



The Access Assurance Suite Implementation Guide

Release 9.1

Core Security SDI Corporation

**1000 Holcomb Woods Parkway
Suite 401**

Roswell, GA 30076

Phone: (678) 304-4500

Fax: (770) 573-3743

Trademarks

Copyright © 2018 by Core Security SDI Corporation. All Rights Reserved. The following are trademarks of Core Security Corporation "Core Impact", "Core Vulnerability Insight", "Core Password", "Core Access", "Core Provisioning", "Core Compliance", "Core Access Insight", "Core Mobile Reset", and "Think Like an Attacker". The following are registered trademarks of Core Security Corporation "WebVerify", "CloudInspect", "Core Insight", and "Core Security". The names of actual companies and products mentioned herein may be the trademarks of their respective owners. The names of additional products may be trademarks or registered trademarks of their respective owners.

Table of Contents

Chapter 1 - Overview of Implementation Options	7
Chapter 2 - Web Access (ASP Page) Configuration	9
Configuration Prerequisites	10
Web Access Configuration Settings	11
Optional Configuration Information	13
PasswordCourier Classic with Web Access	14
PasswordCourier Classic Configuration	14
Using Client Side Components	14
PasswordCourier Support Staff Classic with Web Access	15
PasswordCourier Support Staff Classic Configuration	15
ProfileCourier Classic with Web Access	16
ProfileCourier Classic Configuration	16
The Provisioning Platform	17
Configuration	17
Access Assurance Suite Administration Manager Configuration	17
Web Access Configuration	17
Using Client Side Components	18
Client Side Components	19
Client Side Configuration	19
Core Security Client Manager Installation	19
Using Silent Installations	22
Silent Installation Log Files	22
Creating Tailored Silent Install Response Files	23
Examples of Silent Configuration Commands	23
Configuration Switches	24
Server Side Configuration for the Provisioning Platform	25
Modifying and Submitting the Unique Resource Data Form When Upgrading	26
Updating the Password Reset Action's Summary Page	26
Modifying the JavaScript Template	26
JavaScript Template	27
Formatting the output from the CourLocalControl ActiveX control's processing	28
Server Side Configuration for the Classic Platform	29
Chapter 3 - Guidelines for Modifying Sample Active Server Pages	31
Introduction	32
General ASP Information	33
Back Up of Default and Customized ASPs	35
File Paths in ASP Files	35
Guidelines for Modifying ASPs	36
Customizable ASPs	36
PasswordCourier Classic and PasswordCourier Support Staff Classic	36
ProfileCourier Classic	36
The Provisioning Platform	37
Details on Specific ASPs	37
Helper Routines	37
Cascading Style Sheet Customization	38
JavaScript Customization	39
COM Interfaces for the Classic Platform	40
ValidateUser COM Interface Properties	40
ValidateUser2 Interface Properties	40
AuthenticateUser COM Interface Properties	40
Target COM Interface Properties	41

Password COM Interface Properties	41
Reset Password COM Interface Properties	42
Profile COM Interface Properties	42
ProfileAction COM Interface Properties	43
Chapter 4 - Best Practices for Implementing the Connector for Web Services	45
Implementation Overview	46
Best Practices	48
Avoid Iterations	48
Avoid Tag Attributes	49
Avoid SOAP Header Definitions	49
Handle Namespacing Properly	49
Use SOAP Error/Fault Mechanism for Reporting Errors	50
Keep Request and Response XML Payloads Similar	50
Types of WSDLs That Can Be Auto-Discovered	53
Chapter 5 - Configuring the Trusted Connection for Core Security Services and the Access Assurance Portal	55
Setting Up a Service Account	56
Minimum Privileges Required for the Service Account in Active Directory	56
Configuring Trusted Connection in the Database	57
Configuring Trusted Connection Authentication For Core Security Services	58
Configuring the Identity Mapping Service	58
Configuring the Core Security Notification Service	59
Configuring the Core Security Request Service	60
Configuring the Core Security Scheduler Service	62
Configuring Trusted Connection for the Access Assurance Portal	64
Configuring the Internet Information Services	66
Chapter 6 - Securing the Access Assurance Portal	69
Using Transport Layer Security (TLS)/ Secure Sockets Layer (SSL)	70
Configuring TLS/SSL	70
Configuring the Web Services	74
Configuring the Tomcat Server	77
Configuring Force Protocol Encryption	79
Chapter 7 - Multilanguage Support	81
About Multilanguage Support	82
Overriding the Default Language for any End User	83
Overriding the Default Language and Culture for any End User	83
ISO 639-1 Code	84
Web Access (ASP) Option Multilanguage Mapping	85
Sample Steps to Define ASP Multilanguage Mapping for English and French Languages	85
Sample Steps to Add Support for a Language and Culture for the Access Assurance Portal	89
Creating a Language-Specific XML File	90
Adding Static Text to an End User Page	91
Translating Dynamic Strings	92
AuditLink Multilanguage Mapping	94
Using the %Behavior.Language% Macro to Populate Email and Ticketing Events	94
Accessing the Translation Table with AccountCourier or ComplianceCourier	98
Sample Steps to Define AuditLink Multilanguage Mapping for PasswordCourier or ProfileCourier	98
Simplifying Third-Party Translation	99
Chapter 8 - Enabling Multi-Byte Connectors and PMMs	101
Connectors and PMMs that have been Enabled for Use with Multi-Byte Languages:	101
Overview	102
Enabling Multi-Byte Connectors	103
Standard Connectors	103
RDK Connectors	103
Enabling Multi-Byte PMMs	104
Branded PMMs	104
Branded RDK PMMs	104

Unbranded RDK PMMs	105
Chapter 9 - Configuring the XML Access Option	107
About the XML Access Option	108
SPML Schema Support	109
Creating an SPML Template Document	114
Formatting SPML Documents	114
Sample Modify Request Template	115
SPML Add Request	116
Sample Template for a Delete Request	117
Sample Template for an Extended Request	118
Configuring Options in the Administration Manager for Automating a Workflow	120
SPML Code Sample	120
Authentication Form	121
Action Selection Form	121
Community Restrictions	121
Requirement for Automating the Verify Action	122
Initial Profile and Resource Overrides Automation	122
Testing the SPML Code	124
Configuring ASP Pages to Send SPML to the Core SecurityService	125
Step 1	125
Step 2	125
Chapter 10 - The Provisioning Listener for PeopleSoft	127
Requirements	128
Configuring PeopleSoft for Provisioning Listener for PeopleSoft	130
Chapter 11 - Optimizing the Performance of the Transaction Repository	135
Using the Microsoft SQL Server Index Tuning Wizard	136
Purging the Transaction Repository	137
Chapter 12 - Configuring PasswordCourier for Transparent Synchronization	139
Overview	140
The Transparent Synchronization Service	140
The Transparent Synchronization Listener	140
Transparent Synchronization Workflow	141
The Transparent Synchronization Workflow Template	141
The Tsync.asp Page	141
Requirements	142
Transparent Synchronization Access Key	142
Sample Transparent Synchronization Configuration	143
Customizing the Transparent Synchronization Workflow Template	144
Configuring Targets	144
Set Up the Unique Resource Data Form	146
Installing the Transparent Synchronization Service and the GAT Service	148
Installing the GAT Service	148
Running the Distributed COM Configuration Utility in Environments with Multiple Core Security Servers	
149	
Accessing the CourGatService Properties	149
Installing the Transparent Synchronization Service	152
Using the Start Menu to Manage the Transparent Synchronization Service and the GAT Service ...	156
Enabling or Disabling Logging for the Transparent Synchronization Service and the GAT Service ...	156
Installing the Transparent Synchronization Listener	157
Transparent Synchronization Listener for Microsoft Windows	158
Requirements	158
Installing the Listener	158
Configuring the Listener	159
Transparent Synchronization Listener for Microsoft Server Core for Windows Server	161
The Listener Files	161
ListenerRegistry.xml	162
PSListener.ps1	166
Setup.iss	166

Uninstall.iss	166
Installing the Listener on Microsoft Windows Server Core	166
Modifying the Listener on the Server Core for Microsoft Windows Server	167
Uninstalling the Listener on the Server Core for Microsoft Windows Server	167
Transparent Synchronization Listener for i5/OS	168
Requirements	168
Installing the TSL for i5/OS	168
Configuring the TSL for i5/OS	172
TSL for i5 /OS (OS/400) Menu	173
Maintain the TSL Configuration	174
Maintain TSL Specific Excluded Profiles	178
Purge Log File	179
Print TSL Configuration	180
Print Log File	181
Display TSL Output Queue	182
Uninstalling the TSL for i5/OS	182
Notes and Warnings	184
Chapter 13 - Using the Configuration Repository	185
Configuring a Transaction Repository	185
About Sharing Data	186
Running Configuration Repository in Local Mode	186
Running the Configuration Repository in Remote Mode	186
Transferring Shared Data from the Core Security Server To the Shared Repository	187
Backing up Shared Data on the Shared Repository	187
Using the Configuration Management Wizard	188
Configuring Local Mode	188
Configuring Remote Mode	190
Notes on Running the Core Security Server in a Shared Environment	193
When Configuring a Core Security Server in a Shared Environment	193
When Making Modifications To Applications in a Shared Environment	193
Chapter 14 - Input Validation for End-User Workflows	195
Enabling Server-Side Input Validation	196
Configuring Options in the CustomInputValidation.xml File	197
Notes	197
Internationalization of Input Validation Policy Violations	198
Chapter 15 - Monitoring Configuration and Performance Data	199
Monitoring Data in the CIM Repository	200
Using the Microsoft Windows Performance Monitor	201
Modifying the Graph	203
Chapter 16 - Sample Programs Used to Generate Resource Names	205
Appendix A - Multi-Byte Notes	209
Multi-Byte Notes for the Access Assurance Suite	210
Access Assurance Suite Installation	210
The Sort Order of Lists Returned by the Core Security Server	210
Multi-Byte Notes for Password Management Modules (PMMs) and Connectors	211
The Microsoft Windows Login Interface	211
Administrator User Names and Passwords	211
Multi-Byte Support is for End-User Interfaces	211
Password Strength Rules	211
Password Management Modules that are Not Multi-Byte Capable	212
Branded Custom Password Management Modules	212

Chapter 1: Overview of Implementation Options

This manual describes how to use Access Assurance Suite implementation tools and includes the following chapters:

- [*"Web Access \(ASP Page\) Configuration" on page 9*](#)
- [*"Guidelines for Modifying Sample Active Server Pages" on page 31*](#)
- [*"Best Practices for Implementing the Connector for Web Services" on page 45*](#)
- [*"Configuring the Trusted Connection for Core Security Services and the Access Assurance Portal" on page 55*](#)
- [*"Multilanguage Support" on page 81*](#)
- [*"Enabling Multi-Byte Connectors and PMMs" on page 101*](#)
- [*"Configuring the XML Access Option" on page 107*](#)
- [*"The Provisioning Listener for PeopleSoft" on page 127*](#)
- [*"Optimizing the Performance of the Transaction Repository" on page 135*](#)
- [*"Configuring PasswordCourier for Transparent Synchronization" on page 139*](#)
- [*"Using the Configuration Repository" on page 185*](#)
- [*"Input Validation for End-User Workflows" on page 195*](#)
- [*"Monitoring Configuration and Performance Data" on page 199*](#)
- [*"Sample Programs Used to Generate Resource Names" on page 205*](#)

Chapter 2: Web Access (ASP Page) Configuration

Web Access is a web-based interface for the Core Access Assurance Suite which allows end users to quickly and easily change their own password, profile, and other information using a web browser. Because it is web-based, you can customize its appearance.

Web Access interfaces with the Core Server through a TCP/IP Secure Socket Layer (SSL) connection between the web client and the Core Server.

This chapter describes the procedures necessary to install and configure Core Access Assurance Suite Web Access and includes the following sections:

- [*"Configuration Prerequisites" on page 10*](#)
- [*"Web Access Configuration Settings" on page 11*](#)
- [*"Optional Configuration Information" on page 13*](#)
- [*"PasswordCourier Classic with Web Access" on page 14*](#)
- [*"PasswordCourier Support Staff Classic with Web Access" on page 15*](#)
- [*"ProfileCourier Classic with Web Access" on page 16*](#)
- [*"The Core Provisioning Platform" on page 17*](#)
- [*"Client Side Components" on page 19*](#)

Configuration Prerequisites

Web Access requires a module-specific access key which can be obtained from Core Security. For information about the access key, please contact Core Security.

The following are required for Web Access configuration:

- Name of the host running the Core Server
- TCP/IP port used by the Core Server

Web Access Configuration Settings

The settings in the Web Access configuration managers are used to customize the runtime behavior of Web Access. These settings are encrypted in the configuration file. Only Core Security products can decrypt the information within this file. To access the Web Access Configuration manager from the Start Menu, use one of the following sequences, depending on the product you are using:

For PasswordCourier Classic and ProfileCourier Classic:

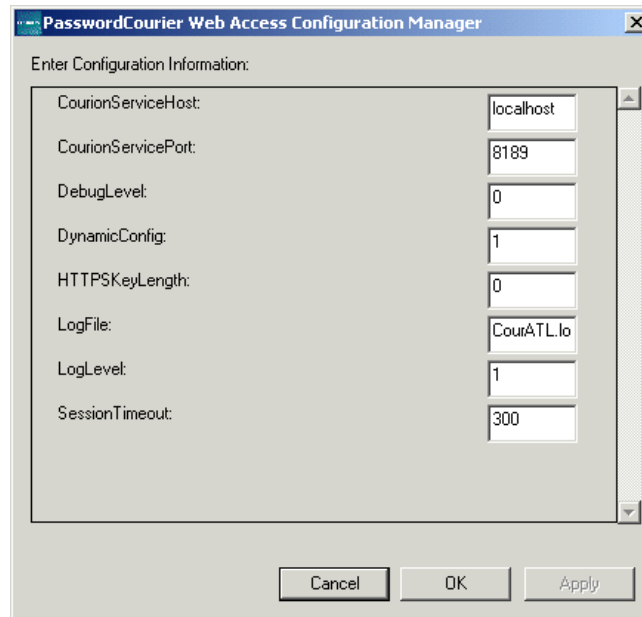
Start>All Programs>Core Access Assurance Suite>*product name*>Web Access Configuration

For Provisioning Platform products:

Start>All Programs>Core Access Assurance Suite>Web Access Configuration

All Web Access Configuration managers use the same configuration settings and the same interface for managing these configuration settings. The only difference between them is the title bar, which is specific for each product. An example of a Web Access configuration manager is shown below in [Figure 1](#).

Figure 1: Sample Web Access Configuration Manager



The configuration settings for all Web Access applications are detailed in [Table 1](#) below.

Table 1: Configuration File Settings for Web Access Applications

Parameter Name	Description	Default Setting
CourionServiceHost	The IP address or hostname of the system running the Core Server.	localhost
CourionServicePort	The port number that was configured during Core Server configuration, which is the port number on which the Core Security Server is listening.	8189

Table 1: Configuration File Settings for Web Access Applications

Parameter Name	Description	Default Setting
DebugLevel	<p>A value of 0 means no debugging information is printed to the browser; value 1 means debugging information is printed. Valid values are 0,1.</p> <p>This flag is useful for debugging your configuration.</p> <p>Note: AccountCourier does not automatically return debugging information when this flag is set to 1. Rather, it displays a checkbox on the interface that, when checked, returns debugging information.</p>	0
DynamicConfig	<p>0 means store configuration information in cache.</p> <p>1 means do not store configuration information in cache.</p> <p>Valid values are 0 or 1.</p> <p>When changing and testing Core Server and application configurations, set this value to 1 to ensure the latest configuration is loaded every time. When you finish changing the configurations, set this flag back to 0 so that the configuration can be reused quickly from the cache.</p>	1
HTTPSKeyLength	<p>Minimum session key length for the SSL connection between the web browser and the IIS Web Server.</p> <p>A value of 0 indicates that SSL is not used. In that case connect to the ASPs using <code>http://hostname/virtualpath/</code>.</p> <p>A value greater than 0 indicates that the session key negotiated between the web browser and the web server must be at least that many bits long. In that case, connect to the ASPs using <code>https://hostname/virtualpath/</code>.</p> <p>Core Security recommends using the highest key length available to the site's web server and browsers.</p> <p>The request is rejected if a key length greater than that supported by the web server or web browsers is chosen.</p>	128
LogFile	Specifies name of the log filename. Do not include the path. The log file is created in the Core Server folder.	CourATL.log
LogLevel	A value of 0 means no logging, Value 1 means log information to the file specified by LogFile. Valid values are 0 or 1.	1
SessionTimeout	Represents the period of time, in seconds, a user session can be left idle before it is terminated.	300 seconds

Optional Configuration Information

When installing the product, the Active Server Pages (ASP) used for the Web Access interface are copied to the installation folder. These ASP pages need to be placed in an area where the web server can execute them. For more information on this topic see the [“Guidelines for Modifying Sample Active Server Pages” on page 31](#) guide.

You can install and configure the Web Access interface on a separate server such as a web server or an intranet server. Install the Core Access Assurance Suite on that server, but instead of configuring the entire product, provide only the access keys for Web Access configuration. Then configure Web Access as needed.

When installation of any Core Security product with web access is complete, create a virtual directory for the web server to point to the “www” folder. This folder may also be copied (not moved) to the web server.

PasswordCourier Classic with Web Access

PasswordCourier Classic with Web Access uses the same configuration parameters as other PasswordCourier access methods such as Core Access Assurance Suite with DIRECT! Access: it authenticates the end user in the same manner, executes the same request tracking actions, and presents the same password targets for resets for all PasswordCourier configurations. PasswordCourier can only be configured in one particular way at any time, and that configuration defines how it works with all access methods. See the manual *Using PasswordCourier and PasswordCourier Support Staff Classic* for more information.

If PasswordCourier Classic with Web Access is installed on a separate machine from the Core Server, you need to specify the location of the Core Server during the configuration of PasswordCourier with Web Access.

PasswordCourier Classic Configuration

Use the settings in the configuration manager to customize the runtime behavior of PasswordCourier with Web Access. To access the PasswordCourier with Web Access Configuration Manager, follow the path below:

Start>All Programs>Core Access Assurance Suite>PasswordCourier Classic>Web Access Configuration

Configure Web Access as appropriate for your environment. For information on how to configure these settings, please refer to the [“Web Access Configuration Settings” on page 11](#).

Using Client Side Components

Client side components are available for use with the PMMs for Active Directory, Lotus Notes, and Windows NT/2000. For details see [“Client Side Components” on page 19](#).

PasswordCourier Support Staff Classic with Web Access

PasswordCourier Support Staff with Web Access authenticates the end user in the same manner as PasswordCourier with Web Access, executes the same request tracking actions (with one exception noted below), and presents the same password targets for resets for all PasswordCourier configurations. PasswordCourier Support Staff can only be configured in one particular way at any time, and that configuration defines how it works with all access methods. See the manual *Using PasswordCourier and PasswordCourier Support Staff Classic* or more information.

Note: If Peregrine ServiceCenter® is being used as the help desk, then additional ticket reuse configuration options are offered. See the manual *Using PasswordCourier and PasswordCourier Support Staff Classic* for more information on these options.

PasswordCourier Support Staff Classic Configuration

Use the settings in the configuration manager to customize the runtime behavior of PasswordCourier Support Staff Classic with Web Access. To access the PasswordCourier Support Staff with Web Access Configuration Manager, follow this path:

Start>All Programs>Core Access Assurance Suite> PasswordCourier Classic>Support Staff Web Access Configuration

Configure Web Access as appropriate for the system environment. For information on how to configure these settings, please refer to [“Web Access Configuration Settings” on page 11](#)

PasswordCourier Support Staff Classic with Web Access allows support staff to prefill user information via a query on existing data. To take advantage of this functionality, open a browser on a computer with access to PasswordCourier Support Staff. In the url address type:

```
http://machinename/path/default.asp?staff1=x;staff2=x;user1=x;user2=x
```

You need to use a full url for this feature to work. “Machinename” represents the name of the server or machine running PasswordCourier Support Staff. “Path” is the path. Everything following the question mark (“?”) is a part of the query string. Support staff users should include the data they want prefilled as part of this query string. In the example above, “x” represents the information that PasswordCourier Support Staff with Web Access should pull from the database to prefill the user information page. This information must be typed into the query string. Entries must be in numeric sequence (example: you cannot add user4 to the query string before users 1–3).

To specify a ticket to reuse, add “TicketID=” to the end of the query string along with the appropriate Ticket ID number after the equal (“=”) sign.

This functionality can be used from a browser even if the support staff user is not logged in to PasswordCourier Support Staff. If the user is not logged in, then they need to authenticate themselves by using the staff1 and staff2 fields.

ProfileCourier Classic with Web Access

You can configure ProfileCourier Classic in only one way at any time, and that configuration defines how it works with all access methods. See the manual *Using ProfileCourier Classic* for more information.

ProfileCourier Classic Configuration

Use the settings in the configuration manager to customize the runtime behavior of ProfileCourier with Web Access. To access the ProfileCourier Classic with Web Access Configuration Manager, follow the path below:

Start>All Programs>Core Access Assurance Suite>ProfileCourier Classic>Web Access Configuration

Configure Web Access as appropriate for your environment. For information on how to configure these settings, please refer to [“Web Access Configuration Settings” on page 11](#).

The Core Provisioning Platform

Products on the Core Provisioning Platform (AccountCourier, ComplianceCourier, RoleCourier, PasswordCourier, and ProfileCourier) require that you configure two Web Access methods: one for the Administration Manager and one for end user Web Access. You must configure the Administration Manager before any further product setup can take place. See the manual *Configuring Workflows with the Core Access Assurance Suite Administration Manager* for more information.

Note: When you configure Web Access for either AccountCourier or ComplianceCourier, the Core Access Assurance Suite enables Web Access for the other product if the appropriate access keys exist for the product. For example, if you configure Web Access for AccountCourier, and you have an access key for ComplianceCourier, the system enables web access for ComplianceCourier.

Configuration

You can configure products on the Provisioning Platform after you have installed the Core Access Assurance Suite and configured the Courier Server. Web Access configuration involves using two separate configuration utilities. See *Configuring Workflows with the Core Access Assurance Suite Administration Manager* for configuring the Administration Manager and [“Web Access Configuration” on page 17](#) for configuring end user access. The Administration Manager must be configured before it can be used to configure the product.

To prevent confusing error messages that could potentially contain confidential information in clear text and to allow AccountCourier to work with IIS version 4.0, IIS server side script debugging should be turned off and **SENT TEXT ERROR MESSAGE TO CLIENT** should be turned on.

Core Access Assurance Suite Administration Manager Configuration

The settings in the Core Access Assurance Suite Administration Manager are used to customize the runtime behavior of the Administration Manager. To access the Administration Manager, follow the path below:

Start>All Programs>Core Access Assurance Suite>Administration Manager Configuration

Configure the Administration Manager as appropriate for your environment. For information on how to configure these settings, please refer to the [“Web Access Configuration Settings” on page 11](#).

Note: The Administration Manager contains one additional configuration parameter to those described in Table 1 on page 11, **SESSIONTIMEOUTMESSAGE**. The **SESSIONTIMEOUTMESSAGE** parameter is used to specify the message that is displayed if a CourATL timeout occurs while someone is using AccountCourier. The session timeout message can contain a brief text message up to 64 characters long. This parameter’s default value is “ASP Session timed out.”

Web Access Configuration

The settings in the Web Access Configuration manager are used to customize the runtime behavior of products on the Provisioning Platform. To access the Web Access Configuration Manager, follow the path below:

Start>All Programs>Core Access Assurance Suite>Web Access Configuration

Configure Web Access as appropriate for your environment. For information on how to configure these settings, please refer to the [*"Web Access Configuration Settings" on page 11.*](#)

Using Client Side Components

Client side components are available for use with the PMMs for Active Directory, Lotus Notes, and Windows NT/2000. For details see [*"Client Side Components" on page 19.*](#)

Client Side Components

There are two client side components used for web access when using certain PMMs: CourLocal control and Courion Client Manager

Courion's ActiveX control, CourLocal, is available with the PMMs for Active Directory, Lotus Notes, and Windows NT/2000. CourLocal can perform two types of actions, depending on the target being reset.

- For Active Directory and Windows NT/2000 password resets, the control updates the locally cached password. This feature prevents an end-user from inadvertently getting locked out of his or her domain account after using PasswordCourier's Web Access Option to reset the domain password.
Note: The end user client system must be on the same network with domain for the client-side component to complete the work of updating the locally cached copy of the password.
- For Lotus Notes password resets, the control downloads the modified Lotus Notes ID file to the local machine. This control can replace the ID file that has the forgotten or expired password with the ID file that has been updated with the user-selected password.

In addition to CourLocal, there is an optionally deployable Windows Service that resides on the end user's client system. A service is needed to make a Microsoft API call that requires "Act as part of operating system" to update the locally stored password that is actively in use in the logged on session. These client side components are called the Courion Client Manager.

Core Security recommends using this optional Service component when NTLM authentication is used in desktop applications in combination with a domain password policy that does include using password history. This Service is not needed for end users that have "Act as part of operating system." Also this Service is not needed if end users lock and unlock their session or log out and re-log-in before the previous password times out in the current session. The time associated with that session varies with the use of domain password history with NTLM or the configured time for a Kerberos ticket.

Client Side Configuration

There are two methods for deploying on client machines:

- Install Courion Client Manager manually
- Install Courion Client Manager via silent installation

Courion Client Manager Installation

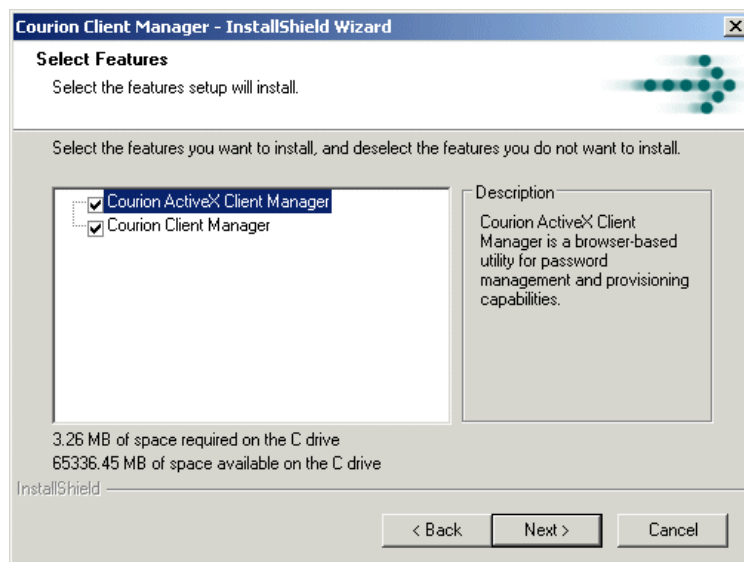
The Courion Client Manager installation allows configurable selection of certificate validation. This can be useful if your environment does not allow for the full default validation. By using this option, you can control how much or how little validation is used. Core Security recommends using the default value to perform all validation checks online.

Download the Courion Client Manager installation kit from the Core Security web site. For assistance, contact Core Security Customer Support.

After launching the installation wizard and accepting the license agreement, the Setup Type screen appears, where you can select a Complete or Custom installation. If you select **COMPLETE** and click **NEXT**, the Certificate Validation screen appears, as shown in [Figure 3](#).

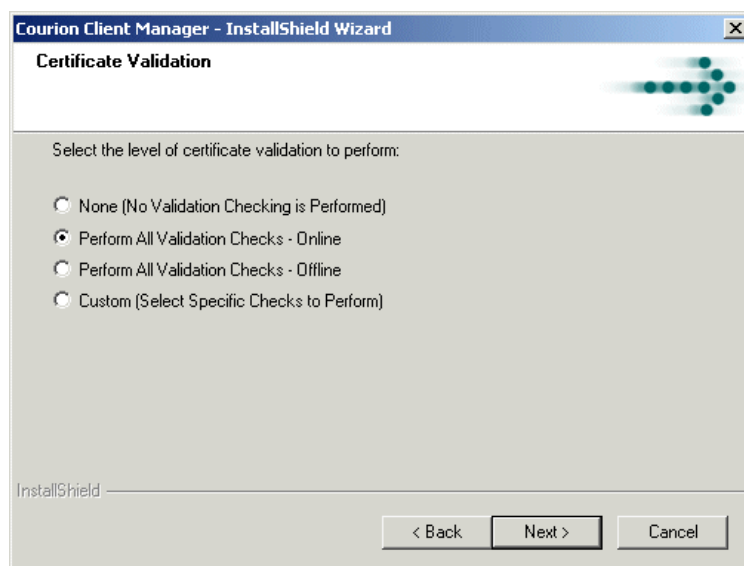
If you select **CUSTOM** and click **NEXT**, a screen appears where you can choose the location for the Courion Client Manager files. Click **NEXT** again and the following screen appears:

Figure 2: Courion Client Manager - Select Features



You can choose to install either the CourLocal ActiveX Client Manager, the Courion Client Manager, or both. If you choose only the CourLocal ActiveX Client Manager and click **NEXT**, the installation wizard proceeds to the final installation confirmation screen. If you choose the Courion Client Manager and click **NEXT**, the Certificate Validation screen appears, as shown in [Figure 3](#).

Figure 3: Courion Client Manager Certificate Validation



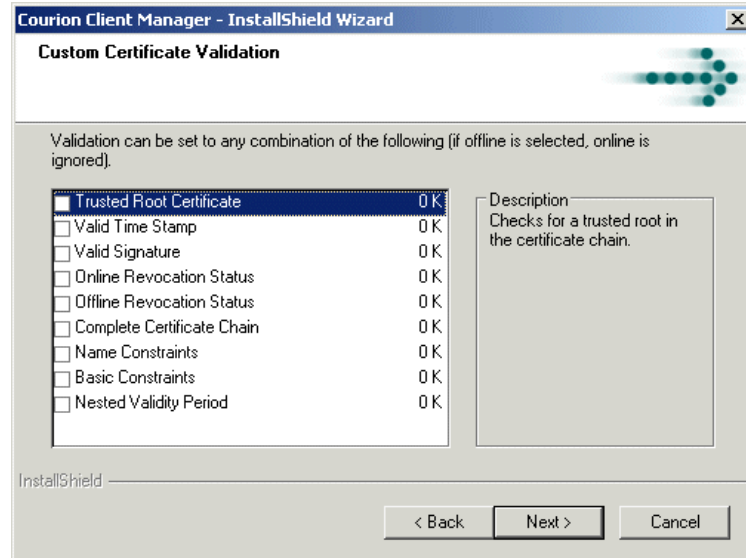
The default value is **PERFORM ALL VALIDATION CHECKS - ONLINE**. This option checks all conditions except offline revocation status. Revocation checks are performed on all certificates in the chain except for the root certificate.

The **PERFORM ALL VALIDATION CHECKS - OFFLINE** option checks all conditions except online revocation status. Revocation checks are performed on all certificates in the chain except for the root certificate.

The **PERFORM ONLINE VALIDATION CHECKS - NO TIME STAMPS** checks all conditions except valid time stamp.

If you select **CUSTOM (SELECT SPECIFIC CHECKS TO PERFORM)** and click **NEXT**, the following screen appears:

Figure 4: Courion Client Manager Custom Certificate Validation



Selecting a validation item displays a description of that item. Select the desired validation items and click **NEXT** to continue with the installation. You can select the following validation items:

- **Trusted Root Certificate** — Checks for a trusted root in the certificate chain
- **Valid Time Stamp** — Checks the ValidFrom and ValidTo dates of the issued certificate against the system time
- **Valid Signature** — Checks for valid signatures on all certificates in the chain
- **Online Revocation Status** — Checks the revocation status of all certificates in the chain using certificate revocation lists (CRLs) available online
- **Offline Revocation Status** — Checks the revocation status of all certificates in the chain using only offline CRLs
- **Complete Chain Certificate** — Checks all certificates in the chain.
- **Name Constraints** — Checks the name length constraints (if any). The certificate authority may limit the length of the name of the certificate
- **Basic Constraints** — Checks basic constraints. Basic constraints can be viewed in the certificate properties. For instance, the Courion certificate has the following basic constraints: Subject Type=end Entity, PATH Length Constraint=None
- **Nested Validity Period** — Checks ValidFrom and ValidTo dates for all certificates in the chain

Using Silent Installations

Silent Installations install the Courion Client Manager on a target PC without any end-user interaction. To run an installation in silent mode, you need to create a silent install response file. This file supplies responses to the various configuration options available during any standard installation.

The following steps show an example of how to create a basic silent install response file and run a silent installation. See [“Creating Tailored Silent Install Response Files” on page 23](#) for detailed information about creating customized response files for large deployments with many different PC types. See [“Configuration Switches” on page 24](#) for detailed information about the switches used in these examples.

1. Create a silent install response file. To do this, run an installation in record mode on a target PC using the CourionClientManager.exe command with the -r command line switch and the name of the silent response file. In this example, the name of the file is Install.iss. You can use the -f1 command line switch to specify the silent install response file:

```
CourionClientManager.exe -r -f1D:\Install.iss
```

Executing the command launches the InstallShield Wizard. Enter the information into each configuration page of the InstallShield Wizard.

CourionClientManager.exe places the responses entered into the silent install response file with the name you specified.

2. Run the installation in silent mode by running it on another Target PC using the -s command line switch. You can use the -f1 command line switch to specify the silent install response file and the -f2 command line switch to specify the name and location for the log file.

```
CourionClientManager.exe -s -f1D:\Install.iss -  
f2%COMPUTERNAME%.log
```

3. Reboot the PC. If the silent install response file specifies to reboot the PC automatically after the installation is complete, the PC reboots without notifying the end user.

Silent Installation Log Files

When you run a silent installation, Courion Client Manager creates a log file with software version information and response codes that indicate whether the installation succeeded or failed. Setup.log is the default name for the silent setup log file, and its default location is same directory as the installation executable). You can specify a different name and location for Setup.log using the -f1 and -f2 switches with CourionClientManager.exe (see [“Configuration Switches” on page 24](#)).

The Setup.log file contains three sections:

- **InstallShield Silent:** this section identifies the version of InstallShield Silent used in the silent setup. It also identifies the file as a log file.
- **Application:** this section identifies the installed application's name and version, and the company name.
- **ResponseResult:** contains the result code indicating whether the silent setup succeeded. An integer value is assigned to the ResultCode keyname in the [ResponseResult] section. InstallShield places one of the following return values after the ResultCode keyname:

- 0 Success.
- 1 General error.
- 2 Invalid mode.
- 3 Required data not found in the Setup.iss file.
- 4 Not enough memory available.
- 5 File does not exist
- 6 Cannot write to the response file.
- 7 Unable to write to the log file.
- 8 Invalid path to the InstallShield Silent response file.
- 9 Not a valid list type (string or number).
- 10 Data type is invalid.
- 11 Unknown error during setup.
- 12 Dialog boxes are out of order.
- 51 Cannot create the specified folder.
- 52 Cannot access the specified file or folder.
- 53 Invalid option selected.

Creating Tailored Silent Install Response Files

In large deployments with many PCs in different locations, you may require multiple silent install response files tailored to different types of installations. You may also require individual log files for each installation. The factors that determine the number of response files you need are:

- Location of installed files
- Selection of installed features: Complete or Custom
- Type of installation: New, Upgrade, or Maintenance

Run the installation in record mode for each type of silent installation you want to create. The `-f1` command line switch overrides the default name for the .iss file. Once you create these response files, copy them to the installation executable directory for use in deploying the various silent installations.

Examples of Silent Configuration Commands

```
CourionClientManager.exe -r -f1D:\CompleteInstall.iss
```

```
CourionClientManager.exe -r -f1D:\Upgrade.iss
```

```
CourionClientManager.exe -r -f1D:\Repair.iss
```

If you do not specify a path, the .iss file is created in the windows directory of the target PC. If you specify a path, the path must exist. UNC path names are allowed. Enclose any path that contains spaces with quotation marks. No spaces are allowed between `-f1` and the specified path.

```
CourionClientManager.exe -r -f1\\server\sharename\Install.iss
```

```
CourionClientManager.exe -r -f1" \\server\sharename with  
spaces\Install.iss"
```

Once you copy the silent install response files to the installation files folder (location of CourionClientManager.exe), launch the setup from the target machine with the `-s` switch. Specify the name of the appropriate .iss file. If you do not use the `-f1` switch, then the install attempts to use the default file name setup.iss. When a silent install runs, it creates a log file named Setup.log in the installation files folder. This file logs the result (success or error) of the silent install.

You can create individual log files for each of the client machines because each install overwrites the default Setup.log file. Use the `-f2` command line switch to override the name and location of the default silent install log files. If you do not specify a path, the installation files folder is used. To create individual log files, either specify an individual log file name for each install or use a unique environment variable enclosed in % signs.

Command Line Example:

```
CourionClientManager.exe -s -f1D:\Install.iss -f2%COMPUTERNAME%.log
```

If you do not specify a path, the .log file is created in the installation files folder of the Source PC. If you specify a path, the path must exist. UNC path names are allowed. Enclose any path that contains spaces in quotation marks. No spaces are allowed between `-f2` and the specified path. The examples below need to be entered on a single command line.

```
CourionClientManager.exe -s -f1\\server\sharename\Install.iss -f2\\server\sharename\%COMPUTERNAME%.log
```

```
CourionClientManager.exe -s -f1\\server\sharename\Install.iss -f2"\\server\sharename with spaces\%COMPUTERNAME%.log"
```

Configuration Switches

These switches provide different configuration options when you create silent response files with the CourionClientManager.exe command. They are not case sensitive. Separate multiple command line switches with a space, but do not put a space between the switch and the parameter associated with it.

Examples:

Valid: `/r /fInstall.inx`

Not valid: `/r/f Install.inx`

When using paths and filename expressions that include spaces, enclose the expressions in double quotation marks. The double quotes tell the operating system that spaces within the quotation marks are not to be treated as command line delimiters.

`/f1<path\ResponseFile>` or `-f1<path\ResponseFile>`

The **"f1"** switch specifies an alternate location and name for the response file (.iss file). If you use this option with InstallShield Silent, the response file is read from the folder/file specified by `<path\ResponseFile>`. If you use this option with the `-r` option, the response file is written to the folder/file specified by `<path\ResponseFile>`. If you specify an alternate compiled script with the `-f` switch, the `-f1` switch entry must follow the `-f` switch entry.

`/f2<path\LogFile>` or `-f2<path\LogFile>`

The **"f2"** switch specifies an alternate location and name for the log file created by InstallShield Silent. By default, InstallShield Silent creates and stores the Setup.log log file in the same directory as that of Setup.inx. If you specify an alternate compiled script with the -f switch, the -f2 switch entry must follow the -f switch entry.

/m<filename> or -m<filename>

The **"m"** switch causes CourionClientManager.exe to generate a Management Information Format (.mif) file automatically at the end of the setup. Do not include a path or an extension. CourionClientManager.exe places the .mif file in the Windows folder. <filename> is optional. If you do not specify a filename, the resulting file is called Status.mif.

/m1<serial number> or -m1<serial number>

The **"m1"** switch causes CourionClientManager.exe to place the serial number you specify in the .mif file.

/m2<locale string> or -m2<locale string>

The **"m2"** switch causes CourionClientManager.exe to place the locale you specify in the .mif file. English (ENU) is the default; refer to Microsoft documentation for a complete listing of locale strings.

/r or -r

The **"r"** switch causes CourionClientManager.exe to generate a silent setup file (.iss file) and place it in the Windows folder. This file is a record of the setup input, in the Windows folder.

/s or -s

The **"s"** switch causes InstallShield Silent to execute a silent setup.

/verbose or -verbose

The **"v"** switch provides detailed information when a CourionClientManager.exe error occurs.

Server Side Configuration for the Provisioning Platform

Complete the following steps on the server side to use CourLocal control and the Courion Client Manager on the Provisioning Platform:

1. Modify and submit the Unique Resource Data Form, if you are upgrading from a previous version
2. Modify the JavaScript Template
3. Format the output from the CourLocalControl ActiveX control's processing
4. Update the Password Reset action's Summary page
5. Configure the Transaction Repository

The following sections describe these steps.

Modifying and Submitting the Unique Resource Data Form When Upgrading

If you have targets for the PMMs for Active Directory, Windows NT, or Lotus Notes that were originally configured in a version of the Core Access Assurance Suite prior to 7.50 update 6, you need to modify the Unique Resource Data Form, and click **SUBMIT**. This updates the form to include the Reset Module Properties attribute, shown in [Figure 5](#).

Figure 5: Unique Resource Data Form with Reset Module Properties

The screenshot shows a web form with a table-like structure. On the left, there is a dropdown menu labeled 'Override'. In the center, the text 'Reset Module Properties' is displayed. On the right, there is a list of checkboxes: 'Global', 'Required', 'Secure', 'Verify', 'Visible', 'Read Only', and 'Global'. Below these checkboxes, the text 'Reset Module P' is visible.

Updating the Password Reset Action's Summary Page

After the password reset completes and the Summary page is displayed to the end user, the CourLocalControl ActiveX control is called to take the necessary actions on the end user's desktop client system. The Form Details of the Summary page must be configured to provide the details to activate the CourLocalControl ActiveX control.

Checking the "Do Not Show Summary Until Action Completes" configuration option on the Summary form ensures that the summary page does not display until the password reset action completes.

Modifying the JavaScript Template

The following JavaScript template provides a sample of what is needed to activate the ActiveX control. The items in the template that should be changed are:

- The names of the macros which include the specific target name as it exists in your workflow

```
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourADDomain.Account Password.Reset Module
Properties%
```

```
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourWNTDomain.Account Password.Reset Module
Properties%
```

```
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourLotusNotesDomain.Account Password.Reset
Module Properties%
```

- The parameters that control the functionality of the CourLocalControl ActiveX Control.

```
reqObj.PARAM_KEY_LOCALCACHEDISABLEUPDATE
```

```
reqObj.PARAM_KEY_LOTUSNOTESDISABLEDOWNLOAD
```

```
reqObj.PARAM_KEY_CACHEUPDATEALLACCOUNTS
```

Modify the following JavaScript template to fit the need for the password reset workflow. This template reflects that updating the locally cached passwords feature will occur, and it will occur for any account in the local cache (not just the currently logged in user), and the Lotus Notes ID file will be downloaded.

JavaScript Template

```
<script language="JavaScript" src="Common/
CourLocalHelper.js"></script>
<script language="JavaScript">var cacheTokenArray = new
Array();
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourADDomain.Account Password.Reset Module
Properties%
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourWNTDomain.Account Password.Reset Module
Properties%
%[COURLOCALJS].Password Reset.Resource Overrides.PMM-
Gateway-Cnctr.YourLotusNotesDomain.Account Password.Reset
Module Properties%
</script>
<div id="CacheResult"></div>
<object id="CourLocal2" classid="clsid:6C64B50D-0472-
4CD6-9312-644BEF37D4E6"
codebase="CourLocal35.CAB#version=8,0,0,35"></object>
<object id="Params" classid="clsid:61CB728E-6267-4494-
B197-8AB826C577A1"
codebase="CourLocal35.CAB#version=8,0,0,35"></object>
<script language="JavaScript">var result = ""; var
reqObj=new Object();
reqObj.AccountArray=cacheTokenArray;reqObj.Params=document.
getElementById("Params");
reqObj.CourLocalObj =
document.getElementById("CourLocal2");
reqObj.PARAM_KEY_LOCALCACHEDISABLEUPDATE = false;
reqObj.PARAM_KEY_LOTUSNOTESDISABLEDOWNLOAD = false;
reqObj.PARAM_KEY_CACHEUPDATEALLACCOUNTS = true;
result = DoControl(reqObj);
var resultTag = document.getElementById('CacheResult');
resultTag.innerHTML = result; </script>
```

Warning: Do not include carriage returns in the JavaScript code inside any of the tags. Carriage returns are encoded as HTML breaks (
). This causes the JavaScript code to be invalid. When you compose the code in the text editor, you can use carriage returns, but you must remove them before pasting the code into the Summary Form Details box.

Warning: Do not include curly quotes in the JavaScript code; use only straight quotes (" "). Use a text editor that does not allow curly quotes, such as Microsoft Notepad, to compose the code.

Warning: When a newer version of the CourLocalControl ActiveX control is released in the future, you will need to update the version number information mentioned above.

