Universal Content Management

Version 10gR3

Security Providers Component Administration Guide

Table of Contents

Chapter 1: Introduction

Chapter 2: Installation

Chapter 3: Using Security Providers

Chapter

1

INTRODUCTION

OVERVIEW

This chapter covers the following topics:

PRODUCT OVERVIEW

The Security Providers component provides additional security for the Content Server by extending the functionality of basic socket providers with two new types of providers:

- Secure socket layer (SSL) providers
- Keepalive providers

Benefits of using the Security Providers component include the following:

- SSL enhances security for web communication by providing communication encryption and authentication.
- Security providers enable use of certificates for socket or server authentication.
- Keepalive and connection pooling logic help avoid SSL expense overhead by reducing the amount of SSL socket creation and teardown.

REQUIREMENTS

The following items are important to consider when deciding to use the Security Providers component:

- This component runs on Content Server under the supported Windows and UNIX operating systems.
- This documentation discusses the component as it runs on Content Server version 10gR3.

COMPONENT CONTENTS

The Security Providers component file, SecurityProvider.zip, is available for download and is included with the samples and extras. It contains the following files:

Description	**File Name**
Component files	*.hda *.class *.htm *.zip

AUDIENCE

This guide is intended for people who design, install, or administer the network connections and providers for server and client communication between the Content Server and web browsers. These architects and administrators must be familiar with networking and security concepts, and with Content Server services and processes.

ABOUT THIS GUIDE

This guide provides instructions to install and configure the Security Providers component on the Content Server. The information contained in this document is subject to change as the product technology evolves and as hardware, operating systems, and third-party software are created and modified.

Conventions

The following conventions are used throughout this document:

- The notation *<install_dir>/<instance>* is used to refer to the location on your system where a specific instance of Content Server is installed.
- Forward slashes (/) are used to separate parts of an Internet address. For example, http://www.microsoft.com/windows2000/. A forward slash might or might not appear at the end of an Internet address.
- Backward slashes (\) are used to separate the levels in a path to a server, directory, or file. For example, C:\stellent\. This is true when referring to files on a Windows file system or on a UNIX system. A backward slash will always appear after the end of a server, directory, or file path.
- Paths to access operating system dialogs or windows use the following formatting structure:

  Start—Settings—Control Panel
- Required user input is distinguished using the following font formatting:

  `xyz_name`

Chapter

2

INSTALLATION

OVERVIEW

This chapter covers the following topics:

- ❖ Installing The Security Providers Component (page 2-1)
- ❖ (page 2-3)

INSTALLING THE SECURITY PROVIDERS COMPONENT

To install and enable the Security Providers component on Stellent Content Server, use either the Component Wizard or the Component Manager.

Component Wizard Installation

Use this procedure to install the component using the Component Wizard.

1. Obtain the Security Providers component file, SecurityProviders.zip, from the component update bundle for release 10g. Place the file in a temporary location.
2. Start the Component Wizard by selecting **Start—Programs—Stellent Content Server—<*instance*>—Utilities—Component Wizard**.

   The Component Wizard main screen and the Component List screen are displayed.
3. On the Component List screen, click **Install**.

   The Install screen is displayed.

4. Click **Select**. Navigate to the location where you downloaded the SecurityProviders.zip file and select it.
5. Click **Open**.

   The zip file contents are added to the Install screen list.
6. Click **OK**.

   The Component Wizard prompts whether you want to enable the Security Providers component.
7. Click **Yes**.

   The Security Providers component is listed as enabled on the Component List screen.
8. Restart the content server.

Component Manager Installation

Use this procedure to install the component using the Component Manager.

1. Obtain the Security Providers component file, SecurityProviders.zip, from the component update bundle for release 10g. Place the file in a temporary location.
2. Open a new browser window and log in to your Stellent Content Server as the system administrator.
3. Open the Administration tray.
4. Click **Admin Applets** to open the Administration page.
5. Click **Admin Server**.
6. Click the applicable content server instance.
7. In the sidebar click **Component Manager**.

   The Component Manager screen is displayed.
8. Select the Browse button next to the Install New Component box. Navigate to the SecurityProviders.zip file and select it.
9. Click **Install**.

   A page is displayed listing the component items that will be installed.
10. Click **Continue** to continue with the installation.

    A Content Server message is displayed that indicates the installation is successful.
11. On the message page click to return to the Component Manager.

12. Click the component name in the Disabled Components box.
13. Click **Enable** to enable the component.

    The component is listed in the Enabled Components box.
14. In the sidebar click **Start/Stop Content Server**.
15. Restart the Content Server.

Chapter

3

USING SECURITY PROVIDERS

OVERVIEW

This chapter covers the following topics:

- About Security Providers (page 3-1)
- Managing SSL and Keepalive Providers (page 3-9)
- Security Provider Interface Pages (page 3-11)

ABOUT SECURITY PROVIDERS

The Security Providers component enables Secure Socket Layer (SSL) encryption and authentication for standard Stellent Content Server incoming and outgoing socket providers. The component also enables keepalive functionality for socket providers, which minimizes the creation and teardown of SSL sockets. Appropriate use of security providers, along with keys and certificates, can improve the security for network and Internet communication with the Content Server.

To use the Security Providers component it is necessary to be familiar with socket providers, security and authentication, SSL, keepalive, and other aspects of security for network communication. The following sources of information can be useful in working with the Security Providers component:

- *Stellent Content Server Managing System Settings and Processes*
- *Stellent Content Server Managing Security and User Access*

- *Sun Java Secure Socket Extension (JSSE) Reference Guide for the Java 2 SDK, Standard Edition, v. 1.4.2*.

