ORACLE

Oracle Private Cloud Appliance X9-2

Automatic Virtual IP Failover in Cluster Deployment

August 25, 2022 | Version 1.05

Public

PURPOSE STATEMENT

This document provides a guide to provision Automatic Virtual IP Failover to ensure continued operations in case of unexpected failure. This functionality is available with the latest 3.0.1 SW posted on PCA-X9 ULN channel Release b697160.

DISCLAIMER

CONTENTS

INTRODUCTION

Oracle Private Cloud Appliance X9-2 supports Secondary Private IP provisioning with the release of the latest PCA-X9 SW 3.0.1. This Technical Brief provides the steps required to provision Secondary IP and ensure Automatic Virtual IP Failover in case of unexpected instance failure.

Oracle PCA X9-2 allows users to assign a secondary Private IP after instance is launched. The Secondary Private IP must be from the CIDR of the VNIC's subnet and can also be moved to another instance's VNIC in the same subnet. Secondary Private IP can be assigned to another instance, which a member of a cluster, if the primary instance encounters a failure. The assigned Public IP moves with the Private IP, and hence Automatic Virtual IP failover provides the capability to keep the application running in case of unexpected instance failure in a cluster environment running on PCA X9-2.

SCOPE AND CONTENT

This document provides a methodology and workflow that solution architects and system administrators can use to ensure Automatic Virtual IP Failover on a corosync pacemaker cluster. Some of the key aspects covered in this document are:

- Steps for Compute Instance Deployments
- Installing OCI CLI (Using the OCI CLI)
- Installing Corosync Pacemaker
- Testing Secondary IP Failover
- Useful Secondary Private IP Commands

ADVANTAGES OF ORACLE PRIVATE CLOUD APPLIANCE

The Oracle Private Cloud Appliance (PCA) is an Oracle Engineered System designed for implementing the application and middleware tiers. PCA is an integrated hardware and software system that reduces infrastructure complexity and deployment time for virtualized workloads in private clouds. It is a complete platform for a wide range of application types and workloads, with built-in management, compute, storage, and networking resources. PCA provides excellent performance and other system properties for a broad range of applications.

PCA X9-2 is the latest member of the Oracle Private Cloud Appliance product family. PCA X9-2 provides cloud and administrative services for general purpose IaaS (Infrastructure as a Service) for a broad range of workloads including modernized Cloud Native applications. It provides an excellent foundation to layer PaaS (Platform as a Service) and SaaS (Software as a Service) solutions on top of the infrastructure. Under the covers, it makes use of modern micro-services architecture, Kubernetes, and related technologies, for a refreshed future-proofed software stack.

