

# Ansible + Hadoop

Deploying Hortonworks Data Platform with Ansible



**Michael Young**

Solutions Engineer

February 23, 2017



# About Me

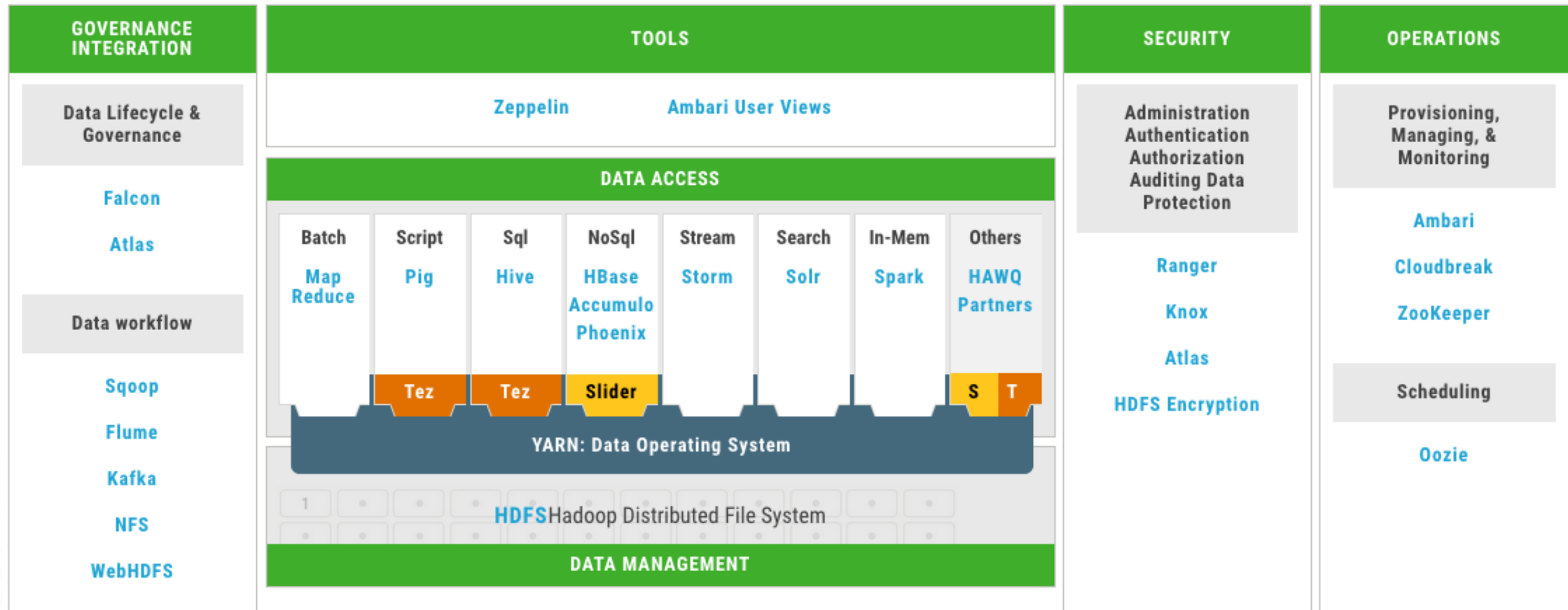
- ◆ Michael Young
  - Solutions Engineer @ Hortonworks
  - 16+ years of experience (Almost all in Public Sector)
  - Information Retrieval (Solr, Elasticsearch)
  - Hadoop (HDP, MapR, Cloudera)
  - DevOps (Ansible, Puppet, Docker, Vagrant)
  - Development (Python, Perl, Node.js)
- ◆  @jaraxal
- ◆  myoung@hortonworks.com



