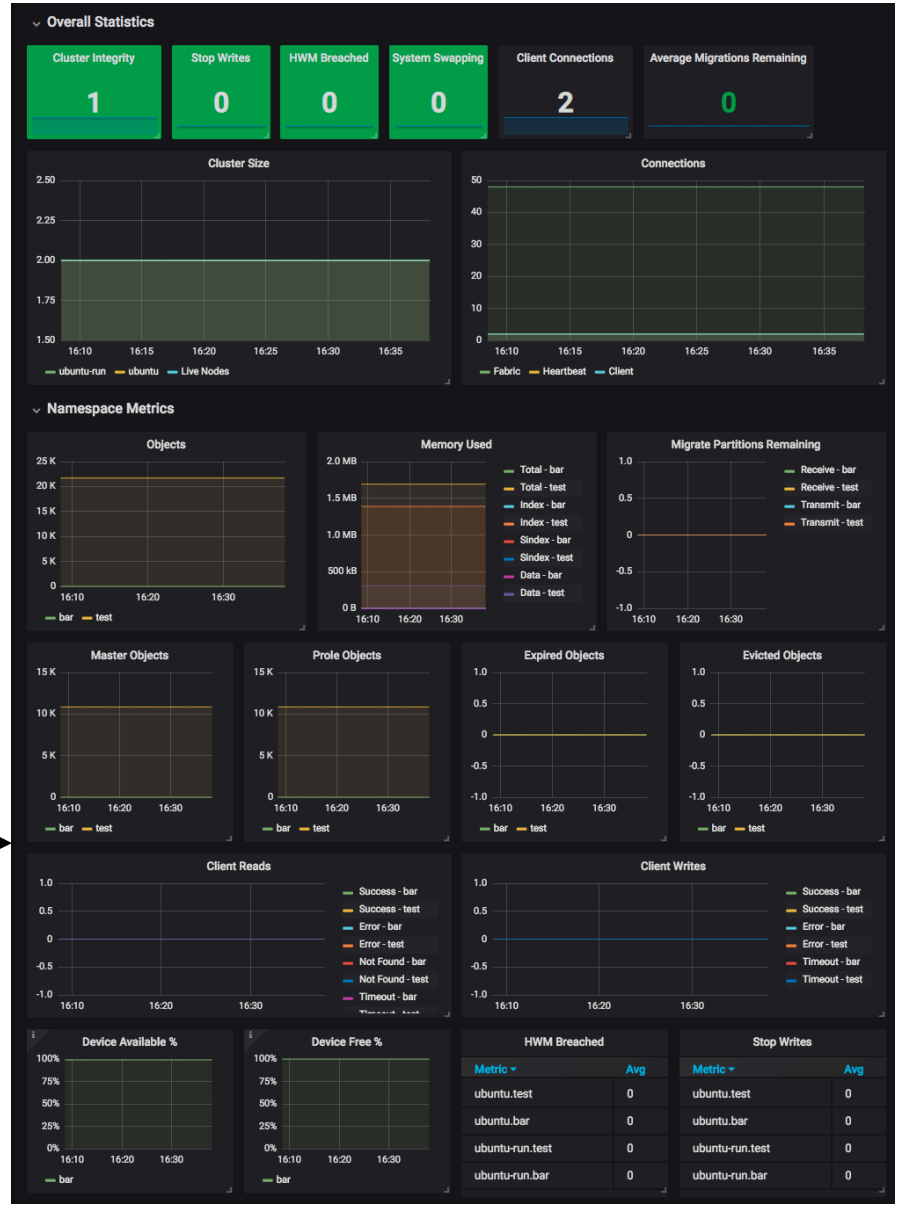
ZABBIX



asgraphite.py

collectd





Source: https://grafana.com/dashboards/8074

Q&A?