An Oracle White Paper
April 2010

# Oracle's Sun Systems for Enterprise 2.0

Oracle's Sun Systems for Enterprise 2.0 ....1
Introduction ....1
Architecture ....2
System Components ....3
Software Stack ....8
Performance Characterization ....10
Test Environment ....10
Workload and Test Description ....12
Data Collection ....13
Test Results ....13
Results Summary ....16
Summary ....17
Appendix ....18
Preliminary Setup ....18
Load Balancer Configuration ....18
Database Server Installation ....20
Application/Oracle GlassFish Server ....25
Oracle WebLogic Server 10*g* Installation and Setup ....27
Tuning ....39
About the Authors ....42
References ....42

# Oracle's Sun Systems for Enterprise 2.0

Improved business collaboration and easier information sharing within an enterprise can provide a strong competitive advantage. Enterprises today are seeking new communication technologies, such as wikis, blogs, and forums, to improve communication and collaboration and increase employee productivity. Oracle's Sun Systems for Enterprise 2.0 solution provides an integrated platform—combining highly scalable servers, next-generation storage, and networking technologies with best-of-breed software and services from Oracle—to deploy a highly reliable Enterprise 2.0 infrastructure that delivers simplicity, speed, and savings.

The following topics are addressed in this paper:


- "Introduction" on page 1 introduces the challenges and trends facing the enterprise today.
- "Architecture" on page 2 describes the architecture used by the Sun Systems for Enterprise 2.0 with Oracle solution, including the hardware and software components.
- "Performance Characterization" on page 10 includes performance information that demonstrates scalability and can be used to help determine proper sizing of systems.
- "Appendix" on page 18 includes the procedures used to set up and configure the test environment.


## Introduction

Information has expanded exponentially in recent years because of the new ways people are capturing, storing, and sharing information. This trend is apparent in social networks, and it is no different in the enterprise. Enterprise users seek to personalize the corporate intranet to their needs, share information with their peers, and easily update content without lengthy and cumbersome IT processes. In addition, employees want ways to be more productive, including single sign-on to the intranet and the various applications they use, an easy way to get information from one application and enter it in another, and incorporation of productivity tools such as calendars, time gadgets, and maps.

The emergence of social networking—with new ways to communicate such as instant messaging, wikis, and blogging—is permeating the corporate world, and early adoption has shown very promising results. Users now expect a faster and more efficient way to communicate with peers, groups, and across the enterprise. These new user requirements and trends are clear signs that there must be changes to the existing corporate intranet. Welcome to the world of Enterprise 2.0.

Web portals allow people to execute applications, process documents and other type of media, self-publish content through blogs and wikis, and interact with peers through email, instant messaging, and other communication tools. Users access the portal through a common Web browser that is customized to their needs based upon their assigned roles, helping users find relevant information and applications more easily.

Planning an Enterprise 2.0 deployment takes time and technical knowledge on many disciplines. Selecting portal server software alone is not enough. Choosing hardware components such as servers

and storage that can provide a solid foundation, and deploying the software infrastructure, including database, Web, and application server software, is equally important. This paper provides an answer for an Enterprise 2.0 architecture—a tested and proven solution that uses Oracle® systems, Oracle GlassFish Server, Oracle WebLogic Server, and the Oracle Database 11*g* with Real Application Cluster (RAC) Technology.

## Architecture

The Sun Systems for Enterprise 2.0 with Oracle solution combines key components in computing, storage, networking, and software into a comprehensive solution. This architecture, which encompasses all needed components from the underlying hardware to the portal software, provides a flexible and adaptable implementation of highly available and scalable enterprise collaboration initiatives, delivered through a Web-based portal interface.

Major components of the Sun Systems for Enterprise 2.0 with Oracle solution include Oracle's Sun Blade 6000 Modular System, with Sun Blade T6340 server modules and the Sun Blade 6000 Virtualized Multi-Fabric 10 Gigabit Ethernet Network ExpressModule (NEM) from Oracle; Oracle's Sun Storage 7000 Unified Storage Systems; the Oracle Solaris operating system; Oracle GlassFish Web Space Server; Oracle WebLogic Server; Oracle Database 11*g* with RAC; and Zeus Extensible Traffic Manager (ZXTM) load balancing software. A logical view of the system architecture, shown in Figure 1, depicts the relationships between the various hardware and software components.

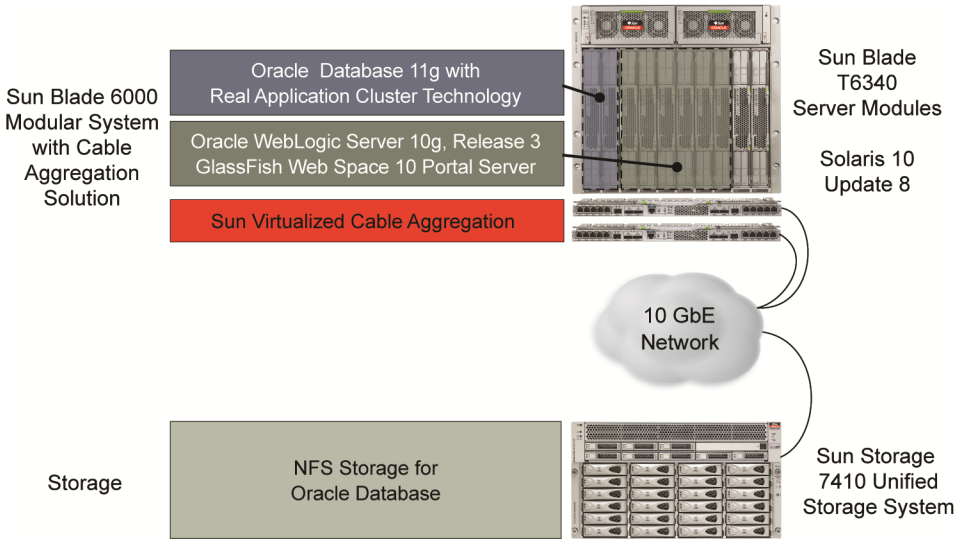



**Figure 1. Architectural logical diagram.**

## System Components

Table 1 lists the specific hardware and software components included in the Sun Systems for Enterprise 2.0 with Oracle solution.

**TABLE 1. HARDWARE AND SOFTWARE COMPONENTS**

| | COMPONENT | PRODUCT |
|---|---|---|
| Hardware | Server hardware | Sun Blade 6000 Modular system<br>• Sun Blade 6000 chassis<br>• Sun Blade T6340 server modules<br>• Sun Blade 6000 Virtualized Multi-Fabric 10 GbE NEM |
| | Storage | Sun Storage 7410 System |
| | Rack | Sun Rack II |
| Software | Operating system | Oracle Solaris SPARC® OS 10, update 8 |
| | Load balancer | Zeus Extensible Traffic Manager |
| | Portal | Oracle GlassFish Server |
| | Web/Application | Oracle WebLogic Server 10*g* Release 3 |
| | Database | Oracle Database 11*g* with RAC |

### Hardware Components

The primary hardware components include the Sun Blade 6000 Modular System, the Sun Storage 7410 System, and the Sun Rack II.

#### The Sun Blade 6000 Modular System

The Sun Blade 6000 Modular System provides the hardware foundation for this Enterprise 2.0 solution. This modular, scalable system with a choice of blade server module options provides a flexible platform suitable for enterprise deployments. The system consists of the following components:

- *Sun Blade 6000 chassis* fits in a compact 10U form factor while supporting up to 10 full-featured blade server modules (see Figure 2). These server modules can be separately provisioned to run different operating systems such as Solaris OS, Linux or Windows. A key factor in selecting this chassis for this solution is its standout I/O throughput. This chassis offers full throughput to every server module for all configurations, industry-standard ExpressModule slots, as well as added-value Network ExpressModule (NEM) boards. In addition, the chassis' modular architecture and hot-swappable components help ensure optimal availability, serviceability, and ease of deployment.

**Figure 2. Sun Blade 6000 chassis.**

- The *Sun Blade T6340 Server Module*, the highest density chip multithreading (CMT) server available in a blade form factor, delivers massive throughput and built-in, no-cost encryption and virtualization (Oracle VM Server for SPARC). The Sun Blade T6340 Server Module, equipped with two 1.4 GHz UltraSPARC® T2 Plus processors, is one of the highest performing and most efficient platforms available. Published results on industry-standard benchmarks, including the fastest dual-processor result on an enterprise Java™ workload, demonstrate the ability of the Sun Blade T6340 server module to handle enterprise-scale Java workloads.[1]

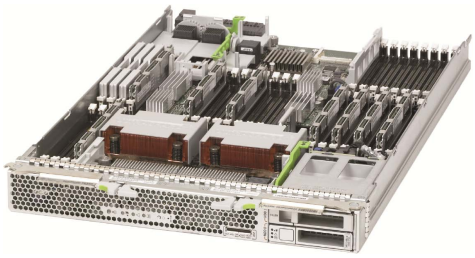


**Figure 3. Sun Blade T6340 server module.**

- The *Sun Blade 6000 Virtualized Multi-Fabric 10 GbE Network ExpressModule* (NEM) allows up to five hosts to share a single 10 GbE network port with a dedicated PCIe endpoint for each host. Each server module appears to have its own 10 GbE NIC through virtualized NEM ASICs. With two virtualized NEM ASICs connected through the Inter-ASIC Link (IAL), a single 10 GbE port can act as the shared I/O for 10 server modules.

[1] See *http://www.sun.com/servers/blades/t6340/performance.xml* for details on the Sun Blade T6340 server module performance on industry-standard benchmarks.

**Figure 4. Sun Blade 6000 Virtualized Multi-Fabric 10 GbE Network ExpressModule**

The Sun Blade 6000 Virtualized Multi-Fabric 10 GbE NEM provides many benefits to this solution by reducing the cost of cabling and port management, simplifying deployment and reconfiguration, as well as eliminating the need for compatibility and interoperability issues with enterprise switches.

In this solution, the Sun Blade 6000 chassis is configured with one Sun Blade 6000 Virtualized Multi-Fabric 10 GbE NEM; a second NEM can offer redundancy and increased network performance, if required.

#### Sun Storage 7410 System

The Sun Storage 7410 system, part of Sun Storage 7000 Unified Storage Systems product line, provides an ideal storage platform that fulfills enterprise requirements along with ease of management. The storage system consists of a controller with 64 GB RAM and dual quad-core processors, and a 12 TB (twelve 1 TB drives) storage array. In this solution, Oracle database files were created using the NFS v3 data protocol.

#### Sun Rack II

All hardware components of this solution are installed into a Sun Rack II, Oracle's newest rack. This enclosure, optimized for vertical density, offers zero-RU power distribution units (PDU). The design, with PDUs installed on the rear side of the cabinet, enables supply of high amperage without taking up any usable space. The PDUs contain a metering unit that enables monitoring of the current being used by the equipment connected to the PDU.

### Software Components

The primary software components include Oracle Solaris OS, Oracle GlassFish Server, Oracle WebLogic Server, Oracle Database 11*g* with RAC, and Zeus load balancing software.

#### Oracle Solaris SPARC OS 10, update 8

This latest version of Oracle's enterprise-class operating system provides a rich, coherent platform for building and running applications. Oracle Solaris includes support for Oracle Solaris Zettabyte File System, an open-source 128-bit file system with built-in reliability and availability features; a high-performance networking stack for IPv4 or IPv6 networking; IP network multipathing to provide fault-tolerance and load spreading for network interface cards (NICs); virtualization features such as Oracle Solaris Containers and Oracle Solaris Zones; Dtrace Analytics, including capabilities that provide instrumentation for real-time performance analysis and debugging; and Predictive Self-Healing

technology that allows automatic diagnosis, isolation, and recovery from many hardware and application faults.