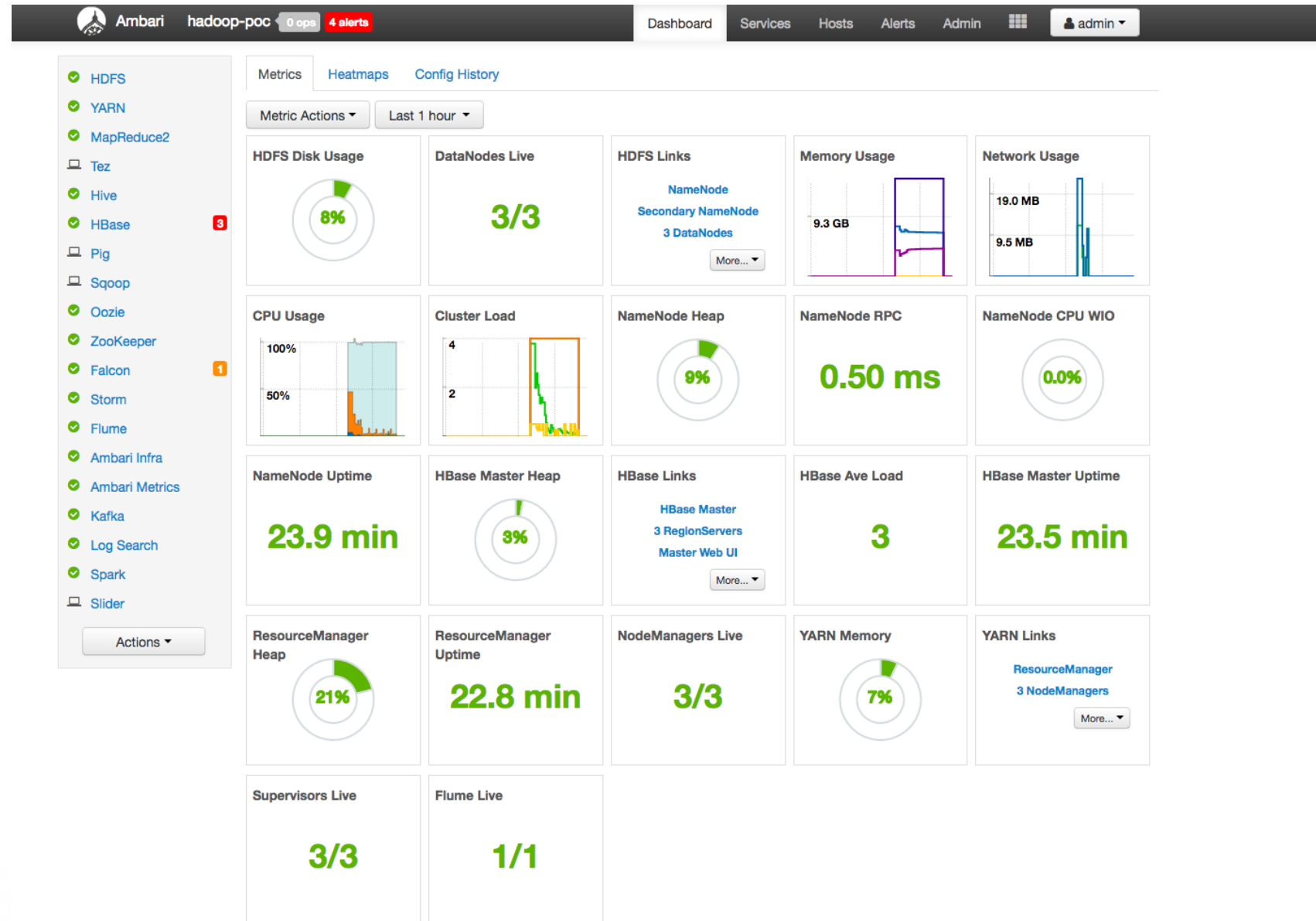
# About Hortonworks

- ◆ Only 100% Open Source Hadoop Company
- ◆ Over 1,000 customers
- ◆ Over 2,100 partners
- ◆ Hortonworks Data Platform (HDP)
- ◆ Hortonworks Data Flow (HDF)
- ◆ Hortonworks Community Connection (HCC)

# Hortonworks Data Platform 2.5



# Ambari: Management and Monitoring



# HDP Provisioning Workflow

- ◆ Prepare Infrastructure
  - Package Repos
  - DNS
  - NTP
- ◆ Prepare OS
  - Disable Transparent Huge Pages
  - Disable Swapping
  - Jumbo Frames
  - Format and mount disk drives
- ◆ Bootstrap Ambari
  - Install Ambari Server
  - Install Ambari Agents
- ◆ Install HDP
  - Interactively via Ambari's web-based UI
  - Automatically via Ambari Blueprints

# Ambari Blueprints for HDP Deployments

- ◆ <https://cwiki.apache.org/confluence/display/AMBARI/Blueprints>
- ◆ Declarative definition of a cluster written in JSON.
- ◆ Preserves best practice configuration across deployments
- ◆ Requires OS configuration prerequisites already in place – Ambari will perform checks and warn you.

```
{
  "configurations" : [
    {
      "configuration-type" : {
        "property-name" : "property-value",
        "property-name2" : "property-value"
      }
    },
    {
      "configuration-type2" : {
        "property-name" : "property-value"
      }
    }
    ...
  ],
  "host_groups" : [
    {
      "name" : "host-group-name",
      "components" : [
        {
          "name" : "component-name"
        },
        {
          "name" : "component-name2",
          "provision_action" : "(INSTALL_AND_START | INSTALL_ONLY)"
        }
        ...
      ],
      "configurations" : [
        {
          "configuration-type" : {
            "property-name" : "property-value"
          }
        }
        ...
      ],
      "cardinality" : "1"
    }
  ],
  "Blueprints" : {
    "stack_name" : "HDP",
    "stack_version" : "2.1",
    "security" : {
      "type" : "(NONE | KERBEROS)",
      "kerberos_descriptor" : {
        ...
      }
    }
  }
}
```

# Automation! Why Ansible?

Ansible has thousands of users, hundreds of customers and over 2,200 community contributors.

