Using Oracle Application Server Web Cache as a Highly Available Load Balancer

An Oracle White Paper
March 2005

ORACLE

Using Oracle Application Server Web Cache as a Highly Available Load Balancer

INTRODUCTION 3

LOAD BALANCING AND HIGH AVAILABILITY 3

ORACLE APPLICATION SERVER AND LOAD BALANCING 4

ORACLEAS WEB CACHE 4

TERMINOLOGY 5

ORACLEAS WEB CACHE LOAD BALANCING 5

DEPLOYMENT CONSIDERATIONS FOR ORACLEAS WEB CACHE LOAD BALANCER 6

HIGH AVAILABILITY FOR ORACLEAS WEB CACHE LOAD BALANCER 7

CONFIGURING HIGHLY AVAILABLE ORACLEAS WEB CACHE LOAD BALANCER 8

CONCLUSION 11

Using Oracle Application Server Web Cache as a Highly Available Load Balancer

INTRODUCTION

HTTP load balancers are critical to application server deployments that need high availability and good resource utilization. There are many HTTP load-balancing solutions available in the market today, the cost of which is forcing many IT organizations to look for alternatives.

Oracle Application Server Web Cache is a powerful, state-of-the-art server acceleration solution shipped with Oracle Application Server. OracleAS Web Cache is a cache engine that also supports load balancing of HTTP and HTTPS traffic. You can deploy OracleAS Web Cache as a front-end HTTP load balancer for some specific configurations of Oracle Application Server.

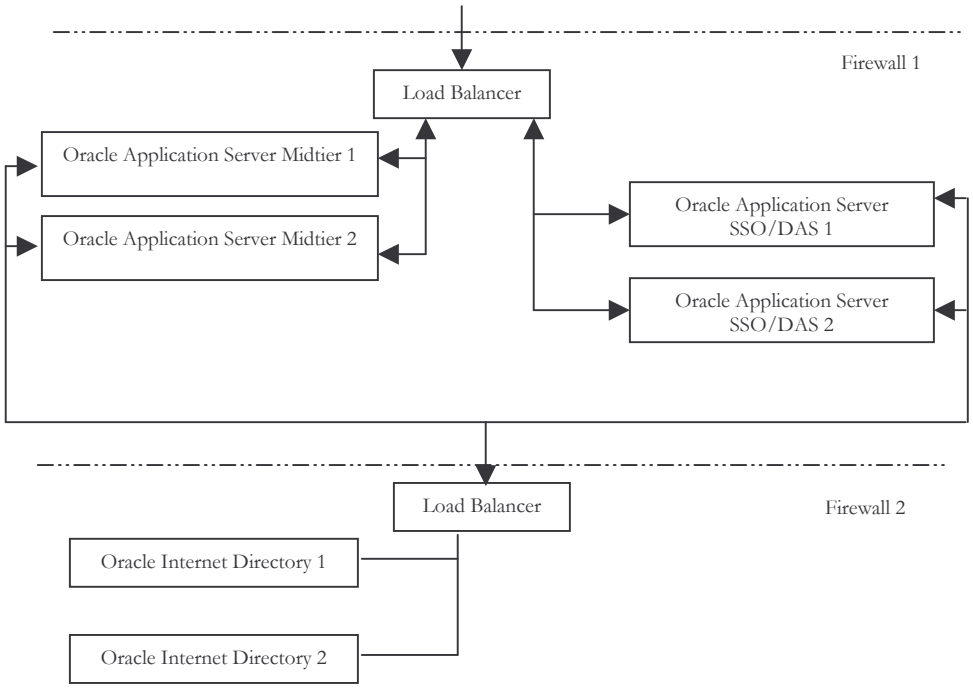
This document describes how to configure OracleAS Web Cache as a highly available load balancer. It describes the relationship between high availability and load balancing and its impact on Oracle Application Server. It also provides detailed steps for configuring high availability for OracleAS Web Cache functioning as a load balancer.

LOAD BALANCING AND HIGH AVAILABILITY

Load balancers are typically devices placed at the edge of the network. All the client requests are first received by the load balancer, which then distributes them among participating backend servers. Most load balancers monitor the health of the backend servers to ensure continuous availability. Availability of the load balancer itself is ensured by deploying a standby load balancer.