  This online document is available from Sun at www.sun.com. It contains an extensive Related Documents section that includes web links to reference books, security standards, government security policies and regulations, and a list of books on cryptography and SSL.

- *keytool Key and Certification Management Tool*

  This online document is available from Sun at www.sun.com.

- RSA's *Public Key Cryptography Standards*.

  This online document is available from RSA at www.rsasecurity.com.

- RSA's *Cryptography FAQ*.

  This online document is available from RSA at www.rsasecurity.com.

- *SSL Certificate FAQ*.

  This online document is available from The Linux Documentation Project at www.tldp.org.

The following list includes definitions for some of the terms used in this guide. For detailed information refer to the previous list of information sources or to security and authentication standards sources.

- Certificate—A digital signature that verifies the identity and public key for an entity (a person or company). A certificate can be issued by a Certification Authority or by an individual entity.

- Certificate Authority (CA)—An entity that issues certificates for other entities, and is recognized as a well-known and trusted source for certificates, such as VeriSign and Thawte.

- Keystore—A file or database of information for keys, used for authentication processing.

- Private key—Information packaged as a key that is known only to the entity that issues it. Private keys are used in generating signatures.

- Public key—Information packaged as a key that is publicly associated with an entity. Public keys are used in verifying signatures.

- SSL—Secure Socket Layer, a protocol for secure network communication using a combination of public and secret key technology.
- Truststore—A file or database of keys that the trust manager has determined can be trusted.

Planning

It is recommended that you determine how you want to use security providers before implementing SSL socket providers or keepalive socket providers. Examine the keepalive and SSL connection types and determine whether additional configuration is required to use the security providers you select, such as a need to create keystores or a truststore. Refer to the additional sources of information listed in About Security Providers (page 3-1).

The following sections provide more information about the SSL and keepalive provider types, including the Java classes used to control the behavior of the provider types, and additional configuration that may be necessary.

- Keepalive Connections (page 3-3)
- SSL Connections (page 3-4)

Keepalive Connections

The keepalive feature enables persistent connections and the pooling of socket connections for service requests. The setup for keepalive connections is most useful in situations where connection setup and teardown can take a considerable amount of time, and you want to minimize the time spent on that activity. The Security Providers component provides two keepalive socket providers: incoming and outgoing.

The following Java classes are used to set up the keepalive incoming socket provider:

Provider Class	idc.provider.ExtendedSocketIncomingProvider
Connection Class	idc.provider.KeepaliveSocketIncomingConnection
Server Thread Class	idc.server.KeepaliveIdcServerThread

The following Java classes are used to set up the keepalive outgoing socket provider:

Provider Class	idc.provider.KeepaliveSocketOutgoingProvider
Connection Class	idc.provider.KeepaliveSocketOutgoingConnection
Request Class	idc.provider.KeepaliveServerRequest

SSL Connections

The SSL provider setup enables the use of SSL connections in a keepalive environment. This setup is recommended over a simple SSL provider setup because it helps minimize the cost of SSL socket setup and teardown. The Security Providers component provides two SSL socket providers with keepalive: incoming and outgoing.

The following Java classes are used to set up the SSL keepalive incoming socket provider:

Provider Class	idc.provider.ssl.SSLSocketIncomingProvider
Connection Class	idc.provider.KeepaliveSocketIncomingConnection
Server Thread Class	idc.server.KeepaliveIdcServerThread

The following Java classes are used to set up the keepalive SSL outgoing socket provider:

Provider Class	idc.provider.KeepaliveSocketOutgoingProvider
Connection Class	idc.provider.ssl.SSLSocketOutgoingConnection
Request Class	idc.provider.KeepaliveServerRequest

Additional Configuration

Depending on which type of security provider you choose, there can be additional required configuration.

- **Keepalive and SSL outgoing providers**—The Add Provider page includes a Num Connections field, which specifies the number of connections to pool.
- **SSL incoming providers**—The Add Provider page includes two additional options:
  - Request Client Auth option—If clients are able, they should authenticate themselves when they make a connection.
  - Require Client Auth option—Clients must authenticate themselves in order to make a connection.

SSL providers may also require setup of a keystore or keystores, and a truststore, for both the client and server, depending on the value of the Request Client Auth option, the value of the Require Client Auth option, and what type of Certification Authority signed the certificates handled by these options. For information on keystores and truststore refer to Keystores and Truststore (page 3-5).

Keystores and Truststore

SSL providers may require use of keystores and may require a truststore. Keystores are files that hold public and secret key information for use in SSL. A truststore contains certificates that have been determined to be trusted. If a certificate used on the server and client is signed by a well-known Certification Authority (CA) such as VeriSign or Thawte, then a truststore isn’t necessary, because the default JVM truststore contains the certificates of these CAs. Truststores are needed when certificates used by the SSL providers are self-signed or signed by a private CA. If SSL providers require keystores, and a truststore, then they must be created and managed.

The following sections provide overview information about keystores and truststore.

- When to Use Keystores and a Truststore (page 3-6)
- Specifying Keystore and Truststore Information (page 3-6)
- Generating a Keystore (page 3-6)
- Creating a Truststore (page 3-8)

For detailed information on keystores and truststores refer to the sources of information listed in About Security Providers (page 3-1).

