Oracle RAC on Linux:
Database Service Utopia?

Date:
August 2003

Overview:
In recent months, Oracle RAC on Linux has received
unprecedented publicity as a technology for
delivering a complete solution for meeting your
enterprise database computing requirements. This
paper provides guidance on core elements required
to deliver these benefits to your enterprise.

Target Audience: IT Executives and DBMS
technologists

"cheaper, faster and more reliable than any other environment around."

Larry Ellison talking about Oracle RAC on Linux at LinuxWorld Expo, August 2002.

About Linxcel Europe Ltd

Linxcel Europe Ltd is a privately held consultancy working to bring you the most cost effective, high performance Oracle environment, whether it's a simple traditional single instance solution or a multi-instance, multi-node Real Application Cluster (RAC) operating on state of the art Intel/Linux blade technologies. The Linxcel philosophy and best practices for delivering a complete end-to-end Oracle system are set out in the following book:

High-Performance Oracle: Proven Methods for Achieving Optimum
Performance and Availability
Author: Geoff Ingram
Publisher: John Wiley Inc, August 2002
ISBN: 0-471-22436-7

Founded by a professional technology management team with support from key customers, we aim to deliver a fast track to your preferred deployment. We deliver these solutions through:

- our consultancy services
- *empower! for Oracle* our graphical tuning tool, which includes RAC support
- our Corporate Advantage membership of the Oracle PartnerNetwork

Specializing in delivering solutions that integrate with your existing hardware and software partners, we have extensive experience with Sun, HP, IBM and Intel/Linux configurations.

We provide consultancy services for:

- RAC on Linux proof-of-concept studies, including platform, OS, and storage selection
- Performance management, tuning, and tools selection
- Oracle environment build standardisation
- Stress testing and benchmarks, including those specified by the TPC organisation
- Database end-to-end backup and recovery based on RMAN
- Oracle and 3rd party non-stop and high availability solutions
- Conversions to Oracle from other DBMS
- Bespoke Java, Pro*C and OCI Development

For more information, contact consulting@linxcel.co.uk

Introduction

The goal of this paper is to provide guidance on the key elements to be considered in order to deliver an enterprise class database solution using Oracle and RAC technology in pursuit of what we call "database service utopia" at a highly attractive price point.

The following topics are covered:

- Defining requirements for an enterprise database solution
- How Oracle on RAC delivers the solution
- A brief history of Linux and Oracle
- Oracle's Unbreakable Linux Initiative
- Choosing your technology stack
- How to run a do-it-yourself "RAC readiness" test

The information has been gathered over time from real-world projects that initially started in Q4 2001.

Deleted:

A Short History of Oracle on Linux

It's helpful to have a historical perspective on Linux itself, in order to understand the sea change that took place in 2002 with respect to the emergence of enterprise versions of Linux, and how those changes lead to Oracle's Unbreakable Linux Initiative.

The initial public release of Linux appeared back in September 1991. Many people are surprised that Linux is as old as that. It's certainly not the new kid on the block. The driver for the creation of Linux itself was somewhat mundane, given the explosion in its use over time. Linus Torvalds, a young professor at the University of Helsinki, was frustrated by the lack of UNIX-style operating systems for 386 architecture PCs. At the time, the alternatives were MS-DOS, or Minix, a stripped down operating system developed by Professor Andrew Tanenbaum. Tanenbaum himself is the author of a legendary book on TCP/IP networking.

Linux was designed from the ground up to be source-code compatible with UNIX; a program that compiles under UNIX will usually compile under Linux. This is one of the key features of Linux, by design, that typically makes the porting of a C program to Linux from other UNIX a straightforward task most of the time. That's worth keeping in mind, when you commit yourself to a pilot project to port an application to Linux.

Linux is Open Source, which means that the source code is available, and anyone can modify and improve it. Whether those changes ever make it back into mainstream Linux depends on the Open Source community itself. Given the huge numbers of programmers working on Linux today, you can be sure that problems are identified quickly, good ideas are incorporated, and bad ones are discarded. Open Source is probably the ultimate meritocratic and democratic environment for software development.

