



# Accelerate Oracle Database 10g Creation and Deployment Using VMware Infrastructure and EMC Celerra Writeable Checkpoints

*Applied Technology*

---

## **Abstract**

This white paper first reviews the business case for and the challenges associated with creating a large Oracle testing and development environment. The paper then describes the benefits and process of using VMware<sup>®</sup> Infrastructure with Celerra<sup>®</sup> Writeable Checkpoints to rapidly create and deploy Oracle Database 10g databases for application development, testing, and reporting. Detailed steps are also provided in this paper to illustrate the working procedure.

March 2008

---

---

Copyright © 2008 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on [EMC.com](http://EMC.com)

All other trademarks used herein are the property of their respective owners.

Part Number H4279

---

## Table of Contents

<b>Executive summary .....</b>	<b>4</b>
<b>Introduction .....</b>	<b>4</b>
Audience .....	5
Terminology .....	5
<b>Technology overview .....</b>	<b>5</b>
VMware Infrastructure 3.....	5
Celerra SnapSure .....	6
Celerra Writeable Checkpoints .....	6
<b>Accelerate Oracle 10g database creation and deployment on VMware Infrastructure 3.....</b>	<b>7</b>
Hardware and software description .....	7
Process overview .....	8
Detailed procedure.....	10
<b>Conclusion .....</b>	<b>16</b>
<b>References .....</b>	<b>16</b>

---

## Executive summary

One of the most challenging tasks for Oracle administrators is to create, deploy, and manage a large number of images of a production database for application development, QA testing, reporting, and data warehousing. Administrators often have to perform these activities frequently and quickly to meet their internal Service Level Agreements (SLAs). The traditional way of deploying databases to multiple environments presents several challenges including lengthy provisioning time, system downtime, and the need for additional server and storage resources.

VMware Infrastructure 3 virtualizes servers, storage, and networks to help transform IT infrastructure by:

- Increasing hardware utilization
- Reducing hardware requirements
- Streamlining the execution of IT management processes
- Improving application availability and business continuity

With VMware Infrastructure, users can encapsulate an Oracle database in a virtual machine on a VMware ESX Server host machine and then rapidly clone and deploy it massively without the need for dedicated servers to host these environments.

EMC<sup>®</sup> Celerra<sup>®</sup> is the ideal networked storage platform for VMware Infrastructure. EMC Celerra offers the most comprehensive suite of built-in features for today's VMware users who need advanced functionality to lower total costs. Among the features, Celerra Writeable Checkpoints can be used to instantaneously create point-in-time writeable checkpoints of a file system to assist in creating and deploying database clones at the storage layer.

This white paper provides an overview of the Celerra Writeable Checkpoints feature and how it can be used to accelerate creation and deployment of an Oracle Database 10g database. To use Celerra Writeable Checkpoints, the Oracle production database must be hosted in a Celerra file system that is exported using NFS, which is then presented to an ESX Server host as a NFS Datastore. The paper then uses an Oracle RAC 10g database as an example to illustrate how a single-instance testing/development (test/dev) database environment can be created on an ESX Server host. This paper also describes how to quickly clone the test/dev database and rapidly deploy a large number of development and test environments without the additional need for servers and storage.

## Introduction

This paper uses a hypothetical scenario that emulates real-world business needs as closely as possible. In the scenario, the production environment is an Oracle RAC 10g Database deployed on EMC Celerra and physical servers. A near-production copy of the database is created and converted into an Oracle Database 10g (that is, a single instance database) deployed on an ESX Server host with storage on Celerra in the form of NFS Datastore. The VM, hosting the guest OS, Oracle software, and the single instance database, is then cloned many times and made available to possibly hundreds of application developers and testers.

VMware Infrastructure provides virtual machine cloning capability that allows users to make multiple copies of a virtual machine from a single installation and configuration process. When the cloning operation is complete, the clone is a separate virtual machine.

The Celerra Writeable Checkpoints feature allows modification of the Celerra file system checkpoints. These writeable checkpoints can be exported and mounted on the client machines with read-write mode in a manner similar to regular file systems. With the inherent checkpoint technology, a writeable checkpoint is created instantly and is a point-in-time replica of the base file system while allowing changes to be made to the checkpoint without affecting the base file system content.

Together, VMware Infrastructure and EMC Celerra make it quick and easy to create and deploy Oracle database clones for other business uses. In order to utilize the Writeable Checkpoints feature, Celerra release 5.6 is required. The process included in this paper was validated against an Oracle RAC 10g

database on Linux; however this process is applicable to the Oracle RAC 11g database and other OS platforms.

## Audience

The intended audience is primarily administrators involved in configuring test/dev environments for Oracle 10g databases on VMware Infrastructure using the Celerra Writeable Checkpoints feature. The readers are expected to be familiar with EMC Celerra and have experiences administering Oracle database.