#### Oracle GlassFish Web Space Server

Oracle GlassFish Web Space Server, a new class of portal server, includes a comprehensive set of capabilities that enables organizations to provide users with personalized Web spaces that can be used for collaboration, business processes, communications, social networking, and much more. The Oracle GlassFish Server leverages the open-source Liferay Portal project and provides the following capabilities that simplify collaboration and development of Web content:

- *Content Management*—Oracle GlassFish Server provides an extensive set of content management capabilities. A WYSIWYG Web editor simplifies content generation. Content staging functionality controls content publishing by staging page content and assigning content managers. Multilevel custom workflow allows more than one reviewer or approver of content, and helps facilitate the scheduling of new content publication. Document and file management and a variety of file hosting capabilities—including user and community files, file check-in and check-out, Office Suite integration, versioning, tagging, search, and image and slide show viewing—are all supported.
- *Collaboration*—The collaboration tools included in the Oracle GlassFish Server enable easy collaboration within the enterprise and can be used to improve communication and the capture and sharing of intellectual property. Users have access to self-Web publishing capabilities such as wikis, blogs, and forums. Social networking and user-defined communities are also supported, allowing anyone to create a community to facilitate ad hoc productivity requirements.
- *Security*—Oracle GlassFish Server provides Web single sign-on (SSO) capabilities utilizing Federated Access Manager (FAM) for authentication and role management. WS-Federation and Security Assertion Markup Language (SAML) identity protocols are supported. Identity-based content delivery (IDBC) helps streamline business processes by directing the right content and applications to the right user, group, or role, to accelerate the search for relevant content.
- *Productivity*—Oracle GlassFish Server offers many features that improve productivity. A wiki server is included in the installation; users can aggregate existing social networks and perform self-Web publishing of content. Widget and gadget integration allows the easy incorporation of productivity tools such as calendars, date and time gadgets, and maps. The Oracle GlassFish Server also includes a Simple API for Workflow (SAW) that provides a common API for integrating with rules engines.

#### Oracle WebLogic Server 10, Release 3

Oracle WebLogic Server 10, Release 3, is a scalable, enterprise-ready Java 2, Enterprise Edition (J2EE) application server. The Oracle WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architecture (SOA). SOA is a design methodology aimed at maximizing the reuse of application services.

Oracle WebLogic Server, a complete implementation of the Sun Microsystems J2EE 5.0 specification, provides a standard set of APIs for creating distributed Java applications that can access a wide variety of services, such as databases, messaging services, and connections to external enterprise systems.

#### Oracle Database 11*g* with RAC

Oracle Database 11*g* with RAC is a clustered database designed for grid computing implementations. A proven technology that allows multiple servers to perform as a single large server, Oracle Database 11*g* with RAC forms a key foundation for enterprise database grids, providing the highest level of database availability along with flexibility in scaling. If a node in the cluster fails, the Oracle software continues running on the remaining nodes. If more processing power is necessary, new nodes can easily be added to the cluster. Key benefits of the Oracle Database 11*g* with RAC include:

- Single system image management
- Automated workload management, including workload monitoring and resource management
- Fast connection failover and recovery
- Data integration for fast disaster recovery
- Cluster verification and diagnostic tools

Note: Sun Cluster Advanced Edition for Oracle RAC software can be used to manage the multiple servers configured in a clustered deployment. Designed to improve service availability, resource utilization, and scalability, the Sun Cluster Advanced Edition for Oracle RAC software can also help accelerate disaster recovery and ease system management. In addition to providing I/O fencing to help guarantee data integrity, this software is designed for tighter integration with the Solaris OS to help minimize application downtime.

#### Load Balancer

Zeus Extensible Traffic Manager (ZXTM) software from Zeus Technology is used for load balancing user requests among the multiple Web and application servers. The benefits of using ZXTM as a load balancer include:

- Price/performance and flexible platform deployment choice
- TrafficScript and Java Extensions offer huge extensibility
- Clustered with distributed configuration
- SSL-Optimized for use of onboard cryptography with Sun CMT server platform.

Note: It is possible to use a hardware load balancer, instead of software such as ZXTM. In both cases, the load balancer can be clustered with a second one for redundancy and to achieve a higher degree of availability.

## Software Stack

Components of the software stack used by the Sun Systems for Enterprise 2.0 with Oracle solution are depicted in Figure 5. The stack can be logically divided into two main groupings: the infrastructure and the Oracle GlassFish Server.

Note: The OpenSSO/FAM 8.0 component is optional and was not used to test this solution

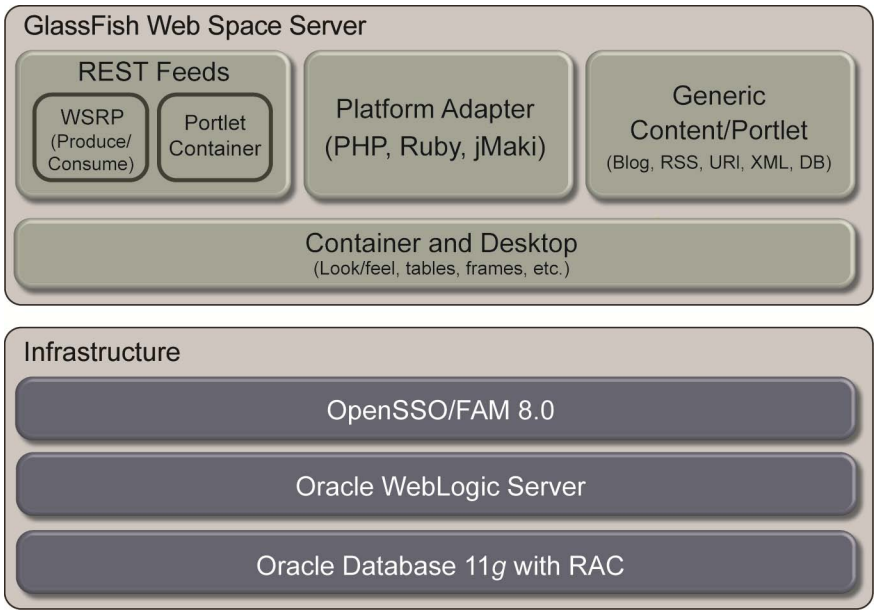



**Figure 5. Software stack.**

The Oracle GlassFish Server includes a container and desktop layer, plus components that generate content for the layout container.

- Container: The page container holds the header, navigation, tabs, and other "look and feel" items. The page container is rendered based on the theme definition and any defined portlets, which are hosted in the portlet container above.
- Other components, shown in the top row of Figure 5, generate content for the layout container. These include:

Web Services for Remote Portlets (WSRP)—allows remote portlets to be hosted on the local page, and also allows local portlets to be exposed (published) for inclusion on other Web properties and venues.

- Portlet container—holds all local portlets and manages their life cycles.
- Platform adapter—allows portlets written in different languages (such as PHP and Ruby) or other widgets from other platforms (such as jMaki and JavaFX) to be rendered locally as though they were local portlets.
- Generic content/portlet—represents all local or built-in portlets, such as wikis, blogs, and applications.

The infrastructure includes three primary components: identity authentication, application and Web server, and database.

- Identity/Authentication—OpenSSO: Provides authentication services and mapping of LDAP user groups and roles to the portal data model (such as communities and organizations).
- Application Server—Oracle WebLogic Server: Provides J2EE services and containment of the overall portal application. This component enables clustering of portal applications and Web service infrastructure for accessing underlying portal data.
- Database—Oracle Database 11*g*: Persists portal data, including user preferences and portal data such as CMS or collaboration content. This component is accessed through Hibernate or a Java Persistence API (JPA) provider, providing Java object-level access to the underlying entities stored in the database.

# Performance Characterization

This section describes the test environment, workload, and test description, and summarizes the results gathered from this testing of the Sun Systems for Enterprise 2.0 with Oracle solution.

## Test Environment

Figure 6 contains a logical representation of the test environment. Eight instances of the Oracle GlassFish Server and Oracle WebLogic Server are deployed in the test environment. In addition, two instances of the Oracle Database 11*g* database server are deployed.

The SLAMD distributed load generation engine is used to manage the test workload. Two instances of the workload agent generate the load for the test environment. Load balancing software distributes the workload across the multiple Oracle GlassFish Server and Oracle WebLogic Server instances.

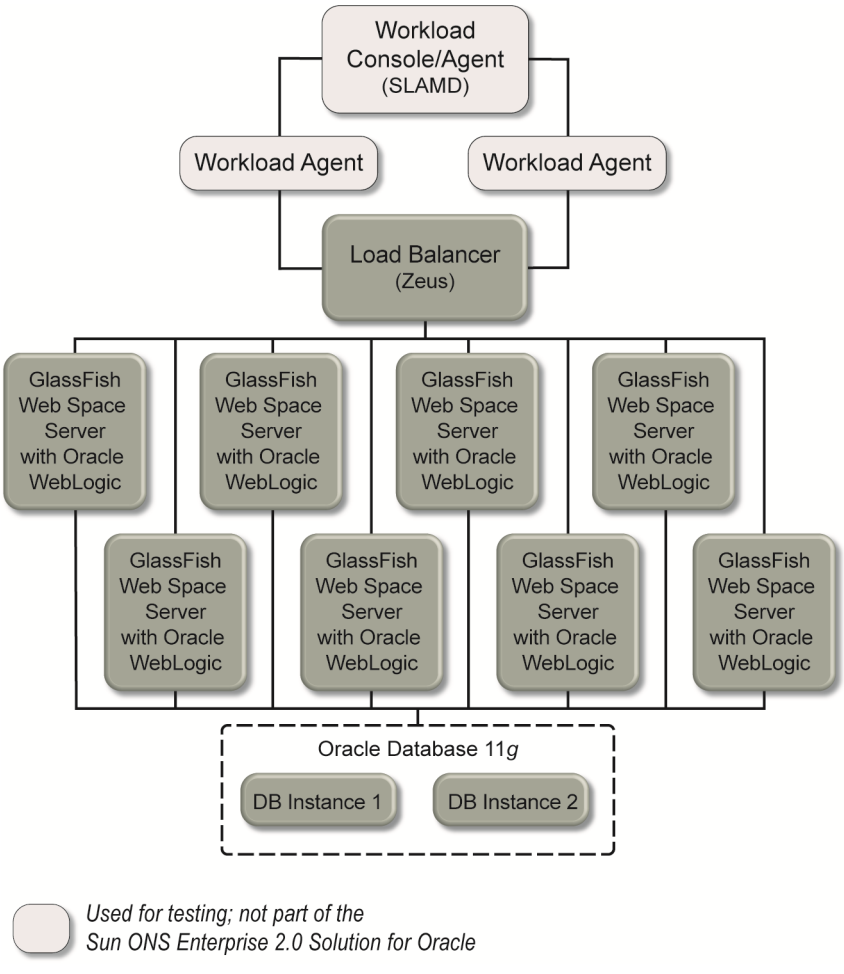



**Figure 6. Logical diagram of test environment**

Figure 7 shows the hardware and software configuration used for this testing and performance characterization. One Sun Blade 6000 chassis contains eight Sun Blade T6340 server modules. Each server module is configured with two UltraSPARC T2 Plus processors, 32 GB RAM, two hard disk drives, one Fabric Expansion Module (FEM), and one Network ExpressModule (NEM). Four server modules are configured to run two instances each of the Oracle GlassFish and Oracle WebLogic Server, and two server modules are configured to run one instance each of the Oracle Database 11*g* software. In addition, two server modules are used during testing as load generators.

