



Creating Oracle Cloud Native Environment (OCNE) on Oracle Private Cloud Appliance

Oracle Private Cloud Appliance

August 20, 2024 | Version 2
Copyright © 2024, Oracle and/or its affiliates
Public

PURPOSE STATEMENT

This technical paper provides a methodology and workflow that solution architects and system administrators can follow to create an environment for Oracle Cloud Native Environment(OCNE) and Kubernetes on Private Cloud Appliance.

DISCLAIMER

This document in any form, software, or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

CONTENTS

Purpose Statement	1
Disclaimer	1
Introduction	3
Scope and content	3
Advantages of Oracle Private Cloud Appliance	3
Oracle Cloud Native Environment creation on Oracle Private cloud appliance	4
Reference architecture	4
Prerequisites	5
Configuration steps	5
Set proxy vars in etc bashrc	5
Set proxy vars	5
Apply OS Updates	6
Identify your OS	6
Configure yum repositories	7
Install chrony	7
Disable swap on all nodes	7
Configure Firewall on Operator Node	8
Configure Firewall on Control Plane	8
Configure Firewall on Worker Nodes	9
Setup bridge netfilter on the Control Plane	9
Configure subnet security list ingress rule	9
Installation steps	11
Install OCNE API server on the Operator Node.	11
Install OCNE Platform Agents on the Control Plane and Worker Nodes	11
Prepare firewall for internal load balancer on the Control Plane	12
Put X509 certificates in standard location on all nodes	13
Start platform agents on the control plane (i.e., the control nodes) and the worker nodes	13
Verify platform agents running	13
Create OCNE environment on the operator node	14
Create Kubernetes module	14
Add Ingress Rule to Subnet	15
Validate Kubernetes module on the operator node.	15
Install Kubernetes module on the operator node	15
See Kubernetes module report on the operator node.	16
Resources	16

INTRODUCTION

Private Cloud Appliance is uniquely compatible with Oracle Cloud Infrastructure (OCI) providing fast and efficient infrastructure for modern software and business applications. Private Cloud Appliance has the same infrastructure constructs (including APIs and SDKs) as OCI. This enables customers to adopt a “develop once and deploy anywhere—on-premises or on OCI” approach to rapidly design and develop high-performance applications and middleware.

SCOPE AND CONTENT

This technical paper provides a methodology and workflow that solution architects and system administrators can follow to create an environment for Oracle Cloud Native Environment(OCNE) and Kubernetes on Private Cloud Appliance.

ADVANTAGES OF ORACLE PRIVATE CLOUD APPLIANCE

Oracle Private Cloud Appliance is a rack-scale engineered system delivering Oracle Cloud Infrastructure compute, storage and networking constructs on-premises. It enables rapid deployment of applications, middleware and workloads that are cloud-compatible via automation in an OCI-like environment while being disconnected from the public cloud. Private Cloud Appliance is the ideal platform alongside Oracle Exadata and Oracle Database Appliance platforms offering lowest latency and highest performance between the application and database layers. Private Cloud Appliance is designed for customers who want a cloud-like development and deployment experience while also meeting data residency requirements.

ORACLE CLOUD NATIVE ENVIRONMENT CREATION ON PRIVATE CLOUD

APPLIANCE

Oracle Cloud Native Environment (OCNE) is a fully integrated suite for the development and management of cloud-native applications. Oracle Cloud Native Environment is a curated set of open-source projects that are based on open standards, specifications and APIs defined by the Open Container Initiative (OCI) and Cloud Native Computing Foundation (CNCF) that can be easily deployed, have been tested for interoperability and for which enterprise-grade support is offered. Oracle Cloud Native Environment delivers a simplified framework for installations, updates, upgrades and configuration of key features for orchestrating microservices.

