ORACLE

Migrating Mid-tier Applications from Oracle Exalogic to Oracle Private Cloud Appliance X9-2

How to migrate workloads from Exalogic to PCA X9-2

PURPOSE STATEMENT

This document provides a guide to exporting virtual machine workloads from Oracle Exalogic to Oracle Private Cloud Appliance X9-2 (PCA X9-2) running software version

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.

TABLE OF CONTENTS

INTRODUCTION

This Technical Brief provides best practices and guidance for importing virtual machines from Oracle Exalogic to Oracle Private Cloud Appliance X9-2 (abbreviated as "PCA" in this document except when making distinction with previous PCA versions) in "lift and shift" migration.

The Oracle Private Cloud Appliance (PCA) has emerged as the premier platform for WebLogic, Fusion Middleware and other application tier software, often in conjunction with Oracle Exadata - the premier database platform. Many of those applications have been deployed on Oracle Exalogic or commodity x86 servers and are now being deployed on PCA for increased performance, scale, and manageability.

This white paper discusses advantages of moving those applications to PCA, describes how to migrate them, with emphasis on Exalogic workloads, and describes deployment methods and best practices.

SCOPE AND CONTENTS

This document provides a methodology and workflow that architects and system administrators can use to migrate virtual machine workloads to PCA X9-2, including:

- Planning steps applicable for any migration to PCA
- Details specific to migrating middleware environments from Exalogic
- The operating system is assumed to be Oracle Linux, running on a virtualized Exalogic environment under Oracle VM

The material presented here uses the process described in the technical paper "Oracle Private Cloud X9-2 Workload Import" at https://www.oracle.com/a/otn/docs/oracle-private-cloud-appliance-x9-2-workload-import.pdf specifically focused on the use case and considerations for migration from Exalogic.

ADVANTAGES OF PRIVATE CLOUD APPLIANCE

The Oracle Private Cloud Appliance (PCA) is an Oracle Engineered System designed for application tier. 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 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 portable between private and public clouds. PCA can be paired with Oracle Exadata to create an ideal infrastructure for scalable, multi-tier applications. Customers using the Oracle Private

Cloud Appliance X9-2 realize the operational benefits of cloud deployments for customers preferring or requiring an on-premises solution

DEFINITIONS

This document uses the following terms:

- The **source** system is the platform where the virtual machine is currently running, and the **target** is the PCA system it will be moved to.
- An **instance** is a virtual machine on the PCA. An instance has a **lifecycle**: it can be created, started ("launched"), stopped, and terminated (removed from the PCA system). Oracle Cloud Infrastructure (OCI) uses the term instance instead of virtual machine because an instance could potentially be on bare metal. PCA conforms to the same definition.
- Every instance has a **shape**, which describes its CPU, memory, network and disk configuration. PCA has a list of standard shapes shown below in Appendix 1.
- An **image** is the template of a virtual disk, containing the operating system and pre-installed applications, plus descriptive metadata.
- An instance is created by creating a **boot volume** from an image, and immediately launched. Other disks belonging to the instance are called **block volumes** and are created after the instance is launched.
- PCA provides **platform images** for Oracle Linux 7, Oracle Linux 8, and Oracle Solaris 11.4.
- **Custom images** can be created from an instance running on PCA, making it possible to use an image as the basis for cloned instances with customized contents.
- **Bring Your Own Image (BYOI)** images are imported from a different platform.

For further description of these terms, see the PCA Concepts Guide sections 7.1 (Compute Services) and 7.2 (Compute Images).

This document emphasizes BYOI using a "lift and shift" approach in which an OS instance is moved from Exalogic to PCA with as few OS and configuration changes as possible. Another approach is to create entirely new images based on the latest version of the desired operating system and application software. This is often used in "technology refresh" situations where the entire hardware and software stack is modernized. Both approaches are valid and widely used, depending on an institution's business and technical needs.

LICENSING

PCA provides platform images for Oracle Linux 7, Oracle Linux 8, and Oracle Solaris 11.4

> **Note**: You must comply with all licensing requirements when you upload and start instances based on OS images that you supply.

MIGRATION WORKFLOW

Moving to PCA X9-2, as with movement to any platform, is similar to performing a datacenter migration. It requires planning and multiple operational steps. The workflow contains these steps:

1. Preparation: data collection on source system.

2. Preparation: updates to instances on source system before they are exported.

3. Export: make the virtual machine boot disk image available outside the hypervisor environment.

4. Transform the boot disk image for operation with PCA X9-2, if needed.

5. Import image into PCA.

6. Create and launch instance from the imported image.

7. Post-install data migration and configuration.

8. Parallel test and switchover.

The above steps 2 to 5 apply to lift and shift migration. If new instances are being created, they would be replaced by installing the application stack onto new Oracle Linux instances.

PREPARATION: DATA COLLECTION

The first phase of migrating workloads to PCA X9-2 is to collect information about the VMs to be moved. This must be done using the source platform management tools or directly within each source virtual machine.

Collect the information described in the following sections. Some parts require logging into the VM, while others can be gathered from the hypervisor environment or management framework such as Oracle Enterprise Manager. Not all of this information will be needed immediately but will be needed to complete the migration process. Commands mentioned below are not intended to be exhaustive. A system administrator for each source environment can easily find those details

VM Shape

Collect the number of cores and memory size assigned to the source VM. This will determine the corresponding PCA shape for the VM's requirements. See Appendix 1 for the list of PCA shapes. Notice the 1:16 ratio of cores to RAM.

