ORACLE VM

An Oracle Technical White Paper
August 2011

Oracle VM:
Designing, Creating and Testing an Oracle VM 2.2 Environment

ORACLE

Introduction

This document describes a plan you could follow to ensure that your Oracle VM deployment is organized, repeatable and tested before you mark it ready for production. The plan is broken down into three key stages, each of which may touch many aspects of the overall deployment.

Contained within this document are a number of tips, hints, instructions and even a test plan check-sheet for you to use when planning, building and testing your Oracle VM environment. At the end of the document are several Appendices containing useful reference material related to the deployment of Oracle VM, including some reference configuration files.

Any plan has a number of stages, phases and steps that lead to the successful completion of that plan. We believe that there are three key stages to the successful deployment of Oracle VM:

- Stage 1: Designing your Oracle VM environment
- Stage 2: Creating your Oracle VM environment
- Stage 3: Testing your Oracle VM environment

Each stage is self-contained but requires the total completion of all previous steps; these stages cannot be run in parallel at any point. Attempting to do this has resulted in numerous failed or buggy deployments, especially where assumptions have been made about the ability to test certain aspects in isolation of the whole.

Stage 1: Designing your Oracle VM environment

Obviously, you have a reason for using Oracle VM and this will have influenced your hardware, storage, networking and software choices up to this point so we are not going to go into the physical installation of the hardware and network here; rather, we are going to assume that you know how to do this or have this done to your specification. Having said that, there are three key design considerations affecting your hardware infrastructure that need to be addressed at this point:

- Oracle VM Management design
- Oracle VM Shared Storage design
- Oracle VM Networking design

These designs can be addressed individually although they will influence the outcome of each of the other designs. It is entirely possible that different teams will have overall responsibility for one or more aspects of these designs, which is why it is essential that someone take overall responsibility for the Oracle VM environment design as a whole. We can tell you from experience that design decisions made outside the overall picture can have serious negative consequences later.

One classic example is for the storage team to just provide the Oracle VM team with a large collection of small LUNs rather than one large one. They have no reason for this other than they thought it best even though they had no idea about Oracle VM or what the team was trying to achieve. This can lead to much wasted space, excessive network overhead and management complexity without any actual benefit.

Oracle VM Management Design

The first thing you need to decide is if you are going to use Oracle VM manager or Oracle Enterprise Manager Grid Control to manage your Oracle VM environment; the two are mutually exclusive for Oracle VM 2 and Enterprise Manager 11g and all versions below those. Each has its own benefits and disadvantages compared to the other and these should be considered carefully before making your decision. The rest of this document assumes you are using Oracle VM manager but all sections are equally applicable to Grid Control; if you are using Grid Control just substitute that for Oracle VM manager wherever you see that mentioned in the remainder of this document.

Oracle VM Management Topology

Oracle VM manager creates Oracle VM guests, allocates resources in the storage pool (discussed later), starts, stops and migrates Oracle VM guests. Oracle VM guests will remain running if Oracle VM manager stops running temporarily; Oracle VM guests can still be started, stopped and migrated without Oracle VM manager using the Xen Hypervisor command line interface on the Oracle VM servers, hence, the continued operation of Oracle VM guests is not dependent on the operational status of Oracle VM manager.

Given the relationship between the Oracle VM manager and the Oracle VM guests, there are many approaches to designing a management topology; Oracle VM manager can control multiple server pools, even across a WAN connection. Commonly, the Oracle VM manager is deployed as an Oracle

VM guest of the same server pool it is managing to make it highly available. However, the Oracle VM manager is often installed on a separate physical server rather than a VM – the choice is really up to you. The options are:

- Centralized management deployed as an Oracle VM guest on a server pool (Figure 1)
- Centralized management deployed on an independent physical server (Figure 2)
- Distributed management deployed as an Oracle VM guest on each server pool (Figure 3)
- Distributed management deployed on independent physical servers (Figure 4)

Centralized management deployed as an Oracle VM guest allows Oracle VM manager to be part of a highly available cluster taking advantage of Oracle VM high availability feature while providing a single point of maintenance.

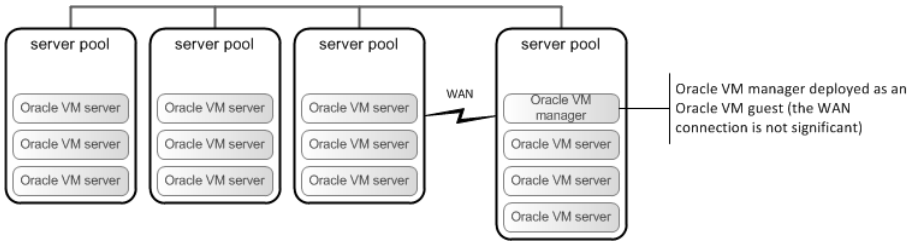


Figure 1: Centralized management deployed as an Oracle VM guest

Although Oracle VM manager would not be highly available, centralized management using a physical server that is independent of any Oracle VM environment is also an acceptable solution. Figure 2 illustrates the concept. Note that it does not matter which network the independent server hosting Oracle VM manger as long as it can reach all Oracle VM servers and guests being managed.

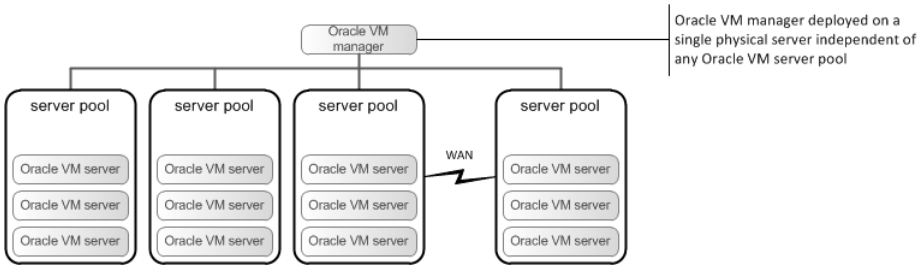


Figure 2: Centralized management deployed on an independent server

The next two examples of Oracle VM management topology are also fine, but are less desirable since either of the schemes is harder to maintain as far as patching and consistency are concerned. Oracle does not recommend either of these management topologies, but they are supported. If you decided to use a distributed management scheme, then Oracle would recommend the following since it takes advantage of Oracle VM's high availability feature.

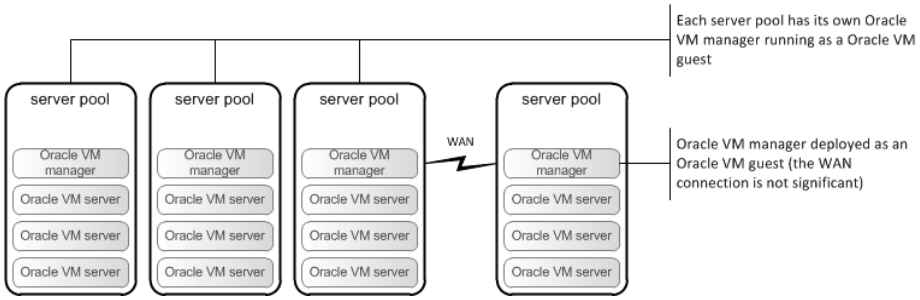


Figure 3: Distributed management topology deployed as Oracle VM guests

The least desirable management topology is one in which each server pool is managed by its own Oracle VM manager running on a physical server that is independent from any Oracle VM environment as shown in Figure 4.

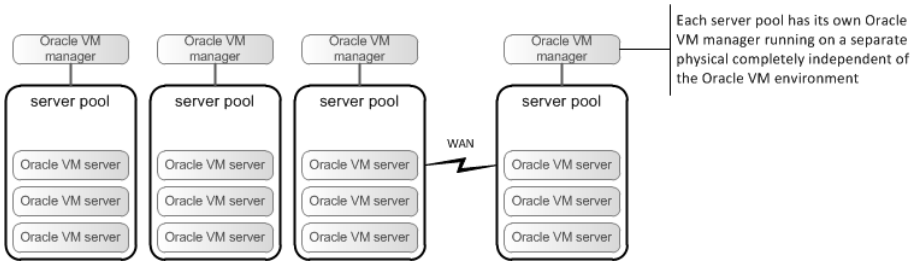


Figure 4: Distributed management topology

Management Network

The next thing to decide is how Oracle VM manager is going to communicate with your Oracle VM servers. The options are:

- In-band network management
- Out-of-band network management

In-band management, illustrated in Figure 1, is pretty typical and simply means the IP address for the Oracle VM manager is on the same subnet as the primary network interface or virtual bridge. In other words, the IP address for the Oracle VM manager is on the same VLAN used by the typical end user to access Oracle VM Guests and other servers.

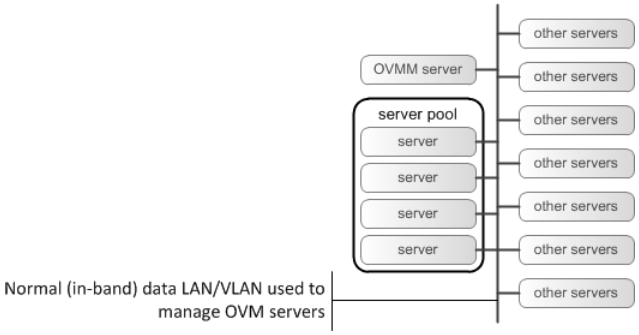


Figure 5: In-band network management

Out-of-band network management, illustrated in Figure 2, means the IP address for the Oracle VM manager is on a network interface or virtual bridge that is dedicated to just managing servers and is usually restricted to the systems administrators only. This would be a VLAN that is not accessible to the typical end user.

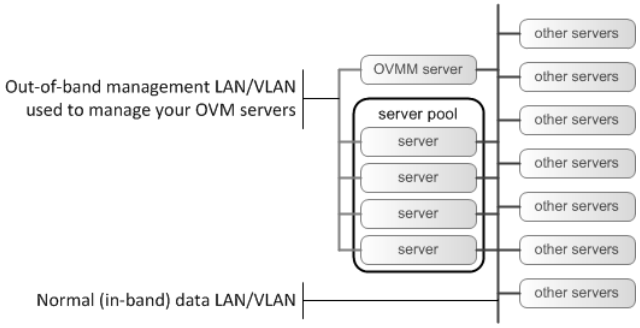


Figure 6: Out-of-band network management

System Patching

Oracle VM servers are able to update themselves to the latest patches and updates via the up2date facility. By default, this facility expects to connect to the Unbreakable Linux Network (ULN) via the Internet and you are able to register your Oracle VM servers with the ULN using your ULN ID and your Oracle Customer Support Identifier (CSI). Doing this will allow you to see any outstanding patches or updates either from the Oracle VM server directly by running the up2date command or from the ULN itself (log in at linux.oracle.com).

It may not always be desirable to have your Oracle VM servers connected to the Internet so it is possible to set up local YUM repositories, registered as such with ULN, to download the latest updates and patches into a local store and to configure your Oracle VM servers to go to this local YUM repository to check and patch/update themselves, across your private WAN as desired. Using local YUM repositories requires manual reconfiguration of the up2date facility on each of your Oracle VM servers that will use them. Clearly, if you are to use local YUM repositories to update and patch your Oracle VM servers you will need to incorporate these into your management (and likely, network) design. You may even chose to extend these local YUM repositories to manage your local Oracle Linux patches and updates in addition to Oracle VM, as they both use the same systems. Instructions on how to do this can be found at:

[http://www.oracle.com/technetwork/topics/linux/yum-repository-setup-085606.html]

Oracle VM Server Pool Design Considerations

Oracle VM Manager is supplied with an embedded Oracle database [Oracle XE] that is somewhat limited in its capacity. Enough for most environments, this can still be replaced with a larger Oracle Database (even up to and including a RAC cluster) depending on your requirements, operating considerations and number of Oracle VM components (servers, pools and VMs) you plan to manage.

Oracle VM servers can perform one or more of three duties within a server pool:

- Server pool Master: one and only one node will always be the server pool Master. If the designated server pool Master fails, one of the remaining nodes in the server pool will automatically assume the role and responsibilities. The server pool Master decides upon which node any new VM is started and coordinates the HA facility of the server pool in High Availability mode. This is the only duty that is automatically reassigned in the event of a failure.
- Utility Server: one or more nodes in each server pool can be designated as Utility Servers, which perform the I/O intensive tasks for the server pool such as VM creation, VM Template creation, cloning, etc.
- Virtual Machine Server: the remaining nodes in the server pool can be marked for the exclusive use of running existing VMs.

These roles need to be designed according to anticipated demand and operational conditions. With only a few Oracle VM servers in a server pool it is quite common to allocate every node the duties of Utility and VM Server so that every node can perform any job. With more than five Oracle VM servers in a given server pool it is quite common to dedicate one node as the Utility Server and the remaining nodes as VM Servers [of course, one random node is always the server pool Master starting as the first node you add to the server pool]; in this way, the creation of VMs, Templates and clones does not affect the performance of the VM Servers (over and above the load it applies to the shared network and storage subsystems themselves). You will need to incorporate these considerations into your management design.

Server pools are maintained as clusters internally, using OCFS2 cluster and locking functions even if no OCFS2 file systems are used (i.e. in a totally NFS based shared Storage Repository configuration). To ensure the server pools maintain their cluster integrity they perform a number of I/O writes across the shared Storage Repositories and Networks. With Oracle VM Server 2.2, these include multiple writes across every shared Storage Repository every second; if more than a threshold number of these writes fail or timeout then the nodes will fence themselves from the server pool by rebooting. A detailed explanation of the clustering technologies used can be found in the Oracle whitepaper: Oracle VM – Creating & Maintaining a Highly Available Environment for Guest VMs [http://www.oracle.com/us/026999.pdf].

In order to reduce the overhead associated with these cluster file system heartbeats you should ensure that you have the minimum number of shared Storage Repositories either by using large Storage Repositories (up to 16TB each) or using network storage directly from the VMs (such as direct LUN or NFS storage from the guest OS). As you cannot split virtual disks across Storage Repositories or create virtual machines that create multiple virtual disks in different Storage Repositories and given that Oracle VM Manager 2.2 automatically creates new virtual disks and VMs on the next Storage Repository with the greatest available free space, it is generally a bad idea to use many small Storage Repositories as they create an I/O overhead as well as inefficient use of the underlying storage itself.