REFERENCE ARCHITECTURE

OCNE manages a cluster and can provide services to support modules. For our case the primary module is Kubernetes. In the cluster there are three categories of nodes:

- Operator node hosts the OCNE API server and is the source of commands configuring all the nodes of the cluster
- Control nodes direct the Kubernetes cluster and can potentially provide HA in combination with a load balancer
- Worker nodes are where the work is done by pods deployed by Kubernetes

The architecture gives a layout of the various resources:

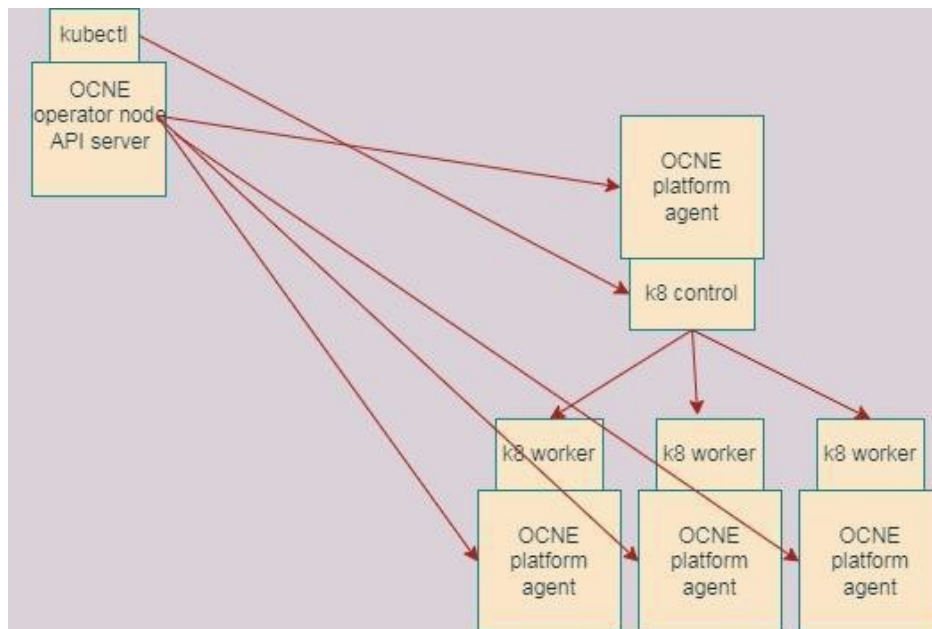


Fig 1: Oracle Cloud Native Environment Reference Architecture

PREREQUISITES

The environment assumes the provisioning of the following resources:

- Create Compartment in Tenancy.
- Create VCN in compartment with DNS label (Select Use DNS hostnames in this VCN)
- Create compute instances for worker nodes, Control nodes and the operator node.
- Apply OS patches to bring all the nodes up to date in terms of security and function.

CONFIGURATION STEPS

The environment assumes the provisioning of the following resources:

- Create Compartment in Tenancy.
- Create VCN in compartment with DNS label (Select Use DNS hostnames in this VCN)
- Create compute instances for 3 worker nodes, 1 control node and the operator node.

Apply OS patches to bring all the nodes up to date in terms of security and function.

Set proxy vars in etc bashrc

Set proxy vars in etc bashrc on all nodes so all future shell sessions will have the benefit of the proxy environment variables. Substitute your appropriate proxy server name and port for `http://<proxy-host>:<proxy-port>`.

Set proxy vars

```
if [ ! -f /etc/bashrc.bak ]; then
    cp -p /etc/bashrc /etc/bashrc.bak
else
    cp -p /etc/bashrc.bak /etc/bashrc
fi
zz=/tmp/ocne.setup.sh.on_host_to_be_setup
cat <<EOD > $zz
export http_proxy=http://<proxy-host>:<proxy-port>
export https_proxy=http://<proxy-host>:<proxy-port>
# substitute your domain name for
"dm.com" HOSTNAME=`hostname`
first_3_octets_of_LAN=`nslookup $HOSTNAME.dm.com | grep Address | tail -1 | sed -e 's/Address:
//' -e 's/\.[0-9]*$//`
export
no_proxy='localhost,127.0.0.1,.<proxyhost>,.oraclecorp.com,.oraclevcn.com,$first_3_octets_of_L
AN.0/24,.svc,/var/run/crio/crio.sock,10.96.0.0/12'
EOD
cat $zz >> /etc/bashrc
```

Apply OS Updates

Identify your OS

```
Identify OEL major release uname -a | sed -e  
's/.*\..el//' -e 's/^(.\).*/^1/'
```

If the result of this command is 8, then you are running OEL8. If you are running OEL7, use yum instead of dnf, and do not pass the additional argument --best --allowerasing. So, for example, if the instructions say to do:

```
yum  
yum --setopt=keepPrivate Cloud Applianceche=1 xyz
```

Instead, if you are running on OEL7, you should then do

```
dnf  
dnf --best --setopt=keepPrivate Cloud Applianceche=1 --allowerasing xyz
```

Next apply OS updates on all nodes after configuring yum and possibly dnf to be aware of your proxy server.

