Oracle9*i*AS Single Sign-On

Release Notes

Release 2 (9.0.2)

April 2002

Part No. A96641-01

This document addresses topics not covered in *Oracle9iAS Single Sign-On Administrator's Guide* and *Oracle9iAS Single Sign-On Application Developer's Guide*.

The document covers the following topics:

- Configuring the Oracle HTTP Server for Legacy Applications
- Reregistering the Oracle HTTP Server with the Single Sign-On Server
- Configuring Globalization Support: Errata and Additions
- Using POST Authentication with Netegrity SiteMinder
- Default Password Policies
- Issues with Protecting Single Sign-On URLs with SSL
- Protecting Single Sign-On URLs with SSL: Erratum
- Security Issues: Single Sign-Off and Application Logout
- Unlocking Single Sign-On Users
- Issues with Oracle Internet Directory Outages
- Locating the Single Sign-On Software Development Kit
- A Note About Sample Programs for Java APIs
- Browser History and the Customized Login Page
- Using a Proxy Server with Oracle Single Sign-On
- Browser Settings for Oracle Single Sign-On
- Setting Up ACLs for Certificate-Enabled Single Sign-On

- PUBLIC User Account Must Not Be Deleted
- Bug: Enabling Oracle9iAS Web Cache

1 Configuring the Oracle HTTP Server for Legacy Applications

The standard way to access legacy[1], or external, applications enabled by single sign-on is through the External Applications portlet of Oracle9*i*AS Portal, an SDK-enabled partner application. Applications accessed in this way can be configured for GET, POST, or Basic authentication.

An alternative method is to use the Oracle HTTP server as a secure proxy for applications that reside on a separate Web server. This method involves configuring the modules mod_osso and mod_proxy to support single-sign-on-enabled Basic authentication. The advantage of the proxy approach is that it eliminates the screen flicker that occurs when external applications are accessed in the standard way.

This section contains the following topics:

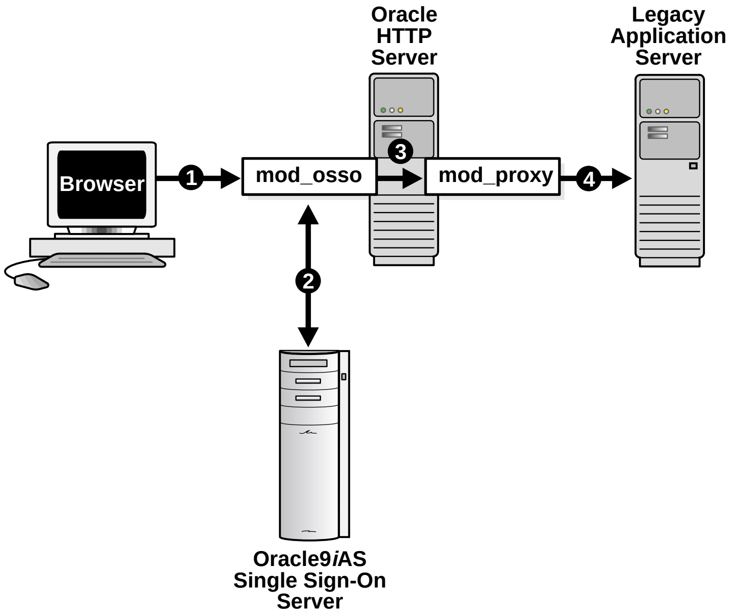
- Authenticating to Legacy Applications Using Mod_osso / Mod_proxy
- Configuration Requirements
- Configuration Steps

1.1 Authenticating to Legacy Applications Using Mod_osso/Mod_proxy

If the Oracle HTTP server is configured correctly, authentication to mod_osso-enabled external applications is similar to what it is for partner applications: mod_osso intercepts a URL request and redirects it to the single sign-on server. Unlike mod_osso-enabled partner applications, mod_osso-enabled external applications must reside on a third server, and they require Basic authentication.

The figure that follows illustrates the process.

[1] Legacy applications are older applications that cannot be modified to delegate authentication to the single sign-on server.



1. The single sign-on user requests an external application by selecting a bookmark or by entering a virtual URL. This URL enables the Oracle HTTP server to intercept the request.
2. Mod_osso adds an authentication header to the intercepted request and retrieves the user's credentials from the Oracle9*i*AS database.
3. Mod_osso sets the header value with the user's credentials, retrieved from the single sign-on server. Mod_osso then passes this header to mod_proxy.
4. Mod_proxy passes the user's credentials—in the form of a Basic authentication header—to the real URL. Mod_proxy does this by using directives that map the virtual URL to the real URL.