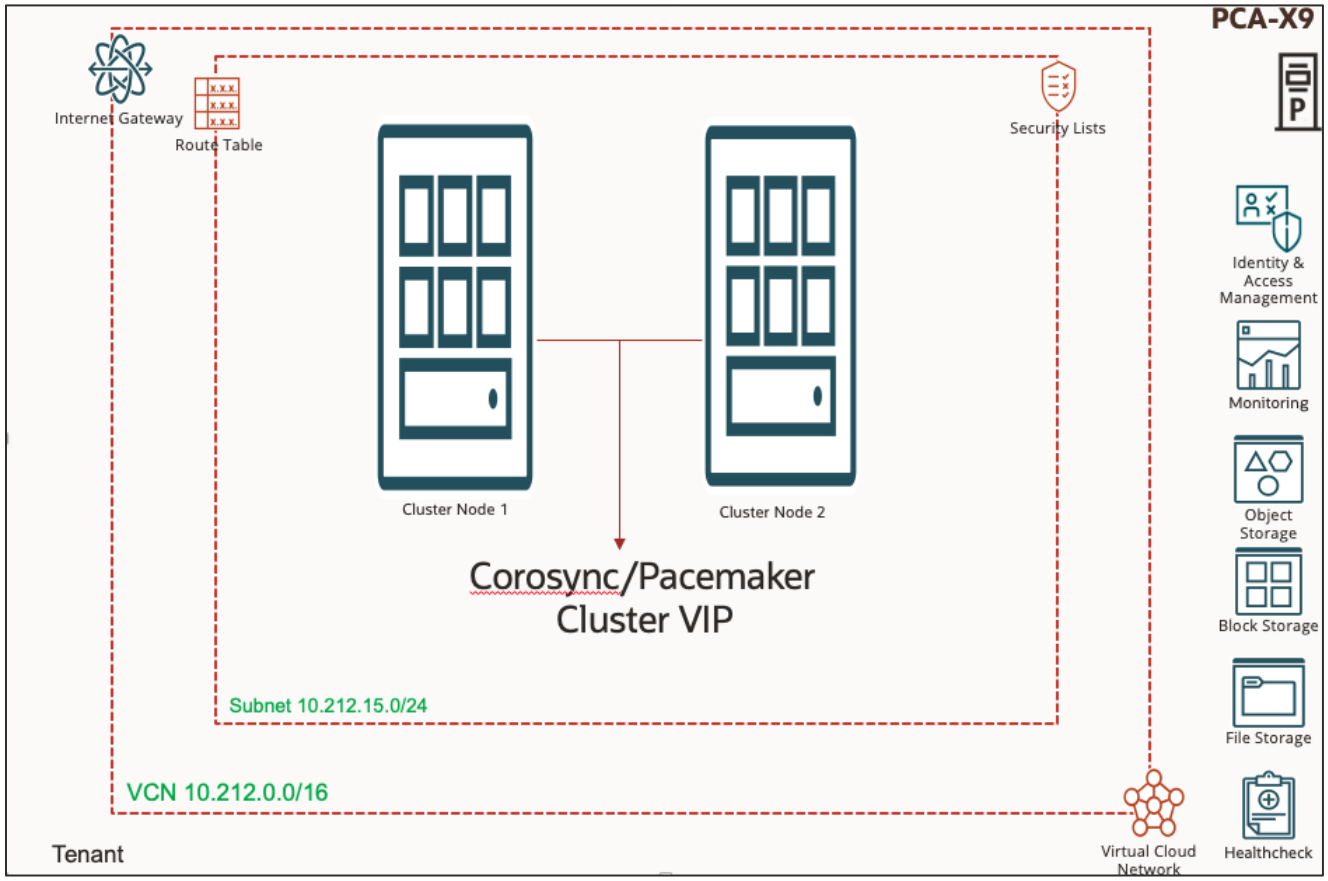
The key new feature of PCA X9-2 compared to previous PCA versions is that it delivers private cloud infrastructure and architecture consistent with Oracle Cloud Infrastructure (OCI). Core IaaS services use the same APIs, methods, tools, and interfaces familiar to OCI users, delivered on a modernized infrastructure, capable of high levels of scale and performance.

PCA X9-2 brings services compatible with Oracle Cloud Infrastructure's (OCI) to an on-premises implementation at rack scale, making workloads, user experience, tool sets and skills portability between private and public clouds. PCA X9-2 can be paired with Oracle Exadata to create an ideal infrastructure for scalable, multi-tier applications. Customers preferring or requiring an on-premises solution, can realize the operational benefits of public cloud deployments, using the Oracle Private Cloud Appliance X9-2.

