ORACLE

Oracle Virtualization

Best practices for installation and deployment

January, 2026 - Version 1.0

Public

Purpose statement

This document provides an overview of the best practices for using Oracle Virtualization. It is intended solely to help you to plan for the deployment of the product features described.

Disclaimer

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, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Contents

ORACLE

Introduction

Oracle Virtualization is a proven, enterprise-grade server virtualization solution that provides KVM-based virtualization and management capabilities for on-premises data centers. The Oracle Virtualization solution is built on two core components: Oracle Linux Virtualization Manager, the management platform built from the open source oVirt project, and Oracle Linux KVM (Kernel-based Virtual Machine), a type-1 hypervisor that delivers the physical server virtualization.

The scope of this document includes Oracle Linux Virtualization Manager and Oracle Linux KVM compute host setup; data center and cluster design; memory features such as KSM, ballooning, and huge pages; storage domain selection; and network design with bonding and end-to-end MTU validation.

For virtual machines (VMs), the guidance covers high-performance profiles, I/O threads and multiqueue tuning, virtio-net, disk interfaces, standardized QEMU machine types, and high-availability policies.

The document also highlights disaster recovery options, routine engine backups, fencing practices, and structured log collection to support resilience and efficient day to day operations.

ORACLE

Getting Ready

Reading the existing Getting Started and Administration guides is essential to understanding the installation process and how to use Oracle Linux Virtualization Manager.

Mission-critical services require proper planning. To prepare the environment, read the Architecture and Planning guide, and to confirm the availability of specific features, check the latest Release Notes.

This paper does not replace any of the documents mentioned above.

Overview

The following provides a high-level summary of the core Oracle Virtualization components and their relationships, outlining control, segmentation, compute, storage, and networking.

- **Engine**: The management server that hosts the ovirt-engine, also known as Oracle Linux Virtualization Manager. Stores inventory, policies, and tasks; schedules placement, migrations, and lifecycle operations.
- **Hosted-Engine**: The engine running as a highly available VM on the same cluster it manages.
- **Data Centers**: Top-level logical domains that group storage domains and clusters. Used to separate tenants or environments (prod/non-prod, sites/DR).
- **Clusters**: Sets of homogeneous hosts that share a CPU baseline. Workloads can live migrate within a cluster.
- **Hosts**: Oracle Linux KVM hypervisors that run VMs.
- **Storage Domains**: Shared storage pools for VM disks and templates. Supported shared types include block (iSCSI/FC) and file (NFS). Each data center requires at least one storage domain.
- **Networks**: Logical networks mapped to host NICs/bonds and VLANs. Common roles include management, storage (iSCSI/NFS), live migration, and VM traffic.

Oracle Linux Virtualization Manager

The Engine (Stand-Alone)

When installed on a bare metal server or in a VM within another virtualization environment, the engine is referred to as a stand-alone engine. In this deployment model, extra care must be taken when performing the operating system (OS) installation, and all requirements described in the Getting Started guide must be met, including the following:

- The Oracle Linux OS and kernel versions
- Installation type (minimal)
- A correctly partitioned disk
- Software repositories and the package installation order
- Minimal network latency between the engine host and the Oracle Linux KVM hosts
- High availability for the engine is strongly recommended

The Self-Hosted Engine

The self-hosted engine is the recommended deployment model for Oracle Linux Virtualization Manager. In this model, the engine runs as a highly available VM within the Oracle Linux KVM hosts, eliminating the requirement for a dedicated physical management server. By setting up the management plane with the virtualization infrastructure, the self-hosted engine improves overall resiliency.

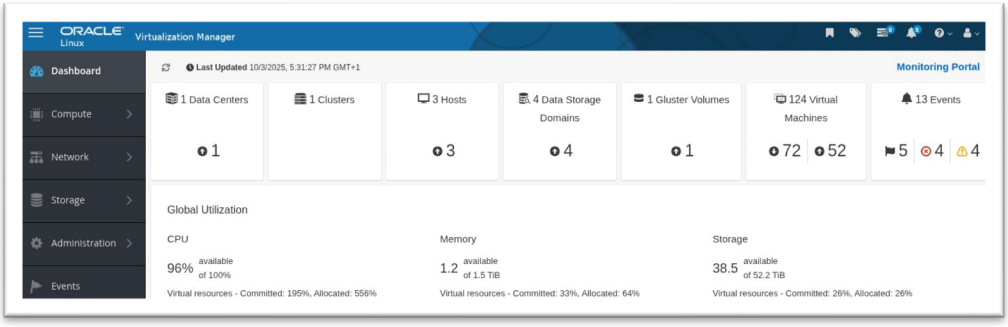
This approach delivers built-in high availability for the Manager and reduces the risk of Manager–host communication issues, as the management services operate within the same environment as the Oracle Linux KVM hosts. The solution is provisioned from a prebuilt VM image that applies correct disk partitioning and a minimal Oracle Linux installation, promoting a consistent and supportable baseline.

If the Manager becomes unavailable, host and VM high-availability policies are not enforced, automated scheduling and migrations are suspended, and changes to storage, networking, or VMs cannot be performed through the Manager. In this state, VMs cannot be started by the Manager, and administrative operations are limited until the management plane is restored. The self-hosted engine deployment model provides a higher level of assurance that the Manager remains available.