When to Use Keystores and a Truststore

The following examples present different situations and uses for keystores and a truststore.

- The server requires a keystore containing a signed SSL certificate in order to create SSL sockets.
- The server requests or requires client authentication, which may require a truststore. If the client's certificate is not signed by a well-known CA, then the server will need a truststore containing that CA's certificate.
- The server requests or requires client authentication, which may require that the client have a keystore in which it stores the certificate the client presents for authentication.
- The server uses a certificate that hasn't been signed by a well-known CA, therefore the client will require a truststore that contains the server's certificate.

Specifying Keystore and Truststore Information

In order to use keystore and truststore information, the SSL incoming and outgoing providers require that a file named sslconfig.hda be set up in the providers directory (next to the provider.hda file). The sslconfig.hda file contains configuration information you specify for your keystore and truststore. It has a format similar to the following example. For security reasons, there is no web interface to assist in editing this file; all edits must be done manually using a text editor.

```
@Properties LocalData
TruststoreFile=/servers/idc/data/providers/ssloutgoing1/truststore
KeystoreFile=/servers/idc/data/providers/ssloutgoing1/keystore
@end
```

Configuration Name	Value Description
TruststoreFile	The full path to the truststore file.
KeystoreFile	The full path to the keystore file.

Generating a Keystore

This section describes the basic process for generating a keystore. You will need to determine the specific requirements and names for keys and keystores you want to create for your SSL providers. You can store keystore files wherever you want, because the sslconfig.hda file contains full paths for its KeystoreFile config settings. However, it is recommended that keystore files are stored in the

<*install_dir*>/data/providers/<*provider_name*> directory (next to the provider.hda and sslconfig.hda files) or in the <*install_dir*>/config/ directory. Aliases and passwords are set using the provider page in Content Server.

For detailed information on how to use the keytool utility to generate a keystore, see the document titled *keytool Key and Certification Management Tool*, available online from Sun at www.sun.com.

> **Note:** The Java keytool utility has a feature that prevents direct interaction with private keys. This means a certificate that is generated using keytool is “stuck” in the keystore; there is no way to retrieve the private key portion of the certificate. Inversely, there is no way for keytool to import a pre-existing certificate into a Java keystore.
>
> The Portecle Java keystore tool allows the import and export of private keys from Java keystores. For information on Portecle refer to portecle.sourceforge.net.

To use keytool you must have the utility in your path when you enter the command.

1. Create a key in a keystore. The following command-line example shows how to create a key entry with the name *alias* in a keystore with the name *keystore*. This command prompts for a keystore password, for information that will be used to generate the key, and for a password for the key itself. If the password on the key is different from the password on the keystore, the values KeystoreAlias and KeystoreAliasPassword will be required to retrieve the key.

   ```
   keytool -genkey -v -alias <alias> -keystore <keystore>
   ```

2. Generate a certificate signing request.The following command-line example shows how to generate a certificate signing request for the key entry named *alias* in the keystore named *keystore*, which is then stored in the file named *csr_file*. This file can be sent to a CA to be signed.

```
keytool -certreq -v -alias <alias> -keystore <keystore> -file <csr_file>
```

3. Import the CA’s certificate into the keystore. The keytool checks the chain of trust on the user’s certificate upon import. If the certificate was signed by a CA that is not well-known and the keytool knows nothing about the CA, the certificate is rejected. Therefore any certificate from a CA that is not well-known must first be imported into the keystore to permit the user’s certificate to successfully be imported in the next step. The following command-line example shows how to import a certificate in a file named *cert_file* into the keystore named *keystore*:

```
keytool -import -v -alias <ca_alias> -keystore <keystore> -file <cert_file>
```

4. Import the signed certificate back into the keystore. Once the certificate signing request has been received by a CA and the signed certificate is sent back from the CA, the certificate can be read into the keystore entry identified by *alias*.The following command-line example shows how to import the signed certificate.

```
keytool -import -v -alias <alias> -keystore <keystore_name> -file <csr_file>
```

5. Check that everything is in the keystore.

```
keytool -list -v -keystore <keystore_name>
```

Creating a Truststore

This section describes the basic process for generating a truststore. A truststore is necessary when an SSL provider uses keys that have not been signed by a well-known Certification Authority. A truststore contains only public certificates that have been verified by the person managing the truststore (the *trust manager*) for the content server. You will need to determine the specific requirements and name for the truststore you want to create. You can store a truststore file wherever you want, because the sslconfig.hda file contains a full path for a TruststoreFile config setting. However, it is recommended that a truststore file is stored in the <*stellent_dir*>/data/providers/<*provider_name*> directory (next to the provider.hda and sslconfig.hda files) or in the <*stellent_dir*>/config/ directory.

For detailed information on how to use the keytool utility to generate a truststore refer to the document titled *keytool Key and Certification Management Tool*, available online from Sun at www.sun.com.

To use keytool you must have the utility in your path when you enter the command.

To create a truststore enter the following command:

```
keytool -import -v -alias <alias> -keystore <keystore> -file <cert_files>
```

Variable	**Description**
<*alias*>	The alias name for the key.
<*keystore*>	The name of the keystore.
<*cert_files*>	The path to the Certification Authority's certificate.

MANAGING SSL AND KEEPALIVE PROVIDERS

The following tasks are involved in managing SSL and keepalive security providers:

❖ Adding an Incoming Security Provider (page 3-9)

❖ Adding an Outgoing Security Provider (page 3-10)

For detailed information on managing providers for the content server and how to edit or delete providers refer to *Stellent Content Server Managing System Settings and Processes*.

