Novell SAML Extension for Novell® iChain®

www.novell.com

SAMPLE SITE SETUP GUIDE

Novell®

Legal Notices

Novell, Inc.
1800 South Novell Place
Provo, UT 84606
U.S.A.

www.novell.com

SAML Extension Sample Site Setup Guide
September 2004

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see www.novell.com/documentation.

Novell Trademarks

ConsoleOne is a registered trademark of Novell, Inc. in the United States and other countries.
iChain is a registered trademark of Novell, Inc. in the United States and other countries.
NetWare is a registered trademark of Novell, Inc. in the United States and other countries.
Novell is a registered trademark of Novell, Inc. in the United States and other countries.
Novell eDirectory is a trademark of Novell, Inc. in the United States and other countries.

Third-Party Trademarks

All third-party trademarks are the property of their respective owners.

Contents

About This Guide 7

Introduction . . . 7
Documentation Conventions . . . 7
Documentation Updates . . . 7

1 Setting Up the iChainSite SAML Sample Site 9

System Layout . . . 9
Prerequisites . . . 10
Overview of the iChainSite Setup . . . 11
Configuring the iChain Accelerator . . . 11
Configuring the iChain Protected Resource and OLAC . . . 12
Deploying the iChainSite Sample Application . . . 13
Installing the SAML Extension for Novell iChain Software . . . 14
Configuring SAML and ConsoleOne . . . 14
Creating the SAML Trusted Affiliate . . . 21
Starting the SAML Extension Server . . . 29
Deploying the SAML Extension Server Application . . . 29
Starting the Servlet Container (Tomcat) . . . 29
Testing the Loopback Affiliate Links . . . 30
Conclusion . . . 32

2 Setting Up the eMartian Sample Site 33

Prerequisites . . . 33
Setting Up the eMartian Site . . . 34
Configuring the iChain Accelerator . . . 35
Defining the iChain Protected Resource and OLAC . . . 35
Deploying the eMartian Sample Application . . . 36
Installing the SAML Extension for Novell iChain Software . . . 39
Configuring SAML and ConsoleOne . . . 39
Creating the SAMLExtensionServer . . . 39
Creating the ISO ProviderSiteID Link . . . 39
Creating the SAMLSiteConfig Object . . . 39
Creating SAML Trusted Affiliates . . . 41
General Tab . . . 41
User Mapping . . . 42
Audiences Tab . . . 43
Assertions Tab . . . 43
User Attributes . . . 44
URLs Tab . . . 45
Starting the SAML Extension Server . . . 45
Testing the Loopback Affiliate Links . . . 45
Conclusion . . . 47

3 Setting Up the iChainSite and eMartian Sample Site Affiliation 49

System Layout . . . 49
Prerequisites . . . 50

Creating the SAML Relationship Between the Sample Sites . . . 50
Creating the Trusted Affiliate Object for eMartian . . . 50
Creating the Trusted Affiliate Object for iChainSite . . . 54
Updating Web Pages . . . 57
iChainSite Intersite Transfer URLs. . . . 57
eMartian Intersite Transfer URLs . . . 59
Troubleshooting the SAML Extension Sample Site Setup . . . 60

A Fine-Tuning the SAML Extension 63

XML Signature Generation and Validation . . . 63
Creating a Signing Key Pair . . . 63
Exporting a Signing Key Pair . . . 68
Setting the PKCS#12 Signature Key on the SAML Extension Server . . . 71
Modifying the SAML Settings in the Directory . . . 72
Exporting the Public Key Certificate . . . 74
Importing Public Key Certificates . . . 75
Configuring SAML to Support SSL Mutual Authentication . . . 79

About This Guide

Introduction

The purpose of this documentation is to help you set up sample sites using the SAML extension for Novell® iChain® software.

The audience for this documentation is network administrators.

This guide is divided into the following sections.

- Chapter 1, "Setting Up the iChainSite SAML Sample Site," on page 9 — Information about how to set up a sample SAML site called iChainSite.
- Chapter 2, "Setting Up the eMartian Sample Site," on page 33 — Information about how to set up a sample SAML site called eMartian.
- Chapter 3, "Setting Up the iChainSite and eMartian Sample Site Affiliation," on page 49 — Infomation about how to set up a SAML affiliation between the iChainSite and eMartian sample sites.
- Appendix A, "Fine-Tuning the SAML Extension," on page 63 — Infomation about how to set up XML signature generation and validation, creating, importing, and exporting Key Pairs, modifying SAML settings in eDirectory, and SSL Mutual Authentication using the SAML back channel.

Documentation Conventions

In Novell documentation, a greater-than symbol (>) is used to separate actions within a step and items in a cross-reference path.

A trademark symbol (®,™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

Documentation Updates

For the latest SAML extension for Novell iChain documentation, including updates to this installation guide, see the online documentation at the Novell documentation Web site (http://www.novell.com/documentation).

1 Setting Up the iChainSite SAML Sample Site

This chapter provides information on how to set up the *iChainSite* SAML sample site. The iChainSite is intended to be a simple example of how to use and deploy SAML-enabled Web applications on iChain using the SAML extension for Novell® iChain® product. The following general topics are covered:

- System Layout
- Overview of the iChainSite Setup
- Starting the SAML Extension Server

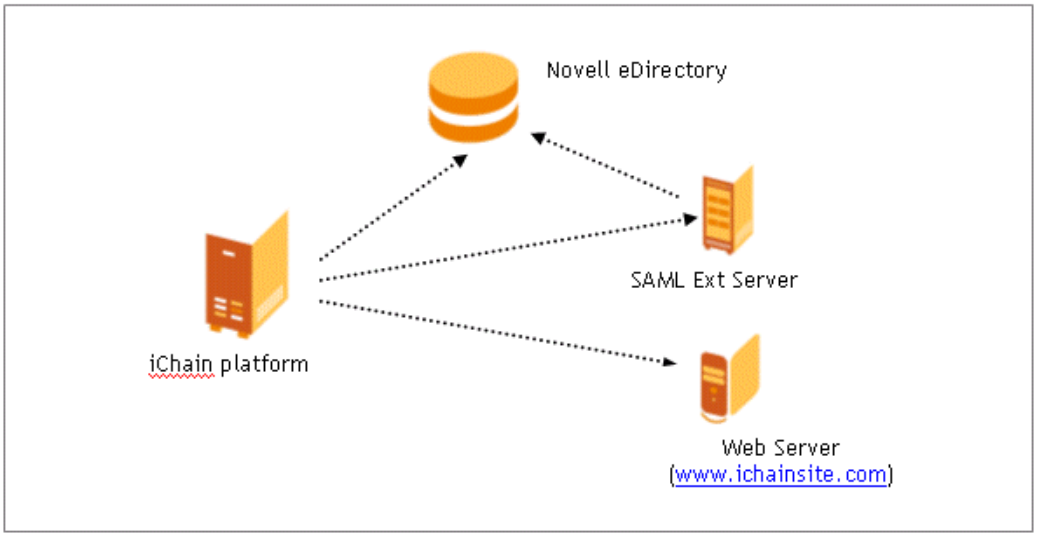
System Layout

The system requires four services that you must run on at least two machines:

- Novell® eDirectory™ for user accounts and iChain configuration
- Novell iChain 2.2 Service Pack 1
- A Web server with a servlet container to run the *iChainSite* sample application
- A Web server with a servlet container to run the SAML extension for Novell iChain product

The following figure shows how these services are connected and related:

Figure 1 System Services: Connections and Relationships



Both the iChain platform and SAML extension service have connections to Novell eDirectory to read configuration information and user attributes. In order to conserve hardware, the Novell eDirectory, Web Server, and SAML extension server can all be run from the same machine

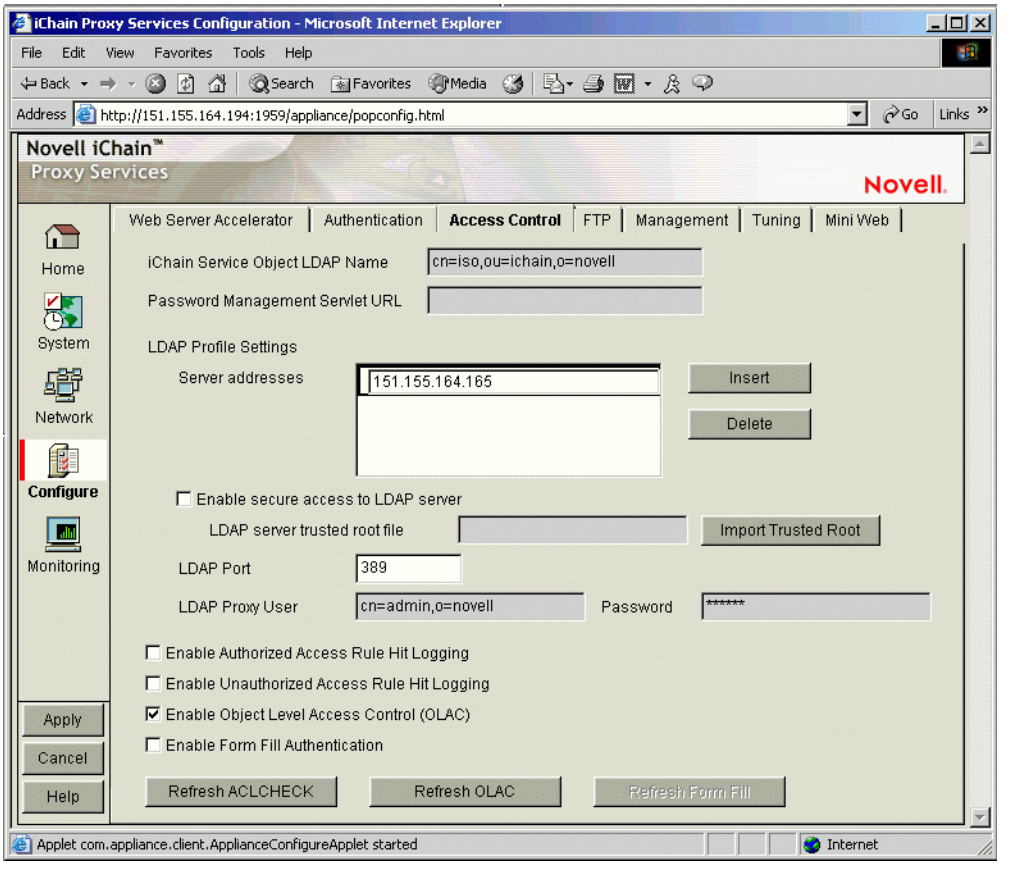
Prerequisites

You should be familiar with the setup and configuration of iChain 2.2. In order to run the iChainSite sample site, the following prerequisites are required:

- iChain 2.2 Service Pack 1
- iChain Authorization server components
- ConsoleOne snap-ins for iChain
- LDAP authentication and OLAC-enabled

Figure 2 shows an iChain installation with the proper authorization and OLAC settings applied. The important points in this figure are the configuration directory settings and the Enable Object Level Access Control (OLAC) setting.

Figure 2 iChain Installation With Correct Settings



For hardware requirements, see the iChain Hardware Guide (http://www.novell.com/products/ichain/hardware22.html).

For additional information and full system requirements for Novell iChain, refer to the Novell iChain Administration Guide, available at the Novell Documentation Web site (http://www.novell.com/documentation/lg/ichain22/index.html).

You can download Novell iChain at Novell Software Downloads (http://download.novell.com).

You can download SAML sample site code at Novell Cool Solutions (http://www.novell.com/coolsolutions/tools/1918.html).

Overview of the iChainSite Setup

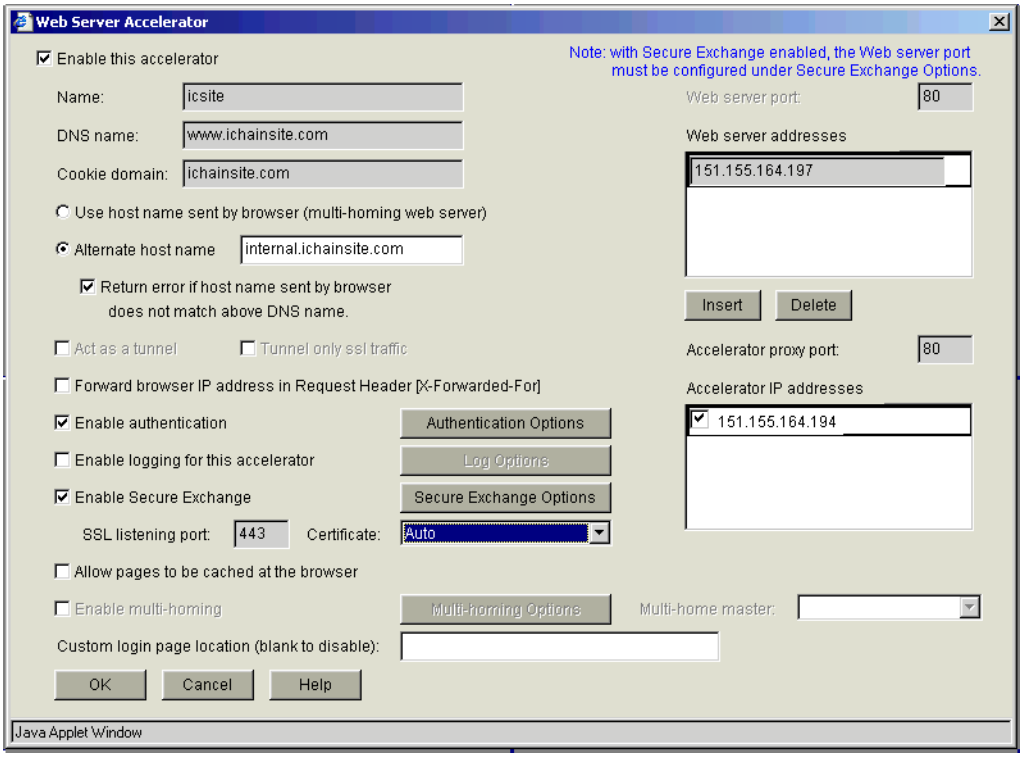
The following list shows the steps you need to complete in order to set up the www.ichainsite.com SAML demo application with the loopback SAML Trusted Affiliate.

1. Configure iChain with the www.ichainsite.com accelerator.
2. Configure the ISO with the www.ichainsite.com protected resources and OLAC parameters.
3. Deploy the www.ichainsite.com sample application.
4. Test the www.ichainsite.com sample application.
5. Install the SAML extension schema and snap-ins.
6. Create SAML extension configuration objects in the directory
   a. Create the loopback SAML Trusted Affiliate site.
7. Install SAML extension server components.
8. Test the SAML extension service.
9. Test the www.ichainsite.com loopback SAML Trusted Affiliate site.

Configuring the iChain Accelerator

In order to run the sample, you must first create a new accelerator using the iChain GUI. You should name this accelerator www.ichainsite.com. Figure 3 shows a basic www.ichainsite.com accelerator configuration:

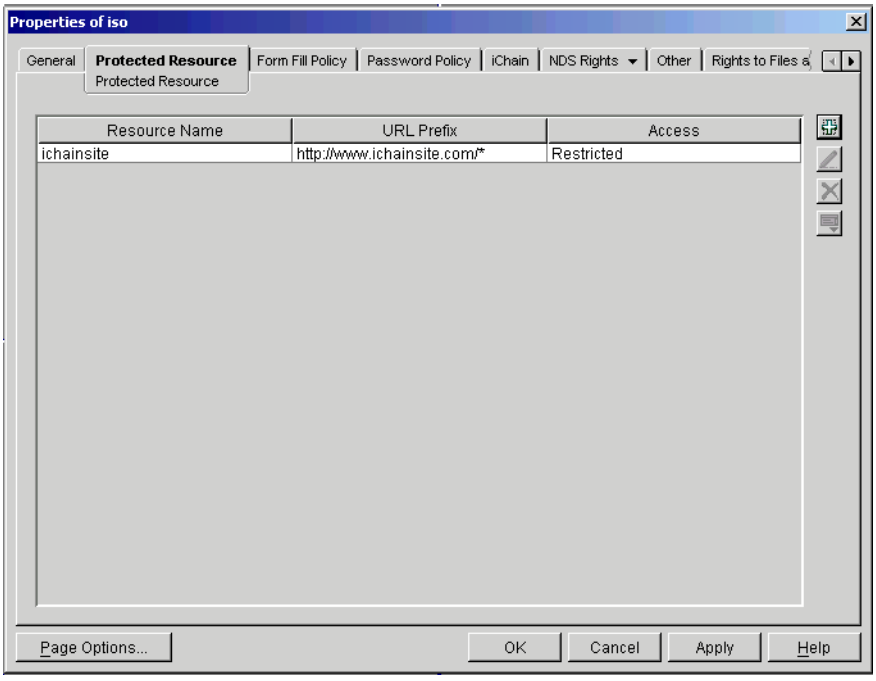
Figure 3 iChainSite Accelerator Configuration



Configuring the iChain Protected Resource and OLAC

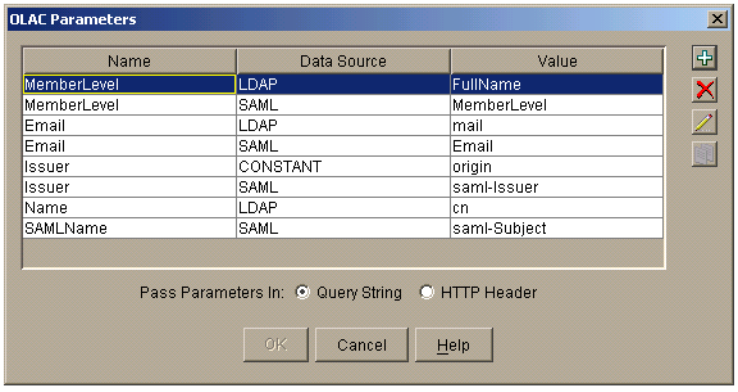
Using ConsoleOne, you must define both a protected resource for the iChainSite application and the OLAC parameters to pass to the application. You make these definitions by selecting the iChainServiceObject you are using in the directory, and selecting the Protected Resources tab. Figure 4 shows the protected resource definitions for the iChainSite application:

Figure 4 Protected Resource Definitions



Next, you must define OLAC parameters for the ichainsite_portal protected resource. Figure 5 shows all of the OLAC parameters required by the iChainSite demo application.

Figure 5 OLAC Parameters



You should make sure that the parameter names (Name) match those in Figure 5 because the iChainSite demo application relies on these names values and if they do not match, the application will not work. The LDAP values names (Value) do not necessarily need to match as long has you have the appropriate LDAP attribute set on the test user objects. (You can use different LDAP values than FullName for MemberLevel and mail for Email.)

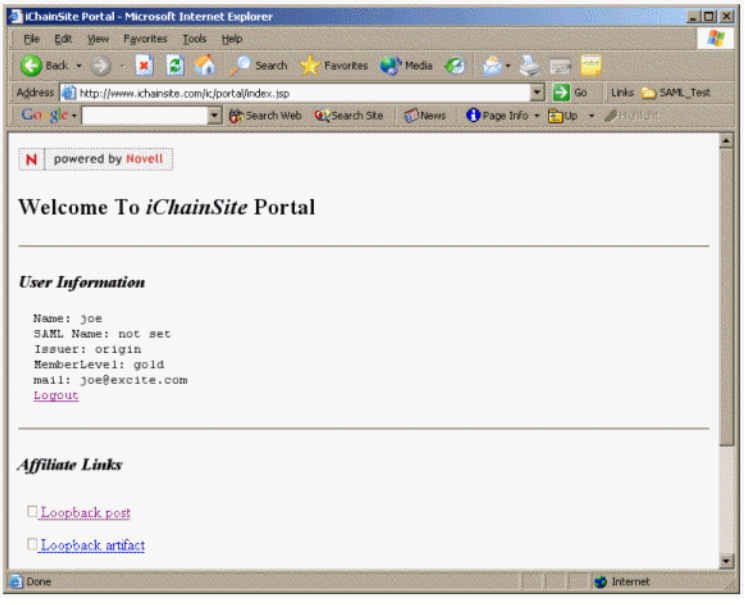
Deploying the iChainSite Sample Application

Because the ichainsite uses simple Java server pages to display its content, you must deploy it into a Java servlet container. If you are running the Apache Tomcat server engine, you can simply take the entire *ichainsite* directory and place it into the *TOMCAT_HOME*/webapps directory. If you

want to deploy the ichain site to the URL extension "ic" (as used in the previous steps), then you must rename the iChainSite directory to ic. See the *SAML Extension for Novell iChain Administration Guide* for details on installing and running Tomcat.

You can test by page by going to your browser and entering http://www.ichainsite.com/ic/portal. After authenticating to iChain, you should see a page that looks the one shown in Figure 6:

Figure 6 Welcome to iChainSite Portal



Make sure that the LDAP properties are being passed through correctly. In this example, the user is logged in as joe.novell, and has an e-mail address of joe@excite.com and FullName (MemberLevel) of gold. See "Testing the Loopback Affiliate Links" on page 30 for information on the Loopback post and Loopback artifact links.

Installing the SAML Extension for Novell iChain Software

Install the SAML extension for Novell iChain components. For details instructions on how to install this software, see the *SAML Extension for Novell iChain Administration Guide*.

The SAML extension installer will install three components:

- SAML extension server
- Snap-ins
- Schema extension

Configuring SAML and ConsoleOne

After you have configured the sample site and have installed the SAML components, you are ready to configure the system.

Creating the SAMLExtensionServer

The first object you need to create is the SAMLExtensionServer. This object contains all of the SAML extension server configuration objects.

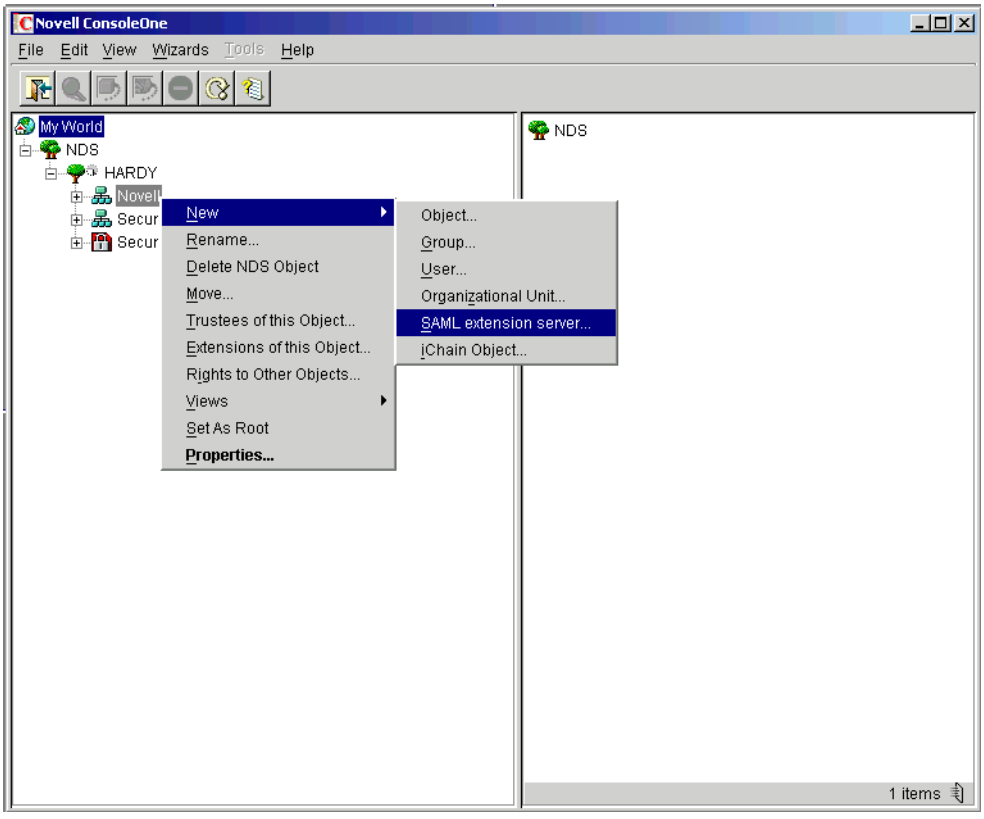
To create an SAMLExtensionServer:

1. Right-click the container where you want to create the SAMLExtensionServer object.

   We recommend that you create this object in the sample container where your iChainServiceObject resides.

2. Select New > Provider Site. See Figure 7.

Figure 7 Provider Site Menu

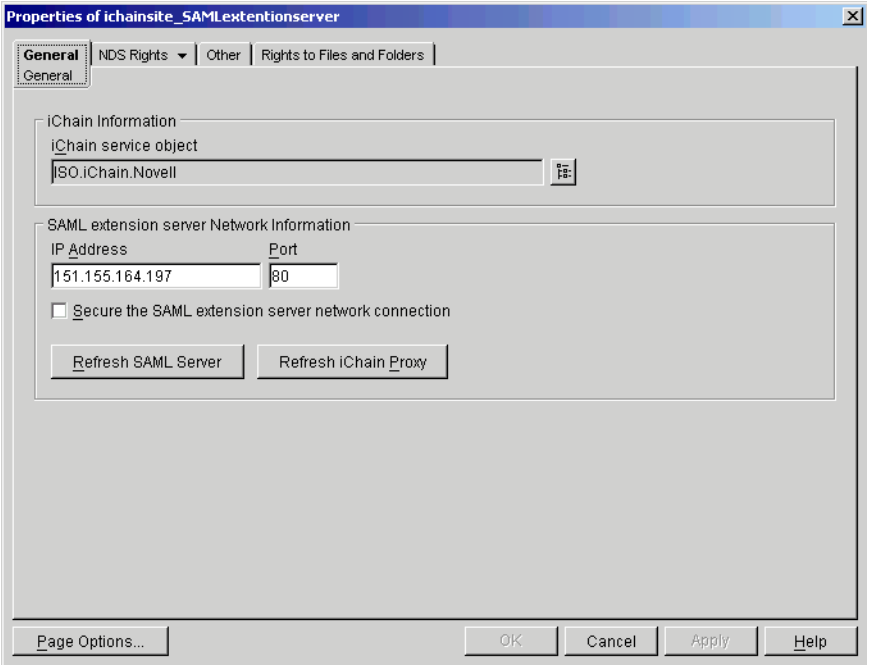


The SAMLExtensionServer object contains configuration information that allows iChain to communicate with the SAML extension server.

3. To set the properties, right-click the SAMLExtensionServer object, then select Properties.

   A Properties page is displayed, as shown in Figure:

Figure 8 Properties of IdentityService



4 Set the following properties:

iChain service object: This field is a back-link to the ISO you want to associate this SAMLExtensionServer with.

Affiliate Connector Network Information: These settings tell iChain where the SAML extension server will be running on the network. It is generally a good idea to have a machine dedicated to running the SAML extension server. Enter the IP address and port the SAML extension Web server will be running on.

Secure the Affiliate Connector Network Connection: This setting allows the use of SSL between the SAML extension server and iChain. See Appendix A, "Fine-Tuning the SAML Extension," on page 63 for details on how to set this up.

Creating the ISO ProviderSiteID Link

After creating the SAMLExtensionServer, you must create a link between it and your iChainServiceObject. The iChainServiceObject must contain an attribute which allows iChain to associate the service object with the appropriate Identity Provider Site object. An attribute called identityProviderSiteRef on the iChainServiceObject must be created. Do the following to create the identityProviderSiteRef:

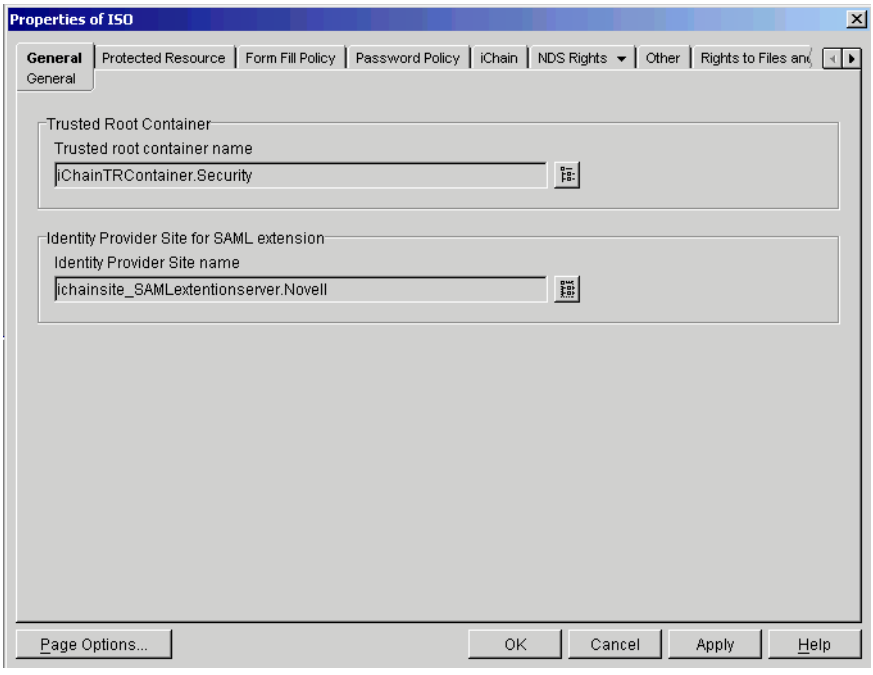
1 Right-click the iChainServiceObject and select Properties.

