CA Single Sign-On

Administration Guide

r12.1

Third Edition

CA Product References

This document references the following CA products:

- CA® Single Sign-On (CA SSO)
- CA® Access Control
- CA® ACF2
- CA® Audit
- CA® Directory
- CA® Top Secret
- Unicenter® Software Delivery

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 8: Managing Services 189

Chapter 9: User Authentication 201

Chapter 10: Launching Applications 209

Appendix B: Configuring the CA SSO Server 357

Appendix C: Configuring Authentication Agents 385

Chapter 1: The CA SSO Client

This section contains the following topics:

About the CA SSO Client

The CA SSO Client is a thick client application that lets users in your enterprise to work with CA SSO. This is the only CA SSO component that the end-user sees and works with.

The CA SSO Client runs on every workstation that uses CA SSO. The CA SSO Client can be installed on each workstation, and can be configured in a number of different ways depending on your organization's requirements.

The CA SSO Client does the following tasks:

- Communicates with primary authentication agents to verify the user's primary authentication, then stores an authenticated ticket for that session
- Displays a list of the applications that the user is authorized to use
- Sends the authenticated ticket to the CA SSO Server to gain access to applications
- Executes a logon script and logs the user into the selected application
- Sends the results of the logon attempt to the CA SSO Server if instructed to do so by the logon script
- Sends password changes resulting from an application script to the CA SSO Server to update the stored password.

Running the CA SSO Client

At installation, a shortcut for the CA SSO Client is placed in the system Start folder. When the CA SSO Client starts, a logon event is initiated, and the CA SSO Client shows a window for primary authentication.

The Authentication window shows the server-set, login name and password for the primary authentication method the user is authorized to use. These settings are located in the CA SSO Client Auth.ini file

The user can select the authentication method they want to use, or the administrator can configure the authentication dialog to display the default authentication method. The user can click Change to choose another available authentication method.

More information:

Configuring the CA SSO Client (see page 311)

CA SSO Client Interfaces

The CA SSO Client has three main interfaces through which users access CA SSO. These tools allow users to:

- Display their application list
- Refresh their application list
- Run their CA SSO-enabled applications
- Change the password for primary authentication
- Change/set logon information for an CA SSO-enabled application
- Create shortcuts to the application on the Windows Desktop or Startup Group

During the installation process you choose which user interface you want your users to have access to: SSO Tools or Launchbar. The SSO Status Icon is always installed and available.

After the CA SSO Client is installed, a shortcut for the user interface you selected is created on the desktop. If you want this interface to start up automatically, copy this shortcut to the Windows Startup Group folder for that user.

It is possible to use both, either or neither of the SSO Tools or SSO Launchbar interfaces.

Status Icon

The SSO Status Icon appears in system tray icon. This icon shows the status of CA SSO on the workstation: whether a user is logged on or not, and whether the workstation is operating in offline mode (when the CA SSO Client cannot contact the CA SSO Server).

The SSO Status Icon is always available and starts when Windows starts. The SSO Status Icon lets users launch the other SSO Client interfaces, like the Launchbar and SSO Tools, and gives the current users access to their applications.

The SSO Status Icon updates its appearance to indicate the state of CA SSO on the workstation, adding a 'tick' symbol when a user is logged onto CA SSO, a 'cross' symbol when the workstation is in offline mode, and a circle with a slash through it when there is no CA SSO user logged on. In addition, moving the mouse to hover over the SSO Status Icon displays the name of the logged on user.

You can enable or disable functionality of the Status Icon using the Client.ini file.

More information:

Configuring the CA SSO Client (see page 311)

Launchbar

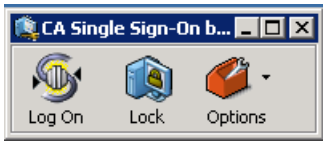
The SSO Launchbar gives users functionality similar to the Microsoft Windows QuickLaunch bar.

The Launchbar can be docked to any of the four sides of the user desktop. The Launchbar also displays container applications (and nested container applications), and provides advanced functionality to the user.

When a user is logged onto CA SSO, the Launchbar displays that user's applications as button icons. The user can click any icon to launch that application.

Like all of the CA SSO Client interfaces, you can enable or disable functionality on the Lauchbar as required. For example, in a Shared Workstation environment, for Kiosk mode, you can disable the advanced functionality and the log off button.

By default, after the user authenticates, the SSO Launchbar is automatically displayed as a floating toolbar on the screen. The first display of the SSO Launchbar looks like this:



The Launchbar has three default buttons: Logon, Lock and Options.

The Logon/Logoff button

Allows the user to log on or log off CA SSO.

If the user clicks this and they are not logged on, they see an authentication window.

If the user clicks this and they are already logged on, this button logs them off CA SSO and if CA SSO GINA or CA SSO Credential Provider is installed, clicking this button logs off the workstation as well as CA SSO. Logging off automatically closes all the applications.

Note: You may want to add logoff scripts to close the applications when the user presses the Logoff button (for example, a logoff script might automatically save changes for all open applications).

The Lock button

Locks the workstation.

The Options button

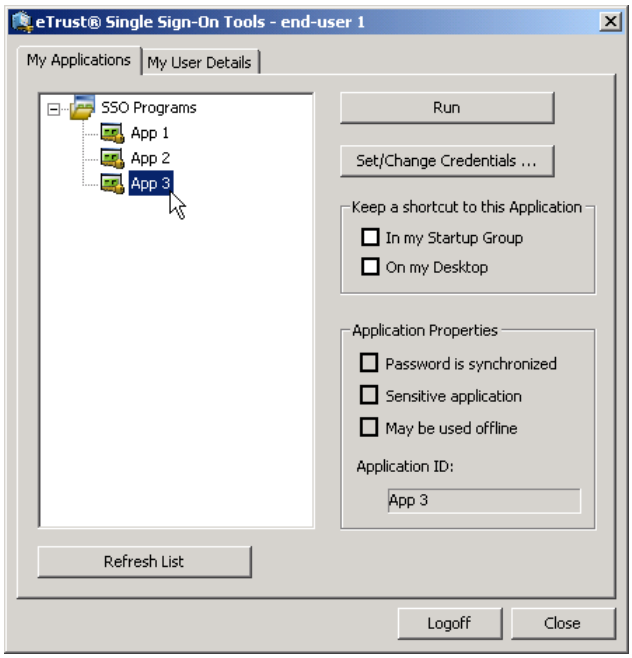
Displays a drop-down menu with additional functionality, such as whether to have the Launchbar Always On Top of other windows, or to exit the Launchbar.

Application button

Lets the user can access any application by clicking the corresponding application button.

SSO Tools

SSO Tools displays the user's application list in a tree format. To open SSO Tools, right-click the SSO Status Icon in the system tray, then select Open SSO Tools.



The SSO Tools window lets users:

- View application properties for the selected application
- Change the application password
- Create a shortcut for a selected application on the desktop, in the startup folder, or in both, by selecting the application in the tree box, and then checking Desktop or Startup under Shortcuts.

An application that is marked as a container application functions as a header for the contained applications. When a container application is selected, a new submenu is displayed, displaying all the contained applications of the selected application. The new sub-menu can also include container applications, so that you can build a multi-level hierarchy.

Configure the SSO Interfaces

You can configure the appearance and behavior of any of the three SSO Client interfaces (Launchbar, Status Icon, SSO Tools) using the Client.ini file.

To configure the Launchbar or Status Icon or SSO Tools

1. Open the Client.ini file.

   By default this is installed in the following location on the local SSO Client computer, but you may have a centralized Client.ini on a server:

   \Program Files\CA\Single Sign-On\Client\cfg

2. Edit the appropriate section of the Client.ini file.
3. Save the changes to the file.

   The launchbar, tools, or status icon of SSO tools are configured.

More information:

Configuring the CA SSO Client (see page 311)

SSO-Enabled Applications

After users are authenticated they can access their SSO-enabled applications in any of the following ways:

- Directly from the computer
- SSO Launchbar
- Status Icon, Applications menu
- SSO Tools
- Windows Start menu, All Programs, SSO Programs
- Windows Startup menu, which means that the application starts automatically when the user logs on

You can add any application to CA SSO, not just password protected ones. You may choose to add these to simplify the user experience so they can launch all of their applications from the one place.

You can change the icon and caption that appears with the icons. You may choose to do this to help users identify familiar applications.

The Application List

The SSO Client gets the list of SSO-enabled applications from the SSO Server and displays it to the user when they log onto CA SSO.

You must configure the SSO Server to build all the application lists for all users at non-peak times and store them in a cache. Users can trigger a real-time calculation of their application list directly from the SSO Server by clicking the Refresh Application List on any of the SSO interfaces. You can disable the Refresh Application List by editing the Client.ini file.

More information:

Users' Application List Updates (see page 189)
Configuring the CA SSO Client (see page 311)

The SSO Programs Menu

The application list is displayed in the Start menu. The menu item SSO Programs is displayed at the top of the menu. When you move the cursor to SSO Programs, the list of applications accessible to you is displayed in hierarchical submenus.

CA SSO builds the SSO Programs menu by dynamically placing application shortcuts in the Start Menu folder, where they can be accessed directly or copied to the desktop. Applications can be selected from the Start Menu or from a desktop shortcut.

Message of the Day

The system administrator can define a Message of the day (MOTD). MOTDs are a way to communicate with users when they log on to CA SSO. For example, you might use an MOTD to notify users about changes in working procedures. The MOTD is invoked in one of two ways:

Global MOTD

This message is displayed when the CA SSO Client starts. This must be configured on the CA SSO Server.

Application MOTD

This message is displayed when a specific SSO-enabled application starts. This is configured on the CA SSO Server.

Client MOTD

This message is displayed before signon on the Client machine. This is configured on the CA SSO Client.

The text of these messages is defined on the CA SSO Server and is sent to the user's workstation when requested by the CA SSO Client. Each message resides in a separate file, in the motd directory in the CA SSO Server installation area. You can change the directory by specifying a value for Widows servers in the Windows Registry.

The global MOTD resides in a file named motd and each application MOTD resides in a file named motd.appl, where appl is the name of the application in the data stores. These files are optional; if a file by this name is not found, CA SSO does not display any message of the day for the specific application.

Components of the SSO Client

The SSO Client includes the following components:

SSO Launchbar

This is the executable for the Launchbar interface, for end-user activities such as password change and running applications.

SSO Tools

This is a GUI for executing end–user activities such as password change and application list refreshes.

SSO Status Icon

This is the persistent System tray icon, which updates to show the SSO status on the workstation. It provides access to the end user to start up their Launchbar or SSO Tools, and is another way to prompt an SSO log on or log off.

Interpreter

This is an interpreter for the extended Tcl. The interpreter has the following functions:

- Communicating with the SSO Server when the user selects an application
- Retrieving the appropriate logon script and logon information
- Executing the logon script to log the user into the application

Window Watcher

Window Watcher is a component of CA SSO Client that continuously monitors the windows or applications that a user launches outside of CA SSO Client. Window Watcher is a part of the CA SSO status icon. When users launch applications from their computer, Window Watcher compares attributes of an application with a list of attributes for the application stored in CA SSO Server. This list of attributes is named Watchlist. If the application is Window Watch-enabled and the application attributes match the attributes of any CA SSO enabled application's Watchlist, CA SSO Interpreter supplies the login credentials for the user and logs the user into the application. Window Watcher essentially extends CA SSO functionality to applications that are not launched using CA SSO Client. The Watchlist, used by Window Watcher, is created and uploaded to CA SSO Server using Application Wizard. You can also use Policy Manager to upload the Watchlist to CA SSO Server.

Client Service

This is a background service that maintains the end-user's SSO session and handles any notifications from the SSO Server.

All of these executables, and the DLLs associated with them are located in the SSO Client bin directory. This directory can be found under the Client install directory; if the default install path has been chosen this is:

```
C:\Program Files\CA\Single Sign-On\Client
```

Config Directory

The config directory is located below the Client directory. The default path is:

```
C:\Program Files\CA\Single Sign-On\Client\cfg
```

Client.ini

This file contains SSO Client configuration information.

Auth.ini

This file contains configuration information for the authentication components of the SSO Client.

Authentication DLLs

These DLLs are located in the Auth subdirectory, under the \bin directory. Examples of dlls are: sso.dll, sso00.dll, NT.dll, NT00.dll, and so forth.

More information:

Configuring the CA SSO Client (see page 311)

Data Directory

Information on the current CA SSO user, the ticket, application list, and scripts is stored in the data directory. This information is updated regularly; if you choose to store this directory in a different location than the default (\data under the SSO Client install directory), the location needs to be one that the Client has permission to write to.

A new registry value has been added in Vista that contains the value of the locations of the data and log directories on Vista. The new registry key is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Computer Associates\SingleSignOn\Client\DataDir.
```

The location of the data directory in Vista is:

```
%ALLUSERSPROFILE%\CA\Single Sign-On\Client\data
```

Log Directory

The SSO Client logging information is saved in the Client\log directory by default. You can change this directory.

A new Registry value has been added in Vista that contains the value of the location of the Data and Log directories on Vista. The new registry key is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Client\DataDir
```

The location of the log directory in Vista is:

```
%ALLUSERPROFILE%\CA\Single Sign-On\Client\log
```

More information:

Configuring the CA SSO Client (see page 311)

Resource Directory

The SSO Client resource directory is where all of the locale- specific resource libraries are located. If the SSO Client is translated into other languages, these resource files are stored here.

Client File Changes

In general, CA SSO does not make changes to Windows DLLs on the client workstation, with the following exceptions:

- During installation of the SSO Client, some out of date Microsoft Windows DLLs may be replaced with up-to-date versions.
- When the SSO Client is installed with Windows desktop security integration, the SSOGina.dll is installed. The registry path is updated so that SSOGina.dll is accessed instead of the original Msgina.dll. The original Msgina.dll is not deleted.

SSO GINA

When a user logs in to a Windows computer, they enter their credentials using the Microsoft GINA. The GINA (Graphical Identification and Authentication library) is the component of Windows that provides secure authentication and interactive logon services. You can replace the Microsoft GINA with the CA SSO GINA.

When to deploy it

Benefits of using the SSO GINA include:

- One-step authentication to both the workstation and to CA SSO
- Using any of the SSO-supported authentication methods for Windows logon which enhances security
- Shared computer mode functionality, if necessary

How to install it

Install the SSO GINA with the CA SSO Client. When the CA SSO Client is installed with the SSO GINA, it replaces the Microsoft GINA. If the CA SSO Client is uninstalled, the SSO GINA is likewise uninstalled, and the Microsoft GINA is reinstated.

After you install the SSO GINA, configure it using the Client.ini file.

How it is controlled

The the Client.ini file controls SSO GINA behavior.

The following are the INI entries related to SSO GINA in the Client.ini file:

```
[GINA]
    LogonBitmap=
    LogonTitle=
    LogonText=
    LockedBitmap=
    LockedTitle=
    LockedText=
    Font=
    FontSize=
    GinaPassThrough=
    LogonCAD=
    FetchDomainsFromSystem=
    Domains=

[GINA/SystemLogon]
    NetWareLogon=
    NetWareServer=

[GINA/StationLock]
    EnableOsUnlock=
    DisableShutdown=
    DisableLogoff=
    UnlockStationMode=
    ShowLockedUsername=
    EnableSSOLogoff=
```

What the SSO GINA looks like

The SSO GINA has four dialogs that the user sees according to their actions and the state of the workstation, these are:

- Welcome
- Authentication
- Security
- Locked

Limitations of the SSO GINA with Terminal Services

The SSO GINA only supports a subset of terminal services functionality: the SSO GINA supports Remote Administration Mode but does not support Application Server Mode.

The SSO GINA with Remote Administration Mode lets administrators gain access to a workstation using an RDP (Remote desktop) client. We recommend that when an administrator connects to a computer:

- it must have no prior logons (this usually means that the computer must be rebooted before the remote administration logon attempt)
- the administrator must select Windows Logon Only instead of using any other SSO authentication method

Standard Operating Systems for GINA

CA SSO GINA is only supported on Windows XP SP2 and SP3, Windows 2000, and Windows 2003 Enterprise Edition SP2. For Windows Vista Ultimate SP1 and SP2, Windows 2008, and Windows 7 deploy the CA SSO Credential Provider.

Note: If Windows Group Policies fail to apply to SSO GINA, then set Network Location Awareness (NLA) service to Startup Type 'Automatic' (the default is Manual).

Configure SSO GINA

Users can log onto Windows locally or through a domain (network) using the SSO GINA.

For users to log onto a network (Windows domain), you must create a domain application on the CA SSO Server, for example, MYCORPDOMAIN (where MYCORPDOMAIN is the name of your domain) and assign users to the application. For users to log onto Windows locally, they can use the default NT_LOCAL_LOGON application, or alternatively, create a local logon application, for example MYLOCALLOGON (where MYLOCALLOGON is the hostname of your machine) and assign users to the application.

When an SSO user logs on to the network (Windows domain), the SSO GINA looks for the domain application, for example, "MYCORPDOMAIN". If the SSO GINA cannot find the application, it uses the NT_LOCAL_LOGON application to log users onto Windows locally. Similarly, when a user wants to log onto Windows locally, the SSO GINA looks for the local logon application if it exists, for example MYLOCALLOGON. If it cannot find the application, it uses the default NT_LOCAL_LOGON application.

Note: The above example assumes the use of MYCORPDOMAIN and NT_LOCAL_LOGON/MYLOCALLOGON for those intending to use SSO for network (Domain) and local logon respectively.

To set up an application for the SSO GINA on the CA SSO Server

1. Open the Policy Manager
2. Create an application and name it according to your naming standards, for example MYCORPDOMAIN.

   Note: For security reasons, we strongly recommend that you make the domain application a sensitive application. This forces users to re-authenticate before changing their domain application password. In this case the sensitive timeout is never set to zero (by default this is set to five).
3. Assign this application to all user(s).

More information:

Auth.ini Configuration (see page 343)
Client.ini Configuration (see page 314)
Registry Configuration (see page 356)

Windows Registry Values

The following registry setting determines whether the CTRL+ALT+DEL screen is displayed within Windows XP and Windows 2003:

```
HKEY_LOCAL_MACHINE\SOFTWARE\>Microsoft\Windows NT\WinLogon\DisableCAD
```

If logonCAD is set to 0 or 1, CA SSO overrides this value. If logonCAD is set to 1, then whatever value the DisableCAD registry setting is, it determines if the CTRL+ALT+DEL screen is displayed).

The following registry setting determines the action that the SSO GINA takes when you remove the smart card from the reader after you have logged on with certificate authentication method and a smart card.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\WinLogon\scremoveoption
```

If scremoveoption is set to 0, the GINA Security dialog is displayed. If it is set to 1, the workstation is locked. If it is set to 2, a Windows logoff is performed.

When you install SSO GINA, a registry key--ginadll is added to the following registry hive:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\WinLogon

ginadll

Specifies the path of the GINA DLL that is used to log a user into the computer. This key is of type REG_SZ.

Important! The ginadll value must point to a valid GINA DLL. If the path points to an invalid GINA DLL, users might not be able to log into the computer.

SSO Credential Provider

When a user logs onto a Windows Vista, Windows 2008, or Windows 7 computer, they enter their credentials using the Microsoft Credential Provider. The Microsoft Credential Provider is the component of Windows that provides secure authentication and interactive logon services. When you install CA SSO Client, CA SSO Credential Provider is added to the Credential Providers.

When to deploy it

Benefits of using the CA SSO Credential Provider include:

- One-step authentication to both the workstation and to CA SSO
- Using any of the CA SSO-supported authentication methods for Windows logon which enhances security
- Shared computer mode functionality, if required

How to install it

Install the CA SSO Credential Provider with the CA SSO Client. When the CA SSO Client is installed with the CA SSO Credential Provider, filters out the Microsoft Credential Provider by default, and provides the CA SSO Microsoft Credential Provider wrapper for logging onto the Vista machine without logging onto SSO. This is similar to the GINA pass through feature.

How it is controlled

The CA SSO Credential Provider behavior is controlled by the Client.ini file.

The following are the INI entries related to Credential Providers in the Client.ini file:

```
[CredentialProvider]
    EnableSSOCredentialProvider=
    AddLocalMachineToDomainSelectionOptions=
    FetchDomainsFromSystem=
    FilterMSProvider=
    Domains=
    LoginBitmap=

[CredentialProviderTileImages]
    ServerSetTile=
    MSCPTile=
    SSOCPTile=
    WINCPTile=
    LDAPCPTile=
    RSACPTile=
    CERTCPTile=

[GINA/StationLock]
    UnlockStationMode=
```

What the CA SSO Credential Provider looks like

The CA SSO Credential Provider has four dialogs that the user sees according to their actions and the state of the workstation, these are:

- Welcome
- Authentication
- Security
- Locked

Limitations of CA SSO Credential Providers with Terminal Services

The CA SSO Credential Provider supports only a subset of terminal services functionality. It supports remote administration mode only, it does not support application server mode.

In the remote administration mode, the CA SSO credential provider lets an administrator login to a computer using a remote desktop client with the following limitations:

- The remote computer must be restarted before an administrator logs in using the remote desktop client.
- The administrator must only use the Windows Only Logon. The administrator must not use any of the authentication methods to logon.

Standard Operating Systems for CA SSO Credential Provider

CA SSO Credential Provider is supported only on Windows Vista Ultimate Edition SP1 and SP2, Windows 2008, and Windows 7.

Configure SSO Credential Provider

Users can log onto Windows locally or through a domain (network) using the SSO Credentail Provider.

For users to log onto a network (Windows domain), you must create a domain application on the CA SSO Server, for example, MYCORPDOMAIN (where MYCORPDOMAIN is the name of your domain) and assign users to the application. For users to log onto Windows locally, they can use the default NT_LOCAL_LOGON application, or alternatively, create a local logon application, for example MYLOCALLOGON (where MYLOCALLOGON is the hostname of your machine) and assign users to the application.

When an SSO user logs on to the network (Windows domain), the SSO Credential Provider looks for the domain application, for example, "MYCORPDOMAIN". If the SSO Credentail Provider cannot find the application, it uses the NT_LOCAL_LOGON application to log users onto Windows locally. Similarly, when a user wants to log onto Windows locally, the SSO Credential Provider looks for the local logon application if it exists, for example MYLOCALLOGON. If it cannot find the application, it uses the default NT_LOCAL_LOGON application.

Note: The above example assumes the use of MYCORPDOMAIN and NT_LOCAL_LOGON/MYLOCALLOGON for those intending to use SSO for network (Domain) and local logon respectively.

To set up an application for the SSO credential provider on the CA SSO Server

1. Open the Policy Manager
2. Create an application and name it according to your naming standards, for example MYCORPDOMAIN.

   Note: For security reasons, we strongly recommend that you make the domain application a sensitive application. This forces users to re-authenticate before changing their domain application password. In this case the sensitive timeout is never set to zero (by default this is set to five).
3. Assign this application to all user(s).

Windows Registry Values

When you install SSO client with credential providers the following keys are added to the registry hive,

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication`

Credential Providers

Specifies the keys of all the credential providers installed on your computer. The credential provider keys are of REG_SZ type. The Class ID(CLSID) of the Credential Provider is as follows:

`{781A7B48-79A7-4FCF-92CC-A6977171F1A8}`

Credential Provider Filters

Specifies the CLSID of the credential provider filters installed on your computer. The CLSID of the credential provider filter is as follows:

`{A54AB0C2-B99A-43ff-A897-865D141960B6}`

MSCPWrapper

The CLSID of MSCPWrapper is as follows:

`{7BBBD3CE-A872-4c35-BCDB-16A78828703E}`

The following registry keys are added to the registry hive,

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon`

DefaultUserName

Specifies the underlying Windows username for a computer shared in a Kiosk mode. By default, the value for this key is empty. When a CA SSO user logs into the computer for the first-time in shared workstation mode 3, the underlying Windows username is captured and stored as the value for this registry key.

DefaultDomainName

Specifies the underlying Windows domain name for a computer shared in a Kiosk mode. By default, the value for this key is empty. When a CA SSO user logs into the computer for the first-time shared workstation mode 3, the domain name of the underlying Windows user is captured and stored as the value for this registry key.

DefaultPassword

Specifies the password for the underlying Windows username of a CA SSO user who is using the computer in a kiosk mode. This value is expected in clear text in the registry key. By default, the value for this key is empty.

Note: If the registry keys DefaultUserName, DefaultDomainName, and DefaultPassword are empty, the user is prompted to enter any valid Windows user credentials.

SSOAutoAdminLogon

Specifies if the credential provider is configured to log in to the Windows desktop automatically upon startup using the Windows username and password stored in the registry keys. This key can take the following values:

0

Indicates that the credential provider does not automatically log into the Windows desktop upon startup.

1

Indicates that the credential provider uses the Windows username and password stored in the DefaultUserName and DefaultPassword to log into the Windows desktop on startup.

Default: 0

The following registry setting determines whether the CTRL+ALT+DEL screen is displayed within Windows Vista, Windows 2008, and Windows 7:
`HKLM\Software\microsoft\windows\currentversion\policies\system\DisableCAD`

The following registry setting determines the action that the Credential Provider takes when you remove the smart card from the reader after you have logged on with certificate authentication method and a smart card.
`HKLM\Software\microsoft\windows\currentversion\policies\system\scremoveoption`

If scremoveoption is set to 0, the Credential Provider Security dialog is displayed. If it is set to 1, the workstation is locked. If it is set to 2, a Windows logoff is performed.

Shared Workstations

You can configure the SSO Client to suit how people use their computers. For example, you might have a kiosk-style workstation environment where lots of people access a single computer each day. Or you might have one computer for each person.

These computer modes affect how the computer is unlocked and how the users of that machine access their SSO-enabled applications and the Windows desktop. Make sure you understand how users access computers in your organization before you decide which mode to work with.

The six different computer modes are:

Single-user workstation mode 0

This is used in non-shared workstation environment. The computer can only be unlocked by the person who locked it (or a systems administrator). This therefore suits a situation where the same person uses this computer all the time. This option provides the greatest security.

Scenario

Nancy sits at one workstation full-time. She does not share her workstation with anyone else. She is the only person who logs into the domain and uses CA SSO from this computer.

Semi-shared workstation mode 1 (single Windows desktop for all users)

This mode caters to multiple users on one computer who can share a single Windows Desktop. This is used in a semi-shared computer environment. The computer can be unlocked by any CA SSO user, as long as that user shares the underlying Windows logon. This suits a situation where two or more people share a customized Windows setup, but need access to their own CA SSO applications on a workstation. All users work as different CA SSO users using the same Windows profile.

You must write a logoff script for each user. When a user is logged off, their logoff script runs, closing their open applications.

Scenario

Hilary and Mike work in Human Resources and spend a lot of their time in interviews, so they share one workstation. They share a Windows desktop that shows the applications that relate to their job, but they need to have separate access to their own CA SSO applications. When either of then unlocks the workstation in their own name they see the same Windows setup, but their own specific CA SSO applications.

Semi-shared workstation mode 2 (unique Windows desktops for each user)

This mode caters to multiple users on one computer who all need their own Windows Desktop.

This is used in a semi-shared computer environment. The computer can be unlocked by any CA SSO user, but if the new user has a different underlying Windows logon, the old Windows user is logged off and the new user is logged on. This suits a situation where two or more people share a computer and each user wants to have their own customized Windows setup as well as their own CA SSO applications. This method is slower, because it completely logs one user off Windows and then logs the next user on. It is not recommended.

You must write a logoff script for each user. When either user is logged off, their logoff script runs, closing their open applications.

Scenario

Peter and Sally share a workstation. They do very different jobs so they each want their own Windows desktop and their own CA SSO sessions. Peter works in the morning and leaves at midday. When Sally starts work in the afternoon she unlocks the workstation in her own name and sees her own Windows desktop and her own CA SSO applications.

Full Shared Workstation (kiosk) mode 3

This is used in a full shared computer environment. The computer can be unlocked by any CA SSO user, but there is no reference to the underlying Windows user. This suits a situation where two or more people share a computer and Windows setup, but each user wants to have their own customized CA SSO applications. This is like the Semi-shared workstation mode 1 option, but is much faster and suits an environment where several people may use one computer in quick succession.

In Vista, you can configure the credential provider to log in to the Windows desktop automatically upon startup using the Windows username and password stored in the registry keys for Credential Provider. So, users share the same underlying Windows credentials but different CA SSO credentials.

Scenario

A busy ward in a hospital has a computer and printer for general use. This computer is used by many doctors who need to access data quickly and print things out without going back to their offices. This computer has a generic Windows desktop and settings that connect to the printer. Each doctor can unlock the computer and see their CA SSO applications. From here they can print to the printer beside the computer because this Lock Option uses the local settings of the shared Windows setup.

Single user Windows session mode 4 (single CA SSO user per Windows session)

This mode is used in a non-shared Windows session environment (one CA SSO user for one Windows user, but multiple Windows sessions (accounts) are allowed).

A Windows session corresponds to exactly one Windows account. The Windows account is not shared by multiple SSO users. The locked Windows session can only be unlocked with the credentials that match those of the last SSO user. If the SSO user that unlocks the station has different Windows credentials, a new Windows session is opened without logging the previous Windows user off. If this session is locked too, each of the signed-on SSO users can only unlock the Windows session associated with it. Other SSO users can log in, only if they have different Windows users. (A new Windows session is created.) This means that multiple users share the same machine resources.

Scenario

Sharon logged into her Windows session to update a patient's file and then she logged off the machine with her Windows session still running. George logged onto the machine, using his credentials, which gave him access to his Windows session that was already running.

Multiple users Windows session mode 5 (multiple SSO users per Windows session)

This mode is used when two or more people share the same Windows account but need access to their own SSO applications on a computer (multiple SSO users for one Windows user, but multiple sessions are allowed).

The locked Windows session can be unlocked by any SSO user as long as that user has the same underlying Windows user. If the unlocking SSO user has a different underlying Windows user, a new Windows session is opened without logging off the previous Windows user. This means that multiple users share the same machine resources.

Scenario

Jack, Roger, and Beth share a Windows account, but each needs to access their own SSO applications on a computer. Jack unlocks the computer and accesses his SSO applications and then logs off. Roger logs on using the same Windows credentials as Jack, and then accesses the SSO applications that he needs. When Beth logs on, using the same credentials, she is able to access her SSO applications. If Jarrod logs on using different Windows credentials, a new Windows session is opened.

Note: GINA supports modes zero through three. Credential Providers support modes zero to five.

Hints and Tips for Shared Computer Mode

Locking the computer on startup

After auto-logging into the computer, you may want to invoke the CA SSO Client and Station Lock automatically. This means that an end user has to pass SSO primary authentication before obtaining access to the desktop.

You can lock the computer using several methods - two options are listed below.

- Define a shortcut in the \Documents and Settings\All Users\Start Menu\Programs\Startup folder to invoke the CA SSO Client interface
- Add String value entries to the HKLM\Software\Microsoft\Windows\Current Version\Run registry key to invoke the CA SSO Client

Hiding the desktop when a user is logging out of CA SSO

In a multi-user environment, ensure that one user does not view another user's SSO applications. This is possible in a Shared Computer environment when one user logs onto CA SSO from Station Lock when a previous user was already logged on. CA SSO invokes the first user's logoff script which may take a few seconds to run; during this time the second user can see the data that user had previously displayed on screen.

CA SSO automatically comes with a utility that you can use to hide the desktop while a logoff script is running on the user's computer. If you invoke this utility at the very start of the logoff script, it "covers" the screen and all open windows with a selected color and an information message, or an image file. You must also invoke the utility at the very end of the logoff script (or whenever the logoff script is to terminate) to remove the "cover".

The utility is called hidedesktop.exe and you can find it in the CA SSO Client installation directory. See the hidedesktop.html help file in the same directory for an overview on how to use it.

Note: This utility is designed to improve end user experience, but it may not work in all circumstances. For example, if one of the open programs is defined to run in a "stay-on-top" window, then the hidedesktop.exe may not be able to "cover" that application window.

Configure Shared Computer Mode

This procedure explains how to configure all six shared workstation modes. To activate shared workstation modes, you must create a shared user account and configure the Windows "AutoAdminLogon" setting in the Windows registry.

To configure shared workstation mode

1. Open the Client.ini file.

   By default this is installed in the following location:

   C:\Program Files\CA\Single Sign-On\Client\cfg

2. Find the [GINA/StationLock] section.
3. Edit the following values:

   UnlockStationMode

   Defines the workstation mode. These values only apply when the SSO GINA and Credential Providers are in use. When GINA is in use, the valid values are zero through three. For Credential Providers the the valid values are zero through five.

   - Single user lock option

     Select option 0. This is used in a regular non-shared computer environment (one Windows user, one SSO user).

   - Multiple SSO user lock option
     Select option 1. This is used when two or more people share a customized Windows setup, but need access to their own SSO applications on a computer (one Windows user, multiple SSO users).

   - Multiple Windows user lock option
     Select option 2. This is used when more than one person shares a computer and each user needs to have their customized Windows setup as well as their own SSO applications (multiple Windows users, multiple SSO users).

   - Kiosk mode lock option
     Select option 3. This is used when more than one person shares a computer and shares a generic Windows setup, but each user needs to have their customized SSO applications. This is like option Multiple Windows user lock mode, but is much faster and suits an environment where several people may have to use one computer in quick succession or may not have a Windows user account.

   - Single user Windows session lock option
     Select option 4. This is used in a non-shared Windows session environment (one CA SSO user for one Windows user, but multiple Windows sessions (accounts) are allowed)

- Multiple users Windows session lock option
  Select option 5. This is used when two or more people share the same Windows account but need access to their own SSO applications on a computer (multiple SSO users for one Windows users, but multiple sessions are allowed).

  Value: [0|1|2|3|4|5]

  Default: 1

4. Save the Client.ini file.

Note: If you use this Lock Option, you must set the "GINAPassThrough" parameter to 'yes' in the Client.ini file.

You must also configure the Windows Registry to automatically log the defined Windows user onto the computer. To do this, edit the "AutoAdminLogon" setting in the Windows registry. You must turn this on so that the user is never prompted to enter the Windows password.

More information:

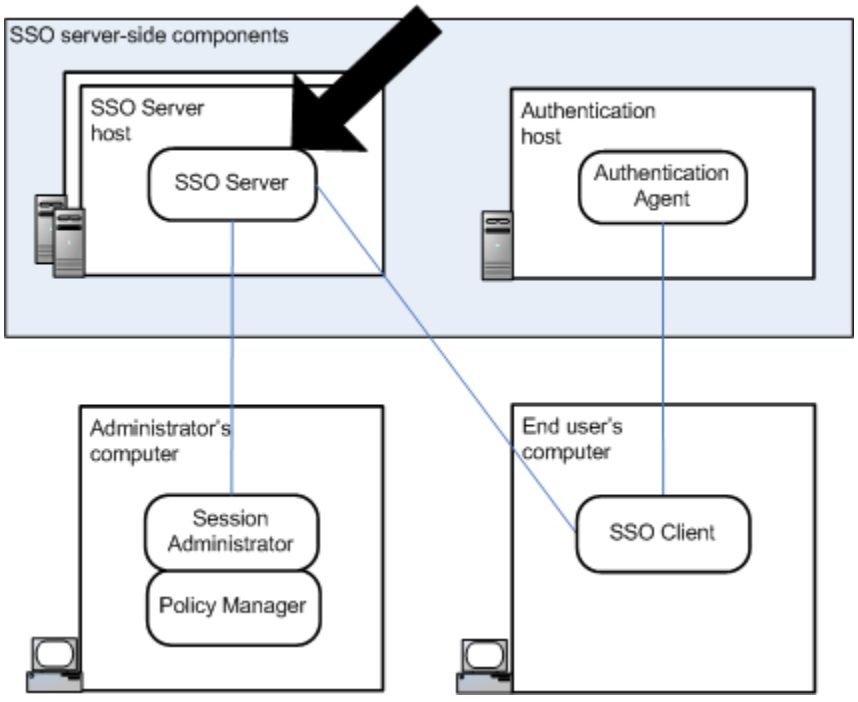
Configuring the CA SSO Client (see page 311)

Chapter 2: The CA SSO Server

This section contains the following topics:

Architecture

The following diagram shows where the CA SSO Server fits into the architecture of a typical CA SSO deployment.



About the SSO Server

The SSO Server is the center of CA SSO. It resides on a UNIX or Windows server and manages resources and provides services to the SSO Client.

The SSO Server performs the following functions:

- Authorization:
  - Builds the list of applications that a user is allowed to access and sends it to the SSO Client
  - Controls who can access the data held in the CA SSO data stores
  - Controls when data can be accessed
- Policy and user management:
  - Manages users and resources in:
    - CA Access Control
    - CA Directory
    - Active Directory and other third party data stores
  - Provides the logon scripts and the user-specific logon data for each application
  - Sets and clears passwords
  - Configures Session Management and offline application support
- Auditing, logging and tracing

Working with the CA SSO Server

The tools you use to manage the CA SSO Server are:

Policy Manager

The Policy Manager is a Windows-based management interface that lets you work with users, groups, and resources.

The Policy Manager is the default interface for managing the CA SSO Server; together with the Session Administrator, these are your configuration tools.

Selang

Selang is a proprietary command language that allows you to interact with the CA Access Control database through the CA SSO Server.

Selang commands are used in the initial stages of CA SSO implementation and for batch operations.

We do not recommend selang as your management interface; it is provided for legacy support only.

Information stored on the CA SSO Server Computer

The computer on which the CA SSO Server is installed also stores the following information:

- Data stores
- Logon scripts
- Utilities:
  - Application List Background Calculation Utility (psbgc)
  - Automatic Password Generation Utility (genkeypair)
- Log files
- "Message of the Day" files containing text that is displayed on the user's workstation at system logon or at application logon

Data Stores

When you install CA SSO, it embeds a copy of CA Access Control, and CA Directory. These store all of the data that CA SSO requires. The data that CA SSO requires is stored in three data stores:

Name of Data	Data	Stored In
User data store	Users and user groups	An LDAP directory (such as Active Directory or CA Directory)
Administrative data store	■ Applications & groups, authentication hosts & groups, password policies ■ Rules of user access to applications ■ Agents ■ Administrators	CA Access Control
Token data store	The user's session details: ■ Session ID ■ Client IP number ■ User name ■ Last heartbeat time	An LDAP directory (typically CA Directory)

You can populate your user data store with user and group information from an existing database in your organization, during or after product installation. You can conveniently import user and group information by running a utility. Alternatively, you can point your CA SSO Server at an existing LDAP datastore (such as Active Directory). In this case, the CA SSO Server recognizes the users and group information in this store and works with it as it currently exists (no changes required).

If you do not have an appropriate existing data store where your users already exist, you can use CA Directory for this purpose. CA Directory is designed to efficiently manage thousands of users, which significantly enhances the performance and scalability of CA SSO. The CA Directory data store is perfect for large enterprise installations.

CA Access Control and CA Directory are used by other CA products. Once you load information, these products can all read and update the shared data stores for their separate and common purposes.

Note: By default the CA SSO Server only recognizes English characters. If you want to enter data with non-English characters you must set the CA SSO Server to recognize those characters.

More information:

Working with the Token Data Store (see page 257)
User Data Store Class Information (see page 161)
Configuring the CA SSO Server (see page 357)

Logon Scripts

SSO scripts, such as logon scripts, are also stored on the same server host as the CA SSO Server. These provide the instructions that the CA SSO Client executes to log an end user into an application.

Services

The CA SSO Server comes with a Watchdog service. The Watchdog service watches the CA SSO Server, and can be configured to restart it if it detects it has stopped. The Watchdog service provides status updates using a number of standard protocols such as HTTP, and a suggested configuration is to point your Hardware Load Balancer (HLB) to the SSO Watchdog service.

This service is installed but not enabled when you install the CA SSO Server. To turn on the Watchdog service go to the Windows Services Manager, locate the CA Single Sign-On Server Watchdog service, and click Start.

More information:

SSO Watchdog Service (see page 268)

Utilities

The CA SSO Server works with two utilities. For more information about these utilities, see the Managing Services chapter.

The Application List Background Calculation Utility (psbgc)

The CA SSO Server supplies the list of applications that users can access. This list of applications does not change frequently, so there is no need to prepare the application list every time a user requests CA SSO services.

However, before rolling out your enterprise setup, run the psbgc once so that application lists are generated and cached on the server, before CA SSO Clients start and request these lists.

The psbgc utility compiles application lists as a background task, which reduces the CA SSO Server load and improves the CA SSO Client performance.

The Automatic Password Generation Utility

This utility automatically generates passwords for user applications. These random passwords are based on password rules set by administrators.

This increases security for two reasons:

- Automatically generated passwords are usually harder to guess than those chosen by users.
- Administrators can set stricter password rules and more frequent password changes, because user resistance is no longer a problem.

Once password auto-generation is implemented, users no longer know their applications' passwords. This means that they can only access their applications through CA SSO. This reduces help desk calls from users who forget passwords and can improve the system administrators' tracking of application use.

More information:

Managing Services (see page 189)

Initialization File (UNIX Only)

On Windows, the SSO Server is configured by changing the following Windows Registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server
```

On UNIX, the SSO Server is configured in the policyserver.ini file stored in the SSO Server directory. By default, this is:

```
/opt/CA/SingleSign-on/Server
```

More Information:

Configuring the CA SSO Server (see page 357)

Server Farms to Increase Reliability

A server farm, where each server backs up and is backed up by all the others, can provide reliable and rapid services at large sites. This is the recommended way to set up the CA SSO Server in your environment. CA SSO Server farms must contain a minimum of two servers.