In summary, the self-hosted engine provides a secure, standardized, and highly available foundation for Oracle Linux Virtualization Manager deployments and should be adopted as the default architecture unless specific, well-justified requirements dictate an alternative with equivalent availability and connectivity assurances.

Administration Portal

The primary user interface for the end user is the Administration Portal. This web browser-based interface can be accessed by system administrators to perform most of their operations.



Data Centers

A data center is a top-level logical group that contains clusters and their shared storage domains. Every cluster in a data center must be able to access every storage domain in that data center. If any Oracle Linux KVM host in a cluster cannot access even one storage domain in the data center, that host becomes non-operational until access is restored.

The recommended way to separate projects or internal customers is to use separate data centers. Since storage domains belong to a single data center, this separation prevents cross-access to VM disks and data.

A data center has the following capacity characteristics:

- There is no limit on the number of data centers per engine.
- There is no limit on the number of clusters per data center.
- There is a limit of 50 storage domains per data center.

Clusters

A cluster is a logical group of Oracle Linux KVM hosts that share the same CPU family. The first time a host is added to a cluster, the CPU family is set automatically. To use the complete CPU feature set, all hosts should be in the same CPU family.

By default, Oracle Linux Virtualization Manager will attempt to select the secure version of a CPU family (for example, Secure Icelake instead of Icelake). A secure family—when available—enables Spectre v2 mitigations (IBRS on) and disables TSX.

When mixing CPU generations in the same cluster, consider the following:

- It is possible to mix different CPU generations within the same cluster, but the cluster CPU family must match the oldest CPU present in the cluster.
- The cluster CPU family is set by the first host added to the cluster; therefore, consider the future and add the oldest host first to avoid potential downgrades later.
- If a newer host is added to a cluster first and an older host is introduced later, the older host remains non-operational until the cluster CPU family is downgraded to match the older generation. Downgrading the cluster CPU family requires a restart of all VMs within the cluster, resulting in VM downtime.

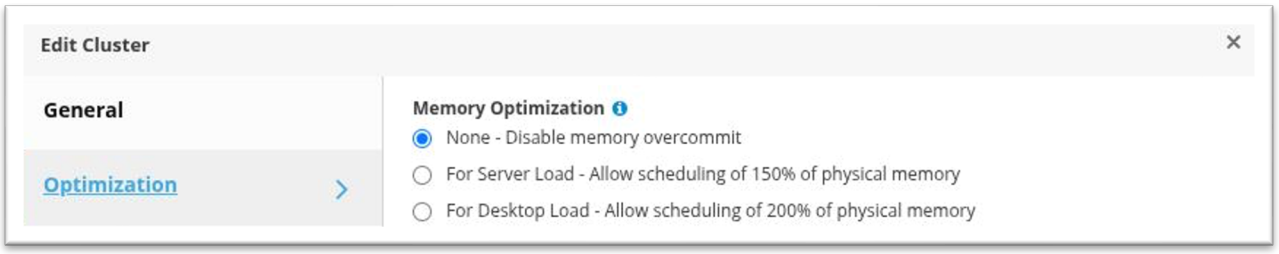
When downgrading the cluster CPU family, the following impacts apply to VMs:

- All running VMs must be restarted to adopt the downgraded cluster CPU feature set.
- Live migration fails between Oracle Linux KVM hosts if a VM was started with CPU flags not available on the target host, which is now configured with an older CPU family.

Memory Optimization

Memory optimization is set at the cluster level; its purpose is to allow more VMs to run than the total physical RAM would normally support, based on the assumption that not all VMs require peak memory simultaneously. The setting is available at 100%, 150%, and 200%.

Memory optimization values above 100% are not recommended for critical clusters or latency-sensitive operations.



Note the following:

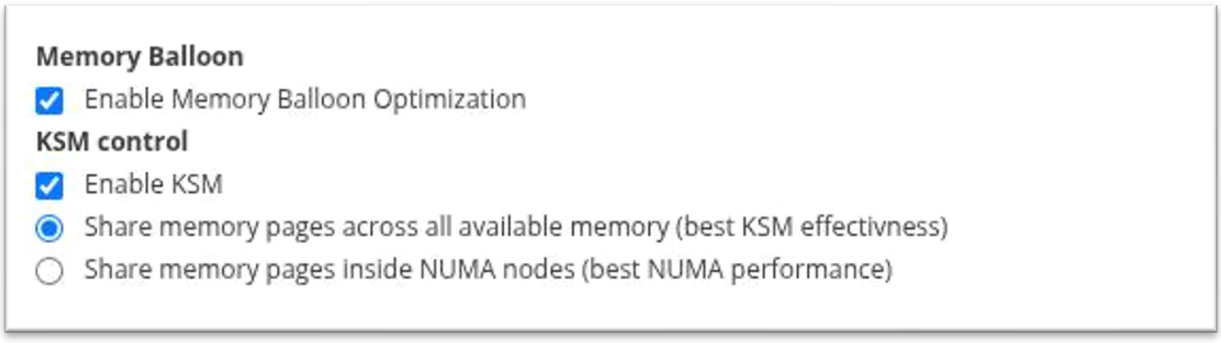
- To take advantage of memory optimization settings above 100%, memory ballooning and KSM must also be enabled.
- Only with both features enabled is it possible to "re-use" real memory and avoid swapping.
- Systems should always be monitored for utilization, and it is recommended to review the VM memory settings (Memory/Reserved Memory) if the Oracle Linux KVM host begins to use swap.

