Arktos and Mizar Single Node Installation Guide (GCP)

Date: 4 Jan 2022

Introduction

This document is intended for new users to install the Arktos platform with Mizar as the underlying network technology.

Installation Steps

- Prepare lab machine, the preferred OS is **Ubuntu 18.04**. If you are using GCP, the recommended instance size is e2-standard-16 vCPUs - 64 GB RAM and the storage size is 128GB or more

cd

git clone https://github.com/CentaurusInfra/mizar.git

cd mizar

chmod 755 setup-machine-arktos.sh

./setup-machine-arktos.sh

```
root@prajwal-ark:~# cd
root@prajwal-ark:~# git clone https://github.com/CentaurusInfra/mizar.git
Cloning into 'mizar'...
remote: Enumerating objects: 6756, done.
remote: Counting objects: 100% (978/978), done.
remote: Compressing objects: 100% (567/567), done.
remote: Total 6756 (delta 575), reused 713 (delta 390), pack-reused 5778
Receiving objects: 100% (6756/6756), 11.53 MiB | 27.53 MiB/s, done.
Resolving deltas: 100% (4500/4500), done.
root@prajwal-ark:~# cd mizar
root@prajwal-ark:~/mizar# chmod 755 setup-machine-arktos.sh
root@prajwal-ark:~/mizar# ./setup-machine-arktos.sh
Setup: Install go (currently limited to version 1.13.9)
```

The lab machine will be rebooted once the above script is completed, you will be automatically logged out of the lab machine.

- Log onto your lab machine, then run `bootstrap.sh` script from the Mizar project folder to bootstrap your lab machine.

- Once bootstrap is completed, you can then compile Mizar. Make sure to run these in sudo mode:

```
root@prajwal-ark:~/mizar# bash bootstrap.sh
NOTE: This script will reboot the system if you opt to allow kernel update.
      If reboot is not required, it will log you out and require re-login for new permissions to take effect.

Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu bionic-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'python-scapy' instead of 'scapy'
build-essential is already the newest version (12.4ubuntu1).
pkg-config is already the newest version (0.29.1-0ubuntu2).
pkg-config set to manually installed.
lcov is already the newest version (1.13-3).
libcmocka-dev is already the newest version (1.1.1-1).
libelf-dev is already the newest version (0.170-0.4ubuntu0.1).
```

cd ~/mizar

sudo su

Install grpcio tools:

python3 -m pip install --user grpcio-tools

make

```
root@prajwal-ark:~/mizar# python3 -m pip install --user grpcio-tools
Collecting grpcio-tools
  Cache entry deserialization failed, entry ignored
  Downloading https://files.pythonhosted.org/packages/55/7a/b6d5a5d69d6ab0df70a7ceed16f0e9a6c0bdc09376c92fa5638d08803fa4/grpcio-tools-1.
    100% |████████████████████████████████| 2.2MB 578kB/s
Collecting grpcio>=1.43.0 (from grpcio-tools)
  Cache entry deserialization failed, entry ignored
  Downloading https://files.pythonhosted.org/packages/c6/6b/5f7cd38ff3ac80f47cbe56618fe45502f90b41a56f5d9e248ee574e14687/grpcio-1.43.0.t
    100% |████████████████████████████████| 21.5MB 59kB/s
Collecting protobuf<4.0dev,>=3.5.0.post1 (from grpcio-tools)
  Cache entry deserialization failed, entry ignored
  Downloading https://files.pythonhosted.org/packages/c1/12/7479ece04931984162698bfaa05cbb2fc23d7f6ee1ab5146cfc6ade56a31/protobuf-3.19.1
    100% |████████████████████████████████| 163kB 7.3MB/s
Requirement already satisfied: setuptools in /usr/lib/python3/dist-packages (from grpcio-tools)
Requirement already satisfied: six>=1.5.2 in /usr/lib/python3/dist-packages (from grpcio>=1.43.0->grpcio-tools)
Building wheels for collected packages: grpcio-tools, grpcio
  Running setup.py bdist_wheel for grpcio-tools ... done
  Stored in directory: /root/.cache/pip/wheels/85/c7/a5/d41a749ff6bbb1e64684ebf5a6deb86bece79c3285278227f5
  Running setup.py bdist_wheel for grpcio ... done
  Stored in directory: /root/.cache/pip/wheels/b2/98/2d/7aac9507590b235d4ed0ca74e24c91d3c3704cc86416adcd31
Successfully built grpcio-tools grpcio
Installing collected packages: grpcio, protobuf, grpcio-tools
Successfully installed grpcio-1.43.0 grpcio-tools-1.43.0 protobuf-3.19.1
root@prajwal-ark:~/mizar# make
LDFLAGS=-Llib/usr/lib64 -l:libbpf.a -l:libelf.a -lz -lnsl -static-liblsan -static-libubsan
mkdir -p core
mkdir -p cov
mkdir -p lcov/report
mkdir -p build/bin
mkdir -p build/tests
mkdir -p build/xdp
mkdir -p test/trn_func_tests/output
mkdir -p test/trn_perf_tests/output
git submodule update --init --recursive
DESTDIR=../../../../lib/ make install -C src/extern/libbpf/src
make[1]: Entering directory '/root/mizar/src/extern/libbpf/src'
if [ ! -d '../../../../lib//usr/include/bpf' ]; then install -d -m 755 '../../../../lib//usr/include/bpf'; fi; install bpf.h libbpf.h bt
```

Build arktos-network-controller (as it is not part of arktos-up.sh yet)

cd $HOME/go/src/k8s.io/arktos

sudo ./hack/setup-dev-node.sh

make all WHAT=cmd/arktos-network-controller

```
root@prajwal-ark:~/mizar# cd $HOME/go/src/k8s.io/arktos
root@prajwal-ark:~/go/src/k8s.io/arktos# sudo ./hack/setup-dev-node.sh
The script is to help install prerequisites of Arktos development environment
on a fresh Linux installation.
It's been tested on Ubuntu 16.04 LTS and 18.04 LTS.
Update apt.
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu bionic-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
Install docker.
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker.io is already the newest version (20.10.7-0ubuntu5~18.04.3).
The following packages were automatically installed and are no longer required:
  accountsservice apport-symptoms bc command-not-found-data libaccountsservice0 libnuma1 python3-attr python3-automat python3-cl
  python3-debconf python3-debian python3-distro-info python3-gdbm python3-httplib2 python3-hyperlink python3-incremental python3
  python3-pyasn1-modules python3-requests-unixsocket python3-service-identity python3-systemd python3-twisted python3-twisted-bi
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Install make & gcc.
Reading package lists... Done
Building dependency tree
Reading state information... Done
make is already the newest version (4.1-9.1ubuntu1).
```

Also, please ensure the hostname and its ip address in /etc/hosts. For instance,

```
127.0.0.1 localhost
10.128.0.7 prajwal-ark
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
169.254.169.254 metadata.google.internal metadata
~
~
~
~
~
~
~
~
~
```

Replace the Arktos containerd:

cd $HOME/mizar

sudo bash replace-containerd.sh

```
root@prajwal-ark:~/go/src/k8s.io/arktos# cd $HOME/mizar
root@prajwal-ark:~/mizar# sudo bash replace-containerd.sh
root@prajwal-ark:~/mizar#
```

