Forms 6*i* Single Sign-On Integration

An Oracle White Paper
August 2001

Updated version covering configuration enhancements introduced in Forms 6i patch 7

ORACLE

EXECUTIVE OVERVIEW

In the client-server world, application access is limited by the point of installation. In the Internet world, the Web offers users a broader access to applications simply by typing a URL from a browser. As corporate employees access multiple applications on the Web, each requiring different login information, the demand for Single Sign-On has increased steadily.

This paper covers how to integrate an Oracle9*i*AS Forms 6*i* Services application with Oracle9i Single Sign-On Server to perform Single Sign-On on the Web.

INTRODUCTION TO SINGLE SIGN-ON

Increasingly, companies are moving their information infrastructure online. Employees, customers, and partners are accessing company information across the Web. Initially, this information was generic, static, and public, not requiring access control or user authentication.

The explosion of self-service applications and online business transactions have moved private and business-critical data onto the Web, allowing, for example, employees to access their personal records, customers to place orders and pay by credit card, or suppliers to provide quotes for products or services. As a result, users who access private data via the Web must be authenticated before being granted access to that data. Typically, users are required to submit a username and password to gain access.

Without Single Sign-On, each user must maintain a separate identity and password for each application being accessed. Maintaining multiple accounts and passwords for each user is expensive, insecure, and most of all impractical.

Single Sign-On eliminates the need for users to remember multiple passwords. It provides a mechanism for users to gain access to multiple applications by entering a username and password only once and gain access to any application for which they are registered users.

INTRODUCTION TO ORACLE9I SINGLE SIGN-ON SERVER

Oracle9i Single Sign-On Server which is part of Oracle9*i* Application Server is Oracle's unified authentication solution for the Web, supporting external

authentication mechanisms like LDAP as well as providing a native authentication mechanism using its own password store in the Oracle database.

Starting with Oracle9*i* Application Server Release 2, the majority of Oracle tools, including Forms Services, Reports Services, and Discoverer, use Oracle9i Single Sign-On Server as a single source of authentication. Other server products like Oracle HTTP Server *powered by Apache* and Oracle Portal already use the Single Sign-On Server to provide lightweight user support for access control.

Oracle9i Single Sign-On Server offers two APIs, one for "Partner Applications" and one for "External Applications."

For more information about the Single Sign-On Server, refer to the Oracle9*i*AS Single Sign-On documentation provided in the Oracle9*i* Application Server documentation library.

Partner Applications

Partner Applications work within the SSO framework. Specifically, they are designed (or have been modified) to delegate user authentication to the Single Sign-On Server.

Since Partner Applications take advantage of the authentication services of the Single Sign-On Server, they do not need to implement their own authentication modules. User administration is simplified for Partner Applications since there is no need to manage passwords for each Partner Application. Thus, deploying a Partner Application can greatly reduce both development and ongoing administrative expenses.

External Applications

External Applications retain their own username and password administration; they do not delegate user authentication to the Single Sign-On Server and have not been developed or modified to work within the SSO framework. A typical External Application might be one developed or deployed by a third party, such as a Web page which requires a username and password to access custom services like e-mail.

Since External Applications do not take advantage of the Single Sign-On Server's authentication mechanism, they must implement their own authentication modules and manage user passwords. It is preferable to deploy applications as Partner Applications, but External Applications are also supported by Single Sign-On Server since it is not always practical to retrofit an existing application as a Partner Application.

Authentication Performed by Oracle Forms Applications

The Oracle Forms product has a very strong database-centric authentication model. As an application developer, you can define a database account for each user, or you can manage your own lightweight user system, maintaining a user table

in the database. In either ways, Forms authenticates via the user database account, similar to the client-server world where users were not required to remember as many passwords.

STARTING A FORMS APPLICATION ON THE WEB - REHEARSED

There are several possible ways to integrate Forms Web applications into a company's Web infrastructure. The easiest way is to call a static HTML file containing the applet tag required by the generic Java client to start the Forms Web application.

A better solution though is to use the Forms 6*i* Services Servlet interface (Forms60\Java\servlet.jar) to dynamically render the Forms start HTML page. By providing a single configuration file (formsweb.cfg), you can reduce the administrative costs for maintaining several different HTML files. In addition, the Forms 6*i* Services Servlet has been designed to support Single Sign-On integration.