Memory Ballooning

Memory ballooning is a guest-aware mechanism that reclaims idle RAM from VMs and reallocate it to other VMs that require additional memory. It requires the balloon driver in the guest OS and can be enabled at both the cluster and VM levels.

Ballooning is most useful in non-critical application clusters, particularly when memory overcommit is in use (for example, 150%-200%). It works best for workloads with fluctuating memory demands and predictable idle periods.

In practice, deploy ballooning where workloads tolerate brief reclaim events, ensure the balloon driver is installed and active, and monitor for swapping or latency. Disable memory ballooning for latency-sensitive or mission-critical applications.



KSM (Kernel Same-page Merging)

Kernel Same-page Merging (KSM) is a host-level memory deduplication feature that combines identical memory pages across VMs to free RAM. In Oracle Linux Virtualization Manager, it is configured at the cluster level only.

Use KSM selectively. It can be appropriate in non-critical application clusters with many similar VMs (for example, standardized OS builds), but it should not be used for Oracle workloads, including Oracle Database and Oracle Real Application Clusters (RAC). The primary trade-offs are that KSM saves memory at the cost of additional CPU cycles for scanning and may add latency during page faults. Its benefit also diminishes as guest images diverge over time.

For critical application clusters, keep settings conservative: set memory optimization to 100%, disable memory ballooning, and leave KSM off. Also, use the Secure CPU Family.

For non-critical application clusters, higher overcommit may be acceptable: consider memory optimization up to 150% with adequate swap, enable ballooning per VM, and optionally enable KSM if the images are sufficiently similar. Pair KSM with standardized images to maximize deduplication and monitor CPU overhead; disable it if latency-sensitive applications are impacted.

For Oracle Database, Oracle RAC, and other Oracle enterprise workloads, disable memory ballooning and KSM, avoid or minimize memory overcommit, and size memory statically. Always validate configurations against the product support matrix and organizational compliance guidelines.

Networks

There are four main network types: management, VM, storage, and migration networks. These should be segregated on distinct VLANs or physical interfaces, configured for high availability (bonding: mode 1 or 4), and planned with clear MTU choices.

Separate each network type to isolate traffic domains and faults. Use NIC bonding (mode 1 [active-backup] or mode 4 [LACP]) for resilience. Keep configurations consistent across all hosts in the cluster and data center.

Jumbo Frames (9000 MTU)

Jumbo frames require a uniform MTU across the entire path. Every device and interface between source and destination—host NICs, bonds, switches, VLANs, LAGs, storage targets—must be configured for the same MTU. Validate with end-to-end testing before production use. The following use cases apply to jumbo frames:

- **Storage (iSCSI and NFS)**: Recommended, as large MTUs reduce overhead and CPU usage for bulk I/O.
- **Migration network**: Recommended to improve live migration throughput and reduce migration time.
- **VM network**: Appropriate only for latency-sensitive application paths (for example, heartbeat or backend services) when endpoints support jumbo frames.

Management traffic involves diverse endpoints (engine, admin workstations, BMCs/ILO/iDRAC, monitoring systems) that often do not use jumbo frames. The management network also carries HA signaling; MTU mismatches may disrupt HA communications. If enabled, ensure all involved endpoints and links are consistently configured. It is not recommended to change the default MTU in the management network.

Storage Domains

Use storage that is isolated from other workloads, with redundant network and fabric paths, and consistent configuration across all hosts.

Keep storage networking on dedicated VLANs/VSANs with appropriate bandwidth (for example, 10/25/40 Gbps for NFS; dual fabrics for FC; redundant paths for iSCSI). Standardize mounts, initiator/target settings, and firmware/driver levels across the environment to ensure consistent behavior and simplify operations.

Recommended Storage Domain types

- **NFS (file-based)**: Use one dedicated NFS export with its own underlying filesystem per storage domain, and do not share it with other workloads. Prefer NFS v4.2 for efficient sparse, clone, and discard operations.
- **iSCSI (block-based)**: Follow array vendor best practices for ALUA, queue depths, timeouts, and CHAP. Multipathing is mandatory and must be configured identically on all hosts.
- **Fibre Channel (block-based)**: Architect with dual fabrics, consistent zoning and masking, and identical HBA settings on every host. As in iSCSI, multipathing is mandatory and must be configured identically on all hosts.

Block-Based Storage Domains

In block-based storage domains, there is no filesystem on the domain; all objects are LVM metadata and logical volumes (LV). Each block-based storage domain is implemented as a single LVM volume group (VG), with virtual disks and snapshots represented as LVs within that group; adding LUNs to an existing storage domain expands the underlying volume group. As scale guidance, keep roughly 1,500 LVs per volume group as a soft limit, because operations and activation times can degrade beyond that; this count includes both disks and snapshots. Oracle Linux Virtualization Manager supports up to 400 physical LUNs per storage domain, but design for fewer, larger LUNs to reduce metadata churn and speed up scans and activations.