You can operate a server farm if you create a network connection between two or more CA SSO Servers. Each Server can then function as the main server for a number of workstations running the CA SSO Client, while at the same time serving as a backup CA SSO Server for all the other CA SSO Servers.

You must set up Multiwrite replication between the CA Directory data stores so that all CA SSO Servers will simultaneously update the local data stores and all the data stores of the other CA SSO Servers in the replication group.

In this way, the data stores on all the server hosts contain all the data maintained in the organization – exact copies of the data store in each of the other server hosts.

Directory Structure on UNIX

The following table explains the directory structure of the SSO Server on a UNIX server relative to a base directory. This base directory is usually /opt/CA/SingleSign-on/Server.

UNIX Directory or File	Description
bin/	SSO Server daemon, startserver, stopserver, deinstall and GenKeyPair.
config/	CA Directory database configuration files. This database was installed when you installed the SSO Server.
data/	Contains the Crypto folder. The sso_rootcert.pem and sso_rootcert.pem files are stored in the Crypto folder.

UNIX Directory or File	Description
exits/	Password exits directory.
lang/	Localization directory.
lib/	SSO Server libraries.
log/	Log directory.
log_ini	Log configuration file.
motd/	"Message of the Day" for applications, one for each application and general Message of the Day. **Note:** These files can be stored in a different directory.
policyserver.ini	Server daemon initialization file.
polsrv.key	File of keys generated.
psbgc/	Application lists calculated by the psbgc utility.
pslog.ini	Log configuration file.
pslog.ini.sample	Sample file that details the settings in the pslog.ini file.
pswd.dat	Contains information that the Watchdog process uses to determine if the SSO Server is running.
pswdlog.ini	Contains logging settings for the Watchdog process.
samples/	Sample directory.
scripts/	Script directory.
_uninst	Uninstaller directory.
utils	Utilities directory.

PSLang Overview

The Pslang utility is a command-line tool for managing data in CA SSO. With the PSlang utility, you can do the following:

- Display user login information
- Edit user login information
- Create user login information
- Remove user information from ps-ldap user store
- Remove user login information for an application
- Agents and agent types
- Applications and application groups

How to Install PSlang Utility

The PSlang utility is shipped with the CA SSO installer on the DVD. Copy the PSlang utility from the following location on the DVD and extract the utility to a folder on your local computer:

`<SSO_Installer_DVD>\Tools\pslang`

Set up PSlang Utility

Before using the PSlang utility, configure the utility to communicate with the CA SSO Server.

To setup PSlang utility

1. Navigate to the folder on your local computer where you have extracted the PSlang utility.
2. Open the following file using a text editor:

   `pslang.dat`
3. Edit the following settings:

   POLICY_SERVER_HOST

   Specifies the hostname of the computer where the CA SSO Server is installed.

   AUTH_METHOD

   Specifies the authentication method used to communicate with the CA SSO Server.

   Default: EAC

   Note: Do not modify this setting.

   AUTH_HOST_NAME

   Specifies the authentication agent hostname.

   Default: EAC_AUTHHOST

   Note: Do not modify this setting.

   USERNAME

   Specifies the username of the ps-ldap admin in the CA SSO Server.

   CREDENTIALS_FILE

   Specifies the file where the username and login credentials of the ps-ldap admin user are stored. This setting is optional. Use this file to automate login to the CA SSO Server.

   DEFAULT_USER_DIR

   Specifies the default user store used by the CA SSO Server.

   Default: ps-ldap

   PRINT_EMPTY_VALUES_AS_NA

   Specifies if the empty values for any parameters must be displayed as 'n/a'. For example, if you have not set a value to Credentials_file, and use the pslang command su to display the user details, Credentials_file setting is not printed. To print the Credentials_file setting even if it is not set, you must the PRINT_EMPTY_VALUES_AS_NA to True.

PROMPT_ON_UPDATE

Specifies if the pslang utility must prompt users for confirmation before executing any update,create, or delete commands.

Value: True|False

Default: True.

ADD_MSG_IDX

Specifies if the pslang messages are indexed. For example, if this option is set to True, all the pslang return messages are indexed. The indexing starts at 0 (zero). So, the first pslang command return message after you have performed any operation is indexed as 0, the next pslang message will be indexed as 1, and so on.

MESSAGE_FILE_PATH

Specifies the absolute location of the CA SSO Server message enu.msg file.

MESSAGE_FILE_NAME

Specifies the message file name.

Default:enu.msg

COMM_MODE=0

Specifies the mode of communication between the PSlang utility and the CA SSO Server.

Value: 0 | 1 | 2

0 - Non-FIPS mode

1 - FIPS-only mode

2 - Mixed mode

Default: 0

Note: In FIPS-only and Mixed modes, a trusted certificate is needed.

PEMFILE_PATH

Specifies the absolute path to the .pem certificate file to use during FIPS-mode.

KEY_FILEPATH

Specifies the absolute path to the private key to use with the authentication certificate.

4. Save and close the file.

Using the PSlang Utility

To use the PSlang utility to manage CA SSO Server, do the following:

1. Navigate to the folder where the PSlang utility is extracted using the command-line.
2. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   `pslang.exe`
3. Run any of the following commands from the pslang prompt:

   su

   Displays the user information in the user store.

   el

   Edit login information for an application.

   nu

   Adds a user to the ps-ldap.

   ru

   Removes a user from the ps-ldap user store.

   rl

   Removes a user login information for an application.

   lm

   Lists the authentication methods available.

Display a User Using PSlang Utility

To display a user record from the user store

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> su "<user_name>" [user_dir(<user_dir_name>)]
   [container(<container_dn>)] [appl(<appl_name>)]
   ```

   Where

   user_name

   Specifies the user name of a specific user you want to display.

   user_dir_name

   Specifies the user directory where the user is stored. If the parameter is not mentioned, pslang utility picks the value from the ini file.

   container_dn

   Specifies the container dn for the specified user.

   appl_name

   Specifies the application name for which the user login information must be displayed. If the application name is blank, pslang displays the login information for all the applications that the specifies user is authorized to access.

Create a User Using PSlang Utility

To create a user record in the ps-ldap user store

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> nu "<user_name>"
   [user_dir(<user_dir_name>)][container(<container_dn>)]

   lastname(<last name>)[fullname(<full name>)][comment(<comment>)]

   [location(<location>)][auth_type(<method_id_1>,<method_id_n>)][pwd_autogen(TR
   UE)] [pwd_sync(TRUE)]
   ```

   Where

   user_name

   Specifies the user name of the new user.

   user_dir_name

   Specifies the user directory where the user is stored. If the parameter is not mentioned, pslang utility picks the value from the ini file.

   container_dn

   Specifies the container dn for the new user.

   last name

   Specifies the last name of the user. This parameter is mandatory.

   full name

   Specifies the full name of the user.

   comment

   Specifies a description about the user.

   location

   Specifies the location of the user.

   auth-type

   Specifies the authentication method IDs as specified by the lm option of pslang utility. For more information about the method ID, see the description for the pslang option lm.

   pwd_autogen

   Specifies if the automatic password generation is enabled.

   pwd_sync

Specifies if password synchronization is enabled.

Edit User Login Information for an Application

To edit a user login information for an application

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> el "<user_name>"  [user_dir(<user_dir_name>)]
   [container(<container_dn>)]
   appl(<appl_name>)[loginid(<login_name>)][currpwd(<password>)]
   [nextpwd(<password>)][ignore][checkonly][expired(TRUE)]
   ```

   Where

   user_name

   Specifies the user for whom you want to edit the login information for an application.

   user_dir_name

   Specifies the user directory where the user is stored. If the parameter is not mentioned, pslang utility picks the value from the ini file.

   container_dn

   Specifies the container dn for the specified user.

   appl_name

   Specifies the application name for which the user login information is being edited. This field is mandatory.

   login_name

   Specifies the user login name for the specified application.

   password

   Specifies the password for the user login for the specified application.

   ignore

   Specifies if the password policy must be ignored.

   checkonly

   Checks the password quality against the password policy.

   expired

   Specifies that the user must change the password at next login.

Remove a User from the ps-Ldap User Store

To remove a user record from the ps-Ldap user store

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> ru "<user_name>"
   [user_dir(<user_dir_name>)][container(<container_dn>)]
   ```

   Where

   user_name

   Specifies the user name of a specific user you want to remove.

   user_dir_name

   Specifies the user directory where the user is stored. If the parameter is not mentioned, pslang utility picks the value from the ini file.

   container_dn

   Specifies the container dn for the specified user.