ORACLE APPLICATION SERVER AND LOAD BALANCING

Load balancers are used at different tiers of the Oracle Application Server stack to distribute load and to provide high availability. Oracle Application Server installations typically employ load balancers to balance HTTP traffic to Oracle midtiers, Oracle Application Server Single Sign-On servers and LDAP traffic to Oracle Internet Directories.



Example of Oracle Application Server Deployment with a Load Balancer
Fig 1

ORACLEAS WEB CACHE

OracleAS Web Cache, part of the Oracle Application Server product suite, is primarily a cache engine that also supports load balancing of HTTP and HTTPS traffic. Deployed before a farm of application servers or globally at the network edge, OracleAS Web Cache provides load balancing, failover, clustering and surge protection features for application servers.

Terminology

This document uses the following terms:

cache

Refers to OracleAS Web Cache being used for caching, with or without load balancing.

OracleAS Web Cache load balancer

Refers to OracleAS Web Cache being used only for load balancing without any caching.

Surrogate-Control response header

Refers to OracleAS web Cache specific response header that enables application developers to specify caching attributes of an object. This response-header field enables the application Web server to override the caching rules configured through the OracleAS Web Cache Manager interface.

VIP

Refers to virtual IP address that clients use to connect.

active/primary

Implies the application process is running and is processing client requests.

passive/standby

Implies the application process is not running.

clusterware

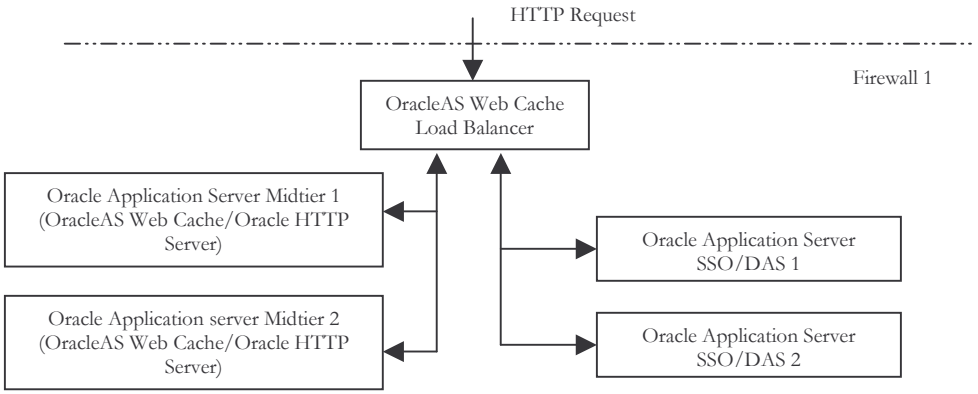
Refers to clustering software provided by operating system vendors and associated hardware.

ORACLEAS WEB CACHE LOAD BALANCING

OracleAS Web Cache supports following load balancing features.

- Stateful and stateless load balancing of HTTP and HTTPS traffic
- Weighted round-robin algorithm for load balancing.
- Cache servers or Oracle HTTP servers as backend origin servers

For more information on stateful and stateless load balancing, see Chapter 1, "Introduction to OracleAS Web Cache," in the *Oracle Application Server Web Cache Administrator's Guide* 10*g* (9.0.4).



Load Balancing Oracle Application Server Using OracleAS Web Cache Load balancer
Fig 2

Deployment Considerations for OracleAS Web Cache Load Balancer

When considering OracleAS Web Cache as the load balancer, consider the following factors:

- OracleAS Web Cache can load balance only HTTP and HTTPS traffic.
- OracleAS Web Cache does not actively monitor the health of the next hop origin server node and will not be able to quickly detect ungraceful node failures, such as those due to power failure, system crash, or NIC failure. However, OracleAS Web Cache can quickly detect graceful failures of origin servers, such as an origin server process exiting or crashing while the system is available with network connectivity.

  NOTE: *OracleAS Web Cache does allow for setting of send and receive timeouts. You can also configure the number of failures after which a node will be considered as down. In typical usage, you set the receive timeout to a high value (default: 3600 seconds) to accommodate for slow running applications in the upstream servers. Transport protocol timers might also aid in the quick detection of origin server unavailability. TCP has timers for connect (default: 75 seconds), write (default: 9 minutes) and keep alive (default: 2 hours).*

- You should not allow `Surrogate-Control` response headers directing cacheability to be passed through an OracleAS Web Cache load balancer. If `Surrogate-Control` response headers are used by any application, then do not use the OracleAS Web Cache load balancer to load balance traffic to that application.