When a client requests to access a Forms 6*i* application on the Web, the request is routed by the Forms 6*i* Services Servlet interface. The Servlet dynamically creates the start HTML page responsible for downloading a fixed set of Java classes (if not already cached by a previous run), building the generic Forms Web client.

Once downloaded, the Forms generic Java applet establishes a connection to the Forms Listener or the Forms Listener Servlet to spawn a Forms runtime engine, creating a SQL*Net connection to the database. The Forms Listener connects the runtime engine directly with the Forms Java client and waits for new incoming requests. Conversely, the new Forms Listener Servlet, introduced in Forms 6*i* patch 4, stays in between, routing any further communication between the runtime engine and the assigned Forms Java client.

To learn more about the new Forms deployment architecture using the Forms Listener Servlet, we recommend to read the Whiteaper "Forms 6*i* Patch5: Forms Listener Servlet for Deployment of Forms on the Internet" at http://otn.oracle.com.

This paper does not focus on the new Forms Listener Servlet architecture. However, note that the Forms Servlet and the Forms Listener Servlet are different parts in this architecture.

The Forms Listener Servlet (Forms60\Java\f60srv.jar) was introduced in patch4 as a substitute for the Forms Listener, performing the same tasks as the Forms 6*i* Services Listener.

The Forms Servlet is a Web interface and interoperates with both the Forms Listener (still supported with Forms 6*i*) and the Forms Listener Servlet. The Forms 6*i* Services Servlet in patch 5 or higher is required for Single Sign-On.

HOW FORMS INTEGRATES WITH SINGLE SIGN-ON

Starting with patch5 of Oracle9*i*AS Forms 6*i* Services, a Forms Web application can be integrated as an "External Application" with the Single Sign-On Server. Having said that, no additional coding is required in the application modules.

New in Forms 6i Patch 7

The two new features in patch set 7 of Forms 6i that relate to Single Sign-On are the hidden username and password in the generated Forms start HTML page and an easier way to configure a Forms application to use Single Sign-On.

Removing username and password from the Forms start HTML page

When using releases of Forms prior to patch 7 with Single Sign-On, the username and password of the provided application would be visible in the Applet tag of the downloaded start HTML file. Although this information wasn't cached in the browser, it was of concern to some customers.. In patch 7 the Forms Servlet is changed to pass the username and password invisibly behind the scenes to the Forms Listener Servlet.

Easier configuration for Single Sign-On

As you will see later in this document, configuring Single Sign-On has greatly simplified with patch 7, in that there is only one configuration file that needs to be changed to make Single Sign-On work. The Forms Servlet has been changed to map a user defined logon format, dynamicallyto the Forms native logon format of username/password@database.

Forms 6*i* Services SSO authentication with Oracle9i Single Sign-On Server

Oracle9i Single Sign-On Server, or Single Sign-On Server in short, was previously called Login Server. You'll find references to the old name when registering the Forms 6i Single Sign-On application with the Single Sign-On Server.

Before going into any details of how to set up a Forms 6*i* Services application to run in Single Sign-On mode, a high level overview of the authentication flow when a user requests a Forms application authenticated by Oracle9i Single Sign-On Server is presented in the Figure 1 below:

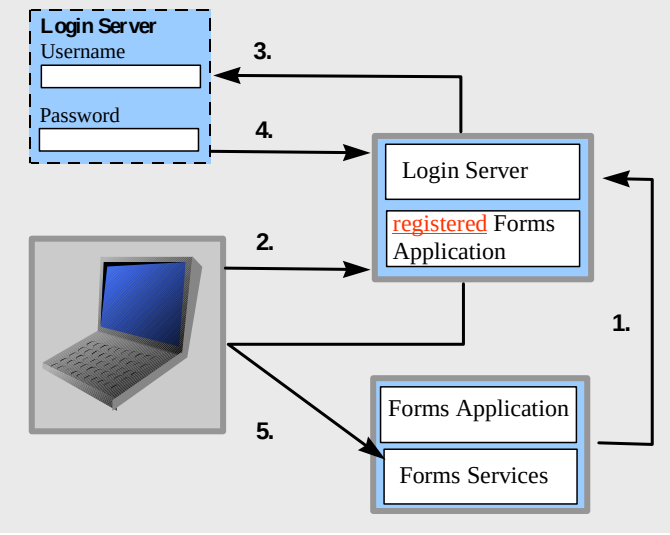

Figure 1: Authentication flow when calling a Form in Single Sign-On mode

The technical details of registering a Forms 6*i* application as an "External Application" to the Oracle9i Single Sign-On Server are covered later in this document. This section presents a high level overview.