2 Select Other > Add Attribute.

3 Select the identityProviderSiteRef attribute, then set the value to SAMLExtensionServer.

Figure 9 shows a sample iChainService object with the identityProviderSiteRef attribute set:

Figure 9 iChainService Object with identityProviderSiteRef Attribute Set



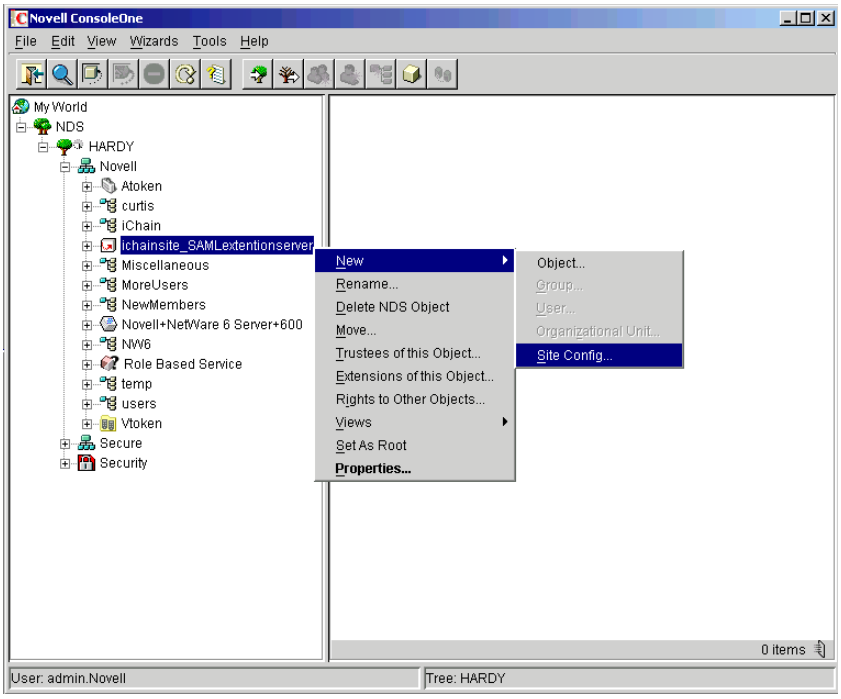
Creating the SAMLSiteConfig Object

Create the SAML Site Config object:

1. Right-click the SAMLExtensionServer object.
2. Select New > Site Config. This launches the samlSiteConfig creation wizard.

   The SAMLSiteConfig object is created as a child object of the SAMLExtensionServer object. Figure 10 shows a new samlSiteConfig object named SAMLConfig:

Figure 10 SAMLConfig Object

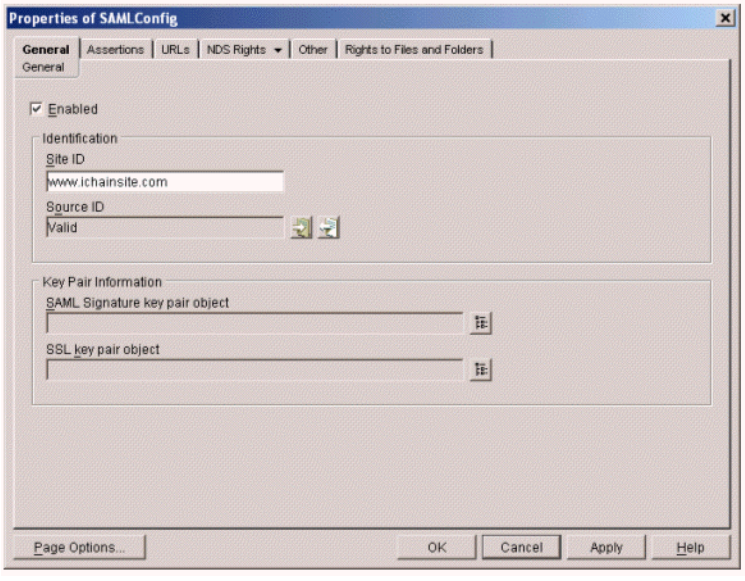


The SAMLSiteConfig object contains the top-level SAML configuration information that is used to identify this site to other SAML sites. The samlSiteConfig Properties page contains the following three main tabs: General, Assertions, and URLs.

General Tab

Figure 11 shows the General tab's fields, followed by a description of the fields.

Figure 11 SAMLSiteConfig Properties: General Tab



Site ID: This is a SAML parameter used to identify this SAML site to your partner SAML sites. In Figure 11, the example shows a SAML system being set up for a company using www.ichainsite.com. In this case, the identifier used by partner sites is www.ichainsite.com.

SourceID: This is a 20-byte value that uniquely identifies the site. This value is used as part of the SAML Browser/Artifact profile. Partner sites use this 20-byte value to determine the origin site of a SAML Artifact. This value can be automatically generated, or you can import a 20-byte hex or base 64 value. Generally, it is sufficient to automatically generate the value. The Valid indicator indicates that a good 20-byte value is set for the configuration.

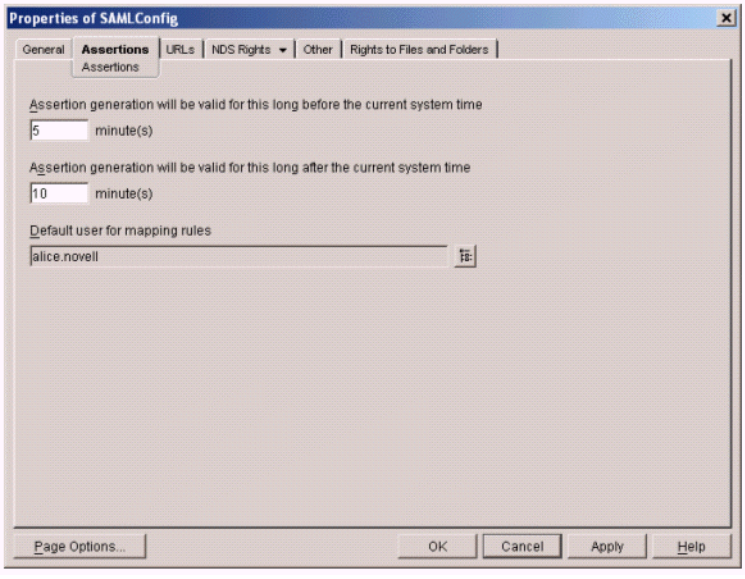
SAML Signature key pair object: This setting allows you to specify a Key Material object in the directory to use to create XML signatures on SAML data. To use it, you must export the Key Material object in password-protected PKCS#12 format and copy it to the SAML extension server. The reference to the key is for convenience to help you keep track of which key should be in use on the SAML extension server.

SSL key pair object: This setting allows you to specify a Key Material object in the directory to use to create outbound SSL connections. To use it, you must export the Key material object in password-protected PKCS#12 format and copy it to the SAML extension server. The reference to the key in the tab is for convenience to help you keep track of which key should be in use on the SAML extension server.

Assertions Tab

Figure 12 shows the Assertion tab's fields, followed by a description of the fields.

Figure 12 SAMLSiteConfig Properties: Assertions Tab



Assertion generation will be valid for this long before the current system time: SAML assertions contain a conditions element. The SAML conditions element contains two timestamp values. One of the timestamps is a "Not Valid Before" timestamp, meaning if the assertion is received before the time specified in the timestamp, it should not be accepted. In some cases, SAML partner systems could have system clocks that are not exactly synchronized. This difference between systems could cause time constraint conditions to fail. Therefore, this value allows you to set the Not Before time condition to be pre-skewed to avoid time synchronization errors. The resulting Not Before time condition is set to the current time minus this value.

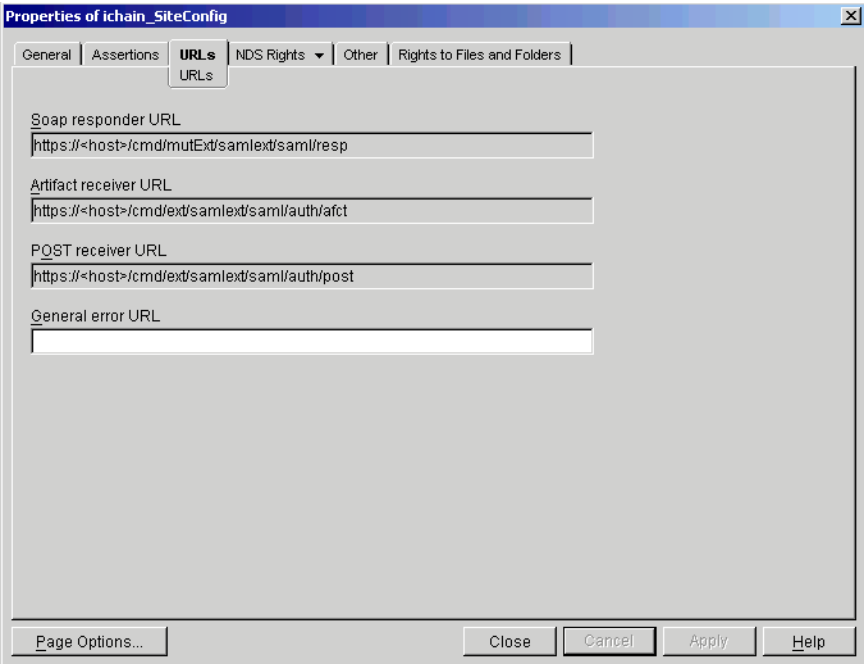
Assertion generation will be valid for this long after the current system time: The SAML conditions element also contains a Not On Or After time stamp. This condition states that an assertion received on or after a specified time must be rejected. SAML assertions that are generated by the system are given a Not On Or After time stamp equal to the current time plus the value of this field.

Default user for mapping rules: This is an optional value that indicates a user object to map an incoming SAML user to in the event that all other user mapping rules fail. Each Trusted Affiliate configuration can contain a set of user mapping rules that define how to map incoming SAML users to identities in the local directory. If a given Trusted Affiliate configuration has no rules, or all of the rules evaluate to false, then this value is used to map the user. If this value is left blank and a user cannot be mapped, the user will not be able to access the system and will instead be shown an error page.

URLs Tab

Figure 13 shows the URL tab's fields, followed by a description of the fields.

Figure 13 SAMLSiteConfig Properties: URLs Tab



SOAP responder URL: This value is used for informational purposes only. You use it for setting up affiliations with other SAML-enabled partner sites. It indicates the URL that partner sites should use to access your SAML SOAP responder service. For the SAML extension product, this URL will always be https://*host*/cmd/mutExt/samlext/saml/resp for mutual authentication required connections, or https://*host*/cmd/ext/samlext/saml/resp if mutual authentication is not required.

Artifact receiver URL: This value is used for informational purposes only. You use it for setting up affiliations with other SAML-enabled partner sites. It indicates the URL that partner sites should send SAML authentication requests to using the SAML Browser/Artifact profile. For the SAML extension product, this URL will always be https://*host*/cmd/ext/samlext/saml/auth/afct.

Post receiver URL: This value is used for informational purposes only. You use it for setting up affiliations with other SAML-enabled partner sites. It indicates the URL that partner sites should send SAML authentication requests to using the SAML Browser/POST profile. For the SAML extension product, this URL will always be https://*host*/cmd/ext/samlext/saml/auth/post.

General Error URL: This value is used in the event that an error occurs while the user is performing an operation on the SAML extension server. The user would then be redirected to the provided URL, which should show a message that indicates the problem and a link back to the host site or resource.

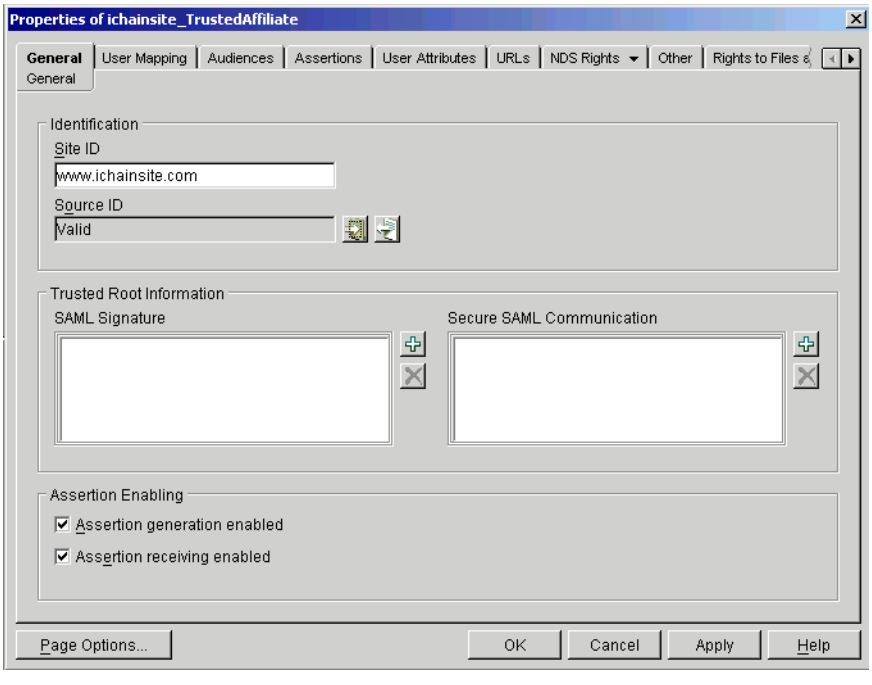
Creating the SAML Trusted Affiliate

After you have defined your site's SAML configuration, you can create SAML trusted partner sites. Each SAML affiliation is configured in a child object of the samlSiteConfig object in the directory. The object used to contain the SAML partner site configuration is a samlTrustedAffiliate object. You can create a new samlTrustedAffiliate object by right-clicking the SAML Config object that you created in "Creating the SAMLSiteConfig Object" on page 17, then selecting New > Trusted Affiliate.

When you set up the SAML extension service for the first time, we recommend that you create a "loopback" SAML Trusted Affiliate site. The loopback site is a copy of your site (for example, (www.ichainsite.com), and can help you determine whether the SAML extension system is working properly. The following screens and directions describe how to create a loopback affiliate for your iChain sample site.

General Tab

Figure 14 iChainSite Properties: General Tab



The General tab's fields are similar to the samlSiteConfig's General tab, but with some key differences, as described:

Site ID: This is the value used to identify this site to other SAML partners. Since you are creating a loopback affiliate, this value is shown in Figure 14 as www.ichainsite.com.

Source ID: This is the same value that you generated in the samlSiteConfig Properties. It is important that the value matches the one used in the samlSiteConfig tab. If you select the auto-generate option, the values will be guaranteed to match.

SAML Signature: Shows links to Trusted Root Certificate object(s) that the SAML extension service will use to validate signed SAML messages.

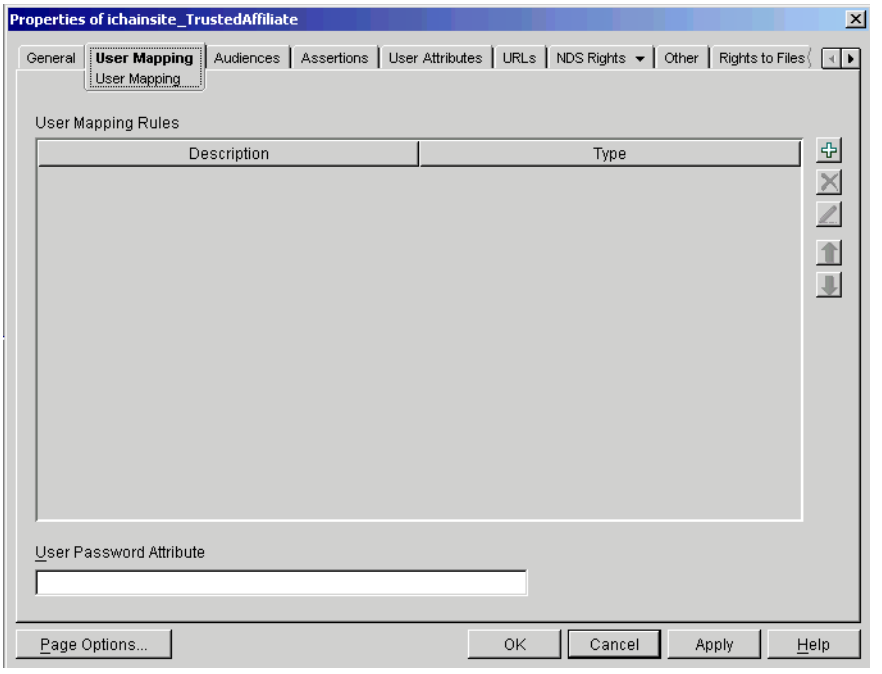
Secure SAML Communication: Shows links to Trusted Root Certificate object(s) that the SAML extension service will use in its SSL trust store when making outbound SSL connections.

Assertion generation enabled: This value indicates whether assertions should be generated for the SAML affiliate site.

Assertion receiving enabled: This is value indicates whether assertions should be accepted from the SAML affiliate site.

User Mapping Tab

Figure 15 iChainSite Properties: User Mapping Tab



The User Mapping tab allows you to define a set of rules that will be evaluated for each incoming user from this SAML partner. The user mapping rules will be evaluated in order and the first successful evaluation will be used as the mapping. There are two different types of user mapping rules:

Static: A static user mapping rule makes a decision based upon the value of attributes in the SAML Assertion. If the condition of a static user mapping rule evaluates to true, a specified user object will be used as the user mapping. This type of rule is ideal for many-to-one user mappings where many users from an affiliate site map to a smaller set of role users at the destination site.

Dynamic: A dynamic user mapping rule performs an LDAP search using values contained the SAML Assertion. An example of this type of user mapping rule would be a search through the users for a given e-mail address or last name. These types of rules are ideal for one-to-one mappings where there is a unique identity at the designation site for each incoming user from the source site.

User Password Attribute: Many applications fronted by iChain still need a user password in order to function properly. This value allows you to specify a SAML attribute value to use as the user's password when the session is created on iChain. We recommend that you avoid using this feature. It is included only for support of legacy applications that require user password for authentication.

User Mapping Rules

As stated in the User Mapping Tab section, you can define dynamic and static types of user mapping rules. The following are examples of dynamic and static user mapping rules used in the iChain sample site loopback affiliation.

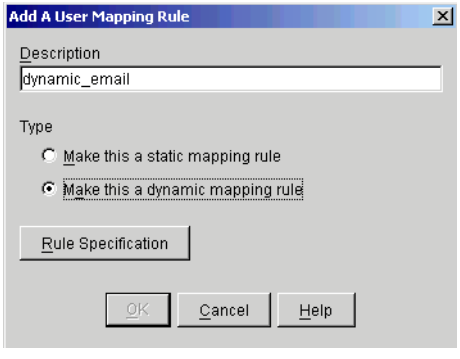
IMPORTANT: Use the dynamic user mapping rule for setting up the iChain site. The static mapping rule is included only as a reference point.

Dynamic Rules: To set a dynamic user mapping rule:

1. At the User Mapping tab, click the plus sign (+) to the right of the User Mapping Rules field.

   The Modify a User Mapping Rule dialog box is displayed, as shown in Figure 16:

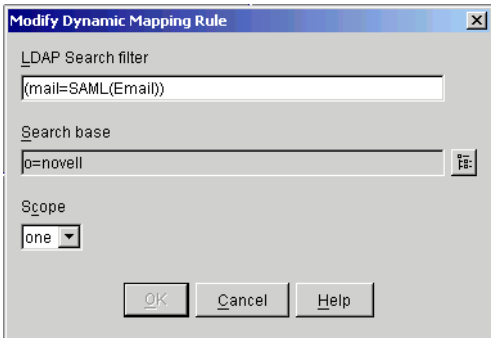
Figure 16 Modify a User Mapping Rule Dialog Box



2. At the Description field, enter an identifiable description to associate with this rule.
3. Choose Make this a dynamic mapping rule.
4. If you want to enter the rule manually, click the Rule Specification button.

   Clicking Rule Specification displays the Modify Dynamic Mapping Rule dialog box, as shown in Figure 17:

Figure 17 Modify Dynamic Mapping Rule Dialog Box



4a Enter the Search filter and Search Base, and choose the Scope.

For a dynamic rule, a search is made for a match between the LDAP mail attribute and the SAML Email attribute sent in the assertion. The Search base value determines the container in which the search will start. Figure 17 shows that for this example, the search will start in the o=novell container. The Scope value determines if a single level (one) or sub-tree (sub) search will be performed. In the example, the search will take place in a single container only.

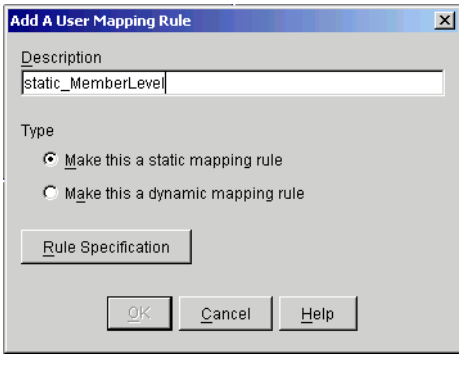
4b Click OK > OK.

Static Rules: To set a static mapping rule:

1 At the User Mapping tab, click the plus sign (+) to the right of the User Mapping Rules field.

The Modify a User Mapping Rule dialog box is displayed, as shown in Figure 18:

Figure 18 Modify a User Mapping Rule Dialog Box



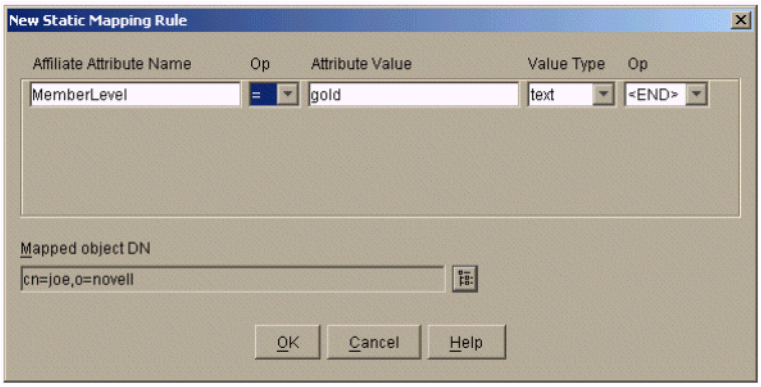
2 At the Description field, enter an identifiable description to associate with this rule.

3 Choose Make this a static mapping rule.

4 If you want to enter the rule manually, click the Rule Specification button.

Clicking Rule Specification displays the New Static Mapping Rule dialog box, as shown in Figure 19:

Figure 19 New Static Mapping Rule Dialog Box

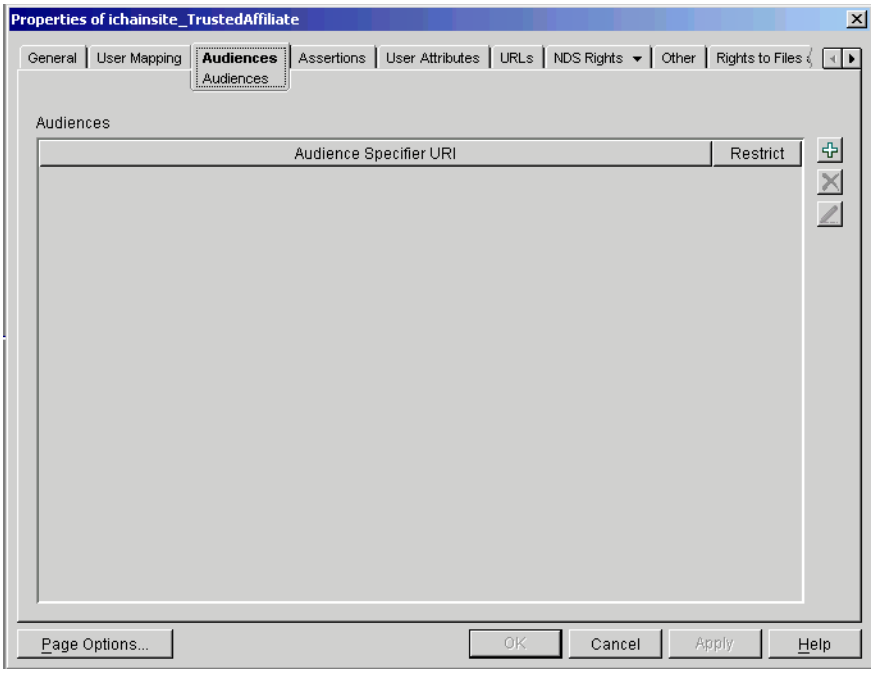


5 Enter the dynamic sample rule with (mail=SAML(Email)). This will perform a one-to-one mapping for all users with matching e-mail addresses.

For the sample application to work properly, you must have users in the directory with valid e-mail addresses set.

Audiences Tab

Figure 20 iChainSite Properties: Audiences Tab

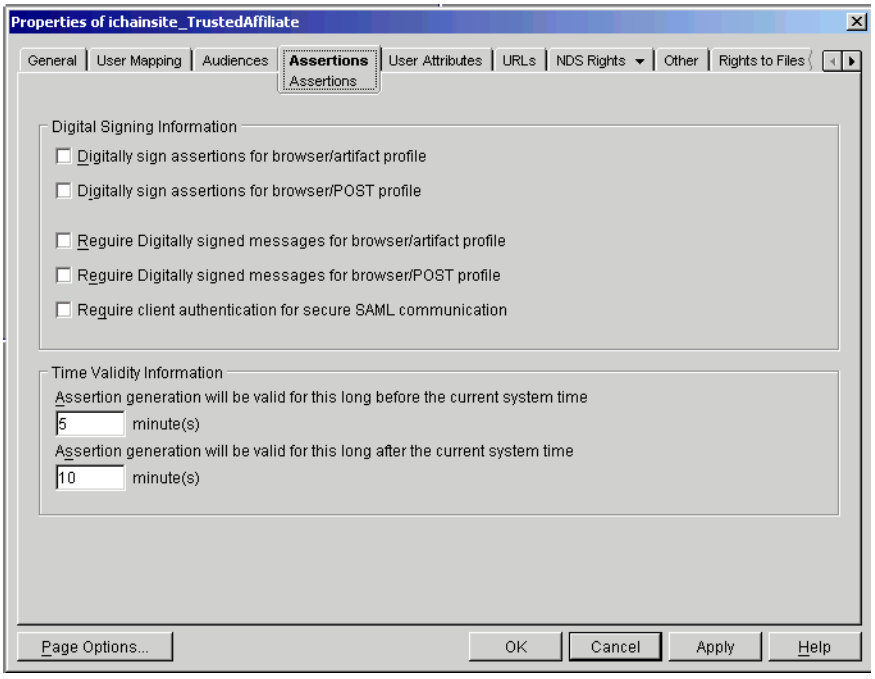


An audience value indicates that a SAML assertion is addressed to one or more specific audiences identified by the provided text value. The value is a URI reference that identifies an intended audience. The URI reference can identify a document that describes the terms and conditions of audience membership. The settings at this tab determine what audiences this affiliate will accept in SAML assertions. If the Restricted value is set, a SAML Audience Restriction condition will be created for generated assertions containing the specified audience value. It is then the responsibility of the receiving party to determine if it wants to accept SAML assertions with the

provided Audience Restriction Condition. When you first set up a SAML partnership, adding audience restriction conditions is likely unnecessary and could add complexity to your system.

Assertions Tab

Figure 21 iChainSite Properties: Assertions Tab



The Assertions tab defines the security constraints associated with SAML assertions generated for this partner site and the SAML assertions incoming from this site. The properties are:

Digitally sign assertions for browser/artifact profile: This value indicates whether SAML Assertions generated for this partner site in the Browser/Artifact profile should include a digital signature. Generally, this value is set to False since in the Browser/Artifact profile, assertions are obtained over a mutually authenticated TLS connection.

Digitally sign assertions for browser/POST profile: This value indicates whether the SAML response generated for this partner site in the Browser/POST profile should include a digital signature. Generally, this value is set to True. The SAML 1.0 bindings specification requires that Browser/POST traffic be signed; however, in low value transaction cases where security is not a great concern, digital signing can be turned off. For initial setup, we recommend that all digital signing and validation be turned off.