Adding an Incoming Security Provider

To add an incoming security provider, follow these steps:

1. Display the Providers (page 3-11).
2. In the Create a New provider table, click **Add** in the Action column for the incoming security provider type.

   The incoming security provider page is displayed.
3. Complete the following fields:

   Required fields

   - Provider Name
   - Provider Description
   - Provider Class (provided)
   - Server Port

   Optional fields

   - Connection Class (provided)
   - Configuration Class
   - Server Thread Class (provided)

   Optional check boxes (sslincoming provider only)

   - Request Client Authentication
   - Require Client Authentication
4. Click **Add**.

   The Providers page is displayed, with the new provider added to the Providers table.
5. Restart the content server.

Adding an Outgoing Security Provider

To add an outgoing security provider, follow these steps:

1. Display the Providers (page 3-11).
2. In the Create a New provider table, click **Add** in the Action column for the outgoing security provider type.

   The outgoing security provider page is displayed.
3. Complete the following fields:

   Required fields

   - Provider Name
   - Provider Description
   - Provider Class (provided)
   - Server Host Name (provided)
   - Server Port
   - Instance Name
   - Relative Web Root

   Optional fields

   - Connection Class (provided)
   - Configuration Class
   - Request Class (provided)
   - Number of Connections (provided)
   - HTTP Server Address
   - Proxied (check box)
   - Notify Target (check box)
   - Users (check box)
   - Released Documents (check box)
   - Enterprise Searchable (check box)
   - Required Roles
   - Account Filter
4. Click **Add**.

   The Providers page is displayed, with the new provider added to the Providers table.

5. Restart the content server.

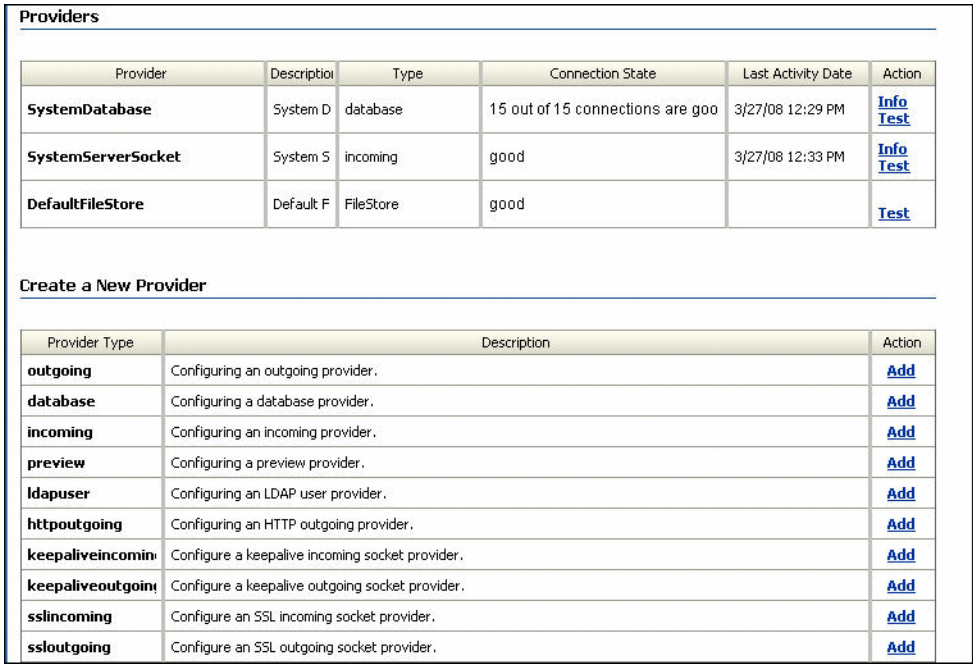
SECURITY PROVIDER INTERFACE PAGES

The Security Providers component enables the following provider interface screens to manage security providers:

- Add keepaliveincoming Provider Page (page 3-13)
- Add keepaliveoutgoing Provider Page (page 3-14)
- Add sslincoming Provider Page (page 3-17)
- Add ssloutgoing Provider Page (page 3-19)

Providers Page

When the Security Providers component has been installed and enabled, the Providers page enables administrators to create SSL and keepalive providers. To access this page click the Providers link on the Administration menu for the Content Server.



Providers

Provider	Descriptioı	Type	Connection State	Last Activity Date	Action
SystemDatabase	System D	database	15 out of 15 connections are goo	3/27/08 12:29 PM	Info Test
SystemServerSocket	System S	incoming	good	3/27/08 12:33 PM	Info Test
DefaultFileStore	Default F	FileStore	good		Test