Oracle VM Server Synchronization

In order to keep the cluster synchronized it is highly advised to synchronize the clocks of the Oracle VM servers in each server pool. The best way to do this is by using a Network Time Protocol server

(or servers) and configuring the NTP service on each Oracle VM server to point to those same NTP servers. There are a number of publicly available Internet NTP servers or you can point to one inside your LAN; you can even configure the Oracle VM Manager Oracle Linux host to be an NTP server (as long as it's running on a physical server, as all VMs are prone to clock-drift to a lesser or greater degree).

Hard Partitioning for Oracle Licenses

If you are planning to use Oracle VM to provide Hard Partitioning for Oracle per-processor licensed products you will need to design how the individual cores in each Oracle VM server will be allocated to those hard partitions. Oracle licensing only recognizes Oracle VM virtual machines as hard partitions if their configuration files [vm.cfg] have been directly edited to include specific instructions to pin the virtual CPUs to actual and specifically identified cores within the host processor. The specifics for hard partitions along with the requirements can be found here:

[http://www.oracle.com/technology/tech/virtualization/pdf/ovm-hardpart.pdf]

When planning your Oracle VM server pools you should take into account the type of machines and storage that you intend to use for the workloads in each Oracle VM server pool. One important consideration for any Oracle VM server pool is that every processor in the server pool should be the same; in fact, the documentation states that every hardware server in a server pool needs to be identical. In reality, the requirement is based on the processors themselves to ensure that no corruption can occur within running VMs if they are moved to another Oracle VM server within the server pool. For example, Oracle databases make extensive use of advanced chipset features including SMMD instruction sets based on what they find available when they start up – if they try to execute one of these instructions on a CPU that does not have them the database will crash.

The production level check to see if non-identical servers can be used in a server pool is:

- Try to create the server pool containing all identical and non-identical servers
- Try to run a Live Migration between the Oracle VM servers in the server pool

If both these tests succeed then it can be considered a supported configuration in the event you need to contact Oracle support. There are settings that can be modified to turn off these checks but such modified server pools are unsupported configurations.

Oracle VM Shared Storage Design

Storage concepts surrounding Oracle VM server and Oracle VM manager implementations are probably the single most important thing to understand, yet the most complex aspect of the overall design.

There are only a few things that need to be taken into account with storage, but there are a multitude of different options and concepts that need to be thoroughly understood before designing a server pool. Basically, the following things need to be considered:

- Presenting a shared storage repository to Oracle VM servers

- Presenting additional "local" and/or "shared" storage to Oracle VM guests

Storage repository

A storage repository is simply a directory structure that resides on a single disk (or NFS export) that can be seen by all Oracle VM servers controlled by Oracle VM manager. The storage repository contains the configuration files and image files for the Oracle VM guests, "local" and "shared" virtual disks (sparse files) and any other resources needed by the hypervisor to run and manage Oracle VM guests.

In order to create a resource pool of processors, memory and storage using Oracle VM you simply install Oracle VM on each physical server and allocate at least one shared storage device as a Storage Repository to be shared equally between each of the Oracle VM servers. You tell all the Oracle VM servers that they are a shared pool by creating a Server Pool using Oracle VM Manager. You can have any number of Server Pools but each one should contain no more than 32 Oracle VM servers. All the Oracle VM servers in a Server Pool knows about and talks to all the others to keep a consistent map of what they are doing. In this way, new VMs are started on the Oracle VM server within the pool that has the most available resource. If High Availability, HA, is enabled for the Server Pool, VMs that fail for any reason will be automatically restarted – on a different Oracle VM server if required. Additionally, running VMs within a Server Pool may be moved between Oracle VM servers within that pool without interruption through the use of Live-Migration.

You must decide how you want to present storage to each VM server. Options include network file system (NFS), and block level storage such as direct attached storage (DAS), SCSI over Ethernet (iSCSI) or SCSI over Fibre Channel (FCP). The figure below illustrates using block level (FCP or iSCSI) disk device to create the storage repository.

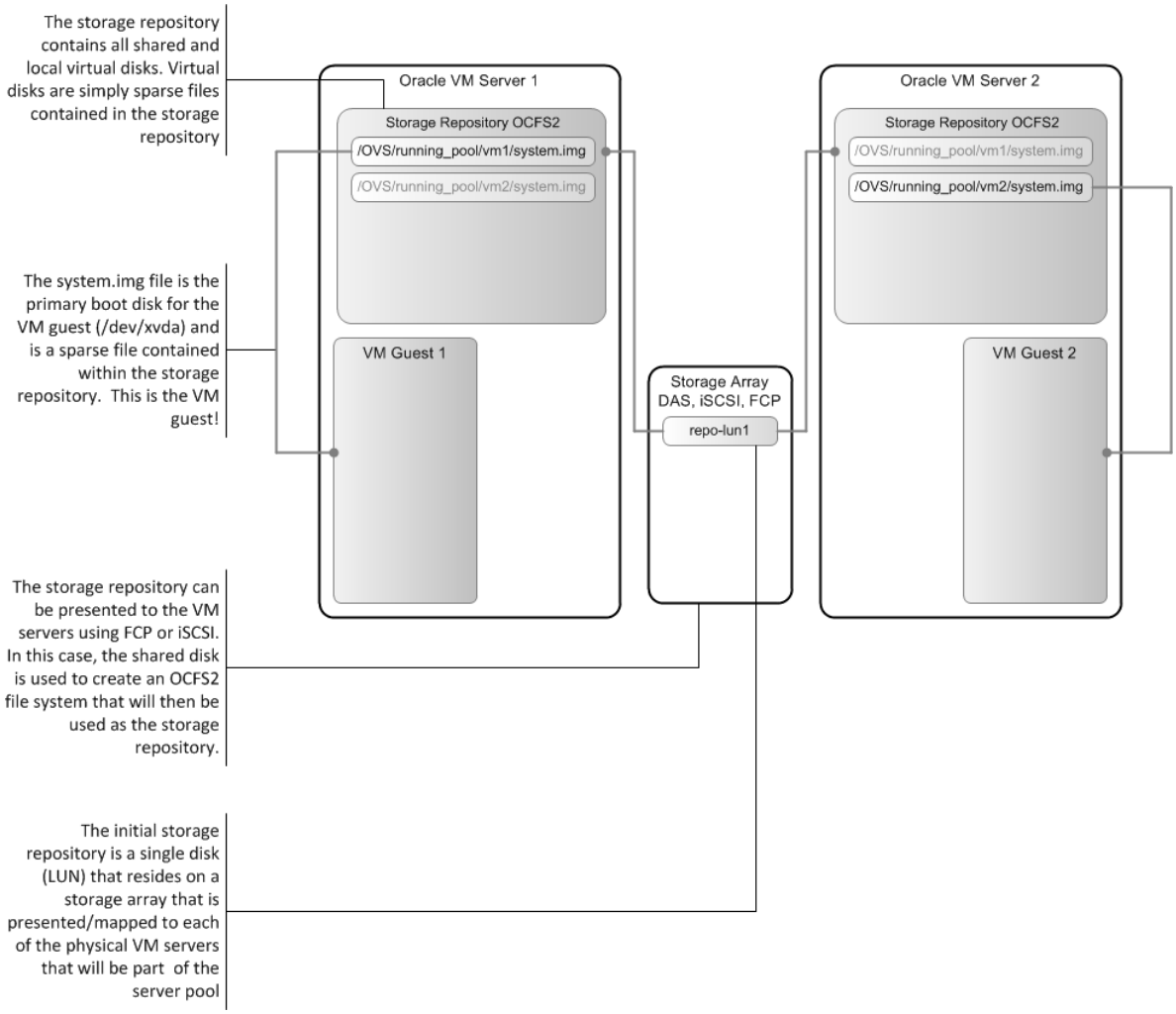


Figure 7: Storage repository using FCP (or iSCSI)

A single disk is presented to each Oracle VM server from a storage array and then OCFS2 is used to create the file system that will be managed by Oracle VM manager. Note that OCFS2 must be used to create a cluster aware file system when using either iSCSI or FCP

Shared storage repositories can also be created using NFS as shown in the figure below. Note that NFS inherently cluster aware and does not use OCFS2.

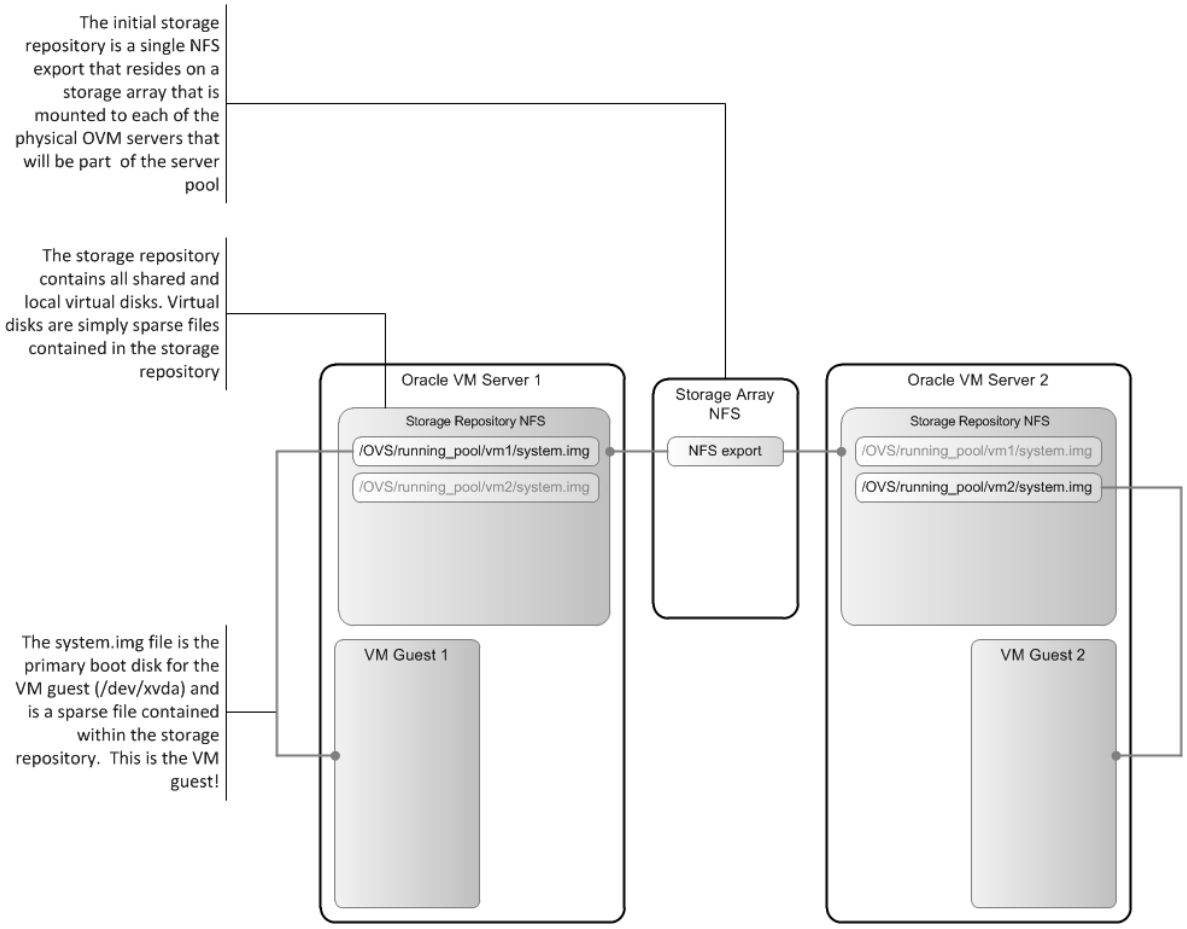


Figure 9: Storage repository using NFS

Here are some additional considerations to ensure this all works:

- For FCP devices you need to ensure every LUN is presented to every Oracle VM server and configure the Zones to ensure that every possible path is exposed and weighted appropriately; if you ever encounter a path that one Oracle VM server can see but that another cannot, you will have a problem when the system is running. You will also need to mask any LUNs that will be kept for 'private' storage for individual VMs or will be otherwise unused by the Oracle VM Server Pool as shared Storage Repositories. If you are planning to use SAN boot for the Oracle VM servers themselves you will need to set this up too [ensure you use the `linux mpath` install option when installing such Oracle VM servers].
- For iSCSI or NFS devices you need to decide if you will use dedicated network LAN or VLANs for the storage or share the configures LANs and VLANs with other traffic. Network bonding or teaming can be used to increase the bandwidth to the storage and/or add resilience. Using dedicated

management networks or VLAN connections with Oracle VM 2.2 requires direct configuration of the Oracle VM servers' network settings.

- For DAS devices you need to ensure they can be access simultaneously by the number of processors you are planning to turn into Oracle VM servers; shared SCSI, for instance, can only be shared by a maximum of two servers.

Presenting Virtual and Physical Disks to Oracle VM Guests

After you create the storage repository and after you create some Oracle VM guests, you may want to present additional disks to one or more of the Oracle VM guests.

There are three different types of disks and ways of presenting them to VM guests:

- Shared virtual disks (managed by Oracle VM manager)
- Local virtual disks (managed by Oracle VM manager)
- Physical disks (managed at OS level)

Shared Virtual Disks

The following figure illustrates shared virtual disks and their relationship to the storage repository.