Capacity Management

Extending the capacity of a storage domain online is possible by growing an existing LUN or adding a new LUN. Shrinking by removing a LUN is only possible when the domain is in maintenance mode, and is risky and strongly discouraged. The first LUN in a storage domain cannot be removed. Plan capacity growth in larger increments to minimize VG fragmentation and monitor thin-provisioned pools closely.

Multipathing

Multipathing is mandatory for block-based domains. Ensure consistent zoning and masking so every host in the data center sees the same paths and WWIDs. Use a standardized `multipath.conf` and verify with `multipath -ll` on each host before placing domains into service. While Oracle Linux provides sensible defaults, the multipath device stanza (including ALUA mode and path selection) is the responsibility of the storage vendor (refer to Using Multipathing for Efficient Storage for configuration guidance). Keep initiator and target configurations consistent across hosts and validate failover behavior during deployment to catch problems early.

SCSI-3 Reservations

Direct-attached LUNs are supported without passthrough by default. When enabling SCSI-3 persistent reservations (PR) for a specific LUN, Oracle Linux Virtualization Manager will automatically enable passthrough for that LUN, because passthrough is required by the underlying stack to support PR. For all other LUNs where SCSI-3 reservations are not required, passthrough remains unsupported and disabled by default.

Engine Backup

Configure engine backups immediately after installing the engine and automate them as part of day-one operations. Use the supported engine backup tooling to capture configuration, database, certificates, and keys. Store backups off the engine VM/host (for example, on secure NFS or object storage), take them daily, and create an on-demand backup before engine updates or any maintenance that may modify the PostgreSQL database. Establish a retention policy and protect backups with appropriate access controls.

With a current backup, the engine VM or host can be restored to the backup's point in time, typically with minimal VM downtime, because storage domains and host inventory are preserved. Regularly test restore procedures to validate recovery time and integrity. If a valid or recent backup is not available, a new engine must be deployed to regain control of the storage domains, and a complete outage will be required to import the storage domain and reattach workloads.

```
[root@node ~]$ /usr/bin/engine-backup --scope=all --mode=backup --
log=/tmp/backup.log --file=/var/backup/backup.file
Start of engine-backup with mode 'backup'
scope: all
archive file: /var/backup/backup.file
log file: /tmp/backup.log
Backing up:
```

```
Notifying engine
- Files
- Engine database 'engine'
- DWH database 'ovirt_engine_history'
- Grafana database '/var/lib/grafana/grafana.db'
Packing into file '/var/backup/backup.file'
Notifying engine
Done.
```

Pre-Check Script

When deploying a new engine or KVM host, Oracle provides the `olvm-pre-check.py` script to verify that the host meets product prerequisites, including:

- The `oracle-ovirt-release` package installed
- A minimal OS installation
- Repositories are properly setup
- Linux kernel version
- Firewall, SELinux and FIPS status
- Ansible, libvirt and qemu versions
- Whether the host has a valid FQDN/hostname

When setting up a new host, follow the Oracle Virtualization Getting Started Guide and run the `olvm-pre-check.py` script to validate all requirements.

```
----------------------------------
    OLVM 4.5.5 PRE-CHECK SCRIPT
----------------------------------


+++ Checking oracle-ovirt-release-45                    [PASS]
+++ Checking if Host is installed                       [PASS]
+++ Checking if a Minimal Installation                  [PASS]
+++ Validating the 'Minimal Install' Group              [PASS]
+++ Checking enabled repositories                       [PASS]
+++ Running 'dnf makecache'                             [PASS]
+++ Dry run 'dnf update --assumeno'                     [PASS]
+++ Checking Linux Kernel                               [PASS]
+++ Checking kernel-uek-modules-extra                   [PASS]
+++ Checking Firewalld status                           [PASS]
+++ Checking SELinux status                             [PASS]
+++ Checking FIPS status                                [PASS]
    FIPS is disabled.
+++ If installed, check ansible version                 [PASS]
+++ If installed, check qemu-kvm version                [PASS]
```

```
+++ If installed, check libvirt version                    [PASS]
+++ Checking Hostname/FQDN                                 [PASS]
```

Oracle Linux KVM Hosts

A KVM host in Oracle Linux 8 is a physical server with the KVM kernel modules (`kvm` and `kvm_intel`/`kvm_amd`) enabled, using QEMU for hardware virtualization and libvirt to manage VMs.

In an Oracle Linux Virtualization Manager setup, the VDSM agent runs on the KVM host and orchestrates VM lifecycle and host configuration by calling libvirt's APIs (over local sockets) to define, start, stop, and monitor VMs, which QEMU executes as Linux processes with accelerated KVM. VDSM also coordinates host networking and storage configuration (bridges/bonds/VLANs, storage domains, and volumes), while VMs use virtio devices for efficient CPU, memory, disk, and network I/O.

