



# Installation Instructions for

# Oracle 9*i* Release 2 (9.2)

on

# SuSE Linux Enterprise Server 8

Powered by United Linux 1.0

and

**PolyServe Matrix Server 1.2**

**SuSE Inc. ©2003**
Contact: oracle@suse.com

# Table of Contents

# Introduction

This document describes the step-by-step installation process of Oracle 9*i* R2 (9.2) on a Linux system **Powered by United Linux 1.0** (SuSE Linux Enterprise Server 8) equipped with the High Availability product and cluster filesystem **PolyServe Matrix Server 1.2**.

For detailed instructions see the Oracle Installation Guide and the Oracle Administration Guide, both very good documents, that come with Oracle 9*i* R2. A must-read is the release notes accompanying all Oracle products! For a selection of more Oracle documentation see http://technet.oracle.com/.

The Oracle Installation Guide provides a much more comprehensive overview over **all** the possible installation options. This document on the other hand uses only **one typical installation method** as an example. If you want to know all the details about the Oracle installer and the Oracle tools like the Database Creation Assistant, see the Oracle documentation.

SuSE maintains a special section of Oracle/Linux support webpages at *http://www.suse.com/oracle/* -> *"Support Matrix"*.

### How we Tested

For writing this document SuSE and PolyServe joined forces at PolyServe's headquarters lab facility. PolyServe built a 12 node cluster of Intel-based servers and attached them to a switched Storage Area Network with several RAID arrays for storage.

SuSE personnel then proceeded to install Oracle 9*i* Real Application Clusters and document the steps in preparation for this document.

Taking advantage of the general-purpose nature of Matrix Server, the Oracle Home ( $ORACLE_HOME) was put on the cluster filesystem along with all Oracle related files! Oracle requires some directories to be per-node and not shared. For instance, the Oracle Cluster Manager directory (at least the configuration file and the log directory) and all other log directories. The PolyServe Clustered Filesystem supports node dependent files and directories via Context Dependent Symbolic Links (CDSL). CDSLs are symbolic links that resolve into a different file on each node. The advantage of keeping all files on the cluster filesystem, including the non-shared ones, is that they can be accessed and edited on any node in the cluster. For additional information of using PolyServe's CDSL technology for a shared Oracle Home please reference PolyServe Tech Tip TT0003.

Subsequent to the Oracle install, an OLTP database was built. All Oracle related files (e.g., control files, OCMS quorum disk, srvconfig file, datafiles, log files) were located in the PolyServe Matrix Server CFS. After the database was created, 12 instances were booted. In order to put a load on the system, a Pro*C application, with workload characteristics similar to TPC-C, was then executed on all 12 nodes. SuSE and PolyServe Engineers then conducted a series of forced fault-insertion tests. PolyServe Quality Assurance Engineering has conducted very intensive 14-point fault insertion testing on clusters running SuSE Linux and PolyServe Matrix Server on an on-going basis for quite some time, however, this was the first opportunity for SuSE and PolyServe engineers to do so jointly.

Once the workload achieved a state of high rate I/O, nodes where chosen at random and powered off. Analysis was conducted to ensure all the other remaining nodes were servicing OLTP transaction after the brief time required for Matrix Server and Oracle9i RAC to perform recovery. The test then proceeded to pick multiple nodes at random which were powered off nearly

simultaneously creating a cascading recovery event for both Matrix Server and Oracle9i RAC. In all cases, non-affected nodes continued to service the OLTP application. A truly high availability technology combination!

We tested and wrote this documentation at PolyServe HQ (Portland, OR). The ORACLE_HOME was put on the cluster filesystem in addition to the data files! Oracle requires some directories to be per-node and not shared. These are the Oracle Cluster Manager directory (at least the configuration file and the log directory) and all other log directories. The PolyServe cluster filesystem supports node dependent files and directories via Context Dependent Symbolic Links (CDSL) which are symbolic links that resolve into a different file on each node. The advantage of keeping all files on the cluster filesystem including the non-shared ones is that they can be accessed and edited on any node in the cluster.

## Why PolyServe Matrix Server?

Oracle Real Application Cluster runs on a regular SuSE Linux Enterprise Server 8 (United Linux 1.0), so why would you want to add a third product, PolyServe's Matrix Server 1.2? The overriding answer is that there is significant value in a complete, general-purpose cluster file system, like Matrix Server, for both Oracle-centric shops and enterprise customers that need a highly available, scalable cluster file system. For Oracle-focused customers, the larger a given Oracle 9*i* RAC cluster, the more value a commercial, enterprise-grade CFS, like Matrix Server, delivers to the data center.

PolyServe Matrix Server provides key functionality to ease the management of an Oracle 9*i* Real Application Clusters (RAC) installation. This functionality includes the following benefits that are not available when Oracle9i RAC is used with raw devices.

All Oracle 9*i* RAC file types are shared. Matrix Server enables multiple servers to concurrently access and share files on a SAN. It also guarantees data integrity by implementing a sophisticated distributed lock management system to arbitrate access to regular file data while offering Oracle9i completely unencumbered Direct I/O to database files. This shared data access provides significant manageability benefits to the administrator.

Matrix Server allows administrators to take full advantage of all Oracle 9*i* features. These features cannot be used without a CFS like Matrix Server.

**Shared Oracle Home.** All nodes in the cluster can use a single shared Oracle Home. The full Oracle9i Home directory has over 100,000 files. With Matrix Server, one copy of these files can be shared across all nodes in the cluster, enabling administrators to configure and run an Oracle9i RAC cluster from a single unified application image. Administrators are no longer burdened with the need to maintain a separate Oracle Home on each node in the cluster and to keep their files in sync.

