

Deploy arktos with mizar cni test

The preferred OS is Ubuntu 18.04.

If you are using AWS, the recommended instance size is t2.xlarge and the storage size is 128GB or more.

Steps:

1. Check the kernel version:

Command:

```
uname -a
```

Output:

```
ubuntu@ip-172-31-31-176:~$ uname -a
Linux ip-172-31-31-176 5.4.0-1045-aws #47~18.04.1-Ubuntu SMP Tue Apr 13 15:58:14 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

Update the kernel if the kernel version is below 5.6.0-rc2

Command:

```
wget https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
```

```
sudo bash kernelupdate.sh
```

Output:

```
ubuntu@ip-172-31-31-176:~$ wget https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
--2021-10-19 12:04:18-- https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 791 [text/plain]
Saving to: 'kernelupdate.sh'

kernelupdate.sh      100%[=====] 791 --.-KB/s  in 0s

2021-10-19 12:04:19 (41.2 MB/s) - 'kernelupdate.sh' saved [791/791]

ubuntu@ip-172-31-31-176:~$
ubuntu@ip-172-31-31-176:~$ sudo bash kernelupdate.sh
--2021-10-19 12:04:19-- https://mizar.s3.amazonaws.com/linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb
Resolving mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)... 52.217.70.204
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|52.217.70.204|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7621020 (7.3M) []
Saving to: './linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb'

linux-headers-5.6.0-rc2_5.6.0-rc2-1 100%[=====] 7.27M 5.25MB/s  in 1.4s

2021-10-19 12:04:20 (5.25 MB/s) - './linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb' saved [7621020/7621020]

--2021-10-19 12:04:20-- https://mizar.s3.amazonaws.com/linux-5.6-rc2/linux-image-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64.deb
Resolving mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)... 52.217.36.156
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|52.217.36.156|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 857827912 (818M) [application/x-www-form-urlencoded]
Saving to: './linux-5.6-rc2/linux-image-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64.deb'

age-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64 95%[=====] 783.57M 8.78MB/s  eta 5s
```

```
ubuntu@ip-172-31-31-176:~$ uname -a
Linux ip-172-31-31-176 5.6.0-rc2 #1 SMP Tue Feb 25 18:54:05 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
```

2. Clone the Arktos repository and install the required dependencies:

Command:

```
git clone https://github.com/Click2Cloud-Centaurus/arktos.git
~/go/src/k8s.io/arktos

cd ~/go/src/k8s.io/arktos

git checkout cni-mizar

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

Output:

```
ubuntu@ip-172-31-31-176:~$ git clone https://github.com/Click2Cloud-Centaurus/arktos.git ~/go/src/k8s.io/arktos
Cloning into '/home/ubuntu/go/src/k8s.io/arktos' ...
remote: Enumerating objects: 61543, done.
remote: Counting objects: 100% (947/947), done.
remote: Compressing objects: 100% (544/544), done.
remote: Total 61543 (delta 531), reused 605 (delta 386), pack-reused 60596
Receiving objects: 100% (61543/61543), 221.37 MiB | 28.53 MiB/s, done.
Resolving deltas: 100% (37767/37767), done.
Checking out files: 100% (20761/20761), done.
ubuntu@ip-172-31-31-176:~$
ubuntu@ip-172-31-31-176:~$ cd ~/go/src/k8s.io/arktos
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ git checkout cni-mizar
Branch 'cni-mizar' set up to track remote branch 'cni-mizar' from 'origin'.
Switched to a new branch 'cni-mizar'
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ sudo bash ./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-west-2.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:4 http://us-west-2.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [8570 kB]
```

3 Command:

```
echo export PATH=$PATH:/usr/local/go/bin\ >> ~/.profile

echo cd \${HOME}/go/src/k8s.io/arktos >> ~/.profile

source ~/.profile
```

Output:

```
you can proceed to run Arktos-up.sh if you want to launch a single-node cluster.
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ echo export PATH=$PATH:/usr/local/go/bin\ >> ~/.profile
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ echo cd \${HOME}/go/src/k8s.io/arktos >> ~/.profile
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ source ~/.profile
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ CNIPLUGIN=mizar ./hack/arktos-up.sh
DBG: mizar CNI plugin will be installed AFTER cluster is up
DBG: effective feature gates AllAlpha=false,WorkloadInfoDefaulting=true,QPSDoubleGCCController=true,QPSDoubleRSController=true,MandatoryArktosNet
work=true
DBG: effective disabling admission plugins
DBG: effective default network template file is /home/ubuntu/go/src/k8s.io/arktos/hack/testdata/default-mizar-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.
WARNING: This script MAY be run as root for docker socket / iptables functionality; if failures occur, retry as root.
cni plugin is mizar; arktos will use mizar to provision pod network
Checking arktos containerd...
arktos containerd not found...
--2021-10-19 12:16:33-- https://github.com/containerd/containerd/releases/download/v1.4.2/containerd-1.4.2-linux-amd64.tar.gz
Resolving github.com (github.com)... 192.30.255.112
Connecting to github.com (github.com)[192.30.255.112]:443... connected.
HTTP request sent, awaiting response... 302 Found
```

