Managing Kubernetes in Air Gap/Offline Environments

刘荣 Rong Zhang, Suning.com (@riverzhang on github)
Agenda

• What is an Air Gap/Offline environments
• Kubeadm Air Gap/Offline installer support
• Trusted Cloud Native Registry – Harbor
• Production Ready Kubernetes Cluster – Kubespray
• How to setup kubespray in an Air Gap/Offline environment
What is an Air Gap/Offline environments?
Challenging and a quite common requirement

- Government
- Large corporations
- China
- Private cloud user

- User’s privacy guarantee
- Already maintaining own infrastructure and CDN
- Costs and not Public cloud
Kubeadm  Air Gap/Offline installer support
[root@k8s-dev home]# kubeadm config images list

- k8s.gcr.io/kube-apiserver-amd64:v1.11.10
- k8s.gcr.io/kube-controller-manager-amd64:v1.11.10
- k8s.gcr.io/kube-scheduler-amd64:v1.11.10
- k8s.gcr.io/kube-proxy-amd64:v1.11.10
- k8s.gcr.io/pause:3.1
- k8s.gcr.io/etcd-amd64:3.2.18
- k8s.gcr.io/coredns:1.1.3

Kubeadm configmap
- kubernetesVersion
- Version must be v1.11.10, v1.12.1, v1.13.3?

How to define the version of etcd and image repo?

https://github.com/kubernetes/kubernetes/pull/71135
Based on feedback from KubeCon China 2018.
Supported v1.13, Thanks @luxas
• **Kubeadm Issues:**
solve the kubeadm offline and air-gapped support issues #1041
https://github.com/kubernetes/kubeadm/issues/1041

• **kubernetes PR:**
kubeadm: fix offline and air-gapped support #67397
https://github.com/kubernetes/kubernetes/pull/67397
Kubeadm Air Gap/Offline install

- Download images:
  - `k8s-release-repo`
  - `docker pull/save images`

- Access:
  - `Host able to access`
  - `scp/tar/docker load`

- Install packages:
  - `Kubeadm init`
  - `k8s Master/Node`

- Join:
  - Master-1
  - Master-2
  - Master-3
  - Master-N
  - Node-1
  - Node-2
  - Node-3
  - Node-N

- Install network plugin:
  - `KubeCon`
  - `CloudNativeCon`
  - `China 2019`
Kubeadm Air Gap/Offline install

Images Registry

Kubeadm + Ansible
Harbor is a trusted cloud native registry that stores, signs, and scans content. The mission is to provide cloud native environments the ability to confidently manage and serve container images.
Harbor Architecture

- Consumers
- 3rd party components
- Harbor components
- Supporting services

API Routing

Core Service (API/Auth/GUI)

Users (GUI/API)

Container Schedulers/Runtimes

Job Service

Admin Service

Vulnerability Scanning

Trusted Content

Image Registry

Key/Value Storage

SQL Database

Local or Remote Storage (block, file, object)

Harbor Packaging

LDAP/Active Directory

Docker

Kubernetes
Typical Use Cases

- Image consistency through software lifecycle
- Shipping images in “binary” format
- Image replication unlocks interesting deployment architectures
- Auth\{Z,N\}
- Vulnerability scanning
- Image signing
- Helm chart management
Shipping “Binaries”

Git → Kubernetes → CI → Dev Registry → Test Registry → Staging Registry

- Images

Kubernetes

Git

CI

Dev Registry

Test Registry

Staging Registry
Kubespray is a sig-cluster-lifecycle’s project to create, configure and manage Kubernetes clusters. It provides optional, additive functionality on top of core Kubernetes.
Kubespray at a glance

- Cluster lifecycle manager
- Flexible and composable
- Production ready
- Ansible based
- One package-based component: Docker, Cri-o etc...
- Multi-arch
- Community driven since 2015
- Base of kubeadm since 2018
- Just bring your own machine
- Certified Kubernetes Installer (CNCF)
Deployment workflow

- Bootstrap OS
- Preinstall step
- Install Docker
- Install etcd
- Install Kubernetes Master
- Install Kubernetes Minion
- Configure network plugin
- Addons
High Availability

- **Etcd**
  - Native support for all clients to connect to all ETCD instances

- **Apiserver**
  - External LB (Cloud LB, F5)
  - Local LB (nginx, proxy), static pod in kubernetes cluster
Local LB (default)
Offline options