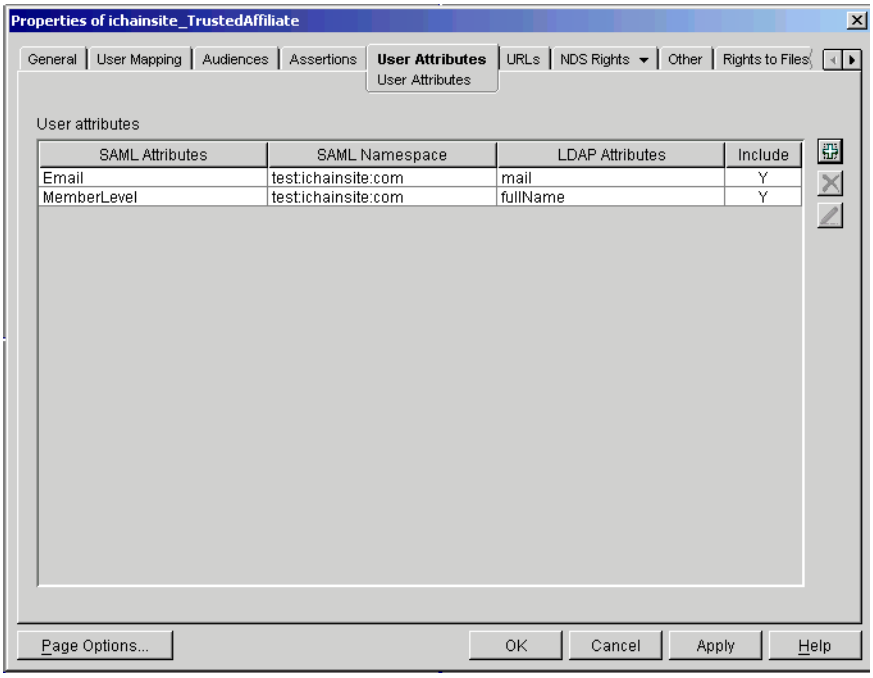
Require Digitally signed messages for browser/artifact profile: This value indicates whether SAML Assertions received from this partner site in the Browser/Artifact should be required to be signed. Generally, this value is set to False. If this value is True and a SAML Assertion is received from this partner under the Browser/Artifact profile that is not signed, the assertion will be rejected.

Require Digitally signed messages for browser/POST profile: This value indicates whether SAML responses received from this partner site in the Browser/POST should be required to be signed. Generally, this value is set to True. If this value is True and a SAML response is received from this partner under the Browser/POST profile that is not signed, the response will be rejected.

Time Validity Information: These two settings allow you to override the defaults set in the samlSiteConfig object. Their definitions and use are identical to those described for the samlSiteConfig. See “Assertions Tab” on page 19 for more information.

User Attributes Tab

Figure 22 iChainSite Properties: User Attributes Tab



The User Attributes tab defines what user attributes will be available to this partner site. These attributes can be set so that they are always sent when authentication assertions are generated for this affiliate. The settings are:

SAML Attributes: This is the name value that will be used to name the attribute in the SAML assertion. Generally, the partner site will indicate to the administrator the names of the attribute it requires in order for users to properly access the partner site. As part of the out-of-band negotiation between this site and the partner site, the names of these attributes will be determined. For this site, the SAML Attribute name can be any text value the partner site requires.

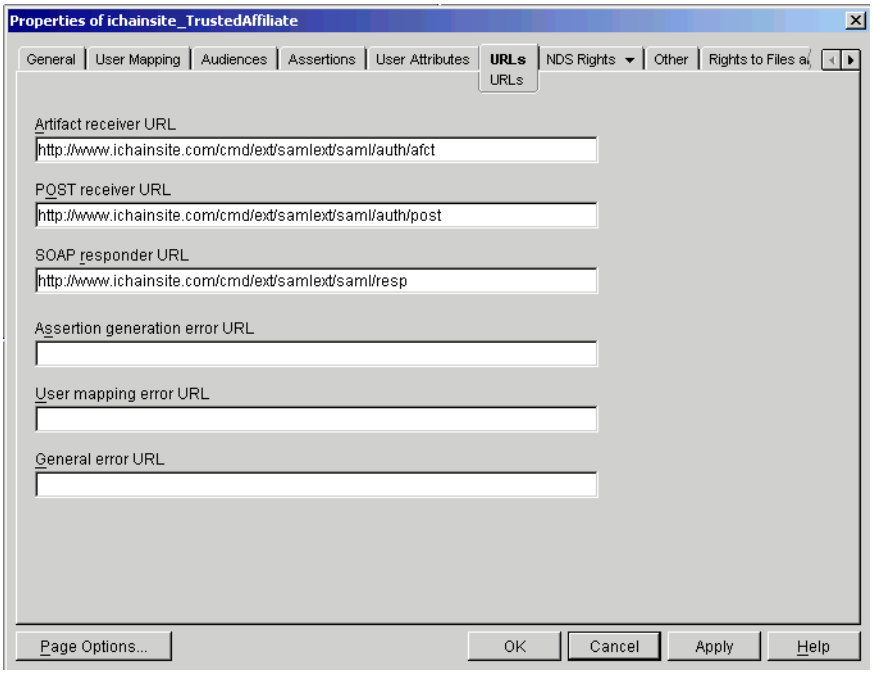
SAML Namespace: This is the namespace value that will be used to name the attribute in the SAML assertion. Since this is also a value that will be requested by the partner site, the SAML Attribute namespace can be any text value the partner site requires.

LDAP Attributes: This is the corresponding LDAP attribute that will be used to as the value of the SAML Attribute.

ICHAIN_PWD: This is a special case LDAP value used to indicate that the user’s password should be sent as a SAML Attribute to the partner site.

Include (Y) or (N): The (Y) value indicates that this attribute will be included in all generated authentication assertions. The (N) value indicates that the attribute will only be obtained when requested via a SAML Attribute Query. If the partner site requests an attribute that is not in this list, the query will be rejected. Thus, you are able to restrict which User attributes you want to expose to the SAML partner sites.

Figure 23 iChainSite Properties: URLs Tab



The URLs tab is the most important page in the Trusted Affiliate configuration. This tab tells the SAML service where to send or re-direct users when a SAML event or error occurs. The properties are:

Artifact receiver URL: The user will be re-directed to this URL when a Browser/Artifact authentication is requested between this site and the partner site.

POST receiver URL: The user will be sent to this URL when a Browser/POST authentication is requested between this site and the partner site.

SOAP responder URL: This URL is the SOAP endpoint that this site will send SAML Artifact requests to when authenticating users from this partner site using the Browser/Artifact profile.

Assertion generation error URL: The user will be sent to this URL in the event that an error occurs during SAML Assertion generation. If there is no value specified in this field, the General error URL or samlSiteConfig Default error URL will be used.

User mapping error URL: The user will be sent to this URL in the event that an error occurs during the SAML user mapping processes. If there is no value specified in this field, the General error URL or samlSiteConfig Default error URL will be used.

General error URL: This user will be sent to this URL in the event that an error occurs during Assertion validation for SAML data received from this affiliate, or if other error URLs are not set. If this value is not set, the samlSiteConfig Default error URL will be used.

Starting the SAML Extension Server

Once the SAML extension objects have been created and configured in eDirectory, you can start the SAML extension server. This section assumes that you have successfully installed the SAML extension server components discussed previously in this chapter.

Deploying the SAML Extension Server Application

You must make sure that the SAML server has been properly deployed. If you installed the SAML extension server components into *TOMCAT_HOME*/webapps, the SAML extension should deploy automatically. If you installed the server components to some other location, you need to modify *TOMCAT_HOME*/conf/server.xml to deploy the application. You can either replace the existing *TOMCAT_HOME*/conf/server.xml file with the server.xml file generated by the installer, or add the following lines to the existing server.xml file:

Figure 24 Modifying the server.xml File

```
<Host ... >

  <Context path="/samlext" docBase="<SAMLEXT_HOME>"/>

</Host ..>
```

The above assumes that you've installed the SAML extension server components to the *SAMLEXT_HOME* directory.

The server.xml file also defines what port the HTTP server will listen on. The server.xml file will contain a section like the following:

Figure 25 Port Definition in server.xml File

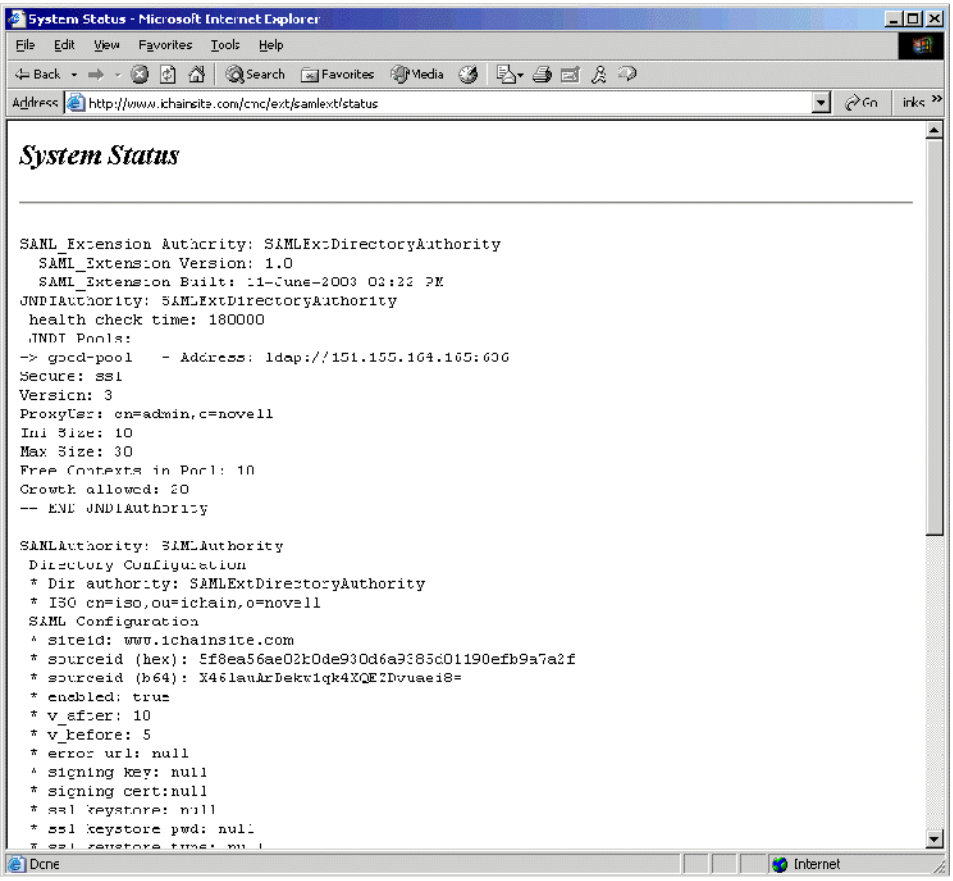
```
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
           port="80"
           minProcessors="5"
           maxProcessors="75"
           enableLookups="true"
           redirectPort="443"
           acceptCount="10"
           debug="0"
           connectionTimeout="20000"
           useURIValidationHack="false" />
```

The port attribute defines what port the HTTP server will run on. This must match the value set as the port on the SAMLExtensionServer object in the directory. Default installations of Tomcat generally are set to listen on port 8080.

Starting the Servlet Container (Tomcat)

There are a number of ways to start the Tomcat Servlet container. The most common way is to run *TOMCAT_HOME*/bin/Catalina.bat with the run command. On Windows operating systems, the Tomcat installer will create a program group in the Windows Start menu. Once Tomcat has been started, you can determine if the SAML extension server is deployed by entering the following URL: http://www.ichainsite.com/cmd/ext/samlext/status. This URL displays information about the running SAML extension server components. If the system is running properly, a page like Figure 26 is displayed:

Figure 26 System Status



This page shows that the system is connected to a single LDAP server at 137.65.159.66:389. If there were other LDAP servers specified, they would be displayed here. If any specified LDAP server is currently down, it will be displayed in the Bad JNDI Pools list. There is also an entry for the SAMLAuthority. This is the component that loads the SAML extension server configuration from the directory. You can validate that the configuration information in the directory matches the information on this screen.

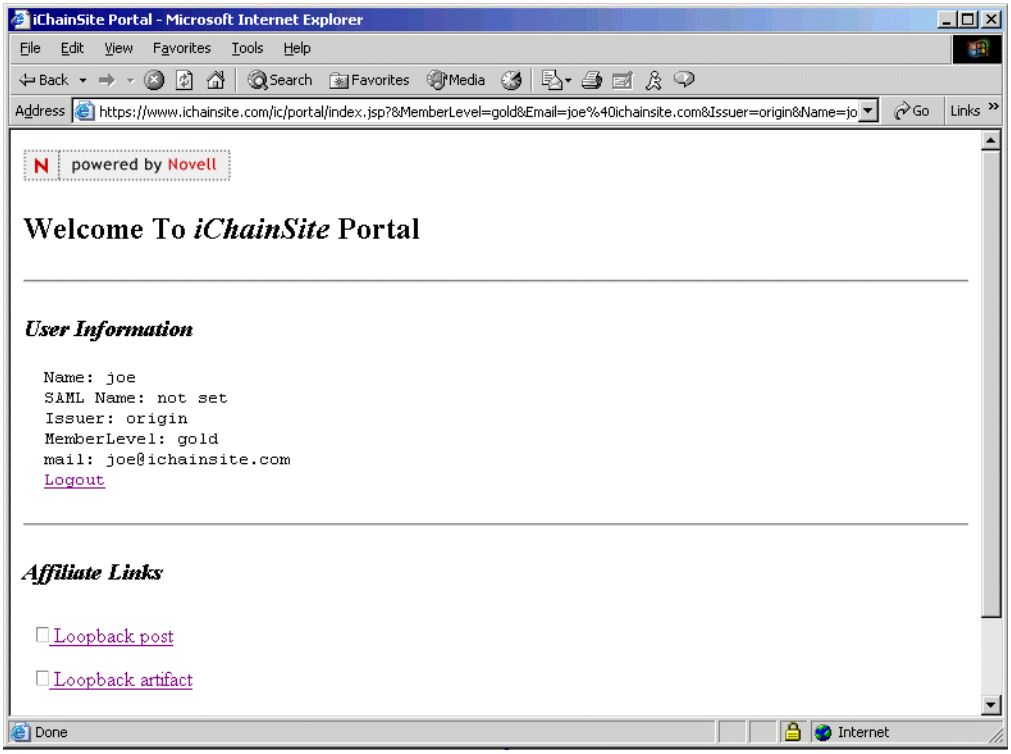
If the SAML extension server was unable to load, you might receive a 404 Page Not Found or similar error. This happens if the SAML extension server was not properly deployed. Check the *SAML Extension for Novell iChain Administration Guide* and Tomcat logs for information on why the application was not deployed.

If the SAML extension server was able to load but encountered errors, this page should tell you generally what went wrong. A common problem is that the configuration LDAP server could not be accessed or the required configuration details were not found in the directory.

Testing the Loopback Affiliate Links

Once the SAML extension server has been configured and deployed, you can begin testing it. Access the main iChain Site sample application by entering the following URL: http://www.ichainsite.com/ic/portal. You should be prompted to authenticate to iChain. After successful login, you should receive a page like the one shown in Figure 27:

Figure 27 iChainSite: Successful Login



The two links of interest here are Loopback post and Loopback artifact. The post link will perform a SAML single sign-on operation using the SAML Browser/POST profile. The artifact link will perform a SAML single sign-on operation using the SAML Browser/Artifact profile. Figure 28 shows the HTML source of the two links:

Figure 28 HTML Source of Loopback Links

```
<!-- POST LINK -->
<A
href="https://www.ichainsite.com/cmd/ext/samlext/saml/gen/post?AID
=www.ichainsite.com&TARGET=http://www.ichainsite.com/ic/portal">
Loopback post</a>

<!-- Artifact LINK -->
<A
href="https://www.ichainsite.com/cmd/ext/samlext/saml/gen/afct?AID
=www.ichainsite.com&TARGET=http://www.ichainsite.com/ic/portal">
Loopback artifact</a>
```

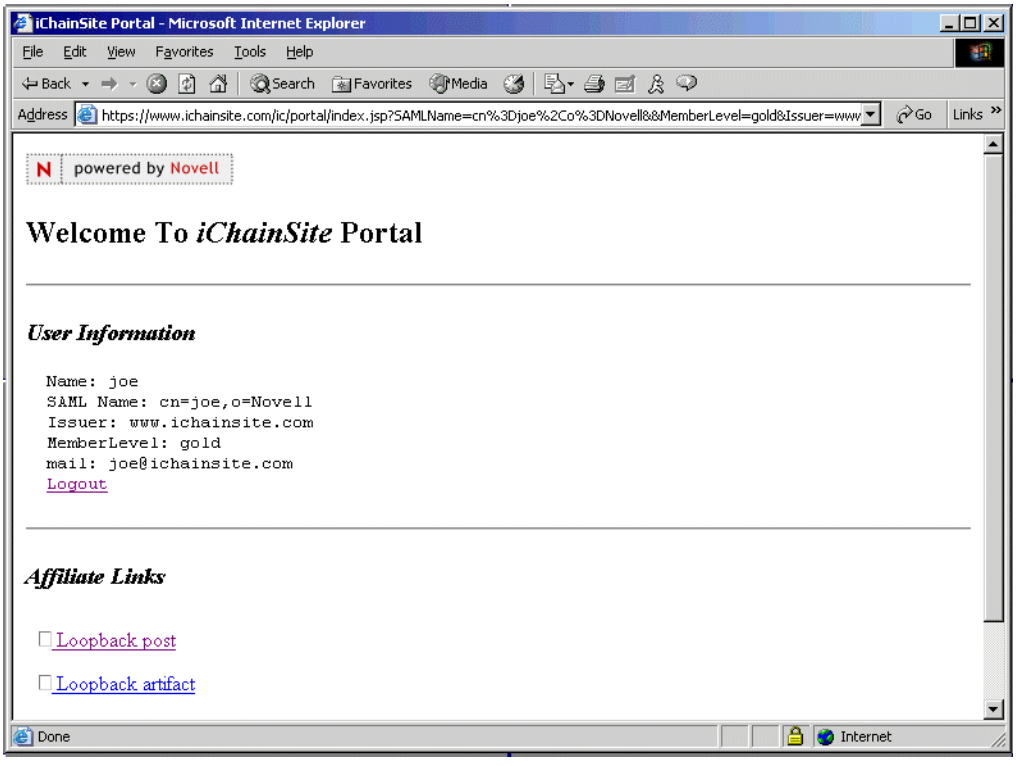
Note that both links point back to the iChainSite host with the /cmd/ext URL switch. This indicates that the traffic is intended for the SAML extension server. The URLs https://www.ichainsite.com/cmd/ext/saml/gen/post and https://www.ichainsite.com/cmd/ext/saml/gen/afct are called Intersite Transfer URLs. They indicate that the user wants to perform a SAML single sign-on operation to a partner site, and they are used to securely send the user to that site. There are two critical URL parameters included on each URL:

Aid: This is the affiliate ID that the user wants to access. In the example above, a loopback affiliation is being performed, so the Aid is www.ichainsite.com. This identifier must match the SiteID value stored in the directory for the partner site.

Target: The target is a URL at the destination site that the user wants to access after single sign-on has been completed. The example above indicates that the user wants to access the /ic/portal resource at www.ichainsite.com.

Clicking either of the Loopback links will cause the SAML extension server to both generate and validate a single sign-on assertion for your user. Figure 29 shows the page that is displayed if you click the POST link:

Figure 29 POST Loopback Link



There is one important difference between this page and the original: The Issuer value is now set to www.ichainsite.com. This indicates that instead of being from the origin site, you have accessed this page from an affiliate www.ichainsite.com. This means you have successfully performed SAML single sign-on.

For extended debug information about what is happening during the SAML single sign-on processes, you can check the console of the servlet engine (Tomcat) running the SAML extension server. By default, debug logging is sent to this console as well as a file, wsslog.xml. The wsslog.xml file should be located at *TOMCAT_HOME*/bin/wsslog.xml.

Conclusion

You have now concluded the setup portion of the stand-alone iChainSite SAML sample site. This documentation is intended to provide details of what is involved in the setup and deployment of the SAML extension for Novell iChain product. For more additional information, including installation and general administration, see the *SAML Extension for Novell iChain Administration Guide*.

2 Setting Up the eMartian Sample Site

This chapter provides information on how to set up the *eMartian* SAML sample site. The eMartian site is intended to be a simple example of how to use and deploy SAML-enabled Web applications on iChain using the SAML extension for Novell® iChain® product. Before reviewing this chapter, you should complete a thorough review of Chapter 1, "Setting Up the iChainSite SAML Sample Site," on page 9. The following general topics are covered:

- Prerequisites
- Setting Up the eMartian Site
- Configuring SAML and ConsoleOne
- Starting the SAML Extension Server
- Testing the Loopback Affiliate Links

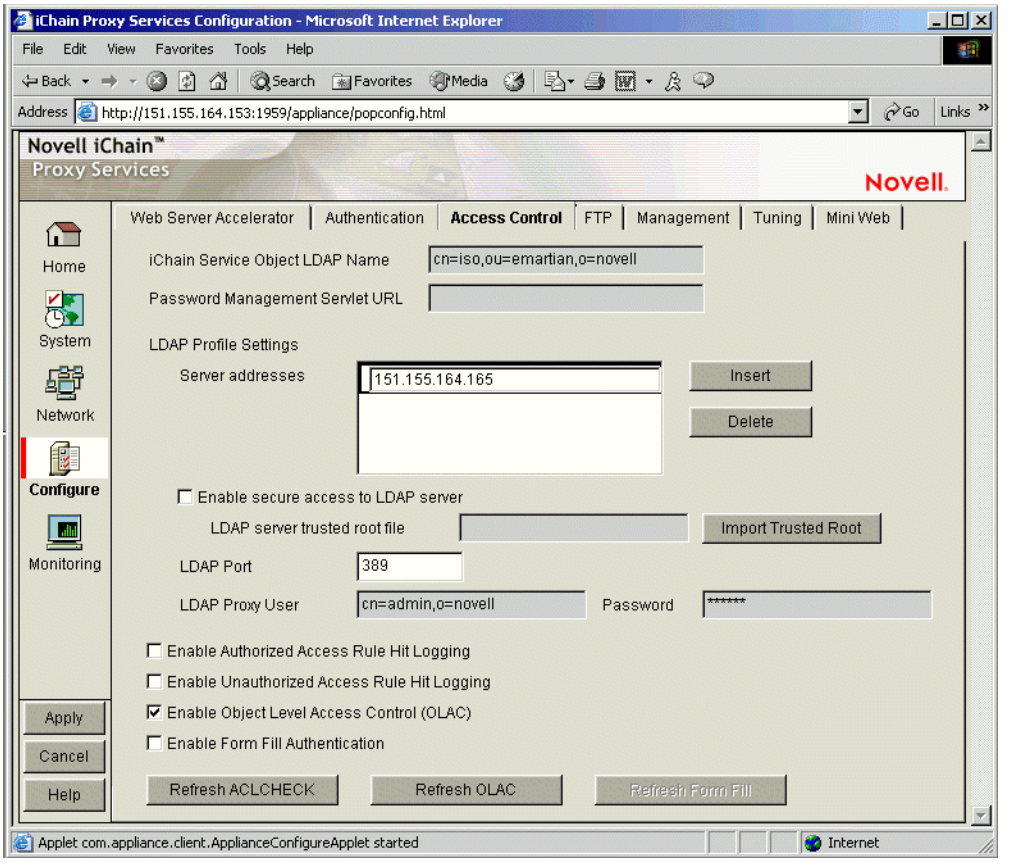
Prerequisites

You should be familiar with the setup and configuration of iChain 2.2. In order to run the eMartian sample site, the following prerequisites are required:

- iChain 2.2 Service Pack 1
- iChain Authorization server components
- ConsoleOne snap-ins for iChain
- LDAP authentication and OLAC-enabled

Figure 30 shows an iChain installation with the proper authorization and OLAC settings applied. The important points in this figure are the configuration directory settings and the Enable Object Level Access Control (OLAC) setting.

Figure 30 iChain Installation With Correct Settings



For hardware requirements, see the iChain Hardware Guide (http://www.novell.com/products/ichain/hardware22.html).

For additional information and full system requirements for Novell iChain, refer to the Novell iChain Administration Guide, available at the Novell Documentation Web site (http://www.novell.com/documentation/lg/ichain22/index.html).

You can download Novell iChain at Novell Software Downloads (http://download.novell.com).

Setting Up the eMartian Site

To setup the www.emartian.com SAML demo application with the loopback SAML Trusted Affiliate, you must complete the following general steps:

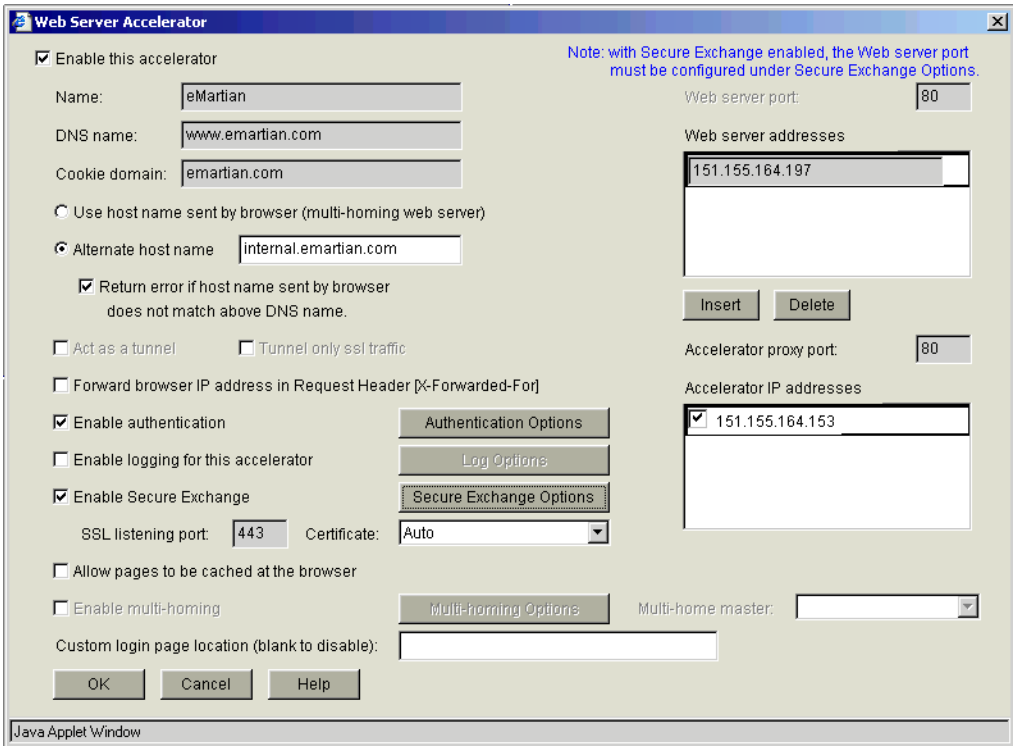
1. Configure iChain with the www.emartian.com accelerator.
2. Configure the ISO with the www.emartian.com protected resources and OLAC parameters.
3. Deploy the www.emartian.com sample application.
4. Test the www.emartian.com sample application.
5. Install the SAML extension schema and snap-ins.
6. Create SAML extension configuration objects in the directory.
    a. Create the loopback SAML Trusted Affiliate site.

7. Install SAML extension server components.
8. Test the SAML extension service.

Configuring the iChain Accelerator

In order to run the sample, you must first create a new accelerator using the iChain GUI. You should name the accelerator www.emartian.com. Figure 31 shows a basic www.emartian.com accelerator configuration:

Figure 31 eMartian Accelerator Configuration



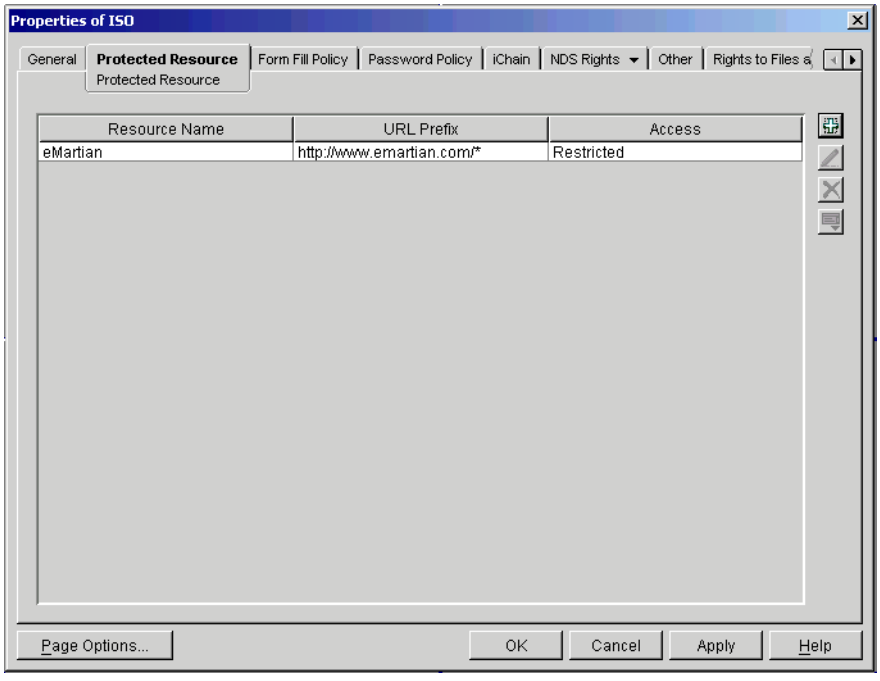
Defining the iChain Protected Resource and OLAC

Using ConsoleOne, you must define both a protected resource for the eMartian application, as well as the OLAC parameters to pass to the application. To do these operations:

1. Select the iChainServiceObject you are using in the directory.
2. Click the Protected Resources tab.

   Figure 32 shows the protected resource definitions for the eMartian application:

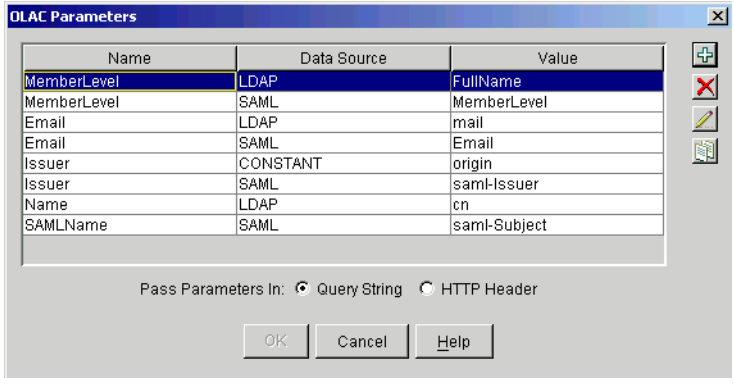
Figure 32 Protected Resource Definitions for the eMartian Application



3 Define OLAC parameters for the eMartian_application protected resource.

Figure 33 shows all of the OLAC parameters required by the eMartian demo application:

Figure 33 OLAC Parameters Required by the eMartian Application



It is important that the parameter names (Name) match those in Figure 33. The eMartian demo application relies on these name values, and if they are different, the application will not work. The LDAP values names (Value) do not necessarily need to match as long has you have the appropriate LDAP attribute set on the test user objects. (You can use different LDAP values than FullName for MemberLevel and mail for Email.)