1. Before a user can call a Forms 6*i* Services application in Single Sign-On mode, the application needs to be registered with the Single Sign-On Server. The registration includes the Forms 6*i* application access URL (e.g. "http://<hostname>/servlet/f60servlet?config=my_app& userid=scott/tiger@orcl") and a new Single Sign-On URL, internally mapped to this application URL, which is created by the Single Sign-On Server (e.g. "http://<hostname>/pls/portal30_sso/PORTAL30_SSO.wwsso_app_admin.fapp_process_login? p_app_id=106").

For this paper, the "SSO URL" is used to refer to the Single Sign-On Server URL which is internally mapped to the application URL, referred to as the "original URL."

2. To run the application in Single Sign-On mode, the user calls the **SSO URL**, generated by the Single Sign-On Server, instead of the **original URL**. The Single Sign-On Server checks for a specific SSO cookie, which is set the first time that a user runs any Single Sign-On application, to determine whether authentication has already been performed.

   If the user has already been authenticated, the Single Sign-On Server skips steps 3 and 4 and proceeds to step 5.

3. The absence of a Single Sign-On Server cookie in the user's request URL indicates that the Single Sign-On Server has not previously authenticated the user. In this case, the Single Sign-On Server redirects the client a logon dialog requiring the user to provide a valid username and password to access the Single Sign-On account. The Single Sign-On Server account is not necessarily the same account that the user needs to run the requested application. The Single Sign-On Server account is the user's lightweight username/ password pair for Single Sign-On.

4. After the user supplies a Single Sign-On username and password, the Single Sign-On Server checks that the information matches a valid SSO account, kept either in Single Sign-On Server or LDAP.

5. Once the user has been authenticated successfully, the Single Sign-On Server redirects the client to the application that the user originally requested, using the application's "original URL" provided by the administrator when registering the application with the Single Sign-On Server. From now on, the user doesn't have to reconnect again for the duration of the user's browser session.

   If a user decides not to store the application's credentials in the Single Sign-On Server password store, clearing the checkbox, the connect information is required each time the application is requested.

   The very first time that a user starts a Forms applications configured for Single Sign-On, the user is prompted for the username/password pair for this particular application. Optionally, this information can be stored permanently in the Single Sign-On Server password store to activate Single Sign-On. This step is required for all External Applications.

HOW FORMS 6I SERVICES BUILDS THE START HTML FILE LAUNCHING THE GENERIC JAVA CLIENT

A start HTML file, containing all applet initialization parameters required by Forms, is used to download and start the generic Forms Java Client. The Forms

Servlet dynamically creates the start HTML file by reading the basic parameter settings from one of three base HTML files, after detecting the client's browser type.

All basic HTML files contain parameter variables marked by enclosing "%" characters. During startup, these variables are replaced by a value provided in the URL or the Forms configuration file, "formsweb.cfg". The formsweb.cfg file is located in the Forms60\server directory and shares the same location with the base HTML files.

CONFIGURING FORMS 6I SERVICES FOR SINGLE SIGN-ON

Setting up Oracle Forms 6*i* Services as an External Application to the Oracle9i Single Sign-On Server requires no additional coding in any application module. All that is required are a few configuration steps performed in the configuration files related to the Forms Services architecture used with the Forms Servlet.

Configuration for Forms 6i patch 7

The current Forms 6*i* logon format for the Web requires a user to provide the connect information as userid=<username>/<password>@database, which does not conform to the standard Web logon format string supported by the Single Sign-On Server. The Oracle9*i* Single Sign-On Server standard logon format, like the default Web format, uses a combination of key/value pairs concatenated with a "&", for instance: username=scott &password=tiger&database=orcl.

In Forms 6*i* patch set 7 the Forms Servlet is modified to read the required target Web format from the formsweb.cfg configuration file and then map it to the internal format used by Forms.

The following file requires modifications to run a Forms application in Single Sign-On mode using Forms 6*i* patch 7

`formsweb.cfg`

Configuring the formsweb.cfg file to support SSO applications

The formsweb.cfg file in the Forms60\server directory is read by the Forms 6*i* Services Servlet when building the start HTML file for the Forms Web application.