- **Binaries**
  - `foo_download_url`
- **Images**
  - When using docker, `docker_insecure_registries` and `docker_registry_mirrors`
- **System packages**
  - When `container_manager=docker`, `docker_foo_repo_base_url`, `docker_foo_repo_gpgkey`, `dockerproject_bar_repo_base_url` and `dockerproject_bar_repo_gpgkey` (where `foo` is the distribution and `bar`is system package manager)
  - When `container_manager=crio`, `crio_rhel_repo_base_url`
- **Helm charts**
  - When using Helm, `helm_stable_repo_url`

https://kubespray_downloads.md#offline-environment
Air /Gap options

pull once, push many

download_localhost: True to make localhost the download delegate. This can be useful if cluster nodes cannot access external addresses. Download container images and binaries only once and then push them to the cluster nodes.

https://air_gap_kubespray_download
Community

Stars: 6400+
Forks: 2600+
Commits: 4400+
Contributors: 450+
Join us

Slack
#kubespray
#kubespray-dev

Github
http://kubespray.io
http://github.com/kubernetes-sigs/kubespray

WeChat
Kubespray China
How to setup kubespray in an Air Gap/Offline environment
Air Gap/Offline: High Availability Install

- Harbor
  - Harbor offline installer
  - Collect Images and Publish Images Or CI pipeline

- Kubespray
  - Install requirements.txt
  - Modify the roles/download images registry
  - Install kubernetes cluster and add private registry(harbor) options, etc...
Lifecycle of cluster operations

- `cluster.yml`
  - Install or reconfigure a cluster
- `upgrade-cluster.yml`
  - Graceful rolling upgrade to a new version
  - Backup, etcd snapshots taken during upgrade
- `scale.yml`
  - Add a node to an existing cluster
- `remove-node.yml`
  - Remove a particular node from a cluster
- `reset.yml`
  - Uninstall an entire cluster
New cluster

ansible-playbook -i inventory/sample/cluster1.ini  cluster.yml  -e kube_version=v1.12.3  -e docker_insecure_registries=[‘mirror.registry.io’,’172.19.16.11’]

cluster1.ini
[all]
kube-master01 ansible_host=10.32.7.143  ip=10.32.7.143
kube-node01  ansible_host=10.32.7.135  ip=10.32.7.135

[kube-master]
kube-master01

[etcd]
kube-master01

[kube-node]
kube-node01

[k8s-cluster:children]
kube-master
kube-node
Scale node

ansible-playbook -i inventory/sample/cluster1.ini scale.yml -e kube_version=v1.12.5 -e docker_insecure_registries=['mirror.registry.io', '172.19.16.11']

cluster1.ini
[all]
kube-master01 ansible_host=10.32.7.143 ip=10.32.7.143
kube-node01 ansible_host=10.32.7.135 ip=10.32.7.135
kube-node02 ansible_host=10.32.7.136 ip=10.32.7.136

[kube-master]
kube-master01

[etcd]
kube-master01

[kube-node]
kube-node01
Kube-node02

[k8s-cluster:children]
kube-master
kube-node
```bash
ansible-playbook -i inventory/sample/cluster1.ini cluster.yml -e kube_version=v1.12.5 -e docker_insecure_registries=['mirror.registry.io','172.19.16.11'] --skip-tags=node,network,apps
```

---

<table>
<thead>
<tr>
<th>Role</th>
<th>Host</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube-master01</td>
<td>ansible_host=10.32.7.143</td>
<td>ip=10.32.7.143</td>
</tr>
<tr>
<td>kube-master02</td>
<td>ansible_host=10.32.7.144</td>
<td>ip=10.32.7.144</td>
</tr>
<tr>
<td>kube-master03</td>
<td>ansible_host=10.32.7.145</td>
<td>ip=10.32.7.145</td>
</tr>
<tr>
<td>kube-node01</td>
<td>ansible_host=10.32.7.135</td>
<td>ip=10.32.7.135</td>
</tr>
</tbody>
</table>

---

[k8s-cluster:children]
kube-master
kube-node
Upgrade cluster

ansible-playbook -i inventory/sample/cluster1.ini upgrade-cluster.yml -e kube_version=v1.13.3 -e docker_insecure_registries=['mirror.registry.io','172.19.16.11']

Other operations:

• Upgrade docker:
  --tags=docker
• Upgrade etcd:
  --tags=etcd
• Upgrade Kubernetes master components:
  --tags=master
• Upgrade kubelet:
  --tags=node  --skip-tags=k8s-gen-certs,k8s-gen-tokens
• Upgrade network plugins:
  --tags=network
• Upgrade l add-ons:
  --tags=apps
Remove node and Uninstall cluster

- **Remove nodes**
  ansible-playbook -i inventory/sample/cluster1.ini remove-node.yml --extra-vars "node=kube-node02"

- **Uninstall cluster**
  ansible-playbook -i inventory/sample/cluster1.ini reset.yml
Thanks

Q&A