Lifecycle and Ownership

- Treat Oracle Linux KVM hosts as managed appliances. Install the supported Oracle Linux version and add the `oracle-ovirt-release` package first (per the Getting Started guide).
- Do not install third-party agents/software or extra repositories; these can introduce unsupported components. Antivirus, malware detection, and IDS agents are not supported on Oracle Linux Virtualization Manager hosts.
- Before onboarding, perform only minimal network configuration on the OS: set up the intended management interface and any required bond/VLAN.
- Use standard interface names (non-conventional names can cause installation failures):
  - **Bonds**: `bond<number>` (for example, `bond0`)
  - **VLANs on bonds**: `bond<number>.<vlan>` (for example, `bond0.111`)

After a host is added to Oracle Linux Virtualization Manager, manage it exclusively through the Oracle Linux Virtualization Manager Administration Portal. Make any persistent network changes in the Administration Portal—local changes (for example, via `nmcli` or `nmtui`) are not preserved and may be reverted on reboot. Apply all operating system updates through the Administration Portal as well.

Follow standard operational practices: keep software current with supported updates, restrict network access using external firewalls and policies, and adhere to the principle of least privilege.

Power Management

All Oracle Linux KVM hosts must have out-of-band management (CMC/iDRAC/iLO/ILOM) enabled and reachable. Create a dedicated account with permissions to power on, power off, and reset the server. For consistent, vendor-agnostic integration, configure IPMI over LAN (`ipmilan`) on each out-of-band interface; ensure the management controller firmware is current and set to a supported cipher/auth configuration.

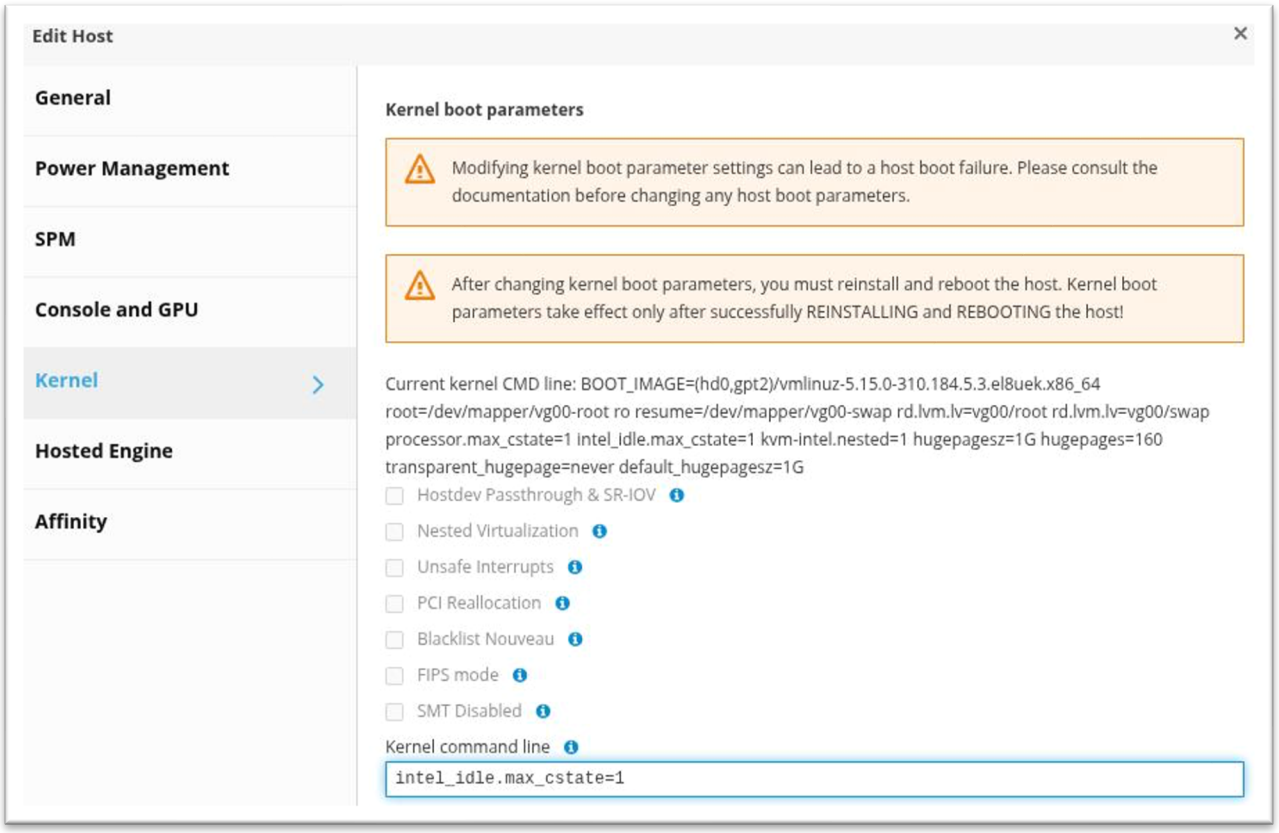
Every host in a cluster must be able to reach the out-of-band IP addresses of its peers. Place these interfaces on a routable, restricted management network and allow the required IPMI ports. The engine host does not need direct access; it delegates power checks and fencing actions to an available Oracle Linux KVM host within the cluster.

Successful power management is a prerequisite for fencing and high availability. Without working power integration, failed hosts cannot be fenced, and HA is not supported. After onboarding, validate power operations (status, on, off, reboot) from Oracle Linux Virtualization Manager for each host and retest after network or firmware changes to ensure continued reliability.