Before deploying Mizar, you will need first start up Arktos API server:

cd $HOME/go/src/k8s.io/arktos

./hack/arktos-up.sh

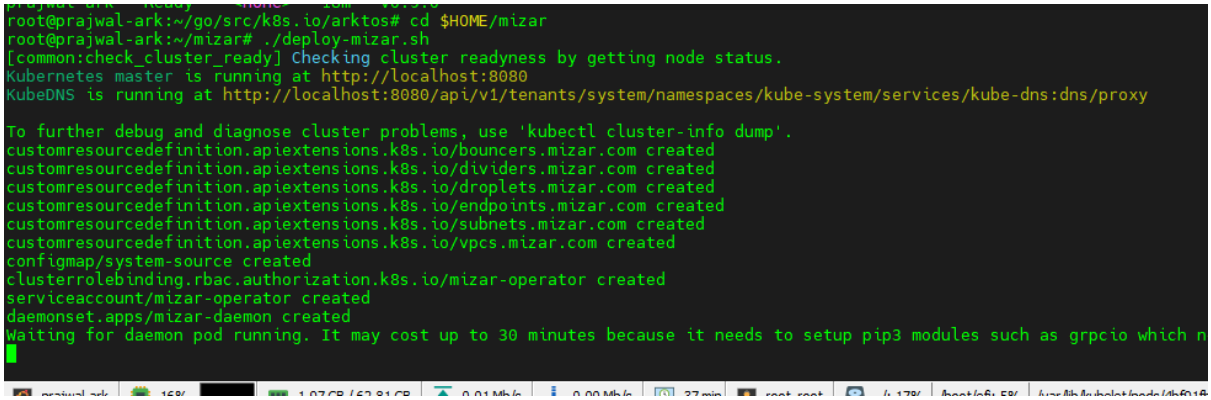
```
root@prajwal-ark:~/mizar# cd $HOME/go/src/k8s.io/arktos
root@prajwal-ark:~/go/src/k8s.io/arktos# ./hack/arktos-up.sh
DBG: Flannel CNI plugin will be installed AFTER cluster is up
DBG: effective feature gates AllAlpha=false,WorkloadInfoDefaulting=true,QPSDoubleGCController=true,QPSDoubleRSController=true,Mandator
DBG: effective disabling admission plugins
DBG: effective default network template file is /root/go/src/k8s.io/arktos/hack/testdata/default-flat-network.tmpl
DBG: kubelet arg RESOLV_CONF is /run/systemd/resolve/resolv.conf
WARNING : The kubelet is configured to not fail even if swap is enabled; production deployments should disable swap.
cni plugin is bridge; arktos will use bridge to provision pod network
Ensuring firewall to allow traffic forward by default
-P FORWARD DROP
-P FORWARD ACCEPT
Ensuring minimum cni plugin installation...
installing cni plugin binaries
./
./flannel
./ptp
./host-local
./firewall
./portmap
./tuning
./vlan
./host-device
./bandwidth
./sbr
./static
./dhcp
./ipvlan
./macvlan
./loopback
./bridge
2022-01-04 06:11:57 URL:https://objects.githubusercontent.com/github-production-release-asset-2e65be/84575398/53318d00-bf61-11e9-9a8f-
4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220104%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220104T061156Z&X-Amz-Expir
```

```
Local Kubernetes cluster is running. Press Ctrl-C to shut it down.

Logs:
  /tmp/kube-apiserver0.log
  /tmp/kube-controller-manager.log


  /tmp/kube-proxy.log
  /tmp/kube-scheduler.log
  /tmp/kubelet.log

To start using your cluster, you can open up another terminal/tab and run:

  export KUBECONFIG=/var/run/kubernetes/admin.kubeconfig
Or
  export KUBECONFIG=/var/run/kubernetes/adminN(N=0,1,...).kubeconfig

  cluster/kubectl.sh

Alternatively, you can write to the default kubeconfig:

  export KUBERNETES_PROVIDER=local

  cluster/kubectl.sh config set-cluster local --server=https://ip-172-31-25-250:6443 --certificate-authority=/var/run/kubernet
  cluster/kubectl.sh config set-credentials myself --client-key=/var/run/kubernetes/client-admin.key --client-certificate=/var
  cluster/kubectl.sh config set-context local --cluster=local --user=myself
  cluster/kubectl.sh config use-context local
  cluster/kubectl.sh
```