Torvalds took a decision to release code under the General Public License (GPL) from the Free Software Foundation. This farsightedness means that you can sell Linux and make changes and customisations to the operating system in any way you like – but you can't turn it into a proprietary operating system and sell it. This restriction is imposed by the GPL, which means that any software derived from GPL license code is also subject to GPL. In the vernacular, when Linux is described as free software, that means free as in "freedom of speech not beer." And clearly there is no restriction in selling "closed-source" software to run on the Linux operating system – Oracle is a prime example.

Regular requests for Oracle to port the DBMS to Linux – for example in Internet USENET discussion groups - began to appear soon afterwards. Bowing to demand, Oracle demonstrated its commitment to Linux as far back as August 1999, when Oracle8i appeared as a download. Several hundred thousand downloads later, the success of this strategy is clear. Such is the success that during 2002, Oracle on Linux was released to run under mainframe Linux environments, such as IBM zLinux, purely on the basis of customer demand.

Enterprise database solution

Typical Requirements

Before going into any kind of technical discussion let's step back for a moment and look at the big picture. Imagine you are setting up a enterprise database computing environment in a new company from scratch.

Let's put aside those Oracle versus Microsoft SQL Server versus IBM UDB versus Sybase flame wars that periodically break out on the Internet. You know the ones, that start "my database is better than yours because…" In reality the choice of database technology you run in your organization is not a question of religion.

It's a matter of hard analysis and facts, and extensive testing: in short it means putting in some hard work. If you're prepared to do that, you can deliver an enterprise database solution for your organization that represents database service utopia.

So what are the requirements you need to meet, in order to get there? Ultimately, these four components should be considered as the core drivers that the end user community in any enterprise are particularly interested in:

- Performance
- Availability
- Scalability
- Cost

It's interesting that technology considerations are nowhere to be seen at this stage when you set out the business-driven requirements like this – and the requirements are database-vendor agnostic.

At first sight, it's reasonable to question whether you can realistically measure the business benefits delivered by the database organization in your company under four simply stated categories. Let's consider a selection of headings that are apparently missing: what about manageability, or supportability, people resources and availability or 3rd party vendor support?

Ultimately, these are sub-headings of availability. If you can't get the right people with the right skills, then you'll suffer unplanned outages – and that compromises availability. If the system is difficult to manage, without the right tools, then mistakes will be made, leading to unplanned outages. Once again this compromises availability.

The remainder of this paper sets out to explain why and how Oracle RAC on Linux is the sweet-spot that can deliver performance, availability, and scalability to form a complete technical solution for your enterprise database service, at a price that in most cases will be substantially below deploying traditional Oracle infrastructures.

How Oracle RAC on Linux delivers on the goals.

At the heart of the RAC architecture is a single database shared co-operatively between multiple servers **at the same time**. As such, RAC is unique and fundamentally different from the traditional definition of clustering in the DBMS world. Traditional clustering usually means having a standby server available to take on the existing workload when a node fails, by remounting storage from the failed node onto the standby node, and migrating services on to it. With RAC, all nodes are active at the same time.

It's important to understand how Oracle RAC on Linux specifically delivers on our stated goals:

- Performance
- Availability
- Scalability
- Cost

Performance

In terms of performance, you deploy Oracle RAC on Linux on low cost commodity servers, running Intel or AMD processors. These platforms provide outstanding performance at a very attractive price. In fact, you could even view Linux as an enabling technology that allows you to leverage the exceptional price/performance of these lower cost commodity servers for your Oracle systems.

Looking to the future, both Intel and AMD have articulated processor development roadmaps for the long term. The existence of roadmaps for all technologies you deploy is important, because it means you have future proofed your architecture and your investment.

RAC itself brings additional performance capability into the equation, because all CPU and memory resources in your cluster as a whole are available to all workloads.