Remove Login Information from an Application

To remove a user login information from an application

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> rl "<user_name>" [user_dir(<user_dir_name>)]
   ```

   ```
   [container(<container_dn>)]appl(<application_name>)
   ```

   Where

   user_name

   Specifies the user for whom the login information is removed from an application.

   user_dir_name

   Specifies the user directory where the user is stored. If the parameter is not mentioned, pslang utility picks the value from the ini file.

   container_dn

   Specifies the container dn for the specified user.

   application_name

   Specifies the application name from which the user login information is deleted. This field is mandatory.

List the Supported Authentication Methods

When creating a new user, you can configure the authentication method used by the user.

To display a list of supported authentication methods

1. Run the following command with any of the listed options from the command prompt to connect to the CA SSO Server:

   ```
   pslang.exe
   ```

2. Run the following command from the pslang prompt:

   ```
   pslang> lm
   ```

The following are the supported authentication methods:

Method Name	Method ID	Description
CERT	Method12	Certificate Authentication
EAC	Method20	eTrust Access Control Authentication
LDAP	Method21	LDAP Authentication
NTLM	Method23	NTLM Authentication
RSA	Method6	RSA SecurID Authentication
SSO	Method5	eTrust SSO Authentication
TICKET	Method22	TICKET Authentication
WIN	Method7	WIN Authentication

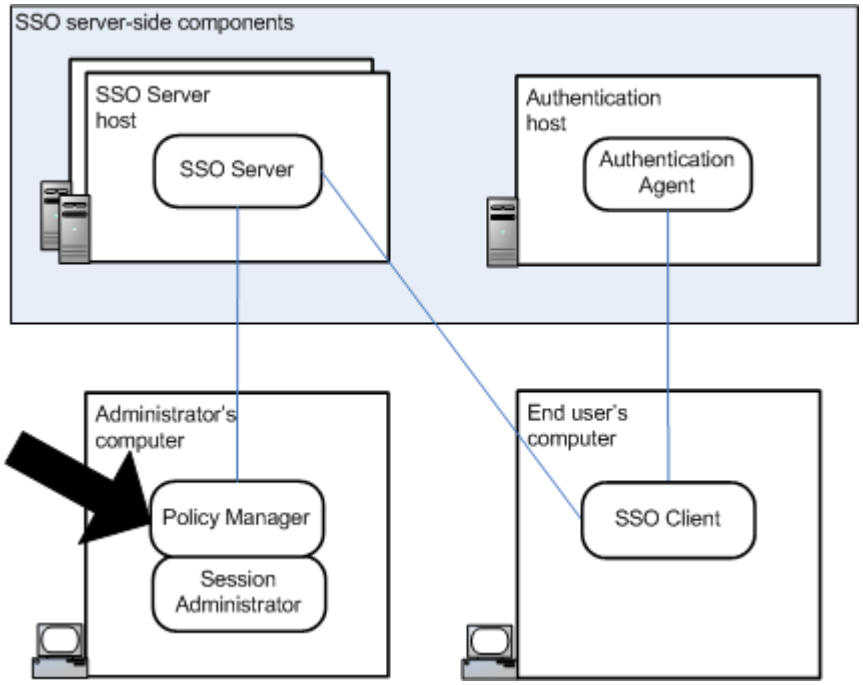
Chapter 3: The Policy Manager

This chapter describes the Policy Manager. The Managing Resources chapter describes how to use the Policy Manager to work with CA SSO data.

This section contains the following topics:

Architecture

The following diagram shows where the Policy Manager fits into the architecture of a typical CA SSO deployment. The Policy Manager can be deployed on any administrator's computer.



About the Policy Manager

The Policy Manager is an easy-to-use interface for managing data in CA SSO. With the Policy Manager, you can add, delete, modify and link the following:

- Users and user groups
- Agents and agent types
- Applications and application groups
- Authentication hosts and authentication host groups
- Password policies
- Session profiles
- Other CA SSO resources

The Policy Manager generates commands and sends them to the CA SSO Server, which updates the data stores (both administrative and user).

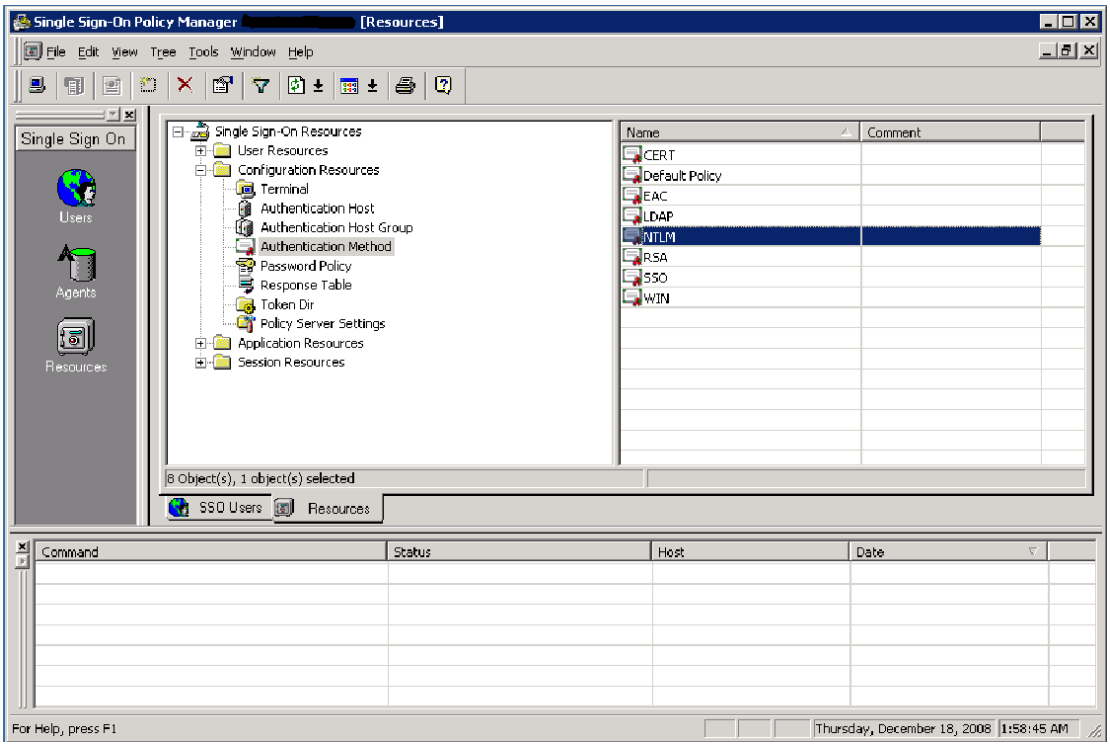
The Policy Manager runs on Windows workstations that have a TCP/IP connection to the CA SSO Server.

The Policy Manager Window

All data management begins at the main window of the Policy Manager. This window appears after you successfully complete the Login dialog.

To login to the Policy Manager for the first time, use the Admin user you created during the CA SSO Server install (ps-admin is the default name, although you may have changed this).

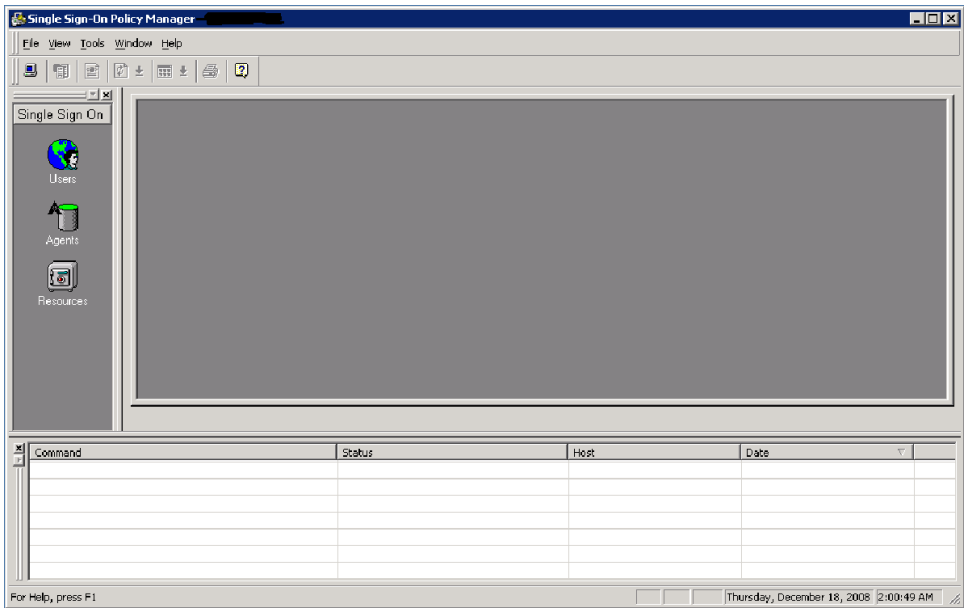
This is a sample Policy Manager window:



To increase the display area, you can use the View menu to hide any of the window areas except the workspace.

Workspace

When the main window of the Policy Manager first appears, there is no application window section—only an empty workspace:



Program Bar

The program bar on the left shows an icon for each of the categories of entries you can manage with the Policy Manager.

Click an icon in the program bar to work with that category in the Policy Manager:

Users

Lets you add, remove, and edit users and user groups in your user and administrative data stores.

Agents

Lets you create, edit and delete agents and agent types.

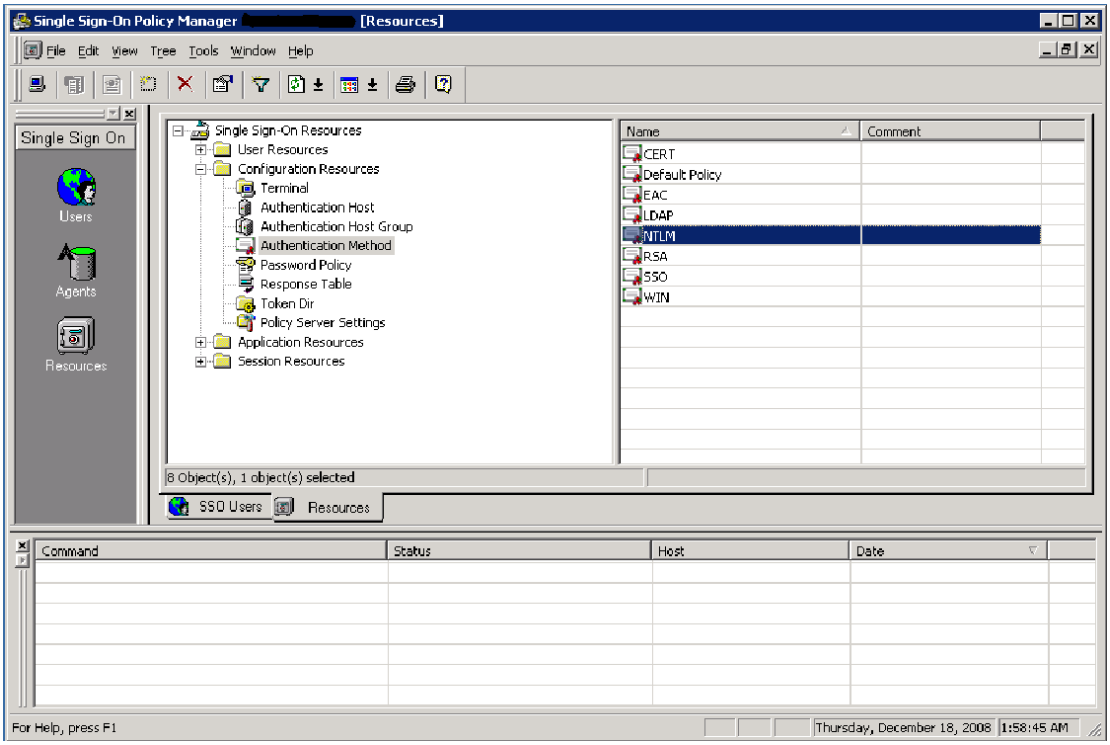
Resources

Lets you manage the user resources, configuration resources, application resources, and session resources in your policy data store.

Application Windows

The purpose of the workspace is to display application windows, which list users and resources.

To open an application window, click an icon in the program bar, or click an option in the File, Open menu.



When an application window is first opened, a list of entry types appears in the left pane of the application window. You can then click an entry type icon to see a list of all entries of that type in the right pane.

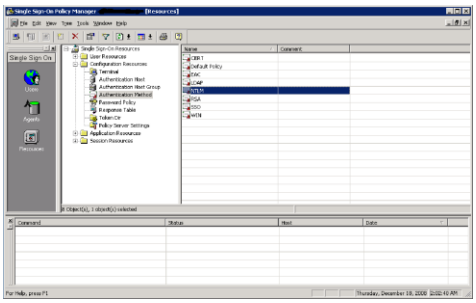
Workbook Mode

As you work with the Policy Manager, you might open several different application windows.

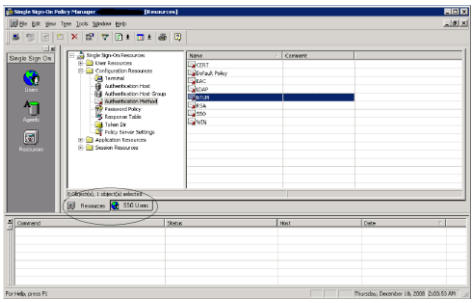
If Workbook Mode is switched on, each application window remains open, and you can quickly return to it by clicking the tab for that window.

You can turn Workbook Mode off and on using either the View menu or the Appearances dialog on the Tools, Options menu.

In the following graphics, the Policy Manager shown on the right has two application windows currently open: CA SSO Users and Resources.



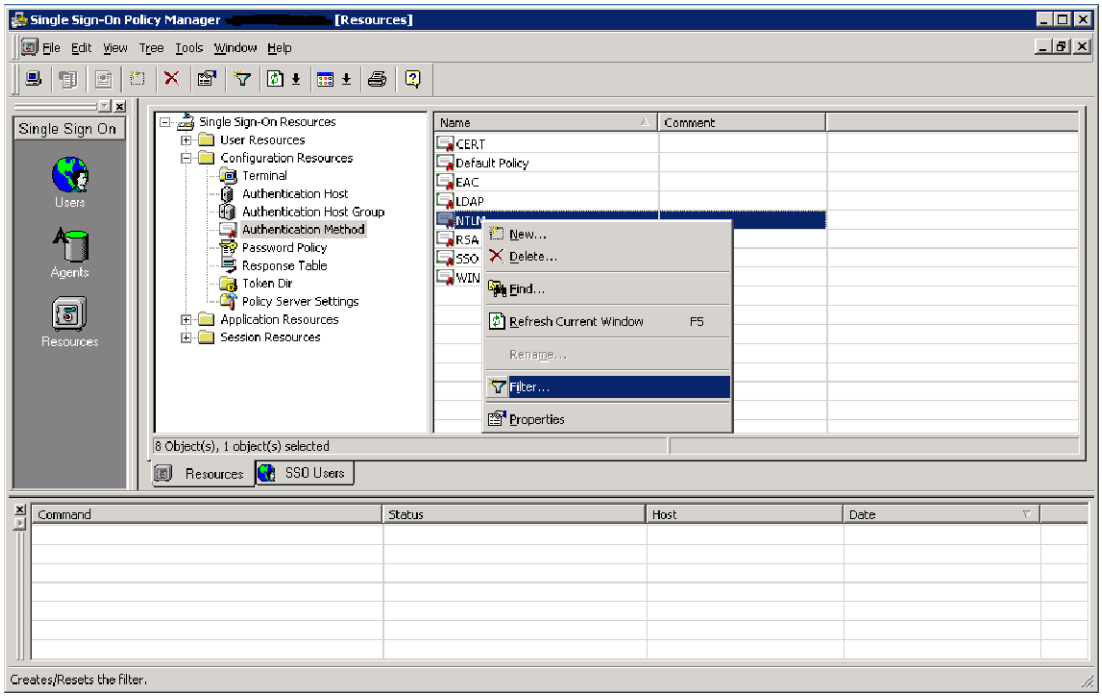
Policy Manager with Workbook mode off



Policy Manager with Workbook mode on (note the tabs at the bottom of the application window)

Work with List Entries

To work with a list entry, right-click on it to open a pop-up menu.

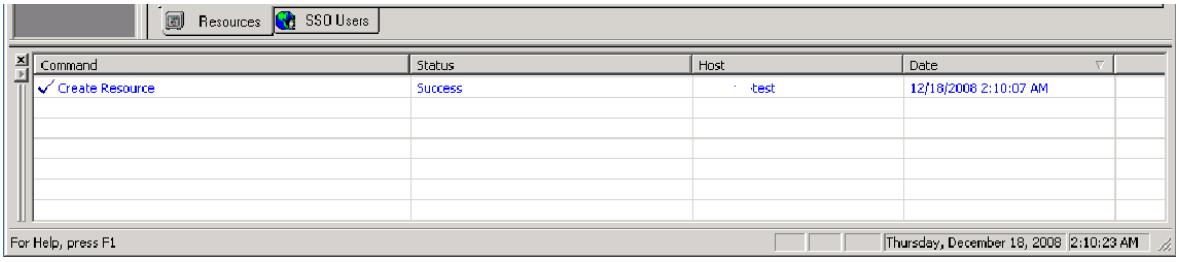


This pop-up menu lets you do some or all of the following:

- Create a new entry
- Delete the selected entries
- Rename an entry
- Find a specific entry in the data store
- Filter the displayed list to view only those entries with specified properties
- View the details of an entry
- View all the links of an entry (for example, the groups a user is linked to)

Output Bar

The Policy Manager generates the commands that are sent to the data store. The output bar displays the result of the command.

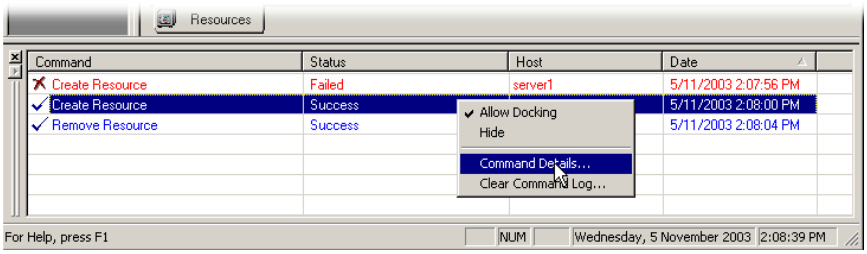


The commands that are displayed in the output bar are also recorded in the command log.

To limit which commands must be logged, use the Command Log tab in the Options dialog. In this dialog, you can choose to not log transactions that use the Find command or the Show command.

Every time you begin a new session of the Policy Manager, a new command log is created. If you want to save the commands from a session, you must save or print the log.

You can right-click on the output bar to open a pop-up menu:



This pop-up menu lets you:

- Allow the output bar to dock with the main window
- Hide the output bar
- Open the Command Details dialog to see details about the selected command. You can print or save the information in this dialog.
- Clear the command log

Menu Bar

The menu bar contains the pull-down menus of commands you can use with the Policy Manager. The menu bar structure is dynamic, with appropriate commands appearing for the action you are taking. For example, the Tree menu appears only when the active window contains a tree structure.

Toolbar

The toolbar provides easy access to frequently used commands. Most of the commands are also accessible from the menu bar. Like the menu bar, the toolbar is dynamic, with appropriate commands appearing for the action you are taking. Common tools are described in the following sections. Tools specific to a particular window are described in the section on that functionality.

Connect

The Connect button displays the Login dialog, which lets you connect to a different host.

New, Delete and Properties

These three buttons let you create a new entry, delete the selected entries, and view the properties of the selected entry.

Filter

The Filter button lets you limit the displayed list to view only those entries with specified properties.

Refresh

The Refresh button lets you redisplay the current window after running a transaction. The arrow lets you refresh all windows or the current window.

Views

The Views button lets you select the view for the active window. The choices are Large Icons, Small Icons, List, and Details. The arrow gives a drop-down list of the choices. The icon toggles you through the list. Each window can have its own view setting.

Print

The Print button displays a Print dialog that lets you print the contents of the active window. You can select different formats for the header, content, and footer. Clicking OK in this dialog opens the Windows Print dialog, where you can set more printer options.

About

The About button displays information about the Policy Manager, including version number and registration information.

Start the Policy Manager

You must be logged onto your computer as a Windows administrator, and then log onto the Policy Manager as a CA SSO administrator.

To start the Policy Manager

1. Find out the following information:
   - The name of the computer on which the CA SSO Server is running.

     If it is not running, start the CA SSO Server from the Services dialog on the Control Panel (on the CA SSO Server machine).
   - The user name and password of a CA SSO administrator.

     If you are signing on for the first time after CA SSO was installed, use the CA SSO Server administrator user. The default CA SSO Server administrator user is ps-admin. This is defined when you install the CA SSO Server.
   - Host name (DNS name or IP address) of the computer on which the CA SSO Server is running.

     If you are logging on from the computer where the CA SSO Server is installed, use localhost as the name of the host.
   - A Windows user who is an Admin group member

     You must be logged on to Windows as an Administrative user in order to run the Policy Manager.
2. On the Policy Manager computer, choose Start, All Programs, CA, Single Sign-On, Policy Manager.

   The Login dialog appears.

3. Fill in the Login dialog with the following authentication information:
   - User name of a CA SSO Server administrator user
   - Password for the administrator, which was specified during installation
   - Host name of the computer on which the CA SSO Server is running. You can use the Browse button to locate the name of the host.

   After you have logged in, the Policy Manager window appears.

Customize the Policy Manager

You can customize the system options of the Policy Manager that were set during installation.

To change the default settings on the Policy Manager

1. Choose Options from the Tools menu to display the Options dialog.
2. Use the tabs on this dialog to locate the options that you want to change.

   The following list describes each tab on the Options dialog.

Tab	Description
Appearance	Defines the way the toolbar and main windows look and activates the Connection dialog.
Startup	Sets various startup options, specifies a custom splash screen, and enables sub-administration.
Command Log	Defines what types of commands appear in the Policy Manager command log window.
Connections	Specifies the connection timeouts between the Policy Manager and the CA SSO Server, and set the buffer size.

Move, Hide, and Display the Output Bar and the Program Bar

The program bar and the output bar can be moved, hidden, and displayed. They both have a handle:

To move the program bar or the output bar, click on the handle and drag the bar off the main Policy Manager window.

To hide the program bar or the output bar, click the X button on the handle.

To display the program bar or the output bar, use the View menu.

Find, Filter, and Sort Entries

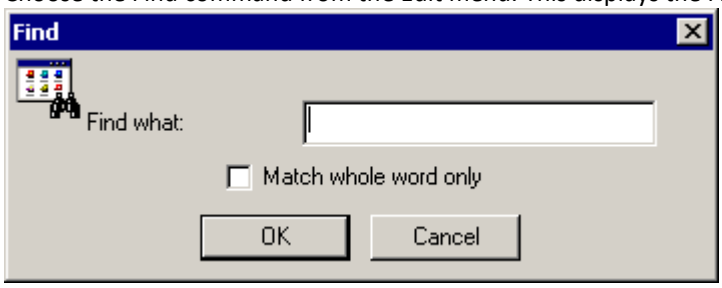
As you continue to use CA SSO, the lists in some of the application windows will become long. To find entries quickly, you can use the Find and Filter commands. You can also use several commands from the View menu to change how entries are displayed in the list.

Find an Entry

To find a particular entry in a list

1. Navigate to the list in which you want to search.

   Choose the Find command from the Edit menu. This displays the Find dialog.



2. Type all or the beginning of the entry in the Find What field, then click OK. The first entry matching your search criteria will be highlighted in the list.

   Note: You cannot use wildcards in the Find dialog.

Filter the List Entries with Wildcards

When you are working with long lists of users, groups, or resources, you can filter the list to display only some of the entries. You can use an asterisk (*****) to replace one or more letters in a search. One ***** can replace several letters in a row. For example, to display a list of entries beginning with **ber**, enter:

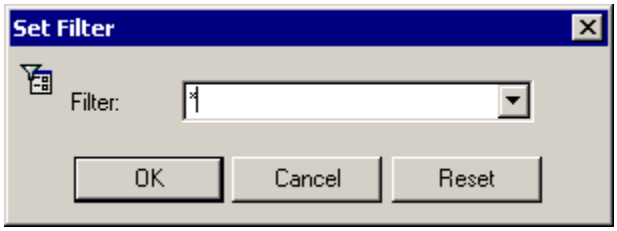
ber*

To display a list of entries containing **USER**, enter:

USER

To filter a list

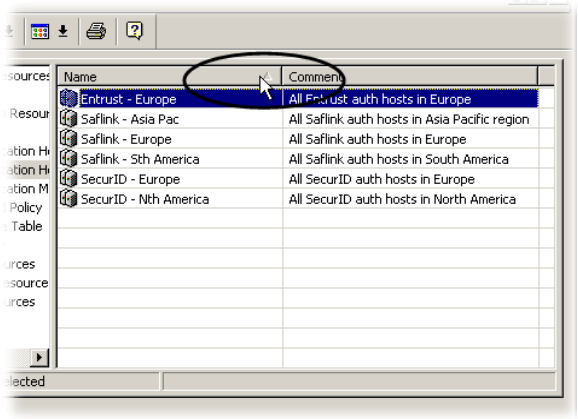
1. Click the Filter button or choose the Edit, Filter menu option to display the Filter dialog:



2. In the Filter field enter a string of characters that is found in all of the names of the entries that you want to display.
3. To redisplay the entire list of entries, click the Reset button, and then click OK.

Sort List Entries

You can sort data by clicking any column header. If you click a column header twice, the entries are sorted in reverse order.



This works in the list of entries in the application window, and also the list of commands in the output bar.

Refresh the Window

You can update the information in the current application window or in all application windows. To update the information in the current window, choose Refresh Current Window from the View menu. To update the information in all windows, choose Refresh All from the View menu.

Chapter 4: Managing Resources

This chapter explains how resources such as users, applications and resources are managed within CA SSO.

This section contains the following topics:

About Managing Resources

A *resource* is an entity that users and groups can access. Resources are grouped by *class*, which is a name for the type of resource. For example, the APPL class contains all applications.

The properties of a resource are stored in the resource's *entry*. An entry is a collection of data that consists of the name and properties of a resource. The properties of resources indicate who defined the resource, the date when the resource was defined, and more.

Every entry in a particular class contains values for the same set of properties—the properties appropriate to the type of resource that the class describes.

You can define and maintain all of the resources in the policy data store from the Resources window. To display the Resources window, click the Resources icon in the program bar.

Resources are grouped into these categories:

User Resources

Defines user data stores and user attributes.

Configuration Resources

Defines authentication hosts, authentication host groups, authentication methods, responses, password policies, token directories, and CA SSO Server settings.

Application Resources

Defines applications and application groups.

Session Resources

Defines Policies for managing user sessions.

Populate the Policy Data Store with Resources

To protect your resources you first need to define them. The policy data store must contain all of the required information on resources, applications, application groups, authentication hosts, password policies, and agents.

Populate the policy data store by using either the Policy Manager or selang.

After populating the policy data store you must assign access permissions by defining the access rules for each resource. Once the policy data store contains the resource definitions and access rules, you can control access to your resources.

Managing User Resources

The User Resources folder includes the user data stores and user attributes.

More information:

Managing Users and User Groups (see page 155)

Resource Configuration

From the Configuration Resources folder in the Policy Manager you can define and change security for these resources:

- Authentication hosts
- Authentication host groups
- Authentication methods

- Password policies
- Response tables
- Token directories
- Policy Server settings

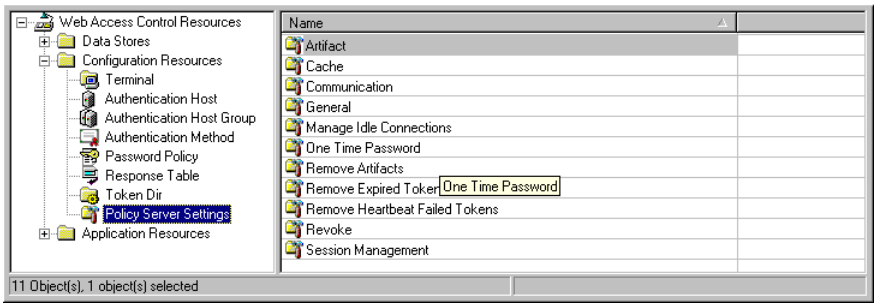
Define CA SSO Server Settings

CA SSO Server Settings control the various aspects of the CA SSO Server configuration.

To display the list of the CA SSO Server settings

1. Expand the Configuration Resources folder
2. Click the CA SSO Server Settings folder.

   The following window displays in the workspace with the subfolders containing the CA SSO Server settings listed in the Name column:



   Note: During the CA SSO Server installation the settings are populated with the appropriate values.

More information:

Configuring the CA SSO Server (see page 357)

Update CA SSO Server Settings

To change existing CA SSO Server settings

1. Display the CA SSO Server settings.
2. Locate the subfolder of the setting you want to change and double-click its entry in the list.

   The View or Set GPSCONFIGPROPERTY Properties dialog appears.

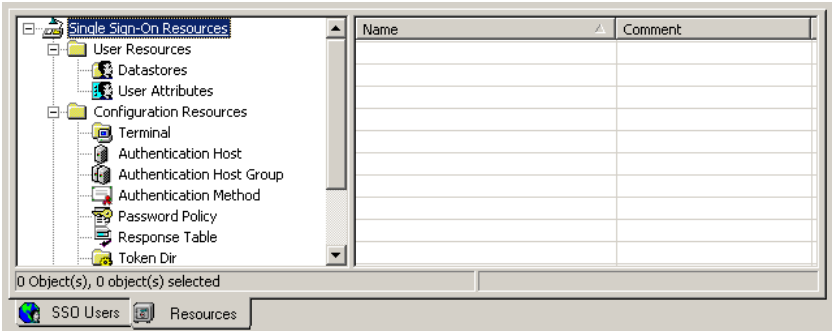
   Select the property setting you want to change and double-click its entry in the list.
3. Make any changes that are necessary and click OK when you are finished.

A *resource* is an entity that users and groups can access.

The properties of a resource are stored in the resource's *entry*. An entry is a collection of data that consists of the name and properties of a resource. The properties of resources indicate who defined the resource, the date when the resource was defined, and more.

Every entry in a particular class contains values for the same set of properties—the properties appropriate to the type of resource that the class describes.

You can define and maintain all of the resources in the policy data store from the Resources window. To display the Resources window, click the Resources icon in the program bar.



Resources are grouped into these categories:

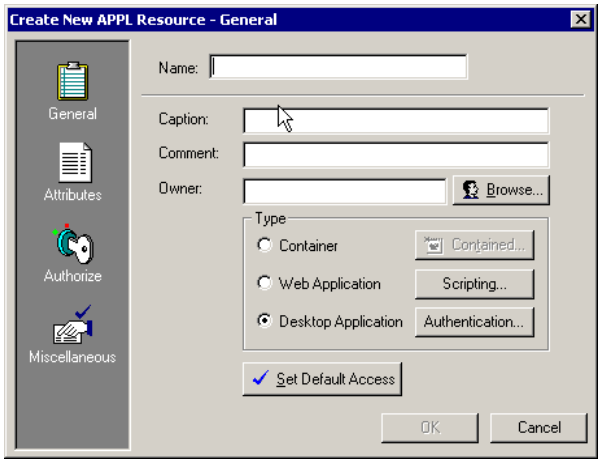
Resources Group	Resources
User Resources	■ **Datastores** Data store in which the CA SSO Server stores user information ■ **User Attributes** Categories of extra information that can be recorded for each user, such as the country they work in, or their children's names.
Configuration Resources	■ **Authentication Host** Computers that run authentication agents. ■ **Authentication Method** The method by which users are authenticated. ■ **Password Policy** The rules that apply to the strength of the password. ■ **Response Table** Table that defines additional information to be returned with an authorization request. ■ **Token Directory** An LDAP directory in which the CA SSO Server stores the users' session information.
Application Resources	Applications that users can log on to using CA SSO.
Session Resources	Policies for managing user sessions.

You can administer all the resources in the policy data store by doing the following:

- Add a resource to any class in the policy data store
- Update a resource in any class in the policy data store
- Delete a resource in any class from the policy data store
- Define holidays when users need extra privileges to log on
- Define task delegation and task groups

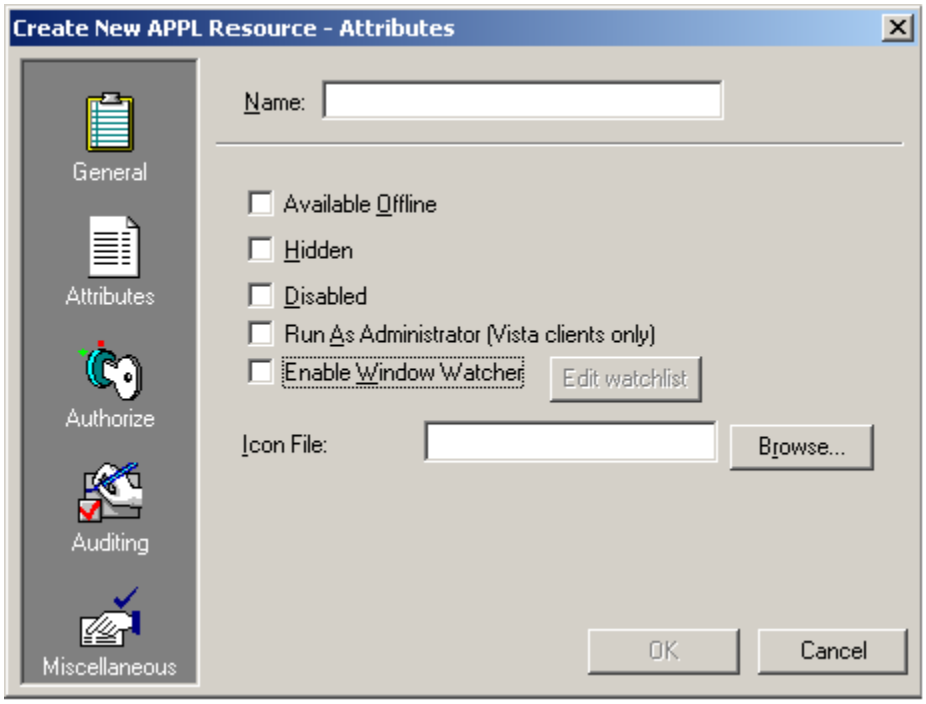
To perform these functions, click Resources in the Single Sign On panel of the program bar, select a resource in the workspace, and then click New, Delete, or Properties on the toolbar. When you click Resources, a new menu option named Upload Watchlist appears in the Tool menu. This option lets you upload the Watchlist generated by the Application Wizard to CA SSO Server.

Here is the dialog for creating a new application resource, which uses the APPL class:

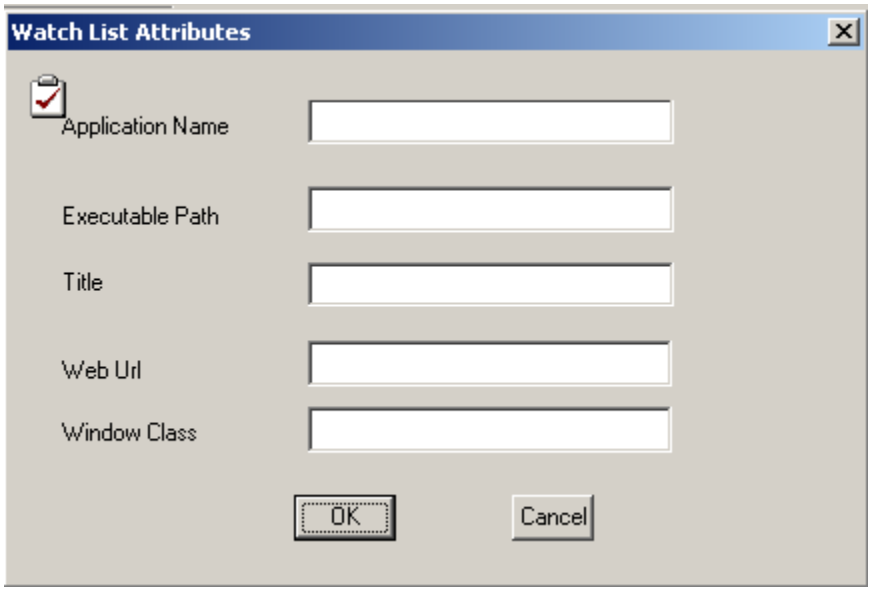


Click the icons on the left to display different dialogs. For example, the General dialog, which is shown, lets you enter the resource name and description, specify the owner, and more.

Use the following dialog in the Policy Manager to define the attributes for a new application resource. When you create an application resource, you have the option to enable Window Watcher. Window Watcher is an CA SSO Client component that lets CA SSO provide single sign-on capabilities to applications that are launched outside of CA SSO Client. Select the Enable Window Watcher to let Window Watcher monitor the new application resources. When you enable Window Watcher, the Edit Watchlist button is activated. Use the Edit Watchlist button to configure the Watchlist settings for this new application resource.



Use the following dialog in the Policy Manager to edit the Watchlist.



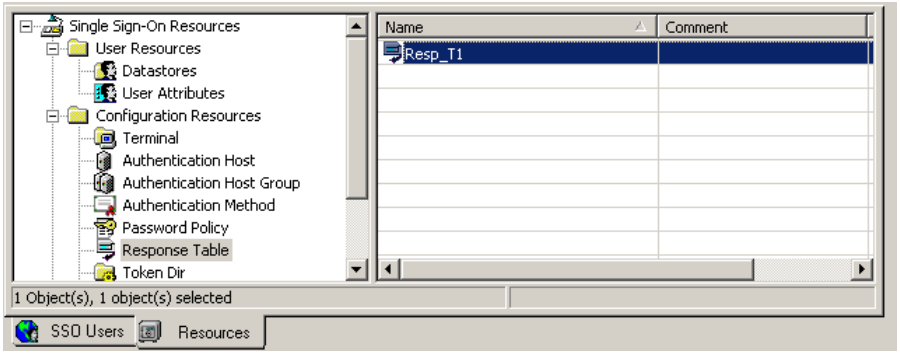
Note: For more information about the Watchlist Attributes, see the *Policy Manager Help*.

Define a Response Table

A response defines additional information to be returned with a Web SSO authorization request. This additional information is static text that can be returned in addition to a successful authorization decision (access granted). The Web Agent returns the response text as a cookie so a developer can use this cookie to customize the look of the web site.

To display a list of response tables

1. Open the Configuration Resources folder.
2. Click the Response Table object.



From the list of response tables, you can do the following tasks:

- Add a new response table
- Remove a response table
- Locate a response table in the list
- Change the properties of a response table.

To perform any of these actions, use the Edit menu or right-click in the list of response tables and then select from the command list.

These settings are associated with personalizing your web page content.

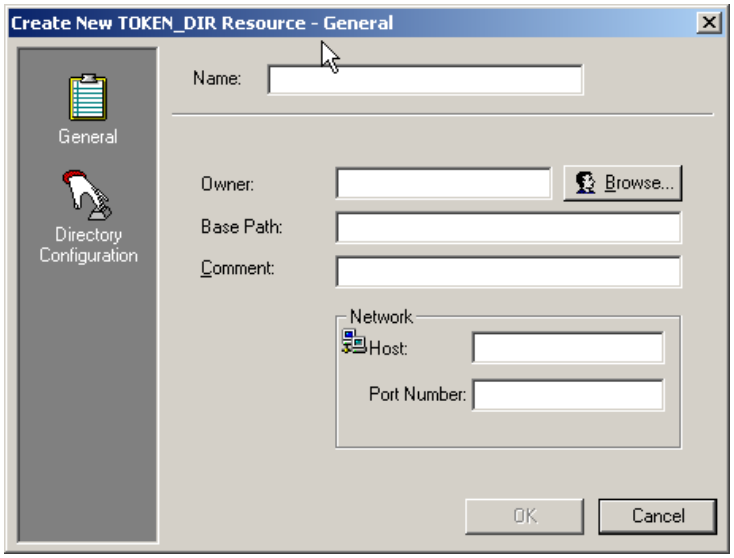
Token Directory

A *token directory* is an LDAP directory in which the CA SSO Server stores the users' session information. Session information includes the session ID, client IP address, username, and the last heartbeat time. During the authentication process, the CA SSO Server uses the token directory to save token and user information. If a CA SSO Server fails, the backup server can use the token directory to fetch token information for unknown tokens.

The token directory solution supports configurations with a very large number of CA SSO Servers. If the CA SSO Servers each stored their own token information, a lot of network traffic would be generated by the need to synchronize memory between servers. Instead, one token directory can be used remotely for all the CA SSO Servers.

Define Properties for a Token Directory

Whenever you define a new token directory, you must specify its properties by completing a set of dialogs associated with the Create New TOKEN_DIR Resource dialog. The following example shows the Create New TOKEN_DIR Resource dialog; the bar on the left lists the associated dialogs.



The dialogs for defining the properties of a token directory are:

General

Defines the token directory's name, owner, base path, host, and port number.

Directory Configuration

Defines administrator information and data store properties.

For a complete explanation of the dialogs and the fields they contain, see the *Policy Manager Help*.

Update Token Directory Properties

To change the properties of an existing token directory

1. Locate the token directory you want to change and double-click its entry in the list.

   This displays the properties dialog.

2. Make any changes that are necessary and click OK when you are finished.

Administrators and Administration Computers

Routine administration of CA SSO includes defining new users, deleting users, assigning users to groups, and resetting user passwords. Other tasks you may occasionally need to perform as an administrator include:

- Updating new users and user groups as your organization changes
- Defining the resources in the policy data store and assigning the resources to user or resource groups when new resources are added to the system
- Writing logon scripts when new applications are added to the system, or modifying the scripts as a result of upgrades to applications

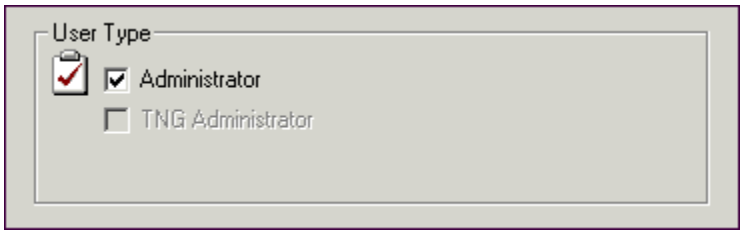
When you installed the CA SSO Server, you defined two administrators. The default administrator names for:

- CA Access Control is ps-admin
- CA Directory is ldap-admin

Additional administrators are defined by adding a special user option called *Administrator* to the user definitions. This option allows the user to do things that an ordinary user cannot do. For example, a user with the administrator option can perform all administrative activities, including define and update users, groups, and resources.

Create an Administrator

You can add the Administrator option when you define a new user or you can add it to an existing user's definition. The Administrator option is specified on the User Options dialog for a particular user:



Important! You can only define administrators in the CA Access Control data store, but normal users must be created in the CA Directory LDAP (ps-ldap) data store.

Important! When defining an administrator, the name you enter in the User Name field cannot contain a space. For example, you cannot enter **John Smith** as the value for User Name, but you can enter **JohnSmith**, **John-Smith**, **jsmith** or **John_Smith**.

To define an administrator

1. Right-click in the right pane and select New, User

   The Create New User – General dialog appears

2. Enter the following information as a minimum:

   User Name: Enter administrator user name

   Authentication Method: EAC (use the browse button)

3. Select the User Options icon from the left pane.

   The Create New User – User Options dialog appears.

4. Check the following options

   User type: Administrator

   Password Features: Password Never Expires

5. Select the General icon from the left pane

   The Create New User – General dialog appears. The Change Password button is now enabled.

6. Click the Change Password button and enter and confirm the new administrator password.

Note: You must now grant the new administrator access rights for the computer on which they will use the Policy Manager to administer the CA SSO Server.

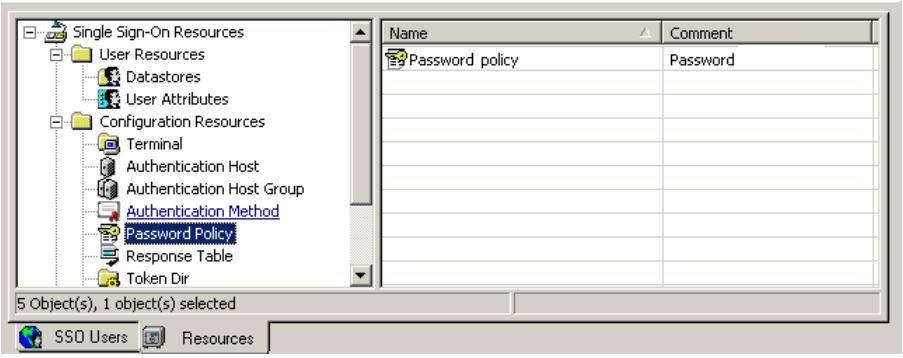
Define a Password Policy

A *password policy* is a set of rules that ensures that users select reliable passwords. The password policy defines password parameters such as the minimum number of characters in a password and the maximum time until expiration.

To display a list of defined password policies

1. Open the Configuration Resources folder
2. Click the Password Policy object.

   The following window displays in the workspace with the defined password policies listed in the Name column.



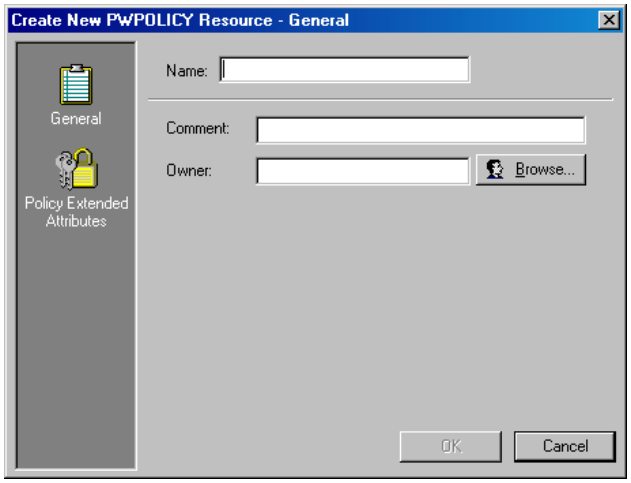
   From the list of password policies you can do the following tasks:

   - Add a new password policy
   - Remove a password policy
   - Locate a password policy in the list
   - Change the properties of a password policy

3. Select the Edit menu or right-click in the list of password policies and then select what you want to do from the command list.

Define Password Policy Properties

Whenever you add a new password policy, you must specify its properties by completing a set of dialogs associated with the Create New PWPOLICY Resource dialog. The following example shows the Create New PWPOLICY Resource dialog; the bar on the left lists the associated dialogs.



The dialogs for defining a password policy's properties are:

General

Defines the password policy's name and owner.

Policy Extended Attributes

Defines specific password characteristics (such as minimum and maximum length of the password, what types of characters can be used in the password, how many days before the password expires, how many past passwords to retain, and other parameters).

When defining password policies, you must complete the General and Password Extended Attributes dialogs.

For a complete explanation of all of the dialogs and the fields they contain, see the *Policy Manager Help*.

Define a Generic Password Policy

Generic password policies apply to all applications. These can be overwritten by application-specific password policies.

To define a generic password policy

1. Right-click the list of password policies, and then choose New.
2. Enter _default in the Name field.

   Any application that does not have a specific password policy linked to it automatically uses a password policy named _default.
3. Enter the name of the policy owner in the Owner field.
4. Define any required password characteristics in the Policy Extended Attributes dialog.

More information:

Managing Passwords (see page 131)

Define an Application-Specific Password Policy

Application-specific password policies are applied to particular applications.

To define a password policy that will apply to specific applications

1. Right-click the list of password policies, and then choose New.
2. Enter a Name for the password policy.
3. Enter the name of the policy owner in the Owner field.
4. Define any required password characteristics in the Policy Extended Attributes dialog, then click OK.
5. Expand the Application Resources folder, and click the Application folder.
6. In the application list window, double-click the application you want to have the password apply to.
7. In the View or Set APPL Properties – General dialog, click the Authentication button.
8. Select the Password Policy you defined in step 2, then click OK.
9. Repeat steps 6-8 for every application you want to have the policy apply to.

More information:

Managing Passwords (see page 131)

Update Password Policy Properties

To change the properties of an existing password policy

1. Display the list of password policies.
2. Locate the policy you want to change and double-click its entry in the list.

   This displays the View or Set PWPOLICY Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

Define Special Characters to use with a Password Policy

The ability to define a subset of Special Characters to use for a Password Policy, instead of relying on a hard-coded pre-defined list, is available from the selang AC command prompt. To specify the set of special characters used by CA SSO auto-generated passwords, use the following procedure:

From a command prompt

1. Run the following command

   ```
   selang
   ```
2. At the selang> command prompt enter:.

   ```
   editres PWPOLICY <PWNAME> gen_prop(SPEC_CHAR_LIST)
   gen_val(list_of_special_chars_in_hex)
   ```

   Note: The above command defines a "list_of_special_chars_in_hex". For example, "23242526", would specify using special characters "#$%&".

   ```
   editres PWPOLICY ("Test Policy") gen_prop(SPEC_CHAR_LIST)
   gen_val(23242526)
   ```
3. Exit selang by issuing the following command from the selang command prompt.

   ```
   exit
   ```
4. Restart the Policy Server.

List of Special Characters

The following lists the special characters that can be defined.

Note: A space character is defined as a special character.

Character	Decimal	Hex
SPACE	32	20
!	33	21

Character	Decimal	Hex
#	35	23
$	36	24
%	37	25
&	38	26
'	29	27
(	40	28
)	41	29
*	42	2A
+	43	2B
-	45	2D
.	46	2E
:	58	3A
;	59	3B
<	60	3C
=	61	3D
>	62	3E
?	63	3F
@	64	40
[	91	5B
\	92	5C
]	93	5D
^	94	5E
_	95	5F
`	96	60
|	124	7C
~	126	7E

Special Characters and HTML Scripts

Most special characters can be used in the password field within an HTML script without causing problems; however, we recommend that you first verify in a test environment that they work correctly before using them in your production environment.

Note: The "\" character will cause problems and must not be used.

Authentication Related Procedures

The following topics describe procedures related to authentication.

Define Authentication Hosts

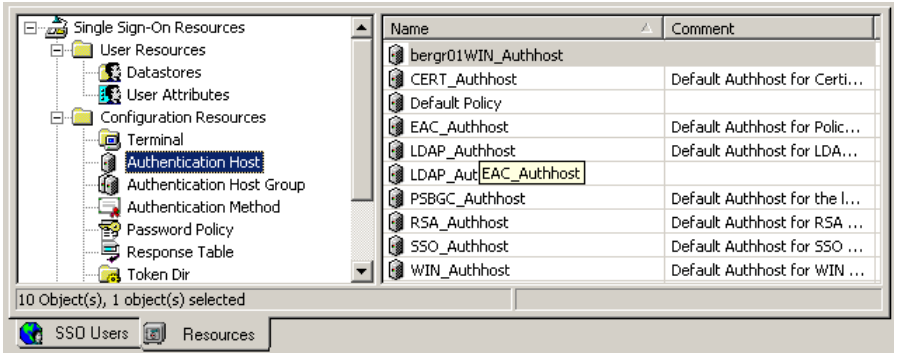
An *authentication service* authenticates entities (people or computers) according to a specified authentication protocol. It is a security service that confirms the identity claimed by or for an entity.

An *authentication host* represents an authentication service running on a local or remote computer and performs identity verification for users. The authentication host is used to define:

- Authentication information that represents the authentication service (authentication method and provider) and attributes used for this authentication host
- Authorization rules that determine who can use the authentication host services
- Mapping rules that map the user that is known to the authentication service to the user defined in one of the user data stores

To display the list of defined authentication hosts

1. Expand the Configuration Resources folder.
2. Click the Authentication Host folder:



From the list of authentication hosts you can do the following tasks:

- Add a new host
- Remove a host
- Locate a host in the list
- Change the properties of a host

To perform any of these actions, use the Edit menu or right-click in the authentication hosts list and then select from the command list.

The Create New AUTHHOST Resource Dialog

Defining an authentication host requires you to specify its properties on a set of dialogs associated with the Create New AUTHHOST Resource dialog:

The bar on the left lists the dialogs used to define the properties of an authentication host. The following list briefly explains the function of each dialog.

General

Defines the name, owner, authentication method, user data store, and default access permissions for the authentication host.

Authorize

Lists who can access this authentication host and what access permissions they have.

Note: You can also grant permission for users to use the authentication host through the default access settings that are defined from the General dialog.

Authentication Information

Defines the provider of the authentication method and any authentication parameters if required.

User Mappings

Specifies user account mapping rules that map the user that is known to the authentication service to the user defined in one of the user data stores.

Backward Compatibility

Specifies optional settings to make the authentication host backward compatible so it accepts identification strings from older versions of the product.

Miscellaneous

Sets day and time restrictions for the authentication host.

Default AUTHHOST Objects

To simplify configuration, the CA SSO Server installs an authentication host object for each of the supported authentication methods. These authentication hosts are created with a default Read access which allows anyone to use them for authentication.

Note: The default authentication host objects created, are assigned an auto-generated ticket encryption key during installation so that they provide an out-of-the-box configuration.

Important! The LDAP_Authhost and EAC_Authhost which are created automatically are used for Policy Manager authentication. You must not delete these as you cannot access the system without them.

Define an Authentication Host

To define an authentication host

1. Define an authentication method.
2. Right-click the list of authentication hosts, and then choose New.
3. On the Create New AUTHHOST Resource – General dialog, specify the properties for this authentication host. You must provide the following details as a minimum:
   - The authentication host name

     Note: For NTLM authentication, the host name can be either the domain name that NTLM authentication uses or any desired logical name. Select the appropriate one for your configuration. The method of authentication used by the authentication host

     The authentication method you specify for this host must match the authentication method that the authentication process associated with this host uses. For example, if an authentication host represents the X.509 authentication method, then choose CERT.
   - The user data store used by the authentication host

4. On the Create New AUTHHOST Resource – Authentication Information dialog, identify the authentication process represented by this host:
   - Choose the provider for the chosen authentication method.
   - If required, enter the authentication method attributes.

     For example, choose a domain for the WIN authentication method.
   - If required, modify the advanced authentication information as needed.

   Note: For LDAP authentication you can use the advanced authentication information to change the way the user name entered during logon is mapped to the user name known to the authentication service.
5. To change the default way the user name used for authentication is mapped to the corresponding user name defined in the user data store, modify the user mapping information on the Create New AUTHHOST Resource – User Mappings dialog.
6. To define who can access this host and what access permissions they have, add accessors on the Create New AUTHHOST Resource – Authorize dialog.

   Note: Users can also be granted permission to use the authentication host through the default access settings as defined from the General dialog.
7. If authentication method used by the authentication host is CERT or NTLM, define backward compatibility options on the Create New AUTHHOST Resource – Backward Compatibility dialog.

For detailed procedures, see the *Policy Manager Help*. The online help also contains a complete explanation of all the dialogs and the fields they contain.

More information:

Define an Authentication Method (see page 109)

Authentication Information

Authentication information is used to match the authentication host to the authentication process. Use the Create New AUTHHOST Resource – Authentication Information dialog to choose the authentication method provider (if there is more than one provider associated with the authentication method) and any additional authentication attributes required for the particular provider selected. Not all authentication methods require additional attributes.

Provider	Instructions for Defining Authentication Information
CERT	In the Advanced Authentication Information dialog, specify the Ticket Encryption Key for the X.509 authentication method. **Note:** For CERT authentication, you also need to specify backward compatibility options.
WIN	Specify the domain name or the name of the stand-alone Windows server that the WIN authentication method is verified against.
LDAP	Specify the authentication directory for the LDAP authentication provider you selected. Use the advanced authentication information to specify pre-authentication user mapping.

Note: The CA SSO Server does not accept CA SSO tickets generated with default or empty encryption keys by default. To change this behavior, change the AllowDefaultEncKey General CA SSO Server setting to Allowed.

More information:

Configuring the CA SSO Server (see page 357)

User Mapping Information

To simplify the logon process, you can define how the user name entered during logon is mapped to a known user. You can set this mapping to happen either before or after authentication:

Pre-authentication user mapping

After the user has entered their logon credentials, the authentication service maps those credentials to the user name stored in the data store. This user name is then used to authenticate the user.

This method of user mapping lets you manipulate the logon information before it is processed for authentication.

Note: Pre-authentication user mapping is only available for the LDAP authentication method.

Post-authentication user mapping

After the user has successfully authenticated, the CA SSO Server maps the authenticated user to a user that is defined in the user data store.

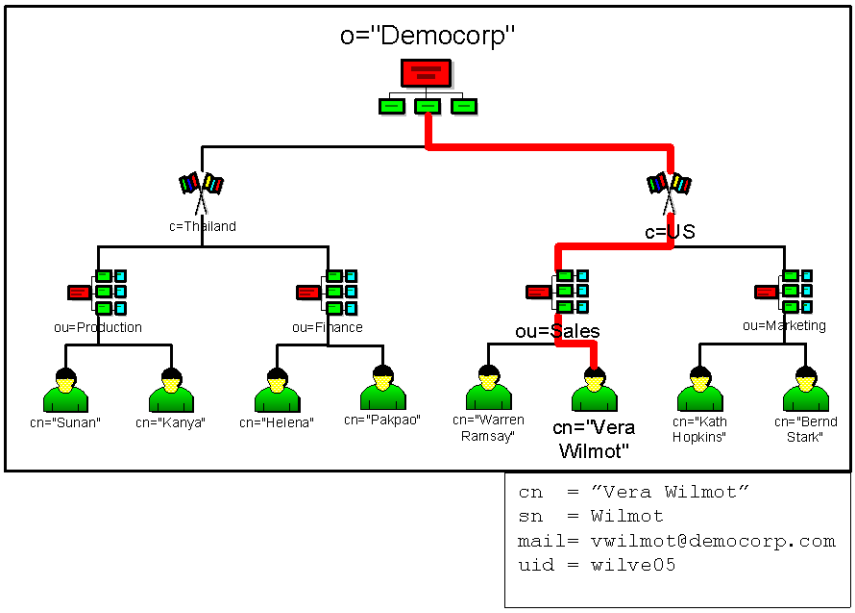
This method of user mapping lets you use a different user data store to check the authorization level of the user. For example, the CA SSO Server checks if the user is permitted to use the selected authentication method.

The information you need in order to define pre-authentication user mapping, is identical to the information you need in order to define post-authentication user mapping. For pre- and post-authentication user mapping, you need to define:

- The container format
- The user format
- The database field that is used to search for the user
- Whether the search is recursive through all sub-containers

  Note: CA SSO Authentication does not support hierarchical user data stores. This means that you cannot use recursive search with CA SSO authentication.

Consider the following illustration of an example LDAP user data store:



In this example, Vera's distinguished name, which is required to identify her, is:

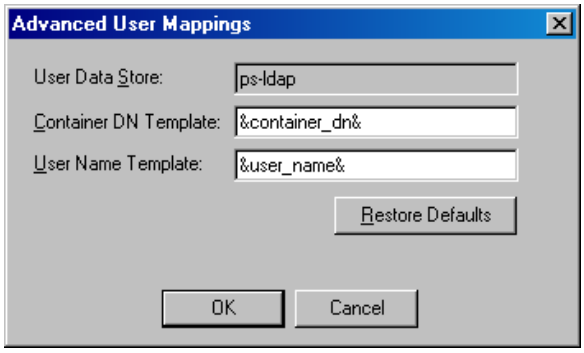
```
cn="Vera Wilmot",ou=Sales,c-US,o="Company A"
```

By defining an authentication host and defining user mappings, you can let Vera logon using her common name (cn), email address, or any other attribute assigned to her user record. This is pre-authentication user mapping.

Once the user is authenticated, you can use post-authentication user mapping to map the authenticated user to a different user data store. The user credentials can be different between the two user data stores. This means that you can use an existing user data store for authentication and a different data store, with an extended schema, to handle authorization.

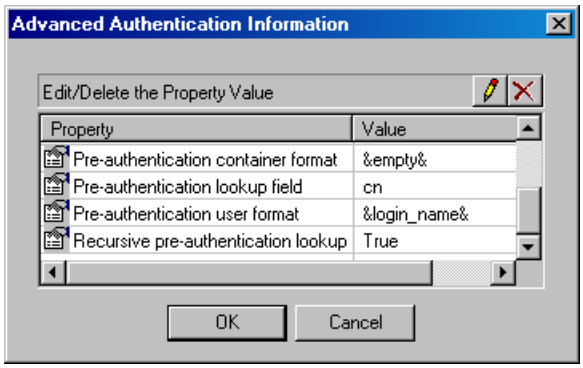
To configure post-authentication user mapping

- Use the Create AUTHHOST Resource – User Mappings dialog.



To configure pre-authentication user mapping

1. Open the Create New AUTHHOST Resource – Authentication Information dialog for an LDAP authentication host
2. Click the Advanced Authentication Information button.



3. Fill out the Advanced Authentication Information dialog.

Container and User Format Keywords

Container and user format keywords are used in the dialogs where you define user mapping (refer to the two preceding screen captures). These keywords define the format of the user name and container name. Use the following keywords for both pre- and post-authentication user mapping:

Keyword	Description
&login_name&	The string that the user enters in the logon screen as it was passed from the authentication plug-in to the CA SSO Server.

Keyword	Description
&user_name&	The portion of the logon string representing the name of the user. If the name is in DN format, it will be truncated to the first comma. If the name begins with “Name By”, this prefix will be removed. For example: ■ If the logon string is “ps-admin”, then &user_name& is also “ps-admin”. ■ If the logon string is: “cn=john, ou=develop, o=ps”, then &user_name& is “john”.
&container_dn&	The domain name of the logon string container, relative to the user data store base path. If there is no container in the logon string then this keyword is empty. For example: ■ If the logon string is “cn=john, ou=develop, o=ps” and the user data store base path is “o=ps”, then &container_dn& is “ou=develop”. ■ If the logon string is “cn=john, o=ps and the user data store base path is “o=ps”, then &container_dn& is empty.
&empty&	Use this keyword to leave the container DN empty.

Post-Authentication User Mapping Scenario

Barry, the administrator for Company1, sets up a TSS directory for authentication and a CA Directory user data store. To let users (for example, Sherry) authenticate with the TSS directory and be mapped to the CA Directory user data store, Barry creates a user directory pointing to the TSS server (tss-dir) and a user directory pointing to the CA Directory server (etr-dir).

Barry sets the TSS directory as follows:

- Sherry’s full user name

  `cn=Sherry, host=host1, o=company1, c=us`

- The base path of the directory

  `host=host1, o=company1, c=us`

  This means that the &container_dn& is empty and &user_name& is Sherry.

Barry sets CA Directory user data store as follows:

- Sherry’s full user name
  `cn=Sherry_LDAP, ou=develop, o=ps`
- Sherry has permissions to use the LDAP authentication method.

To map between the authentication directory (tss-dir) and the user data store (etr-dir), Barry now needs to create an authentication host with the following properties:

Dialog	Field	Value
General	Authentication Method	LDAP
	User Data Store	etr-dir
Authentication Information	Provider	TSS
	Authentication Directory	tss-dir
Advanced User Mapping	Container DN Template	ou=develop
	User Name Template	&user_name&_LDAP

Database Lookup

Database lookup lets you extend pre- and post-authentication user mapping by mapping a user to a specific field of a directory.

Note: Database lookup is only supported by LDAP authentication providers.

Pre-Authentication Lookup Scenario

Rita, the administrator of Company2, sets up an Active Directory for user authentication and an CA Directory user data store. To let users (for example, Daniel) authenticate with Active Directory using their NT account name (SAM account name), Rita creates a user directory pointing to the Active Directory server (ad-dir) and a user directory pointing to the local CA Directory data store (etr-dir).

Rita sets Active Directory as follows:

- Daniel's full user name

  `cn=Daniel, ou=dev, ou=comp, dc=domain1, dc=company2, dc=com`

- The base path of the directory

  `ou=comp, dc=domain1, dc=company2, dc=com`

- Daniel's SAM account name is dani

  This means that the &container_dn& is ou=dev and &user_name& is Daniel.

Rita sets the CA Directory user data store as follows:

- Daniel is the common name
- Daniel has permissions to use the LDAP authentication method.

To let Daniel log on using his NT account, Rita now needs to create an authentication host with the following properties:

Dialog	Field Value	
General	Authentication Method	LDAP
	User Data Store	etr-dir
Authentication Information	Provider	AD
	Authentication Directory	ad-dir
Advanced Authentication Information	Pre-authentication container format	&container_dn&
	Pre-authentication lookup field	SAMAccountName
	Pre-authentication user format	&user_name&
	Recursive pre-authentication lookup	True
Advanced User Mapping	Container to search in	&empty&
	Value to search	&user_name&

Note: The lookup needs to be recursive because user Daniel is not directly under the base path.

When Daniel now enters dani in the LDAP authentication dialog, he is mapped to “cn=Daniel, ou=comp, dc=domain1, dc=company2, dc=com” in the Active Directory.

Post-Authentication Lookup Scenario

John, the administrator of Company3, sets up an Active Directory for user authentication and a CA Directory user data store. John creates a user directory pointing to the Active Directory server (ad-dir) and a user directory pointing to the local CA Directory data store (etr-dir). The existing Active Directory has users defined using their email address (for example, Amelia@Company3.com).

John sets the CA Directory user data store as follows:

- Amelia's full user name

  `cn=Amelia, ou=dev, ou=comp, dc=domain1, dc=company3, dc=com`

- The base path of the directory

  `ou=comp, dc=domain1, dc=company3, dc=com`

- Amelia's email address stored as a comment (description field)

  `Amelia@Company3.com`

- Amelia's LDAP password

To let Amelia log on and then be mapped to the CA Directory user data store, John now needs to create an authentication host with the following properties:

Dialog	Field	Value
General	Authentication Method	LDAP
	User Data Store	etr-dir
Authentication Information	Provider	AD
	Authentication Directory	ad-dir
Advanced Authentication Information	Recursive pre-authentication lookup	True