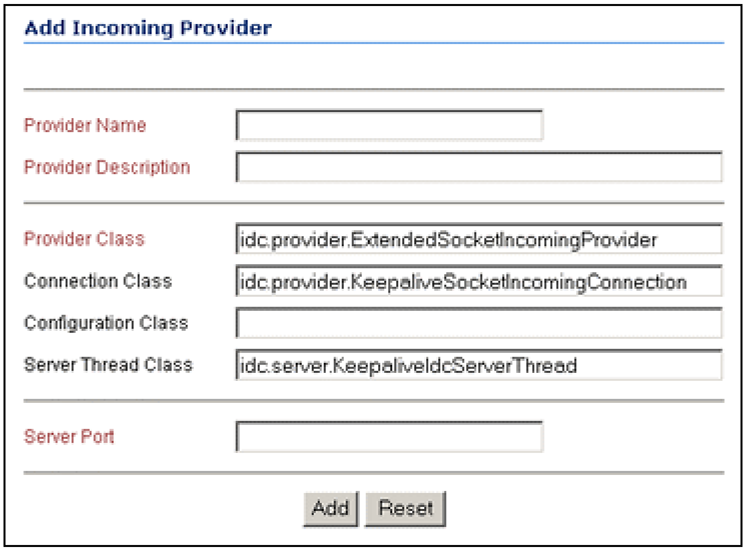
Create a New Provider

Provider Type	Description	Action
outgoing	Configuring an outgoing provider.	Add
database	Configuring a database provider.	Add
incoming	Configuring an incoming provider.	Add
preview	Configuring a preview provider.	Add
ldapuser	Configuring an LDAP user provider.	Add
httpoutgoing	Configuring an HTTP outgoing provider.	Add
keepaliveincomin	Configure a keepalive incoming socket provider.	Add
keepaliveoutgoin	Configure a keepalive outgoing socket provider.	Add
sslincoming	Configure an SSL incoming socket provider.	Add
ssloutgoing	Configure an SSL outgoing socket provider.	Add

Feature	Description
Providers table	
Provider column	The name of the provider that establishes connection to outside entities.
Description column	A description of the provider.
Provider Type column	The type of provider. For example, *incoming*, or *database*.
Connection State column	Possible states are: • misconfigured • good • down • requires restart
Last Activity Date column	The last date and time that the provider was active.
Actions column	• The Info link displays the Provider Information Page for the provider. • The Test link refreshes the Connection State and Last Activity columns for the provider.
Create a New Provider table	
Provider Type column	The type of provider.
Description column	A description of the provider type.
Action column	Clicking an Add button displays the Add/Edit Provider page for that type of provider.

Add keepaliveincoming Provider Page

The Add Incoming Provider page for the keepalive function is used to create a keepalive socket incoming provider. To access this page click the Providers link on the Administration menu for the Content Server, then click **Add** in the Action column for the keepaliveincoming provider type.



Add Incoming Provider

Provider Name

Provider Description

Provider Class: idc.provider.ExtendedSocketIncomingProvider

Connection Class: idc.provider.KeepaliveSocketIncomingConnection

Configuration Class

Server Thread Class: idc.server.KeepaliveIdcServerThread

Server Port

Feature	Description
Provider Name field*	The name of the provider.
Provider Description field*	A description of the provider.
Provider Class field*	The name of the Java class for the provider. For example: *idc.provider.ExtendedSocketIncomingProvider*
Connection Class field	The name of the Java class that implements the provider connection. For example: *idc.provider.KeepaliveSocketIncomingConnection*
Configuration Class field	The name of a Java class that performs some extra configuration. This class is very useful for database providers, where the connection classes are already providers.

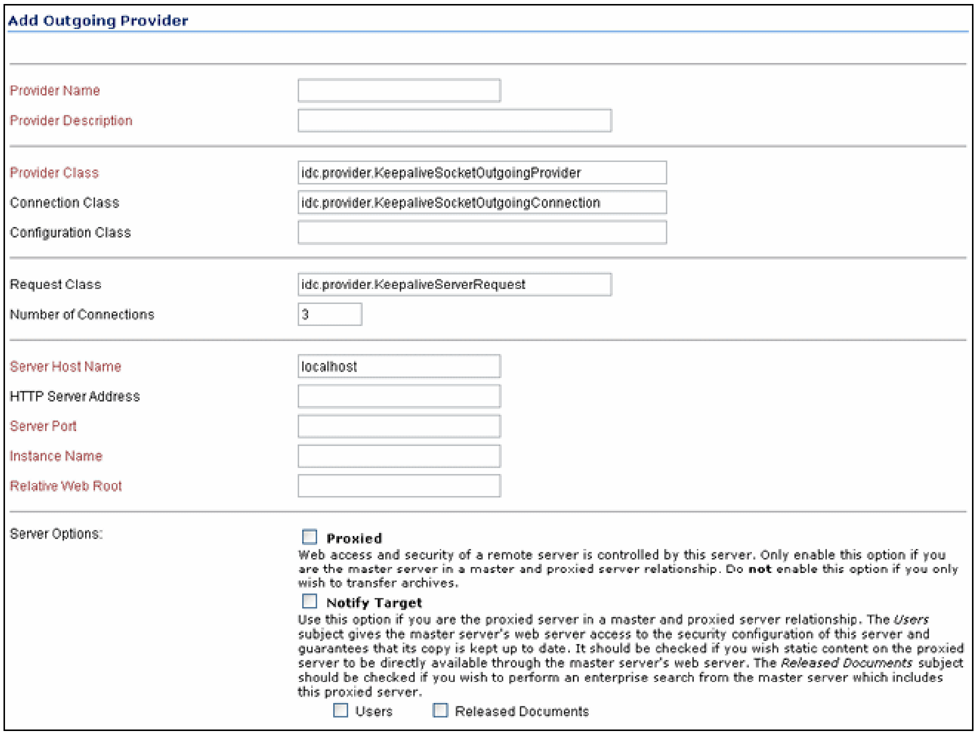
Feature	Description
Server Thread field	The name of the server thread for incoming connections. For example: *idc.provider.KeepaliveIdcServerThread*
Server Port field*	The port the provider listens on for incoming connections. For example, the incoming system provider listens on port 4444 by default.
Add button	Saves the provider information.
Reset button	Resets the provider information to the last saved values.