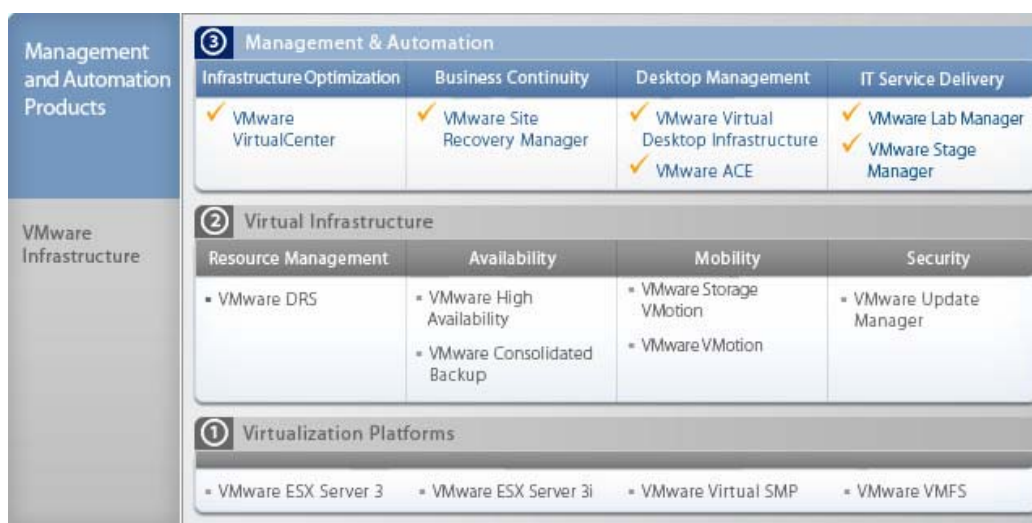
## Terminology

- **Baseline Checkpoint:** The read-only checkpoint from which a writeable checkpoint is created.
- **Celerra SnapSure®:** A Celerra software feature that creates instant checkpoints.
- **Checkpoint:** A read-only point-in-time logical copy of a Celerra file system.
- **Datastore:** A file system, either Virtual Machine File System (VMFS) or Network File System (NFS), that serves as a virtual representation of an underlying pool of physical storage resources. These physical storage resources can be comprised of SCSI disks from a local server, Fibre Channel SAN disk arrays, iSCSI SAN disk arrays, or NAS storage arrays.
- **Primary File System (PFS):** A Celerra file system from which checkpoints are created.
- **Virtual Machine:** A virtualized x86 PC on which a guest operating system and an associated application run. A virtual machine is also a set of discrete files that primarily include a .vmx configuration file and one or many .vmdk virtual disk files.
- **Writeable Checkpoint:** A read / writeable point-in-time logical copy of a Celerra file system.

## Technology overview

### VMware Infrastructure 3

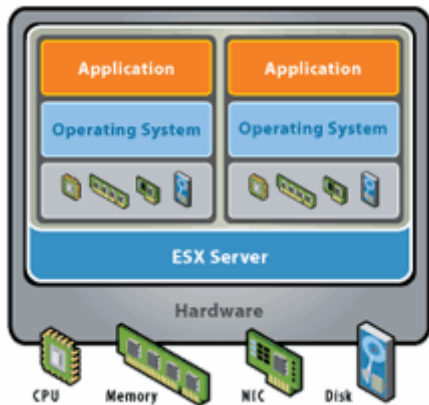
VMware Infrastructure 3 provides a suite of rich features to deliver efficiency, availability, and dynamic management capabilities for production use in a data center. [Figure 1](#) summarizes the VMware Infrastructure architecture and rich feature set as well as the management and automation products that work in conjunction with the underlying virtual infrastructure.



**Figure 1 VMware Infrastructure 3 features with VMware Management and Automation Products**

---

Within VMware Infrastructure, ESX Server is a market-leading hypervisor that abstracts server processor, memory, storage, and networking resources into multiple self-contained virtual machines. ESX Server employs a bare-metal architecture by inserting a robust virtualization layer directly on server hardware. ESX Server delivers enhanced virtual machine performance, advanced memory management, improved power-management, and four-way virtual Simple Management Protocol (SMP) to virtualize the most process intensive applications. Advanced management capability, high availability, and security are genuinely offered for virtual machines.



**Figure 2 VMware ESX Server**

An ESX Server host leverages high-performance, shared storage to centralize virtual machine file storage. A virtual machine is encapsulated in one or a set of virtual disk files. The virtual disk files can reside on a NAS storage array that presents storage to ESX Server in various forms including NFS Datastore. A virtual machine can be quickly cloned and customized before deployment in VMware Infrastructure.

## ***Celerra SnapSure***

The Celerra SnapSure feature creates a read-only, logical point-in-time image (checkpoint) of a production file system (PFS). SnapSure can maintain up to 96 PFS checkpoints while allowing PFS applications continued access to the real-time data. The principle of SnapSure is *copy old on modify*. When a block within the PFS is modified, a copy containing the block's original content is saved to a separate volume called the SavVol. Subsequent changes made to the same block in the PFS are not copied into the SavVol. The original blocks from the PFS (in the SavVol) and the unchanged PFS blocks (remaining in the PFS) are read by SnapSure according to a bitmap and blockmap data-tracking structure. These blocks combine to provide a complete point-in-time file system image called a checkpoint.