Select a shape that meets the instance's capacity needs, adjusting for CPU speed and resource utilization, and accommodating future growth plans. You may be able to select a smaller shape if peak CPU utilization is low, or if memory is over-allocated. That would help maximize the number of instances that can be hosted.

> **Note:** The shape also determines the maximum number of virtual NICs the instance can have. You may need a larger shape than the CPU and memory requirements dictate. See Appendix 1 for details.

Tools in the source environment and operating system can be used to collect this information. For example, with Exalogic and Oracle VM, the VM's shape is directly visible from the Oracle VM Manager browser interface, or by issuing the Oracle VM CLI command `"show vm name=VMNAME"`. The following image shows an Exalogic image running under Oracle VM:

You can also obtain shape information within the guest operating system: On Linux, issue commands: `awk '($1 == "processor")' /proc/cpuinfo | wc -l ; mpstat -P ALL` for CPU count, and `"free -h"` or `"cat /proc/meminfo |grep MemTotal"` for memory size.

You can access the VM's 'vm.cfg' file to get its details in ASCII text format. Note the highlighted lines, which describe the virtual disk and network configurations and Exalogic-specific details. The example shows the same VM displayed above, with the name "EL1_base", a single Ethernet virtual NIC, 16GB of RAM, 4 virtual CPUs, and a single virtual disk. The new instance that replaces this VM will use Ethernet, not InfiniBand.

```
# cat \
/OVS/Repositories/0004fb0000030000fb8bfb7348173fea/VirtualMachines/0004fb00000600005d0c282a60
9633d3/vm.cfg
kernel = '/usr/lib/xen/boot/hvmloader'
vif = ['mac=00:21:f6:7e:30:85,bridge=xenbr0']
OVM_simple_name = 'EL1_base'
vnclisten = '127.0.0.1'
```

```
vncconsole = 1
serial = 'pty'
disk =
['file:/OVS/Repositories/0004fb0000030000fb8bfb7348173fea/VirtualDisks/0004fb0000120000b912f8
5b31cb3130.img,hda,w']
vncunused = 1
uuid = '0004fb00-0006-0000-5d0c-282a609633d3'
on_reboot = 'restart'
EL_Template_Version = 1
cpu_weight = 27500
pae = 1
memory = 16384
cpu_cap = 0
maxvcpus = 4
OVM_high_availability = False
acpi = 1
timer_mode = 2
on_poweroff = 'destroy'
vnc = 1
OVM_os_type = 'Oracle Linux 6'
expose_host_uuid = 1
on_crash = 'restart'
exalogic_ipoib = [{'pkey': ['0x8004'], 'port': '1'}, {'pkey': ['0x8004'], 'port': '2'}]
apic = 1
name = '0004fb00000600005d0c282a609633d3'
guest_os_type = 'linux'
device_model = '/usr/lib/xen/bin/qemu-dm'
builder = 'hvm'
vcpus = 4
keymap = 'en-us'
exalogic_vnic = [{'pkey': ['0xffff'], 'guid': '0xbbb34d724b05dd73', 'port': '1'}, {'pkey':
['0xffff'], 'guid': '0xbbb34d724b05dd74', 'port': '2'}]
OVM_cpu_compat_group = ''
OVM_domain_type = 'xen_hvm'
```

Note the lines that describe the VM's virtual disks, network, CPU configuration, domain type, and lines for Exalogic features "`exalogic_ipoib`" and "`exalogic_vnic`". The PCA implementation will use Ethernet-based virtual NICs.

Storage Configuration

Collect the number and sizes of disks owned by the VM.

> **Note**: This document exclusively describes moving the boot disk, referred to as the boot volume on PCA. Disks containing data, swap, `/tmp`, application binaries are not transferred as part of this process. Their sizes will be needed after the boot volume is transferred and the instance is created. Data disks are referred to as *block volumes* and are defined and populated after the instance is created.

As with CPU and memory size this can be determined from the source platform hypervisor. In PCA 2 and Oracle VM the details are visible from Oracle VM Manager browser interface and CLI as described above.

Alternatively, from within the Oracle Linux VM, issue "`df -h; lsblk; mount`".

Filesystem mounts (NFS and SMB/CIFS) and iSCSI block storage

Though not part of the instance definition, you will need details on NFS and SMB mounts and iSCSI targets accessed by the VM. If the new platform has access to the same storage platform they may still be used. Use the same "within the VM" commands as in the Storage configuration section above.

> **Note**: Changing the virtual machine will change its iSCSI IQN, and continued access to the same iSCSI block storage will require adjustment within the VM instance and at the iSCSI SAN storage layer.

Network Configuration

OS instances on Exalogic use InfiniBand network connections that are replaced with Ethernet. The details for removing them from an image to be moved to PCA in a "lift and shift" migration are described below. The current network environment must be collected whether a lift and shift or a fresh OS installation.

For each virtual NIC in the VM, collect its IP address, netmask, MTU, routing, DNS and NTP settings. On Linux use `ifconfig -a; ip a; netstat` commands and `'cat /etc/resolv.conf'`. You can also use graphical tools: on GNOME you can use the desktop navigation System→Preferences→Internet and Network→Network Connections and then display IP settings. It will be necessary to remove any InfiniBand devices in the virtual machine.