Patch OS

```
if [ ! -f /etc/yum.conf.bak ]; then
    cp /etc/yum.conf /etc/yum.conf.bak
else
    cp /etc/yum.conf.bak /etc/yum.conf
fi

echo proxy=http://<proxy-host>:<proxy-port> >> /etc/yum.conf

if [ -d /etc/dnf ]; then
    if [ ! -f /etc/dnf/dnf.conf.bak ]; then
        sudo cp /etc/dnf/dnf.conf /etc/dnf/dnf.conf.bak
    else
        sudo cp /etc/dnf/dnf.conf.bak /etc/dnf/dnf.conf
    fi

    cp /etc/dnf/dnf.conf.bak /tmp/dnf.conf

    echo proxy=http://<proxy-host>:<proxy-port>:80 >> /tmp/dnf.conf
```

```
sudo mv /tmp/dnf.conf /etc/dnf/dnf.conf

fi

dnf --best --setopt=keepPrivate Cloud Applianceche=1 --allowerase update -y

reboot
```

Configure yum repositories on all nodes

configure repos

if we are on OEL7, do the following:

```
sudo yum-config-manager --enable ol7_OCNE15 ol7_kvm_utils ol7_addons ol7_latest ol7_UEKR6
```

```
sudo yum-config-manager --disable ol7_OCNE14 ol7_OCNE13 ol7_OCNE12 ol7_OCNE11
ol7_OCNE ol7_developer
```

but if we are on OEL8, do the following:

```
sudo dnf -y install oracle-OCNE-release-el8
```

```
sudo yum config-manager --enable ol8_OCNE15 ol8_addons ol8_baseos_latest
ol8_appstream ol8_UEKR6
```

```
sudo yum config-manager --disable ol8_OCNE12 ol8_OCNE13 ol8_OCNE14 ol8_developer
```

Install chrony

#Install chrony on all nodes.

```
sudo dnf --best --setopt=keepPrivate Cloud Applianceche=1 --allowerase -y install chrony
```

```
sudo systemctl enable --now chronyd.service
```

Disable swap on all nodes

```
no swap

swapoff -a

if [ ! -f /etc/fstab.bak ]; then

    sudo cp /etc/fstab /etc/fstab.bak

fi

cat /etc/fstab.bak | sed '/[\t ]swap[\t ]/d' > /tmp/fstab

echo diff /etc/fstab.bak /tmp/fstab
```

```
diff      /etc/fstab.bak /tmp/fstab
sudo cp /tmp/fstab /etc/fstab
echo cat /etc/fstab
cat      /etc/fstab
```

Configure Firewall on Operator Node

```
operator node firewall
sudo firewall-cmd --add-port=8091/tcp --permanent
sudo firewall-cmd --reload
```

Configure Firewall on Control Plane

```
control plane firewall
sudo firewall-cmd --zone=trusted --add-interface=cni0 --permanent
sudo firewall-cmd --add-port=8090/tcp --permanent
sudo firewall-cmd --add-port=10250/tcp --permanent
sudo firewall-cmd --add-port=10255/tcp --permanent
sudo firewall-cmd --add-port=8472/udp --permanent
sudo firewall-cmd --add-port=6443/tcp --permanent
# HA ports:
sudo firewall-cmd --add-port=10251/tcp --permanent
sudo firewall-cmd --add-port=10252/tcp --permanent
sudo firewall-cmd --add-port=2379/tcp --permanent
sudo firewall-cmd --add-port=2380/tcp --permanent
sudo firewall-cmd --reload
```