The inputs from the reset modules are in the macros that start with %[COURLOCALJS].Password Reset. Insert a macro for each target in the workflow that supports this capability. Use COURLOCALJS transformation on it to properly format the data.

Use the macro Password Reset.Resource Overrides.PMM-Gateway-Cnctr.TARGET.Account Password.Reset Module Properties:

Figure 6: CourLocal Password Reset Macro

%Password Reset.Resource Overrides.PMM-Gateway-Cnctr.bowie.Account Password.Password%
%Password Reset.Resource Overrides.PMM-Gateway-Cnctr.bowie.Account Password.Reset Module Properties%
%Password Reset.Resource Overrides.PMM-Gateway-Cnctr.bowie.Account Password.User Must Change Password

Use the transformation COURLOCALJS as the **DATA FORMAT TO APPLY**.

Insert these in the Template after the line: var cacheTokenArray = new Array();

Note: Use the CourLocalJS transformation to take the value of the ResetModuleProperties macro and put it into the data format needed for the CourLocalControl ActiveX control. An example of how to use this new transformation is provided in the JavaScript code template provided here.

In the JavaScript code template the current values for the object ID tags are correct. If there is an update to CourClientManager, you need to get the version information of CourLocalControl.DLL that you are going to use. The default values in the Template should be fine.

The options available on the CourLocalControl remain the same as they were for the use of this control on the classic platform. To set the client-side processing options, insert the appropriate code after the following:

```
reqObj.CourLocalObj = document.getElementById("CourLocal2");
```

- **Disable Local Cache Update** - This option prevents the client-side control from updating the local password cache for the packets it is processing. To activate this option, use this code:

```
reqObj.PARAM_KEY_LOCALCACHEDISABLEUPDATE = true;
```
- **Update All Cached Accounts** - By default, the control only updates the cache for the logged-in user. If you run this workflow in a kiosk or in a situation where the logged-in user does not match the reset user, activate this option with this code:

```
reqObj.PARAM_KEY_CACHEUPDATEALLACCOUNTS = true;
```
- **Disable Lotus Notes ID Download** - This option prevents the control from downloading Lotus Notes ID files when it is set to true. By default, the control downloads any Lotus Notes ID files it is presented. To activate this option, use this code:

```
reqObj.PARAM_KEY_LOTUSNOTESDISABLEDOWNLOAD = true;
```

Formatting the output from the CourLocalControl ActiveX control's processing

Decide where you want the output of the client-side processing. Place the code

```
<div id="CacheResult"></div> somewhere in the Form Details.
```

To hide the output, use the code `<div id="CacheResult" style="display:none"></div>`

By default the output is generated as an HTML concatenated string. The string is pushed into the innerHTML of the <div> tag in the code snippet (id=CacheResult). If you want to process the output in a different way, you can completely customize the output with additional JavaScript.

Follow these steps:

1. Copy the original CourLocalHelper.js to another file (i.e. myCourLocalHelper.js)

(If AAS was installed in the default location, this file can be found in C:\Program Files (x86)\Courion Corporation\www\WebSamples\AccessOptions\HTML\AccountCourier\common)

2. Open the new file, myCourLocalHelper.js, and edit near the end of the DoControl() function to generate the output you want to return. Look for places that modify the value of the status_msg variable. Example:

```
status_msg += "<p>The control returned code:
"+rCode.text+".</p>";
```

3. Change your code snippet in the Summary page Form Details box to reference the new file:

```
<script language="JavaScript" src="Common/
CourLocalHelper.js"></script>
to
<script language="JavaScript" src="Common/
myCourLocalHelper.js"></script>
```

Server Side Configuration for the Classic Platform

If CourLocal is implemented, workflow users should use IE8, IE9, and IE10. Mozilla Firefox and Google Chrome may not support some Microsoft technologies such as ActiveX.

This release supports version 8.00.00.35 of CourLocalControl.dll. The name of the CourLocal34.CAB file has been changed to CourLocal35.CAB

The PasswordCourier Classic platform uses a variable codebase="CourLocal35.CAB#version=<%=COUR_VERSION%>" COUR_VERSION is defined in CourVersion.asp. The value for COUR_VERSION is updated by CourVersion.asp as part of Access Assurance Suite installation.

To use CourLocal control on the Classic platform, configure it using one of the following options:

- Modify www\WebSamples\AccessOptions\HTML\PasswordCourier\Password.asp to point to Reset3.asp instead of Reset.asp.

Open the Password.asp file with a text editor and change all instances of "Reset.asp" to "Reset3.asp", then resave Password.asp

OR

- Delete the file
www\WebSamples\AccessOptions\HTML\PasswordCourier\Reset.asp
then rename the file
www\WebSamples\AccessOptions\HTML\PasswordCourier\Reset3.asp
to
www\WebSamples\AccessOptions\HTML\PasswordCourier\Reset.asp

Note: Instead of deleting the original Reset.asp file, you may want to give it a different name to have a copy of the original file.

Chapter 3: Guidelines for Modifying Sample Active Server Pages

This chapter provides some guidelines for modifying sample active server pages and includes the following sections:

- [*"Introduction" on page 32*](#)
- [*"General ASP Information" on page 33*](#)
- [*"Back Up of Default and Customized ASPs" on page 35*](#)
- [*"Guidelines for Modifying ASPs" on page 36*](#)
- [*"COM Interfaces for the Classic Platform" on page 40*](#)

Introduction

This chapter is a collection of information and guidelines for modifying the Active Server Pages (ASPs) shipped and used by the Core Access Assurance Suite. The ASPs shipped with the Core Access Assurance Suite work as-is and require no customization to perform their intended tasks. Though you can customize some of the HTML and text to modify the look and feel of these pages, do not modify the functional code elements within the ASPs.

This chapter does not detail the mechanics of how to alter the ASPs; it assumes you have the knowledge and skills to make these alterations.

Internet Information Server (IIS) is configured to return a detailed ASP error message to the browser. To change this on the server where CourATLService is running:

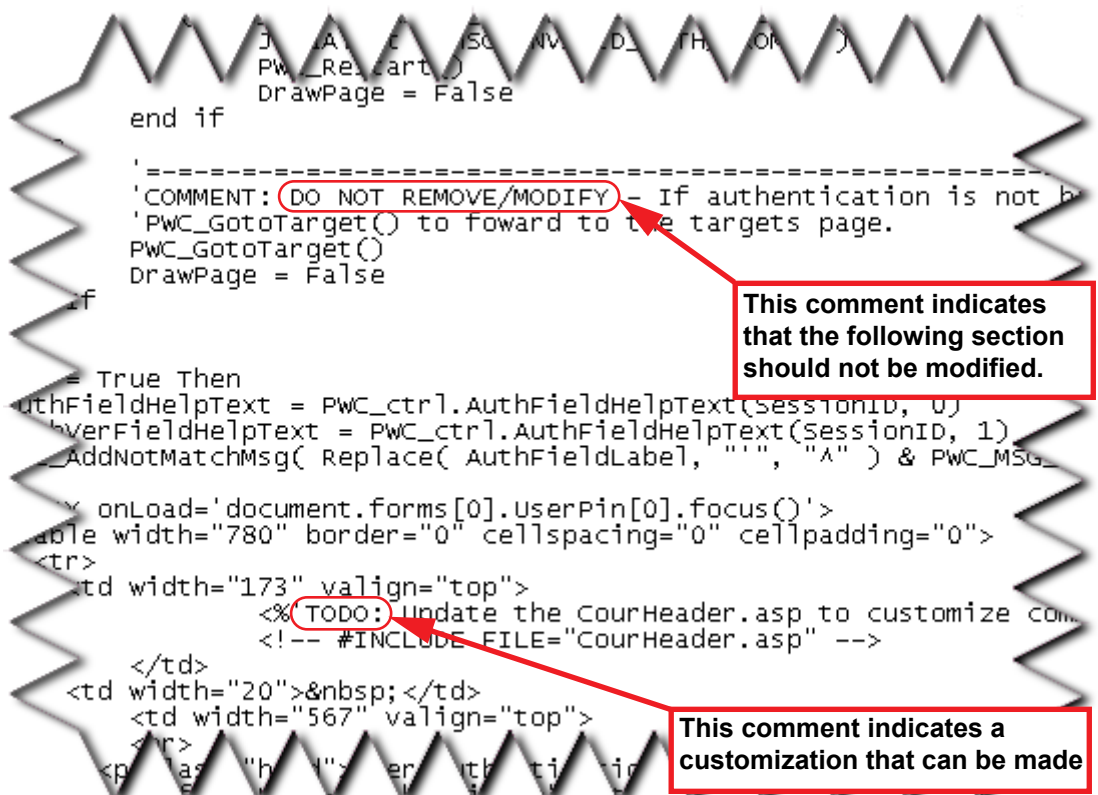
1. Go to the Control Panel and double-click on the Administrative Tools icon.
2. Double-click the **INTERNET INFORMATION SERVICE MANAGER** icon in the window that comes up.
3. In the tree in the left pane, expand the local computer, then expand the **WEB SITES** folder.
4. Click on **DEFAULT WEB SITE** and then right-click on the Virtual Directory where the Core Security ASP files are stored.
5. Select **PROPERTIES** in the window that opens and click on the **CONFIGURATION** box. This will open the App Window.
6. Select the **APP DEBUGGING** tab and select the **SEND TEXT ERROR MESSAGE TO CLIENT**. In the box below, write the error message you wish users to receive.

General ASP Information

Each ASP file contains comments which indicate what can be customized (“TODO”) and what should not be changed (“DO NOT REMOVE/MODIFY”). [Figure 7](#) shows an example of what can and what should not be modified. If you change required code, user actions such as password resets will not properly function. The ASPs already handle the various configurations possible for the Core Access Assurance Suite. If, for example, escrow password is used, the code within the ASP receives the information and does not prompt the end user for the password and verify password fields.

Only change the code marked with the TODO: tag. You can add extra HTML and graphics around the pertinent Core Security information. Be sure to test all created code thoroughly before publishing on the corporate web site.

Figure 7: Sample ASP Comments



The first ASP that executes makes a connection with the Core Security Server. Once the connection is established, a session ID is returned. Web Access options for the Classic Platform use the session ID to keep track of the steps through the workflow. This means that if authentication is configured, an end user is not able to bypass the authentication step and move into another step and create a security risk.

Note: If the **BACK** button on the browser is used when there is no **PREVIOUS** button presented on the HTML page, it is possible to get into the wrong state of the workflow. It is best to ask end users to use the **PREVIOUS** and **NEXT** buttons provided in the HTML page rather than their browsers' navigation buttons.

AccountCourier, PasswordCourier Classic, ProfileCourier Classic, and CertificateCourier also ship with a Resource ASP file (CourACStrings.asp, PWC_Resource.asp, PC_Resource.asp, and CC_Resource.asp, respectively) that contains the default text for messages, buttons, page titles, and

help text used in the end user interface. You can modify the text within the quotation marks in these files as needed, but you should not modify any text not in quotations. Also, do not remove the spaces within the quotation marks, as they are used for spacing purposes in the text string.

Back Up of Default and Customized ASPs

During installation, all Core Access Assurance Suite web pages are saved to

Program Files>Courion Corporation>www_DoNotModify-X.XX.XXX (where X.XX.XXX is the version number)

When installation is complete, the files are copied to

Program Files>Courion Corporation>www

You should place your customized files in the www directory. If any ASP functionality is lost or damaged during customization, replace the damaged files with the default files contained within www_DoNotModify directory.

After you customize and test ASP files, make a copy of the www folder containing the customized ASPs and save that copy to another location on your server for safe keeping. You can use this copy of the customized files to refresh any working files if they become corrupt due to further customization. They can also be useful as reference files when customizing the ASPs in later Core Access Assurance Suite installations.

If you upgrade from a previous release, the installer creates a backup copy of the current www directory before installing the www_DoNotModify-X.XX.XXX directory and overwriting the current www directory. The name of the backup folder is based upon the version of the suite installed when the upgrade install is started. For example, if you upgrade from 7.70 update 02 to 7.80, the backup directory is "www-7.70.002-Backup".

Note: The ASPs shipped with the Core Access Assurance Suite are version-specific and you cannot use them with previous or future versions of the Core Access Assurance Suite. Use the backup customized files as references when customizing the newer ASPs. Do not copy or paste code from an older ASP to a newer ASP (or vice versa), because the functionality within the ASP will fail.

File Paths in ASP Files

Once the ASP files are copied to the www directory, any ASP pages containing "#INCLUDE FILE=" statements with relative paths are modified to point to the virtual directory "Core" that is created during installation. This change is required because the default configuration of IIS 6.0 virtual directories does not allow parent paths. The installer logs all of these file changes.

For example, the statement

```
<!-- #INCLUDE FILE="../../../WebSamples/AccessOptions/HTML/AccountCourier/
Common/CourACUtils.asp" -->
```

is changed to

```
<!-- #INCLUDE VIRTUAL="Core/WebSamples/AccessOptions/HTML/AccountCourier/
Common/CourACUtils.asp" -->
```

These changes are made only to the files in the www directory, not the www_DoNotModify directory. Therefore, if you restore an ASP file by copying it from the www_DoNotModify directory to the www directory, you will need to change any #INCLUDE FILE= statements to #INCLUDE VIRTUAL= statements. Or, you can change the #INCLUDE FILE= statements from relative to absolute file paths.

Guidelines for Modifying ASPs

This section describes ASP, CSS (Cascading Style Sheets), and JavaScript™ files that you can modify.

Customizable ASPs

The customizable ASPs for each Core product are listed below. These files are located in the product folders under:

Program Files (x86)>courion>www>WebSamples>AccessOptions>HTML

Note that although these files are labeled as “customizable”, there are elements within each of these files that should not be altered in any way. Please see [“General ASP Information” on page 33](#) for more information on how to identify what is and is not customizable within an ASP file. Many of these files are very complexly constructed. Please use caution when customizing files.

PasswordCourier Classic and PasswordCourier Support Staff Classic

The following files can be customized:

- Authenticate.asp
- CourFooter.asp
- CourHeader.asp
- CourUtils.asp
- Default.asp
- DisableApp.asp
- Password.asp
- PWC_Resource.asp
- Reset.asp and Reset3.asp
- Targets.asp
- Validate.asp
- ValidateStaff.asp (PasswordCourier Support Staff only)

ProfileCourier Classic

The following files can be customized:

- CourPBUtils.asp
- Default.asp
- PC_Resource.asp
- PCFooter.asp

- PCHeader.asp
- PCLogin.asp
- PCUserInfo.asp
- UpdateProfile.asp

The Provisioning Platform

Please refer to the manual *Configuring Workflows with the Core Access Assurance Suite Administration Manager* for information about ASP file customization for the provisioning platform.

Details on Specific ASPs

Most customization can be handled by modifying the Header and Footer ASPs for each product. These ASPs contain logos and default text that display on every page the users view, as the Header and Footer ASPs are referenced by all other ASP files for general formatting and layout purposes. The figure below illustrates how the Header and Footer ASPs are used in the end user interface.

Figure 8: ASP Layout Illustration



The provided Reset.asp and Reset3.asp files include a mechanism that provides a way to redirect end users to another URL after they successfully complete the password reset process. You can edit ASPs to customize a name for the button to display and the URL to which it redirects end users.

Helper Routines

The CourUtils.asp file contains helper routines, some of which every ASP should call:

- `pwc_CheckKeyStrength()` checks the key length of the browser against the required key length (configured).
- `pwc_CheckAndInit()` checks the product ID and client IP address to make sure the provided information is valid. This routine also generates a session ID and, therefore, should only be called on the first ASP page that is part of the password reset process.
- `pwc_CheckClient` checks whether the product ID, client IP address, and session ID are valid. The reset process should be terminated if any of the information is not valid. If it is not valid, check if the request came from the same source for the next page, other than the source for the first page.

- `pwc_CourDebug` prints out session information for debugging purposes if debugging is configured. This is useful when testing created ASPs before deployment.

Other helper routines that can be found in `CourUtils.asp` are:

- `pwc_GotoTarget()` is used to forward the state to the target state when there is no authentication configured.
- `pwc_GoBackAndAlert (reason)` goes back to the previous page and displays the supplied reason.
- `pwc_Restart` goes back to the home page, which defaults to `Courion.asp`.

The `CourPBUtls.asp` file contains helper routines, some of which every ASP should call:

- `CheckKeyStrength()` checks the key length of the browser against the required key length (configured).
- `CheckAndInit()` checks the product ID and client IP address to make sure that the provided information is valid. This routine also generates a session ID and, therefore, should only be called on the first ASP that is part of the password reset process.
- `CheckClient` checks if the product ID, client IP address, and session ID are valid. If any of the information is found not to be valid the reset process should be terminated.
- `CourDebug` prints out session information for debugging purposes if debugging is configured. This capability is useful to have when testing ASPs before deployment.

Other helper routines that can be found in `CourPBUtls.asp` are:

- `GoBackAndAlert (reason)` goes back to the previous page and displays the supplied reason.
- `Restart` links back to the home page, which defaults to `Default.asp`.
- `onSubmit(fieldName, checkMatch)` checks if fields are equal if `checkMatch=1`, otherwise it checks if fields are empty.

Cascading Style Sheet Customization

A cascading style sheet (CSS) is shipped with all Core products. This CSS defines the basic text formatting in the end-user interface. It is located in:

Program Files (x86)>Courion Corporation>www>WebSamples>AccessOptions>AccessOptions_styles.css

You can modify the style definitions in this CSS file as needed.

JavaScript Customization

There is also a JavaScript file shipped with the product that contains common client side routines. This file is called `CourUtils.js`. It includes the following helper routines:

- `onSubmit (fieldName, checkMatch)` checks if fields are equal if `checkMatch = 1`, otherwise it checks to see if fields are empty.
- `onSubmitSelectTarget ()` checks to see if a target is selected.
- `onSubmitCheckPassword ()` checks to see if the entered password meets the established length and alphanumeric requirements.
- `onUpdateGroup (selectIndex)` updates the target option list and the group help text when the end user selects a different group.
- `onUpdateTarget (selectIndex)` updates the target help text when the end user selects a different target.

`CourPBUtils.js` contains similar common client side routines.

COM Interfaces for the Classic Platform

Aside from the helper routines, Core Security provides the Component Object Model (COM) interfaces that are necessary to go through each step of a Classic Platform workflow.

ValidateUser COM Interface Properties

To use the validation COM interfaces, the following must be in the code:

- `CreateObject("CourATLService.ValidateUser")` Assign it to a local object variable. Then, using this local parameter, which represents the `ValidateUser` object, use the properties listed below:
- `FieldCount (sessionID)` returns a short integer that is the number of configured user validation fields.
- `FieldLabel (sessionID, index)` returns a BSTR that is the configured label for the field specified by the index. Valid indexes are 0 to `FieldCount-1`.
- `FieldHelpText (sessionID, index)` returns a BSTR that is the configured help text for the field specified by the index. Valid indexes are 0 to `FieldCount-1`.
- `IsSecureField (sessionID, index)` returns a boolean to indicate whether the field specified by the index is secured or hashed so it can be appropriately displayed. `True` indicates a hashed or secure field and `False` indicates a nonsecure field. Valid indexes are 0 to `Fieldcount-1`.
- `IsNumericField (sessionID, index)` returns a boolean to indicate whether the field specified by the index requires numeric entry. `True` indicates that it requires numeric entry and `False` indicates that numeric and/or text is required. Valid indexes are 0 to `Fieldcount-1`.

ValidateUser2 Interface Properties

A COM interface cannot be changed in any way after it is published, including the addition of properties or methods. When an interface needs to be changed, it is common to have another interface with the same name and an ordinal number indicating that it is the next in the series. In this case, a new property is available. For this reason, a `ValidateUser2` interface has been published with the new property. To use this COM interface, include `CreateObject("CourATLService.ValidateUser2")` in the code and assign it to a local parameter. Using this local parameter, which represents the `ValidateUser2` object, you can use the following property:

- `PBLoginDirections (sessionID)` returns a BSTR that is the configured login directions.

AuthenticateUser COM Interface Properties

To use the authentication COM interfaces, include `CreateObject("CourATLService.AuthenticateUser")` in the code and assign it to a local parameter. With this local parameter, which represents the `AuthenticateUser` object, use the following properties:

- `ValidateUser (sessionID, UserInfo, AccountDisable)` returns a short integer to indicate the success or failure of the end user information validation. `AccountDisable` is `True` if the maximum number of attempts is reached and the end user account is disabled from using `PasswordCourier`.
- `UsePin (sessionID)` returns a boolean to indicate whether or not authentication is set. `True` indicates that end user authentication is required and `False` indicates that no authentication is required.
- `AuthFieldLabel (sessionID, index)` returns a BSTR that is the configured authentication label when the index is 0. When the index is 1, the configured authentication verification label is returned.
- `AuthFieldHelpText (sessionID, index)` returns a BSTR that is the configured authentication help text when the index is 0. When the index is 1, the configured authentication verification help text is returned.

Target COM Interface Properties

To use the target COM interfaces, include `CreateObject ("CourATLService.TargetList")` in the code and assign it to a local parameter. Using this local parameter, which represents the `TargetList` object, use the following interfaces:

- `AuthenticateUser (sessionID, UserPin, AccountDisable)` returns the result of authenticating the end user with the provided information. `AccountDisable` is `True` if the maximum number of attempts are reached and the end user account is disabled from using `PasswordCourier`.
- `NumberOfGroups (sessionID)` returns a short integer that indicates the number of configured password target groups.
- `GroupName (sessionID, index)` returns a BSTR that is the group name for specified index. Valid indexes are 0 to `NumberOfGroups-1`.
- `GroupHelpText (sessionID, index)` returns a BSTR that is the configured help text for the specified group index. Valid indexes are 0 to `NumberOfGroups-1`.
- `NumberOfTargets (sessionID, groupName)` returns a short integer that indicates the number of configured password targets.
- `TargetName (sessionID, groupName, index)` returns a BSTR that is the target name for the specified target index for the specified group. Valid indexes are 0 to `NumberOfTargets-1`.
- `TargetAlias (sessionID, groupName, index)` returns a BSTR that is the target alias for the specified target index for the specified group. Valid indexes are 0 to `NumberOfTargets-1`.
- `TargetHelpText (sessionID, groupName, index)` returns a BSTR that is the configured help text for the specified group's specified target index. Valid indexes are 0 to `NumberOfTargets-1`.

Password COM Interface Properties

To use the password COM interfaces, include `CreateObject ("CourATLService.Password")` in the code and assign it to a local parameter. Then, using this local parameter, which represents the `Password` object, use the following properties:

- `StartAction (sessionID, groupName, targetName, result)` returns a BSTR that is the configured start action message or error message returned from server. The `groupName` and `targetName` indicate the password reset target. The returned result is `True` if the server successfully creates and updates the ticketing information, otherwise the return is `False`.”
- `UseEscrowPassword (sessionID)` returns a boolean that indicates whether escrow passwords are used. `True` indicates that escrow passwords are used and, therefore, there is no need to prompt the end user for a password. `False` indicates that there are no escrow passwords and, therefore, end users should be prompted for a new password and password verification.
- `PasswordFieldLabel (sessionID, index)` returns a BSTR that is the configured password field label when `index = 0`. When the `index = 1` it returns the verify password field label.
- `PasswordFieldHelpText (sessionID, index)` returns a BSTR that is the configured password field help text when `index = 0`. When the `index = 1` it returns the configured verify password help text.
- `UseComments (sessionID)` returns a boolean that indicates whether or not comments are used. `True` indicates that the end user should be presented with a comments field. `False` indicates that no comments field should be presented.
- `CommentLabel (sessionID)` returns a BSTR that is the configured comment field label.
- `CommentHelpText (sessionID)` returns a BSTR that is the configured help text for the comment field.

Reset Password COM Interface Properties

To use the `ResetPassword` COM interfaces, include `CreateObject(“CourATLService.ResetPassword”)` in the code and assign it to a local parameter. Then using this local parameter, which represents the `ResetPassword` object, use the following properties:

- `ValidatePassword (sessionID, password)` returns a BSTR that indicates the success or failure of checking the specified password according to password strength rules.
- `ResetPassword (sessionID, password, comment)` returns a BSTR that is the configured success or failure message sent after the success or failure of the actual password reset with the specified password and comment.

Profile COM Interface Properties

This COM interface is used in the sample `PBUserInfo.asp` to get the list of configured fields for profile creation/update and their configured labels (for example, help text). To use this interface, include `CreateObject(“CourATLService.Profile”)` in the code and assign it a local parameter. With this local parameter, which represents the `Profile` object, use the following properties:

- `PBAuthenticateUser(sessionID, UserPin, AccountDisable)` returns the end user authentication string result. `AccountDisable` is `True` if the maximum attempts are reached and the end user account is disabled from using `ProfileCourier`.

- `PBFieldCount(sessionID)` returns a short integer that is the number of configured fields for profile creation/update.
- `PBFieldAttr(sessionID, index)` returns a BSTR that is the field attribute for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.
- `PBFieldHelp(sessionID, index)` returns a BSTR that is the configured help text for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.
- `PBFieldLabel(sessionID, index)` returns a BSTR that is the configured field label for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.
- `PBFieldMaxLength(sessionID, index)` returns a short integer that is the configured maximum entered input length for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.
- `PBFieldMinLength(sessionID, index)` returns a short integer that is the configured minimum entry length for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.
- `PBFieldValue(sessionID, index)` returns a BSTR that is the current database value for an existing end user profile record for the specified field index. Valid indexes are 0 to `PBFieldCount-1`.

ProfileAction COM Interface Properties

This COM interface is used in the sample `UpdateProfile.asp` to perform profile creation/update. To use this interface, include `CreateObject("CourATLService.ProfileAction")` in the code and assign it a local parameter. With this local parameter, which represents the `ProfileAction` object, use the following property:

- `UpdateProfile(sessionID, sFieldValues, bCreateNew)` returns the result of a profile creation/update and creates a new profile with the `sFieldValues` if the `CreateNew` flag is set to `True`. The existing profile is updated if the flag is `False`.

Chapter 4: Best Practices for Implementing the Connector for Web Services

This chapter describes some recommendations for how to implement the Connector for Web Services at your site, and includes the following sections:

- [*“Implementation Overview” on page 46*](#)
- [*“Best Practices” on page 48*](#)
- [*“Types of WSDLs That Can Be Auto-Discovered” on page 53*](#)

Implementation Overview

The connector for Web Services allows you to connect to other applications that support web services. The connector has the ability to auto-discover published Web Services Definition Language (WSDL) files using HTTP. This chapter provides a guide for developing Service Oriented Architecture (SOA) solutions using Core Security's RDK technology, specifically focusing on web service development and deployment.

The auto-discovery mechanism builds a sample XML document representing the request-side of a web service. Each XML document represents an individual operation exposed by the web service.

When configuring the web service connector within Core Security's Administration Manager, operations exposed by the WSDL file appear as target objects. XML document elements appear as a list of attributes. Each attribute name is an XPath representation of where you can find the element in the XML document.

The following is an example of this representation:

XML Document fragment:

```
<profile>
  <name/>
  <address>
    <street/>
    <city/>
    <state/>
    <zipcode/>
  </address>
</profile>
```

Attribute naming convention (XPath):

```
"profile/name"
"profile/address/street"
"profile/address/city"
"profile/address/state"
"profile/address/zipcode"
```

The web service connector uses the "rdkwebbservice.js" library to integrate Web Services with the Core Access Assurance Suite using the RDK. This library performs WSDL parsing, SOAP packet building, and HTTP(S) communication with the published web service.

Once a web service is auto-discovered, a global javascript object (gWebService_WSDLCache[<wsdl name>]) is created that caches the discovery of the WSDL, its operations, and the XML payloads associated with those operations. Calling the WebService_RequestPkt() function returns an instance of the global javascript object. You can manipulate this instance without altering the global object, which allows multiple threads of execution to provision against the same Web Service at the same time.

Once an instance of the web service is retrieved with the WebService_RequestPkt() call, you can change the XML payload, add values, and modify SOAP headers. After completing this process, the resulting request packet object is used as the single parameter to WebService_InvokeWS().

The `WebService_InvokeWS()` is responsible for final SOAP preparations, authentication, and the eventual transaction of sending the SOAP packet to the destination web service. A `WebService_ResponsePkt` object returns from the `WebService_InvokeWS()` call. This object contains the response XML payload in SOAP format along with any SOAP-level faults that were returned from the web service.

There are target-level parameters that enable you to define XSLT files that you can use to transform the response XML payload back into an XML structure that is understandable by the RDK. You can place this resulting XML into the RDK-Defined "respObj.xmlDoc" structure for passage back up to the Core Security workflow for further processing. In the case of AcctInfo and Query type interfaces, you can define XSLT files that translate the response back into requests so that generic algorithms can be developed to correlate response elements with what was requested.

Best Practices

There are a few best practices that you can follow that make it easier to build and integrate a web service solution into the Core Access Assurance Suite. Being able to dictate how the web service on the remote side is constructed provides the best method for achieving success with building a web services connector.

Avoid Iterations

Because the Administration Manager normalizes all complex structures into attribute/value pairs, building a web service that contains iteration results in only the first iteration being populated. An example of a specific iteration could be to provide multiple addresses on a profile (home and work).

The XML payload representing this could be:

```
<profile>
  <address type="home">
    <street/>
    <city/>
    <state/>
    <zip/>
  </address>
  <address type="work">
    <street/>
    <city/>
    <state/>
    <zip/>
  </address>
</profile>
```

To minimize integration requirements and allow the auto-discovery mechanisms to work to your advantage, create an interface that expands the iteration out into its specific representation (if possible).

The following is an example of the previous XML payload expanded out to remove the iteration:

```
<profile>
  <homeaddress>
    <street/>
    <city/>
    <state/>
    <zip/>
  </homeaddress>
  <workaddress>
    <street/>
    <city/>
    <state/>
    <zip/>
  </workaddress>
</profile>
```


Note: If you cannot define each type of iteration explicitly, be aware that only the first iteration is managed by the auto-discovery mechanism and Core Security's workflow engine. The RDK writer needs to manage all additional iterations of the same structure through custom javascript.

Avoid Tag Attributes

Tag attributes help to describe the behavior or representation of a particular XML element. However, representing tag attributes as XPath can result in a visually unattractive attribute name. To avoid this, create an XML document that only contains elements.

The following is an example of a tag attribute and how you can convert it into an XML element:

Tag attribute:

```
<product brand="Dole"/>
```

Converted to an XML element:

```
<product>
  <brand>Dole</brand>
</product>
```

Avoid SOAP Header Definitions

The SOAP protocol allows a web service operation to define an XML payload in the <soap:Header> and <soap:Body>. Typically, you use SOAP headers to provide authentication information and other global parameters. In most cases, this kind of information is placed in the header portion of the SOAP packet to help separate XML definition and implementation.

If the auto-discovery mechanism exposes both Header and Body elements, this requires the attribute names to evolve into structures that also include the words "soap:Header/" and "soap:Body/" as the first part of the name. This creates takes up space on Administration Manager forms, especially with attributes that are nested multiple levels deep in the XML payload. For this reason, the web services connector only supports XML definitions contained in the soap:Body section of a web service allowing the Core Security Service to assume "soap:Body/<operation>/" during configuration and execution.

If <soap:Header> structures are unavoidable, the global javascript object built from the WSDL discovery contains an XML document representing the Header.

Handle Namespacing Properly

When a SOAP packet is built and delivered, every element in the XML payload contains a namespace as defined by the WSDL. It is up to the web service to handle receiving such a qualified document. One XPath technique that you can use to extract XML elements that always work is as follows:

XML document:

```
<m:profile xmlns:m="http://yourwebservice.com">
  <m:name>John</m:name>
</m:profile>
```

XPath for retrieving the "name" tag:

```
"/*[local-name()='profile']/*[local-name()='name']"
```

The XPath function `local-name()` isolates tag names without their corresponding namespace. This type of XPath query is slightly slower to perform than an explicit query of `"/profile/name."` However, always works regardless of the namespaces being applied to the elements.

If namespacing is important to the implementation of the web service, you can read in all the namespaces encountered for each element and apply the appropriate prefix when building the XPath. This alternative is much more implementation intensive and may outweigh the advantage of having quicker XPath lookups.

Use SOAP Error/Fault Mechanism for Reporting Errors

By default, the `rdkwebbservice.js` file examines the `<soap:fault>` structure looking for errors that occurred during operation. These errors are then sent back to the Core Security application to be logged and sent to the end-user.

The RDK treats all errors the same regardless of where they occur (at the application or system level), to save on implementation cost, it would be beneficial to use the `<soap:fault>` capability to send back application level errors in the same way that system-level errors are normally returned in this area (such as when the XML payload is malformed or the WebService is down).

When the `webservices.js` library detects a `<soap:fault>`, the `WebService_ResponsePkt` object sets the `ErrorOccurred` attribute to true and populates the `errorMsg` attribute with the contents of the `<soap:fault>` tag. This results in a simplified implementation for determining errors during web service interaction. The following javascript illustrates this:

```
var respPkt = WebService_InvokeWS(reqPkt);
if (respPkt.bErrorOccurred){
    respond_statusFailure(respObj, respPkt.errorMsg);
}
```

Keep Request and Response XML Payloads Similar

Ensure that the request and the response payload are as similar as possible. This results in lower implementation costs when building XSLT files for translation from the web service back into the original request structure discovered and shown in the Administration Manager.

This ease in translation allows for the javascript implementation to determine what response elements correlate to what was requested. This is especially important for "query" type interfaces (such as "Query", "Native Query", "Update Ticketing", and "AcctInfo"). It allows for a generic algorithm for correlating and assigning return values to attributes whose names are known by the Core Security workflow engine based on the request payload (and not the response payload).

The following is an example of a request and response that are similar in structure:

Request Payload:

```
<profileReq>
  <name/>
  <birthday/>
  <email/>
  <phone/>
</profileReq>
```

Response Payload:

```
<profileResp>
  <name/>
  <birthday/>
  <email/>
  <phone/>
</profileResp>
```

The following is a sample XSLT document that you can use to transform response packets back into request packets (minus namespacing) where the only difference is the top level element:

```

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/
Transform">
  <xsl:output method="xml" version="1.0" encoding="UTF-8"
    indent="yes" omit-xml-declaration="yes"/>
  <xsl:template match="/">
    <!-- These variables represent the top level tagname
         differences between request and response -->
    <xsl:variable name="reqTagName">
      <xsl:text disable-output-escaping="yes">courstart</xsl:text>
    </xsl:variable>
    <xsl:variable name="respTagName">
      <xsl:text disable-output-escaping="yes">Courion_bpel</
        xsl:text>
    </xsl:variable>

    <!-- Output the XML structure -->
    <xsl:for-each select="*">
      <xsl:call-template name="ReplaceTag">
        <xsl:with-param name="tagName">
          <xsl:choose>
            <xsl:when test="local-name()=$respTagName">
              <!-- Replace the response tag with the request
                   tag -->
              <xsl:value-of select="$reqTagName"/>
            </xsl:when>
            <xsl:otherwise>
              <xsl:value-of select="local-name()"/>
            </xsl:otherwise>
          </xsl:choose>
        </xsl:with-param>
      </xsl:call-template>
    </xsl:for-each>
  </xsl:template>

  <xsl:template name="ReplaceTag">
    <xsl:param name="tagName"/>
    <xsl:text disable-output-escaping="yes">&lt;</xsl:text>
    <xsl:value-of select="$tagName"/>
    <xsl:for-each select="@*">
      <xsl:text disable-output-escaping="yes"> </xsl:text>
      <xsl:value-of select="local-name()"/>
      <xsl:text disable-output-escaping="yes">="</xsl:text>
      <xsl:value-of select="."/>
      <xsl:text disable-output-escaping="yes">"</xsl:text>
    </xsl:for-each>
    <xsl:text disable-output-escaping="yes">&gt;</xsl:text>
    <xsl:value-of select="text()"/>
    <xsl:for-each select="*">
      <xsl:call-template name="ReplaceTag">
        <xsl:with-param name="tagName" select="local-name()"/>
      </xsl:call-template>
    </xsl:for-each>
    <xsl:text disable-output-escaping="yes">&lt;</xsl:text>
    <xsl:value-of select="$tagName"/>
    <xsl:text disable-output-escaping="yes">&gt;</xsl:text>
  </xsl:template>
</xsl:stylesheet>

```

Types of WSDLs That Can Be Auto-Discovered

Most WSDL definitions defined using industry leading RAD tools work with the WSDL auto-discovery algorithm in the `rdkwebservices.js` library. Here are two public WSDL definitions that can be successfully discovered by the RDK. These definitions are not necessarily provisioning-type services; however you can still use them as a learning tool to determine what can be discovered and exposed by the Administration Manager.

<http://api.google.com/GoogleSearch.wsdl>

<http://webservices.amazon.com/AWSECommerceService/AWSECommerceService.wsdl?>

Chapter 5: Configuring the Trusted Connection for Core Security Services and the Access Assurance Portal

This chapter describes how to enable connection strings for trusted connection. The connection strings in the Core Security Services and the Access Assurance Portal support trusted connection. Using a trusted connection prevents the need for credentials to be stored in clear text.

To enable the trusted connection in the connection strings, additional configuration is required that is described in detail in the following sections:

- [*“Setting Up a Service Account” on page 56*](#)
- [*“Configuring Trusted Connection in the Database” on page 57*](#)
- [*“Configuring Trusted Connection Authentication For Core Security Services” on page 58*](#)
- [*“Configuring Trusted Connection for the Access Assurance Portal” on page 64*](#)
- [*“Configuring the Internet Information Services” on page 66*](#)

Before you set up the trusted connection, you must configure a service account on the SQL server and the Active Directory. Refer to the Microsoft documentation to set up the service accounts.

Setting Up a Service Account

Each Core Security Service executes in the security context of a user account. This user account that is created on a Windows operating system within a particular domain is referred to as a service account. The service account appears as DOMAIN\Username. For example, user account “xyz” created on “abcdomain.com” appears as “ABCDOMAIN.COM\xyz”.

You must first set up a service account before configuring the Core Security Services and the Access Assurance Portal with a trusted connection. Refer to the Microsoft documentation to create a service account on the SQL server and in Active Directory.

Throughout this document, the service account mentioned in screen shots uses the convention “\$\$ServiceAccountName\$”.

Minimum Privileges Required for the Service Account in Active Directory

The following minimum privileges are required for a service account:

- The service account should be a domain user and also the local administrator of the Core Security server (the application server). Additionally, the service account requires the “Log on as batch job” privilege on the Core Security server.
- On the SQL server, the service account should have db_datawriter and db_datareader database roles with the Execute permission.
- Since the Identity Mapping Solution interacts with other databases besides the Core Security database for creating a data feed, the service account must have the same roles and permissions explained above on the external databases.

After the service account is set up, you can start configuring the connection strings for the Core Security Services, the Access Assurance Portal and the Internet Information Services (IIS) Manager with the trusted connection.

Configuring Trusted Connection in the Database

By setting the trusted connection to true in the database connection string, the Access Assurance Portal uses Windows Authentication to connect to the SQL server using the service account. Using `Trusted_Connection=True`, `Integrated Security=SSPI`, or `Integrated Security=True` enables the trusted connection between the Access Assurance Portal and the SQL server. Use one of the following examples to set the trusted connection to true in the database connection string:

```
connectionString="Data
Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Trusted_Connection=True"
```

```
connectionString="Data
Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Integrated Security=SSPI"
```

```
connectionString="Data
Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Integrated Security=True"
```

Note: To point the connection strings to the correct database, replace instances of `$$DatabaseServerName$$` with the name of the database server, and replace `$$Core SecurityDatabaseName$$` with the name of the Core Security database. These instances appear throughout this document. Replace the reference with the respective value as described in the note.

Configuring Trusted Connection Authentication For Core Security Services

Use the information described here to set up trusted connection for the Core Security Services that are described in detail in this section.

Configuring the Identity Mapping Service

To configure the Identity Mapping Service:

1. Open the IdentityMapping.WCFHost.ConnectionStrings.config located in the <Core Security Installation Folder>\Courion Corporation\Core SecurityService\Config folder.
2. Update the connection string as follows in the .config file:

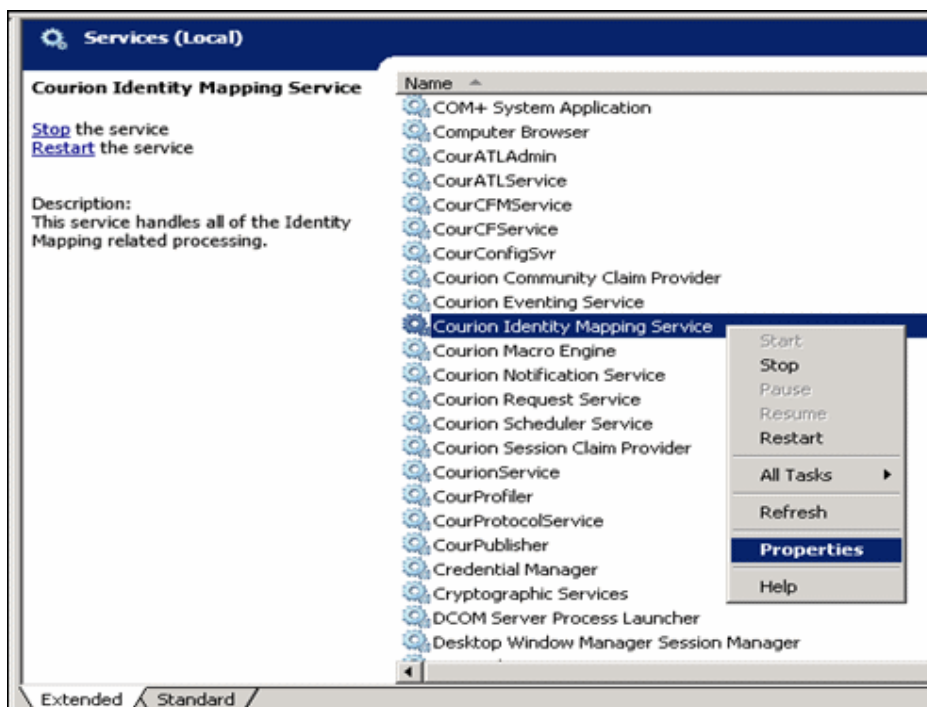
```
<connectionStrings>

<add name="dbConnectionString" connectionString="Data
Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

</connectionStrings>
```

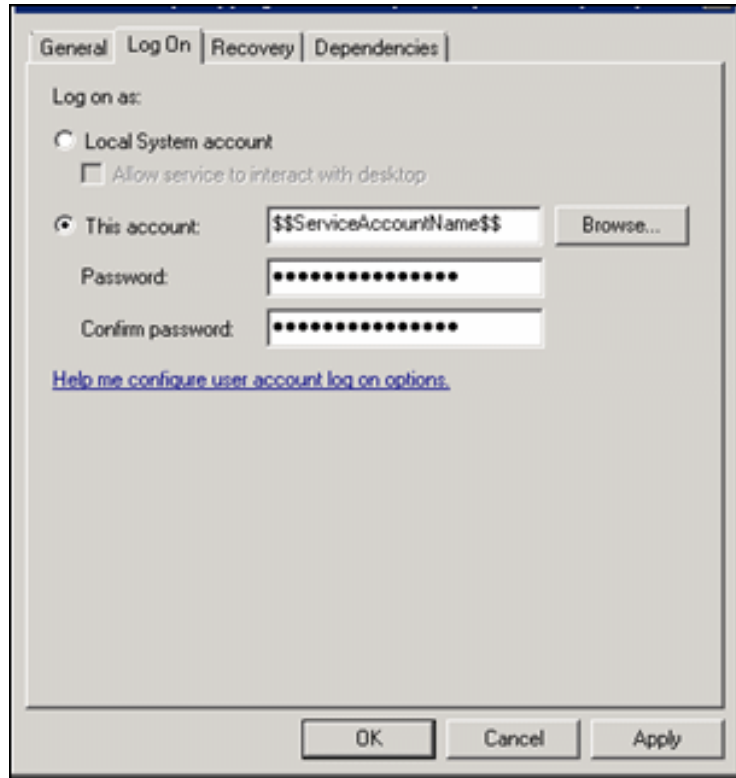
3. Open the Services window from the Control Panel, and right click on the Core Security Identity Mapping Service. Select **PROPERTIES** as shown.

Figure 9: Core Security Identity Mapping Service Properties



- On the **LOG ON** tab, select the option **THIS ACCOUNT**. Enter the service account credentials and select **OK** to save.