**Oracle Disk Manager Support (ODM)**. ODM is an Application Programming Interface created by Oracle to replace the non-optimized I/O and file management API calls traditionally available on Unix/Linux systems. ODM offers asynchronous, direct I/O, specialized I/O priorities, fully optimized scatter/gather I/O, atomic file creation/deletion, and cluster-wide file keys that eliminate all risk of file mismanagement in complex clustered environments. Additionally, PolyServe's ODM (MxODM) offers a feature-rich cluster wide I/O monitoring package allowing monitoring at the cluster, database,instance or node level. MxODM monitoring delivers a clear single-image, real-time view of such desirable statistics as Log Writer and Database Writer I/O operations and

latency, parallel query activity, table scan I/Os, temp segment writes, read/write percentages and asynchronous I/O latency a feature not available without MxODM.

**OMF.** Matrix Server allows administrators to have full Oracle Managed Files (OMF) capability in an Oracle 9*i* RAC cluster. Oracle 9*i* provides a new and improved ability to monitor and manage data files that greatly reduces the complexity of creating, configuring, and managing the storage for an Oracle database. Administrators running Oracle 9*i* RAC with a shared raw device cannot use OMF functionality.

**ETL.** Matrix Server dramatically improves the speed of Extract, Transform, and Load (ETL) operations on Oracle 9*i*. The ETL capabilities provided with Oracle 9*i* allow a data warehouse to be loaded directly from external files (External Tables). In a cluster, this would be a bottleneck if only one server could read and process the External Table. However, with Matrix Server, the Oracle Parallel Query Option can parallel process the external file, resulting in a dramatic reduction in the time it takes to load a data warehouse.

## Partners - SuSE and PolyServe

PolyServe and SuSE have a strategic alliance and bilateral support agreement, and it is this relationship that provides enterprise customers with the comfort and confidence required to deploy a SuSE/PolyServe-based mission-critical environment.

PolyServe Matrix Server runs out-of-the-box on standard SuSE kernels. PolyServe and SuSE work together to run rigorous performance and quality assurance tests to ensure stability of the kernel and the operating environment.

# What is “United Linux” (UL)

For a detailed description see *http://www.unitedlinux.com/*, this explanation here concentrates on technical and Oracle certification aspects of UL. We used the *SuSE Linux Enterprise Server 8* “Powered by UnitedLinux-1.0” as an example.

### What is UL?

UL is a consortium of currently four companies to jointly develop a core Linux distribution used by all UL members. The focus is on “development” only, how the member companies market their products and what they do with the UL CDs they receive as a result of the development process is entirely up to the respective members (except that they are not allowed to make **any** changes to those UnitedLinux CDs)!

When you open your SLES-8 box you will find 4 CDs: One for “SuSE Linux Enterprise Server 8” and 3 labeled “United Linux 1.0, CD #1...#3”.

The UL CDs are **self-contained**, which means they are installable and contain a complete Linux system! This is what was used for the Oracle certification. The fourth CD contains add-on (“value add”) packages, the contents of this CD is different for each UL member company. The UL CDs on the other hand are **exactly** the same for each UL member.

The vendor “value add” CD **does not replace** any packages on the UL CDs! For example, there are no vendor specific kernels packages, there is only **one** kernel – the UnitedLinux kernel – for all UL based products.

### What is on the vendor specific CD?

That depends on the vendor. On the SuSE Linux Enterprise Server (SLES) 8 CD, for example, you will find packages that give you additional YaST2 modules and other add-ons. None of them are essential for running Oracle though!

### What if I install from the vendor CD?

You can start the installation with either UL CD #1 or with the vendor add-on CD. The difference is that if you start from the SuSE vendor CD your screen will read “SuSE Linux Enterprise Server 8” instead of “United Linux 1.0”, and you see the vendor add-on packages in the software selection instead of only the UL packages. Otherwise there is no difference! The installer will ask you to insert UL CD #1 immediately after start.

To install a minimum and Oracle certified system you need UL CDs #1 and #2 (#2 has the *gcc_old* package with gcc 2.95.3), and the “UL Service Pack #1” CD (or at least the latest kernel update).

### What is a “Service Pack (SP)”?

Several times per year there will be a *Service Pack* released for United Linux. This SP is going to contain all patches and updates that have accumulated thus far.

If you are a SuSE customer: If you use YaST2 Online Update (YOU) to access the SuSE Maintenance Web you do not need the SP CDs since your system already has all the updates the SP contains. The SP is just a collection of patches that are available via *YOU*. It too is bootable!

# Requirements

### *Hardware*

For a detailed description of hardware- and disk space requirements see the Oracle Installation Guide that comes with Oracle 9*i*R2 for Linux. An Oracle cluster requires *shared storage*, i.e. disks that are equally accessible by all nodes of the cluster. The most common technology used is a fiber channel attached storage array.

### *Software*

- United Linux 1.0 based Linux system
- PolyServe Matrix Server 1.2
- Oracle 9*i* R2
- Patches/Service Packs for UnitedLinux 1.0

  Install the UnitedLinux patches after the operating system installation, and before the Oracle installation. The source of UL patches is the vendor you purchased your “Powered by UnitedLinux” system from!
- Patches/Patchsets for Matrix Server 1.2
- Patches/Patchsets for Oracle 9*i* R2

  See Oracle Metalink. You must be an Oracle support customer to have access. The Oracle patches must be installed after the Oracle installation, obviously. You install patchsets using the Oracle Universal Installer. Each patch and patchset comes with detailed installation instructions.
- SuSE Support Website for Oracle: **http://www.suse.com/oracle/**

  SuSE maintains a special section of support webpages for SuSE/UnitedLinux customers running Oracle.