Configure Firewall on Worker Nodes

```
worker node firewall

sudo firewall-cmd --zone=trusted --add-interface=cni0 --permanent

sudo firewall-cmd --add-port=8090/tcp --permanent

sudo firewall-cmd --add-port=10250/tcp --permanent

sudo firewall-cmd --add-port=10255/tcp --permanent

sudo firewall-cmd --add-port=8472/udp --permanent

sudo firewall-cmd --add-masquerade --permanent

sudo systemctl restart firewalld.service
```

Setup bridge netfilter on the Control Plane

```
bridge netfilter

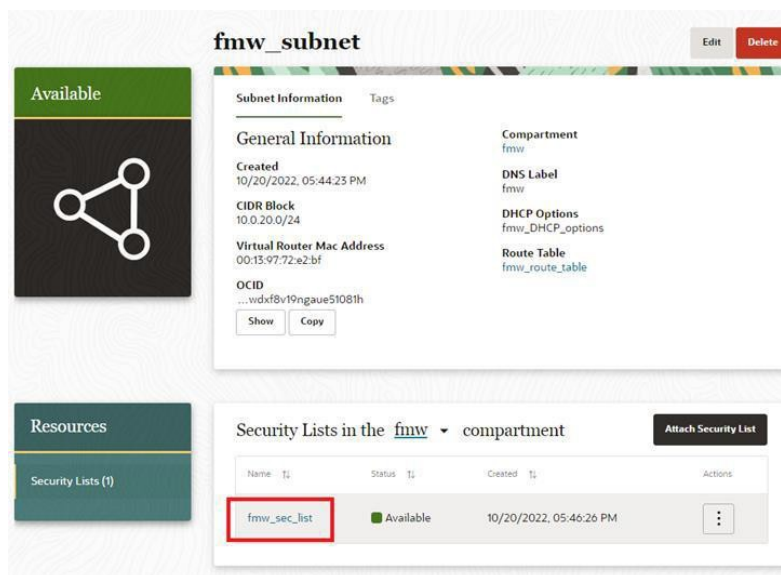
sudo modprobe br_netfilter

export PATH=/sbin:/usr/sbin:$PATH

if [ -n "`lsmod | grep br_netfilter`" ]; then
    sudo sh -c 'echo "br_netfilter" > /tmp/br_netfilter.conf'
    sudo sh -c 'echo "br_netfilter" > /etc/modules-load.d/br_netfilter.conf'
fi
```

Configure subnet security list ingress rule

To permit OCNE/Kubernetes traffic on the new cluster, you must adjust the subnet security list.



The screenshot displays the Oracle Cloud console interface for a subnet named 'fmw_subnet'. The page is divided into several sections:

- Available:** A green header with a network diagram icon.
- Subnet Information:** A tabbed section showing details for 'fmw_subnet'.
 - General Information:** Created (10/20/2022, 05:44:23 PM), CIDR Block (10.0.20.0/24), Virtual Router Mac Address (00:15:97:72:e2:bf), and OCID (...wdx8v19ngaue51081h).
 - Compartment:** fmw
 - DNS Label:** fmw
 - DHCP Options:** fmw_DHCP_options
 - Route Table:** fmw_route_table
- Resources:** A sidebar menu with 'Security Lists (1)' selected.
- Security Lists in the fmw compartment:** A table listing security lists. The 'fmw_sec_list' entry is highlighted with a red box. It has a status of 'Available' and was created on 10/20/2022 at 05:46:26 PM.

Fig 2: Subnet Security Lists

Drill down into the security list to add ingress rules:

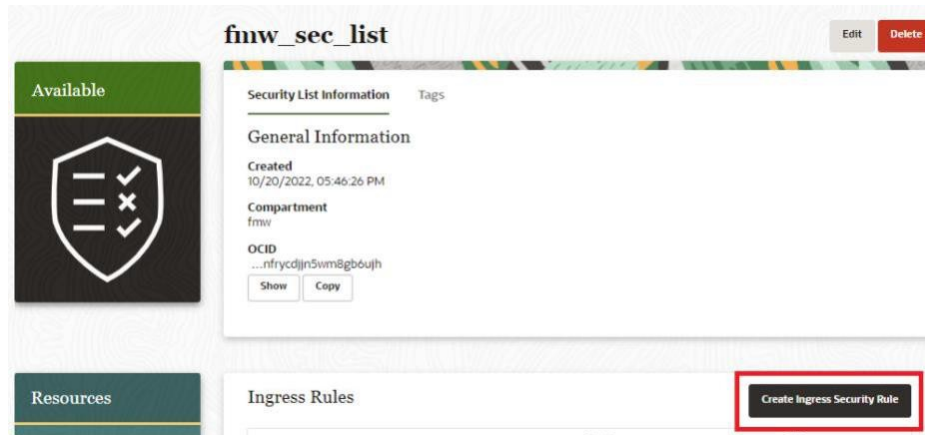


Fig 3: Ingress Rules within Subnet Security List

Add ingress rules to the security list for the subnet used for the nodes' compute instances to allow traffic on the necessary ports:

The image shows a 'Create Security List Rule' dialog box. It has a 'Stateless' checkbox which is unchecked. Below it is the 'Ingress CIDR' field with the value '0.0.0.0/0'. The 'IP Protocol' is set to 'TCP'. There are two optional fields: 'Source Port Range' with the example '80, 20-22' and 'Destination Port Range' with the value '2379-2380'. The 'Description' field contains the text 'OCNE' and has a note 'Enter 1 to 255 characters.' at the bottom. At the bottom right of the dialog are two buttons: 'Create Security List Rule' and 'Cancel'.

Fig 4: Security List Rule

Allow traffic with ports:

- 2379-2380
- 6443-6444
- 8090-8091
- 8472
- 10250-10252
- 10255

INSTALLATION STEPS

Install OCNE API server on the Operator Node

API server

```
if ! systemctl status OCNE-api-server.service | grep 'Loaded: loaded'; then
    echo "No platform OCNE-api-server.service seen on `hostname`, so the way is clear
to install it..."
    pm_action=install
else
    sudo systemctl stop OCNE-api-server.service
    pm_action=reinstall
fi

sudo dnf --best --setopt=keepPrivate Cloud Applianceche=1 --alloweraseing $pm_action -y
OCNEctl OCNE-api-server OCNE-utils
sudo systemctl enable OCNE-api-server.service
```

Install OCNE Platform Agents on the Control Plane and Worker Nodes

platform agents

```
if ! systemctl status OCNE-agent.service | grep 'Loaded: loaded'; then
    echo "No platform OCNE-agent.service seen on `hostname`, so the way is clear to install it..."
    pm_action=install
else
    sudo systemctl stop OCNE-agent.service pm_action=reinstall
fi

sudo dnf --best --setopt=keepPrivate Cloud Applianceche=1 --alloweraseing $pm_action -y OCNE-agent OCNE-utils
sudo systemctl enable OCNE-agent.service

sudo mkdir -p /etc/systemd/system/crio.service.d
cat <<EOD> /tmp/t
[Service]
Environment="HTTP_PROXY=http://<proxy-host>:<proxy-port>"
Environment="HTTPS_PROXY=http://<proxy-host>:<proxy-port>"
Environment="NO_PROXY=localhost,127.0.0.1, <proxy-
host>us.oracle.com,oraclecorp.com,oraclevcn.com,10.0.1.0/24,10.0.0.0/24,svc,/var/run/crio/c_rio.sock,10.96.0.0/12"
EOD
sudo mv /tmp/t/etc/systemd/system/crio.service.d/proxy.conf
if ! systemctl status docker.service 2>&1 | grep -l 'could not be found.' > /dev/null 2>&1; then
```

```

        sudo systemctl disable --now docker.service
fi
if ! systemctl status containerd.service 2>&1 | grep -l 'could not be found.' > /dev/null
2>&1; then
        sudo systemctl disable --now containerd.service
fi

```

Prepare firewall for internal load balancer on the Control Plane

control plane firewall for HA

```

sudo firewall-cmd --add-port=6444/tcp
sudo firewall-cmd --add-port=6444/tcp --permanent
sudo firewall-cmd --add-protocol=vrrp
sudo firewall-cmd --add-protocol=vrrp --permanent
scp /path/to/your/id_rsa ocneoperator.dm.com:/home/opc/.ssh/id_rsa