* Required metadata fields.

Add keepaliveoutgoing Provider Page

The Add Outgoing Provider page for the keepalive function enables administrators to create a keepalive socket outgoing provider. To access this page, select the Providers link from the Administration menu for the Content Server, then select **Add** in the Action column for the keepaliveoutgoing provider type.

The following two images show this page.



Add Outgoing Provider

Provider Name	
Provider Description	
Provider Class	idc.provider.KeepaliveSocketOutgoingProvider
Connection Class	idc.provider.KeepaliveSocketOutgoingConnection
Configuration Class	
Request Class	idc.provider.KeepaliveServerRequest
Number of Connections	3
Server Host Name	localhost
HTTP Server Address	
Server Port	
Instance Name	
Relative Web Root	

Server Options:

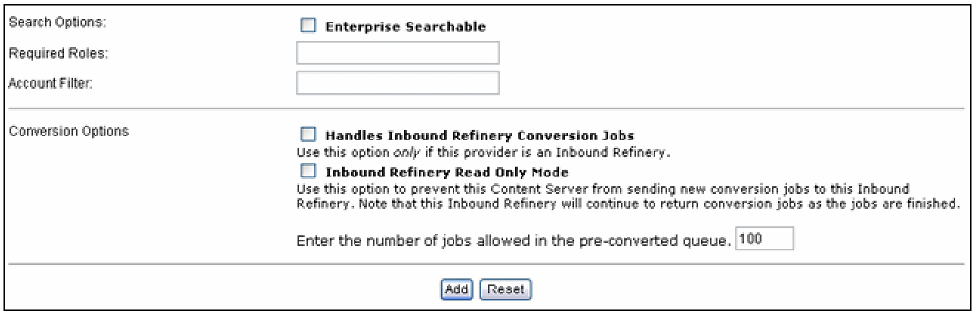
□ **Proxied**

Web access and security of a remote server is controlled by this server. Only enable this option if you are the master server in a master and proxied server relationship. Do **not** enable this option if you only wish to transfer archives.

□ **Notify Target**

Use this option if you are the proxied server in a master and proxied server relationship. The *Users* subject gives the master server's web server access to the security configuration of this server and guarantees that its copy is kept up to date. It should be checked if you wish static content on the proxied server to be directly available through the master server's web server. The *Released Documents* subject should be checked if you wish to perform an enterprise search from the master server which includes this proxied server.

□ Users □ Released Documents



Search Options: □ **Enterprise Searchable**

Required Roles:

Account Filter:

Conversion Options

□ **Handles Inbound Refinery Conversion Jobs**

Use this option *only* if this provider is an Inbound Refinery.

□ **Inbound Refinery Read Only Mode**

Use this option to prevent this Content Server from sending new conversion jobs to this Inbound Refinery. Note that this Inbound Refinery will continue to return conversion jobs as the jobs are finished.

Enter the number of jobs allowed in the pre-converted queue. 100

Feature	Description
Provider Name field*	The name of the provider.
Provider Description field*	A description of the provider.
Provider Class field*	The name of the Java class for the provider. For example: *idc.provider.KeepaliveSocketOutgoingProvider*
Connection Class field	The name of the Java class that implements the provider connection. For example: *idc.provider.KeepaliveSocketOutgoingConnection*

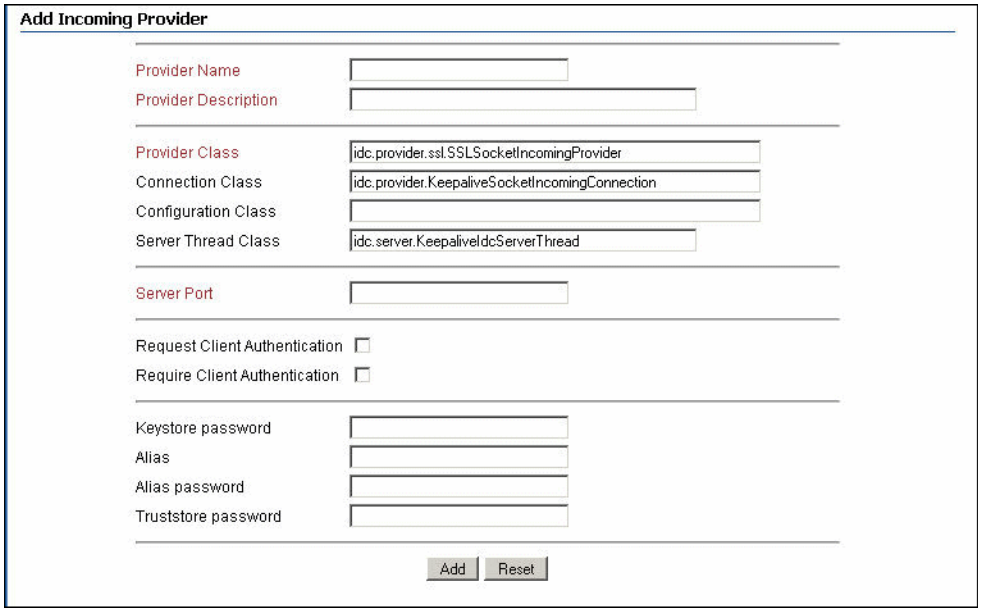
Feature	Description
Configuration Class field	The name of a Java class that performs some extra configuration.
Request Class field	The name of the Java class that implements the server request. For example: *idc.provider.KeepaliveServerRequest*
Number of Connections field	The maximum number of connections. For example, 3.
Server Host Name field*	The server host name of the other content server instance. For example, *localhost*.
HTTP Server Address field	The HTTP address of the other content server instance.
Server Port field*	The port on which the provider communicates with the other content server.
Instance Name field*	the instance name of the other content server instance.
Relative Web Root field*	The relative web root of the other content server instance.
Proxied check box	Enable this option if the provider is connecting to a content server that will be controlled by the current instance.
Notify Target check box	Enable this option if the provider is connecting to a content server that is acting as a controlling instance, and you want this content server to notify the controlling instance when user information and/or content item information changes.
Users check box	Enable this option if you want this content server to notify the controlling instance when user information changes.
Released Documents check box	Enable this option if you want this content server to notify the controlling instance when content item information changes.