  NOTE: *This restriction applies only to the Oracle AS Web Cache load balancer and does not apply to the cache. Surrogate-Control headers with control directive "no-store"*

can be passed to the Oracle AS Web Cache load balancer. The Oracle HTTP server module "mod-osso" uses Surrogate-Control headers with control directive "no-store".

- OracleAS Web Cache load balancer should not be used to load balance traffic to Oracle Application Server Portal.
- OracleAS Web Cache load balancer is not highly available by default. However, the OracleAS Web Cache load balancer can be made highly available. See the next section.

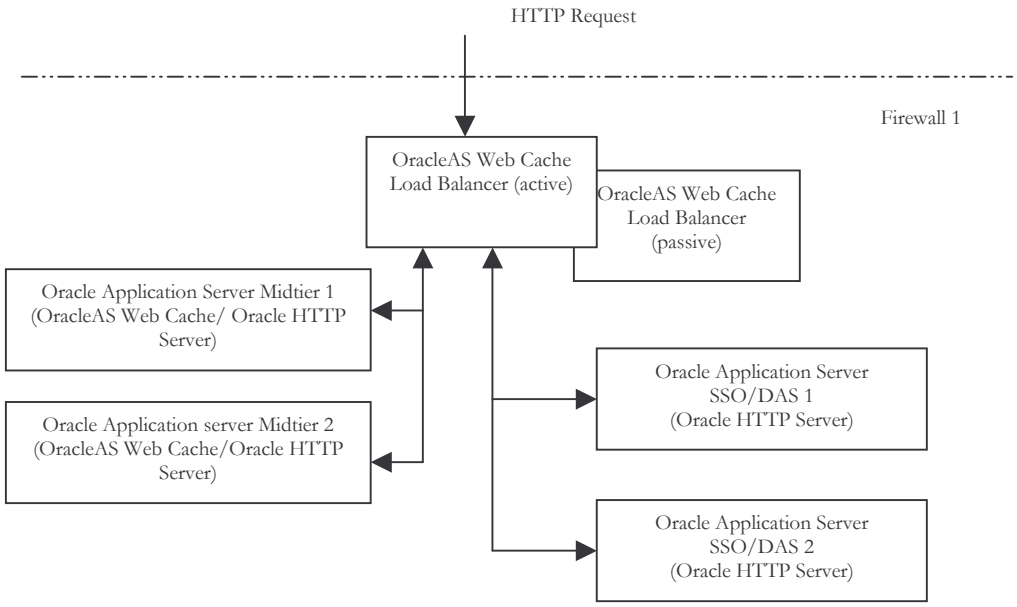
HIGH AVAILABILITY FOR ORACLEAS WEB CACHE LOAD BALANCER

OracleAS Web Cache load balancer is not highly available by default. You can make OracleAS Web Cache load balancer highly available by using a standby node in conjunction with mechanisms for failure detection and failover.

In this configuration, you deploy two OracleAS Web Cache load balancer nodes, one active node and one passive node. When an active OracleAS Web Cache load balancer is detected to be down, then VIP is migrated to the passive node and the passive OracleAS Web Cache load balancer is started up.

This document does not make any assumptions about mechanisms used for failure detection or for failover; but provides instructions for configuring the OracleAS Web Cache load balancer for a successful failover. The specific method used for detection of node death, migration of the VIP and starting up of the passive OracleAS Web Cache load balancer is left to the user.

The most commonly used method in the industry for failure detection and failover is the usage of a clusterware. In this approach, clusterware monitors the health of the active node, migrates the VIP, and starts up the application on the passive node as necessary.



Load Balancing Oracle Application Server Using Highly Available OracleAS Web Cache Load Balancer
Fig 3

Configuring Highly Available OracleAS Web Cache Load Balancer

1. Configure OracleAS Web Cache as a load balancer:
    a. Install OracleAS Web Cache on both primary and standby nodes. If a clusterware is being used, then on UNIX platforms you can install OracleAS Web Cache on the shared disk in which only a single install is needed.
    b. Make sure OracleAS Web Cache is listening on the VIP by connecting to OracleAS Web Cache administration port.
    c. Restrict traffic to OracleAS Web Cache invalidation, administration, and statistics ports from outside the firewall.
    d. Add backend servers, among which load has to be distributed, to the origin servers list. See Chapter 7, "Basic Setup and Configuration," in the *Oracle Application Server Web Cache Administrator's Guide* 10*g* (9.0.4).

e. Enable pinging of origin servers and specify a suitable ping interval.