```

Generate X509 certificates on the operator node

```

generate certificates
if [ -f /home/opc/.OCNE/certificates/127.0.0.1:8091/node.cert ]; then # should we call
        #       OCNEctl --api-server 127.0.0.1:8091 environment report --environment-name myenvironment
        # this file is searched for. Skip the call and just check for the file; if we see it, attempt to delete existing env:
        echo signs of myenvironment seen, will try to delete it...
        OCNEctl --api-server 127.0.0.1:8091 environment delete --environment- name myenvironment
else
        echo "no environment seen, as expected. But will attempt to delete anyway to account for the case that the
        envis not reflected in that file check above"
        OCNEctl --api-server 127.0.0.1:8091 environment delete --environment- name myenvironment
fi
cd /etc/OCNE
if systemctl status OCNE-api-server.service; then
        echo running OCNE-api-server.service seen, stopping it... sudo systemctl stop
        OCNE-api-server.service
else
        echo no running OCNE-api-server.service seen, as expected
fi

```

```
sudo ./gen-certs-helper.sh --cert-request-organization-unit "Paper Sales" --cert-request-organization "Dunder Mifflin" --cert-request-locality "Scranton" --cert-request-state "WA" -- cert-request-country "US" --cert-request-common-name "dm.com" --nodes ocneoperator.dm.com,ocnecontrol.dm.com,ocneworker.dm.com,ocneworker2.dm.com,ocneworker3.dm.com
```

Distribute X509 certificates to all nodes from the operator node

distributed certificates

```
sudo chown -R opc:opc /etc/OCNE/configs/certificates/production
sudo chown -R opc:opc /etc/OCNE/configs/certificates/tmp-OCNE
```

```
# copies certificates to all nodes: bash -ex
/etc/OCNE/configs/certificates/OCNE-transfer-certs.sh ls -l
/etc/OCNE/configs/certificates/production
```

Start OCNE API server on the operator node.

start API server

```
sudo bash -x /etc/OCNE/bootstrap-OCNE.sh --secret-manager-type file --OCNE-node-cert-path
/etc/OCNE/configs/certificates/production/node.cert --OCNE-ca-path
/etc/OCNE/configs/certificates/production/ca.cert --OCNE-node-key-path
/etc/OCNE/configs/certificates/production/node.key --OCNE-component api-server systemctl
status OCNE-api-server.service
```

Put X509 certificates in standard location on all nodes

copy certificates to standard place

cp certs to standard location:

```
sudo rm -f /etc/OCNE/certificates/* > /dev/null 2>&1
```

```
sudo cp /etc/OCNE/configs/certificates/production/* /etc/OCNE/certificates
```

Start platform agents on the control plane (i.e., the control nodes) and the worker nodes

start platform agents

```
sudo /etc/OCNE/bootstrap-OCNE.sh --secret-manager-type file --OCNE-node-cert-
path /etc/OCNE/configs/certificates/production/node.cert --OCNE-ca-path
/etc/OCNE/configs/certificates/production/ca.cert --OCNE-node-key-path
/etc/OCNE/configs/certificates/production/node.key --OCNE-component agent
systemctl status OCNE-agent.service
```

Verify platform agents running

Verify platform agents running on the control plane (i.e., the control nodes) and the worker nodes.

```
verify platform agents up
ps auxww | grep /usr/libexec/OCNE-agent | grep -v grep > /tmp/kk.26597
if [ -s /tmp/kk.26597 ]; then
```



```
echo "OK /usr/libexec/OCNE-agent running on `hostname`"
if [ -n "" ]; then
    cat /tmp/kk.26597
fi
else
    echo "FAIL /usr/libexec/OCNE-agent NOT running on `hostname`"
fi
```