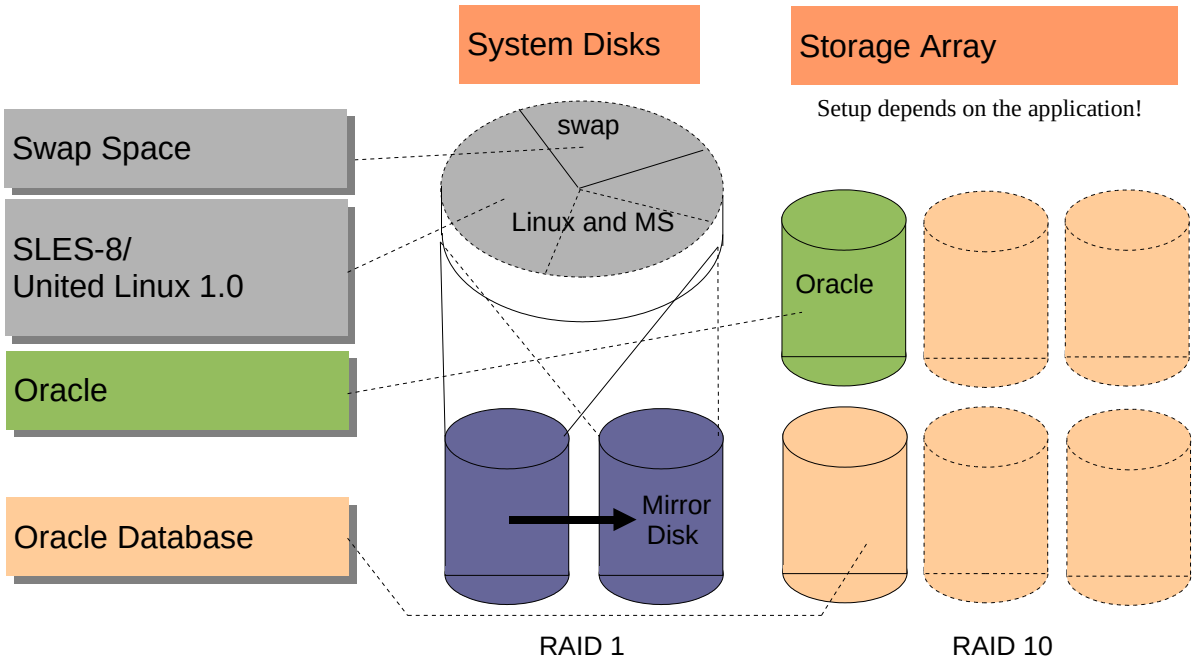
  Please check the page for *Oracle 9i R2 on SuSE Linux Enterprise Server 8* for any news and additional information, reachable by clicking on the appropriate icon in the **Support Matrix** (direct URL: http://www.suse.com/oracle/db/9iR2_sles8.html).

# Installation: SuSE Linux Enterprise Server

For complete instruction please have a look at the documentation that comes with SuSE Linux Enterprise Server. Here we only describe special aspects of an Oracle RAC cluster installation.

**Partitioning:** Ideally you have partitioned your system in a way that allows you to install Oracle and the database files on different partitions than the OS. The advantage is that when you have to update the operating system at some point you can do a complete new, fresh installation and reformat the OS partition without loosing your Oracle installation and/or data! We recommend to place the Oracle installation in */opt/oracle*, for the only reason that this is what our add-on package *orarun.rpm*, described below, *sets* as default and because this is a Linux standard location (FHS – Filesystem Hierarchy Standard, part of LSB – Linux Standard Base) for such software. Feel free to choose any other location, though!

Here is the setup we used:

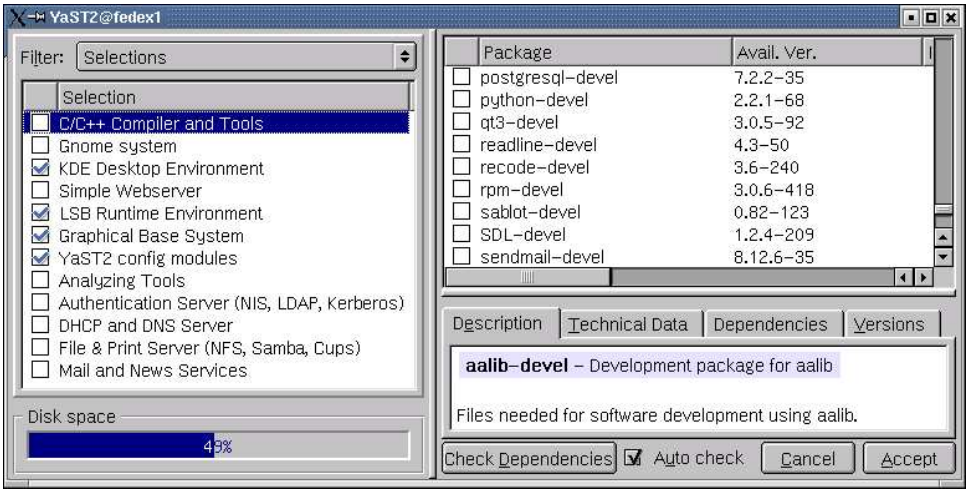



We suggest that you have at least two internal drives in the server unit, configured in a RAID-1 (mirroring) configuration in the hardware RAID controller of the server to protect against the loss of any one of the drives. There should be at least three main areas (partitions) on the disk (the virtual disk provided by the hardware RAID controller). One is the swap space, the other one is for the operating system - possibly divided further e.g. to put logfiles and the temporary directory in a different partition - and the third one is for the Oracle software.

You should give ca. 2-4 GB to Linux, and at least 2 GB for swap. Note that on 32bit Intel the swap partition cannot be larger than 2 GB, so if you need more swap space you need more than one swap partition.

**Software package selection:** A good base to start is if you make these selections shown in the screenshot below in the software installation dialog during the Linux installation (or later). By default many more items will be installed, but this is really all you need.

The items in the left column are *Selections*, i.e. they are not individual packages, they represent a collection of packages. Each of these *Selections* is self-contained, so if you select just (any) one you will get a fully functional system because each one contains the same core set of packages! By de-selecting a *Selection* you de-select all packages contained in this particular *Selection* but those packages also needed by another selection which is still active are not de-selected.