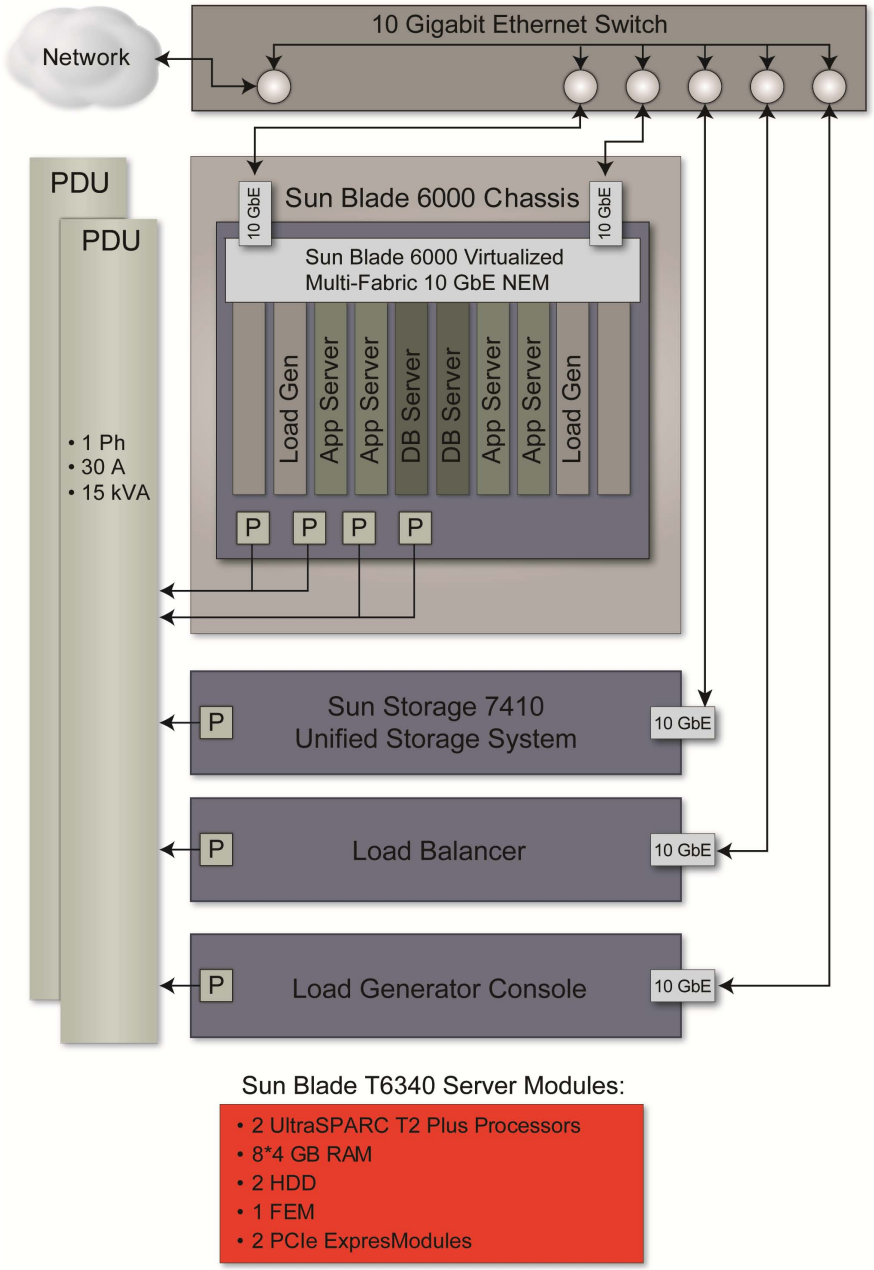



**Figure 7. Test environment.**

The Sun Blade 6000 chassis supports up to 10 server modules. This reference architecture incorporates six Sun Blade T6340 server modules: two configured as database servers and four configured as application/Web servers. For test purposes, two additional server modules were configured as load generators. During normal operation, these additional chassis slots remain available for expansion.

A Sun Storage 7410 Unified Storage System is used to store the Oracle Database 11*g* database files. All files are created using the NFS v3 data protocol.

A 10 Gigabit Ethernet switch is used for communication. A Sun Blade 6000 Virtualized Multi-Fabric 10 GbE NEM is installed in the Sun Blade 6000 chassis. This NEM has two 10 GbE network ports, allowing up to 10 server modules to have virtualized 10 GbE network access. The Sun Storage 7410 Unified Storage System and the systems used during testing for load balancing and the load generator console also connect via the 10 GbE switch.

## Workload and Test Description

Testing was performed to show the scalability of the solution. Oracle sees most portal deployments as having mostly read-heavy workloads; therefore the workload mix consisted of performing 85 percent reads and executing 15 percent writes.

Since user login operation is the most expensive operation in terms of resource utilization, a workload of users performing login operations was used to stress the scalability of this solution. When a user logs in for the first time in any given session, the portal server contacts the database to retrieve all user login information and then builds the custom home page that the user sees upon the first login.

The user login and home page information is written to the database when the user's home profile and authorized content list is created for the first time, and there are no additional writes to the database until the user creates additional content or edits existing content in the portal. Therefore, the testing was heavily biased towards reporting read scalability, with a small percentage of writes for initial home page compilation.

The workload consists of escalating numbers of users logging in over a predetermined time, and was emulated using SLAMD, a distributed load generation engine. The workload involved users logging in to the portal, navigating to one or more pages and not logging out; the focus was on navigating through pages that make use of the CMS that is part of the portal. Also, there are no delays between user logins; that is, no "think time." For the purposes of this test, a login is considered complete when the anonymous pages have been retrieved, the user/password has been sent, and the user has navigated to a specific page.

A database of over one million users, generated from actual names and public records provided by a local government census bureau, was used for the testing. Oracle Database 11*g* with RAC with two nodes was used as the database. Although Oracle Real Application Cluster Technology can provide scalability, it was mainly used to provide high availability because the nature of the workload did not impose a heavy load to the database.

This study distributed the workload across multiple servers using load balancing software provided by Zeus Technology. The Zeus Extensible Traffic Manager (ZXTM) product was extremely easy to

configure and powerful in its feature set. ZXTM is a licensed product, and could be swapped out for a hardware-based load balancer. This testing used ZXTM for speed of deployment and for ZXTM's ability to preserve JSessionID cookies.

## Data Collection

Test results were captured from the reports created by SLAMD after the execution of each test. Data from Oracle AWB snapshots and basic Oracle Solaris OS performance metrics, such as `sar` and `vmstat`, were captured during the execution of the tests.

Results are reported as logins per second. This number is much smaller than page views per second because, as described previous (see "Workload and Test Description" on page 12), a login is considered complete only after navigating through several pages.

## Test Results

Each server under test had 16 cores (128 threads), 32 GB of RAM, and virtualized NEM connectivity. For increased vertical scalability, multiple instances of the application/portal servers were run in six zones. And for increased horizontal scalability, multiple application/portal servers were deployed on more than one blade server module.

### Results with One Blade Server Module

Figure 8 on page 14 depicts the results from tests run with one Sun Blade T6340 server module, showing both the number of authenticated user logins per second and the response time for various thread counts. Recorded values from the test cycle are shown with a cubic spline interpolation.

As shown in Figure 8, the number of user logins scales in a near-linear fashion with increasing thread counts: as more threads are applied to the testing, simulating increasing user requests, a corresponding larger number of logins per second (LPS) are attained—until throughout reaches around 60 logins per second. Consequently, this test demonstrated near linear scaling while response times remained acceptable throughout the test.

The dip observed in the LPS towards the end of the test (and the corresponding increase in latency) corresponds to the start of garbage collection in the old (sometimes called tenured) generation of the Java heap. This is the Concurrent Mark and Sweep (CMS) collector which, even though it runs concurrently with the regular Java threads, does have an impact in overall throughput during the run.

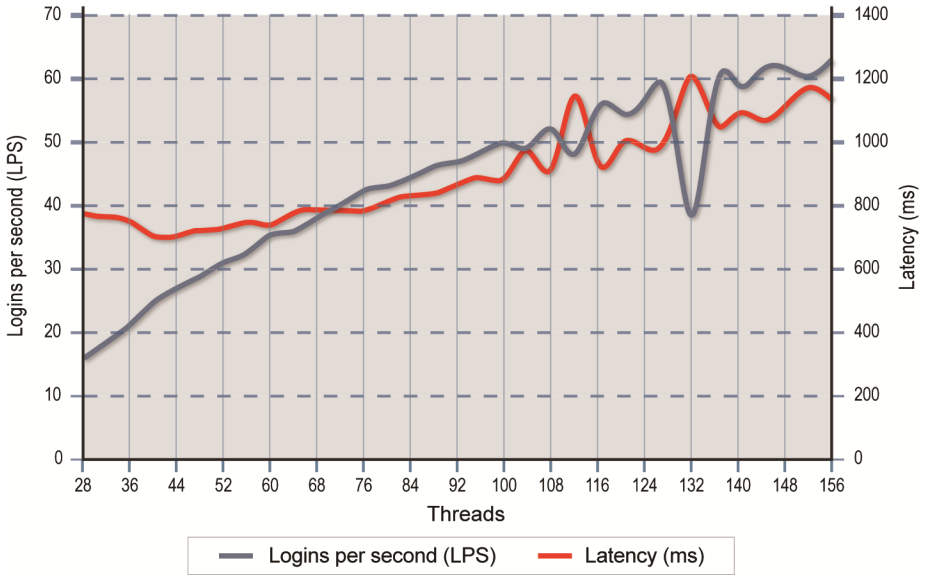



**Figure 8. Throughput and latency using single Sun Blade T6340 server module, six zones (Oracle GlassFish Server and Oracle WebLogic Server software).**

The memory utilization was about 4 GB for each Java Virtual Machine running on each zone of the Application/Portal server. At the point of peak throughput, CPU utilization reached 85 percent on the application/portal server. The resource utilization of the database server was low and the database response time was constant as the workload increased.

CPU utilization was measured at five second intervals using the UNIX® `sar` command. The trend of the CPU utilization time by user and system is depicted in Figure 9. The percentage of CPU utilization time spent idle was zero.

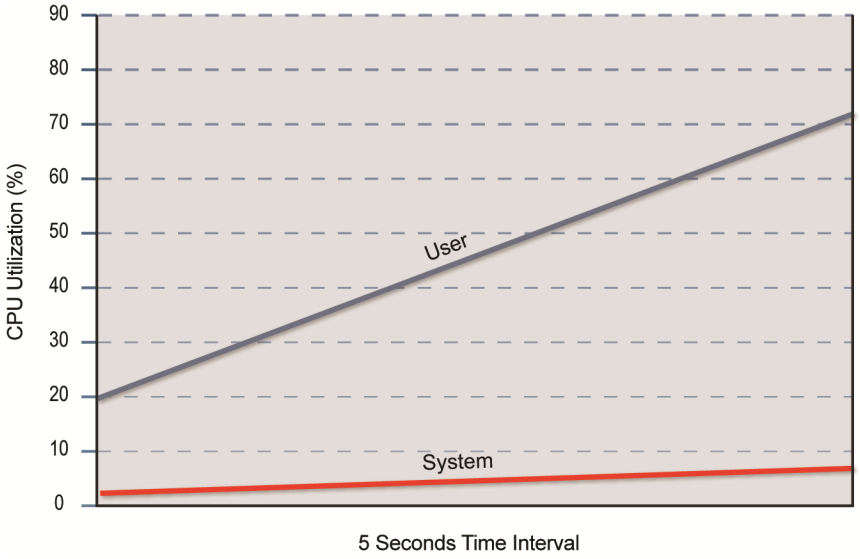



**Figure 9. Snapshot of CPU utilization—single Sun Blade T6340 server module.**

### Results with Two Blade Server Modules

Figure 10 depicts the results from tests run with two Sun Blade T6340 server modules, showing both the number of authenticated user logins per second and the response time for various thread counts. Recorded values from the test cycle are shown with a cubic spline interpolation.