4 Start Arktos cluster **with error**

Command:

```
CNIPLUGIN=mizar ./hack/arktos-up.sh
```

Output:

```
... [1019 12:17:02] building go targets for linux/amd64:
... /vendor/k8s.io/kube-openapi/cmd/openapi-gen
ln: failed to create symbolic link '/home/ubuntu/go/src/k8s.io/arktos/_output/bin': File exists
!!! [1019 12:17:02] call tree:
!!! [1019 12:17:02] 1: hack/make-rules/build.sh:28 kube::golang::place_bins(...)
Makefile.generated_files:295: recipe for target '_output/bin/defaulters-gen' failed
make[1]: *** [_output/bin/defaulters-gen] Error 1
make[1]: *** Waiting for unfinished jobs....
make[1]: Leaving directory '/home/ubuntu/go/src/k8s.io/arktos'
Makefile:560: recipe for target 'generated_files' failed
make: *** [generated_files] Error 2
make: Leaving directory '/home/ubuntu/go/src/k8s.io/arktos'
!!! Error in ./hack/arktos-up.sh:142
Error in ./hack/arktos-up.sh:142. 'make -j4 -C "${KUBE_ROOT}" WHAT="cmd/kubectl cmd/hyperkube cmd/kube-apiserver cmd/kube-controller-manager c
md/workload-controller-manager cmd/cloud-controller-manager cmd/kubelet cmd/kube-proxy cmd/kube-scheduler cmd/arktos-network-controller"' exited
with status 2
call stack:
```

Start Arktos cluster **without error**

Command:

```
CNIPLUGIN=mizar ./hack/arktos-up.sh
```

Output:

```
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/admin(N=0,1,...).kubeconfig
cluster/kubect1.sh

Alternatively, you can write to the default kubeconfig:
export KUBERNETES_PROVIDER=local

cluster/kubect1.sh config set-cluster local --server=https://ip-172-31-31-176:6443 --certificate-authority=/var/run/kubernetes/server-ca.crt
cluster/kubect1.sh config set-credentials myself --client-key=/var/run/kubernetes/client-admin.key --client-certificate=/var/run/kubernetes/cl
ient-admin.crt
cluster/kubect1.sh config set-context local --cluster=local --user=myself
cluster/kubect1.sh config use-context local
cluster/kubect1.sh
```

5. Verify mizar pods i.e. mizar-operator and mizar-daemon pods are in running state, for that run:

Command:

```
./cluster/kubect1.sh get pods
```

Output:

```
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ ./cluster/kubect1.sh get pods
NAME                                HASHKEY                                READY    STATUS    RESTARTS    AGE
mizar-daemon-pfpb8                  7084511080620833354                  1/1     Running    0           11m
mizar-operator-79bf96959-fk4hj      5618739021028152324                  1/1     Running    0           11m
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
```

deploy test pod

Command

```
./cluster/kubect1.sh apply -f https://raw.githubusercontent.com/Click2Cloud-Centaurus/Documentation/main/test-yamls/test\_pods.yaml
```

Output:

```
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ ./cluster/kubect1.sh apply -f https://raw.githubusercontent.com/Click2Cloud-Centaurus/Documentation/main/test-yamls/test\_pods.yaml
pod/netpod1 created
pod/netpod2 created
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
```

Command:

```
./cluster/kubect1.sh get pods -A
```

```
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ ./cluster/kubect1.sh get pods -A
NAMESPACE   NAME                                HASHKEY                                READY    STATUS    RESTARTS    AGE
default     mizar-daemon-pfpb8                  7084511080620833354                  1/1     Running    0           16m
default     mizar-operator-79bf96959-fk4hj      5618739021028152324                  1/1     Running    0           16m
default     netpod1                             4164102727579144295                  0/1     ContainerCreating    0           95s
default     netpod2                             5510621305704266765                  0/1     ContainerCreating    0           95s
kube-system coredns-default-5f7f97949d-2lmmd    2531655630568084883                  0/1     ContainerCreating    0           16m
kube-system kube-dns-7f4bf79dc-nz2qs       3275302932980669175                  0/3     ContainerCreating    0           16m
kube-system virtlet-6d5ks              6121837199323958649                  3/3     Running    0           11m
```

Comment:

Pods stuck in creating container state

get VPCs

Command

```
./cluster/kubect1.sh get vpc -A
```

Output:

```
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ ./cluster/kubect1.sh get vpc -A
NAMESPACE   NAME   IP           PREFIX   VNI   DIVIDERS   STATUS   CREATETIME           PROVISIONDELAY
default     vpc0   20.0.0.0     8        1      1          Init     2021-10-19T12:21:14.102260
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$
```

get subnet

command

`./cluster/kubectl.sh get subnet -A`

Output:

```
ubuntu@ip-172-31-31-176:~/go/src/k8s.io/arktos$ ./cluster/kubectl.sh get subnet -A
NAMESPACE   NAME   IP       PREFIX  VNI   VPC   STATUS   BOUNCERS   CREATETIME           PROVISIONDELAY
default     net0   20.0.0.0  8       1     vpc0   Init     1          2021-10-19T12:21:14.170816
```