Oracle® HTTP Server

Release Notes

Release 2 (9.0.2) for UNIX

April 2002

Part No. A90332-01

This document summarizes the differences between Oracle HTTP Server and its documented functionality.

See Also: *Oracle9i Application Server Release Notes*

1 General Issues and Workarounds

This section describes general issues and their workarounds for Oracle HTTP Server.

1.1 Enabling JServ in Oracle9*i*AS Release 2 (9.0.2)

Oracle Corporation recommends that you use the Oracle9*i*AS Containers for J2EE (OC4J) for your servlet environment; it is the default configuration for Oracle9*i*AS Release 2 (9.0.2). However, you may want or need to use JServ in your Oracle9*i*AS Release 2 installation. These instructions are provided to explain how to enable JServ, and, if necessary, use it for some applications and OC4J for others. The instructions assume a working familiarity with the Oracle HTTP Server and JServ configuration.

This section has the following topics:

- Directives Reference: This section describes directives used to enable JServ with `mod_oprocmgr`.
- Enabling JServ with mod_oprocmgr:This section explains how to enable the Oracle default mode for JServ. Use this mode if you want process management and load balancing capabilities for multiple JVMs. The `ApJServManual` directive has a new mode, 'auto', that enables using JServ with the Oracle module `mod_oprocmgr`. The *`ORACLE_HOME`*`/Apache/JServ/etc/jserv.conf` file contains LoadModule directives for `mod_jserv` and `mod_oprocmgr`.

- Enabling JServ in Automatic Mode: This section explains how to enable JServ in automatic mode. Use this mode if you need only one JVM. In this mode, the `ApJServManual` directive is set to 'off' and the `mod_jserv` module launches and monitors the JVM. If the Oracle HTTP Server is restarted or stopped, `mod_jserv` restarts or stops the JVM.
- Enabling JServ in Manual Mode: This section explains how to enable JServ in manual mode. Use this mode if you need to run multiple JVMs. In this mode, the `ApJServManual` directive is set to 'on' and you have to stop and start the JVM manually. To monitor the JVM, you must use an external monitoring facility.
- Using JServ and OC4J Together: This section explains how to use `mod_rewrite` to enable some applications to execute on JServ, and others on OC4J.

1.1.1 Directives Reference

This section describes some JServ configuration directives that are related to using JServ with `mod_oprocmgr`. All directives are described thoroughly in the configuration files.

See Also: *Oracle HTTP Server Administration Guide*

1.1.1.1 Directives in jserv.properties

This section describes the following directive:

`Port`

Use this directive to specify the ports to which JServ will bind, as shown in the example below.

```
port=8007
```

If no ports are specified, the JServ processes will choose their ports. The `port` directive as shown below enables the JServ processes to choose their own ports (the default).

```
port=
```

If you eliminate the `port` directive entirely, an error will occur.

You can specify multiple ports, and separate the values with commas as shown in the example below. Note that a range of ports (9000-9010) is a valid value.

```
port=8007,9000-9010,8010
```

1.1.2 Directives in jserv.conf

This section describes the following directives:

ApJServManual

This directive accepts a new mode, auto, which invokes the new infrastructure functionality (in which mod_oprocmgr manages processes). The syntax is:

```
ApJServManual auto
```

You can set the mode to on or off to use standard JServ functionality.

ApJServGroup

This directive defines groups for the process manager to manage for mod_jserv. If you have worked with mod_jserv, you will note that this directive replaces the ApJServBalance, ApJServHost, ApJServRoute and ApJServShmFile directives.

All JServ processes to be managed must belong to a group, and each group has its own ApJServGroup directive. If you only have one JServ process, you must define a group with just that process in it. The processes in a group are identical except for their listening ports, so requests directed to the group are distributed evenly among the processes.

