An Oracle White Paper
September 2010

Deploying Oracle Database on the Oracle Solaris Platform – An Introduction

Introduction

This document is intended for Oracle DBAs who want to understand the benefits of deploying Oracle databases on the Oracle Solaris 10 environment.

The Oracle Solaris Operating System delivers enterprise-grade performance, massive scalability, flexible virtualization, and unparalleled security. It runs across the entire range of Sun SPARC and x64 platforms from entry level servers to 64-processor servers like the Oracle's Sun SPARC Enterprise M9000 server. Not only does the Oracle Solaris Operating System scale across systems of different sizes, it scales across different technologies and hardware platforms by delivering binary compatibility within the SPARC and x86 processor families.

The Oracle Solaris 10 Operating System introduced new features to enhance manageability, performance and availability to unprecedented levels. The key new features include Solaris Containers for virtualization, Predictive Self-healing for continuous availability, Dtrace for advanced observability, ZFS for next-generation volume management and file system support, and user and process rights management for enhanced security. An Oracle database deployment can take advantage of each of these unique features found in Oracle Solaris 10 Operating System to enhance the manageability, scalability, availability and security of both single and multiple Oracle database instances – all across multiple platform and processor architectures

Proven Scalability

The Oracle database has a proven track record of scaling well both vertically as well as horizontally on the Oracle Solaris 10 platform. For instance, Oracle Database 11g with Oracle Real Application Clusters demonstrated excellent horizontal scalability across 12 Sun SPARC Enterprise T5440 servers on Oracle Solaris 10 running the industry-standard TPC-C workload. The single instance Oracle 11g database also scaled well on a single Oracle Sun SPARC Enterprise M9000, server with 32 sockets, running the industry-standard TPC-H data warehousing benchmark. Oracle believes in empowering its customers to use both horizontal and vertical scalability dimensions to best meet their critical performance and availability criteria.

Business applications deployed on both single and multiple-instances Oracle databases have consistently demonstrated exceptional performance and scalability running online as well as batch based workloads on the Oracle Solaris 10 platform. For instance, the SAP ERP 6.0 2-tier Sales and Distribution benchmark deployed on a single instance Oracle database 10g instance performed best on Sun SPARC Enterprise M9000 server running Oracle Solaris 10. The same benchmark demonstrated near linear scalability of Oracle Real Application Cluster 10g in an SAP environment when deployed on a four node Sun Blade X6270 cluster running Oracle Solaris 10. The PeopleSoft Payroll for North America benchmark, a batch performance benchmark, exhibited linear scalability using 16 job streams on a single Oracle's Sun SPARC Enterprise M4000 server for 240,000 employees and 32 job streams on a single Oracle's Sun SPARC Enterprise M5000 servers for 500,000 employees running Oracle 11g database on the Oracle Solaris 10 platform. The Siebel CRM 8.0 application, an online transaction processing application, scaled linearly from 5000 concurrent users on a single Sun Fire T5220 system to 10,000 concurrent users on two such systems running Oracle database 10gR2 on the Oracle Solaris 10 Operating System.

The Oracle database deployment on the Oracle Solaris 10 platform provides customers the flexibility to select the scaling method that best suites the business systems that they are implementing, from scaling out across smaller building blocks, to scaling up on a large SMP configuration and leveraging the Oracle Solaris Containers features to "scale within". It also enhances application performance and scalability on OLTP and batch workloads across SPARC and x64 systems.

Refer to Appendix 1 for a comprehensive list of industry standard benchmark deployed on Oracle 11g database on the Oracle Solaris 10 environment that continue to set world records in performance, horizontal and vertical scalability and cost-effectiveness.

Protect against faults: Enhance uptime

The Oracle Solaris Operating System provides a proven architecture for building and deploying systems and services capable of Predictive Self Healing, which is a cohesive architecture and methodology for automatically diagnosing, reporting and handling software and hardware fault conditions, thereby enhancing the systems availability. Solaris Fault Manager and Solaris Service Management facility (SMF) are the two key components of Predictive Self Healing technology. The following section describes how an Oracle database deployment can take advantage of Oracle Solaris Predictive Self Healing technology and can continue uninterrupted even when there are hardware and software fault conditions.

Protect against hardware faults: Solaris Fault Manager

Solaris Fault Manager monitors data relating to hardware errors and automatically diagnoses the underlying problem. Once diagnosed, Solaris Fault Manager automatically responds by off-lining faulty components such as a CPU, memory region or I/O channel. The net benefit is that the system continues to operate with the remaining system resources, achieving a graceful degradation rather than an undesired disruption of the entire system.

Figure 2 demonstrates the fault management architecture in a simplistic manner. The fault management architecture is divided into three areas: error handlers, diagnosis engines and agents. A fault or defect in hardware is associated with a set of observed symptoms called errors. The error events are dispatched to software components called diagnosis engines designed to diagnose the underlying problems corresponding to those symptoms. The diagnosis engine then produces fault event that is broadcast to any agents deployed on the system that know how to respond to that particular fault.

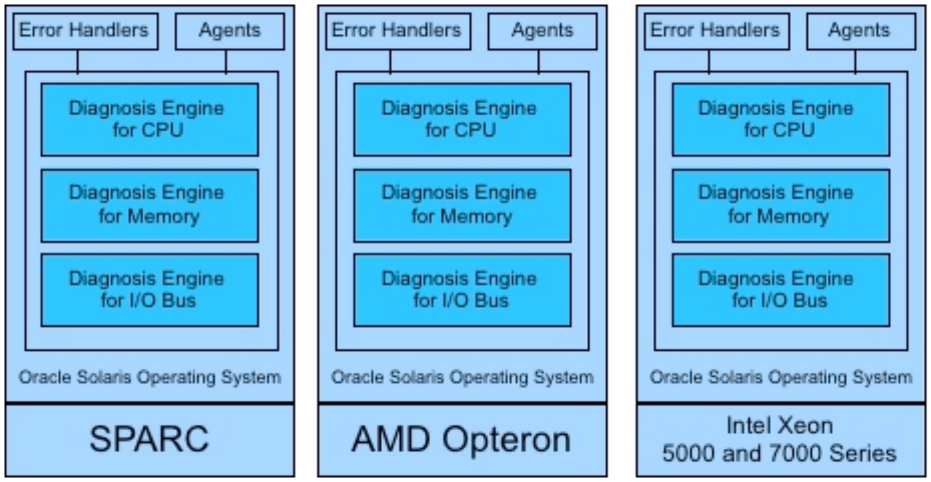

Error Handlers
Agents
Diagnosis Engine for CPU
Diagnosis Engine for Memory
Diagnosis Engine for I/O Bus
Oracle Solaris Operating System
SPARC
AMD Opteron
Intel Xeon 5000 and 7000 Series

Figure 2: Fault Management Architecture defines hardware specific diagnosis engines.