MIGRATION OVERVIEW

There are two approaches to migrating applications from Exalogic to PCA: fresh OS install and "lift and shift", described later in this document. The result is that the Exalogic infrastructure stack is replaced by the PCA infrastructure stack

Figure 1. Exalogic stack with Exalogic Control and InfiniBand optimizations
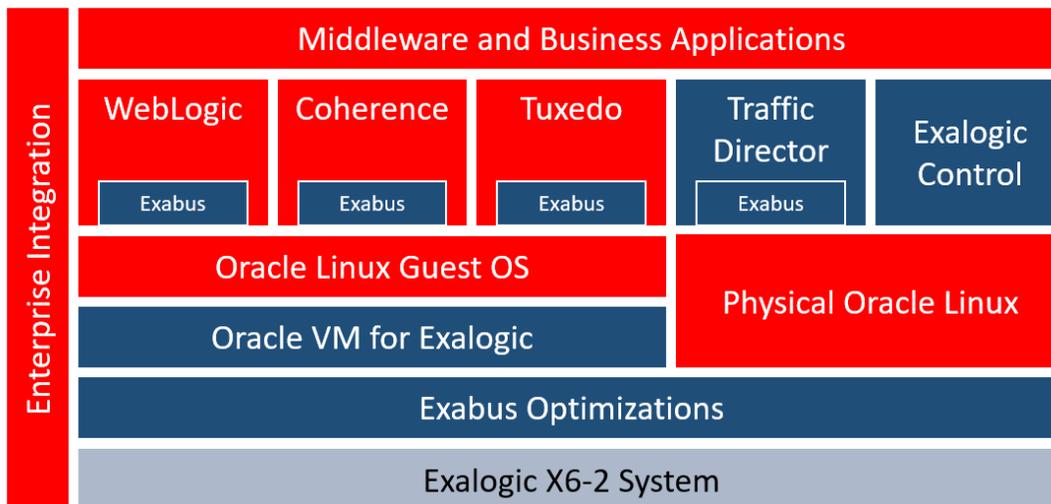
= Exalogic specific software

In general, applications can be migrated without change.   Customers should consider the following general comparisons between the environments when planning migrations and evaluate whether they want to create a new environment that matches the old one as much as possible, or use the migration as an opportunity for an application and OS technical refresh as well as a hardware refresh. These considerations apply:

- Oracle WebLogic Server 10.3.6 and 12.1.3 versions are nearing end of life.   New features such as updated REST support, JSON processing, auto-scaling and REST management in Oracle WebLogic Server 12.2.1.X enable better integration with cloud systems. Customers using prior versions should plan to migrate to 12.2.1.3 or later as part of the migration process.

- The migration will change the underlying compute infrastructure used by applications

- Although Oracle Traffic Director (OTD) is supported for migration to Oracle Private Cloud Appliance, native Kubernetes load balancers such as Traefik and Voyager are more appropriate for Kubernetes, and are recommended as replacements for Oracle Traffic Director. Simple load balancers can be implemented by using `haproxy`, which is included with Oracle Linux. Load Balancer as a Service (LBaaS) will be the preferred native load balancer on PCA X9-2 when available.

- Access to external systems via HTTP and T3 protocol is supported, including access to databases, and Oracle RAC clusters running in Oracle Exadata systems. SDP protocols are not supported on Oracle Private Cloud Appliance, so any existing usage of SDP within domains running on Oracle Exalogic Cloud systems must be removed. That is consistent with current recommendations for applications running on Exalogic.

- Oracle WebLogic Server, Oracle Coherence, and Oracle Application Development Framework are supported for use in Kubernetes with the WebLogic Kubernetes tools.

FRESH OS INSTALL

This method deploys new virtual machines containing WebLogic Server or other application software, and then migrate application contents (binaries, scripts, tools) from current instances. This also applies to applications currently running on commodity servers.

The advantage of this approach is that it is a software technology refresh to complement the hardware technology refresh. This permits a more significant transformation and modernization of the application environment, including the opportunity to modernize applications, run them inside Docker containers and use Kubernetes. PCA fully supports container-based application delivery using Oracle Cloud Native Environment, representing the modern trend for delivering application systems.

The administrator would download the latest Oracle Linux OS and application versions from Oracle Software Delivery Network at https://edelivery.oracle.com to perform a fresh installation of the OS and application, and copy application data onto it. The administrative steps would be no different on PCA than on any other platform.

Oracle recommends this method, which brings the software stack fully up to date. However, this may require additional analysis to capture the current system's contents.

Customers often prefer to move application environments to new hardware with as few changes as possible, as described in the next section.

LIFT AND SHIFT PREPARATION: PRE-EXPORT UPDATES

It is advisable and often necessary to prepare a VM for PCA by making changes to the VM on the original source system.

Operate on a clone of the production image on the source system. This protects the production environment and provides a known fallback if the process needs to be repeated. On Exalogic, use the Oracle VM Manager browser interface. Select the VM in the Servers and VMs tab, right click and select Clone. In all cases, stop the source VM to ensure that the clone has consistent disk contents.