The formsweb.cfg file contains information that is used to fill in the values for substitution variables in the underlying base.htm, baseie.htm, or basejini.htm templates.

The formsweb.cfg file allows the creation of user-defined, application-specific parameters, which can be invoked using a named shortcut as the "*config* " parameter on the URL.

Example:

http://<hostname>/servlet/f60servlet?**config=my_app**&userid=scott/tiger@orcl

In the above example, my_app is the shortcut for a user-defined application setting in the formsweb.cfg file:

[my_app]

Width=850

Height=500

form=reptest

lookandfeel=oracle

colorScheme=blue

serverURL=/servlet/oracle.forms.servlet.ListenerServlet

Notice that the connect information in above example is provided in the Forms native format username/password@database rather than the common format on the Web, username=<value>&password=<value>&database=<value>.

To use the Web logon format with Forms 6i Services applications, changes are required in this custom application section of the formsweb.cfg file.

[my_app]

Width=850

Height=500

form=reptest

lookandfeel=oracle

colorScheme=blue

serverURL=/servlet/oracle.forms.servlet.ListenerServlet

userid=%username%/%password%@%database%

Use following URL to call the defined application in the Web.

http://<hostname>/servlet/f60servlet?config=my_app**&username=scott&password=tiger&database=orcl**

There is no rule for defining the key names. Thus, instead of defining "username" as a parameter name, holding the username value, you can use anything else as

long as the formsweb.cfg file contain a matching "%<key name>% variable. In this paper, we use "username", "password," and "database" as the key names for defining the connect information.

The presence of username and password in the URL will go away when configuring the application with the Oracle 9*i* Single Sign-On Server.

Configuration for Forms 6i Patch 5 and 6

The Single Sign-On configuration for the Forms patches 5 and 6, explained in this section, is generic and continues working with patch 7. However, there is a good reason to use patch 7 as it includes a new feature that hides username and password from the downloaded Applet start HTML file, making Single Sign-On more secure.

The following files require modification to run a Forms application in Single Sign-On mode using Forms 6I patch 5 and patch 6.

- `formsweb.cfg`
- `base.htm`
- `baseie.htm`
- `basejini.htm`

To allow other applications to run in Forms native database authentication mode, make a copy of all the above-mentioned HTML files and prefix each name with "sso_" before performing any modifications.

Creating Forms 6*i* Base HTML Files for Single Sign-On

The current Forms 6*i* logon format for the Web requires a user to provide the connect information as userid=<username>/<password>@database, which is not conforming to the standard Web logon supported by the Single Sign-On Server. The Single Sign-On Server standard logon format, like the default Web format, foresees a combination of key/value pairs concatenated by a "&", like username=scott &password=tiger&database=orcl.

There is no rule for defining the key names. Thus, instead of defining "username" as a parameter name, holding the username value, you can use anything else as long as the target base HTML files contain a matching "%<key name>% variable. In this paper, we use "username", "password," and "database" as the key names for defining the connect information.

base.htm to sso_base.htm

The base.htm file defines the user connect string format as follows:

<PARAM NAME="serverArgs" VALUE="module=%form% ***userid=%userid%*** *%otherParams%">*

For Single Sign-On, change this line to the following:

<PARAM NAME="serverArgs"

VALUE="module=%form% ***userid=%username%/%password%@%database%*** *%otherParams%">*

baseie.htm to sso_baseie.htm

The baseie.htm file defines the user connect string format as follows:

<PARAM NAME="serverArgs" VALUE="module=%form% ***userid=%userid%*** *%otherParams%">*

For Single Sign-On, change this line to the following:

<PARAM NAME="serverArgs" VALUE="module=%form% ***userid=%username%/%password%@%database%*** *%otherParams%">*

basejini.htm to sso_basejini.htm

The basejini.htm file defines the user connect string format twice. For Internet Explorer and Netscape, the lines are as follows:

<PARAM NAME="serverArgs" VALUE="module=%form% ***userid=%userid%*** *%otherparams%">*

and

serverArgs="module=%form% ***userid=%userid%*** *%otherparams%"*

For Single Sign-On, change these lines to the following:

<PARAM NAME="serverArgs" VALUE="module=%form% ***userid=%username%/%password%@%database%*** *%otherParams%">*

and

serverArgs="module=%form% ***userid=%username%/%password%@%database%*** *%otherparams%"*

Configuring the formsweb.cfg file to support SSO applications

The formsweb.cfg file in the Forms60\server directory is read by the Forms 6*i* Services Servlet when building the start HTML file for the Forms Web application.