<b>750+</b> ANSIBLE MODULES	<b>20,000+</b> GITHUB STARS	<b>200+</b> MEETUPS WORLDWIDE	<b>2,400+</b> COMMUNITY CONTRIBUTORS
-----------------------------------	-----------------------------------	-------------------------------------	--

[JOIN OUR COMMUNITY](#)

```
1 ---
2 # Basic provisioning example
3 - name: Create AWS resources
4   hosts: localhost
5   connection: local
6   gather_facts: False
7
8   tasks:
9     - name: Create a security group
10      ec2_group:
11        name: ansible
12        description: "Ansible Security Group"
13        region: "{{aws_region}}"
14        vpc_id: "vpc-f09e8592"
15        aws_access_key: "{{aws_access_key}}"
16        aws_secret_key: "{{aws_secret_key}}"
17        rules:
18          - proto: all
19            cidr_ip: 100.16.203.185/32
20          - proto: all
21            group_name: ansible
22        rules_egress:
23          - proto: all
24            cidr_ip: 0.0.0.0/0
25        register: firewall
26
27     - name: Create an EC2 instance
28       ec2:
29         aws_access_key: "{{aws_access_key}}"
30         aws_secret_key: "{{aws_secret_key}}"
31         key_name: "{{key_name}}"
32         region: "{{aws_region}}"
33         group_id: "{{firewall.group_id}}"
34         instance_type: "m4.xlarge"
35         image: "{{ami_id}}"
36         wait: yes
37         volumes:
38           - device_name: /dev/sda1
39             volume_type: gp2
40             volume_size: 100
41             delete_on_termination: true
42         exact_count: 4
43         count_tag:
44           Name: hadoop-demo
45         instance_tags:
46           Name: hadoop-demo
47         register: ec2
48
```



# Ansible for HDP Deployments

## ◆ Playbooks

- Bootstrap baseline configuration
- Install DBs
- Install HDP software

## ◆ Roles

- Master Servers
- Slave Servers
- Ambari Server
- Ambari Agent

## ◆ Tasks

- Install prerequisite packages
- Install Ambari Server packages
- Install Ambari Agent packages
- Disable SELinux
- Turn on NTP

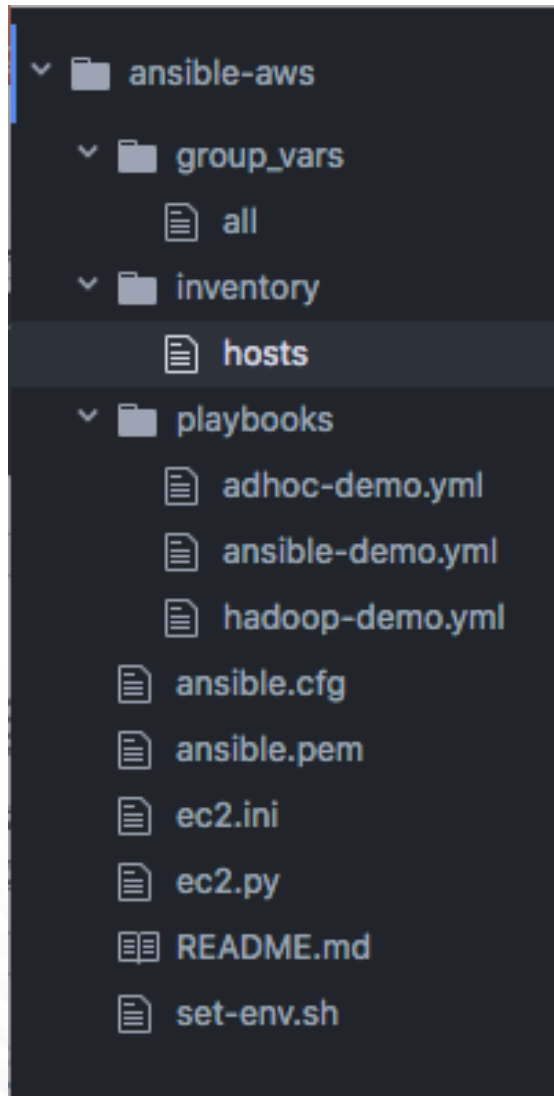
## ◆ Templates

- /etc/hosts
- Ambari Blueprints

## ◆ Files

- Disable THP
- Disable Swapping

# Create 6-node Environment



- Using Amazon AWS
  - 6 x c4.4xlarge instances
- Simple Ansible solution
  - AWS provisioning using ec2 and ec2\_group modules
  - Simple inventory
  - Simple playbook
  - Simple ansible.cfg