1.2 Configuration Requirements

The following criteria must be met before the Oracle HTTP server can be configured for Basic authentication to legacy applications:

- The application requiring Basic authentication must be registered as an legacy application with the single sign-on server. See "Adding an External Application" in Chapter 2 of *Oracle9iAS Single Sign-On Administrator's Guide*.
- The Oracle HTTP server must have mod_osso installed and enabled.
- The Oracle HTTP server must have the default mod_proxy installed and enabled.
- The Web server that hosts the external application must *not* have mod_osso enabled. If mod_osso is enabled, the Basic authentication module mod_auth is bypassed altogether.

1.3 Configuration Steps

To configure the Oracle HTTP server for Basic authentication to legacy applications, add the following section to the mod_osso.conf file.

```
<IfModule mod_proxy.c>
<Location /application_virtual_path>
   require valid user
   AuthType Basic
   OssoLegacyApp on | off
</Location>

ProxyPass /application_virtual_path/ http://host:port/application_real_
path/
ProxyPassReverse /application_virtual_path/ http://host:port/
application_real_path/
</IfModule>

Listen 5000
```

The directive `OssoLegacyApp` indicates whether the protected URL is a legacy application. If the directive is missing or is set to `off`, the code that retrieves the application user name and password from the single sign-on database is not executed. The two mod_proxy directives `ProxyPass` and `ProxyPassReverse` map the virtual URL to the real URL.

Note that mod_osso accesses information about external applications through the non-SSL port 5000.

Notes:

- The directory where the virtual URL resides need not be specified. For convenience, this URL may consist of only the application name.
- If SSL is enabled, substitute `https` for `http` in the real URL of the application.

2 Reregistering the Oracle HTTP Server with the Single Sign-On Server

Mod_osso, the single sign-on module for the Oracle HTTP server, can be registered with the single sign-on server automatically when Oracle9*i*AS is installed. Under certain circumstances, you must reregister mod_osso manually, using the single sign-on registration tool. These circumstances are as follows:

- The host name and port number of the Oracle HTTP server are changed after Oracle9*i*AS is installed
- The osso.conf file is deleted or corrupted
- SSL is enabled on the single sign-on server after Oracle9*i*AS is installed

In the last case, running the single sign-on registration tool updates the mod_osso registration record in the osso.conf file to reflect SSL settings on the single sign-on server. The single sign-on registration tool generates this file whenever it is run.

To run the tool, execute the following command:

```
$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/sso/lib/ossoreg.jar
-oracle_home_path orclHomePath
-host database_host_name
-port database_port_number -sid database_SID
-site_name sitename
-success_url successurl
-logout_url logouturl
-cancel_url cancelurl
-home_url homeurl
[-admin_id adminid]
[-admin_info admininfo]
-config_mod_osso TRUE
-u userid
-sso_server_version v1.2
```

A description of the parameters passed to the tool follows.

oracle_home_path

Absolute path to the Oracle home.

site_name

Name of the site—typically, the host name and port (for example, the contiguous string *host*:*port)*.

success_url

URL to the routine responsible for establishing the partner application session and session cookies. This URL is as follows:

```
http://apachehost.domain:port/osso_login_success
```

Note that `osso_login_success` is *not* a variable. Note, too, that the protocol passed must be `https` if you want the Web client and the Oracle HTTP server to communicate using SSL.

logout_url

URL for the logout routine of the application. This URL is as follows:

```
http://apachehost.domain:port/osso_logout_success
```

Note that `osso_logout_success` is *not* a variable. Be sure to substitute `https` for `http` if you want the Web client and the Oracle HTTP server to communicate using SSL.

cancel_url

URL that specifies where the single sign-on server redirects the user when he or she cancels authentication. This URL is as follows:

```
http://apachehost.domain:port/
```

home_url

This URL is as follows:

```
http://apachehost.domain:port/
```

admin_id

User name of the mod_osso administrator (optional).

admin_info

Any additional information, such as e-mail address, about the administrator (optional).

config_mod_osso

If set to TRUE, this parameter indicates that the application being registered is mod_osso. You must include `config_mod_osso` for the osso.conf file to be generated.

u

The user name that will start the Oracle HTTP server. This name is usually "root." The parameter u is mandatory.

sso_server_version

This parameter must be set to v1.2 in Oracle9*i*AS, Release 2.