The formsweb.cfg file contains information that fill in the values for the variables of the underlying base.htm, baseie.htm, or basejini.htm template.

The formsweb.cfg file allows the creation of user-defined, application-specific parameter settings which can get called by a named shortcut using the "*config* " parameter in the URL.

Example:

```
http://<hostname>/servlet/f60servlet?config=my_app&userid=scott/tiger@orcl
```

In the above example, my_app is the shortcut for a user-defined application setting in the formsweb.cfg file:

[my_app]

WIDTH=850

HEIGHT=500

form=reptest

lookandfeel=oracle

colorScheme=blue

serverURL=/servlet/oracle.forms.servlet.ListenerServlet

To run Forms 6*i* with Single Sign-On, an application configuration needs to be added, overriding the default information in the system section of the formsweb.cfg file, referencing base.htm, baseie.htm and basejini.htm. The overriding information should point to the modified HTML base file templates as created earlier (sso_basehtm, sso_basie.htm and sso_basejini.htm).

[sso_app]

baseHTMLJInitiator=d:\oracle\806\forms60\server\sso_basejini.htm

baseHTMLIE=d:\oracle\806\forms60\server\sso_baseie.htm

baseHTML=d:\oracle\806\forms\server\sso_base.htm

IE50= native

WIDTH=850

HEIGHT=500

form=reptest

lookandfeel=oracle

colorScheme=blue

serverURL=/servlet/oracle.forms.servlet.ListenerServlet

The overriding HTML files require absolute URL addresses.

The "sso_app" configuration overwrites the default base.htm file reference and points to the modified HTML source files created for Single Sign-On. The following URL is valid, using the new login format.

http://<hostname>/servlet/f60servlet?**config=sso_app**&username=scott&password=tiger &database=orcl

You don't need to provide overriding information for all client configuration files as this is very much dependent on the clients that you are supporting.

The appearance of username and password in the URL will go away when configuring the application with the Oracle 9i Single Sign-On Server.

USING ORACLE PORTAL TO REGISTER FORMS 6*i* WITH ORACLE9*i* SINGLE SIGN-ON SERVER

The Oracle9*i* Single Sign-On Server is part of Oracle9*i* Application Server and is shipped integrated in Oracle9*i*AS Portal. The following section, explaining the steps required to setup Forms as an External Application with the Single Sign-On Server, uses Oracle9*i*AS Portal as a starting point.

Introducing Oracle9*i*AS Portal

For more information about Oracle9*i*AS Portal, refer to the appropriate Oracle Technology Network product Web page at otn.oracle.com.

Oracle9*i* Application Server Portal is Oracle's portal solution for the Web to build customizable entry points for a corporate Intranet or Internet presentation. Portal is accessible and manageable by any Web browser and supports the creation of individual portal pages holding business applications, business information, and other content of interest.

If, during installation of Oracle9*i* AS, you decide to use a different user than portal30 as the Portal administration user, then you must change the "portal30" string in the mentioned URL to this user.

Oracle9*i*AS Portal is shipped with Oracle9*i* Application Server and installs into an Oracle database. The Web access path for the portal administration page, using the default database scheme, is http://<hostname>/pls/portal30. The default administrator connect string is portal30/portal30.

Launching the Oracle9i Single Sign-On Server External Application Form

The Single Sign-On Server administration page can be reached from the Oracle9*i*AS Portal "Administer" tab, which contains the "Single Sign-On Server Administration" link.

Figure 2: Administer tab on Oracle9*i*AS Portal administration page

After clicking the "Single Sign-On Server Administration" link, click the "Administer External Applications" link, and then click the "Add External Application" link which displays the Single Sign-On Server registration form.