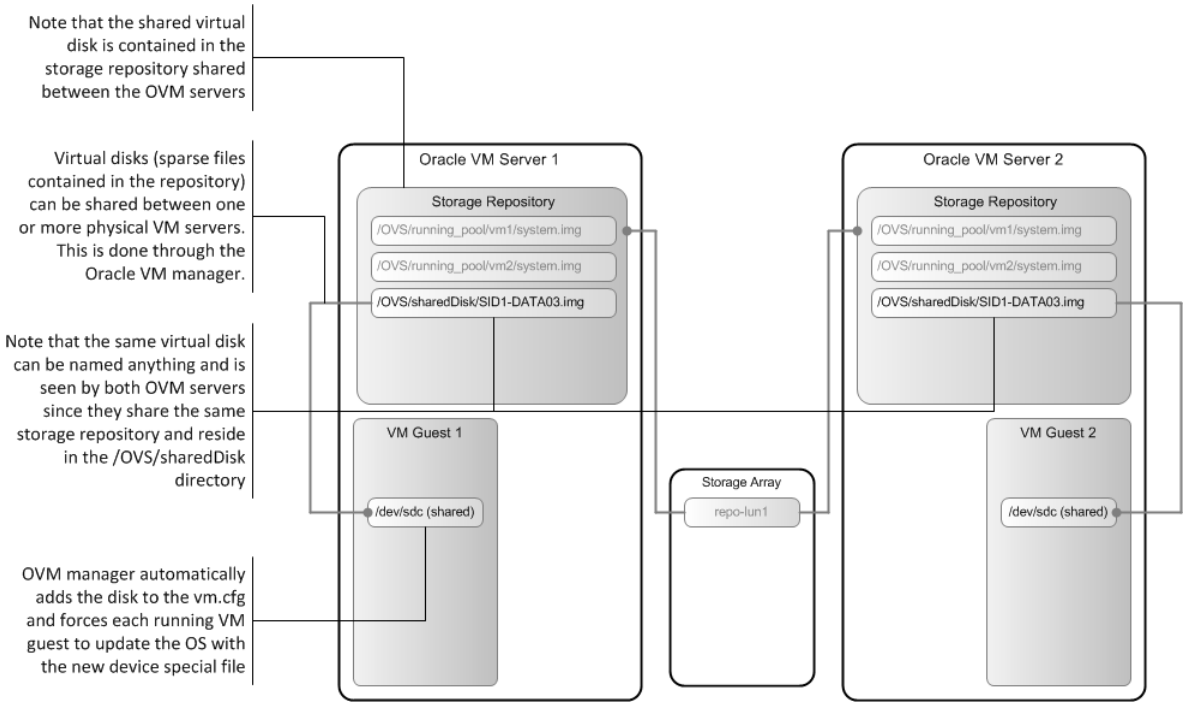


Figure 10: Shared virtual disks managed by Oracle VM manager

Shared virtual disks are created using Oracle VM manager and are really just sparse files that reside within the storage repository under /OVS/sharedDisk directory. These "disks" can be seen by multiple Oracle VM guests and are assigned to specific guests using Oracle VM manager. Shared

virtual disks are accessible by all Oracle VM servers by virtue of residing in the storage repository; therefore Oracle VM guests can access them no matter which Oracle VM server they are running. Notice in the illustration that there is only one "shared" disk in the storage repository, but each guest can see the same "disk".

Local Virtual Disks

The following figure illustrates local virtual disks and their relationship to the storage repository.

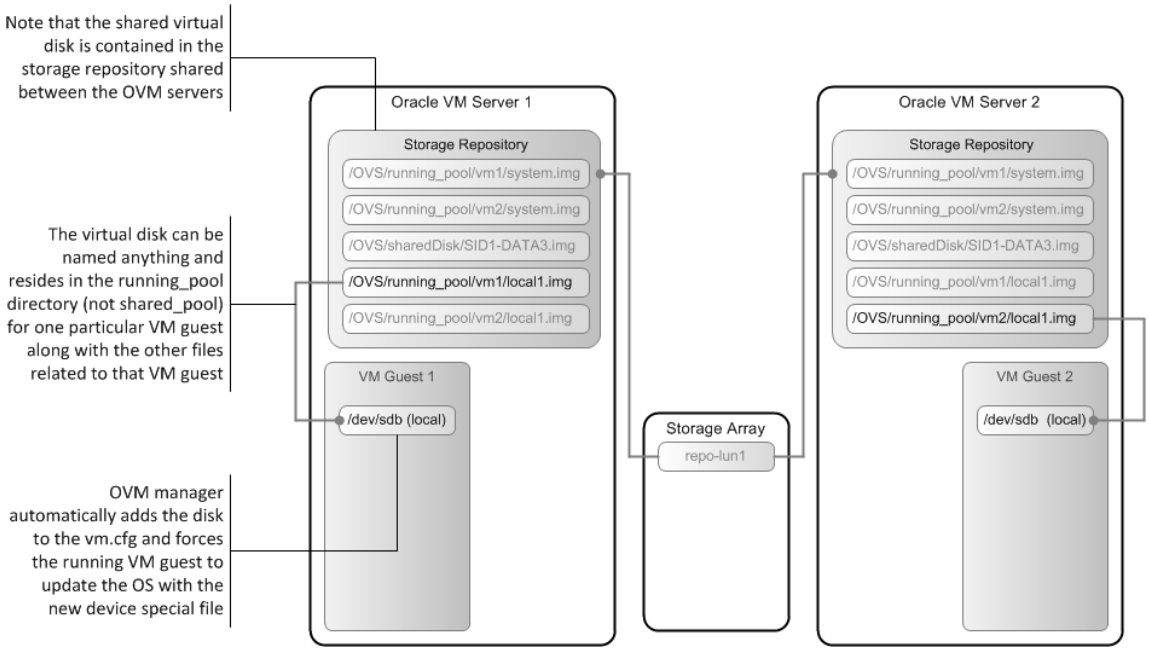


Figure 11: Local virtual disks managed by Oracle VM manager

Local virtual disks are created using Oracle VM manager and are really just sparse files that reside the storage repository under /OVS/running_pool/<vmguest> within the directory containing the files associated with a particular Oracle VM guest. Notice in the illustration that there are two "local" disks in the storage repository, but each guest can only see their assigned "disk".

Local virtual "disks" are dedicated just to a single Oracle VM guest but are accessible by all Oracle VM servers by virtue of residing in the storage repository, therefore each Oracle VM guest can access them no matter which Oracle VM server they are running.

Physical Disks

Physical disks are not part of the storage repository and are LUNs or physical disks created or made available from an external storage array or direct attached storage (DAS). These also need to be made available to all Oracle VM servers so Oracle VM guests that need to access them can do so no matter which Oracle VM server a particular guest is running. Once the disks have been presented to all Oracle VM servers, then they need to be added to the vm.cfg file of each Oracle VM guest that needs to access the shared disk.

The following figure illustrates shared physical disks and their relationship to the Oracle VM guests outside of the storage repository (not managed by Oracle VM manager).

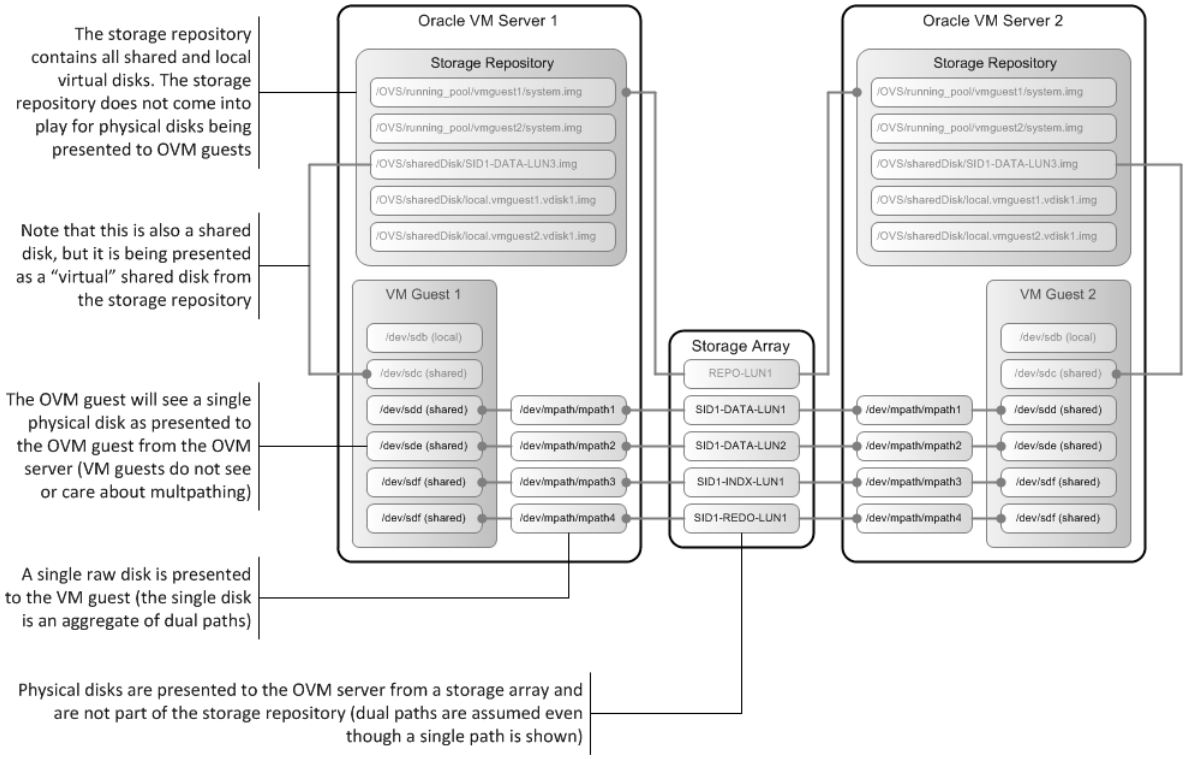


Figure 12: Shared physical disks not managed by Oracle VM manager

Physical disks can also be presented from the storage array as "local" disk. Local physical disks still need to presented/mapped to all Oracle VM servers, but you will only manually add the disk to the vm.cfg file of the particular Oracle VM guest that needs to access the disk as "local".

Basic Sizing Considerations for a Storage Repository

Now that you are familiar with the role of the storage repository and the differences between virtual disks and physical disks, you can think about the size of the disk space needed for the storage repository. The storage repository contains all the files associated with each Oracle VM guest. Sizing requirements will vary widely depending on your specific implementation, but the following table should provide a general idea of the information you will need to gather in order to calculate a repository size that will fit your needs:

DIRECTORY STRUCTURE OF THE STORAGE REPOSITORY

DIRECTORY	CONTENTS OF DIRECTORY	CONSIDERATION FOR SIZING
/OVS/iso_pool	Contains ISO images of operating systems that can be used to install on newly created OVM guests. These would be used instead of	Determine if you will be using ISO images for installs of operating systems, how many different images you will want to import and

	templates or physical media. (see Oracle VM manager user's guide)	the size of each ISO file.
/OVS/publish_pool	The location of all files associated with each Oracle VM guest image. This includes the system.img file (the sparse file containing the virtual OS for the guest image) and any "local" virtual disks. This is the same as running_pool, but for OVM guests that can be managed in OVM manger by anyone. (see Oracle VM manager user's guide)	Determine how many OVM guests you will have running on all servers in the server pool and an approximate file size needed to contain each OVM guest. Also determine how many "local" virtual disks you will need for each OVM guest and approximate size for each file.
/OVS/running_pool	The location of all files associated with each Oracle VM guest image. This includes the system.img file (the sparse file containing the virtual OS for the guest image) and any "local" virtual disks. This is the same as publish_pool, but for OVM guests that can be managed in OVM manger by anyone. (see Oracle VM manager user's guide)	Determine how many OVM guests you will have running on all servers in the server pool and an approximate file size needed to contain each OVM guest. Also determine how many "local" virtual disks you will need for each OVM guest and approximate size for each file.
/OVS/seed_pool	The location for all OVM guest templates (see Oracle VM manager user's guide)	Determine how many standard templates you need to quickly deploy additional OVM guest images and
/OVS/sharedDisk	The location of all "shared" virtual disks for all OVM guest images (see Oracle VM manager user's guide)	Determine how many "shared" virtual disks you need for any clustered OVM guests you will be running on the server pool and an approximate file size.

All of the concepts are covered in more detail in the Oracle VM Manger User's guide and the Oracle VM Server User's guide if you need more information about the directories contained in the storage repository.

There are many variables involved that will impact the size of your repository – too many to cover in this particular document, but an example of calculating size would probably be a good idea. As a very rough guide to calculating the size requirements, let's assume the following in a very simplistic scenario:

SIMPLE EXAMPLE OF CALCULATING REPOSITORY SIZE

QUANTITY	OBJECT	UNIT SIZE (MB)	EXTENDED SIZE (MB)
3	Oracle VM servers in a server pool	N/A	N/A
6	Oracle VM guests running any supported operating systems	12,288	73,728

1	Additional 36 gigabyte local virtual disk per Oracle VM guest	36,864	221,184
1	Additional 36 gigabyte shared virtual disk accessible by all six VM guest images	36,864	36,864
10	Raw physical disks per Oracle VM guest to be used by Oracle databases for data files, redo logs, etc (not counted because they are not part of the storage repository)	N/A	N/A
3	Oracle VM guest templates for quick deployment of new guest images	12,288	36,864
2	ISO images of an operating system such as Windows	430	960
6	Additional space to allow for the creation of 6 more Oracle VM guests (including 1 additional local virtual disk per guest)	49,152	294,912
Total repository size needed (rounded up to nearest gigabyte)			664,512 (MB) 650 (GB)

So, in this very simple scenario, you would need to create a 650 gigabyte (plus overhead for file system metadata) LUN/Disk or NFS file system and make it available to all three Oracle VM servers.

Using DM- Multipath with the storage repository

DM-Multipath can be used for shared disks being presented to the Oracle VM servers using DAS, iSCSI or FCP.

Using DM- Multipath with Oracle VM guests

There is usually some confusion over how device mapper Multipath (DM-Multipath) is incorporated into the operating system contained within VM guests. The short explanation is that DM-Multipath is not used on the operating systems contained within VM guests because the underlying Oracle VM server (physical server) is already providing multiple paths to a single disk using DM- Multipath. Therefore, VM guests will only see and use single paths even if those single paths represent multiple physical paths aggregated into the single "physical" disk that is being presented by the Oracle VM server to the OS contained within the VM guest.

Section 5.4 of the Oracle VM Manager User's Guide explains how to present physical disks to the VM guest using vm.cfg. Please note that the section "5.4 Managing Shared Virtual Disks" in the guide uses the term "virtual" when in fact the term "virtual disks" only refers to virtual disks created using the Oracle VM Manger, which in turn are sparse files

DM-Multipath is relevant only to Oracle VM servers (physical server), and does not pertain in any way to Oracle VM guests. Just to reinforce a concept discussed earlier, DM-Multipath is only relevant to "physical" disks, not "virtual" disks created within the storage repository and managed by Oracle VM manager. So, the disks used to create the storage repository itself might use DM-Multipath to provide protection from failed controllers but DM-Multipath will not pertain to virtual disks, which as the reader might recall, are simply sparse files contained within the storage repository.