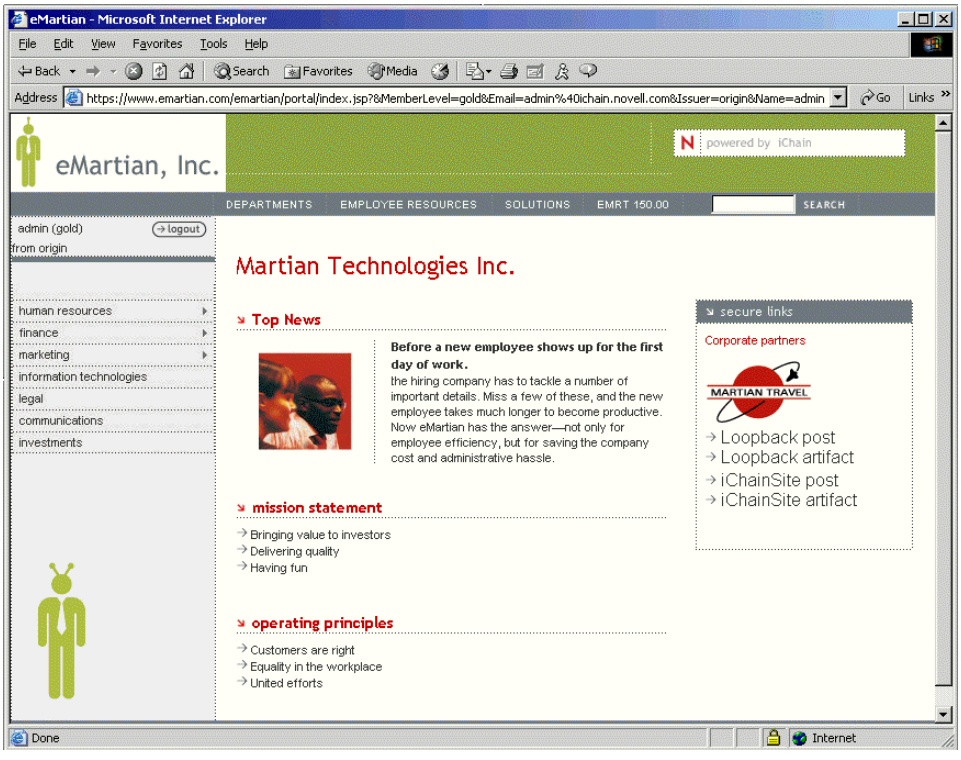
Deploying the eMartian Sample Application

The eMartian application uses simple Java server pages to display its content. Thus, you must deploy it into a Java servlet container. If you are running the Apache Tomcat server engine, you can simply take the entire *eMartian* directory and place it into the *TOMCAT_HOME*/webapps

directory. See Chapter 3 for more details on installing and configuring Tomcat. After deploying the application, enter the following URL to access the eMartian portal:

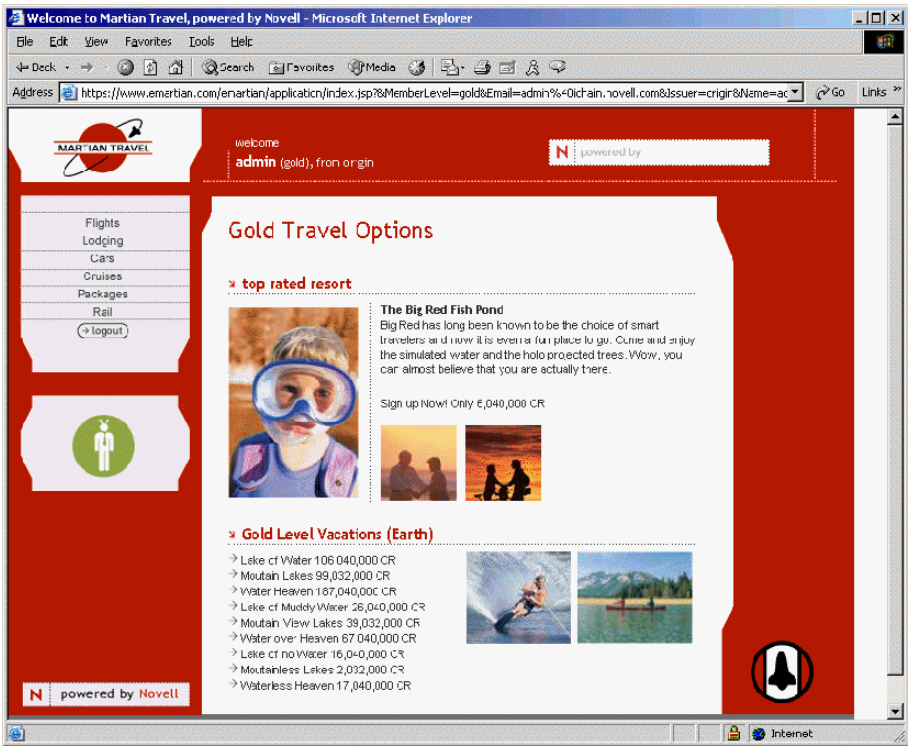
http://www.emartian.com/emartian. After you authenticate to iChain, a page as shown in Figure 34 should be displayed:

Figure 34 Authentication to iChain



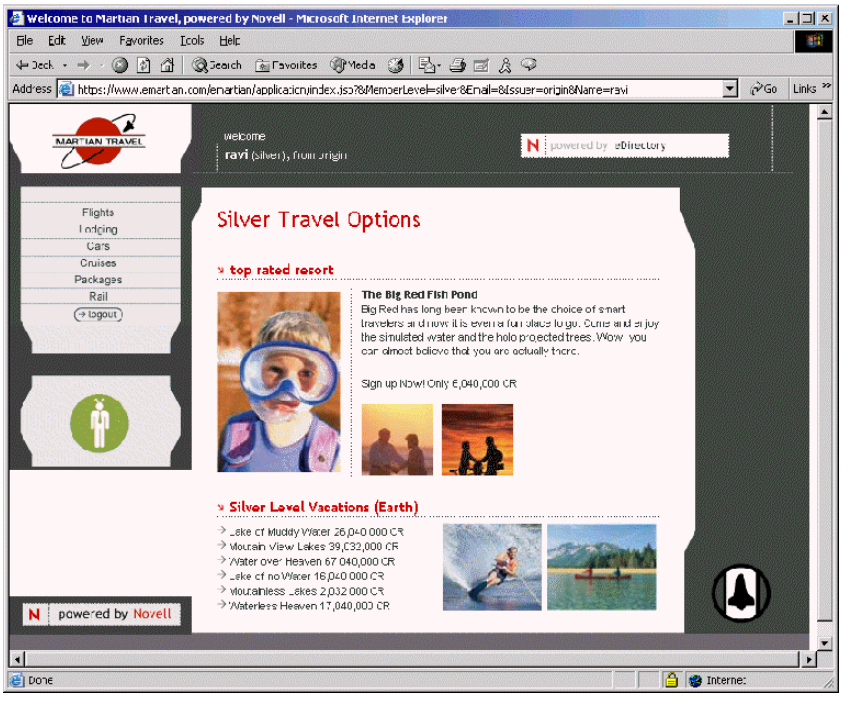
You should verify that the LDAP properties are being passed correctly. In the example shown in Figure 34, the user is logged in as Admin and has a FullName (MemberLevel) of gold. By selecting the Martian Travel link on the right-hand side of the page, the eMartian application is accessed. A page should display as shown in Figure 35:

Figure 35 Accessing the eMartian Application



You can again validate that the proper OLAC attributes are being sent. Different content is displayed, depending upon the MemberLevel of the user accessing the application. If you were to access the eMartian application with a user whose MemberLevel (FullName) were set to silver, you should see a page as shown in Figure 36:

Figure 36 MemberLevel Set to Silver



As shown in Figure 36, a user named r_ravi accessed this page. R_ravi has a MemberLevel of silver.

Installing the SAML Extension for Novell iChain Software

Install the SAML extension for Novell iChain components. For details instructions on how to install this software, see the *SAML Extension for Novell iChain Administration Guide*.

The SAML extension installer will install three components:

- SAML extension server
- Snap-ins
- Schema extension

Configuring SAML and ConsoleOne

After you have configured the sample site and have installed the SAML components, you are ready to configure the system.

Creating the SAMLExtensionServer

Follow the same steps you used for the iChainSite setup. For details on how to create and configure the SAMLExtensionServer, see "Creating the SAMLExtensionServer" on page 15.

Creating the ISO ProviderSiteID Link

For details on how to create the ISO ProviderSiteID link, see"Creating the ISO ProviderSiteID Link" on page 16.

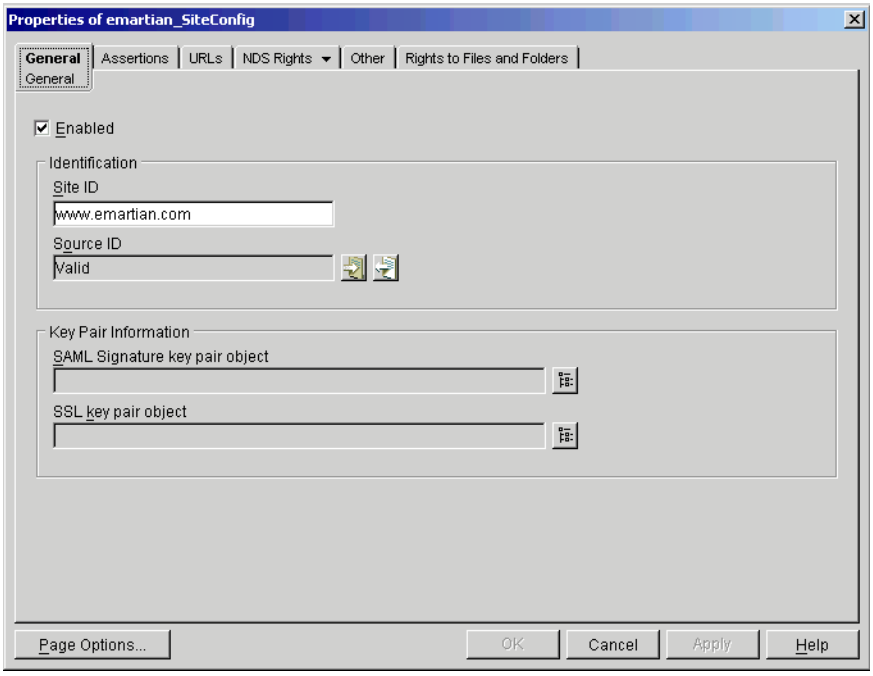
Creating the SAMLSiteConfig Object

Create a samlSiteConfig object by following the same steps you used for the iChainSite setup. For details on how to create the samlSiteConfig object, see "Creating the SAMLSiteConfig Object" on page 17. The setup parameters in this case will be different than the ones you used for the iChainSite. Figure 37, show the appropriate values set for the Properties pages:

General Tab

For details on the General tab, see "General Tab" on page 18. Figure 37 shows the values that should be used for the eMartian site:

Figure 37 eMartian SAMLConfig: General Tab

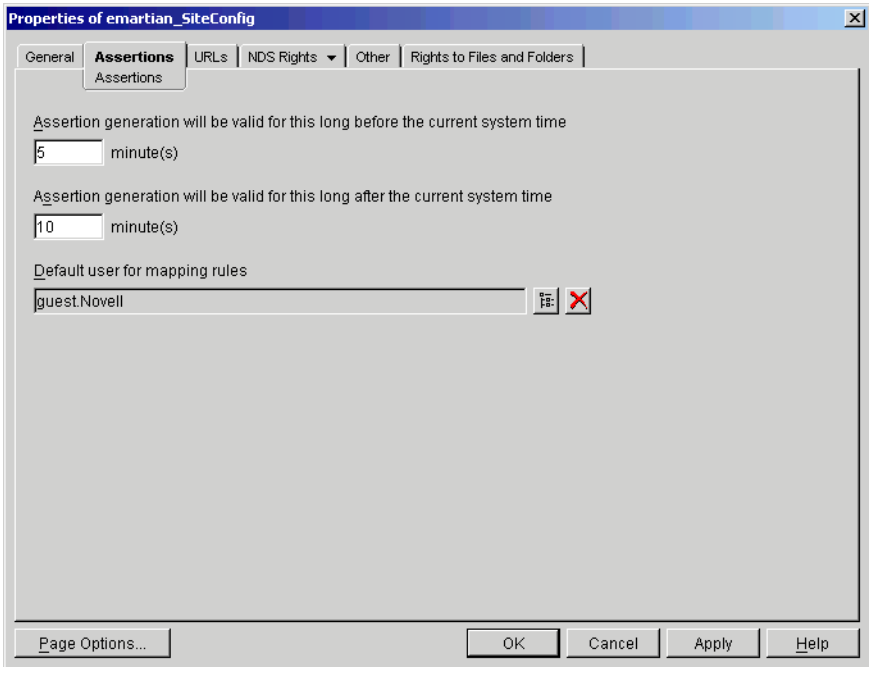


NOTE: The Key Pair Information fields should be left blank at this point. See Appendix A, “Fine-Tuning the SAML Extension,” on page 63 for information on Key Pair management.

Assertions Tab

For details on the Assertions tab, see “Assertions Tab” on page 19. Figure 38 shows the values that you could use for the eMartian setup:

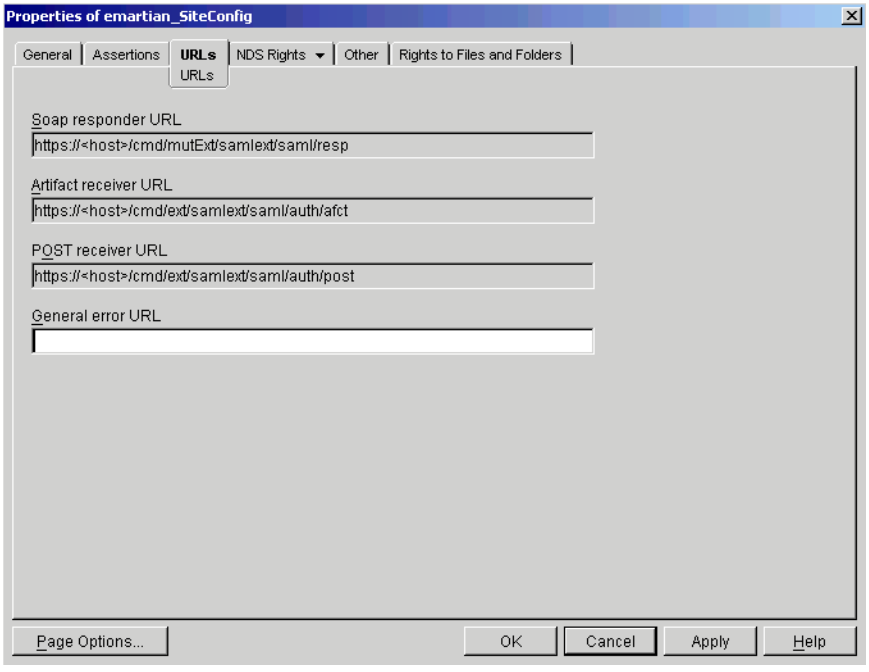
Figure 38 eMartian SAMLConfig: Assertions Tab



URLs Tab

For details on the URLs tab, see “URLs Tab” on page 20. Figure 39 shows the values that you could use for the eMartian application:

Figure 39 eMartian SAMLConfig: URLs Tab



Creating SAML Trusted Affiliates

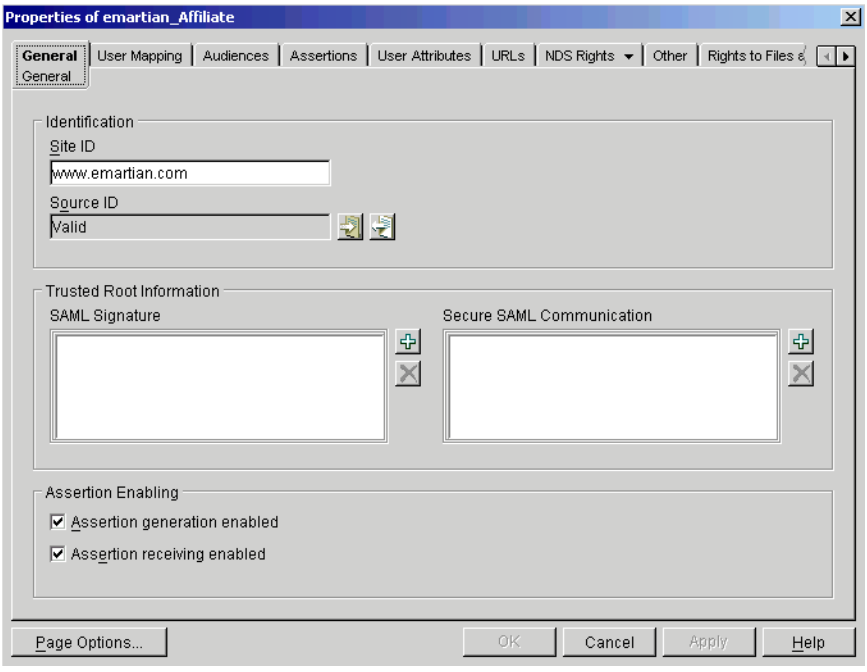
After you have defined your site’s SAML configuration, you can create SAML trusted partner sites. Each SAML affiliation is configured in a child object of the samlSiteConfig object in the directory. The object used to contain the SAML partner site configuration is a samlTrustedAffiliate object. You can create a new samlTrustedAffiliate object by right-clicking the SAML Config object that you created in “Creating the SAMLSiteConfig Object” on page 39, then selecting New > Trusted Affiliate.

When you set up the SAML extension service for the first time, we recommend that you create a loopback SAML Trusted Affiliate site. The loopback site is a copy of your site (for example, (www.emartian.com), and can help you determine whether the SAML extension system is working properly. The following screens and directions describe how to create a loopback affiliate for your iChain sample site.

General Tab

For details on the General tab properties, see “General Tab” on page 21. Figure 40 shows the values to use for the eMartian site:

Figure 40 General Tab Properties



NOTE: The Trusted Root Information fields can be left blank. See Appendix A, "Fine-Tuning the SAML Extension," on page 63 for more information on Trusted Roots.

User Mapping

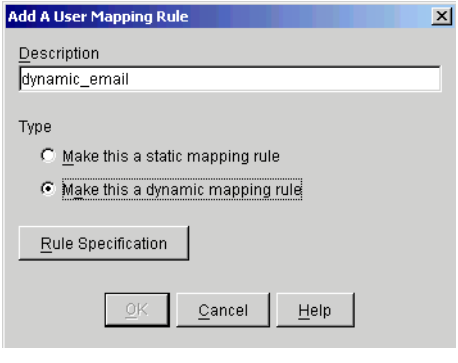
For details on User Mapping, see "User Mapping Tab" on page 22. In order for loopback to work properly, a single dynamic user mapping rule should be used.

Dynamic Rules: To set a dynamic user mapping rule:

1. At the User Mapping tab, click the plus sign (+) to the right of the User Mapping Rules field.

   The Modify a User Mapping Rule dialog box is displayed, as shown in Figure 41:

Figure 41 Modify a User Mapping Rule Dialog Box

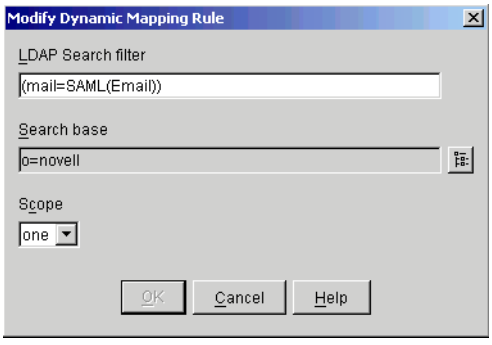


2. At the Description field, enter an identifiable description to associate with this rule.
3. Choose Make this a dynamic mapping rule.

4 If you want to enter the rule manually, click the Rule Specification button.

Clicking Rule Specification displays the Modify Dynamic Mapping Rule dialog box, as shown in Figure 42:

Figure 42 Modify Dynamic Mapping Rule Dialog Box



4a Enter the Search filter and Search Base, and choose the Scope.

In this dynamic rule, a search is made for a match between the LDAP mail attribute and the SAML Email attribute sent in the assertion. The Search base value determines the container in which the search will start. Figure 42 shows that for this example, the search will start in the o=novell container. The Scope value determines if a single level (one) or sub-tree (sub) search will be performed. In the example, the search will take place in a single container only.

4b Click OK > OK.

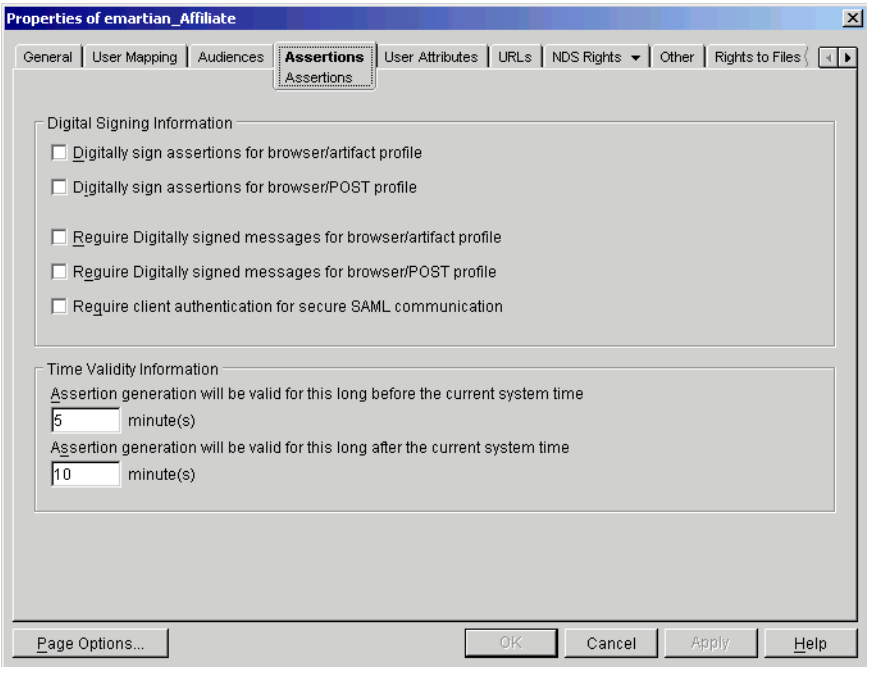
Audiences Tab

For details on the Audiences tab, see "Audiences Tab" on page 25. You can leave this tab unaltered for now.

Assertions Tab

For details on the Assertions tab, see "Assertions Tab" on page 26. You should configure this page the same way you did for the iChainSite sample. Figure 43 shows the iChainSite Assertions tab.

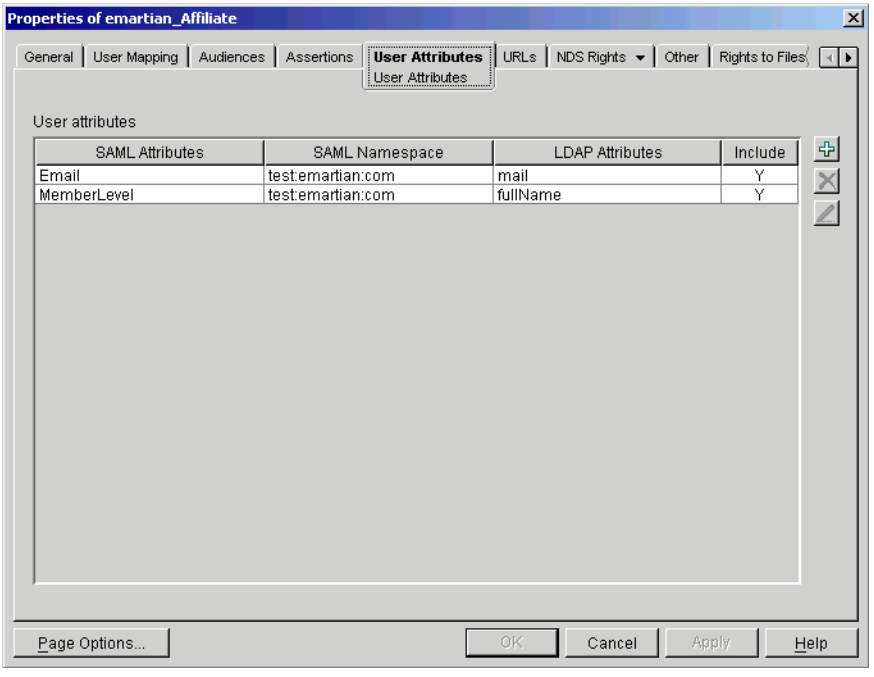
Figure 43 iChainSite Loopback: Assertions Tab



User Attributes

For details on the User Attributes tab, see "User Attributes Tab" on page 27. The eMartian sample site should be configured the same as you did for the iChainSite. Figure 44 shows the iChainSite sample User Attribute tab:

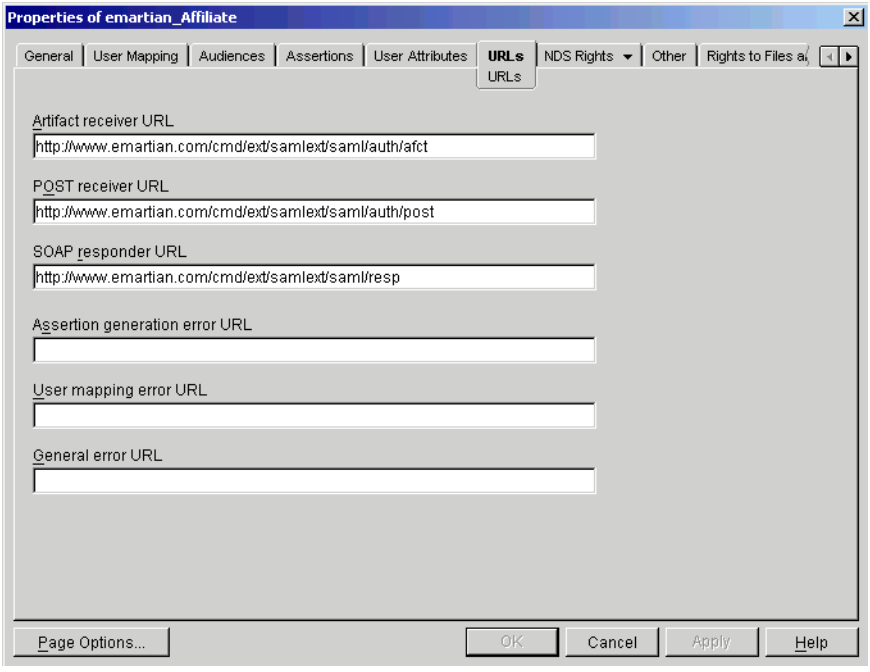
Figure 44 iChainSite Properties: User Attributes Tab



URLs Tab

For details on the URLs tab, see “URLs Tab” on page 28. The URLs to use for the eMartian sample site are shown in Figure 45:

Figure 45 iChainSite Loopback: URLs Tab



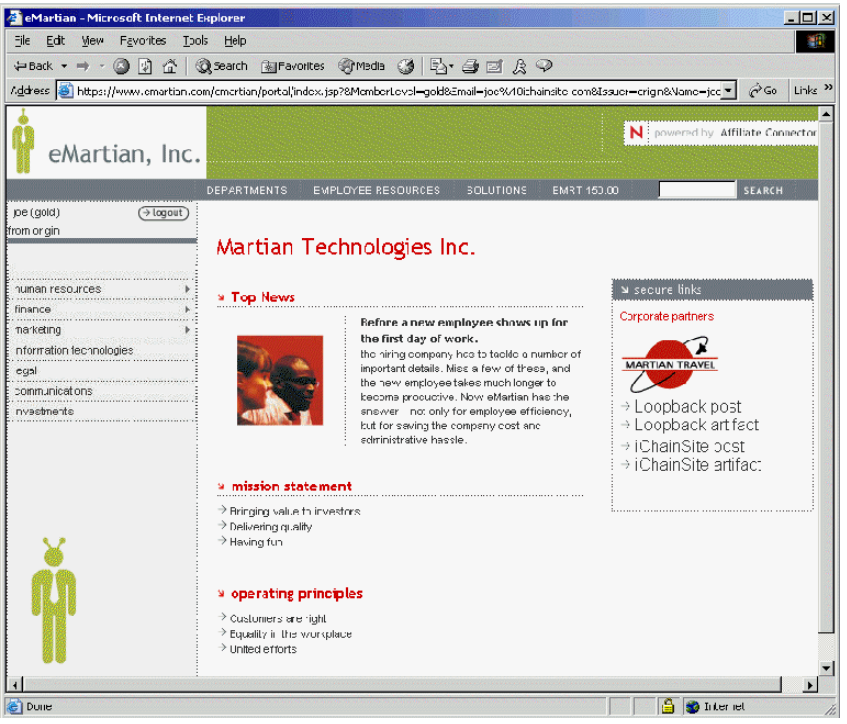
Starting the SAML Extension Server

See “Starting the SAML Extension Server” on page 29 for details on how to start the SAML extension server.