Perform the following steps on the clone, not the original VM. **Use a clone!**

Patch to current level of the OS

Update the clone VM to the current patch level of the same OS. The instructions depend on the guest operating system. For example, `'yum update'` to the latest patch level for Oracle Linux . Reboot after patching to ensure proper operation post-patching. This ensures that the most recent device drivers are available and prevents risk from recently closed bugs and security exposures.

NOTE: We strongly urge customers using Oracle Linux 6 or earlier to move to a current Oracle Linux version, ideally Oracle Linux 8. That will ensure that the appropriate device drivers for PCA X9-2 are installed, as well as many bug fixes and enhancements. We encourage users of Oracle Linux 6 and earlier to use a fresh OS installation rather than a lift and shift.

The steps for using an older version of Oracle Linux are described for customers who need to do that as an interim procedure.

Change virtual network devices to DHCP

Several changes may be needed to the guest VM's virtual network in order for it to work after being installed on PCA.

PCA X9-2 uses DHCP to provide a network address to the new instances. Since multiple instances can be cloned from the same image, it makes sense to assign new MAC and IP addresses when each instance is created.

> **Note**: The IP address acquired by DHCP when the instance is created is stable.

> **Note:** the IP address visible to the instance is a private IP address on a virtual cloud network (VCN). The instance is accessed from outside the PCA via a Network Address Translation (NAT) layer. See the Network tab in the Compute Enclave's instance description to see both private and public IP addresses.

On Linux, edit the file `/etc/sysconfig/network-scripts/ifcfg-NICNAME` and remove `HWADDR` or `MACADDR` lines and set `BOOTPROTO="dhcp"`. Alternatively, you can use the graphical tool provided with your distribution. Additionally, remove the files `/etc/udev/rules.d/70-persistent-ipoib.rules` and `/etc/udev/rules.d/70-persistent-net.rules` if present. Those files contain MAC addresses that won't match the new instance, They are recreated automatically if needed.

Remove Exalogic-specific tools/drivers

At this point, remove Exalogic OS components.

Remove the InfiniBand RPM used with Exalogic `infinibus-N.N.x86_64`

Then remove or move away the network definition files for the Infiniband devices:
```
# cd /mnt/sysimage/etc/sysconfig/network-scripts
# rm ifcfg-ib0.0004; rm ifcfg-ib1.0004
```

Remove the OFED component:
```
# /usr/sbin/ofed_uninstall.sh --force
```

If Weblogic is in use, connect to the WLS console, and turn off the Exalogic enhancements by reversing the steps used to turn them on for Exalogic, in particular disabling Socket Direct Protocol (SDP). This is a current best practice for Exalogic and is not supported for PCA. See the Oracle Fusion Middleware Exalogic Enterprise Deployment Guide at https://docs.oracle.com/cd/E18476_01/doc.220/e18479/optimization.htm#ELEDG779

When Exalogic VMs are identical other than hostname and IP address, you can stop the VM and clone it for additional members of the application cluster. Boot each VM up in turn to change its hostname and address to make them unique.

## Install kernel and driver features for PCA

The operating system may need to have VirtIO device drivers explicitly included at boot time. On Oracle Linux use the `dracut` command to add device drivers to initramfs: "`dracut --force -N`" or "`dracut -N --regenerate-all`".

- The `-N` option disables hostonly mode, which creates an `initramfs` only for the current host. This option produces a generic image.
- If you are running on an old version of Linux that does not have the "-N" option, specifically list the kernel modules needed for PCA X9-2. Use a command like "`dracut --add-drivers 'sg sd_mod virtio virtio_pci virtio_ring virtio_scsi'`"
- The `--regenerate-all` option regenerates `initramfs` for all kernel versions in `/boot`, not just the one currently booted. This is useful for Oracle Linux to ensure that both UEK and RHCK kernels get the needed drivers, regardless of which is currently booted
- The option `--force` overlays the current `initramfs` file already present.

You can see the updated `initramfs` files in `/boot` - they will be larger than previous versions because of the added device drivers.

You can optionally backup kernels before running `dracut`, e.g.: `mv /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r).img.bak`

> **Note**: If the `dracut` step is omitted, it can be performed the first time the instance is launched on PCA X9-2: Log into the Compute Enclave, select the instance, connect to the console, and select the rescue line in the Grub menu. If boot fails before you can select the rescue boot, issue a Reset action on the instance from the Compute Enclave. After the instance boots, login and issue the `dracut` commands described above.
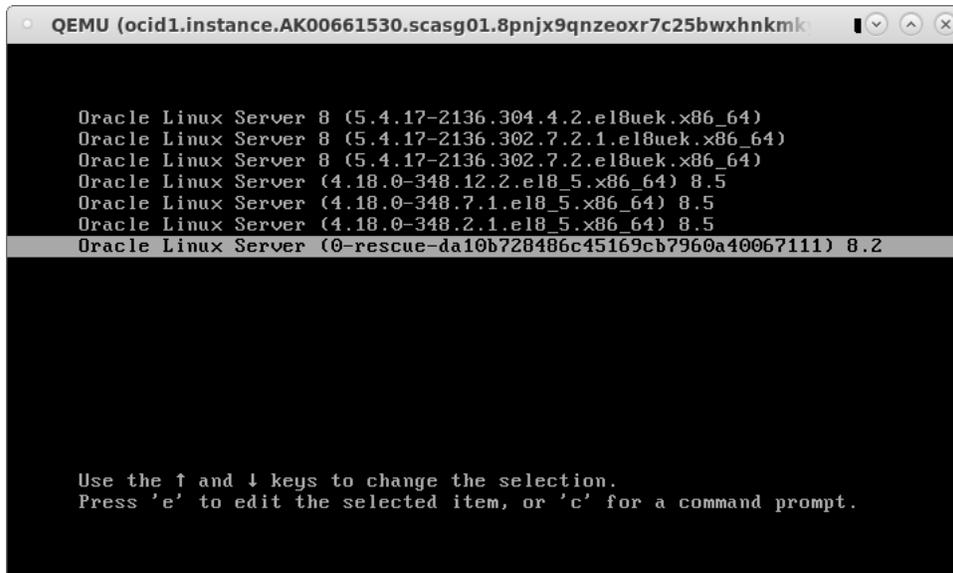
Prepare cloud-init (optional)