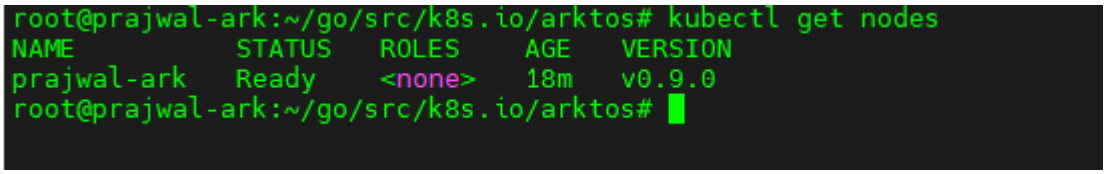
Deploy Mizar. Open a new terminal window, and run:

cd $HOME/mizar

./deploy-mizar.sh



Once your arktos server and Mizar are running. To verify, you can open a new terminal and run `kubectl get nodes`, you should see a node running with the name starts with "IP" followed by the private IP address of your lab machine.



You also want make sure the default kubernetes bridge network configuration file is deleted:

sudo ls /etc/cni/net.d

sudo rm /etc/cni/net.d/bridge.conf

Start Arktos network controller. From a new terminal window, run:

cd $HOME/go/src/k8s.io/arktos

./_output/local/bin/linux/amd64/arktos-network-controller --kubeconfig=/var/run/kubernetes/admin.kubeconfig --kube-apiserver-ip=xxx.xxx.xxx.xxx

where the `kube-apiserver-ip` is your lab machine's **private ip address**

Open another terminal:

Deploy test pods:

kubectl apply -f https://raw.githubusercontent.com/Click2Cloud-Centaurus/Documentation/main/test-yamls/test_pods.yaml

kubectl get pods -A

```
root@prajwal-ark:~/go/src/k8s.io/arktos# kubectl apply -f  https://raw.githubusercontent.com/Click2Cloud-Centaurus/Documentation/main/test-yaml
pod/netpod1 created
pod/netpod2 created
root@prajwal-ark:~/go/src/k8s.io/arktos# kubectl get pods -A
NAMESPACE     NAME                                HASHKEY              READY   STATUS              RESTARTS   AGE
default       mizar-daemon-slvdj                  4888288601711211922  1/1     Running             0          6m33s
default       mizar-operator-64d6f9fc8d-cwfpb     7291039560737290798  1/1     Running             0          3m33s
default       netpod1                             5375616082334601755  0/1     ContainerCreating   0          14s
default       netpod2                             3602922169560548014  0/1     ContainerCreating   0          14s
kube-system   coredns-default-d44bb7497-q9j79     505578764204138568   1/1     Running             0          25m
kube-system   kube-dns-554c5866fc-npd89           1640584905441628638  3/3     Running             0          25m
kube-system   virtlet-qxnxf                       2095782094958600669  3/3     Running             0          7m24s
root@prajwal-ark:~/go/src/k8s.io/arktos# kubectl get pods -A
NAMESPACE     NAME                                HASHKEY              READY   STATUS              RESTARTS   AGE
default       mizar-daemon-slvdj                  4888288601711211922  1/1     Running             0          6m42s
default       mizar-operator-64d6f9fc8d-cwfpb     7291039560737290798  1/1     Running             0          3m42s
default       netpod1                             5375616082334601755  0/1     ContainerCreating   0          23s
default       netpod2                             3602922169560548014  0/1     ContainerCreating   0          23s
kube-system   coredns-default-d44bb7497-q9j79     505578764204138568   1/1     Running             0          25m
kube-system   kube-dns-554c5866fc-npd89           1640584905441628638  3/3     Running             0          25m
kube-system   virtlet-qxnxf                       2095782094958600669  3/3     Running             0          7m33s
root@prajwal-ark:~/go/src/k8s.io/arktos# kubectl get pods -A
```

Pods are getting stuck in ***ContainerCreating*** state