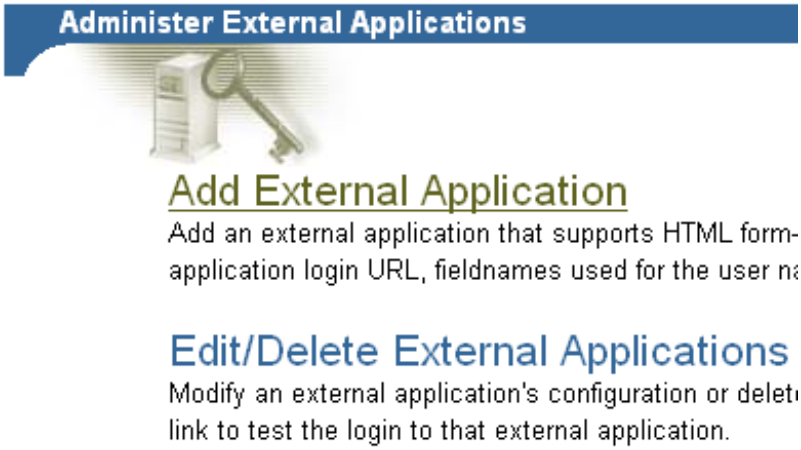

Figure 3: Administer External Application page

Registering a Forms application as an External Application

The Single Sign-On Server registration form for External Applications requires information about the application that is included in a Single Sign-On architecture.

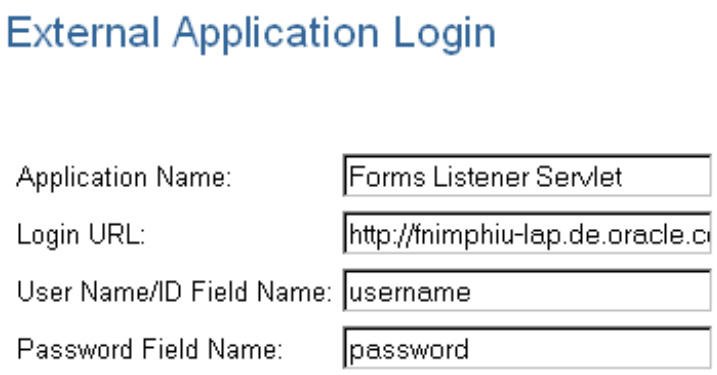

Figure 4: External Application registration and Login information

In the "**Application Name**" field, specify a meaningful name that helps to identify the configuration in the Single Sign-On Server's list of registered applications.

In the "**Login URL**" field, specify the Forms Servlet URL, e.g. http://fnimphiu-lap.de.oracle.com/servlet/f60servlet, **without** a question mark at the end.

In the "**Username/ID field name**" specify the name of the parameter containing the application username when calling the application directly from a URL. If the parameter name is "my_user_param," then this name is the value that would be inserted into the **Username/ID field name.**

In the "**Password field name**" field, specify the name of the parameter that holds the application password of the user, when calling the application directly from a URL. If the parameter name is "my_password_param" then this name is the value that would be inserted in the **Password field name** field**.**

By default, the type of authentication is set to **POST**, otherwise explicitly set it to this value.

Complete the "Additional Fields" section with any optional parameters that your application may require.

For Forms 6*i* there are at least two more required parameters, one for the database connect and one for the config parameter . In the example below, a parameter "database" has been added as "%database%" to the start HTML template files:

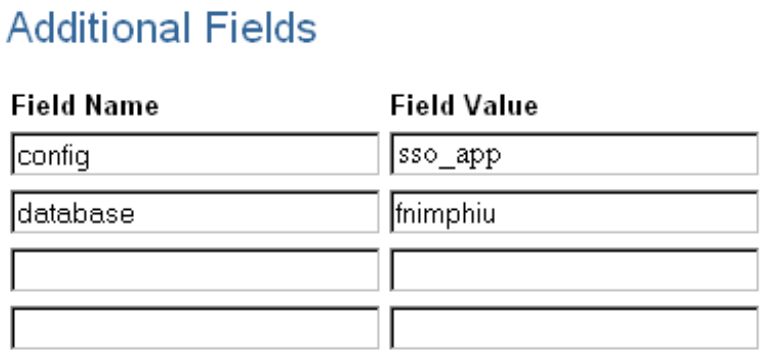

Figure 5: Additional fields taking additional parameters

If you don't provide values in these fields, then the user must enter them. The current version of Single Sign-On Server does not support hidden parameters.

The use of double quotes in a text string is used to highlight the specified value. Please omit double quotes when you are typing the field values.