Testing the Loopback Affiliate Links

Once the SAML extension server has been configured and deployed, you can begin testing it. Access the main eMartian Site sample application by entering the following URL: http://www.emartian.com/emartian/portal. You should be prompted to authenticate to iChain. After successful login, you should receive a page like the one shown in Figure 46:

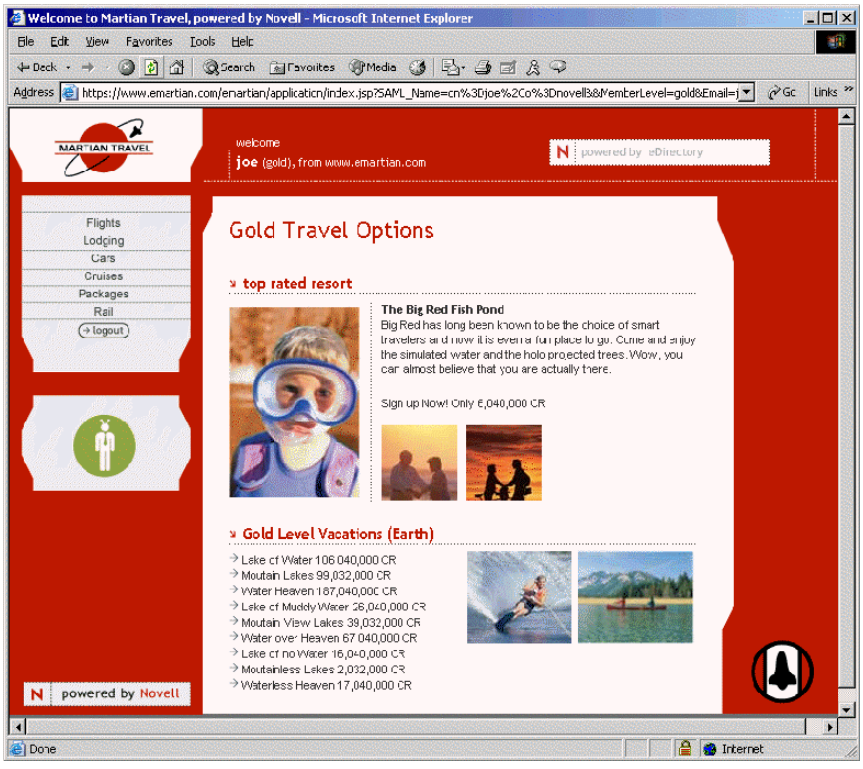
Figure 46 eMartian: Successful Login



The two links of interest here are Loopback post and Loopback artifact. The post link will perform a SAML single sign-on operation using the SAML Browser/POST profile. The artifact link will perform a SAML single sign-on operation using the SAML Browser/Artifact profile. See “Testing the Loopback Affiliate Links” on page 30 for the link format for the partner (SAML) links.

One important difference between the eMartian site and the iChainSite sample is that the eMartian Target parameter is set to the eMartian application, rather than to the portal. When you click the loopback links, you go to the eMartian application instead of the portal where you started. After you click the loopback link, a page is displayed as shown in Figure 47:

Figure 47 eMartian: Loopback



Note the "from www.emartian.com" at the top of the page. If you had accessed this page directly, the text would read, "from origin site."

Conclusion

You have now concluded the setup portion of the stand-alone eMartian SAML sample site. This documentation is intended to provide details of what is involved in the setup and deployment of the SAML extension for Novell iChain product. For more additional information, including installation and general administration, see the *SAML Extension for Novell iChain Administration Guide*.

3 Setting Up the iChainSite and eMartian Sample Site Affiliation

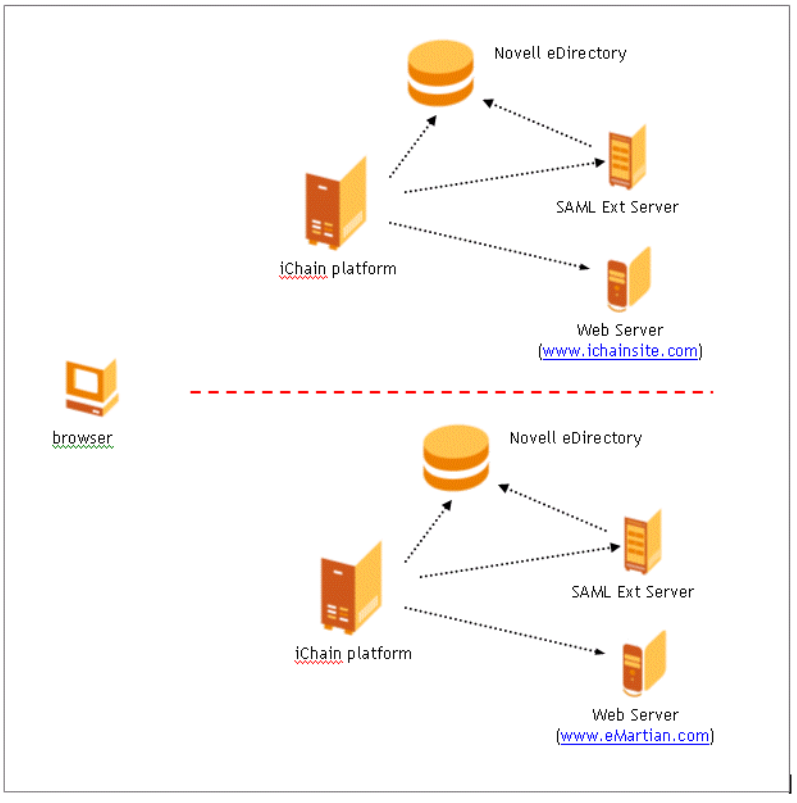
This chapter provides information on how to set up a SAML affiliation between the iChainSite and eMartian sample SAML sites using the SAML extension for Novell® iChain® product. This chapter assumes you have set up the iChainSite and eMartian sample sites, as discussed in the previous two chapters. The following general topics are covered:

- System Layout
- Creating the SAML Relationship Between the Sample Sites
- Updating Web Pages
- Troubleshooting the SAML Extension Sample Site Setup

System Layout

You should have two separate iChain/SAML extension systems set up. Each should have its own iChain, Web server, directory, and SAML extension server. The goal is for a user to be able to log in to either the iChainSite or the eMartian application and get single sign-on access to the other site. Not only should the user be authenticated at the partner site, but the partner should be able to display customized content to that user based upon attributes sent from the partner site. Figure 48 shows this configuration:

Figure 48 iChainSite/eMartian Configuration



Prerequisites

It is assumed that you have already created the scenario shown in Figure 48, with a complete iChain/SAML extension system setup for both iChainSite and eMartian. Refer to the previous chapters in this guide if you have not set up these sample sites.

Creating the SAML Relationship Between the Sample Sites

In order to create a SAML relationship between iChainSite and eMartian, you must do the following:

1. Create a Trusted Affiliate object for eMartian in the iChainSite configuration.
2. Create a Trusted Affiliate object for iChainSite in the eMartian configuration.

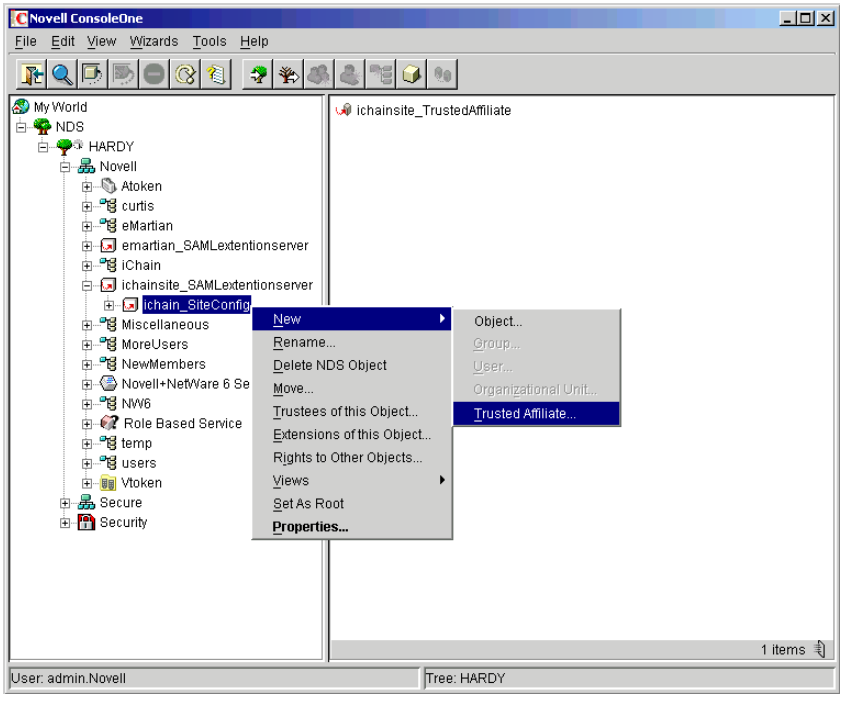
Creating the Trusted Affiliate Object for eMartian

To create an affiliation between iChainSite and eMartian, iChainSite must have an entry in its list of trusted affiliates for eMartian. To create this entry:

1 Select the iChainSite SAML Config Object.

2 Select New > Trusted Affiliate. See Figure 49:

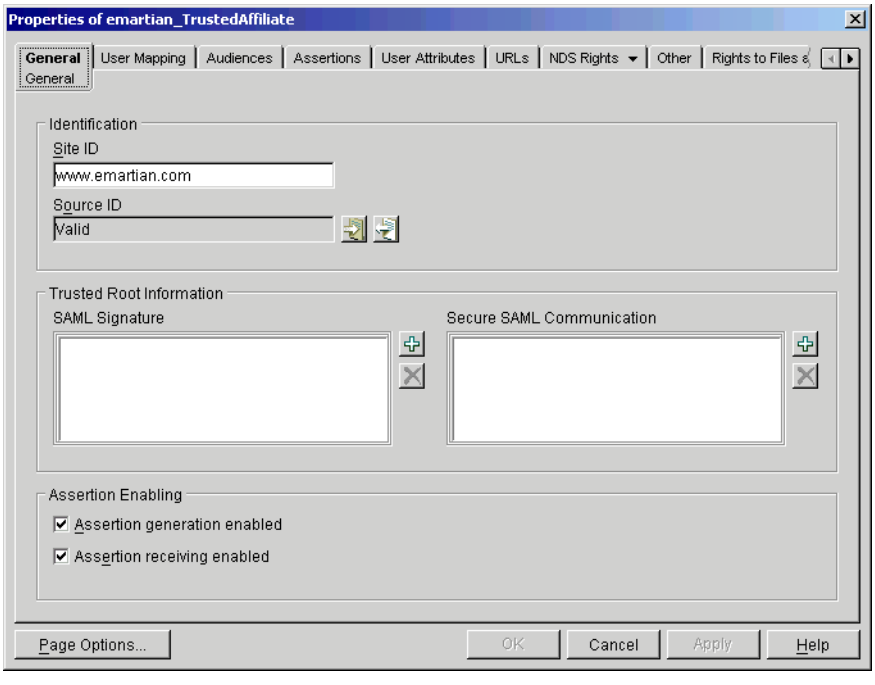
Figure 49 iChainSite: Trusted Affiliate



In this example, the Trusted Affiliate object that represents www.emartian.com is named eMartian.

3 Open the eMartian Trusted Affiliate object's Properties page.

Figure 50 eMartian Trusted Affiliate Object: Properties Page



4 Set the Site ID to www.emartian.com.

5 Auto-generate the SourceID.

Leave the Trusted Root Information fields blank.

6 Click the User Mapping tab.

7 Enter your desired user mapping scheme.

If you want to quickly get your sites running, you can leave the rules blank and use the default user mapping defined in the SAML Config object. Alternatively, you can use the e-mail attribute to create a dynamic user mapping rule, as used in the iChainsite and eMartian samples.

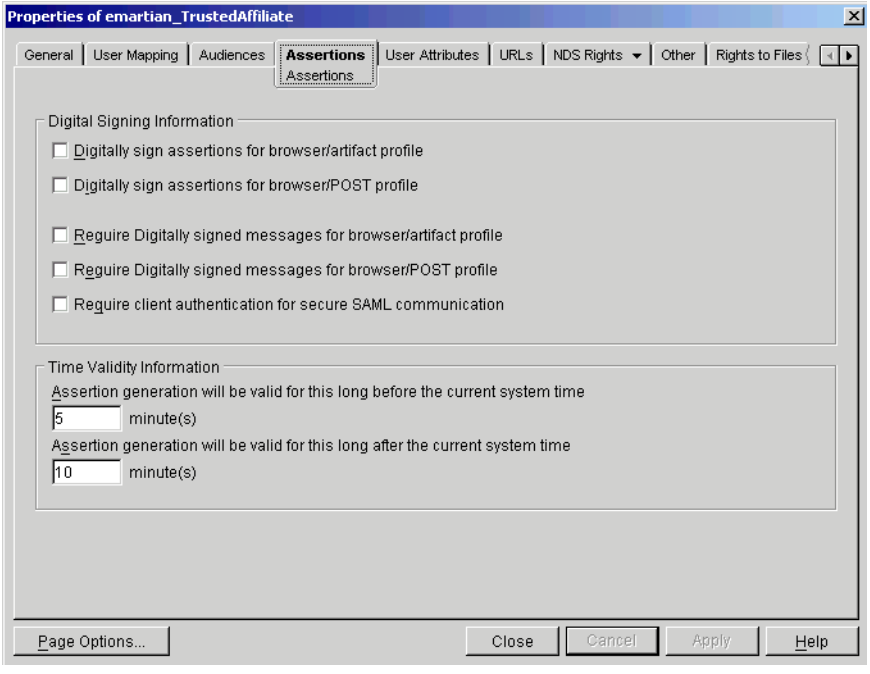
8 Click the Assertions tab.

9 Uncheck the Digital Signing Information check boxes.

For details on how to set up security between the two sites, see Appendix A, “Fine-Tuning the SAML Extension,” on page 63.

Figure 51 shows what the Assertions tab should look like for the eMartian Trusted Affiliate:

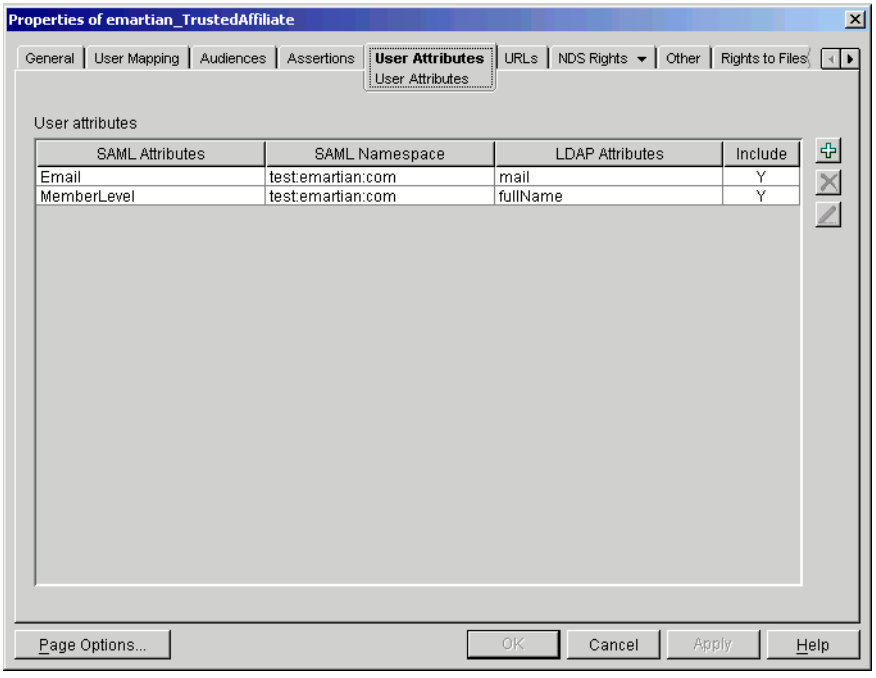
Figure 51 eMartian Properties: Assertions Tab



10 In order for the eMartian application to display custom-tailored content for the iChainSite users, the Email and Password attributes should be sent.

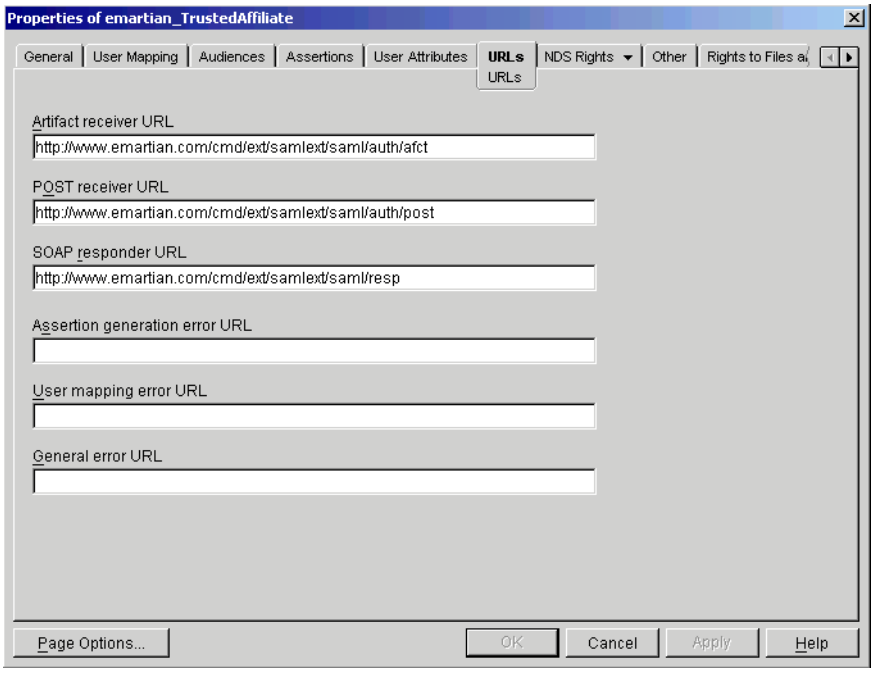
Figure 52 shows what the User Attributes tab should look like:

Figure 52 eMartian Properties: User Attributes Tab



11 The iChainSite to eMartian Trusted Affiliate should have all of the URLs necessary to let iChain contact eMartian. Follow the example in Figure 53 to set up these URLs:

Figure 53 eMartian Properties: URLs Tab



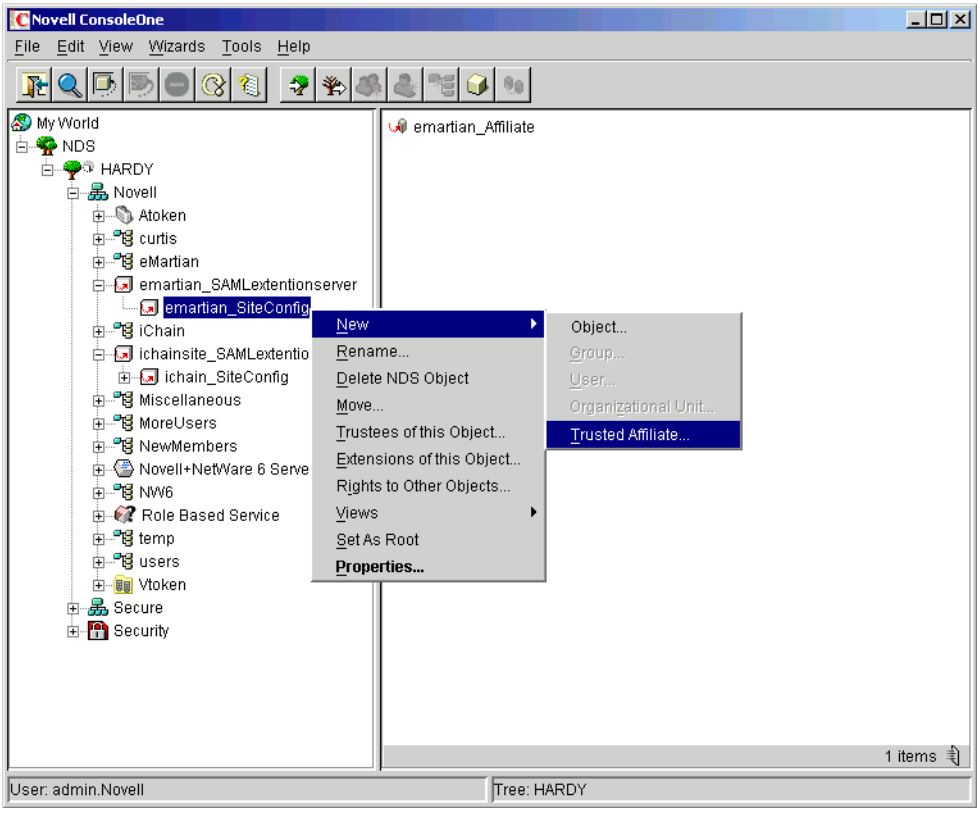
At this point, iChainSite can trust eMartian.

Creating the Trusted Affiliate Object for iChainSite

Now that iChainSite can trust eMartian, you must configure eMartian to trust iChainSite in return. To do this, you must create a Trusted Affiliate entry in the eMartian SAML configuration representing iChainSite.

1. Select the eMartian SAML Config Object.
2. Select New > Trusted Affiliate. See Figure 54.

Figure 54 eMartian: Trusted Affiliate



In this example, ichainsite is the chosen name for the Trusted Affiliate object. Once you create this object, open its Properties page.

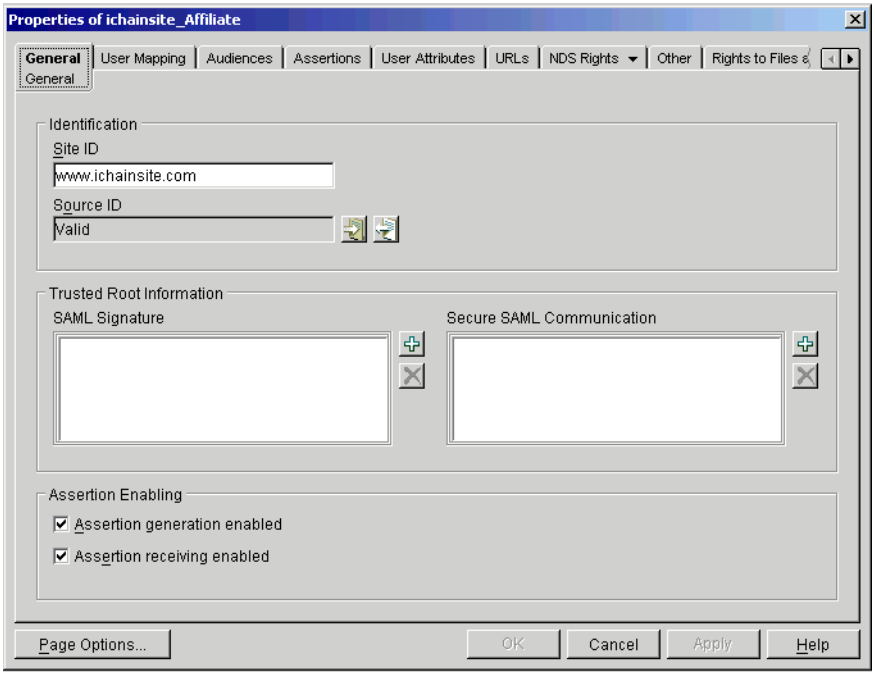
3. Right-click the object and select Properties.

   Do the following to configure this object.

4. Set the SiteID to www.ichainsite.com.
5. Auto-generate the SourceID.

   Leave the Trusted Root Information fields blank.

Figure 55 iChainSite: Properties Page



6 Click the User Mapping tab.

7 Enter your desired user mapping scheme.

If you want to quickly get your sites running, you can leave the rules blank and use the default user mapping defined in the SAML Config object. Alternatively, you can use the e-mail attribute to create a dynamic user mapping rule, as used in the iChainsite and eMartian samples.

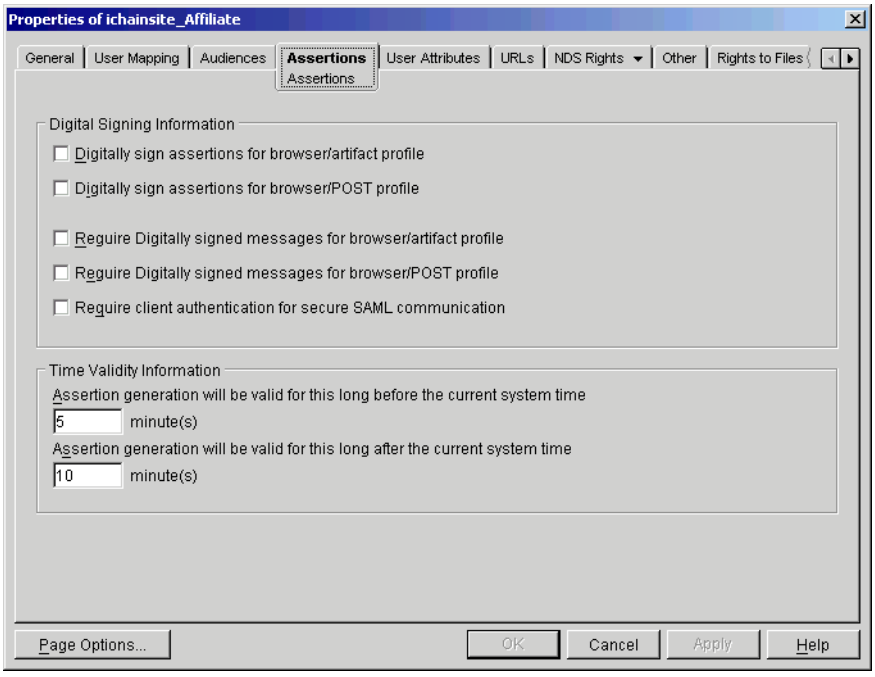
8 Click the Assertions tab.

9 Uncheck the Digital Signing Information check boxes.

For details on how to set up security between the two sites, see Appendix A, “Fine-Tuning the SAML Extension,” on page 63.

Figure 55 shows what the Assertions tab should look like for the iChainSite Trusted Affiliate:

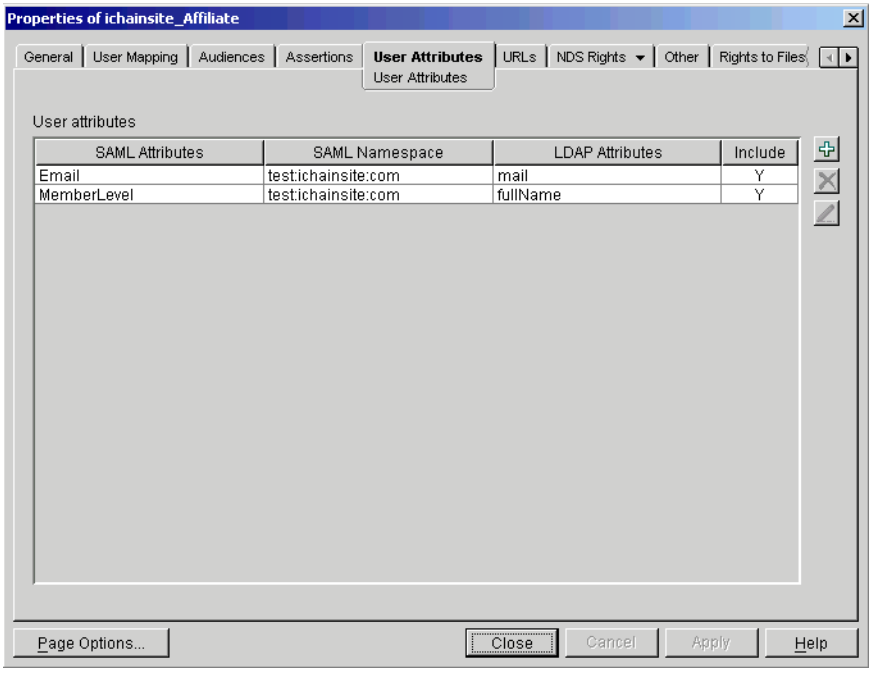
Figure 56 iChainSite Properties: Assertions Tab



10 In order for the iChainSite application to display custom-tailored content for the eMartian users, the Email and Password attributes should be sent.