# Simple Inventory

```
1 [local]
2 localhost ansible_python_interpreter=/Users/myoung/anaconda/bin/python
```

- ◆ All Ansible commands run locally
- ◆ Uses AWS API
- ◆ Using Anaconda Python

# Simple Playbook: hadoop-demo.yml

```
1 ---
2 # Basic provisioning example
3 - name: Create AWS resources
4   hosts: localhost
5   connection: local
6   gather_facts: False
7
8   tasks:
9     - name: Create a security group
10      ec2_group:
11        name: ansible
12        description: "Ansible Security Group"
13        region: "{{aws_region}}"
14        vpc_id: "vpc-f09e8592"
15        aws_access_key: "{{aws_access_key}}"
16        aws_secret_key: "{{aws_secret_key}}"
17        rules:
18          - proto: all
19            cidr_ip: 100.16.203.185/32
20          - proto: all
21            group_name: hadoop-demo
22        rules_egress:
23          - proto: all
24            cidr_ip: 0.0.0.0/0
25        register: firewall
26
27     - name: Create an EC2 instance
28      ec2:
29        aws_access_key: "{{aws_access_key}}"
30        aws_secret_key: "{{aws_secret_key}}"
31        key_name: "{{key_name}}"
32        region: "{{aws_region}}"
33        group_id: "{{firewall.group_id}}"
34        instance_type: "m4.xlarge"
35        image: "{{ami_id}}"
36        wait: yes
37        volumes:
38          - device_name: /dev/sda1
39            volume_type: gp2
40            volume_size: 100
41            delete_on_termination: true
42        exact_count: 4
43        count_tag:
44          Name: hadoop-demo
45        instance_tags:
46          Name: hadoop-demo
47        register: ec2
48
```

- 2 Tasks
  - Create Security Group
  - Create EC2 Instances

# Task: Provision Security Group

```
9 - name: Create a security group
10   ec2_group:
11     name: ansible
12     description: "Ansible Security Group"
13     region: "{{aws_region}}"
14     vpc_id: "vpc-f09e8592"
15     aws_access_key: "{{aws_access_key}}"
16     aws_secret_key: "{{aws_secret_key}}"
17     rules:
18       - proto: all
19         cidr_ip: 100.16.203.185/32
20       - proto: all
21         group_name: ansible
22     rules_egress:
23       - proto: all
24         cidr_ip: 0.0.0.0/0
25     register: firewall
26
```

- ec2\_group module
  - Region
  - VPC
  - Rules

# Task: Provision Servers

```
27 - name: Create an EC2 instance
28   ec2:
29     aws_access_key: "{{aws_access_key}}"
30     aws_secret_key: "{{aws_secret_key}}"
31     key_name: "{{key_name}}"
32     region: "{{aws_region}}"
33     group_id: "{{firewall.group_id}}"
34     instance_type: "m4.xlarge"
35     image: "{{ami_id}}"
36     wait: yes
37     volumes:
38       - device_name: /dev/sda1
39         volume_type: gp2
40         volume_size: 100
41         delete_on_termination: true
42     exact_count: 4
43     count_tag:
44       Name: hadoop-demo
45     instance_tags:
46       Name: hadoop-demo
47     register: ec2
48
```

- ◆ ec2 module
  - Region
  - Group
  - Instance type
  - AMI
  - Volumes
  - Counts
  - Tags

# Run Playbook

```
HW11380:ansible-aws myyoung$ time ansible-playbook -i inventory/hosts playbooks/hadoop-demo.yml
PLAY [Create AWS resources] *****
TASK [Create a security group] *****
ok: [localhost]
TASK [Create an EC2 instance] *****
changed: [localhost]
PLAY RECAP *****
localhost : ok=2  changed=1  unreachable=0  failed=0

real    0m35.028s
user    0m3.267s
sys     0m2.927s
```

- ◆ `ansible-playbook -i inventory/hosts playbooks/hadoop-demo.yml`
- ◆ Takes ~35 seconds

# DEMO



# Ansible AWS Ad-Hoc Examples

## ◆ Dynamic Inventory

- <https://aws.amazon.com/blogs/apn/getting-started-with-ansible-and-dynamic-amazon-ec2-inventory-management/>
- <https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.py>
- <https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.ini>
- Handy Python script allows you to interact with AWS instances

```
export ANSIBLE_HOSTS='/Users/myoung/Development/ansible-aws/ec2.py'  
export EC2_INI_PATH='/Users/myoung/Development/ansible-aws/ec2.ini'
```

```
HW11380:ansible-aws myoung$ ansible key_aws -m ping -u centos  
174.129.62.122 | SUCCESS => {  
  "changed": false,  
  "ping": "pong"  
}
```