## ***Celerra Writeable Checkpoints***

Celerra Writeable Checkpoints is a feature that enables the Celerra administrator to make a copy of a file system checkpoint that is writeable. A file system can be restored from a writeable checkpoint, so this technology is suitable for use as mechanism for implementing and testing contained changes to production environment. A writeable checkpoint is always created from a baseline checkpoint. Each baseline checkpoint may only have one writeable checkpoint associated with it at a time. A writeable checkpoint's initial file system reflects the point-in-time view that is stored in its baseline checkpoint. Writeable checkpoint blocks (for example, writes to the writeable checkpoint) are stored in the SavVol that is associated with the file system. A writeable checkpoint does not count toward the maximum number of checkpoints for each PFS. However, a writeable checkpoint does count toward the maximum number of file systems for each data mover and for each Celerra cabinet. There may be no more than 16 writeable checkpoints created per PFS. A writeable checkpoint can be mounted on Celerra and then exported using NFS or shared using CIFS.

# Accelerate Oracle 10g database creation and deployment on VMware Infrastructure 3

In EMC lab validation, the following scenario was tested. The production environment is an Oracle RAC 10g database deployed on EMC Celerra and physical servers. A near-production copy of the database is created and converted into an Oracle Database 10g (that is, a single instance database) deployed on an ESX Server host with storage on Celerra in the form of NFS Datastore. The virtual machine, hosting the guest OS, Oracle software, and the single instance database, is then cloned many times and made available to a large number of application developers and testers.

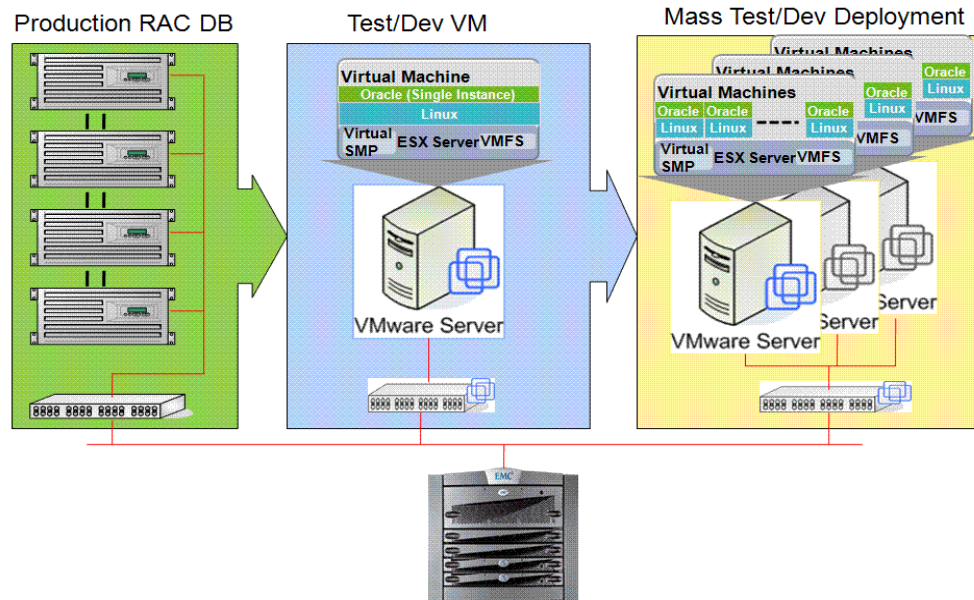


Figure 3 Test scenario and process

## Hardware and software description

Table 1 describes the hardware and software used in the validation scenario.

Table 1 Hardware and software

Component	Description	Usage
<b>Hardware</b>		
Celerra NS40G	Gateway system with 2 Data Movers	NAS storage
CLARiiON CX3-40	3 shelves of FC drives	Backend storage
Dell PowerEdge 1850	Two dual-core CPU 3.00 GHz, 12 GB memory	Oracle RAC 10g servers
Dell PowerEdge 1850	Two single-core CPU 3.00 GHz, 12 GB memory	ESX Server hosts
Dell PowerEdge 1850	Two single-core CPU 3.00 GHz, 4 GB memory	Virtual Center Management Server
<b>Software</b>		
Celerra DART	5.6.30.2	
RHEL 4	Update 4, 64-bit	Production database servers, and Virtual Machines for