C-States

Recommendation across all clusters: disable C-states deeper than C1 to minimize wake-up latency and jitter. Limiting to C1 improves predictability for VM networking, storage I/O, live migration, and low-latency applications, with a modest increase in power consumption.

Configure C-state policies through the Oracle Linux Virtualization Manager Administration Portal to keep hosts consistent and supported. On systems using the Intel `intel_idle` driver, set `intel_idle.max_cstate=1`; on older Intel CPUs and on AMD platforms, set `processor.max_cstate=1`. Applying these kernel parameters requires host reconfiguration: migrate all VMs off the host, reboot it, and perform the restart through the Oracle Linux Virtualization Manager Administration Portal.



Complement the OS settings with firmware controls. Select a Maximum Performance power profile and, if necessary, disable deep package C-states and C1E while keeping turbo and P-states enabled unless instability is observed. Avoid extreme options such as `idle=poll`, which dramatically increase power consumption and thermals.

Virtual Machines

In Oracle Virtualization, a VM is a software-defined compute instance that runs an operating system and applications as if on dedicated hardware. VMs in Oracle Linux Virtualization Manager are backed by Oracle Linux KVM and managed through the Oracle Linux Virtualization Manager, which orchestrates CPU, memory, storage, and network resources from defined clusters and hosts.

Oracle Linux Virtualization Manager provides enterprise features for VM lifecycle and resilience, including snapshots, live migration between hosts, high availability, and scheduling with resource quotas and permissions.

Emulated Machine Version

In a new installation, new VMs appear with a Custom Emulated Machine of `pc-q35-4.0`. This legacy baseline persists because:

- Upgrades do not modify existing templates or VMs.
- During Oracle Linux KVM upgrades, there will be different Oracle Linux KVM hosts running on different QEMU versions.
- To be able to live-migrate VMs between Oracle Linux KVM hosts during an upgrade, the emulated machine is unchanged and persists in an older version.
- Changing the emulated machine will always require the VM downtime.

For new workloads on Oracle Linux Virtualization Manager 4.5, `pc-q35-7.2` is recommended.

Aligning the VM chipset with QEMU 7.2 exposes current device models and VirtIO capabilities, incorporates stability and performance fixes accumulated across releases, improves UEFI/Secure Boot handling, and enhances IOMMU/VFIO behavior for passthrough. Modern Oracle Linux and Windows guests are validated on recent Q35 generations, reducing driver quirks and avoiding legacy emulation fallbacks.

In Oracle Linux Virtualization Manager 4.5, this control is currently not available at the cluster level. Set the Custom Emulated Machine per VM and standardize through updated templates so new VMs inherit `pc-q35-7.2` by default. After an Oracle Linux Virtualization Manager upgrade (from 4.4 to 4.5 for example) plan the uplift of existing VMs during maintenance windows.

High Availability

High availability for VMs in Oracle Linux Virtualization Manager depends on three core elements:

- Host power management.
- The VM's Highly Available setting.
- A VM storage lease.

Power management must be enabled and correctly configured on all hosts so the engine can fence failed or isolated hosts decisively. Fencing prevents split-brain and authorizes automated restarts on healthy hosts, which is essential for production.

Marking a VM as Highly Available instructs the engine to restart it after host failure or planned maintenance. A VM lease provides storage-level decisions; only one host can hold the lease at a time, avoiding double-run and helping enable fast failover. The lease should reside on the same storage domain as the VM's root disk to align failure domains and reduce dependencies.

Some recovery is possible without power management, but decisions become conservative or delayed; production deployments should always treat power management as mandatory.

A guest watchdog is optional. It does not influence host-level failover but is valuable in specific scenarios:

- Triggering a controlled reset and core dump when the VM kernel crashes.
- Taking action when storage I/O stalls cause the guest to hang (for example, pausing or resetting the VM to recover service).

High Performance

When creating a VM in Oracle Linux Virtualization Manager, it can be created as Desktop, Server or a High-Performance VM. The High-Performance option configures the VM to run as close to bare metal as possible.

Do not forget that this is a virtual environment where all VMs running on the same Oracle Linux KVM host share host resources, and when configuring high-performance in a single VM, the other VMs running in the same Oracle Linux KVM host may be affected.

If planning for high performance, never oversubscribe CPUs in the Oracle Linux KVM host. By selecting High Performance, the following changes are automatically applied to the VM:

- Headless mode, serial console, and a paravirtualized RNG PCI device are enabled.
- All USB, sound card, smart card, and memory balloon devices are disabled.
- CPU Pass-Through is enabled, and VM migrations are completely disabled.
- High Availability is enabled only for pinned hosts, not for all hosts in the cluster.
- I/O and emulator threads are pinned to the first two cores of each NUMA node, but only when vNUMA is configured, otherwise, I/O and emulator threads will be free to run across all the Oracle Linux KVM CPUs.

When creating high performance VMs, note the following recommendations:

- CPUs should be pinned using one of the available policies; the default is “manual.”
- vNUMA is currently in technology preview (under development and made available for testing and evaluation purposes). Unless the number of vCPUs exceeds the number of pCPUs available in a NUMA node, or the amount of VM memory exceeds the memory available in a single NUMA node, you may ignore the recommendation.
- Huge Pages, which are valuable for Oracle Database workloads, should be configured accordingly.
- KSM must be disabled at the cluster level; this impacts all VMs in the cluster, high-performance VMs or not.
- Memory ballooning should be disabled.