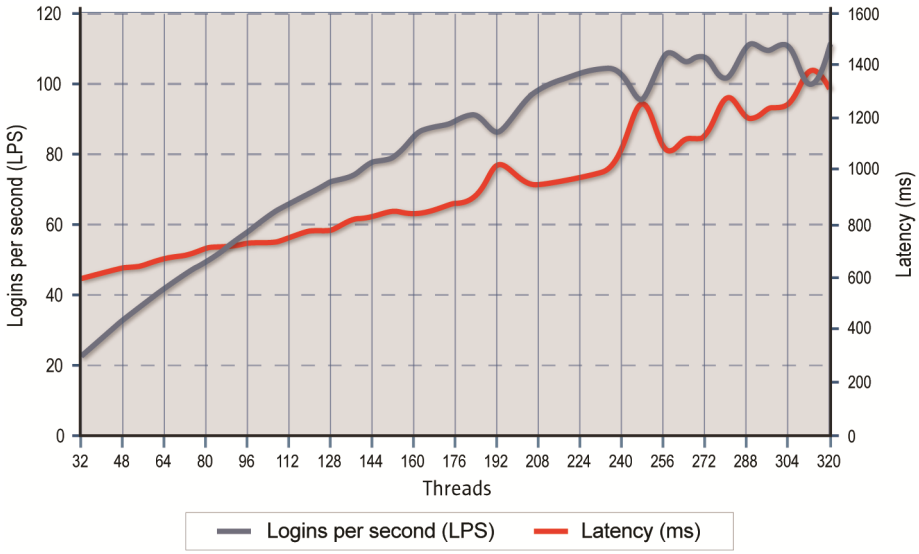



**Figure 10. Throughput and latency using dual Sun Blade T6340 server modules, six zones (Oracle GlassFish Server and Oracle WebLogic Server software).**

These results with two blade server modules are very similar to those with one blade server module. The linear scaling of user logins is evident, as is the acceptable response time across all test runs. Again, towards the end of the test, there is a dip in throughput caused by a CMS event as well as recovery and continued throughout increase after the CMS run is completed.

In a duration of about half hour, over 178,000 users logged in and over 500,000 Web pages were generated. The resource utilization of the database server was low and the database response time was constant as the workload increased.

CPU utilization was measured at five second intervals using the UNIX sar command. The trend of the CPU utilization time by user and system is depicted in Figure 11. The percentage of CPU utilization time spent idle was zero.

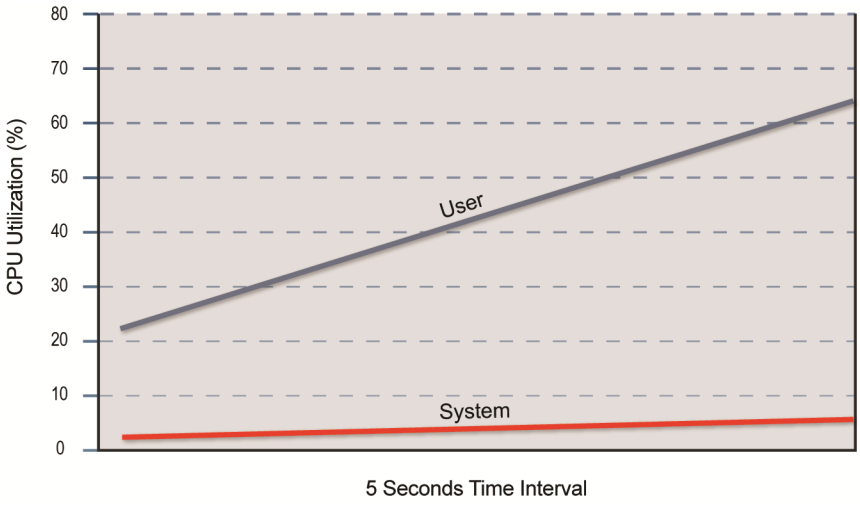



**Figure 11. Snapshot of CPU utilization—dual Sun Blade T6340 server modules.**

## Results Summary

This study measures the rate as logins per second. It is assumed that logins are the most expensive operation and that the number of actual users to logins is approximately half the number of people logging in any given time to the number of active users accessing the portal. These assumptions represent a busy and casual portal site, which may need to be adjusted for a specific use case.

Based on the workload used during this test, It is estimated that a configuration of one Sun Blade T6340 server module with six zones can actively serve portal Web pages to 73,200 active users at any given time. This number is calculated by taking the number of logins per hour (61 logins/second * 60 seconds * 60 minutes = 219,600 logins/hour) and then estimating that at any given time approximately a third of those users will be actively doing something that requires network traffic to the site. The estimated throughput might vary if additional processes—such as connecting the portal to back end services like a mail server—are added to the workload, as these processes would increase the response time.

As application needs increase, it is a straightforward task to grow this solution to meet new demands:

- Physically add another blade server module and install the OS with unique networking (or have the OS disk preinstalled ahead of time).
- Configure an Oracle WebLogic application server running the Oracle GlassFish portal software on the new blade server module (can be part of the OS preinstallation).
- Connect that application server to the Oracle database.
- Add knowledge of the new application server to the load balancer.

## Summary

Designed for medium to large deployments in both extranets and intranets, Oracle's Sun Systems for Enterprise 2.0 solution combines all required components—compute, storage, networking, software, and services—in a fully-tested, integrated implementation.

Key hardware components include Oracle's Sun Blade 6000 Modular System, configured with up to 10 Sun Blade server modules from Oracle running a choice of operating systems; Oracle's Sun Blade 6000 Virtualized Multi-fabric NEMs; and Oracle's Sun Storage 7410 system. Proven highly available and scalable enterprise-level software, including Oracle Solaris 10 OS, Oracle Database 11*g* with RAC, Oracle WebLogic Application Server, and the widely popular Oracle GlassFish Server (portal) software, provides an instantiation of the Oracle's Sun Systems for Enterprise 2.0. This solution provides flexibility, scalability, and unparalleled performance with RAS capabilities throughout the entire hardware and software stack, and can help enterprises quickly deploy Web services and adapt their existing applications to Enterprise 2.0 requirements.

# Appendix

The following sections describe the procedures used to set up and configure the test environment.

## Preliminary Setup

Perform the following steps on all servers.

Note: Use of a "multiple terminal window at once" program (such as iTerm on Mac, PuTTY Command Sender on Windows, or ClusterSSH on Linux and Oracle Solaris platforms) can simplify installation by enabling the administrator to enter commands once and have them sent to all servers.

1. Install Oracle Solaris on all Sun Blade server modules.
2. Set root login permission:

```
# su - root
# rolemod -K type=normal root
```

3. Edit the `/etc/ssh/sshd_config` file and change the following line to permit root logins:

```
PermitRootLogin yes
```

4. Use the svcadm command to restart the `ssh` remote login program:

```
# svcadm restart ssh
```

5. Edit the `/etc/default/login` file and comment out the `CONSOLE` entry to allow root logins on all connections.

Caution: The previous steps compromise security by allowing remote root logins to each server module. It is recommended to remove these login permissions for root at the completion of the setup procedure.

## Load Balancer Configuration

Perform the following steps on the server that will be running the load balancing software.

1. Decompress and extract compressed tar archive file `ZXTM_51r1_SunOS.tgz` residing in the `/opt/installs/zeus directory`.
2. As root, execute the install script `zinstall`:
   a. Accept license agreement.
   b. Choose the directory to install Zeus software (this directory will be known as `ZEUSHOME`). This example uses `/opt/zeus`.

c. For the first time, configure ZXTM software by executing the configure script. (This script resides in `ZEUSHOME/zxtm`.). A configuration file `global.cfg`, pointing to `global.cfg -> conf/zxtms/oraweb-49.sfbay.sun.com`, is created with the following content:

```
gid root
uid root
controlport 9080
java!port 9060
```

3. Once Zeus is up and running, perform the following steps:

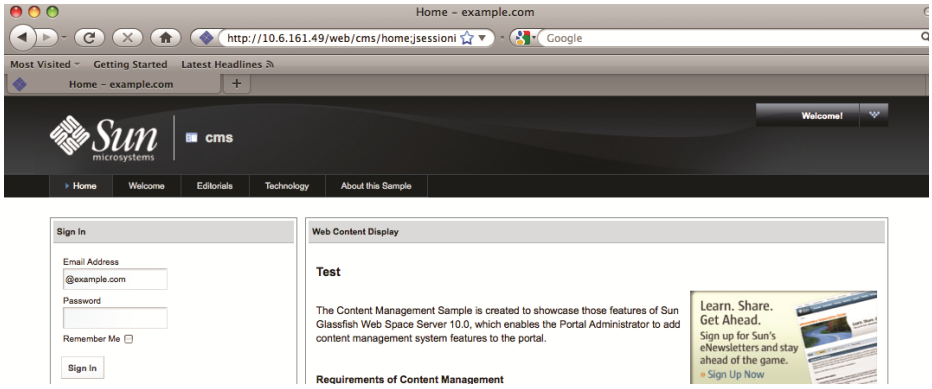
   a. Log in as admin and from the **Wizards** drop-down menu choose **Manage a new Service**.

   b. Give the service a name (like "Portal on UST2 blades," choose the protocol (HTTP) and the port (normally 80), and click **Next**. Next, enter all the nodes (or, as in this case, zones) that will handle the load, and click **Next**.

   c. Then, review the **Summary** and click **Finish**.

   d. After the service has been created, click on the service name and scroll down to **Session Persistence**. Click **Edit** and then choose either **Transparent session affinity** or **J2EE Session Persistence**—both will work.

4. Check that the service has been started and manually test the service by opening a browser connection to the load balancer machine. If you not only get the anonymous portal page, but you can also successfully log in, then the service has been configured correctly.




5. From the ZXTM Administrator GUI, execute **Manage a New Service Wizard** (box in the upper right corner).

   Note:

   - To start ZXTM, use the command: `#ZEUSHOME/start-zeus`
   - To stop ZXTM, use the command: `#ZEUSHOME/stop-zeus`
   - To access the ZXTM Administrator Server information: `https://oraweb-49:9090` (login/password is `admin/admin123`)

## Database Server Installation

Oracle Database 11*g* with RAC was selected to provide scalability and high availability for the solution.

The database was deployed on two nodes, with each node running on a Sun Blade T6340 server module. The database files were created with NFS from a Sun Storage 7410 Unified Storage System.

The following steps were followed to setup the database:

1. Using the Unified Storage System user interface (via a browser), create the following NFS file systems for the database files under a project called `oraweb`:

| FILE SYSTEM NAME | RECORD SIZE | USAGE | MOUNT POINT |
|---|---|---|---|
| ocr | 8 KB | Oracle Cluster Registry files and voting file | /export/oraweb/ocr |
| oradata | 8 KB | Database files | /export/oraweb/oradata |
| oralog | 8 KB | Database online logs and control file | /export/oraweb/oralog |
| oraarchive | 128 KB | Archived redo logs | /export/oraweb/oraarchive |

   - Be sure to add the hostname of the database server to the NFS exception list for each of the systems and check the root access option.

Note: For more information, refer to the Sun article "Deploying Oracle Real Application Clusters on Sun Storage 7000 Unified Storage System," available online at: *http://www.sun.com/bigadmin/features/articles/7000_oracle_rac.jsp*

2. Prepare the two nodes where the Oracle database instance is going to run:
   a. Update the `/etc/system` file as follows:

```
* Parameters for Oracle DB
set noexec_user_stack=1
set semsys:seminfo_semmns=1024
set semsys:seminfo_semvmx=32767
set shmsys:shminfo_shmmax=21474836480
#set maxphys=2097152

* Tunings for soft rings (nxge and e1000g)
set ip:ip_squeue_fanout=1
set ip_squeue_soft_ring=1
set ip:ip_soft_rings_cnt=16
```

b. Configure public network ports and private interconnect (for the Oracle database).