For more information about when to run the single sign-on registration tool, see Chapter 7, "Component Configuration Dependencies," in *Oracle9i Application Server Administrator's Guide.*

3 Configuring Globalization Support: Errata and Additions

The section "Configuring National Language Support" in Chapter 2 of *Oracle9iAS Single Sign-On Administrator's Guide* states that users can select a language when Oracle9*i*AS is installed. Please note though that a language is available only if the administrator installs it *after* Oracle9*i*AS is installed. Twenty-nine languages are available. Only one, English, is installed by default. To install additional languages, execute the following command:

```
$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/sso/lib/ossoca.jar langinst
lang make_lang_avail $ORACLE_HOME
```

For the variable *lang*, substitute the code for the language to be installed. For the variable *make_ lang_avail*, substitute 1 if you want to make the language available. Substitute 0 if you want to make the language unavailable.

For a complete list of the language codes supported, see Table 3-2 in *Oracle9i Application Server Globalization Support Guide*.

See Also: *Oracle9i Application Server Release Notes*

4 Using POST Authentication with Netegrity SiteMinder

Netegrity SiteMinder® must be configured to challenge users with Basic authentication if these users invoke SiteMinder by requesting a protected Oracle URL. If they use form-based POST authentication, they must log in to SiteMinder first. Requesting the application first returns the error "Page cannot be found" to the browser.

Both SiteMinder and Oracle Single Sign-On must reside in the same domain if POST authentication is used. If the domains are separate, SiteMinder requires the user to authenticate twice.

For more information about installing and deploying Netegrity SiteMinder, see Chapter 5, "Third-Party Single Sign-On," in *Oracle9iAS Single Sign-On Administrator's Guide.*

5 Default Password Policies

As stated in *Oracle9iAS Single Sign-On Administrator's Guide* (Chapter 3, "Directory-Enabled Single Sign"), user names and passwords are stored in Oracle Internet Directory. The administrator can use the GUI tool Oracle Directory Manager to configure password expiry. Barring that, passwords are subject to defaults. These are as follows:

- Password expiration: after 60 days
- Account lockout: after 10 consecutive login failures
- Minimum length of user password: five characters
- Numeric characters in password: at least one
- Password lockout duration: one day

6 Issues with Protecting Single Sign-On URLs with SSL

The section "Enabling the Single Sign-On Server for SSL" in Chapter 2 of *Oracle9iAS Single Sign-On Administrator's Guide*, instructs the administrator to enter the following directive in the dads.conf file to protect single sign-on URLs with SSL:

```
<IfDefine SSL>
   <Location /pls/orasso>
      SSLRequireSSL
   </Location>
</IfDefine>
```

Please note though that this directive must be used with specific URLs—for instance, the login and change password URLs—not with the entire single sign-on database access descriptor (DAD), `orasso`. If the directive is applied to the entire single sign-on DAD, the Oracle9*i*AS installer, Oracle9*i*AS Portal, and mod_osso are denied access to the single sign-on schema.

The installer uses the procedure `WWSSO_APP_ADMIN.SSOPING` to contact the single sign-on server when it installs a new Oracle9*i*AS middle tier. Oracle9*i*AS Portal and mod_osso use the single sign-on schema to obtain information about external applications. These URLs must *not* be protected by SSL. Instead you must configure them with directives that make them accessible only to hosts for the Oracle9*i*AS middle tier, Oracle Portal, and mod_osso.

To protect the URL for the procedure `WWSSO_APP_ADMIN.SSOPING`, navigate to the dads.conf file:

```
$ORACLE_HOME/Apache/modplsql/conf/dads.conf
```

Then enter the following directive:

```
<Location "pls/orasso/*[Ss][Ss][Oo][Pp][Ii][Nn][Gg]">
  Order deny, allow
  Deny from all
  Allow from <your domain name>
</Location>
```

For Oracle Portal, enter the following lines:

```
<Location "pls/orasso/*[Aa][Pp][Pp][Ss]_[Ll][Ii][Ss][Tt]">
  Order deny, allow
  Deny from all
  Allow from <your domain name>
</Location>
```

For mod_osso, enter the following lines:

```
<Location "pls/orasso/*[Gg][Ee][Tt]_[Ee][Xx][Tt]_[Aa][Pp][Pp]*">
  Order deny, allow
  Deny from all
  Allow from <your domain name>
</Location>
```