Note: This window shows only United Linux selections. If you install SLES-8 you get one more item “SLES Administration Tools” at the bottom with additional YaST2 modules.

You do not need to select “C/C++ Compiler and Tools”!

**Tip: Security** related packages, some installed by default (only a sample):

*pam* (pluggable authentication modules, always installed)
*openssh* (Secure Shell – powerful encrypted default remote login service)
*cipe*, *vtun*, *freeswan* (IP tunneling to build VPNs)
*sslwrap*, *stunnel* (turn any inetd service into an encrypted service)
*cracklib* (password check)
*bastille* (security hardening program)
*aide* (intrusion detection system)
*seccheck* (security reports via via *cron*)
*nessus* (security scanner)
*saint* (security scanner)

After selecting the *Selections* above switch to *Search* in the top left of the installer window, marked “Filter:”. Search for - and make sure that they are selected for installation – the following packages: ***gcc_old***,***glibc-devel***, ***libaio-devel***. All other packages you may need have already been selected since they are part of the *Selections* above!

Do **not** install/de-select any Java packages (since Oracle includes them already): Search for “java” and de-select all packages displayed, which is 4 IBM-Java packages, plus the two packages *java2-jre* and *postgresql-jdbc*.

**For your information, if you do not follow the above procedure:**

**Must Install:**

- ✔ (Included in the above selection) Make sure that **X-Window** is installed and running, or that at least the X-Window libraries are present for a remote installation.
- ✔ You will also need to have the basic developments tools installed, like **make**, **gcc_old (2.95.3)**, and the **binutils** package.
- ✔ **Which compiler:** UnitedLinux-1.0 has been certified using the ***gcc_old*** (gcc 2.95.3) package rather than the *gcc* package (gcc 3.2). The *gcc_old* package installs gcc 2.95.3 in */opt/gcc295/*, so you can have both gcc versions installed at the same time – which one you use depends on your path (which "gcc" executable is found first)!
- ✔ The *orarun* package.
- ✔ To view the Oracle documentation you need a **web browser** (e.g Mozilla or KDEs Konqueror) and a **PDF reader** (e.g. Adobe Acrobat Reader™, gv/ghostscript or xpdf).
- ✔ You may need to install package **pdksh**, which provides a Korn shell. Some Oracle scripts require this shell.
- ✔ **Java:** The Oracle Java GUI tools like *dbca* use their own JRE (Java Runtime Environment) bundled with Oracle. Oracle includes both version 1.1.8 and 1.3.1 of the JRE and also the JDK, so there is no need to install any additional Java package bundled with UnitedLinux 1.0.
- ✔ You should install the *orarun* package (see appendix and text below). The instructions below **require** this package.
- ✔ You **must** install the latest UnitedLinux kernel update! Oracle was certified against an update kernel, the original UL-1.0 kernel is NOT certified!

**Should NOT Install:**

- ✗ Do not install OpenLDAP, an LDAP directory server, if you do not plan to use it or if you plan to use Oracle Internet Directory (OID), which is Oracle's LDAP service.
- ✗ We recommend you do not install any other server software. Production Oracle servers should run only Oracle and nothing else. One exception could be a Mail Transfer Agent (MTA) like sendmail or postfix.
- ✗ Do not replace any core components of the Linux system with packages from outside sources.

### After the OS Installation

**Required patches for the operating system:** In order to get a certified Oracle system you have to at least update the kernel, but we highly recommend to install all applicable patches because they fix everything from stability to security problems and also provide support for new hardware.

To get an Oracle certified system you need at least (i.e. by now more current updates are available and should be used) the following kernel version:

k_smp-2.4.19-196.i586.rpm - SMP kernel
k_athlon-2.4.19-200.i586.rpm - Optimized for AMD Athlon
k_deflt-2.4.19-207.i586.rpm - Single CPU
k_debug-2.4.19-164.i586.rpm - Debug kernel
k_psmp-2.4.19-201.i586.rpm - Support for *very* old Pentium CPUs
kernel-source-2.4.19.SuSE-133.i586.rpm - for a self-compiled kernel (discouraged)

SuSE and Oracle tested and expect you to use the SMP kernel – a single CPU works but is not recommended for a production Oracle server.

**Service Packs** vs. **SuSE Maintenance Web:** There are two ways to install the required patches. You can use a United Linux Service Pack or you can use the SuSE Maintenance Web. The Maintenance Web contains the same patches as the Service Pack (naturally), and usually more – since the Maintenance Web is constantly updated with the latest patches while the Service Packs are released only a few times (2-4) per year. Since Service Pack 2 the Maintenance Web also has an entry for the Service Pack, selecting it installs all packages of the Service Pack.

**Remote access:** After the OS installation the only way to access the system remotely is via Secure Shell (*ssh*). Anything else, e.g. *telnet*, *ftp*, or *rsh*, will not work! For security reasons UnitedLinux by default does not activate the *inetd* daemon providing those services, they have to be enabled explicitly e.g. by editing */etc/inetd.conf* to enable services and then by calling (as *root*) *"chkconfig inetd on; rcinetd start"*.

**Special Partitioning:** Set up the space for Oracle if you didn't do so already. The database files should be on different disks than the operating system, but the Oracle software itself should at least go into a different partition than the operating system (see the discussion above). We recommend a clean separation! Read the *Oracle 9i Administrator's Reference* (distributed with 9*i*R2) about the Oracle Flexible Architecture (OFA), the disk- and mount point setup Oracle recommends.

Since there are many different methods and all have their use and depend a lot on how the database is going to be used, we will not make any suggestions. A small demo/test system can very well have everything, OS, Oracle and data files in one partition, but a big production database on a server with many disks may have a very different setup.