The Oracle Solaris Operating System has implemented diagnostic engines for CPU, memory, and I/O bus nexus components for a variety of hardware platforms incorporating SPARC, AMD Opteron and Intel Xeon 5000 series and 7000 series processors, exploiting the specific hardware reliability, availability and serviceability (RAS) features provided by the underlying system.

For example, the diagnostic engines on Sun SPARC Enterprise systems offer the following capabilities:

- CPU “off lining” takes cores and threads (strands) deemed faulty offline. They are recorded and remain offline on reboot until the faulty processor has been replaced, at which point they are made available again.
- Memory patrol: Memory patrol periodically scans memory for errors, proactively preventing the use of faulty areas of memory before they can cause system or application errors, improving system reliability.
- Memory Extended ECC: The memory Extended ECC function of these servers enables single-bit error correction, enabling processing to continue despite events such as burst read errors that are sometimes caused by memory device failures.

Similarly, the Oracle Solaris Operating System running on Intel’s Xeon 5000 series and 7000 series processor based system provides diagnosis engines that are completely integrated with Intel's Machine Check Architecture (MCA). Intel's MCA recovery enables the system to detect and correct errors in memory and cache that were previously “uncorrectable” through ECC or other means. MCA accomplishes this by first detecting and containing errors before the data is consumed by an application, then works in conjunction with Solaris to determine the best course of action to keep the system and application running. This advanced recovery capability means that systems based on the Intel Nehalem processor running the Oracle Solaris Operating System will be able to recover and remain running in situations where other x86-based systems would not. Hence, an Oracle database deployment on any SPARC or x64 platform running Oracle Solaris 10 will provide correct diagnosis and recovery should a hardware fault occur since Solaris Fault Manager has specialized diagnosis engines for specific processor families.

Protect against memory faults: Memory Page Retirement

Additionally, the Oracle Solaris Operating System provides a platform neutral technology, Memory Page Retirement (MPR), to ensure that both the Oracle Solaris Operating System and user applications continue to operate in the face of main memory faults. The MPR technique allows memory pages suffering from correctable errors and relocatable clean pages suffering from uncorrectable errors to be removed from use in the virtual memory system without interrupting user applications. It also allows relocatable dirty pages associated with uncorrectable errors to be isolated with limited impact on affected user processes, avoiding an outage for the entire system.

Oracle Solaris MPR technology ensures that Oracle database deployments can continue uninterrupted even when the underlying system has memory errors. Consider the scenario of an Oracle database instance deployed on a system that is experiencing memory errors. The diagnosis engine of the Solaris fault manager, which is continuously examining both correctable errors (CEs) and uncorrectable memory errors (UEs), will see a series of correctable errors in a memory location as an indication of uncorrectable memory. If the Oracle database has memory pages that contain CEs then Solaris MPR will retire those pages from memory without interrupting Oracle processes. If the Oracle database references memory pages that have uncorrectable memory errors, then Solaris MPR will retire clean pages containing UEs, again without interrupting Oracle processes. In the unlikely case of the Oracle database having dirty memory pages with UEs, the Oracle processes will come down. However, even in this scenario, if Oracle is configured with Service Management Facility, as explained in the next section, it can restart automatically.

Protect against software faults: Service Management Facility

Service Management Facility is a core part of the Oracle Solaris Predictive Self-Healing technology, which provides automatic recovery from software failures as well as administrative errors. With SMF, system administrators can use simple command line utilities to easily identify, observe, and manage both the services provided by the system and the system itself.

A Solaris service is any long-lived software object with a well-defined state, start and stop, and relationship to other services on the system. In Oracle Solaris 10, each software service has an advertised state. Should a failure occur, the system automatically diagnoses it and locates/pinpoints the source of the failure. Failing services are automatically restarted whenever possible, reducing the need for human intervention. Should manual intervention be required, system administrators can quickly identify the root cause of the service's failure and significantly reduce the times-to-repair and recover from said failure.

Adding the Oracle database and Oracle listeners as a service to the Solaris Service Management Facility (SMF) provides the following advantages:

- If the Oracle database service comes down for any reason including administrator error, software error or uncorrectable hardware error, it will be automatically restarted in dependency order.

- If any service from dependency order fails, the Oracle database service will gracefully come down and a complete explanation of why a service isn't running, as well as individual, persistent log files for each service will be available for debugging purposes.
- The task of managing the Oracle services can be delegated to Oracle administrators; SMF is integrated with Solaris RBAC which ensures that the services can be securely managed by non-root users, including the ability to configure, start, stop, or restart services.

Configure Oracle as a service in Service Management Facility

This section describes the steps required for adding Oracle database as an SMF service so it can be automatically restarted in case of any type of failure.

- Create a service manifest file *oracledatabase.xml* in /var/svc/manifest/application/database directory.
  - You need to create the directory if it doesn't exist and have the appropriate privileges to perform this action. Appendix B has a sample *oracleDatabase.xml* file that you can tailor to your environment.
- Create a methods script file to define how to start and stop this service.
  - Create a shell script *oracledb* in /lib/svc/method directory and change its permission to 555. This script will have methods to start and stop the Oracle database.
- Validate and import the manifest file into the Solaris service repository to create the service in SMF by issuing the following command
  - svccfg validate /var/svc/manifest/application/database/oracledatabase.xml
  - svccfg import /var/svc/manifest/application/database/oracledatabase.xml
- Enable the service using the following svcadm command.
  - svcadm enable svc:/application/database/oracle
- Verify that the service is online
  - svcs -a | grep oracle
- Monitor and troubleshoot the service
  - You can monitor the log file of this service at /var/svc/log/application-database-oracle:default.log.
  - If the service is in maintenance mode (invoke svcs –x command to list failing services), look at the log file to find the cause. Once you resolve the error, clear the maintenance flag on the service by issuing the following command :

```
# svcs clear /application/database/oracle
```

Table 1 shows all the files associated with the Oracle Solaris SMF service

SMF	FILE LOCATION
Service Identifier (FMRI)	Svc:/application/database/oracle
Service Log	/var/svc/log
Service Manifest	/var/svc/manifest/application/database/oracledatabase.xml
Service Start Method	/lib/svc/method/oracledb

Enhance out-of-box accountability

The Oracle Solaris 10 Operating System, arguably the most secure OS on the planet, provides security features previously only found in Sun's military-grade Trusted Solaris OS. User and Process Rights Management work in conjunction with Oracle Solaris Containers to let you securely host thousands of applications and multiple customers on the same system. Solaris Trusted Extensions is a standard part of Oracle Solaris and allows customers who have specific regulatory or information protection requirements to take advantage of labeling features previously only available in highly specialized operating systems or appliances.