After entering the configuration information and submitting the page, a new link, with the name that was provided in the "Application Name" field, is displayed in the External Applications list.

RUNNING A FORMS 6*i* APPLICATION IN SINGLE SIGN-ON MODE

To start a registered External Application in Single Sign-On mode, click the link that is provided by the Single Sign-On Server after registering the application.

You can copy the SSO URL for a registered Forms 6*i* Services application by right mouse clicking the link provided in the list of Single Sign-On Server External Applications with the name that you defined in the "Application Name" field on it. Publish this URL to the end users.

Example URL:

http://<hostname>/pls/portal30_sso/PORTAL30_SSO.wwsso_app_admin.fapp_process_login?p_app_id=106

When the end user starts the application, entering the application's URL for the very first time, two logon dialogs are displayed.

- The first dialog is the Single Sign-On Server login, in which the user provides the appropriate Single Sign-On username and password.

Figure 6: Single Sign-On login form

- The second dialog requires the user to provide and/or store the username, password, and database information required by Forms to start the application. Subsequent connects will only display the Single Sign-On logon dialog if the user was not previously authenticated by the Single Sign-On Server.

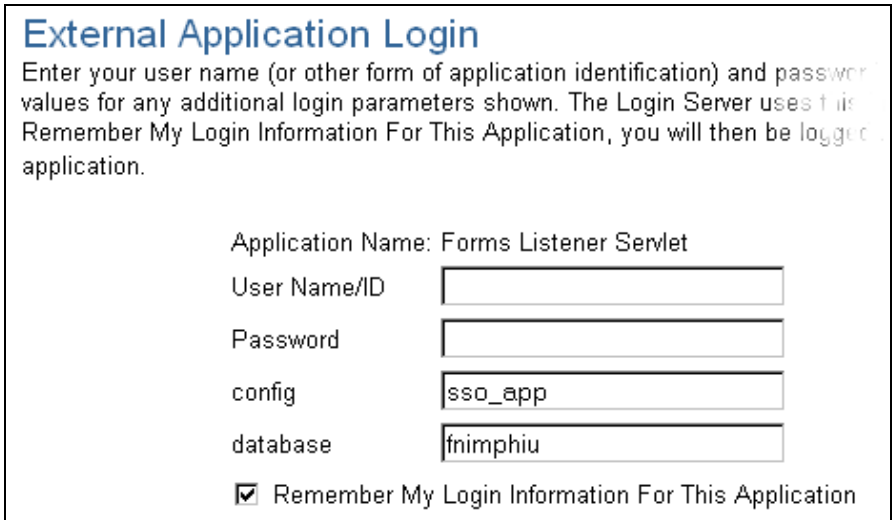

Figure 7: External Application login form

WHAT YOU SHOULD KNOW

Single Sign-On is not a replacement for Web security. Single Sign-On simply allows an application to start without requiring the user to first enter application credentials. The userid string used by Forms to connect a user to the database is still visible in the source code of the downloaded HTML page used to launch the Forms generic Java client unless Forms 6*i* patch 7 is used.

Also, because the syntax for providing the user's database connect information has changed, you must provide connect information when running in Single Sign-On mode. If the value for username, password and database is omitted, the final connect string becomes "/@" which leads to a TNS error message before the native Forms dialog gets displayed.

To handle an omitted connect information, avoiding a TNS error message, an on-logon trigger in Forms can be used in combination with the logon(…) built-in.

SUMMARY

Having users accessing multiple applications on the Web led to a demand for a Web-based Single Sign-On solution which allows users to have only one username/ password pair for authentication. Oracle9i Single Sign-On Server is Oracle's unified Single Sign-On solution and is part of Oracle 9*i* Application Server.

Starting with Forms 6*i* patch5, a Forms6i Services application can be launched using Single Sign-On. Doing this doesn't require any additional coding in your application because all that needs to be done is a number configuration steps on the middle-tier Server. Forms 6*i* Services is configured as an External Application to the Single Sign-On Server.

For Forms 6*i* patch 7 and above the Single Sign-On integration has been enhanced to hide the username/password pair from the HTML start page. In addition configuring Single Sign-On has become a one step task.

Forms 6/ Single Sign-On Integration
June/August 2001

Author: Frank Nimphius
Contributing Authors: Duncan Mills

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

Oracle Corporation provides the software
that powers the internet.