The ApJServGroup directive takes four arguments: groupname, number of processes, node weight, and properties file. In the example below, the groupname is mygroup, the number of processes is 1, the node weight is 1, and the full path of the properties file used to start the JServ processes is

ORACLE_HOME/Apache/JServ/etc/jserv.properties

```
ApJServGroup mygroup 1 1
/private2/up_1022/Apache/Jserv/etc/jserv.properties
```

ApJServGroupMount

This directive defines a mount point and maps it to a process group and zone. In the example below, the mount point is /servlets, the group is mygroup, and the zone is root. Note that the balance protocol is in use for routing, as in the standard JServ configuration.

```
ApJServGroupMount /servlets balance://mygroup/root
```

Place this directive after the ApJServGroup directive in the configuration file.

ApJServGroupSecretKey

This directive specifies the secret key that JServ needs to authenticate clients. It can be disabled, as shown below:

```
ApJServGroupSecretKey disabled
```

When activated, the directive takes one or two arguments. In the example below, with group and filename arguments, the filename `mysecretkey` applies to the group `mygroup`.

```
ApJServGroupSecretKey mygroup /usr/local/apache/jserv/mysecretkey
```

You can supply only the filename argument, as shown below. No group is named, so the secret key filename applies to all groups.

```
ApJServGroupSecretKey /usr/local/apache/jserv/mysecretkey
```

You cannot combine directives using the one-argument syntax with directives using the two-argument syntax. If you use the two-argument syntax, the default for groups without a group-specific secret key is 'disabled'.

Place this directive after the `ApJServGroup` directive in the configuration file.

> **Warning: The secret in the secret key file specified in `ApJServSecretKey` must be the same as that specified by the `security.secretKey` directive in the jserv.properties file. If the secrets are not the same, the death detection mechanism assumes that all the servlet engine processes are dead, eliminates them, and starts new processes to replace them (repeating the cycle endlessly).**

1.1.3 Enabling JServ with mod_oprocmgr

This section explains how to implement process management and load balancing services for JServ processes with `mod_oprocmgr`. Terms used in this section to describe the module and its functions are defined below:

- **`mod_oprocmgr`**: A module that starts, stops, and detects death of processes (starting new processes to replace them), and provides load balancing services to the processes. `mod_oprocmgr` gets the topology management information via HTTP requests from internal servers such as JServ, and does its job based on this information.
- **`group`**: A set of processes across which request traffic is distributed.

- **servlet engine process**: A JVM instance that runs a servlet engine, such as JServ.

1.1.3.1 How mod_oprocmgr Works With mod_jserv

`mod_oprocmgr` provides infrastructure capabilities, such as automatic starting of processes, death detection and restart, and load balancing. These capabilities are enabled by a new mode, auto, for the `ApJServManual` directive.

Based on the configuration information provided by `mod_jserv`, `mod_oprocmgr` starts the specified number of JServ processes, managing them for the life of the servers.

Follow these steps to enable JServ with `mod_oprocmgr`:

1. Uncomment the Include directive for the `jserv.conf` file in *ORACLE_HOME*`/Apache/Apache/conf/httpd.conf`.

   ```
   #include "/ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

2. Configure directives in the *ORACLE_HOME*`/Apache/Jserv/etc/jserv.conf` file, if needed.
3. Configure directives in the *ORACLE_HOME*`/Apache/Jserv/etc/jserv.properties` file, if needed.
4. Configure directives in the *ORACLE_HOME*`/Apache/Jserv/etc/zone.properties` file, if needed.
5. Restart the Oracle HTTP Server.

1.1.4 Enabling JServ in Automatic Mode

Follow these steps to enable JServ in automatic mode:

1. Uncomment the Include directive for the `jserv.conf` file in *ORACLE_HOME*`/Apache/Apache/conf/httpd.conf`.

   ```
   #include "/ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

2. Configure the `ApJServManual` directive in the *ORACLE_HOME*`/Apache/Jserv/etc/jserv.conf` file:

   ```
   ApJServManual off
   ```

3. Configure other directives as needed in `jserv.conf`.
4. Set the `port` directive in the *ORACLE_HOME*`/Apache/Jserv/etc/jserv.properties` file to the same value as that specified in the `ApJServDefaultPort` directive.