Feature	Description
Enterprise Searchable check box	Enable this option if you have enabled Enterprise Search and you want this content server instance to be searchable. See the *Stellent Enterprise Search Administration and User Guide* for more information.
Required Roles field	Enter roles that have permission to search this content server instance using Enterprise Search. If no roles are entered, all users will have permission.
Account Filter field	Enter accounts that have permission to search this content server instance using Enterprise Search. If no accounts are entered, all users will have permission.
Conversion options	Used to indicate if the provider is an Inbound Refinery.
Add button	Saves the provider information.
Reset button	Resets the provider information to the last saved values.

* Required metadata fields.

Add sslincoming Provider Page

The Add Incoming Provider page for the sslincoming function enables administrators to create an SSL socket incoming provider. To access the page click the Providers link from the Administration menu for the Content Server, then select **Add** from the Action column for the “sslincoming” provider type.



Add Incoming Provider

Provider Name	
Provider Description	
Provider Class	idc.provider.ssl.SSLSocketIncomingProvider
Connection Class	idc.provider.KeepaliveSocketIncomingConnection
Configuration Class	
Server Thread Class	idc.server.KeepaliveIdcServerThread
Server Port	
Request Client Authentication	□
Require Client Authentication	□
Keystore password	
Alias	
Alias password	
Truststore password	

Feature	Description
Provider Name field*	The name of the provider.
Provider Description field*	A description of the provider.
Provider Class field*	The name of the Java class for the provider. For example: *idc.provider.ssl.SSLSocketIncomingProvider*
Connection Class field	The name of the Java class that implements the provider connection. For example: *idc.provider.KeepaliveSocketIncomingConnection*
Configuration Class field	The name of a Java class that performs some extra configuration. This class is very useful for database providers, where the connection classes are already providers.
Server Thread field	The name of the server thread for incoming connections. For example: *idc.provider.KeepaliveIdcServerThread*
Server Port field*	The port the provider listens on for incoming connections. For example, the incoming system provider listens on port 4444 by default.

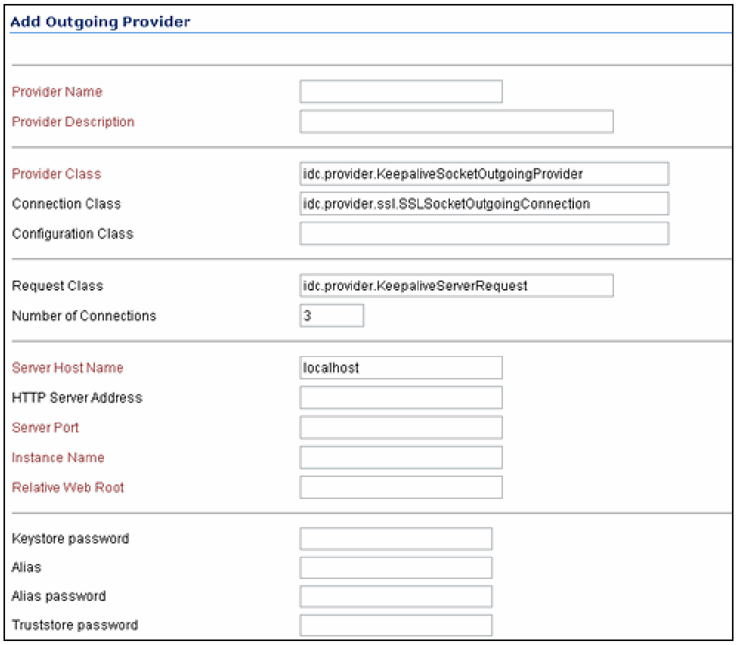
Feature	Description
Request Client Authentication check box	Enable this option if you want the provider to request client authentication from the incoming connection.
Require Client Authentication check box	Enable this option if you want the provider to require client authentication from the incoming connection.
Keystore/Alias/Truststore information	If necessary, enter the keystore password name, the alias, the alias password, and the truststore password.
Add button	Saves the provider information.
Reset button	Resets the provider information to the last saved values.

* Required metadata fields.