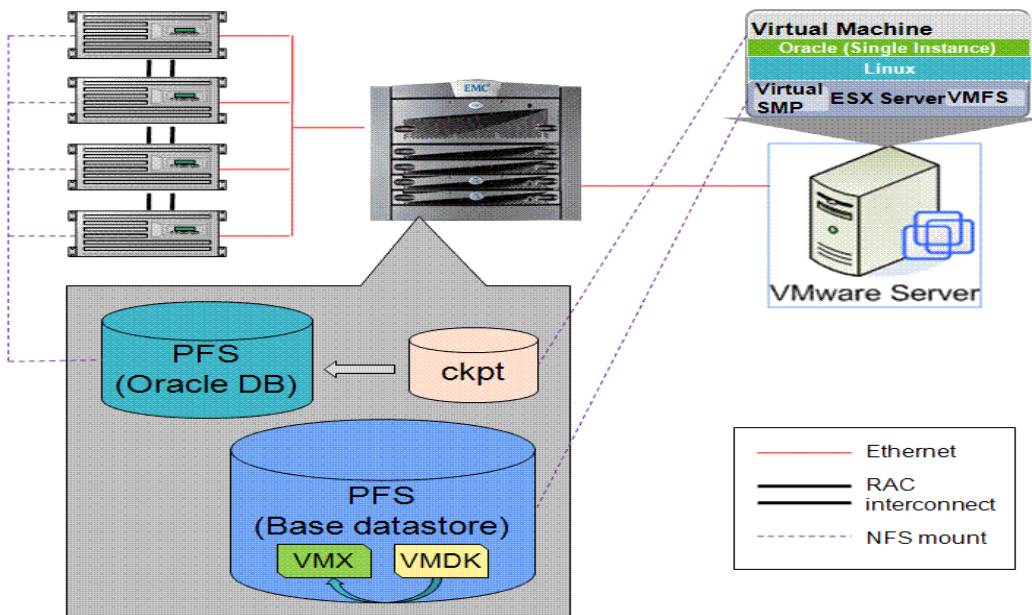
Component	Description	Usage
		Test/Dev
Oracle RAC 10gR2	10.2.0.2.0, 64-bit	Production database
Oracle Database 10gR2	10.2.0.2.0, 64-bit	Test/Dev database
ESX Server	3.5	
VMware VirtualCenter Server	2.5	Management server

## Process overview

The process is primarily comprised of the following stages.

- Stage 1 - build a self-contained virtual machine
- Stage 2 - create a base datastore and its writeable checkpoints

Stage 1 builds a virtual machine that encapsulates the Linux guest OS, Oracle Database 10g software, and a replica of the production database. Stage 2 makes clones of the virtual machine to establish a base datastore, and then creates writeable checkpoints of the base datastore for mass deployment of the virtual machine.

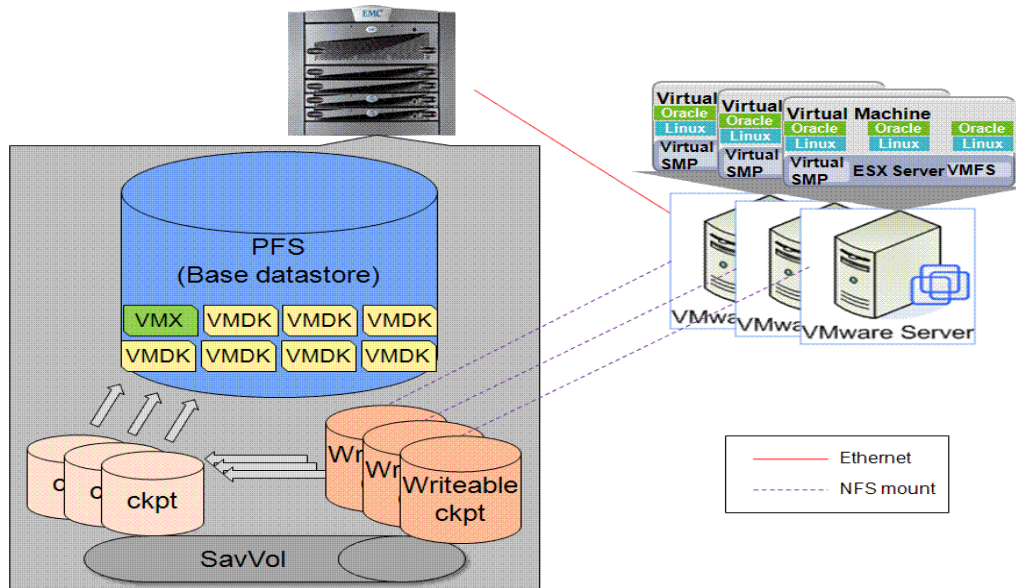


**Figure 4 Stage 1 - build a self-contained virtual machine**

Complete step 1 through 8 to complete stage 1 of this process.

1. Create a file system on Celerra and export it using NFS to hold the base datastore.
2. Install ESX Server and create a NFS Datastore using the file system just created.
3. Create a virtual machine in the Datastore previously created and install Red Hat Enterprise Linux and Oracle Database 10g on the virtual machine.
4. Create a regular checkpoint of the Celerra file system that holds the production RAC database and export it using NFS. Mount the checkpoint on the virtual machine.

5. Copy all the data files and archive log files of the production database from the checkpoint onto the virtual machine.
6. Configure Oracle Database 10g to use the files copied and bring up the single instance database.
7. Unmount the checkpoint, created in step four of this procedure, and remove it from Celerra.
8. Power down the virtual machine, and clone it to a virtual machine Template for future use (optional).