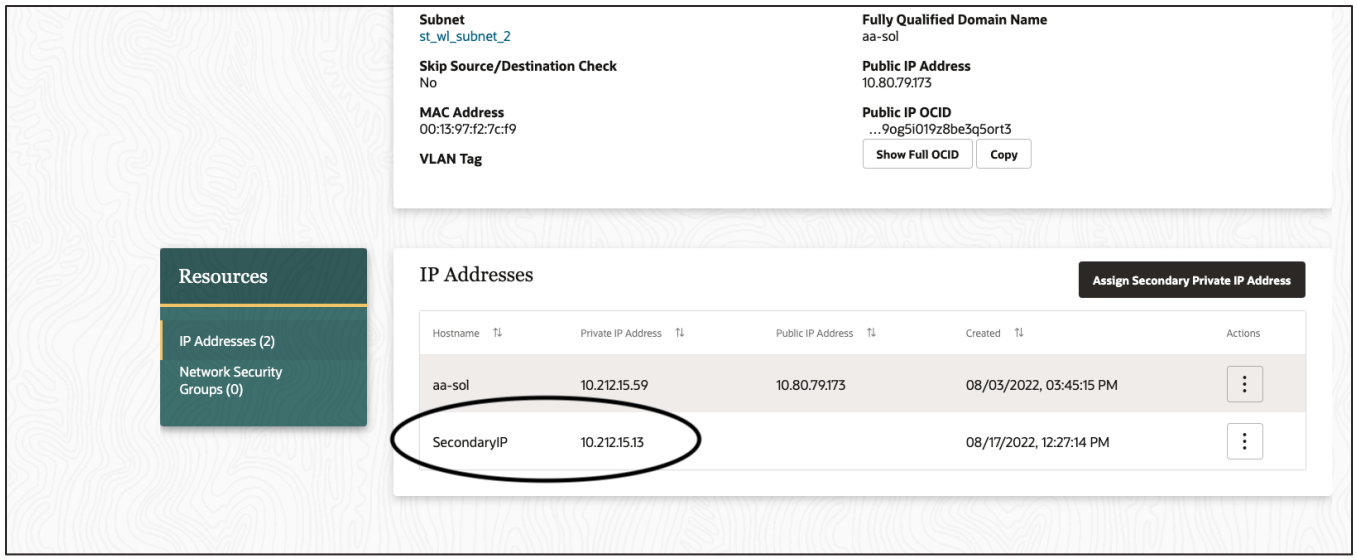
SETUP REQUIREMENTS

- PCA OL7.9 or OL8.0 Platform images
- VCN/Subnet/Sec list defined
- Internet Gateway for instance external access
- 2xInstances with min shape VM.PCAStandard1.4
- corosync pacemaker packages installed on both instances
    - Corosync and Pacemaker
- OCI CLI Installed on both instances (Using the OCI CLI)

REFERENCE ARCHITECTURE



Reference Architecture for PCA X9 Automatic Virtual IP Failover on a corosync pacemaker cluster



Console Showcasing Secondary IP Assignment

QUICK INSTALL STEPS ON CLUSTERED INSTANCES

1. Package Install

- yum install corosync pacemaker pcs
- yum install python36-oci-cli

Note: Setup proxy as per the lab requirements to reach to yum repos

2. Take the backup of heartbeat configuration of the clusters

- sudo cp /usr/lib/ocf/resource.d/heartbeat/IPaddr2 /usr/lib/ocf/resource.d/heartbeat/IPaddr2.bck

3. firewalld defined as per customer security needs for cluster

4. Update the /etc/hosts to Add node's IP and host name information for all cluster nodes to setup a local DNS system

- Example:
  - <Instance-Node1 Private IP> node1-name
  - <Instance-Node2 Private IP> node2-name

5. Create a user and set password:

- sudo passwd hacluster

6. Start and Enable the Cluster Services

- sudo systemctl start pcsd
- sudo systemctl enable pacemaker
- sudo systemctl enable corosync
- sudo systemctl enable pcsd

7. Verify cluster authentication

- sudo pcs cluster auth <node1-name> <node2-name>
  - <node1-name> : Cluster Node1/Hostname
  - <node2-name> : Cluster Node1/Hostname
- Example:
  - sudo pcs cluster auth a-node1 a-node2