**Kernel parameters** are something you do not need to worry about if you use our *orarun* package, we will set them dynamically during runtime! Refer to either the *Oracle Installation Guide for Linux* for which variables to set to what values, or simply read on, install the *orarun* package (see Appendix), and get that information from the file `/etc/sysconfig/oracle`. For the installation you do not need to do this in any case, only for actually running Oracle.

**Users and groups** for Oracle (user *oracle*, groups *oinstall*, *dba*) are setup automatically when you install package *orarun* as described below.

**Environment variables** are another subject we will take care of during the installation, by installing a package *orarun* SuSE created to make these things easier.

# Step by Step

**Starting point:** A plain SuSE Linux Enterprise Server 8 “Powered by UL” has been installed on all nodes, including all currently available patches and also the ***orarun*** package. All partitioning has been done. **Now select one machine to be the installation system.**

### Option 1: Local installation

1. Depending on if you have a GUI login window or are on a text console:
   Start X-Window and login (as yourself, the user you setup during the system installation). Or the other way around, depending on if you booted into the *xdm* runlevel 5 (X-Window is always on) or into the console runlevel 3.

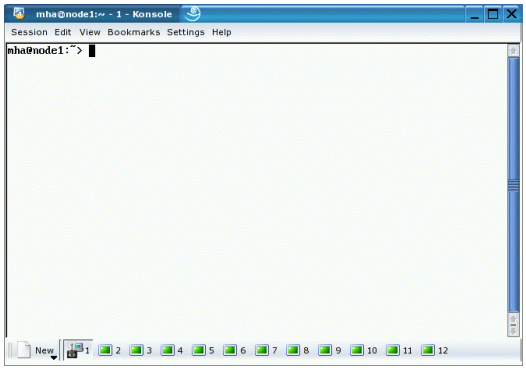
   
   

   Click on the icon representing the user, or enter the user name manually. Then enter the password - note that you will not see anything you type for security reasons (someone might be looking over your shoulder, for example). We suggest to keep *KDE*, then press *GO*.

2. Open two KDE *Konsoles*. One is for working on the local node which is the installation machine, the other one is for working on all other nodes – simultaneously, via a special *Konsole* feature!

   On Konsole 1 you work as *oracle* and occasionally as *root* on the local (installation) node only.

   On Konsole 2 you open as many shells as you have cluster nodes. Then you enable *Send input to all sessions* in one of them – from now on everything you type there is happening on all other shells, too!
   Warning: Be aware that you need to do some node specific things occasionally, and also that the shell history on the nodes may be different, so using the shell history may not bring the expected result on all nodes when you work in this mode!

   The screenshot shows the *Konsole* we used for our 12-node cluster. We renamed the sessions “1...12” and put session #1 into “Send Input to All Sessions” mode. We configured a shortcut for triggering this mode (Menu: Settings -> Configure Shortcuts) since we needed to turn it on and off many times.

**ALL NODES – Preparing the Systems (as user *root*)**

3. Set the password for user oracle*:*

```
passwd oracle
```

   Optional: create extra */home/oracle* directory:

```
cp -a /etc/skel /home/oracle
chown oracle:oinstall /home/oracle
usermod -d /home/oracle oracle
```

   Alternatively, maybe you would like to change just the home directory to wherever you plan to have your Oracle home directory.

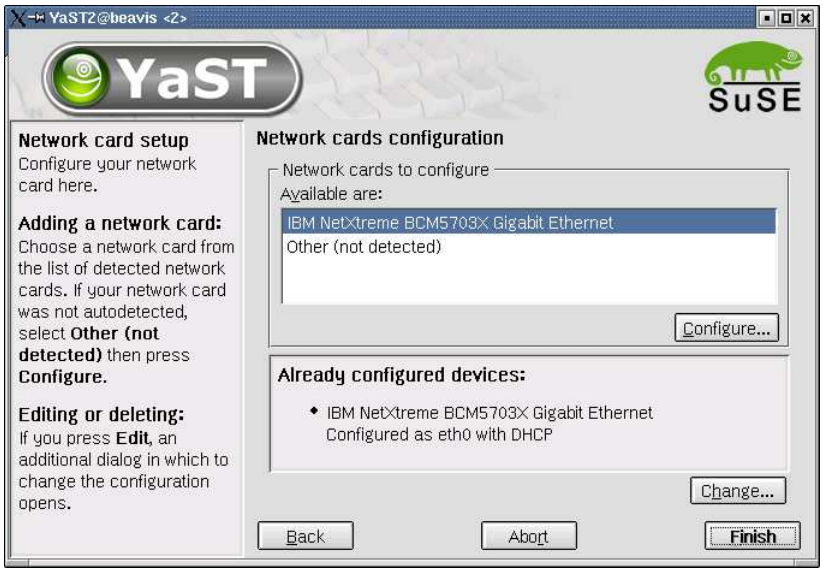
4. Remove gcc 3.2 (**`rpm -e gcc --nodeps`**) to make sure it is not used by the Oracle installer - we prefer an error message during installation over inadvertently using gcc 3.2 (If you choose not to remove it you have to edit *$ORACLE_HOME/bin/genclntsh* as well as *$ORACLE_HOME/bin/genagtsh* and add */opt/gcc295/bin* **in front** of the PATH variable set in those scripts! Then do **`relink all`** after the Oracle installation has finished.)

   Just in case... some Oracle makefiles don't look for "gcc" but for "/usr/bin/gcc". We found files *$ORACLE_HOME/network/lib/env_*.mk* as examples during the 9.2.0.3 patchset installation. As *root* do this:

```
cd /usr/bin
ln -s /opt/gcc29/bin/gcc gcc
ln -s /opt/gcc29/bin/gcc cc
```

5. Set up the network interfaces (internal/external) using the *YaST2* system administration tool (picture). See the SuSE Linux Administration Manual for details.

   You should have at least two network interfaces: one connected to an external network that connects the cluster nodes with the “world”, and one that connects them with each other and which is used for cluster internal communication only.