Oracle Solaris provides two resources for auditing: BART (Basic Audit Reporting Tool) and BSM (Basic Security Module). Solaris BSM , when enabled, creates an audit trail for specified users. BART is a file tracking tool that operates entirely at the file system level. BART gives you the ability to quickly, easily, and reliably gather information about the components of the software stack that is installed on deployed systems. Refer to reference section for a list of collateral that describes Oracle Solaris auditing in greater detail.

The following section explains how an Oracle Database installation can be made more secure with enhanced accountability by exploiting the user rights management feature of Oracle Solaris 10.

Track activities of individual DBAs

User rights management reduces security risks by providing privileged users only the capabilities needed to run a select number of commands consistent with their needs rather than granting full super-user access to the system. This increases security by reducing the chances of administrative errors or accidental/malicious use of systems. User rights management, based on Oracle Solaris Role-Based Access Control (RBAC) capabilities, is centrally managed for reduced administration cost and increased flexibility for rapidly changing business requirements. Effective security reduces downtime, raises quality of service, and keeps costs low.

In RBAC, roles are assigned to users. When a user assumes a role, the capabilities of the role are available. Roles get their capabilities from rights profiles. Rights profiles can contain authorizations, privileged commands, and other supplementary rights profiles. Privileged commands are commands that execute with security attributes.

Default installations of the Oracle database can be made more secure by exploiting the user rights management feature of Oracle Solaris 10 security. In a typical Oracle deployment, all Oracle DBAs login as the UNIX user *oracle*. Hence, it is not possible to track the DBA-related activities of an individual user; only the combined activities of all DBAs are tracked by the Operating system and the database server. User rights management enables you to create an oracle role and assign it to users with DBA responsibilities. In this scenario, the users will login to the database server system with their regular UNIX logins and assume the oracle role when they need to do any Oracle DBA-related tasks. This approach ensures that multiple Oracle administrators do not share a single login. They login in as individual users and are accountable for their individual actions; yet they have the flexibility to perform all the functions of an Oracle administrator by assuming the oracle role. Complete accountability for individual users can be enforced by enabling auditing of the oracle role; which in turn will provide a detailed description all Oracle DBA-related activities for each individual UNIX user.

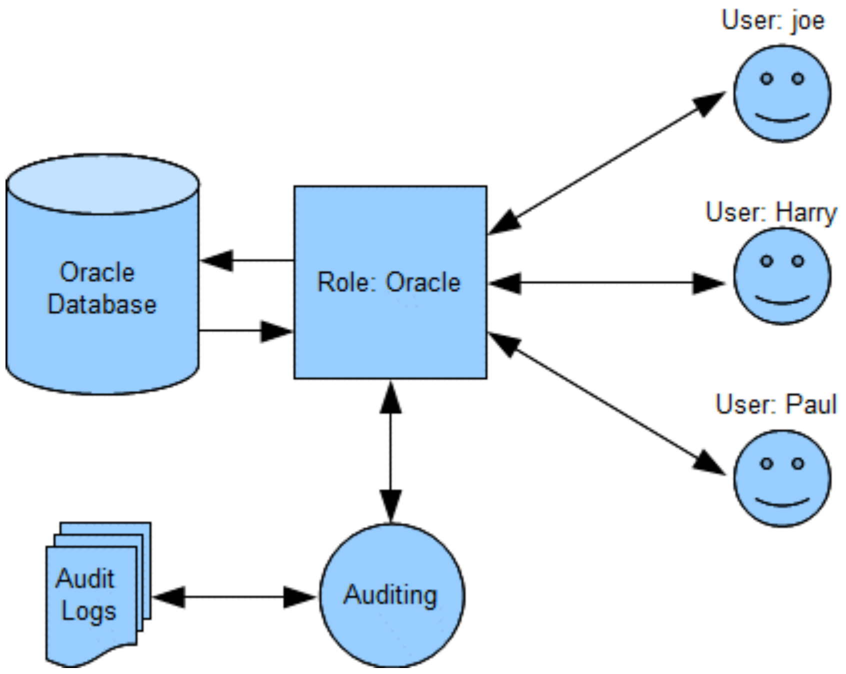


Figure 4 : Using *Oracle* role enhances security and accountability

If additional security is required, the privileges of the UNIX user can be adjusted such that individual UNIX users cannot view Oracle processes. Similarly, the privileges of the *Oracle* role can be adjusted such that they can view only the Oracle processes.

Create an Oracle role

The pre-requisite for creating an *Oracle* role is to define a rights profile for the *Oracle* role, which will define the capabilities of this role. An Oracle administrator would need access to all commands under the $ORACLE_HOME/bin directory. He would need access to commands found in the /usr/bin and /usr/sbin directories. An Oracle database administrator would additionally need authorization to manage Oracle database and listener SMF services, if they exist.

Figure 5 illustrates the relationship between an *Oracle* role and an Oracle database administration rights profile. The *Oracle* role would have the permissions for all the executables under $ORACLE_HOME as well as executables under /usr/bin and usr/sbin. Additionally, it will have the authority to manage Oracle SMF services.