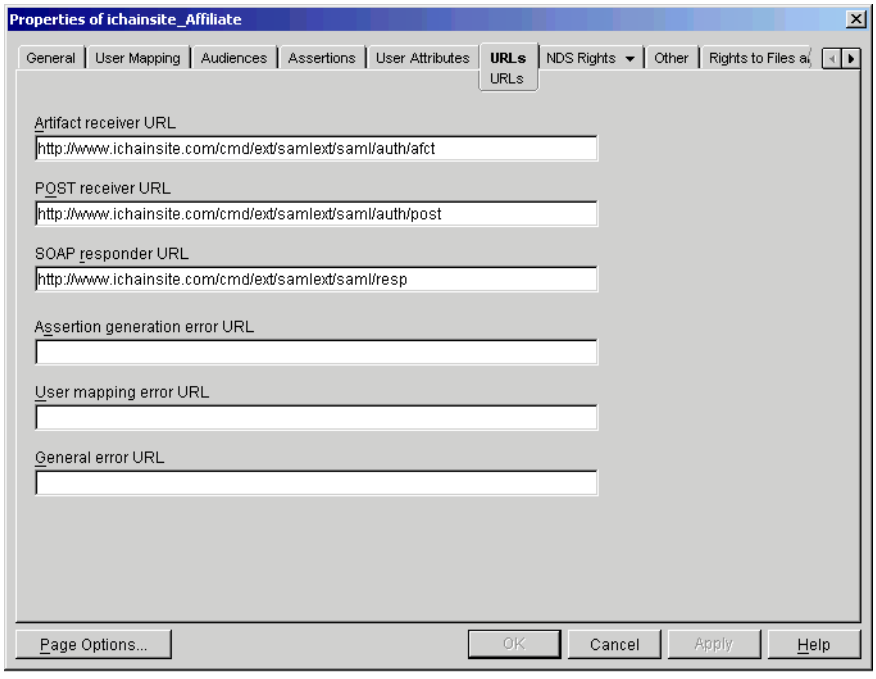
Figure 57 shows what the User Attributes tab should look like:

Figure 57 iChainSite Properties: User Attributes Tab



11 The eMartian to iChainSite Trusted Affiliate should have all of the URLs necessary to let eMartian contact iChainSite. Follow the example in Figure 58 to set up these URLs:

Figure 58 iChainSite Properties: URLs Tab



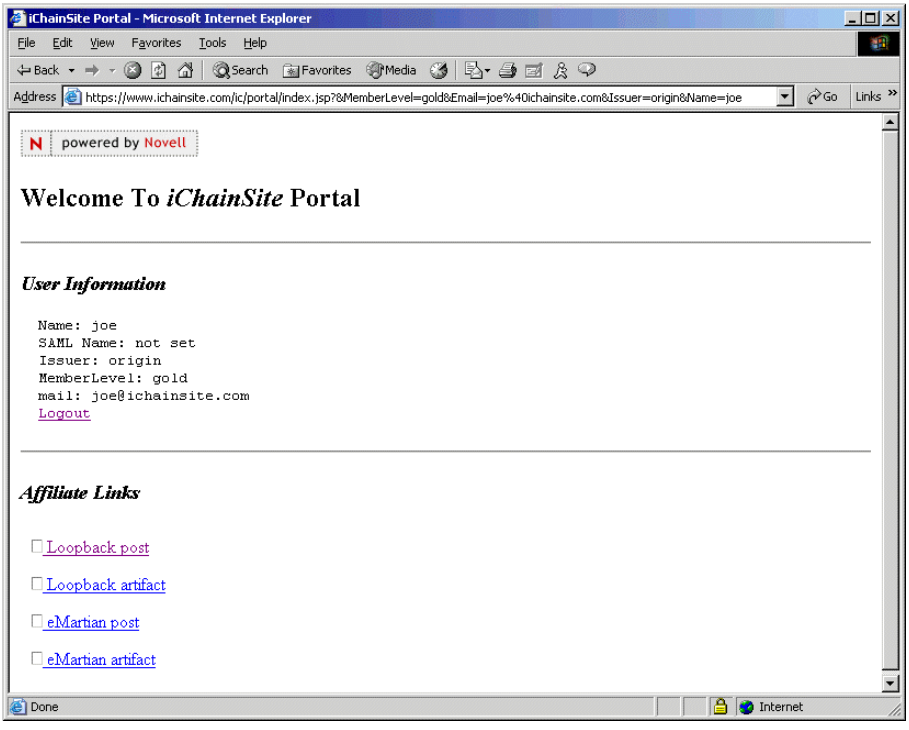
Updating Web Pages

After configuring the SAML systems for both iChainSite and eMartian, you need to create Intersite Transfer URLs for each portal page. The iChainSite portal needs an affiliate link pointing to eMartian, and vice versa.

iChainSite Intersite Transfer URLs

At the iChainSite portal/index.jsp page there are some affiliate links. The first two are loopback links that perform SAML single sign-on operations with the iChainSite. Intersite transfer URLs for eMartian must be added to the page. The eMartian post and eMartian artifact URLs are provided to send users from the iChainSite portal to the eMartian application.

Figure 59 iChainSite Portal



The source of the eMartian intersite transfer URLs is shown in Figure 60:

Figure 60 eMartian Intersite Transfer URLs

```
<!-- eMartian POST -->
<A
href="https://www.ichainsite.com/cmd/ext/samlext/saml/gen/post?AID=ww
w.eMartian.com&TARGET=http://www.eMartian.com/eMartian/application">
 eMartian post</a>

<!-- eMartian Artifact -->
<A
href="https://www.ichainsite.com/cmd/ext/samlext/saml/gen/afct?AID=ww
w.eMartian.com&TARGET=http://www.eMartian.com/eMartian/application">
eMartian artifact</a>
```

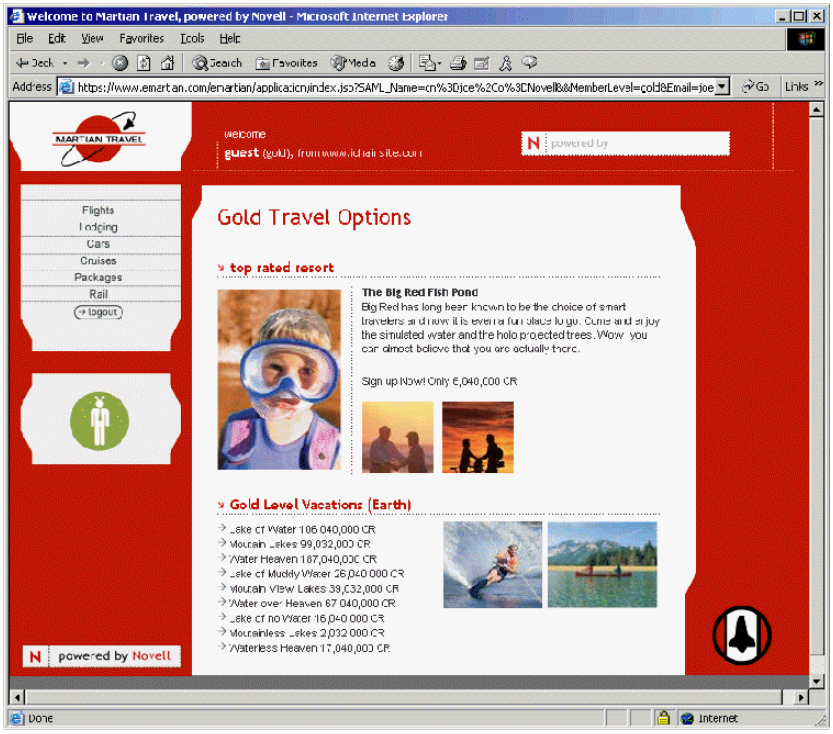
Note that each of the links points back to the iChainSite resource used to generate assertions; https://ww.ichainsite.com/cmd/ext/samlext/saml/gen/post is used to generate assertions in the Browser/POST profile, and https://www.ichainsite.comd/cmd/ext/samlext/saml/gen/afct is used to generate assertions in the Browser/Artifact profile. There are two critical parameters that must be included in the URL in order for the assertion generation to work:

Aid: This is the site ID for which the assertion is being generated. In the example above, assertions are generated for www.emartian.com. The Aid must match the SiteID configured in the directory.

Target: The target is the resource the user wants to access at the partner site. The eMartian site link target is http://www.emartian.com/application.

Clicking on either of the eMartian links will cause the iChainSite to generate a SAML Assertion intended for eMartian. The user will then be sent to the appropriate SAML receiver URL on the eMartian site. The SAML service running at eMartian will validate the provided SAML Assertion, evaluate the user mapping rules, and provide the user with the target resource. See Figure 61.

Figure 61 From iChain to eMartian



eMartian Intersite Transfer URLs

In order to send users from eMartian to iChainSite, intersite transfer URLs must be added to the eMartian portal. By default the eMartian portal contains these URLs for iChainSite. The source for these URLs is shown in Figure 62:

Figure 62 eMartian Portal: URLs for iChainSite

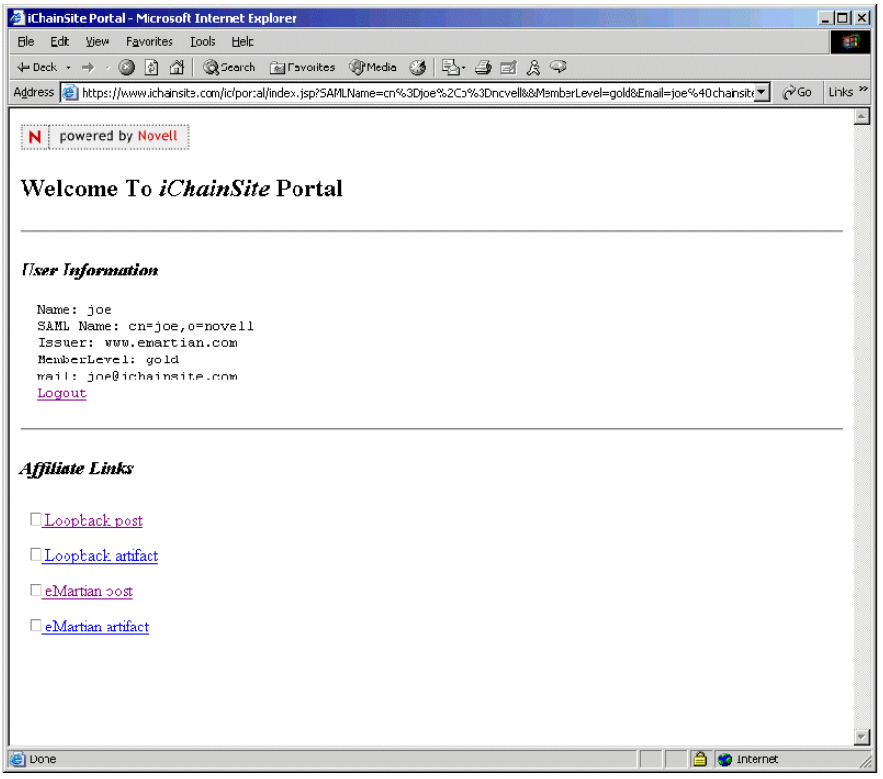
```
<!-- ichainsite POST -->
<A
href="https://www.eMartian.com/cmd/ext/samlext/saml/gen/post?AID=www.
ichainsite.com&TARGET=http:// www.ichainsite.com/ic/portal ">
 eMartian post</a>

<!-- ichainsite Artifact -->
<A href="https://www.
eMartian.com/cmd/ext/samlext/saml/gen/afct?AID=www.
ichainsite.com&TARGET=http://www.ichainsite.com/ic/portal">
eMartian artifact</a>
```

The eMartian Intersite Transfer Links follow the same pattern as those in the iChainSite portal. They both point to the SAML Assertion generation service running at eMartian: https://www.emartian.com/cmd/ext/samlext/saml/gen/post, and https://www.emartian.com/cmd/ext/samlext/saml/gen/afct. The AID is www.ichainsite.com, indicating that the SAML Assertion is to be generated for iChainSite. The target URL is set to the iChainSite portal.

Clicking on either of the iChainSite links will cause the eMartian system to generate a SAML Assertion for iChainSite, sending the user to the SAML receiver at the iChainSite. The SAML service running on the iChainSite will process the provided assertion, map the user to an identity, and provide the user with the requested resource. See Figure 63.

Figure 63 From eMartian to iChain



Troubleshooting the SAML Extension Sample Site Setup

At this point, you should be able to perform SAML single sign-on between the iChainSite and eMartian applications. If you are having difficulty, the following list contains common problems and possible workarounds:

Assertion Generation error, unknown affiliate: This is causes by an Aid parameter in the intersite transfer URL that does not anything in the site's trusted affiliates. Make sure that the Aid has a corresponding Site ID in the trusted affiliates. The CN of the object is not what matters; the Site ID value in the General page is the value that counts.

Assertion Receive error, unknown affiliate: This is similar to the Assertion Generation error, however, in this case the receiving site does not have any information about the issuer of the assertion. Make sure that you have a Trusted Affiliate object with a Site ID that matches the incoming SAML assertion's issuer. Again, the CN of the object does not matter; the Site ID value on the Trusted Affiliate object is the important value.

Assertion Not Yet Valid and/or Assertion No Longer Valid: SAML assertions contain timestamp values that limit how long they are considered valid. These values are set on the Trusted Affiliate Assertions Properties page. Generally, the validity window for assertions is in minutes. Thus, if two partner sites have clocks that do not match closely, you will encounter validity period problems. Make sure that the partner sites have system clocks that are synchronized within one or two minutes.

Untrusted Certificate: This is a problem that comes up in the Browser/Artifact profile when a Site attempts to access an assertion over the server-to-server back channel. See Appendix A, “Fine-Tuning the SAML Extension,” on page 63 for detailed instructions on how to set up the

security of this back channel. For quick reference, verify that the appropriate SOAP Responder URL uses http:// protocol rather than https:// protocol.

Unsigned, or Unable to Sign: This error is due to the Digitally Sign Assertions or Require Digital Signature settings in the Assertions Properties page being set. See the Appendix A, “Fine-Tuning the SAML Extension,” on page 63 for detailed instructions on how to setup XML signature generation and validation. For quick reference, verify that neither the generate or require check boxes are checked in the Assertions Properties page.

A Fine-Tuning the SAML Extension

This appendix includes information for fine-tuning the SAML extension. The following topics are discussed:

- XML Signature Generation and Validation
- Creating a Signing Key Pair
- Configuring SAML to Support SSL Mutual Authentication

XML Signature Generation and Validation

You must complete the following general steps in order to configure the SAML system to generate digital signatures:

1. A Signing Key Pair (SKP) must be generated or imported into the system and stored in eDirectory.
2. The SKP must be exported from eDirectory in PKCS#12 format and stored on the SAML extension server.
3. The appropriate settings must be set on the Trusted Affiliate object whose SAML data you want to sign.
4. The Public Key Certificate associated with the SKP must be exported and sent the Trusted Affiliate which will be validating the signatures you generate.

Creating a Signing Key Pair

There are a number of ways to get a KeyPair into an eDirectory system. You can:

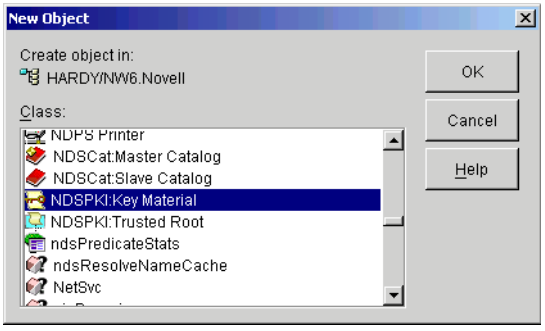
- Use Novell Certificate Server to generate the Key Pair and certificate, signed by your Tree CA
- Use Novell Certificate Server to generate the Key Pair and Certificate Signing Request to be signed by an external CA
- Use Novell Certificate Server to import an external Key Pair stored in PKCS#12 format

The following instructions describe how to generate a Key Pair using Novell Certificate Server and the Novell Certificate Server Snap-ins. For more detailed information on key and certificate management, see the Novell Certificate Server documentation.

1 Create a NDSPKI:Key Material object.

This is way the Novell Certificate Server calls a public - private Key Pair object. Novell Certificate Server stores and associates Key Pair objects with servers, so you must create the Key Material object in the same container as the server you want to use to host the key. For these purposes, the server you choose to use is immaterial. Figure 64 shows how a Key Material object is created:

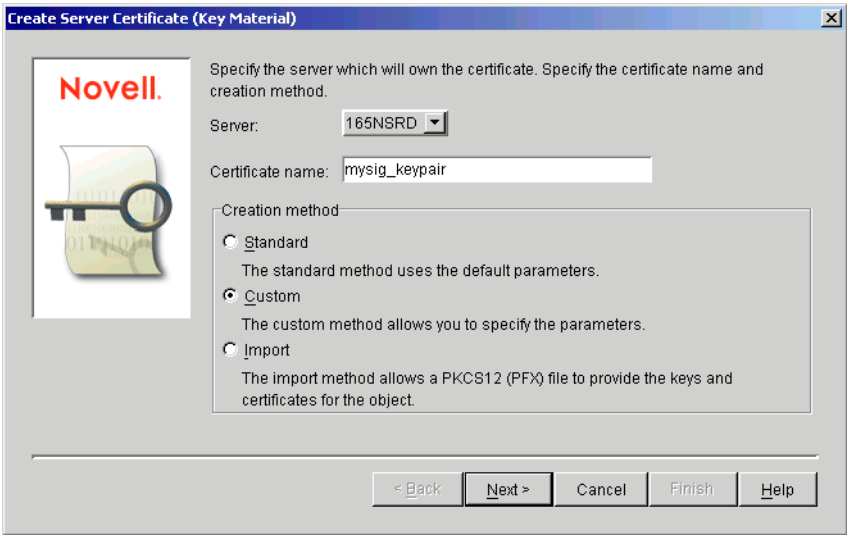
Figure 64 Creating the Key Material Object



2 Select the Key Material object and click OK.

A wizard is launched to guide you through the certificate generation process, as shown in Figure 65.

Figure 65 Certificate Generation Wizard



3 Select the server where you want to create the private key.

The Creation method selector determines what type of operation you are going to perform.

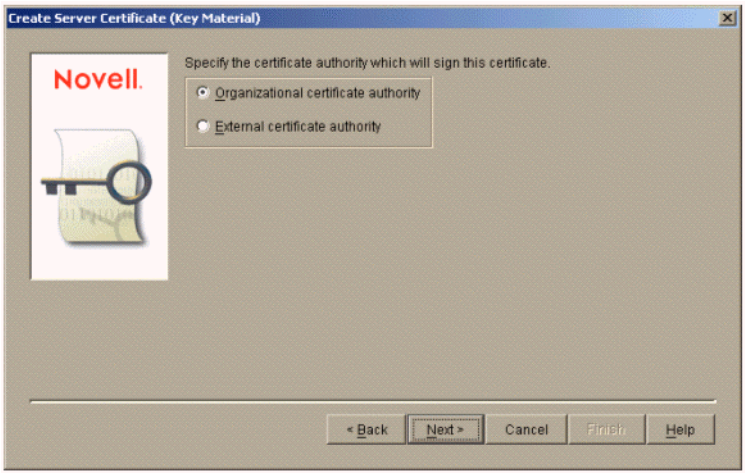
4 If you already have a certificate you want to use, select Import. Otherwise, select Custom.

4a If you choose the Import option, you are prompted to enter the file name and password associated with the PKCS#12 file you want to import. (This is a relatively straightforward operation; see the Novell Certificate Server documentation for details on importing external certificates.)

5 Click Next.

6 You are prompted about whether you want to create a public key certificate signed by your Tree's Certificate Authority (CA) or an external CA, as shown in Figure 66:

Figure 66 Specify the CA

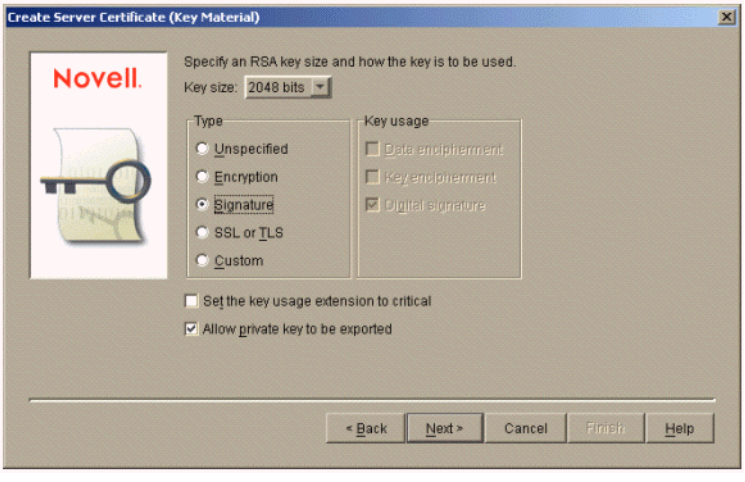


Typically, if you are creating a certificate for testing, you can use your built-in Organizational Certificate Authority. For production, you will probably want a certificate signed by a well-known CA, such as Verisign or Entrust. If you want to select an external CA, select External certificate authority.

7 Click Next.

8 Define the Key Pair properties. Since this certificate will be used to sign SAML data, make sure Signature is selected, as shown in Figure 67:

Figure 67 Define Key Pair Properties



Typically, a key size of 2048 bits is sufficient.

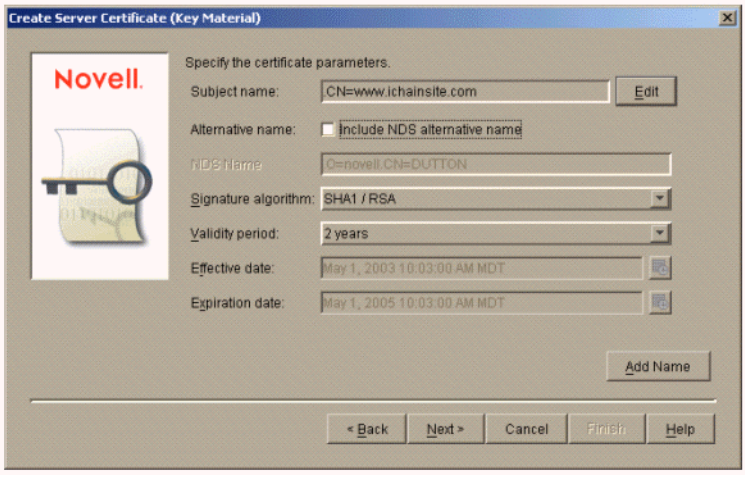
9 Click Next.

10 Define additional key and certificate properties.

Make sure you create a subject name that will allow your partner sites to identify the certificate as yours. A general rule is to make the subject name the same as your Site ID. In this example, a certificate is being generated for the iChainSite sample site, so the Subject

name is .CN=www.ichainsite.com. Subject names in the Novell Certificate Server must begin with a period (.) character, as shown in Figure 68:

Figure 68 Define Additional Key and Certificate Properties



11 Click Next.

12 Finish this operation: If you are signing the certificate with your Tree CA (Organizational), you are prompted to select the certificate that will sign your certificate.

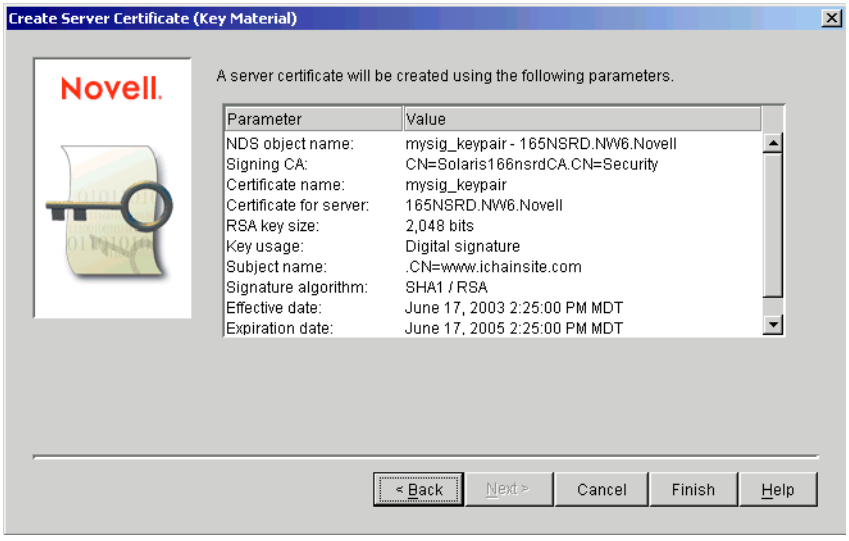
Either of these selections will work for testing purposes. For information about the differences between the two, refer to the Novell Certificate Server documentation.

Figure 69 Specify the Trusted Root Certificate

13 Click Next.

You are presented with a summary page that outlines all of the selections you made using the wizard. Verify that your selections are correct, then select Finish.

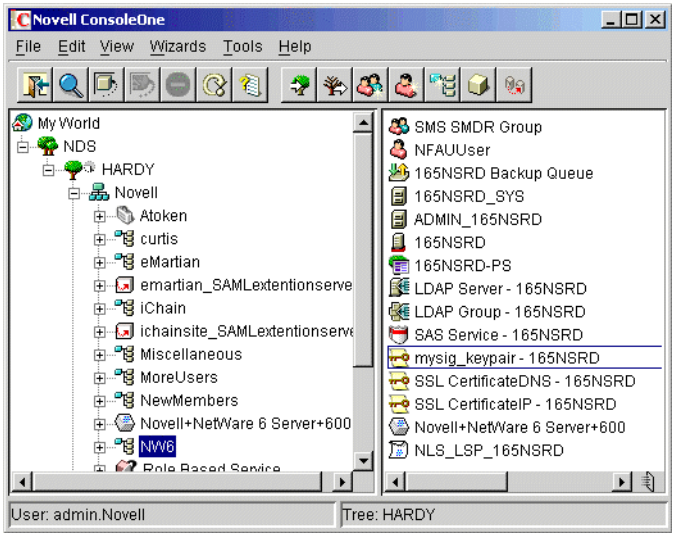
Figure 70 Summary of Your Selections



The Key Pair will be generated.

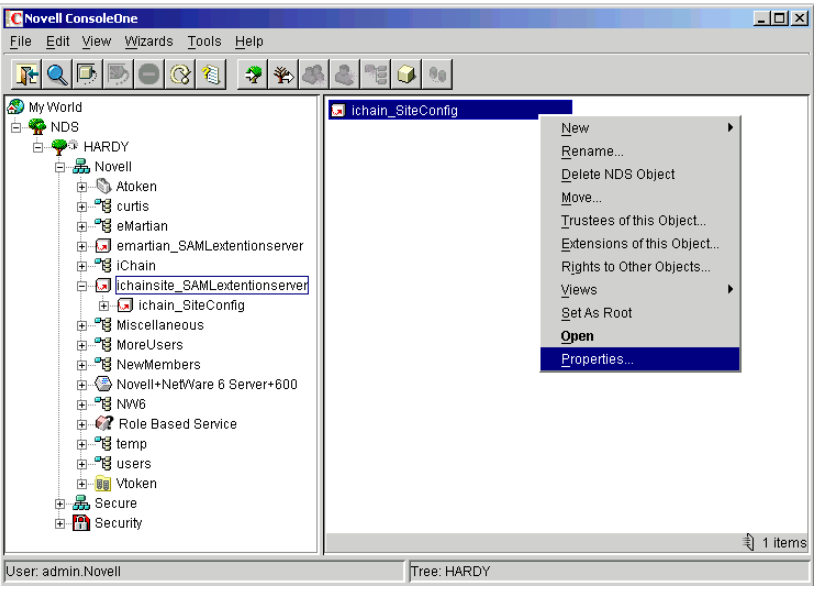
You should show a new NDSPKI: Key Material object in the directory. The name of this new object is the name you specified in the wizard, followed by the hosting server name. Figure 71 shows the Key Pair generated in this example. The certificate name is mysig_keypair - Dutton.