Figure 10: Enter the Service Credentials



- In the **SERVICES** window, right-click on the **CORE SECURITY IDENTITY MAPPING SERVICE** and select **RESTART** to accept the configuration changes.

Configuring the Core Security Notification Service

To configure the Notification Service:

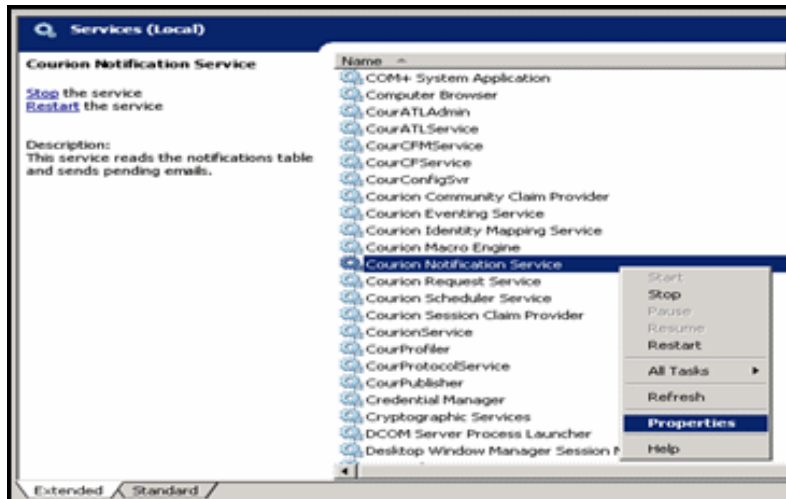
- Open the Core Security.Framework.NotificationService.exe.config file located in the <Core Security Installation Folder>\Courion Corporation\Core Security\Service\ folder.
- Update the connection string as follows:

```
<connectionStrings>

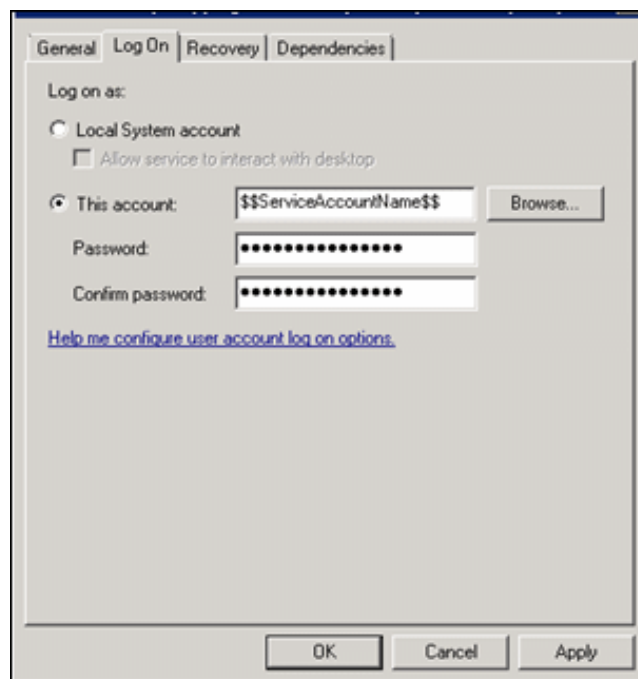
  <add name="dbConnectionString" connectionString="Data
Source=$$DatabaseServerName$$;Initial Catalog=$$Core
SecurityDatabaseName$$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

</connectionStrings>
```

- Open the **SERVICES** window from the Control Panel and right-click on the **CORE SECURITY NOTIFICATION SERVICE**. Select **PROPERTIES**.

Figure 11: Core Security Notification Service Properties

4. On the **LOG ON** tab, select the option **THIS ACCOUNT**. Enter the service account credentials and select **OK** to save.

Figure 12: Enter the Service Credentials

5. From the **SERVICES** window, right-click on the **CORE SECURITY NOTIFICATION SERVICE** and select **RESTART** to accept the configuration changes.

Configuring the Core Security Request Service

To configure the Request Service:

1. Open the Core Security.Framework.RequestService.exe.config file located in the <Core Security Installation Folder>\Courion Corporation\Core SecurityService\ folder.
2. Update the connection string as follows:

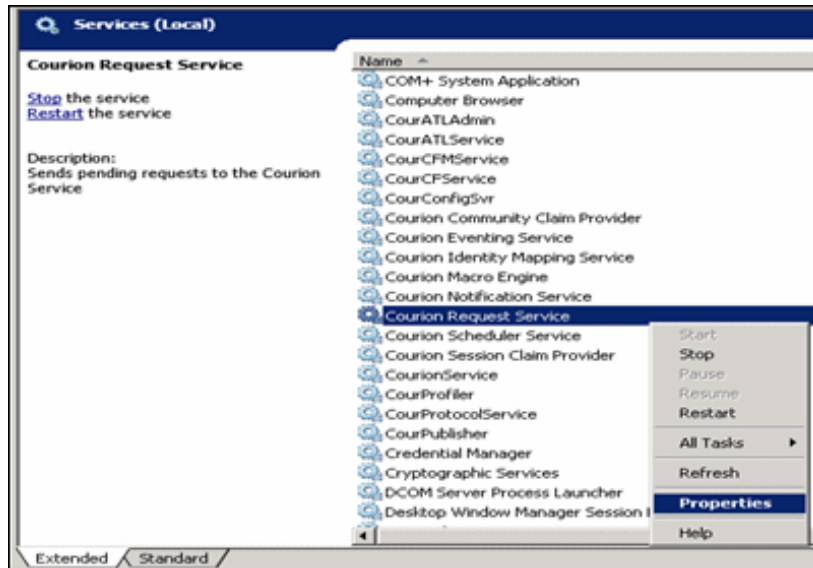
```
<connectionStrings>
```

```
<add name="dbConnectionString" connectionString="Data
Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />
```

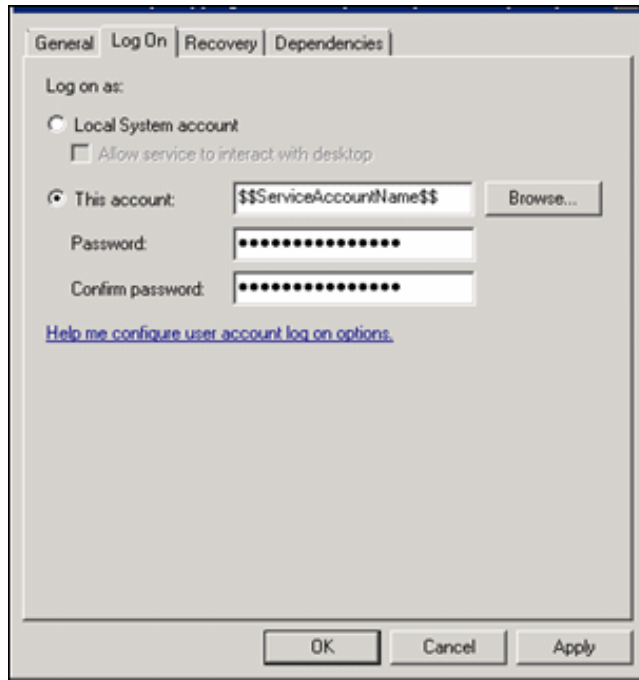
```
</connectionStrings>
```

3. Open the **SERVICES** window from the Control Panel, right-click on the **CORE SECURITY REQUEST SERVICE** and select **PROPERTIES**.

Figure 13: Core Security Request Service Properties



4. On the **LOG ON** tab, select the option **THIS ACCOUNT**. Enter the service account credentials and select **OK** to save.

Figure 14: Enter the Service Credentials

5. From the **SERVICES** window, right-click on the **CORE SECURITY REQUEST SERVICE** and select **RESTART** to accept the configuration changes.

Configuring the Core Security Scheduler Service

To configure the Scheduler Service:

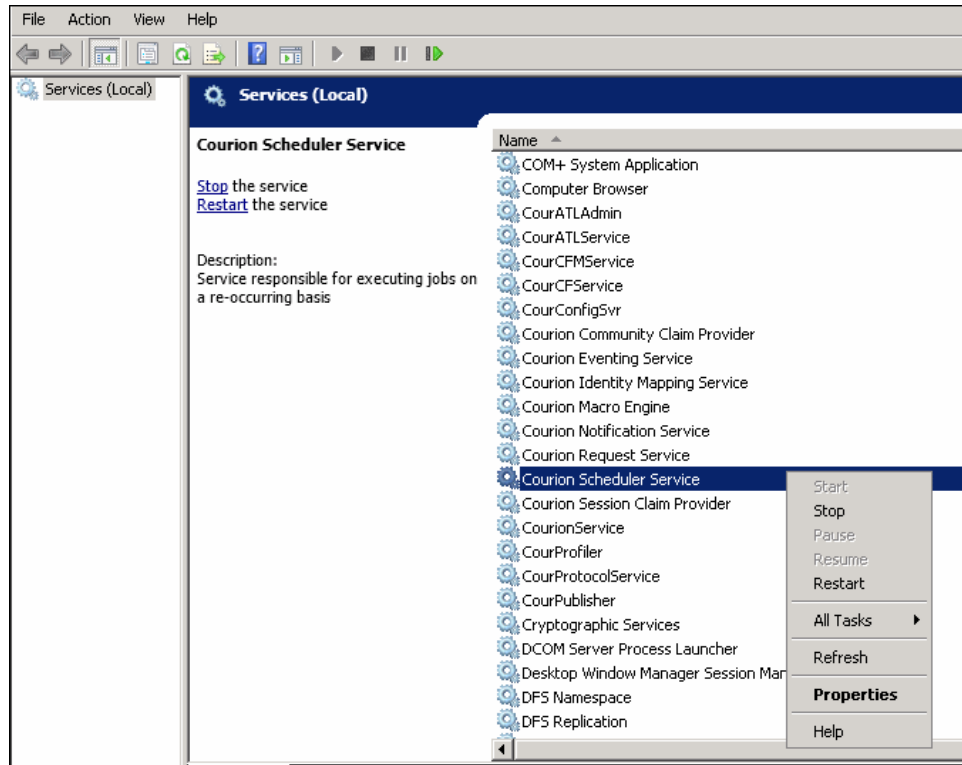
1. Open the Scheduler.WCFHost.ConnectionStrings.config file located in the <Core Security Installation Folder>\Courion Corporation\Core SecurityService\Config\ folder.
2. Update the connection string as follows:

```
<connectionStrings>

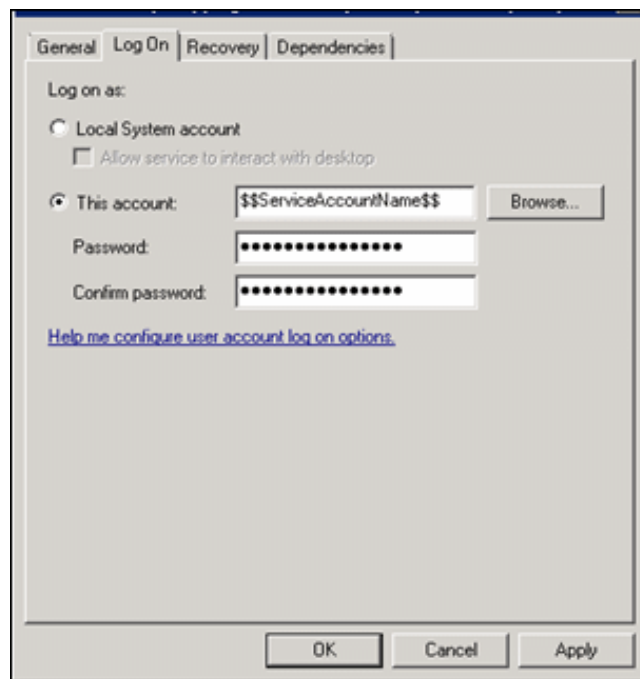
<add name="dbConnectionString" connectionString="Data
Source=$$DatabaseServerName$$;Initial Catalog=$$Core
SecurityDatabaseName$$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

</connectionStrings>
```

3. Open the **SERVICES** window from the Control Panel and right-click on the **CORE SECURITY SCHEDULER SERVICE**. Select **PROPERTIES**.

Figure 15: Core Security Scheduler Service Properties

4. On the **LOG ON** tab, select the option **THIS ACCOUNT**. Enter the service account credentials and select **OK** to save.

Figure 16: Enter the Service Credentials

5. From the **SERVICES** window, right-click on the **CORE SECURITY SCHEDULER SERVICE** and select **RESTART** to accept the configuration changes.

Configuring Trusted Connection for the Access Assurance Portal

To configure the trusted connection for the Access Assurance portal:

1. Open the CustomerConnStrings.config file located in the <Core Security Installation Folder>\Courion Corporation\CourionARMS\ folder.
2. Update the following connection strings:
 - MetricRepositoryDefault
 - Default
 - dbConnectionString
 - ARMEntities
 - AccessEntities

```
<add name="MetricRepositoryDefault"

connectionString="Data Source= $$DatabaseServerName $$;

Initial Catalog= $$Core
SecurityDatabaseName $$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

<add name="Default"

connectionString="Data Source= $$DatabaseServerName $$;

Initial Catalog= $$Core
SecurityDatabaseName $$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

<add name="dbConnectionString"

connectionString="Data Source= $$DatabaseServerName $$;

Initial Catalog= $$Core
SecurityDatabaseName $$;Trusted_Connection=True"
providerName="System.Data.SqlClient" />

<add name="ARMEntities"

connectionString="metadata=res://*/
;provider=System.Data.SqlClient;provider connection
string=&quot;

Data Source= $$DatabaseServerName $$; Initial
Catalog= $$Core
SecurityDatabaseName $$;Trusted_Connection=True;

MultipleActiveResultSets=True&quot;;"
providerName="System.Data.EntityClient" />
```



```
<add name="AccessEntities"

connectionString="metadata=res://*/AccessModel.csdl|res://
/*/*AccessModel.ssdl|res://*/
AccessModel.msl;provider=System.Data.SqlClient;provider
connection string=&quot;

Data Source= $$DatabaseServerName$$;Initial Catalog= $$Core
SecurityDatabaseName$$;Trusted_Connection=True;

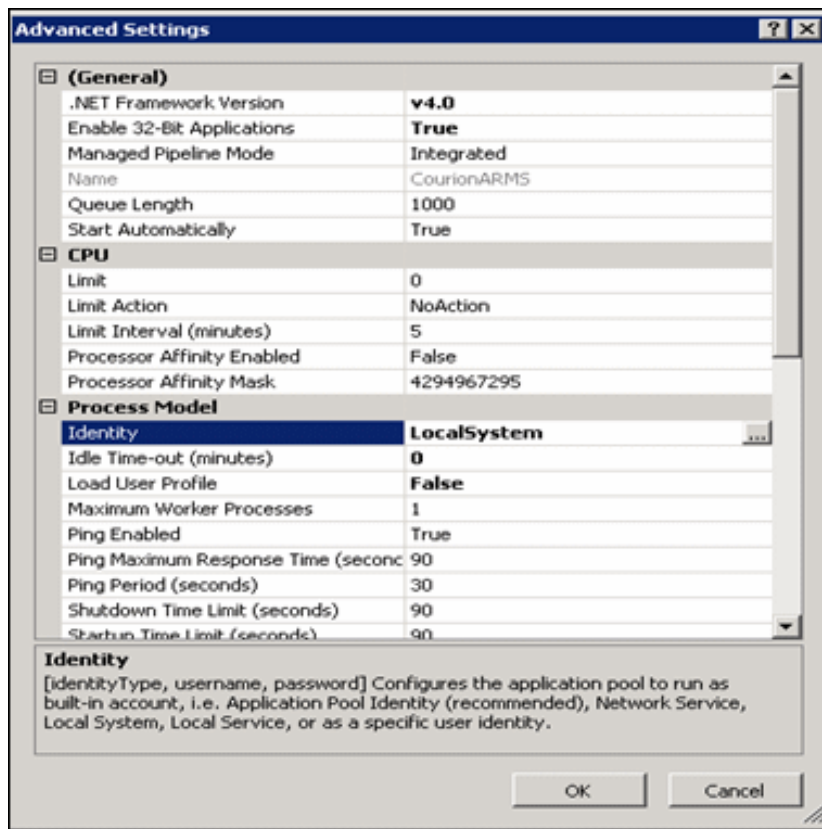
multipleactiveresultsets=True;App=EntityFramework&quot;;"
providerName="System.Data.EntityClient" />
```

Configuring the Internet Information Services

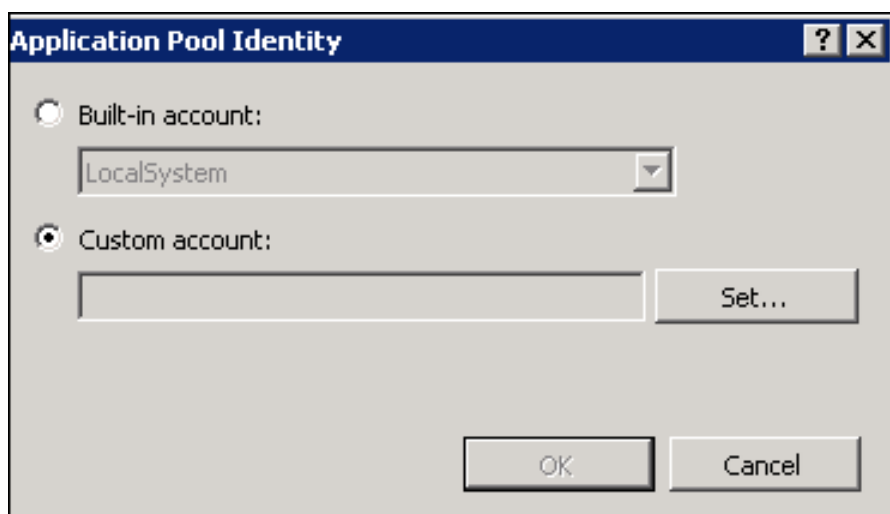
When a database connection string in the CustomerConnectionStrings.config file is set with the Trusted_Connection=True, Windows Authentication is enabled to log into the SQL server. To leverage this setting, follow these steps:

1. Go to Start and select Run. In the Run window, enter *inetmgr* to launch the Internet Information Service (IIS) Manager.
2. From the Connections pane on the left, expand the server node and click Application Pools.
3. On the Application Pools page, right-click the CoreARMS application pool, and select Advanced Settings.

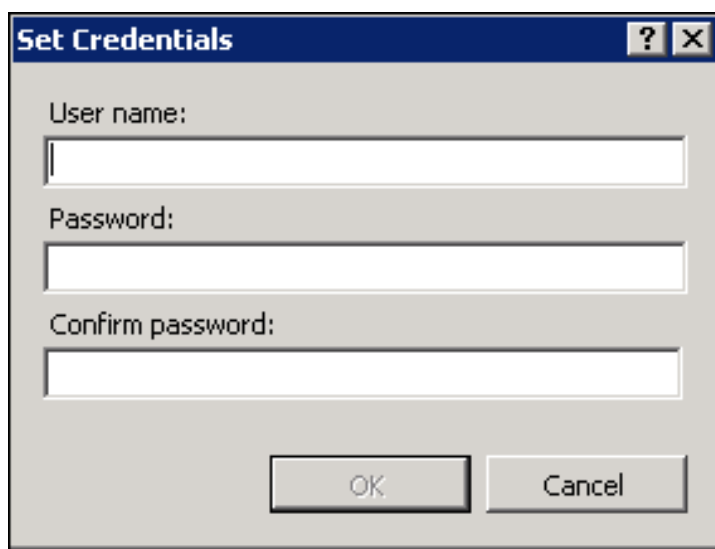
Figure 17: Advanced Settings



4. For the Identity property, click the ... button on the right to open the Application Pool Identity dialog box.

Figure 18: Application Pool Identity

5. To use custom identity, select Custom account, and click Set to open the Set Credentials dialog box.

Figure 19: Set Credentials

6. Enter the Service Account name in the User Name text box, and the Service Account password in the Password text box. Confirm the password and click OK.
7. Click OK to close the Application Pool Identity dialog box.
8. Stop and Start the CoreARMS application pool.
9. Restart the IIS Manager and the Core Security Services.
10. Go to [http://\[machine-name or IP address\]/CoreARMS/AspxCommon/PortalHome.aspx](http://[machine-name or IP address]/CoreARMS/AspxCommon/PortalHome.aspx) to confirm that everything is working correctly.

Chapter 6: Securing the Access Assurance Portal

This chapter explains how the Access Assurance Portal can be secured by encrypting the data communication between the AAS application server and an Internet browser as well as between the AAS application server and the Microsoft SQL Server. You can implement encryption to secure communication between a browser and the AAS application server by using Transport Layer Security (TLS)/ Secure Sockets Layer (SSL). To secure communication between the AAS application server and Microsoft SQL Server, you can use Force Protocol Encryption. These encryption techniques are described in the following sections:

- [*“Using Transport Layer Security \(TLS\)/ Secure Sockets Layer \(SSL\)” on page 70*](#)
- [*“Configuring Force Protocol Encryption” on page 79*](#)

Using Transport Layer Security (TLS)/ Secure Sockets Layer (SSL)

The Transport Layer Security (TLS)/ Secure Sockets Layer (SSL) protocols encrypt the data so that only the intended recipient can understand it. They use certificates to authenticate the systems with whom they are communicating and to negotiate a symmetric session key, which is used to encrypt the data communication between the systems. To use TLS/SSL for the Access Assurance Portal, you need to take the following steps, which are explained in the following sections:

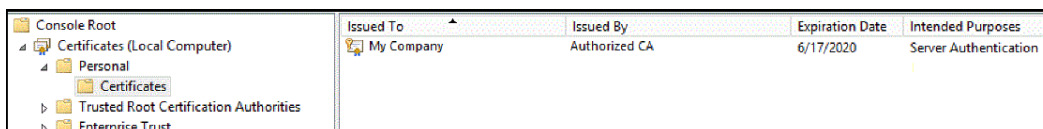
- [“Configuring TLS/SSL” on page 70](#)
- [“Configuring the Web Services” on page 74](#)
- [“Configuring the Tomcat Server” on page 77](#)

Configuring TLS/SSL

Perform the following steps to configure TLS/SSL on the Access Assurance Portal:

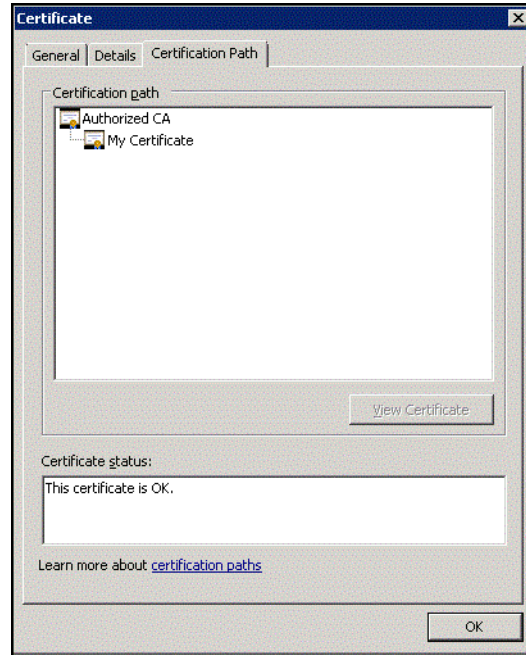
1. Copy the server certificate (*.pfx) file to the Core Security Web Server where the AAS is installed.
2. Open the Certificate Manager Tool by executing `mmc.exe` from Start Menu > Run. The Management Console appears.
3. Perform the following steps to add the Certificates Snap-in:
 - a. On the **FILE** menu, select **ADD/REMOVE SNAP-IN...** to open the Add or Remove Snap-ins screen.
 - b. From the **AVAILABLE SNAP-INS** box, select **CERTIFICATES** and click **ADD>**. The Certificate Snap-in pop-up appears.
 - c. Select the **COMPUTER ACCOUNT** option and click **NEXT**. The Select Computer pop-up appears.
 - d. Select the **LOCAL COMPUTER** option and click **FINISH** to return to the Add or Remove Snap-ins screen. Click **OK** to return to the Management Console.
4. On the left panel, under the **CONSOLE ROOT**, click on **CERTIFICATES (LOCAL COMPUTER)** to expand it.
5. Select the **TRUSTED PEOPLE** store and take a backup of the Server certificate provided by Core Security, which appears in the right panel.
For the client certificate, you can use your company-specific certificate or the certificate provided by Core Security. Make sure that the certificate is in the PFX format.
6. Import your company-specific SSL server certificate (*.pfx) in the **PERSONAL** store and the **TRUSTED PEOPLE** store.

Figure 20: Personal Store



7. To ensure that the certificates are installed properly, perform the following steps:
 - a. Double-click on the certificate. The Certificate screen appears.
 - b. Click the **CERTIFICATION PATH** tab and check if the certificate is under the Trusted Root Certification Authorities > Certificates as shown in the figure.

Figure 21: Certification Screen



- c. If the root Certificate is not present, search for it and import it to Trusted Root Certification Authorities > Certificates.

Figure 22: Trusted Root Certification Authorities

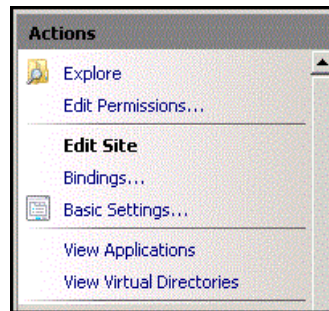
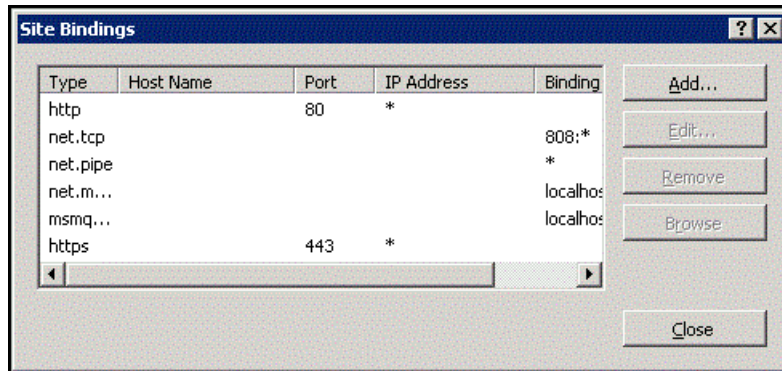
	Issued To	Issued By	Expiration Date
Console Root			
Certificates (Local Computer)			
Personal			
Certificates			
Trusted Root Certification Authorities			
Certificates			
Enterprise Trust			
Intermediate Certification Authorities			
Trusted Publishers			
	Microsoft Root Certificate Authority...	Microsoft Root Certificate Authority ...	3/22/2036
	Microsoft Root Certificate Authority...	Microsoft Root Certificate Authority ...	6/23/2035
	Microsoft Root Certificate Authority	Microsoft Root Certificate Authority	5/9/2021
	My Company	Authorized CA	6/17/2020
	Microsoft Root Certificate Authority	Microsoft Root Certificate Authority	5/9/2021
	Microsoft Root Certificate Authority...	Microsoft Root Certificate Authority ...	6/23/2035
	Microsoft Root Certificate Authority	Microsoft Root Certificate Authority	5/9/2021

8. From the Start menu, open the Internet Information Services (IIS) Manager. In the left pane, click on the + icon to expand the **MACHINE NAME** and then expand **SITES** and select the **DEFAULT WEB SITE** node.

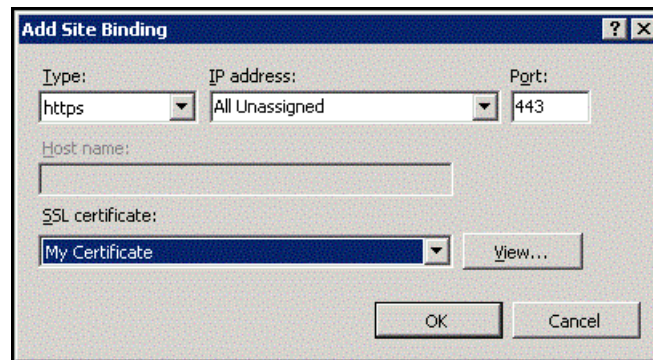
Figure 23: Default Web Site Node



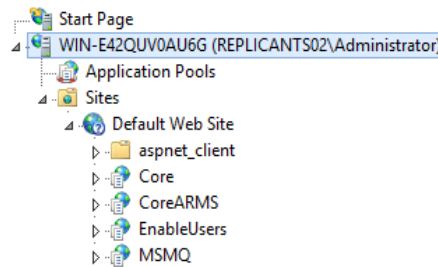
9. On the Actions pane, select **BINDINGS...** The Site Bindings pop-up appears.

Figure 24: Actions Pane**Figure 25: Site Bindings**

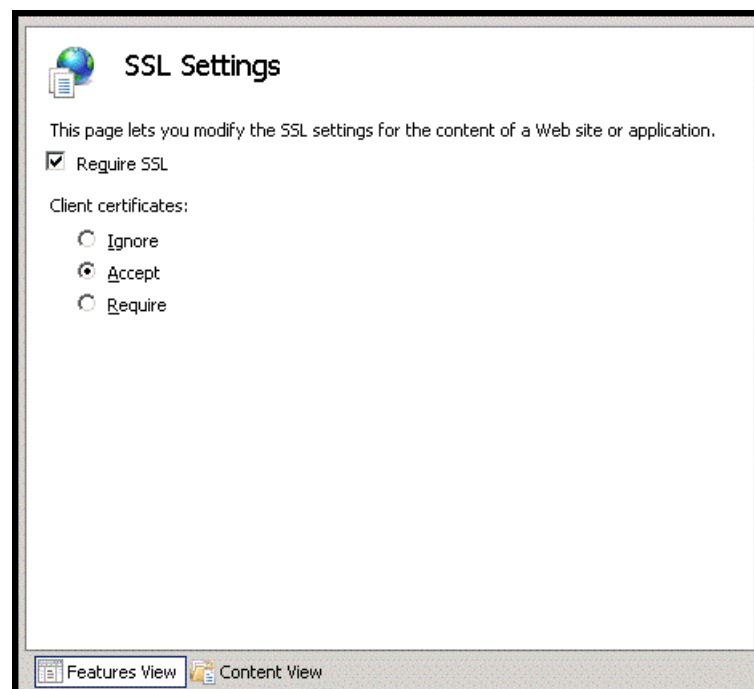
- Click **ADD...** The Add Site Binding pop-up appears. In the **TYPE** drop-down list, select **HTTPS** and in the **HOST NAME** drop-down list, select the name of the certificate you have imported. Click **OK**.

Figure 26: Add Site Bindings

- You will return to Site Bindings pop-up. Click **CLOSE** to return to the Information Services (IIS) Manager.
- On the left pane, expand the **DEFAULT WEB SITE** node and select the **COREARMS** directory.

Figure 27: CoreARMS Directory

13. On the **FEATURE VIEW** pane, double-click the **SSL SETTINGS** icon under the **IIS** panel to open SSL Settings.

Figure 28: SSL Settings

14. Select the **REQUIRE SSL** check box. The **ACCEPT** option is automatically selected.
15. Close the Internet Information Services (IIS) Manager. You can now open the Access Assurance Portal using the HTTPS protocol. For example:
<https://<FQDN>/CoreARMS>.

Configuring the Web Services

After configuring TLS/SSL, you must configure the web.config file to ensure that the Certifications and DataCollectionService web services communicate properly when the HTTPS protocol is used.

Perform the following steps to configure the web.config file:

1. Open the [Core Security\InstallPath]\CoreARMS\web.config file in a text editor.
2. Under the <behavior name="SecureCertificationsBehavior"> node, un-comment the HTTPS (SSL) related sections and comment out the HTTP related sections as shown below. Also, change the value in the findValue attribute with the subject in your SSL server certificate.

```
<behavior name="SecureCertificationsBehavior">

    <dataContractSerializer maxItemsInObjectGraph="2147483647" />

    <!-- Use this setting when running AccessAssuranceSuite
    Server under HTTP -->

    <!-- <serviceMetadata httpGetEnabled="true" /> -->

    <!-- Use this setting when running AccessAssuranceSuite
    Server under HTTPS (SSL) -->

    <serviceMetadata httpsGetEnabled="true" />

    <serviceDebug includeExceptionDetailInFaults="true" />

    <serviceCredentials>

        <serviceCertificate findValue="$$Subject in SSL server
        certificate$$" storeLocation="LocalMachine"
        storeName="TrustedPeople" x509FindType="FindBySubjectName" />

        <clientCertificate>

            <!-- Use this setting when running AccessAssuranceSuite
            Server under HTTP -->

            <!-- <authentication certificateValidationMode="PeerTrust" /> -->

            <!-- Use this setting when running AccessAssuranceSuite
            Server under HTTPS (SSL) -->

            <authentication certificateValidationMode="PeerTrust"
            revocationMode="NoCheck" />

        </clientCertificate>

    </serviceCredentials>

</behavior>
```

Change \$\$Subject in server certificate\$\$ to the subject in your SSL server certificate.

3. Under the `<behavior name=" SecureDataCollectionBehavior ">` node, un-comment the HTTPS (SSL) related sections and comment out the HTTP related sections as shown below. Also, change the value in the `findValue` attribute with the subject in your SSL server certificate.

```
<behavior name="SecureDataCollectionBehavior">

    <!-- Use this setting when running AccessAssuranceSuite
    Server under HTTP -->

    <!-- <serviceMetadata httpGetEnabled="true" /> -->

    <!-- Use this setting when running AccessAssuranceSuite
    Server under HTTPS (SSL) -->

    <serviceMetadata httpsGetEnabled="true" />

    <serviceDebug includeExceptionDetailInFaults="true" />

    <serviceCredentials>

        <serviceCertificate findValue="$$Subject in SSL server
        certificate$$" storeLocation="LocalMachine"
        storeName="TrustedPeople" x509FindType="FindBySubjectName" />

        <clientCertificate>

            <!-- Use this setting when running AccessAssuranceSuite
            Server under HTTP -->

            <!-- <authentication certificateValidationMode="PeerTrust"
            /> -->

            <!-- Use this setting when running AccessAssuranceSuite
            Server under HTTPS (SSL) -->

            <authentication certificateValidationMode="PeerTrust"
            revocationMode="NoCheck" />

        </clientCertificate>

    </serviceCredentials>

</behavior>
```

Change `$$Subject` in `server certificate$$` to the subject in your SSL server certificate.

4. Under the `<binding name="DataCollectionBinding" maxReceivedMessageSize="4000000">` node, un-comment the HTTPS (SSL) related sections and comment out the HTTP related sections as shown below.

```
<binding name="DataCollectionBinding" maxReceivedMessageSize="4000000">

    <readerQuotas maxDepth="400" maxStringContentLength="4000000" />

    <!-- Use this setting when running AccessAssuranceSuite
Server under HTTP -->

    <!-- <security mode="Message">

        <message clientCredentialType="Certificate" />

    </security> -->

    <!--Use this setting for security node when running
AccessAssuranceSuite Server under HTTPS (SSL)-->

    <security mode="TransportWithMessageCredential">

        <message clientCredentialType="Certificate"
negotiateServiceCredential="true"/>

        <transport clientCredentialType="Certificate"/>

    </security>

</binding>
```

5. Under the `<binding name="CertificationsBinding" maxReceivedMessageSize="104857600" maxBufferPoolSize="104857600">` node, un-comment the HTTPS (SSL) related sections and comment out the HTTP related sections as shown below.

```
<binding name="CertificationsBinding" maxReceivedMessageSize="104857600"
maxBufferPoolSize="104857600">

    <readerQuotas maxDepth="2147483647"
maxStringContentLength="104857600" />

    <!-- Use this setting when running AccessAssuranceSuite
Server under HTTP -->

    <!-- <security mode="Message">

        <message clientCredentialType="Certificate" />

    </security> -->

    <!--Use this setting for security node when running
AccessAssuranceSuite Server under HTTPS (SSL) -->

    <security mode="TransportWithMessageCredential">
```

```

        <message clientCredentialType="Certificate"
negotiateServiceCredential="true"/>

        <transport clientCredentialType="Certificate"/>

    </security>

</binding>

```

6. Under the `<behavior name="ClientSecuredByCertificates">` node, change the value in the `findValue` attribute with the subject in your client certificate if you want to use your company-specific client certificate.

```

<behavior name="ClientSecuredByCertificates">

    <clientCredentials>

        <clientCertificate findValue="$$Subject in Client
Certificate$$" storeLocation="LocalMachine"
storeName="TrustedPeople" x509FindType="FindBySubjectName" />

        <serviceCertificate>

            <authentication revocationMode="NoCheck" />

        </serviceCertificate>

    </clientCredentials>

</behavior>

```

Change `$$Subject in Client Certificate$$` to the subject in your client certificate.

7. Restart the Internet Information Services. The web services will now be accessible and can communicate over the HTTPS protocol.

Configuring the Tomcat Server

You must configure the Tomcat server to ensure that the data collection rules run properly on an application server with SSL configuration.

Perform the following steps to configure the Tomcat server:

1. Ensure that you have a server certificate's .cer (public key) file available on your server.
2. Open the Command Prompt and navigate to the JRE bin folder. For example:
`cd c:\Program Files\Java\jre7\bin`
3. Run the following command:

```

keytool.exe -import -alias Server -keystore
"[JRE-location]\lib\security\cacerts" -file
"[Certificate-file-location]\[Certificate-File-Name].cer"

```

Note: Use the location of your JRE folder and the Certificate file folder in the above command.

4. Create a configuration file with the same name as the Tomcat server executable file in the folder where the default Tomcat server executable file is located. The format of the configuration file name must be as follows:

{ExecutableFileName}.{extension}.config

For example, if the Tomcat server executable file name is tomcat7.exe and it is located in the C:\Tomcat\apache-tomcat-7.0.42\bin folder, you must create the configuration file named tomcat7.exe.config in the same folder.

5. Add the following content in the file:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="ServerAuthenticationCertificateOverride"
value="value=$$Subject in server certificate$$" />
    <add key="ClientAuthenticationCertificateOverride"
value="value=$$Subject in client certificate$$" />
    <add key="ServerAuthenticationCertificateOverrideStore"
value="My" />
    <add key="ClientAuthenticationCertificateOverrideStore"
value="TrustedPeople"/>
  </appSettings>
</configuration>
```

Change \$\$Subject in server certificate\$\$ to the subject in your server certificate.

Change \$\$Subject in client certificate\$\$ to the subject in your client certificate.

6. Restart the Tomcat server.

Configuring Force Protocol Encryption

Force Protocol Encryption ensures the data communication between the AAS application server and the Microsoft SQL Server is encrypted using a valid server certificate installed on the Microsoft SQL Server. To enable Force Protocol Encryption, you must update all the connection strings in the CustomerConnStrings.config file to add the attribute, `Encrypt=true`;

Following is an example showing how to update the DataCollectionEntities and dbConnectionString connection strings:

- Update the DataCollectionEntities connection string to add the attribute, 'Encrypt=true' while using SQL authentication or Windows authentication.

The following is an example of SQL authentication:

```
<add name="DataCollectionEntities"
connectionString="metadata=res://*/
DataCollectionModel.csdl|res:
//*/DataCollectionModel.ssdl|res://*/
DataCollectionModel.msl;
provider=System.Data.SqlClient;
provider connection string="
Data Source=$$YOURSERVERHERE$$;Encrypt=true;
Initial Catalog=$$YOURDBHERE$$;Integrated Security=false;
User ID=$$YOURUSERIDHERE$$;Password=$$YOURDBPASSWORDHERE$$;
multipleactiveresultsets=True;App=EntityFramework"
"providerName="System.Data.EntityClient" />
```

The following is an example of Windows authentication:

```
<add name="DataCollectionEntities"
connectionString="metadata=res://*/
DataCollectionModel.csdl|res:
//*/DataCollectionModel.ssdl|res://*/
DataCollectionModel.msl;
provider=System.Data.SqlClient;
provider connection string="Data
Source=$$YOURSERVERHERE$$; Encrypt=true;Initial
Catalog=$$YOURDBHERE$$;
Integrated Security=true;multipleactiveresultsets=True;
App=EntityFramework"
"providerName="System.Data.EntityClient" />
```

- Update the `dbConnectionString` connection string to add the attribute, 'Encrypt=true' while using SQL authentication or Windows authentication.

The following is an example of SQL authentication:

```
<add name="dbConnectionString"
connectionString="Data Source=$$YOURSERVERHERE$$;
Initial Catalog=$$YOURDBHERE$$;Integrated Security=false;
User ID=$$YOURUSERIDHERE$$; Encrypt=true;
Password=$$YOURDBPASSWORDHERE$$"
providerName="System.Data.SqlClient" />
```

The following is an example of Windows authentication:

```
<add name="dbConnectionString"
connectionString="Data Source=$$YOURSERVERHERE$$;
Initial Catalog=$$YOURDBHERE$$; Encrypt=true;
Integrated Security=true"
providerName="System.Data.SqlClient" />
```


Chapter 7: Multilanguage Support

This chapter describes how to use the Access Assurance Suite Multilanguage support option, and includes the following sections:

- [*“About Multilanguage Support” on page 82*](#)
- [*“Web Access \(ASP\) Option Multilanguage Mapping” on page 85*](#)
- [*“AuditLink Multilanguage Mapping” on page 94*](#)
- [*“Simplifying Third-Party Translation” on page 99*](#)

This chapter assumes that you have installed the Access Assurance Suite, then configured and tested all applications. As a best practice, make sure that your ASP pages and email/ticketing configurations operate properly before you begin. This allows you to pinpoint problems resulting from translation implementation.

About Multilanguage Support

The multilanguage support feature of the Access Assurance Suite provides support for languages other than American English, including French, German, UK English, and other languages.

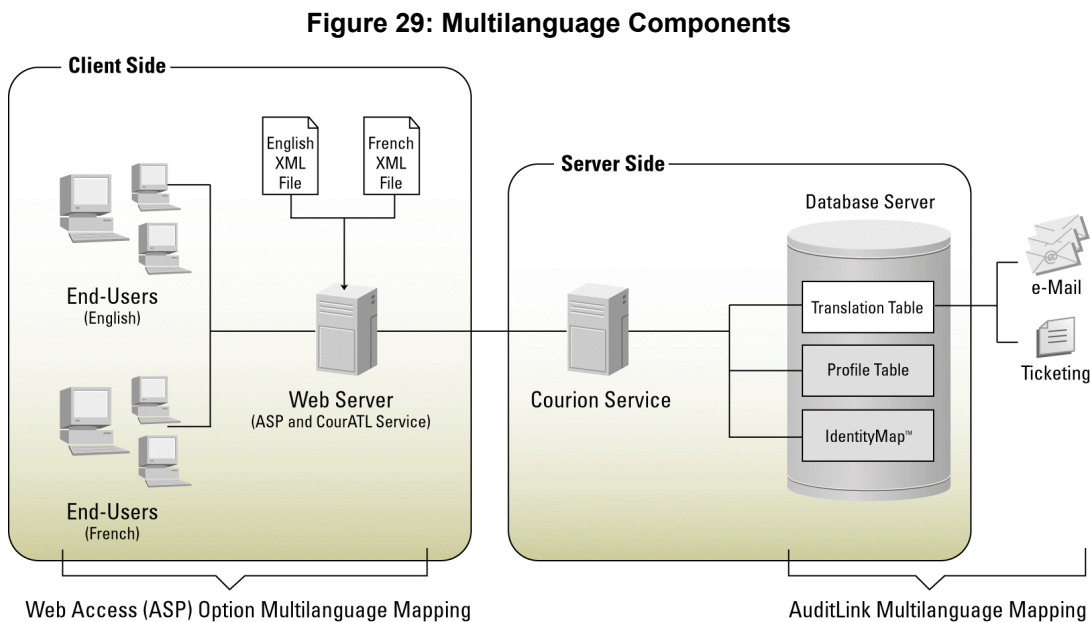
The multilanguage support feature has two parts:

- **Web Access (ASP) Option Multilanguage Mapping** (also referred to as, ASP Multilanguage Mapping), which lets end users of Core Security products such as PasswordCourier receive messages and prompts in other languages. ASP Multilanguage Mapping uses XML documents to provide mapping between languages.
- **AuditLink Multilanguage Mapping**, which lets an administrator—through a translation table he or she creates—configure email and tickets to appear in the language specified by the end user. AuditLink Multilanguage Mapping uses %Select% macros to perform the mapping.

You can use one or both mapping options. The ASP Multilanguage Mapping is explained in [“Web Access \(ASP\) Option Multilanguage Mapping” on page 85](#) and AuditLink Multilanguage Mapping in [“AuditLink Multilanguage Mapping” on page 94](#).

Dates in local formats are supported; however currency and sorting based on local settings are not supported.

[Figure 29](#) shows the multilanguage components in relation to the framework of Core Security's product suite.



The system default language is based on ASP pages. By default, these ASP pages assume the language code `en` (for English), send this code to the Courion Server, and read the default translation files (`pwc_resource_en.xml`, `pc_resource_en.xml`, and `string_resource_en.xml`) shipped with the product.

You can change the system default behavior of the ASP pages to assume a language other than `en` (English). To do this, open the `default.asp` page, copy it, and in the copied file, modify the line that sets `Session("lang") = "en"`. Then redirect the portal to point to the modified copy of `default.asp`.

To support a non-English language for all users, you can do the following:

You can change the system default behavior of the XML or ASP? pages to assume a language other than `en` (English). To do this, open the `String_resource_en.xml` page, copy it, and in the copied file, modify the line that sets `Session("lang") = "en"`. Then redirect the portal to point to the modified copy of `String_resource_en.xml`

Overriding the Default Language for any End User

You can override the system default language at the end user level by specifying the appropriate language in an end user's URL. For example, if a division of your company prefers French, you could use the following URL format in each person's URL to enable English-to-French translation for the Access Assurance Suite Portal products:

```
http://Core Security.web.server/virtualdir/websamples/accessoptions/html/PasswordCourier/default.asp?lang=fr
```

For the Access Assurance Suite Portal products, the `?lang=fr` is a URL parameter that assigns the language code `fr` to the variable `lang`, thereby overriding the system default.

For the Provisioning Platform products, the `&lang=fr` is a URL parameter that assigns the language code `fr` to the variable `lang`, thereby overriding the system default.

Overriding the Default Language and Culture for any End User

In addition to specifying a language at the end user level, you can also specify a culture on the user interface of the Access Assurance Portal. Attributes such as date format and currency symbols on the user interface change according to the selected culture. The language and culture are set using the following options:

- **Language Parameter:** You can specify the required language and culture in an end user's URL. For example, if a division of your company prefers French as well as the date format and currency symbol used in France, you could use the following URL format in each person's URL:

```
http://<Core Security ServerIPAddress>/CoreARMS/AspxCommon/Login.aspx?lang=fr-FR
```

Where 'fr' is the language and 'FR' is the culture according to which attributes like date format, currency symbols appear.

The resource file `string_resource_fr-FR.xml` must be copied to the following location:

```
\Courion Corporation\CourionARMS\Areas\AccessRequest\common
```

For more information, refer to the section, ["Sample Steps to Add Support for a Language and Culture for the Access Assurance Portal" on page 89](#).

- **Browser Language:** If no language and culture are specified in the end user's URL, the first language and culture specified in the browser languages list, which is displayed in the Language options of the browser settings is used.

- **Global Configuration:** When neither the language parameter at the end of the URL nor browser language are specified, the default culture and language specified in the UICulture global option is used. The default value specified in the UICulture option is 'en' which can be changed by the administrator using the Configure Global Options screen. Refer to the manual, *Configuring the AccountCourier Access Request Manager Solution* for more information on configuring the global options.

ISO 639-1 Code

Core Security supports only "ISO 639-1 Code" standards (two-letter) for the language parameter as described in the following webpage:

http://www.loc.gov/standards/iso639-2/php/code_list.php

Web Access (ASP) Option Multilanguage Mapping

Web Access (ASP) Multilanguage Mapping (for brevity, ASP Multilanguage Mapping) uses XML documents to provide mapping between languages. The language is determined by a language code passed as part of the URL. This language code determines which XML document the web server uses as a source for translation.

For example, when the Web Access software passes the value `fr` as the `lang` variable, the following occurs:

1. For PasswordCourier and PasswordCourier Support Staff, the ASP pages import the XML file `pwc_resource_fr.xml`. For ProfileCourier, the file is `pc_resource_fr.xml`. For AccountCourier and ComplianceCourier, the XML filename is `string_resource_fr.xml`
For the Access Assurance Suite, the ASP pages import all three XML files.
2. Web Access software uses the XML file to translate other internal ASP parameters along with any dynamic text sent to the ASP pages from the Courion Server.
3. The software sends the language code to the Courion Server using a `SetBehavior()` call which makes the language code available to the workflow through the macro `%Behavior.Language%`.

Sample Steps to Define ASP Multilanguage Mapping for English and French Languages

To create an Access Assurance Suite installation that supports English and French languages, follow these steps. For other languages, the steps are similar. For more information, refer to the [“Creating a Language-Specific XML File”](#) section.

1. Copy the `../HTML/AccountCourier/common/String_resource_en.xml` file and rename it `String_resource_fr.xml`.
2. Open the `String_resource_fr.xml` file and change the `id` attribute of the `<language>` tag from `en` to `fr`. That is, change

```
<language codepage="1252" id="en" lcid="1033"....  
to  
<language codepage="1252" id="fr" lcid="1036"....
```

3. Add the `<prompt>`, `<key>`, `<value>` tags as indicated in the following example:

```
<languages>  
  
<language codepage="1252" id="fr" lcid="1036">  
  
...  
  