As usual, you should always view any claims that a particular computing architecture is superior to the rest, with a healthy cynicism. There is no substitute for running a proof of concept exercise yourself to measure and understand performance of RAC on Linux against your existing infrastructure. To facilitate this, a later section of this paper describes how to run a do-it-yourself "RAC readiness" test using free Open Source software available on the Internet.

Availability

The RAC architecture itself is a key factor in delivering availability. All your client applications have access to the shared cluster database through any single node in your cluster. The cluster database is truly a shared resource. If a node dies unexpectedly, or you need to perform hardware maintenance on a node, then Oracle features such as Transparent Application Failover (TAF) are available to ensure that existing sessions are re-directed to any available node. Oracle's networking and load balancing technology directs new connections to the least heavily loaded nodes automatically.

Client applications use a simple name, held in a centralized name server (including LDAP), to connect to any node in the cluster database. If a node is relocated, or nodes are added to or removed from the cluster, no client application changes whatever are required. Not surprising then that Oracle positioned RAC as the first stage in their Grid computing architecture delivered in Oracle 10G. Grid computing means computing-as-utility, where the end user neither knows nor cares about the technology used to deliver business data, just that the data is available where and when required. Just like electricity from the grid.

So should you consider RAC on another hardware platform or operating system? Of course you should. But keep in mind that potential availability enhancements exist specifically for RAC on Linux, as compared to RAC on other platforms. This happens for two main reasons. First, through the Unbreakable Linux Initiative (described later), Oracle can provide support for the operating system as well as the Oracle DBMS, simplifying the solution. Second, for RAC on Linux, Oracle itself develops the code for some of the components (e.g. the cluster management software) that are developed and supported by the operating system vendor or ISVs on other platforms. Ultimately, the integration and quality of your 3rd party support process determines how quickly you can get things up and running in a crisis.

Scalability

DBMS platform hardware purchases typically have involved a large initial capital outlay, in order to accommodate resources capable of dealing with the projected workload for day one, up to some point in the future. As database loads grow, then new processors and CPU are installed, until eventually the server is fully configured.

The next step is another large capital outlay on a new server., whilst the previous system is either retired or moved onto new duties. Whilst there is nothing wrong per se with this approach, is does have significant cost and service implications for the end solution. This approach is often referred to as "scale up".

The RAC architecture is fundamentally different. Using RAC allows you to grow your computing resources in smaller incremental steps as demands increases. RAC allows you to "scale out" by purchasing additional low cost commodity servers and including them in the RAC configuration. Its worth noting that even with RAC, the model of "scale up" can also be adopted.

Once again, you need to demonstrate the scalability of the RAC Architecture itself to your own satisfaction, as explained later in this paper.

Cost

So how do Oracle, RAC, Linux and Intel commodity servers impact the cost model for your database service offering in your organisation?

While it's not realistic to try and create a generic model that fits all organisations, key things to consider in the cost of ownership model that are common to nearly all organisations are: storage, network, server, backup, job scheduling, system and database administration costs. The following chart provides a real world example of how one client compared the costs of their existing proprietary UNIX/RISC configurations running Oracle and their proposed Oracle, RAC, Linux deployment.
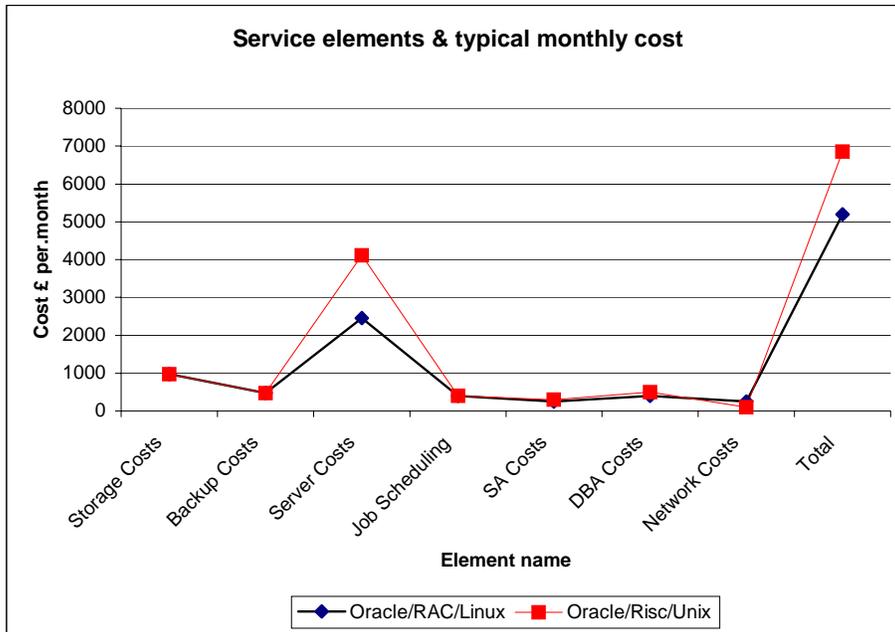
A note about RAC versus Oracle Parallel Server