CPU Pinning

In Oracle Linux Virtualization Manager, CPU pinning policies control how a VM's virtual CPUs (vCPUs) are assigned to the physical CPUs (pCPUs) of a host. The three policies—manual, dedicated, and isolated—offer different levels of control and resource allocation and are often used for performance optimization.

Manual CPU Pinning

This policy gives you granular control over where each vCPU is placed. It is a static method where you explicitly define a mapping between specific vCPUs and pCPUs. This is often used for:

- **Performance**: To reduce latency and improve cache efficiency by ensuring a VM's vCPUs always run on the same pCPUs, especially for high-performance workloads such as databases or real-time applications.

The downside to this policy is that it can be inflexible. The VM is tied to a specific host (or a set of hosts) with the specified pCPU IDs, which can complicate migration and lead to resource fragmentation.

Dedicated CPU Pinning

Unlike manual pinning, this is an automated policy where Oracle Linux Virtualization Manager handles the pCPU selection. When you choose this policy, the system:

- Automatically finds a set of physical cores that matches the VM's vCPU requirements.
- Exclusively assigns those pCPUs to the VM, meaning other VMs won't be scheduled on them.

This approach provides the performance benefits of pinning without the rigid, host-specific configuration of manual pinning. The VM can be migrated to other hosts as long as a suitable set of dedicated pCPUs is available.

Isolated CPU Pinning

This policy is a stricter version of the dedicated policy. When enabled, the Oracle Linux Virtualization Manager scheduler takes two key actions:

- **Dedicated Allocation**: Similar to the dedicated policy, it assigns a set of physical cores to the VM.
- **Host Isolation**: It prevents the host operating system itself from scheduling any other workloads on those designated pCPUs, including host processes and other VMs.

This is the highest level of CPU exclusivity and is primarily used for extremely demanding workloads that require minimal interference from the host or other VMs. It ensures that the VM has a completely "quiet" and uncontended set of cores.

To summarize, manual is a user-defined, static mapping. Dedicated is an automatic, exclusive assignment by the engine scheduler. Isolated is the most extreme form of dedicated pinning, where the host OS itself is told to stay off those CPUs, providing maximum performance and resource predictability for critical workloads.

Note that to comply with the Oracle Partitioning policy, CPU pinning must be performed through the `olvm-vmcontrol` utility. Refer to the Oracle technology paper on Hard Partitioning for details.

VirtIO Network Interface

New VMs attach a VirtIO network adapter by default. Emulated adapters such as rtl8139 (100 Mb/s) and e1000 (1 Gb/s) load without extra drivers in Windows guests but deliver poor throughput and higher CPU overhead, so they are reserved for legacy compatibility. VirtIO-net supports modern offloads and multiqueue for high bandwidth and low latency.

Oracle Linux Virtualization Manager exposes multiqueue as an on/off control and assigns queues automatically based on vCPU count:

- 1 vCPU → 1 queue
- 2 vCPUs → 2 queues
- 4+ vCPUs → 4 queues

Higher queue counts are possible via an `engine-config` parameter. Open a Service Request in My Oracle Support if increasing the number of queues is required.

Disk Interfaces

Three interface types are supported.

- VirtIO-SCSI is recommended for most workloads because it scales to many devices, supports SCSI features such as UNMAP/trim and persistent reservations, and offers multiqueue; Linux guests present devices as /dev/sdX.
- VirtIO (virtio-blk) has slightly lower per-device overhead but scales poorly because each device consumes its own PCI function; devices appear as /dev/vdX.
- IDE or SATA (depending on the selected chipset, i440fx or q35) require no extra Windows drivers but provide limited performance and features, fitting only transitional or legacy needs.

Modern Linux distributions include VirtIO drivers; Windows guests require the `virtio-win` package for VirtIO-net and VirtIO-SCSI. For further details, refer to Oracle Linux KVM VirtIO Drivers for Microsoft Windows.

VirtIO-SCSI Multiqueue and I/O Threads

VirtIO-SCSI multiqueue allows a guest to submit I/O in parallel across multiple queues, improving throughput at the cost of additional guest CPU cycles. The default is automatic; where higher I/O performance is needed, a specific queue count can be set per VM in the Administration Portal. Optimal results assume `blk-mq` in the guest (default in modern kernels) and suitable queue depth at the filesystem or database layer.

I/O threads shift block I/O work from vCPU threads to dedicated host threads, reducing contention and benefiting VMs with multiple disks.

With VirtIO-SCSI, disks attach to one controller by default. After setting I/O threads (for example, to 4) and detaching/reattaching disks so the configuration is updated, the next boot creates four PCI SCSI controllers and spreads disks across them. Each I/O thread maps to a host CPU thread, so the configured count must not exceed available host CPU threads.