- Using two private interconnects requires manual setup of the IPMP (IP Multipathing) option if Oracle Database 11*g* with RAC is not deployed with Oracle Solaris Cluster. Oracle Solaris Cluster will take care of configuring the two ports for IPMP automatically.
- To provide IPMP capability, all private ports must be in the same subnet.

c. Update the `/etc/hosts` file with private interfaces and virtual IP addresses.

```
# oracle RAC virtual IP@
192.168.1.88 oraweb10g-40-vip
192.168.1.89 oraweb10g-47-vip
# oracle RAC private interconnect IP@
10.10.10.10 oraweb10g-40-priv
10.10.10.20 oraweb10g-47-priv
```

Note: With Oracle Solaris 10 OS, it is recommended to set up kernel parameters through resource control via projects instead of defining them in the `/etc/system` file; however, Oracle Cluster Ready System (CRS) is responsible for starting all registered Oracle-related services and components including Oracle databases. CRS starts the database as oracle user without inheriting the kernel settings specified by the project assigned to the oracle user; instead it looks for the `shmsys:shminfo_shmmax` parameters of the /etc/system file before bringing up the database. Others IPC parameters work fine by being defined to the project assigned to oracle user.

d. Create group `dba` and user `oracle`.

e. Establish password-less `ssh` between the nodes:

First, login as `oracle` user, and execute the following commands to create the `rsa.pub` and `dsa.pub` files:

```
oracle# ssh-keygen -t rsa
oracle# ssh-keygen -t dsa
```

Then, change to the `~/.ssh` directory and update the `authorized_keys` file with the content of the `rsa.pub` and `dsa.pub` files:

```
oracle# cat id_rsa.pub > authorized_keys
oracle# cat id_dsa.pub >> authorized_keys
```

f. Create Oracle base, home, and crs directories and change ownership to `oracle`.

g. Update the `/etc/vfstab` file and mount the file systems that were created in step 1 (see page 20).

```
oraweb10g-uss:/export/oraweb/ocr - /oracle/products/11gR1/ocr
nfs - yes rw,bg,hard,nointr,rsize=32768,wsize=32768,
proto=tcp,vers=3,noac,forcedirectio

oraweb10g-uss:/export/oraweb/oradata - /oradata/data
nfs - no rw,bg,hard,nointr,rsize=32768,wsize=32768,
proto=tcp,vers=3,noac,forcedirectio

oraweb10g-uss:/export/oraweb/oralog - /oradata/logs
nfs - no rw,bg,hard,nointr,rsize=32768,wsize=32768,
```

```
proto=tcp,vers=3,noac,forcedirectio

oraweb10g-uss:/export/oraweb/oraarchive - /oradata/archive
nfs - no rw,bg,hard,nointr,rsize=32768,wsize=32768,
proto=tcp,vers=3,noac,forcedirectio
```

h. Set the following Oracle environment variables:

- `ORACLE_BASE = /oracle/products/11gR1`
- `ORACLE_CRS_HOME = /oracle/products/11gR1/crs`
- `ORACLE_HOME = /oracle/products/11gR1/db`
- `ORACLE_OCR = /oracle/products/11gR1/ocr`
- `ORACLE_SID = ORASUN1` (for node 1; use `ORASUN2` for node 2)

by editing the following two files in the `/export/home/oracle` directory:

File: `.profile`

```
. ~/set_oracle_env.sh
PS1='${LOGNAME}:${ORACLE_SID}>'
export PS1
umask 022
```

File: set_oracle_env.sh

```
#!/usr/bin/bash
clear
echo “Choose Oracle Home & hit RETURN”
echo “Enter 1 to use Oracle Home DB”
echo “Enter 2 to use Oracle Home CRS”
echo “Hit any other key to skip Oracle Home selection”
read answer
if      [[ $answer = 1 ]]
    then
    ORACLE_HOME=/oracle/products/11gR1/db ;
    ORACLE_SID=ORASUN1
elif    [[ $answer = 2 ]]
    then
    ORACLE_HOME=/oracle/products/11gR1/crs
    ORACLE_SID=SUNCRS1
else
    ORACLE_HOME=/export/home/oracle
    ORACLE_SID=xxx
fi

ORACLE_BASE=/oracle/products/11gR1
export ORACLE_BASE
export ORACLE_HOME
export ORACLE_SID
cd $ORACLE_HOME
CRS_HOME=/oracle/products/11gR1/crs
export CRS_HOME
ORACLE_OCR=/oracle/products/11gR1/ocr
export ORACLE_OCR
ORACLE_STAGE=/OracleDVD
export ORACLE_STAGE
export PATH=/export/home/oracle/local/bin:
/oracle/products/11gR1/db/bin:/usr/bin:
/usr/sbin:/usr/openwin/bin:/usr/X11/bin:/usr/dt/bin:
/oracle/Opatches/OPatch:$PATH
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:
/oracle/products/11gR1/db/lib
EDITOR=vi
export EDITOR
if [ -t 0 ]; then
   stty intr ^C
fi
# stty erase ^?
alias sql=’sqlplus / as sysdba’
alias crsi=’/oracle/products/11gR1/crs/bin/crs_stat -t’
alias xxx=”. ~/set_oracle_env.sh”
alias oh=’cd $ORACLE_HOME’
alias stage=’cd $ORACLE_STAGE’
```

Note: The next steps (Step 3 to Step 6) are executed on only one node.

3. On one node, install the Oracle Clusterware software as `oracle`:
   a. Check the Oracle Database 11*g* infrastructure configuration by executing the `runcluvfy.sh` script.
   b. Execute the `runInstaller` command.
4. Install the Oracle Database 11*g* binaries by executing the `runInstaller` command.
5. Create and configure the database using the Oracle Database Configuration Assistant, `dbca`, with the following parameters:
   - sga_target = 3072M
   - processes = 2000
6. Configure the listener by executing the Oracle Network Configuration Assistant, `netca`. Example `listener.ora` file from node 1:

```
LISTENER_ORAWEB10G-40 =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-40-vip)
(PORT = 1521)(IP = FIRST))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.100)
(PORT = 1521)(IP = FIRST))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
      )
    )
  )
```

And `listener.ora` file from node 2:

```
LISTENER_ORAWEB10G-47 =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-47-vip)
(PORT = 1521)(IP = FIRST))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.107)
(PORT = 1521)(IP = FIRST))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
      )
    )
  )
```

7. Configure the `tnsnames.ora` file on the database server and all client machines that need to access the database. For example:

```
ORASUN2 =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-47-vip)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ORASUN.sfbay.sun.com)
      (INSTANCE_NAME = ORASUN2)
    )
  )

ORASUN1 =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-40-vip)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ORASUN.sfbay.sun.com)
      (INSTANCE_NAME = ORASUN1)
    )
  )

ORASUN =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-40-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = oraweb10g-47-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ORASUN.sfbay.sun.com)
    )
  )
```

## Application/Oracle GlassFish Server

This example sets up the physical servers to run multiple instances of the application server. The Oracle WebLogic Application Server is shown to scale well when running multiple instances of the application server on a single physical machine. Therefore, this test sets up six zones on each physical blade server and installs the application server on each zone.

### Preliminary Setup and Configuration

Run the following commands on all physical servers that will host application servers.

1. As a convenience, create a login profile. It is advantageous to create a variable for each blade server module that references the name of the application server zones. The `.bash_profile` must be edited on each application server to have unique names. For each instance, replace `appserv-XXX-zoneX` with the names of the application server zones. When using a utility that sends a command to multiple terminals, the same variable name can be used. This can be a significant time saver.

For example, the following `~/.bash_profile` can be created on each server module, substituting the appropriate values for `ASZONE0` to `ASZONE6`:

```
source ~/.bashrc
export JAVA_HOME=/usr/jdk/latest/
export PATH=$PATH:/usr/local/bin
export ASZONE0=appserv-XXX-zone0
export ASZONE1=appserv-XXX-zone1
export ASZONE2=appserv-XXX-zone2
export ASZONE3=appserv-XXX-zone3
export ASZONE4=appserv-XXX-zone4
export ASZONE5=appserv-XXX-zone5
export ASZONE6=appserv-XXX-zone6
```

After creating the `~/.bash_profile`, evaluate the file:

```
# source ~/.bash_profile
```

2. Create the ZFS pools and set the mount points:

```
# zfs create rpool/zones
# zfs set mountpoint=rpool/zones
```

3. Next, six zone creation scripts are created. This example installs six zones per application server, as the architecture can better utilize resources with six application servers running on any given blade server module when higher memory configurations are used. The six zone creation scripts, `zone0.cfg` through `zone6.cfg` are essentially identical. The only differences are the zonepath and the IP address used. Only one IP address is defined, a private 10 GbE connection used for database/storage connectivity.

   Create the following six scripts, substituting the correct IP address for your network:
   Example configuration script: `zone1.cfg`:

```
create -b
set zonepath=/zones/zone1
set autoboot=false
add net
set address=192.168.1.40
set physical=hxge0
end
```

4. Configure the six zones using the script files defined in the previous step, and then install the zones using the `zonecfg` and `zoneadm` commands. For example:

```
# zonecfg -z $ASZONE1 -f zone1.cfg
# zoneadm -z $ASZONE1 install
```

5. After the installation process has finished, set the `autoboot` property to true on all zones. For example:

```
# zoneadm -z $ASZONE1 set autoboot=true
# zoneadm -z $ASZONE1 commit
```

6. Boot the newly created zones. For example:

```
# zoneadm -z $ASZONE1 boot
```

7. Log in to the console of the domains. For example:

```
# zlogin -C $ASZONE1
```

Note: If a multiple-terminal command sender utility is being used, these steps can be performed in parallel. In not, two sessions per physical application server are needed for all of the zones. Once the zones are installed and networked, then can be logged into directly.

## Oracle WebLogic Server 10*g* Installation and Setup

There are three methods to install Oracle WebLogic products: graphical, console, and silent mode. The following screen shots show the steps taken to interactively install Oracle WebLogic Server using the Graphical User Interface (GUI).

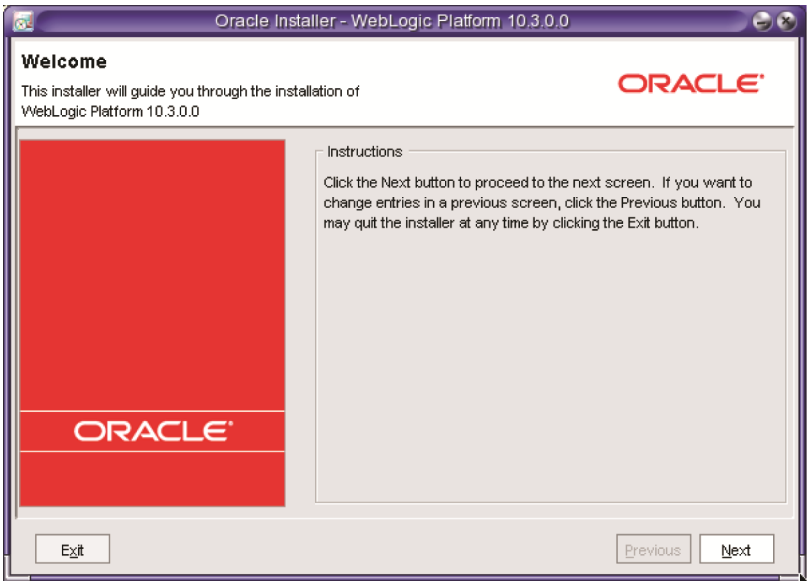
1. Download Oracle WebLogic Server 10*g* Release 3 (10.3) Sun Solaris SPARC (32-bit) from the Oracle Web site `server103_solaris32.bin` into a temporary place (this example used the directory `/opt/installs`).
2. Execute the following command from a terminal window:

```
# /opt/intalls/server103_solaris32.bin
```