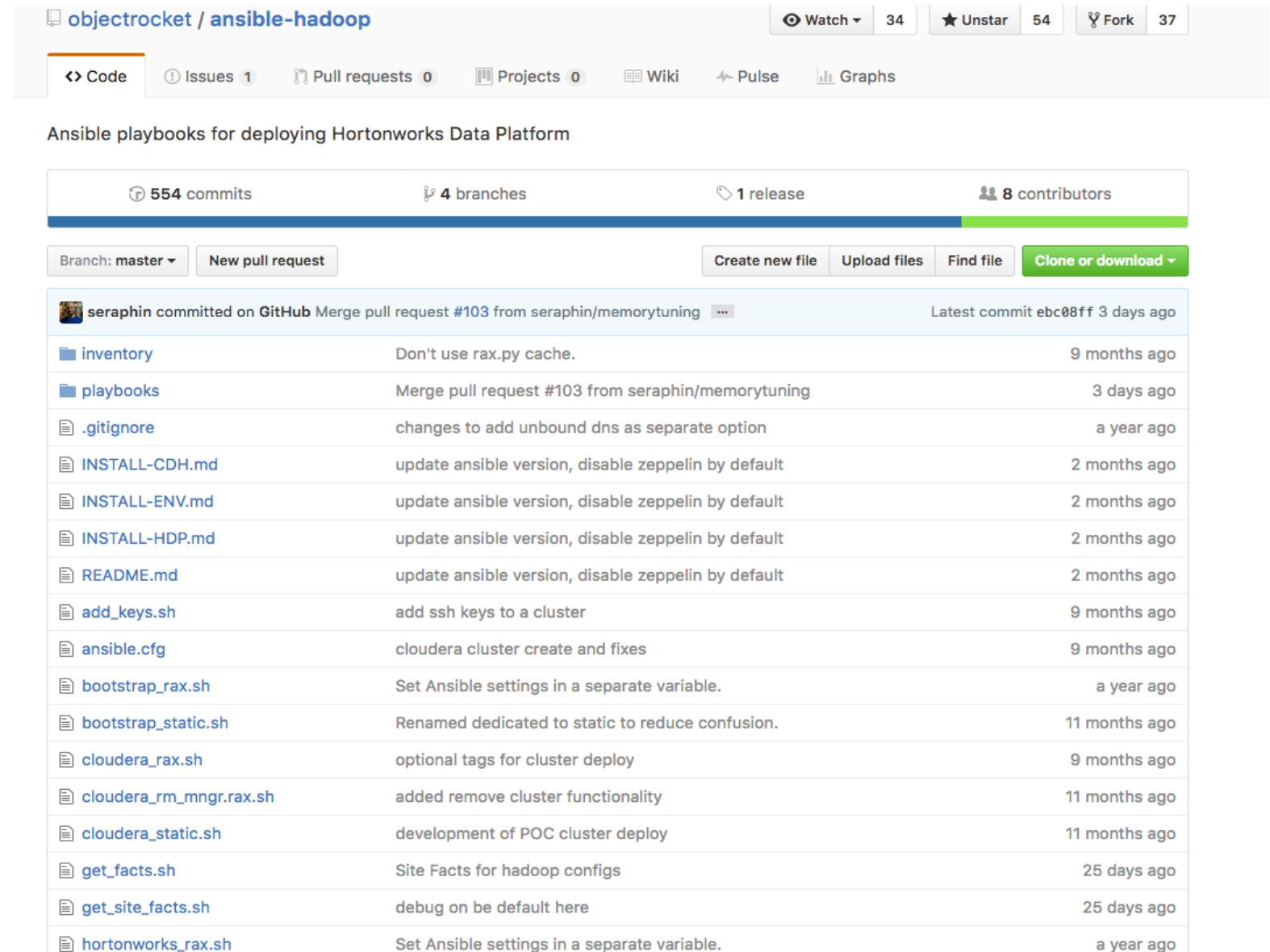
```
HW11380:ansible-aws myoung$ ansible key_aws -m yum -a "name=git state=present" -u centos  
174.129.62.122 | SUCCESS => {  
  "changed": false,  
  "msg": "",  
  "rc": 0,  
  "results": [  
    "git-1.8.3.1-6.el7_2.1.x86_64 providing git is already installed"  
  ]  
}
```

# Ready to Create?

- ◆ Inventory
  - Dev
  - Test
  - Prod
- ◆ Playbook
  - Roles
  - Tasks
  - Templates
  - Files
  - Handlers
- ◆ Generally an iterative process
- ◆ Start small, move towards more complex
- ◆ Entire process could take a couple of days to a couple of weeks

# Why re-invent the wheel?

- <https://github.com/objectrocket/ansible-hadoop>
- ObjectRocket is a Rackspace company.
- Enables deployment of hadoop clusters using Ansible
- Supports Rackspace cloud and existing environments
- Ansible == 2.1.3.0 (2.2 is not supported at the moment)
- Expects RHEL/CentOS 6/7 or Ubuntu 14 hosts.
- Simple – Configure, then run two scripts