...  
  
</prompts>
```

```

<prompt>

<key>STRING TO BE TRANSLATED</key>

<value>TRANSLATED STRING</value>

</prompt>

</prompts>

</language>

</languages>

```

4. Enter the exact string from the Core Security web pages you want translated in place of the "STRING TO BE TRANSLATED".

Note: The values that you add must be XML-compliant as described in section 2.4 of the following World Wide Web Consortium Recommendation:

<http://www.w3.org/TR/REC-xml/>

5. Repeat steps 3 and 4 for each string you want translated.
6. Optionally, open the `CommonStringsForLocalization.xml` file and copy the `<prompt>` tags of strings that you feel are important to translate. This file contains about 1,600 different strings that the Courier Server sends to the ASP pages for the provisioning products.
7. Save the `String_resource_fr.xml` file.
8. Copy the `../HTML/PasswordCourier/Pwc_resource_en.xml` file and rename the copy as `pwc_resource_fr.xml`.
9. Open the `pwc_resource_fr.xml` file for editing and change the `id` attribute of the `<language>` tag from `en` to `fr`. That is, change

```

<language codepage="1252" id="en".....
to
<language codepage="1252" id="fr"....

```

10. Add the `<prompt>`, `<key>`, `<value>` tags as indicated in the following example:

```

<languages>

<language codepage="1252" id="fr" lcid="1036">

...

...

</prompts>

<prompt>

<key>STRING TO BE TRANSLATED</key>

<value>TRANSLATED STRING</value>

```

```
</prompt>
```

```
</prompts>
```

```
</language>
```

```
</languages>
```

11. Enter the exact string from the Core Security web pages you want translated in place of the "STRING TO BE TRANSLATED".

Note: The values that you add must be XML-compliant as described in section 2.4 of the following World Wide Web Consortium Recommendation:

<http://www.w3.org/TR/REC-xml/>

12. Save the `pwc_resource_fr.xml` file and exit the editor.
13. Copy the `../HTML/ProfileCourier/Pc_resource_en.xml` file and rename the copy as `Pc_resource_fr.xml`
14. In an editor, open the `Pc_resource_fr.xml` file and change the `id` attribute of the `<language>` tag from `en` to `fr`. That is, change

```
<language codepage="1252" id="en".....
to
<language codepage="1252" id="fr"....
```

15. Add the `<prompt>`, `<key>`, `<value>` tags as indicated in the following example:

```
<languages>

<language codepage="1252" id="fr" lcid="1036">

...

...

<prompts>

<prompt>

<key>STRING TO BE TRANSLATED</key>

<value>TRANSLATED STRING</value>

</prompt>

</prompts>

</language>

</languages>
```

16. Enter the exact string from the Core Security web pages you want translated in place of the "STRING TO BE TRANSLATED".

Note: The values that you add must be XML-compliant as described in section 2.4 of the following World Wide Web Consortium Recommendation:

<http://www.w3.org/TR/REC-xml/>

17. Save the `pc_resource_fr.xml` file and exit the editor.
18. If you have a third-party ready to perform the translations, send the `String_resource_fr.xml` file, `pwc_resource_fr.xml` file and the `pc_resource_fr.xml` file to be translated. If you use both the Web Access and AuditLink Multilanguage Mapping and you have the translation done by a third party, you might want to combine the translation table and XML files temporarily. If so, go to ["Simplifying Third-Party Translation" on page 99](#). Otherwise, continue here.
19. After the `String_resource_fr.xml` file contains the translated strings, copy the file to the following locations:

`\Courion Corporation\www\WebSamples\AccessOptions\HTML\AccountCourier\common`

`\Courion Corporation\www\AccountCourier\common`

When the `pwc_resource_fr.xml` file contains the translated strings, use it to replace the following file:

`../HTML/PasswordCourier/Pwc_resource_fr.xml`

When the `Pc_resource_fr.xml` file contains the translated strings, use it to replace the following file:

`../HTML/ProfileCourier/Pc_resource_fr.xml` file

This ensures that the translations are available to AAS and end user workflows.

20. Restart the IIS Web server in one of the following ways:
- Run `iisreset` from the command prompt
 - Select Restart using the IIS Manager Console.
21. Inform your end users that their browser language setting should include the language(s) you have made available. For more information about browser language settings, refer to the following World Wide Web Consortium webpage:
- <http://www.w3.org/International/questions/qa-lang-priorities.en.php>

The valid end user workflow URL for the French language should look similar to the following:

<http://<Core Security ServerIPaddress>/Core/WebSamples/AccessOptions/HTML/AccountCourier/default.asp? Workflow=WorkflowName?lang=fr>

Sample Steps to Add Support for a Language and Culture for the Access Assurance Portal

Following is an example to add support for the French language and culture for the Access Assurance Portal. For other languages and culture, the steps are similar.

1. Copy the `../HTML/AccountCourier/common/String_resource_en.xml` file and rename it to `String_resource_fr-FR.xml`.
2. Open the `String_resource_fr-FR.xml` file and change the `id` attribute of the `<language>` tag from `en` to `fr-FR`. That is, change

```
<language codepage="1252" id="en" lcid="1033"
```

to

```
<language codepage="1252" id="fr-FR" lcid="1036"
```

Refer to the following URLs for more information on the language and culture LCID codes:

[https://msdn.microsoft.com/en-us/library/ms533052\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms533052(v=vs.85).aspx)

<https://msdn.microsoft.com/en-us/global/bb964664.aspx>

3. Add the `<prompt>`, `<key>`, and `<value>` tags as indicated in the following example:

```
<languages>
```

```
<language codepage="1252" id="fr-FR" lcid="1036">
```

```
...
```

```
...
```

```
<prompts>
```

```
<prompt>
```

```
<key>STRING TO BE TRANSLATED</key>
```

```
<value>TRANSLATED STRING</value>
```

```
</prompt>
```

```
</prompts>
```

```
</language>
```

```
</languages>
```

4. Enter the exact string from the Core Security web pages you want translated in place of the "STRING TO BE TRANSLATED".

Note: The values that you add must be XML-compliant as described in section 2.4 of the following World Wide Web Consortium Recommendation:

<http://www.w3.org/TR/REC-xml/>

5. Repeat steps 3 and 4 for each string you want translated.
6. Optionally, open the `CommonStringsForLocalization.xml` file and copy the `<prompt>` tags of strings that you feel are important to translate. This file contains about 1,600 different strings that the Courion Server sends to the ASP pages for the provisioning products.
7. Save the `String_resource_fr-FR.xml` file.
8. After the `String_resource_fr-FR.xml` file contains the translated strings, copy the file to the following location:
`\Courion Corporation\CourionARMS\Areas\AccessRequest\common`
9. Restart the IIS Web server in one of the following ways:
 - Run `iisreset` from the command prompt
 - Select Restart using the IIS Manager Console

The Access Assurance product end user link for French users should be similar to the following:

<http://<Core Security ServerIPaddress>/CoreARMS/AspxCommon/Login.aspx?lang=fr-FR>

Creating a Language-Specific XML File

The following section provides a closer look at the contents of the multilanguage support related-XML files. Each XML file contains a root node called `<languages>` followed by a child node defining the language being translated called `<language>` that defines the output language. The `<language>` tag contains three attributes:

- The `codepage` and `lcid` are values given to each language as defined by organizations like Unicode (www.unicode.org) and ANSI.
- The `id` is the language code that is used to link the `?lang=` URL parameter to the proper `<language>` tag.

When the Courion Server passes a `?lang=` code, for example, `?lang=fr` to indicate French, the Web Access software uses the language code `fr` to find the appropriate XML document and the appropriate `<language>` tag within that XML document. An example of this follows from file `pwc_resource_fr.xml`.

```
<languages>
<language codepage="1252" id="fr" lcid="1036">
.
</language>
</languages>
```

Within the `<language>` tag, there are two different types of subtags: `<param>` and `<prompts>`. Subtag `<param>` represents mapping of strings for internal ASP variables used by the Core Security application. The `<param>` tags define the look and feel of the pertinent application; you need to modify them only when you want to change that look and feel.

Each `<param>` tag includes an `id` attribute that uniquely identifies it (such as `<param id="MyASPVariable">`). Best practice suggests you assign this attribute the same name as the variable that it represents. (However, the software does allow more than one ASP variable to reference the same `<param>` tag if the need arises.) Later, the ASP variable can then employ the function call `GetXMLParamValue("MyASPVariable")` to initialize the variable based on the value placed in the XML file. The `GetXMLParam` function call is shown later on page 92.

Subtag `<prompts>` encapsulates individual `<prompt>` tags, which map strings to dynamic text received by the Courion Server. Tag `<prompt>` contains two subtags: `<key>` and `<value>`. Subtag `<key>` provides the string as received from the Courion Server; subtag `<value>` provides the localized version.

For example, assume that the ASP receives the following string from the Courion Server: `I love translating`. For a French translation, you would construct the `<prompt>` tag as follows:

```
<prompt>
<key> I love translating</key>
<value> J'aime la traduction</value>
</prompt>
```

Then, when the key string `I love translating` comes from the Courion Server, the end user would see the value string `J'aime la traduction`.

Note: For Provisioning Platform products, you may need to modify the `<param>` subtag if translation fails with the `<prompt>` subtag.

For example, you use the `<prompt>` subtag to translate the text on the Next button to French:

```
<prompt>
<key>Next</key>
<value>Suivant</value>
</prompt>
```

However, if the translated text does not appear on the Next button, Core Security recommends that you modify the `<param>` subtag as shown:

```
<param id="NEXT_BUTTON_CAPTION"> Next </param>
to
<param id="NEXT_BUTTON_CAPTION"> Suivant </param>
```

Adding Static Text to an End User Page

Static text is text that you do not enter using the Administration or Configuration Manager but that you want to display from a specific end user page.

To add static text to an end user page, use an ASP variable to represent the text so that it can be translated for different localizations. The steps for doing this are as follows:

1. Give the ASP variable a name and add it to the `../HTML/PasswordCourier/Pwc_resource_en.xml` file as a `<param>` tag, where the `id` attribute represents the variable name and the contents of the `<param>` tag represent the static string. For example,

```
<param id="Welcome_page">Welcome to the XYZ Corporation
PasswordCourier implementation</param>.
```
2. Open the file in which you want the static text to appear and insert a call to the ASP function `GetXMLParamValue("[ASP-Variable]")` where you want the static text to be shown. For example,

```
<p><%=GetXMLParamValue("Welcome_page")%></p>
```

If after adding text you receive the error message "ERROR - XML file does not contain following parameter:...", this means the ASP page called the `GetXMLParamValue([ASP-Variable])` but the `[ASP-Variable]` could not be found in the XML file as a `<param>` tag. Add a `<param>` tag where the `id` attribute equals the ASP variable name to resolve this error.

Translating Dynamic Strings

In some strings, part of the string changes frequently. Such a changeable string is called a dynamic string.

In a dynamic string, the software tries to find a substring that matches a `<prompt>` tag in the XML file. If the software finds a `<prompt>` that matches a substring, it replaces that substring with the contents of the `<value>` tag and then processes the remainder of the string. It continues until it has tried all possible combinations. It shows any substring it cannot match with a `<prompt>` in the native language as received by the Courion Server.

For best matching, place `<prompt>` tags with larger `<key>` strings before `<prompt>` tags with smaller `<key>` strings. Also, if the translation uses a single variation on the dynamic string more frequently than others, you should add that entire string variation to a `<prompt>` tag so translation occurs only with less common variations.

You may find that an ASP page takes many seconds to display and perhaps also that the Internet Information Service is consuming all CPU time. If so, you may be trying to translate many large dynamic strings (perhaps with thousands of words) using an XML file that also contains thousands of `<prompt>` entries. Pattern matching is an expensive operation that requires many CPU cycles.

One way to improve matching is to isolate the most frequent variations of the large strings received from the Courion Server and add each variation (in its entirety) to the XML file. Place the `<prompt>` tags ahead of other smaller `<prompt>` tags so the software can find the strings faster.

Note: If you do not need to translate dynamic strings returned from the Courion Server, you can turn off the multilanguage support. To do this, open the `default.asp` page, copy it to a new file, and then in your copy change one of the following Sessions from `True` to `False`, depending on whether you are using PasswordCourier (PWC), PasswordCourier Support Staff (PWCSS), ProfileCourier (PC), or AccountCourier (AC).

```
Session ("PWC_SEARCH_SUBSTRING_FOR_XML_REPLACEMENT")
Session ("PC_SEARCH_SUBSTRING_FOR_XML_REPLACEMENT")
Session ("AC_SEARCH_SUBSTRING_FOR_XML_REPLACEMENT")
```

For example:

```
Session("PWC_SEARCH_SUBSTRING_FOR_XML_REPLACEMENT") =  
False
```

Then save changes and redirect the portal to point to your copied `default.asp` file.

AuditLink Multilanguage Mapping

AuditLink Multilanguage Mapping software resides on the database server and uses `%Select...%` macros to perform the language mapping capability against a data source. [Figure 29 on page 82](#) shows the multilanguage components in relation to the framework of Core Security's product suite.

For AuditLink Multilanguage Mapping, the language is determined by the end-user client passing a language code in the portal link. This language code determines strings used from the translation table.

Using the `%Behavior.Language%` Macro to Populate Email and Ticketing Events

The statement `?lang=` is a URL parameter—which you can use in an end user's initial ASP page or calling URL—that assigns the language code to the variable `lang`. The value passes to the Courion Server that, in turn, can reference the value through a `%Behavior.Language%` macro. The `%Behavior.Language%` macro takes effect before the first page is displayed to the user. Therefore, you can call this macro in any field label, help text, value, or message that accepts macros.

You can use the `%Behavior.Language%` macro in conjunction with `%Select...%` macros to provide language-specific text for retrieval by an AuditLink Multilanguage Mapping data source, such as ticketing and email notification. In this way you can configure an email body (or ticketing field) to show a language-specific string based on the language code passed by ASP Multilanguage Mapping.

To do this, build a translation table in the data source and then query that table using `%Select...%` macros. The strict format enforced on `%Select...%` macros lets you query the translation table easily, by retrieving a value based on a key where the key is the concatenation of the language code and string being searched for; for example, `frStartActionEmailBody`.

In this section we use a code value of `fr` for `?lang=` and `%Behavior.Language%`. The string `fr` is a fixed code for the French language; use it consistently in `lang=`, the `%Behavior.Language%` macro, and the translation table.

Table 2 shows a typical translation table in a database.

Table 2: Sample Translation Table

UID (auto-sequencing)	Key	Value
0	<code>frStartActionEmailBody</code>	Ceci est mon corps anglais d'e-mail pour l'événement d'action de début.
1	<code>enStartActionEmailBody</code>	This is my English email body for the start action event.
2	<code>frSuccessActionEmailBody</code>	Et ceci le corps anglais d'e-mail pour un événement d'action de succès.
3	<code>enSuccessActionEmailBody</code>	And this is the English email body for a success action event.

After you construct the translation table, it is simple to build the `%Select...%` macro to pull the proper field based on the language code passed in by the user through the `%Behavior.Language%` macro.

Here is an example of the `Select` statement that you can use in the Start Action email body field:

- `%Select Value,TranslationTable,Key,|Behavior.Language|StartActionEmailBody%`

Resolved SQL Syntax if `%Behavior.Language%` macro = `fr`:

```
SELECT Value FROM TranslationTable WHERE Key =
"frStartActionEmailBody"
```

For PasswordCourier Classic, PasswordCourier Support Staff Classic, and ProfileCourier Classic, we suggest that you use the customization manager GUI screens to define `SELECT` statements to access the translation table.

In the Administration Manager, you need to enter the `SELECT` statements manually, or create a custom macro that accesses the translation table.

[Figure 30](#) and [Figure 31](#) show the customization manager screens that define the Select Start Action statement for email, before and after the administrator has pressed the Apply button.

The figures use the values `StartActiononEmailBody` and `SuccessActionTicketStatusField`. These values can be any string as long as the same string appears in the `%SELECT...%` macro to associate the text value with the field.

Figure 30: Define a Macro - Email (step 1)

Make Selection for 'Msg:'

Define field input information

☐ Select a Macro
☒ Select from a Table
☐ Enter a Text Value

Select field from table...

Value/TranslationTable

Where this field...

Key

☐ equals Macro selected below
☒ equals value entered below

Behavior.Language|StartActiononEmailBody

Apply

Your Selection

Clear Selection

OK Cancel

Java Applet Window

Figure 31: Define a Macro- Email (step 2)

Make Selection for 'Msg:'

Define field input information

☐ Select a Macro
☒ Select from a Table
☐ Enter a Text Value

Select field from table...

Value/TranslationTable

Where this field...

☒ equals Macro selected below
☐ equals value entered below

Apply

Your Selection

*SELECT,Value,TranslationTable,Key,
|Behavior.Language|StartActionEmail
Body*

Clear Selection

OK Cancel

Java Applet Window

[Figure 32](#) and [Figure 33](#) show the Administration Manager screens that define the Select Success Action statement for ticketing.

Figure 32: Define a Macro - Ticketing (step 1)

Make Selection for 'Msg:'

Define field input information

- ☐ Select a Macro
- ☒ Select from a Table
- ☐ Enter a Text Value

Select field from table...

Value/TranslationTable

Where this field...

Key

☐ equals Macro selected below

☒ equals value entered below

Behavior.Language|SelectActionTicketStatusF

Apply

Your Selection

Clear Selection

OK Cancel

Java Applet Window

Figure 33: Define a Macro - Ticketing (step 2)

Make Selection for 'Msg:'

Define field input information

- ☐ Select a Macro
- ☒ Select from a Table
- ☐ Enter a Text Value

Select field from table...

Value/TranslationTable

Where this field...

☐ equals Macro selected below

☐ equals value entered below

Apply

Your Selection

Select, Value, TranslationTable, Key, Behavior.Language|SuccessActionTicketStatusField

Clear Selection

OK Cancel

Java Applet Window

Accessing the Translation Table with AccountCourier or ComplianceCourier

For AccountCourier or ComplianceCourier, you need to enter the SELECT statements manually, or create a custom macro that accesses the translation table. The following is a sample custom macro for use with ODBC:

```
SELECT Value FROM TransTable WHERE  
key="%Behavior.Language%ActionStartTicket"
```

Place the custom macro where you want the translation to occur (for example, in a ticketing field).

Sample Steps to Define AuditLink Multilanguage Mapping for PasswordCourier or ProfileCourier

For PasswordCourier, PasswordCourier Support Staff, or ProfileCourier, follow these steps. You must perform them for each application for which you want to use language translation.

1. Configure PasswordCourier and PasswordCourier Support Staff (including Request Tracking) and ProfileCourier so that each application works with ASP Multilanguage Mapping and messaging is correctly defined.
2. Create a translation table in the authentication data source for request tracking. The table should include messaging for email and ticketing. For the table format, refer to the sample translation table on page 94.
3. Add English entries to the translation table that represents each field that is configured in the Request Tracking tab of the application.
4. Configure ticketing and notifications within the application by using `Select...%` macros in combination with the `%Behavior.Language%` macro. Refer to the section discussing `%Behavior.Language%` for details on building these SELECT macros.
5. Have the pertinent English translation table entries translated to French. If you use both the Web Access and AuditLink Multilanguage Mapping and you have the translation done by a third party, you might want to combine the translation table and XML files temporarily so the translator needs to deal with only one file. If so, go to ["Simplifying Third-Party Translation" on page 99](#). Otherwise, continue here.
6. Test your workflow to make sure that the email and ticketing are getting the strings from the SELECT macros.
7. Repeat the above steps in the other languages to ensure that the Audit links are localizing correctly.

Simplifying Third-Party Translation

If you use both the Web Access and AuditLink Multilanguage Mapping, then you need a translation of both XML file and translation table entries.

If you are having the translation done by a third party, you can save time and expense by adding the translation table to the XML file entries temporarily so the translator has to work with only one file for your entire site.

To add the table and key entries, open the language XML files in a text editor and follow these steps.

1. Add the following XML tags right before the ending `</language>` tag. Here is a sample format:

```
<language id="fr" lcid="9228" codepage="1252">
<email>
    <key>StartActionSubject</key>
    <value>This is my start action email subject for
translation</value>
</email>
<email>
    <key>StartActionBody</key>
    <value> This is my start action email body for
translation</value>
</email>
<ticketing>
    <key>SuccessActionTicketStatus</key>
    <value> This is the ticket status for translation</value>
</ticketing>:
</language>
```

2. Instruct the translator to translate only the contents of the `<param>` tags and the contents of the `<value>` tags in the XML document. The contents of the `<key>` tag should NOT be modified and neither should any of the XML attributes such as `id` or `codepage`. Doing so prevents the application from running properly.
3. When you receive the finished XML document from the translator, copy it into the same directory as the original English language resource file.
4. Add entries to the translation table in the database such that the field representing the key shows the contents of the `<key>` tag preceded by the language code. For example, `frStartActionEmailBody` or `deStartActionEmailBody` and in which the value field represents the contents of the `<value>` tag in the XML file.
5. Run the application and the pertinent other product runtimes and make sure that everything works as expected in an English environment. That is, both the ASP pages show the right messaging and the email/tickets have the fields properly filled through `%Select...%` macros.

Chapter 8: Enabling Multi-Byte Connectors and PMMs

To enable a connector or PMM to receive multi-byte data from the Access Assurance Suite, you first need to enable it for multi-byte usage and then label it as multi-byte ready, unless it has already been enabled for use with multi-byte languages. See [“Connectors and PMMs that have been Enabled for Use with Multi-Byte Languages:” on page 101](#) for a list of these connectors and PMMs.

To do this, edit the following file:

CourionService\cnctr_targets_constraints_default.xml file.

This chapter describes how to edit this file to enable a connector or PMM to accept multi-byte data in the following sections:

- [“Overview” on page 102](#)
- [“Enabling Multi-Byte Connectors” on page 103](#)
- [“Enabling Multi-Byte PMMs” on page 104](#)

Connectors and PMMs that have been Enabled for Use with Multi-Byte Languages:

The following connectors and PMMs have been enabled by Core Security for use with multi-byte languages:

- The Connector for Microsoft ADO – Authentication, query, ticketing, notifications, triggers, profile update
- The Connector for Microsoft Active Directory – Authentication, Query, Provisioning Actions
- The Enhanced Connector for HTML Email – Notifications
- The Connector for Core Security VBScript Custom Macros – Query
- The Connector for Microsoft Exchange 2003 – Provisioning Actions
- The Connector for Microsoft Exchange 2007 – Provisioning Actions
- The PMM for Microsoft Active Directory - Password Reset

Overview

At the end of the `CourionService\cnctr_targets_constraints_default.xml` file there is an XML node labeled `utf8Capable`. By adding children to this node, you can explicitly enable or disable the ability for a connector or a PMM to receive multi-byte data through the Connector Framework Manager.

Each XML node consists of a name attribute and a ready attribute. The name is the connector's vendor triplet, or the PMM's name attribute. Ready is either "yes" or "no." "No" is the default. The following example shows the Microsoft ADO connector as enabled for multi-byte data:

```
<utf8Capable default="no">
  <connector name="Microsoft-ADO-3.0" ready="yes" />
</utf8Capable>
```

If ready is set to "no" the CFM converts any Unicode data to the default codepage of the server.

Enabling Multi-Byte Connectors

To enable a connector to receive multi-byte data, add a connector node to the `utf8Capable` block in the `cnctr_target_constraints_default.xml` file with the name of the connector's vendor triplet.

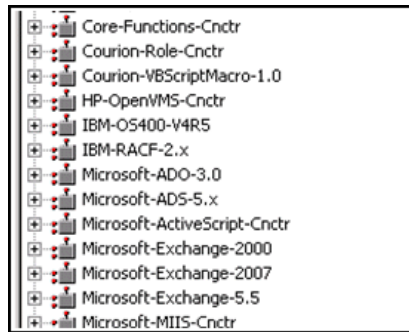
Standard Connectors

For standard connectors, you can obtain the vendor triplet through the Connector Configuration Manager. From the Start menu, select:

Start>Courion Access Assurance Suite>Connector Configuration Manager

Figure [Figure 34](#) shows a list of vendor triplets displayed by the Connector Configuration Manager.

Figure 34: Connector Configuration Manager



If you cannot obtain the vendor triplet from the Connector Configuration Manager, contact Core Security Customer Support.

RDK Connectors

For RDK connectors, you can find this information under the `Cnctr.xml` file in the `CourionService` directory, as shown in [Figure 35](#).

Figure 35: Cnctr.xml File for an RDK Connector

```
<VpvtTriplet vendor="Courion" product="HTML Email" version="Cnctr2" />
```

The vendor triplet for the connector is:

Courion Corporation-HTML Email-Cnctr2

To enable this connector to receive multi-byte data, update the `utf8Capable` block as follows:

```
<utf8Capable default="no">
  <connector name="Courion Corporation-HTML Email-Cnctr2" ready="yes" /
  >
</utf8Capable>
```

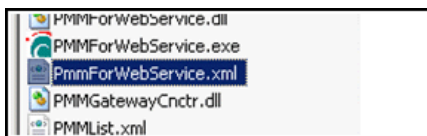
Enabling Multi-Byte PMMs

To enable a PMM to receive multi-byte data, add a connector node to the utf8Capable block in the cnctr_target_constraints_default.xml file with a name attribute of the PMMs name.

Branded PMMs

You can find the name attribute for the name of a branded PMM in the PMM branding XML file. These files exist in the Core Security Service directory. Figure [Figure 36](#) shows the PMM branding XML file for the PMM for Web Services, PmmforWebService.xml:

Figure 36: PMM Branding XML File for the PMM for Web Services



The PmmforWebService.xml file includes the name attribute, as shown in [Figure 37](#):

Figure 37: Name Attribute

```
<?xml version="1.0"?>
<ModuleSettings>
  <Name>Web Services</Name>
</ModuleSettings>
```

The name attribute in this example is Web Services. To enable this PMM to receive multi-byte data enter the following into the cnctr_target_constraints_override.xml file:

```
<utf8Capable default="no">
  <connector name=" Web Services" ready="yes" />
</utf8Capable>
```

Branded RDK PMMs

Branded RDK PMMs are created with the Enterprise Provisioning Suite Rapid Development Kit (RDK). These PMMs have a script-based file, typically JavaScript, and supporting documentation about configuring the PreCheck, PasswordRestEx, and ConfigValidation command lines. Because of this, branded RDK PMMs require an additional configuration step to enable them to receive multi-byte data than standard, branded PMMs.

Follow these steps:

1. Open the branding XML file in the CourionService directory, as shown in [Figure 38](#):

Figure 38: Branding XML File for a Branded RDK PMM

The CourionSamplePMM.xml file includes the name attribute, as shown in [Figure 39](#):

Figure 39: Name Attribute

```
<?xml version="1.0"?>
<ModuleSettings>
  <Name>Courion Sample PMM</Name>
</ModuleSettings>
```

2. Add a new child node after the <Name> tag in the .XML file:

```
<Unicode>yes</Unicode>
```

[Figure 40](#) shows the updated file:

Figure 40: Updated Branded XML File for a Branded RDK PMM

```
<?xml version="1.0"?>
<ModuleSettings>
  <Name>Courion Sample PMM</Name>
  <Unicode>yes</Unicode>
</ModuleSettings>
```

3. Edit the Add/Edit Target Configuration dialog box in the Connector Configuration Manager when you configure the target PMM with command lines similar to the following:
cscript.exe /nologo /b Courion SamplePMM.js
Add a /u option to the command line for cscript. The result appears as:
cscript.exe /u /nologo /b Courion SamplePMM.js
The /u option notifies the Windows Scripting Host to expect and return Unicode on the command line interface.
See the manual *Configuring Password Management Modules (PMMs), Connectors, and Agents* for more information about the Connector Configuration Manager.
4. Follow the instructions in [“Branded PMMs” on page 104](#) to add the PMM name attribute to the cnctr_target_constraints_default.xml file.

Unbranded RDK PMMs

To enable a PMM for multi-byte data it must be branded. To brand a PMM, make an XML file with the same name as the script file. For example, with a PMM script file of UnBrandedPMM.js, create an XML file with the name UnBrandedPMM.xml. Open the new XML file and enter the following:

```
<?xml version="1.0"?>
<ModuleSettings>
  <Name>UnBranded PMM</Name>
  <Unicode>yes</Unicode>
```