Thin Provisioned or Pre-Allocated

Thin-provisioned (sparse) images allocate storage on demand and reduce capacity consumption, fitting most general workloads. Pre-allocated images reserve full capacity at creation, minimize allocation-time latency, reduce fragmentation, and typically deliver the best and most consistent I/O performance; they are preferred for latency-sensitive databases and critical applications.

Snapshot behavior is important for performance and capacity planning because snapshot layers are created as sparse images. Taking a snapshot of a pre-allocated disk freezes the base image and writes new data to a sparse top image; when the snapshot is deleted, the delta merges and the result returns to a pre-allocated layout. Long snapshot chains increase copy-on-write overhead; keeping chains short preserves performance.

VirtIO Drivers

VirtIO provides paravirtualized devices for networking, storage, memory ballooning, RNG, and serial. Modern Linux distributions (including Oracle Linux) ship these drivers in-kernel, but Windows guests require the Oracle Linux VirtIO for Windows driver package (`virtio-win`) for optimal performance and stability. Without VirtIO drivers, Windows falls back to emulated devices (e1000, IDE/SATA) that significantly limit throughput and increase CPU overhead.

Keep Oracle Linux VirtIO drivers current. New releases deliver performance improvements (multiqueue scaling, offloads, improved I/O paths), stability fixes under heavy load and backup/snapshot conditions, more features (TRIM/UNMAP, SCSI persistent reservations, modern TLS/UEFI/Secure Boot compatibility), and security updates.

Aligning driver versions with the Oracle Linux Virtualization Manager/QEMU level in use helps avoid edge cases in migrations, snapshot merges, and storage path recovery.

Required components for Windows typically include:

- **Storage**: `viostor` (VirtIO-blk) and/or `vioscsi` (VirtIO-SCSI).
- **Network**: `netkvm` (VirtIO-net).
- **Balloon**: balloon service for dynamic memory operations.
- **RNG and serial**: `viorng` and `vioser` for entropy and console.
- **Crash/health**: `pvpanic` and watchdog (if watchdog use is desired).

Installation and lifecycle considerations:

- During OS installation, load storage drivers from the virtio-win media so Windows can see VirtIO disks; post-install, add network, balloon, RNG, and other devices as needed.
- For existing VMs, update drivers via Device Manager using the latest `virtio-win` package. A maintenance window is recommended, as NIC updates can create a new Windows network interface (requiring IP configuration reapplication), and storage driver changes may prompt a reboot.
- Keep drivers and guest agents consistent across templates to ensure predictable behavior; refresh templates when new stable `virtio-win` packages are released.

Using up-to-date Oracle Linux VirtIO drivers enables the high-performance paths Oracle Linux Virtualization Manager is designed to expose, reduces crash and timeout risks during heavy I/O (snapshots, backups, live storage operations), and avoids the limitations inherent in legacy emulated devices.

Backups and Snapshots

In Oracle Virtualization, third-party backup solutions operate through snapshots:

- Create a snapshot.
- Copy the base image to backup storage.
- Remove the snapshot and merge the delta.

On file-based storage domains, sparse file growth is handled by the underlying filesystem. On block-based storage (FC or iSCSI), virtual disks are LVM LVs that grow only via LV extensions. A single host acts as the Storage Pool Manager (SPM) and performs LV extensions for thin images across all hosts.

During heavy snapshot or thin-growth activity—common during backups—the SPM can become a bottleneck; if extensions are delayed, affected VMs may pause until growth completes. Environments with frequent backup snapshots or a high proportion of thin-provisioned disks should apply the action plan in Doc ID 2727849.1 to minimize I/O stalls during volume growth and merges.

Log Collection

The standard tool for gathering diagnostics in Oracle Linux Virtualization Manager is the `ovirt-log-collector`. Run it on the engine host. By default, it will collect:

- Engine database dump.
- The engine sosreport.
- Sosreports from all Oracle Linux KVM hosts.

The command packages everything into a single archive with consistent timestamps and configuration context. This enables correlation across the engine (events, scheduler decisions, migration records, storage domain metadata) and hypervisors (VDSM, libvirt, sanlock, networking, storage paths).

Standalone sosreports miss Oracle Linux Virtualization Manager context and the engine database, making analysis more difficult. Always prefer `ovirt-log-collector`. Large environments can produce large archives. Scope collection to specific hypervisors when the issue is localized, or omit hypervisors entirely for engine-only problems:

1. Collect from two hosts (for example, a failed migration source and destination):
   ```
   # ovirt-log-collector -H KVM1,KVM2
   ```
2. Engine-only collection (no Oracle Linux KVM host sosreports):
   ```
   # ovirt-log-collector --no-hypervisors
   ```

Performance Co-Pilot

Performance Co-Pilot (PCP) is a lightweight, extensible toolkit for system performance monitoring and analysis. It provides a common framework (`pmcd` daemon and plug-in PMDAs) to collect, expose, and archive metrics from the OS, hardware, and services (including libvirt/KVM), with tools for live inspection and historical replay (`pmlogger`).

PCP is enabled by default on all KVM hosts. The metrics it collects help Oracle Virtualization Support engineers troubleshoot resource contention and other performance-related issues. All PCP metrics will be automatically collected when running sosreports or the `ovirt-log-collector`.

It is strongly recommended to keep PCP enabled and disable it only if explicitly instructed by Oracle Virtualization Support engineers.