Customers are encouraged to install the `cloud-init` (Linux) toolkit. It is the industry-standard, vendor agnostic methods for cross-platform instance initialization that identify the cloud the instance is running on during boot, read any provided metadata from the cloud and initialize the system accordingly. It has a number of capabilities, among which is converting from password-based `ssh` to using a key pair exchange as described in section 6.4 of the User Guide and at https://cloudinit.readthedocs.io/en/latest/

On Oracle Linux, install the `cloud-init` and `oci-utils` packages suitable for your Linux distribution. For Oracle Linux 8 this is:

```
# dnf install yum-utils
# yum-config-manager --enable ol8_addons
# dnf install cloud-init oci-utils
```

You would then `ssh` into the launched instance with userid **opc** without a password, using the key pair exchange described in the references cited above.

The preceding steps disable password-based authentication. If you want to continue using password based ssh authentication on the target system to keep it more like how it operated on the source platform, it can be re-enabled before export by editing `/etc/cloud.cfg` and adding the line:

```
 ssh_pwauth:    1
```

Alternatively, this can be done before export by commenting out the following line in `/etc/ssh/sshd_config`:
```
# PasswordAuthentication no
```

Prepare startup and file system

The image only contains the boot disk, so disable any boot-time services and applications that require other disks. References to non-boot file systems must be removed until re-created later. If the boot image refers to other disks for swap, `/tmp`, or `/var/logs` they must be temporarily moved to the boot volume.

On Linux: edit `/etc/fstab` and remove entries to file system mounts that are not on the boot disk. Ensure any boot disk partitions are not directly referenced by device names such as `/dev/xvda`. Instead use "`LABEL=`" or "`UUID=`". References to LVM devices don't need to be modified if referring to the boot disk. Run the 'pvs' command to ensure that the only LVM physical volume (PV) is the boot volume. If a volume group (VG) is on other physical volumes it should be commented out of `/etc/fstab`. If a logical volume consists of both the boot and other disks then the migration is more complicated and out of the scope of this document.

Disable any services or applications that launch at boot time and require full configuration, since the initial launch on PCA X9-2 will not have the complete operational environment.

EXPORT BOOT DISK

Copy the boot disk from the source environment to a utility server that will be used to prepare the image for upload to PCA. Use a method that depends on the source hypervisor and management platform.

For example, obtain the `.img` file for the boot disk by copying it from the storage repository shown in the VM's `vm.cfg` file. The command would be run on the compute node or Oracle VM Server (OVS) and look like

```
cd /OVS/Repositories/0004fb00000300007b2cd14ca0d48e8b/VirtualDisks
scp 0004fb0000120000fcd80512881d4979.img
utilityhost:/data/0004fb0000120000fcd80512881d4979.raw
```

In the above example, the output file is renamed with the file extension .raw, an optional change that indicates that the Oracle VM `.img` file uses 'raw disk' image format.

Speed up export by compressing disk image files

Disk image files are large and copying them to other hosts can take a long time. This may go much faster if you use compression, especially for sparse disk images. You can optionally compress the disk image file using a command like `gzip` or `lzop` to reduce the time to copy to the utility host. Here is an example that uses `lzop` and pipes output to `scp`:

```
lzop -1 < 0004fb0000120000fcd80512881d4979.img | ssh utilityhost 'lzop -d >
/data/0004fb0000120000fcd80512881d4979.raw'
```

We use `lzop` for compression in this case because a fast network is available, so compression speed was more important than compression ratio. If you have a slower network, you may opt for another approach such as using `gzip` or `bzip2`.

```
gzip 0004fb00001200008190ce0329fdacdf.img
scp 0004fb00001200008190ce0329fdacdf.img.gz utilityhost:/data/
```

All of these methods, or using no compression at all, are equally valid and depends on the administrator's preferences.

SET UP THE UTILITY HOST (ONE TIME OPERATION)

Create a utility host that runs Oracle Linux. This can be a virtual machine or a bare metal instance. Prepare the utility host by installing the OCI CLI, as described in section 1.2 of the User Guide.

Also, use the following commands to install `qemu-img` and `virt-sparsify`. This example assumes an Oracle Linux 7 host:

```
$ sudo yum upgrade
$ sudo yum-config-manager --enable ol7_kvm_utils
$ sudo yum install qemu-img libguestfs-tools
```

TRANSFORM BOOT DISK

The boot disk may need to be converted into the open standard `qcow2` format used by the PCA hypervisor. Perform this on the utility host that has access to OCI CLI, `qemu-img` command and other optional utilities.