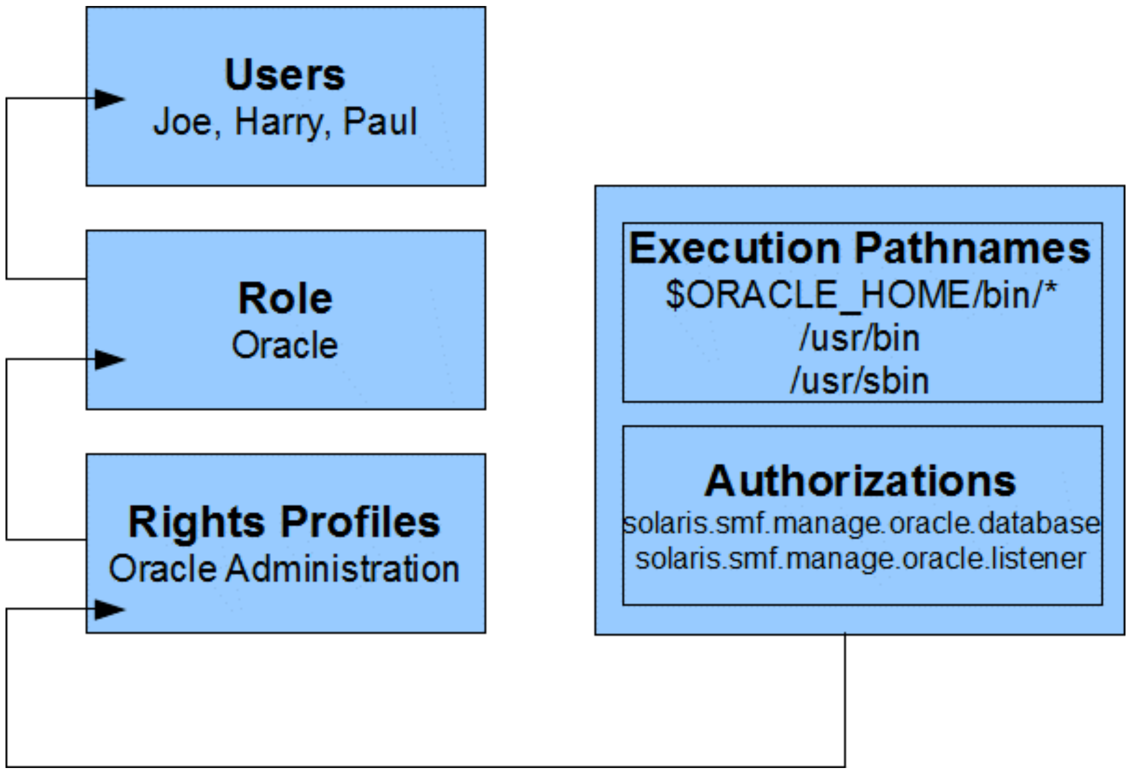


Figure 5 Oracle role has permissions to access Oracle database deployment and control Oracle SMF services

Creating an Oracle role is a two step process, the first step is to create an Oracle database administration rights profile and the second step is to create the role and assign it the Oracle administration rights profile.

Step 1: Create Oracle Administration rights profile

- Start the Solaris Management Console (smc) as Superuser :

- %/usr/sadm/bin/smc &
- Click on the 'This Computer' icon in the Navigation pane
- Click on System Configuration->Users->Rights
- Click Action->Add Rights. The Add Rights wizard opens.
- Create the Oracle Administration rights profile with the Add Rights wizard by entering the following information in the wizard:

TABLE 2. CREATE ORACLE ADMINSTRATOR RIGHTS PROFILE

TAB	FIELD	VALUE
General	Name	Oracle Administrator
	Description	Rights profile for Oracle DBAs
Commands	Add Directory	Click Add Directory, type $ORACLE_HOME/bin in the dialog box and click OK
	Commands Denied/ Commands Permitted	Move $ORACE_HOME/bin to the Commands permitted column
	Set Security Attributes	Select , click Set Security Attributes and set Effective UID=oracle
Authorizations	Authorizations Excluded/ Authorizations Included	Select Oracle SMF authorization, if configured. Refer to Appendix B for details
Supplementary Rights	Rights Excluded/ Rights Included	No Supplementary rights profiles.

Step 2 Create an Oracle role and associate Oracle administration rights profile with this role

- Start the Solaris Management Console (smc) as superuser
  - /usr/sadm/bin/smc &
- Click on the 'This Computer' icon in the navigation pane
- Click on System Configuration->Users->Administrative Roles
- Click Action->Add Administrative Role. The Add Administrative Role wizard opens.
- Create the Oracle role with the Administrative Role wizard by following these steps
  - Set the role name to Oracle, full role name to Oracle DBA role Description to Role for Oracle DBA. Click Next

- Set and confirm the role password. Click Next.
- Select the Oracle Administrator rights profile from the Available Rights column and add it to Granted Rights column. Click Next
- Add UNIX logins of all Oracle DBAs to the list of users who can assume this role.

Simplify Deployment

Prior to Oracle Solaris 10, installing the Oracle database on the Oracle Solaris Operating System required changes to the /etc/system file. Every reconfiguration required a reboot for the changes to take effect. The System V IPC implementation in Oracle Solaris 10 no longer needs changes to the /etc/system file. Instead the new resource control facility is used, which allows changes to become effective immediately, without a system reboot. Furthermore the default settings of the System V IPC parameters have been set to reasonable defaults enabling Oracle database instances to run out-of-the-box without requiring special parameters to be set.

Oracle deployments on Oracle Solaris 10 work out of the box, with no additional system configuration, if the System Global Area (SGA) uses less than 25% of the system's total memory. If the deployment plans to use more than 25% of the systems memory, then the shared memory resource parameter can be dynamically set to the required value using the resource control facility.

Create a project for Oracle Database Installation

By default, the Oracle Solaris OS provides all workloads running on the system equal access to all system resources. Oracle Solaris uses projects facility to identify a workload. Every user in the Oracle Solaris OS system is assigned a default project. Users cannot login to the system unless they are associated with a project. Oracle Solaris 10 provides a resource control facility to set resource limits for projects. The resource control facility provides project wide resource controls to define Oracle Solaris kernel's inter process communication (IPC) facilities. These resource controls replace the /etc/system tunables and can be set dynamically.

In order to set the shared memory to more than 25% of the sytem, you need to create a project, assign it to the *oracle* user and set the max-shm-memory resource control to the desired value.

The following command creates a project named oracle, assigns it to user *oracle* and group *dba* and sets max-shm-memory resource to 10 gigabytes :

```
$  projadd -U oracle -G dba -K 'project.max-shm-memory= (privileged, 10G, deny)' oracle
```

You can optionally set the project id with the -p option and comment with the -c option

```
$ projmod -p 100 -c "Project for Oracle database deployment' oracle
```

If an Oracle database is deployed on a non-global zone on Oracle Solaris 10 8/07 update, the SystemV IPC resource controls are added zone -wide. Hence, these resources can be set during the process of creating the zone or altered on a zone wide basis; there is no need to create a project to set System V IPC variables on a non-global zone.