The key to understanding the relationship between DM-Multipath and Oracle VM guests is illustrated in the following two figures. The first figure shows a single disk being presented to a single VM guest. Note that you do not attempt to set up DM-Multipath on the VM guest operating system; the OS running inside the VM guest does not need to duplicate the hardware path failover mechanism already being handled by DM- Multipath on the physical Oracle VM server.

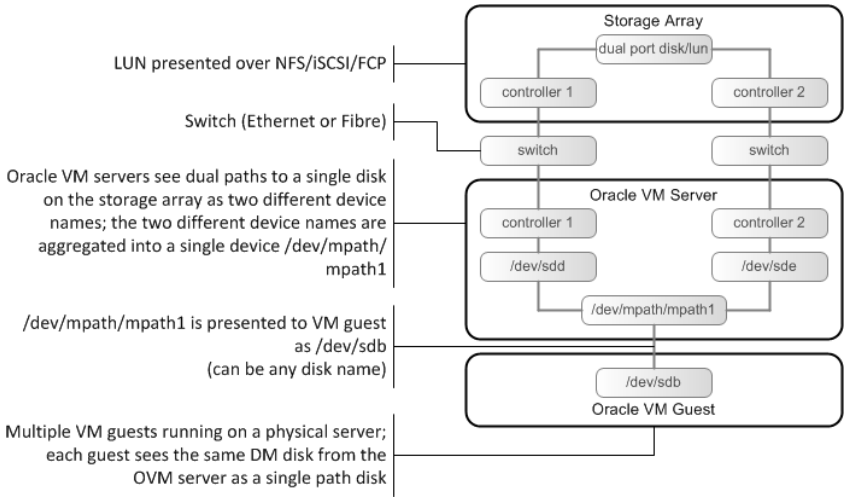


Figure 13: DM-Multipath with a single VM guest

The second figure shows the same single disk being presented to multiple Oracle VM guests as a shared physical disk. Any disks that you want to present to Oracle VM guests as shared will need to be added to the vm.cfg file for each of the Oracle VM guests.

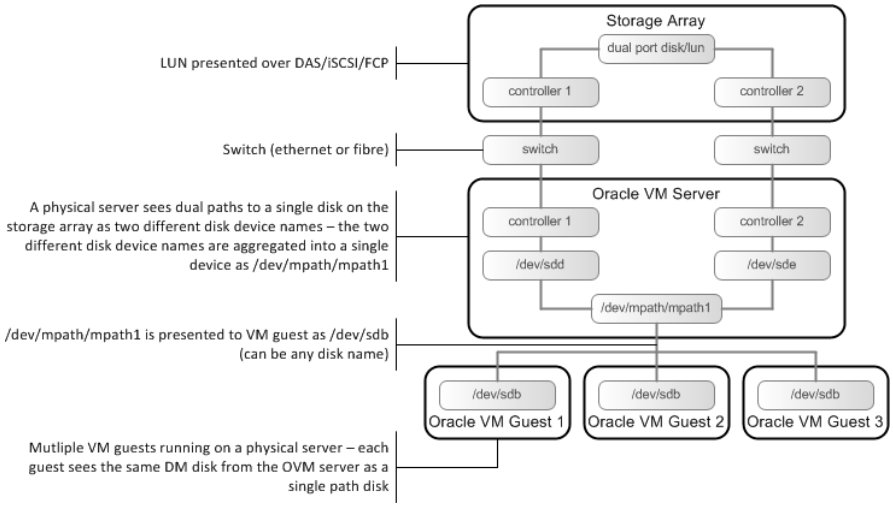


Figure 14: DM-Multipath with multiple VM guests

Best Practices for Storage

The following subsections discuss some Oracle recommended best practices related to storage on your Oracle VM clustered environment.

File Systems for Shared Virtual Disks on Oracle VM Guests

There are some very subtle differences between the various scenarios where it makes sense to use OCFS2; you may want to review this section more than once to get it clear in your own mind. The file system and the way you use shared virtual disks presented to Oracle VM guests will depend on the storage repository:

- Don't use OCFS2 on virtual disks being presented to Oracle VM guests if the storage repository is formatted using OCFS2
- Do use OCFS2 on virtual disks being presented to Oracle VM guests if the storage repository is using NFS

If you decided to create a storage repository using FCP, iSCSI or DAS, then you would have used OCFS2 for the file system of the storage repository. The following figure illustrates shared virtual disks being presented to multiple Oracle VM guests from a storage repository that is formatted using OCFS2.

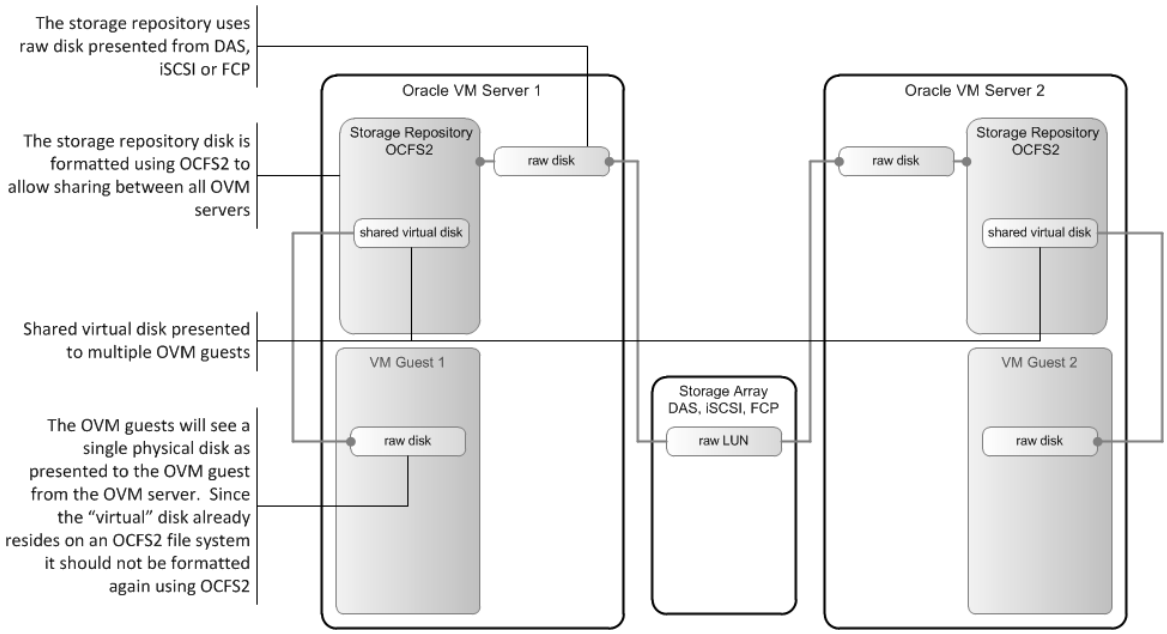


Figure 15: Shared virtual disk from an OCFS2 formatted storage repository

The Oracle VM guests will see a single physical disk as presented from the Oracle VM server. Since the shared "virtual" disk already resides on an OCFS2 file system it should not be formatted again using OCFS2 on the guest operating system. So, shared virtual disks should only be used by applications in the Oracle VM guest image such as databases that can manage raw disk.

If you decided to create a storage repository using NFS, then you would not use OCFS2 for the file system of the storage repository. The following figure illustrates shared virtual disks being presented to multiple Oracle VM guests from a storage repository that is using NFS.

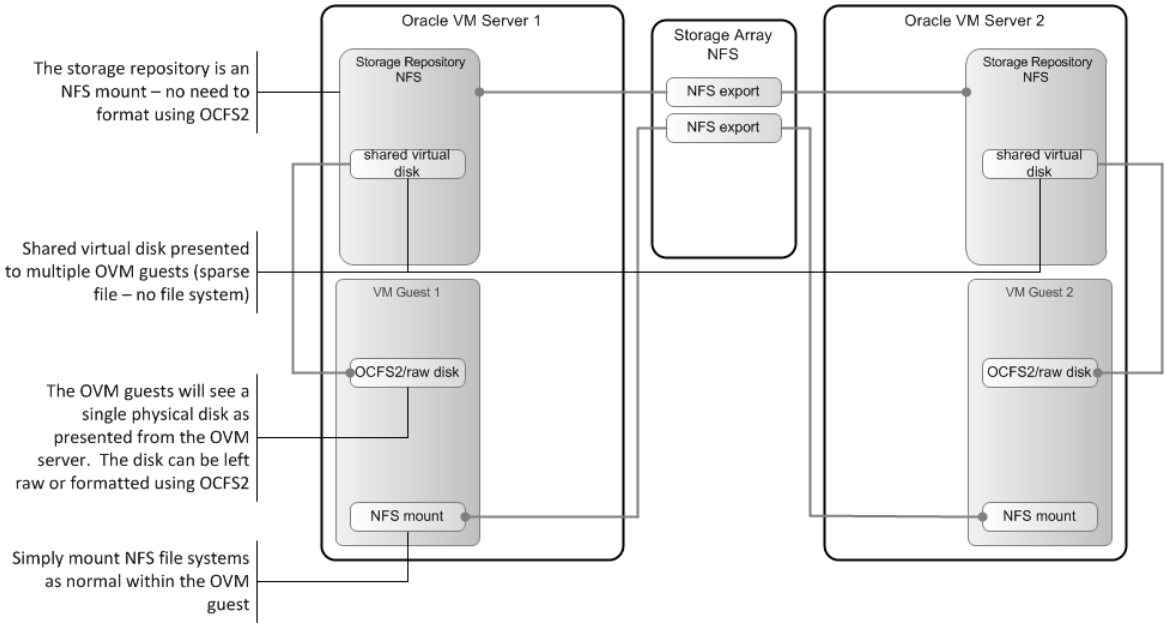


Figure 16: Shared virtual disk from an NFS storage repository

NFS is special in that the volumes being presented by NFS can be any file system on the disk, being solely managed by the NFS servers themselves. As NFS clients, the Oracle VM servers just see the contents of the volumes in exactly the same way as they see those volumes using OCFS2. NFS can be very useful for utilizing specialized storage devices such as ZFS or even collections of distributed storage [such as the unused space of desktop servers] or even providing complex and flexible storage topologies such as geo-clusters or DR; its performance is also very good and it is used in many production Oracle VM environments.

File Systems for Shared Physical Disks on Oracle VM Guests

When using FCP, iSCSI or DAS as shared storage you will use the OCFS2 clustered file system to create storage volumes that can be consistently shared between all the Oracle VM servers. OCFS2 is the standard Linux clustered file system and, as such, supported not only by Oracle VM but any standard Linux kernel.

The following figure illustrates shared storage being presented using FCP, iSCSI or DAS completely unrelated to the storage repository.

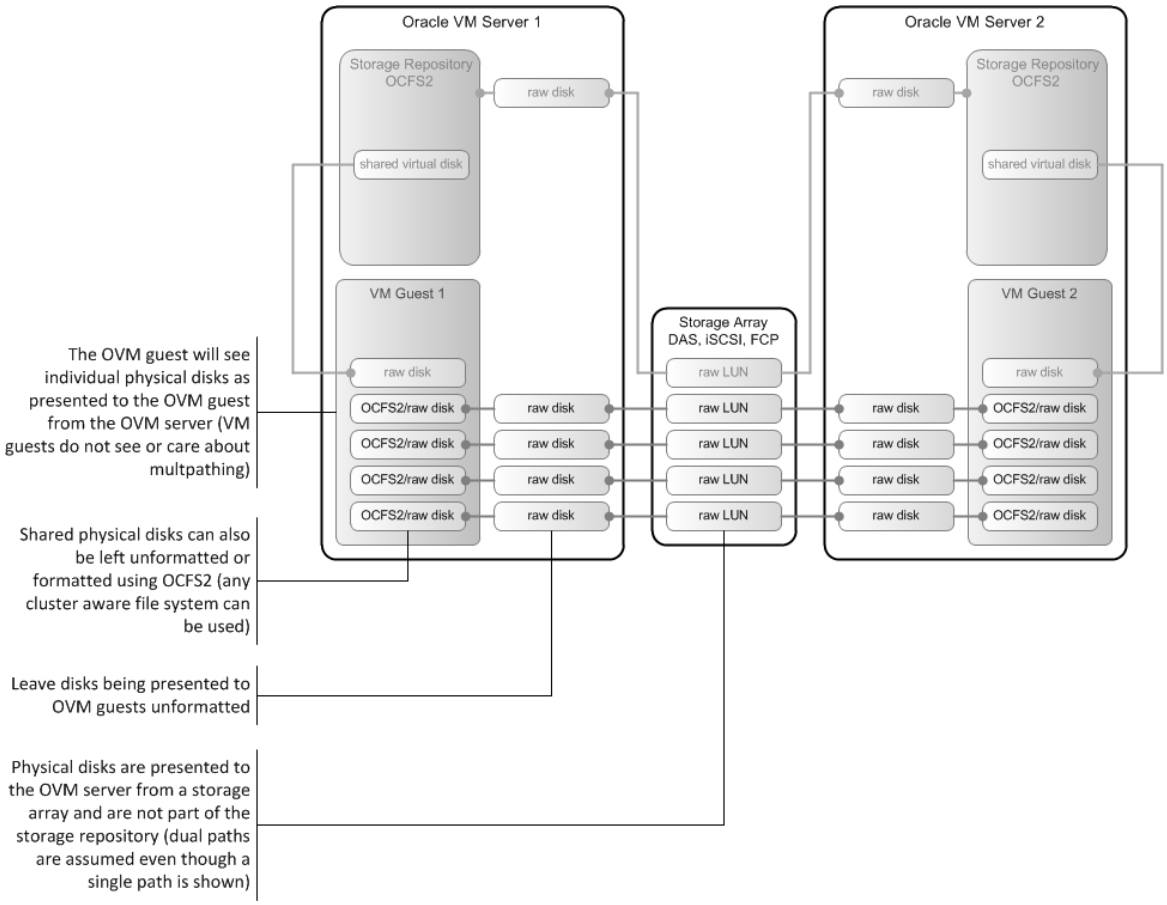


Figure 17: Shared physical disks being presented to Oracle VM guests and formatted as OCFS2

LUNs are presented to the Oracle VM servers as raw disk and the Oracle VM guests will see shared physical disks presented from the Oracle VM server. It is important to note that the disks presented to the Oracle VM guests should remain unformatted. The Oracle VM guest will see the shared physical disks as unformatted, so any cluster aware file system can be used on the guest operating system including OCFS2.