5. Configure directives in the *ORACLE_HOME*/Apache/Jserv/etc/zone.properties file, if needed.
6. Restart the Oracle HTTP Server.

1.1.5 Enabling JServ in Manual Mode

Follow these steps to enable JServ in manual mode:

1. Uncomment the Include directive for the jserv.conf file in *ORACLE_HOME*/Apache/Apache/conf/httpd.conf.

   ```
   #include "/ORACLE_HOME/Apache/Jserv/etc/jserv.conf"
   ```

2. Configure the ApJServManual directive in the *ORACLE_HOME*/Apache/Jserv/etc/jserv.conf file:

   ```
   ApJServManual on
   ```

3. Configure other directives as needed in jserv.conf.
4. Configure directives in the *ORACLE_HOME*/Apache/Jserv/etc/jserv.properties file, if needed.
5. Configure directives in the *ORACLE_HOME*/Apache/Jserv/etc/zone.properties file, if needed.
6. Before or while starting the JVM, set the arguments passed to the Java interpreter, and the classpath passed to the JVM (as specified by wrapper.bin.parameters and wrapper.classpath in the jserv.properties file).

   An example of the command to do this is shown below:

   ```
   java -classpath ORACLE_HOME/lib/servlet.jar:$CLASSPATH
   org/apache/jserv/JServ </path/to>/jserv.properties
   ```

7. Restart the Oracle HTTP Server.

1.1.6 Using JServ and OC4J Together

Perform the following configuration steps to enable JServ and Oracle9*i*AS Containers for J2EE (OC4J) to coexist. This is important if you have the Portal and Wireless installation type, because of the Portal dependency on OC4J.

1. Specify the engine on which applications should execute. Suppose you have these URLs:

   `/application1/file1.jsp` to execute on JServ, and

   `/application2/file2.jsp` to execute on OC4J.

   You must rewrite the URL for application1.

   a. Edit *ORACLE_HOME*`/Apache/Apache/conf/httpd.conf` and ensure that the following directives are active (uncommented) and present:

      ```
      LoadModule rewrite_module libexec/mod_rewrite.so
      AddModule mod_rewrite.c
      RewriteEngine on
      ```

   b. Edit *ORACLE_HOME*`/Apache/jsp/conf/ojsp.conf` to add these directives:

      ```
      RewriteRule /application1/(.*)/(.*)\.jsp$
      /application1/$1/$2.jsp1
      ApJServAction .jsp1 /servlets/oracle.jsp.JspServlet
      ```

   c. Remove this directive:

      ```
      ApJServAction .jsp /servlets/oracle.jsp.JspServlet
      ```

   d. Edit *ORACLE_HOME*`/Apache/Jserv/etc/jserv.conf` and mount `/servlets` to the JVM that will service the JSP requests. Use the `ApJServMount` or `ApJServGroupMount` directive (depending on how the JServ processes are started).

2. Configure JServ using the Enterprise Manger Web site:

   a. Navigate to the Instance Home Page on the Enterprise Manager Web site. Scroll to the Administration section.

   b. Select **Configure Components**. This opens the Configure Components Page.

   c. Choose JServ in the Component menu, enter the `ias_admin` password, and click **OK**.

3. Restart Oracle HTTP Server.

1.2 Oracle HTTP Server Virtual Hosts Metrics Distinguished by Hostname Only

The performance metrics collected for virtual hosts for Oracle HTTP Server are categorized by virtual host name only. So if an Oracle9*i*AS site deploys multiple virtual hosts that differ only in their listening port or IP address, then their performance metrics, as shown by the Enterprise Manager, will be displayed summed together.

1.3 Available Header Size Is Significantly Less for Cookie and/or Header Data

The amount of space available for cookie data and / or header data has greatly decreased from Oracle HTTP Server / Jserv to Oracle HTTP Server / OC4J. Oracle HTTP Server / Jserv can approach 8K in size for cookie data while for Oracle HTTP Server / OC4J, this is limited to about 4K in size.