When the single sign-on server is SSL enabled, use the following directive to make login and change password URLs accessible only in SSL mode:

```
<IfDefine SSL>
  #Login URL for SSO Server and External Application
  <Location  "/pls/orasso/*[Ll][Oo][Gg][Ii][Nn]">
     SSLRequireSSL
  </Location>
  #Change password page
  <Location  "/pls/orasso/*[Pp][Aa][Ss][Ss][Ww][Oo][Rr][Dd]">
     SSLRequireSSL
  </Location>
  #External Application password page
  <Location  "/pls/orasso/*[Ff][Aa][Pp][Pp][Uu][Ss][Ee][Rr]">
     SSLRequireSSL
  </Location>
</IfDefine>
```

7 Protecting Single Sign-On URLs with SSL: Erratum

The section "Enabling the Single Sign-On Server for SSL" in Chapter 2 of *Oracle9iAS Single Sign-On Administrator's Guide* directs the reader to Chapter 3 of *Oracle9i Application Server Security Guide* to learn how to configure the Oracle HTTP server to use SSL. The cross-reference should be to Chapter 4—specifically to the section "Using Secure Sockets Layer (SSL) to Authenticate Users."

8 Security Issues: Single Sign-Off and Application Logout

Application developers who build custom applications using Oracle9*i*AS, Release 2, should note the following: when global logout, or single sign-off, is invoked, only the single sign-on and mod_osso cookies are cleared. This means that an Oracle9*i*AS application must be coded to store single sign-on user and subscriber names in either the OC4J session or in the application session. The application must then compare these values to those passed by mod_osso. If a match occurs, the application must show personalized content. If no match occurs, which means that the mod_osso cookie is absent, the application must clear the application session and force the user to log in.

This section covers the following topics:

- Application Login: Code Examples
- Application Logout: Recommended Code

8.1 Application Login: Code Examples

The first two code examples in this section do not incorporate the logic prescribed in the section immediately preceding. The third example does incorporate this logic. Although these are Java examples, they could be examples written in other languages such as Perl, PL/SQL, and CGI.

8.1.1 Bad Code Example #1

```
// Get user name from application session. This session was
// established by the application cookie or OC4J session cookie
String username = request.getSession().getAttribute('USER_NAME');

// Get subscriber name from application session. This session was
// established by the application cookie or OC4J session cookie.
String subscriber = request.getSession().getAttribute('SUBSCRIBER_
NAME');

// Get user security information from application session. This session
was established  by the application cookie or OC4J session cookie
```

```
String user_sec_info = request.getSession().getAttribute(’USER_APP_
SEC’);

if((username != null) && (subscriber!= null))
{
  // Show personalized user content
  show_personalized_page(username, subscriber, user_sec_info);
}
else
{
  // Send Dynamic Directive for login
  response.sendError( 499, "Oracle SSO" );
```

8.1.2 Bad Code Example #2

```
// Get SSO username from http header
String username = request.getRemoteUser();

// Get subscriber name from SSO http header
String subscriber = request.getHeader(’OSSO-SUBSCRIBER’);

// Get user security information from application session.  This session
// was established by the application or OC4J session
String user_sec_info =request.getSession().getAttribute(’USER_APP_SEC’);

if((ssousername != null)&&(subscriber!= null))
{
  // Show personalized user content
show_personalized_page(username, subscriber, user_sec_info);
}
else
{
  // Send Dynamic Directive for login
response.sendError( 499, "Oracle SSO" );
}
```

8.1.3 Recommended Code

```
// Get user name from application session. This session was
// established by the application or OC4J session
String username = request.getSession().getAttribute(’USER_NAME’);

// Get subscriber name from application session. This session was
// established by the application or OC4J session
String subscriber = request.getSession().getAttribute(’SUBSCRIBER_
NAME’);

// Get user security information from application session. This session
// was established by the application or OC4J session
```

```
String user_sec_info = request.getSession().getAttribute('USER_APP_
SEC');

// Get username and subscriber name from JAZN API */
JAZNUserAdaptor jaznuser  = (JAZNUserAdaptor)requset.getUserPrincipal();
     String ssousername   = jaznuser.getName();
     String ssosubscriber = jaznuser.getRealm().getName();

// If you are not using JAZN api then you can also get the username and
// subscriber name from mod_osso headers
String ssousername   = request.getRemoteUser();
String ssosubscriber = request.getHeader('OSSO-SUBSCRIBER');

// Check for application session. Create it if necessary.
if((username == null) || (subscriber == null) )
  {
    ...Code to create application session. Get the user information from
    JAZN api(or mod_osso headers if you are not using JAZN api) and
    populate the application session with user, subscriber and user
    security info...
  }

if((username != null)&&(subscriber != null)
  &&(ssousername != null)&&(ssosubscriber != null)
  &&(username.equalsIgnoreCase(ssousername) == 0 )
  &&(subscriber.equalsIgnoreCase(ssosubscriber) == 0))
{
  // Show personalized user content
show_personalized_page(username, subscriber, user_sec_info);
}
else
{
     ...Code to Wipe-out application session, followed by...

// Send Dynamic Directive for login
// If you are using JAZN then you should use following code
// response.sendError( 401);

// If you are not using JAZN api then you should use following code
// response.sendError( 499, "Oracle SSO" );
}
```