```
</ModuleSettings>
```

Substitute the name "UnBranded PMM" with an appropriate branding for this PMM. Additionally, when you configure the target for this newly branded PMM, include the /u option when appropriate.

Chapter 9: Configuring the XML Access Option

This chapter explains how to use the XML Access Option and includes the following sections:

- [*“About the XML Access Option” on page 108*](#)
- [*“SPML Schema Support” on page 109*](#)
- [*“Creating an SPML Template Document” on page 114*](#)
- [*“Configuring Options in the Administration Manager for Automating a Workflow” on page 120*](#)
- [*“Testing the SPML Code” on page 124*](#)
- [*“Configuring ASP Pages to Send SPML to the CourionService” on page 125*](#)

About the XML Access Option

The XML Access Option (XMLAO) lets you automate a workflow, eliminating end-user access with the workflow. The XMLAO uses SPML (Service Provisioning Markup Language) documents to provide input to the workflow that would otherwise be provided by the end user. The XML Access Option can simplify implementations that require input to provisioning actions from an external application to provision individual resources.

For example, a company hires a new employee and the Human Resources department enters information about the employee into a PeopleSoft® HR system. This triggers the PeopleSoft HR system to generate XML code with information about the new employee. You request an Add action for the new employee, and the request converts the XML code into an SPML document. The XML Access Option delivers the SPML document to AccountCourier using CourATLService, and AccountCourier performs the Add action. In this way the new employee gets an account without providing information to AccountCourier.

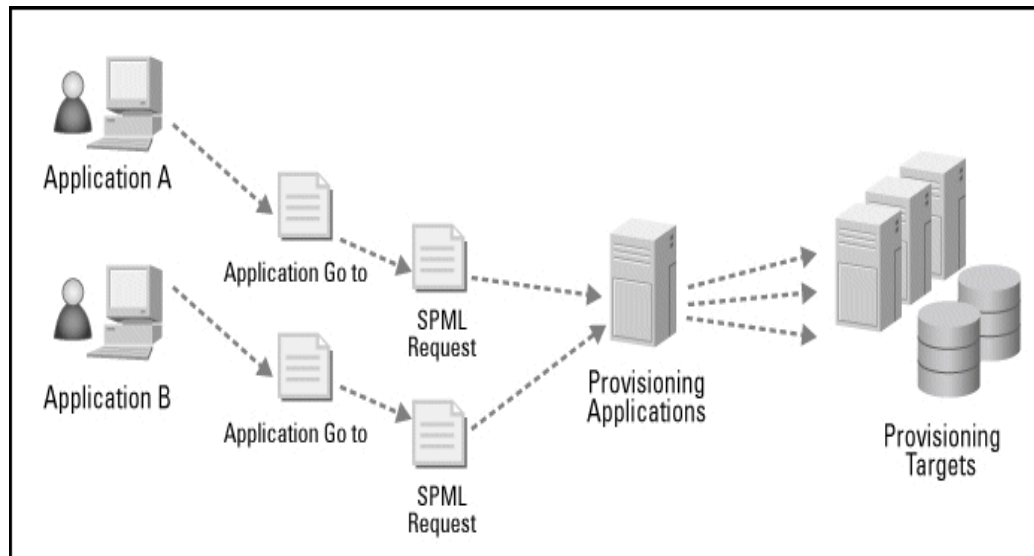
The XML Access Option can automate the following provisioning and compliance actions:

- Add
- Change
- Create
- Delete
- Disable
- Enable
- Verify (view only)
- View
- Password Reset

Note: Automating the Verify action through the XML Access Option allows you to view the status of account data on the Resource Data Selection form, but not compare values.

Each SPML document supports one workflow and one action within the workflow.

[Figure 41](#) represents the flow of information from an SPML document to the provisioning applications. The SPML data becomes available for use in macros that you use to retrieve values that automate the workflow.

Figure 41: Information Flow with the XML Access Option

The XML Access Option receives an SPML-encoded request and submits it to the workflow for processing.

SPML Schema Support

The Access Assurance Suite supports the following SPML 1.0 and 2.0 requests:

- addRequest
- modifyRequest
- deleteRequest
- extendedRequest

These SPML requests map to the provisioning actions shown in Table 3.

Table 3: SPML Standards and Provisioning Actions

SPML Standard	Provisioning Actions
addRequest	Create, Add
modifyRequest	Change, Disable, Enable, Password Reset
deleteRequest	Delete, Disable
extendedRequest	View, Verify

The following diagrams represent schemas for the SPML Protocol standards for addRequest, modifyRequest, deleteRequest, and extendedRequest. The elements and attributes with a check mark in these diagrams represent the components of the schema that the Access Assurance Suite supports.

Figure 42: addRequest

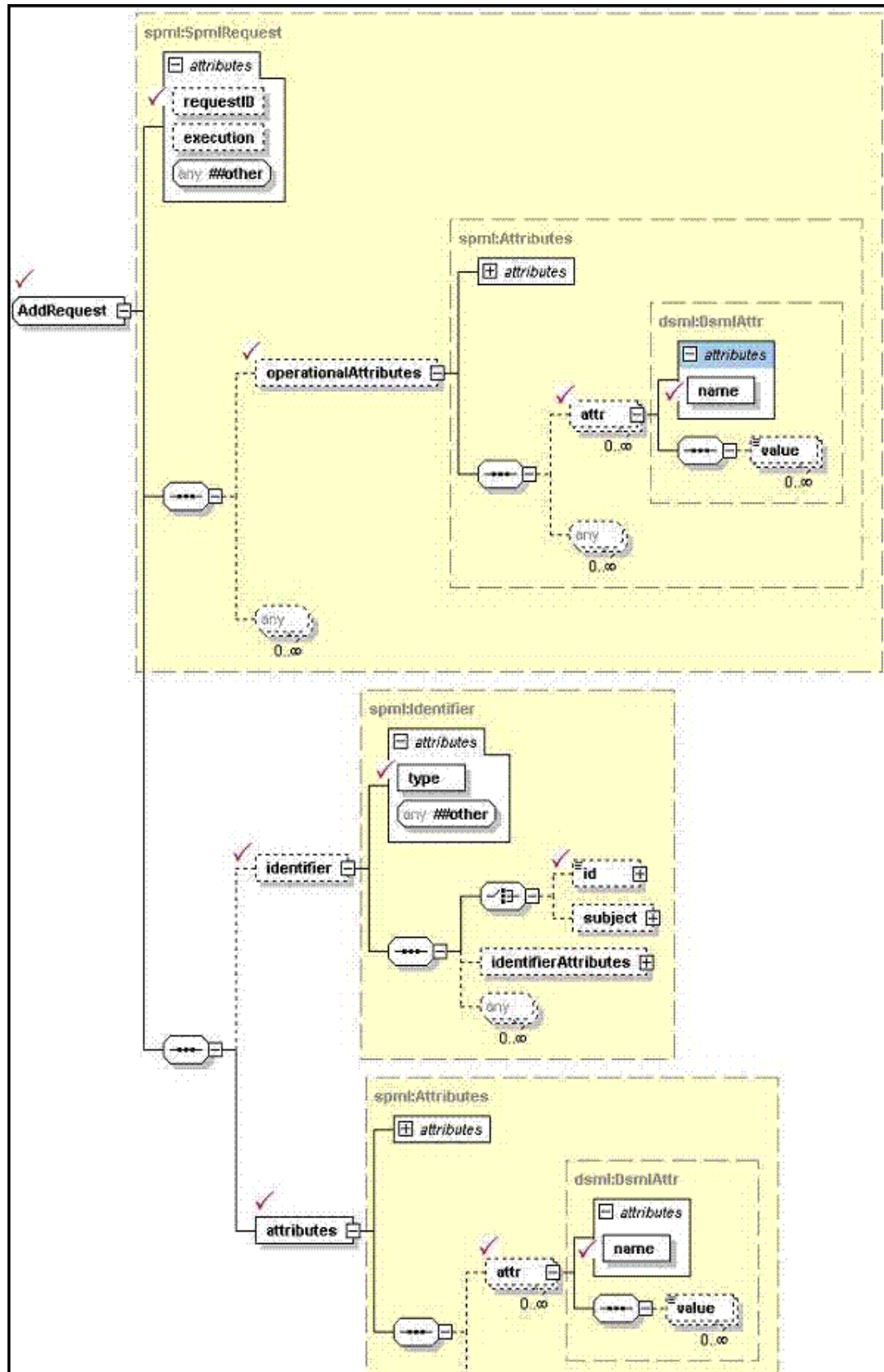


Figure 43: modifyRequest

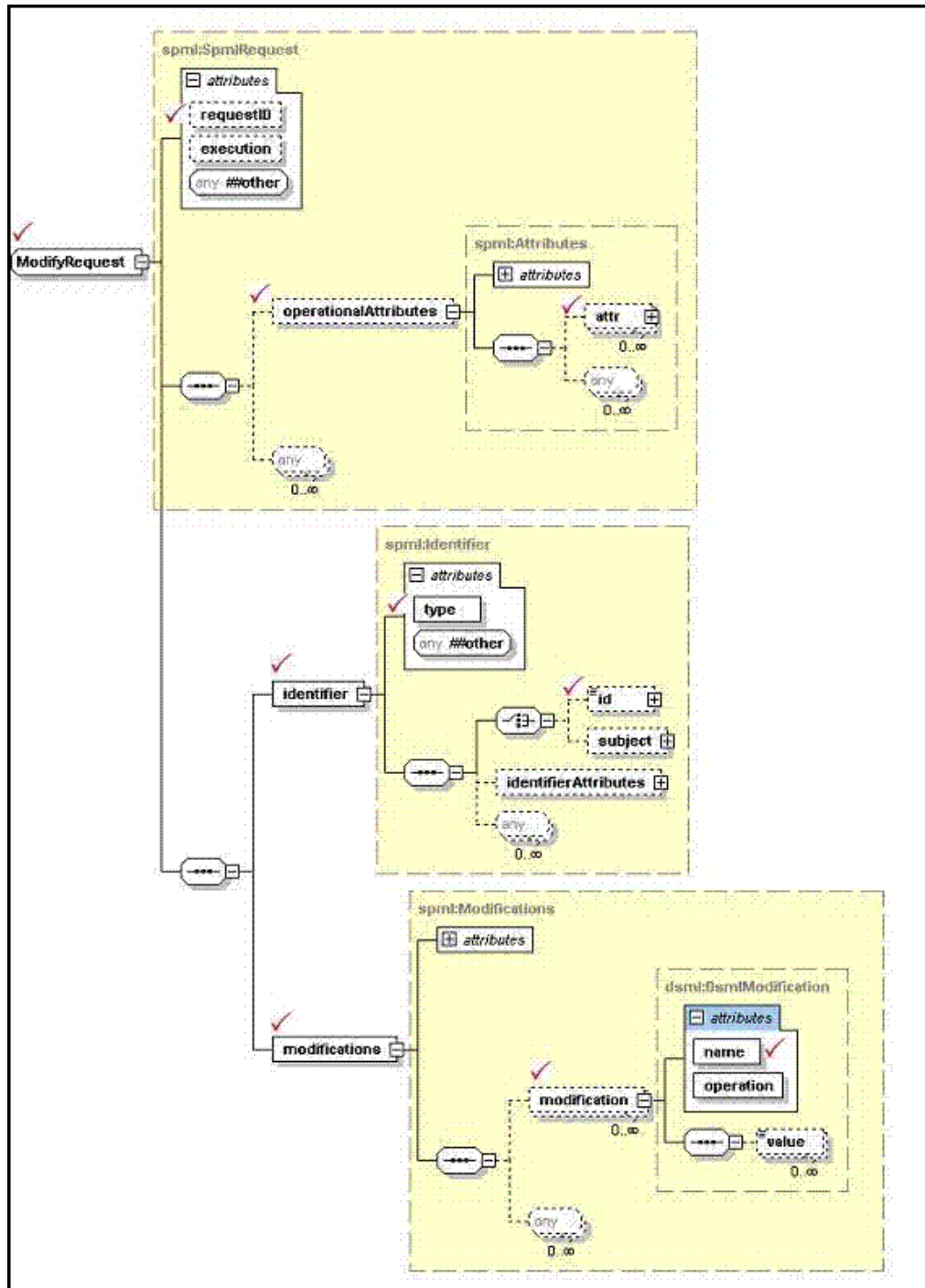


Figure 44: deleteRequest

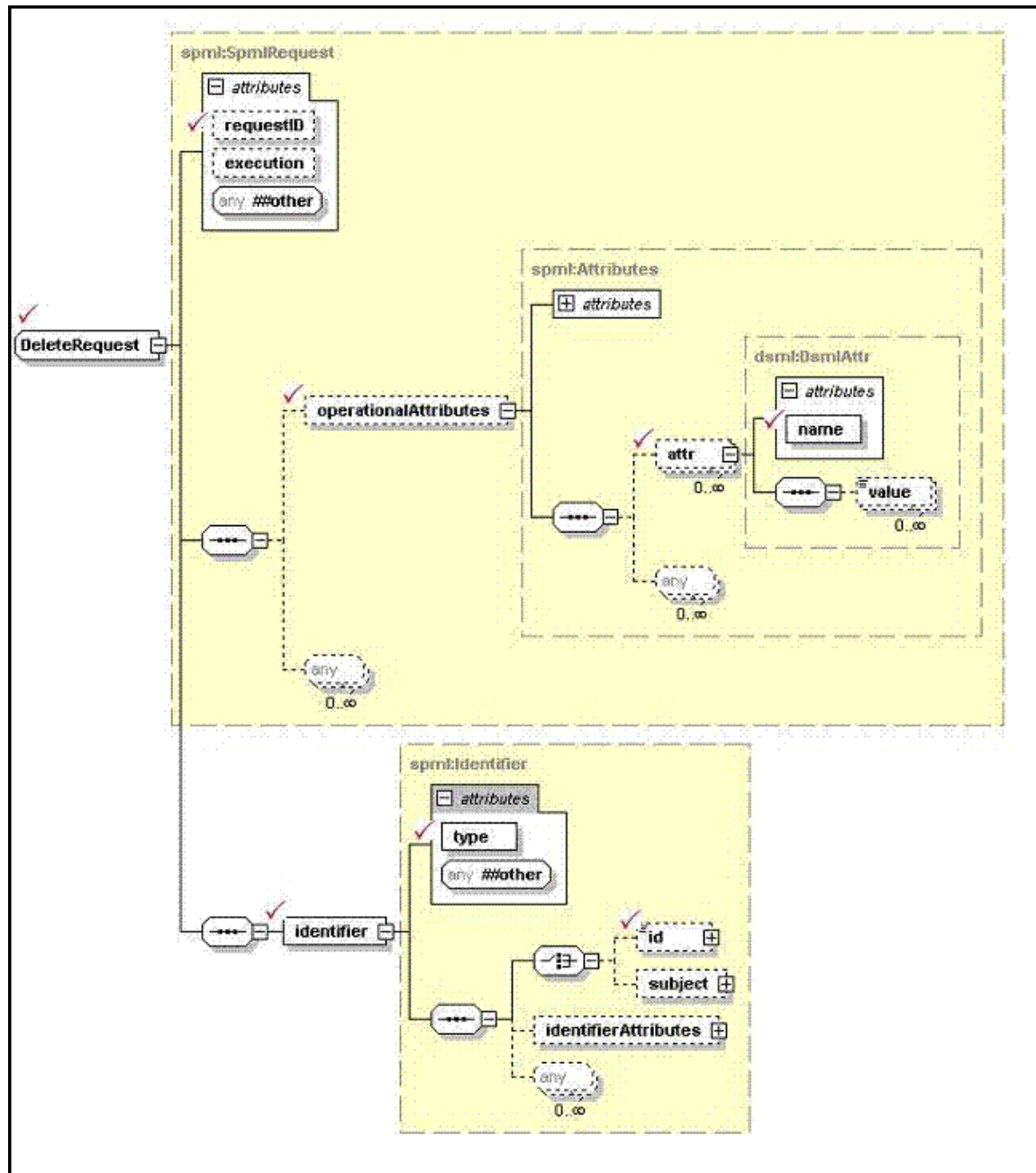
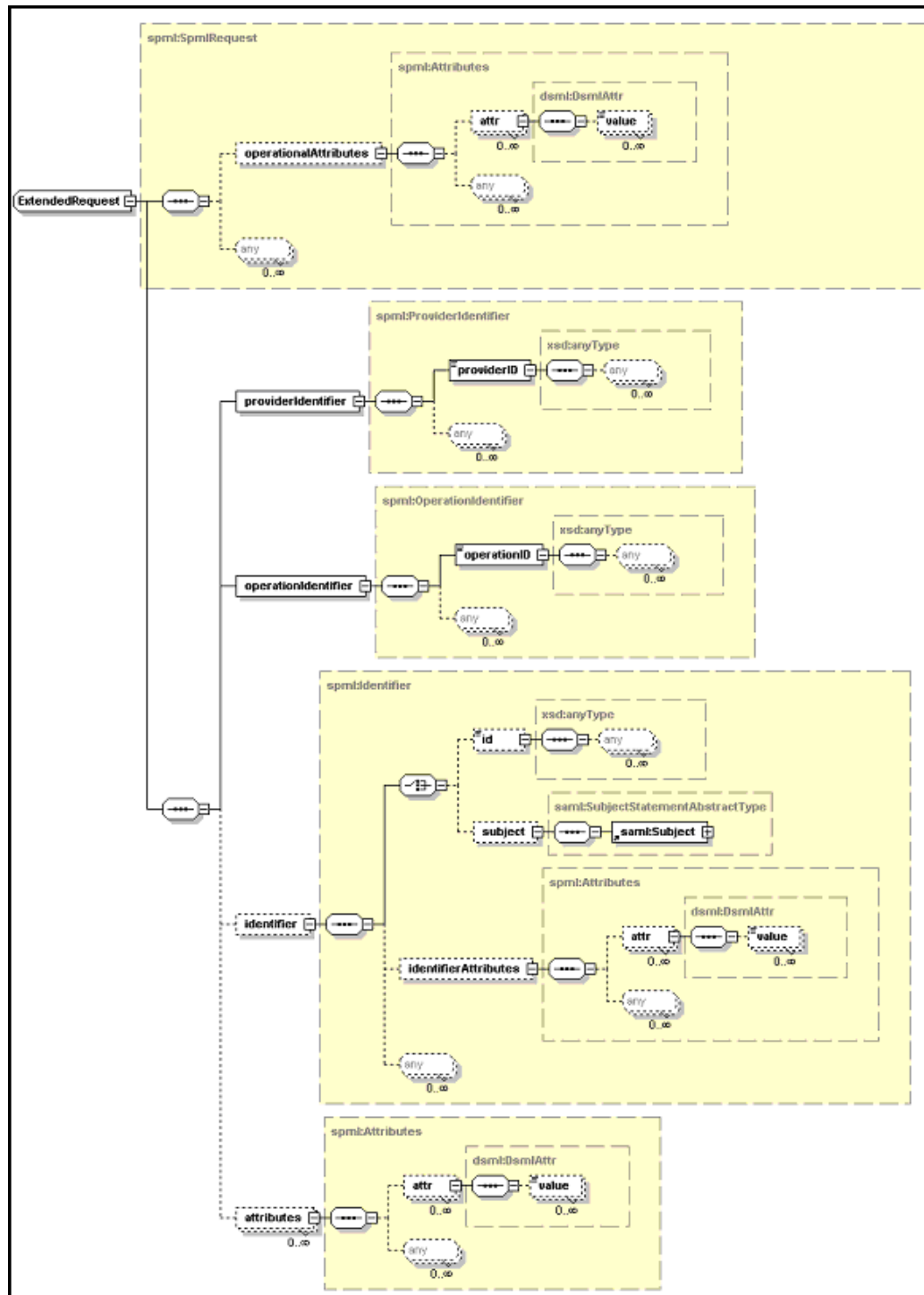


Figure 45: extendedRequest

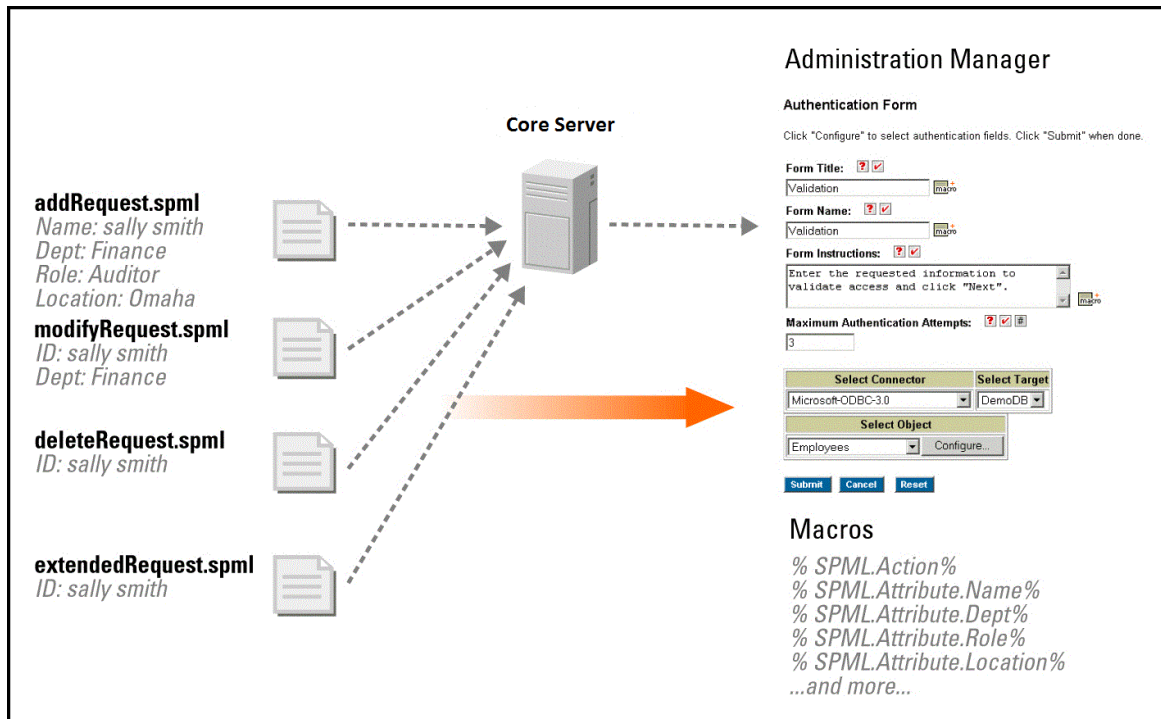


Creating an SPML Template Document

Each SPML request you want a provisioning application to process requires a template. The software translates the template into values that macros in a workflow can use to obtain the user information it needs to provision resources. The template serves as a source for the macros you use from the workflows.

The SPML requests move information into the Core Server for access by macros used in a workflow as shown for a sample user in [Figure 46](#).

Figure 46: How SPML Requests Make User Information Available to Macros in a Workflow



Installation of the Access Assurance Suite creates directory `\CourionService\SPML Samples`. This directory is initially empty but after it contains files, the Administration Manager Macro Selection screen is updated to let you select macros from it. Be sure to save all SPML template documents (with the `.spml` extension) in this `Samples` directory.

Formatting SPML Documents

Template files are case sensitive.

If you create an SPML template using Microsoft Notepad, remove the `.txt` extension from the filename.

Sample Modify Request Template

The following is a sample SPML template for a modify request.

```
<modifyRequest requestId="A1B2C3D$">
  <identifier type="SSN">
    <id>555551234</id>
  </identifier>
  <operationalAttributes>
    <attr name="Authentication Key">
      <value>{F6FD3E74-79D3-4c1a-AA35-D80536D9A345}</value>
    </attr>
  </operationalAttributes>
  <modifications>
    <modification name="Expiration Date">
      <value>12/31/2005</value>
    </modification>
  </modifications>
</modifyRequest>
```

The modify template includes the following components:

- `modifyRequest` — Specifies that this is a modify action request.
- `Identifier type` — Defines the type of identification passed.
- `Identifier id` — Identifies the user to modify.
- `operationalAttributes` — Defines a collection of attributes that are useful for the automation of the provisioning application.
- `modifications` — Specifies each change to make.

Table 4 shows macros that when used in a workflow retrieve information passed in the sample SPML Modify template shown above.

Table 4: Macros to Retrieve Values from the Sample Modify Template

Macro Name to Use in Workflow	Description	Value Macro Retrieves Passed from Sample Modify Template
<code>%SPML.Action%</code>	Retrieves the top tag of the SPML document. <code>xxxRequest</code> , where <code>xxx</code> is Add, Modify, Delete, or Extended.	<code>modify</code>
<code>%SPML.RequestID%</code>	Retrieves the "requestId" XML attribute associated with the top tag.	<code>A1B2C3D\$</code>
<code>%SPML.Identifier%</code>	Retrieves the value of the <code>id</code> tag within the <code>identifier</code> tag.	<code>555551234</code>
<code>%SPML.IdentifierType%</code>	Retrieves the type XML attribute of the <code>identifier</code> tag.	<code>SSN</code>

Table 4: Macros to Retrieve Values from the Sample Modify Template (Continued)

Macro Name to Use in Workflow	Description	Value Macro Retrieves Passed from Sample Modify Template
%SPML.Attribute.xxx%	The <i>xxx</i> is the “name” XML attribute you used in the template. The macro as a whole retrieves the value of the tag <i>modification</i> or <i>attr</i> . For an add, delete, or extended request document, the macro retrieves the <i>name</i> XML attribute associated with <i>attr</i> tags underneath the parent tag <i>attributes</i> .	Expiration Date=“12/31/2005”
%SPML.OperationalAttribute.xxx%	The <i>xxx</i> is the “name” XML attribute you used in the template. The macro as a whole retrieves the value of the tag <i>attr</i> underneath tag <i>operationalAttributes</i> .	Authentication Key = “F6FD3E74-79D3-4c1a-AA35-D80536D9A345”

SPML Add Request

Refer to Table 4 for information about macros you can use in a workflow to retrieve information from this and the other sample templates in this section. As mentioned earlier, these examples include new line, space, and tab characters for easier reading, but actual SPML files use a single-line format, without new lines, spaces, or tabs.

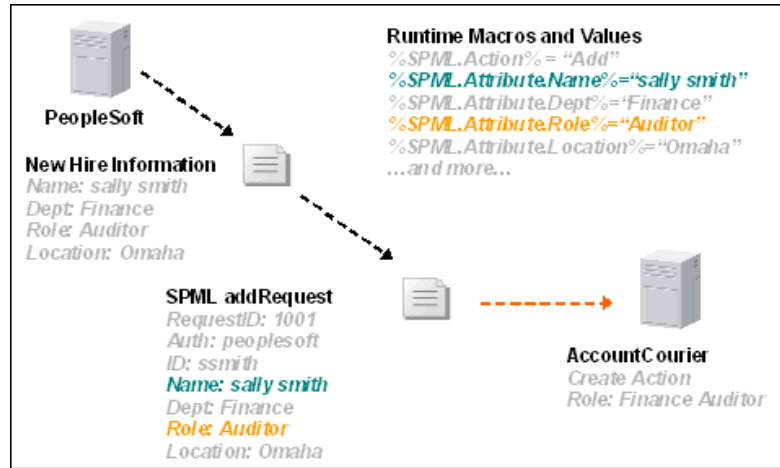
```
<addRequest requestId="A1B2C3D4">
  <operationalAttributes>
    <attr name="Authentication Key">
      <value>{F6FD3E74-79D3-4c1a-AA35-D80536D9A345}</value>
    </attr>
  </operationalAttributes>
  <attributes>
    <attr name="Seed SSN">
      <value>555551234</value>
    </attr>
    <attr name="Source SSN">
      <value>555550001</value>
    </attr>
  </attributes>
</addRequest>
```

The Add template includes the following components:

- **addRequest** — Specifies that this is an Add action request.
requestId — Identifier for the request. This is a tracking mechanism and is useful to pass on to ticketing or notification requests.
- **attr** — Defines a name/value pair in the collection.
- **attributes** — Defines a collection of the data that relates to the individual that the action is occurring against. This is generally the data to be used to provision the individual.

The SPML requests move information into the Core Server for access by macros used in a workflow as shown for a sample user in [Figure](#).

Figure 47: How SPML Requests Make User Information Available to Macros in a Workflow



Sample Template for a Delete Request

```
<deleteRequest requestId="A1B2C3D4">
  <identifier type="SSN">
    <id>555551234</id>
  </identifier>
  <operationalAttributes>
    <attr name="Authentication Key">
      <value>{F6FD3E74-79D3-4c1a-AA35-D80536D9A345}</value>
    </attr>
  </operationalAttributes>
</deleteRequest>
```

The Delete template includes the following components:

- `deleteRequest` — Specifies that this is a Delete action request.
- `requestId` — Identifier for the request. This is useful to pass on to ticketing or notification requests.
- `identifier` and `id` — Identifies the user to delete.
`type` — Defines the type of identification passed.
- `operationalAttributes` — Defines a collection of attributes that are useful for the automation of AccountCourier.
- `attr` — Defines a name/value pair in the collection.

Sample Template for an Extended Request

This sample includes the Verify action.

```
<extendedRequest requestID="8675309">
  <identifier type="SSN">
    <id>123-45-6789</id>
  </identifier>
  <operationalAttributes>
    <attr name="employeebadgenumber">
      <value>ABC123</value>
    </attr>
  </operationalAttributes>
  <providerIdentifier>
    providerIDType="urn:oasis:names:tc:SPML:1:0#URN">
      <providerID>1.3.6.1.4.1.6657
    </providerID>
  </providerIdentifier>
  <operationIdentifier>
    operationIDType="urn:
    oasis:names:tc:SPML:1:0#GenericString:>
    <operationID>Verify</operationID>
  </operationIdentifier>
  <attributes>
    <attr name="RACF-ID">
      <value>CAD001</value>
    </attr>
  </attributes>
</extendedRequest>
```

The Extended request template includes the following components:

- `extendedRequest` — Specifies that this is an extended action request; the component `operationIdentifier` specifies `Verify`.
- `identifier type` — Defines the type of identification passed.
- `identifier id` — Identifies the user to verify.
- `operationalAttributes` — Defines a collection of attributes that are useful for the automation of ComplianceCourier.
- `providerIdentifier` — Defines a collection of provider information that is useful for the automation of ComplianceCourier.
- `operationIdentifier` — Specifies the kind of extended request.

Table 5 shows macros that when used in a workflow could retrieve information passed in the sample SPML Extended Request template shown above.

Table 5: Macros to Retrieve Values from the Sample Extended Request Template

Macro Name to Use in Workflow	Description	Value Macro Retrieves Passed from Sample Extended Template
%SPML.action%	The top tag of the SPML document. <code>xxxRequest</code> , where <code>xxx</code> is Add, Modify, Delete, or Extended.	extended
%SPML.RequestID%	The “requestId” XML attribute associated with the top tag.	8675309
%SPML.IdentifierType	The type XML attribute of the identifier tag.	SSN
%SPML.Identifier%	The value of the <code>id</code> tag contained within the identifier tag.	123-45-6789
%SPML.OperationalAttribute.xxx%	The <code>xxx</code> represents the “name” XML attribute. The macro as a whole retrieves the value of the tag <code>attr</code> underneath tag <code>operationalAttributes</code> .	ABC123
SPML.ProviderIdentifier	The value of the <code>ProviderID</code> attribute of the <code>ProviderIdentifier</code> tag.	1.3.6.1.4.1.6657
SPML.ProviderIdentifierType	The type XML name attribute of the <code>operationalAttributes</code> tag.	urn:oasis:names:tc:SPML:1:0#URN
SPML.OperationalIdentifier		Verify
SPML.OperationalIdentifierType		urn:oasis:names:tc:SPML:1:0#GenericString
%SPML.Attribute.xxx%	The <code>xxx</code> represents the “name” XML attribute. The macro as a whole retrieves the value of tag <code>modification</code> or <code>attr</code> . For <code>addRequest</code> or <code>deleteRequest</code> or <code>extendedRequest</code> documents, the macro retrieves the <code>name</code> XML attribute associated with <code>attr</code> tags underneath the parent tag <code>attributes</code> .	RACF-ID CAD001

Configuring Options in the Administration Manager for Automating a Workflow

When you configure an automated workflow using the XML Access Option, you need to set values on certain forms within the workflow to correspond with the macros in the SPML document. These include

- [*"Authentication Form"*](#)
- [*"Action Selection Form"*](#)
- [*"Community Restrictions"*](#)
- [*"Requirement for Automating the Verify Action"*](#)
- [*"Initial Profile and Resource Overrides Automation"*](#)

SPML Code Sample

Use the following sample SPML sample for an addRequest as a reference for the following sections:

```
<addRequest requestID="1DE2FED$">
  <operationalAttributes>
    <attr name="AuthEmployeeBadgeNumber">
      <value>12345</value>
    </attr>
    <attr name="AuthEmployeeName">
      <value>Fred Manager</value>
    </attr>
    <attr name="TargetID">
      <value>RACF</value>
    </attr>
  </operationalAttributes>
  <identifier type="EmailAddress">
    <id>john.doe@mycompany.com</id>
  </identifier>
  <attributes>
    <attr name="FirstName">
      <value>John</value>
    </attr>
    <attr name="LastName">
      <value>Doe</value>
    </attr>
    <attr name="Description">
      <value>Acct Provisioned Via XML Access Option</value>
    </attr>
  </attributes>
</addRequest>
```


Authentication Form

You can automate authentication from incoming data or configure defaults in the Administration Manager Authentication steps form. See the manual *Configuring Workflows with the Access Assurance Suite Administration Manager* for general information about setting up Authentication in a workflow.

Using the code sample on page 120, use the macros

`%SPML.Operational Attribute.AuthEmployeeBadgeNumber%` and

`%SPML.Operational Attribute.AuthEmployeeName%` in the **DEFAULT VALUE** column of the Authentication Fields Selection form to pre-populate and automate two exposed attributes that would represent the "Badge Number" and "Name" of an authenticated user.

Action Selection Form

You automate action selection through the Resource Actions form. See the manual *Configuring Workflows with the Access Assurance Suite Administration Manager* for general information about this form. In the **ACTION DEFAULT VALUE** text box of this form, specify text or a macro that resolves to one account provisioning action (Add, Change, Create, Delete, Disable, Enable, View, Verify, or Password Reset) that corresponds to the action you specified in the SPML document.

You can use the macro `%SPML.Action%` in the **ACTION DEFAULT VALUE** text box to force the workflow to automate the Add action. If you want to automate the Create action, you can build a custom macro with the VBScript connector that sets the action name based on the value of `%SPML.Action%`. For instance, you could build a resource actions macro such as the following:

```
If "%SPML.Action%" = "Add"
  Then NativeScript = "Add"
  ElseIf "%SPML.Action%" = "Modify"
    Then NativeScript = "Change"
  ElseIf "%SPML.Action%" = "Delete"
    Then NativeScript = "Disable"
  Else NativeScript = ""
End If
```

You would then set the value of the Resource Actions **ACTION DEFAULT VALUE** text box to be the name of custom macro you created. In the **FORM TITLE** and/or **FORM NAME** box, you would specify the action you wanted.

Note: On the Properties page for the workflow action, make sure the **REQUIRE ACTION CONFIRMATION** check box is not checked. If it is checked, the XMLAO action will not occur because the workflow is waiting for a confirmation that cannot occur.

Community Restrictions

If you are using the XML Access Option, you must restrict the seed community (for the `Create` action) and provisionee community (for the `Change`, `Delete`, `Disable`, `Enable`, `View`, and `Verify` actions) to return only one user. You do this on the Seed Community and Provisionee Community Field Selection forms. See the manual *Configuring Workflows with the Access Assurance Suite Administration Manager* for information about these forms.

You can restrict the seed and provisionee communities by adding a **COMMUNITY RESTRICTION** value that represents the user being acted upon. In the code sample on page 120, this user is identified with the macro `%SPML.Identifier%`, which represents the user's email address. With that information, you can construct a community restriction such as `EmailAddress = "%SPML.Identifier%"` where the execution of this value results in the following:

```
EmailAddress = "john.doe@mycompany.com"
```

Optionally, you can leverage the IdentityMap community restriction to help refine the number and types of resources that get provisioned by the SPML request. Using the code sample on page 120, you can use the macro `%SPML.Operational Attribute.TargetID%` to force the workflow to provision only RACF accounts even though the workflow is configured to provision a wider range of systems. An example of this syntax could be `TargetID = "%SPML.Operational Attribute.TargetID%"`.

Requirement for Automating the Verify Action

If you are using the XML Access Option to automate the Verify action, set the **SHOW DECISION OPTIONS** selection on the Properties for Verify Action form to **OFF**. This list is set to **USER LEVEL** by default.

Initial Profile and Resource Overrides Automation

You can use the Default Value column shown on the Initial Profile and Unique Resource Data Selection forms to associate attributes within the workflow with attributes defined in the SPML document. To do this, identify those attributes in the SPML document that directly correspond to the resource being provisioned. This includes any attributes that may also describe the person receiving the provisioned resource (in the case that the SPML request represents a "Create" action).

Note: Any MULTIVALUE type attribute must have the Visible Presentation setting unchecked (not visible).

For the Unique Resource Data Selection form, you also need to set the **OBTAIN VALUE FROM** field to "Default".

From the SPML code sample on page 120, you could use the SPML macros in Table 6 on the Initial Profile and Unique Resource Data Selection forms.

Table 6: SPML Macros on the Initial Profile and Resource Data Selection Forms

Macro	Initial Profile	Unique Resource Data Selection
<code>%SPML.Identifier%</code>	Used as Email Address in Profile Record	Used as email address attached to newly created RACF resource.
<code>%SPML.Attribute.LastName%</code>	Used as Last Name in Profile Record	If pushed into Initial Profile field, can aid AcctIdGeneration of new Resource Name (Create action only).

Table 6: SPML Macros on the Initial Profile and Resource Data Selection Forms

Macro	Initial Profile	Unique Resource Data Selection
%SPML.Attribute.FirstName%	Used as First Name in Profile Record	If pushed into Initial Profile field, can aid AcctIdGeneration of new Resource Name (Create action only).
%SPML.Attribute.Description%	N/A	Used to populate the description attribute of the newly created RACF resource.

Testing the SPML Code

To test your SPML code, use the XMLAO_test.asp page that Core Security ships with the Access Assurance Suite. If AAS is installed in the default location, the page is in the following location:

C:\Program Files (x86)\Courion Corporation\www\WebSamples\AccessOptions\XMLAO\XMLAO_test.asp

Open this page with your web browser, and you see a form with a text box and a text area displayed. The following is a sample URL:

http://localhost/Core/WebSamples/AccessOptions/XMLAO/xmlao_test.asp

In the box labeled Specify Workflow Name, specify the workflow name.

In the large text area below, paste the text of your SPML document (copied from the SPML template file).

Click **SEND...**

The software tests the SPML code against the workflow and reports errors. Also, you can view the `Courion.log` file to verify that the action completed successfully or to view error information.

Configuring ASP Pages to Send SPML to the CourionService

After you successfully test your SPML document against your automated workflow, then you can configure your ASP pages to use the SPML document.

To deliver the SPML document to the CourionService, use the CourATLService method CourATLService.SetBehavior. CourATLService.SetBehavior passes runtime information to the CourionService before the workflow starts.

Step 1

Place a function block similar to the following code sample at the end of ACLogin.asp, which sends the SPML document to the CourionService.

```

\ .....
\ Routine used to pull SPML document and add it to the array
\ that will be sent to the CourionService as part of the
\ "SetBehavior()" call.
\ .....
Function AddSPMLDocument (ByRef inArray, ByRef arrayCnt)

    Dim SSPML
    \ .....
    \ Retrieve your SPML document here
    \ (Can be passed in on the URL, pulled from ADODB, or read
    \ from XML file on disk)
    \ .....
    \ Use a statement of the following form to add requests:
    SSPML = "<addRequest>.....</addRequest>"

    arrayCnt = arrayCnt + 2
    ReDim Preserve inArray(arrayCnt)
    inArray(arrayCnt-2) = "ACXmlInputDoc" 'This is a well-known
    \ identifier indicating that the value is in the SPML document
    inArray(arrayCnt-1) = SSPML

End Function

```

Step 2

Place a function call in the ACLogin.asp page, similar to the one in bold in the following code sample.

Note: Be sure to add AddSPMLDocument before the statement ACComm.SetBehavior().

```

If (STicketID <> "") Then
    arrayCnt = arrayCnt + 2
    ReDim Preserve inArray(arrayCnt)
    inArray(arrayCnt-2) = "Ticket"
    inArray(arrayCnt-1) = sTicketID
End If

` Send down SPML document for processing
AddSPMLDocument(inArray,arrayCnt)

` Used to automate the Runtime Client (Stress Testing Purposes)
Session("AC Client-AutomateClient") =
Request.QueryString("Auto")

SetStateID (Accomm.SetBehavior(SessionID, InArray, Errors))

` Determine if any errors occurred
If (vbTrue = ErrorFound(Errors,MSG_AC_INIT_FAIL)) Then
    exit Function
End If

Set ACComm = Nothing
SetSessionID(SessionID)
SetStateID(StateID)

end Function

```

Chapter 10: The Provisioning Listener for PeopleSoft

This chapter describes how to configure the Provisioning Listener for PeopleSoft® and includes the following sections:

- [*“Requirements” on page 128*](#)
- [*“Configuring PeopleSoft for Provisioning Listener for PeopleSoft” on page 130*](#)

Requirements

The Provisioning Listener for PeopleSoft allows events in the PeopleSoft application, such as adding a new employee, to trigger provisioning activity in AccountCourier.

The Provisioning Listener for PeopleSoft requires the following:

- Microsoft Internet Information Server (IIS)
- MSXML 4.0
- MSVBM60.dll

The Provisioning Listener for PeopleSoft uses the following components:

- **PSHandler.asp** — PSHandler.asp is the target that PeopleSoft notifies when an event is triggered.
- **PSEventHandler.dll** — A COM object that is required by the PSHandler.asp file.
- **Recipient.asp** — A sample file that allows you to print the transformed request that PSHandler.asp receives.
- **ASPUtil.dll** — A utility that contains library functions used by PSEventHandler.dll.
- **Config.xml** — A file that allows you to configure how XML from PeopleSoft is interpreted.

Note: The .dll and .xml files are located in the CourierService folder. The .asp files are located at: Program Files (x86)\Courier Corporation\WWW\WebSamples\AccessOptions\XMLAO\PeopleSoft

Note: You need to manually register PSEventHandler.dll, when necessary. This file must be registered for the PeopleSoft Listener to function.

The following excerpt is a sample code from Config.xml:

```
<tracefile>c:\Program Files\Courion Corporation\Courion
Server\pstrigger.log
</tracefile>
<attributeMapping>
<mapping map="sn">//Transaction/PERSON/NAME_TYPE_VW/
NAMES/LAST_NAME</mapping>
<mapping map="Lastname">//Transaction/PERSON/
NAME_TYPE_VW/NAMES/LAST_NAME</mapping>
</attributeMapping>
<subscribers>
<subscriber name="Courion Corporation">http://squig/
PeopleSoft/
recipient.asp</subscriber>
</subscribers>
```

You can interpret the attributes in the code sample as follows:

tracefile — Path to a valid file for the PSEventhandler.dll to write basic information about tracing.

attributeMapping — Contains one or more mapping tags. You can use several mapping tags which tell the PSEventhandler.dll to map a specific PeopleSoft field to the name specified in the map attribute. The value of each mapping tag must be a valid xpath query.

subscribers — Contains one or more subscriber tags. If a PeopleSoft event is processed successfully, the PSEventHandler.dll notifies each subscriber with the processed data.

Configuring PeopleSoft for Provisioning Listener for PeopleSoft

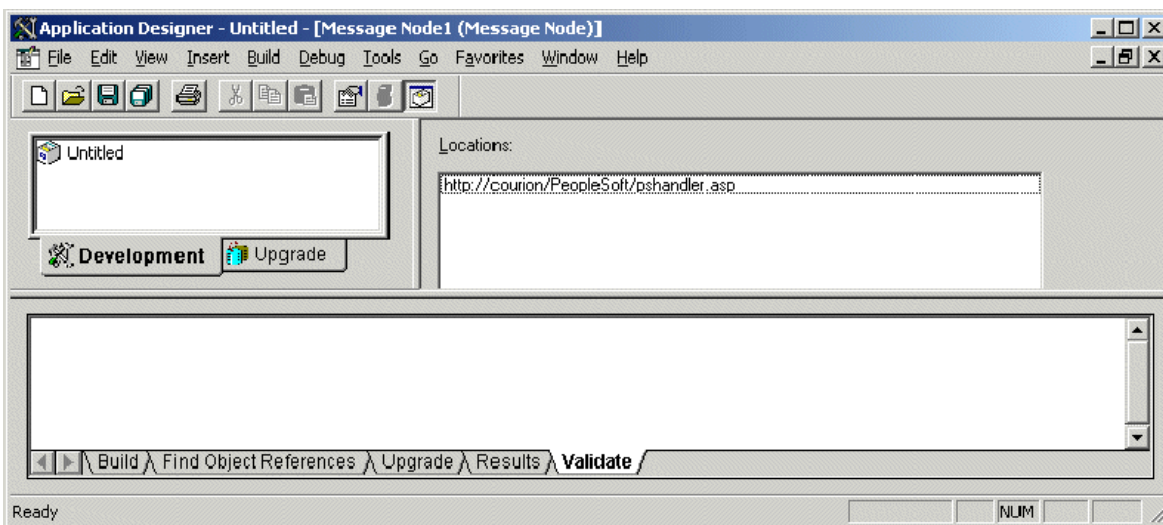
You must configure the PeopleSoft application to trigger provisioning activity in AccountCourier through the Provisioning Listener for PeopleSoft.

Use the PeopleSoft Application Designer to create a new Message Node:

1. Select **FILE > NEW**.
2. Select **MESSAGE NODE** from the list of object types.
3. Select **INSERT LOCATION**. Enter the URL of the Courion handler.

For example, <http://Courion/PeopleSoft/pshandler.asp>, as in [Figure 48](#).

Figure 48: Message Node



4. Select **FILE > SAVE AS** and save the Node as Courion_NODE.

Ensure that you have enabled the following Messages and Message Channels:

- PERSON_BASIC_SYNC
- WORKFORCE_SYNC
- PERSON_BASIC
- PERSON_DATA

These Messages and Message Channels are enabled by default.

To ensure that each Message and Message Channel is enabled, follow these steps:

1. Select **FILE > OPEN**.
2. Select **MESSAGE** or **MESSAGE CHANNEL** from the list of object types. Click **OPEN** and select the object you want from the list of defined objects.
3. Select **FILE > OBJECT PROPERTIES**.

4. Select the **USE** tab to view and disable the **ACTIVE** check box for a Message Property, as in [Figure 49](#).
Select the **USE** tab to view and disable the **RUN** option for a Message Channel Property, as in [Figure 50](#).

Figure 49: Message Property

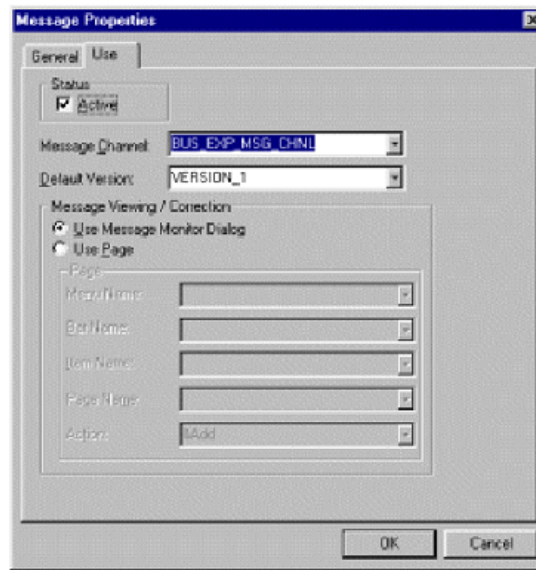
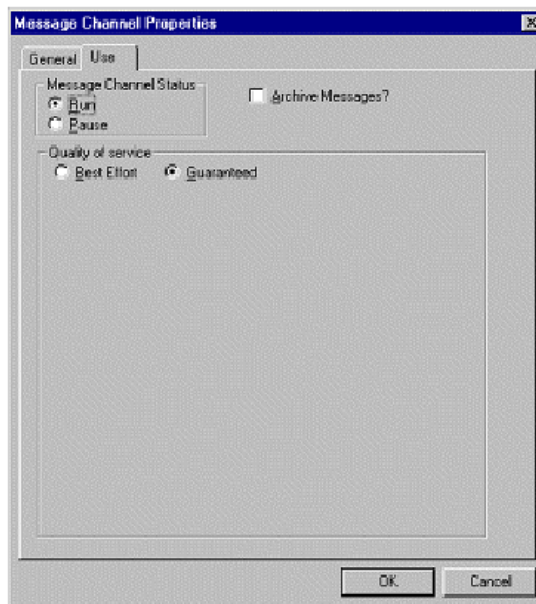


Figure 50: Message Channel Property



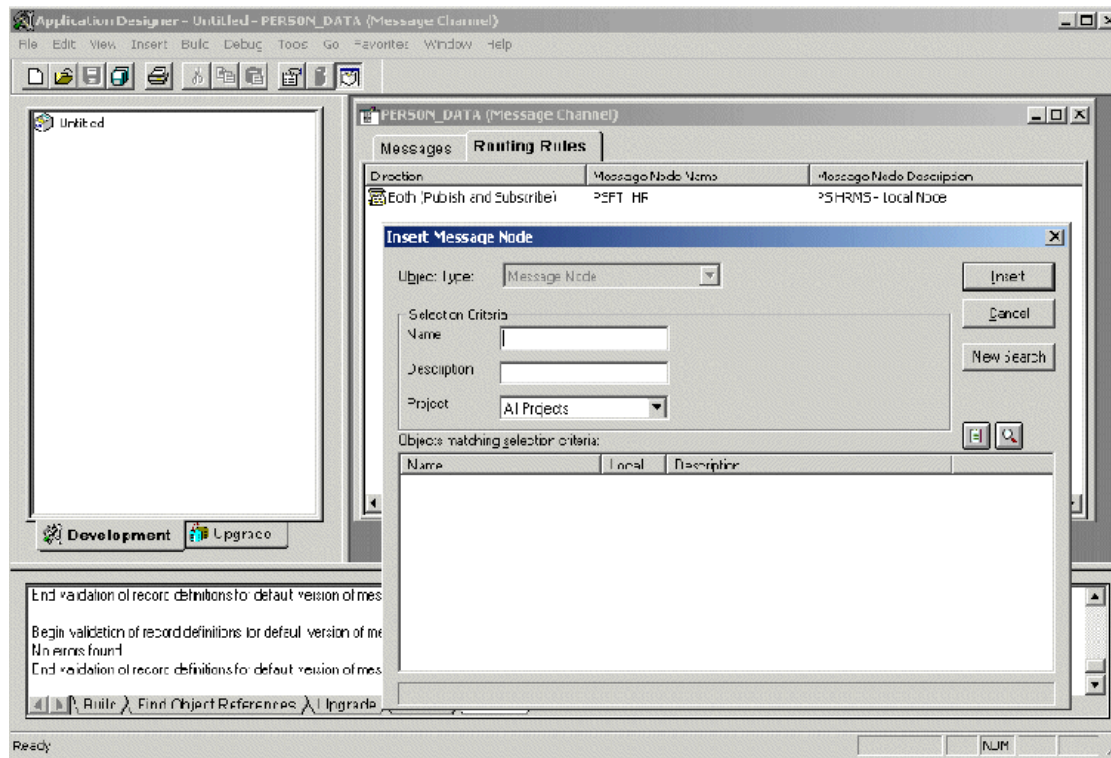
5. Select **FILE > SAVE** to save the object.

Add the Message Node Courion_NODE TO to the Channels PERSON_BASIC and PERSON_DATA.
Repeat the following steps for each Message Channel:

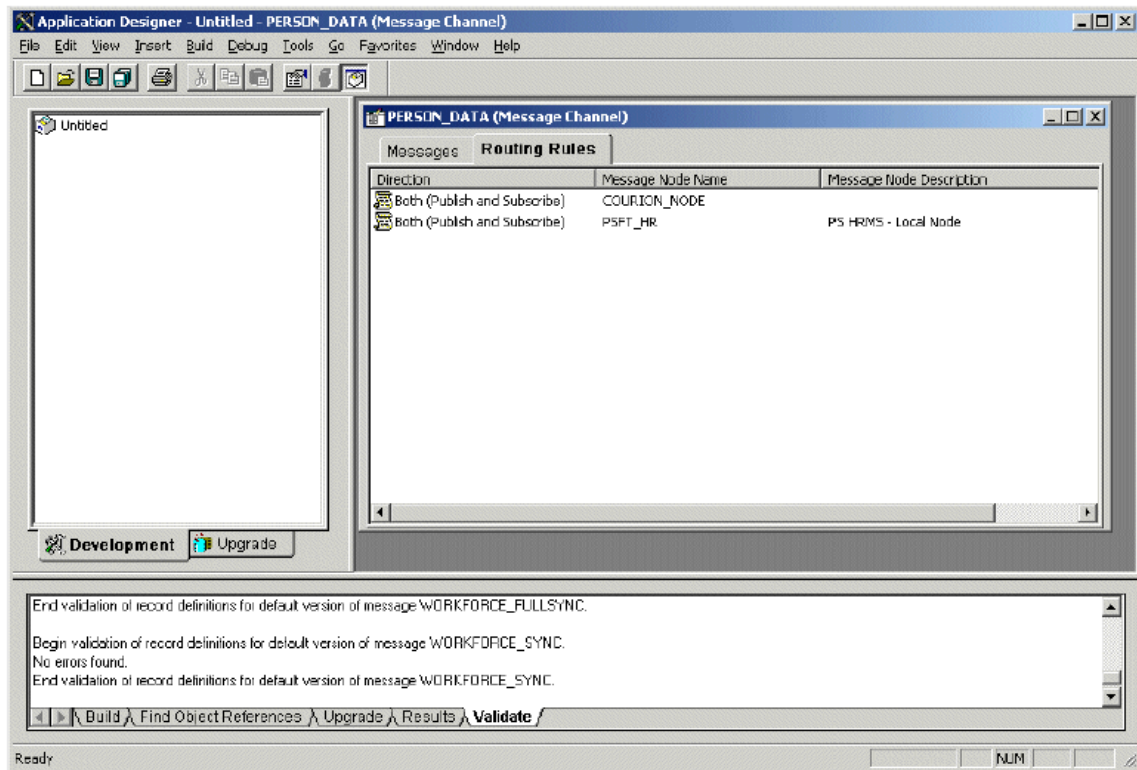
1. Select **FILE > OPEN**.
2. Select **MESSAGE CHANNEL**. Click **OPEN** to open the Message Channel.

3. Click the **ROUTING RULES** tab.
4. Select **INSERT > INSERT MESSAGE NODE** and enter Courion_NODE, as in [Figure 51](#).

Figure 51: Insert Message Node



5. Right-click on the screen to set the **ROUTING DIRECTION**. Choose **PUBLISH TO**, as in [Figure 52](#).

Figure 52: Routing Direction

6. Select **FILE > SAVE** to save the object.

Use the PeopleSoft PSADMIN tool to ensure that the Application Server and the Messaging Channel are activated. For more information, refer to the PeopleBooks Library.

Chapter 11: Optimizing the Performance of the Transaction Repository

Core Security recommends that you optimize the Transaction Repository to improve performance of the Access Assurance Suite. This chapter describes some ways you can do this:

- [*"Using the Microsoft SQL Server Index Tuning Wizard" on page 136*](#)
- [*"Purging the Transaction Repository" on page 137*](#)

Using the Microsoft SQL Server Index Tuning Wizard

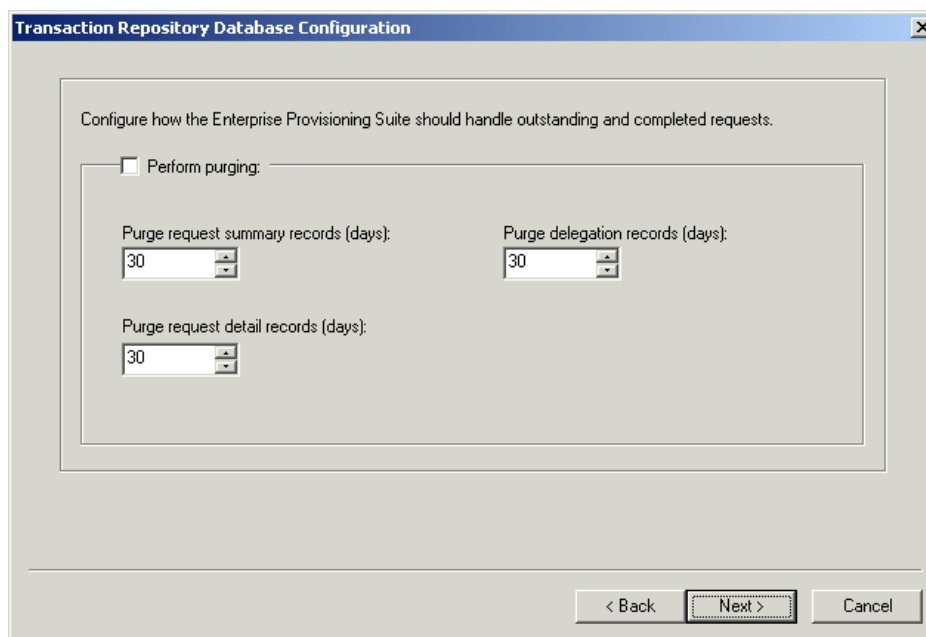
You can use the Microsoft SQL Server Index Tuning Wizard to capture information on the usage of the Access Assurance Suite within your environment during a pilot or simulated use of all the workflows that use the Transaction Repository. You can then make changes to your database based on the recommendations of the Wizard.

For more information about how to use the Microsoft SQL Server Index Tuning Wizard, refer to the Microsoft documentation on this product.

Purging the Transaction Repository

You can regularly purge the transaction repository to limit the size of the database. Since the transaction repository is used for data that is “in process” (requests, approvals, *etc.*), data that has been acted upon may be safely purged. You configure purging of the Transaction Repository in the Configuration Manager on the Database Configuration screen, as shown in [Figure 53](#). See the manual *Installing the Access Assurance Suite* for details.

Figure 53: Transaction Repository Database Configuration



The image shows a Windows-style dialog box titled "Transaction Repository Database Configuration". Inside the dialog, there is a text label: "Configure how the Enterprise Provisioning Suite should handle outstanding and completed requests." Below this, there is a checkbox labeled "Perform purging:". The checkbox is currently unchecked. To the right of the checkbox, there are three spin box controls for specifying purge durations in days. The first spin box is labeled "Purge request summary records (days):" and has the value "30". The second spin box is labeled "Purge delegation records (days):" and also has the value "30". The third spin box is labeled "Purge request detail records (days):" and has the value "30". At the bottom of the dialog, there are three buttons: "< Back", "Next >", and "Cancel". The "Next >" button is highlighted with a dashed border.

If you want to preserve the transaction repository data for reporting purposes, you can archive the transaction repository using the standard procedures for archiving a SQL Server database. See the Microsoft SQL Server documentation for details.

Chapter 12: Configuring PasswordCourier for Transparent Synchronization

This chapter describes how to use the PasswordCourier Transparent Synchronization feature on the provisioning platform, and includes the following sections:

- [*“Overview” on page 140*](#)
- [*“Requirements” on page 142*](#)
- [*“Sample Transparent Synchronization Configuration” on page 143*](#)
- [*“Customizing the Transparent Synchronization Workflow Template” on page 144*](#)
- [*“Installing the Transparent Synchronization Service and the GAT Service” on page 148*](#)
- [*“Installing the Transparent Synchronization Listener” on page 157*](#)

Overview

The PasswordCourier Transparent Synchronization feature allows PasswordCourier to capture password changes from native operating system tools, such as the Microsoft® Windows® 2000 Professional password change dialog box, and propagate them to the Core Security Server for synchronization with a range of targets.

The Transparent Synchronization feature:

- Provides end-users with access to PasswordCourier password resets through their native operating system change password interface.
- Ensures that passwords are synchronized on all targets.
- Provides a high level of security for password resets.
- Counts native password resets against the PasswordCourier history list

The Transparent Synchronization feature has three main components:

- ["The Transparent Synchronization Service"](#)
- ["The Transparent Synchronization Listener"](#)
- ["Transparent Synchronization Workflow"](#)

The Transparent Synchronization Service

The Transparent Synchronization service runs on a Core Server. The Transparent Synchronization service:

- Maintains a lists of Transparent Synchronization Listeners in the network.
- Maintains a Global Arrestor Table (GAT) service. The Transparent Synchronization service uses the information in this table to prevent circular password resets when it receives reset requests from a Transparent Synchronization Listener. The GAT service can reside on the Core Server where you install the Transparent Synchronization service or on another Core Server.

The Transparent Synchronization Listener

A copy of the Transparent Synchronization Listener runs on each domain controller in any domain where you want to detect password changes and use Transparent Synchronization to synchronize your targets. The Listener:

- Detects native operating system password changes and notifies the Transparent Synchronization service.
- Performs both native operating system strength checking and Core Security password strength checking on the new password before it allows the native reset to occur.

For more information about how to install and configure the Transparent Synchronization Listeners currently supported, see ["Installing the Transparent Synchronization Listener" on page 157.](#)

Transparent Synchronization Workflow

The Transparent Synchronization workflow uses the XML Access Option to automate password resets on target systems associated with the user's profile in the IdentityMap datastore. A Transparent Synchronization .asp page (Tsync.asp) that resides on the Core Server submits password reset requests from the Transparent Synchronization Service to the workflow via an SPML request. The workflow resets the password on the target systems that you have configured in the workflow.

When you configure the Transparent Synchronization Service, you specify the name of the Transparent Synchronization workflow and the location of Tsync.asp when you specify information about each listener in the network (see [“Installing the Transparent Synchronization Service” on page 152](#)).

The Transparent Synchronization Workflow Template

During the Courion Server installation, the Express Configuration option includes a Transparent Synchronization workflow template (as well as other sample workflows) that you install as part of the Express Configuration. (See the manual *Installing the Access Assurance Suite* for more information about Express Configuration.) This template is called CourionTransparentSync Reset and you modify it to work in your network by configuring the appropriate synchronized reset targets. See [“Customizing the Transparent Synchronization Workflow Template” on page 144](#) for information about how to do this.

The Tsync.asp Page

The installation of the CourionTransparentSync Reset workflow template automatically installs the Tsync.asp page on the Courion Server.

Requirements

For Core Access Assurance Suite software requirements, please refer to the Product Requirements chapter in *Installing the Core Access Assurance Suite*.

The Core Server where you run the Transparent Synchronization service software and the GAT service software must have access to the profile or ticketing database that contains IdentityMap information.

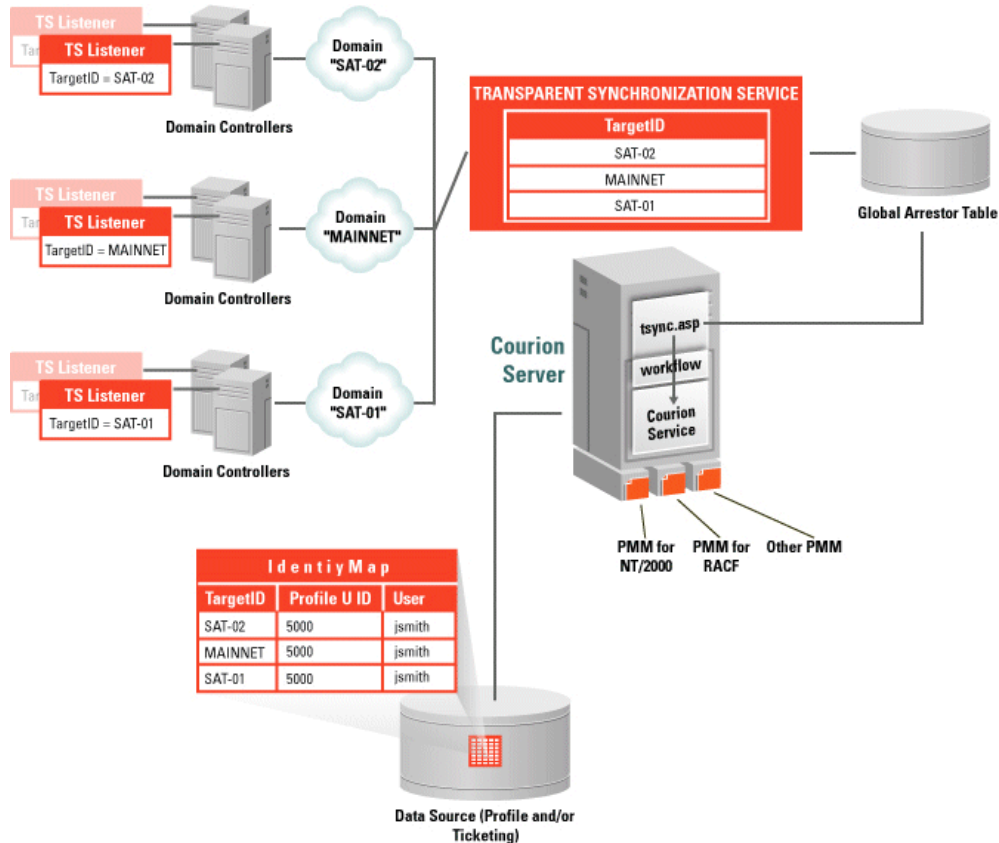
Transparent Synchronization Access Key

The Transparent Synchronization feature requires an access key. If you have questions about how to obtain an access key for Transparent Synchronization, contact Core Security Customer Support.

Sample Transparent Synchronization Configuration

[Figure 54](#) represents transparent synchronization configuration.

Figure 54: Transparent Synchronization Configuration



In this configuration, the Transparent Synchronization service and the GAT service reside on the same Core Security Server with the tsync.asp file. The tsync.asp file submits the password reset request to the targets through the Transparent synchronization workflow. Three Transparent Synchronization listeners, (SAT-02, MAINNET, and SAT-01) are configured. A password reset being triggered on any one of these listeners causes password synchronization for the user's other resources. The target IDs are configured in the IdentityMap for user jsmith.

If, for example, a password reset request occurs on listener SAT-02, the listener passes the reset request to the Transparent Synchronization Service. The service passes the request to the Tsync.asp page which submits the request to the Transparent Synchronization workflow. The workflow uses the XML Access Option to process the reset request on other targets associated with the user profile in the IdentityMap.

Note: It is not necessary for the listener to be a member of the synchronized reset target. It is necessary to have an entry in the IdentityMap that corresponds to the user performing the native password reset on the listener target.

Note: The synchronized reset target can contain members that are not Transparent Synchronization listeners.

Customizing the Transparent Synchronization Workflow Template

During the Core Server configuration, run Express configuration and install the sample workflow for Transparent Synchronization called **CourionTransparentSync Reset**. This is the workflow you customize to enable the Transparent Synchronization feature at your site.

To customize the workflow, configure each of the synchronized reset targets in your network through the global Target Configuration form. Then configure Unique Resource Data for each of these targets by specifying the %SPML_Attribute.Password% macro as the default value for the Text Control Type under Attribute Display Options on the Unique Target Data Field Selection Form.

Other forms within the workflow template, including Authentication forms and forms for other Global settings, are preconfigured and you do not need to modify them.

Note: You can create a new Transparent Synchronization workflow using the CourionTransparentSync Reset workflow as a model for the new workflow. When you create the new workflow, select CourionTransparentSync Reset from the **COPY FROM A PREVIOUSLY CONFIGURED WORKFLOW:** drop-down list.


Configuring Targets

To add targets for transparent synchronization resets:

Flowchart View

1. Log in to the Administration Manager Flowchart View.
2. Click on **GLOBAL SETTINGS** from the navigation bar.
3. Click on **TARGET CONFIGURATION**.

Tree View

1. Log in to the Administration Manager Tree View.
2. Expand the workflow.
3. Expand **ACTIONS**, expand **GLOBAL SETTINGS**, and click the  icon next to **TARGET CONFIGURATION**.

The Administration Manager displays the Target Configuration Form, shown in [Figure 55](#). The form lists all the connectors configured with the Connector Configuration Manager.

Figure 55: Target Configuration Form

Target(s) Configuration Form

The list below displays all currently configured targets. To view / modify the details about one or more targets check the button next to each target you wish to view / modify and then click on the "View / Modify". To delete one or more targets check the button next to each target you wish to delete and then click the "Delete" button. To add a new target click on the "Add" button.

Configured Targets:

[Select All] [Clear All]

Select	Connector ▲	Target ◆	Resource ◆	Target ID ◆	Category ◆	Capabilities ◆	Available ◆
<input type="checkbox"/>	PMM-Gateway-Cnctr	TSyncListener	Account Password	TSyncListener			<input checked="" type="checkbox"/>
<input type="checkbox"/>	PMM-Gateway-Cnctr	ADPMMExpress	Account Password	ADPMMExpress		Password	<input checked="" type="checkbox"/>

[Select All] [Clear All]

View / Modify **Delete** **Add**

- Click the **ADD** button to configure a new target. The Add Target form appears, as in [Figure 56](#).

Figure 56: Add Target Form

Add Target Form

In order for a target configured in Connector Configuration Manager to be accessible to this workflow you must use this form to add a new provisioning target for it. To do so, select the connector and the name of the target to be associated with the new provisioning target. You must also specify the resource (table, user, or account password) and the Unique Attribute (field that uniquely identifies a resource on this target).

Next specify a value for the IdentityMap(TM) Target ID which will be used when creating IdentityMap(TM) entries for the target. You can optionally specify an alias and/or description for the target to make it easier for you to identify it). Checking the Available check box will make the target available in the workflow.

Click the "Save" button to add the target or the "Cancel" to return to the Provisioning Target Configuration Form without creating a new provisioning target.

Connector: <input checked="" type="checkbox"/>	<input type="text"/>
Target: <input checked="" type="checkbox"/>	<input type="text"/>
Resource: <input checked="" type="checkbox"/>	<input type="text"/>
Unique Attribute: <input checked="" type="checkbox"/>	<input type="text"/>
IdentityMap(TM) Target ID: <input checked="" type="checkbox"/>	<input type="text"/> <small>macro</small>
Alias:	<input type="text"/> <small>macro</small>
Description:	<input type="text"/> <small>macro</small>
Category:	<input type="text"/> <small>macro</small>
Available:	<input checked="" type="checkbox"/>

Save **Cancel**

Fill in the following information:

- From the **Connector** drop-down list, select the PMM Gateway Connector (Pmm-Gateway-Cnctr).
- From the **TARGET** drop-down list, select a target associated with the connector.
- From the **RESOURCE** drop-down list, select Account Password.

8. From the **UNIQUE ATTRIBUTE** drop-down list, select Password.
9. In the **IDENTITYMAP TARGET ID** column, enter the corresponding PasswordCourier Target ID for the selected target. If the IDs do not match, PasswordCourier and the IdentityMap feature do not function properly.

If PasswordCourier is not installed, any ID works as long as it is unique and used consistently with other alias definitions for the same target.
10. Leave the **ALIAS** field blank.
11. Leave the **DESCRIPTION** field blank.
12. Leave the **CATEGORY** field blank.
13. Check the **AVAILABLE** check box to make the connector/target combination available in the workflow (this is enabled by default).
14. Click **SAVE**.

Set Up the Unique Resource Data Form

For each target you configured through the Target Configuration form, you configure certain overrides on the Unique Resource Data Form:


- You must configure the Attribute Display Options for the Password attribute with Text as the Control Type and the %SPML.Attribute .Password% macro as the default value for this Control Type. You can select this macro from the Macro Selection table or copy it from the ADPMMExpress target, which is automatically configured with the Transparent Synchronization workflow template.
- You can optionally specify a password strength function as a Validation override. Two password strength functions are configured in the workflow: DefaultStrength and NumericReq. Refer to the chapter on Configuring Functions in the manual *Configuring Workflows with the Access Assurance Suite Administration Manager* for information about how to create your own functions.

To set up the Unique Resource Data Form:

Flowchart View

1. Log in to the Administration Manager Flowchart View.
2. Select **BUSINESS PROCESS** and **PASSWORD RESET** from the navigation bar.
3. Click on **PASSWORDRESET REQUEST** and click on **UNIQUE RESOURCE DATA**.

Tree View

1. Log in to the Administration Manager.
2. Expand the workflow.
3. Expand **ACTIONS**, expand **PASSWORD RESET**, expand **ACTION SETTINGS**, and click the  icon next to **UNIQUE RESOURCE DATA**.

The Administration Manager displays the Unique Resource Data Form, similar to the one shown in [Figure 57](#).

Figure 57: Unique Resource Data Form

Unique Target Data Form

Enter the unique data you want to apply to each target resource. In the "Configuration" column of the Connector/Target that will contain the new resource, click "Create" for the initial configuration or "Modify" to change a configuration. Click "Submit" when done.

Form Title:

Form Name:

Form Instructions:

Select Configuration to Override with Unique Data:

Connector	Target	Object	Configuration	Dependency
PMM-Gateway-Cnctr	ADPMMExpress	Account Password	Modify	None
PMM-Gateway-Cnctr	cour_hp	Account Password	Modify	None

- In the **CONFIGURATION** column of the connector/target that contains the reset password, click **CREATE**. If a Unique Target Data form for this connector/target has already been configured, click **MODIFY** to change the form.

The Administration Manager displays the Unique Resource Data Field Selection Form. [Figure 58](#) shows the portion of the form showing the Attribute display options column and the validation column.

Figure 58: Unique Resource Data Field Selection Form

Resource Attribute	Presentation	Label	Help Text	Obtain Value From	Attribute Display Options
Password	<input checked="" type="checkbox"/> Required <input checked="" type="checkbox"/> Secure <input checked="" type="checkbox"/> Verify <input type="checkbox"/> Visible <input type="checkbox"/> Read Only <input type="checkbox"/> Global	<input type="text" value="Password:"/>	<input type="text" value="New password for the provid"/>	<input type="text" value="Default Value"/>	Control Type: <input type="text" value="Text"/> Default Value: <input type="text" value="%SPML.Attribute.Password%"/>
Force Change	<input type="checkbox"/> Required <input type="checkbox"/> Secure <input type="checkbox"/> Verify <input checked="" type="checkbox"/> Visible <input type="checkbox"/> Read Only <input type="checkbox"/> Global	<input type="text" value="Force change at next login?"/>	<input type="text" value="Should the user's password"/>	<input type="text" value="Default Value"/>	Control Type: <input type="text" value="Boolean"/> Default Value: <input type="checkbox"/>

- In the Attribute Display Options column, choose Text from the drop-down list if this does not appear as the default. Insert the %SPML.Attribute.Password% macro as the default value. You can select it from the macro table, or copy it from the ADPMMExpress target, which is automatically configured with the Transparent Synchronization workflow template.
- Select a password strength function from the Validation drop-down box.
- Click **SUBMIT**. The Unique Target Data Form appears. Click **SUBMIT** on this form.

Repeat this process for each target you configured in the Transparent Synchronization workflow.

Installing the Transparent Synchronization Service and the GAT Service

Installing the Transparent Synchronization feature includes two major steps: Installing the Global Arrester Table (GAT) service and Installing the Transparent Synchronization service. These components can reside on different Core Servers or the same server. You can install these components in any order. These examples show the GAT service installed first.

Note: If the GAT service and the Transparent Synchronization service are on separate Core Servers, they must share the same security pass phrase. To update the security pass phrase on a Core Server from the Microsoft Windows Start menu, select:

Programs>Core Access Assurance Suite>Core Server>Configuration Manager

Click **NEXT** on the Access Key Selection dialog box. On the Site-specific Information dialog box, enter a pass phrase. Click **NEXT** through the remaining dialog boxes, and follow the instructions to accept changes.

After you install and configure the Transparent Synchronization service and the GAT service, you can manage these components through shortcuts on the Microsoft Windows Start menu. See [“Using the Start Menu to Manage the Transparent Synchronization Service and the GAT Service” on page 156](#).

Installing the GAT Service

Note: In an environment with more than one Core Server, one GAT service is used by all Core Servers whether they use Transparent Synchronization or not.

To install the GAT service from the Microsoft Windows Start menu, select:

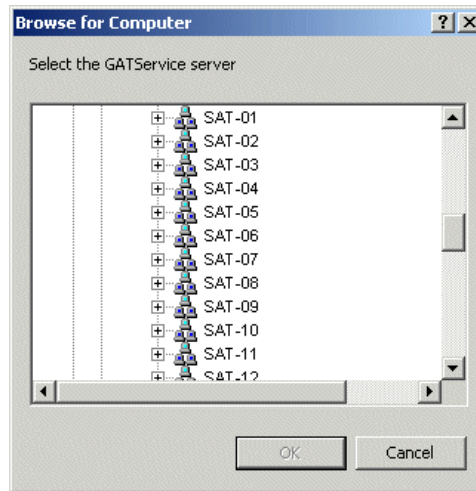
Programs>Core Access Assurance Suite>Transparent Synchronization>Configure GAT Service

The Configure GAT Service dialog box appears, as in [Figure 59](#).

Figure 59: Configure GAT Service Dialog Box



- To install the GAT Service on the host where you installed the Transparent Synchronization software, select **ON THE LOCALHOST (SERVICE)** and click **OK**. You can now install the Transparent Synchronization Service (see [“Installing the Transparent Synchronization Service” on page 152](#)).
- To browse the network for another host, select **ON THIS SERVER**. The browser window in [Figure 60](#) appears.

Figure 60: Browse for Computer Window

- Select a server from the list and click **OK**.

If you selected a Core Server on the network, you must configure the GAT service on that server. From the Microsoft Windows Start menu on the Core Server where you install the GAT service, select:

Programs>Core Access Assurance Suite>Transparent Synchronization>Configure GAT Service

The Configure GAT Service dialog box appears as in [Figure 59](#). Select **ON THE LOCALHOST (SERVICE)** and click **OK**.

If your environment has only one Core Security Server, continue with the steps in the section [“Installing the Transparent Synchronization Service” on page 152](#).

Running the Distributed COM Configuration Utility in Environments with Multiple Courion Servers

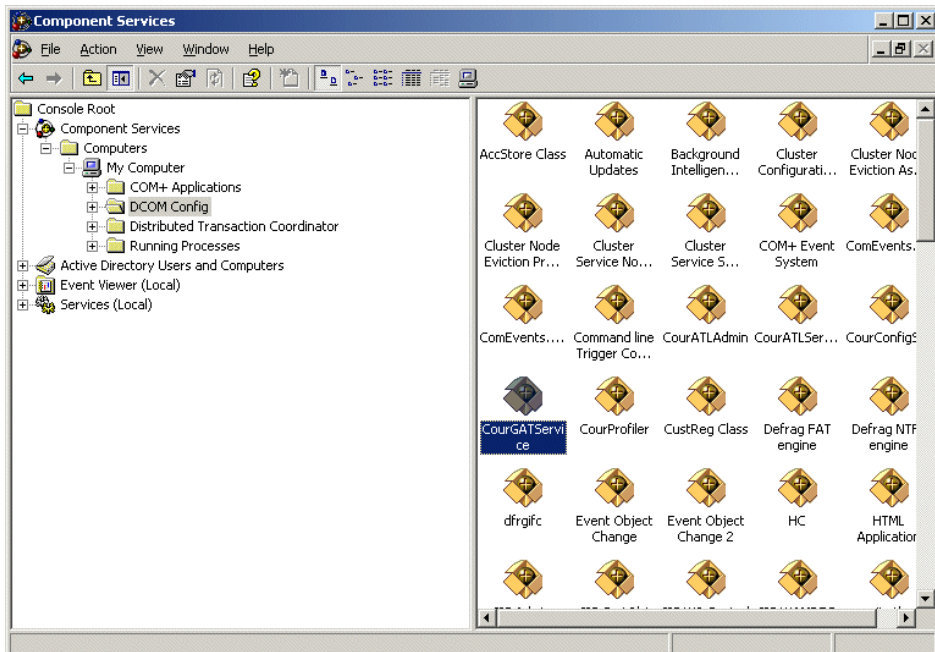
If you have more than one Courion Server in your networking environment, you need to choose one Core Security Server to run the GAT Service. Then you need to configure the Distributed COM (DCOM) profile of the GAT service on that Core Server to set the correct security parameters for that environment.

The dialog box used to change the DCOM profile is called “CourGATService Properties”.

Accessing the CourGATService Properties

Launch the Component Services MMC snap-in on the server where you installed the GAT service:

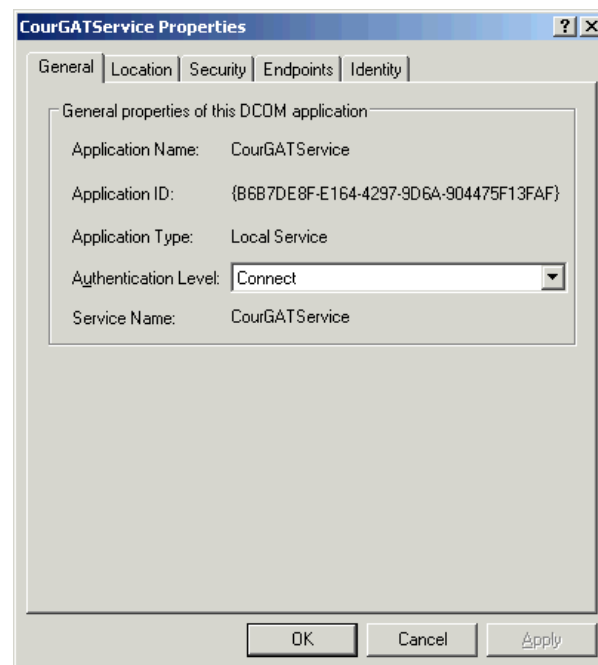
1. From the Start Menu, select:
Settings>Control Panel>Administrative Tools>Component Services
This launches the MMC snap-in, show in [Figure 61](#).

Figure 61: MMC Snap-In

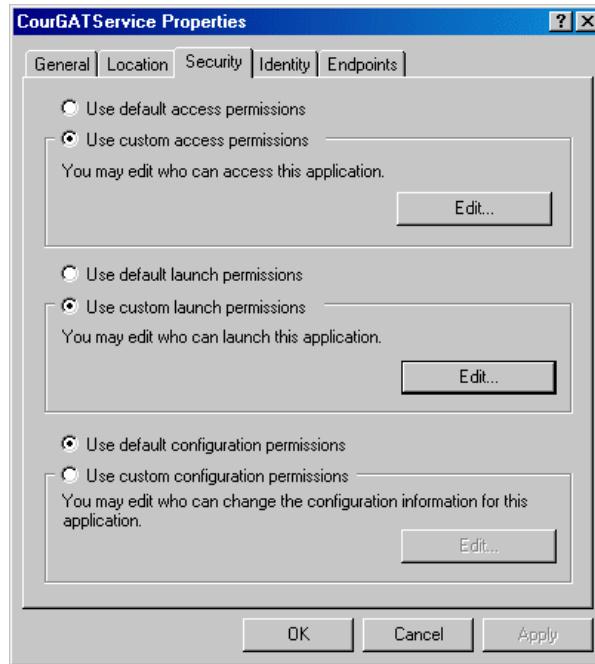
- From the directory tree on the left side of the screen, select:

Component Services>Computers>My Computer>DCOM Config

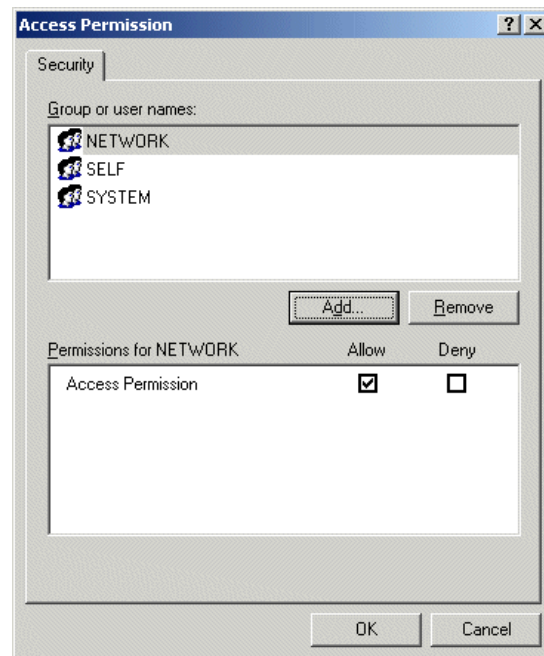
On the container view in the right side of the screen, select CourGATService, right-click the mouse, and select Properties. The CourGATService Properties dialog box appears as in [Figure 62](#).

Figure 62: CourGATService Properties Dialog Box

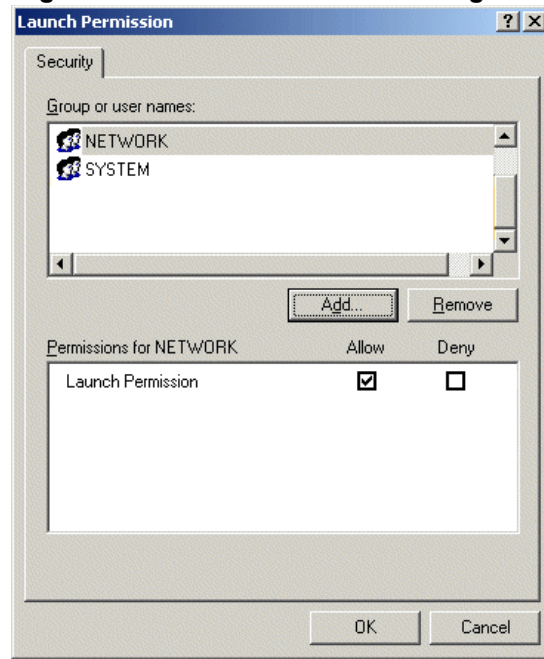
- Select the **SECURITY** tab. The CourGATService security properties dialog box appears as in [Figure 63](#).

Figure 63: CourGATService Security Properties Dialog Box

4. Select **Use custom access permissions** and click **EDIT...**. The Access Permission dialog box appears as in [Figure 64](#).

FIGURE 64: ACCESS PERMISSION DIALOG BOX

5. Ensure that the **NETWORK** user is on the list of **GROUP OR USER NAMES**. If it is not on the list, click **ADD...** and select the **NETWORK** user from the list of users and click **OK**. From the CourGATService Properties dialog box, select **USE CUSTOM LAUNCH PERMISSIONS** and click **EDIT...**. The Registry Value LaunchPermissions dialog box appears as in [Figure 65](#).

Figure 65: Launch Permission Dialog Box

6. Check that the NETWORK user is on the list of names. If it is not on the list, click **ADD...** and select the NETWORK user from the list of users and click **OK**.
7. On the CourGATService Properties dialog box click **OK**.

You can now configure the Transparent Synchronization Service.

Installing the Transparent Synchronization Service

From the Microsoft Windows Start menu, select:

Programs>Core Access Assurance Suite>Transparent Synchronization>Install Transparent Synchronization Service

The Transparent Synchronization Configuration Manager dialog box appears as in [Figure 66](#).

Figure 66: Transparent Synchronization Configuration Manager

The screenshot shows the 'Transparent Synchronization Configuration Manager' dialog box. It is divided into several sections:

- SSL Settings:** Includes a 'Port Number' field set to 26000 and a 'Timeout (seconds)' field set to 90.
- Non-SSL Settings:** Includes a 'Port Number' field set to 26010 and a 'Timeout (seconds)' field set to 90.
- Global Arrestor Table Settings:** Includes a 'Hostname or IP address' field set to LOCALHOST and a 'Select...' button.
- Configured Listeners:** A table with two columns: 'Target ID' and 'HostName or IP Address'. The table is currently empty.

At the bottom of the dialog are three buttons: 'Add...', 'Edit...', and 'Delete'. On the right side of the dialog are three buttons: 'OK', 'Cancel', and 'About...'.

1. **SSL SETTINGS** — The Secure Socket Layer (SSL) protocol ensures security for the communications between the Transparent Synchronization service and the Listener.
 - **PORT NUMBER** — The port number you enter here must be the same as the port number in the Listener Configuration dialog box. (See [“Transparent Synchronization Listener Configuration” on page 160.](#))
 - **TIMEOUT (SECONDS)** — How long the Transparent Synchronization service waits for communication from the Listener before it assumes the Listener is unavailable and logs an error.
2. **NON-SSL SETTINGS** — (These settings do not apply to this version of the Access Assurance Suite).
3. **GLOBAL ARRESTOR TABLE SETTINGS** — This is the host name or IP address of the Core Server where you installed the GAT service. LOCALHOST appears by default if you have not installed the GAT service or if you have installed the GAT service on this server. If you have not installed the GAT service and choose to do so now, click the **SELECT** button. The dialog box in [Figure 59](#) appears. Follow the instructions in the section [“Installing the GAT Service” on page 148](#) to complete this dialog box.
4. **CONFIGURED LISTENERS** — From the Configured Listeners section of the Transparent Synchronization Configuration Manager dialog box, click the **ADD** button to add a Listener. The Listener Configuration Dialog box appears as in [Figure 67](#).

Figure 67: Listener Configuration

Listener Configuration

Listener Information

Target ID:

Hostname or IP address:

Security

Security Phrase:

Verify Security Phrase:

Workflow

☒ Use PasswordCourier
☐ Use PasswordCourier Classic

Workflow Details

Workflow Name:

Transparent Synchronization URL:

OK Cancel

5. **LISTENER INFORMATION** — The Listener information you enter here corresponds to the information you enter in the Transparent Synchronization Listener Configuration dialog box when you install the Listener.

- **TARGET ID** — Enter the name of the target you created through PasswordCourier Support Staff Customization Manager. This is the name of the domain or workstation where the Listener software is installed. It must match the name of the Target ID in the Transparent Synchronization Listener Configuration dialog box (see [Figure 70](#) on page 160).
- **HOSTNAME OR IP ADDRESS** — Enter the hostname or IP address of the Target ID of the Listener on the domain controller (see [Figure 70](#) on page 160).

6. **SECURITY** — Enter the security phrase.

- **SECURITY PHRASE** — Enter a text string to use as a security phrase (128 characters maximum). This security phrase must match the security phrase that you enter in the Listener Configuration window in the Transparent Synchronization Listener Configuration dialog box (see page 160).

Re-enter the security phrase in the Verify Security Phrase window.

7. **WORKFLOW** — Select **USE PASSWORDCOURIER**.

8. **WORKFLOW DETAILS**

- **WORKFLOW NAME** — This is the name of the Transparent Synchronization workflow, configured through the Access Assurance Suite Administration Manager. The name of the workflow template that is automatically installed during the Express Configuration is **COURIONTRANSPARENTSYNC RESET**.

- **TRANSPARENT SYNCHRONIZATION URL** — Enter the Web address to the `tsync.asp` page. This web page submits the reset request to the Transparent Synchronization workflow. The default address is:
<http://localhost/Core/WebSamples/AccessOptions/XMLAO/Tsync/Tsync.asp>

Note: The `tsync.asp` file must be modified if the password strength failure message is changed in the workflow. If the string "Please re-enter your password." is not included in the dictionary failure message, replace that string in `tsync.asp` to match a string that is part of the message configured for password strength check failure.

9. Click **OK**.

Using the GAT server host name and Listener names in this example, the Transparent Synchronization Configuration Manager dialog box appears as in [Figure 68](#).

Figure 68: Transparent Synchronization Configuration Manager with Configured Listeners

The screenshot shows the 'Transparent Synchronization Configuration Manager' dialog box. It has a title bar with standard window controls. The dialog is divided into several sections:

- SSL Settings:** Port Number: 26000, Timeout (seconds): 90.
- Non-SSL Settings:** Port Number: 26010, Timeout (seconds): 90.
- Global Arrestor Table Settings:** Hostname or IP address: \\VMS1, with a 'Select...' button.
- Configured Listeners:** A table with two columns: 'Target ID' and 'HostName or IP Address'. It lists two entries: 'MAINNET' with 'mainsrv.corp.widget.com' and 'SAT-01' with 'sat01.corp.widget.com'. Below the table are 'Add...', 'Edit...', and 'Delete' buttons.

On the right side of the dialog, there are three buttons: 'OK', 'Cancel', and 'About...'.

If you are satisfied with the configuration information, click **OK**.

Using the Start Menu to Manage the Transparent Synchronization Service and the GAT Service

You can update the Transparent Synchronization service and the GAT service settings, or remove these services, through the Microsoft Windows Start menu.

To access these options from the Microsoft Windows Start menu, select:

Programs>Core Access Assurance Suite>Transparent Synchronization

You can select these options:

CONFIGURE GAT SERVICE — Displays the Configure GAT Service Dialog box ([Figure 59](#)) and allows you to specify a GAT server.

INSTALL TRANSPARENT SYNCHRONIZATION SERVICE — Installs the Transparent Synchronization service and displays the Transparent Synchronization Configuration Manager dialog box ([Figure 66](#)). Use this option for first-time installations or after you have removed the Transparent Synchronization Service and want to re-install it.

REMOVE GAT SERVICE — Removes the GAT service from the Core Server where you installed it. Removing the GAT service disables the Transparent Synchronization feature.

REMOVE TRANSPARENT SYNCHRONIZATION SERVICE — Removes the Transparent Synchronization service from the Core Server where you installed it. Removing the Transparent Synchronization service disables this feature.

TRANSPARENT SYNCHRONIZATION CONFIGURATION MANAGER — Displays the Transparent Synchronization Configuration Manager dialog box. Use this option to modify information associated with the Transparent Synchronization service.

Enabling or Disabling Logging for the Transparent Synchronization Service and the GAT Service

You can enable or disable logging for the Transparent Synchronization Service and the GAT Service by using a registry key. To access this option, you need to manually enter a registry key, with or without a string value.

To enable logging, enter either of the following registry keys:

```
HKEY_LOCAL_MACHINE\Software\Courion Corporation\{3BEDAB27-38CE-4A28-B0CD-E6B469CE9FDC}
```

OR

```
HKEY_LOCAL_MACHINE\Software\Courion Corporation\{3BEDAB27-38CE-4A28-B0CD-E6B469CE9FDC}\EnableLogging=true
```

To disable logging, enter the registry key with any string value besides "true". For example:

```
HKEY_LOCAL_MACHINE\Software\Courion Corporation\{3BEDAB27-38CE-4A28-B0CD-E6B469CE9FDC}\EnableLogging=false
```

Installing the Transparent Synchronization Listener

This section explains how to install and configure the Transparent Synchronization Listeners.

- [*“Transparent Synchronization Listener for Microsoft Windows” on page 158*](#)
- [*“Transparent Synchronization Listener for Microsoft Server Core for Windows Server” on page 165*](#)
- [*“Transparent Synchronization Listener for i5/OS” on page 172*](#)

For general definition of the Transparent Synchronization Listener and functionality details, see [*“The Transparent Synchronization Listener” on page 140*](#).

Transparent Synchronization Listener for Microsoft Windows

A copy of the Transparent Synchronization Listener runs on each domain controller in any domain where you want to detect password changes and use Transparent Synchronization to synchronize your targets.

Requirements

Install the latest version of the Microsoft XML Core Services (MSXML). To download MSXML go to:

`http://www.microsoft.com/downloads`

Reboot the computer after the installation is complete.

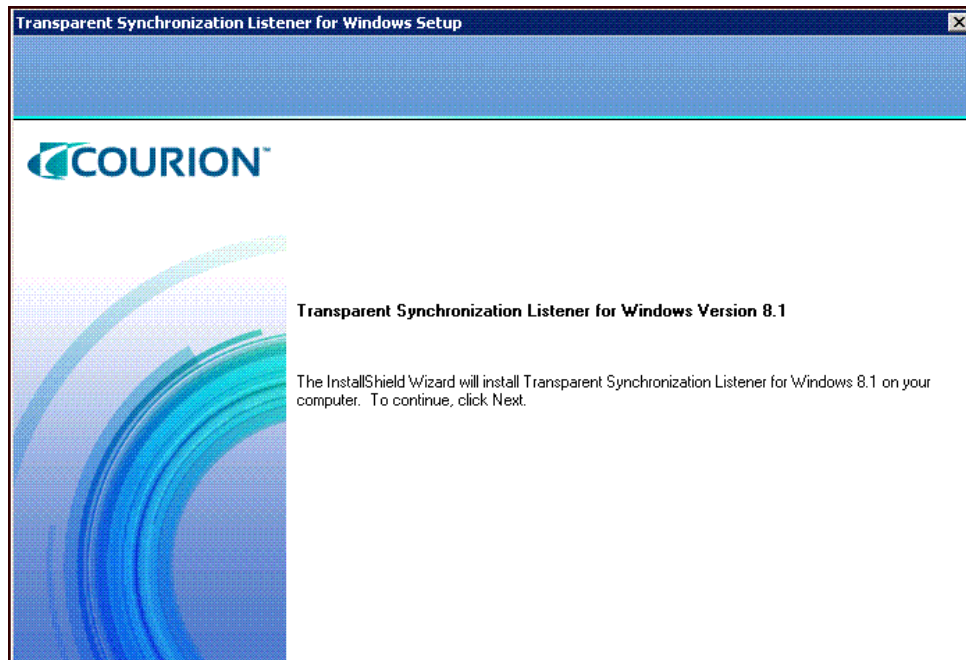
Note: A reboot of the domain controller is needed after uninstalling the listener. For upgrades of this listener you must uninstall the existing, reboot, install the newer, and reboot. The listener is loaded by the LSASS.exe system process, hence the need for the reboot to load and unload it, respectively.

The only time the listener may need to be upgraded is for a bug fix or if the encryption algorithms need upgrading. For the most part this listener is version independent from the Access Assurance Suite, except as noted about the encryption algorithm used for the data sent between the two.

Installing the Listener

To Install the Listener:

1. Locate the folder for the Listener software in the following subdirectory on the Core Server where you installed the Transparent Synchronization software. This example shows the location of the Microsoft Windows listener:
C:\Program Files (x86)\Courion Corporation\Listeners\Windows
2. Copy the folder to the domain controller or workstation where you want to install the Listener or launch the Listener setup.exe file from the network.
3. Launch the setup.exe file. The Transparent Synchronization Listener for Windows Install Shield Wizard appears as in [Figure 69](#).

Figure 69: Transparent Synchronization Listener Install Shield Wizard

4. Click Next and follow the Installation Wizard instructions.
5. Reboot the computer after the installation is complete.

Configuring the Listener

To configure the listener, from the Microsoft Windows Start menu, select:

Programs>Core Access Assurance Suite>Transparent Synchronization>Transparent Synchronization Listener for Windows

The Transparent Synchronization Listener Configuration dialog box appears as in [Figure 70](#).

Figure 70: Transparent Synchronization Listener Configuration

Transparent Synchronization Listener Configuration

☐ Disable listener

Target ID:

Transparent Synchronization servers (in order of use)

Hostname or IP	Port Number	Timeout

Add... Edit... Remove

Security

Security phrase:

Verify security phrase:

☒ Use Diffie-Hellman
☐ Use certificate

Server CA certificate file path:
 ...

Options

☐ Allow native resets to continue if Transparent Synchronization server connection fails

☒ Trigger Transparent Synchronization on password resets performed administratively

☒ Do not check password strength on password change requests

☒ Do not process account names that end with \$

Filter accounts

Add Remove

OK Cancel About

DISABLE LISTENER — As a precaution, the Listener is installed in a disabled state. Uncheck this box after you have completed configuring the Listener and are ready to use it for password resets.

TARGET ID — This is the name of the computer or domain where you are installing this listener. If it is a computer name, use UNC format (\\mymachine). This must match the name in the Target ID field in the Listener Configuration dialog box of the Transparent Synchronization Configuration Manager (see page 154).

Note: Each server associated with this listener must have the same Target ID.

TRANSPARENT SYNCHRONIZATION SERVERS (IN ORDER OF USE) — This feature allows you to specify multiple Transparent Synchronization servers and the priority of use for these servers. You can associate more than one Transparent Synchronization server with the same listener to provide redundancy in case the primary server fails.

FILTER SERVERS — an administrator can configure multiple regex (Regular Expression) to filter out certain AD accounts from executing a Transparent Synchronization request. Preventing the Transparent Synchronization request for the account solely depends upon the list of RegEx expressions configured in “**Transparent Synchronization Listener Configuration.**”

To add a server:

1. Click the **ADD** button in the Transparent Synchronization servers section.

The Server Configuration dialog box appears as in [Figure 71](#)

Figure 71: Transparent Synchronization Server Configuration

- **HOSTNAME OR IP ADDRESS** — Enter the hostname or IP address of the Transparent Synchronization server associated with the Listener.
- **PORT NUMBER** — Enter the port number on the Transparent Synchronization server where the Listener sends notifications of password resets. This port number must match the port number that you specify under SSL Settings on the Transparent Synchronization Configuration Manager dialog box (see page 154).
- **TIMEOUT (SECONDS)** — How long the Listener waits for communication from the Transparent Synchronization service before it assumes the Transparent Synchronization service is unavailable and logs an error.

2. Click **OK**.

The new server name appears in the list of Transparent Synchronization servers. Repeat this process to add more servers.

When you add more servers, the servers appear in the list in the order that you added them. The primary server is the first in the list. In the event that the listener cannot communicate with this server, the listener automatically begins using the next server in the list. To change the order of the servers in the list, single click on the server name or IP address and click the up or down arrows.

Double clicking on a server in the list causes the Transparent Synchronization Server configuration box to appear.

You can edit server configuration parameters or remove a server from this listener configuration with the **EDIT** and **REMOVE** buttons.

SECURITY — Enter security information.

- **SECURITY PHRASE** — Enter a text string to use as a security phrase (128 characters maximum). This security phrase must match the security phrase that you entered in the Listener Configuration window in the Transparent Synchronization service configuration (see page 154).
- Re-enter the security phrase in the Verify Security Phrase window.
- **USE CERTIFICATE OR USE DIFFIE-HELLMAN** — Select which method of authentication and encryption to use for communication between the Listener and the Transparent Synchronization service. Diffie-Hellman is the default. If you have a digital certificate to use for authentication and encryption between the Listener and the Transparent Synchronization service, select Use Certificate. Use the Server CA Certificate File Path dialog box to locate your certificate.

OPTIONS — Enable or disable password reset options (these options are disabled by default).

- **ALLOW NATIVE RESETS TO CONTINUE IF TRANSPARENT SYNCHRONIZATION SERVER CONNECTION FAILS** — When you select this option, password resets occur on the native operating system even if the listener cannot communicate with the transparent synchronization server. The native strength rules still apply in this case.
- **TRIGGER TRANSPARENT SYNCHRONIZATION ON PASSWORD RESETS PERFORMED ADMINISTRATIVELY** — When you select this option, password synchronization occurs when a system administrator resets a password (not only when a user performs a Ctrl/Alt Delete, changes the password, and verifies the change).
- **DO NOT CHECK PASSWORD STRENGTH ON PASSWORD CHANGE REQUESTS** — When you check this option, the native operating system strength check determines whether the native reset occurs. The Listener does not perform the Core Security password strength check that normally follows the native operating system strength check.

However if you have configured PasswordCourier password strength settings in the PasswordCourier Customization Manager, that check will still occur. Therefore a password change may occur on the domain controller, fail the strength check, and as a result, no synchronization will occur.

- **DO NOT PROCESS ACCOUNT NAMES THAT END WITH \$** — This option allows the filtration of password change notifications for machine accounts in Active Directory® and Microsoft Windows domains.

To complete a Transparent Synchronization operation, entries must exist in the IdentityMap target table for the originating listener as well as the targets to be synchronized. Synchronization does not occur if a Transparent Synchronization listener detects a reset but no corresponding entries exist for the targets to be synchronized. Entries of log messages are made in the Core Security log files following a Transparent Synchronization operation.

If your Active Directory domain or Microsoft Windows domain contains a significant number of machines, then the number of entries logged in the Core Security log files could be significantly high. Since machine accounts end with "\$", enabling this option would eliminate the log entries for machine accounts and substantially reduce the size of the log files.

This option is checked (enabled) by default.

3. Click **OK** when you have completed this dialog box.

4. Add regex expression. Enter the RegEx expression that corresponds with an account naming pattern that the listener should not process.

Figure 72: Transparent Synchronization Listener Configuration - Filter Accounts

The screenshot shows the 'Transparent Synchronization Listener Configuration' dialog box. The 'Filter accounts' section at the bottom is highlighted with a red rectangle. It contains a text input field with the regex expression '[admin][.]*', an 'Add' button, and a 'Remove' button. The 'Options' section above it has four checkboxes: 'Allow native resets to continue if Transparent Synchronization server connection fails' (unchecked), 'Trigger Transparent Synchronization on password resets performed administratively' (checked), 'Do not check password strength on password change requests' (checked), and 'Do not process account names that end with \$' (checked). The 'Security' section on the right includes fields for 'Security phrase' and 'Verify security phrase', radio buttons for 'Use Diffie-Hellman' (selected) and 'Use certificate', and a 'Server CA certificate file path' field with a browse button.

5. Click **Add** to add the configured RegEx expression in to the list of “Filter accounts”.

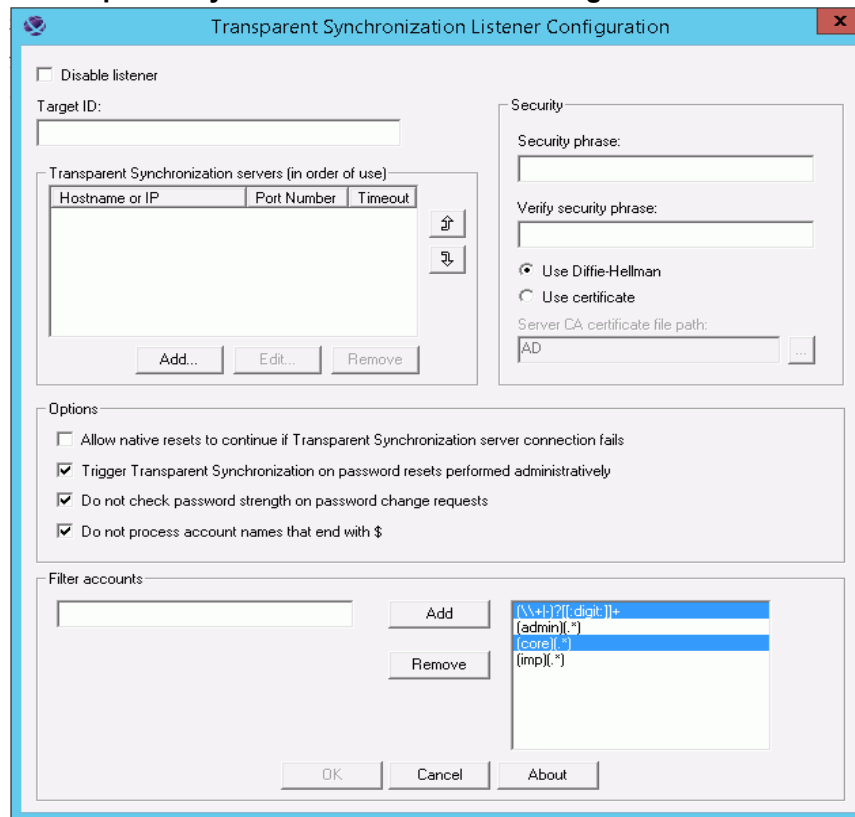
Figure 73: Transparent Synchronization Listener Configuration - Filter Accounts Added

The screenshot shows the 'Transparent Synchronization Listener Configuration' dialog box. It has several sections:

- Disable listener:** A checkbox that is currently unchecked.
- Target ID:** A text input field.
- Transparent Synchronization servers (in order of use):** A table with columns 'Hostname or IP', 'Port Number', and 'Timeout'. Below the table are 'Add...', 'Edit...', and 'Remove' buttons.
- Security:**
 - Security phrase:** A text input field.
 - Verify security phrase:** A text input field.
 - Use Diffie-Hellman:** A radio button that is selected.
 - Use certificate:** A radio button that is unselected.
 - Server CA certificate file path:** A text input field containing 'AD' and a browse button ('...').
- Options:**
 - Allow native resets to continue if Transparent Synchronization server connection fails:** An unchecked checkbox.
 - Trigger Transparent Synchronization on password resets performed administratively:** A checked checkbox.
 - Do not check password strength on password change requests:** A checked checkbox.
 - Do not process account names that end with \$:** A checked checkbox.
- Filter accounts:**
 - A text input field for adding new filter accounts.
 - Add** and **Remove** buttons.
 - A list box containing the entry '(admin)[. *]', which is highlighted by a red rectangle.

At the bottom of the dialog are **OK**, **Cancel**, and **About** buttons.

6. To remove an existing RegEx expression (Multi-select is allowed), select the desired configured RegEx expressions from the list, (*please note that here you can select single or multiple list items at a time*) and click **Remove**.

Figure 74: Transparent Synchronization Listener Configuration - Filter Accounts Removed

The selected RegEx items are removed from the list.

Transparent Synchronization Listener for Microsoft Server Core for Windows Server

The Listener Files

This section describes the Listener files and how to interact with each of them.

The Listener files are located at:

[Core SecurityInstallPath]\Listener\Windows\Windows Core

This folder contains the following files:

- ListenerRegistry.xml
- PSListener.ps1
- setup.iss
- uninstall.iss

ListenerRegistry.xml

Note: Edit this file before running the PSListener.ps1 script.

This file contains all the configuration and registry entry key values that the PSListener.ps1 script needs for installing or uninstalling the Listener setup. By default, all the keys contain the value #ValidData. Replace this value with the correct value for the corresponding key.

The keys from the XML are inserted as registry entries. For a 64 bit system, the entry is into:

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Courion Corporation\Transparent Synchronization Listener for NT\Configuration

For a 32 bit system, the entry is into:

HKEY_LOCAL_MACHINE\SOFTWARE\Courion Corporation\Transparent Synchronization Listener for NT\Configuration

The XML key is as follows:

- **Root:** This key specifies the root path for the Listener setup (.exe) file, which is executed by the PS script generally located at %Install path%\Listener\Windows.

Table 7 indicates the XML keys that correspond to the UI values displayed in [Figure 70](#).

Table 7: XML Keys with the Corresponding UI Values (Sheet 1 of 3)

XML Key	UI Value
Admin Reset	<p>TRIGGER TRANSPARENT SYNCHRONIZATION ON PASSWORD RESETS PERFORMED ADMINISTRATIVELY — When you select this option, password synchronization occurs when a system administrator resets a password (not only when a user performs a Ctrl/Alt Delete, changes the password, and verifies the change).</p> <p>The accepted values include:</p> <p>0: Unchecks the option</p> <p>1: Checks the option</p> <p>Any other value results in failure.</p> <p>For example: <Key Name="Admin Reset" Value="1"></Key></p>
Use Certificate	<p>USE CERTIFICATE or USE DIFFIE-HELLMAN — Select which method of authentication and encryption to use for communication between the listener and the Transparent Synchronization service. Diffie-Hellman is the default. If you have a digital certificate to use for authentication and encryption between the listener and the Transparent Synchronization service, select Use Certificate. The accepted values include:</p> <p>0: Use Diffie-Hellman option is selected</p> <p>1: Use Certificate option is selected</p> <p>Any other value results in failure.</p> <p>For example: <Key Name="Use Certificate" Value="0"></Key></p>

Table 7: XML Keys with the Corresponding UI Values (Sheet 2 of 3)

XML Key	UI Value
CA File Path	<p>If USE CERTIFICATE value is 1, use the Server CA Certificate File Path dialog box to locate your certificate. For example:</p> <pre><Key Name="CA File Path" Value=" C:\CertificateFile.cer"></Key></pre>
Disable	<p>DISABLE LISTENER — The Disable Listener checkbox appears in the upper right corner. By default, it is checked (enabled) the first time you run the Listener configuration. When it is checked, the Listener does not perform password resets. Uncheck this box if you want to activate this Listener.</p> <p>For example: <pre><Key Name="Disable" Value="0"></Key></pre></p>
Disable Strength Check	<p>DO NOT CHECK PASSWORD STRENGTH ON PASSWORD CHANGE REQUESTS — This option ignores the PasswordCourier password strength settings. The accepted values include:</p> <p>0: Unchecks the option</p> <p>1: Checks the option</p> <p>Any other value results in failure.</p> <p>For example: <pre><Key Name="Disable Strength Check" Value="0"></Key></pre></p>
Filter Machine Accounts	<p>DO NOT PROCESS ACCOUNT NAMES THAT END WITH \$ — This option allows the filtration of password change notifications for machine accounts in Active Directory® domains. The accepted values include:</p> <p>0: Unchecks the option</p> <p>1: Checks the option</p> <p>Any other value results in failure.</p> <p>For example: <pre><Key Name="Filter Machine Accounts" Value="0"></Key></pre></p>
Native Reset	<p>ALLOW NATIVE RESETS TO CONTINUE IF TRANSPARENT SYNCHRONIZATION SERVER CONNECTION FAILS — When you select this option, password resets occur on the native operating system even if the listener cannot communicate with the transparent synchronization server. The native strength rules still apply in this case. The accepted values include:</p> <p>0: Unchecks the option</p> <p>1: Checks the option</p> <p>Any other value results in failure.</p> <p>For example: <pre><Key Name="Native Reset" Value="0"></Key></pre></p>

Table 7: XML Keys with the Corresponding UI Values (Sheet 3 of 3)

XML Key	UI Value
Security Phrase	<p>SECURITY PHRASE — Enter a text string to use as a security phrase (128 characters maximum). This security phrase must match the security phrase that you entered in the Listener Configuration window in the Transparent Synchronization service configuration.</p> <p>For example: <code><Key Name="Security Phrase" Value="PassPhrase"></Key></code></p> <p>This key needs an encrypted security phrase for the Listener to work correctly. You can enter the value in plain text in the XML, but you must change it manually from the UI after running the PowerShell script from where it is inserted in an encrypted form in the registry.</p> <p>Or</p> <p>If you have an existing setup of the Transparent Synchronization Listener on another domain controller, copy the encrypted security phrase from the registry and add it to the XML.</p> <p>For example:</p> <pre><Key Name="Security Phrase" Value="GJ4H6SJGJ5KAG6KJ1GSKJA9GKJS0HG"></Key></pre> <p>The registry path:</p> <p>For example: HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Courion Corporation\Transparent Synchronization Listener for NT\Configuration</p>
Server Count	<p>Represents the number of Transparent Synchronization servers that are specified under <code><TServers> </TServers></code>. The value should be an integer.</p> <p>For example: <code><Key Name="Server Count" Value="1"></Key></code></p>
Target Name	<p>TARGET ID — This is the name of the computer or domain on which you are installing this Listener. If it is a computer name, use the UNC format (\\mymachine). This must match the name in the Target ID field in the Listener Configuration.</p> <p>For example: <code><Key Name="Target Name" Value="ADPMMExpress"></Key></code></p>

The XML node `<TServers> </TServers>` represents **TRANSPARENT SYNCHRONIZATION SERVERS (IN ORDER OF USE)** — This feature allows you to specify multiple Transparent Synchronization servers and the priority of use for these servers. You can associate more than one Transparent Synchronization server with the same listener to provide redundancy in case the primary server fails.

Servers can be added under this tag, for example:

```
<TServers>

  <TServer1>

    <Keys>
      <Key Name="Port Number" Value="n"></Key>
      <Key Name="Server Name" Value="x.x.x.x"></Key>
      <Key Name="Timeout" Value="90"></Key>
    </Keys>

  </TServer1>

  <TServer2>

    <Keys>
      <Key Name="Port Number" Value="n"></Key>
      <Key Name="Server Name" Value=" TestServerName "></Key>
      <Key Name="Timeout" Value="90"></Key>
    </Keys>

  </TServer2>

  <TServer3>

    :

  </TServer3>

    :

    So on.

</TServers>
```

Note: where 'n' stands for the port number.

Each server tag has the information keys described as follows:

- **Port Number** — Enter the port number on the Transparent Synchronization server where the Listener sends notifications of password resets. This port number must match the port number that you specify under the SSL Settings on the Transparent Synchronization Configuration Manager.
For example: <Key Name="Port Number" Value="n"></Key>
- **Server Name (Hostname or IP Address)** — Enter the hostname or IP address of the Transparent Synchronization server associated with the Listener.
For example: <Key Name="Server Name" Value="TestServerName"></Key>

- **Timeout (Timeout (seconds))** —The amount of time the Listener waits for communication from the Transparent Synchronization service before it assumes the Transparent Synchronization service is unavailable and logs an error. The value should be an integer. For example: <Key Name="Timeout" Value="90"></Key>

PSListener.ps1

This is the script file which has commands to install or uninstall the Listener on the Server Core for Microsoft Windows Server. The script file accepts three commands:

- `/?`: This Command can be used to get help for script syntax.
For example: `is PS>PSListener.ps1 /?`
- `Install`: This command can be used to trigger the installation of the Listener.
For example: `PS>PSListener.ps1 install 'C:\WindowsCore\ListenerRegistry.xml'`
- `Uninstall`: This command can be used to trigger the uninstallation of the Listener.
Forexample: `PS>PSListener.ps1 uninstall 'C:\WindowsCore\ListenerRegistry.xml'`

Setup.iss

This file is used to specify installation parameters in silent mode.

Uninstall.iss

This file is used to specify uninstallation parameters in silent mode.

Installing the Listener on Microsoft Windows Server Core

1. The Listener software is in the following location: [Core SecurityInstallPath]\Listeners\Windows\WindowsCore
where [Core SecurityInstallPath] represents the default path where the Core Security software is installed: C:\Program Files (x86)\Courion Corporation.
2. Copy the WindowsCore folder to the domain controller where you want to install the Listener.
3. Open the PowerShell command prompt on the Server Core system. For example: C:\Windows\system32>powershell
4. Edit ListenerRegistry.xml entries in the Windows Core folder. For example:
`PS C:> notepad.exe C:\WindowsCore\ListenerRegistry.xml`
5. Run PSListener.ps1 script with the parameter **Install** and the path to the XML file to install the Listener. For example:
`PS>PSListener.ps1 install 'C:\WindowsCore\ListenerRegistry.xml'`
6. Reboot the domain controller after the installation is complete.
7. Navigate to SysWOW64. For example, C:\Windows\SysWOW64) folder by using the command prompt.

8. Run CommConfig.exe and modify the Listener configuration to supply the Security Phrase as shown in [Figure 70](#). Click on the **OK** button to apply the configuration changes.

Modifying the Listener on the Server Core for Microsoft Windows Server

Run CommConfig.exe and modify the Listener configuration.

Uninstalling the Listener on the Server Core for Microsoft Windows Server

1. If the files used during installation of the listener are missing, follow Step 1 to Step 4 as described in [“Installing the Listener on Microsoft Windows Server Core”](#).
2. Run PSListener.ps1 script with the parameter **Uninstall** and the path to the XML file to uninstall the Listener. For example:

```
PS>PSListener.ps1 uninstall 'C:\WindowsCore\ListenerRegistry.xml'
```
3. Reboot the domain controller after the uninstallation is complete.

Transparent Synchronization Listener for i5/OS

The Transparent Synchronization Listener installed on i5/OS® systems detects the native i5/OS operating system's password change events, and propagates them to the Courier Server for synchronization with a range of targets.

Requirements

Requirements for the Transparent Synchronization Listener (TSL) for i5/OS:

- Install i5/OS or OS/400 V5R1 or higher
- Install and configure the Password Management Module (PMM) Agent for i5/OS
- FTP capabilities to the i5/OS system

Installing the TSL for i5/OS

To install the Transparent Synchronization Listener (TSL) for i5/OS, the user profile used in the install process needs to have *SECADM and *ALLOBJ special authorities. Core recommends using QSECOFR during the installation process. To install the Transparent Synchronization Listener for i5/OS:

1. Sign on to the i5/OS as QSECOFR or a user profile with *ALLOBJ and *SECADM authority.
2. End PMM Agent for i5/OS
 - d. Display the PMM Agent for i5/OS menu by entering: ([Figure 75](#))
GO COURAGENT/PMMNU

Figure 75: PMM Agent for i5/OS (OS/400) Menu

```

PMMNU          PMM Agent for OS/400 Menu          System:  OS400A
Select one of the following:

  1. Start PMM Agent for OS/400
  2. End PMM Agent for OS/400
  3. Configure PMM Agent for OS/400
  4. Maintain Excluded Profiles
  5. Purge Message and Output Queues
  6. Print Configuration Files
  7. Print Message Queue
  8. Display Message Queue
  9. Display Output Queue

 90. Sign Off

Selection or command
==> _____

F1=Help  F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel
COPYRIGHT (C) 2002 COURION CORPORATION.
  
```

- e. Select **END PMM AGENT FOR OS/400** and press **ENTER** to confirm.

- f. Enter the following command to ensure that the PMM Agent for i5/OS is not active. ([Figure 76](#) displays the PMM Agent for i5/OS with active jobs).

```
WRKUSRJOB USER (COURAGENT) STATUS (*ACTIVE)
```

Figure 76: PMM Agent for i5/OS (OS/400) Active Jobs

Work with User Jobs					08400A
					02/27/02 15:07:33
Type options, press Enter.					
2=Change		3=Hold	4=End	5=Work with	6=Release
8=Work with spooled files		13=Disconnect		7=Display message	
Opt	Job	User	Type	-----Status-----	Function
—	QJVACMSRV	COURAGENT	BATCHI	ACTIVE	
—	QZRCRSRV	COURAGENT	BATCHI	ACTIVE	
—	SSLAG08191	COURAGENT	BATCH	ACTIVE	CMD-STRPMM
					Bottom
Parameters or command					
==>					
F3=Exit	F4=Prompt	F5=Refresh	F9=Retrieve	F11=Display schedule data	
F12=Cancel	F17=Top	F18=Bottom	F21=Select assistance level		
Intermediate assistance level used.					

3. There should not be any active jobs for COURAGENT ([Figure 77](#))

Figure 77: PMM Agent for i5/OS (OS/400) Inactive Jobs

Work with User Jobs					
				02/27/02	15:53:28
Type options, press Enter.					
2=Change 3=Hold 4=End 5=Work with 6=Release 7=Display message					
8=Work with spooled files 13=Disconnect					
Opt	Job	User	Type	-----Status-----	Function
(No jobs to display)					
					Bottom
Parameters or command					
==>					
F3=Exit	F4=Prompt	F5=Refresh	F9=Retrieve	F11=Display schedule data	
F12=Cancel	F17=Top	F18=Bottom	F21=Select assistance level		

4. Press **F3** until you exit the PMM Agent for i5/OS menu.
5. Enter the following command to launch a Command Entry screen:
- ```
CALL QCMD
```
6. Grant temporary authority
- The COURAGENT user profile temporarily needs authority to the RSTLIB and RST commands.
- Note:** The install process requires access to the COURAGENT user profile.
- Temporarily grant authority to the RSTLIB and RST commands to COURAGENT to perform product restore operations.

```
GRTOBJAUT OBJ(RSTLIB) OBJTYPE(*CMD) USER(COURAGENT)
AUT(*USE)
```

```
GRTOBJAUT OBJ(RST) OBJTYPE(*CMD) USER(COURAGENT)
AUT(*USE)
```

- c. Sign off as QSECOFR or the user ID used to make the changes.

## 7. FTP the save-file (.savf) file to the i5/OS

- a. Sign on to the i5/OS as COURAGENT.

**Note:** The administrator issuing the FTP, RST and RSTLIB commands needs to sign on as COURAGENT, otherwise the system does not set up correctly.

- b. Create a save file in QGPL

```
CRTSAVF QGPL/COURTSL
```

The extracted i5/OS programs are delivered in a \*SAVF file and must be uploaded to the i5/OS system.

- c. Copy the COURTSL.savf file from the Access Assurance Suite install directory (OS400 subfolder) to the root directory of C:\ on your Windows NT/2000/XP system.
- d. Using FTP, enter the following commands from the PC where the COURTSL.savf file is stored and perform the following steps:

```
FTP myAS400ipAddress
User: COURAGENT
Password: (COURAGENT password)
cd QGPL
lcd C:\
binary
put COURTSL.savf COURTSL
quit
```

## 8. Restore the TSL for i5/OS product library

- a. On the i5/OS, signed on as COURAGENT, issue this restore command:

```
RSTLIB SAVLIB(COURTSL) DEV(*SAVF) SAVF(QGPL/COURTSL)
OUTPUT(*PRINT)
```

## 9. Run the TSL for i5/OS installation program

```
CALL COURTSL/INSTALLTSL
```

This step:

- restores the pmm400.jar file
- deletes the QGPL/COURTSL save-file
- grants \*PUBLIC \*USE authority to the product library COURTSL.

- 10. Configure the TSL for i5/OS. See [“Configuring the TSL for i5/OS” on page 176](#). Save your configuration and press **F3** to return to the PMM agent for i5/OS menu.

## 11. Start the PMM Agent for i5/OS

- a. Enter the command: ([Figure 75](#))

```
GO COURAGENT/PMMNU
```

- b. Select **START PMM AGENT FOR OS/400**

This command submits the PMM Agent for i5/OS to the job queue you specified in the configuration.

## 12. Revoke Temporary Authority

After installing the TSL for i5/OS, you can remove the temporary authority you granted to the RSTLIB and RST commands.

- a. Sign off as the user COURAGENT
- b. Sign on again as QSECOFR, or a user profile with \*ALLOBJ and \*SECADM authority.

- c. Revoke the authority you gave to COURAGENT to restore objects:

```
RVKOBJAUT OBJ(RSTLIB) OBJTYPE(*CMD) USER(COURAGENT)
AUT(*ALL)

RVKOBJAUT OBJ(RST) OBJTYPE(*CMD) USER(COURAGENT)
AUT(*ALL)
```

### 13. Configure the i5/OS system values

The instructions below include details about how to set the QPWDVLDPGM system value to \*REGFAC, and how to add the exit point programs using the WRKREGINF command.

- a. Signed on as QSECOFR or a user profile with \*ALLOBJ and \*SECADM authority, display the current system value:

```
DSPSYSVAL SYSVAL(QPWDVLDPGM)
```

- b. Save the displayed system value for your records.

- c. If the QPWDVLDPGM system value is not already set to \*REGFAC, then change it:

```
CHGSYSVAL SYSVAL(QPWDVLDPGM) VALUE(*REGFAC)
```

- d. Work with registration information for QIBM\_QSY\_VLD\_PASSWORD exit point: [\(Figure 78\)](#)

```
WRKREGINF EXITPNT(QIBM_QSY_VLD_PASSWORD)
```

**Figure 78: Work with Registration Information**

| Work with Registration Information                  |                       |                   |            |                   |
|-----------------------------------------------------|-----------------------|-------------------|------------|-------------------|
| Type options, press Enter.                          |                       |                   |            |                   |
| 5=Display exit point    8=Work with exit programs   |                       |                   |            |                   |
| Opt                                                 | Exit Point            | Exit Point Format | Registered | Text              |
| 8                                                   | QIBM_QSY_VLD_PASSWORD | VLDP0100          | *YES       | Validate Password |
| <div style="text-align: right;"><b>Bottom</b></div> |                       |                   |            |                   |
| Command                                             |                       |                   |            |                   |
| ===>                                                |                       |                   |            |                   |
| F3=Exit                                             | F4=Prompt             | F9=Retrieve       | F12=Cancel |                   |

e. Select option 8 = **WORK WITH EXIT PROGRAMS**. ([Figure 79](#))

Add the exit point programs TSLPSR and TSLTSR. If other exit point programs exist, then change the keyword PGMNBR() sequence so that the TSLPSR exit point program executes before any other exit point program, and TSLTSR executes after all other exit point programs.

**Figure 79: Work with Exit Programs**

| Work with Exit Programs                             |                     |                  |         |
|-----------------------------------------------------|---------------------|------------------|---------|
| Exit point: QIBM_QSY_VLD_PASSWORD                   |                     | Format: VLDP0100 |         |
| Type options, press Enter.                          |                     |                  |         |
| 1=Add 4=Remove 5=Display 10=Replace                 |                     |                  |         |
| Opt                                                 | Exit Program Number | Exit Program     | Library |
| —                                                   | 30                  | CHGPWDEXIT       | QGPL    |
| —                                                   |                     |                  |         |
| Command                                             |                     |                  | Bottom  |
| ===>                                                |                     |                  |         |
| F3=Exit F4=Prompt F5=Refresh F9=Retrieve F12=Cancel |                     |                  |         |

In the example below, the TSLPSR and TSLTSR exit point programs are added, and the keyword PGMNBR() sequence is set to 20 and 40 respectively.

Enter the following two commands at the command line:

```
ADDEXITPGM EXITPNT(QIBM_QSY_VLD_PASSWORD) FORMAT(VLDP0100)
PGMNBR(20) PGM(COURTSL/TSLPSR)
TEXT('TSL for OS/400 Password Strength Request')
ADDEXITPGM EXITPNT(QIBM_QSY_VLD_PASSWORD) FORMAT(VLDP0100)
PGMNBR(40) PGM(COURTSL/TSLTSR)
TEXT('TSL for OS/400 Transparent Sync Request')
```

**Note:** The TSLPSR is the exit point program to indicate a Password Strength Request (PSR). The TSLTSR is the exit program to indicate a Transparent Sync Request (TSR).

14. Sign off as QSECOFR. The TSL for i5/OS configuration is now complete.

### Configuring the TSL for i5/OS

Once the Transparent Synchronization Listener (TSL) for i5/OS has been installed, it is ready for configuration. To configure the listener, you must be signed on as COURAGENT. Enter the command:

```
GO COURTSL/TSLMNU
```

The TSL for i5/OS Menu appears as in [Figure 80](#).



## TSL for i5 /OS (OS/400) Menu

**Figure 80: TSL for 5/OS (OS/400) Menu**

```

TSLMNU TSL for OS/400 Menu System: XXXXXXXX

Select one of the following:

 1. Maintain the TSL configuration
 2. Maintain TSL specific excluded profiles
 3. Purge log file
 4. Print the TSL configuration
 5. Print log file
 6. Display output queue

 90. Sign Off

Selection or command
==> _____

F1=Help F3=Exit F4=Prompt F9=Retrieve F12=Cancel
COPYRIGHT (C) 2004 COURION CORPORATION.

```

In [Figure 80](#), you can either select the menu option or type the command for each option at the command line. See Table 8 for the menu options and the corresponding commands to execute.

## TSL for i5/OS Menu Commands

**Table 8: TSL for i5/OS Menu Commands**

| Option # | Menu Options                            | Command to Execute |
|----------|-----------------------------------------|--------------------|
| 1        | Maintain the TSL Configuration          | TSLMNTCFG          |
| 2        | Maintain TSL Specific Excluded Profiles | CALL TSLMNTEXPC    |
| 3        | Purge Log File                          | TSLPRGLOG          |
| 4        | Print the TSL Configuration             | TSLPRTCFCG         |
| 5        | Print Log File                          | TSLPRTLLOG         |
| 6        | Display Output Queue                    | WRKOUTQ TSLOUTQ    |
| 90       | Sign Off                                | SIGNOFF            |

## Maintain the TSL Configuration

When you select **MAINTAIN THE TSL CONFIGURATION** in [Figure 80](#), the configuration screen appears as in [Figure 81](#). This screen allows you to configure the TSL for i5/OS.

**Figure 81: TSL for i5/OS Configuration (1 of 3)**

```

 Maintain the TSL Configuration (TSLMNTCFG)

Type choices, press Enter.

Library where installed > COURTSL Name
TSL enabled *YES *YES, *NO
Trace enabled *NO *YES, *NO
Use PMM excluded profiles . . . *NO *YES, *NO
Process native if connect fail *YES *YES, *NO
Enforce password rules in PWC . *YES *YES, *NO
TSL client settings:
 Port 08902 1024-65536
 Timeout 00010 SECONDS
 Retry attempts 01 01-99

 More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

```

**LIBRARY WHERE INSTALLED** — Displays the name of the current library where the product TSL for i5/OS is installed. This is a read-only field.

**TSL ENABLED** — This option allows you to decide whether or not to allow the TSL for i5/OS to process password change requests. The default value for this option is **\*NO**.

**TRACE ENABLED** — This option allows you to decide whether or not to allow the Transparent Synchronization Listener (TSL) Validate Password Exit Program to produce trace level debugging information. The default value for this option is **\*NO**.

**USE PMM EXCLUDED PROFILES** — This option allows you to decide whether or not to allow the TSL Validate Password Exit Program to pass requests to change passwords for the profiles that were specified for exclusion on the PMM Agent for i5/OS on to the Transparent Synchronization Service. The default value for this option is **\*NO**.

If you select **\*YES**, the TSL Validate Password Exit Program checks for profiles that were excluded on the TSL for i5/OS, and also checks for user profiles that were excluded on the PMM Agent for i5/OS.

If you select **\*NO**, only the profiles specified on the TSL for i5/OS are excluded.

**PROCESS NATIVE IF CONNECT FAILS** — This option controls what happens to a password change in the event of a communications failure. The default value for this option is **\*YES**.

If you select **\*YES**, then any communications failure between the TSL Validate Password Exit Program and the Listener for i5/OS Communication Module, or the Listener for i5/OS Communication Module and the Transparent Synchronization Service still processes a native password change successfully in i5/OS.

If you select **\*NO**, then any communications failure between the TSL Validate Password Exit Program and the Listener for i5/OS Communication Module, or the Listener for i5/OS Communication Module and the Transparent Synchronization Service does not process a native password change in i5/OS.

**ENFORCE PASSWORD RULES IN PWC** — This option controls whether or not to enforce the password strength rules specified in PasswordCourier when changing a password. The default value for this option is **\*YES**.

If you select **\*YES**, the PasswordCourier password strength rules are used before changing a password. If you select **\*NO**, the PasswordCourier password strength rules are not applied for a password change. The **NO** option can speed up processing of password changes.

**Note:** The TSL for i5/OS detects the native password change event on the i5/OS system. Upon detecting this password change, the TSL Validate Password Exit Program captures the username and password information and begins the communication exchange to the Transparent Synchronization Service. Before the TSL Validate Password Exit Program communicates the exchange to the Transparent Synchronization Service, the password may be forced to uppercase depending on the password level (QPWDLVL). If your i5/OS QPWDLVL system value is set to 0 or 1, then all i5/OS passwords are captured in uppercase, regardless of how the user has entered the password. If the QPWDLVL system value is set to 2 or 3, then all i5/OS passwords are captured in mixed case (upper and lower) — exactly based on the case the user has entered.

For example, if you select **\*YES**, the password is checked against the PasswordCourier password strength rules. If a password strength rule says that one or more characters must be in lowercase, then that rule does not pass if QPWDLVL is set to 0 or 1. You must give special consideration when configuring Password Strength checks, since the QPWDLVL system value determines if lowercase is allowed in i5/OS passwords. For more information on QPWDLVL, see the appropriate i5/OS manual.

**TSL CLIENT SETTINGS** — This option is specific to the TSL Validate Password Exit Program and the Listener for i5/OS Communication Module.

- **Port** — The port on which the Listener for i5/OS Communication Module talks to the TSL Validate Password Exit Program. The default value is 08902.
- **Timeout** — The time out in seconds for the Listener for i5/OS Communication Module when it attempts to communicate with the TSL Validate Password Exit Program. The default value is 00010.
- **Retry attempts** — The number of attempts the TSL Validate Password Exit Program makes to communicate with the Listener for i5/OS Communication Module. The default value is 01.

**Note:** If you make any changes to the TSL client settings, then you must stop and restart the PMM Agent for i5/OS.

Press the **PAGE DOWN** key on your keyboard. The second screen for the menu option **MAINTAIN THE TSL CONFIGURATION** appears as in [Figure 82](#).

**Figure 82: TSL for i5/OS Configuration (2 of 3)**

```

 Maintain the TSL Configuration (TSLMNTCFG)

Type choices, press Enter.

TSL server settings:
Server name - '192.168.1.5'

Port 26000 1024-65536
Timeout 00005 SECONDS
 + for more values _

```

**TSL SERVER SETTINGS** — This option allows you to specify values for the Transparent Synchronization Service.

**Note:** You can provide information for up to 10 servers. If you enter details for a particular server, then all the of the options must be filled.

- **SERVER NAME** — This is the host name or the IP address that the Listener for i5/OS Communication Module uses to communicate with this server.
- **PORT** — The port on which the Listener for i5/OS Communication Module communicates with the Transparent Synchronization Service. You can configure up to 10 ports.
- **TIMEOUT** — The time out in seconds for the Listener for i5/OS Communication Module when it attempts to communicate with the Transparent Synchronization Service. You can configure up to 10 timeouts.

**Note:** If you make any changes to the TSL server settings, then you must stop and restart the PMM Agent for i5/OS.

Press the **PAGE DOWN** key on your keyboard. The third screen for the menu option **MAINTAIN THE TSL CONFIGURATION** appears as in [Figure 82](#).

**Figure 83: TSL for i5/OS Configuration (3 of 3)**

```

 Maintain the TSL Configuration (TSLMNTCFG)

Type choices, press Enter.

Target ID server5

Security phrase

Verify security phrase

F3=Exit F5=Refresh F12=Cancel

```

**TARGET ID** — Enter the name of the system where the listener is installed. The system name must match the name entered in the Target ID field in the Listener Configuration dialog box of the Transparent Synchronization Configuration Manager. For more information about the Transparent Synchronization Configuration Manager, please see [“Installing the Transparent Synchronization Service” on page 152](#).

**Note:** Each server name associated with this listener must have the same Target ID.

**Note:** If you make any changes to the target id, then you must stop and restart the PMM Agent for i5/OS.

**SECURITY PHRASE** — Enter a text string to use as a security phrase (128 characters maximum). This security phrase must match the security phrase that you enter in the Listener Configuration window in the Transparent Synchronization Listener Configuration dialog box. For more information about the Transparent Synchronization Configuration Manager, please see [“Installing the Transparent Synchronization Service” on page 152](#).

**Note:** If you change the **SECURITY PHRASE** entry, another field appears which allows you to re-enter the security phrase to verify that it has been entered correctly.

**Note:** If you make any changes to the security phrase, stop and restart the PMM Agent.



## Purge Log File

When you select **PURGE LOG FILE** in [Figure 80](#), the following screen appears ([Figure 85](#)).

This option allows you to purge some or all of the entries from the TSL for i5/OS Log file (TSLLOG).

**Figure 85: Purge Log File**

**Purge Log File (TSLPRGLOG)**

Type choices, press Enter.

|                                       |                |
|---------------------------------------|----------------|
| #Days to retain information . . . 7   | 1 - 365, *NONE |
| Reorganize file after purge . . . *NO | *YES, *NO      |

**Bottom**

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display  
F24=More keys

**DAYS TO RETAIN INFORMATION** — Select **\*NONE** to purge all records in the log file. If you enter a number from 1 through 365, the log file retains log entries for those many days going back from the current date.

**Note:** If you make any changes to the number of days to retain information, stop and restart the PMM Agent for i5/OS.

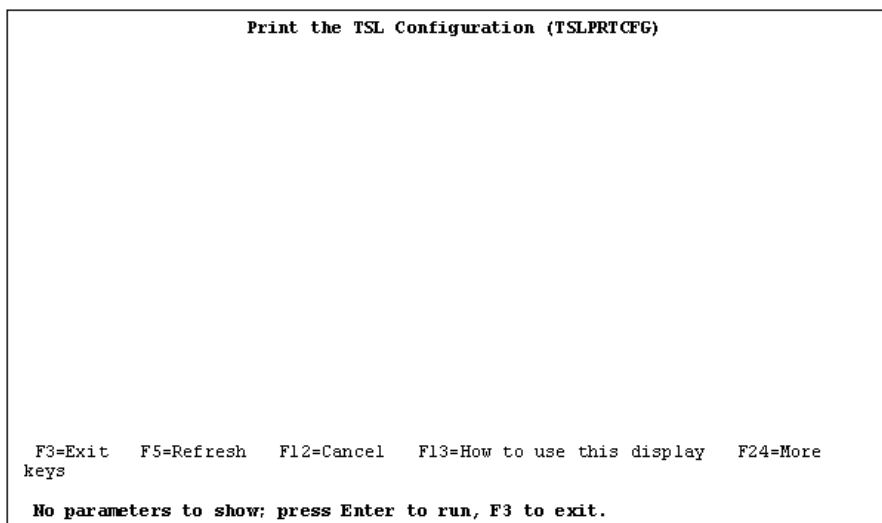
**REORGANIZE FILE AFTER PURGE** — This option controls whether or not the TSL for i5/OS log file is reorganized after the records were purged. Select **\*YES** if you want to reorganize the log file.

## Print TSL Configuration

When you select **PRINT THE TSL CONFIGURATION** in [Figure 80](#), the following screen appears ([Figure 86](#)).

This option creates a listing that shows the current TSL for i5/OS configuration parameters and their associated default values.

**Figure 86: Print the TSL Configuration**





## Print Log File

When you select **PRINT LOG FILE** in [Figure 80](#), the following screen appears ([Figure 87](#)).

**Figure 87: Print Log File**

```

Print Log File (TSLPRTL06)

Type choices, press Enter.

Start Time and Date:
 Begin Time *AVAIL Time, *AVAIL
 Begin Date *CURRENT Date, *CURRENT, *BEGIN
End Time and Date:
 End Time *AVAIL Time, *AVAIL
 End Date *CURRENT Date, *CURRENT, *END

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
Bottom

```

**START TIME AND DATE** — This option allows you to include log entries on the print log file list that were created on or after the date and time specified.

- **BEGIN TIME** — Select **\*AVAIL** if you want to include all of the logged entries for the specified starting date, or you can enter the starting time for the specified starting date. The time is specified in a 24-hour format.
- **BEGIN DATE** — Select **\*CURRENT** to include all of the logged entries for the current date, and those that get logged between the starting and ending times (if specified).

Select **\*BEGIN** to include log entries from the beginning of the log.

Or, you may specify the starting date in the job date format with or without a separator.

**Note:** If you make any changes to the start time and date, then you must stop and restart the PMM Agent for i5/OS.

**END TIME AND DATE** — This option allows you to include log entries on the print log file list that were created on or before the date and time specified.

- **END TIME** — Select **\*AVAIL** if you want to include all of the logged entries for the specified ending date, or you can enter the ending time for the specified ending date. The time is specified in a 24-hour format.
- **END DATE** — Select **\*CURRENT** to specify that the current date is the last day for which logged entries are to be included on the list.

Select **\*END** to include entries of the last day on which the entries were logged.

Or, you may specify the ending date in the job date format with or without a separator.

**Note:** If you make any changes to the end time and date, then you must stop and restart the PMM Agent for i5/OS.

## Display TSL Output Queue

When you select **DISPLAY TSL OUTPUT QUEUE** in [Figure 80](#), the following screen appears ([Figure 88](#)).

**Figure 88: Display TSL Output Queue**

```

 Work with Output Queue
Queue: TSLOUTQ Library: COURTSL Status: RLS

Type options, press Enter.
 1=Send 2=Change 3=Hold 4=Delete 5=Display 6=Release 7=Messages
 8=Attributes 9=Work with printing status

Opt File User User Data Sts Pages Copies Form Type Pty
-- -
 - TSLPRTLOGP COURAGENT TSLPRTLOGR RDY 2 1 *STD 5
 - TSLPRTCFGP COURAGENT TSLPRTCFGR RDY 1 1 *STD 5

Parameters for options 1, 2, 3 or command
====>
F3=Exit F11=View 2 F12=Cancel F20=Writers F22=Printers
F24=More keys
Bottom

```

This option displays a standard i5/OS screen from where you can print the COURTSL output queue spool file entries (TSLOUTQ).

## Uninstalling the TSL for i5/OS

To uninstall the Transparent Synchronization Listener (TSL) for i5/OS product library, you must sign on to the i5/OS as QSECOFR or a user profile with \*ALLOBJ, \*SECADM, and \*SPLCTL authority. You must have access to a 5250 emulation session.

**Note:** The uninstall procedure removes the TSL for i5/OS product library, but leaves the Listener for i5/OS Communication Module intact.

### 1. Remove TSL for i5/OS commands

If you have added any TSL for i5/OS commands to the QSTRUP program or to the i5/OS Job Scheduled (WRKJOBSCDE), then you must remove these entries before proceeding with uninstall. Look for any of the following commands and remove them:

COURTSL/TSLMNTCFG (TSL Configuration Maintenance)

COURTSL/TSLPRGLOG (Purge Transparent Sync Log)

COURTSL/TSLPRTCFG (Print TSL Configuration)

COURTSL/TSLPRTLOG (Print Transparent Sync Log)

**Note:** You must compile the QSTRUP program, if you removed any of the above TSL for i5/OS commands.

### 2. End PMM Agent for i5/OS

a. Display the PMM Agent for i5/OS menu by entering: ([Figure 75](#))

```
GO COURAGENT/PMMNNU
```

b. Select **END PMM AGENT FOR OS/400** and press **ENTER** to confirm.

- c. Enter the following command to ensure that the PMM Agent for i5/OS is not active: See [Figure 76](#) and [Figure 77](#).