**Figure 5 Stage 2 - create a base datastore and its writeable checkpoints**

Complete step 9 through 13 to complete stage 2 of this process:

9. If more than 17 test/dev databases are required, clone the virtual machine as many times as needed and feasible in the base datastore. (optional)
10. Create a regular checkpoint of the base datastore, and then create a writeable checkpoint off of the regular checkpoint. Repeat this step to create up to 16 writeable checkpoints.
11. Export the writeable checkpoints on Celerra, and create NFS Datastores on the ESX Server hosts using the writeable checkpoints.
12. Register all copies of the virtual machine to add them into inventory.
13. Power on all copies of the virtual machine and customize them as needed. Each of them is a fully functional near-production Oracle test/dev environment.

**Important:** Depending on how many test/dev databases are needed, carefully plan to meet your business requirements. Some of the important factors to consider include:

- Size of the encapsulated virtual machine and size of the file system that holds the base datastore. The virtual machine size is largely determined by the size of the database, guest operating system, and Oracle software.
- Size of the SavVol of the file system. Any changes to a virtual machine will lead to SavVol consumption.

- 
- Growth rate of each of the test/dev databases should be considered. For virtual machines in the base datastore, any growth in size of a test/dev database will consume more disk space of the base datastore, and any database modification will increase SavVol space usage. For those in the writeable checkpoints, any such growth or modification will consume more disk space of the SavVol.
  - The maximum number of writeable checkpoints per PFS is 16.
  - Hardware configuration of the ESX Server host machines and resource requirements by each test/dev database. Some may demand more CPU cycles and memory than others. Based on these requirements, you need to estimate the optimal number of virtual machines for each ESX Server host.
- 

## Detailed procedure

With the process established of creating a test/dev database on VMware Infrastructure and then rapidly deploying it, this section includes a detailed procedure to demonstrate step by step how the process works.

---

**Note:** Variables are put in “<>”; in practice names or values of these variables should be replaced for the specific environment.

---

### Step 1: Create a file system for the base datastore

Use the following Celerra CLI commands to create a file system and make it available using NFS. For details, please refer to the relevant Celerra documentation. Alternatively, you can use Celerra Manager to create, mount, and export a file system easily.

```
# nas_fs -name <datastore_fs> -create size=<50G> pool=<clar_r5_performance>
# server_mount <server_2> <datastore_fs> </datastore_fs>
# server_export <server_2> -Protocol nfs </datastore_fs>
```

You should plan for storage layout and sizing carefully based on the requirements, especially when a large number of test/dev databases are requested and the databases will be actively augmented or updated.

---

**Note:** Using the process described in this paper, the test/dev databases will be created in a separate file system from the file systems that hold the production database. Therefore, test/dev database I/O and production database I/O will not share the same disk spindles. The production database remains online throughout this process and its performance will not be negatively impacted. In addition to separate disk layouts, you can assign ownership of the file system to another Data Mover available on your Celerra to take advantage of Celerra’s high performance architecture. Moreover, you can create this file system on another Celerra to completely separate the production environment and the test/dev environment. No additional technology or step is required following this procedure.

---

### Step 2: Install ESX Server and create NFS Datastore

1. Install ESX Server 3.5 on the physical server. Refer to the ESX Server installation guide for details about configuring the VMkernel.
2. Create a NFS Datastore using the <datastore\_fs> file system created previously. In order to access the NFS file system you need to create a VMkernel port with access to the NFS network. This can be done in the **Networking** window accessible from the ESX Server **Configuration** tab. To add a NFS Datastore, click the **Storage (SCSI, SAN and NFS)** link to bring up the **Add Storage** wizard. Use the following values when prompted by the wizard:

```
Storage Type: Network File System
Server (name of NFS Server): Use the IP address of one of the ports on the Data Mover
Folder (shared directory on NFS Server): <datastore_fs>
Mount NFS read only: Leave the checkbox cleared
Datastore name: <BaseStore>
```

---

### Step 3: Build the base virtual machine

1. Create a new virtual machine using VirtualCenter. In the **Inventory** panel expand the datacenter and right-click your ESX Server host. Select **New Virtual Machine** from the menu to bring up the wizard. Use the following values when prompted by the wizard:

**Virtual Machine Configuration:** Custom  
**Virtual Machine Name:** <BaseVM>  
**Datastore for the Virtual Machine:** <BaseStore>  
**Guest Operating System:** Linux  
**Version:** Red Hat Enterprise Linux 4 (64-bit)  
**Location:** Store with the virtual machine

For other parameters such as **Number of virtual processors** and **Memory**, please assign values based on your overall planning. After the virtual machine is created, install the guest OS (RHEL 4 Update 4 64-bit) into the virtual machine using the proper OS installation media.

2. Power on the virtual machine and install Oracle Database 10g Release 2 on the guest OS. Please follow the Oracle installation guide for procedure and configuration.