At this point it's worth mentioning Oracle's previous incarnation of cluster database technology: Oracle Parallel Server (OPS). When Oracle RAC was first announced, there was a feeling expressed in some parts of the Oracle community that RAC was simply a re-badging of Oracle Parallel Server. Although in broad terms OPS set out to deliver on the same goals as RAC, they aren't really comparable.

RAC is fundamentally different because, whereas in OPS data block migration between nodes in the cluster was communicated by expensive disk writes followed by reads, RAC uses a high-speed private network (referred to as the interconnect), to perform the same function. This is typically orders-of-magnitude faster. And whereas OPS needed careful partitioning of data and workloads for best performance, leading to manageability challenges, RAC has no such restrictions: all nodes in a RAC cluster are peers in terms of running the available workload. It's no longer necessary to partition workloads. The interconnect network is a key component of "global cache fusion", a term used to describe the process by which Oracle keeps the individual database memory caches (SGAs) on each node in sync, cluster-wide.

Oracle's Unbreakable Linux Initiative

Alongside the successes, the widespread take-up of Oracle on Linux highlighted some of challenges in deploying Oracle on Linux in the enterprise. This section analyzes the state of Linux, in terms of performance, scalability, and how those requirements led to Oracle's Unbreakable Linux initiative.

At the close of 2001, some Oracle on Linux performance issues were evident and widely known under heavy workload - the operating system had a tendency to "hit the wall". It certainly wasn't unbreakable.

For commercial organizations, this was a critical showstopper preventing widespread deployment.

Alongside the availability aspects, the proliferation of Linux distributions caused a logistical problem for

- Oracle in terms of both providing multiple ports of Oracle for those distributions
- Independent Software Vendors (ISVs) that provide the products such as backup/restore, monitoring, and performance management, needed to support Oracle in a production environment in a commercial organization
- Independent Hardware Vendors (IHVs) that provide servers, storage, and network infrastructure

The challenge that needed to be met was to provide a complete infrastructure and environment where all products and technologies required to run an Oracle database were guaranteed to be interoperable and certified to run alongside each other. In short, the challenge was one of **technology alignment**. With the (at that time) 3 month release cycles of many Linux distributions, it was almost impossible to deliver a stable environments with a complete set of certified and interoperable components need to run a production Oracle database on Linux. Such deployments typically suffered unplanned downtime, and compromised availability. The state of flux also meant that your in-house enterprise-wide technologies, such as the Storage Area Network (SAN) were unlikely to be compatible with the database and operating system.

A quantum leap was required in order to overcome the performance and vendor alignment challenges of Oracle on Linux. This leap took the form of the emergence of enterprise-ready versions of Linux, alongside the Oracle Unbreakable Linux Partner Initiative. Under the Initiative, Oracle, ISVs, Linux vendors, and IHVs work together to deliver an enterprise ready environment in which supportability and interoperability is guaranteed.