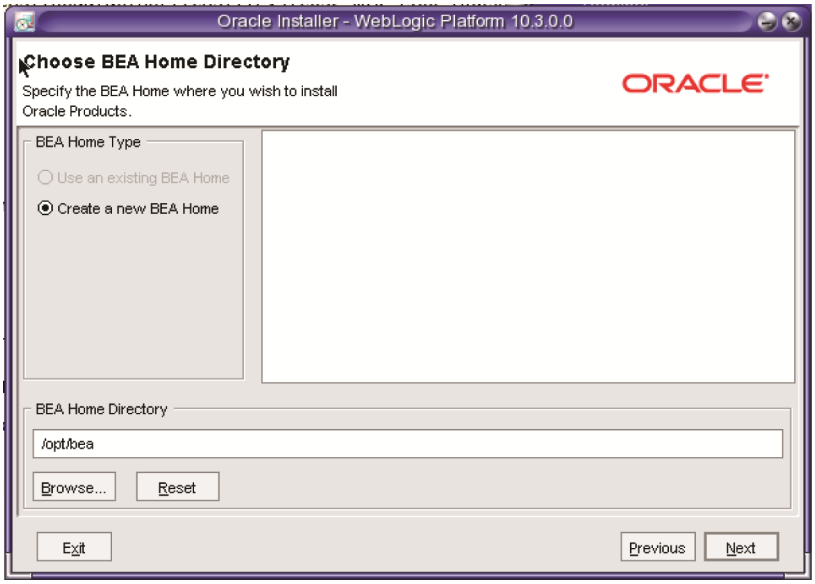
   The following message is displayed:

```
“Extracting 0%.....................................................
.....................................................100%”
```

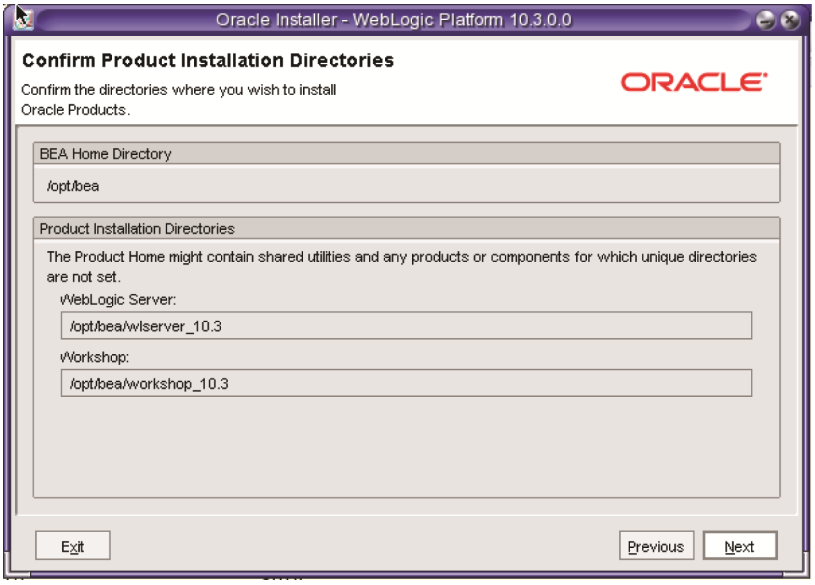
Before the following screen is displayed:



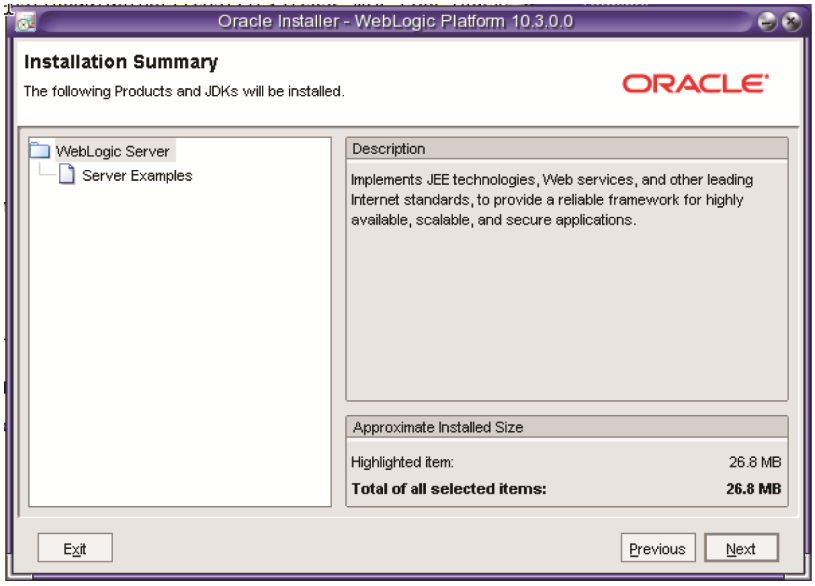

3. Enter the BEA Home Directory; this example uses `/opt/Bea`:



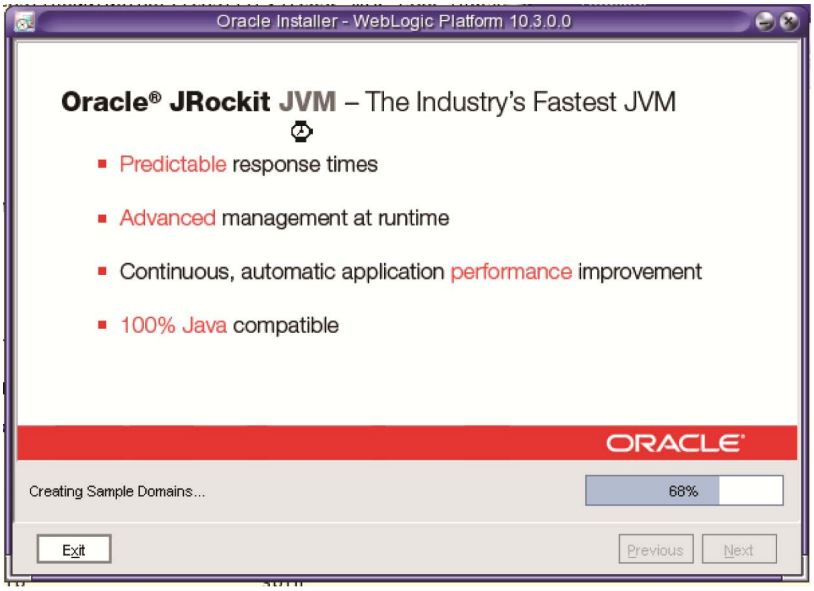
4. Choose install type (Complete: WebLogic Server and Workshop, or Workshop). This example uses Complete. Select **Product Installation Directories** and confirm the selection:



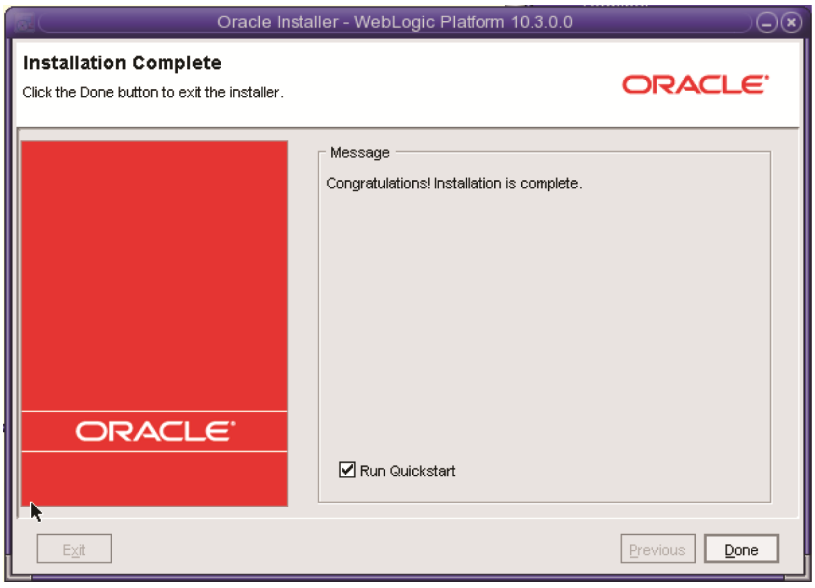

5. The following Installation summary is displayed:



6. Wait until the installation completes. The following screen displays the installation progress:




7. When the installation is complete, the following screen is displayed:



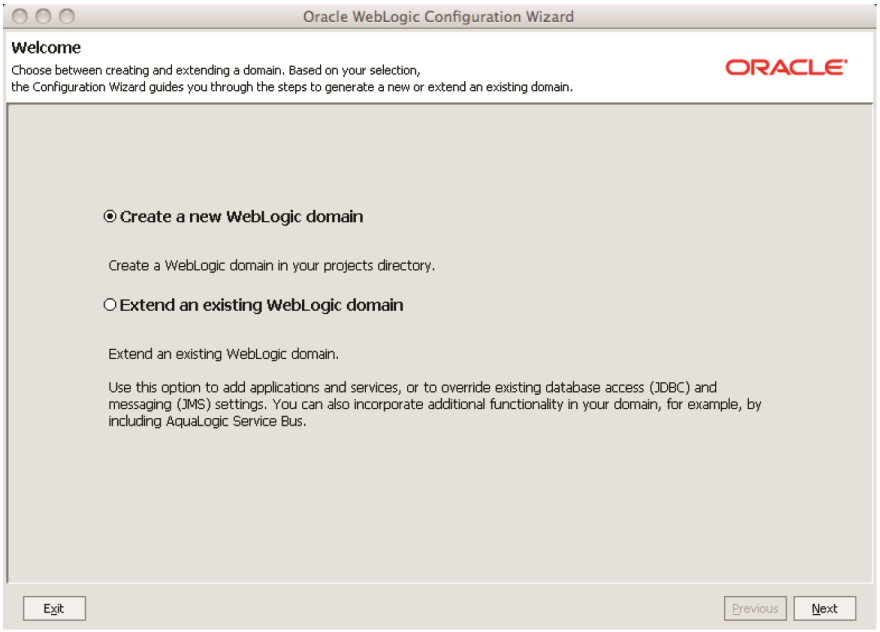
### Oracle WebLogic Server Domain Creation

This step is optional, but creates a cleaner deployment. A domain is created by default during the installation and setup of Oracle WebLogic (see "Oracle WebLogic Server 10*g* Installation and Setup" on page 27).

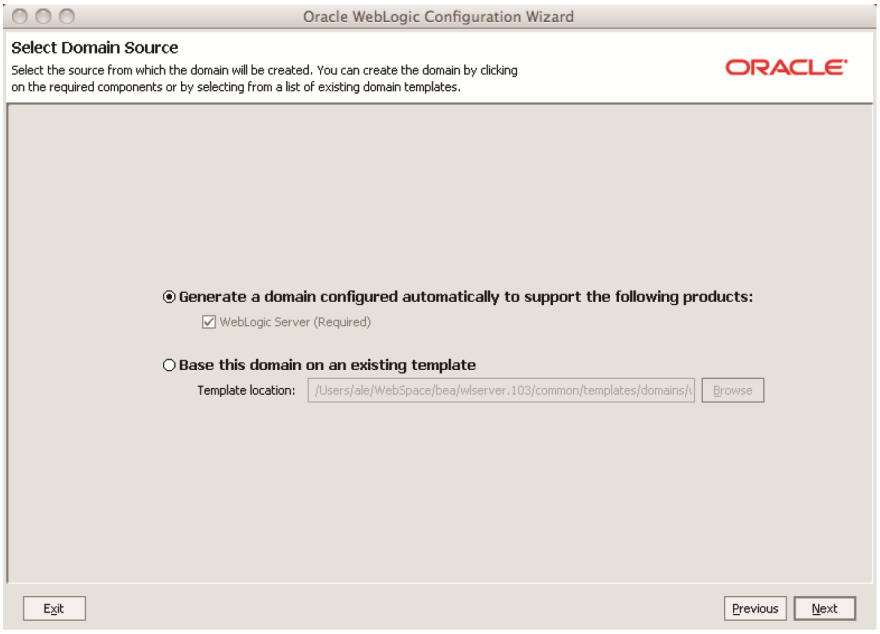
1. Run the following script:

```
# common/bin/setup.sh
```