8.2 Application Logout: Recommended Code

Most applications that authenticate users have a logout link. In a single sign-on enabled application, the user invokes the dynamic directive for logout in addition to other code in the logout handler of the application. Invoking the logout directive initiates single sign-off, or global logout. The example that follows shows what single sign-off code should look like.

```
Clear application session, if any
String l_return_url := return url to your application e.g. home page
response.setHeader( "Osso-Return-Url", l_return_url);
response.sendError( 470, "Oracle SSO" );
```

> **See Also:** Chapter 2, "Developing Applications Using Mod_osso" in *Oracle9iAS Single Sign-On Application Developer's Guide*

9 Unlocking Single Sign-On Users

Oracle9*i*AS Single Sign-On, Release 2, includes the script `ssounlck.sql`. This script is obsolete. It should no longer be used to unlock users who have submitted an incorrect user name and password combination more times than is permitted by Oracle Internet Directory. Instead, use Delegated Administration Service (DAS) to unlock users. For information about DAS, see Chapter 9, "The Delegated Administration Service," in *Oracle Internet Directory Administrator's Guide.*

10 Issues with Oracle Internet Directory Outages

For performance reasons, the single sign-on server caches connections to Oracle Internet Directory. If the directory server has a scheduled or unscheduled outage, the single sign-on server is left holding bad directory connections, and users may encounter directory setup errors when they try to log in. If the LDAP connection cache is invalid, the Oracle HTTP server must be restarted.

Use the following steps to determine whether the LDAP connection cache must be refreshed:

1. Connect to the single sign-on schema.
2. Issue the following command:

   ```
   SELECT * FROM WWSSO_LOG$
   ```

3. Restart the HTTP server if you see the following error in the log:

   ```
   'INVALID LDAP CONNECTION CACHE: RESTART ORACLE HTTP
   SERVER'
   ```

4. Delete the error message from `WWSSO_LOG$`.

11 Locating the Single Sign-On Software Development Kit

The Oracle9*i*AS Single Sign-On Software Development Kit is included in an Oracle9*i*AS infrastructure installation. The kit contains installation instructions, required libraries, and sample programs in PL/SQL and Java. It can be found at the following location:

```
$ORACLE_HOME/sso/lib/ssosdk902.zip
```

12 A Note About Sample Programs for Java APIs

The Java programs documented in Chapter 4 of *Oracle9iAS Single Sign-On Application Developer's Guide* are written for JServ on the Oracle HTTP server. Note though that the code actually contained within the Oracle9*i*AS Single Sign-On Development Kit has been slightly modified for OC4J.

13 Browser History and the Customized Login Page

Older browsers cache the form input data for the customized single sign-on login page. This is a security risk because someone can log in as the user by selecting the browser back button. To prevent the user name and password from being cached, enter the following tags on the customized login page:

```
Pragma: no-cache
Cache-Control: no-cache
Expires: Thu, 01 Jan 1970 12:00:00 GMT
```

14 Using a Proxy Server with Oracle Single Sign-On

In network configurations where a range of distinct proxy addresses "front" the single sign-on server, the single sign-on IP check should be turned off. In such cases, a proxy routs the client's URL request to the single sign-on server. After authenticating, the client receives a cookie with information about the proxy address. When the client requests the application again, he or she might be routed by another proxy. If rerouting occurs, the single sign-on server refuses the request because the cookie that seems to originate with the second proxy records its origin as the first proxy.