1.4 FastCGI, CGI, CGI-Bin/Test-CGI and mod_perl Demos Should be Disabled

`FastCGI`, `CGI`, `mod_perl`, and `CGI-bin` / `test-CGI` demo scripts should be disabled from production systems as they might serve a good source of information for hackers. The demo scripts are shipped with the distribution in order for users to verify that the installation is successful. Note that there are no known security problems, such as buffer overflow, with these demo programs. The information leakage can be addressed by proper configuration.

1.5 Accessing mod_osso Protected Pages from Netscape 4.7 Requires Manual Configuration

You may not be able to access `mod_osso` protected pages from Netscape 4.7. If you want to access `mod_osso` protected pages from Netscape 4.7, then the partner application corresponding to `mod_osso` should be modified from the Single Sign-On server configuration console to point to Oracle9*i*AS Web Cache port number, which is usually 7777. For details on how to use the Single Sign-On console, refer to the appropriate Single Sign-On administration documentation on the documentation CD.

2 Configuration Issues and Workarounds

This section describes configuration issues and their workarounds for Oracle HTTP Server.

2.1 External Application Basic Authentication SSO API Error

The parameter `nonssl_sso_port=<port>` and / or `nonssl_sso_host=<alternative sso server name>` need to be specified in the Oracle HTTP Server's `mod_osso` configuration file `conf/osso/osso.conf` when a SSO Server is operated in SSL mode. In particular, parameter `nonssl_sso_port=<port>` must be specified in the `osso.conf` file in order to facilitate internal communications between an Oracle HTTP Server and a SSL enabled SSO server. By default, the `nonssl_sso_port` is initiated automatically by the `SSORegistrar` tool at Oracle9*i* Application Server install time, and its default value is 5000.

There are other configuration steps need to be completed on the SSO apache server.

Add the following line in the `httpd.conf` file to expose Basic Authentication URL to the `mod_osso`.

```
Listen <port number>
Example: Listen 5000

Note: If you have used the following configuration in the Apache
configuration in SSO Server URLs then you need to turn it off. (Details
in the Oracle9iAS Single Sign-on Administrator's Guide, Chapter 2,
Enabling the Single Sign-On Server for SSL)

<IfDefine SSL>
  <Location /pls/orasso>
     SSLRequireSSL
  </Location>
</IfDefine>
Instead of this, use following configuration to protect of SSO Server
URLs:

# SSO Server Login URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_admin.ls_login>
    SSLRequireSSL
  </Location>
</IfDefine>
```

```
# Change password URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_user.mgr.change_password>
    SSLRequireSSL
  </Location>
</IfDefine>

# External Application Login URL
<IfDefine SSL>
  <Location /pls/orasso/orasso.wwsso_app_user.mgr.change_password>
    SSLRequireSSL
  </Location>
</IfDefine>
```

2.2 Use UpdateConfig For Manual Edits to OC4J, Oracle HTTP Server and OPMN Configuration Files

DCM command line utility must be used to notify the DCM repository of all manual edits to the Oracle HTTP Server, OC4J or OPMN configuration files. Failure to run this tool could result in loss of configuration data.

The valid forms of this command are as follows:

- `dcmctl updateConfig`: This copies the configuration on disk to the DCM Repository for Oracle HTTP Server, OC4J or OPMN.
- `dcmctl updateConfig oc4j`: This copies the configuration files for OC4J to the DCM Repository.
- `dcmctl updateConfig ohs`: This copies the configuration files for Oracle HTTP Server to the DCM Repository.
- `dcmctl updateConfig opmn`: This copies the configuration files for OPMN to the DCM Repository.

2.3 OPMN/Oracle HTTP Server Infrastructure Requires Special Setting for a Secure Web site

When using OPMN / Oracle HTTP Server infrastructure, the user has to specify at least one `non_ssl` port. For a purely secure Web site, meaning it only accepts SSL connection, the user has to provide an extra `non_ssl` port in `httpd.conf`. The user can do so by adding the following to maintain a secure Web site:

```
Listen <port>

<VirtualHost _default_:port>
SSLEngine Off
<Location />
Order deny,allow
Deny from all
Allow from local host
Allow from <ip1 of a localhost>
Allow from <ip2 of a localhost>
Allow from <ip3 of a localhost>
</Location>
</VirtualHost>
```

This way security is maintained by restricting the non-ssl port to only accept traffic from local host.