Figure 71 Key Pair



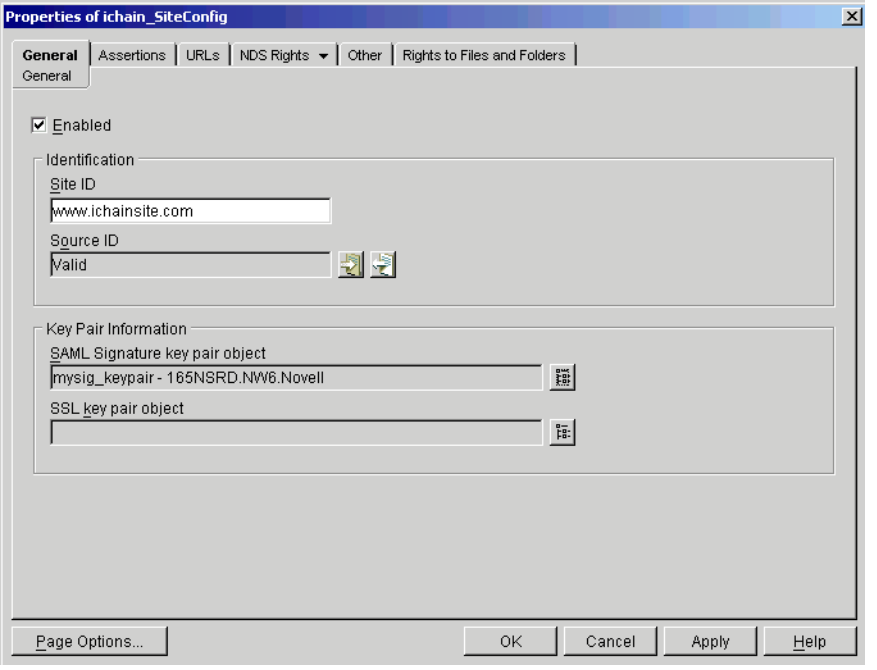
13a You can associate your SAML configuration object with the Key Pair you just created. While this is not required, we recommend it because it keeps the association between the Key Pair you generated and the SAML configuration. You create this link by right-clicking the SAML configuration object and selecting Properties, as shown in Figure 72:

Figure 72 SAML Configuration Object: Properties



If you create this association, when you are working with the configuration of the SAML system, you now have a link back to the Key Pair you're using to sign SAML data, as shown in Figure 73:

Figure 73 Key Pair Link



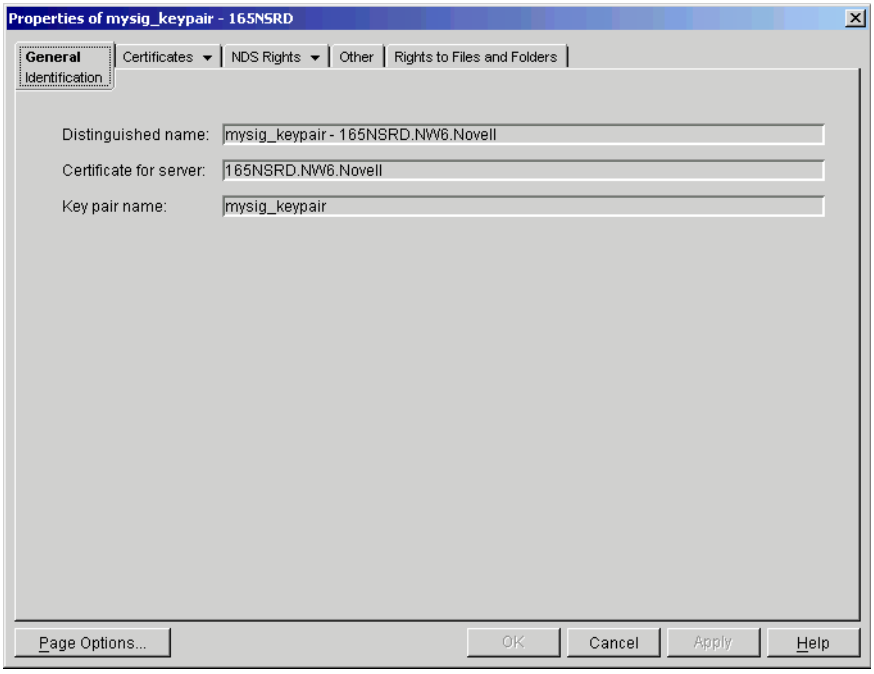
Exporting a Signing Key Pair

You must now export the Key Pair you just created to a disk to store on the SAML extension server:

1 Double-click or right-click the Key Pair object and select Properties.

The Key Pair's Properties page is displayed, as shown in Figure 74:

Figure 74 Key Pair Properties

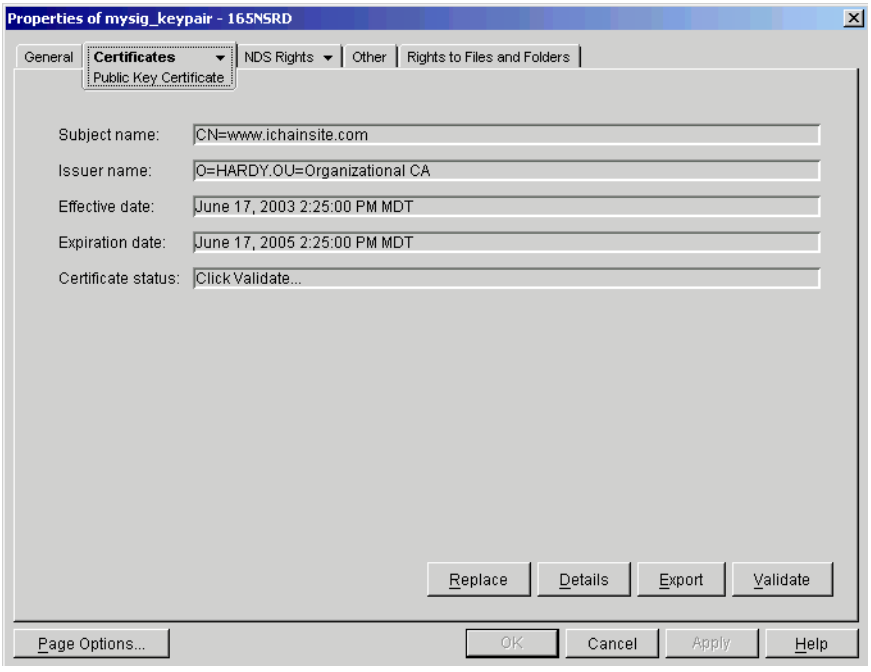


2 Click the Certificates tab.

3 Select Public Key on the drop-down menu.

A Properties page is displayed, as shown in Figure 75:

Figure 75 Public Key Properties

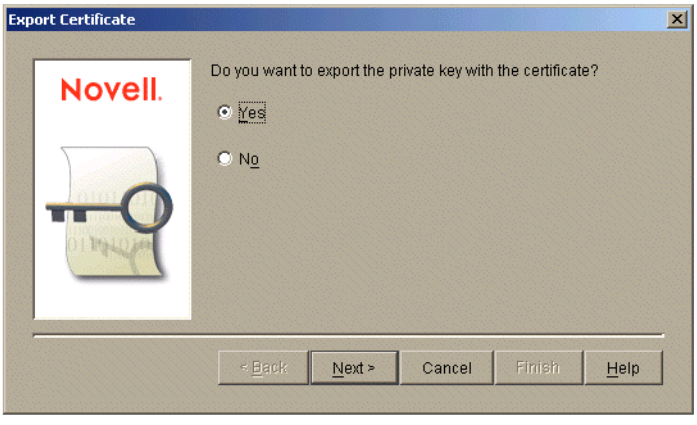


4 Verify that the Subject name, Effective date, and Expiration date match the values you entered during the certificate creation process.

5 Click Export.

This will export the Key Pair. A wizard dialog is displayed, as shown in Figure 76:

Figure 76 Key Pair Export Wizard



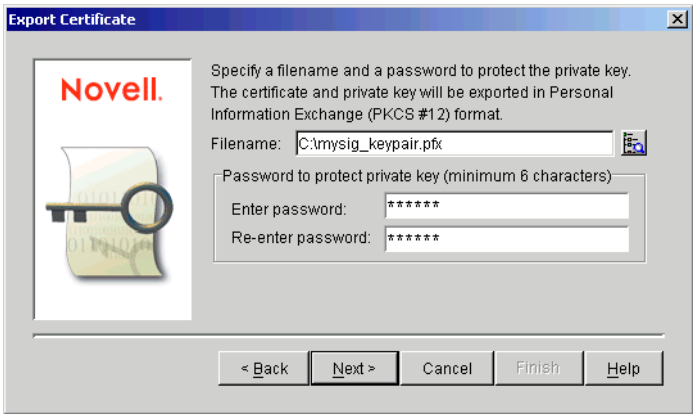
If you want to export only the Public Key Certificate, select No. In this example, since you are exporting the Public - Private Key Pair, Yes is selected.

Selecting Yes exports the Public and Private keys in password-protected PKCS#12 format. This is what you must move to the SAML extension server.

Selecting No exports the Public Key only. This is what you must provide to your partner sites later.

6 Select Next. The Export Certificate wizard is displayed, as shown in Figure 77:

Figure 77 Export Certificate Wizard



7 Enter a filename and password.

The filename can be whatever you desire. Take note of the password you enter. It will be required later as part of the SAML extension server setup.

8 Click Next.

Setting the PKCS#12 Signature Key on the SAML Extension Server

To import the Signature Key file into the SAML extension server for use:

1 Copy the PKCS#12 file exported in the previous process to the local drive of the SAML extension server.

2 Modify the SAML extension server configuration file to point to the PKCS#12 file.

When you installed SAML extension, a configuration file was automatically generated. This file is located at *SAMLEXT_HOME*/conf/samlextConfig.xml. (For example, this path could be C:\Program Files\Apache Group\Tomcat 4.1\webapps\samlext\conf\samlextConfig.xml.) If you did not specify any Key Pairs during the installation, the configuration file should look like the one shown in Figure 78:

Figure 78 SAMLextConfig.xml File

```
<authority name="SAMLExtDirectoryAuthority">
        <class><name>com.novell.wss.authority.saml.SAMLExtDirectoryA
uthority</name></class>
        <data>
                <servers>
                        <url>ldap://137.65.159.66:389</url>
                </servers>
                <proxyUser>
                        <dn>cn=admin,o=novell</dn>
                        <password>novell</password>
                </proxyUser>
                <isoDn>cn=iso,o=novell</isoDn>
                <initialCapacity>10</initialCapacity>
                <maxCapacity>30</maxCapacity>
                <keypairs>
                        <keypair usage="ssl" type="jks">
                                <password></password>
                                <file></file>
                        </keypair>
                        <keypair usage="signing" type="jks">
                                <password></password>
                                <file></file>
                        </keypair>
                </keypairs>
        </data>
</authority>
```

Modify the signature keypair element with usage signing to include the file name and password of the PKCS#12. In the example, the file would be modified to read as shown in Figure 79:

Figure 79 Modified Signature KeyPair Element

```
<authority name="SAMLExtDirectoryAuthority">
        <class><name>com.novell.wss.authority.saml.SAMLExtDirectoryA
uthority</name></class>
        <data>
                <servers>
                        <url>ldap://137.65.159.66:389</url>
                </servers>
                <proxyUser>
                        <dn>cn=admin,o=novell</dn>
                        <password>novell</password>
                </proxyUser>
                <isoDn>cn=iso,o=novell</isoDn>
                <initialCapacity>10</initialCapacity>
                <maxCapacity>30</maxCapacity>
                <keypairs>
                        <keypair usage="ssl" type="jks">
                                <password></password>
                                <file></file>
                        </keypair>
                        <keypair usage="signing" type="pkcs12">
                                <password>novell</password>
                                <file>c:\mysig_keypair.pfx</file>
                        </keypair>
                </keypairs>
        </data>
</authority>
```

This example assumes that the exported PKCS#12 file was copied to the SAML extension server as c:\mysig_keypair.pfx, using novell as the password.

Modifying the SAML Settings in the Directory

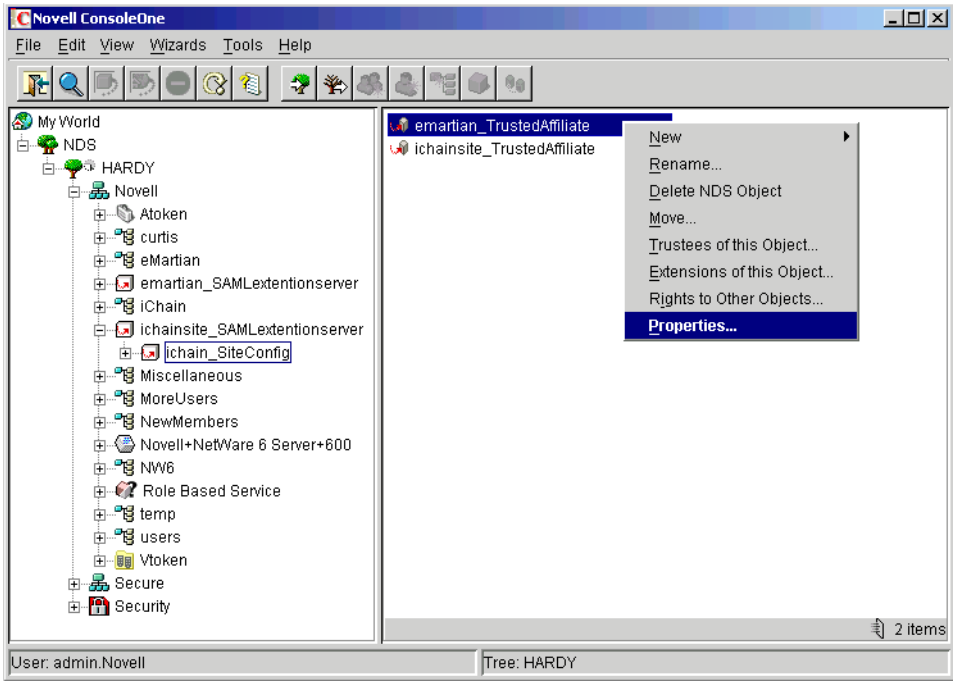
Although you have created and imported the Key Pair, it will only be used if it is required by the SAML configuration stored in the directory. The signing of SAML data is a setting made on a per-affiliate basis. This means that the SAML administrator can decide which SAML partner sites will receive signed data and which will not. The following steps show how signing is turned on for the eMartian affiliate:

1. Select the Trusted Affiliate object you want to sign data for.

   In this example, data that is intended for the eMartian affiliate will be signed.

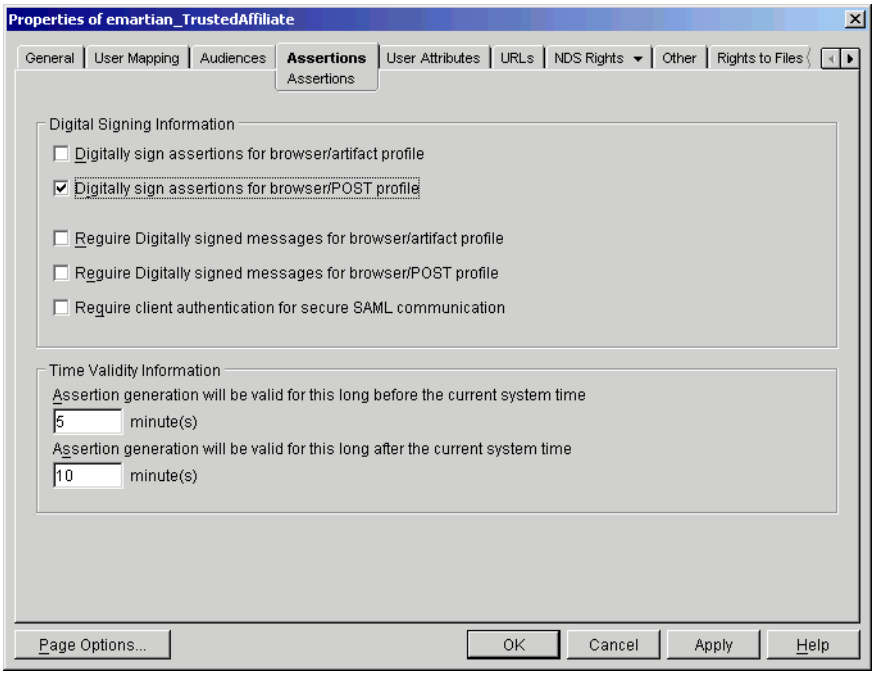
2 Open the Trusted Affiliate object's property page by right-clicking the object and selecting Properties, as shown in Figure 80:

Figure 80 Trusted Affiliate Properties



3 Click the Assertions tab.

Figure 81 eMartian Properties: Assertions Tab



4 Select Digitally sign assertions for the browser/POST profile.

This causes the system to use your Key Pair to sign SAML data sent to the eMartian Trusted Affiliate using the browser/POST file.

5 Click OK.

Exporting the Public Key Certificate

Now that you will be signing the SAML data, you need to provide your SAML partner site with a way of validating the signatures you generate. You do this by providing the partner with your Public Key Certificate which the partner can import into its system and use to validate the signatures you generate.

To export the public key certificate:

1 Open the Properties page associated with the Key Pair you are using to generate your digital signatures (the same as you did when you exported the Key Pair in PKCS#12 format).

2 Click the Certificates tab, then select public key certificate. (This is also the same as you did when you exported the Key Pair in PKCS#12 format.).

3 Click Export.

The Export Certificates wizard is displayed, as shown in Figure 82:

Figure 82 Export Certificate Wizard

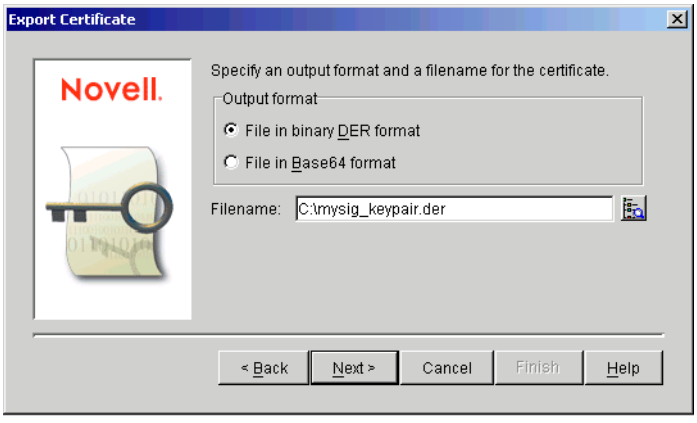
4 Select No. You will only export the public key portion of the pair.

5 Click Next.

6 Select the file name and format to save the file as.

The most common file format is binary DER encoding.

Figure 83 Output Format: Binary DER Format



7 Click Next.

8 Click Finish.

Next, you will want to send the public key certificate to your partner site(s) that want signed data. The partner site(s) would then import the certificate so that they could validate your signatures.

Importing Public Key Certificates

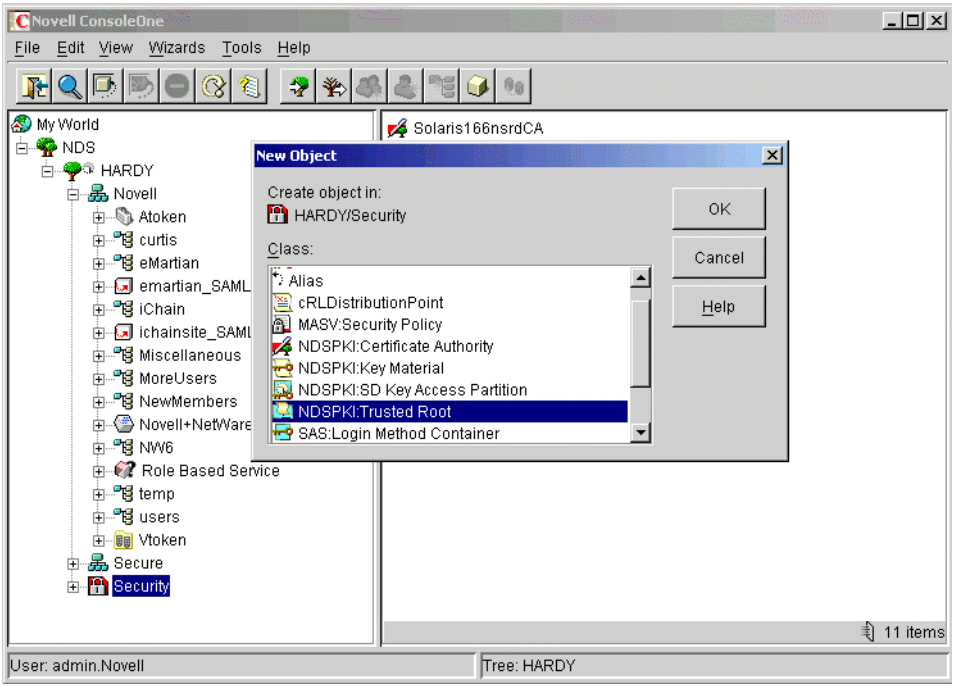
This section is presented from the point of view of the SAML administrator of eMartian. At this point you have received a public key certificate from www.ichainsite.com that you can use to validate signatures generated by that site. You must now import this public key certificate into your system and associate it with the Trusted Affiliate object representing www.ichainsite.com. The following steps show how to import a public key certificate into eDirectory and then associate the certificate with the Trusted Affiliate object representing www.ichainsite.com.

In Novell eDirectory, trusted public key certificates must be placed in a Trusted Roots container. If you do not already have one you will need to create this container. Typically, the Trusted Roots container is created under the [ROOT].security container.

1 Right-click the security container, and select New > Object.

2 Select NDSPKI:Trusted Root as shown in Figure 84:

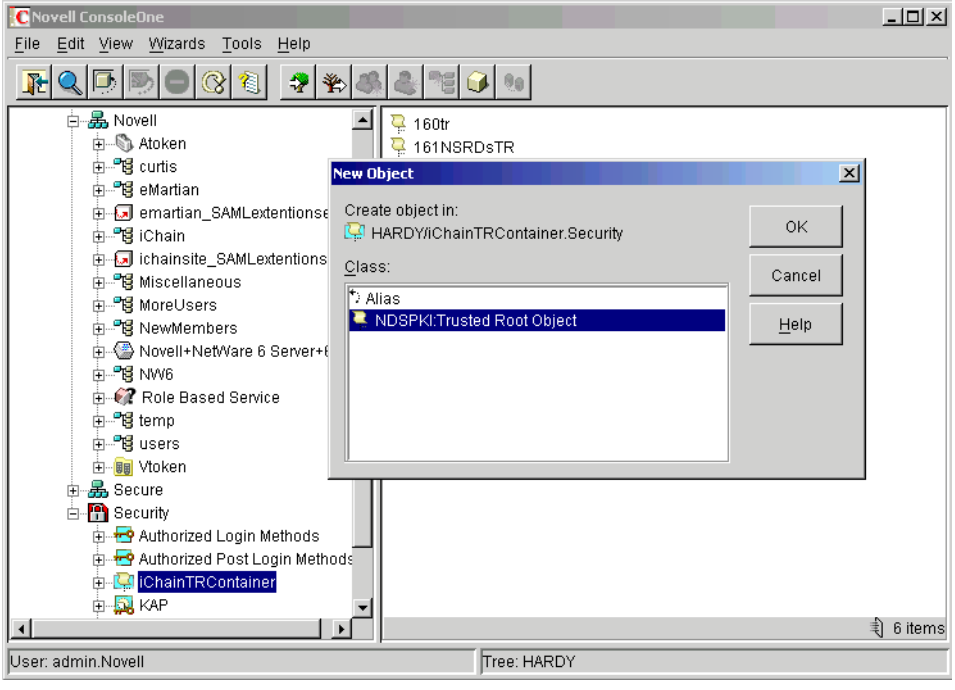
Figure 84 NDSPKI Trusted Root



When the Trusted Roots container has been created, you can add a new Trusted Root public key certificate to it.

3 Right-click the Trusted Roots container and select New > Trusted Root.

Figure 85 New Trusted Root

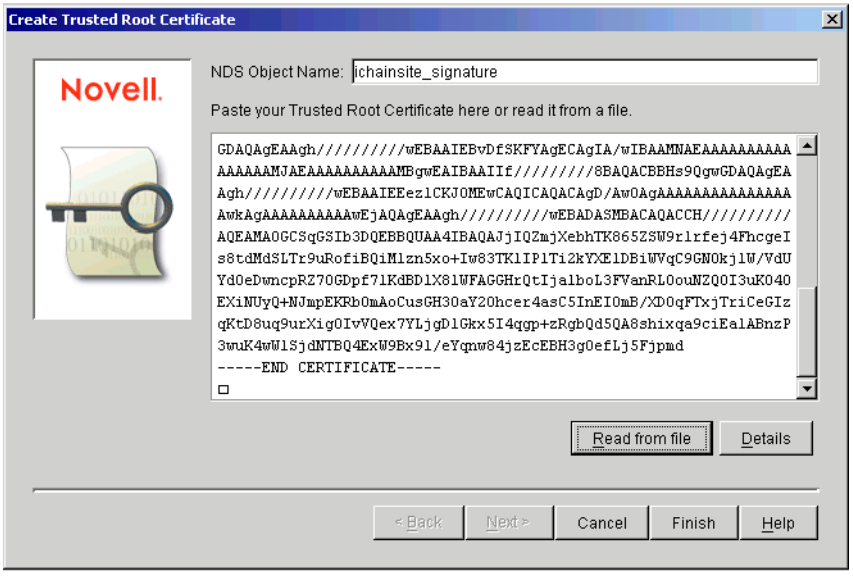


4 Click OK.

The Create Trusted Root Wizard launches.

5 Enter the NDS Object name for the trusted root. In Figure 86, ichainsite_signature has been entered as the name.

Figure 86 Trusted Root Name



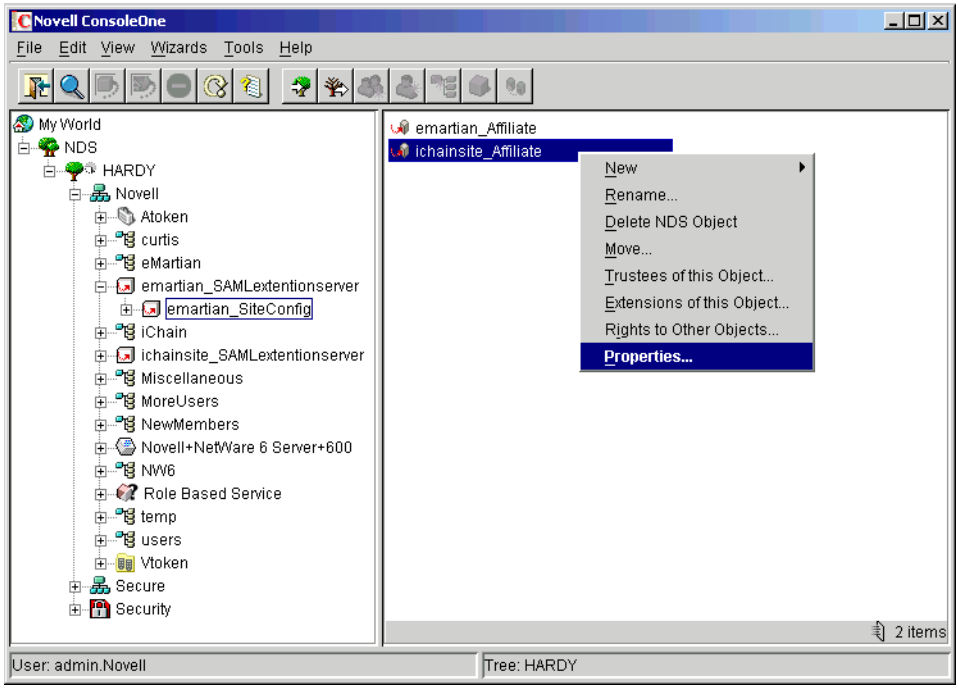
6 Click Finish.

This completes the trusted root import. You should now have a new object in the Trusted Roots container.

Next, you need to associate this object with your Trusted Affiliate entry for www.ichainsite.com.