6. In file */etc/sysconfig/suseconfig* set

```
CHECK_ETC_HOSTS="no"
BEAUTIFY_ETC_HOSTS="no"
```

7. Recommended: Set up /etc/hosts to be completely independent from any DNS server. List all nodes with their external and internal (cluster internal network) names and IP addresses, and specify both long and short names (FQDN – with domain name, and just the host name).

   Example */etc/*hosts for a 2 node cluster (we only list the new, additional entries and not the ones already there by default, e.g. for *localhost*):

```
...
# external names/network
10.0.0.101   node1.mydomain.com node1
10.0.0.102   node2.mydomain.com node2
# cluster internal names/network
192.168.0.101   node1-local.mydomain.com node1-local
192.168.0.102   node2-local.mydomain.com node2-local
```

8. Enable the network services *rsh* (remote shell) anc *rcp* (remote copy).

   Edit */etc/inetd.conf* (or use the YaST2 module for *inetd*): remove the "#" in front of "shell..." and "login...". One service is needed for *rsh*, the other one for *rcp*. There are two lines for each, the one with the additional "-a" option performs hostname verification via reverse DNS lookup before accepting a connection. Since we don't rely on DNS but on hard-coded hostnames in */etc/hosts* - see above - we don't need this.)

   Two commands to turn the service on at boot time and to start it immediately:

```
chkconfig inetd on
rcinetd on
```

9. Enable password-less *rsh* and *rcp* for user *oracle* (see `man rhosts` for details). This is needed for the Oracle installer and some Oracle tools like *netca* (Oracle Network Configuration Assistant), but it is useful for the cluster administrator too since it provides a very quick way to visit other nodes in the cluster:

   * Working as user *oracle*, create a new file *.rhosts* in the Oracle users home directory. Contents of this file: the names of each node (incl. the one you are on) in the cluster, including the domain name. Example (using the 2-node example introduced above):

```
node1-local.mydomain.com
node2-local.mydomain.com
```

   This example only uses the local node names, i.e. you also have to use the local names in your *rsh* and *rcp* commands because the external names won't work. This restricts the password less *r*-commands to the internal cluster network.

   * Restrict the permissions of the new *.rhosts* file:

```
chmod 600 ~/.rhosts
```

   * Check if you can *rsh* and *rcp* - as user oracle - from any node to any other node in the cluster (no need to test all possible combinations).

Example: We are on *node1*. First we copy local file */etc/passwd* to node2. Then we login on node2 and check if the file is there. This tests both services *rsh* and *rcp*.

```
oracle@node1:~> rcp /etc/passwd node2:
oracle@node1:~> rsh node2
oracle@node2:~> ls passwd
passwd
oracle@node2:~> rm passwd
oracle@node2:~> ^D (press CTRL-D to logout)
oracle@node1:~>
```

10. Run **`rcoracle start`**. This sets the kernel parameters which have been set to Oracle recommended default values in the default configuration file */etc/sysconfig/oracle* that comes as part of package *orarun*.

11. Recommended: Install and configure *xntpd* to synchronize the time/date on all nodes. This keeps the system clocks synchronized to within less than a tenth of a second on all nodes.

12. Edit */etc/profile.d/oracle.[c]sh* and set ORACLE_SID to some SID[#nodenumber] for each node (like *rac1, rac2*). Make sure to edit BOTH files (.sh and .csh) because there are Oracle scripts used during installation that use a C-shell, a Korn shell or a Bourne shell.

13. **Install and configure PolyServe Matrix Server –** see the next page.

## Installation: PolyServe Matrix Server

Please refer to the PolyServe Matrix Server Release Notes and the Installation Guide on the PolyServe MatrixLink website.

The basic Oracle installation will require a minimum of two PolyServe filesystems. One to contain the shared Oracle Home and another for all Oracle database files (e.g., datafiles, redo logs, archived logs, OCMS quorum disk, control files, etc).

Mount the cluster filesystem for the database files using the special "DBOPTIMIZED" mount option. The cluster filesystem for the shared Oracle Home should be a standard shared mount. As such, the normal buffered I/O interface will be used for I/O to and from files in the Oracle Home, but all database I/O will be *direct I/O* (it will not have to pass through the operating systems buffer).

In this document we use /opt/oracle for the shared Oracle home (ORACLE_BASE) and */var/opt/oracle* for the database files. Change the owner of the directories to user "oracle", group "dba".

Matrix Server includes a cluster filesystem and its own cluster management tools (GUI, CLI). For the purposes of the installation all we use in the following steps is the cluster filesystem. In addition to some unique features of the cluster filesystem PolyServe Matrix Server product offers a cluster management, SAN management and HA framework. This is not to be confused, however, with the Oracle Cluster Management Services (OCMS). OCMS, often referred to as OSD clusterware, is still used for Oracle 9*i* RAC when using PolyServe Matrix Server. Oracle uses its own clusterware for node membership services. On the other hand, Matrix Server uses its own internal clusterware and DLM for filesystem integrity and HA features.

Note: The cluster filesystem for the Oracle cluster manager quorum and srvconfig files must be on a PolyServe filesystem that is mounted using the DB_OPTIMIZED option! These files should be created before using Oracle Universal Installer for the first time. To create these files, use the following type of dd(1) command:

```
$ dd if=/dev/zero of=/var/opt/oracle/quorum.dbf bs=1024k count=8
$ dd if=/dev/zero of=/var/opt/oracle/srvconf.dbf bs=1024k count=50
```

Additional documentation for PolyServe MxS 1.2:

- MxS Tech Tip #3: Creating a Shared Oracle Home
  https://ssl.polyserve.com/matrixlink/products/MxSv1.2.0/FAQs+and+Tech+Tips/TT0003.pdf
- PolyServe Matrix Server Installation Guide (SLES8)
  https://ssl.polyserve.com/matrixlink/productdetail.php?pid=8&detail=Product+Documentation

# Installation: Oracle 9*i* R2 RAC

**NODE #1 (installation node)**

Info: Unlike most other RAC installations where the Oracle home directory is shared, the entire installation can in our case be done completely on just one - on any! - of the nodes in the cluster!