Oracle VM Networking Design

The default configuration for any new Oracle VM 2.2 server is to use one NIC as the management interface (specified during installation), which will be given an IP address [either manually or via DHCP] and then it and every other NIC will be configured as a separate bridge for VMs to connect to. The first bridge will be called xenbr0, the next xenbr1 and so on. Xenbr0 will share the same network as the management traffic. When you create a VM, you can give it virtual NICs and link each of those vNICs to bridges to establish the physical LANs they will participate in. An Oracle VM server with a single NIC would have only a single bridge, xenbr0, and all the management traffic and all the VMs would use that same NIC to talk to the LAN [if network storage is being used that traffic would also share that single NIC and LAN connection].

The default network configuration can be replaced with a specific, customized configuration including all the networking options available in a modern Linux kernel; including bonding, teaming and VLANs, with all the modes and options. There are three key considerations when creating customized networks for your Oracle VM servers:

- Ensure that every single Oracle VM server in a Server Pool has exactly the same bridge configuration for the VMs. Every bridge available to the VMs must have exactly the same name on every single Oracle VM server and connect to exactly the same LAN and VLAN in the network – ideally the same bandwidth and resilience configuration should be applied too. If this is not done correctly, individual VMs will have random and serious networking problems when running.
- Ensure that every Oracle VM server's management traffic can pass unhindered between the Oracle VM server and the Oracle VM Manager. Every Oracle VM server in a Server Pool must be able to see every other Oracle VM server in the Server Pool and communicate freely. If you are not using DNS for the management network you must ensure that every Oracle VM server in the Server Pool is listed in the /etc/hosts file of every Oracle VM server in the pool. Additionally, ensure that 127.0.0.1 only points to localhost, and not the hostname, in every /etc/hosts file [this can be set wrongly if you install Oracle VM Server without specifying a DNS server].
- Ensure that any storage networks are configured so that the storage is always visible and available from every Oracle VM server in the Server Pool. Ideally you should configure identical bandwidth and resilience options on each Oracle VM server in the Server Pool.

VLANs deserve particular attention to detail; one of the most common problems with networking is that not every switch in the LAN/WAN has been configured correctly for VLANs. Bad VLAN configurations can appear to work but cause intermittent or weird problems and these are exacerbated in virtual environments where a VM may move from one processor to another or even move its traffic from one switch to another in the network.

We recommend that you draw up a network topology diagram not just between processors, switches, the Oracle VM Manager and storage devices where appropriate but also between virtual networks and the physical LANs and VLANs they will connect to, including the bridges, bonds, teams, etc.

An example virtual network configuration is found in Appendix C – Virtual Network Configuration.

One important consideration for the network design with respect the management of Oracle VM Server Pools is that each Server Pool will require its own Virtual IP address (VIP) in addition to the IP addresses assigned to the individual Oracle VM servers themselves. This VIP will be used by the Oracle VM Manager and the Server Pools themselves to contact the controlling node (Server Pool Master) at any one time; this role is always taken up by one and only one Oracle VM server in every Server Pool even in the event of a node failure.

Stage 2: Creating your Oracle VM environment

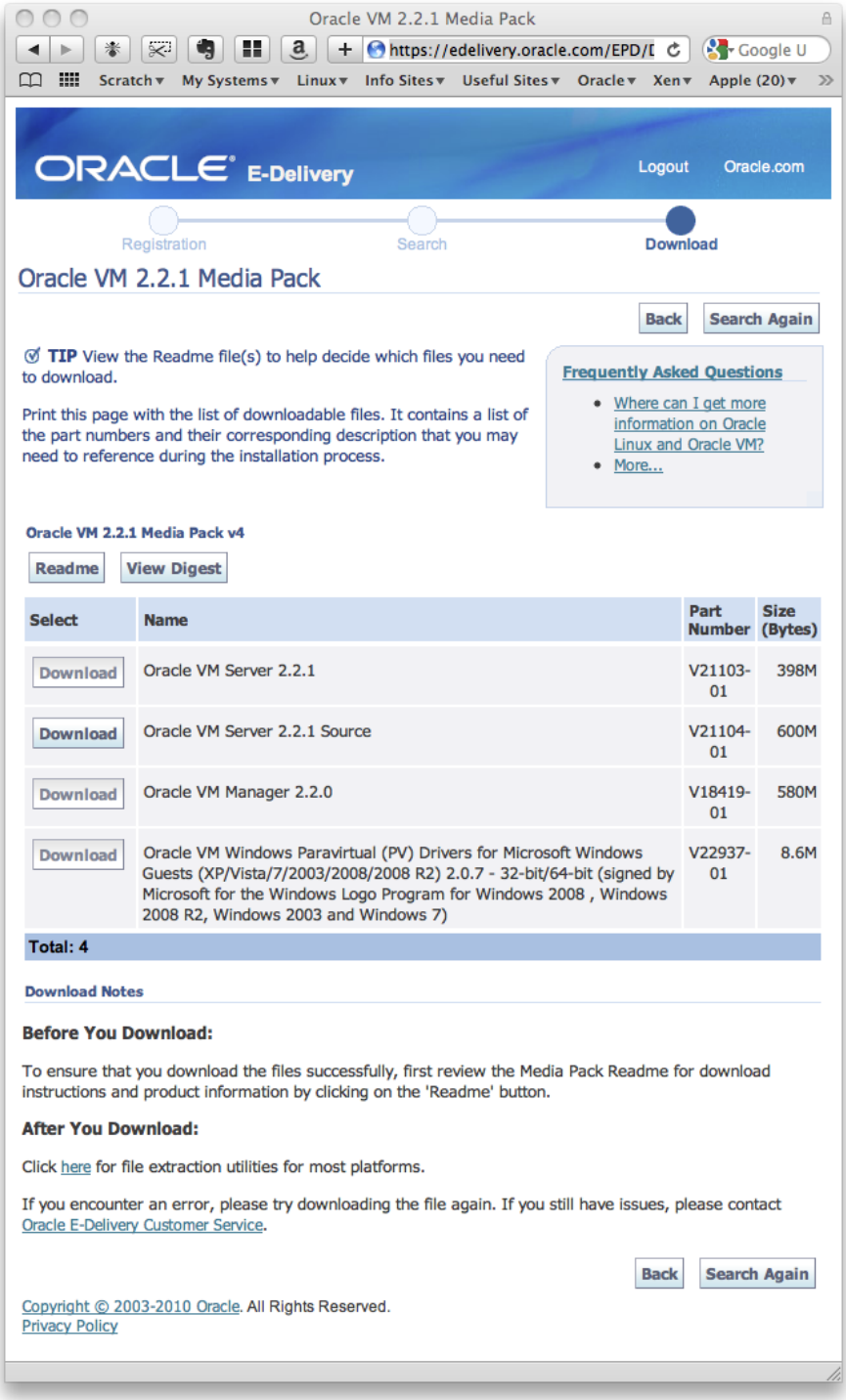
Once you have a clear design for your Oracle VM environment, creating that environment can begin.

Hardware Preparation

The first thing to do is install the hardware and plug in the storage and networking connections according to your design. Ensure that all the storage devices (LUNs, Zones, Arrays, etc.) and network devices (NICs, switches, routers, etc.) are correctly configured. If you have a test rig that can be plugged in and engaged for this purpose we advise using it – some customers have a basic Linux installation that can be simply and quickly installed on each host to run through some basic checks like storage probing, network teaming and VLAN connectivity.

Software Preparation

Download the latest version of the Oracle VM software from edelivery.oracle.com/linux. You will need to download at least the Oracle VM Server and Oracle VM Manager files (they are ZIPPED ISO files) from the list similar to the one shown in the following figure of the E-Delivery site for Oracle VM Software.



You only need to burn the Oracle VM Server ISO to disc as the Oracle VM Manager ISO can be simply mounted by the target Linux host when needed.

As we have said, some configuration of the Oracle VM server once it is installed is required in order to set it up according to your designs. This will result in the creation or modification of a set of specific configuration files on each unique Oracle VM server.

The simplest way to add Oracle VM servers to your Server Pools with the correct network and storage configurations is to take copies of each of the files you have changed and prepare a complete set of files to be applied to each Oracle VM server according to your design. In each set of files, ensure that the unique elements are set in preparation for each unique Oracle VM server. By preparing these sets of files you will also be able to recreate any one of the individual Oracle VM servers from scratch very quickly, meaning that you do not have to worry about backing up the Oracle VM server boot volumes.

Create an Oracle VM Server Pool

Our recommendation is to choose one host as the first one to be installed as an Oracle VM server and follow the steps below:

- Install Oracle VM [ensure you use the `linux mpath` install option if you are installing the software on a multipathed SAN boot disk; failure to do this will result in unexpected behavior and problems with failover in the event of a SAN issue].
- We highly recommend you update your Oracle VM server to the latest patch level using `up2date`. By default, new kernel updates/patches will not be downloaded – if you wish to ensure you have the latest kernels installed (both for Dom0 and Xen) use the `update -fu` command.
- Check that `/etc/hosts` file does not contain the hostname (or anything other than localhost) for address 127.0.0.1 (see below for how this should look): the hostname can sometimes appear in this line if the DNS settings were not set correctly during installation.

  /etc/hosts

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1        localhost.localdomain localhost
::1      localhost6.localdomain6 localhost6
```

- Edit the `/etc/hosts` file to ensure that it contains the hostnames and IP addresses for every other Oracle VM server you plan to put in the same Server Pool and any TCP storage hosts. Even if you are using DNS it is a good idea to do this so that in the event of a problem with the DNS server or the network to the DNS server the Oracle VM Server Pool will still function correctly.
- Create your custom network environment according to your design.
- If you are using TCP based networking either reboot your Oracle VM server or restart the network and xend services (it's a good idea to reboot to ensure that it comes up with the network the way you expect).
- If you are using block attached storage ensure the devices/LUNs and paths are all probed, configured and masked according to your storage design but do not mount any of the devices.
- Configure the NTP service to synchronise the time with your designated NTP server by editing the file `/etc/ntp.conf` and validating at least two entries (make sure you can ping them from the

Oracle VM server). You will need to turn on the NTP service as it is not running by default. Running ntpstat will show you the status of the running NTP service.

```
# service ntpd start
# chkconfig ntpd on
```

- You will need to format the file systems of all shared Storage Repositories to OCFS2 while they are un-mounted. Ensure you set the maximum number of nodes that will be accessing the shared file system correctly.

```
# mkfs.ocfs2 -Tdatafiles -N8 /dev/sdb
```

- Remove any locally created Storage Repositories:
  - Run: `/opt/ovs-agent-latest/utils/repos.py -l`
  - Make a note of the UUID of any existing repositories
  - Run: `/opt/ovs-agent-latest/utils/repos.py -d <UUID>`
- Add your planned shared Storage Repositories

