

Load Balancing Oracle Web Applications

An Oracle White Paper
November 2004

Load Balancing Oracle Web Applications

Introduction	3
Load Balancing Implementation	3
Architecture Overview	3
Architecture Redesign	3
Multiple Apache Configuration.....	4
Multiple JServ Configuration.....	5
Cisco CSS SCA Configuration	5
Summary	5
Diagrams.....	6

Load Balancing Oracle Web Applications

INTRODUCTION

Humana was faced with the challenge of supporting 13,000 Oracle Web Application users for Payslip Viewing, Self Service Benefits, Personal Information, iProcurement and iSupplier. This level of user support required redesigning the systems architecture. The final solution included a Cisco CSS and SCA device for load balancing multiple Apache Servers and providing SSL encryption.

LOAD BALANCING IMPLEMENTATION

Oracle Applications can be implemented using various topologies. The flexible architecture allows for load balancing components of the application independently based upon business requirements. This paper provides an outline of how Humana implemented an HTTP Load-Balancer and JServ Layer Load Balancing solution in order to support a large number of Web users.

Architecture Overview

Humana's architecture prior to deployment of the Oracle Web Applications was as follows: One Application Node running Oracle Applications 11.5.9, iAS 1.0.2.2.2 (Apache 1.3.19), Forms, and Concurrent Manager; One Database Node running Oracle Enterprise Edition Database 9.2.0.4 and the Database Listener.

Secure Socket Layer (SSL) Encryption was implemented as supported by the Apache server. Users accessed the application with the following URL: <https://myappserver.domain.com:443> (See Diagram 1 – Architecture Overview) The Web Node consisted of one Apache Server with one JVM (See Diagram 2 Web Node Configuration)

Architecture Redesign

In order to support the large Web user base, a multiple Apache Server configuration was required. This was due to the limitation of the Apache Server's maxclient directive. Furthermore each JServ is limited to supporting approximately 50 concurrent users. In order to reduce the number of Apache Servers required, multiple JVMs were added per Apache Server.

Apache Server and Jserv concurrent user limitations necessitated a system redesign

A Secured Socket Accelerator (SCA) is a hardware device that provides SSL encryption

Adding a SCA device can improve performance by offloading SSL Encryption

Defining a Web Entry Point is a critical step in enabling Oracle Applications to use an HTTP Load Balancer

Multiple Apache Servers can be implemented with a single or multiple Web node configuration

To implement a multiple Apache Server configuration, a method for load balancing the Apache Servers was necessary. A Cisco CSS 11000 Content Switch was implemented as the HTTP Layer Hardware Load Balancer. It was determined that SSL encryption would be handled with hardware instead of software; therefore a Secure Socket Layer Accelerator (SCA) device was also added to the architecture solution. This device migrates the SSL traffic off of the Web node, thus reducing its workload. (See Diagram 3 – Architecture Redesign)

Following the hardware redesign, a series of tests were conducted to answer the following: 1. How to configure the application in order to utilize the new architecture, 2. How to implement multiple Apache Servers on one application server, 3. How to configure the CSS SCA.

Based upon the behavior of the application, it was determined that the instance needed to be referenced by a “virtual host name” or “web entry point”. The web entry point was given a unique IP address and added to the DNS. (See Diagram 4 – Web Entry Point)

Any reference in the application configuration of [server_name]:[port] was modified to the web entry point. All system level profiles, application functions, and the value in ICX_PARAMETERS.HOME_URL were changed accordingly. The following configuration files were also modified to reflect the virtual host name: aplogon.htm, applist.html, jserv.properties, xmlsvcs.properties, formservlet.ini, and disco4iviewer.properties.

Multiple Apache Configuration

In order to setup multiple Apache Servers, \$APACHE_TOP was copied to different locations on the application server. Additionally, the Apache startup script was copied to a new script file. Due to the Apache Servers running on the same server, unique ports were defined for each instance of Apache. All necessary files that needed to reference the new Apache location and unique ports were then changed. The following files required location changes: DB.env, [INSTANCE].env, apachectl, java.sh, apps.conf, httpd.conf, oracle_apache.conf, httpd_pls.conf, jserv.properties, jserv.conf, zone.properties, forms.properties, xmlsvcs.properties, ssp_init.txt viewer4i.properties, disco4iviewer.properties, emailcenter.properties, ojsp.conf, mobile.properties, xml.conf, plsqli.conf, plsqli_pls.conf, formservlet.properties, jservSoap.properties. The following files required port modifications: formservlet.ini, oprocmgr.conf, disco4iv.xml, htmlvars.js, httpd.conf, httpd_pls.conf, jserv.properties, zone.properties, forms.properties, xmlsvcs.properties, viewer4i.properties, oem.conf, plsqli.conf. Location and port modifications are only a requirement if multiple Apache Servers are configured to run on the same Web Node.