Add ssloutgoing Provider Page

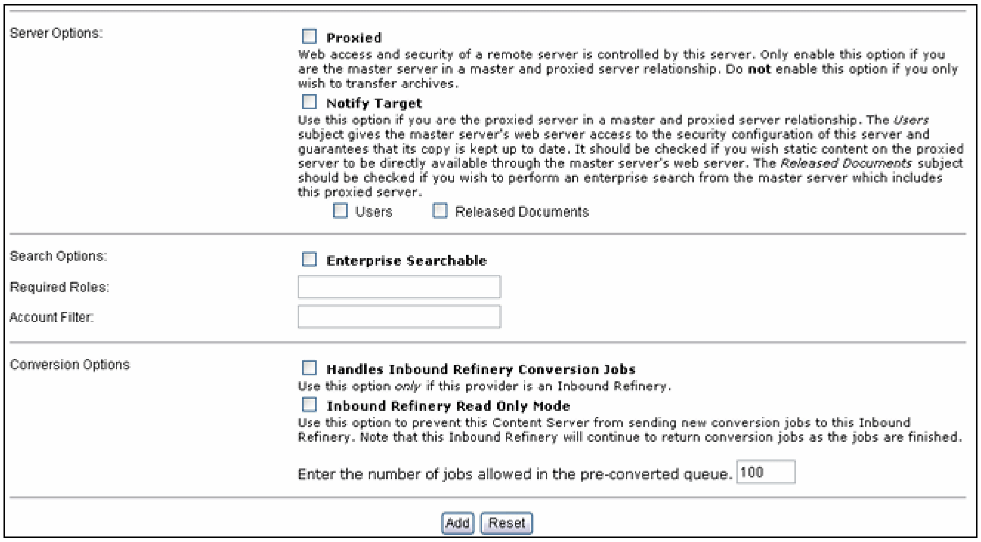
The Add Outgoing Provider page for the ssloutgoing function enables administrators to create an SSL socket outgoing provider. To access the page click the Providers link from the Administration menu for the Content Server, then select **Add** from the Action column for the “ssloutgoing” provider type.

The following two images show this page.



Add Outgoing Provider

Provider Name	
Provider Description	
Provider Class	idc.provider.KeepaliveSocketOutgoingProvider
Connection Class	idc.provider.ssl.SSLSocketOutgoingConnection
Configuration Class	
Request Class	idc.provider.KeepaliveServerRequest
Number of Connections	3
Server Host Name	localhost
HTTP Server Address	
Server Port	
Instance Name	
Relative Web Root	
Keystore password	
Alias	
Alias password	
Truststore password	



Server Options:

□ **Proxied**

Web access and security of a remote server is controlled by this server. Only enable this option if you are the master server in a master and proxied server relationship. Do **not** enable this option if you only wish to transfer archives.

□ **Notify Target**

Use this option if you are the proxied server in a master and proxied server relationship. The *Users* subject gives the master server's web server access to the security configuration of this server and guarantees that its copy is kept up to date. It should be checked if you wish static content on the proxied server to be directly available through the master server's web server. The *Released Documents* subject should be checked if you wish to perform an enterprise search from the master server which includes this proxied server.

□ Users □ Released Documents

Search Options: □ **Enterprise Searchable**

Required Roles:

Account Filter:

Conversion Options

□ **Handles Inbound Refinery Conversion Jobs**

Use this option *only* if this provider is an Inbound Refinery.

□ **Inbound Refinery Read Only Mode**

Use this option to prevent this Content Server from sending new conversion jobs to this Inbound Refinery. Note that this Inbound Refinery will continue to return conversion jobs as the jobs are finished.

Enter the number of jobs allowed in the pre-converted queue. 100

Feature	Description
Provider Name field*	The name of the provider.
Provider Description field*	A description of the provider.

Feature	**Description**
Provider Class field*	The name of the Java class for the provider. For example: *idc.provider.KeepaliveSocketOutgoingProvider*
Connection Class field	The name of the Java class that implements the provider connection. For example: *idc.provider.KeepaliveSocketOutgoingConnection*
Configuration Class field	The name of a Java class that performs some extra configuration.
Request Class field	The name of the Java class that implements the server request. For example: *idc.provider.KeepaliveServerRequest*
Number of Connections field	The maximum number of connections. For example, 3.
Server Host Name field*	The server host name of the other content server instance. For example, *localhost*.
HTTP Server Address field	The HTTP address of the other content server instance.
Server Port field*	The port on which the provider communicates with the other content server.
Instance Name field*	the instance name of the other content server instance.
Relative Web Root field*	The relative web root of the other content server instance.
Keystore/Alias/Truststore information	If necessary, enter the keystore password name, the alias, the alias password, and the truststore password.
Proxied check box	Enable this option if the provider is connecting to a content server that will be controlled by the current instance.
Notify Target check box	Enable this option if the provider is connecting to a content server that is acting as a controlling instance, and you want this content server to notify the controlling instance when user information and/or content item information changes.

Feature	**Description**
Users check box	Enable this option if you want this content server to notify the controlling instance when user information changes.
Released Documents check box	Enable this option if you want this content server to notify the controlling instance when content item information changes.
Enterprise Searchable check box	Enable this option if you have enabled Enterprise Search and you want this content server instance to be searchable. See the *Stellent Enterprise Search Administration and User Guide* for more information.
Required Roles field	Enter roles that have permission to search this content server instance using Enterprise Search. If no roles are entered, all users will have permission.
Account Filter field	Enter accounts that have permission to search this content server instance using Enterprise Search. If no accounts are entered, all users will have permission.
Conversion options	Used to indicate if the provider is an Inbound Refinery.
Add button	Saves the provider information.
Reset button	Resets the provider information to the last saved values.

* Required field

Index

I