Process the boot disk

Perform this operation on the boot disk being prepared for PCA. This is necessary for disk images from Oracle VM on Exalogic or PCA 2.x.

On the utility node, issue a command like the following, substituting in the actual filename for the disk image file:

```
$ qemu-img convert -f raw -O qcow2 0004fb0000120000fcd80512881d4979.raw output.QCOW2
```

Note: The output file must be named `output.QCOW2`.

Note: The `qemu-img` command can take a substantial time to run depending on the image file size and the system speed. You can use the flag "`-p`" to show progress displayed as a percentage in the form `(nn.nn/100%)`.

You may be able to reduce the size of the disk image by using the `virt-sparsify` tool. This tool works on many file systems: ext2/3/4. xfs, btrfs, NTFS, LVM. and reduces sparse disk image size by eliminating disk blocks not owned by a file. The amount of reduction varies based on the contents of the disk image before exporting. Depending on your utility environment, you may have to also issue "`export LIBGUESTFS_BACKEND=direct`" before running the command:

```
$ virt-sparsify --in-place output.QCOW2
```

Preparing for import: the OCI file type

PCA X9-2 image and Oracle Cloud Infrastructure (OCI) images use the file type `.oci`, which has two components: an `image_metadata.json` file that describes the image in JSON format, and the actual disk image named `output.QCOW2` prepared in the previous step. The `image_metadata.json` file describes the image's settings (BIOS vs. UEFI, how virtual devices are implemented), and what operating system it runs. The examples shown in Appendix 2 can be used with little change. Select the BIOS or UEFI version based on the source and (for neatness sake) change the operating system description. Exalogic images will be in BIOS mode.

```
$ tar zcf MyImage.oci image_metadata.json output.QCOW2
```

IMPORT TO PCA

Once the `oci` file is available, it can be imported into PCA without further change. Upload it from any web server that has network connectivity to the PCA management nodes. The file can be copied to an existing web server or you can create an impromptu web server using a Python command similar to the following (adjusting for port number as needed). If using Python 2.x use `python -m HTTPSimpleServer 8080` and If using Python 3.x use `python3 -m http.server 8000`

Import images into PCA by using the PCA Compute Enclave browser user interface (BUI) or the OCI command line interface (CLI). See the User Guide section 5.1 for a full description.

Import using browser interface

If using the browser interface, log into the Compute Enclave, go to the Compute Images page, click on 'Import Image', and then fill in details on the pop-up:

Import Image

Name

demo-image

Create In Compartment

pca3autoTenant ▾

Source Type

○ Import from an Object Storage Bucket

◉ Import from an Object Storage URL

Object Storage URL

|

Image Type

◉ VMDK

Virtual machine disk file format. For disk images used in virtual machines.

○ QCOW2

For disk image files used by QEMU.

Launch Mode

[Import Image] [Cancel]

Select 'Object Storage URL' and enter the URL to the file containing the boot image. Specify QCOW2, which applies to either a `.qcow2` file or a `.oci` file containing both metadata and the `output.QCOW2` disk image file. Click on the 'Import image' button to start the import. This will take you to a page showing the status of the image, initially 'Importing' and then 'Available' when the import completes.

The time to perform the import depends on the size of the image and the speed of the network between the PCA system and the web server. The user interface is not locked during the import so you can proceed to other tasks or log off. Check on the status of the image by viewing the image's display.

Import using OCI CLI

If using the OCI CLI, use a line similar to the example below, substituting in the image URL and the OCID for your tenancy. The import command produces a display of the import work unit, and you can use a 'get' command to display its status.

```
$ oci compute image import from-object-uri --uri http://mynfs/shares/export/images/mydemo.oci
--display-name "demo-image" --compartment-id $OCI_CLI_TENANCY
{
  "data": {
    "agent-features": null,
    "base-image-id": null,
    "compartment-id":
"ocid1.tenancy.AK00661530.scasg01.lk0p5dyyyrdctdo3vvb9vdgr8zvdwznk7tcqtfmvzv1xxym66oak00090146",
    "create-image-allowed": true,
    "defined-tags": {},
    "display-name": "demo-image",
    "freeform-tags": {},
    "id":
"ocid1.image.AK00661530.scasg01.lt0xzicgaiv9plkwurk2xswia4x00e4ds2vyxa78d1xmk17b1indxamf6i4h",
    "launch-mode": "PARAVIRTUALIZED",
    "launch-options": null,
    "lifecycle-state": "IMPORTING",
    "operating-system": "UNAVAILABLE",
    "operating-system-version": "UNAVAILABLE",
    "size-in-mbs": 0,
    "time-created": "2022-02-03T00:41:24.982617+00:00"
  },
```

```
  "etag": "ae01e390-eab5-449c-b78d-c655235af23f",
  "opc-work-request-id": "ocid1.workrequest.AK00661530.scasg01.storage-
1qsyui8onuv2q9evycdrnwtp13nj9l3eko37fzvhpahbobbav18r"
}
```