Red Hat was the first Linux distribution to step forward and address these issues, with its Red Hat Enterprise Linux AS (formerly Red Hat Linux Advanced Server) in March 2002. Red Hat Enterprise Linux AS addressed the performance challenges by collaborating closely with Oracle to eliminate performance bottlenecks in Linux to produce an OS that scaled well to 8 processors in a Symmetric Multi-Processing (SMP) architecture. The most important of these are briefly described below:

- Support for asynchronous I/O leads to higher I/O throughput that enterprise database architects already take for granted on more mature RISC based operating systems.

- Reductions in I/O spinlock contention reduce queues for execution of critical sections of code on SMP systems, leading to more efficient hardware resource usage.

- Improved process scheduling leads to better scalability of multi process workloads, through more efficient CPU usage.

- Elimination of unnecessary memory copies (usually referred to as bounce buffer elimination) for systems with > 1GB of memory, lead to reduced memory and CPU utilization.

It's a sign of the success of the collaboration that these changes have been agreed for incorporation back into mainstream Linux code.

Alongside the performance enhancements, Red Hat Enterprise Linux AS included as one of its goals a roadmap to produce new releases on a 12 to 18 month cycle. This relatively long release cycle in turn enabled Oracle, ISVs and IHVs to align their technologies on a stable and infrequently changing platform, leading to a stable environment that can deliver the performance, availability, and scalability required.

The UnitedLinux industry consortium, including SuSEs enterprise version of Linux (SLES 8 at the time of writing) soon joined the Unbreakable Linux Initiative to provide an alternative to Red Hat Enterprise Linux AS, with the same goals of stable releases and enhanced performance.

Supportability (that leads directly to increased availability) is a feature of Unbreakable Linux. You can now use Oracle as a one-stop-shop for both your DBMS and operating system support – this support integration can fast track your support problem to its root cause, be it the DBMS or the operating system, without requiring the customer to manage the process. This finally puts Oracle DBMS supportability (on Linux at least) on a level playing field with Microsoft who, through their ownership of both the DBMS platform (SQL Server) and the operating system (Windows) have always been able to provide a well-integrated support experience.

Choosing your end-to-end technology stack

Before you even decide you run a pilot project to test the claims of RAC on Linux, you should consider the end game as early as possible on your roadmap – and the end game is production deployment. Clearly, there's a lot more technology required to run a production Oracle database than simply an operating system and DBMS software.

This section aims to provide a comprehensive checklist of the technology components you need to deliver into your production environment, with some examples provided as a reference. If your technologies don't appear in the categories mentioned then you need to start working with your technology partners as soon as possible to ensure everything is in place in time for your production rollout - with sufficient lead time for complete testing.

The example assumes that you put a roadmap in place for delivery of your enterprise Linux on RAC architecture prior to the availability of Red Hat AS in Q1 2002, and that you deploy on the corporate Storage Area Network (SAN).

Before you begin, choose your Oracle and operating system version (SuSE SLES or Red Hat AS) carefully. In particular, your ISV and HSV partners probably certify their products against a specific release of the operating system. Some of them may have specific requirements for a particular Linux kernel erratum patch.

You need to be realistic from the outset, by recognizing that you may hit problems during testing that lead to a requirement for operating system patches. When this occurs, the dependencies need to be re-visited. The emergence of enterprise Linux, and longer release cycles, means such issues should be more manageable over time.

Note: In general, storage-related technologies are more sensitive to kernel releases.

Technology	Vendor/Product	Certified on Red Hat 2.1 AS
Storage Array	EMC Symmetrix	Q1 2003
Host Bus Adapter (SAN to storage connectivity)	Emulex	Q1 2003
	Qlogic	Q1 2003
SAN Multipathing	EMC PowerPath	Q1 2003
Network interconnect multipathing (known as NIC teaming)	Hanicd (Open Source)	Q4 2002
	Intel iANS	Q3 2002
Linux Kernel	Red Hat AS 2.4.9 e.12	Q4 2002
Shared Filesystem for Cluster Database	Raw partitions	Q1 2002
	Oracle Cluster Filesystem	Q1 2003
	NetApp Filer	Q4 2002
	PolyServe	Q1 2003
	VERITAS DBAC for Linux	Q4 2003
Monitoring/Alerting	BMC Patrol	Q1 2003
Database Performance Management	VERITAS i3	Q1 2003
	IBM Tivoli	Q1 2003
Cross Platform Scheduling	CA AutoSys	Q4 2002
File system backup/restore	Legato Networker	Q4 2002
	VERITAS Netbackup	Q3 2002
Database backup/restore (via RMAN)	Legato Networker for Oracle	Q4 2002
	VERITAS Netbackup	Q3 2002

This list is by no means comprehensive and is subject to change frequently - it should however provide some guidance on key components and their availability.