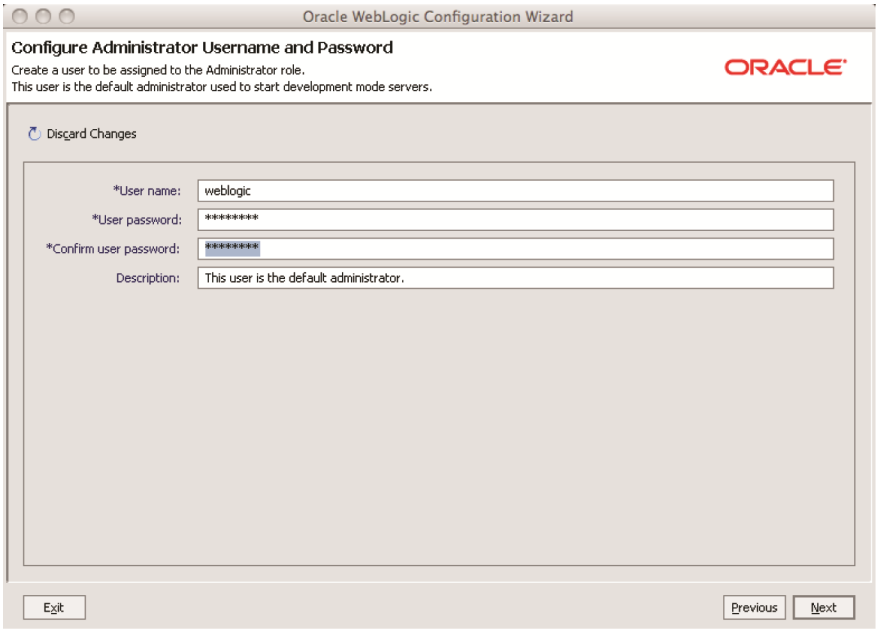
2. Select **Create a new WebLogic domain** and click **Next**:



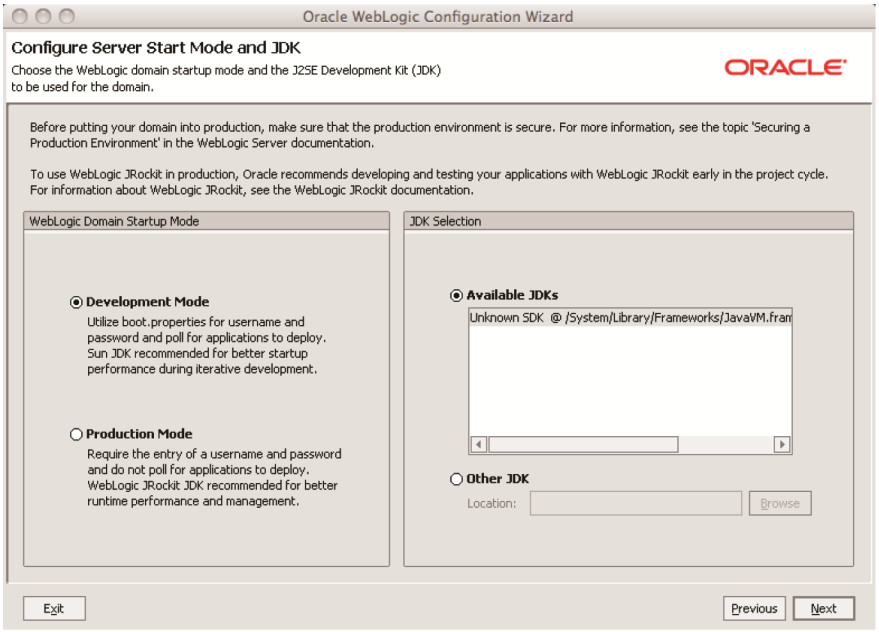
3. Choose **Generate a domain configured automatically to support the following products: WebLogic Server**, unless you have previously defined a domain template. Having domain template is optional, but can be useful if you frequently create and then configure domains. Then click **Next**:




4. Enter the user name and password. In this example, `weblogic` is chosen as the user name and password. Then click **Next**:

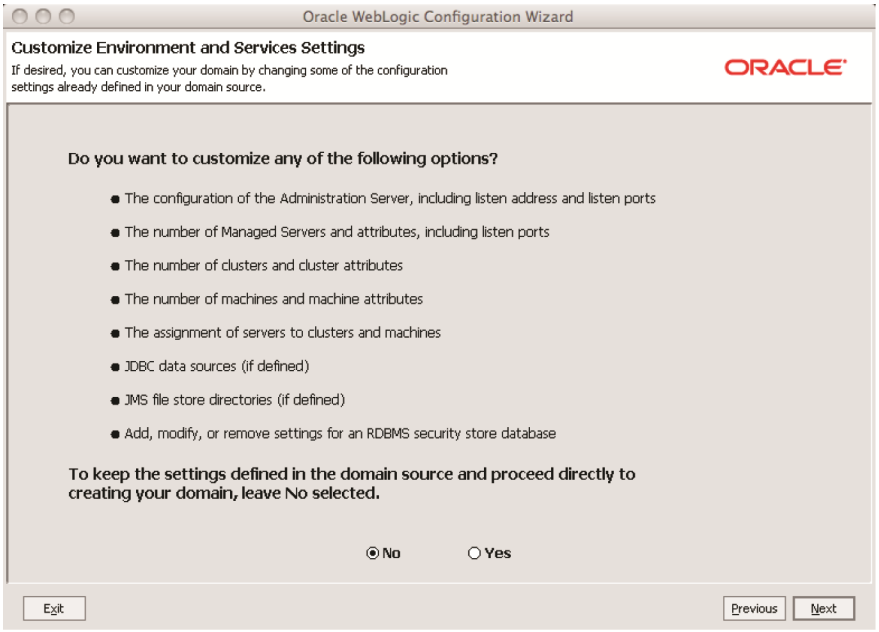


5. Choose Development or Production mode and the JDK version, then click **Next**:

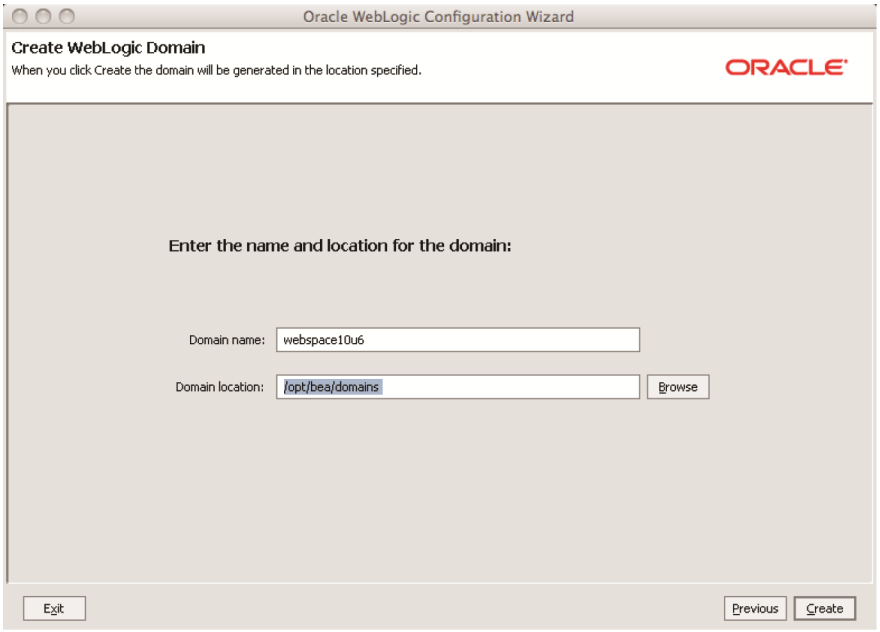



Note: For SPARC® processor deployments, Oracle WebLogic ships the Sun JDK 1.6.0_05. However, we chose to use the newer JDK 1.6.0_16.

6. Choose whether to customize the Oracle WebLogic options, and then click **Next**. In this example, no further customizations were done using this wizard.



7. Choose the Domain name and location, and click **Create**:



### Oracle GlassFish Web Space Server 10.0 Installation And Setup

Oracle GlassFish Web Space Server 10.0 software is available as a downloadable zip file from the Sun GlassFish Web Space Server page (`https://webspace.dev.java.net/`) in three general package configurations:

- *Integration bundle*: Includes Oracle GlassFish Web Space Server 10.0 software only, and is meant to be installed into an existing GlassFish Enterprise Server v2.1 or Oracle WebLogic Server 11*g* environment. These bundles do not include application server software or the GlassFish Web Space Server sample applications.
- *Deployment bundle:* Platform-specific packages that include GlassFish Web Space Server 10.0 software and GlassFish Enterprise Server software. These bundles do not include the GlassFish Web Space Server sample applications.
- *Evaluation bundle:* Platform-specific packages that include GlassFish Web Space Server 10.0, Sun GlassFish Enterprise Server v2.1, plus all sample Web Space Server applications. These bundles are recommended primarily for evaluation purposes rather than for deployment in a live production environment.

The following steps are used to install the Sun GlassFish Web Space Server software. Before starting the installation, please note:

- To avoid the chance of overwriting existing Oracle WebLogic configuration settings or files, it is recommended to unzip the GlassFish Web Space Server package outside of the existing Oracle WebLogic directory structure. See the *GlassFish Web Space Server 10.0 Installation Guide* for more detailed information.
- Make sure Oracle WebLogic Server 10*g* or 11*g* software and Apache Ant 1.7 are installed and running before proceeding with the following instructions.

1. Open the Oracle WebLogic administration console:

```
http://localhost:7001/console
```

2. Navigate to the domain in which the GlassFish Web Space Server will be installed, and then click on the **Web Applications** tab.
3. Enable the **Optimistic Serialization** option and save the setting.
4. Stop the Oracle WebLogic domain.
5. Download the `webspace-10.0.6-for-weblogic.zip` integration bundle and unzip in the directory of your choice (`/opt/installs/webspace10u6` was used in this test).

6. Change to the `/opt/installs/webspace10u6/webspace-for-weblogic/webspace/application` directory:

```
# cd /opt/installs/webspace10u6/webspace-for-weblogic/webspace/
application
```

7. Run the install.xml script (residing in `/opt/installs/webspace10u6/webspace-for-weblogic/webspace/application`):

```
# ant -f ./install.xml
```

Note: Refer to the *GlassFish Web Space Server 10.0 Installation Guide* for detailed instructions on installing the GlassFish Web Space Server, including deploying on Oracle WebLogic server and specifying the use of the Oracle database.

8. Follow the prompts to complete the GlassFish Web Space Server installation in the Oracle WebLogic domain to be used and deploy the GlassFish Web Space Server WAR files.

   The Web Space Server WAR files are copied to the `ws-install-dir/va/webspace/war-workspace/finals/` directory. It is recommended that you choose the installer option "do not deploy the WAR files". With this option, the installer just creates and saves the WAR files, but does not automatically deploy them. Later on, you should either deploy the WAR files with the regular deployment tools that are part of Oracle WebLogic (GUI or console), or just drop them one by one in the autodeploy directory if autodeploy is enabled.

### Additional Configuration and Setup

The following steps are used to complete the configuration.

1. Configure the GlassFish Web Space Server to work with the Oracle database.

   This configuration is done either after the execution of the `install.xml` script (see Step 7 in previous section, "Oracle GlassFish Web Space Server 10.0 Installation and Setup") or after the generation of the WAR files. Changes can be performed by manually modifying the `portal-ext.properties` file and executing the following `synchronize.xml` command:

```
# ant -f ./synchronize.xml
```

Note: The following modifications were made to the `portal-ext.properties` file for this study. While performance results measured in this document used these settings, results may vary and use of these settings may not increase performance in all deployments.