To turn off IP check, log in to the single sign-on schema and set `url_cookie_ip_check` in the `WWSEC_ENABLER_CONFIG_INFO$` table to `N`. Use the following sequence of commands.

```
SQLPLUS orasso/password
UPDATE WWSEC_ENABLER_CONFIG_INFO$ SET url_cookie_ip_check='N';
COMMIT;
EXIT;
```

To enable a proxy server, do the following:

1. Run the script `ssocfg.sh` to change the host name stored in the single sign-on schema to the proxy host name on the infrastructure computer. The script can be found at the following location:

   ```
   $ORACLE_HOME/sso/bin
   ```

   Use the following command syntax, entering values for the protocol, host name, and port of the proxy server:

   ```
   ssocfg.sh protocol host port
   ```

2. Change the `ServerName` directive in the httpd.conf file of the single sign-on server to use the name of the proxy host.
3. Log in to the single sign-on server, using the single sign-on login URL:

   ```
   http://Single_Sign-On_host:Single_Sign-On_port/pls/orasso/
   ```

   This URL takes you to the single sign-on home page. If you are able to log in, you have configured the proxy correctly.
4. Reregister mod_osso with the single sign-on server. This step configures mod_osso to use the proxy host name instead of the actual host name. You can delete the old mod_osso registration record using the Edit Partner Application page in the single sign-on user interface. For details, see "Administering Partner Applications" in Chapter 2 of *Oracle9iAS Single Sign-On Administrator's Guide.*

15 Browser Settings for Oracle Single Sign-On

To log in and out of Oracle Single Sign-On successfully, you must have certain browser settings in place. These are as follows:

Cache Settings

To enable the correct cache settings:

1. Go to the cache settings dialog box:
   - Internet Explorer: Tools->Internet Options->General->Settings
   - Netscape Communicator: Edit->Preferences->Advanced->Cache
2. Select "Every visit to the page" in Internet Explorer or "Every time" in Netscape Communicator.

Image Settings

To ensure that images are automatically loaded:

1. Navigate as follows:
   - Internet Explorer: Tools->Internet Options->Advanced
   - Netscape Communicator: Edit->Preference->Advanced
2. Select "Show pictures" in Internet Explorer or "Automatically load images" in Netscape Communicator.

16 Setting Up ACLs for Certificate-Enabled Single Sign-On

Oracle9*i*AS users who use digital certificates to authenticate must not be able to update the `userCertificate` attribute of their directory entry. The reason is the potentially long lapse time between the revocation of a certificate and the update of the certification revocation list. Please note that Oracle Internet Directory, by default, denies the user access to the `userCertificate` attribute. It should be modified only by trusted entities such as the single sign-on server, Oracle Certificate Authority, or a third-party certificate authority.

17 PUBLIC User Account Must Not Be Deleted

When Oracle Single Sign-On is installed, a public, or unauthenticated, user is created in Oracle Internet Directory. The entry is created within the default subscriber subtree. It has the following distinguished name:

```
cn=PUBLIC,cn=users,o=subscriber,dc=com
```

This entry must *not* be removed. The single sign-on server cannot function properly without it. If the entry is missing, server performance suffers because of repeated attempts to locate the entry.

In the event that the entry must be added to the default user search base of an existing directory, care must be taken to omit the attribute `userPassword` from the entry. Omitting this attribute prevents the `PUBLIC` user account from being used to log in.

The entry should look something like this:

```
dn: cn=PUBLIC,cn=users,o=DEFAULT SUBSCRIBER,dc=com
cn: PUBLIC
sn: PUBLIC
orclactivestartdate: 01/01/02
objectclass: top
objectclass: person
objectclass: inetOrgPerson
```

```
objectclass: organizationalPerson
objectclass: orclUser
objectclass: orclUserV2
```

18 Bug: Enabling Oracle9*i*AS Web Cache

When the Oracle9*i*AS installer initially registers mod_osso with the single sign-on server, it passes registration URLs containing the port number of the Oracle HTTP server instead of the port number of Oracle9*i*AS Web Cache. Because the installer passes the incorrect port number, Web Cache is, in effect, bypassed.

To correct this error, reregister mod_osso manually, changing the port number in the registration URLs to the Web Cache port number. To learn how to run the single sign-on registration tool, see "Reregistering the Oracle HTTP Server with the Single Sign-On Server".

For detailed information about how changes in host name and port affect Oracle 9*i*AS—specifically, mod_osso and the single sign-on server—see Chapter 7, "Component Configuration Dependencies" in *Oracle9i Application Server Administrator's Guide*.