Multiple JServs can be configured with AutoConfig or by manually editing the jserv.conf and jserv.properties files

Multiple JServ Configuration

Multiple JServs can be configured with AutoConfig. The AutoConfig Context Editor can be used to input values for the following parameters: ApJServManual, OA Core zone name, OA Core JVM Processes, OA Core Node Weight, and OA Core Servlet Port Range. ApJServManual must be set to “auto” to enable multiple JServs. The number of JServs that are automatically started is set with OA Core JVM Processes. OA Core Node Weight is only used if load balancing JVMs across multiple nodes. The OA Core Servlet Port Range is the range of ports that will be used to start the JVM processes. If AutoConfig is not enabled, then a multiple JVM configuration can be implemented by editing the appropriate parameters in the jserv.conf and jserv.properties files.

Cisco CSS SCA Configuration

To support the SSL encryption, a certificate request was initiated from the SCA device using the virtual host name. The certificate, once received from VeriSign was then loaded on the SCA. The CSS was configured with the host and port services that were then mapped to the virtual host. During the course of testing, it was determined that the load balancing algorithm had to be round robin. In order to sustain a persistent connection, the advanced arrowpoint-cookie setting was required. The sticky bit setting directs the session to the same server for all subsequent HTTP requests. Without the sticky bit set, a user that was logged into the application would lose session information and subsequently be disconnected.

A sticky bit setting is required to maintain a persistent user connection

After completing initial user testing to verify configuration, load testing was performed with LoadRunner. The purpose of the load testing was to validate that user concurrency requirements were met. The stress tests also allowed verification of the Apache Server configuration. It also served to measure the hardware to determine if additional memory or CPU was required for the application or database servers.

SUMMARY

Load Balancing Oracle Applications can be accomplished in many ways. Humana has implemented Apache Server and JServ Balancing. A Web Entry Point was created to access the application via a Cisco CSS HTTP Load Balancer. A SCA was also introduced to the architecture in order to provide SSL encryption and offload the encryption from the Web Node to the hardware. This methodology can be implemented in a multiple or single Web Node environment.