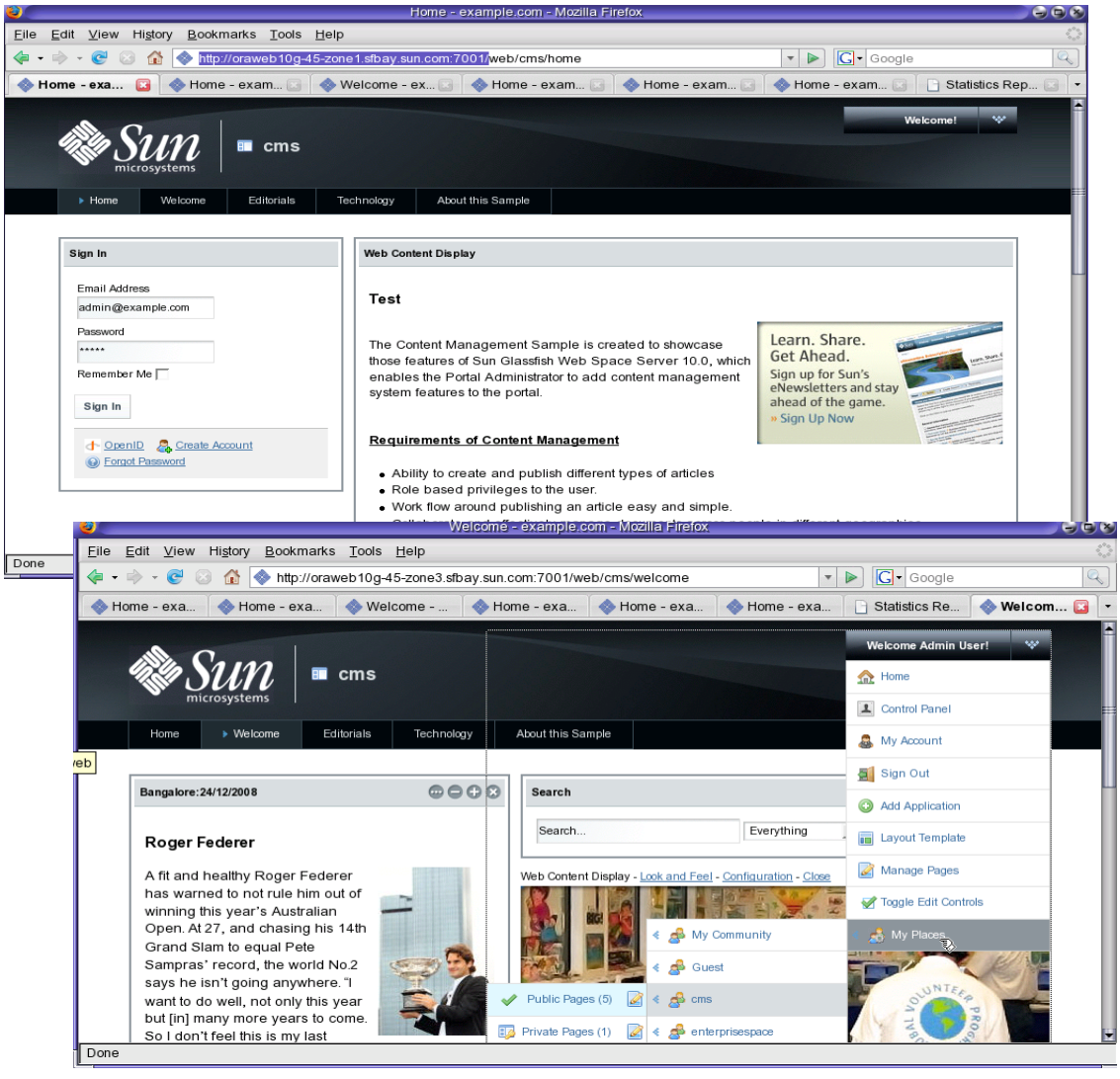
```
# Point Web Space Server to Oracle DB
jdbc.default.driverClassName=oracle.jdbc.driver.OracleDriver
jdbc.default.url=jdbc:oracle:thin:@//<DBSERVER>:1521/<DBNAME>
jdbc.default.username=<DBUSER>
jdbc.default.password=<DBPASSWORD>

# Increase the DB connection pool
jdbc.default.maxPoolSize=72
jdbc.default.minPoolSize=16

# Disable autodeploy for the portal server
auto.deploy.enabled=false
auto.deploy.interval=0

# Settings for typical production site
last.modified.check=false
theme.css.fast.load=true
javascript.fast.load=true
portlet.css.enabled=false
velocity.engine.resource.manager.cache.enabled=true
com.liferay.portal.servlet.filters.layoutcache.LayoutCacheFilter=true
auto.deploy.enabled=false
auto.deploy.interval=0
```

2. Start the Oracle WebLogic domain by changing to the `domain1/bin` directory and executing the domain startup command:

```
# /startWeblogic.sh
```

3. Open the GlassFish Web Space Server site.

   Access the portal manually once per instance to make sure the software is working correctly, using `http://<WebLogic-domain>:7001/web/cms/home`. For example, access `http://oraweb10g-44-zone6.sfbay.sun.com:7001/web/cms/home` and sign in using the user/password `admin@example.com/admin`:



## Tuning

The following sections describe tuning that was performed for the performance evaluation described in this paper.

### JVM tuning

The Oracle WebLogic application server uses a shell script, `setDomainEnv.sh`, to set JVM tuning parameters, including the following:

```
<jvm-options>-d64</jvm-options>
      <jvm-options>-server</jvm-options>
      <jvm-options>-Xms4G</jvm-options>
      <jvm-options>-Xmx4G</jvm-options>
      <jvm-options>-Xloggc:/opt/webspace/10u4/gfv2/
glassfish2/domains/domain1/logs/gc.log</jvm-options>
      <jvm-options>-XX:NewSize=768M</jvm-options>
      <jvm-options>-XX:MaxNewSize=768M</jvm-options>
      <jvm-options>-XX:PermSize=256M</jvm-options>
      <jvm-options>-XX:MaxPermSize=256M</jvm-options>
      <jvm-options>-XX:ParallelGCThreads=8</jvm-options>
      <jvm-options>-XX:+UseCompressedOops</jvm-options>
      <jvm-options>-XX:+UseParNewGC</jvm-options>
      <jvm-options>-XX:+UseConcMarkSweepGC</jvm-options>
      <jvm-options>-XX:+CMSClassUnloadingEnabled</jvm-options>
```

These parameters:

- Enable the 64-bit JVM
- Set the maximum and minimum size of the heap to 4 GB
- Allocate 768 MB (of the 4 GB heap) for the new generation, and leave the rest of the heap for the tenured generation
- Allocate 256 MB for the permanent part of the heap, where classes are loaded
- Enable the ParNew and CMS garbage collectors
- Allow the CMS collector to unload classes

The `setDomainEnv.sh` script also includes a section for the memory-related tuning parameters `MEM_ARGS, MEM_PERM_SIZE,` and `MAX_MEM_PERM_SIZE`. On machines with sufficient memory, those variables should be set as follows:

```
MEM_ARGS="-d64 -server -Xms4G -Xmx4G -Xloggc:/var/tmp/gc.log \
-XX:NewSize=1G -XX:MaxNewSize=1G -XX:ParallelGCThreads=8 \
-XX:+UseParNewGC -XX:+UseConcMarkSweepGC \
-XX:+PrintGCDateStamps -XX:+PrintGCDetails \
-XX:+PrintClassHistogram"
export MEM_ARGS

MEM_PERM_SIZE="-XX:PermSize=256M"
export MEM_PERM_SIZE

MEM_MAX_PERM_SIZE="-XX:MaxPermSize=256M"
export MEM_MAX_PERM_SIZE
```

### Oracle WebLogic Tuning

Tuning parameters are stored in the `portal-ext.properties` file. Modifications were made to the default values to point the portal to the Oracle database, configure the size of the connection pool to that database, and also add tuning parameters that are normally used in production deployments.

```
jdbc.default.driverClassName=oracle.jdbc.driver.OracleDriver
jdbc.default.url=jdbc:oracle:thin:@//oraweb10g-40-vip:1521/
  ORASUN.sfbay.sun.com
jdbc.default.username=test04
jdbc.default.password=test04

jdbc.default.maxPoolSize=72
jdbc.default.minPoolSize=16

auto.deploy.enabled=false

# Make the default page to be the content management sample
# community's public page
company.default.home.url=/web/cms/home
default.landing.page.path=/web/ /welcome
default.logout.page.path=/web/cms/home
auth.forward.by.last.path=true
auto.deploy.enabled=false

last.modified.check=false
theme.css.fast.load=true
javascript.fast.load=true
portlet.css.enabled=false
velocity.engine.resource.manager.cache.enabled=true
com.liferay.portal.servlet.filters.layoutcache.LayoutCacheFilter=true
auto.deploy.enabled=false
auto.deploy.interval=0

hibernate.dialect=org.hibernate.dialect.Oracle10gDialect

jcr.initialize.on.startup=true
jcr.jackrabbit.repository.root=/Users/ale/WebSpace/10u4/gfv2/
  var/webspace/data/jackrabbit
jcr.jackrabbit.config.file.path=/Users/ale/WebSpace/10u4/gfv2/
  var/webspace/data/jackrabbit/repository.xml
dl.hook.impl=com.liferay.documentlibrary.util.JCRHook
```

## About the Authors

Pedro Lay is an Enterprise Solutions Architect in Sun's Systems Technical Marketing Group. He has over 20 years of industry experience that spans application development, database and system administration, and performance and tuning efforts. Since joining Sun in 1990, Pedro has worked in various organizations include Technology, the Customer Benchmark Center, the Business Intelligence and Data Warehouse Competency Center, and the Performance Applications Engineering group.

Alejandro Medrano has focused on performance engineering in his 10 years at Sun. In the role of performance architect, Alejandro is responsible for throughput, scalability, and capacity of the Sun Web Space server and the previous Portal Server products. Analyzing and fixing performance issues in these products have led to the resolution of bottlenecks at all layers of Sun's software—from the Virtual Machine for the Java platform (Java Virtual Machine or JVM) to the identity stack to the Directory Sever. The direct results of the combined performance engineering efforts on those fronts have produced a tripling in throughput since the Liferay 4.4 release to today's Oracle GlassFish Web Space server offering.

Srikanth Konjarla is a deployment engineering architect for the Sun Web Space Server product. Srikanth has 17 years of industry expertise, including 10 years of experience in design and architecture of Web and infrastructure products with Sun. During his tenure at Sun, Srikanth has worked with many customers to provide solutions.

## References

**TABLE 2. REFERENCES FOR MORE INFORMATION**

| **WEB SITES** | |
|---|---|
| Sun Blade 6000 Chassis | sun.com/servers/blades/6000chassis/ |
| Sun Blade T6340 server module | sun.com/servers/blades/t6340/ |
| Sun Blade 6000 Virtualized Multi-Fabric 10GbE Network ExpressModule | sun.com/servers/blades/6000iomodule/10gbnem/index.xml |
| Sun Storage 7000 systems | sun.com/storage/disk_systems/unified_storage |
| Oracle Solaris operating system | sun.com/software/solaris/ |
| Oracle GlassFish Web Space Server | sun.com/software/products/webspace/ |
| Oracle Real Application Clusters | oracle.com/database/rac_home.html |
| Oracle WebLogic Server 10*g* Release 3 Documentation Site Map | download.oracle.com/docs/cd/E12840_01/wls/docs103/sitemap.html |

| | |
|---|---|
| **WHITE PAPERS** | |
| Sun Reference Architecture for Oracle 11*g* Grid | sun.com/third-party/global/oracle/collateral/SunRA-Oracle11gGrid.pdf |
| **ARTICLES** | |
| Deploying Oracle Real Application Clusters on Sun Storage 7000 Unified Storage System | sun.com/bigadmin/features/articles/7000_oracle_rac.jsp |

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

Oracle's Sun Systems for Enterprise 2.0
April 2010
Author: Pedro Lay
Contributing Authors: Alejandro Medrano, Srikanth Konjarla

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



**SOFTWARE. HARDWARE. COMPLETE.**