Clearly, depending on the technology you chose to deploy under each heading, it wasn't possible to actually deliver your production Oracle RAC configuration on Linux until Q1 2003 at the earliest, despite the availability of the operating system nearly a year earlier. Of course you could have omitted some of them. But in this case, the goal is to deliver unbreakable Linux, which means no omissions. As a result of the alliances now in place under the Unbreakable Linux Initiative, these lag times should be considerably reduced for future release cycles of Oracle and the operating system.

To deliver a true high availability solution, redundancy of components is required. For example, you need multiple-SAN paths to ensure that access to the shared disk doesn't containing a single point of failure in the fiber network itself. This leads to a single option for the configuration used in the example: EMC PowerPath. You need also need multiple interconnect paths between your nodes, to ensure a single network card, or network path, can't act as a single point of failure for the interconnect network.

Storage Considerations and Unbreakable Linux

As explained previously, under the Unbreakable Linux Initiative, Oracle can provide a one-stop-shop for both operating system and Oracle DBMS issues. But there's one very important caveat you need to be aware of. The Unbreakable Linux support option only applies to non-modified Linux kernels.

Any kernel package or extension that modifies your Linux kernel, and which is not available as Open Source and certified for use with RAC, is not supported by Oracle. In effect, it "breaks" Unbreakable Linux. Such modifications are most likely to take place for components in the I/O stack between the host and your shared database storage.

In technical speak, you can run the lsmod command on Linux and look for the word Tainted in the output. If you see it, your kernel has been built with non-GPL (i.e. Open Source) code and contains unsupported kernel modifications.

Here's an example. Say that you install a kernel driver for the host bus adapter card (HBA) that is used to connect your host to the shared SAN storage where your cluster database is located. This driver is not available as Open Source – its non-GPL. In a worst-case scenario, a database corruption occurs during production operation. Because the driver is effectively a "black-box" in the I/O stream from host to storage, and no source code is available to show how it works, no one but the HBA driver supplier is privy to exactly what it does. It's not certain that the driver is responsible for the corruption – but Oracle can't tell. In this scenario, you would need to work with the HBA manufacturer to get support. An alternative is to use an Open Source certified HBA driver instead. But of course this may conflict with your company's standard hardware configuration. In this case, you need to choose which option to take based on the costs and benefits of each approach.

Keep in mind that because RAC requires shared storage for the cluster database, supportability requirements for such storage are stricter than non-RAC configurations. This is because access to the storage must be carefully synchronized. Such synchronization requirements do not apply to a single instance (i.e. non RAC) database being accessed by a single host at a time.

You definitely shouldn't rule out other storage and filesystems just because they make kernel modifications. Those vendors carry out extensive test programs themselves, certify their own products, and provide the support if things go wrong – they are just as committed to a working system as Open Source solutions. Whatever option you choose for your shared storage, extensive testing of realistic workloads during a proof of concept exercise is essential so that you can satisfy yourself that the technology is performance and reliable. That testing should include power cycling hosts that are carrying out an I/O intensive Oracle workload, to ensure that Oracle recovers properly.

You should give consideration to using Oracle Cluster Filesystem (ocfs) for storing your cluster database. This is a purpose built cut-down shared filesystem specifically designed for Oracle RAC databases on Linux. Although you must be careful not to treat it as a generic shared file system by placing **only** Oracle database files upon it, it's free, open source, written and supported by Oracle, and provides the performance of raw partitions with the manageability of a filesystem. At the time of writing, you can't yet use it to share a single Oracle software installation across all cluster nodes. That means a 4-node Oracle cluster requires 4 separate Oracle software installations to support and maintain. As an alternative, generic cluster filesystems such as those from PolyServe and VERITAS provide fully-fledged generic cluster filesystems compatible with Oracle RAC, at a price. They also provide the capability to share a single Oracle software installation between all cluster nodes.