```
# /opt/ovs-agent-latest/repos.py -n filer:/my/repository
```

If you have multipath devices make sure that devices are mounted via their `/dev/mapper/mpath`*n* devices rather than `dm-`*n* or raw paths and that your `/etc/mutltipath.conf` file is correct for your SAN devices to ensure correct operation in the event of path failure. You will also need to ensure the `multipathd` service is enabled and running. Please review section, Multipath Storage Tips, below for additional tips on configuring multipath devices as Storage Repositories.

If you have not already rebooted, reboot your Oracle VM server now.

- Install Oracle VM Manager onto your Oracle Linux target; if you are planning to use an existing or dedicated Oracle database rather than the supplied embedded one ensure you set this up now as part of the installation.
- Using Oracle VM Manager, create your first server pool containing the single Oracle VM server you have just installed and configured; it is always a good idea to test the connection using the facility provided before actually committing the create job. Obviously, as you only have one Oracle VM server node in your Server Pool you should tell it to be all three roles upon Server Pool creation.

If you do not allocate a VIP to the Server Pool, its server failover and HA functionality will not function correctly in the event of a failure in of the Sever Pool Master.

- Logging into your Oracle VM server's Dom0 you should find that all your shared Storage Repositories are now mounted as their appropriate file systems.

You now have your first Oracle VM Server Pool comprising a single Oracle VM server, which is a completely valid and operationally viable configuration. Assuming that your design calls for more than

Oracle VM server in a Server Pool (perhaps so that you can use Live Migration and/or HA functionality) you should now add the remaining Oracle VM servers to your Server Pools accordingly.

Add Oracle VM Servers to the Server Pool

When you install the remaining Oracle VM servers to be added to an existing Server Pool you will run through many of the same steps as above except adding the shared Storage Repositories, as this will be done automatically when you join them to your existing Server Pools using Oracle VM Manager. These steps are summarized here:

- Install Oracle VM server
- Update the installation to latest patch level
- Set your networking files (paying attention to `/etc/hosts`)
- Make the shared storage visible (probe and mask as required – ensuring you pay attention to the multipath configuration files and the note below) making sure you do not try to mount the devices or add them manually via the `/opt/ovs-agent-latest/utils/repos.py` command.

  Changes you make to partitions/devices on iSCSI or SAN LUNs may not be automatically seen by already running Oracle VM servers you plan to add to the Server Pool. To ensure that these already running Oracle VM servers see the correct paths, devices, partitions run `partprobe` on each Oracle VM server before you add it to the Server Pool.

- Reboot and ensure everything comes up
- Add the Oracle VM server as a new node in an existing Server Pool using Oracle VM Manager

Once you have installed all the Oracle VM servers and added them to your Server Pool according to your designs you will move onto the final stage: Testing.

Stage 3: Testing your Oracle VM Environment

The key stage before signing off your Oracle VM environment for production (even if it is just production in 'Test & Development') is to check that everything is working the way it should to identify potential problems that should be fixed before you sign it off.

Oracle has created a TechNote specifically addressing this topic: 1290587.1 - Performing Site Reviews and Cluster Troubleshooting with VMPInfo, which even provides a tool and instructions for automated health-checking of an Oracle VM Server Pool. In addition or as an alternative to that process we present here a simple checklist of elements of functionality to check the status of your Oracle VM Server Pool installation.

BASIC FUNTCIONALITY TESTS

TEST	ACTION	RESULT OF YOUR TEST
Create a HA Server Pool	Create a server pool with shared storage and enable it for HA, assigning a valid Virtual IP address for the server pool	

Check shared storage	Log into one of the servers in the pool as root. Perform the following:
	`#cd /OVS`
	`#touch foo`
	Log into each of the other servers in the pool and run the following:
	`#cd /OVS`
	`#ls -l`
	You should see the file foo listed along with the normal running_pool and other directories. If you don't see this file from one of the servers it means that it is not seeing the shared storage correctly and there is a problem with the cluster.
	Remove the file `/OVS/foo` once the tests are complete.	

Check server pool (re)starts	Log into each node as root and run the following command: `#init 0` Once they have all powered down, power them all up together. They should all come up ok, mounting their shared storage repositories and registering themselves with Oracle VM Manager (or OEM if you are using that instead of Oracle VM Manager). Check that one of the servers in the pool has	

	taken the role of Server Pool Master and that the server pool can be accessed via the Virtual IP address assigned to it. [`ping` or `ssh` should work]	
Create a Guest VM	Create a guest VM using Oracle VM Manager (or OEM); set this guest to be HA enabled and configure its native networking so that it has an active IP address on some known network. Once created, start it up and running if not left that way after its creation.	
Test Live-Migration	Open a network connection to the guest VM you created; this can be a continuous `ping` or a regular `ssh` session from some other system to this guest VM. Whilst this network connection is open/active initialize a live-migration of this guest VM to one of the other nodes in the server pool. The guest VM should be moved to the other node without any interruption to the active network session; i.e. the continuous ping or ssh session should still be running even after the guest VM has been migrated from the original Oracle VM server to the other within the server pool. If the live-migration fails, look in the log to see what the problem was. If the live-migration succeeds but the live network session breaks (i.e. the continuous ping or ssh session looses connection to the guest VM) then it means that the networking configuration on the two Oracle VM servers is different. Common issues are different VLAN configs, switch configuration or routing errors.	
Test HA failover	Switch off or disconnect the electrical power from the Oracle VM server on which the HA enabled guest VM is running. After three minutes, that same guest VM should automatically start up on one of the other Oracle VM servers in the HA enabled server pool. An even more complete test of this HA mechanism is to perform the same test whilst the Oracle VM Manager (or OEM) is either stopped, or disconnected from the network, as HA failover is independent of active	

	management involvement. If it takes a long time (more than 5 minutes) for the guest VM to restart on another Oracle VM server within the server pool then check the logs to see if any errors were reported. If the guest VM fails to start on one of the other Oracle VM Servers in the server pool check that both the server pool and the guest VM were started with the HA options enabled. Note: it is not enough to enable the HA option on a running guest VM - it needs to have been enabled when it booted to ensure correct operation.	
Test server pool master failover	Only one of the Oracle VM Servers in a server pool can be the acting server pool master and, by the same token, one of the Oracle VM Servers in a server pool must be a server pool master. This is assigned automatically by negotiation between the running nodes in a server pool. Test this is working correctly by shutting down the active server pool master Oracle VM Server; you can perform this shutdown gracefully (by issuing `init 0` or similar) if you wish. After three minutes, one of the other Oracle VM Servers in the server pool should become the server pool master. If you have assigned a virtual IP address to the server pool, the new server pool master will be accessible via this IP address. If you are using OEM10g ensure you have installed the latest patches to ensure this functionality operates correctly.	
Test any VLAN networks	Make sure that guest VMs configured to use VLANs can connect correctly when they are running on every Oracle VM Server in the server pool (live-migrate them from node to node to test this). Common problems for VLANs are errors in the network scripts for the VLANs on the Oracle VM Servers themselves, mistakes in the VLAN configuration within the network switches (VLANs on some switches will appear to work with some operating systems even when incorrectly configured but not with others).	

Once you have passed the tests and are happy that all the essential elements of your Oracle VM environment are operating within expected parameters we recommend you take an archive of the configuration files you created or modified for each Oracle VM server so that you can recreate them or add new Oracle VM servers as required.

Appendix A – Configuration Maxima

The configuration maxima for Oracle VM Server for x86 can be found at

- Oracle VM Server for x86 Technical specification: http://www.oracle.com/us/technologies/virtualization/oraclevm/024974.htm

Appendix B – Supported Guest Operating Systems

Oracle VM supports Linux, Oracle Solaris and Microsoft Windows operating systems running in its Oracle VM 2.2 virtual machines. Paravirtualised (PV) VMs are recommended for Oracle software running on Oracle VM for both performance and the ability to dynamically change the amount RAM and CPUs on the fly. Hardware Virtualised (HVM) VMs can run any unmodified OS and may have very high performance on the latest processors but usually do not recognize dynamic changes in RAM or CPU cores. For up to date guest OS support, please visit Oracle VM documentation: http://www.oracle.com/technetwork/documentation/vm-096300.html

Appendix C – Virtual Network Configuration

Here are example configuration files to create a custom network setup for an Oracle VM server. Specifically, these configuration files will set up the following networking environment on an Oracle VM 2.2 server with two NICs.

Physical Environment:

2x gBit network cards

- Oracle VM 2.2
- 1x Managed gBit Switch
- Both NICs are plugged into the same switch using tagged VLAN ports
- Gateway and DHCP server is 192.168.2.1

Networks Design:

- Two IEEE 802.1q VLANs are configured in the switch: VLAN50 and VLAN51
- No dedicated storage network
- No dedicated management network
- VMs can access shared network and both VLANs
- Oracle VM server is given an IP address of 192.168.2.101 which is used to access and control it via ssh and Oracle VM Manager.

Actions:

- Edit the existing `/etc/sysconfig/network-scripts/ifcfg-eth0` and `/etc/sysconfig/network-scripts/ifcfg-eth1` files to look like those below making sure to keep your existing HWADDR settings.
- Edit your `/etc/modprob.conf` file to add the following lines:

```
alias bond0 bonding
options bonding miimon=100 downdelay=200 updelay=200
```

- Create the remaining `/etc/sysconfig/network-scripts` files as required (examples are at the end of this appendix below).
- Take a copy of the existing `/etc/xen/scripts/network-bridges` file.
- Replace the existing `/etc/xen/scripts/network-bridges` file with the example below [this script is called from `/etc/xen/xend-config.sxp` to create the automatic bridges and could be replaced by commands to create the networks, VLANs and bridges rather than the network-scripts if you chose].
- Reboot the Oracle VM server

Results:

- There are three bridges made available for VMs to connect their vNICs to:
  - Xenbr0: part of the 192.168.1.0 network
  - Vlan50: a VLAN tagged as VLAN50
  - Vlan51: a VLAN tagged as VLAN51
- A VM connected to VLAN50 will only see other devices or VMs also connected to VLAN50. Similarly for devices or VMs connected to VLAN51. Devices or VMs not connected to either VLAN are not able to see those VLANs or any devices or VMs connected to them. [The commented sections in the bridges for those VLANs show settings that can be given to VMs on those VLANs to see this in operation.]
- Creating a new VM in Oracle VM Manager will select xenbr0 as the default bridge for each new vNIC connected to that VM but a drop-down selection can be made of all three bridges.
- The bandwidth into Dom0 and the VMs is aggregated up to a maximum 2Gbit while both NICs are connected and functioning.
- If one of the NICs is disconnected all the network connections remain intact (but bandwidth is reduced to 1Gb).

Configuration Files

/etc/network

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=OVM1
GATEWAY=192.168.2.1
```

/etc/modprobe.conf

```
alias eth0 e1000e
alias eth1 e1000e
alias scsi_hostadapter ahci
alias snd-card-0 snd-hda-intel
options snd-card-0 index=0
options snd-hda-intel index=0
remove snd-hda-intel { /usr/sbin/alsactl store 0 >/dev/null 2>&1 || : ; }; /sbin/modprobe -r
--ignore-remove snd-hda-intel
alias bond0 bonding
options bonding miimon=100 downdelay=200 updelay=200
```

/etc/xen/scripts/network-bridges

```
#!/bin/sh
true
```

/etc/sysconfig/network-scripts/ifcfg-eth0

```
# Intel Corporation 82567LM-3 Gigabit Network Connection
DEVICE=eth0
BOOTPROTO=none
HWADDR=00:21:86:F2:E0:AA
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
# enable JUMBO frames
MTU=9000
```

/etc/sysconfig/network-scripts/ifcfg-eth1

```
# Intel Corporation 82574L Gigabit Network Connection
DEVICE=eth1
BOOTPROTO=none
HWADDR=00:1B:21:4F:FE:48
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
# enable JUMBO frames
MTU=9000
```

/etc/sysconfig/network-scripts/ifcfg-bond0

```
BRIDGE=xenbr0
DEVICE=bond0
TYPE=Bonding
BOOTPROTO=none
ONBOOT=yes
USERCTL=no
IPADDR=0.0.0.0
NETMASK=0.0.0.0
MII_NOT_SUPPORTED=yes
BONDING_OPTS="mode=5"
# enable JUMBO frames
MTU=9000
```

/etc/sysconfig/network-scripts/ifcfg-bond0.50

```
#
# Global bonded VLAN entries
#
DEVICE=bond0.50
ONBOOT=yes
VLAN=yes
#
# basic VLAN bonded network device
#
#BOOTPROTO=static
```

```
#USERCTL=no
#IPADDR=192.168.1.101
#NETWORK=192.168.1.0
#GATEWAY=192.168.1.101
#NETMASK=255.255.255.0
#MII_NOT_SUPPORTED=yes
#
# Bonded VLAN bridge device
#
BOOTPROTO=none
TYPE=Ethernet
BRIDGE=vlan50
```

/etc/sysconfig/network-scripts/ifcfg-bond0.51

```
#
# Global bonded VLAN entries
#
DEVICE=bond0.51
ONBOOT=yes
VLAN=yes
#
# basic VLAN bonded network device
#
#BOOTPROTO=static
#USERCTL=no
#IPADDR=192.168.3.101
#NETWORK=192.168.3.0
#GATEWAY=192.168.3.101
#NETMASK=255.255.255.0
#MII_NOT_SUPPORTED=yes
#
# Bonded VLAN bridge device
#
BOOTPROTO=none
TYPE=Ethernet
BRIDGE=vlan51
```

/etc/sysconfig/network-scripts/ifcfg-vlan50

```
#
# VLAN file for Bonded VLAN Bridge
#
DEVICE=vlan50
BOOTPROTO=none
ONBOOT=yes
TYPE=Bridge
```

/etc/sysconfig/network-scripts/ifcfg-vlan51

```
#
# VLAN file for Bonded VLAN Bridge
```

```
#
DEVICE=vlan51
BOOTPROTO=none
ONBOOT=yes
TYPE=Bridge
```

/etc/sysconfig/network-scripts/ifcfg-xenbr0

```
DEVICE=xenbr0
ONBOOT=yes
Type=Bridge
DELAY=0
BOOTPROTO=static
IPADDR=192.168.2.101
NETMASK=255.255.255.0
NETWORK=192.168.2.0
GATEWAY=192.168.2.1
```

Appendix D – Additional Oracle VM Hints and Tips

Here we present a collection of little hints and tips for working with Oracle VM.

Multipath Storage Tips

Multipath storage via SAN or iSCSI is complex and prone to configuration errors. This document is not the place to address the whole subject of multipath configuration but here are several tips that may help you avoid common pit-falls.

Friendly Names for LUNs

By default, your multipath configuration will enable friendly names for each of the LUNs that can be addressed by multiple paths: this is what is responsible for creating the `/dev/mapper/mpathn` devices we recommend you use. You can change these friendly names to whatever you like using the alias option in the multipaths section of the multipath configuration file to set the name of the multipath device: as suggested, the default, hidden, alias for these paths is in the form `mpathn` in sequence of devices found during the initial probe. The alias for the multipath device is consistent across all the nodes in a cluster. For information on the multipaths section of the multipath configuration file, see Section 4.4, "Multipaths Device Configuration Attributes" of the Red Hat Enterprise Linux documentation.

The multipath bindings file (by default, `/var/lib/multipath/bindings`) must be available at boot time. If `/var` is a separate filesystem from `/`, then you must change the default location of the file. `/etc` is a good location for this file as it is always available at boot time. For example, create the directory `/etc/multipath`, relocate the `/var/lib/multipath/bindings` file to that directory and use the **`bindings_file`** directive in the defaults section of `/etc/multipath.conf` to make dm-multipath aware of the new location:

```
defaults {
user_friendly_names yes
        bindings_file /etc/multipath/bindings
}
```

To ensure the system-defined user-friendly names are consistent across all Oracle VM servers in the Server Pool, set up all of the multipath devices on Server Pool Master (i.e. the first Oracle VM Server you are configuring here) and copy the `/var/lib/multipath/bindings` file from that machine to all the other machines to be added to the Server Pool.

Scan for Changed Partitions Without Rebooting

When configuring the shared Storage Repositories on the first Oracle VM server in a new Server Pool (this first Oracle VM server will become the Server Pool Master initially) you may make changes to the partition tables of the shared storage devices – these will not be automatically reflected in the partition tables of any other running Oracle VM servers that are attached to the same shared storage, which will cause a failure to add them to the Server Pool. We recommend rebooting Oracle VM servers after

initial configuration of their multipath and network configuration before attempting to add them to a Server Pool, which will pick up any changes in the shared storage partitions but may take a long time if there is an extensive SAN fabric. To ensure these changes are reflected in any running Oracle VM servers after you have configured them without rebooting you can force the system to rescan the partition table by running the `partprobe` command on each of them:

```
# partprobe
```

Quick Patch Update

When you don't have time or resources to set up a local YUM repository but you have limited download bandwidth or access you can download the patches and updates to a single Oracle VM server that can be given direct access to the ULN and propagate them to the remaining servers manually.