Consolidate multiple Oracle Database instances

Oracle Solaris Containers

Oracle Solaris Containers, Oracle's operating system level virtualization technology, provide complete, isolated, and secure run time environments for applications. This technology allows application components to be isolated from each other using flexible, software-defined boundaries. Oracle Solaris Containers are designed to provide fine-grained control over resources that the applications use, allowing multiple applications to operate on a single Oracle Solaris 10 OS instance while maintaining specified service levels (Figure 6).

Unlike other commercially-available virtualization solutions, Solaris Containers are included with the Oracle Solaris Operating System at no additional cost. Further, both Oracle Database 10g and 11g have been certified on Oracle Solaris Containers and are fully supported by Oracle. Oracle Solaris Containers can be used to deploy virtualized application environments, both on x86 and SPARC platforms, at significant cost savings and much lower risk compared to alternative solutions.

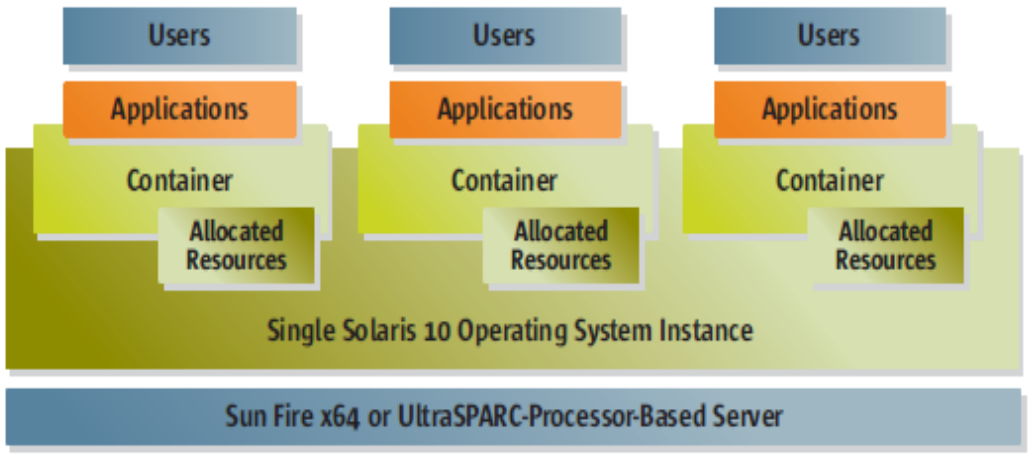


Figure 6 : Solaris Containers enable multiple applications to operate while maintaining specified service levels

Oracle Solaris Containers use Oracle Solaris Resource Manager (SRM) features along with Oracle Solaris Zones software partitioning technology to deliver a virtualized environment that can have fixed resource boundaries for application workloads. For more detailed information about these technologies, see the references section.

Unlike virtual machines, Oracle Solaris Containers provide operating system level virtualization by giving the functionality and isolation of multiple OS instances without requiring multiple physical machines or hypervisor-based virtual machines. Isolation between Oracle Solaris Containers is accomplished by restricting the scope of system calls, rather than the CPU-intensive task of emulating hardware architectures and instruction sets in software. This makes it possible to create hundreds, even thousands, of Oracle Solaris Containers on a single system. Because of this negligible overhead, and unlike physical partitioning or hypervisor-based virtual machines, Oracle Solaris Containers can be created in large numbers. You can create up to 8191 Oracle Solaris Containers in a single system. Computing resources—CPUs, physical memory, network bandwidth, and more—can be dedicated to a single application one moment and then shared with others in an instant, all without moving applications or rebooting the system, physical domain, or logical domain where the Oracle Solaris Container resides.

Manage license

By deploying Oracle databases in Oracle Solaris 10 Containers customers can license only the CPUs or cores located in capped Oracle Solaris 10 Containers, since they are recognized as licensable entities, known as hard partitions.

Oracle licensing policy defines hard partitioning as “a physical subset of a server that acts like a self-contained server” (for more details see reference section). The following example (Figure 7) illustrates how an eight processor system can be partitioned into a three processor sub-system using Oracle Solaris Containers technology in the Oracle Solaris 10 OS.