- Username: hacluster
- Password:
- a-node1: Authorized
- a-node2: Authorized

8. Cluster Setup:

- sudo pcs cluster setup --name <clustername> <node1-name> <node2-name>
  - <clustername> : Cluster Name
  - <node1-name> : Cluster Node1/Hostname
  - <node2-name> : Cluster Node1/Hostname
- Example:
  - sudo pcs cluster auth a-node1 a-node2
    - Username: hacluster
    - Password:
    - a-node1: Authorized
    - a-node2: Authorized
    - [root@a-node1 opc] # sudo pcs cluster setup HACluster a-node1 a-node2
    - Error: A cluster name (--name <name>) is required to setup a cluster
    - [root@a-node1 opc] # sudo pcs cluster setup --name HACluster a-node1 a-node2
    - Destroying cluster on nodes: a-node1, a-node2...
    - a-node1: Stopping Cluster (pacemaker)...
    - a-node2: Stopping Cluster (pacemaker)...
    - a-node2: Successfully destroyed cluster
    - a-node1: Successfully destroyed cluster
    - Sending 'pacemaker_remote authkey' to 'a-node1', 'a-node2'
    - a-node1: successful distribution of the file 'pacemaker_remote authkey'
    - a-node2: successful distribution of the file 'pacemaker_remote authkey'
    - Sending cluster config files to the nodes...
    - a-node1: Succeeded
    - a-node2: Succeeded
    - Synchronizing pcsd certificates on nodes a-node1, a-node2...
    - a-node1: Success
    - a-node2: Success
    - Restarting pcsd on the nodes in order to reload the certificates...
    - a-node1: Success
    - a-node2: Success

9. Start cluster for all the cluster nodes from any of the cluster nodes

- sudo pcs cluster start --name clustername –all
- Example:
  - sudo pcs cluster start --name HACluster --all
  - a-node1: Starting Cluster (corosync)...
  - a-node2: Starting Cluster (corosync)...
  - a-node2: Starting Cluster (pacemaker)...
  - a-node1: Starting Cluster (pacemaker)...

10. Set pacemaker property

- sudo pcs property set stonith-enabled=false
- sudo pcs property set no-quorum-policy=ignore

11. Verify the running cluster status :

- sudo pcs cluster status
- Example:
  - sudo pcs cluster status
  - Cluster Status:

- Stack: corosync
- Current DC: a-node2 (version 1.1.23-1.0.1.el7_9.1-9acf116022) - partition with quorum
- Last updated: Fri Aug 19 03:07:25 2022
- Last change: Fri Aug 19 03:06:13 2022 by root via cibadmin on a-node1
- 2 nodes configured
- 0 resource instances configured
- PCSD Status:
- a-node1: Online
- a-node2: Online

12. Set the OCI Config as per your PCA settings on all the Cluster nodes

- Reference OCI: Using the OCI CLI
  - Note: Setting up config profile is very important to connect to PCA
    - Example: i.e. /root/.oci/config
      - [DEFAULT]
      - user=<User-ocid1>
      - fingerprint=<fingerprint>
      - key_file=<Key-Location>
      - tenancy=<Tenancy ocid1>
      - region=<PCA FQDN>
    - Get the cert for PCA X9
    - Ref: Obtaining the Certificate Authority Bundle

DEFINE THE HEARTBEAT SETTINGS ON ALL CLUSTER NODES AND INTEGRATE WITH PCA X9 INSTANCES FOR VIP FAILOVER

Note:

- **sudo sed -i '633i\export OCI_CLI_CERT_BUNDLE=/root/.oci/ca-chain.cert.pem\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2**
- Above is a must for PCA
- The other option is passing the pca certificate with assign-private-ip command itself i.e. --cert-bundle <Certification Location for PCA> option