objectrocket / ansible-hadoop

Watch 34 Unstar 54 Fork 37

Code Issues 1 Pull requests 0 Projects 0 Wiki Pulse Graphs

Ansible playbooks for deploying Hortonworks Data Platform

554 commits 4 branches 1 release 8 contributors

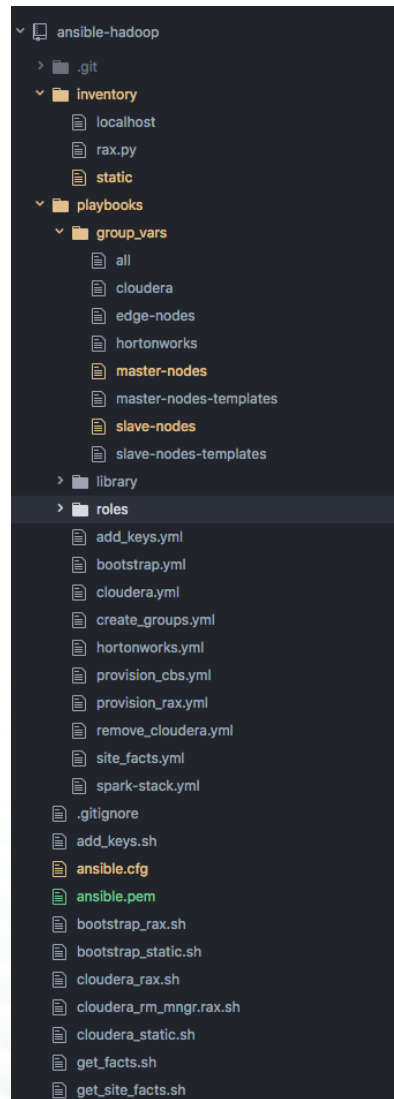
Branch: master New pull request Create new file Upload files Find file Clone or download

seraphin committed on GitHub Merge pull request #103 from seraphin/memorytuning Latest commit ebc08ff 3 days ago

inventory	Don't use rax.py cache.	9 months ago
playbooks	Merge pull request #103 from seraphin/memorytuning	3 days ago
.gitignore	changes to add unbound dns as separate option	a year ago
INSTALL-CDH.md	update ansible version, disable zeppelin by default	2 months ago
INSTALL-ENV.md	update ansible version, disable zeppelin by default	2 months ago
INSTALL-HDP.md	update ansible version, disable zeppelin by default	2 months ago
README.md	update ansible version, disable zeppelin by default	2 months ago
add_keys.sh	add ssh keys to a cluster	9 months ago
ansible.cfg	cloudera cluster create and fixes	9 months ago
bootstrap_rax.sh	Set Ansible settings in a separate variable.	a year ago
bootstrap_static.sh	Renamed dedicated to static to reduce confusion.	11 months ago
cloudera_rax.sh	optional tags for cluster deploy	9 months ago
cloudera_rm_mgr.rax.sh	added remove cluster functionality	11 months ago
cloudera_static.sh	development of POC cluster deploy	11 months ago
get_facts.sh	Site Facts for hadoop configs	25 days ago
get_site_facts.sh	debug on be default here	25 days ago
hortonworks_rax.sh	Set Ansible settings in a separate variable.	a year ago

# DEMO

# Minimal Configuration Needed



- ◆ inventory/static
- ◆ playbooks/group\_vars/master\_nodes
- ◆ playbooks/group\_vars/slave\_nodes
- ◆ playbooks/group\_vars/hortonworks
- ◆ ansible.cfg
- ◆ Optional: custom repos and blueprints

# Modify inventory/static

```
1 [master-nodes]
2 master01 ansible_host=ec2-54-152-145-254.compute-1.amazonaws.com ansible_ssh_user=centos
3
4 [slave-nodes]
5 slave01 ansible_host=ec2-54-164-169-79.compute-1.amazonaws.com ansible_ssh_user=centos
6 slave02 ansible_host=ec2-54-209-36-208.compute-1.amazonaws.com ansible_ssh_user=centos
7 slave03 ansible_host=ec2-107-20-69-195.compute-1.amazonaws.com ansible_ssh_user=centos
8
9 #[edge-nodes]
10
```

- ◆ Add information for **master**, **slave** and **edge** nodes
- ◆ Use public IP for **ansible\_host**
- ◆ Default user for my AMI is **centos**. Set **ansible\_ssh\_user** appropriately.
- ◆ Using key, so no password specified
- ◆ Don't forget to comment unused node types (edge-nodes)

# Modify playbook/group\_vars/\*\_nodes

```
1 ---
2 #####
3 # use template file for example references          #
4 # ~/ansible-hadoop/playbooks/group_vars/master-nodes-templates#
5 #####
6
7 cluster_interface: 'eth0'
8
```

- ◆ Refer to template files for examples
- ◆ Most options are geared towards Rackspace cloud