```
WRKUSRJOB USER(COURAGENT) STATUS(*ACTIVE)
```

3. Remove the TSL Validate Password Exit Program

**Note:** You must be signed on as QSECOFR or a user profile with \*ALLOBJ, \*SECADM, and \*SPLCTL authority,

- a. Enter the following command to work with registration information for QIBM\_QSY\_VLD\_PASSWRD exit point: ([Figure 89](#))

```
WRKREGINF EXITPNT(QIBM_QSY_VLD_PASSWRD)
```

**Figure 89: Work with Exit Programs**

| Work with Exit Programs             |                     |                  |                        |
|-------------------------------------|---------------------|------------------|------------------------|
| Exit point: QIBM_QSY_VLD_PASSWRD    |                     | Format: VLDP0100 |                        |
| Type options, press Enter.          |                     |                  |                        |
| 1=Add 4=Remove 5=Display 10=Replace |                     |                  |                        |
| Opt                                 | Exit Program Number | Exit Program     | Library                |
| —                                   | 20                  | TSLPSR           | COURTSL                |
| —                                   | 30                  | CHGPWDEXIT       | QGPL                   |
| —                                   | 40                  | TSLTSR           | COURTSL                |
|                                     |                     |                  | <b>Bottom</b>          |
| Command                             |                     |                  |                        |
| ==>                                 |                     |                  |                        |
| F3=Exit                             | F4=Prompt           | F5=Refresh       | F9=Retrieve F12=Cancel |

- b. Select option 4 = **REMOVE** to remove the TSLPSR and TSLTSR exit point programs
  - Note:** The TSLPSR is the exit point program to indicate a Password Strength Request (PSR). The TSLTSR is the exit program to indicate a Transparent Sync Request (TSR).
  - c. Press **ENTER** to confirm the removal
  - d. Press **F3** to return to the previous menu
4. Delete TSL for i5/OS Product Library

**Note:** You must be signed on as QSECOFR or a user profile with \*ALLOBJ, \*SECADM, and \*SPLCTL authority,

- a. Enter the following command to clear the TSLOUTQ entries.

```
CLROUTQ COURTSL/TSLOUTQ
```

- b. Delete the COURTSL product library

```
DLTLIB COURTSL
```

- c. Sign off as QSECOFR

The TSL for i5/OS product library is now uninstalled. At this point you can restart the PMM Agent for i5/OS to resume PasswordCourier processing.

*Notes and Warnings*

- Password changes using the CHGPWD command and the Change User Password (QSYCHGPW) API are supported. Password changes using the CHGUSRPRF command are not supported.
- Passwords are forced to uppercase when i5/OS QPWDLVL is set to 0 or 1. Passwords are not forced to uppercase with QPWDLVL set to 2 or 3.

## Chapter 13: Using the Configuration Repository

---

The Core Access Assurance Suite Connector Configuration Manager includes a Configuration Repository option that allows you to share common volatile types of configuration data on your Core Server, such as disabled users lists, ticket IDs, and email IDs with other Core Servers in the network. The shared configuration data is stored in a transaction repository database on an SQL server. You can choose to share data from your Core Server or to not share data.

This chapter includes the following information about using the Configuration Repository:

- [\*“About Sharing Data” on page 186\*](#)
- [\*“Using the Configuration Management Wizard” on page 188\*](#)
- [\*“Notes on Running the Core Server in a Shared Environment” on page 193\*](#)

### Configuring a Transaction Repository

Configure a transaction repository target using the Microsoft-ADO-3.0 connector. The target system appears on the Remote mode drop-down list when you use the Configuration Management wizard (see [Figure 92](#)). See the manual *Configuring Password Management Modules (PMMs), Connectors, and Agents* for information about how to configure connectors, including the Microsoft-ADO-3.0 connector.

## About Sharing Data

The data you can share includes:

- Disabled Users Lists for the Provisioning Platform (AccountCourier, ComplianceCourier, PasswordCourier, and ProfileCourier), and the Classic Platform (PasswordCourier Classic, PasswordCourier Support Staff Classic, ProfileCourier Classic)
- Email IDs on the Classic Platform
- Peregrine start ticket IDs for AccountCourier (Provisioning Platform)
- ODBC ticket IDs for PasswordCourier Classic, PasswordCourier Support Staff Classic, ProfileCourier Classic, and CertificateCourier
- LDAP ticket IDs for PasswordCourier Classic, PasswordCourier Support Staff Classic, ProfileCourier Classic, and CertificateCourier

You can choose not to share this data by running the Configuration Repository in Local mode, or to share it with other Core Servers on the network by running the Configuration Repository in Remote mode.

**Note:** When you run in Local mode, shared data is stored with non-shared data in the `CfgFile.db` file in the `CourionService` folder. For information about using the Core Access Assurance Suite Administration Manager Archive option to archive local configuration data, see the manual *Configuring Workflows with the Core Access Assurance Suite Administration Manager*.

### Running Configuration Repository in Local Mode

When you run the Configuration Repository in Local mode, shared data is stored with non-shared data in the `CfgFile.db` file on the Core Security Server in the following location:

```
Core Security>CourionService>CfgFile.db
```

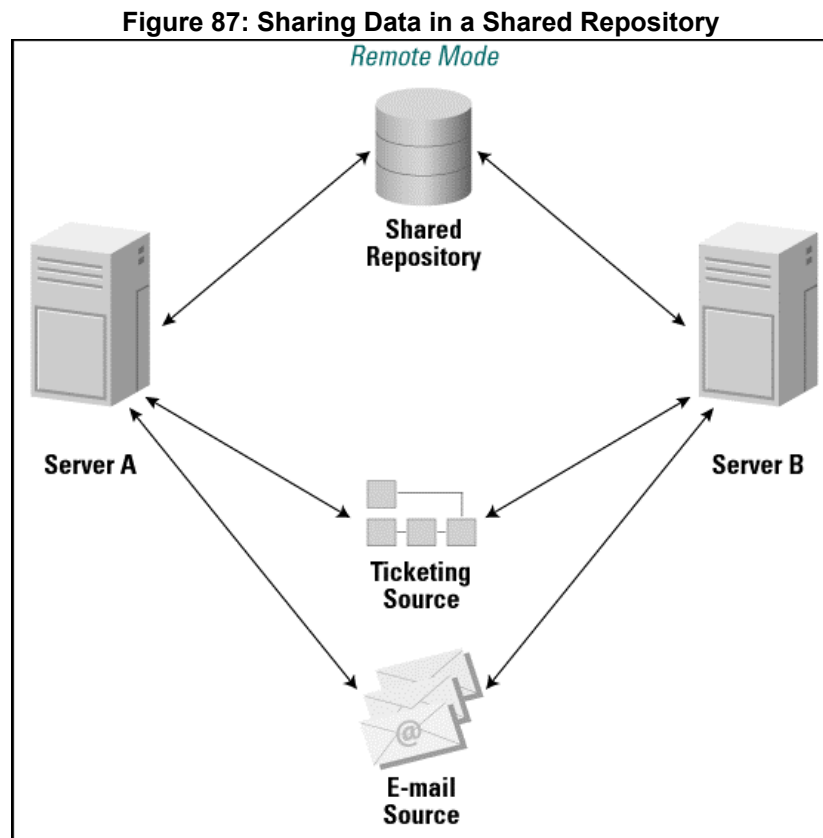
The Core Server updates the shared data in the `CfgFile.db` file when disabled users lists are updated, Emails are generated, and new tickets are created. The data in `CfgFile.db` is not shared and cannot be accessed by other Core Security Servers.

### Running the Configuration Repository in Remote Mode

When you run the Configuration Repository in Remote mode, shared data is transmitted to a shared repository on the network as disabled users lists are updated, Emails are generated, and new tickets are created. When running in Remote mode, the Core Security Server is “attached” to the shared repository with other Core Security Servers in the network. (The shared repository is the transaction repository you configured using the Microsoft-ADO-3.0 connector with the Connector Configuration Manager.)

**Note:** You cannot delete the `CfgFile.db` file on the Core Security Server even if you choose to share data in the shared repository, as non-shared data is still stored in this file.

[Figure 87](#) shows an example of a network with two Core Servers, an SQL server with a shared repository, a ticketing source and an email source.



If user John Smith is disabled on Core Server A, for example, his disabled user status is transferred and stored in the shared repository. If he attempts to log in to Core Server B, he is prevented from doing so because Core Server B is sharing the disabled users list on the shared repository.

### *Transferring Shared Data from the Core Server To the Shared Repository*

When you switch to running in Remote mode, you have the option to transfer shared data from your local Core Server to the shared repository. When you do this, it overwrites any shared data currently in the shared repository. Or, if yours is the first Core Server to join a network sharing data in a shared repository, you can transfer data to populate the shared repository database.

### *Backing up Shared Data on the Shared Repository*

Core Security recommends that the SQL Server administrator use native SQL utilities to periodically back up the shared repository with the shared configuration data. If the shared repository should become unavailable for some reason, it can be restored from the backed-up copy.

## Using the Configuration Management Wizard

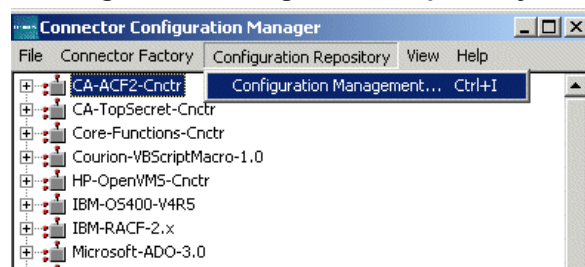
Use the Configuration Management Wizard to configure Local mode or Remote mode for storing shared configuration information.

To access the Configuration Management Wizard from the Start menu, select:

Start>All Programs>Core Core Access Assurance Suite>Connector Configuration Manager

From the Connector Configuration Manager menu, click on **CONFIGURATION REPOSITORY**, and select **CONFIGURATION MANAGEMENT** (see [Figure 88](#)).

**Figure 88: Configuration Repository**



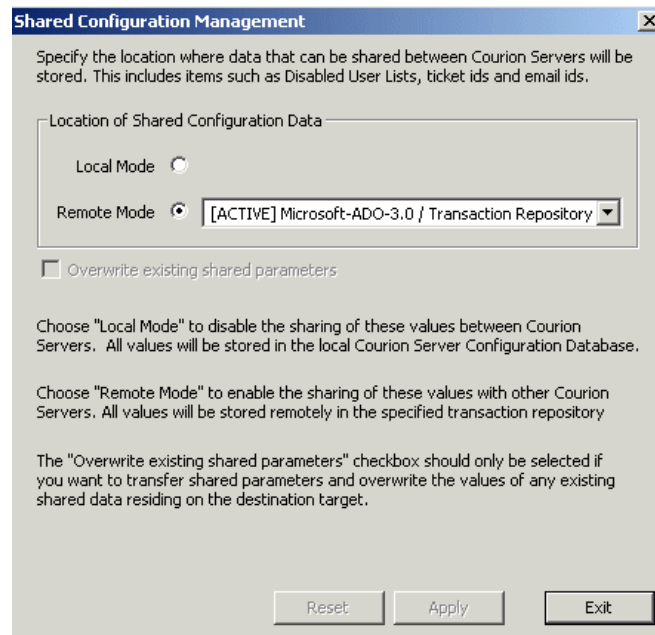
The Configuration Management Wizard appears with your current settings as the default. The first time you use the Configuration Management Wizard, Local mode is the default and you do not need to change this setting if you want to use Local mode to store shared data.

## Configuring Local Mode

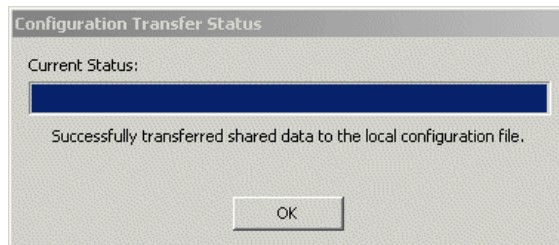
Local mode stores shared data in the `CfgFile.db` file on the local Core Server and does not share data with other servers on the network. When you first use the Configuration Repository after installing the Core Server, Local mode is the default.

[Figure 89](#) shows the Configuration Management Wizard previously configured in Remote mode.



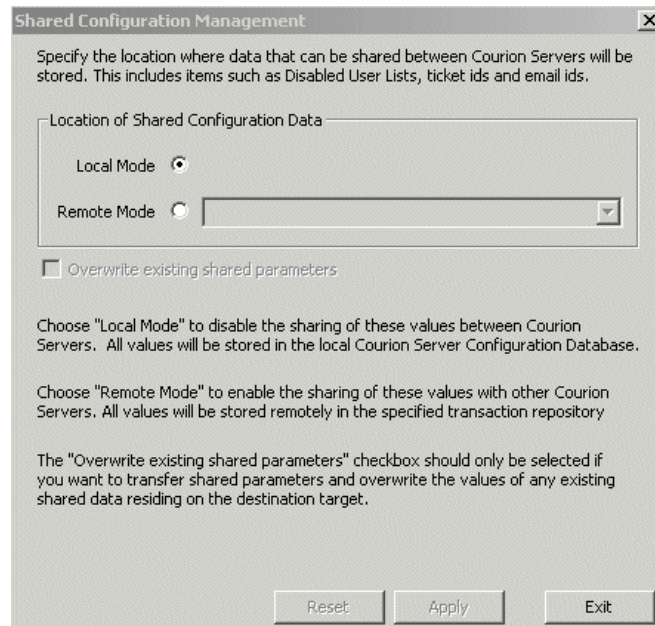
**Figure 89: Configuration Management Wizard (configured in Remote mode)**

1. To run in Local mode, select the **LOCAL MODE** radio button, and click **APPLY**. The wizard displays a prompt asking if you want to continue switching to Local mode.
2. Click **YES** to Continue. Click **NO** to stop the transition to Local mode. When you click Yes, the Configuration Transfer Status screen appears, as in [Figure 90](#).

**Figure 90: Configuration Status Screen (Local mode)**

During this transition, the configuration manager transfers shared data from the remote database file to your local system.

3. Click **OK**. The Configuration Management Wizard appears as in [Figure 91](#).

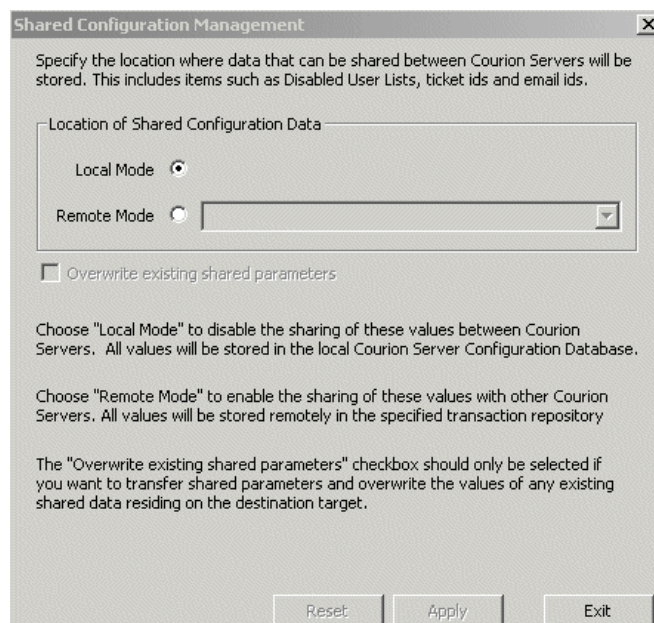
**Figure 91: Local Mode Configuration Complete**

4. Click **EXIT** to exit from the Configuration Management Wizard.

## Configuring Remote Mode

Remote mode stores shared data on the remote shared repository you specify. If you select the Overwrite existing shared parameters option, the shared data on the local Core Server overwrites the data in the shared repository when you switch to Remote mode. If you do not select this option, the Core Server attaches to the shared repository without overwriting data there.

[Figure 92](#) shows the Configuration Management Wizard previously configured in Local mode.

**Figure 92: Configuration Management Wizard (configured in Local mode)**

1. To run in Remote mode, select the **REMOTE MODE** radio button. When you do, the Remote mode drop-down list of transaction repository targets appears. This list includes all transaction repository targets you have configured through the Connector Configuration Manager.
2. Select a shared repository from the list of targets in the drop-down list. This is the location where the Configuration Manager stores the shared data.
3. Select the **OVERWRITE EXISTING SHARED PARAMETERS** option if you want the shared parameters on your local system to overwrite shared parameters currently stored in the shared repository.

Click **APPLY**. The wizard displays this prompt if you do not select the Overwrite existing shared parameters option:

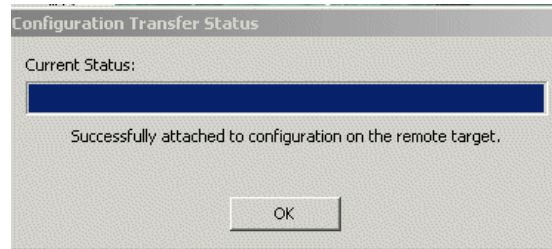
**IF YOU CONTINUE, ALL SHARED DATA CONFIGURATION WILL BE OBTAINED FROM THE SPECIFIED LOCATION OR TARGET. DO YOU WISH TO CONTINUE?**

The wizard displays this prompt if you do select the Overwrite existing shared parameters option:

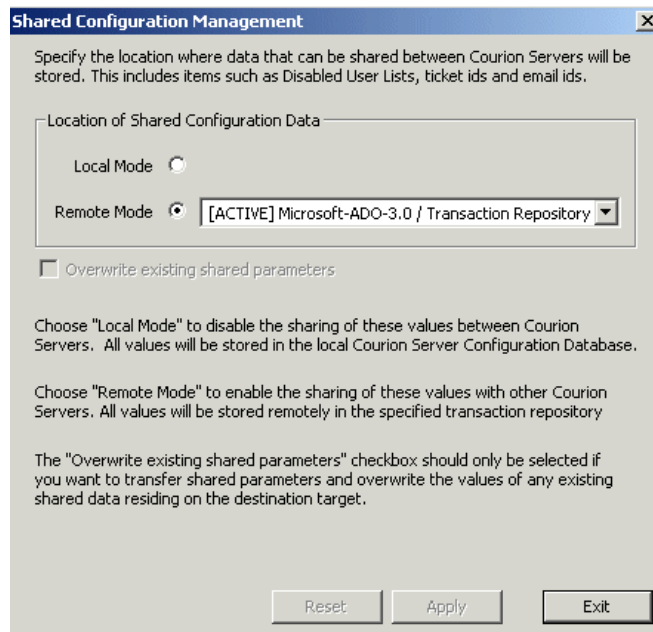
**ALL EXISTING SHARED DATA ON THE SPECIFIED TARGET WILL BE OVERWRITTEN. THIS WILL IMPACT ALL SERVERS USING THIS TARGET FOR THEIR SHARED DATA CONFIGURATION. DO YOU WANT TO CONTINUE?**

4. Click **YES** to Continue. Click **NO** to stop the transition to Remote mode. When you click Yes, the Configuration Transfer Status screen appears. When the process is complete, the status appears as in [Figure 93](#) which shows an “attached” status (not using the Overwrite existing shared parameters option).

**Figure 93: Configuration Status Screen (Remote mode)**



5. Click **OK**. The Configuration Management Wizard appears as in [Figure 94](#).

**Figure 94: Remote Mode Configuration Complete**

The Configuration Management Wizard screen displays the transaction repository you selected on the remote system and marks it as **[ACTIVE]**.

# Notes on Running the Core Server in a Shared Environment

## *When Configuring a Core Server in a Shared Environment*

- Do not run in Remote mode until applications are configured. At the minimum, configure ticketing and email for PasswordCourier Classic, ProfileCourier Classic, and AccountCourier if you are using Peregrine Ticketing.
- Test the configurations to ensure that they work properly.
- If this is the first Core Server in the shared environment, you can select the Overwrite existing shared parameters option when switching to Remote mode to populate the database.
- If other Core Servers are already running in the shared environment, do not select the Overwrite existing shared parameters option when you switch to Remote mode unless you want the shared data on your local system to overwrite the shared data currently stored in the shared repository.

## *When Making Modifications To Applications in a Shared Environment*

- If you need to make modifications to PasswordCourier Classic or ProfileCourier Classic, switch to Local mode before you make the modifications.
- If you need to make modifications to Peregrine Connector targets, switch to Local mode before you make the modifications.
- Test the changes to make sure they are correct before you switch to Remote mode.



## Chapter 14: Input Validation for End-User Workflows

---

The Access Assurance Suite can validate input into end-user workflows and reject Cross Site Scripting (XSS) and SQL injection attacks with the server-side input validation framework. The server-side input validation framework detects signs of XSS and SQL injection attacks using two scanners:

- The XSS scanner
- The SQL scanner

This chapter describes how to enable server-side input validation, configure settings that affect the behavior of the server-side input validation feature, and translate input validation policy violations using the Multi-Language Framework:

- [\*“Enabling Server-Side Input Validation” on page 196\*](#)
- [\*“Configuring Options in the CustomInputValidation.xml File” on page 197\*](#)
- [\*“Internationalization of Input Validation Policy Violations” on page 198\*](#)

## Enabling Server-Side Input Validation

By default, server-side input validation is disabled. To enable this feature, follow these steps:

1. For 64-bit systems, server-side input validation requires a hot fix from Microsoft to function properly. To ensure that the security vulnerability has been addressed in your environment, you should refer to Microsoft Knowledge Base Article 974455 and confirm with any system administrators as needed.
2. Browse to the installation folder:  
`\CourionService\InputValidation`
3. Copy the file named `InputValidation.xml` to `CustomInputValidation.xml`.
4. Edit the `Bypass XML` tag in `CustomInputValidation.xml` and change it from `True` to `False`.

This feature is now enabled. You do not need to restart the Core Security Server.

**Note:** *Do not* edit the original `InputValidation.xml` file as it may be overwritten by an update as Core Security adds more malicious detection signatures to the configuration files.



## Configuring Options in the CustomInputValidation.xml File

Once you have created the CustomInputValidation.xml file (see [“Enabling Server-Side Input Validation” on page 196](#)), you can modify a number of configurable settings that affect the behavior of the server-side input validation feature. Table 9 lists these settings, their values, and how they affect server-side input validation.

**Table 9: Configurable Options**

| Setting                            | Valid Values                                                   | Description                                                                                                                                                                                                                                           |
|------------------------------------|----------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bypass                             | True<br>False                                                  | Bypasses input validation altogether. A value of True turns off input validation for the entire Core Security Server                                                                                                                                  |
| ScannerMode                        | Blacklist<br>Whitelist                                         | In Blacklist mode, any match by either the XSS scanner or the SQL scanner results in a rejected form post.<br><br>In Whitelist mode, each form post item must match at least one of the regular expressions.                                          |
| ClientErrorDisplay                 | General<br>Detailed                                            | General indicates that only the form post item that triggered a policy violation is displayed. No details on what the policy violation are displayed.<br><br>Detailed indicates that the form post item is displayed along with the rule it violated. |
| ServerLogMode                      | Trace<br>Info<br>Success<br>Warning<br>Error<br>Critical Error | Shows all messages below the current level in this order. The default is set to Error and Critical Error.                                                                                                                                             |
| ScanPreviousDirection<br>FormPosts | True<br>False                                                  | This setting controls whether validation is performed when the user presses the Previous button. Leave this setting in the False state to prevent edge validation loop conditions.                                                                    |
| DecodeStrings                      | Base64<br>HTMLEncoder<br>URLEncoder                            | Controls which decoding engines are run on the input form post to look for malicious signatures. They are all enabled by default.                                                                                                                     |
| DecodeString Base64                | <Decoder<br>MinimumLength<br>="20">Base64<<br>/Decoder>        | Base64 has a minimum length flag that defaults to 20. The decoder does not try to decode strings less than 20 characters by default.                                                                                                                  |

### Notes

Server-side input validation does not support the following:

- Base64 encoded strings smaller than 20 characters are not detected by default. You can configure the minimum base64 decoding level with the MinimumLength option on the Base64 DecodeString tag.
- Administration Manager tools available from the Toolbox section of the Portal page:
  - Data Security Utility
  - Enable Users Utility
  - Configuration Migration Utility

## Internationalization of Input Validation Policy Violations

You can translate input validation policy violations using the same Multi-Language Framework as you can with other end-user workflow strings. Translate the following strings:

IDS\_INPUT\_VALIDATION\_POLICY\_VIOLATION\_OCCURRED

IDS\_INPUT\_VALIDATION\_GENERIC\_POLICY\_VIOLATION

IDS\_INPUT\_VALIDATION\_POLICY\_VIOLATION\_PREFIX

IDS\_INPUT\_VALIDATION\_ONLY\_SINGLE\_VALUE\_PERMITTED

IDS\_INPUT\_VALIDATION\_UNRECOGNIZED\_VALUE

IDS\_INPUT\_VALIDATION\_INVALID\_XML\_CTRL

IDS\_INPUT\_VALIDATION\_INVALID\_ADD\_LIST\_IDX

IDS\_INPUT\_VALIDATION\_INVALID\_NO\_CHANGE\_LIST\_IDX

IDS\_INPUT\_VALIDATION\_INVALID\_REMOVE\_LIST\_IDX

IDS\_INPUT\_VALIDATION\_INVALID\_MULTIVALUE\_CTRL

IDS\_INPUT\_VALIDATION\_INVALID\_INFO\_XML

See [\*“Multilanguage Support” on page 81\*](#) for more information.

## Chapter 15: Monitoring Configuration and Performance Data

---

The Access Assurance Suite stores configuration data in the Microsoft Windows Management Interface (WMI) Configuration Information Model (CIM) repository. It also supports the use of the WMI Performance Monitor to provide statistics about provisioning and password reset actions. This chapter explains how to monitor information in the CIM repository and describes the statistics you can view using the Performance Monitor:

- [\*“Monitoring Data in the CIM Repository” on page 200\*](#)
- [\*“Using the Microsoft Windows Performance Monitor” on page 201\*](#)

## Monitoring Data in the CIM Repository

The Access Assurance Suite stores custom installation data in the Common Information Model (CIM) repository available through Microsoft's Windows Management Instrumentation (WMI). This data includes information about connectors, targets, and major components such as the Connector Framework, Connector Framework Manager, and Core Security Server.

You can monitor this data using WMI-enabled administrative tools, including the following:

- Microsoft WMI CIM Studio
  - Microsoft System Center, Operations Manager
- Both of these tools are available from the Microsoft web site.
- Macerations VBScript, Windows Powershell (including Microsoft provided scripts)
  - HP OpenView
  - Computer Associates Unicenter

With a tool such as WMI CIM Studio, browse the CMI to the following Namespace:

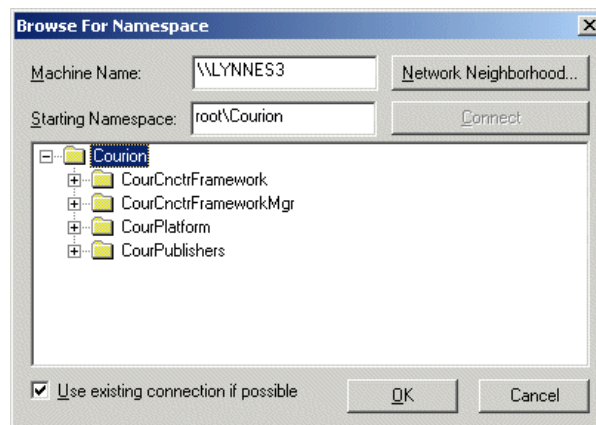
root\Courion Security

The configuration data is stored in the following folders:

- CourConctrFramwork
- CourCnctrFrameworkMgr
- CourPlatform
- CourPublishers

[Figure 95](#) shows these folders in the Courion Namespace.

**Figure 95: Courion Namespace**



# Using the Microsoft Windows Performance Monitor

You can use the Microsoft Window Performance Monitor to view and monitor statistics about ongoing provisioning and password reset actions and events.

The CourionService must be running for the Performance Monitor to obtain counter information. When the CourionService is not running, the counters always appear as zero (0). If you stop the CourionService and then restart it, you need to restart the Performance Monitor to establish a new connection to get live counter data again.

Table 10 lists the counters you can monitor, explains what they represent, and indicates whether they apply to the Provisioning Platform only or both the Provisioning Platform and the Classic Platform:

**Table 10: CourionService Performance Counters**

| Counter                            | What it Monitors                                                                                                                                             | Platform                 |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| Completed <action> Actions         | The number of completed <action> actions that the CourionService processed since it was last started.                                                        | Provisioning only        |
| Client Connections Currently Open  | The number of network-based client connections currently active in the CourionService.                                                                       | Provisioning and Classic |
| Client Connections Wait Queue Size | The number of network-based client connections waiting for the CourionService to process.                                                                    | Provisioning and Classic |
| Jobs Running                       | The number of jobs that the CourionService is currently processing.                                                                                          | Provisioning only        |
| Job Wait Queue Size                | The number of jobs waiting for the CourionService to process.                                                                                                | Provisioning only        |
| Job Workflow Exceptions            | The number of job exceptions that have occurred in the CourionService since it was last started.                                                             | Provisioning only        |
| SPML Jobs Running                  | The number of SPML jobs currently running in the CourionService. SPML jobs are those initiated from the XML Access Option and Transparent Synchronization.   | Provisioning only        |
| SPML Wait Queue Size               | The number of SPML jobs waiting for the CourionService to process. SPML jobs are those initiated from the XML Access Option and Transparent Synchronization. | Provisioning only        |
| SPML Workflow Exceptions           | The number of SPML workflow exceptions that have occurred in the CourionService since it was last started.                                                   | Provisioning only        |

To use the Performance Monitor:

1. Start the Performance Monitor. For example, from the Start Menu select **RUN...**

Enter:

```
perfmon.exe /wmi
```

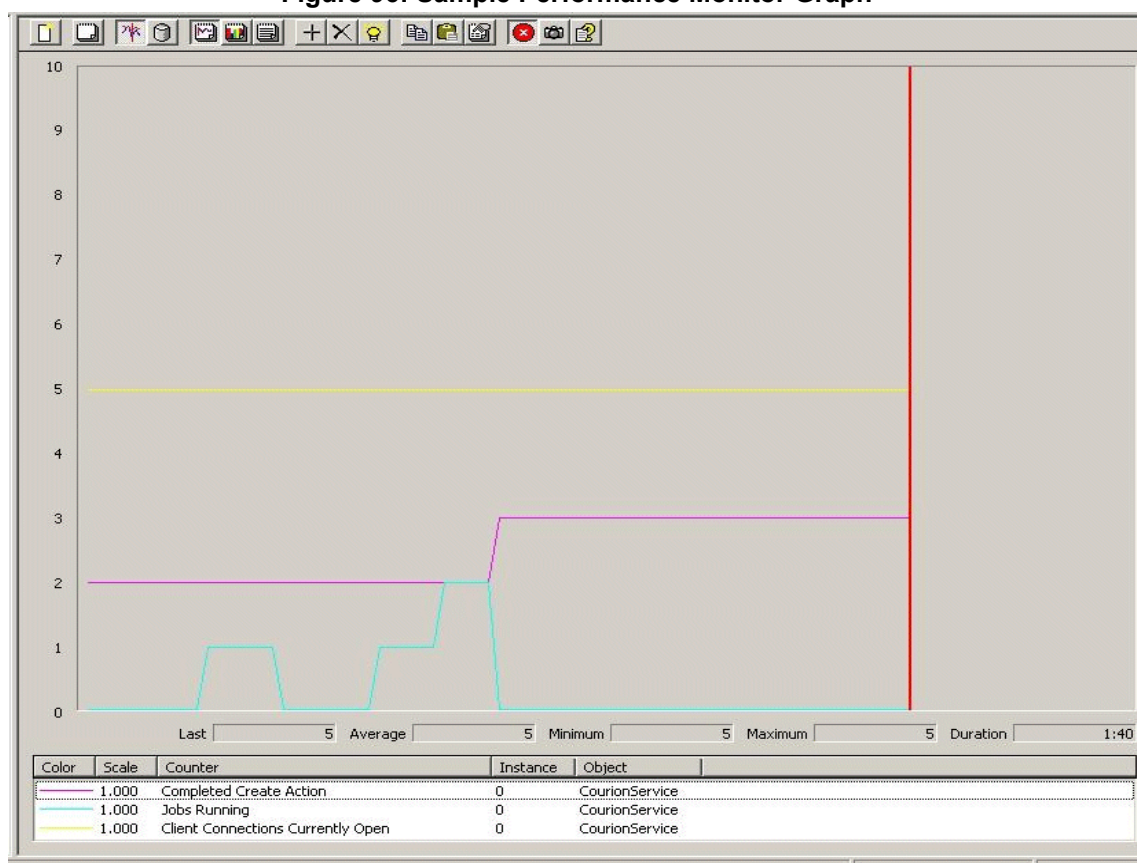
2. Add the CourierService.exe performance object from the Performance Object drop-down list.
3. Select counters from the list and click the **ADD** button.

When you do, the counter appears at the bottom of the Performance Monitor main window, and statistics for that counter appear on the graph in the window. Each counter has a color associated with its line on the graph.

To see an explanation of what a counter does, select a counter and click the Explain button.

[Figure 96](#) shows a Performance Monitor graph.

**Figure 96: Sample Performance Monitor Graph**



The yellow line is the "Client connections currently open". It shows that there are five clients connected to the CourierService. One is the Performance Monitor itself. The other four consist of one Administration Manager client and three end-user clients. (This number would be much higher in a full production environment.)

The green line shows "Jobs Running." The first spike indicates the Account Info job, which extracts the data to populate the unique account overrides screen. The second (longer) job indicates that data is being written to the transaction repository, and performs the requested action. (In a full production environment, this counter might be much higher.)

The blue line shows "Completed Create Actions." Two actions were completed since the CourionService was started, and when the job engine is finished with the current job, the number of completed actions goes up to three. This number is cumulative for however long the CourionService has been running. When the service restarts, the number resets to zero. This Completed Create Actions count only increases. The other two can increase and decrease as different users connect and disconnect, and the job engine gets more or less busy.

## Modifying the Graph

By default, the Performance Monitor assumes a Y axis graph from 0 to 100. Depending on system usage, you may want to change the properties of the graph to something other than the default.

To change the properties of the vertical scale of the graph:

1. Use "Ctrl+Q" to bring up the "System Monitor Properties" tab sheet.
2. Select the "Graph" tab.
3. Change the values in the "vertical scale."

For low activity set the Maximum to 10.

By default Performance Monitor assumes updating every second. This provides a duration of 1:40 (one minute and 40 seconds) of data visible. Changing the value to 30 seconds provides a duration of 50 minutes. The minimum, average, last and maximum data is relative to the duration. Depending on the usage, the properties may need to be changed from the default.

By default, the Performance Monitor sets tracking activity to one (1) second. Depending on system usage, you may want to extend the tracking activity to a longer interval.

To change the properties of the tracking activity:

1. Use "Ctrl+Q" to bring up the "System Monitor Properties" tab sheet.
2. Select the "General" tab
3. Change the value in the "Sample Automatically every \_\_ seconds" field.

For longer tracking of activity, set the value to 30 seconds. For example, to track the average number of open network connections for a 24 hour period set this value to 865 to get a duration of 1 day and 1 minute.





## Chapter 16: Sample Programs Used to Generate Resource Names

---

When you set up the form a provisioner accesses to create a new user resource with the Create action, you can specify that new resource names be generated automatically.

The Access Assurance Suite includes two sample files that contain Microsoft Visual Basic® scripting for automatic name generation. The two supplied programs create resource names based on a user's first initial and last name, or on the user's first name and last name. The programs assume database fields called "FirstName" and "LastName". You can use either of these sample programs, edit these sample programs to use different fields, or create new programs based on company naming conventions.

**Note:** Writing programs requires experience in Microsoft Visual Basic software or some other programming language, and is beyond the scope of this guide. Customer-written programs are the responsibility of the customer and are not supported by Core Security. Customer-written programs must be copied to the following directory:

D:\Program Files\Courion Corporation\CourionService\AcctlIDGen

Programs can be written that enforce resource naming conventions for different target systems. For example, a program could be written that generates IDs according to resource naming rules on a Windows NT® operating system, and another program that generates names according to resource naming rules on an z/OS® RACF® system.

[Figure 97](#) illustrates the contents of the program that creates a resource name based on a user's first name initial and last name.

[Figure 98](#) illustrates the contents of the program that creates a resource name based on a user's first name and last name.

**Figure 97: FirstInitialLastName.vbs**

```
' Copyright (c) Core Security 2017
' All Rights Reserved.
'
' Permission to use, copy, modify, and distribute this software
is prohibited.
'
' THE AUTHOR(S) MAKE NO REPRESENTATIONS OR WARRANTIES ABOUT THE
SUITABILITY OF THE
' SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT
LIMITED TO THE IMPLIED
' WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE, OR NON-INFRINGEMENT.
' THE AUTHORS AND PUBLISHER SHALL NOT BE LIABLE FOR ANY DAMAGES
SUFFERED BY
' LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS
SOFTWARE OR ITS DERIVATIVES.
'
' THIS SCRIPT IS PROVIDED AS A SAMPLE FOR THE ACCOUNT NAME
GENERATION FOR AccountCourier
' PRODUCT.
```

```
Option Explicit
```

```
' Generates account id's by:
' (First Initial) + (Last Name) [+ Autonumber]
Function IdEngine_GenerateId(lMaxTries)
 Dim strFirstName
 Dim strLastName
 Dim strId
 Dim bValid
 ' Retrieve parameters to build account id with
 GetParameter "FirstName", strFirstName
 GetParameter "LastName", strLastName

 strId = LCase(Left(strFirstName, 1) + strLastName)

 ' Validate this id
 ValidateId strId, bValid

 If Not bValid Then
 ' Now we resort to autonumber tactics
 Dim strBaseId
 Dim lCounter

 strBaseId = strId
 lCounter = 1

 ' Try up to lMaxTries times
 Do While Not bValid And lCounter < lMaxTries
 ' Append autonumber to end of base id
 strId = strBaseId + CStr(lCounter)

 ' Validate this id
```

**Figure 98: FirstNameLastName.vbs**

```

' Copyright (c) Core Security 2017
' All Rights Reserved.
'
' Permission to use, copy, modify, and distribute this software is
prohibited.
'
' THE AUTHOR(S) MAKE NO REPRESENTATIONS OR WARRANTIES ABOUT THE
SUITABILITY OF THE
' SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
IMPLIED
' WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR
NON-INFRINGEMENT.
' THE AUTHORS AND PUBLISHER SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED
BY
' LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE
OR ITS DERIVATIVES.
'
' THIS SCRIPT IS PROVIDED AS A SAMPLE FOR THE ACCOUNT NAME GENERATION FOR
AccountCourier
' PRODUCT.

Option Explicit

' Generates account id's by:
' (First Name) + (Last Name) [+ Autonumber]
Function IdEngine_GenerateId(lMaxTries)
 Dim strFirstName
 Dim strLastName
 Dim strId
 Dim bValid
 ' Retrieve parameters to build account id with
 GetParameter "FirstName", strFirstName
 GetParameter "LastName", strLastName

 strId = LCase(strFirstName + strLastName)

 ' Validate this id
 ValidateId strId, bValid

 If Not bValid Then
 ' Now we resort to autonumber tactics
 Dim strBaseId
 Dim lCounter

 strBaseId = strId
 lCounter = 1

 ' Try up to lMaxTries times
 Do While Not bValid And lCounter < lMaxTries
 ' Append autonumber to end of base id
 strId = strBaseId + CStr(lCounter)

 ' Validate this id

```



## Appendix A: Multi-Byte Notes

---

This appendix includes information about the support of multi-byte languages in the Access Assurance Suite in the following sections:

- [\*“Multi-Byte Notes for the Access Assurance Suite” on page 210\*](#)
- [\*“Multi-Byte Notes for Password Management Modules \(PMMs\) and Connectors” on page 211\*](#)

# Multi-Byte Notes for the Access Assurance Suite

## Access Assurance Suite Installation

You must install the Access Assurance Suite to a pathname that contains ASCII characters only. You can install the suite on a non-English operating system, as long as you do not use non-ASCII characters in the install pathname.

## The Sort Order of Lists Returned by the Core Security Server

When you change the original sort order of lists of names or accounts in workflows to an alphabetical list, foreign language characters do not appear alphabetically. For example, if you sort the list of names in the Choose Provisionees table alphabetically, and that list includes Asian and other non-English names, the sorted list includes the following characteristics:

- Sort ascending places names with non-English characters first.
- Character sets are generally grouped together, and alphabetized correctly within a single language.
- Accented English characters are *not* treated as if they were unaccented. They are sorted as non-English characters.

# Multi-Byte Notes for Password Management Modules (PMMs) and Connectors

## The Microsoft Windows Login Interface

The Microsoft Windows login interface does not support Asian characters. Although you can use the PMM for Active Directory to set your password to include Asian characters, you cannot login with that password through the Microsoft Windows login interface.

## Administrator User Names and Passwords

The Administrator username and password that you enter through the PMM configuration manager or the Connector Configuration Manager must use English characters only.

## Multi-Byte Support is for End-User Interfaces

The Administration Manager, the Connector Configuration Manager, and the PMM Configuration Manager support unaccented English characters only.

## Password Strength Rules

Password strength rules may not be enforceable, or may not be enforced as expected for non-English values. The following are notes about password strength rules using non-English values:

- Lower-case letters, upper-case letters, numeric characters, and punctuation detection works as expected.
- The “Duplicate character” rule works as expected.
- The “Do not allow substrings in username” rule is case-sensitive for all non-English languages.
- The “Do not allow repeating characters” rule works as expected.
- The minimum number of consecutive alphanumeric characters is limited to a-z A-Z 0-9.

See the chapter on Configuring Functions in the manual *Configuring Workflows with the Access Assurance Suite* for more information on password strength rules.

## Password Management Modules that are Not Multi-Byte Capable

Multi-byte characters passed to any PMM that is not multi-byte capable are converted to the default code page of the server. This can result in characters being mapped to different characters than were originally entered. This can cause either a failed password reset or a successful password reset, but the password will not contain the multi-byte characters originally entered by the user.

## Branded Custom Password Management Modules

If you have created a multi-byte ready PMM or modified an existing RDK PMM to be multi-byte ready, the Access Assurance Suite needs to be made aware that the PMM is multi-byte capable. To configure a multi-byte capable custom PMM (created with the PMM Rapid Development Kit) to receive multi-byte data, follow these steps to modify the XML branding file that comes with the PMM:

1. In the branding file add the following node: `<Unicode>yes</Unicode>`. The “yes” is not case sensitive. Any string value but “yes” is interpreted behaviorally as “no.”
2. If this is a cscript PMM, make sure the pre, execute, and post command lines all have the `/u` command incorporated.

Example: `cscript.exe /u /b /nologo mypmm.js`

**Note:** If Unicode is set to no, and cscript.exe has the `/u` command in it, the PMM does not function properly.

3. Add the PMM name, as denoted in the `<Name>` node of the branding file, to the `cnctr_constraints` override file under the UTF8 section.