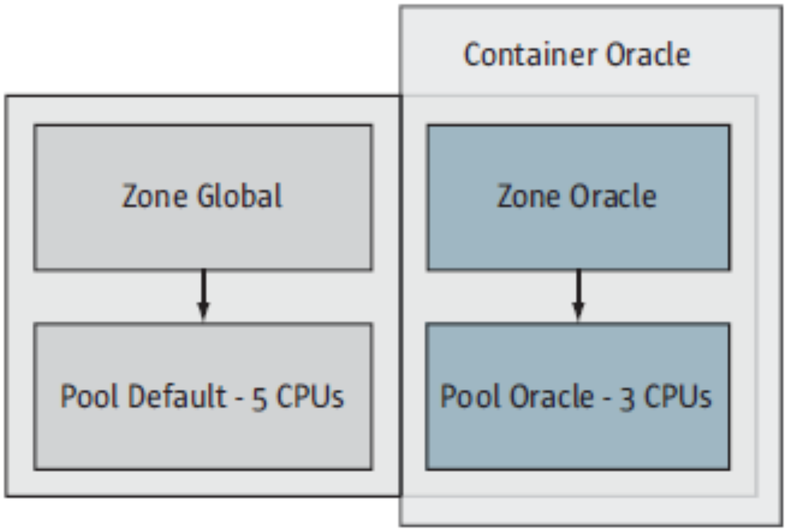


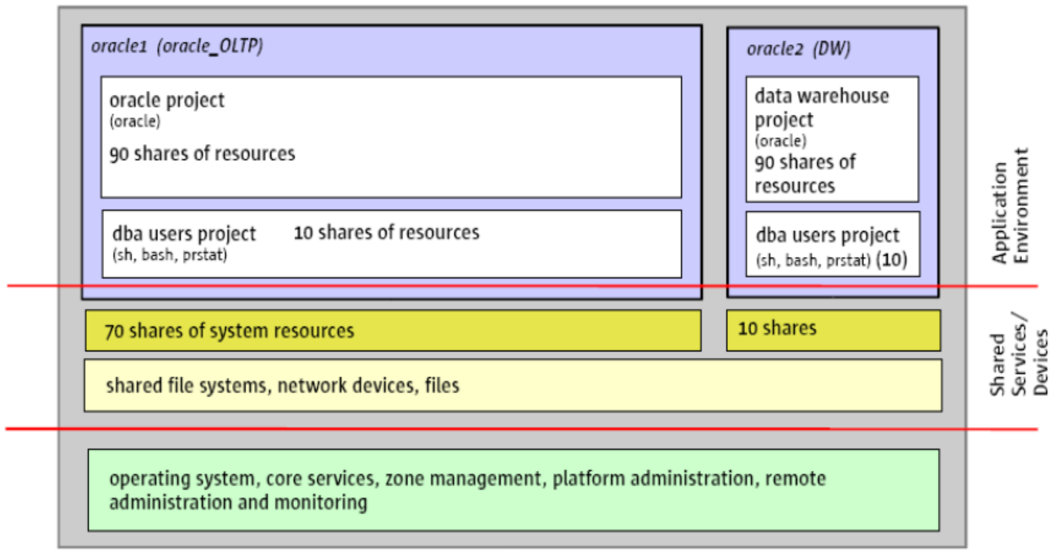
Figure 7: Figure 7 illustrates creation of an Oracle Solaris Container for database deployment.

To create an Oracle Solaris 10 Container that fits the licensing requirements set by Oracle, the Oracle Solaris system administrator needs to create a resource pool with the desired number of CPUs or cores and bind a zone to this resource pool. Alternatively, the administrator may set up a container to use a dynamic pool with a specified CPU maximum limit. The license is driven by the maximum number of CPUs or cores in this pool.

Maintain Quality of Service (QoS)

Customers can consolidate multiple Oracle database instances into separate containers on the same system to enable competing applications, such as online transaction processing (OLTP) and data warehousing applications, to run with predefined resource allocation, changing as business needs change. For example, in Figure 8, the OLTP container is allocated 70 shares of the CPU resources, while the data warehouse is allocated 10 shares, resulting in a 7:1 ratio of CPU resources allocated to each container. Shares allow unused cycles to be used by other applications, or the allocation can be changed dynamically during peak times to provide more CPU resources to either container. In addition, the resources for each container are further subdivided, allocating a portion of resources to each project within the container. This helps ensure that each project always has the resources it requires to function predictably. Database administrators can have complete control over their isolated environment. In addition, a separate project can be created specifically for database administrators in order to limit their access to resources, which can keep other processes from consuming critical CPU resources and negatively affecting the performance of the database.

Figure 8 : This figure illustrates consolidation of multiple databases and restriction on database



administrator access.

The reference section has a list of collateral that provides step-by-step instructions to set up Oracle Solaris Container for an Oracle database deployment.

Enhance Observability

With the advent of multi-tier architectures today's applications have become very complex. While individual levels of the application tier may have excellent tools for observability and debugging, there are no tools to observe and optimize the entire application stack. This problem becomes even more complicated for observing applications in production which are likely sensitive to performance impacts. Also, it is not always easy to stop and start these applications to enable debug flags. Adding debug versions of applications into production may not be permitted. Even if permitted, bringing debug versions into production involves expensive and time consuming QA cycles. All of these issues complicate the problem of observation.

DTrace, a Dynamic Tracing framework, was developed to address this very problem. It can be used to observe any or all tiers of the application stack, it is truly dynamic and does not require application code changes or even an application restart. One can observe fully optimized applications using DTrace. The overhead of observation is low and there is no overhead when observation is turned off. Instrumentation can be turned on and off dynamically thus only collecting information when it is needed. DTrace is safe and turns itself off when observation overhead affects system performance.

DTrace can be used to observe applications developed in, C, C++, Java, JavaScript, Ruby, PHP, Perl, Python among other programing and scripting languages. Other system layers, like I/O, networking, application and kernel locks, CPU counters etc, can also be observed using DTrace.

DTrace scripts are used to enable and program points of instrumentation. D-script format does not change based on the application tier being observed and a single script can be used to observe multiple tiers at the same time.

DTrace can be used to look at Oracle database processes in isolation or concurrently with any other processes running on the system and can be an invaluable tool for identifying performance bottlenecks and many other real world issues. Oracle administrators can use DTrace probes, in conjunction with Oracle's AWR report, to quickly understand and resolve performance issues on the Oracle Solaris 10 platform.

Over two decades of engineering collaboration and innovations

For over 20 years, Oracle Solaris internals have been improved to enhance the scalability and performance of Oracle database deployments. Some of these changes include:

- The first 64bit version of Oracle (Oracle 8i) was available on Oracle Solaris as Solaris was the first commercially available UNIX to offer a 64-bit version. This enabled Oracle database to break the 4 GB memory barrier and utilize the 64GB of memory available on the Sun Enterprise 10000 range of servers.
- Sun collaborated with Oracle to enable Memory Placement Optimizations (MPO). Memory Placement Optimization enables processors to have an affinity for the closest memory on Non-uniform Memory Access (NUMA) systems—the types of multisocket, large memory systems that are powered by SPARC processors and Oracle Solaris. These optimizations were made default on

Oracle 10g running on Sun NUMA based machines. These optimizations help increase the locality of reference for the SGA and Process Global Area (PGA, a dedicated memory cache). The performance improvements can be worthwhile depending on the server and the application. Oracle Solaris MPO innovations are key to scaling on servers with significant NUMA behavior.

- The Oracle Solaris 's Intimate Shared Memory (ISM) feature provides permanently locked shared memory and shares translation tables involved in the virtual to physical address translation for shared memory pages, as opposed to just sharing the actual physical memory pages. ISM was a critical technology which enabled Oracle to efficiently scale on large SMP systems as well as smaller machines.
- Oracle Solaris provides large Page support and automatically uses large pages for Oracle Database instruction pages and for the database SGA on all SPARC systems, and for the database PGA.
- Dynamic ISM (DISM) enabled Oracle support for the dynamic SGA feature introduced in Oracle9i. This allowed a DBA to dynamically increase or decrease the size of the SGA (up to a limit defined by sga_max_size) without needing to restart the Oracle instance. Using the Solaris Reconfiguration Coordination Manager (RCM), it is also possible to write a script that allows Oracle to be alerted when new CPUs/memory are to be removed from the domain, so that the SGA can be dynamically scaled back to allow the board to be removed without shutting down the database. DISM is the default option with Oracle Database 11gR2.
- For many years Oracle Solaris Cluster software has been evolving to complement and integrate with Oracle Database solutions including Oracle Real Application Clusters (RAC). The result is thoroughly tested, tightly integrated, end-to-end solutions that extend the advantages of Oracle Solaris and Sun SPARC Enterprise systems into multiserver, high-availability environments.