### Step 4: Create checkpoints of the production database and export them to the base virtual machine

To create a point-in-time copy of the production database and make it available to the virtual machine:

1. Place the production database in archive log mode with the following sequence of commands in case it is not currently in archive log mode (optional).

```
# srvctl stop database -d <primdb>  
# sqlplus / as sysdba  
SQL> startup mount;  
SQL> alter database archivelog;  
SQL> alter database open;  
SQL> archive log list;  
# srvctl start instance -d <primdb> -i <primdb2>
```

2. Place the production database in hot backup mode.

```
SQL> alter database begin backup;
```

3. Create a checkpoint of the database data file system.

```
# fs_ckpt <datafs> -Create -readonly y
```

4. Move the production database out of hot backup mode.

```
SQL> alter database end backup;
```

5. Archive the current redo log files.

```
SQL> alter system archive log current;
```

6. Create a checkpoint of the database archive log file system.

```
# fs_ckpt <archfs> -Create -readonly y
```

7. Create NFS exports for the data and archive file system checkpoints.

```
# server_export <server_2> -P nfs </datafs_ckpt1>  
# server_export <server_2> -P nfs </archfs_ckpt1>
```

- 
- Mount these checkpoints on the virtual machine.

```
# mount <DM IP>:/<datafs_ckpt1> </mnt/datafs>
# mount <DM IP>:/<archfs_ckpt1> </mnt/archfs>
```

### Step 5: Copy the data and archive log files from the checkpoints

On the virtual machine, create the same directory structure for the test/dev database as that for the production database:

- Create the directory structure for the database, assuming </u02> is for data files, </u03> is for redo log files, </u04> is the mirror of the redo log files, and </u05> is for archive log files.

```
# mkdir -p </u02>; chmod -R 775 </u02>; chown -R oracle:oinstall </u02>
# mkdir -p </u03>; chmod -R 775 </u03>; chown -R oracle:oinstall </u03>
# mkdir -p </u04>; chmod -R 775 </u04>; chown -R oracle:oinstall </u04>
# mkdir -p </u05>; chmod -R 775 </u05>; chown -R oracle:oinstall </u05>
```

- Copy the data files and archive log files from the checkpoints. Depending on the size of the production database, this may be a time-consuming step.

```
# cp -pR </mnt/datafs/*> </u02/>
# cp -pR </mnt/archfs/*> </u05/>
```

### Step 6: Re-create the database on the virtual machine

Using the copied data files and archive log files, re-create the near-production database and bring it up:

- On the virtual machine, set the environment parameters ORACLE\_SID, ORACLE\_BASE and ORACLE\_HOME to the same settings as on the production database servers.

```
# export ORACLE_BASE=</u01/app/oracle>
# export ORACLE_HOME=</u01/app/oracle/product/10.2.0/db_1>
# export ORACLE_SID=<primdb>
```

- Create a pfile from the spfile of the production database.

```
SQL> create pfile='</home/oracle/pfile.ora>' from spfile;
```

- Copy the pfile from to the virtual machine.

```
# scp <pfile.ora> oracle@<VM>:/u01/app/oracle/product/10.2.0/db_1/dbs/<initprimdb.ora>
```

- Create the required database dump directories on the virtual machine.

```
# mkdir $ORACLE_BASE/admin
# mkdir $ORACLE_BASE/admin/primdb
# mkdir $ORACLE_BASE/admin/primdb/adump
# mkdir $ORACLE_BASE/admin/primdb/bdump
# mkdir $ORACLE_BASE/admin/primdb/cdump
# mkdir $ORACLE_BASE/admin/primdb/dpdump
# mkdir $ORACLE_BASE/admin/primdb/hdump
# mkdir $ORACLE_BASE/admin/primdb/udump
# mkdir $ORACLE_BASE/admin/primdb/pfile
```

- Configure the listener and tnsnames.ora files on the virtual machine in a manner similar to those on the production database servers.

- Modify the pfile copied in step 3. Ensure that **cluster\_database** is set to **false**, and none of the **instance\_number** parameters exists..

- Copy the control file from </u02/primdb> to the log areas on the virtual machine.

```
# cp </u02/primdb/control01.ctl> </u03/primdb/control02.ctl>
```

---

```
# cp </u02/primdb/control01.ctl> </u04/primdb/control03.ctl>
```

8. List the available archive logs. This list might be necessary for recovering the test/dev database.

```
# ls -ltr </u05/primdb>
```

9. Connect to **sqlplus**.

```
# sqlplus sys as sysdba
```

10. Start up the test/dev database in mount mode.

```
SQL> startup mount;
```

11. Recover the test/dev database. Specify file names of the archive logs from the list displayed in step seven of this procedure. At the end, specify the current log file name from the same thread.

```
SQL> recover database until cancel;
```

12. Open the test/dev database with resetlogs option.

```
SQL> alter database open resetlogs;
```

### Step 7: Remove the checkpoints of the production database

The checkpoints of the data and archive file systems of the production database are no longer needed; they can be removed after being unmounted from the virtual machine.

```
# umount /mnt/datafs
# umount /mnt/archfs
# server_export <server_2> -u -p nfs </datafs_ckpt1>
# server_export <server_2> -u -p nfs </archfs_ckpt1>
# nas_fs -delete datafs_ckpt1
# nas_fs -delete archfs_ckpt1
```

### Step 8: Create a template for the virtual machine (optional)

Remove the hostname and IP address from the VM, then power off the VM. In the VirtualCenter server, right-click the VM and select **Clone to Template...** from the menu to create a template. This template can be used in the future for quickly deploying a single VM. This is an optional step.