You can view the import via the browser interface, or check in its status via the CLI, using the OCID displayed in the import-image command.

```
$ oci compute image get --image-id
ocid1.image.AK00661530.scasg01.lt0xzicgaiv9plkwurk2xswia4x00e4ds2vyxa78d1xmk17b1indxamf6i4h
{
  "data": {
    "agent-features": null,
    "base-image-id": null,
    "compartment-id":
"ocid1.tenancy.AK00661530.scasg01.lk0p5dyyyrdctdo3vvb9vdgr8zvdwznk7tcqtfmvzv1xxym66oak0009014
6",
    "create-image-allowed": true,
    "defined-tags": {},
    "display-name": "demo-image",
    "freeform-tags": {},
    "id":
"ocid1.image.AK00661530.scasg01.lt0xzicgaiv9plkwurk2xswia4x00e4ds2vyxa78d1xmk17b1indxamf6i4h"
,
    "launch-mode": "CUSTOM",
    "launch-options": {
      "boot-volume-type": "PARAVIRTUALIZED",
      "firmware": "BIOS",
      "is-consistent-volume-naming-enabled": false,
      "is-pv-encryption-in-transit-enabled": false,
      "network-type": "PARAVIRTUALIZED",
      "remote-data-volume-type": "PARAVIRTUALIZED"
    },
    "lifecycle-state": "AVAILABLE"
    "operating-system": "Fedora Linux"
    "operating-system-version": "35"
    "size-in-mbs": 512000
    "time-created": "2022-02-03T00:41:24.982617+00:00
  }
  "etag": "d96dda18-e10b-4d87-83ab-425bb5cf582e
}
```

INITIAL LAUNCH ON PCA

You can launch the image without any special additional steps, as illustrated below. See section 6 of the User Guide for further details.

Instance launch using browser interface

From the page describing the image, click on the Controls drop-down menu, then click on 'Create Instance'. Fill in the pop-up window with details on the instance's name, fault domain, shape and virtual network and subnet. If using `cloud-init`, fill in the public key to permit password-free `ssh`.
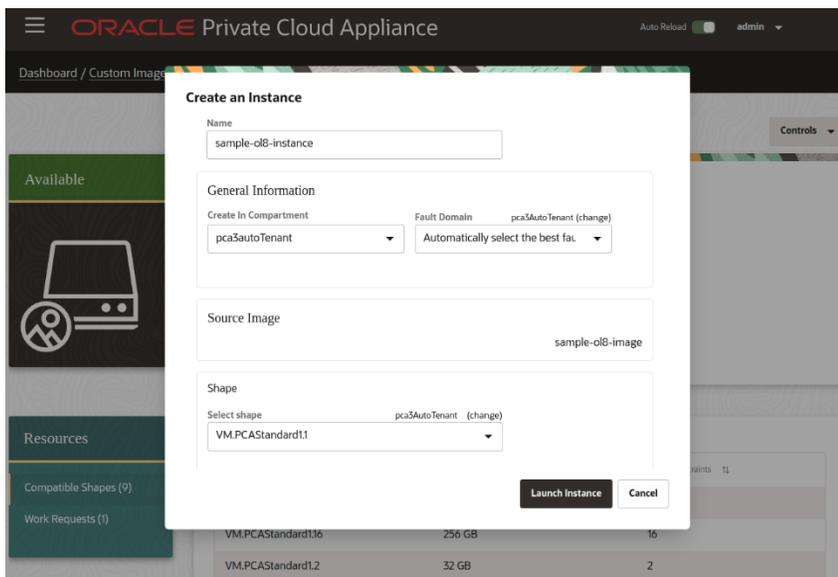
The instance will be in Provisioning state while it is being created, and then is automatically launched. After the instance is provisioned you can create a console connection to connect to the instance console, as described in the User Guide.

Instance launch using OCI CLI

You can also use OCI CLI with a pattern similar to this, substituting in desired values for the variables displayed, shape, source details, subnet, keys and names:

```
$ oci compute instance launch -c $OCI_CLI_TENANCY \
    --availability-domain ad1 \
    --shape VM.PCAStandard1.2 \
    --source-details '{"boot-volume-size-ingbs"
:100,"imageId":"$MYIMAGEID"},"sourceType":"image"}' \
    --subnet-id $MYSUBNET_OCID \
    --metadata '{"ssh_authorized_keys":"ssh-rsa ...$RESTOFMYKEY myuserid@mydesktop"}' \
    --display-name "myinstance" \
    --assign-public-ip true
```

The advantage of the CLI is that it is scriptable and can be repeated in bulk with fewer manual steps.

If you need to override the launch mode or parameters, you can specify them on the command line:

```
$ oci compute instance launch [...] \ --launch-options '{"boot-volume-
type":"PARAVIRTUALIZED","firmware":"UEFI_64", "is-consistent-volume-naming-
enabled":false,"is-pv-encryption-in-transit-enabled": false, "network-
type":"PARAVIRTUALIZED","remote-data-volume-type": "PARAVIRTUALIZED"}'
```

POST-LAUNCH: COMPLETE THE INSTANCE

Add block volumes (non-boot disks)

Previous steps created the boot environment on the PCA X9-2. Review the data collected in the preparation steps to size disks that needed to complete the image. Use the PCA Compute Enclave navigation to create block volumes and add them to the instance. Then, create file systems on the block volumes using the tool appropriate to the guest operating system, and copy contents from the source system. The approach will depend on the applications and nature of the data, so details are out of scope for this document.