On the Oracle VM server connected to ULN run `up2date` to register with ULN if you have not already done so but do not tell it to perform any updates. Once this is done, run `up2date` in a way to tell it only download the latest updates but not install them:

```
# up2date -df
```

The `-f` option will force `up2date` to download updated kernels, which are skipped by default. If you don't want to install new kernels omit the -f option.

This will check for all the updates and patches applicable to your Oracle VM server and download the relevant RPM files to `/var/spool/up2date`. Create a directory on some shared storage (can be one of the regular shared Storage Repositories but it is usually easier just to use an NFS share – Oracle VM installs the NFS automounter so you can simply look in `/net` to mount any available share) and copy all the downloaded RPM files to it.

Moving to an Oracle VM server at the same patch level as the source machine, you can mount that same shared directory and use YUM to perform a local install of those RPMs.

```
# yum localinstall --nogpgcheck -y <patch>.rpm
```

We use the `--nogpgcheck` if we have not imported the PGP key for ULN into the target Oracle VM server; we can trust the files as we used the real key to download the original files when using up2date on the source Oracle VM server.

When installing Oracle VM from the installation media it is the best time to download all the relevant patches and updates available for that release so that you can update all the local Oracle VM servers without worrying about which specific patches to install.

An example of this is shown in the following figure:

```
Login to OVM1 as root

[root@ovm1 ~]# rpm --import /usr/share/rhn/RPM-GPG-KEY
[root@ovm1 ~]# up2date
[root@ovm1 ~]# up2date -l

Fetching Obsoletes list for channel: ovm22_i386_latest...
########################################

Fetching rpm headers...
########################################

Name                                    Version        Rel
----------------------------------------------------------
crash                                   4.1.2          4.0.2.el5         i386
ovs-agent                               2.3            42                noarch
xen                                     3.4.0          0.1.21.el5        i386
xen-64                                  3.4.0          0.1.21.el5        noarch
xen-debugger                            3.4.0          0.1.21.el5        noarch
xen-devel                               3.4.0          0.1.21.el5        i386
xen-tools                               3.4.0          0.1.21.el5        i386

The following Packages were marked to be skipped by your configuration:

Name                                    Version        Rel  Reason
-------------------------------------------------------------------------------
kernel                                  2.6.18         128.2.1.4.28.el5Pkg name/pattern
kernel-ovs                              2.6.18         128.2.1.4.28.el5Pkg name/pattern

[root@ovm1 ~]# up2date -df crash ovs-agent xen xen-64 xen-debugger xen-devel xen-tools kernel kernel-ovs

Fetching Obsoletes list for channel: ovm22 i386 latest...

Fetching rpm headers...
########################################

Name                                    Version        Rel
----------------------------------------------------------
crash                                   4.1.2          4.0.2.el5         i386
ovs-agent                               2.3            42                noarch
xen                                     3.4.0          0.1.21.el5        i386
xen-64                                  3.4.0          0.1.21.el5        noarch
xen-debugger                            3.4.0          0.1.21.el5        noarch
xen-devel                               3.4.0          0.1.21.el5        i386
xen-tools                               3.4.0          0.1.21.el5        i386
kernel                                  2.6.18         128.2.1.4.28.el5  i686
kernel-ovs                              2.6.18         128.2.1.4.28.el5  i686


Testing package set / solving RPM inter-dependencies...
########################################
crash-4.1.2-4.0.2.el5.i386. ########################### Done.
ovs-agent-2.3-42.noarch.rpm ########################### Done.
xen-3.4.0-0.1.21.el5.i386.r ########################### Done.
xen-64-3.4.0-0.1.21.el5.noa ########################### Done.
xen-debugger-3.4.0-0.1.21.e ########################### Done.
xen-devel-3.4.0-0.1.21.el5. ########################### Done.
xen-tools-3.4.0-0.1.21.el5. ########################### Done.
kernel-2.6.18-128.2.1.4.28. ########################### Done.
kernel-ovs-2.6.18-128.2.1.4 ########################### Done.
[root@ovm1 ~]# cd /net/filer
[root@ovm1 filer]# cd Software/OVM2/OVM221Patches
[root@ovm1 OVM221Patches]# cp /var/spool/up2date/*.rpm .


Login to OVM2 as root

[root@ovm2 ~]# cd /net/filer
[root@ovm2 filer]# cd Software/OVM2/OVM221Patches
[root@ovm2 OVM221Patches]# yum localinstall --nopgpcheck -y *.rpm
Setting up Local Package Process
Examining crash-4.1.2-4.0.2.el5.i386.rpm: crash-4.1.2-4.0.2.el5.i386
Marking crash-4.1.2-4.0.2.el5.i386.rpm as an update to crash-4.0-7.2.3.0.1.i386
Examining kernel-2.6.18-128.2.1.4.28.el5.i686.rpm: kernel-2.6.18-128.2.1.4.28.el5.i686
Marking kernel-2.6.18-128.2.1.4.28.el5.i686.rpm as an update to kernel-2.6.18-128.2.1.4.25.el5.i686
Examining kernel-ovs-2.6.18-128.2.1.4.28.el5.i686.rpm: kernel-ovs-2.6.18-128.2.1.4.28.el5.i686
Marking kernel-ovs-2.6.18-128.2.1.4.28.el5.i686.rpm as an update to kernel-ovs-2.6.18-128.2.1.4.25.el5.i686
Examining ovs-agent-2.3-42.noarch.rpm: ovs-agent-2.3-42.noarch
Marking ovs-agent-2.3-42.noarch.rpm as an update to ovs-agent-2.3-38.noarch
Examining xen-3.4.0-0.1.21.el5.i386.rpm: xen-3.4.0-0.1.21.el5.i386
Marking xen-3.4.0-0.1.21.el5.i386.rpm as an update to xen-3.4.0-0.1.10.el5.i386
Examining xen-64-3.4.0-0.1.21.el5.noarch.rpm: xen-64-3.4.0-0.1.21.el5.noarch
Marking xen-64-3.4.0-0.1.21.el5.noarch.rpm as an update to xen-64-3.4.0-0.1.10.el5.noarch
Examining xen-debugger-3.4.0-0.1.21.el5.noarch.rpm: xen-debugger-3.4.0-0.1.21.el5.noarch
Marking xen-debugger-3.4.0-0.1.21.el5.noarch.rpm as an update to xen-debugger-3.4.0-0.1.10.el5.noarch
Examining xen-devel-3.4.0-0.1.21.el5.i386.rpm: xen-devel-3.4.0-0.1.21.el5.i386
Marking xen-devel-3.4.0-0.1.21.el5.i386.rpm as an update to xen-devel-3.4.0-0.1.10.el5.i386
Examining xen-tools-3.4.0-0.1.21.el5.i386.rpm: xen-tools-3.4.0-0.1.21.el5.i386
Marking xen-tools-3.4.0-0.1.21.el5.i386.rpm as an update to xen-tools-3.4.0-0.1.10.el5.i386
Resolving Dependencies
--> Running transaction check
---> Package kernel-ovs.i686 0:2.6.18-128.2.1.4.28.el5 set to be installed
---> Package kernel.i686 0:2.6.18-128.2.1.4.28.el5 set to be installed
---> Package xen.i386 0:3.4.0-0.1.21.el5 set to be updated
---> Package xen-tools.i386 0:3.4.0-0.1.21.el5 set to be updated
---> Package crash.i386 0:4.1.2-4.0.2.el5 set to be updated
---> Package xen-debugger.noarch 0:3.4.0-0.1.21.el5 set to be updated
---> Package ovs-agent.noarch 0:2.3-42 set to be updated
---> Package xen-devel.i386 0:3.4.0-0.1.21.el5 set to be updated
---> Package xen-64.noarch 0:3.4.0-0.1.21.el5 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

```
================================================================================================================================================
 Package                                 Arch                              Version                                        Repository
Size
================================================================================================================================================
Installing:
 kernel                                  i686                              2.6.18-128.2.1.4.28.el5                        kernel-2.6.18-
128.2.1.4.28.el5.i686.rpm                                 17 M
 kernel-ovs                             i686                           2.6.18-128.2.1.4.28.el5                        kernel-ovs-2.6.18-
128.2.1.4.28.el5.i686.rpm                              17 M
Updating:
 crash                                   i386                               4.1.2-4.0.2.el5                                crash-4.1.2-
4.0.2.el5.i386.rpm                                 1.6 M
 ovs-agent                              noarch                            2.3-42                                         ovs-agent-2.3-
42.noarch.rpm                                     625 k
 xen                                     i386                                3.4.0-0.1.21.el5                               xen-3.4.0-
0.1.21.el5.i386.rpm                                   4.1 M
 xen-64                                  noarch                            3.4.0-0.1.21.el5                               xen-64-3.4.0-
0.1.21.el5.noarch.rpm                                 9.5 M
 xen-debugger                           noarch                         3.4.0-0.1.21.el5                               xen-debugger-3.4.0-
0.1.21.el5.noarch.rpm                              64 k
 xen-devel                               i386                              3.4.0-0.1.21.el5                               xen-devel-3.4.0-
0.1.21.el5.i386.rpm                                 878 k
 xen-tools                               i386                              3.4.0-0.1.21.el5                               xen-tools-3.4.0-
0.1.21.el5.i386.rpm                                 3.8 M

Transaction Summary
================================================================================================================================================
Install      2 Package(s)
Update       7 Package(s)
Remove       0 Package(s)

Total download size: 55 M
Downloading Packages:
Running rpm check debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Updating       : crash                                                 [ 1/16]
  Updating       : ovs-agent                                             [ 2/16]
  Installing     : kernel                                                [ 3/16]
  Updating       : xen-tools                                             [ 4/16]
  Updating       : xen-devel                                             [ 5/16]
  Updating       : xen-debugger                                          [ 6/16]
  Updating       : xen                                                   [ 7/16]
  Installing     : kernel-ovs                                            [ 8/16]
  Updating       : xen-64                                                [ 9/16]
  Cleanup        : crash                                                 [10/16]
  Cleanup        : xen-tools                                             [11/16]
  Cleanup        : xen-64                                                [12/16]
  Cleanup        : ovs-agent                                             [13/16]
  Cleanup        : xen-devel                                             [14/16]
  Cleanup        : xen                                                   [15/16]
  Cleanup        : xen-debugger                                          [16/16]