And More …..

Oracle Solaris ZFS

Oracle Solaris Zettabyte File System (ZFS) technology offers a dramatic advancement in data management with a virtual storage pool design, integrated volume manager, and data services that provide an innovative approach to data integrity.

Oracle Solaris ZFS software enables more efficient and optimized use of storage devices, while dramatically increasing reliability and scalability. Physical storage can be dynamically added or removed from storage pools without interrupting services, providing new levels of flexibility, availability, and performance.

Oracle Solaris ZFS protects all data by 256-bit check sums, resulting in 99.99999999999999999-percent error detection and correction. Oracle Solaris ZFS constantly reads and checks data to help ensure it is correct, and if it detects an error in a storage pool with redundancy (protected with mirroring, Oracle Solaris ZFS RAIDZ, or Oracle Solaris ZFS RAIDZ2), Oracle Solaris ZFS automatically repairs the corrupt data. This contributes to continuous availability by helping to protect against costly and time-consuming data loss due to hardware or software failure, and by reducing the chance of administrator error when performing file system-related tasks.

Oracle Solaris ZFS software optimizes file system reliability by maintaining data redundancy on commodity hardware. It seamlessly and transparently supports new hybrid disk storage pools that include Flash technology for superior application performance

Since Oracle Database 10g, Oracle added Automatic Storage Management (ASM) provides the database administrator with a simple storage management interface, which is consistent across all server and storage platforms. The Automatic Storage Management feature in Oracle Database 11g Release 2 extends ASM functionality to manage ALL data: Oracle database files, Oracle Cluster ware files and non-structured general-purpose data such as binaries, externals files and text files.

Recommended best practice is to use Oracle ASM as an integrated volume manager for Oracle database deployment and Oracle Solaris ZFS for other files. However, if you choose to use Oracle Solaris ZFS for Oracle database deployment, which is certified for Oracle database 10g and 11g , refer to the resources section for a list of collateral on best practices for using Oracle Solaris ZFS with Oracle Database.

Oracle Solaris Clusters

Oracle Solaris Cluster supports Oracle Database and Real Application Clusters, and tightly integrates with Oracle Cluster ware. It also provides flexibility for the cluster infrastructure by supporting a wide range of networking and storage options such as InfiniBand, ASM, NAS, QFS, and hardware in thoroughly tested configurations.

Solaris Cluster also provides Oracle Solaris Containers cluster, which provides virtual clusters and support the consolidation of multiple cluster applications into a single cluster. Multiple Oracle database versions can be consolidated into one physical cluster for highly reliable service at a much lower cost while still benefiting from Oracle Solaris Container's advantages of security isolation, resource management, and fault isolation. Oracle Solaris Containers is supported with Oracle RAC 10g R2 and 11g R1 with Oracle Solaris Cluster on Sun SPARC Enterprise servers. Oracle Solaris Container cluster is the most complete Oracle Solaris-based ahigh-availability solution that leverages software licensing models based on CPU utilization. In some situations, the costs of the applications and databases that co-exist in the same cluster of hardware can be reduced by using Oracle Solaris Containers clusters.

Oracle Solaris Cryptographic Framework

Oracle Advanced Security option in Oracle Database 10g and later provides Transparent Data Encryption (TDE) of data stored in the database and network encryption for data traveling across the network. Since Oracle Database 11gR1, transparent data encryption (TDE) supports Hardware Security Module (HSM) to provide secure key management features for the master encryption key.

Oracle Solaris Cryptographic Framework facilitates encryption and decryption of data stored in an Oracle Database. It supports Master key management using Solaris PKCS#11 Soft Token or third-party Hardware Security Modules (HSM) as well as Master key backup and recovery for an Oracle database deployment.

Using Solaris Cryptographic Framework and its PKCS#11 interfaces, Oracle TDE can take advantage of configuring Oracle Wallet with Solaris PKCS#11 Soft Token or a Hardware Security Module (HSM) to support enabling specific TDE features :

- Centralized key store for securing the master key used to encrypt and decrypt the keys performing actual data encryption.
- Encryption/decryption of tablespace and column encryption keys.
- Encryption/decryption support for Oracle Data Pump utility.
- Encryption/decryption of backup/restore using Oracle Recovery Manager (RMAN)
- Network encryption between Oracle client and server applications supporting SSL/TLS protocols.
- NIST specified cryptographic algorithms.

Conclusion

The Oracle database has a proven track record of scaling well both vertically as well as horizontally on the Oracle Solaris 10 platform. Additionally, an Oracle database deployment on Oracle Solaris 10 platform can easily take advantage of the unique features found in Oracle Solaris 10 Operating System to enhance the manageability, scalability, availability and security of both single and multiple Oracle database instances – all across multiple platform and processor architectures. The reference section has a list of collateral that provides detailed information on how an Oracle database deployed on SPARC servers can further take advantage of Oracle Solaris 10 platform.

References

The following table contains links to useful information related to this paper

RESOURCES	
Oracle Solaris Operating System	http://www.oracle.com/us/products/servers-storage/solaris/index.html
Oracle Solaris : Virtualization	http://www.oracle.com/us/technologies/virtualization/index.htm
Oracle Solaris : Reliability	http://www.oracle.com/us/products/servers-storage/solaris/reliability-066071.html
Oracle Solaris Security	http://www.oracle.com/security/index.html
Oracle Solaris ZFS	http://www.sun.com/bigadmin/topics/zfs/
Configuring Sun Storage 7000 Unified Storage Systems for Oracle Databases	http://www.oracle.com/technetwork/articles/systems-hardware-architecture/ss700-config-oracle-db-163902.pdf
Oracle Advanced Security Transparent Data Encryption using Sun Crypto Accelerator 6000 PCIe Card	http://www.oracle.com/technetwork/articles/systems-hardware-architecture/adv-encryption-sca6000-163879.pdf
How to create an Oracle Solaris SMF Manifest	http://developers.sun.com/solaris/docs/smf-manifest-053110.pdf