Note: OracleAS Web Cache will start pinging origin servers only after it has detected the origin server to be down. This pinging enables the OracleAS Web Cache load balancer to automatically detect when origin servers come back up.

f. Create site definitions, and map site definitions to origin servers. See Chapter 7, "Basic Setup and Configuration," in the *Oracle Application Server Web Cache Administrator's Guide* 10*g* (9.0.4).

g. Turn off caching in OracleAS Web Cache by changing **Site Specific** and **For All Sites** rules to **Don't Cache** for all content for all the operations. Add the rule `.*` with a caching policy of **Don't Cache** to both **Site Specific** and **For All Sites** rules. See Chapter 9, "Creating Caching Rules," in the *Oracle Application Server Web Cache Administrator's Guide* 10*g* (9.0.4) for more details on configuring caching rules.

2. Configure Oracle Application Server to be a front-ended by the OracleAS Web Cache load balancer:

   a. Enable session binding if session stickiness is required. If the application that requires session stickiness is an OC4J application, then use **JSESSIONID** cookie. If all of the backend Oracle HTTP Server and OC4J are cross-registered then you may not have to enable session binding for the OracleAS Web Cache load balancer. See Chapter 7, "Basic Setup and Configuration," in the *Oracle Application Server Web Cache Administrator's Guide* 10*g* (9.0.4) for more details on OracleAS Web Cache session binding. For example, OracleAS Single Sign-On and Oracle Delegated Administration Services require turning on session binding (stateful load balancing).

   b. Partner and provider applications that use OracleAS Single Sign-On may need to be reregistered. See Chapter 4, "Configuring and Administering Partner Applications" and Chapter 9 "Advanced Configurations" in the *Oracle Application Server Single Sign-On Administrator's Guide* 10*g* (9.0.4) for more information on reregistering applications with OracleAS Single Sign-On.

   c. Make sure the HTTP `ServerName` and `Port` directives in `httpd.conf and ssl.conf` specify the correct URL and port. Typically, `ServerName` and `Port` specify the site name and port of the load balancer.

3. Configure high availability for the OracleAS Web Cache load balancer

    a. If the OracleAS Web Cache load balancer is installed on a shared disk, then create a two-node OracleAS Web Cache cluster using OracleAS Web Cache Manager. Both of the entries in the cache cluster members will be identical except for the host name.

    b. If the OracleAS Web Cache load balancer is not installed on a shared disk and if there are separate installs for active and passive nodes, then make sure that these separate installs have identical configuration files (`webcache.xml`) that differ only on the host name. The host name is specified in the `HOSTNAME` attribute of the `CACHE` element in `webcache.xml`.

    c. Both of these cache entries (`CACHE` element in `webcache.xml`) should always be kept in sync except for the host name. Any changes made to one node should be made to the other node. In the case of a shared disk install, the OracleAS Web Cache `admin` server depicts the configuration as a cache cluster where one of the nodes in the cache cluster is down.

    d. Set up mechanisms for failure detection and failover of the OracleAS Web Cache load balancer. Typically, this step involves continuous monitoring of the active OracleAS Web Cache load balancer, failover of the VIP, and starting up of the OracleAS Web Cache load balancer on the standby node when the active load balancer is detected to be down. If using clusterware, then create cluster agent scripts for monitoring the OracleAS Web Cache load balancer, migrating the VIP, and for starting up the OracleAS Web Cache load balancer on the standby node.

CONCLUSION

Most of the load balancing solutions available in the market today are expensive and offer additional functionalities such as intrusion detection and firewall support. OracleAS Web Cache can be used as an inexpensive alternative when only HTTP load balancing is required.

OracleAS Web Cache, part of Oracle Application Server suite, is a cache engine that also supports load balancing of HTTP and HTTPS traffic. Oracle Web Cache can be used as a load balancer to balance HTTP and HTTPS traffic to some configurations of Oracle Application Server. OracleAS Web Cache can be made highly available by using redundant deployment.

Using Oracle Application Server Web Cache as a Highly Available Load Balancer
March 2005
Author: Laljo John Pullokkaran , Oracle Application Server High Availability Group, Oracle Corporation.

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