2.4 ORACLE_HOME Path Limit for FastCGI

FastCGI will fail if the `ORACLE_HOME` path length is longer than 92 characters. This is due to the limitation of UNIX domain socket pathname length. As a workaround, you may override the default path by using "`FastCgiIpcDir<dir>`" directive in `ORACLE_HOME/Apache/Apache/conf/httpd.conf` within the `<IfModule mod_fastcgi.c>` block.

For example:

```
<IfModule mod_fastcgi.c>
FastcgiIpcDir "/tmp/fcgi"
...
</IfModule>
```

2.5 Proxy-plugin for *i*Planet Creates Mapping Error

When *i*Planet is initially installed, it automatically maps all URL beginning with "/servlet" to its own servlet handler resulting usually in a "Not Found" error returned to client if /servlet request intended to be handled by the proxy plugin is handled by *i*Planet.

If you wish to map /servlet to the proxy plugin, all references to servlet need to be commented out in the listener's `obj.conf` file. This file is located in the `config` directory. No change are needed to `magnus.conf`.

The following should be commented out:

```
NameTrans fn="NSServletNameTrans" name="servlet"
NameTrans fn="pfx2dir" from="/servlet"
dir="c:/iplanet/Servers/docs/servlet"
name="ServletByExt"

<Object name="servlet">
ObjectType fn=force-type type=text/html
Service fn="NSServletService"
</Object>

<Object name="ServletByExt">
ObjectType fn=force-type type=magnus-internal/servlet
Service fn="magnus-internal/servlet" fn="NSServletService"
</Object>
```

2.6 Single Sign-On Initially Gets 503 Errors When Attempting to Access Protected Page

When attempting to access a protected resource, user is redirected to the SSO Server, user receives 503 errors initially. To avoid this, when using a server load balancer, the `KeepAlive` directive should be disabled.

3 Administration Issues and Workarounds

This section describes administration issues and their workarounds for Oracle HTTP Server.

3.1 *i*ASOBF and SSO Wallet Support Is User-dependent

Both *i*ASOBF, a tool used to obfuscate Wallet password and/or OSSO configuration file, and SSO Wallet-support use Oracle obfuscation algorithm which requires the same user to obfuscate and deobfuscate it. For `mod_ossl`, typically the Oracle9*i*AS administrator does the obfuscation and Apache runtime does the deobfuscation. However, Apache is typically run as root (this is true if `root.sh` is executed at install time) and therefore will not start. It is important to make sure that obfuscation is also run as `root`, or as the user that Apache runtime is running under.

3.2 ORASSO Schema Password Change Using OEM Requires Oracle HTTP Server to be Restarted

If you use Oracle Enterprise Manager to change the `orasso` schema password, the portal SMI plugin that is invoked by Oracle Enterprise Manager does not automatically restart Oracle HTTP Server. This will cause access to Single Sign-On Web pages to fail. The administrator requires to manually restart Oracle HTTP Server to pick up the new `orasso` password.

4 Documentation Errata

This section describes known errors in the documentation.

4.1 Remove SOAP Demo from Oracle HTTP Server Administration Guide

The Oracle HTTP Server Administration Guide contains information about a SOAP demo located on the Oracle HTTP Server main page. This demo is no longer a part of the Oracle HTTP Server main page and this information should be removed from the Oracle HTTP Server Administration Guide.

4.2 Incorrect Path Name for Configuring *i*Planet Listener for Single Sign-on

Step 6 of the "Configuring the *i*Planet Listener for Single Sign-on" section in the Oracle HTTP Server Administration Guide provides incorrect information. It states the following:

Add the following line:

```
<object path="/path/login_success">
Service fn="osso_success_service"
</object>
```

The correct line that should be added is:

```
<object ppath="/path/osso_login_success">
Service fn="osso_success_service"
</object>
```