Appendix A: Benchmarks published on Oracle11g on Oracle Solaris10

List of industry standard benchmarks published on Oracle 11g on Solaris 10

Oracle Database 11g deployment on Oracle Solaris 10 deployment continues to set world records in performance and affordability. This includes:

- SPECjAppServer2004 JOPS@Standard: Oracle Solaris, Oracle WebLogic 10.3.3 Application Server, and Oracle Database 11g Enterprise Edition power five Sun SPARC Enterprise T5440 servers, six Sun Storage F5100 Flash Arrays, and one Sun SPARC Enterprise M9000 server to a world record result of 28,648.74 SPECjAppServer2004 JOPS@Standard on the SPECjAppServer2004 benchmark.
- TPC-C: SPARC Enterprise T5440 on Oracle Solaris Server Cluster is the world's fastest OLTP system (7,646,486.7 tpmC), and achieved with the best $/tpmC ($2.81/tpmC) out of all top 10 performers. TPC-C demonstrates Oracle Solaris and SPARC combine to cost-effectively deliver heavily multithreaded and I/O capabilities on OLTP workloads.
- TPC-H@3000GB: Sun SPARC Enterprise M9000 server and Oracle Solaris delivered a single-system TPC-H 3000GB world record performance and price performance results.7 The Sun SPARC Enterprise M9000 server, running Oracle Database 11g Release 2 proves the power of the Oracle solution.
- Oracle Business Intelligence Enterprise Edition: SPARC Enterprise 5440 systems, Oracle 11g Database, Oracle Solaris, Oracle Solaris Containers, and Oracle Solaris ZFS set world records in supporting 50,000, 28,000, and 10,000 concurrent users.
- PeopleSoft Payroll (North America) 9.0 benchmark. Oracle Solaris running on the Sun SPARC Enterprise M4000 server with four 2.53GHz SPARC64 VII processors and the Sun Storage F5100 flash array on the PeopleSoft Payroll (NA) 9.0 benchmark with Oracle 11g Database.9 The Sun SPARC Enterprise M4000 server combined with Oracle FlashFire technology demonstrated a speedup of 81 percent going from 1 to 8 streams on the PeopleSoft Payroll (NA) 9.0 benchmark using the Oracle 11g Database.
- SAP SD 2-tier: Oracle delivers world-record leadership, including:
  - SPARC Enterprise T5440 system and Oracle Solaris set 4-CPU world record—4,720 SD users—for SAP ERP 6.0 application Enhancement Package 4 (Unicode).
  - Oracle Solaris and the Sun SPARC Enterprise M9000 server with 2.88 GHz SPARC64 VII processors achieved 32,000 users.

Appendix B: Manifest file to add Oracle database as an SMF service

The reference section has a list of collateral that provides step by step instructions to create the manifest. The following section explains the different sections of the sample Oracle database manifest file.

- Service name:
  - Service name section defines the name of the service as *oracle*. To change the service name, update the following section of the manifest file:
  - `<service_bundle type='manifest' name='oracle'>`
- Dependency section:
  - This section identifies a group of FMRIs upon which the service is dependent. In the sample Oracle SMF manifest file the dependency is set to network, local file system and name-services milestone. If listener and ASM are added to SMF, they should be added as dependency for Oracle database startup.
- Exec method:
  - Defines the methods to be executed for starting, stopping or restarting the service. The sample Oracle manifest file defines these methods in a shell script oracledb. This shell script should be placed in /var/svc/lib directory.
- Method Context:
  - This element combines credential and resource management attributes for execution methods. Edit this section to change the project or resource pool under which Oracle database is deployed.
- Instance name:
  - Defines the instance name. The sample manifest file defines only a single instance. If you want SMF to control multiple instances, this element needs to be replicated for as many instances as required.
- Method Credential
  - Edit this element to update user, group, supp_groups, privileges and limit_privileges attributes
- Method Environment
  - This element defines all the environment variables that are required by methods. This section needs to be edited to suit your environment.
- SMF and RBAC
  - By default, only root user can manage SMF services. The sample manifest allows any user or role that has solaris.smf.manage.oracle authorization to be able to use and configure this service.

- You can configure the oracle user to have authorization to manage this service as under:

Edit the /etc/security/auth_attr file and add the following lines:

- solaris.smf.manage.oracle:::Manage Oracle service states::

Next assign *solaris.smf.manage.oracle* authorization to *oracle* user.

- usermod -A solaris.smf.manage.oracle oracle

The sample Oracle database manifest file for single instance database is as under:

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>

<service_bundle type='manifest' name='oracle'>

     <service name='application/database/oracle' type='service' version='0'>

     <!--
          Wait for network interfaces to be initialized
     -->

     <dependency
          name='network'
          grouping='require_all'
          restart_on='none'
          type='service'>
          <service_fmri value='svc:/milestone/network:default'/>
     </dependency>

     <!--
          wait for all local filesystems to be mounted
     -->

     <dependency
          name='filesystem-local'
          grouping='require_all'
          restart_on='none'
          type='service'>
          <service_fmri value='svc:/system/filesystem/local:default'/>
     </dependency>

     <instance name='default' enabled='false'>
          <method_context
               project=':default'
               resource_pool=':default'
               working_directory=':default'>

               <method_credential
                    group='dba'
                    limit_privileges=':default'
                    privileges=':default'
                    supp_groups=':default'
                    user='oracle'/>
```

```
                <method_environment>
                     <envvar name='ORACLE_HOME'
                          value='/oracle/app/oracle/product/12.1.0/testdb'/>
                     <envvar name='ORACLE_SID'
                          value='testdb'/>
                </method_environment>
           </method_context>

           <exec_method
                name='start'
                type='method'
                exec='/lib/svc/method/oracledb start'
                timeout_seconds='300'>
           </exec_method>

           <exec_method
                name='stop'
                type='method'
                exec='/lib/svc/method/oracledb stop'
                timeout_seconds='300'>
           </exec_method>

           <exec_method
                name='refresh'
                type='method'
                exec='/lib/svc/method/oracledb start'
                timeout_seconds='60'>
           </exec_method>

      </instance>

      <stability value='Evolving'/>

      </service>

</service_bundle>
```

ORACLE®

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

Deploying Oracle Database on the Oracle Solaris Platform
September 2010 Updated January 2014
Author: Ritu Kamboj

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

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.
This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

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

AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. UNIX is a registered trademark licensed through X/Open Company, Ltd. 0110

SOFTWARE. HARDWARE. COMPLETE.