Running a RAC-Readiness Proof of Concept

There's no question of dispensing with your existing enterprise DBMS architecture and replacing it with Oracle RAC on Linux without first running a comprehensive proof-of-concept exercise. This should include quantitative measurements of how Oracle RAC on Linux delivers performance, availability, and scalability compared to what you have today. While this section addresses performance considerations, backup/recovery needs to be included at this stage also.

The proof of concept exercise is the first stage on the route to production deployment. Keep in mind the production deployment must include monitoring, performance management, and backup/recovery amongst others, and should only be considered when you have demonstrated that RAC is ready for deployment in your organization on your infrastructure.

Stress Testing Tools

Performance and scalability should include multi-user CPU, memory, and I/O stress testing. The good news here is that the tools available to do this are available on the Internet for free, as Open Source. Some of them are based on the standard TPC-C OLTP benchmark described at www.tpc.org.

Although TPC-C benchmarks are often portrayed as marketing exercises, the TPC-C itself is a realistic simulation of a real-world OLTP workload, and is a good test of your database performance, and a valid way to compare Oracle performance between different server platforms and operating systems.

Orabm is a multi-user CPU stress test for Oracle databases, based on the read-only transactions in TPC-C. It runs entirely on the server side. Orabm deliverables include a data loader, orabmload, which creates TPC-C compliant test data:

http://linxcel.co.uk/orabm

Deleted: www.dbcool.com

The following chart shows the transactions per second for a single session in Orabm on different 32-bit processors. Using OS monitoring tools, you would expect to see a single CPU at 100% utilization and no physical I/O during the test – in effect, the test measures the Oracle-related performance of a single CPU.
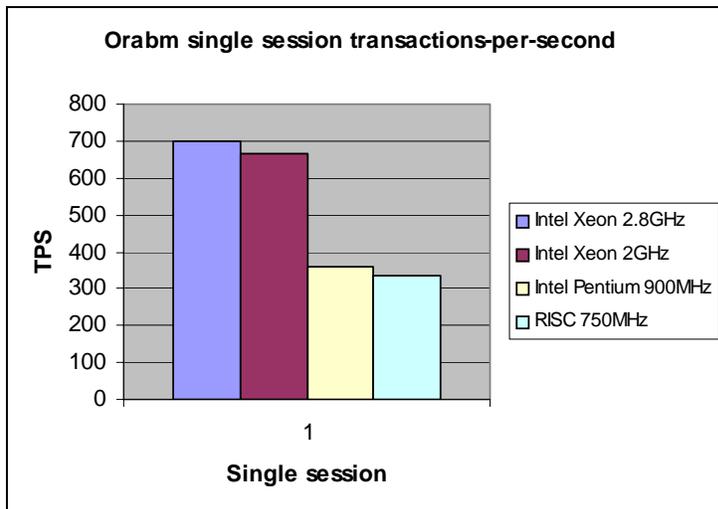
Hammerora is a TPC-C style OLTP test, based on the five standard TPC-C transactions. Like TPC-C it provides a mixed workload that stresses memory, CPU, and I/O at the same time. Data loading takes place using orabmload:

http://hammerora.sourceforge.net

For I/O intensive stress testing, Oracle's own text file loader, SQL*Loader, running in both SQL based and direct-path modes, is ideal for an I/O-only stress tester. SQL*Loader supports the loading of data streams in parallel.

Finally, **empower_checkpoint** is a simple Open Source command line utility that enables you to replay the real I/O generated during an Oracle checkpoint operation without requiring an Oracle database. During a checkpoint, Oracle writes dirty blocks to disk. As such, empower_checkpoint can give you a ballpark figure for the performance of your storage under an intensive Oracle I/O workload in just a few minutes:

http://www.linxcel.co.uk/

Performance Management Tools

Once you have your stress tests up and running, you need tools to analyze the performance. Here is a selection.