Ref: Obtaining the Certificate Authority Bundle

Below sed update example is for OL7 and for other operating systems like OL8 or Centos look for add_interface () function line number in IPaddr2 and change line number accordingly for updating Linux HA IPaddr2 resource entries

```
sudo sed -i '628i\server="`hostname -s`"\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '629i\node1vnic="ocid1.vnic.pca.NODE1-vNIC-OCID"\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '630i\node2vnic="ocid1.vnic.pca.NODE2-vNIC-OCID"\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '631i\vnicip="10.212.15.13"\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '632i\export LC_ALL=C.UTF-8\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '633i\export LANG=C.UTF-8\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '633i\export OCI_CLI_CERT_BUNDLE=/root/.oci/ca-chain.cert.pem\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '634i\touch /tmp/error.log\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '635i\##### OCI/IPaddr Integration\' /usr/lib/ocf/resource.d/heartbeat/IPaddr2
```

```
sudo sed -i '636i\        if [ $server = "node1" ]; then\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '637i\                oci network vnic assign-private-ip --unassign-if-already-
assigned --vnic-id $node1vnic  --ip-address $vnicip >/tmp/error.log 2>&1\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '638i\        else \' /usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '639i\               oci network vnic assign-private-ip --unassign-if-already-
assigned --vnic-id $node2vnic  --ip-address $vnicip >/tmp/error.log 2>&1\'
/usr/lib/ocf/resource.d/heartbeat/IPaddr2

sudo sed -i '640i\        fi \' /usr/lib/ocf/resource.d/heartbeat/IPaddr2
```

Changes needed in IPaddr2 code

- Replace ocid1.vnic.pca.NODE1-vNIC-OCID and ocid1.vnic.pca.NODE2-vNIC-OCID by your own OCI vNICs OCIDs.
- Replace "node1" and "node2" hostname entries by your own Clusternodes hostname ones.
- OCI_CLI_CERT_BUNDLE : Define the CERT bundle location for PCA
- vnicip : define the vnicip as per your configuration/subnet/vcn and make sure this is uniq ip and not allocated to any other vnic

Setup Cluster Resource

- pcs resource create <Cluster-Resource-Name> ocf:heartbeat:IPaddr2 ip=10.212.15.13 cidr_netmask=24 op monitor interval=20
  - NOTE:
    - The cidr_netmask=24 in the Pacemaker command is dependent on the subnet size being /24
    - The ip=10.212.15.13 is the secondary private ip
- Example:
  - pcs status
  - Cluster name: HACluster
  - Stack: corosync
  - Current DC: a-node2 (version 1.1.23-1.0.1.el7_9.1-9acf116022) - partition with quorum
  - Last updated: Fri Aug 19 03:34:51 2022
  - Last change: Fri Aug 19 03:34:44 2022 by root via cibadmin on a-node1
  - 2 nodes configured
  - 1 resource instance configured
  - Online: [a-node1 a-node2]
  - Full list of resources:
  - HAFailover (ocf::heartbeat:IPaddr2): Started a-node1
  - Daemon Status:
  - corosync: active/disabled
  - pacemaker: active/disabled
  - pcsd: active/enabled

Testing Failover of the Secondary IP

- sudo pcs resource move <cluster-name> <node2-name>
  - <clustername> : Custer Name
  - <node2-name> : Cluster Node1/Hostname
- Example:
  - sudo pcs resource move HAFailover a-node2

Verify Successful Failover

Resources should be started on node2

- # pcs status

USEFUL SECONDARY PRIVATE IP COMMANDS

Assign Private IP:

- oci network vnic assign-private-ip --unassign-if-already-assigned --vnic-id <vnic-id> --ip-address <private-ip>

Assign Public IP to Private Secondary IP:

- oci network public-ip create --private-ip-id <privateip ocid> --lifetime RESERVED

List Public IPs:

- oci network public-ip list --scope AVAILABILITY_DOMAIN

Unassign Public IP:

- oci network public-ip update --public-ip-id <ocid1.publicip> --private-ip-id ""

REFERENCES

See these reference documents for additional information:

- Automatic Virtual IP Failover on Oracle Cloud Infrastructure
- How to configure Automatic Virtual IP
- Oracle Private Cloud Appliance Release Notes

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

Oracle Private Cloud Appliance X9-2
October 2222
Author: Amardeep Dhillon