14. As user "oracle" run `./runInstaller` to **install cluster manager**. For quorum file enter something like this: */var/opt/oracle/quorum* (or wherever you mounted the cluster filesystem for the datafiles)

> **What to install/settings:** Install the *Cluster Manager* only. When the installer asks you for the names of the nodes to install it on, you **must specify only one node** and no others! After all, we install on a shared Oracle home so we install on all nodes simultaneously when we install on any one of them! We are turning this installation into the full cluster later.

Caution: For this shared Oracle home installation you must enter only ONE node in the Oracle cluster manager installation window! We install only on ONE node this time, and because of the cluster filesystem all other nodes share this installation!

Info: A big advantage of this installation type is that we avoid all the complications of the Oracle installer copying part of the installed Oracle home to the other nodes, which in 9*i* R2 still has some bugs and sometimes even hangs at one or the other step (randomly).

Exit the installer when you're done installing the cluster manager. If you select "Next install" to install the patchset (next point) right from there the installer will crash (known Oracle installer bug).

15. As oracle: `./runInstaller` - change source to where you saved the 9.2.0.2 (or later) patchset and install the 920x ($x >= 2$) patch for "Cluster Manager"

Info: The installation of the patchset is HIGHLY recommended since beginning with 9.2.0.2 Oracle no longer uses "*watchdogd*" but a kernel-module (written by Oracle and included in SuSE kernels) called *hangcheck-timer*, which has many significant advantages over the old "*watchdogd*"! That is why we recommend not to bother using *watchdogd* and to upgrade **immediately** to the latest Oracle patchset.

16. Create the quorum file:

```
dd if=/dev/zero of=/var/opt/oracle/quorum bs=1024k count=4
```

17. Edit */etc/sysconfig/oracle* to enable start of OCM and GSD (GSD will work only later after the full software is installed)

```
START_ORACLE_DB_OCM="yes"
START_ORACLE_DB_GSD="yes"
```

18. Start OCM and *hangcheck-timer*. We need to create this "one-node cluster" because the installer offers the RAC installation option only when it detects a running *oracm*:

```
rcoracle start
```

If you didn't install the (latest) SuSE/UnitedLinux update kernel you will get an error about a missing module "iofence-timer". Please note that in order to get a supported and certified system for Oracle RAC you must install *at least* Service Pack #1.

Ignore the error about GSD not found – of course it's not there yet, since we didn't install anything else but the cluster manager.

19. Check processes and *$ORACLE_HOME/oracm.log/cm.log* if *oracm* is running. Check */var/log/messages* and *cm.log* if there are problems.

The end of cm.log should look somewhat like this:

```
....
HandleUpdate(): SYNC(2) from node(0) completed {Thu May 1 18:20:19 2003}
HandleUpdate(): NODE(0) IS ACTIVE MEMBER OF CLUSTER {Thu May 1 18:20:19 2003}
NMEVENT_RECONFIG [00][00][00][00][00][00][00][0f] {Thu May 1 18:20:20 2003 }
Successful reconfiguration,  1 active node(s) node 0 is the master, my node
num is 0 (reconfig 1)
```

20. As user *oracle*, pre-create the shared configuration file:

```
dd if=/dev/zero of=/var/opt/oracle/quorum bs=1024k count=50
```

(On large clusters it needs to be much larger than the recommended value of 20 MB - the above command creates a 50 MB file for the 12-node cluster we used.)

21. As user *oracle* and in the same shell you are going to start the installer from, tell the installer where the shared configuration file should be placed:

```
export SRVM_SHARED_CONFIG=/var/opt/oracle/SharedConfig
```

22. **Install the Oracle RAC software.** As user *oracle*:

```
./runInstaller
```

The installer is going to detect the running Oracle Cluster Manager and through it all nodes that are part of the cluster, and show them to you. In this case with a shared Oracle home there should be only one node up if you correctly followed the steps above.

Select "Software Only", i.e. no database creation (we want to upgrade to the latest Oracle 9.2.0.x patchset first).

Finish the installation, then exit the installer.

23. As user *oracle*, in *$ORACLE_BASE/oui/bin/linux/* do this to fix a bug in the OUI installation:

```
cd $ORACLE_BASE/oui/bin/linux/
ln -s libclntsh.so.9.0 libclntsh.so
```

24. As oracle: **`./runInstaller`**

As source select the 920x patchset directory (*./stage/products.jar*)
Install 920x patchset (we already patched the Cluster Manager earlier)

25. Copy the installation node files */etc/oratab* and */etc/oraInst.loc* to the same location on all other nodes, making sure the owner remains the same, "oracle:dba" for *oratab*, "root" for *oraInst.loc*!

26. Stop everything currently running:

```
rcoracle stop
```

27. As user *oracle*: Create per-node cluster manager directories because the configuration and the logfiles for the cluster manager are per-node. We still keep it on the cluster filesystem because that has the advantage that we can keep it on the shared storage, independent from the individual nodes, which makes them easier to exchange and the configuration easier to administer.

First check the permissions of *$ORACLE_HOME/oracm/*, if it is world writable do a "chmod -R o-w $ORACLE_HOME/oracm" (bug in Oracle installer).

We use PolyServe's Context Dependent Symbolic Links (CDSL) feature to accomplish this. Those links are interpreted differently for each node so each node will see a different - its own - directory.

For our 12-node cluster and nodenames conveniently consisting of "node##" the commands are:

```
for n in 1 2 3 4 5 6 7 8 9 10 11 12; do
  cp -a oracm .oracm.node$i
done
rm -r oracm
ln -s .oracm.{HOSTNAME} oracm
```

Note: Make sure to use the correct $HOSTNAMEs, i.e. check this variable on all nodes! The directories must have the correct name "*oracm.$HOSTNAME*". We use "dot" files (*.oracm.**) but this is a matter of personal preferences.