### Step 9: Clone the virtual machine (optional)

Celerra supports a maximum of 16 writeable checkpoints for each PFS. Therefore, if no more than 17 test/dev databases need to be created, you can skip this step. Otherwise, you need to clone the base virtual machine a certain number of times in the base datastore before creating writeable checkpoints. For example, if you need to create 136 test/dev databases, you could clone the base virtual machine seven times and then create 16 writeable checkpoints of the base datastore:

$$(\text{Base virtual machine} + 7 \text{ clones}) \times (\text{Base datastore} + 16 \text{ writeable checkpoints}) = 136 \text{ virtual machines}$$

Alternatively, you could clone the base virtual machine 16 times and then create only 7 writeable checkpoints of the base datastore:

$$(\text{Base virtual machine} + 16 \text{ clones}) \times (\text{Base datastore} + 7 \text{ writeable checkpoints}) = 136 \text{ virtual machines}$$

You can have many different options. Please plan carefully based on the factors discussed previously in this paper.

---

To make copies of the virtual machine in the base datastore, in the VirtualCenter server, right-click the virtual machine and select **Clone...** to open the **Clone Virtual Machine Wizard**. Use the following values when prompted by the wizard:

**Name:** <CloneVM1>  
**Select the Host or Cluster:** select where you want the VM to run  
**Choose a Datastore for the Virtual Machine:** <BaseStore>  
**Select Guest Customization Option:** Do not customize

#### **Step 10: Create writeable checkpoints of the base datastore**

After the base datastore is established, if more test/dev databases are needed, create a writeable checkpoint of the base datastore. This writeable checkpoint becomes a logical copy of the base datastore and contains the same set of virtual machines. This step basically doubles the number of test/dev databases available for use. If you need more test/dev databases, repeat this step to create up to 16 writeable checkpoints.

```
# fs_ckpt <datastore_fs> -Create -readonly y  
# fs_ckpt <datastore_fs_ckpt1> -Create -readonly n
```

#### **Step 11: Export the writeable checkpoints to ESX Server host**

After the writeable checkpoints are created, they can be exported using NFS just like regular file systems. These writeable checkpoints can then be used to create NFS Datastores on ESX Server host.

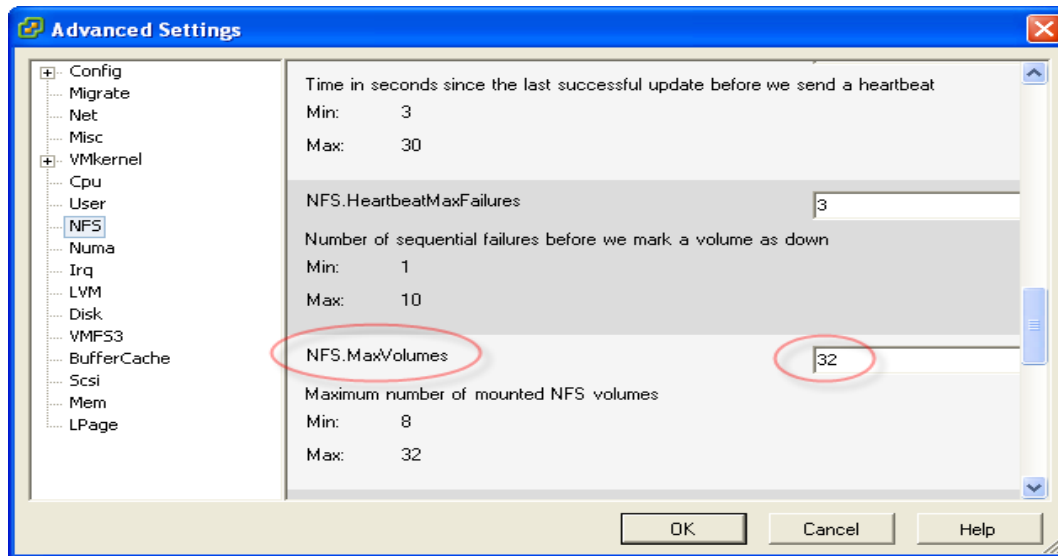
```
# server_export <server_2> -P nfs </datastore_fs_ckpt1_writeable1>
```

A VMkernel port with access to the NFS network has already been created in step 2; you can proceed to add a NFS Datastore. Click the **Storage (SCSI, SAN and NFS)** link to bring up the **Add Storage** wizard. Use the following values when prompted by the wizard:

**Storage Type:** Network File System  
**Server (name of NFS Server):** Use the IP address of one of the ports on the Data Mover  
**Folder (shared directory on NFS Server):** <datastore\_fs\_ckpt1\_writeable1>  
**Mount NFS read only:** Leave the checkbox cleared  
**Datastore name:** <CloneStore1>

Be aware that a single ESX Server host can only support a certain number of virtual machines running concurrently due to resource constraints. Additional ESX Servers hosts should be used so that the virtual machines can be distributed reasonably. For each additional ESX Server host to be able to access the NFS network, a VMkernel port needs to be created as described in Step 2: Install ESX Server and create NFS Datastore.