# Modify playbook/group\_vars/hortonworks

```
1 ---
2 cluster_name: 'hadoop-poc'
3 adminnode: 'ambari-node'
4 hdp_version: '2.5'
5 ambari_version: '2.4.2.0'
6 admin_password: 'admin'
7 services_password: 'AsdQwe123'
8 alerts_contact: 'root@localhost.localdomain'
9 wait: true
10 wait_timeout: 1800 # 30 minutes
11
12 install_spark: true
13 install_zeppelin: false # >= HDP2_5
14 install_flume: true
15 install_hbase: true
16 install_storm: true
17 install_kafka: true
18 install_falcon: true
19 tachyon_service: false
20
21 data_disks_filesystem: xfs
22 configure_firewall: false
23 custom_blueprint: false
24 custom_blueprint_template: blueprint-custom.j2
25 custom_cluster_template: cluster-template-custom.j2
26 custom_repo: false
27 custom_repo_url:
28   * http://public-repo-1.hortonworks.com/HDP-LABS/Projects/Erie-Preview/2.5.0.0-7/centos7/
29   * api/v1/stacks/HDP/versions/2.5/operating\_systems/redhat7/repositories/HDP-2.5
```

- ◆ Specify Configuration Details
  - version of HDP and Ambari to install
  - components to install
  - admin and service passwords
  - repo URL
- ◆ I left this as-is



# Modify ansible.cfg

```
1 [defaults]
2 host_key_checking = False
3 timeout = 60
4 ansible_keep_remote_files = True
5 library = playbooks/library/site_facts
6 private_key_file=/home/centos/ansible-hadoop/ansible.pem
7
8 [ssh_connection]
9 ssh_args = -C -o ControlMaster=auto -o ControlPersist=60s
10
```

- ◆ Change **library** value to **playbooks/library/site\_facts**
- ◆ Specify location of **private\_key\_file**.

# Run bootstrap\_static.sh

```
PLAY RECAP *****
localhost      : ok=3    changed=2    unreachable=0    failed=0
master01       : ok=21   changed=15   unreachable=0    failed=0
slave01        : ok=21   changed=15   unreachable=0    failed=0
slave02        : ok=21   changed=15   unreachable=0    failed=0
slave03        : ok=21   changed=15   unreachable=0    failed=0

real    7m9.776s
user    0m55.377s
sys     0m6.874s
```

- ◆ Performs the common bootstrap configurations
- ◆ ***\$ bash bootstrap\_static.sh***
- ◆ Takes ~8 minutes
- ◆ Consistent timing regardless of node count
- ◆ Same tasks done on all servers in parallel – Ansible approach.

# DEMO

# Run hortonworks\_static.sh

```
PLAY RECAP *****
localhost      : ok=5    changed=3  unreachable=0  failed=0
master01      : ok=32   changed=13 unreachable=0  failed=0
slave01       : ok=10   changed=5  unreachable=0  failed=0
slave02       : ok=10   changed=5  unreachable=0  failed=0
slave03       : ok=10   changed=5  unreachable=0  failed=0

real    18m49.317s
user    1m21.293s
sys     0m11.096s
```

- ◆ Performs the Hortonworks installation
- ◆ ***\$ bash hortonworks\_static.sh***
- ◆ Takes ~19 minutes (4-node m4.xlarge cluster)
- ◆ master01 had significantly more tasks to implement

# Retrying Tasks is Normal

```
TASK [ambari-server : Wait for the cluster to be built] *****
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (180 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (179 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (178 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (177 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (176 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (175 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (174 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (173 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (172 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (171 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (170 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (169 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (168 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (167 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (166 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (165 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (164 retries left).
FAILED - RETRYING: TASK: ambari-server : Wait for the cluster to be built (163 retries left).
```

- ◆ The last task is waiting for the cluster to be built
- ◆ Normal to see many failed checks with retry attempts.

# Monitor Ambari During Install

**0 Background Operations Running** X

Operations	Start Time	Duration	Show: All (8)
✓ Start components on host ip-172-31-20-179	Today 19:01	30.51 secs	100%
✓ Install components on host ip-172-31-20-179	Today 18:53	485.88 secs	100%
✓ Start components on host ip-172-31-28-50	Today 19:01	24.57 secs	100%
✓ Install components on host ip-172-31-28-50	Today 18:53	485.70 secs	100%
✓ Start components on host ip-172-31-18-122	Today 19:02	25.34 secs	100%
✓ Install components on host ip-172-31-18-122	Today 18:53	521.46 secs	100%
✓ Start components on host ip-172-31-31-157	Today 19:04	321.43 secs	100%
✓ Install components on host ip-172-31-31-157	Today 18:53	642.83 secs	100%

Do not show this dialog again when starting a background operation OK

Monitor Ambari during cluster installation.