STATSPACK

Oracle's own widely used STATSPACK utility is an excellent way to get a high-level view of how your Oracle system is performing. However, it doesn't easily enable you to take a cluster-wide view of performance across all your cluster database instances at once: it's designed for single instance (i.e. non-RAC configurations). You can upload your report to www.oraperf.com for a free analysis of the contents of your STATSPACK report.

Oracle Enterprise Manager (OEM)

OEM can provide useful information on server and database performance for your cluster database, once you have configured a repository database and installed agents on all servers in your cluster.

empower! for Oracle

empower! for Oracle, from Linxcel Europe Ltd, is an agent-free lightweight desktop Oracle performance management and tuning tool, incorporating many features to analyze RAC performance. For example, it takes STATSPACK a step further and enables you take snapshots of performance across the cluster as a whole, using RACPACK, for later comparison. It also includes Event Profile charts to enable you to view Oracle performance for all nodes in your cluster side-by-side in real time, and many other built-in features for identifying and analyzing cluster-wide performance issues. On Linux, *empower!* for Oracle also includes real time graphs of your interconnect network performance. A download (including a free trial of the RAC-enabled Enterprise Edition) is available from http://www.linxcel.co.uk

The following screenshot shows performance of two Oracle RAC instances, EMPOWER3 and EMPOWER4, and the interconnect network traffic between them, while an Oracle Import is in process on each node concurrently. The charts are sliding windows, updated in real time.
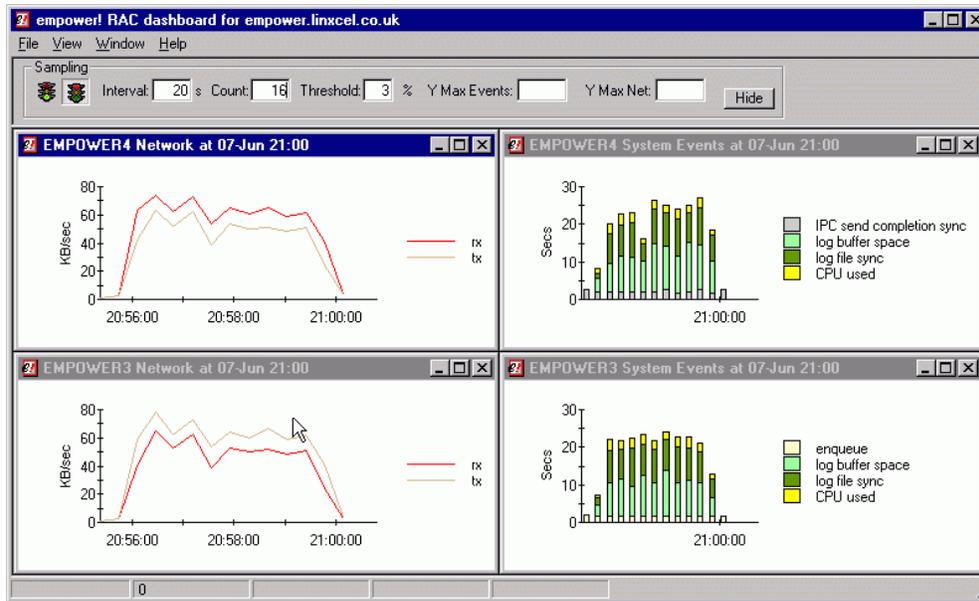
VERITAS i3

VERITAS i3 provides a complete end-to-end performance management solution for Oracle databases, from client to storage. It records extensive performance metrics, 24 x 7, and enables all SQL and PL/SQL executed to be presented across different dimensions such as user, program, and host, at low impact. For mission critical Oracle systems with the highest performance and availability requirements, an enterprise-level Oracle performance management suite like VERITAS i3 is essential. Due to the investment required to purchase and deploy any enterprise-level technology, purchase of such software should include a formal evaluation processes.