Installed: kernel.i686 0:2.6.18-128.2.1.4.28.el5 kernel-ovs.i686 0:2.6.18-128.2.1.4.28.el5
Updated:  crash.i386  0:4.1.2-4.0.2.el5  ovs-agent.noarch  0:2.3-42  xen.i386  0:3.4.0-0.1.21.el5  xen-64.noarch  0:3.4.0-0.1.21.el5  xen-
debugger.noarch 0:3.4.0-0.1.21.el5 xen-devel.i386 0:3.4.0-0.1.21.el5 xen-tools.i386 0:3.4.0-0.1.21.el5
Complete!
[root@ovm2 OVM221Patches]# init 6
[root@ovm2 OVM221Patches]# Connection to ovm2 closed by remote host.
Connection to ovm2 closed.
```

Appendix E – Network Security Guidelines for Oracle VM 2.2 Deployments

The goal of this section is to provide general security guidelines for deploying virtualized solutions based on Oracle VM. The areas that will be covered are:

1. Oracle VM Security Overview
2. Network Security Tiers
3. Controlling Administrative Access
4. Activity Monitoring and Logging

Oracle VM Security Overview

For the balance of the network scenarios, here are some design considerations and default settings for Oracle VM:

1. The VM Manager XE database is restricted to localhost connections and is not remotely accessible via port 1521.
2. VM Server is a minimalist OS implementation derived from EL 5. By design, it has few moving parts and a minimum of network exposed services.
3. Oracle VM Server and the VM Manager template reside on an EL5 OS that includes an iptables firewall with an appropriate ruleset and default policies.
4. Default installations of Oracle VM Server or Manager do not provide physical security. They can be booted (using runlevel 1 or a rescue cd) and compromised by anyone with access to the physical console.

Network security tiers

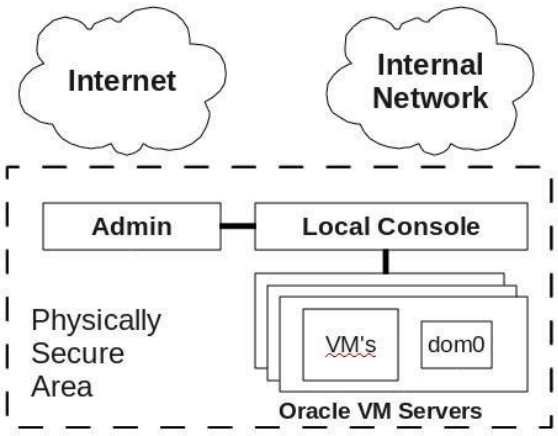
We will consider remote host connectivity and access control in the remainder of this section for the following Security Configuration Models:

1. No Network Connection
2. Isolated Local Network
3. Trusted Internal Network
4. Untrusted Internal Network
5. Internet Facing Service

The basic network topologies presented are examples. Network security architectures are application and threat dependent and must be developed and tested for specific environments.

No Network Connection

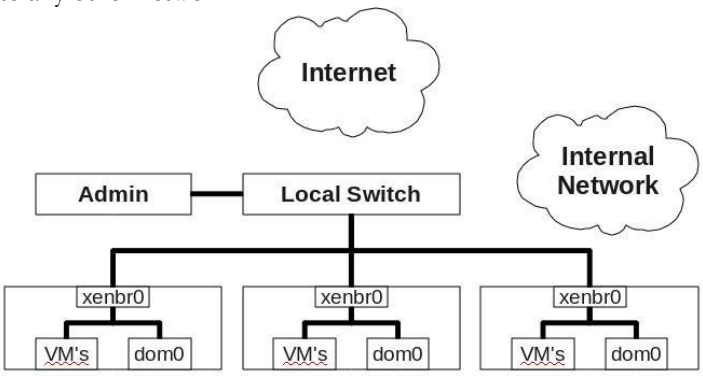
Some highly confidential applications cannot tolerate the possibility of any remote connection or exploit. Machines that require this level of security are usually kept in access-restricted rooms and often built without network cards. This type of configuration is a recommended best practice for credential originating systems such as a root certificate authority where the compromise of the root keys would put all the certificates and applications that were signed by that authority at risk.



Although some of Oracle VM's features such as High Availability and Master Failover aren't available without a network, peer servers and shared storage, the product can be configured for single node service where VMs can provide these secure applications using a local console to access them.

Isolated Local Network

An isolated local network consists of machines that are connected in a local network environment that has no connection to any other network.



In this model, there is no network connectivity to a larger internal network or the Internet. Since there is no potential for remote exploits from a large number of unknown sources, this environment can provide well defined physical, network and security characteristics.

By definition, access to this configuration is limited to personnel with access to the trusted admin hosts (including OEM Grid Control or VM Manager) on the closed, local network. Threats consist of an accidental "convenience" connection being made to other networks or a trusted admin installing an unsigned package or application (via a floppy, usb drive or CD) that may introduce a malware agent.

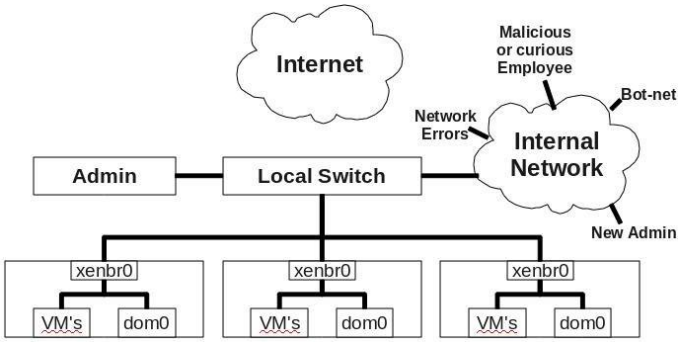
Guidelines for Isolated Local Networks

1. Set all default passwords for uniqueness and complexity. For using the High Availability/VIP features where the master role may be moved to any node, insure that the ovs-agent password on each one is complex, but identical across the pool.
2. Limit physical dom0 and VM Manager access to essential personnel

3. Avoid installing untrusted 3rd party software
4. If clustering with NFS shared storage, make sure the storage network is also restricted to the cluster subnet

Trusted Internal Network

A Trusted Internal Network is a well maintained and probably small internal network where network traffic is unrestricted to and from the cluster subnet.



While this configuration provides advantages of easy access to a company's infrastructure, services and storage, it also introduces a few threats. Normal safeguards, such as an Intrusion Detection Systems, anti-virus software and active network monitoring, can handle most of these. Human error via network and new admin mistakes are the most common causes of cluster service outages.

Guidelines for Trusted Internal Networks

1. Implement all the guidelines for Isolated Local Networks
2. Insure that the `iptables` service is active on each node and at least the default policy and ruleset is present in the file `/etc/sysconfig/iptables`:

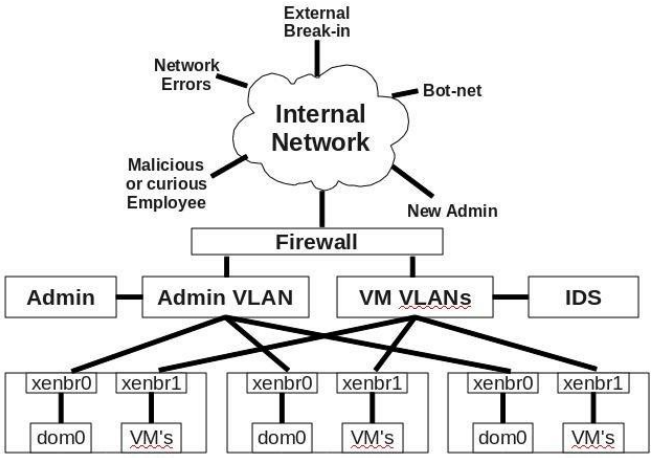
```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -p tcp -m state --state NEW -m tcp --dport 21 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j DROP
-A INPUT -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m udp -p udp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 2049 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 5900:5950 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 8002 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 8003 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 8899 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 7777 -j ACCEPT
```

```
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited COMMIT
```

3. If shared storage is mounted from outside the pool subnet, restrict the export to the dom0 and VM IP's... don't make nfs and san luns globally accessible

Untrusted Internal Network

An untrusted internal network has the characteristics of being loosely maintained and unmonitored, able to be compromised from the outside or from ongoing and uncorrected malware infested workstations on the inside.



In this model, the dom0 admin control and VM's have been partitioned into separate VLANs where traffic flow is controlled by a 3 zone router/firewall. The firewall policy is to allow outbound admin connections to anywhere, but blocks inbound connections from the untrusted internal network. Firewall policies to the VM network would depend on the application but should only expose service ports to the internal network that are needed. A signature driven Intrusion Detection System (IDS) on the VM network will monitor for traffic patterns that indicate an attack is underway and alert the administrators.

This model views any traffic from the internal network as potentially hostile. Additional hardening of this network can be done by implementing iptables based firewalls and policies on the admin and dom0 hosts that block inbound traffic. The dom0 firewall rules can also be enhanced to reject peer dom0 traffic on all ports except ocfs2 (7777) ovs-agent (8899) and the Xen administration ports (8002 and 8003).

Additional Host Security for Untrusted Internal Networks

1. Implement all the guidelines for Trusted Internal Networks
2. Disable `ssh` root logins. Admins should log in as themselves (using their global uid) with user privileges. Set up `sudo` to allow specific admin commands per user or root, if needed. See the next section "Controlling Administrative Access" for details.
3. Add failed connection logging to the existing `iptables` firewall just prior to the last `REJECT` line:

```
-A RH-Firewall-1-INPUT -m limit --limit 15/minute -j LOG --log-prefix "FW Drop:"
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
```

4. Set up a remote syslog server to track all user logins and firewall connection failures.
5. Disable the VNC connections on the dom0's:

```
#-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 5900:5950 -j ACCEPT
```

... and port forward VNC connections via `ssh` and the `vncviewer` rather then VM Manager:

```
ssh -L 5900:vmserverhost:5900 vmserverhost
```

6. Disable port 8888 and IPP ports on VM Manager:

```
#-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
#-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
#-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 8888 -j ACCEPT
```

This will leave secure ports 22, 443 and 4433 for admin command line access and VM manager connections.

7. Define a trusted admin network or host and limit VM Manager/VM Server `ssh` connections to that network or host. To implement this, comment out the existing `ssh` rule in the default `/etc/sysconfig/iptables` configuration file:

```
#-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
```

And replace it with the following for the 192.168.0.0/24 network:

```
¯A RH-Firewall-1-INPUT -p tcp -s 192.168.0.0/24 --dport 22 -j ACCEPT
```

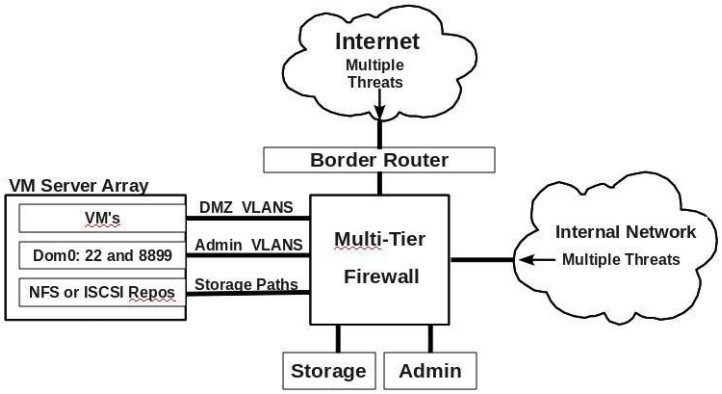
Or The following to narrow the connection to a single admin host at 192.168.0.67:

```
-A RH-Firewall-1-INPUT -p tcp -s 192.168.0.67 --dport 22 -j ACCEPT
```

Connections that fail will fall through and be logged by the rule provided in item 2. All the rules in `/etc/sysconfig/iptables` can also be changed to restrict access to selected networks or hosts.

Internet Facing Service

The most challenging model for securing a network connected host is placing VM based services directly on the internet. In this case, great care must be taken to insure that the environment won't be compromised through a wide variety of aggressive penetration techniques and exploits from malicious individuals, criminal gangs and hostile, well equipped governments.



There are a variety of topologies to provide zone isolation that can help to repel attacks and mitigate the effects of intrusion attempts. The following describes some basic approaches.

Internet Facing Service: General guidelines

1. Implement all the guidelines for Untrusted Internal Networks
2. All internet facing hosts and VM's reside on an isolated network stub called a DMZ. Connections into and out of the network are managed and monitored by a stateful firewall that enforces well defined rules and policies. If possible, an individual DMZ vlan per VM is optimal.
3. Never place VM Manager or any other mission critical administration tool on the internet. Use a vpn to access these hosts on secure internal admin networks.
4. Place the dom0's that are hosting internet exposed VM's in an administrative dmz. This zone cannot initiate connections to the internal network or internet, but is reachable from an admin vlan. The dom0 itself should not the reachable from the internet or the internal network.
5. Use selinux on internet exposed Linux VM guests whenever possible

Controlling Administrative Access

The Oracle VM Dom0 is a highly privileged environment. For maximum security, access to it must be strictly controlled and logged. The following provide methods for implementing these controls.

Limiting Physical Access

The Oracle VM Dom0 is a Linux system that can be booted in runlevel one or with a rescue CD that will allow the root password to be changed. Without physical access restrictions, the server configurations are at risk from these methods.

Disable Root Access

In any network environment where administrative responsibilities are shared, each administrator should log in with a uniquely assigned global ID (not as root) and use the sudo command to get root access or perform a specified, pre-defined set of privileged operations.

To disable root logins, find the default in `/etc/ssh/sshd_config`, uncomment the following line, change the yes to no and restart the **sshd** service:

Default: `#PermitRootLogin yes`

Change to: `PermitRootLogin no`

Use **sudo** to Provide root and Granular Administrator Privileges

For automated Oracle VM deployments (see Oracle KM Document 1288854.1), sudo can be used as an initial security mechanism that is easy to implement. Per the automated installation scripts, the sshd root login can be disabled before the machine's first boot. The initial administrator's public ssh keys can also be installed and the `/etc/sudoers` file configured to allow root privileges for that user.

To provide root access to a user with the global id "fred", add the user's id to `/etc/sudoers`:

```
## Allow root to run any commands anywhere
root ALL=(ALL) ALL
fred ALL=(ALL) ALL
```

To restrict the user to a specific command:

```
noob ALL=(ALL)NOPASSWD:/usr/sbin/xm
```

This will allow the "noob" user to start, stop and query the VM's on the Dom0, but won't allow access to the configuration, logs or other more privileged root commands.

While sudo is quite useful and provided with Oracle VM, many other options are available for user authentication. The basic security requirement is that each admin log in with a unique ID rather then as root. In the next section, we will use this to create a monitoring system that tracks these logins and provides an audit trail for all activities on the Oracle VM nodes.

Activity Monitoring and Logging

Tracking administrative logins and actions on the dom0 is an essential component of an overall security model. This is easily accomplished by setting up a restricted access, remote syslog server and configuring the dom0's to forward security oriented logs to it.

Setting up the Remote Syslog Server

The following assumes an OEL 5.5 remote server that is access restricted to security auditors. By default, the syslog configuration for a standard OEL 5.5 install won't accept log data from remote hosts. The first step is to change that default and restart the syslog service by editing `/etc/sysconfig/syslog` and adding a `-r` to the following line:

```
SYSLOGD_OPTIONS="-m 0 -r"
service syslog restart
```

Insure that the syslog daemon is listening on udp port 514 with the following command:

```
# netstat -uapnl | grep :514
udp    0      0 0.0.0.0:514     0.0.0.0:*       20213/syslogd
```

The iptables service should be active on a restricted server, so we'll need to create a rule that will allow our monitored Oracle VM server array to inbound connect to udp port 514. Edit the `/etc/sysconfig/iptables` file and add the following line prior to the last rule that shows `"REJECT --reject-with icmp-host-prohibited"`:

```
-A RH-Firewall-1-INPUT -p udp -s 192.168.0.0/24 --dport 514 -j ACCEPT
```

Then restart the iptables service:

```
service iptables restart
```

Configuring the Oracle VM Servers for Remote Logging

Each Oracle VM Server must be configured to enable remote security logging. This involves copying and modifying a single line in `/etc/syslog.conf`:

```
authpriv.* /var/log/secure
authpriv.* @10.0.0.59
```

The top line is the existing local log for authentication and sudo operations. The second also directs this information to our logging server that is set up at IP address 10.0.0.59. Restart the syslog service to begin the redirected logging:

```
service syslog restart
```

To test that remote logging is working, log into the Oracle VM server. Runing a log tail on `/var/log/secure` on the remote logging server will show:

```
# tail -f /var/log/secure
Apr 25 18:40:34 blade0 sshd[9647]: Accepted publickey for fred from
10.0.0.17 port 17152 ssh2
Apr 25 18:40:34 blade0 sshd[9647]: pam_unix(sshd:session): session
opened for user fred by (uid=0)
Apr 25 18:40:39 blade0 sudo: fred : TTY=pts/2 ; PWD=/home/tlisjac ;
USER=root ; COMMAND=/bin/su -
Apr 25 18:40:39 blade0 su: pam_unix(su-l:session): session opened for
user root by fred(uid=0)
```

Setting up all the VM Servers to do remote authentication logging will allow admin activities on the array to be monitored and tracked. In this case, sudo provides "fred" with root access, but the logs will also reflect the specific commands that are executed by "noob" where sudo only provided the xm command.

Restricting physical and remote access to the logging server is recommended and will prevent malicious or accidental modification of the security logs and insure that they accurately reflect all administrative activities on the server array.

Oracle VM: Designing, Creating and Testing an Oracle VM 2.2 Environment
2011
Author: Wayne Lewis
Contributing Authors: Tom Lisjac, Greg King

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com

Oracle is committed to developing practices and products that help protect the environment

Copyright © 2011, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

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

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. UNIX is a registered trademark licensed through X/Open Company, Ltd. 1010

Hardware and Software, Engineered to Work Together