7 Right-click the iChainSite Trusted Affiliate object, and select Properties.

Figure 87 iChainSite Trusted Affiliate Object Properties

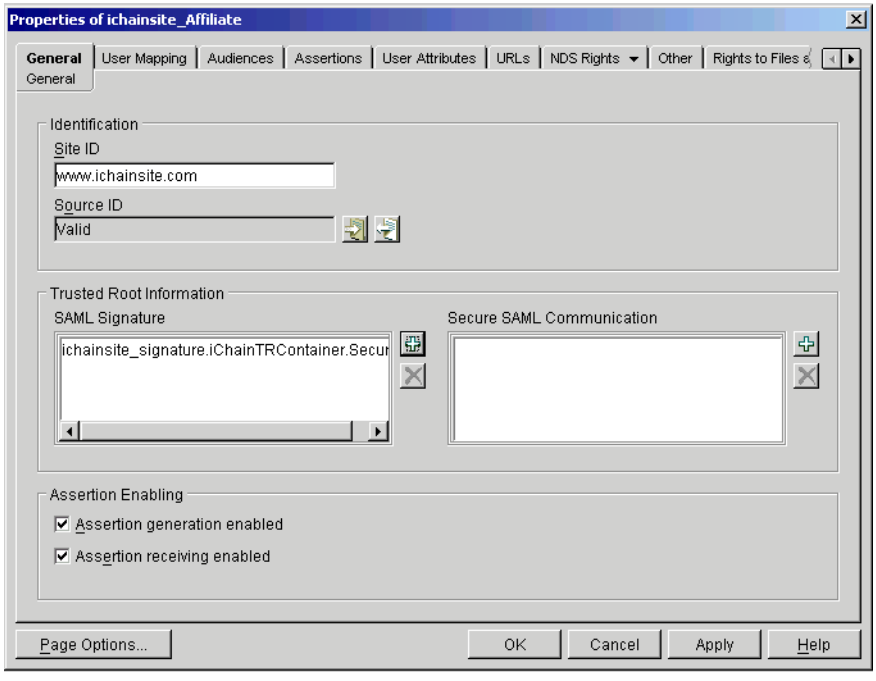


8 Click the General tab, then select the plus sign (+) under the SAML Signature section.

9 Browse to the Trusted Roots container and select the desired trusted root object.

Figure 88 shows the www.ichainsite.com Properties page with the trusted root reference set:

Figure 88 iChainSite Properties: Trusted Root Reference Set

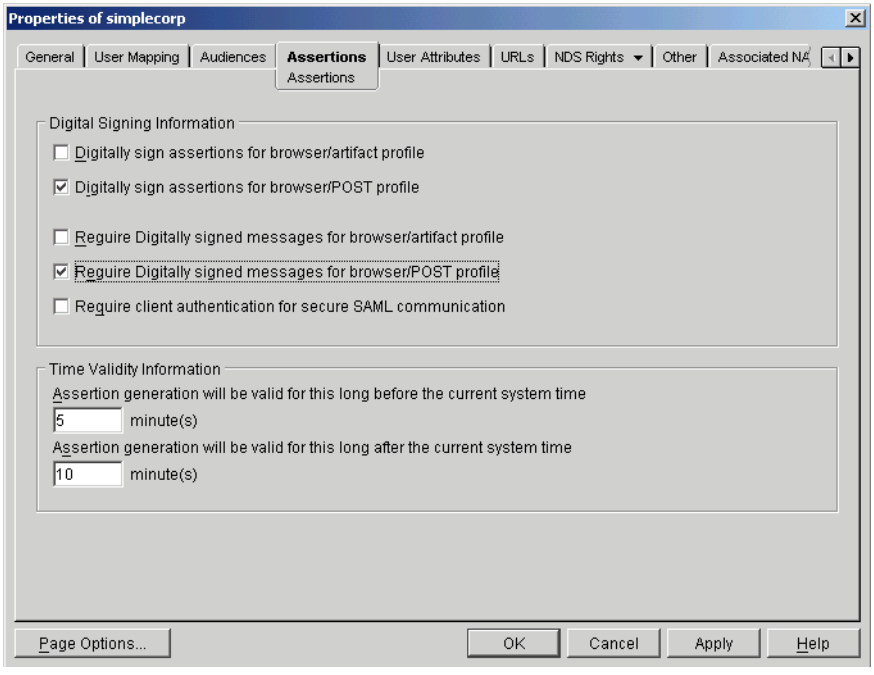


Creating this setting will allow the SAML system to validate data signed by www.ichainsite.com.

10 Click OK.

11 Verify that signature validation occurs by clicking the Assertions tab and selecting the Require Digitally signed messages for browser/POST profile check box, as shown in Figure 89:

Figure 89 iChainSite Properties: Assertions Tab



Choosing this setting ensures that all SAML messages sent from www.ichainsite.com to eMartian using the browser/POST profile must be signed and validated or they will not be accepted.

Testing Digital Signatures

After completing the steps in the previous section, SAML data sent from www.ichainsite.com to eMartian using the browser/POST profile will be digitally signed. You can validate this by checking the log messages on each SAML extension server. When assertions are generated, you can see the XML signature being generated.

Configuring SAML to Support SSL Mutual Authentication

Do the following steps to configure the SAML system to support SSL Mutual authentication over the SAML back-channel:

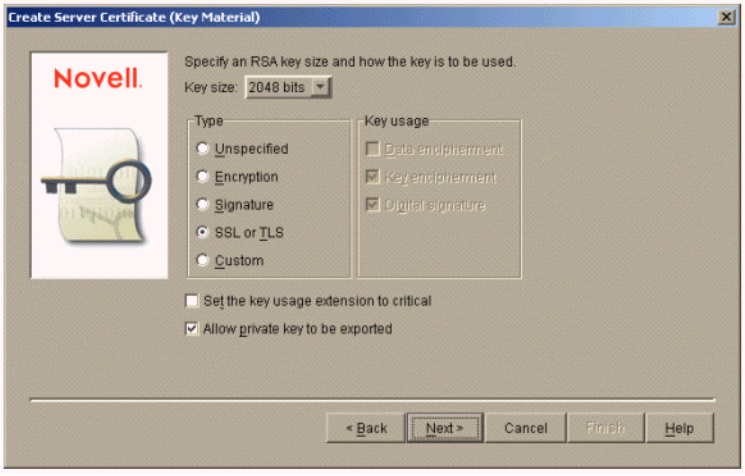
1. Generate an SSL Key Pair (SKP) or import it into the system and store it in eDirectory.
2. Export the SKP from eDirectory in PKCS#12 format and store it on the SAML extension server.
3. Use the SKP as the SSL Key Pair on the appropriate iChain accelerator.

4. Import the SKP into the SAML extension server.
5. Export the Public Key Certificate associated with the SKP and send it to the Trusted Affiliate.
6. Import the partner's SSL public key certificate (that it sends to you) into your trust store.
7. Configure the appropriate settings on the Trusted Affiliate who will be communicating with your site over SSL-M.

Generating the SSL Key Pair

You can use the Novell Certificate Server snap-ins to generate your SSL Key Pair. If you choose to do this, the steps required are nearly identical to those followed to generate the data signing key. The only difference is at the Create Server Certificate tab, rather than selecting the Signature radio button as you did when you were generating the signature Key Pair, you should select the SSL or TLS button as shown in Figure 90:

Figure 90 Create Server Certificate (Key Material)



Exporting the SSL Key

The steps for exporting the SSL key are identical to those you used to export the signing Key Pair. Be sure to select the public key certificate tab and remember the export password (you will need it when the Key Pair is imported).

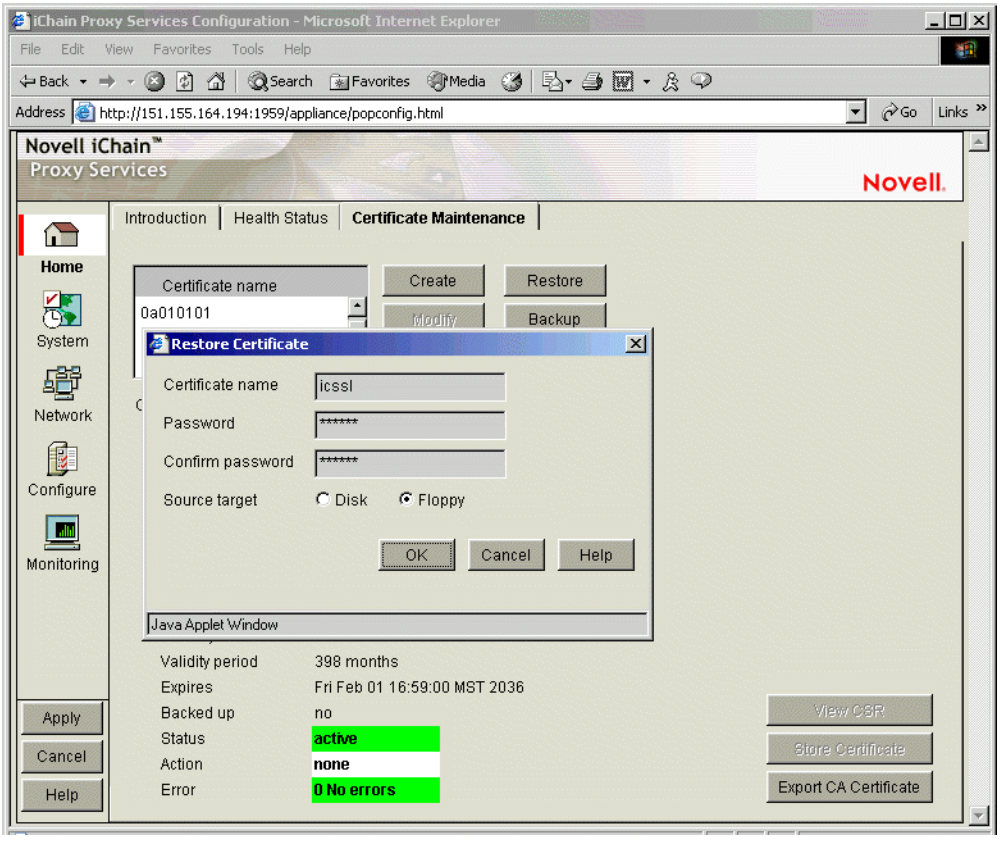
Importing the SSL Key Into iChain

You can use the same Key Pair for SAML SSL as you do for SSL on your iChain accelerator. If you already have an acceptable certificate in use on the iChain accelerator, you can use the iChain GUI to export it into a PKCS#12 file and then import it into eDirectory. Alternately, you may import a PKCS#12 Key Pair exported from eDirectory into iChain.

NOTE: In order to import a PKCS#12 file into iChain, the file must be in 8.3 format (the file must have a maximum of 8 characters in the name, and a 3 character extension). Copy the PKCS#12 file you exported onto a floppy and rename it to fit the 8.3 format.

Next, open iChain GUI on the iChain server. Select the Certificate Maintenance option and click the Restore button. A screen is displayed as shown in Figure 91:

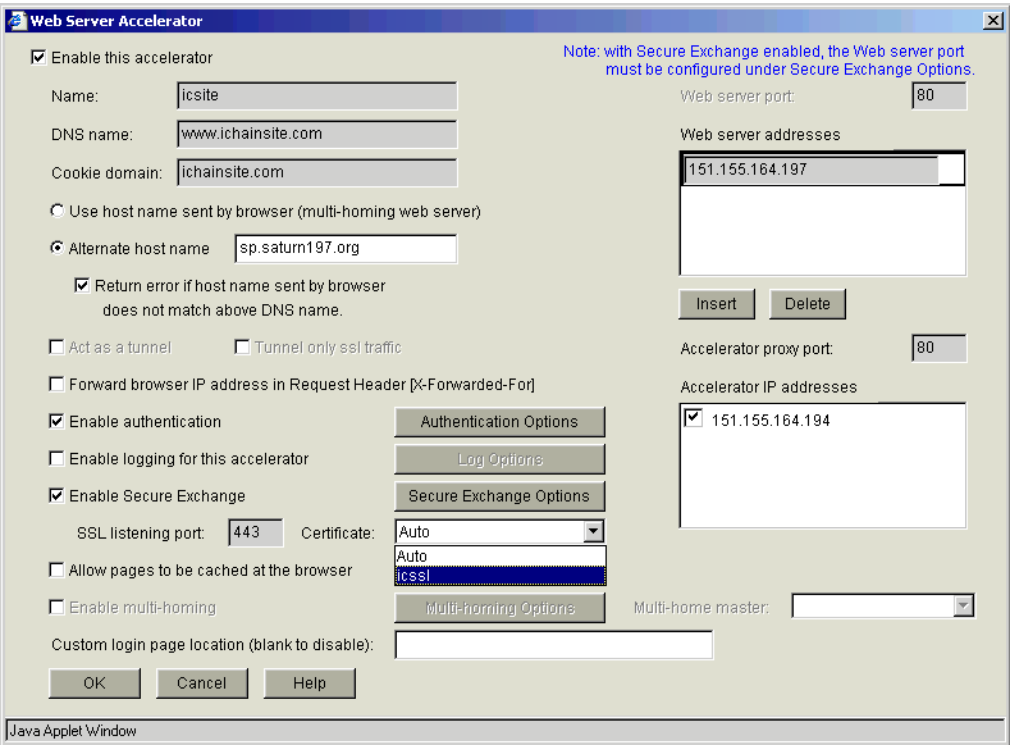
Figure 91 Certificate Maintenance: Restore Certificate



Enter the PKCS#12 file name, enter the appropriate password, and select Floppy. Click OK, then click the Apply button on the left-hand side of the GUI. The status and action indicators on the Certificate Maintenance page will show whether the operation was successful.

Next, you need to configure your www.ichainsite.com accelerator to use this certificate for SSL. Select the Configure option and select the Web Server Accelerator tab. Select the appropriate accelerator and click on the Modify button. A page is displayed as shown in Figure 92:

Figure 92 Web Server Accelerator



Click the Certificate menu and select the appropriate certificate, then click OK to apply the changes.

Importing and Configuring the SAML Extension Server to Use the SSL Certificate

The same as you did in the signing key case, you must get the PKCS#12 file exported from eDirectory onto the local file system of the SAML extension server. Then you must modify the SAML extension server's configuration file to use it.

To import the Signature Key file into the SAML extension server for use:

1. Copy the PKCS#12 file exported in the previous process to the local drive of the SAML extension server.
2. Modify the SAML extension server configuration file to point to this file.

When you installed the SAML extension software, a configuration file was automatically generated. This file is located at *SAMLEXT_HOME*/config/samlextConfig.xml. After modifying this file to handle the signing key, it should look similar to Figure 93:

Figure 93 Modifying the samlextConfig.xml File

```
<authority name="SAMLExtDirectoryAuthority">
        <class><name>com.novell.wss.authority.saml.SAMLExtDirectoryA
uthority</name></class>
        <data>
                <servers>
                        <url>ldap://137.65.159.66:389</url>
                </servers>
                <proxyUser>
                        <dn>cn=admin,o=novell</dn>
                        <password>novell</password>
                </proxyUser>
                <isoDn>cn=iso,o=novell</isoDn>
                <initialCapacity>10</initialCapacity>
                <maxCapacity>30</maxCapacity>
                <keypairs>
                        <keypair usage="ssl" type="jks">
                                <password></password>
                                <file></file>
                        </keypair>
                        <keypair usage="signing" type="pkcs12">
                                <password>novell</password>
                                <file>c:\mysig_keypair.pfx</file>
                        </keypair>
                </keypairs>
        </data>
</authority>
```

Modify the signature keypair element with ssl usage to include the file name and password of the SSL Key Pair PKCS#12. As shown in Figure 94, you would modify this file to read as:

Figure 94 Modifying the Signature KeyPair Element with SSL Usage

```
<authority name="SAMLExtDirectoryAuthority">
        <class><name>com.novell.wss.authority.saml.SAMLExtDirectoryA
uthority</name></class>
        <data>
                <servers>
                        <url>ldap://137.65.159.66:389</url>
                </servers>
                <proxyUser>
                        <dn>cn=admin,o=novell</dn>
                        <password>novell</password>
                </proxyUser>
                <isoDn>cn=iso,o=novell</isoDn>
                <initialCapacity>10</initialCapacity>
                <maxCapacity>30</maxCapacity>
                <keypairs>
                        <keypair usage="ssl" type="pkcs12">
                                <password>novell</password>
                                <file>c:\myssl_keypair.pfx</file>
                        </keypair>
                        <keypair usage="signing" type="pkcs12">
                                <password>novell</password>
                                <file>c:\mysig_keypair.pfx</file>
                        </keypair>
                </keypairs>
        </data>
</authority>
```

This assumes that you copied the exported PKCS#12 file to the SAML extension server as c:\myssl_keypair.pfx using novell as the password.

Exporting the iChain Server SSL Public Key Certificate

In order for your partners to accept SSL connections from you, they must have and trust the public key associated with your SSL Key Pair. You must export the public key certificate and send it to your partners so that they can create this trust relationship.

The easiest way to get the SSL Public Key for the ichainsite is to do the following:

1. Connect to the ichainsite accelerator using Internet Explorer.
2. Double-click on the lock in the lower right-hand corner, then click the Certification Path tab.
3. Highlight the CA (the top item).

4. Click View Certificate, then click the Details tab.
5. Copy to file.
6. Select Base-64, then click Next.
7. Name the file ichainsite.b64, then save it.

Importing the Partner's SSL Public Key Certificate

In order to create mutually authenticated SSL connections, the following two conditions must be in place:

1. Your partners must validate and trust your server certificate.
2. You must trust your partners' server certificates.

Thus, just as you sent your SAML partner site your SSL public key certificate, you must receive from your SAML partner site(s) their SSL public key certificate(s) and import them into your system's trust store.

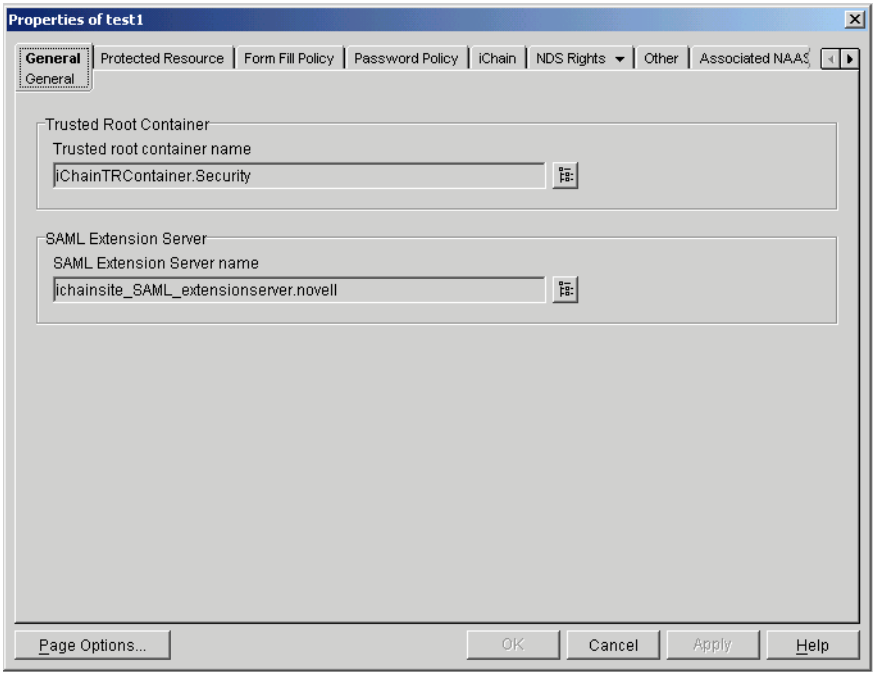
It is assumed that your partner site, eMartian, has generated an SSL Key Pair and sent you (iChainSite) its SSL public key certificate.

1. Connect to the eMartian site accelerator using Internet Explorer.
2. Double-click on the lock in the lower right-hand corner, then click the Certification Path tab.
3. Highlight the CA (the top item).
4. Click View Certificate, then click the Details tab.
5. Copy to file.
6. Select Base-64, then click Next.
7. Name the file eMartian.b64, then save it.

You import eMartian's SSL certificate into your Trusted Root container and associate it with the eMartian Trusted Affiliate object.

NOTE: Each iChainServiceObject can associate with a Trusted Root container. You must verify that the SSL trusted roots that you use are imported into this container. Otherwise, SSL connections will not work through iChain. The Trusted Root container associated with the iChainServiceObject can be determined by opening the iChainServiceObject's Properties page and selecting the General tab, as shown in Figure 95:

Figure 95 iChainServiceObject Properties Page

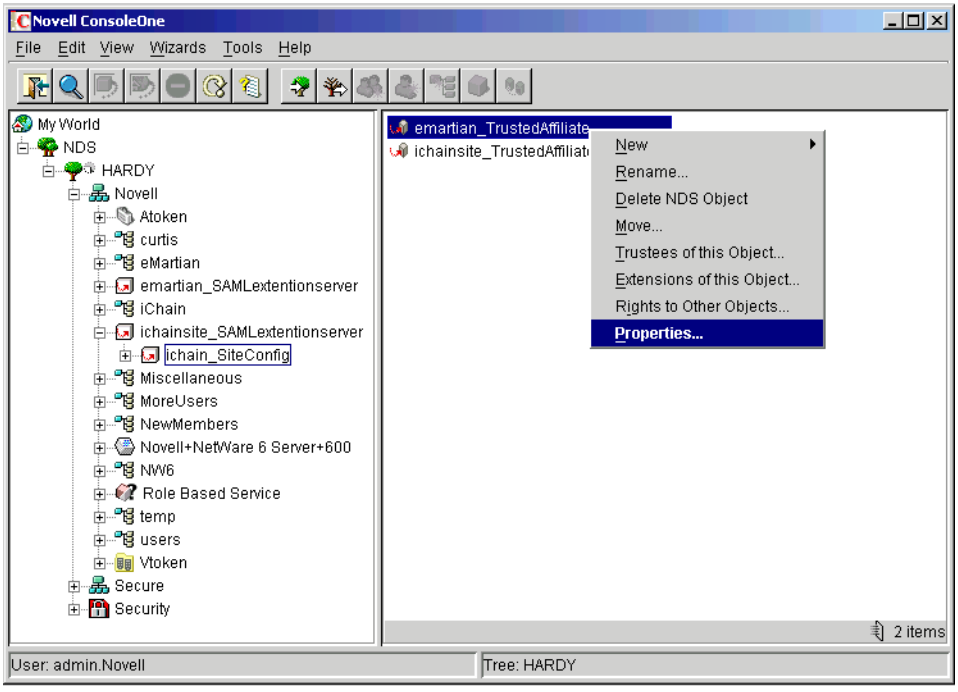


If you haven't set the the value, you can set it by selecting the Browse button on the right. If no Trusted Root container has been created, follow the steps outlined in "Importing Public Key Certificates" on page 75 to create one.

After setting this Trusted Root container name attribute in the iChainServiceObject, browse to the container and import the eMartian.b64 certificate. The certificate can be imported by following the steps outlined in "Importing Public Key Certificates" on page 75.

After importing the certificate into the appropriate Trusted Root container, you must configure the SAML extension server to use it. In this example, you are administering ichainsite and are receiving a certificate from eMartian, so you would select the Trusted Affiliate object associated with eMartian as shown in Figure 96:

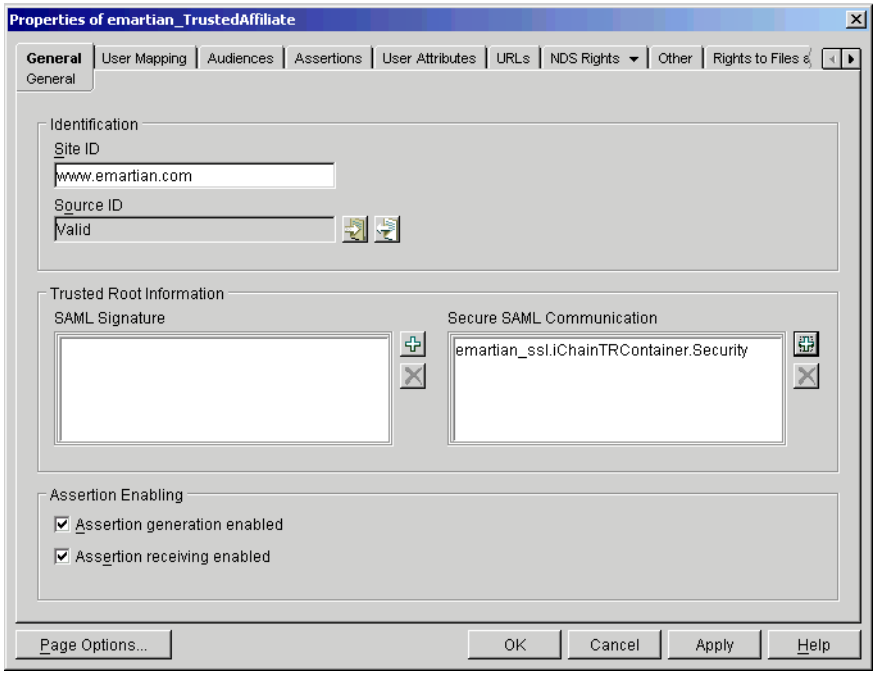
Figure 96 eMartian Trusted Affiliate Object



To configure the SAML extension server to use the certificate:

1. Right-click the object, then select Properties.
2. Select the General tab, then click the plus sign (+) associated with the Secure SAML Communication group.
3. Select the SSL certificate associated with this Trusted Affiliate. In this example, the certificate is the eMartian SSL public key certificate, as shown in Figure 97:

Figure 97 eMartian SSL Public Key Certificate



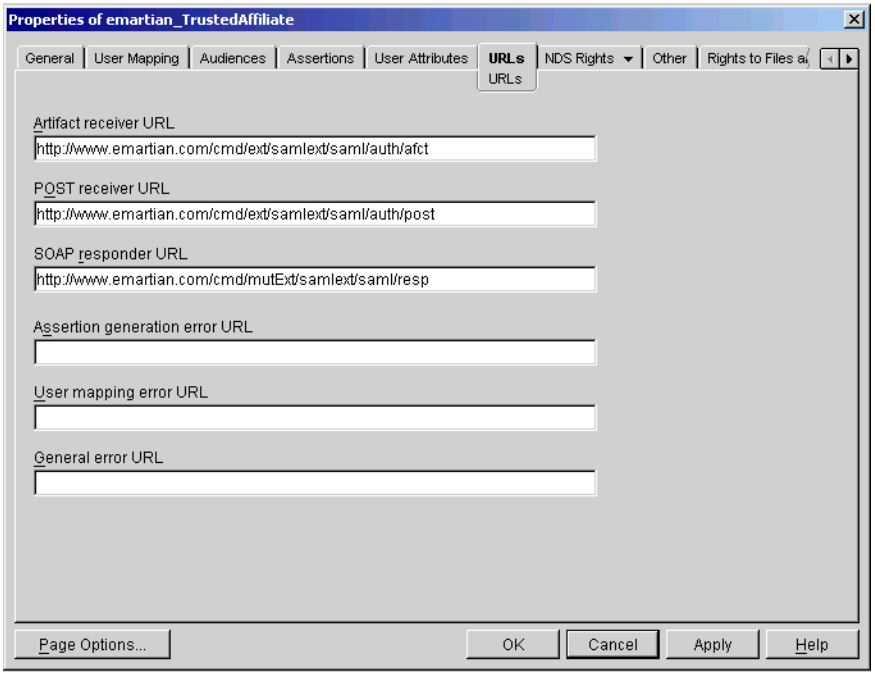
4 Click OK.

5 Go to the eMartian site and import the ichainsite.b64 you saved earlier.

Modifying the SAML SOAP Endpoint URL

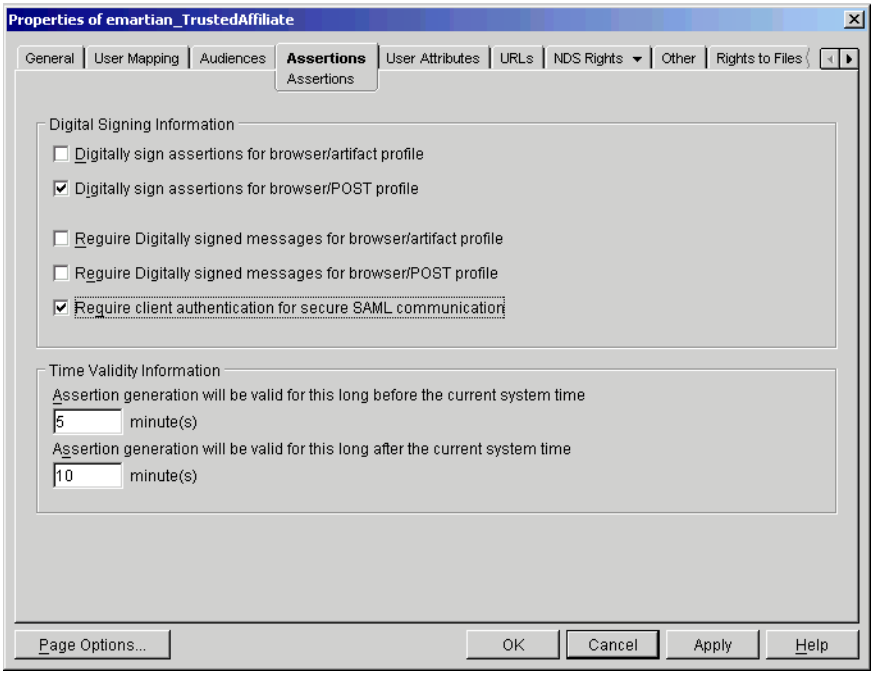
After you are set up for SSL, you can modify the SOAP Endpoint associated with the eMartian affiliate to make use of this new security. There are two separate ways of accessing the SAML back-channel. The first is through the /cmd/ext URL switch. This extension can handle clear text and server side SSL. The second is through the /cmd/mutExt URL switch. This runs only SSL-M. If you modify the SOAP endpoint of the eMartian site to include the /cmd/mutExt switch rather than the /cmd/ext switch, SSL-M will be used. This setting is made at the URLs tab. Figure 98 shows this setting:

Figure 98 Modifying the SOAP Endpoint for eMartian



Note that now the SOAP Responder URL contains /cmd/mutExt rather than /cmd/ext. You can require that affiliates communicating with you over the SAML back-channel use SSL-M. This setting is made at the Assertions tab. Figure 99 shows this setting:

Figure 99 eMartian Properties: Require Client Authentication



With the Require client authentication for secure SAML communication setting enabled, only connections with SSL-M with a certificate matching that in the Secure SAML Communication field will be accepted.