DIAGRAMS

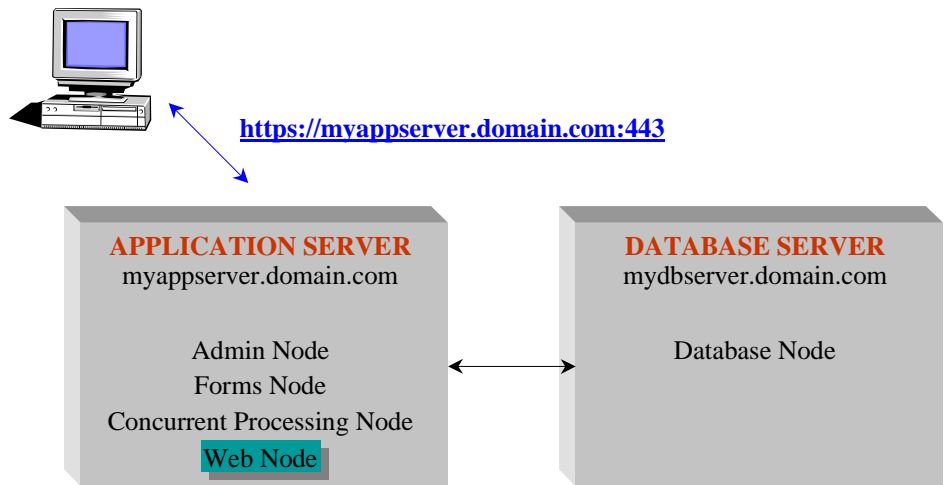


Diagram 1
Architecture Overview

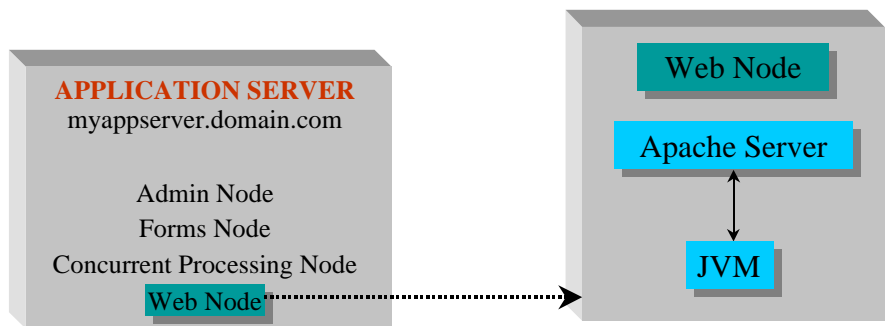


Diagram 2
Web Node Configuration

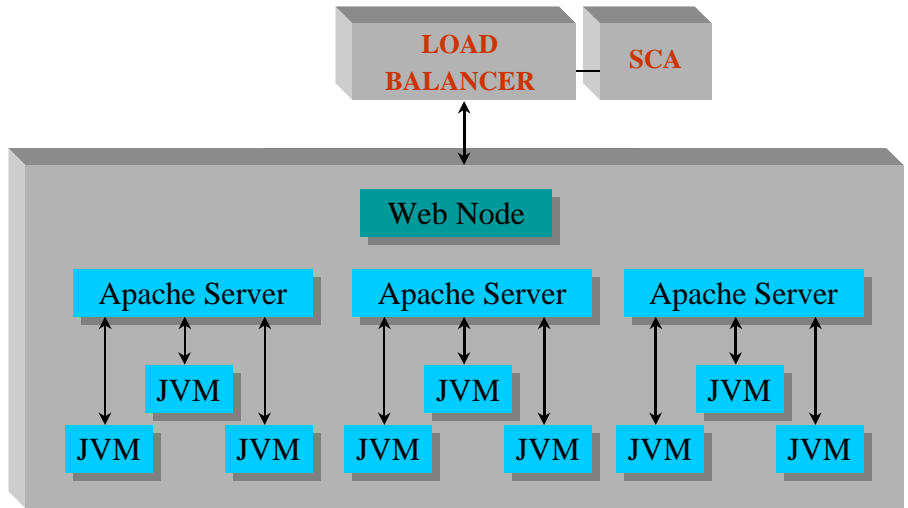


Diagram 3
Architecture Redesign

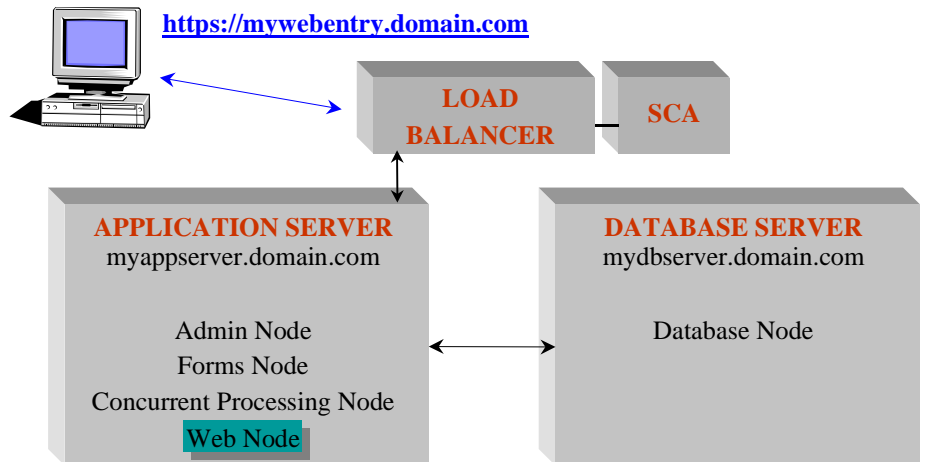


Diagram 4
Web Entry Point



White Paper Title
November 2004
Author: Elke P. Phelps
Contributing Authors: Paul Jackson

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
www.oracle.com

Copyright © 2003, Oracle. 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 is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective owners.