The maximum number of NFS mounts per ESX Server host is 32. Figure 6 shows how to increase the number from the default eight.



**Figure 6** Modification of maximum number of NFS mounts for each ESX Server host

### Step 12: Inventory the virtual machines

Up to this point, a large number of self-contained virtual machines have been created. Each virtual machine is a fully functional test/dev database server that encapsulates the guest OS, the Oracle software, and the database. All these virtual machines except for the base virtual machine must be registered with the ESX Server host to allow the VirtualCenter server to manage them.

Continuing the previous step, please plan for distribution of the virtual machines across different ESX Servers hosts. If needed, allocate those ESX Servers hosts to different clusters and even different datacenters to meet the business requirements. To register a virtual machine:

1. Select the ESX Server host and click the **Configuration** tab.
2. Click **Storage (SCSI, SAN, and NFS)** to display the available Datastores on the ESX Server host.
3. Right-click the Datastore that holds the virtual machine and choose **Browse Datastore**. The Datastore browser dialog box appears.
4. Either enter the search criteria on the **Search** tab or navigate the Datastore's hierarchy in the **Folders** tab to locate the virtual machine to be registered.
5. Right-click on the virtual machine (the .vmx file) and choose **Add to Inventory** to register it. Use the following values during the registration:

**What do you want to call this virtual machine:** <CloneVM1>

**Select a Resource Pool:** Select the appropriate ESX Server host or cluster

You can also register a virtual machine with the following CLI on the service console of the ESX Server host:

```
# vmware-cmd -s register <config_file_path>
```

### Step 13: Customize the virtual machines

The cloned virtual machines need to be customized prior to use. Each cloned virtual machine has the same network identity as the base virtual machine; the network configuration must be modified to avoid network conflicts. Furthermore, the change of network identity needs to be reflected in the Oracle database on the virtual machine.

After powering on a virtual machine, log in to it and modify the network configuration. Change the hostname of the virtual machine first. If DHCP is desired, use DHCP to obtain a new IP address. Otherwise, set a static IP address, the subnet mask, and gateway for the virtual machine.

---

Leave the Oracle database SID unchanged. Edit the listener.ora file to have the new hostname of the virtual machine updated. You should be able to bring up the database now.

## Conclusion

VMware Infrastructure provides a great framework for creating and deploying encapsulated virtual machines. The benefits include cost savings, improved manageability, added flexibility, and shortened service time, among many others. VMware Infrastructure has proven to be the ideal architecture for many business applications such as Oracle.

EMC Celerra Writeable Checkpoints help Oracle DBAs to speed up creation and deployment of Oracle test/dev environments for application development and testing. It eliminates unnecessary system down times while keeping the production environment safe from unexpected changes and interruptions. Together, VMware Infrastructure and Celerra Writeable Checkpoints offer an optimum solution to help administrators meet the frequent and changing requirements of creating, deploying, and managing a large number of clones of a production database for other business uses.

This paper described a process for the solution. A detailed procedure is provided to illustrate how the process works. Although the procedure is targeted to a certain software configuration, it can be easily tailored to fit other environments. To best utilize the process, careful planning beforehand is critical to the success of this solution.

## References

The following documents, located on Powerlink<sup>®</sup>, provide additional, relevant information. Access to these documents is based on your login credentials. If you do not have access to the content listed below, contact your EMC representative:

- *Using SnapSure on Celerra* technical module
- *Managing Celerra Volumes and File Systems Manually* technical module
- *Using EMC Celerra IP Storage with VMware Infrastructure 3 over iSCSI and NFS - Best Practices Planning* white paper
- *EMC Solutions for Oracle RAC 10g EMC Celerra NS Series NFS – Best Practices Planning* white paper

The following document on the Oracle MetaLink website (<https://metalink.oracle.com/>) provides more information about Oracle databases:

- *Oracle Database, Oracle Clusterware and Oracle Real Application Clusters Installation Guide 10g Release 2 for Linux*, Part No. 14203

The following documents, located on the VMware website (<http://www.vmware.com/>), provide more information about VMware Virtual Infrastructure 3:

- *ESX Server 3 Installation Guide – ESX Server 3.5 and VirtualCenter 2.5*
- *ESX Server 3 Configuration Guide – ESX Server 3.5 and VirtualCenter 2.5*