# One Node: ~1,000 seconds

✓ Start components on host ip-172-31-31-157	Today 19:04	321.43 secs	<div style="width: 100%;"></div>	100%	▶
✓ Install components on host ip-172-31-31-157	Today 18:53	642.83 secs	<div style="width: 100%;"></div>	100%	▶

Name	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions	Components
ip-172-31-18-122	172.31.18.122	/default-rack	4 (4)	15.26GB	<div style="width: 10%;"></div>	0.04	HDP-2.5.3.0-37	20 Components
ip-172-31-20-179	172.31.20.179	/default-rack	4 (4)	15.26GB	<div style="width: 10%;"></div>	0.06	HDP-2.5.3.0-37	20 Components
ip-172-31-28-50	172.31.28.50	/default-rack	4 (4)	15.26GB	<div style="width: 10%;"></div>	0.01	HDP-2.5.3.0-37	20 Components
ip-172-31-31-157	172.31.31.157	/default-rack	4 (4)	15.26GB	<div style="width: 10%;"></div>	0.49	HDP-2.5.3.0-37	41 Components

- One node took ~1,000 seconds to complete install and startup
- This node is the master node, has more components
- Room to decrease deployment time by adding more master nodes

# Five Node Cluster

```
PLAY RECAP *****
localhost      : ok=5   changed=3   unreachable=0   failed=0
master01       : ok=10  changed=5   unreachable=0   failed=0
master02       : ok=32  changed=13  unreachable=0   failed=0
slave01        : ok=10  changed=5   unreachable=0   failed=0
slave02        : ok=10  changed=5   unreachable=0   failed=0
slave03        : ok=10  changed=5   unreachable=0   failed=0

real    15m27.286s
user    1m10.594s
sys     0m9.754s
```

✓ Start components on host ip-172-31-13-199	Today 10:33	160.26 secs	100%
✓ Install components on host ip-172-31-13-199	Today 10:24	534.72 secs	100%
✓ Start components on host ip-172-31-11-113	Today 10:33	219.67 secs	100%
✓ Install components on host ip-172-31-11-113	Today 10:24	512.16 secs	100%

Name	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions	Components
ip-172-31-11-113	172.31.11.113	/default-rack	4 (4)	15.26GB		0.55	HDP-2.5.3.0-37	31 Components
ip-172-31-13-199	172.31.13.199	/default-rack	4 (4)	15.26GB		0.69	HDP-2.5.3.0-37	33 Components
ip-172-31-15-130	172.31.15.130	/default-rack	4 (4)	15.26GB		0.17	HDP-2.5.3.0-37	22 Components
ip-172-31-4-66	172.31.4.66	/default-rack	4 (4)	15.26GB		0.04	HDP-2.5.3.0-37	20 Components
ip-172-31-7-11	172.31.7.11	/default-rack	4 (4)	15.26GB		0.03	HDP-2.5.3.0-37	20 Components

- 5 x m4.xlarge
- 2 master and 3 slave nodes
- Took ~15 minutes
- ~3 minutes faster than 4-node cluster.
- More even distribution of components on master servers





# Six Node Cluster

```
PLAY RECAP *****
localhost      : ok=5    changed=3    unreachable=0    failed=0
master01      : ok=10   changed=5    unreachable=0    failed=0
master02      : ok=10   changed=5    unreachable=0    failed=0
master03      : ok=32   changed=13   unreachable=0    failed=0
slave01       : ok=10   changed=5    unreachable=0    failed=0
slave02       : ok=10   changed=5    unreachable=0    failed=0
slave03       : ok=10   changed=5    unreachable=0    failed=0

real    15m17.136s
user    1m10.345s
sys     0m10.227s
```

- 6 x m4.xlarge
- 3 master and 3 slave nodes
- Took ~15 minutes
- No apparent improvement in deployment times over 5-node cluster.

# Comparing Instance Sizes - Six Node Cluster

```
PLAY RECAP *****
localhost      : ok=5    changed=3    unreachable=0    failed=0
master01       : ok=10   changed=5    unreachable=0    failed=0
master02       : ok=10   changed=5    unreachable=0    failed=0
master03       : ok=32   changed=13   unreachable=0    failed=0
slave01        : ok=10   changed=5    unreachable=0    failed=0
slave02        : ok=10   changed=5    unreachable=0    failed=0
slave03        : ok=10   changed=5    unreachable=0    failed=0

real    11m52.938s
user    0m58.018s
sys     0m7.829s
```

- ◆ m4.xlarge vs c4.4xlarge
- ◆ Same cluster configuration
- ◆ 3 master and 3 slave nodes
- ◆ Took ~12 minutes
- ◆ ~3 minutes faster than m4.xlarge cluster

# Number & Size of Nodes

- ◆ Factoring the number and size of nodes to decrease deployment time is interesting, but not generally important
- ◆ Size your cluster on based on data size and workload
  - More Data: more local storage per slave node, more slave nodes
  - More Queries: more memory and cpu per slave node, more slave nodes
  - High Availability: Use at least 3 master nodes, at least 3 slave nodes
- ◆ Minimum recommended cluster size for production is ~12 nodes

# Summary

- ◆ Easily created an AWS environment using a simple Ansible playbook
  - Takes ~1-2 minutes, includes modifying playbook
- ◆ Easily deployed 6-node HDP cluster
  - Ran playbook from an AWS node with Ansible
  - Modify a couple of configuration files
  - Run 2 commands and have an HDP cluster in < 20 minutes
- ◆ Demonstrated how cluster size and instance type affected deployment times

# Questions?