Create OCNE environment on the operator node

If this is not a freshly allocated cluster, and an OCNE environment had previously been created on the operator node, then delete that now:

```
optionally remove previous environment
OCNEctl --api-server 127.0.0.1:8091 environment delete --environment-name myenvironment

create environment
sudo chown -R opc:opc /etc/OCNE/configs
OCNEctl --api-server 127.0.0.1:8091 environment create --environment-name myenvironment -
- update-config --secret-manager-type file --OCNE-node-cert-path
/etc/OCNE/configs/certificates/production/node.cert --OCNE-ca-path
/etc/OCNE/configs/certificates/production/ca.cert --OCNE-node-key-path
/etc/OCNE/configs/certificates/production/node.key
```

Create Kubernetes module

If this is not a freshly allocated cluster, and an OCNE Kubernetes module had previously been created on the operator node, then remove that module now:

```
optionally remove previous k8s module
OCNEctl module uninstall --environment-name myenvironment --module kubernetes --name mycluster
```

Create Kubernetes module on the operator node. For this command we need to determine what network interface to use. In this example code, we run `ifconfig` and pick the first non-loop network interface. So, for example, if `ifconfig` produces the following output:

```
ifconfig output
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>          mtu 9000
    inet 172.16.8.117          netmask 255.255.252.0      broadcast 172.16.11.255
    inet6 fe80::213:97ff:fe3c:8b34          prefixlen 64  scopeid 0x20<link>
    ether 00:13:97:3c:8b:34          txqueuelen 1000  (Ethernet)
    RX packets 2284          bytes 392817  (383.6  KiB)
        RXerrors0  dropped 0  overruns 0  frame 0
    TX packets 1335          bytes 179539  (175.3  KiB)
        TXerrors0  dropped 0  overruns0  carrier0          collisions 0
```



```
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 6 bytes 416 (416.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6 bytes 416 (416.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions
```

Then the sed expression on the second line below will end up setting iface to be 'ens3':

```
create k8 module
sudo chown -R opc:opc /etc/OCNE/configs/certificates/restrict_external_ip/production

iface=`ifconfig | sed -n -e '/^ /d' -e /LOOPBACK/d -e 's/:.*//p'``

# substitute the IP of your control node for CONTROL_NODE_IP below:
OCNEctl module create --environment-name myenvironment --module kubernetes --name mycluster --
container-registry container-registry.oracle.com/OCNE --virtual-ip CONTROL_NODE_IP --master-
nodes ocnecontrol.dm.com:8090 --worker-nodes
ocneworker.dm.com:8090,ocneworker2.dm.com:8090,ocneworker3.dm.com:8090 --selinux enforcing --
restrict-service-externalip-ca-cert
/etc/OCNE/configs/certificates/restrict_external_ip/production/production/ca.cert --restrict-
service-externalip-tls-cert
/etc/OCNE/configs/certificates/restrict_external_ip/production/production/node.cert --
restrict-service-externalip-tls-key
/etc/OCNE/configs/certificates/restrict_external_ip/production/production/node.key --pod-
network-iface $iface
```

Add Ingress Rule to Subnet

- In the PRIVATE CLOUD APPLIANCE UI, go to Dashboard / Virtual Cloud Networks / your_VCN / your_security_list
- Add a rule for source 0.0.0.0/0 of type tcp allowing a destination port range 2379-10255.

Validate Kubernetes module on the operator node.

```
validate k8
OCNEctl module validate --environment-name myenvironment --name mycluster
```

Install Kubernetes module on the operator node

```
install k8
OCNEctl module install --environment-name myenvironment --name mycluster
```

See Kubernetes module report on the operator node.

```
report on k8  
OCNEctl module report --environment-name myenvironment --name mycluster
```

Show Kubernetes nodes on the operator node

Note: To use kubectl within your cluster,:

```
run kubectl  
kubectl get nodes -o wide
```

RESOURCES

For additional information please refer to:

- [Oracle Linux 7.9 Setup](#)
- [OCNE Setup Documentation](#)
- [Oracle Linux 8 Setup](#)

CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com.

Outside North America, find your local office at oracle.com/contact.

 blogs.oracle.com

 facebook.com/oracle

 twitter.com/oracle

Copyright © 2024, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0120

Creating Oracle Cloud Native Environment (OCNE) on Oracle Private Cloud Appliance
Feb-24