**ALL NODES**

28. Edit *$ORACLE_HOME/oracm/admin/cmcfg.ora* and add the other nodes to "PrivateNodeNames=..." and "PublicNodeNames=...". Set "HostName=..." to the individual hostname on each node! We use the same path on each node but because "*oracm*" is a PolyServe CDSL (link) we end up in a per-host directory just like we wanted. Read the PolyServe Matrix Server manual for details about CDSL, it is important that you are familiar with this concept.

29. Let's now start the cluster manager on all nodes. This time GSD is started too, which we only just installed. GSD is needed by OEM and by "dbca".

    As *oracle*: **`srvctl -init`**
    As *root*: **`rcoracle start`**

    If GSD does not start: In */etc/init.d/*oracle (*rcoracle*) we use *gsdctl*. Should "gsdctl" be unable to start GSD try using *$ORACLE_HOME/bin/gsd.sh* instead. It is almost identical to *$ORACLE_HOME/bin/gsdctl* which is a shell script, too.

30. Go to $ORACLE_BASE and create a link to the shared NFS mounted directory

```
cd $ORACLE_BASE
ln -s /var/opt/oracle oradata
```

    (assuming you mounted the directory under */var/opt/oracle*)

31. Go to $ORACLE_HOME and create a link to the shared NFS mounted directory

```
cd $ORACLE_HOME
rm -r dbs
ln -s /var/opt/oracle dbs
```

    (assuming you mounted the directory under */var/opt/oracle*)

**FINISHED**

Now the software is installed and ready, and the cluster manager and the GSD are up and running, we are ready to create a database! The following steps are for creating a demo database and can be ignored if you prepare your system for a “real” database, using your own scripts and configuration. Sometimes the Java Oracle tools won't work and you have to create the setup manually. Check Metalink if Oracle has any bugfixes.

***Optional* - NODE #1 (installation node)**

32. As user *oracle*: run **`netca`** to create an Oracle network configuration
    The Network Configuration Assistant should detect the running cluster manager and offer a cluster configuration option! You must at least configure a listener. You can accept all the defaults, i.e. simply press NEXT until the listener configuration is done.

33. Run **`lsnrctl start`** on ALL NODES.

34. We highly recommend to create the database manually. "dbca" has various issues and limitations especially when trying to create a cluster database, and here again especially when trying to do this on a cluster filesystem - "dbca" expects raw devices and is not able to detect a cluster filesystem.

35. Should you want to try, as oracle, run:

    ```
    dbca -datafileDestination $ORACLE_HOME/dbs
    ```

    Set up a database. Without the -datafileDestination parameter dbca assumes (and checks for!) raw devices which we don't use here! If there's an error right at the start, try restarting the cluster manager and GSD via "rcoracle stop; rcoracle start".

36. Edit */etc/sysconfig/oracle* to start additional services, e.g. the Oracle listener. If you set START_ORACLE_DB="yes" you have to edit /etc/oratab (on ALL NODES) and change the last letter in the line for your database (usually just one line, at the bottom) to "Y" or no database will be started.

**Please check *http://www.suse.com/oracle/* -> *Support Matrix* for any updates to these instructions!**

# END

# Appendix – Package *orarun.rpm*

## *Package information*

**Name :** orarun
**License :** GPL
**Summary :** Environment for running Oracle products

**Description:** This package
- sets the Oracle environment variables for each user, like ORACLE_HOME and PATH
- sets the recommended kernel parameters, e.g. SHMMAX
- provides for automated start/stop of Oracle processes at system startup/shutdown

You may want to or even have to edit /etc/profile.d/oracle.[c]sh (environment variables), /etc/sysconfig/oracle (which components to start/stop, values for kernel parameters).

**File list:**

| | |
|---|---|
| Shell script: | `/etc/init.d/oracle` |
| Shell variables: | `/etc/profile.d/oracle.csh` |
| C-Shell variables: | `/etc/profile.d/oracle.sh` |
| Shell variables: | `/etc/sysconfig/oracle` |
| Documentation: | `/usr/share/doc/packages/orarun/README` |

Plus: Several symbolic links.

**1.)** It **provides the environment variables** for running Oracle. It does so for all users, by placing it in the `/etc/profile.d/` directory, and it provides a file each for Bourne shell users and for C shells (UnitedLinux default shell is *bash*, a Bourne shell). Some administrators may not like to have an Oracle environment set for all users, in this case just put the word “exit” at the very top of the file!

**2.)** The package also provides a script for **automated startup and shutdown** (when the system starts up or shuts down) of the Oracle database and of other Oracle components like the Apache webserver or the listener. Which components should be started by the script can be controlled by setting certain variables to *yes* or *no* in the text file `/etc/sysconfig/oracle`

**3.)** The startup script also takes care of **setting the kernel parameters** for running Oracle. This is because as of kernel 2.4 all the parameters that need be be set according to the Oracle recommendations are dynamically (during runtime) adjustable, so no kernel rebuilding is necessary! The file storing the values is `/etc/sysconfig/oracle`. Have a look and edit it according to the instructions embedded in that text file! For small to medium databases no values need to be adjusted at all since UnitedLinux already provides reasonable defaults.

Note that the startup script does its two tasks - setting kernel parameters and starting/stopping Oracle processes - independent of one another, so even if START_ORACLE_DB is set to “no” the kernel parameters will still be set if this is set to “yes” (which is the default).

**4.)** This package has dependencies on the UL packages *pdksh*, *gcc_old*, *make*, *binutils*, *glibc-locale*, *glibc-devel*, *libaio*, *libaio-devel* – all of them are necessary for a successful Oracle installation. *libaio* is needed for asynchronous I/O.