If the VM on the source system uses NFS, SMB/CIFS or iSCSI targets for its data, it may be possible to expose them to the new instance without having to create new storage resources. Permissions and iSCSI parameters may need to be changed on the storage device.

Note: PCA X9-2 does not support Fibre Channel devices.

TAKE A BACKUP

Create a backup after first successful boot for fallback after any further changes. See the Users Guide for details on creating a new custom image and taking boot and block volume backups

DATA MIGRATION, ACCEPTANCE TESTING AND CUTOVER

At this point you have a running instance on PCA. Copy over the application data using the tools appropriate to the operating system and application (e.g.: rsync or scp for file data and conduct user acceptance testing in parallel with undisturbed operation in the source system. Once acceptance testing is successful, perform a final data synchronization from the source stem to the new environnment. The procedures for this last, essential, step will be specific to the operating system and applications.

SUMMARY

This document provides a guide with instructions and process for moving workloads from source system to PCA X9-2. Architects and administrators transitioning virtual machines to PCA can use this as a basis for their migrations.

APPENDIX 1: STANDARD SHAPES

PCA X9-2 supports the following shapes. When sizing an instance, select the shape that provides sufficient CPU and memory capacity. If the source VM's CPU speed is slower than the X9-2 server is or is overprovisioned with more capacity than it needs, you can select a smaller shape.

Note that the shape also includes a maximum number of virtual network devices, so if an instance needs more than provided by the smallest shape meeting its CPU cores and memory requirements you'll have to promote to a larger shape

Shape Name	Cores	Memory (GB)	Virtual NICs
VM.PCAStandard1.1	1	16	2
VM.PCAStandard1.2	2	32	2
VM.PCAStandard1.4	4	64	4
VM.PCAStandard1.8	8	128	8
VM.PCAStandard1.16	16	256	16
VM.PCAStandard1.24	24	384	24
VM.PCAStandard1.32	32	512	24
VM.PCAStandard1.48	48	768	24
VM.PCAStandard1.Max	60	960	24

APPENDIX 2 IMAGE_METADATA.JSON EXAMPLES

The contents below can be used as the basis of the `image_metadata.json` file included in a `.oci` bundle for upload. Exalogic images will be in BIOS mode, and both BIOS and UEFI metadata file versions are shown for reference. Values for "`operatingSystem`" and "`operatingSystemVersion`" should be set to the values for the imported image. Other fields should be left as shown.

BIOS images

```
{
  "version": 2,
  "externalLaunchOptions": {
    "firmware": "BIOS",
    "networkType": "PARAVIRTUALIZED",
    "bootVolumeType": "PARAVIRTUALIZED",
    "remoteDataVolumeType": "PARAVIRTUALIZED",
    "localDataVolumeType": "PARAVIRTUALIZED",
```

```json
    "launchOptionsSource": "CUSTOM",
    "pvAttachmentVersion": 1,
    "pvEncryptionInTransitEnabled": false,
    "consistentVolumeNamingEnabled": false
  },
  "imageCapabilityData": null,
  "imageCapsFormatVersion": null,
  "operatingSystem": "Oracle Linux",
  "operatingSystemVersion": "8.5"
}
```

UEFI images

```json
{
  "version": 2,
  "externalLaunchOptions": {
    "firmware": "UEFI-64",
    "networkType": "PARAVIRTUALIZED",
    "bootVolumeType": "PARAVIRTUALIZED",
    "remoteDataVolumeType": "PARAVIRTUALIZED",
    "localDataVolumeType": "PARAVIRTUALIZED",
    "launchOptionsSource": "CUSTOM",
    "pvAttachmentVersion": 1,
    "pvEncryptionInTransitEnabled": false,
    "consistentVolumeNamingEnabled": false
  },
  "imageCapabilityData": null,
  "imageCapsFormatVersion": null,
  "operatingSystem": "Oracle Linux",
  "operatingSystemVersion": "7.9"
}
```

REFERENCES

See these reference documents for additional information:

- Oracle PCA 3.01 documentation library: https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/
  - Oracle PCA Concepts Guide: https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/3.0/concept-3.0.1/index.html
  - Oracle PCA User Guide: https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/3.0/user-3.0.1/index.html
- Bring Your Own Image page for Oracle Cloud Infrastructure: https://docs.oracle.com/en-us/iaas/Content/Compute/References/bringyourownimage.htm
- Oracle VM Server for x86 and Oracle VM Manager documentation library at https://docs.oracle.com/en/virtualization/oracle-vm/index.html
- cloud-init documentation https://cloudinit.readthedocs.io/en/latest/
- "Introducing VirtIO Drivers with Oracle Solaris on OCI" at https://blogs.oracle.com/solaris/post/introducing-virtio-drivers-with-oracle-solaris-on-oci
- https://cloudbase-init.readthedocs.io/en/latest/
- Oracle Private Cloud Appliance 2.4 Backup Guide at https://www.oracle.com/technetwork/server-storage/private-cloud-appliance/documentation/pca-backup-x8-0-6-5676918.pdf

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

Migrating Mid-tier Applications from Oracle Exalogic to Oracle Private Cloud Appliance X9-2
May 2222
Author: [OPTIONAL]
Contributing Authors: [OPTIONAL]