User Mapping	Search	selected
	Attribute to search by	description
Advanced User Mapping	Container to search in	&empty&
	Value to search	&user_name&

Note: The lookup needs to be recursive because user Amelia is not directly under the base path.

When Amelia now logs on using her email address (Amelia@Company3.com), she is mapped to user "cn=Amelia, ou=dev, ou=comp, dc=domain1, dc=company3, dc=com" in the CA Directory user data store.

Post-Authentication Lookup Example

John, the administrator of Company3, sets up an Active Directory for user authentication and a CA Directory user data store. John creates a user directory pointing to the Active Directory server (ad-dir) and a user directory pointing to the local CA Directory data store (etr-dir). The existing Active Directory has users defined using their email address (for example, Amelia@Company3.com).

John sets the CA Directory user data store as follows:

- Amelia's full user name

  `cn=Amelia, ou=dev, ou=comp, dc=domain1, dc=company3, dc=com`

- The base path of the directory

  `ou=comp, dc=domain1, dc=company3, dc=com`

- Amelia's email address stored as a comment (description field)

  `Amelia@Company3.com`

- Amelia's LDAP password

To let Amelia log on and then be mapped to the CA Directory user data store, John now needs to create an authentication host with the following properties:

Dialog	Field	Value
General	Authentication Method	LDAP
	User Data Store	etr-dir
Authentication Information	Provider	AD
	Authentication Directory	ad-dir
Advanced Authentication Information	Recursive pre-authentication lookup	True
User Mapping	Search	selected
	Attribute to search by	description
Advanced User Mapping	Container to search in	&empty&
	Value to search	&user_name&

Note: The lookup needs to be recursive because user Amelia is not directly under the base path.

When Amelia now logs on using her email address (Amelia@Company3.com), she will be mapped to user "cn=Amelia, ou=dev, ou=comp, dc=domain1, dc=company3, dc=com" in the CA Directory user data store.

Backward Compatibility Information

CA SSO no longer requires users to know the authentication host parameters in order to authenticate. Instead, CA SSO uses logical authentication host naming, which lets users authenticate using the name of the authentication host.

Note: Backward compatibility information must be filled for CERT and NTLM authentication methods.

For backward compatibility, CA SSO uses extra authentication host data to match the authentication host to the authentication process. This information is used for upgraded AUTHHOST objects to support existing identification strings from older versions of the product.

The data you specify on the Create New AUTHHOST Resource – Backward Compatibility dialog depend on the authentication method, because different authentication methods require different data.

For example, for an authentication host that represents the WIN authentication method that is verified against a domain called Domain1, select WIN as the authentication provider and enter Domain1 for DOMAIN on the Backward Compatibility dialog.

The following table lists the authentication methods and the data required for backward compatibility for each one:

Authentication Method	Backward Compatibility Data	Description
NT	DOMAIN	The name of the Windows domain or the name of the stand-alone Windows server that will perform the authentication
LDAP	HOSTNAME	The host name of the computer that runs the LDAP server or directory
	PORT	The LDAP server port
SecurID	HOSTNAME	The host name of the computer that runs the RSA SecurID Agent
SSO	HOSTNAME	The host name of the CA SSO Server computer
CERT	AUTH_HOST_NAME	The host name of the computer running the X.509 authentication agent where the X.509 authentication is performed

Define Groups of Authentication Hosts

By creating a group of authentication hosts, authorized users can be authenticated with all of the hosts that are members of the group.

To display a list of defined groups, open the Configuration Resources folder and click the Authentication Host Group folder.

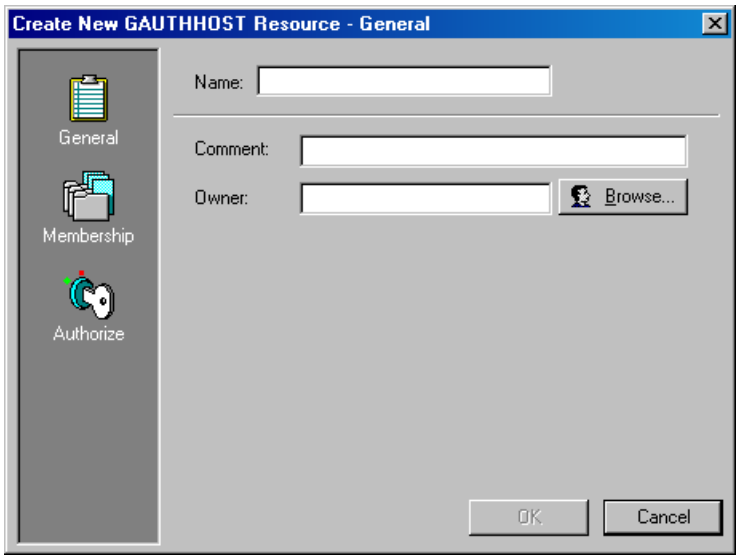
From the list of groups you can do the following tasks:

- Create a new group
- Remove a group
- Locate a group in the list
- Change the properties of a group.

To perform any of these actions, use the Edit menu or right-click in the list of groups and the select from the command list.

Define Properties for a Group of Authentication Hosts

Whenever you add a new authentication host group, you must specify its properties by completing the Create New GAUTHHOST Resource dialog:



The dialogs to define the properties of an authentication host group are:

General

Defines the group's name and owner.

Membership

Lists the authentication hosts in the group.

Authorize

Lists who can access the authentication hosts in this group and what access permissions they have.

For a complete explanation of all of the dialogs and the fields they contain, see the *Policy Manager Help*.

Update the Properties of a Group of Authentication Hosts

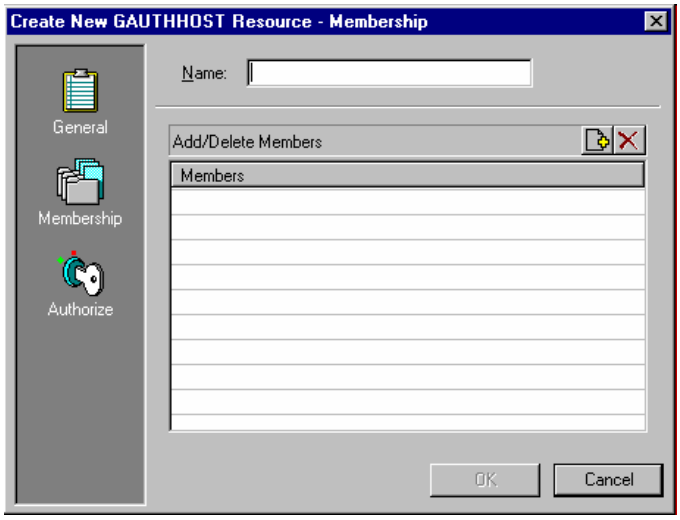
To change the properties of an existing authentication host group

1. Display the list of groups.
2. Locate the group you want to change and double-click its entry in the list. This displays the View or Set GAUTHHOST Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

Add Members to a Group of Authentication Hosts

Only existing authentication host entries can be members of an authentication host group; therefore, each entry that is to be a member of the group must first be defined.

You can add members to a group either while you are creating the group or after the group is created. In either case, members are added using the Membership dialog:



To add a member to a group

1. Click the Add icon in the Add/Delete Members section to display a list of defined entries.
2. Select the entries that are to be members of this group from the list and click OK.

   The entries you selected are added to the list of members.
3. If this is:
   - a new group, click OK on any dialog in the dialog list to create the authentication host group containing the specified members.
   - an existing group, click OK to update the list of members.

Remove Members from a Group of Authentication Hosts

To remove members from the group

1. Display the Membership dialog.
2. Select the member you want to remove from the group.
3. Click the Delete icon in the Add/Delete Members section to remove the member.

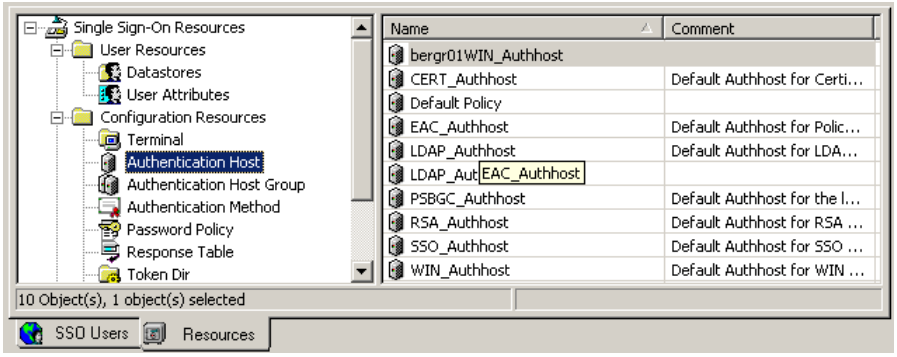
Define an Authentication Method

Authentication method objects are used to grant or deny access to an authentication method for users or groups.

To display a list of defined authentication methods

1. Open the Configuration Resources folder.
2. Click the Authentication Methods object.

   The following window displays in the workspace with the defined authentication methods listed in the Name column.

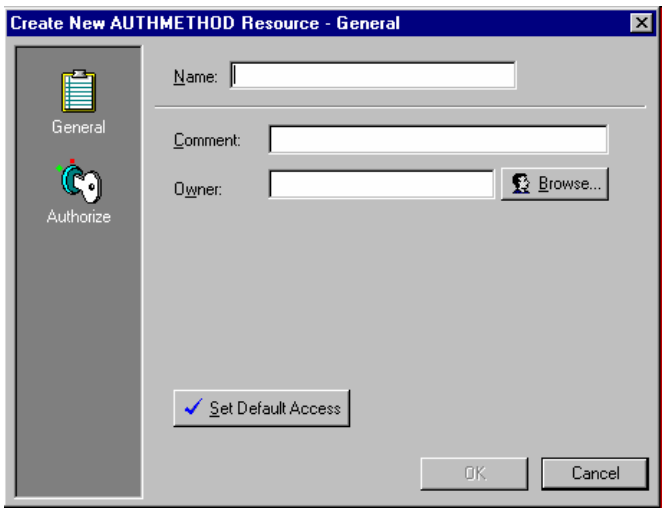


   From the list of authentication methods you can locate an authentication method in the list or change the properties of an authentication method.

   To perform either of these actions, use the Edit menu or right-click in the list of authentication methods and then select from the command list.

Define Authentication Method Properties

Whenever you add a new authentication method, you must specify its properties by completing a set of dialogs associated with the Create New AUTHMETHOD Resource dialog. The following example shows the Create New AUTHMETHOD Resource dialog; the bar on the left lists the associated dialogs.



The dialogs to define the properties of an authentication method are:

General

Defines the authentication method's name, owner, and other basic information about this authentication method.

Authorize

Lists who can use this authentication method and what access permissions they have.

For a complete explanation of all of the dialogs and the fields they contain, see the *Policy Manager Help*.

Update Authentication Method Properties

To change the properties of an existing authentication method

1. Display the list of authentication methods.
2. Locate the authentication method you want to change and double-click its entry in the list. This displays the View or Set AUTHMETHOD Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

Managing Application Resources

Every application to be available through CA SSO must be represented by a record in the data store. Applications can be grouped together to simplify data store administration. When applications are grouped, you can grant users permission to access any application in the group.

From the Application Resources folder in the Policy Manager you can define:

- Applications
- Application groups

Application Types

Application types are defined by the properties that are assigned to them. You can define the following types of applications:

- Common applications
- Restricted applications
- Master applications
- Container applications
- Sensitive applications

More information:

Working with the Administrative Data Store (see page 229)

Common Applications

A common application is one that all users in the system are allowed to use. It common application appears on the application list of every user.You can define an application as common but deny certain users access to the application. To revoke the access of specific users to an application, change the access type in the user's listing in the application's access control list. An application's access control list is defined by the AZNACL property.

Restricted Applications

There are three types of restricted applications:

Disabled

No logon is allowed to a disabled application. This feature is useful when you need to make a change to an application and you do not want any users to log on to the application while you make it. The disabled application appears in the application menu list, but if a user selects the application, the logon is terminated with an appropriate message.

Restricted

An application can be restricted for use to certain days of the week, certain hours of the day, or any combination of days and hours. The application always appears in the application list, but users can only log onto the application during the specified days and times.

Hidden

If an application is marked as hidden, it does not appear in the application menu; however, the application is still a valid application. Usually master applications are marked as hidden.

Master Applications

An application that supplies the application authentication method and passwords (if needed) to other applications is called a *master application*. Master applications are useful when there are several password-based applications running on the same application host, all performing the same password verification. For example, if a Telnet application and an FTP application are defined and running on the same host and both use the same basic authentication method, then the Telnet application can be defined as the master application of the FTP application.

When an application is linked to a master application, it uses the authentication method and password of the master application. CA SSO does not reference any password-related properties in the linked (dependent) application's record. However, if there is a listing for the application host property in the record of the linked application, then the name of the application host is taken from the record of the linked application and not from the master application's record.

Note: Since the password of a master or linked application is the same password and the password is physically located only in the database where the master application resides, if you change the password of a linked application using the selang utility you must also change the password in the master application. However, if you use the Policy Manager to change the password, then you can change it in either the linked application **or** the master application. The Policy Manager automatically changes the password in the master application when you change it in the linked application.

Master applications can be marked as hidden, meaning that they do not appear in the application list presented to the user. Generally, a master application is marked as hidden if it is a dummy (placeholder) application that was created only to supply passwords to a collection of applications.

Container Applications

A *container application* assembles related applications that are to be displayed together on the user's workstation. When a user selects an icon representing a container application, CA SSO displays a window or submenu that shows the applications contained by the container application—it displays the contained applications. Usually a container application has no logon script attached to it, since it is used only as a visual placeholder.

Being authorized to log onto a container application does not automatically mean that the user is authorized to log on to all of the applications it contains. The user must be specifically authorized to each application individually or be assigned to a group that is authorized to that application. Users see only the applications they are authorized to access.

A container application can be nested in a higher-level container application, which means that an application can be both a container and a contained application. An application can also be assigned to more than one container application.

An application is defined in the database as a container application by the IS_CONTAINER property. For container applications, the properties in the APPL record that are used are CAPTION, ICONFILE, and ICONID. The other properties in the record are ignored.

Sensitive Applications

When an application is marked as sensitive, the user is forced to reauthenticate more frequently to use the application. The frequency is set using a default. For a normal application, the CA SSO Server checks the normal expiration time of the CA SSO security token where the default expiration time is eight hours. For a sensitive application, the CA SSO Server checks the sensitive expiration time where the default is five minutes.

For example, when using the default expiration times, if a user carries out primary authentication at 9:00 a.m. and selects an application at 9:10 AM, if the application is a normal application, the user can access it directly. However, if it is a sensitive application, the user is prompted to reauthenticate before using the application

The password a user must enter to log on to a sensitive application is the password used for primary authentication, not the password of the specific application. For example, if primary authentication is set to LDAP and the Telnet application is marked as sensitive, the user is asked to provide the LDAP password when selecting the Telnet application. When entered, the primary authentication process is run before beginning the logon process.

To mark an individual application as sensitive, set the IS_SENSITIVE property in its application record. The expiration time for all sensitive applications is set via the Policy Manager:

1. Open CA SSO Server Settings
2. Double-click Communication
3. Double-click SensitiveExpiration and set to required value.
4. Click OK twice to close the dialogs.

Applications on Windows Vista, Windows 2008, and Windows 7

Windows Vista, Windows 2008, and Windows 7 manage applications differently than other platforms; so, the CA SSO Client also behaves differently when you try to launch applications.

On Windows Vista, Windows 2008, and Windows 7, applications are classified into two groups:

- Standard applications
- Elevated applications

Standard applications are applications that do not need to modify or write to system files. *Elevated applications* are applications that may need to modify or write to restricted system files or registry keys. As a standard user on these operating systems you can run standard applications only. To run elevated applications on Windows Vista, Windows 2008, or Windows 7, you must have administrator privileges.

Administrators can create an application on the CA SSO Server using Policy Manager and mark it "Run as Administrator". Select this check box if the application must run in an elevated mode. If you select this check box, a Run As Administrator option is visible in the menu when you right-click an application. Once this option is enabled, the shield icon appears on the application.

User Account Control (UAC)

User Account Control (UAC) distinguishes between applications that need an elevated execution level. UAC uses color-code prompts to identify an applications security risk. The following prompts are available in UAC:

Elevation prompts for standard users

Vista requires standard users to run without elevated privileges until a task requires elevation. When a task requires elevation, a dialog box prompts the user to input the password of the local administrator account. Once the password has put, the application or operating system feature runs.

Elevation prompts for administrators in Admin approval mode

When an administrator is running on a Vista computer, they function as a standard user until the task that needs to be completed requires elevated privileges. When a task requires elevated privileges, a dialog box is prompts the user to approve the elevation to administrator status.

Launching Applications on Windows Vista, Windows 2008, and Windows 7

Windows Vista, Windows 2008, and Windows 7 manage applications differently from other platforms; so, the CA SSO Client also behaves differently when you try to launch applications.

On these operating systems, applications are classified as standard applications and elevated applications. *Standard* applications are applications that do not need to modify or write to system files. *Elevated* applications are applications that may need to modify or write to restricted system files. As a standard user on these operating systems, you can run standard applications only. To run elevated applications, you must have administrator privileges.

Whenever you try to access an elevated application, Windows displays a User Access Control dialog prompting you for consent to run applications in elevated mode. You may be prompted with a UAC dialog twice based on your application resource settings set in the CA SSO Policy Manager.

To launch applications on Windows Vista, Windows 2008, or Windows 7

1. Log into CA SSO.

   The CA SSO LaunchBar or CA SSO Tools displays a list of applications you can access.

2. Click an application from the application list.

   Based on the settings configured for your application, the CA SSO client does one of the following actions:

   - Displays a User Access Control (UAC) dialog prompting you for consent to run the selected application in an elevated mode.
   - Launches the selected application or a login dialog if you are accessing the application for the first time.

3. (Optional) Enter the administrator credentials in the UAC dialog, and click OK.

   Note: This step is required only if a UAC prompt is displayed on step 3.

   Based on the settings configured for your application, the CA SSO client does one of the following actions:

   - Displays a UAC dialog prompting you to consent to launch the elevated application.
   - Launches the selected application or a login dialog if you are accessing the application for the first time.

4. (Optional) Enter the administrator credentials in the UAC dialog, and click OK.

   Note: This step is required only if a UAC prompt is displayed on step 3.

5. Enter the user credentials for the selected application, and click OK.

The CA SSO client launches the selected application.

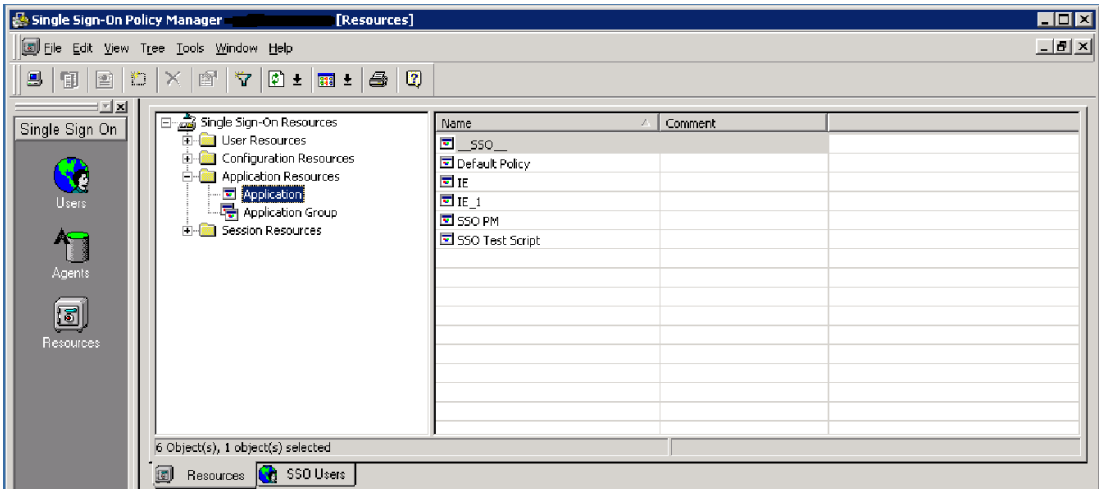
Define an Application

Unlike other resources, applications have user name and password information and an HTML script associated with them. These associations allow CA SSO to supply the user names and passwords of users allowed to access the application and automate the logon process for the applications. The HTML file contains a JavaScript piece that handles and automates logon to the web-based application. Authorized users can access the applications using a personalized application list.

To display a list of defined applications

1. Open the Application Resources folder
2. Click the Application entry.

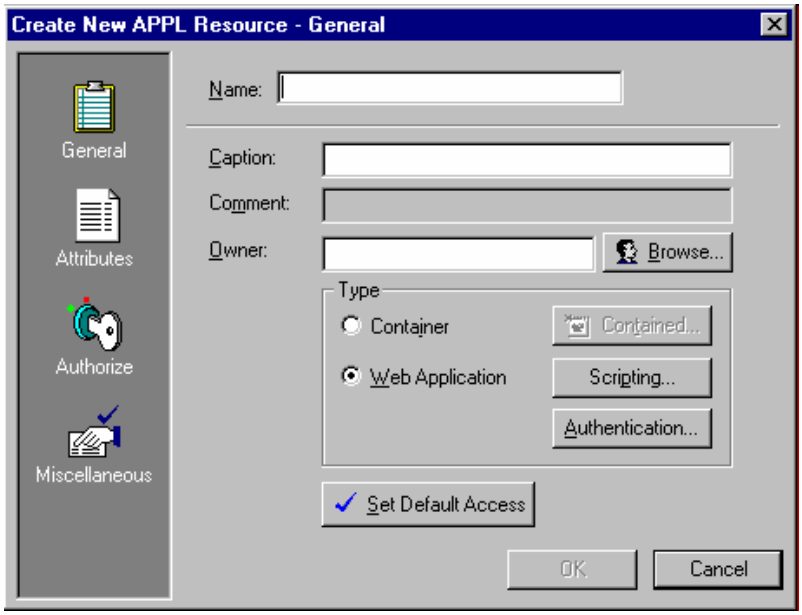
   The following window displays in the workspace with the defined applications listed in the Name column.



   From this list of applications you can add a new one, remove an application, locate an application in the list, or change the properties of an application. To perform any of these actions, select the appropriate command from the Edit menu or right-click in the list of applications to display the pop-up menu and select the command from the command list.

Define Application Properties

Whenever you add a new application, you must specify its properties by completing a set of dialogs associated with the Create New APPL Resource dialog. The following shows the Create New APPL Resource dialog; the bar on the left lists the associated dialogs.



The dialogs for defining an application's properties are:

General

Defines the name and owner of the application, the type of application it is, and other basic information about this application.

Attributes

Specifies application attributes such as a hidden, disabled, Available offline, Run As Administrator used for Windows Vista, Windows 2008, and Windows 7 clients, and an option to attach an icon file.

Authorize

Lists who can access this application and what access permissions they have.

Auditing

Specifies the auditing modes Success, Failure, and Warning.

Miscellaneous

Sets day and time restrictions for using this application.

Update an Application's Properties

To change the properties of an existing application

1. Navigate to the list of applications.
2. Locate the application you want to change and double-click its entry in the list.

   This displays the View or Set APPL Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

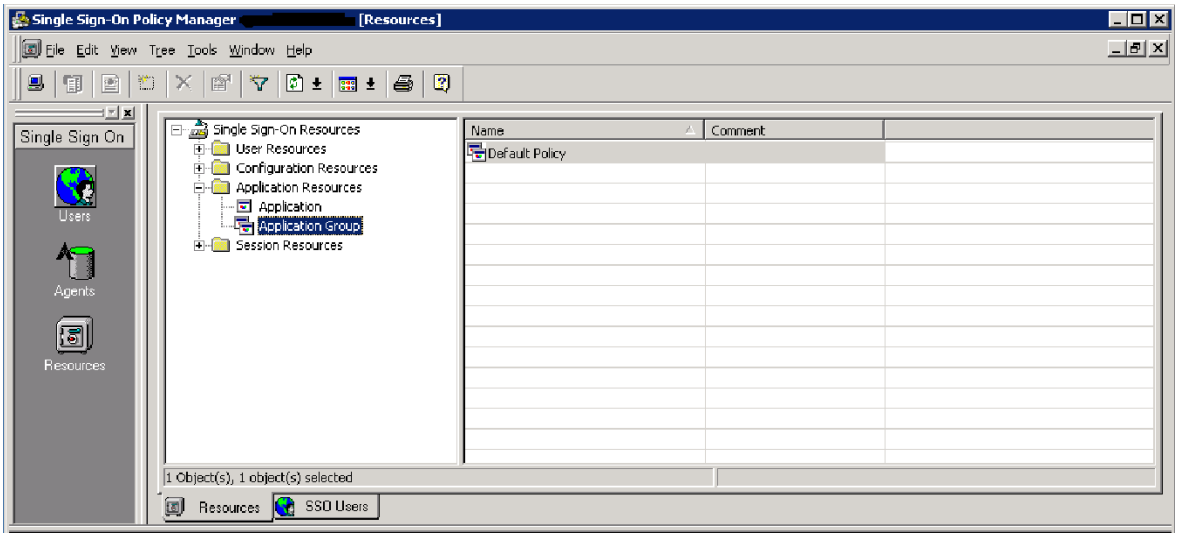
Define an Application Group

Certain users or groups of users tend to use the same set of applications. You can create groups of applications to mirror this work environment. Using application groups makes it easy to change the access rules to multiple applications when user needs change (for example, when a user moves to a different department in the organization).

To display a list of defined application groups

1. Open the Application Resources folder
2. Click the Application Group entry.

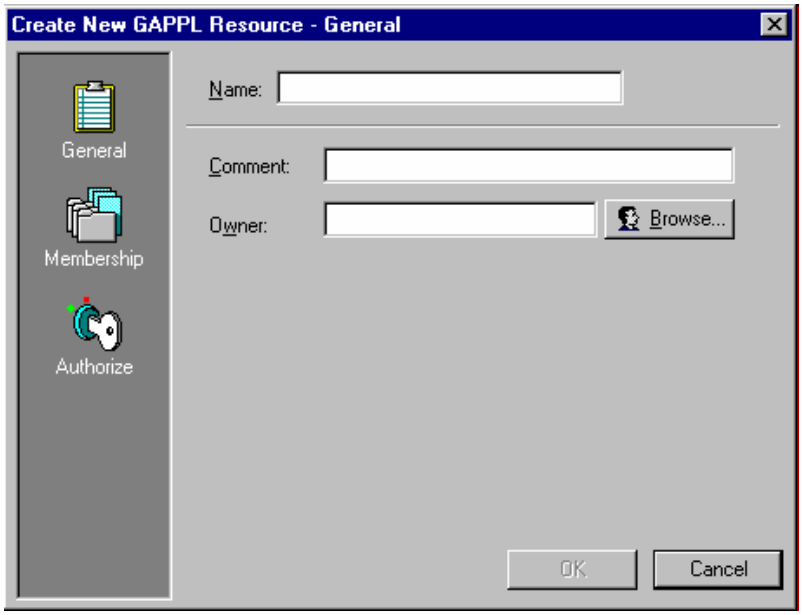
   The following window displays in the workspace with the defined application groups listed in the Name column.



From the list of application groups you can create a new group, remove a group, locate a group in the list, or change the properties of a group. To perform any of these actions, select the appropriate command from the Edit menu or right-click in the list of groups to display the pop-up menu and select the command from the command list.

Define Properties for an Application Group

Whenever you add a new application group, you must specify its properties by completing a set of dialogs associated with the Create New GAPPL Resource dialog. The following example shows the Create New GAPPL Resource dialog; the bar on the left lists the associated dialogs.



The dialogs for defining the properties of an application group are:

General

Defines the group's name and owner.

Membership

Lists the applications in the group.

Authorize

Lists who can access this group of applications and what access permissions they have.

Update an Application Group's Properties

To change the properties of an existing application group

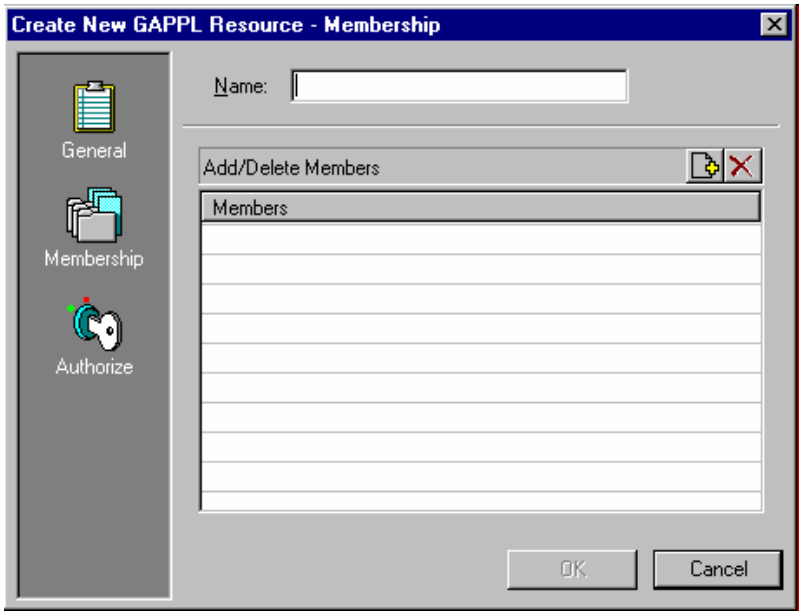
1. Navigate to the list of application groups.
2. Locate the group you want to change and double-click its entry in the list.

   This displays the View or Set GAPPL Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

Add Members

Only existing applications can be members of an application group; therefore, each application that is to be a member of the group must be defined first.

You can add members either while you are creating the group or after the group is created. In either case, members are added using the Membership dialog. The Membership dialog that appears when you are creating a new group is shown next.



1. Click the Add icon in the Add/Delete Members section to display a list of defined applications.
2. Select the applications that are to be members of this group from the list and click OK.

   The selected applications are added to the list of members. If this is a new group, clicking OK on any dialog in the dialog list (shown on the left in the previous example) creates the application group containing the specified members; if this is an existing group, clicking OK updates the list of members.

Removing Members

To remove members from the group

1. Navigate to the Membership dialog.
2. Highlight the group you want to remove.
3. Click the Delete icon in the Add/Delete Members section to remove the member.

   The member is deleted.

Update an Application Record

To update an existing application record in the data store

1. Click on the Applications icon to display the application entry list with a list of applications in the data store.
2. To edit an existing application record, click on it in the list and make the necessary changes in the Details box that appears.
3. When you have made all the changes, click Apply. If you have not made any changes, the Add button remains disabled.

Adding and Updating Application Groups

You add or update application groups in the data store in the same manner using the Application Groups entry list.

Access Permissions

An *access rule* is a piece of information stored in a resource's record that governs the permission of the accessor to work with the resource. Access rules define whether to grant or deny a particular accessor access to a particular resource.

Access types are the kind of access an accessor can have. These access types are different for different resource types. The CA SSO Server lets you store access permissions for all of your resources, making it easier for you to manage and control user access. For example, two access types—EXECUTE and NONE—are used for controlling access to applications.

An access rule can grant one or more different types of access. For example, for the APPL class an access rule can:

- Grant all users within a user data store access to a specific application
- Grant a specific user access to a specific application
- Deny a specific user access to a specific application
- Grant a specific user access to all of the applications in an application group
- Grant all of the users in a user group access to a specific application
- Grant a user with a specific string attribute access to a specific application
- Grant all users (within a user data store) with a numeric attribute within a defined range access to a specific application

There are two types of access rules you can define for all resources: a default access, and an Access Control List (ACL). A resource's ACL overrides any default access permissions.

Note: Owner rules override access permissions defined in an ACL.

Set Default Access Permissions

Default access permissions control what access is granted to any authorized user that does not appear in the resource's list of authorized users (the access control list). The types of access permissions you can assign vary by resource type.

You can set default access permissions for these resources:

- Authentication hosts (AUTHHOST)
- Authentication methods (AUTHMETHOD)
- Applications (APPL)

To specify the default access permission for a resource, click the Set Default Access button on the resource's General dialog. This displays the Set Default Access dialog on which you select the default permissions for this resource.

Case study

Company1 wants to permit all users apart from casual staff (CasualStaff user group) to access its intranet (compIntranet) and to only permit finance (Finance user group) to access the compAccounts application.

For the compIntranet application, Bob the administrator sets EXECUTE to be the default access and adds the CasualStaff group to the ACL with NONE access. For the compAccounts application, Bob sets NONE to be the default access and adds the Finance group to the ACL with EXECUTE access.

Note: Since a Web Agent uses regular expressions to define access rules for more than one resource, default access permission settings are ignored.

Set Access Permissions for a Specific Accessor

In addition to setting default access permissions for resources, you can also set access permissions for individual accessors. Access permissions for a specific accessor control how the accessor can use that resource.

The types of access permissions you can assign vary by resource type.

To specify the access permissions to a resource for individual accessors, you must create an access control list (ACL) for that resource.

You can also ease administration by creating groups and assigning access permissions by group.

More information:

Access Control List (see page 124)
Assign Access Permissions Using Groups (see page 129)

Access Control List

The resource's access control list (ACL) specifies the access permissions that accessors have for the resource. The types of access permissions you can grant an accessor depend on the type of resource. For example, applications use the access types EXECUTE and NONE.

Using the Authorize dialog you can define and manage an ACL for a resource. The Authorize dialog lets you add access rules to the list, remove access rules from the list, and change the values assigned to an access rules in the list. The Authorize dialog is included in the set of dialogs that appear when you create a new resource and in the set of dialogs for specifying the properties of a resource.

Define Generic or Application Resource ACL

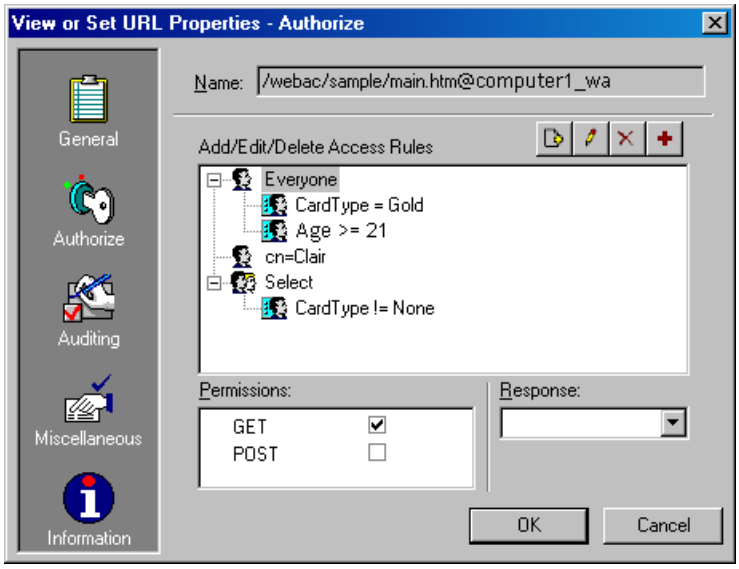
The access control list for generic and application resources holds a list of accessors defined by a set of conditions and paired with access types. Every element in this list defines a specific access type for a specific accessor. A specific accessor is identified by:

- A user or group name from a user data store, or everyone
- Additional conditions that further narrow the accessor according to its attributes or the groups it belongs to.
- Additional conditions must be specified if the accessor is *Everyone* but are optional if the accessor is a user or user group. Additional conditions can use one of the following comparison operators:

Operator	Meaning
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
!=	Not equal to

In this type of ACL, you can compound multiple conditions into one rule. This means that a user can access the resource with the specified access, only if they match all the conditions of at least one of the rules.

An example of the Authorize dialog for a URL resource is shown next:



From this dialog you can see that there are three ACL rules for the URL /example/sample/main.htm that belongs to the Web Agent computer1_wa:

- All users with a Gold card that are of age 21 or over, have GET access to the URL.
- A user with the common name of Clair has a defined access to the URL.

  To view what access permissions Clair has, you have to select cn=Clair in the ACL.
- All members of the *Select* group that have a card (do not have a card type of None) have a defined access to the URL.

  To view what access permissions these users have, you have to select the group in the ACL.

If more than one rule applies to a user, then the highest priority rule takes precedence. The priority of a rule is determined as follows:

1. A rule that specifies a user has the highest priority.
2. A rule that specifies the user's group has a medium priority
3. A rule that does not specify a particular user or group receives the lowest priority.

In the preceding example, even if the criteria for *Everyone* matches Clair (she has a Gold card and is 21 years old or over), the specific access that was defined for her takes precedence.

Similarly, even if a user that has a card and belongs to the *Select* group also matches the criteria defined for *Everyone*, the rules for the group take precedence.

Define Authentication Host and Authentication Method Resource ACL

The access control list for authentication host, group authentication host, and authentication method resources holds a list of accessors paired with access types and a user directory. Every element in this list defines a specific access type for a specific accessor. A specific accessor is identified by a user data store, an attribute, and an attribute value (accessor column).

The following table shows an access control list for the URL /heb/example.htm.

Accessor	Attribute	User Data Store	Access
jane	(User@Mem.com)	Directory(Mem.com)	Access (GET)
cn=Dev	(Group@ps-ldap)	Directory(ps-ldap)	Access (GET)
Developer	(JobTitle@ps-ldap)	Directory(ps-ldap)	Access (GET POST)
Gold	(CardType@ps-ldap)	Directory(ps-ldap)	Access (GET POST)

From this table you can see that user jane from the data store Mem.com has GET access to the /heb/example.htm URL. All users in the group Dev from the data store ps-ldap also have GET access. All users with a job title of Developer from the data store ps-ldap have GET and POST access. In addition, all users from the ps-ldap data store that have a gold card have GET and POST access to the URL.

When defining access to the resource for a user or a group of users in an CA Access Control-type user data store, use the user or group object **name** (for example, jane). However, if you are working with an LDAP or Microsoft Active Directory user data store, use the **RDN** (relational distinguished name) of the user (for example, cn=jane). If you are using the Policy Manager, the cn= prefix is added automatically.

Add an Access Rule

To add an access rule

1. Click the Add Rule icon in the Add/Edit/Delete Access Rules area C
2. Complete the dialog that appears.

   By completing this dialog, you specify the conditions that define who can access this resource.

Editing an Access Rule

To edit an access rule in the ACL, highlight the entry you want to change and click the Edit icon in the Add/Edit/Delete Access Rules area. This displays the conditions that define an accessor.

Removing an Access Rule

To remove an access rule from the ACL

1. Highlight the entry you want to remove
2. Click the Delete icon in the Add/Edit/Delete Access Rules area.

   The selected access rule will be removed from the list immediately.

Define Access Rules Using Regular Expressions

You can use regular expressions to match multiple characters or names when you define URL resources. Using regular expressions simplifies entering resources, since defining one resource with a regular expression can match several resources. For example, a resource named /example/samples/.* matches all URLs under /example/samples including any URLs in subdirectories.

Regular expressions are a combination of literal characters and special characters. Literal characters are the actual characters that are displayed on your screen. Special characters help create a pattern to be matched against an actual resource name. The supported special characters are:

Character	Matches	Example
.	Any character	a.b returns any string that begins with an ‘a’ and ends with a ‘b’
*	The preceding regular expression zero or more times	ab* returns ‘a’, ‘ab’, or ‘a’ followed by any number of ‘b’s
?	The preceding regular expression zero or one time	ab? returns ‘a’ or ‘ab’
+	The preceding regular expression one or more times	ab+ returns ‘ab’ or ‘a’ followed by any number of ‘b’s
^	The start of the string	ab^ returns strings that begin with ‘ab’
$	The end of the string	ab$ returns strings that end with ‘ab’

Note: To use a special character as a literal character, precede it with a backslash. For example, the regular expression *ab returns the string '*ab'.

Additionally, you can use:

- [ ] to indicate a set of characters, either as a list or a range.

  For example, [a-zA-Z0-9] matches any letter or digit.

- ( ) to indicate a set of characters that form a group.

  For example, (ab)* returns the string 'ab', 'abab', or 'ab' followed by any number of 'ab' strings.

- { } to indicate the number of times the preceding regular expression is matched.

  For example, ab{3} returns the string 'abbb' and ab{1,3} returns any string that begins with 'a' and is followed by one, two, or three 'b's.

The most common use of special characters is as a wildcard for part of a path name. Unlike the operating system shells that permit an * (asterisk) to represent all or part of a file name, regular expressions use the * to repeatedly match the previous character in the expression (technically, it matches zero or more times). As a result, just an * or *.* are not valid. Use .* to match any file name; .* means match any character (signified by the dot) and repeat until you find the next regular expression character or the end of the file name (signified by the asterisk).

Regular expressions can be very complex, with literal and special characters strung together to match virtually any resource name imaginable in a variety of ways. The possibilities are too extensive to describe in this guide. Regular expressions are supported in various programming languages such as Perl and awk, and on UNIX operating systems in commands such as grep and vi. For more information on regular expressions, see the man pages on regex or regexp on UNIX or search for "regular+expression+syntax" using an Internet search engine.

Assign Access Permissions Using Groups

To simplify administration, you can define groups of users, applications, authentication hosts, and URLs, and give these groups access permissions to various resources. Doing so gives all of the group members the same access permissions, which reduces administration by avoiding the need to grant access permissions individually. Careful planning of groups can save much administrative overhead.

Here are some ways you can use groups to control access to resources:

- A user group can be granted access to an application, granting all members of the group access to the application.
- A user can be granted access to an application group. The user can then access all of the applications that are members of the group.
- A user group can be granted access to an application group. In that case, every member of the user group can access every member of the application group.
- Members of a user group can be permitted to authenticate themselves with one or more authentication hosts.
- Individual users can authenticate themselves with hosts from one or more authentication host groups. Users are automatically authorized to authenticate themselves with all of the hosts that are members of the authentication host group.
- Members of a user group can be authorized to authenticate themselves with a group of authentication hosts. Members of the user group are automatically authorized to authenticate themselves with all of the hosts that are members of the authentication host group.

You assign access permissions to groups just as you would assign access permissions to individual users.

Note: If the access permission of a member differs from the access permission of the group, the member's access permission overrides the group access permission. This lets you give some members of a group different access permissions than the rest of the group members without having to repeat all of the group's access permissions for each group member—you need only specify the access permissions that are different from the permissions of the group to which the member belongs.

Chapter 5: Managing Passwords

This chapter explains how passwords are managed within CA SSO.

This section contains the following topics:

About Passwords

In general application use, passwords are the most popular mechanism for user authentication, but password protection has well-known problems:

- Trivial passwords are easy to guess.
- Passwords that last for years are eventually broken.
- Cyclic passwords are eventually broken.
- Listeners can trap passwords that are sent in clear text over the network.

CA SSO is designed to minimize the overall amount of password management. You can use logon scripts to handle most of the work of changing passwords, and you can force the users themselves to do some of the work.

The most important rule about passwords is that users must not compromise their passwords by either giving them to another person or by using trivial or insecure passwords. The only way to achieve acceptable password security is by educating the users. CA SSO cannot replace education, but it can enforce rules and policies that force users to use strong passwords. Your company must teach end-users that they are ultimately responsible for protecting their own authentication data, whether that is for primary authentication (such as, a smartcard) or for their end applications.

Password Management Tasks

Password management in CA SSO is controlled by the CA SSO administrator. CA SSO administrators perform these password actions using the Policy Manager:

- Set password policies for the whole CA SSO system
- Set password policies for applications
- Set up password synchronization
- Set up automatic password generation
- If necessary, set initial user logon names and passwords, and change and update passwords
- Change user passwords for applications
- Change passwords for primary authentication

Managing User Passwords

This section describes some common tasks for managing passwords.

Specify a User's Primary Authentication Password

Use the Policy Manager to set a user's primary authentication password.

To specify a user's primary authentication password

1. Select the Users icon in the Single Sign-On program bar to display the DataStores window.
2. Expand the Datastore folder, and User Datastores folder to display the list of available User Datastores. Select the appropriate user datastore to display a list of available users.
3. Double-click the user you want and then choose Properties.

   The View or Set User Properties - General dialog appears.
4. Select an authentication method if you have not already done so.
5. If the authentication method you choose supports passwords, such as SSO, the Change Password button is active. Click Change Password and select an authentication method to display the Change Password dialog.
6. Enter a password of your choice and then reenter it to confirm. You can also choose whether the user must change the password at the next logon.
7. Click OK to return to the General dialog.
8. Click OK to finish.

Change the Primary Authentication Password

When the primary authentication method is CA SSO native authentication (SSO authentication method), an administrator can use the Policy Manager to change the user password or users can change their passwords using the CA SSO Client. This is also the case for LDAP authentication; the user can change this by accessing the My Details functionality on the Launchbar or Tools.

Note: The user can only change their LDAP password if the user exists in the Active Directory data store and the LDAP authentication Host has been configured to be "Active Directory aware".

With all other methods of primary authentication, the user password must be changed in the operating or security system.

Changing Application Passwords

An application password is used to access an application. An application password can be changed by any of the following:

Users

The user can use CA SSO Tools, CA SSO Launchbar, or the application to change their application password. We recommend that the user does not use the application to change the application password as CA SSO does it for them.

CA SSO - Automation

If using the CA SSO password auto-generation facility, passwords for the application are generated by CA SSO and rolled over as required. Similarly, if using One-Time-Passwords for the application, when the password is used by CA SSO, a new one is generated and stored on the CA SSO Server, to be used the next time the application is run.

CA SSO - Application Scripts

When writing a script to start up and plug the user's credentials into an application, administrators are encouraged to allow for password change. This involves adding functionality to a script to respond to a request for a new password from the application, and prompt the user to enter a new password. Once the new password has passed the Server's password policy, the application is provided with it. For this reason, administrators are encouraged to make password policies on the Server stronger than that of any application.

Administrators

CA SSO administrators can change or set an application password using the Policy Manager.

The application

The application may prompt for a password change, but this must generally be avoided. This is because CA SSO will not have been notified of this change, and there is a danger that the stored passwords could become out of sync.

The exception to this is if using the Windows Password Synchronization Agent: this listens to password changes from the domain controller/CA SSO Server and notifies the other party.

How the Application Password Is Stored

The logon info section of the user record in the PSTD contains two password fields for each application the user has access to: the current password and the next password.

The current password, CURRPWD, is the current password for the application and is identical to the password already stored in the application or in the password file used by the application. The application’s logon dialog uses CURRPWD to log onto the application the next time the application is selected.

The next password, NEXTPWD, has a value only when a password change is to be carried out at the next logon.

When a correctly written logon script runs, it accesses the values of CURRPWD and NEXTPWD which the CA SSO Client has received from the CA SSO Server. The logon script must log into the application using the current password CURRPWD, and check if there is a value for NEXTPWD field. If there is a NEXTPWD value, the script must change the password in the application from the current password to the next password (the value of NEXTPWD), and notify the CA SSO Server to set the value of CURRPWD to that of NEXTPWD and to null the value of NEXTPWD.

There are two ways for a password change to occur:

- Application-initiated

  In this instance, the application has requested a password change. The Tcl script for the application must recognize this request, and respond appropriately, such as displaying a dialog to gather the user's new password.

  The following is a section of a logon script for an Application-initiated password change:

```
if  #applicaton dialog for password change

    #get new password from user

sso pwdbox -retype y -prompt $prompt

    . . .

    }
```

- SSO-initiated

  When using CA SSO to generate passwords, when the Password Policy on the Server determines that it is time for a password change, the application script returned for that instance includes a value assigned to NEXTPWD. If this is not null, the script must generate a password change request to the application.

  The following sample is a section of a logon script for an CA SSO-initiated application password change:

```
if  {$_NEXTPWD != "" } {

    # do password change

    . . .

    . . .

    }
```

In both cases, after a password change occurs on the application, the script must return the result to the CA SSO Server. This prompts the Server to update the CURRPWD with the value of NEXTPWD. The new password becomes the application password and is used from then on.

The following samples is a section of a script that notifies the CA SSO Server that the password change on the application has occurred:

```
{
…
#send new password to CA SSO Server
sso notify -event pwdchange -status 0 -appname $NAME
. . .
}
```

Password Change Initiated by the User

Users can change a password by using CA SSO Tools, CA SSO Launchbar or by changing the password directly in the application.

By using CA SSO Tools or CA SSO Launchbar, CA SSO sets NEXTPWD to the value of the password entered.

Changing the password directly in the application creates a potential synchronization problem, as the value of CURRPWD stored by CA SSO does not match the password expected by the application.

The user can correct the situation either by using CA SSO Tools or CA SSO Launchbar to update their SSO profile with the new application password (by setting CURRPWD to the value of the new password), or by informing a CA SSO administrator, who then must use the Policy Manager to update the value of CURRPWD.

If the value of CURRPWD is not changed, the next CA SSO logon fails and the user gets a wrong-password message. However, it is often possible to develop a logon script that will recognize the application's wrong-password message, prompt the user to enter a new password, use the new password to log onto the application, and update the value of CURRPWD.

Password Change Initiated by CA SSO

When the current password for an application in CA SSO expires, as determined by policy rules, CA SSO prompts the user for a new password and puts the new password in NEXTPWD.

If the CA SSO password auto-gen feature is enabled, the CA SSO Server generates the new password and updates NEXTPWD. The user does not know of the password change unless the script sends a message that a password change took place.

Password Change Initiated by an Administrator

In a fully implemented CA SSO system, administrators must only initiate application password changes where there is a need to reset values (for example, in cases of user error).

Note: Administrator-initiated password changes must be carried out using the Policy Manager.

To change a user's application password using the Policy Manager:

1. Select the Users icon in the Single Sign-On program bar to display the DataStores window.
2. Expand the Datastore folder, and within this expand the User Datastores folder to display the list of available User Datastores. Select the appropriate user data store to display a list of available users.
3. Double-click the user you want and choose Properties.

   The View or Set User Properties - General dialog displays.
4. Select the Application List icon on the Properties bar on the left (you may have to scroll down) to view the application list window.
5. Click on the appropriate application and press the Update Login Information…. button.
6. Enter the new password, and retype the new password in the Confirm Password field.

   If the same password has already been set directly by the user in the relevant application, deselect Commit password change at next login.

   If the password has not been changed directly by the user in the relevant application, do not deselect Commit password change at next login.

Password Change Initiated by the Application

Preferably a logon script must handle application requests for a new password. Where this is not possible, the user must carry out the password change as described in Password Change Initiated by the User.

To avoid application-initiated password changes, CA SSO administrators must set the password interval in CA SSO to be shorter than that in the application itself.

Resolving Password Error Messages

If you are setting passwords for users on Windows systems, the following message may appear:

```
The password is shorter than required.
```

This error means that the password does not meet the policy requirements. This can be caused by any of the following:

- The password is shorter or longer than the required length.
- The password has been used recently and exists in the Windows NT Change History field.
- The password does not have enough unique characters.
- The password does not meet other password policy requirements (such as those set with CA SSO password policies).

To avoid this error, make sure you set a password that meets all applicable requirements.

Letting Users Update Their Own Credentials

The CA SSO Client can be configured to let users update two kinds of credential information; application credentials, and primary authentication credentials.

Set application credentials

Allows users to enter a user name and password the first time they log onto an application through CA SSO.

Change application credentials

Allows users to change an application's username and password.

The Set Login Information function may be accessed by either:

- Using CA SSO Tools select the My Applications tab. Select the required application and press the Change Password button.
- Using CA SSO Launchbar, right-click on the required application and select Change Password from the context menu.

Application Credentials

Set Credentials

Allows users to enter a user name and password the first time they log onto an application. An administrator can prompt this mode by going into the user's record in the Policy Manager and selecting 'Clear login info' for the appropriate application; this deletes the information stored by the Server, and the Client then prompts the user to enter this information the next time the application is run.

Change Credentials

Allows users to change a password or username for the application, once they have already run the application successfully once previously.

To access the Set Credentials and Change Credentials functions:

In SSO Tools:

Select the application, then click Change Password. This displays the Set Login Information dialog.

In the Launchbar

Right-click on the desired application, and select Change Password from the list that appears. This displays the Set Login Information dialog.

If there is no information for the application in the User's record in the CA SSO user data store, the Set Login Information dialog appears. If there is already information - in other words, the user has entered this information once previously - the Change Credentials dialog appears.

Note: This functionality can be en/disabled by the CA SSO administrator when configuring the CA SSO Client.

Automatically Generated Passwords

CA SSO has a password auto-gen option for automatically generating passwords for user applications. When the option is enabled, CA SSO generates random passwords based on password rules set by administrators.

To use the feature:

1. Enable this option in the appropriate application:
   a. In the Policy Manager, open an application entry.
   b. In the General tab, click Authentication.
   c. In the Authentication dialog, select Password from the Login Type drop-down list.
   d. Select Password Generation Enabled.
   e. Click OK on both dialogs to save the change
2. Create password policies and attach them to the applications. The password policy interval defines how often a new password will be auto generated.

   Note: You must use password policies; however, the auto-gen option will function even if no password polices are attached to the application.
3. Enable this option for users:
   a. In the Policy Manager, open a user entry.
   b. In the User Options tab, select Password General Enabled.
   c. Click OK to save the change.

After enabling the password auto generation feature, no additional changes are required. There is no need to change the logon scripts. CA SSO works as before, the only difference being that when the password expires (because of the password rules attached to the application), CA SSO automatically generates a new password, places it in NEXTPWD, and the logon script changes the password in the application during the next logon. If the user notices anything at all, it will be that there are fewer, if any, dialogs asking for a new password.

Note: Password auto-gen and password synchronization *canno*t be used together.

Grace Logons, Revoke, Forced Change and Password History

CA SSO provides a number of related optional features that can be used with application password authentication and native (SSO) primary authentication:

Grace logons

The administrator can set the number of logons that a user can carry out using an expired password.

Password history

CA SSO can be set to save up to eight previous user passwords and use them as criteria for checking a new password entered by the user. If the proposed password is identical to one of the passwords in the password history, the password is not accepted and the user is prompted for another new password.

Forced change

When an administrator changes a user's password, the administrator can specify that CA SSO will force the user to change that password during the first logon for which it is used.

Revoke

CA SSO can be set to suspend a user after a specified number of logon attempts that failed because of an incorrect password. The suspension can be for a pre-determined period or until the user is reinstated by a CA SSO administrator.

Grace logons and password history are items of password policy and can be linked to an application for all the users that log on to the application. Forced change can be set for a user when his password is changed. Revoke is a global feature which, when set, operates on all users with all password-authenticated applications and CA SSO native primary authentication.

You set grace logons and password history with the Policy Manager in a Password Policies dialog. Forcing password change at the next logon is also set with the Policy Manager.

The revoke feature is set in the Revoke property in the Policy Manager: Resources, Configuration Resources, CA SSO Server Settings, Revoke.

Synchronizing Application Passwords

It is possible to synchronize all of a user's application passwords, including the primary authentication password if CA SSO native authentication is being used. When one password must be changed, whether it is the password for primary authentication or the password used to log on to an application, CA SSO prompts the user for a new password and changes all the other application passwords of the user.

The benefit of password synchronization is that the user can have only one password to remember. However, this arrangement is relatively insecure: as soon as someone acquires a user's password, that person can log on and then access all the applications permitted to the user. To improve security, it is also possible to maintain one password for primary authentication and a different password for accessing applications.

Note: Password synchronization and the automatically generated password feature cannot be used together.

Enable Password Synchronization

To set up password synchronization, you need to enable password synchronization for the user, and then enable password synchronization for each of the applications.

If passwords only need to be synchronized for a subset of the applications accessible to the user, enable synchronization for only those applications.

To enable password synchronization for a user

1. In the Policy Manager, open a user entry.
2. Select the User Options tab.
3. Check the Password Synchronization Enabled checkbox.
4. Click OK to close the dialog.

To enable password synchronization for an application

1. In the Policy Manager, open an application entry.
2. In the General tab, click the Authentication button to open the Authentication dialog.
3. In the Authentication dialog, select Password from the Login Type drop-down list.
4. Check the Password Sync Enabled check box.
5. Click OK to close the dialog.

Password Policy Algorithm for Synchronized Applications

When a user changes a password of a synchronized application, the CA SSO Server consolidates all of the password policies of the synchronized applications into one policy. The CA SSO Server then uses this consolidated password policy as the default policy.

The consolidation is done for all of the synchronized applications that this user has logon records defined for in the CA SSO Server. This consolidation is computed by taking the most secured rule for every rule in a PWPOLICY record.

Password Synchronization and the CA SSO Native Password

When the user changes the password of one synchronized application using the CA SSO Client, all of the passwords of the other synchronized applications are changed to the new password.

There are three ways to define how a CA SSO native password works with password synchronization:

- The CA SSO native password is not synchronized with the synchronized applications.
- The CA SSO native password is partly synchronized with the synchronized applications.
- The CA SSO native password is fully synchronized with the synchronized applications.

  These three ways are described in the sections below.

CA SSO Native Password Not Synchronized

When the user changes the password of one synchronized application, the password is synchronized with all other synchronized applications, but the SSO native password is not changed.

When the user changes the SSO native password, the passwords of the synchronized applications are synchronized with this password.

To make sure that the CA SSO native password is not synchronized:

1. In the Policy Manager, open the __SSO__ application entry.
2. In the General tab, click the Authentication button to open the Authentication dialog.
3. Make sure that the Password Sync Enabled check box is not checked.
4. Click OK to close the dialog.

CA SSO Native Password Partly Synchronized

When the user changes the SSO native password, the password is synchronized with all other synchronized applications.

Changing a password of a synchronized application changes the passwords of the synchronized applications but does not change the SSO native password.

To partly synchronize the CA SSO native password:

1. On the CA SSO Server machine, open the Policy Manager
2. Open the __SSO__ application entry.
3. In the General tab, click the Authentication button to open the Authentication dialog.
4. Check the Password Sync Enabled check box.
5. Click OK on both dialogs to save the change.
6. Select the Resources icon in the single Sign-On program bar.
7. Expand the Configuration Resources folder and select CA SSO Server Settings.
8. Double-click Communication.
9. Double-click Auto_RollOut and set the value to 0.
10. Click OK on both dialogs to save the change.

CA SSO Native Password Fully Synchronized

When the user changes the SSO native password, the password is synchronized with all the other synchronized applications.

Changing a password of a synchronized application changes the passwords of the synchronized applications and also changes the SSO native password. The CA SSO Client notifies the user that their SSO native password has changed.

To fully synchronize the CA SSO native password:

1. On the CA SSO Server computer, open the Policy Manager.
2. Open the SSO application entry.
3. In the General tab, click the Authentication button to open the Authentication dialog.
4. Check the Password Sync Enabled check box.
5. Click OK on both dialogs to save the change.
6. Select the Resources icon in the single Sign-On program bar.
7. Expand the Configuration Resources folder and select CA SSO Server Settings.
8. Double-click Communication.
9. Double-click Auto_RollOut and set the value to 1.
10. Click OK on both dialogs to save the change.

Password Synchronization Agents

CA Single Sign-On supports password synchronization between Microsoft Active Directory and the CA SSO Server. The Windows Password Synchronization Agent (PSA) is responsible for replicating password changes made by Windows users to the CA SSO Server and vice-versa.

PSA is intended for use in the CA SSO environment where some CA SSO applications use Windows credentials as the application login name and password. For example, an application that launches Microsoft Outlook and authenticates on the behalf of the currently logged-in Windows user, or telnet-type application that establishes a connection to ftp server and logs in automatically.

The Windows PSA is bi-directional, and is comprised of two components:

- Active Directory to CA SSO Server synchronization (password filter)
- CA SSO Server to Active Directory synchronization (password exit)

Both components can be installed on the same or different machines depending on whether the CA SSO Server is installed on the Domain Controller (DC). Typically, the Active Directory to CA SSO Server synchronization component is installed on every domain controller (PDC/BDC). The CA SSO Server to Active Directory synchronization component is installed on every CA SSO Server.

The Active Directory to CA SSO Server synchronization components may be installed without the CA SSO Server to Active Directory synchronization components, facilitating a one-way password synchronization from AD to the CA SSO Server, This is applicable in scenarios where users manage their own domain credentials.

The bi-directional password synchronization functionality, which may be added by installing the CA SSO Server to Active Directory synchronization components, allows CA SSO to completely manage users' domain credentials automatically. In this scenario the CA SSO Server auto-generates and synchronizes users' domain passwords and users must log into their workstations using the CA SSO Client's GINA with a primary authentication method other than Windows authentication.

Note: If bi-directional password synchronization is implemented, all password changes will need to adhere to CA SSO Server password rules in addition to Windows rules.

How PSA Works - Password Change Initiated on the Primary Domain Controller

The following describes how password synchronization works when a password change occurs on a domain controller (DC) This can occur when user chooses to change their Windows password, or when administrative user explicitly sets/resets the user's password via Active Directory snap-in on the domain controller.

1. The user initiates a change to their Windows domain password.
2. The Password Synchronization Agent (password filter component) is invoked by the Local Security Authority (LSA) and contacts CA SSO Server, attempting to locate the user corresponding to the Windows user. Once the user is uniquely identified, new password information is sent to the CA SSO Server.
3. CA SSO Servers verifies whether the new password meets the password policy requirements for the configured SYNC_APPL application.. If the checks are passed, the 'NEXT' password for the application is updated with a new value.
4. Password filter notifies LSA that the new password passes SSO password quality checks.
5. The DC completes the password change and updates the Security Accounts Manager.
6. LSA invokes password filter with the confirmation that Windows databases have been updated with the new password.
7. Password filter notifies the CA SSO Server that password change was completed successfully, and that it is okay to propagate the 'NEXT' password value to 'CURRENT' password value for SYNC_APPL for a user involved.
8. The DC sends a positive reply to the user.

   Note: Any errors that might occur during the synchronization process are logged by the password synchronization agent to the configured log file.

How PSA Works - Password Change Initiated on the CA SSO Server

The following describes how password synchronization works when a password change is initiated via 'autogen' functionality (automatic password generation) on the CA SSO Server. This scenario involves having the Password Synchronization Agent component on the CA SSO Server side (password exit) installed and configured. Synchronization applications must be using Windows credentials (login name, password), with the domain application (<DOMAIN NAME>) being set as the master application.

Note: Password synchronization in this direction is only functional for password autogen applications.

1. A user logs in to the CA SSO Server using the CA SSO Client and launches an SSO application that is flagged for PwdAutogen and is included in the SYNC_APPLS list in the CA SSO Server's PSA Plug-in configuration INI file.

   The CA SSO Server retrieves the user's application credentials. The CA SSO Server checks the expiry of the application password. If the password has expired, it generates a new password for that application and stores the new auto-generated password. The user's application credentials are returned to the client.

2. The client runs the application using the current password. After the application has successfully launched, the application script sends a login notify event to the server to indicate a successful login and to invoke synchronization of the new password with the user's domain password.

3. Upon receipt of the successful login event, the CA SSO Server calls the Password Exit, passing along the user's newly generated application password. If the logged in SSO user is from Active Directory-based data store, full user distinguished name (DN) is also passed in. Otherwise, application's current login name (in case of non-AD-type data store) is used, which must correspond to user's Windows login name

4. The Password Exit checks whether the application name is one of the configured synchronization applications. (The configured application names are the only applications whereby any change in those applications' passwords will be synchronized to the Windows Domain Controller). If the application name matches a synchronization application name as configured in the INI file, the Password Exit attempts to change the password on the Domain Controller.

5. If full user DN was provided in step 3, password change attempt uses the DN to identify a user. Otherwise, a LDAP search below a configured base path is performed on Active Directory, using a filter (sAMAccountName=<application login name>). Since sAMAccountName attribute value is unique throughout the domain, the search must uniquely identify a Windows user for the synchronization.

6. The DC reports whether the operation to change the Windows Domain password was successful or not.

7. If the password change on the domain controller was successful, the CA SSO Server moves the new password to the current password for the application.

8. The CA SSO Server returns a SUCCESS or FAILURE return code corresponding to the login notify call.

Password Policies

The most important rule about passwords is that users must not compromise the integrity of their passwords by either giving them to another person or by using trivial or insecure passwords. The only way to achieve acceptable password security is by educating the users. CA SSO cannot replace education, but it can enforce rules and policies that force users to use strong passwords. The company must emphasize that end-users are ultimately responsible for protecting their own authentication data.

Password policies define the qualities of an acceptable password. CA SSO password policies can determine the minimum number of alphabetic, numeric, or special characters in a password, the maximum period of time a password may be valid, and so forth.

When an application is linked to a password policy, any new passwords must meet the criteria of the policy. When a user or administrator enters a new password, CA SSO checks the new password against the rules in the password policy. If it does not conform to the password policy, it is rejected. Passwords automatically generated by CA SSO will conform to the relevant password policy.

Note: When setting up a password policy, be aware that the SSO policy effectively replaces any password policy the application may have. For this reason, make sure that the password policy you create in SSO is more stringent than that of the application. Otherwise, it will be possible to have a situation where a new password passes SSO's password policy, but is rejected by the application's own policy.

Only one password policy can be linked to an application, but many applications can be linked to the same password policy.

This section describes possible password rules, how to add password policies, and how to link applications and password policies.

Rules for Password and Lockout Policies

Windows has a set of password rules and policies that force users to use passwords that avoid most of these common pitfalls. CA SSO has additional rules that ensure that users select even more secure passwords.

Password Rules You Can Change

The following password rules apply to both password-based primary authentication and application passwords:

- The maximum and minimum number of characters a password must contain.
- The number of characters that must be alphanumeric, alphabetic, uppercase, lowercase, and numeric.
- The minimum number of special characters a password must contain. A special character is one that, like $ or #, is not a letter or a number.
- The maximum number of times the same character can repeat successively in the new password.
- The maximum number of days each password can be used.
- Forbidding the use of a previously used password and defining the number of previous passwords to retain as criteria for this rule.
- The maximum number of grace logons (the number of times a password can be used after it has expired).

  Note: Password policy rules can set password length to a minimum of one character and a maximum of 255 characters.

Create a New Password Policy

To create a password policy

1. Select the Resources icon in the Single Sign-On program bar.

   The Resources window appears.
2. Expand the Configuration Resources folder.
3. Right-click Password Policy and select New.

   The Create New PWPOLICY Resource - General dialog displays.
4. Enter the necessary information on the General dialog.

   For an explanation of all fields, refer to Defining Password Policy Properties section.
5. Select Policy Extended Attributes to define the additional password policy rules such as Password Length restrictions and History requirements.
6. Click OK to finish.

Remove a Password Policy

To remove a password policy

1. Select the Resources icon in the Single Sign-On program bar to display the Resources window.
2. Expand the Configuration Resources folder.
3. Select the Password Policy folder to display the list of available password policies.
4. Right-click the password policy you want to remove and choose Delete. The Delete the Selected PWPOLICY(s) dialog appears.
5. A check mark appears in the environment in which the password policy was created. Click OK to remove the password policy.

Link Policies and Applications

To link an application to a password policy

1. In the Policy Manager, open the Properties of an application resource.
2. In the General tab, click the Authentication button.
3. In the Authentication dialog, select a password policy from the Password Policy list.
4. Click OK on both dialogs to save the change.

Lifetime of Passwords

You can set the maximum password interval, in days, for passwords. The password expires when the specified number of days has passed. The user must then change the password.

Set a Password Interval

The password interval can be set in several different locations. For a specific application, you can set the password interval in the password policy to which it is linked. More than one application can be linked to the same password policy. A global password interval value can be set for the entire system using the default password policy. However, this value can be overridden for an application if it is linked to a specific password policy.

If an application has an integral password policy, you must set the password interval value in CA SSO to a shorter time than the integral policy's. In this way, if logon is performed through CA SSO, the user is prompted by CA SSO for a new password before the application itself requires one.

Each time the CA SSO Server is requested to provide logon information for an application, it checks whether the password for the application has expired.

To set the maximum password interval for a password policy

1. In the Policy Manager, open a password policy entry.
2. Select the Policy Extended Attributes option.
3. Enter a number in the Password Interval box. This is the number of days that a password remains valid. When this time expires the user must create a new password.

User Password Never Expires

You can configure the data store so that a password of a specific user never expires. This applies to both an application password and the SSO password.

To configure a user password to never expire:

1. In the Policy Manager, open a user's entry.
2. Select the User Options tab.
3. Check the Password Never Expires checkbox.
4. Click OK to close the dialog.

Configure Data on the CA SSO Server

The Password Synchronization Agent for Windows communicates with the CA SSO Server in order to implement password policy and password synchronization.

You must define the following in the CA SSO Server:

- AdminUser
- Synchronization application(s)
- Specific user declarations
- Applications settings
- Password policy

Define the Synchronization Application(s)

The synchronization applications and the relationships between them differ depending on whether you configure uni-directional or bi-directional password synchronization between Active Directory and CA SSO.

Uni-directional Synchronization Application Configuration

Uni-directional synchronization requires the Windows PSA to be installed on the Active Directory domain controllers without the PSA Plug-ins being installed on the CA SSO Servers.

In this configuration, when a user's domain password is changed, the password of the SyncAppl application, which is identified in the PSA Filter's registry settings, is synchronized.

It is recommended that this SyncAppl be the domain application, that is, have the same name as the NetBIOS domain name such as 'CORPDOMAIN', for use with the CA SSO Client's GINA.

If you have other applications which also use domain credentials, you can synchronize these applications with the SyncAppl, and automate the maintenance of the credentials of these applications also using the CA SSO Server's 'Password Sync' functionality.

To define synchronized applications in the Policy Manager

1. Go to the Application Properties
2. Press the Authentication button
3. Select the 'Password Sync Enabled' check box.
4. Press OK twice to close the Authentication and Application Properties dialogs.

Bi-directional Synchronization Application Configuration

Bi-directional synchronization requires the Windows PSA to be installed on the Active Directory domain controllers and the PSA Plug-ins to be installed on the CA SSO Servers.

In this configuration, the user does not manage their domain password, but the CA SSO Server auto generates and synchronizes this according to the appropriate password policy defined on the CA SSO Server. The SSO GINA must be installed on the end user workstations to facilitate primary authentication using an authentication method other than Windows (as the users do not know their auto-generated domain passwords).

The domain and other applications which use domain credentials are advised to be linked according to the following configuration which is dictated by the SyncAppl and SyncAppls configuration parameters of the PSA Filter and Exit.

PSA Filter SyncAppl

This is recommended for the domain application. This application must have the "Password Generation Enabled" checkbox selected. When the PSA Filter synchronizes a password change, it will update the password for this application.

PSA Exit SyncAppls

This list must be composed of all other applications which use the domain credentials, and must link to the credentials of the previously described "PSA Filter SyncAppl". This is facilitated by setting the "PSA Filter SyncAppl" as the Master application for each of the "PSA Exit SyncAppls" child applications.

Set Specific User Declarations

For uni-directional password sync configurations that use the "Password Synchronization Enabled" functionality of the SSO Server, you must set up specific user declarations for each user in the domain.

To set specific user declarations in the Policy Manager

1. Go to User Properties.
2. Select User Options.
3. Select the Password Synchronization Enabled checkbox.

More information:

Managing Users and User Groups (see page 155)

Chapter 6: Managing Users and User Groups

This chapter explains how users and user groups are managed within CA SSO.

This section contains the following topics:

CA Directory as the User Data Store

Most large companies use Active Directory (ADS) to store their user information. However, you can use CA Directory as your user data store if you do not already have ADS or another LDAP-compliant data store setup within your company to store your user information.

If you use Active Directory, or another external use data store, you must make all changes to user information in that system external to CA SSO.

About Users and User Groups in CA Directory

If you do not plan to use an existing third-party LDAP data store, such as Active Directory, you can define and maintain all of the users and user groups stored in CA Directory from the Policy Manager. By default the CA Directory user data store is called ps-ldap.

From the Policy Manager you can create, remove, locate, or change the properties of a user or user group.

After defining the users and groups in the user data store, you must specify which resources and applications each group or user is allowed to access and under what conditions. You can do this on a user-by-user basis, but it is more efficient to define rules for groups and add users to the relevant group or groups.

The user data store must include all the users who will use CA SSO.

Note: Administrative users must be created in the Administrative data store. These users will not have SSO functionality such as an application list. If this is something you would like to give these users, create a duplicate user of the same name in the CA SSO user data store.

To simplify the task of assigning and removing authorizations, you must create groups of users who share the same functions or responsibilities. It is much easier to create a group, add users to the group, and then enable the group to use a particular host for initial authorization or to invoke an application than it is to assign the same permissions separately to each user. You add users and groups to the data store using the Policy Manager.

User Group Creation and Definition in CA Directory

The following sections explain how to create and maintain groups of users in CA Directory. A group of users usually represents people who work together on specific projects or belong to a specific department or to the same division in the organization. How you group users depends on how your company is organized and how your users work.

Companies already sort their staff into teams, projects, departments, or other types of groups, and you can create CA SSO user groups that correspond to these groups. Using this method, you can easily remove users from the groups to which they belong when they leave the organization or when a change is required.

To simplify the task of assigning and removing authorizations, you must create groups of users who share the same functions or responsibilities. It is much easier to create a group, add users to the group, and then enable the group to use a particular host for initial authorization or to invoke an application than it is to assign the same permissions separately to each user.

If you are using Active Directory as your user data store, there is no need to create user groups. You probably already have users grouped according to department or functional area; these existing groups are picked up by SSO so you can apply rules or permissions to them.

Note: The SSO Server considers user groups to act just like users.

More information:

Assign Access Permissions Using Groups (see page 129)

Create a User Group in CA Directory

This procedure explains how to create a user group in CA Directory. You might want to create a user group to help manage your users more efficiently. For example, it is much easier to assign an application to a single group of users, rather than assign an application to 30 individual users.

To create a user group in the data store using the Policy Manager

1. Select the Users icon from the left pane.
2. Navigate to the ps-ldap User Datastore.

   A list of any existing users in that data store appears.
3. Right-click in the right pane and select New, User Group.

   The Create New Group - General window appears.
4. Enter the group name in the Group Name field.

   You can also enter a longer name and a comment about this group in this dialog.
5. Click the Members icon on the left.

   The Create New Group – Members dialog appears.
6. Use the Add and Delete and Filter buttons to help you add users to the group.

   Note: The filter can help you organize and filter usernames. You can use an asterisk as a "wild card" character, for example, if you type "G*" you will see all users whose username starts with a "G".
7. Once you have added all the users you want to the group, select OK.

   The new User Group is saved and now appears in the list of users with a different icon from individual users.

Delete a User Group in CA Directory

This procedure explains how to delete a user group. You might need to do this if a department has been merged with another department within your company.

To delete a user group from CA Directory using the Policy Manager

1. Select the Users icon from the left pane.
2. Navigate to the User Datastore.

   A list of all users and user groups in that data store appears.
3. Right-click on the user group you want to delete and select Delete.

   The Delete the selected item(s) window appears.
4. Mark the check box for the user group and select OK.

Create a User in CA Directory

To create a new user

1. Click the Users icon.

   A list of use data stores appears. The CA Directory data store is listed as “ps-ldap” unless you are using your ADS user group in which case you must manage users in ADS.

2. Select the ps-ldap data store in which you want to create the new user.

   A list of any existing users in that data store appears.

3. From the Edit menu, select New, User. You can also right-click on the user and use the pop-up menu.

   The Create New User - General dialog appears.

4. Enter the user details in the New User dialog. Use the icons in the left pane to open the other variables you can define for the user.

5. Select OK.

On the Create New User dialog, the bar on the left shows the associated windows:

General

Defines general information about the user, including the user’s name, where the user is located, and the organization the user belongs to.

User Options

Specifies password features for this user.

User Attributes

Specifies defined attributes and their values.

Groups

Lists the groups this user belongs to.

Restrictions

Indicates whether the user’s account expires, whether the account is suspended, and what days and times the user can access the system.

For a complete explanation of all of the dialogs and the fields they contain, see the *Policy Manager Help*.

Delete a User in CA Directory

Follow this procedure to delete a user from the ps-ldap user data store using the Policy Manager.

To delete a user

1. Click the Users icon.

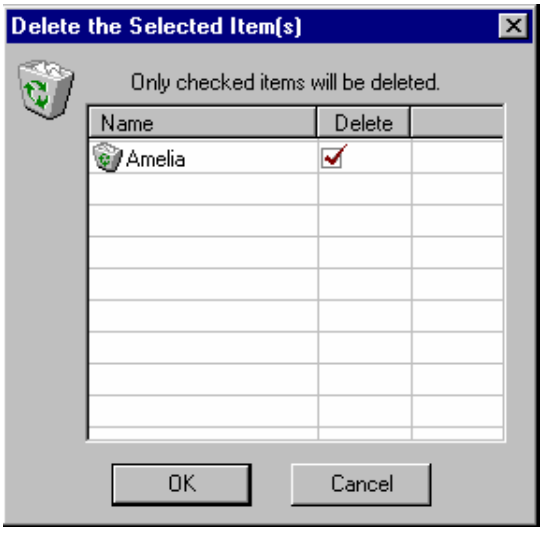
   A list of data stores appears.

2. Select the ps-ldap data store from which you want to delete the user.

   A list of any existing users in the data store appears.

3. Right-click on the user you want to remove, and select Delete from the pop-up menu.

   The Delete the Selected Items dialog appears.



4. Be sure a checkmark appears in the check box in the Delete column beside the user, and then click OK to delete the user.

User Data Store Administration

The topics that follow describe concepts about maintaining the user data store.

Populating the User Data Store

For CA SSO to function properly, the user data store must contain the required information on users and user groups.

Note: By default the CA SSO Server only recognizes English characters. If you are entering users with non-English characters you must set the CA SSO Server to recognize those characters.

Once the user data store contains the user and group definitions, CA SSO can begin to provide single sign-on functionality to the users.

You can populate a user data store with the required information using the following management tools:

CA Directory

- The Policy Manager
- JXplorer

Microsoft Active Directory

Microsoft Management Console

Other Third-Party Directories

- The Policy Manager
- Native management tools provided with the directory.

More information:

General (see page 371)

The LDAP Query Limit

If you are working on an LDAP user data store with a large number of users or group objects, some of the objects may not appear in the list when you open the Users view. This is due to the LDAP query limit.

To change LDAP query limit

1. Go to the CA Directory configuration file:

   On UNIX:

   `$DXHOME/config/limits/default.dxc`

   On Windows:

   `%DXHOME%\config\limits\default.dxc`

2. Change the number of the following attribute to the number of users that you want to display when you do an LDAP query:

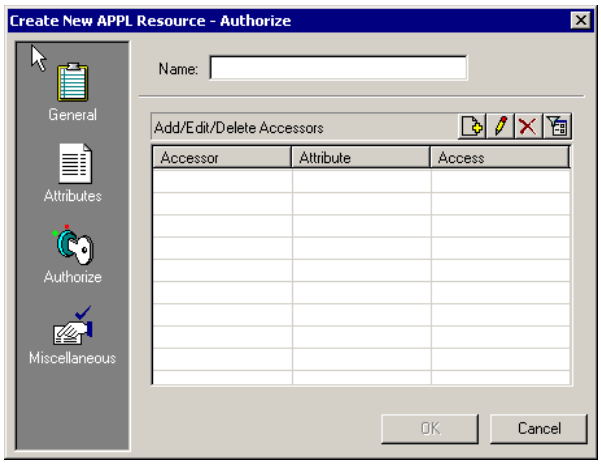
   ```
   set max-op-size
   ```

Authorizing Users and Groups to Hosts and Applications

Users and groups must be authorized to log in to specific authentication hosts or groups of hosts. If users are to access applications through CA SSO, they must also be authorized to the applications and groups of applications.

Note: CA SSO Server does not recursively search nested Active Directory groups to compile authorization data. We recommend that you put the user objects in parent groups that are authorized for any SSO resource.

Use the Create New APPL Resource - Authorize dialog to specify user and group access permissions to the selected resource, or use the View or Set APPL Properties - Authorize dialog to view or modify access permissions to the resource.



If you give an application default access “Execute”, all users can use (execute) that application.

User Data Store Class Information

This section explains the user data class information within CA SSO.

About User Stores

CA SSO supports LDAP-based directories to store user information. A *user data store* contains definitions of users, user groups, and logon data (user IDs and passwords). You can define and use more than one type of user data store in your CA SSO system.

Data Classes in CA SSO

CA SSO uses three user data classes:

USER

Stores information about a user, such as the user's full name, the times the user is allowed to log on, the authentication methods that the user is allowed to use, and the groups to which the user belongs.

GROUP

Stores information about groups of users, including the list of users who are members of the group.

LOGININFO

Stores the information needed for logging the user in to a specific application, including user credentials for the application (such as the logon name, password, and other details), and statistical information (such as last logon, first logon, logon count, and last password change)

The CA SSO user data store can reside on the local host (where the CA SSO Server is installed) or it can reside on a remote host.

User IDs and Logon Names

The *user ID* in the data store must match the primary authentication username.

The user ID is entered as the entry name when a new user is defined. For example, if primary authentication is set to Windows, CA SSO must use the same user ID as that used by Windows or an alias name that is defined in the user entry.

The *logon name* is the name that is used to log onto an application. It can be different from the user ID. A user can have different logon names for different applications.

Forbidden Characters in Property Values

For all classes, property values can contain most of the ASCII characters, including blanks. The following characters **cannot** be used:

- **[ ]** Left and right square brackets
- **** Backslash
- **:** Colon
- **?** Question mark
- **|** Pipe
- **"** Double quote marks
- ***** Asterisk
- **,** Comma

LDAP-enabled Directories

CA SSO supports a variety of LDAP-based directories as user data stores without any additional programming effort on your part. Supported LDAP-based directories include:

- Microsoft Active Directory
- CA Directory
- CA Top Secret (TSS)
- CA ACF2 (ACF2)
- RACF

If you choose to use an LDAP user data store, you can map the user information, group information, and logon information properties to existing fields in your directory. This allows the CA SSO Server to work with any LDAP-enabled database.

To do this, you can either create a new user directory class with the mapping that you want, or you can edit the existing mapping by modifying the user attribute class properties for this directory.

The following table lists the default class mapping for each of the supported user data stores:

CA Directory	Active Directory	TSS, ACF2, RACF
eTssoUser	User	-
eTssoGroup	Group	-

CA Directory	Active Directory	TSS, ACF2, RACF
eTssoLoginInfo	eTssoLoginInfo Only available if the schema has been extended	-
eTssoLoginInfos organization organizationalUnit country	container organizationalUnit	* The ***** causes the CA SSO Server to find the container class by the **nameby** attribute rather then the class name.

Directory Schema

A *schema* is the definition of classes that are used in a directory. You can change the schemas that are supplied with CA SSO to include more fields.

For example, if you need to store the names of each employee's children in their user account, you can extend the schema by adding a field to the User class. CA SSO does not maintain this information.

If you are defining an LDAP-enabled data store, you can specify how users and groups are defined in your data store schema by completing the Advanced Data Store Properties dialog. This dialog lets you define the object classes that represent each class used by the CA SSO Server (users, groups, logon information, and a container for logon information).

These four classes can be mapped to an existing class name or a new one, depending on your implementation. You can modify and extend your directory schema to get new classes.

The CA SSO Server installation creates an CA Directory instance with an extended schema. It also provides a utility to modify an Active Directory schema if needed.

The Nameby Attribute

Some attributes are used to name an entry, forming its relative distinguished name (RDN). Each entry must have at least one naming attribute. Although attributes can have more than one attribute value, only one of these can be chosen as the naming attribute. For example, the CommonName attribute may have two values, **Fred** and **Freddie**, but only one of these values can be the naming attribute.

In each of the four user data store classes, the NameBy attribute lists the naming attribute for the class. This allows the data store to search for an object by its name.

For example, if CommonName is used as the group name, the NameBy property is **cn**.

The NameBy attribute is only used for user data that is stored in an LDAP-enabled directory.

Microsoft Active Directory

The Microsoft Active Directory user data store is an LDAP-based directory for holding user information. You can connect to this data store after you install the CA SSO Server. The Active Directory user data store can reside on the computer where the CA SSO Server is installed or on a remote host.

You cannot use the Policy Manager to add or delete users from an Active Directory user data store. Use the Microsoft Management Console to add and delete users.

CA Directory

The CA Directory user data store is an LDAP-enabled directory for holding user information. You install this data store when you install the CA SSO Server. The CA Directory user data store can reside on the localhost (where the CA SSO Server is installed) or on a remote host.

CA Directory's schema is stored in a file. The default CA Directory user data store schema created by the CA SSO Server installation is in the file PolSrv.dxc located in the **%DXHOME%\config\schema** directory in your path.

OS/390 LDAP Directories

CA SSO provides strong and direct authentication from your applications to user repositories on OS/390. By using a direct access methodology, you can directly issue LDAP authentication requests to your CA ACF2, CA Top Secret, and RACF user repositories from the intercepts provided in CA SSO.

Note: To reduce the time spent waiting to view the list of user entries, we strongly recommended that you use filters when displaying users from an CA ACF2, CA Top Secret, or RACF user data store.

Queries in CA ACF2, CA Top Secret, and RACF Data Stores

CA Top Secret, CA ACF2 and RACF do not support LDAP queries by class name. This means that CA SSO does not map those fields - they are left blank.

Instead, queries in these directories are done by the nameby attribute, because the nameby is different for USER, GROUP and CONTAINER.

Since in the Containers Classes field you are required to specify all the classes names that must be treated as containers, in order to get the server to find containers by nameby rather than class name you must use * in the container class names (instead of leaving it blank as in the other class mappings) and set the container objClassName to the nameby string.

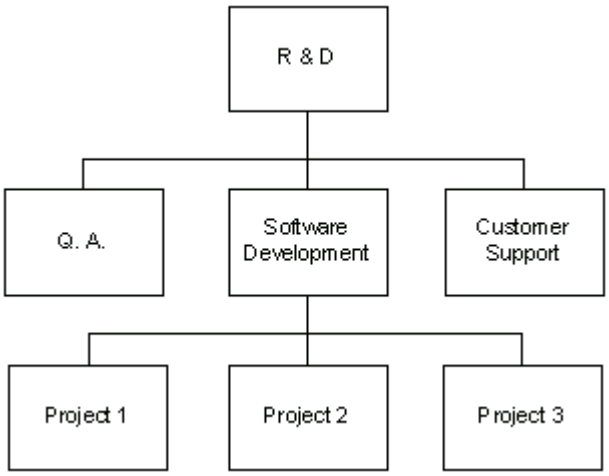
If you leave the container class name blank, the CA SSO Server disregards all containers in the data store.

The CA ACF2, CA Top Secret, and RACF data stores do not support the Logininfo class.

Parentage

The concept of subordinate and superior groups is known as parentage. One group can be the parent—superior—of one or more groups. A *child*, or subordinate group, can have only one parent. Assigning a parent to a group is optional.

Consider the following diagram:



Research & Development is the parent of the three groups: Q. A., Software Development, and Customer Support. Software Development is also the parent of three groups—Project 1, Project 2, and Project 3. Project 2 has only one parent—Software Development. Research & Development has no parent.

Q. A., Software Development, and Customer Support are groups in their own right and are also subgroups of R&D.

The User Class (USER)

Mapping

In an LDAP user data store, the eTssoUser class is a subclass of the inetOrgPerson class.

The following table lists the USER class property names used by the different kinds of user data store. These are mapped to the property names displayed in the Attribute Mapping dialog in the Policy Manager.

Display Name	LDAP (CA Directory)	LDAP Active Directory	CA Top Secret	CA ACF2	RACF
AuthenticationMethod	eTssoAuthnMethod	eTssoAuthnMethod	-	-	-
Comment	description	description	User-Type	-	-
CommonName	cn	sAMAccountName	tssacid	acf2lid	racfid
DayRestrictions	eTssoDayRestriction	eTssoDayRestriction	-	-	-
ExpireAt	eTssoExpireAt	eTssoExpireAt	-	-	-
FullName	displayName	displayName	Name	Name	-
IsDisabled	eTssoIsDisabled	eTssoIsDisabled	-	-	-
Location	l (lower-case L)	l (lower-case L)	-	-	-
MemberOf	-	memberOf	memberOf	memberOf	racfConnectGroupName
Organization	o	o	Zone	-	-
OrganizationalUnit	ou	ou	Department	-	-
Password	userPassword	-	-	-	-
PasswordAutoGeneration	eTssoIsPwdAutoGen	eTssoIsPwdAutoGen	-	-	-
PasswordInterval	eTssoPwdInterval	eTssoPwdInterval	-	-	-
PasswordSynchronization	eTssoIsPwdSync	eTssoIsPwdSync	-	-	-

Display Name	LDAP (CA Directory)	LDAP Active Directory	CA Top Secret	CA ACF2	RACF
Phone	telephoneNumber	telephoneNumber	-	-	-
ResumeAt	eTssoResumeAt	eTssoResumeAt	-	-	-
RevokeCount	eTssoRevokeCount	eTssoRevokeCount	-	-	-
Surname	sn	sn	tssacid	acf2lid	racfid
SuspendAt	eTssoSuspendAt	eTssoSuspendAt	-	-	-
TimeRestrictions	eTssoTimeRestriction	eTssoTimeRestrictions	-	-	-

Properties

Property	Description	Data Type
User	The name of the user. This is the key property. It defines if the CA SSO Server uses the user name for authorization. In LDAP-enabled directories, there is no field that holds this value. Instead, this value is the distinguished name of the user.	String
AuthenticationMethod	The 32 authentication methods the user is allowed to use. The meaning of each value is taken from the AuthMap section in the CA SSO Server configuration. You can view these settings in the Policy Manager: Resources, Configuration Resources, Authentication Methods.	CSV
Comment	A remark, which is not usually used during authorization. It can be used for authorization if a user_attr is mapped to it, and the AZN flag is checked.	String
CommonName	The common name of the user. This can be a first name, username, nickname or full name representation.	String

Property	Description	Data Type
DayRestrictions	The day restrictions on user logon access. This represents the days of the week.	A bitmask of allowed days: ■ ETWAC_SUN 0x80 ■ ETWAC_MON 0x40 ■ ETWAC_TUE 0x20 ■ ETWAC_WED 0x10 ■ ETWAC_THU 0x08 ■ ETWAC_FRI 0x04 ■ ETWAC_SAT 0x02
Email	The email address of the user	String
ExpireAt	The date the user entry expires and becomes invalid.	**Date—**time_t (sec since 1970)
FullName	The full name of the user. This string is not used during authentication.	String
IsDisabled	Indicates if the user is enabled or disabled.	Boolean: ■ **0**—FALSE ■ **1**—TRUE
Location	The location of the user	String
MemberOf	A list of the groups that the user belongs to.	Multi-value property
Organization	The organization of the user	String
OrganizationalUnit	The organizational unit of the user	String
PasswordAutoGeneration	Indicates whether passwords are to be generated automatically by the CA SSO Server. This applies to the user's SSO password as well as passwords for applications that are managed by the CA SSO Server.	Boolean: ■ **0**—Passwords are not generated automatically ■ **1**—Passwords are generated automatically
PasswordInterval	Indicates whether the user's password for an application expires or never expires.	Unsigned short ■ **Blank—**The user's password can never expire (default) ■ **0—**The password never expires ■ **Integer**—The number of days that the password will remain valid after the day is it set

Property	Description	Data Type
PasswordSynchronization	Indicates whether the user's password can be automatically kept identical for all of the user's applications that also have the PwdSync flag set.	Boolean
Phone	The user's telephone number	String
ResumeAt	The date on which the user object becomes valid after it was suspended by the SuspendAt property.	■ **Blank**—No resumption (default) ■ **Date**—time_t (sec since 1970)
RevokeCount	The maximum number of repeated unsuccessful logons to an application a user can have until their logon privileges are revoked. This value is defined in the def_fail_count setting in the Policy Manager: Resources, Configuration Resources, Policy Server Settings, Revoke.	Unsigned short ■ **Blank**—(default) Unsuccessful logons will not cause the user to be suspended or last logon was successful. ■ **Integer**—The number of failed logon attempts before the user is suspended. At every failed logon attempt, this number decreases by 1 until it equals 0, at which point the user is suspended.
Surname	The user's second name. This is used for directory data only.	String
SuspendAt	The date on which the user object is suspended. Use the ResumeAt property to set the date on which the user object becomes valid again.	■ **Blank**—User is not suspended ■ **Date**—time_t (sec since 1970)
TimeRestrictions	The times between which the user is allowed to log on (for example, 8:00 to 18:30).	■ **Blank**—No restriction on logon times (default) ■ **Date**—time_t (sec since 1970)

The Group Class (GROUP)

The group entry contains information about a user group. The most important information in the group entry is the list of users who are members of the group. Each group of users is represented by a entry in the GROUP class.

The CA SSO Server has three different ways of storing information about users and groups:

- The **memberOf** field of the USER class can be used to store the groups that the user belongs to.

  To work in this mode, make sure that the **memberOf** field of the USER class includes a list of groups, and the Member field of the GROUP class is empty.

- The **Member** field of the GROUP class can be used to store the distinguished names (DNs) of the users that are members of this group as multi-valued in this field

  To work in this mode, make sure that the member field of the GROUP class includes a list of users, and the **memberOf** field of the USER class is empty.

- If the **memberOf** and **uniqueMember** fields are both mapped, the CA SSO Server assumes that the directory cross-references information on the user and group objects.

  The CA SSO Server reads the list of user groups from the user object and the list of group members from the group. This avoids extensive searching and increases performance.

  Note: When updating in this mode the CA SSO Server updates the group object only.

Mapping

The following table lists the GROUP class property names used by the different kinds of user data store. These are mapped to the property names displayed in the Attribute Mapping dialog in the Policy Manager.

Display Name	LDAP (CA Directory)	LDAP (Active Directory)	CA Top Secret	CA ACF2	RACF
Comment	eTssoComment	info	User-Type	-	-
CommonName	cn	sAMAccountName	tssprofile	acf2lidgrp	racfid
FullName	eTssoDisplayName	Description	Name	-	-
Member	uniqueMember	member	uniqueMember	uniqueMember	racfGroupUserIds

Properties

Property	Description	Data Type
Group	The name of the group. This is the key property.	String
Comment	A remark; not used during authorization.	String (256 character limit)

Property	Description	Data Type
CommonName	The common name of the group. This is whatever the group is commonly known as. This is used for directory data only.	String (47 character limit)
FullName	The full name of the group. This string is not used during authentication.	String (47 character limit) No default value
Member	The list of users that belong to this group.	

The Logon Information Class (LOGINFO)

The logon information is the information needed for logging the user into a specific application. The CA SSO Client sends application logon information to the application host, after the CA SSO Server checks the user's request to log on to an application and returns the application script and login information.

There is a separate logon information entry for each application. This set of properties can have different values for each application the user is allowed to access.

Each logon information section contains:

User credentials for the application

Including the logon name, password, and other details

Statistical information

Such as last logon, first logon, logon count, and last password change

This set repeats itself to accommodate different values for every application for which the user is authorized. For an application that is not a master application only the statistical information is relevant; the user credentials in the logon information are irrelevant since they are taken from the master application.

The set of information in the logon information property is listed next. This set repeats itself to accommodate different values for every application for which the user is authorized. The LoginInfo class is a subclass of the top class.

When creating a resource in this class, you must create it under the class container eTssoLoginInfos. Start the resource name with the prefix cn=.

Mapping

The following table lists the LOGININFO class property names used by the different kinds of user data store. These are mapped to the property names displayed in the Attribute Mapping dialog in the Policy Manager.

Display Name	LDAP (CA Directory)	LDAP (Active Directory)	CA Top Secret	CA ACF2	RACF
ApplicationName	eTssoApplName	eTssoApplName	-	-	-
CommonName	cn	cn	-	-	-
CurrentPassword	eTssoCurrPwd	eTssoCurrPwd	-	-	-
failedLoginTime	eTssoFailedLogin	-	-	-	-
FirstLoginTime	eTssoFirstLogin	eTssoFirstLogin	-	-	-
GraceCount	eTssoGraceCount	eTssoGraceCount	-	-	-
LastLoginTime	eTssoLastLogin	eTssoLastLogin	-	-	-
LoginCount	eTssoLoginCount	eTssoLoginCount	-	-	-
LoginID	eTssoLoginID	eTssoLoginID	-	-	-
NextPassword	eTssoNextPwd	eTssoNextPwd	-	-	-
oldPasswords	eTssoOldPasswords	-	-	-	-
PasswordChangedTime	eTssoPwdChangedAt	eTssoPwdChangedAt	-	-	-
pwdHistCount	eTssoPwdHistCount	-	-	-	-
UserDN	eTssoUserDN	eTssoUserDN	-	-	-

Properties

Property	Description	Data Type
ApplicationName	The name of the application the logon information describes. This is the key value.	String
commonName	The name of the LoginInfo object. The value is usually in the format *applicationName*.	String
CurrentPassword	The current password of the user.	String
FirstLoginTime	Date and time the user first logged in to the application through CA SSO.	**Date—**time_t (sec since 1970)

Property	Description	Data Type
GraceCount	The remaining number of attempts that the user has to log on to the application until the expired password must be changed. The number is taken from the password policy at the time a password expires and decremented untill it reaches 0, at which time the password is no longer valid.	Integer
failedLoginTime	Date and time that the user last failed to log on to the application.	**Date**—time_t (sec since 1970)
LastLoginTime	Date and time that the user last logged in to the application through CA SSO.	**Date**—time_t (sec since 1970)
LoginCount	The number of times the user has logged in to the application through CA SSO.	Integer
LoginID	The logon ID or user name for the target application.	String
NextPassword	The value of the next password for the application. This field only has a value when the user requested a password change but the password has not yet been changed in the application.	String
OldPasswords	Previous passwords for the application. The passwords are encrypted and stored as multi-value. A maximum of eight old passwords can be stored.	Multi-value list
PasswordChangedTime	Date and time the application password was changed through CA SSO.	**Date**—time_t (sec since 1970)
UserDN	The distinguished name of the user that this logon information relates to.	DN

Chapter 7: Managing User Sessions

This section contains the following topics:

CA SSO Sessions

A CA SSO session is the period of time that a user is logged on to CA SSO. During a CA SSO session, the user may be logged onto other CA SSO-enabled applications.

By default, CA SSO lets users have multiple concurrent sessions on different machines. Using CA SSO, you can set up automatic session management rules to limit the number of concurrent sessions a user has open and the behavior of those sessions. You can also install the SSO Session Administrator which lets you manually view and close user sessions.

Note: SSO Session Management does not work when the Client is operating in Offline mode. If you want to use Session Management then you cannot use the Offline Operation functionality, and vice versa.

Benefits of Session Management

Using the Policy Manager and Session Administrator to control user sessions, you can save system resources and improve system security.

To discourage users sharing logon IDs and to save system resources, the Policy Manager and Session Administrator lets you:

- Set the maximum number of sessions a user can have open at the same time
- Define what happens when a user attempts to exceed this number of sessions
- Manually terminate sessions

To protect sensitive information, you can:

- Set an idle time-out for locking the Client workstation
- Set an idle time-out for logging the user off the CA SSO session as well as logging out the underlying Windows user

Session Management Options

CA SSO enables you to manage user sessions in the following ways. You can:

- Set up session management rules with the CA SSO Server
- Set up the Session Administrator and manually view and close user sessions

Manage User Sessions with the CA SSO Server

Using the Policy Manager, you can set up session profiles that define how the CA SSO Server works with user sessions. Session profiles are groups of settings applied to users or groups of users.

Session profiles include the following settings:

- The number of sessions a user can have open at once
- The result when the user reaches their maximum number of sessions:
  - Terminate the oldest session
  - Terminate the newest session
  - Terminate all sessions
  - Ask the user which of their sessions they want to terminate
  - Reject the registration of the new session – the user is denied log-on
- The result when the system is not used for a time:
  - Define a screen-lock timeout
  - Define a logoff timeout for CA SSO. This timeout must be greater than the screen-lock timeout otherwise it is not considered. CA SSO is logged out after (logoff timeout - screen-lock timeout) time.

Manage User Sessions with the Session Administrator

In addition to storing automatic session profiles on the CA SSO Server, you can also manually track and terminate sessions using the SSO Session Administrator. The Session Administrator is a web-based tool that lets you:

- View and terminate users' sessions
- Check how long a session runs
- Check what machines a session is running on

User Session Termination

Direct Notification and Heartbeat response are two notification methods that CA SSO Server can use to notify an CA SSO Client to terminate user's session. Alternatively, user termination can happen automatically, or you can initiate it manually from Session Administration tool. The methods described below are set up using the Policy Manager and occur automatically.

Direct Notification

This is the default method. This is the faster method to notify the CA SSO Client that its session has been terminated, but it only works if there is no network address translation, such as a firewall or IP chaining, between the CA SSO Client and the CA SSO Server.

Heartbeat Response

This method relies on the CA SSO Client sending a heartbeat to the CA SSO Server and receiving a heartbeat response from the server. You must use the Heartbeat Response method if your system uses network address translation between the CA SSO Client and the CA SSO Server.

Method 1: Direct Notification (Default)

This is the faster method to notify the CA SSO Client that its session has been terminated.

When the user attempts to start a session, the CA SSO Server uses the session management policy and information about current sessions to determine whether to permit the new session.

If the new session is not permitted, the CA SSO Server sends a notification to close the relevant CA SSO Client.

For the Direct Notification method to work, the CA SSO system must have the following:

- The CA SSO Client must be listening for notification messages from the CA SSO Server on a particular port.
- The network must permit the CA SSO Server to send a message.

  The Direct Notification method may not be suitable for systems that contain internal firewalls or gateway machines that affect IP addressing, because server-to-client communication may be impossible.

Method 2: Terminate Message in Heartbeat Response

The CA SSO Client sends a heartbeat to the CA SSO Server at a regular interval.

The Direct Notification method may not be suitable for your system if the CA SSO Server cannot send messages directly to the CA SSO Client. If this is the case, you can configure the CA SSO Server to use the routine heartbeat response to send a termination message to an CA SSO Client.

This method causes a delay between the time at which the CA SSO Server decides to terminate a session, and the time at which the terminate session message is actually sent to the CA SSO Client. This is because the CA SSO Server must wait for a heartbeat from the CA SSO Client before it can notify the Client.

Heartbeats from the CA SSO Client to the CA SSO Server

The CA SSO Client is set to terminate a user session if the CA SSO Server does not reply to a certain number of heartbeats. You can turn this off, but this can compromise the security of your system.

This is useful for two reasons:

- If communications are interrupted, the user sessions do not continue indefinitely.
- It prevents a user from starting a CA SSO session, and then stopping the CA SSO Client heartbeat reaching the CA SSO Server (for example, by disconnecting their machine from the network). Permitting this can enable the user to start a subsequent malicious session.

Manual Termination

Manual Termination involves using the SSO Session Administrator to manually terminate user sessions. Manual Termination triggers the CA SSO Server to terminate a specific session.

This requires the installation of the Session Administrator (web interface) to manage user sessions.

You can install the SSO Session Administrator to manually view and close user sessions. This method is not required for session management but provides the extra option of being able to view all open user sessions.

To use this method, you must:

- Install the SSO Session Administrator
- Enable session management
- Create a Session Administrator user with administrative rights

Multiple Session Profiles

You can apply more than one session profile to a user or group.

Also, you can apply a session profile to a group, and then apply additional session profiles to some individual users in that group.

When applying more than one profile to the same user, the most restrictive settings apply. The table in the CA SSO Server Settings section lists all session management settings, and the parameters in order of increasing restrictiveness.

To see the user's effective session profile, in the Policy Manager open the user properties dialog (Users, Datastore, User Datastores, <user datastore name>, User) and select the Session Profile List pane. Pressing the Effective Profile button causes the CA SSO Server to calculate the user's effective profile at that time.

Example: Two Session Profiles Assigned to One User

For example, the following table shows what can happen if a user has two session profiles assigned - each with different values. The effective session management behavior is a combination of the two profiles: the more restrictive settings apply for each option.

Option	Session Profile Applied to Group	Session Profile Applied to User	Effective Profile for User
Name	Finance Group	Three Sessions	-
Comment	Use for the Finance group and Admin staff	Permits three sessions, logs off after 10 minutes	-
Owner	ps-admin	ps-admin	-
Limit Choice	Close Oldest Session	Select Specific Session	Close Oldest Session
Heartbeat Fail Behavior	None	Logoff from SSO	Logoff from SSO
Logout Timeout	5 minutes	10 minutes	5 minutes
Screen Lock Timeout	1 minute	2 minutes	1 minute
User Max Sessions	1 session	3 sessions	1 session

Manage User Sessions with the CA SSO Server

To manage user sessions using the CA SSO Server, you :

1. Enable Session Management
2. Create one or more session profiles using the Policy Manager.
3. Apply the session profiles to individual users or groups of users.

Enable Session Management

You can configure Session Management settings on the CA SSO Server using the Policy Manager.

Note: If you have CA SSO Clients working in backward compatibility mode, you must configure parameters to enable and configure session management for these clients.

Configure the CA SSO Server

This topic describes how to configure the CA SSO Server to enable automatic session management.

To configure the CA SSO Server

1. Launch the Policy Manager.
2. Click Agents and select the default CA SSO Client.
3. Right-click the default CA SSO Client (Agent) and select Properties and then select Session Management.
4. Check the Enable Session Management check box.
5. Complete the remaining fields.
6. Restart the CA SSO Server for the changes to take effect.

More information:

Session Management Settings (see page 182)

Session Management Settings

To set up the CA SSO Server for Session Management, you must configure the following settings:

Setting	Description	Parameters
Enable Session Management	Whether Session Management is enabled or not.	■ **Disabled** (default) ■ **Enabled** Lets CA SSO Clients work with SSO Sessions
Session Expiration	How long the CA SSO Server waits for a client to shut down before continuing to log a user on the new session. Only relevant if using the Direct Notification method.	■ **Integer** Period in seconds (the default is 0 seconds)
Heartbeat Interval	The period between heartbeats.	■ **0** Disabled (no heartbeat is sent) ■ **Integer** Period in seconds between heartbeats (the default is 30 seconds)
Missed Heartbeat Count to Fail Session	Maximum number of heartbeats missed before the Client session terminates. For example, if set to 3, the CA SSO Client terminates a session after the third missed heartbeat response.	■ **0** The session does not terminate if a heartbeat response is missed ■ **Integer** The number of heartbeats missed (the default is 3)
Notification Message Protocol	The protocol that sends the heartbeat. This determines whether a message is sent with the heartbeat.	■ **UDP** This protocol cannot include messages with heartbeat responses ■ **TCP** This protocol can include messages with heartbeat responses (default)

Create a Session Profile

You can use the Policy Manager to create session profiles. These profiles can then be assigned to users to determine CA SSO session behavior.

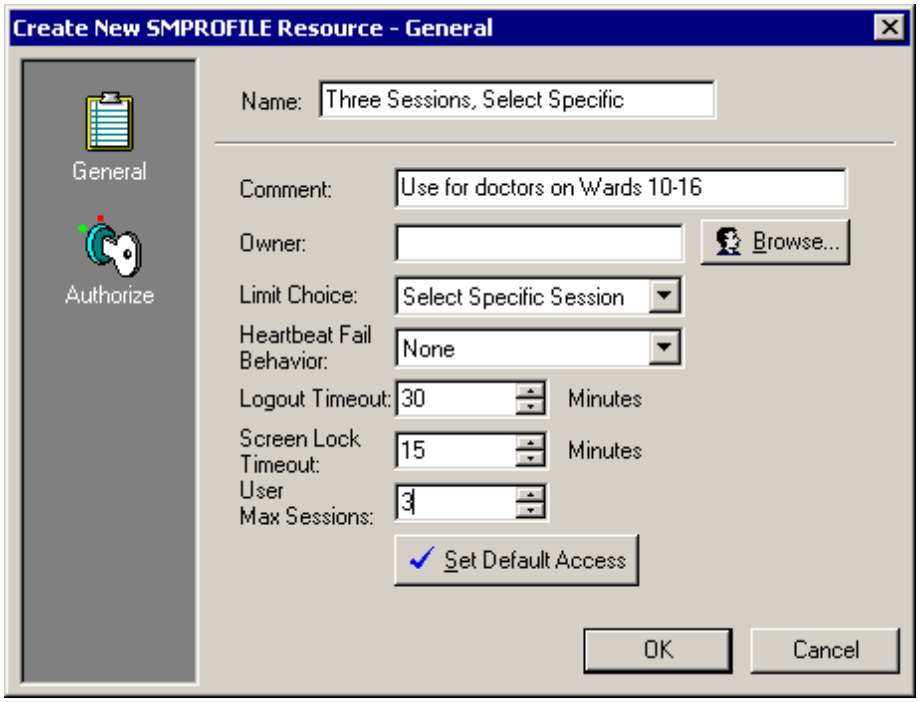
Note: Multiple session profiles can be assigned to a single user. When applying more than one profile to the same user, the most restrictive settings apply.

To create a session profile

1. Launch the Policy Manager.
2. Click Resources, Session Resources, Session Profile.

   The list of existing session profiles appears.
3. Right-click anywhere in the list area, and select New.

   The Create New SMPROFILE dialog opens.



4. In the General dialog, set the behavior for the profile.

   For more information, see SMPROFILE Resource - General Dialog in the *Policy Manager online help*.
5. Click Authorize and set permissions for users or groups to access the new session profile.
6. Click OK.

   The new profile is saved.

Apply a Session Profile to a Single User

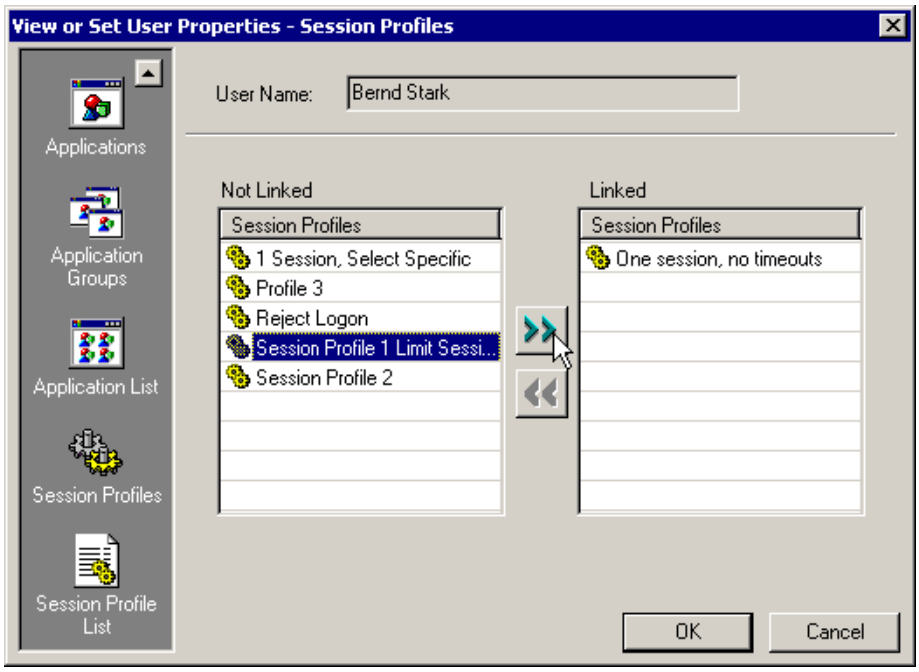
You can use the Policy Manager to apply a session profile to a user to control their CA SSO session behavior.

Note: Multiple session profiles can be assigned to a single user. When applying more than one profile to the same user, the most restrictive settings apply.

To apply a session profile to a single user

1. Launch the Policy Manager.
2. Click Users and select the appropriate data store.
3. Double-click a user to open the User Properties dialog.
4. Click Session Profiles.

   A list of all session profiles that can be applied to the user appears.



5. Select one or more session profiles and move them to the right so they are linked to the user.
6. Click OK.

   The session profiles are assigned to the user.

Apply a Session Profile to a Group

You can use the Policy Manager to apply a session profile to a group to control their CA SSO session behavior.

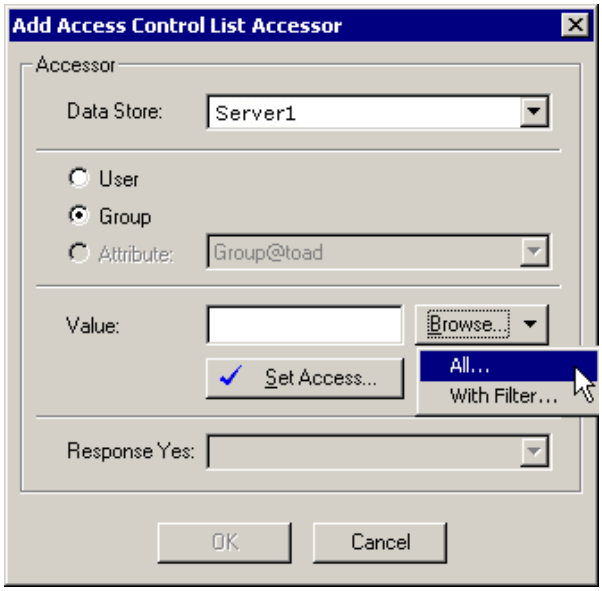
Note: Multiple session profiles can be assigned to a group. When applying more than one profile to a group, the most restrictive settings apply.

To apply a session profile to a group

1. Launch the Policy Manager.
2. Click Resources, Session Resources, Session Profile, and double-click the session profile name you want to assign a group to.

   The View or Set SMPROFILE Properties dialog appears.
3. Click Authorize.
4. Right-click and select Add.

   The Add Access Control List Accessor dialog appears.



5. In the Add Access Control List Accessor dialog, select:
   a. The data store that stores the group.
   b. The Group option.
   c. Group name. If you know the exact name of the group, enter the group name in the Value field.

      Alternatively, click Browse to open the Group Selection dialog, which shows a list of group names. You can either view all group names, or you can filter the list.

6. Click OK when finished.

   The session profile is assigned to the group.

   Note: This procedure can also be used as an alternative for applying a session profile to a user (select the User option at step 5b).

Manage User Sessions with the Session Administrator

The SSO Session Administrator is a web-based application that lets you manually view and terminate CA SSO sessions. The Session Administrator is a standalone tool.

Using the Session Administrator you can:

- Search and view all active CA SSO user sessions
- Search and view a single user's active CA SSO sessions
- Terminate some or all sessions including workstation lockout.

  Note: If you use the Session Administrator to end a user's active session, it only shuts down their CA SSO session(s).

To close all open applications, you must write a logoff script to use with the Session Administrator.

Create a Session Administrator User

Before you can manage user sessions using the SSO Session Administrator, you must set up and assign Session Administrator privileges to a user in the LDAP data store (CA Directory or Active Directory). This lets you launch the Session Administrator and monitor user sessions.

To create a new Session Administrator user

1. In the Policy Manager, create a new user in the LDAP user data store.
2. Run the following selang commands:

   To assign a user administrator rights in ps-ldap, run this command:

   ```
   authorize ROLE ADMIN user_attr("User@ps-ldap") attr_val("cn=<username>") \
   user_dir("ps-ldap") access(Read)
   ```

   The new users are now Session Administrator users meaning they can view and shut down sessions using the SSO Session Administrator.

Launch Session Administrator

The Session Administrator lets you monitor and end user sessions. You must be an SSO Session Administrator to manage user sessions in the Session Administrator.

To Launch the Session Administrator

1. Launch the Session Administrator.
2. Click Start, Programs, CA, Single Sign-On, CA SSO Session Administrator.
3. Log in with your session administrator credentials.

View User Sessions

The Session Administrator lets you view all active user sessions. You must be an SSO Session Administrator to manage user sessions in the Session Administrator.

To view user sessions

1. Launch the Session Administrator.
2. Click Start, Programs, CA, Single Sign-On, CA SSO Session Administrator.
3. Log in with your session administrator credentials.
4. To view user sessions, you can either:
   - Click All Users to view all user sessions.
   - Enter a single user's name and click Single User.
5. Click on the user's name to view their active SSO sessions.

   All active sessions are displayed.

Close User Sessions

The Session Administrator lets you manually terminate user's sessions. You must be an SSO Session Administrator to manage user sessions in the Session Administrator.

To close user sessions

1. Launch the Session Administrator.
2. Click Start, Programs, CA, Single Sign-On, CA SSO Session Administrator.
3. Log in with your session administrator credentials.
4. To view user sessions, you can either:
   - Click All Users to view all user sessions.
   - Enter a single user's name and click Single User.
5. Click on the user's name to view their active SSO sessions.
6. Check the End? check box for each session you want to close.
7. Click Kill Session.

   All relevant sessions are closed.

Chapter 8: Managing Services

This chapter explains how services are managed within CA SSO.

This section contains the following topics:

Users' Application List Updates

Each time a user logs onto CA SSO they are presented with a list of applications that they can access using CA SSO. This application list is retrieved from an application list cache stored on the CA SSO Server. Retrieving this data from the cache is faster than calculating the full application list and typically users' applications do not change frequently.

As new applications are added and deleted, users' application lists can change therefore cache files must be updated periodically. This is essential to avoid the users' application lists becoming out of date.

How the Application List Cache on the CA SSO Client Works

The application list is cached on the CA SSO Client as well as the CA SSO Server. During the course of usual operations, the following steps occur:

1. The CA SSO Client checks with the CA SSO Server to see if the copy it has is up to date.
2. If it is not, it pulls down the application list from the CA SSO Server's cache.

   This saves on network bandwidth because the check is much smaller than the application list.

How the Application Lists Cache on the CA SSO Server Works

The application list cache is created by the CA SSO Server in one of the following ways:

- The Server's background cache generation tool is run
- The user selects Refresh Application List on the Client

Run ps-bgc to create the cache as part of implementing your CA SSO system. Do this before users start logging onto their CA SSO Clients to improve CA SSO Server performance and reduce login time on the CA SSO Client.

When the user clicks Refresh Application List on the Client, the following steps occur:

1. The CA SSO Client requests the application list from the CA SSO Server.
2. The CA SSO Server prepares an application list and compares this list with the cached application list. If there is a mismatch, the CA SSO Server updates the cache with the latest application list.
3. The CA SSO Server gets the application list information from the application list cache.
4. The CA SSO Server sends the application list to the CA SSO Client.

How the Application List Background Calculation (psbgc) Utility Works

The psbgc utility regenerates the application list cache as a background task, which reduces the CA SSO Server load and improves the CA SSO Client performance during peak times.

The utility must be run on a regular basis. This means that the application list caches are always up-to-date, so users do not have to request a refresh of their application list as often.

Note: The CA SSO administrator must ensure that the Application List Background Calculation utility runs at least once a week to ensure that each user's application list is up-to-date.

You can define the scope of the psbgc utility to determine which particular user or group of users will be updated. You can specify:

- All users within an LDAP container
- All users within a data store
- All users below a certain branch of a directory
- An individual user
- A group of users

Note: Every time the Application List psbgc utility runs, the utility looks at each user entry and creates a cache of each user's application list.

psbgc - Update Application List Cache

Valid on UNIX and Windows.

Use the psbgc to keep users' applications lists up-to-date on the SSO Server. Specifically this command will update the application list cache. You can use a number of different options with the psbgc utility. Each option lets you perform a different task or configure how that task runs.

Windows syntax:

```
psbgc -a [administrator name] -p [administrator password] parameters
```

UNIX syntax:

```
./psbgc.sh -a [administrator name] -p [administrator password] parameters
```

-h/-help

(Optional) Displays the help.

-a/-administrator

Specifies the administrator's user name. This user must have administrative rights to the SSO Server.

When you install the SSO Server, a psbgc utility administrator is created by default. This user is called “ps-bgc” and the corresponding password is “ps-bgc”. You should change this password to be more secure.

Note: When you run any psbgc utility command, you must specify an administrator's username and corresponding password. By default, the administrative user name and password are ps-bgc.

-p/-password

Specifies the administrator's password ("ps-bgc" by default).

-c/-container

(Optional) Specifies the LDAP container name where the users are stored. This lets you calculate the application list for a specific subset of users within this container rather than all users in the directory. If this parameter is not specified, the psbgc updates all users within the user data store's base container (base path).

-d/-datastore

(Optional) Specifies the user data store where the users are stored. This lets you calculate the application list for all users within this data store. If not specified psbgc operates on all data stores.

-g/-group

(Optional) Specifies the name of the group of users for whom you want to calculate an application list.

-u/-user

(Optional) Specifies a single user for when you want to calculate a single user's application list.

-r/-recursive

(Optional) Specifies that the psbgc utility should calculate application lists recursively for all users within a specified container. This should be used in conjunction with the -c option if the specified container holds sub-containers that you also wish to search.

This means that you can use the psbgc utility to update all users' application lists within a hierarchy.

-x

(Optional) Specifies that the psbgc should use the paging technique and only return 200 users at a time. We highly recommended this is if you have more than 200 users to update and you are using a data store that supports paging.

-i/-ini

(Optional) Specifies the path to psbgc.ini configuration file.

By default this is stored in the SSO Server's bin directory on the SSO Server computer. If you are in the bin directory you do not need to specify where the psbgc.ini file is located.

Examples:

The following example shows how to perform the following tasks on a UNIX platform:

- Log onto the SSO Server (this example uses the default administrator name “ps-bgc” and password “password”)
- Calculate and cache application lists for all users in all data stores
- Return results in batches of 200

```
psbgc -a ps-bgc -p password -x
```

- Calculate the application list for one user

```
psbgc -a ps-bgc -p password -d ps-ldap -c “ou=QA” -u “John Smith”
```

- Calculate the application list for all users under a specified container in an Active Directory data store

```
./psbgc -a ps-bgc -p password -d AD -c “ou=QA”
```

psbgc.ini Configuration

This table lists all the sections and keynames (also called tokens or settings) of the psbgc.ini file together with a brief description about each value and how it affects the behavior of the psbgc utility.

By default this file is installed in the bin directory of the SSO Server and the psbgc will automatically refer to the psbgc.ini in this location unless you use -i to specify that the psbgc.ini is in a different location.

ServerHost

Specifies the name of the machine (or IP address) where SSO Server is installed.

Value: Computer name or IP address

Default: localhost

AuthHost

Specifies the authentication host used to authenticate psbgc administrative user (whose name and password are provided via -a and -p command line parameters).

Value: Computer name

Default: PSBGC_Authhost

AuthMethod

Specifies the authentication method used to authenticate the administrative user.

Value: [SSO|LDAP|EAC]

Default: SSO

NameAttribute

Specifies the user name attribute for authentication.

Default: USERNAME

PasswordAttribute

Specifies the password attribute for authentication.

Default: PASSWORD

HostAttribute

Specifies authentication attribute containing the name of the authentication host used for authentication.

Default: AUTH_HOST_NAME

MsgFilePath

Specifies the path to the message file.

Windows default: C:\Program Files\CA\Single Sign-On\Server\Lang

UNIX default: /opt/CA/SSO/Server/Lang

MsgFileName

Specifies the name of the message file. The message file contains error descriptions that are presented to the end user.

Value: File name

Default: ENU.msg

KeyFileName

Specifies the name of the temporary file to cache the SSO Server's public keys used for encryption of the communication channel between psbgc and the SSO Server.

Value: File name

Default: PolSrv.key

Interval

Specifies the time interval between requests. This defines how long the psbgc will wait before it sends each request to the SSO Server.

Value: Time in milliseconds

Default: 1000

RecvBuffSize

Specifies the size of the psbgc communication buffer. You may need to increase the value of this token if you are working with large numbers of users, especially if you are not using the page mode. This helps the system to transmit the list of user names from the server to psbgc. You must also set the SendRecvSize parameter, in the SSO Server, to be the same value.

Value: File size in KB

Default: [no default]

CommMode

Specifies the communication mode that the utility uses to communicate with CA SSO Server.

Value: [0|1]

Where

0

Specifies compatible mode

1

Specifies FIPS-mode.

Default: 0

CertFilePath

Specifies the path to the CA SSO Server public certificate file. The file is a .pem file used in CommMode 1.

CreateUserAPPLCache

Specifies that the psbgc utility requests the SSO Server to cache authorization rules to build the application lists.

Value: [Yes|No]

Yes

Specifies that the psbgc utility retrieves and caches the user-application list.

No

Specifies that the psbgc utility does not cache the user-application list.

Default: No

Note: We recommend that you set the CreateUserAPPLCache to Yes for increased performance under heavy loads on the CA SSO Server.

Keys for Session Encryption

Key management is one of the essential tasks of CA SSO management.

Key pair generation for session encryption is performed with a utility named GenKeyPair supplied with CA SSO. The lifetime of the public/secret key pair is unlimited. However, it is recommended to regenerate the pair every one to five years, as well as whenever CA SSO is re-installed. Because key pair generation is a resource intensive process, it is not practical to run the process frequently.

CA SSO components are installed using a default public/secret key pair.

When a new key is set at the CA SSO Server, the server sends the new public key to the CA SSO Clients connected to it.

Running GenKeyPair

The file name is different on UNIX and Windows.

UNIX

GenKeyPair

Windows

GenKeyPair.exe

The utility can be run at any time, but for the new keys to take effect you have to shut down the SSO Server and then start it up again.

The Automatic Password Generation Utility

The CA SSO Server has a password auto-gen utility for automatically generating passwords for user applications. This option increases security, because when it is enabled, it generates random passwords based on password rules set by administrators. Auto-gen passwords are usually harder to guess than those chosen by users. In addition, when this feature is employed, administrators can set stricter password rules and more frequent password changes, since user resistance is no longer a problem.

Once password auto-gen is implemented, users will no longer know their applications' passwords and therefore will not be able to access applications directly, but only by means of CA SSO. This must reduce help desk calls from users who forget passwords and can improve the system administrators' tracking of application use.

Communication Between Components

The following topics explain the SSO communication protocols:

Communication Protocols Between Components

CA SSO has three modes of communication that are supported:

- Compatible mode
- TLS mode of communication
- TLS mode of communication using FIPS-compliant libraries

Encrypting Communications Between Components in Compatible Mode

Communications between the components is encrypted using a system based on a combination of ElGamal Public Key and Triple DES encryption.

Session Encryption

The session is encrypted with Triple DES using three keys. These three keys are referred to as the *session key*. They are generated using a strong random number generator.

The session key is encrypted using ElGamal (also known as Half-Certified Diffie-Hellman). ElGamal encryption keys are formed by three public keys, referred to collectively as *the public key*, and one secret key. Encryption is done using the public key. Decryption is possible only with the secret key.

Any component can encrypt a message using the server's public key, but *only* the server can decrypt the message by using its secret key.

How Encryption Works

1. The CA SSO Client generates a session key.
2. The CA SSO Client then initiates a connection with the CA SSO Server.
3. Once the connection is established, the CA SSO Client encrypts the session key with the CA SSO Server's public key, and sends the encrypted session key to the CA SSO Server.
4. The CA SSO Server decrypts the session key using the server's secret key.
5. The CA SSO Server checks the decrypted session key. If it is accepted, the secure session is established.
6. If the process fails, the reason may be an incorrect public key. The CA SSO Server sends its public key to the CA SSO Client, which prompts the user to accept it. If the user accepts it, the public key sent is cached and the CA SSO Client resumes the process at step 3 above. Otherwise, the connection is terminated.

TLS Communication Between Components

Communication between components is over an SSL channel. A Pem certificate file and a Key file are used in the process of establishing the SSL channel between the components.

FIPS-only Mode

This mode of Communication is TLS communication using FIPS-compliant libraries.

Mixed Mode of Communication

This mode of communication is supported by CA SSO Server and Authentication Agents. In this mode, the server and authentication agents, support the previously mentioned modes of communication.

Ports Used by CA SSO

CA SSO uses the following default ports:

Component	Protocols	Port Number	Description and Configuration
SSO Server	TCP	13980 13981 - SSL port for SSO Server	This is the TCP port where the SSO Server will listen. El Gamel/3DES This is configured on the Policy Manager:
	UDP	13990	Configuration Resources, SSO Server Settings, Communication
SSO Server Token Directory “pstd”	LDAP	13390	The LDAP communication port. This uses SSL. This is configured in the CA Directory configuration files + Policy Manager: Configuration Resources, Token Dir, PSTD, General
	Telnet	13380	This is the console port for debugging. This is configured in the CA Directory Config files. This uses SSL.
SSO Server User Directory “ps”	LDAP	13389	This is the LDAP communication port. This uses SSL. This is configured in: CA Directory config files+ Policy Manager: Data Stores, User Data Stores, ps-ldap, General

Component	Protocols	Port Number	Description and Configuration
	Telnet	13379	This is the console port for debugging. This uses SSL. This is configured in: CA Directory Config files.
CA Access Control	TCP protocol	8891	This is the TCP port where CA Access Control listens. This does not use SSL. This is not available for editing.
SSO Client (Session Management)	UDP	20001-20201	This is the Session Management communications port. This does not use SSL. This is configured in the Client.ini file.
Certificate Authentication Agent	TCP/IP	13987 13988 SSL port Not on UNIX	This is the port for the ticket granting agent (TGA). This uses SSL. This is configured in the CA_certtga.ini file in the Certificate authentication agent install location.
LDAP Authentication Agent	TCP/IP	17979 17980 SSL port Not on UNIX	This is the port for the ticket granting agent (TGA). This does not use SSL. This is configured in the CA_ldaptga.ini file in the LDAP authentication agent install location.
RSA/SecurID Authentication Agent	TCP/IP	13969 13970 SSL port Windows	This is the port for the ticket granting agent (TGA). This does not use SSL. This is configured in the CA_rsatga.ini file in the RSA authentication agent install location.

Chapter 9: User Authentication

This chapter discusses how CA SSO authenticates users. It then describes the different authentication methods that CA SSO can work with.

For information about authenticating to applications, see the “Implementing Authentication" chapter in the *Implementation Guide.*

This section contains the following topics:

About Authentication

Primary authentication is the process by which users prove their identity to CA SSO. To do this, the user might provide a valid user name and password, or biometric information such as a fingerprint or retina scan. After users have proven their identity to CA SSO, they receive their application lists and are entitled to access CA SSO services.

Usually, users only go through primary authentication the first time they use CA SSO. However, users are sometimes asked to re-authenticate (do primary authentication again):

- When they unlock StationLock
- When they log onto a sensitive application
- When the time since the last primary authentication exceeds a designated value

To log onto an application, users also go through application authentication. Application authentication is the process by which a user is authenticated for a particular application.

The method of primary authentication does not dictate the type of application authentication that will be used.

Primary Authentication

Primary authentication is the process by which users prove their identity to CA SSO. To do this, the user might provide a valid user name and password, or biometric information such as a fingerprint or retina scan. After users have proven their identity to CA SSO, they receive their application lists and are entitled to access CA SSO services.

How Primary Authentication Works

The primary authentication process involves the CA SSO Server, the CA SSO Client, and the authentication agent. The primary authentication process is as follows:

1. The user starts the CA SSO Client on their workstation.
2. The CA SSO Client reads the list of server sets and authentication methods from the auth.ini file.
3. The authentication dialog appears, prompting the user to:
   - Select the appropriate server set
   - Provide their credentials, such as username and password, biometric information, or a smart card
4. The CA SSO Client gives the user's credentials to the authentication plug-in specified in the auth.ini configuration file, for example, LDAP, Certificate or Windows.
5. The authentication plug-in verifies that the credentials are valid, either using its own built-in mechanism or (the more common scenario) by sending them to an authentication agent with a verification request.
6. If the credentials are deemed invalid, the authentication agent sends an error message to the CA SSO Client, informing it that the primary authentication has failed.

   If the verification is successful, the authentication agent creates an SSO ticket, encrypts it using a configured encryption key, and sends it to the CA SSO Client. The SSO ticket is a string that includes user identification, authentication method, and time stamp. The ticket is valid for a defined number of hours.
7. The CA SSO Client sends the SSO ticket to the CA SSO Server.
8. The CA SSO Server verifies the SSO ticket.
9. If the ticket is not valid, authentication fails and the user receives an error message.

   If the ticket is valid, the CA SSO Server retrieves the list of the applications that the user is authorized to use from the user data store and sends the list to the CA SSO Client.
10. The CA SSO Client displays the list of applications. The user can now start work.

Primary Authentication Components

This section describes the following system components that are used in the primary authentication process:

- The authentication host
- The authentication agent
- The SSO ticket
- The SSO security token
- The application list

The Authentication Host

The authentication host is a computer on which the authentication agent resides. The authentication software is the component that verifies the user credentials.

The authentication host can be a general-purpose server running NetWare, Windows OS, or a specialized server such as RSA ACE, or SAFLINK.

The Authentication Agent

Authentication agents are used to provide a bridge between the CA SSO Client and various authentications systems. These authentication systems could be an algorithm, authentication server, or data store that can be used to perform authentication on a set of credentials.

CA SSO includes ready-made agents for various authentication systems.

Note: The SSO Authentication method does not require an authentication agent as the CA SSO Server deals with the credentials directly.

The SSO Ticket

The SSO ticket is a partially encrypted string containing the information needed for authenticating the user to the CA SSO Server. This information includes:

- The name of the CA SSO Server AUTHHOST that must be used to decrypt the ticket and then perform authentication of the decrypted values. This is the only unencrypted part of the ticket.

- User identification. Used by the CA SSO Server to identify the user.
- Authentication method.
- Time stamp.

How the SSO Ticket Is Used

The SSO ticket is used by the CA SSO Client to prove to the CA SSO Server that a given user's credentials were successfully validated by the authentication method's authentication agent.

For all client-side authentication methods (that is, everything except the SSO authentication method), the CA SSO Client Authenticator passes the user's method specific credentials, for example, username and password, biometric information, X509 certificate, to an authentication agent specific to that authentication method.

If the authentication agent deems the credentials to be valid, it creates a ticket for the user, which is encrypted using a secret key that the authentication agent shares with the associated CA SSO Server AUTHOST.

This ticket is passed back to the CA SSO Client and then onto the CA SSO Server. If the CA SSO Server decides that the ticket is valid, it issues the CA SSO Client with an SSO Security Token, associated with the user specified in the ticket.

Encrypting the SSO ticket

The ticket is encrypted using a secret key that both the primary authentication agent and the CA SSO Server know.

We recommend that you use a different key for each authentication host.

The CA SSO Server stores the key for every authentication host in an AUTHHOST record. The authentication host stores the key in a protected area, depending on the platform. These two copies of the key must match.

Note: An Authentication Agent installer asks for an encryption key. The key you provide must match the key defined on the AUTHOST entry for this authentication agent in the policy data store. If no key was set when an AUTHHOST entry was created, an administrator needs to modify the Authhost entry with a key provided during authentication agent installation.

The key value on the authentication host must be protected to prevent unauthorized access and keep it from being used as a convenient starting point for gaining illicit access to applications permitted to the user.

The SSO Security Token

After the CA SSO Server has validated the SSO Ticket, or in the case of authentication using the SSO authentication method, the user's credentials, the CA SSO Server issues the engine service with a security token that can be used to retrieve the user's application list or perform other operations associated with that particular user.

If the security token must expire while it is still being used by the CA SSO Client, the CA SSO Server requires a re-authentication to generate a new security token.

The Application List

Depending on the CA SSO Client configuration and the CA SSO Client application used, the list can be displayed as application icons in a floating toolbar, in a tree view, as a program group, or as a part of the Start menu.

After primary authentication has been carried out, the CA SSO Client application requests an application list. To build the application list, the CA SSO Server uses the authorizations in the data stores. There are two options for handling CA SSO Client requests: building or rebuilding the application list each time the client requests it, and building and caching the application list.

Since the application list needs to be rebuilt only if there has been a change in user authorizations, the second option improves performance by reducing the frequency of application list rebuilds. Upon logon, a cached application list is received.

Primary Authentication Methods

Each CA SSO user is associated with one or more primary authentication methods. The primary authentication method determines which primary authentication method that the CA SSO Server accepts for that user.

Within the one CA SSO system, users can be defined to use a number of different primary authentication methods.

Choosing the Authentication Method

Various authentication methods can be used to confirm the end user’s identity. The primary authentication process can be adapted to support a wide variety of device-assisted logon, including smart cards and biometrics.

CA SSO lets you choose the method of primary authentication, and which system component will provide the proof of the user's identity.

Authentication Method	Component That Authenticates User
Native CA SSO	The CA SSO Server (acting to validate the user's credentials in addition to its other functions)
WIN	A primary authentication agent that resides on an operating-system server (such as a Windows NT server)
LDAP	A primary authentication agent that resides on a security server
RSA SecurID	

The property AuthenticationMethod in the user entry specifies the primary authentication methods that are valid for the user. The system administrator can enter values for this property when defining a new user or when updating an existing user.

Authenticating with Native CA SSO

If you use the native SSO authentication method, the CA SSO Server performs the actual credential validation, checking the user ID and password against a password stored in the data stores for primary authentication.

All the components needed for CA SSO native authentication are installed automatically with the CA SSO Server software.

CA SSO native authentication is completely independent from the user's network sign-on.

Depending on configuration of the SSO Authenticator, the user may be taken directly to the SSO Authentication method, or the user may have to select it from the server set selection page,

The SSO Authentication method requires the user to enter a username and a password. These values are passed back to the CA SSO Server, which checks the user record in the data stores and if it finds the password in the SSO logon info, it returns an SSO token to the CA SSO Client.

The CA SSO Client caches the SSO token and uses it whenever it needs to request information from the CA SSO Server specific to that particular user.

To use SSO native authentication, the user has to be both defined as using SSO native authentication and allowed to authenticate against the CA SSO Server (that is, the CA SSO Server also acts as the authentication host).

Note: SSO Authentication does not support hierarchical name spaces.

Integrate the Operating System and SSO Authentication Without the GINA

Two options are available to the SSO users when logging onto a workstation depending on the presence of the SSO GINA and various configuration values.

Users can:

1. Log onto the workstation using OS specific credentials and then, once logged on, authenticate separately to SSO using SSO specific credentials.
2. Use the SSO GINA and logon to the workstation using SSO specific credentials.

   Note: For Vista users, use the SSO Credential Provider.

Some authentication methods (Windows and Certificate) provide a mechanism for automatically authenticating a user without having to prompt the user for credentials.

For such authentication methods, SSO primary authentication can be integrated with the operating system logon so that the user only has to perform their normal OS logon; that is users do not have to change their work habits, they simply log onto the OS as they always have.

Perform the following steps to set this up:

1. The OS user needs to be configured to automatically start up one of the CA SSO Client applications once the OS user is logged on to the work station.
2. The CA SSO Client application needs to be configured (if appropriate) to automatically perform primary SSO logon when it is started.
3. The SSO Authenticator needs to be configured to skip server set selection in favor an authentication method that supports automatic authentication.
4. The chosen authentication method needs to be configured to use automatic authentication.

Windows Authentication

The Authentication Host Name (Hostname) for Windows authentication cannot contain more than 15 characters. This is because Windows limits the name of a server or station to no more than 15 characters.

The name of the server that hosts the Windows authentication agent must be specified in the ServerSet section of the Auth.ini file.

When the Windows authentication agent and the user belong to different domains, the following situations arise regarding user authentication:

- The Windows authentication agent is installed on a machine that belongs to a trusting domain and the user is installed on a machine that belongs to a trusted domain: the user cannot authenticate
- The Windows authentication agent is installed on a machine that belongs to a trusted domain and the user is installed on a machine that belongs to a trusting domain: the user can authenticate.
- Both domains are trusted and trusting (a two-way relation): the user can authenticate.

The SSO Client communicates with the Windows authentication agent via a mechanism called named pipes. This mechanism lets the agent identify the user at the other end of the connection, via a standard operating system service. The agent queries the Windows operating system to find out who logged into the connection from which the request arrived.

Chapter 10: Launching Applications

This section contains the following topics:

How Applications Are Launched

Here is a summary of what happens when a user selects an SSO-enabled application from their application list. This process assumes that the user has already successfully authenticated, and has a valid SSO token on their computer.

1. The user selects an application from the list of SSO-enabled applications on their computer and the CA SSO Client sends the token and an application identifier to the CA SSO Server.
2. The CA SSO Server retrieves the appropriate logon script and the relevant logon variables for that user (user ID and password or ticket), and then sends them back to the CA SSO Client.
3. The CA SSO Client then runs the logon script, which launches the application and inserts the logon variable when required. The logon script has two main responsibilities:
   - to start up the application
   - to enter the user's logon variables (user ID and password or ticket)

Launching Applications on Windows Vista, Windows 2008, and Windows 7

Windows Vista, Windows 2008, and Windows 7 manage applications differently from other platforms; so, the CA SSO Client also behaves differently when you try to launch applications.

On these operating systems, applications are classified as standard applications and elevated applications. *Standard* applications are applications that do not need to modify or write to system files. *Elevated* applications are applications that may need to modify or write to restricted system files. As a standard user on these operating systems, you can run standard applications only. To run elevated applications, you must have administrator privileges.

Whenever you try to access an elevated application, Windows displays a User Access Control dialog prompting you for consent to run applications in elevated mode. You may be prompted with a UAC dialog twice based on your application resource settings set in the CA SSO Policy Manager.

To launch applications on Windows Vista, Windows 2008, or Windows 7

1. Log into CA SSO.

   The CA SSO LaunchBar or CA SSO Tools displays a list of applications you can access.

2. Click an application from the application list.

   Based on the settings configured for your application, the CA SSO client does one of the following actions:

   - Displays a User Access Control (UAC) dialog prompting you for consent to run the selected application in an elevated mode.
   - Launches the selected application or a login dialog if you are accessing the application for the first time.

3. (Optional) Enter the administrator credentials in the UAC dialog, and click OK.

   Note: This step is required only if a UAC prompt is displayed on step 3.

   Based on the settings configured for your application, the CA SSO client does one of the following actions:

   - Displays a UAC dialog prompting you to consent to launch the elevated application.
   - Launches the selected application or a login dialog if you are accessing the application for the first time.

4. (Optional) Enter the administrator credentials in the UAC dialog, and click OK.

   Note: This step is required only if a UAC prompt is displayed on step 3.

5. Enter the user credentials for the selected application, and click OK.

The CA SSO client launches the selected application.

Application List Refresh

The Application List Refresh is a setting that you can configure to periodically check the CA SSO Server for any changes to the users' application list and, if any changes have occurred, automatically update the user’s application list on the CA SSO Client, if any changes have occurred.

When to Use Application List Refresh

The user's application list is only updated under the following circumstances:

- When the user requests a refresh (for example, by pressing the 'refresh application list' item on the Launchbar properties menu)
- When the automatic application list refresh is run
- When the Client comes back online after working offline

In the first case, the Client tells the Server to refresh the cache, and then send down the updated application list. This is obviously the more expensive operation as it forces the Server to recalculate, and is only performed when the end-user explicitly requests it.

In the last two cases, the Client checks the Server asking for a comparison between the application list the Client has cached, and the one the Server has cached. If there is a difference, the Server will return the updated application list. If not, a match message is returned; this is cheaper than returning the application list every time.

The application list refresh functionality is extremely important in an environment where users stay logged onto the CA SSO Client for extended periods of time, for example, more than a day.

Configure Automatic Application List Refresh

You can configure how often the CA SSO Client downloads the users' precalculated application lists from the CA SSO Server. To regularly calculate users' application lists on the CA SSO Server we recommend that you periodically run the psbgc utility using a scheduler.

To configure an automatic regular application list refresh

1. Open the Client.ini file.

   By default this is installed in the following location:

   C:\Program Files\CA\Single Sign-On\Client\cfg

2. Find the [AppListRefresh] section.

3. Edit the following values:

   Enabled

   Defines whether you want to turn the SSO Client's automatic application list refresh on.

   If this is set to 'no', the rest of the tokens in this section are ignored.

   Value: [yes|no]

   Default: no

   Interval

   Defines the time between checks for an updated application list.

   Note: If this value is set, then the EarliestStartTime and LatestStartTime values are ignored.

   Value: *time in [days]d[hours]h[minutes]m[seconds]s*

   Default: [no default]

   EarliestStartTime

   In conjunction with LatestStartTime, defines the time period within which a daily refresh occurs (random point between EarliestStartTime and LatestStartTime). You might want to schedule this during low-network traffic periods.

   If these values are set, they are only used if the 'Interval' token is not set.

   The time is specified as a 24 hour clock: for example, 21:31 indicates 9:31 pm.

   Value: *time in [hours]:[minutes]*

   Default: 09:00

   LatestStartTime

   For a full description, see EarliestStartTime in this section.

   Value: *Time in [hours]:[minutes]*

   Default: 17:00

4. Save the Client.ini file.

SSO Scripts

A CA SSO script is a script written in a special extended version of a command language called Tcl. The CA SSO Client runs a Tcl script and performs a task, or series of tasks, for the user. Scripts can be used for a wide variety of tasks. A logon script, for example, is written to automatically log a user in to an application (automatically insert the correct user's name and password in the relevant fields of the logon screens). Scripts can also be written and configured to be executed when a specific event occurs, such as CA SSO Signoff or a Windows lock event.

There is some overhead to create these scripts up front, but they provide excellent flexibility and let you write scripts to automate almost any task that a user could perform. You should think about tasks that users perform that could be automated.

CA SSO scripts are written in a special extended version of the Tcl scripting language that gives you the use of variables, conditions, loops, procedures, and other common programming devices with a minimum of complexity. Prior experience with Tcl is not required to be able to write these, but some programming experience is an advantage. The security or system administrator in charge of CA SSO is responsible for preparing the logon scripts. These scripts are written during implementation and typically do not affect the day-to-day administration of CA SSO.

You can also use the CA SSO Application Wizard to add your applications to CA SSO without the need to learn Tcl scripting for basic scripting tasks.

Here is an example of a very simple Tcl script that launches a password-protected Word document:

```
sso run -path "C:\\Program\ Files\\Microsoft\ Office\\Office11\\Winword.exe" -args
"C:\\SSO Test.doc"
sso window -title "Password"
sso type -text "$_PASSWORD"
sso type -text "{enter}"
```

The logon variables that appear in many logon scripts are $_HOST, $_LOGINNAME, and $_PASSWORD. The CA SSO Client on the user's workstation replaces these variables with the values received from the CA SSO Server.

Symbol	Meaning
$	Tcl variables
$_	SSO logon variables
#	Comment

For more information about SSO scripts, see the *CA SSO Tcl Scripting Reference Guide*.

Logon Variables

The logon variables include the logon script and the logon data sent to the CA SSO Client. These variables are fetched from the CA SSO Server's data stores. Some variables pertain to the current application, some are specific to the current user in relation to the current application, and some may hold installation-wide data.

The logon variables are stored as properties of the LOGININFO section of the user's record in the LDAP data store. Some of the logon variables are used for authentication (logon credentials) and others provide operational and auditing information (such as time of last logon).

For an illustration of how the logon variables are used, look at the following scenario.

1. Terri selects CICS_TEST from the application list.

   The application record of CICS_TEST in the Access Control data store contains:

   - DIALOG_FILE property with the value CICS.TCL
   - LOGIN_TYPE property with the value AppTicket
   - HOST property with the value MVS_TEST

   In Terri's user record, in the LOGONINFO section relating to CICS_TEST, the property LOGINNAME contains the value UTST021.

2. The CA SSO Server generates an AppTicket and stores the result in the Tcl variable _PASSWORD.
3. The CA SSO Server places the logon name UTST021 in the Tcl variable _LOGINNAME.
4. The server sends the CICS.TCL logon script and the two logon variables _PASSWORD, _LOGINNAME, and _HOST to the CA SSO Client.
5. The CA SSO Client interpreter executes the supplied script, entering the username (_LOGINNAME) and ticket (__PASSWORD) as required.

Sensitive Applications

When the SSO administrator marks an application as *sensitive*, the user is forced to re-authenticate more frequently to use the sensitive application. For a normal application, the CA SSO Server checks the normal expiration time of the SSO token (default expiration time is 8 hours). For a sensitive application, the CA SSO Server checks the sensitive expiration time (default is 5 minutes).

For example, when default expiration times are used, if a user carries out primary authentication at 9:00 A.M. and then selects an application at 9:10 A.M.:

- If the application is a normal application, the user can access it directly
- If it is a sensitive application, the user will be prompted to re-authenticate before using the application

The password a user must enter to log onto a sensitive application is *the password used for SSO primary authentication*, not the password of the specific application. For example, if the chosen primary authentication method is LDAP authentication, and the telnet application is marked as sensitive, the user is asked to provide the LDAP password when selecting the telnet application. When he or she enters the LDAP password, the CA SSO Client runs the primary authentication process before beginning the logon process.

To mark an individual application as sensitive, set the 'Sensitive' checkbox on the Authenticate dialog of the Application properties (in Policy Manager, Resources, Application Resources, Applications). The expiration time for all sensitive applications is set by the Sensitive Expiration value in the Settings dialog of the Communication dialog (in Policy Manager, Resources, Configuration Resources, Policy Server settings)

Note: This can also be done using Policy Manager. Navigate to Resources, configuration resources, policy server settings, communication.

Application Authentication

All application logons supported by CA SSO follow the same overall process. The specific sub-section of application logon that handles the way the user is authenticated to the application is called application authentication.

Different Types of Application Authentication

CA SSO offers two different methods of application authentication:

- Password authentication, which can be used for applications on any platform (Windows or Mainframe)
- Ticket authentication, which is only used for Mainframe applications and can be broken down into the following section:
  - PassTickets

The application authentication method used for an SSO-supported application is specified by the value of the LOGIN_TYPE property of the application's record in theCA Access Control data store. Each SSO application record in the data store can have only one application authentication method associated with it. If a value for the LOGIN_TYPE property is not specified, the default method used is native SSO password.

Change Application Authentication Method

An administrator can change the authentication method of an application at any time.

To change the application authentication method for an SSO-enabled application

1. Open the Policy Manager.
2. Select the Resources icon in the left pane.
3. Navigate to Application Resources, Applications, and select your application.

   The View or Set APPL Properties - General window appears.
4. Click the Authentication button.

   The Authentication window appears.
5. Edit the Login Type field.
6. Click OK twice to save the change.

Application Authentication Using Passwords

The most common way to implement SSO for an application is to use password-based authentication. When using the password authentication method, CA SSO supplies the application with a user ID and a password for the user who is attempting to use the application.

The following steps describe the process of logging in using passwords:

1. The CA SSO Server gets logon variables from the LoginInfo section of the user record that relates to the application. If the application is linked to a master application, the credentials are taken from the LoginInfo section that relates to the master application.

   Logon variables include the current password (PASSWORD) and the next password (NEXTPWD). The current password, which is used to perform the current logon, is identical to the password already stored in the application or in the password file used by the application. If the next password is non-null it indicates that the script must change the password after logging in to the application.

   To check/change this information, use Policy Manager to view the user's application list:

   In the relevant data store in the Users worksheet, select the User, Application List. Select the Application and then press the Update Login Information dialog. Under Login Information is the current Login Name and Password; if you change these they are applied to the NEXTPWD (the next time the application is run).

2. The CA SSO Server checks whether the current user password has expired (an event controlled by CA SSO, not by the application itself).

   If the User Password Never Expires feature is enabled, or the Ignore Password Policy check is set, then the CA SSO Server skips this check.

3. If the current password has expired and the Password Generation feature is not enabled, the CA SSO Server sends a message to the CA SSO Client that the password has expired and the CA SSO Client then prompts the user for a new password.

   If a new password is provided by the user, the CA SSO Client sends it to the CA SSO Server, which checks if the new password meets the rules of the password policy linked to the application. If the new password is valid, it is recorded in LoginInfo as the next password (NEXTPWD). If the new password does not meet the password policy's requirements, the password is rejected by the CA SSO Server and the user is prompted to supply a new password.

   If the user is working offline, the password policy cannot be checked. A way to apply the password policy in these circumstances is to write a script procedure that goes through all of the password policy settings (for example, checks password length, checks for the presence of special characters). This script function would be included in the global script that the CA SSO Server can include with every application script. The application script must then check if the CA SSO Client is operating in offline mode and, if it is, use the global password policy procedure. If not, the script can rely on the CA SSO Server doing the password policy check.

   If the current password has expired and the Password Generation feature is enabled, the CA SSO Server automatically generates a new password and puts it in the NEXTPWD property in the database, and the process continues.

4. If the current password has not expired, the CA SSO Server sends the CA SSO Client the logon variables and the appropriate logon script.

5. When the CA SSO Client receives a logon script and logon variables from the CA SSO Server, it plugs the variables into the logon script and then begins to execute the logon script. If working offline, these variables and script are taken from the cache on the Client.

6. A properly written logon script for a password-authenticated application checks if a new password was sent by the server (that is, whether the CA SSO Client received a non-null value for the NEXTPWD variable). If a new password is received, then a logon script must change the password during the logon process (set the password with the value returned in NEXTPWD), in the way required by the application.

A properly written logon script must also notify the CA SSO Server about the password change after it is carried out. The CA SSO Server moves the value of the new password (NEXTPWD) to the current password field (PASSWORD), clears the NEXTPWD property, and updates the PWDLCHANGE field value with the time that the password was changed.

Application Authentication Using Tickets (for Mainframe)

A ticket is an encrypted one-time alternative to a password supported by a mainframe. A ticket is generated using parameters from logon info and system time.

Normally, tickets are not reusable-if they are intercepted, they cannot be used to log on again; therefore, they are immune to "replay attacks" and are very secure.

Since a ticket replaces the password, there is no need for password maintenance and password policies. There is no need to set password expiration periods, no need to change passwords, and there is no need to check the quality of new passwords. Applications using ticket authentication do not have a password stored in the login info section of user records.

Ticket-Based Application Method

CA SSO offers built-in support for one ticket format that works with z/OS mainframe applications and common z/OS external security packages:

- The PassTicket format, which uses a ticket algorithm created by IBM. CA SSO support for PassTickets is compatible with all versions of external security packages that themselves support PassTickets: RACF (z/OS Security Server), CA Top Secret, and CA ACF2.

For information about specific versions, see the CA SSO Readme or refer to Support Online.

Ticket-based application authentication is recommended wherever one of the following situations exists:

- There is a way for CA SSO to intervene in the password validation process (like the pre-logon exit of RACF)
- The target system supports PassTickets

In addition to CA SSO's built-in solution for z/OS, it is possible to implement ticket-based authentication for applications in other environments. The primary requirement is that either the application has an exit point (known as a *hook*, or a *callback*, in some environments) by which a ticket can be validated, or the system has built–in ticket support.

PassTicket Authentication Features

The following table contains the features for PassTicket-based authentication:

Feature	PassTicket
Support for applications	Only TSO, IMS, CICS, and APPC applications.

Feature	PassTicket
Support for security packages	RACF, CA Top Secret, CA ACF2.
Control of ticket validation parameters	Attributes fixed by the IBM algorithm and not modifiable by the system administrator.
Key handling	Allows only one key level (Application Key).
Installation	PassTicket support is built into CA SSO and no changes need to be made to the mainframe or to the security package.

PassTicket Authentication

PassTicket is a ticket algorithm created by IBM, similar in concept to the AppTicket concept. It defines a one-time-only password used as an alternative to the password of a z/OS external security package: RACF (zOS Security Server), CA Top Secret, or CA ACF2.

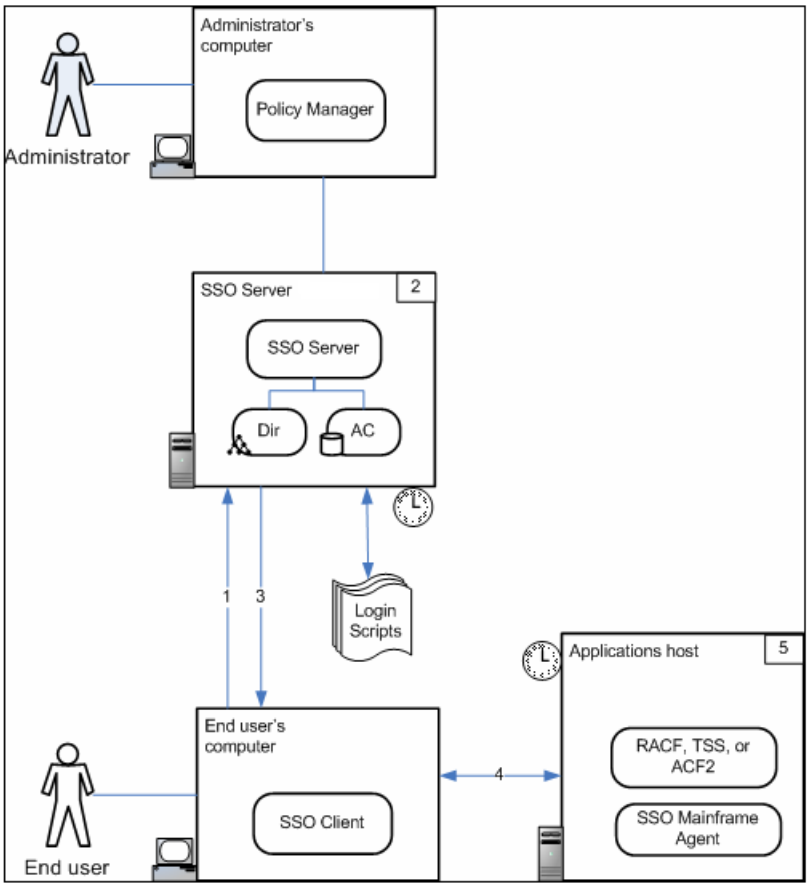
CA SSO support for PassTickets is compatible with all the versions of external security packages that themselves support PassTickets: RACF from v1.9.2, CA Top Secret from v5.0, and CA ACF2 from v6.2.

CA SSO's implementation of PassTicket application authentication does not involve any changes to applications or to the mainframe security package.

In CA SSO's implementation of PassTicket authentication, the CA SSO Server generates PassTickets according to the IBM algorithm. This support is built into CA SSO and requires no installation or configuration on the CA SSO Server side except for defining application profiles and keys (as required by the PassTicket algorithm). The mainframe security package can recognize and verify PassTickets for CA SSO users, assuming that the security package's PassTicket definitions (that is, application profiles and keys) are properly set up. PassTickets can be used only for the following z/OS applications: TSO, IMS, CICS, and APPC. The use of PassTickets does not prevent the use of the usual passwords of the security package.

Logging in Using a PassTicket

The following diagram illustrates application authentication using a PassTicket:



1. The user selects an application and the SSO Client sends user authentication and the application identifier to the SSO Server.
2. If the request is valid and the authentication method is PassTicket, the SSO Server generates a PassTicket based on the user ID, application profile and key (stored in the Access Control data store), and timestamp. System clocks, profiles, and keys on the SSO Server and on the mainframe application host must be synchronized.

   However, system clocks on SSO Client computers do not have to be synchronized with either the SSO Server host or the mainframe application host, since system time on the SSO Client computer is not used in generating or checking the PassTicket.
3. The SSO Server adds the PassTicket as a _PASSWORD value to its response and sends the response to the SSO Client.
4. The SSO Client runs the logon script, sending the user ID and the PassTicket to the application host.
5. The mainframe security package checks whether the password is a valid password or a valid PassTicket. If the PassTicket is valid, the security package allows the logon.

Note: CA SSO does not make any changes in the mainframe security package or in the way the security package handles a PassTicket-based logon. For example, if a valid PassTicket is presented, but according to the security package's database the user is not allowed to log on at the time, the security package will prevent the logon.

For additional information about PassTickets, refer to the documentation of your mainframe security package.

Chapter 11: Launching Web Applications

This section contains the following topics:

About Authenticating Users to Web Applications

CA SSO can work with web applications in the same way that it works with other applications.

You can use the CA SSO Client to handle web applications as well as other applications, or you can use the Web Agent. The Web Agent resides on a web server, and it intercepts each user request for a web application. If the application is unprotected, the request is simply honored. However, if the application is protected, the user requesting the application must be authorized and authenticated.

The Web Agent has the following functions:

- Intercepting any user request to access a web application
- Interacting with the CA SSO Server to authenticate the user and determine if access to the specific resource must be allowed.
- Passing a response to the web server to acknowledge authorization
- Personalizing the web application content to the needs and entitlements of each user.

The term ‘web application’ in this chapter includes restricted web pages.

Supported Authentication Methods

When you install the CA SSO Server, you choose the authentication methods that you want to support. The authentication methods that can be selected when installing the CA SSO Server include:

- LDAP
- Windows
- Certificate
- SSO (proprietary password-based authentication)
- RSA

The Web Agent can support all of the authentication methods supported by your CA SSO Server, or it can support one or more of these authentication methods. Supported authentication methods are defined during the installation of the Web Agent or by customizing the AuthMethods token in the webagent.ini file after the Web Agent is installed.

How the Web Agent Works

To use the Web Agent to authenticate users to web applications, you must have the following in place:

- The Web Agent is installed and running on each of the web servers that host the web sites to be protected.
- The resources and applications that are to be protected are defined in the policy data store.
- The access rules that protect the web applications are defined in the policy data store. Until these definitions are created, the Web Agent grants all requests (access is unlimited).

After you install and start the Web Agent, the web server that hosts the web site requested by the user cannot send information to the user unless the Web Agent permits it. However, once the Web Agent permits the user access to one resource, the Web Agent handles the user's logon to additional web applications and applications without requiring the user to enter their credentials again. Every request by the user for additional web applications is evaluated by the Web Agent to see if the user has authorization to access the application.

When a user tries to access a resource that is not defined as protected, access is granted without going through the authorization process. In this situation, the request is passed to the web server for regular processing.

Three Ways to Authenticate Users to Web Applications

There are three ways to implement CA SSO for web applications:

- Client logon
- Browser logon – requires the Web Agent
- Cookie logon – requires the Web Agent

There are multiple web logon methods because different methods are suited to different web applications and different architectures. You can install all of these methods within the same CA SSO system.

Client Logon

This logon method launches web applications in the same way as any other CA SSO windows application. A Tcl script launches the web browser, inserts the application or page address, and then performs the logon actions.

To use this method you must have:

- The CA SSO Client installed on every user's computer
- The CA SSO Client cookie generation functionality enabled
- Tcl scripts written for each web application or page

This is the most robust and flexible of the three logon methods.

The method can handle complex logon procedures and automatic password changes, but has some limitations when dealing with web applications that use Java applets, Flash, or other non-text methods for their logon procedure.

In the CA SSO Client installation on each client machine, the ssohtmlext.dll file contains a set of HTML extensions for the desktop CA SSO Client to provide script functions to help with writing the scripts for this function.

Example of Web Logon Using the CA SSO Client

You want to allow users to log onto the company's preferred airline Frequent Flyer web site using CA SSO. To do this you use the client logon method because you cannot install the CA SSO Web Agent on the airline's HTML server, and because this logon has a simple repeatable logon process for which you can easily write a Tcl script.

Cookie Logon

The cookie logon method requires the CA SSO Client to create a cookie from the SSO ticket. This cookie is recognized as valid authentication by the Web Agent, which grants the user access to web applications and pages running on that server.

To use this method you must have:

- The CA SSO Client installed on every user's computer
- The CA SSO Client configured to create a cookie when the user authenticates
- The CA SSO Web Agent installed on all servers that host the restricted web applications
- Javascript to handle the logon

This method does **not** require Tcl logon scripting for each web application. Also, it works in conjunction with CA SSO desktop authentication.

To configure the client to create a cookie during authentication, you must modify the [WebAgentIntegration] section of Client.ini to specify the URL(s) of the web server(s) in the CookieURLs keyname. For example, if you were running a web server on a machine on zz.com, you might set the token as CookieURLs=http://www.zz.com

Example of Web Logon using the Cookie

You want to log onto a restricted area on your Intranet using CA SSO. To do this you use the cookie logon method for two reasons. First, you can install the CA SSO web server on the intranet server. Second, you can reuse the CA SSO ticket that was created on your computer during the primary authentication to create a cookie.

Cookie Logon and CA SiteMinder

It is possible to use an SSO cookie to authenticate to SiteMinder protected resources. This requires your enterprise to be using CA SiteMinder (version 6.0 SP4 or greater), and for both SiteMinder and CA SSO to be pointing to the same LDAP datastore.

For further details see the CA SiteMinder documentation set.

Browser Logon

This logon method challenges users for web-based authentication when they try to access a web application or page that is protected by the Web Agent.

You can use this logon method in a thin-client environment, which means that you do not need to have the CA SSO Client installed on users' computers, and no Tcl scripting is required.

To use this method you must have:

- The CA SSO Web Agent installed on all servers that host the restricted web applications
- JavaScript to handle the logon

This method cannot handle complex logon or password management procedures and is not as flexible as the other logon methods because the scripting facilities are restricted, and the password management facilities are not as flexible.

How Browser Logon Works

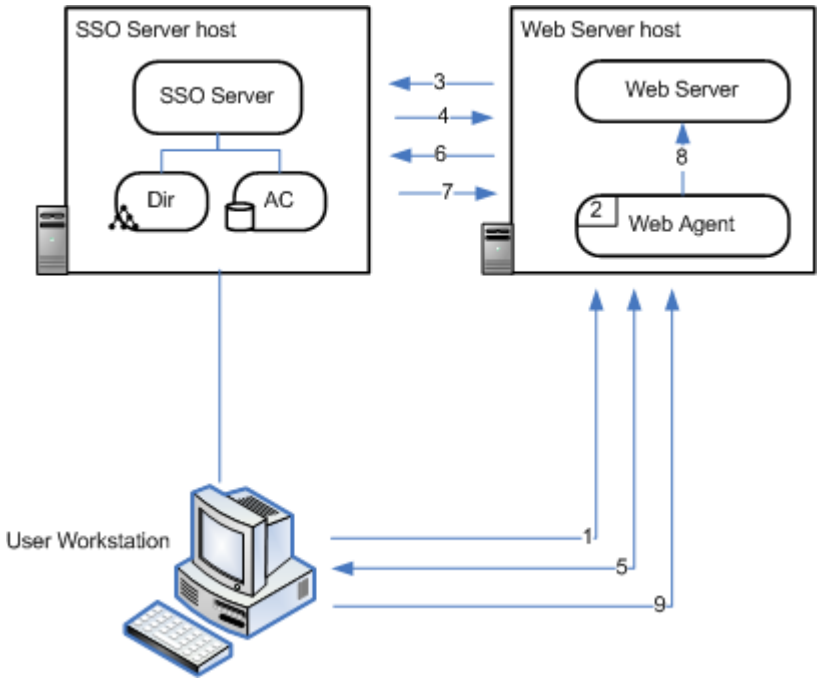
1. The user requests a web application through their web browser.
2. The Web Agent intercepts the request to access the web application
3. The Web Agent connects to the CA SSO Server to check if the web application has access rules.
4. The CA SSO Server sends a message to the Web Agent specifying any access rules for the web application.
5. The Web Agent requests a cookie from the user's computer. This cookie contains the user credentials.

   If there is a cookie, it is sent to the Web Agent.

   If there is no cookie on the user's machine, the Web Agent works with the CA SSO Server to authenticate the user.
6. The web agent sends the user's authentication credentials to the CA SSO Server to check whether this user has rights to access the web application.
7. The CA SSO Server checks whether the user is allowed to access that web application and if they are, sends back a script or HTML file to launch the application.
8. The Web Agent sends any information to the web server in order to allow the page content to be personalized.
9. The user receives access to the web resource.

Diagram of How Browser Logon Works

The following diagram shows how the browser logon method works. The numbers on the diagram correspond to the numbers in the How Browser Logon Works section.



Chapter 12: Working with the Administrative Data Store

This section contains the following topics:

About the Administrative Data Store

CA SSO uses the CA Access Control database as the administrative data store. Even though the complete CA Access Control produce is installed on the CA SSO Server host, CA SSO uses only a subset of this functionality for its administrative data store.

The CA Access Control information used by CA SSO consists of:

- Definitions for resources, applications, application groups, authentication hosts, authentication host groups, and other classes
- Rules of user access to applications and other resources and responses for those rules
- Agent definitions for the CA SSO Client, SSO Web Agents, Application Server Agents, and any custom agents you have defined

You can populate the administrative data store using the Policy Manager. CA Access Control also comes with a command line interface for making changes to its data store, however for CA SSO purposes the Policy Manager is the interface you must be using.

SSO Administrative Users

This section describes a special type of SSO User: the SSO Administrator.

These privileged users have a group authorization attribute set in their own user records. The group authorization attributes are GROUP-ADMIN and GROUP-PWMANAGER are also referred to as SSO Administrators. These are users in the Administrative data store with the administrator property enabled. The Administrative data store is seen in the Policy Manager by selecting Users, Data Store, Administrative Data Stores, *data store name*. The Administrative data store name is always the name of the machine the data store is installed on.

The Administrative data store stores your resources, policies, application information and access control lists. This information is stored in a CA Access Control database.

In addition to this, the Administrative data store has some users in it. These users can have the SSO Administrative privilege, allowing them to assign things like applications to users in the User data store and change passwords.

When you install the CA SSO Server, an administrative user is created for you: ps-admin (you may have renamed this user during your server installation). You can use this user's privileges to log on to the Policy Manager and perform administrative actions.

However, for a user to have SSO functionality, they must be in one of the User data stores. For this reason, we recommend that for the users you want to make administrators that are in the User data store, you create duplicate users in the Administrative data store with the same name and password. These users will be able to perform Administrative tasks like assigning password policies and adding applications to a user's application list, as well as general SSO User tasks like logging onto a CA SSO Client and accessing their SSO-enabled applications.

Note: After logging on to the Policy Manager for the first-time, it is recommended that the default administrator password be changed.

Security Administrative Privileges

To carry out their duties, CA SSO administrators require various security administrative privileges. There are several different types of security administrative privileges in the data stores. The privileges are granted by:

- Global authorization attributes
- Ownership
- Group authorization attributes
- Entries in the ADMIN class

This section discusses administration security privileges, and what each privilege allows its owner to do. The limits of each privilege are also explained. The first three topics listed are covered in this section.

For information about entries in the Admin class and for more details on administration privileges see the CA Access Control documentation.

Global Authorization Attributes

Global authorization attributes are set in the user record. Each global authorization attribute permits the user to perform certain types of functions. These functions and the limits of each global authorization attribute are described in the following sections:

ADMIN

This allows the user to execute almost all administrative activities.

This is the most powerful attribute. It has the limitation that users with the ADMIN attribute are not allowed to log on without typing a password.

If there is only one user in the data store with the ADMIN attribute, that user cannot be deleted, and the ADMIN attribute cannot be removed from that user record.

AUDITOR

This is a special CA Access Control attribute. For information, see the CA Access Control documentation.

Grant Security Administration Privileges

Staff can be assigned as CA SSO administrators and SSO password managers with the Policy Manager.

However, for a user to have SSO functionality, they must be in one of the user data stores. We recommend that you create duplicate users with the same user name and password in the administrative data store for those users in the user data store that require administrative privileges. These users will be able to perform administrative tasks like assigning password policies and adding applications to a user's application list, as well as general SSO User tasks like logging onto an CA SSO Client and accessing their SSO-enabled applications.

For this example we use Pani Patel (user id: patpa01), who exists in the AD-Democorp user data store.

To create a CA SSO user with administrative privileges

1. Log onto the Policy Manager
2. Select Users, Data Store, Administrative Data Stores, <machine name>
3. Right-click on this data store and select New User
4. Enter patpa01 as the User Name, and add the comment "This is a duplicate user with Administrative privileges for Pani Patel."
5. Select Authentication Methods and press the Browse button.

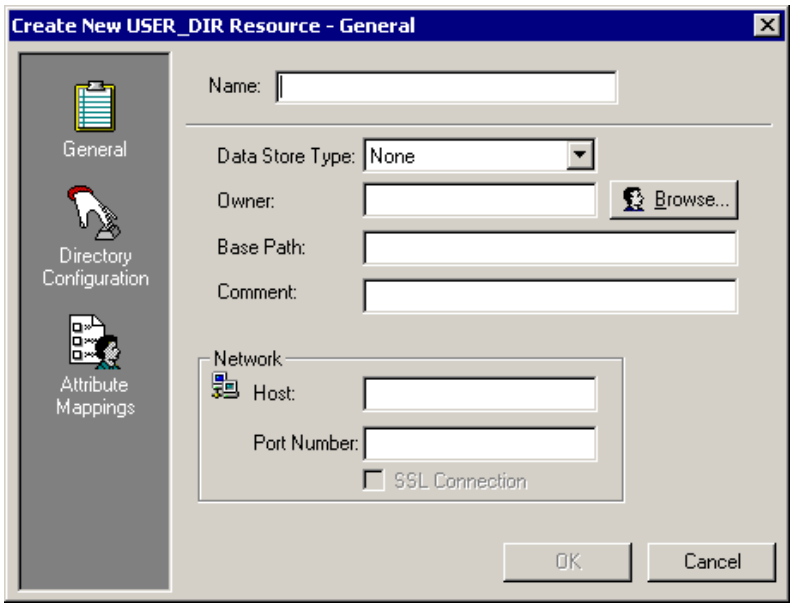
   The Select Authentication Method(s) dialog appears.
6. Assign the EAC authentication method to Pani Patel and press OK.
7. Select User Options from the left pane.
8. Check the Administrator box.
9. Check the Password Never Expires box.
10. Select General from the left pane.
11. Select the Change Password button and select the EAC authentication method. The Change EAC Password dialog appears. Enter the password Pani and click OK.
12. Press OK to save the user and exit the View or Set User Properties dialog.

    You can now see the new Administrative user patpa01 in the Administrative data store. The icon next to the username is blue, indicating that this user is an administrator.

To test that you have created the Administrative user correctly, exit the Policy Manager. Start the Policy Manager up again and log in as patpa01, using the password you have just set. You must be able to log in successfully and perform administrative tasks.

Define and Update Properties for a User Data Store

Whenever you define a new user data store, you must specify its properties by completing a set of dialogs associated with the Create New USER_DIR Resource dialog. The following example shows the Create New USER_DIR Resource dialog; the bar on the left lists the associated dialogs.



The dialogs to define the properties of a user data store are:

General

Defines the data store's name, the type of data store, owner, host, and other properties.

Data Store Configuration

Identifies the administrator and specific information about the data store.

Attribute Mappings

Specifies, for each class, mapping between fields in the user data store and the attributes used by the CA SSO Server.

To change the properties of an existing user data store, you must first display the list of user data stores. Locate the one you want to change and double-click its entry in the list. This displays the View or Set USER_DIR Properties dialog. Make any changes that are necessary and click OK when you are finished.

Classes

A *class* is the type of an object where the objects are the instances of that class. For example, the APPL class defines all the fields and properties that the objects of that type (the applications) will have. The information in these objects is stored in properties. Each object lists values for the same set of properties—the properties appropriate to the type of resource that the class describes.

You can use the following predefined classes or you can define your own classes:

Class	Purpose
AGENT	Agents
AGENT_TYPE	Types of agents
APPL	Applications
GAPPL	Application groups
AUTHHOST	Authentication hosts
GAUTHHOST	Authentication host groups
PWPOLICY	Password policies
RESOURCE_DESC	Resource-allowed accesses
RESPONSE_TAB	Responses
USER_ATTR	User attributes
USER_DIR	User data stores

Note: Names of user-defined classes **cannot** be all uppercase.

The Agent Class (AGENT)

The AGENT class objects contain agent-specific configuration, and are used to associate protected resources with a certain agent instance.

Note: Since resources that belong to different agents can have the same name, you must point to which agent the resource belongs.

Each record in the AGENT class contains the following properties.

Property	Description	Parameter
AGENT_TYPE	The type of agent.	

Property	Description	Parameter
CREATE_TIME	The time the record was created.	Cannot be modified
OWNER	The user or group that owns the record. This user, or anyone with administrator privileges in the group that owns the record, can manage the record.	The default value is the creator of the record.
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
COMMENT	Informational. Agent description/remark.	
SESS_MGMT_ENABLE	The name of the agent; this is the key value.	0 – No Non-zero – Yes
HEARTBEAT_FAIL_AFTER	Number of times an agent /client can miss sending heartbeat before session becomes invalid.	
HEARTBEAT_INTERVAL	The period of time (in sec) between heartbeats	
HEARTBEAT_PROTOCOL	Defined the protocol the client must use to send its heartbeat (keepalive) messages	0 – TCP 1 – UDP
SEND_SESSION_TERMINATION	Defines whether the server must send a session termination notification to the client.	0 – Disabled 1 – Enabled
SESSION_EXPIRATION	Defines how long (in sec) a session can be used before it expires.	
ENFORCE_STATION_ID	Register all authentication tokens to a certain IP address and prevent use of this token form other IP addresses.	0 – Disabled 1 – Enabled 2 – Required
ADD_BASIC_TOKEN_VARIABLES	Specifies whether to store basic authentication header in Token Directory.	1 – Yes 2 – No
ADD_DEFAULT_TOKEN_VARIABLES	Specifies whether to store default headers in Token Directory.	1 – Yes 2 – No
ADD_DYNAMIC_TOKEN_VARIABLES	Specifies whether to store dynamic headers in Token Directory, requires ADD_DEFAULT_TOKEN_VARIABLES to be enabled as well.	1 – Yes 2 – No
CACHE_TO	Specifies the number of seconds for which the authorization decision is valid.	
UPDATE_WHO	The name of the user or group who made the last update.	Cannot be modified

The Agent Type Class (AGENT_TYPE)

For every agent you create, you must define its type using the class AGENT_TYPE. The AGENT_TYPE class defines class lists that the agents work with.

Each record in the AGENT_TYPE class contains the following properties.

Property	Description	Parameter
AGENT_FLAG	Contains information about the attribute in the form of a flag containing the value custom_agent_type. This flag is used whenever you define new custom agent types.	
AGENT_LIST	A list of objects in the AGENT class that were created with this AGENT_TYPE object as the value for the agent_type parameter; for example, this property is updated implicitly when creating an object in the AGENT class.	Cannot be modified
CLASSES	A multi-value list of the resource classes that are relevant to this type of agent.	
COMMENT	A remark.	
CREATE_TIME	The time the record was created.	Cannot be modified
OWNER	The user or group that owns the record. This user, or anyone with administrator privileges in the group that owns the record, can manage the record.	The default is the creator of the record
UPDATE_TIME	The time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who made the last update.	Cannot be modified

The Application Class (APPL)

Applications are objects in the policy data store. The application objects contains information that is used when displaying the application on the user's SSO desktop client computer and when logging the user into the application.

The application record contains details that are needed in two different circumstances:

- Displaying the application to the user—including the icon, the icon name, and any contained applications.
- Logging the user into the application—including the name of the logon script file, the type of logon, and the name of the application host.

Every application that must be accessed through CA SSO must be defined in the APPL class in the policy data store.

You can define the following types of applications:

- **Common** applications—The default access field of a common application is set to ALL or EXECUTE.
- Three types of **restricted** applications:

  Disabled—Use the IS_DISABLED property

  Restricted—Use the DAYTIME property

  Hidden—Use the IS_HIDDEN property
- **Master** applications
- **Container** applications
- **Sensitive** applications

Each record in the APPL class contains the following properties.

Property	Description	Parameter
AZNACL	Authorization rules that specifies who is allowed to access the application, and their access types. Each element in this list contains the following information: ■ User Attr-The user attribute on which the permission is defined (user/group/other field of the user object). ■ Attr Value-The value of the user attribute. ■ User Dir-The user directory to which this user attribute refers. ■ Permitted access-The types of access the granted. The valid access types are: all, execute, none.	
CAPTION	The text under the application icon on the user's desktop.	String (47 character limit) Default value is the name of the APPL record.
COMMENT	A remark; not used during authorization.	String (255 character limit)
CONTAINED_ITEMS	The object names of the contained applications, if the current application object describes a container.	
CREATE_TIME	The time this object was created.	Cannot be modified
DAYTIME	The day and time restrictions that govern when the resource can be accessed.	
DIALOG_FILE	The name of the script in the directory containing the logon sequence for the application. The default directory location is <PolicyServer_full_path>/scripts.	No script (default)
HOST	The name of the host where the application resides.	No host (default)
ICONFILE	The file name or full path of the file (on the client machine) containing the icon that will represent the application on the user's desktop. If just a file name is entered, the search order for the file is: Current directory: ■ Windows system directory ■ Windows directory ■ Directories listed in the PATH environment variable	The default is the default icon of each user's computer

Property	Description	Parameter
ICONID	The numeric ID (if necessary) of the icon within the icon file. If the ICONID is not specified, the default icon is used.	The default is the default icon
IS_CONTAINER	Identifies the application as a container.	Not a container (default)
IS_DISABLED	Identifies the application as disabled. Note: If the application is disabled, users cannot log on to it using CA SSO.	Not disabled (default)
IS_HIDDEN	Identifies an application that does not appear on the desktop even for users who can invoke it. Note: You may want to hide a master application, whose only purpose is to supply passwords to other applications.	Not hidden (default)
IS_SENSITIVE	Identifies whether the user is required to re-authenticate when they open the application again after a preset time.	Not sensitive (default)

Property	Description	Parameter
LOGIN_TYPE	How user passwords are to be provided.	■ pwd-Plain password (default) ■ none-No password required. ■ appticket or passticket-Generated by the CA SSO Server.
MASTER_APPL	The object name of the application supplying the password.	No master (default)
OWNER	The CA Access Control user or group that owns the record.	Blank The creator of the record (default)
PWD_AUTOGEN	Indicates whether the application's password is automatically generated by the CA SSO Server upon password expiration.	No automatic password generation (default)
PWD_SYNC	Indicates whether the application's password can be identical to the user's other application passwords.	No sync (default)

Property	Description	Parameter
PWPOLICY	The name of the password policy object for the application.	The default is no validity checks
RAUDIT	Determines whether to perform auditing on the resource.	■ ALL-Audits all access requests. ■ SUCCESS-Audits all granted access requests. ■ FAILURE-Audits only denied access requests. ■ NONE-Audits no access requests.
SCRIPT_POSTCMD	One or more commands to be executed after the logon script.	
SCRIPT_PRECMD	One or more commands to be executed before the logon script.	
UACC	The default access for the resource, which specifies the access granted to users who are not defined to CA SSO or who do not appear in the resource's access control list. Refer to the ACL property for a list of valid values.	■ EXECUTE or [A]LL-Permission to invoke the application. ■ [N]one-(default) No permission.

Property	Description	Parameter
UPDATE_TIME	The time the record was updated.	Cannot be modified
UPDATE_WHO	The user or group who made the last update.	Cannot be modified
WARNING	Indicates whether to operate in warning mode. In warning mode, all access requests are granted. Instead of preventing an unauthorized user from accessing an application, a record is written to the audit log.	■ Off (default) ■ On

The Application Group Class (GAPPL)

Each object in the GAPPL class defines a group of applications. Application groups are useful when defining access rules, since you can allow or deny access to a group of applications instead of specifying the same access rule for many applications.

Group properties override application properties.

Create an object for each application in the APPL application class before linking specific applications to the relevant group in the GAPPL class.

For example, suppose there is a user group called **ordersdept**, which represents the users in the Orders Department. The users in the Orders Department need three different CICS applications to perform their jobs. To create the appropriate records in the policy data store, you must:

1. Define a record for each of the three CICS applications in class APPL.
2. Define a GAPPL record called **orderapps** and link the three CICS applications to **orderapps**.
3. Allow the group **ordersdept** access to the application group **orderapps**.

Now, all of the users in the group **ordersdept** are allowed to use the three CICS applications.

Each application group is represented by an object in the GAPPL class. Each object in the GAPPL class contains the following properties.

Property	Description	Parameter
AZNACL	Authorization rules that specify who is allowed to access the application, and their access types	
COMMENT	A remark; not used during authorization.	
CREATE_TIME	The time the object was created.	Cannot be modified
MEMBERS	A list of applications that belong to the group.	
OWNER	The CA Access Control user or group that owns the record.	The default is the creator of the record

Property	Description	Parameter
RAUDIT	Determines whether to perform auditing on the resource.	■ **ALL**—Audits all access requests whether successful or not. ■ **ALLOW**—Audits all granted access requests. ■ **DENY**—Audits only denied access requests. ■ **NONE**—Audits no access requests.
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who made the last update.	Cannot be modified

The Authentication Host Class (AUTHHOST)

An authentication host is a host (server) that performs primary authentication of CA SSO users. You determine which hosts can authenticate which users by adding an object to the AUTHHOST class in the policy data store.

Each authentication host is represented by an object in the AUTHHOST class. The name of an AUTHHOST object can be any logical name.

Each record in the AUTHHOST class contains the following properties.

Property	Description	Parameter
AUTH_PARAMETERS	A multi-value field in the key=val format that contains advanced authentication information. The parameter names needed for each authentication provider are taken from the AUTHPROVIDER class.	
AZNACL	Authorization rules that specifies who is allowed to log on using the authentication host.	
AUTH_METHOD	The AUTHHOST object's method ID.	
COMMENT	A remark; not used during authentication.	

Property	Description	Parameter
CONT_FORMAT	A format string that the CA SSO Server uses to manipulate the container's relative distinguished name entered during the authentication process to adjust it to the container name in the user data store.	&container_dn& - the user's container (relative to the user_dir pase path) &empty& - leave empty. &login_name& - the login name (not formatted) &user_name& - the user name (RDN only, stripped of the full DN)
CREATE_TIME	The date and time the record was created.	Cannot be modified
DAYTIME	The day and time restrictions for using the authentication host. Taken from the restrictions parameter used when you define or modify an authentication host.	■ **Blank**—No restrictions (default) ■ mm/dd/yyyy@hh:mm
KEY	The private encryption key of the authentication host. This property is used to allow trust between external authentication agents and the CA SSO Server, using this key as a shared encryption key.	
OWNER	The CA Access Control user or group that owns the record.	The creator of the record (default)
PROPERTIES	Key-value pairs representing all of the AUTHHOST object identification properties. These pairs change from one authentication method to another and are used to identify the AUTHHOST object; for example, HOSTNAME=Machine01.DemoCorp.com PORT=13389.	
RAUDIT	Determines whether to perform auditing on the resource.	■ **ALL**—Audits all access requests whether successful or not. ■ **ALLOW**—Audits all granted access requests. ■ **DENY**—Audits only denied access requests. ■ **NONE**—Audits no access requests.
SERNUM	The serial number of the authentication host. This is used by the Novel authentication agent to provide more security by adding another authhost identification information	

Property	Description	Parameter
UACC	The default access for the resource, which specifies the access granted to accessors who are not defined to CA SSO or who do not appear in the resource's access control list. Refer to the ACL property for a list of valid values.	■ **READ** or **ALL**—Permission to use the host. You can use the abbreviation A. ■ **None**—No permission. You can use the abbreviation N. The default is the value of the AUTHHOST field in the DEFACCESS class.
UPDATE_TIME	The date and time the record was last modified.	Cannot be modified
UPDATE_WHO	The user who last updated the record.	Cannot be modified
USER_DIR_PROP	The name of the user data store object where the user is defined.	Cannot be modified
USER_FORMAT	A format string that the CA SSO Server uses to manipulate the user name entered during the authentication process to make it match the user name in the user data store. The CA SSO Server replaces every occurrence of the value in the fields &user_name& and &user_dir& with the user name and the USER_DIR name, respectively.	&container_dn& - the user's container (relative to the user_dir pase path) &empty& - leave empty. &login_name& - the login name (not formatted) &user_name& - the user name (RDN only, stripped of the full DN)
WARNING	Indicates whether to operate in warning mode. In warning mode, all access requests are granted. Instead of preventing an unauthorized user from accessing an application, a record is written to the audit log.	■ **Off**—(default) Only authorized access requests are granted ■ **On**—All access requests are granted, and unauthorized access is logged

The Authentication Host Group Class (GAUTHHOST)

Every authentication host group object defines a group of authentication hosts. Instead of specifying the access rule for each authentication host, you can allow or deny access to the group.

You must create an object for every host in the AUTHHOST authentication host class, and then link specific hosts to the relevant group in the GAUTHHOST class.

For example, suppose the name of the user group representing the Accounting Department is acctgrp. There are twenty users in this group who use three different NetWare servers for primary authentication. To create the appropriate records in the policy data store, you must:

1. Create one AUTHHOST object for each of the three authentication hosts in the AUTHHOST class.
2. Create a GAUTHHOST object named **accthosts**.
3. Add the three AUTHHOST objects to the GAUTHHOST object.
4. Allow the group **acctgrp** access to the authentication host group **accthosts**.

Now, all of the users who belong to the group **acctgrp** are allowed to authenticate to any of the servers that belong to **accthosts**.

Each authentication host group is represented by a record in the GAUTHHOST class. Each record in the GAUTHHOST class contains the following properties:

Property	Description	Parameter
AZNACL	Authorization rules that specifies who is allowed to use the authentication hosts. Values are determined by the authorize and authorize- selang commands.	
COMMENT	A remark, not used during authentication.	
CREATE_TIME	The date and time this object was created.	Cannot be modified
MEMBERS	A list of the authentication hosts that belong to the group.	

Property	Description	Parameter
OWNER	The name of a CA Access Control user or group that owns the record.	The creator of the record (default)
RAUDIT	Determines whether to perform auditing on the resource.	■ **ALL**—Audits all access requests whether successful or not. ■ **ALLOW**—Audits all granted access requests. ■ **DENY**—Audits only denied access requests. ■ **NONE**—Audits no access requests.
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who last updated the record.	Cannot be modified

The Password Policy Class (PWPOLICY)

Each password policy is represented by an object in the PWPOLICY class. Each record in the PWPOLICY class contains the following properties.

Property	Description	Parameter
APPLS	The list of applications that are linked to the password policy.	
COMMENT	A remark, not used during authentication.	String
CREATE_TIME	The date and time the record was created.	Cannot be modified
OWNER	The name of the CA Access Control user or group that owns the record.	The default is the creator of the record
PASSWORDRULES	For information about password rules, see the “Managing Passwords” chapter.	The default depends on the platform
UPDATE_TIME	The date and time the record was updated.	Cannot be modified

Property	Description	Parameter
UPDATE_WHO	The name of the user or group who last updated the record.	Cannot be modified

The Resource Description Class (RESOURCE_DESC)

The RESOURCE_DESC class represents the access names for the objects of user-defined classes.

Note: You cannot create a new object in the RESOURCE_DESC class; you can only modify the existing ones.

Each object in the RESOURCE_DESC class contains the following properties.

Property	Description	Parameter
CLASS_RIGHT1 to CLASS_RIGHT32	Identifies 32 optional access rights. The defaults for the first four rights are: ■ CLASS_RIGHT1 Read ■ CLASS_RIGHT2 Write ■ CLASS_RIGHT3 Execute ■ CLASS_RIGHT4 Rename	
COMMENT	A remark, not used during authentication.	
CREATE_TIME	The date and time the record was created.	Cannot be modified
OWNER	The name of the user or group that owns the record. This user, or anyone with administrator privileges in the group that owns the record, can manage the record.	The creator of the record (default)
RESPONSE_LIST	The list of responses associated with the user-defined class.	Cannot be modified
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who last updated the record.	Cannot be modified

The Response Class (RESPONSE_TAB)

A response is a personalized answer that is returned to the application after an authorization request is granted or denied. It consists of KEY=VALUE pairs that are understood by the specific application. The response provides the ability to personalize the portal site according to the user's specific needs and authorization permissions.

The RESPONSE_TAB class contains responses to different authorization decisions. A response can be created only for resource classes.

Each record in the RESPONSE_TAB class contains the following properties.

Property	Description	Parameter
CLASS_RIGHT1 to CLASS_RIGHT32	The response to the access right that points to the first to nth place in the Access Control bitmap.	
COMMENT	A remark, not used during authentication.	String
OF_RESOURCE	The name of the resource class that RESPONSE_TABLE belongs to.	Cannot be modified
CREATE_TIME	The date and time the record was created.	Cannot be modified
OWNER	■ The name of the user or group that owns the record. This user, or anyone with administrator privileges in the group that owns the record, can manage the record.	The creator of the record (default)
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who last updated the record.	Cannot be modified

The User Attribute Class (USER_ATTR)

The USER_ATTR class contains all of the valid user attributes for each user directory. You can grant access to a particular resource by using the user attribute. For example, you can set an access rule that allows only managers (where **manager** is the value of a title attribute) to access a certain application.

Valid user attributes consist of predefined user attribute objects and other objects you have defined based on your needs. The predefined objects are:

- User name
- Group name

Use the format name@user_dir for the name of the object. Replace name with the attribute name and user_dir with the user's directory name.

Property	Description	Parameter
ATTRNAME	The name of the attribute.	Cannot be modified
ATTR_PREDEFS	The list of allowed values for a specific attribute.	
COMMENT	A remark, not used during authentication.	
CREATE_TIME	The date and time the record was created.	Cannot be modified
DBFIELD	The name of the field in the user data store database (i.e. field of user class in the ldap directory). Note that different databases can contain different attributes fields or fields are called differently.	
FIELDID	For internal use only; the internal number of the USER_DIR object attribute in the mapping table managed by the CA SSO Server.	Cannot be modified
OWNER	The name of the CA Access Control user or group that owns the record.	The creator of the record (default)
UPDATE_TIME	The date and time the record was updated.	Cannot be modified
UPDATE_WHO	The name of the user or group who last updated the record.	Cannot be modified

Property	Description	Parameter
USERATTR_FLAGS	Contains information about the attribute.	■ aznchk-Indicates that this attribute will be used for authorization. ■ predef or freetext - specified the type of data expected to be set to this attribute. predef - a indicated the existence of predefined values. freetext - indicate that information cal also be entered manually by the user. userdir -Internal flag; associated with the field id property. ■ user or group-Whether the attribute (accessor) is a user or a group
USER_DIR_PROP	The name of the user data store (user_dir) object this user_attr is associated with.	Cannot be modified

The User Directory Class (USER_DIR)

The USER_DIR class contains information about the user data store. CA SSO can work with different types of user data stores.

Each record in the USER_DIR class contains the following properties.

Property	Description	Parameter
ADMIN_NAME	The name of the directory's administrator. Admin name and Admin password are the logon credentials of the administrator of the directory.	
ADMIN_PWD	The password of the directory's administrator. Admin name and Admin password are the logon credentials of the administrator of the directory.	
COMMENT	A remark, not used during authentication.	

Property	Description	Parameter
CONTOBJ_CLS	A list of the object classes that are containers.	Object classes that are wrapped in <> will be used for search operations. Object classes that are not wrapped in <> will be used for create operations. ! is used as not **Note:** This is the only property where the search object classes are ORed together in the search query - a container is an object that is one of the search object classes. (organization OR organizationUnit OR eTssoContainer etc.)
CREATE_TIME	The date and time the record was created.	Cannot be modified
DIR_TYPE	The type of user directory.	■ **ETRUST_AC**—(default) CA Access Control ■ **LDAP**—Any LDAP-enabled directory, including CA Directory ■ **AD**—Microsoft Active Directory ■ **TSS**—CA Top Secret ■ **ACF2**—CA ACF2 ■ RACF
GRPOBJ_CLS	The name of the classes that the group object inherits from. This property is needed for the creation and search of new groups in an LDAP directory.	Object classes that are wrapped in <> will be used for search operations. Object classes that are not wrapped in <> will be used for create operations. ! is used as not Note that the search object classes are ANDed together in the search query - object must have all the object classes names to be considered as a group
LICONTOBJ_CLS	The name of the classes that the logon info container object inherits from. This property is needed when creating new logon info containers in an LDAP directory.	

Property	Description	Parameter
LIOBJ_CLS	The name of the classes that the logon info object inherits from. This property is needed when creating new logon information in an LDAP directory.	Object classes that are wrapped in <> will be used for search operations. Object classes that are not wrapped in <> will be used for create operations. ! is used as not Note that the search object classes are ANDed together in the search query - object must have all the object classes names to be considered as a login info object
LOCATION	Optional. The container where login info objects must be created for this user_dir users. if left empty the login info objects are going to be created in a container called logininfos on the same container where the user resides.	Usually would be set to the following locations: ou=<user_dir>, ou=LoginInfos, o=PS
MAX_RET_ITEMS	The maximum number of items retrieved per query.	
OWNER	The name of the CA Access Control user or group that owns the record.	The creator of the record (default)
PATH	The distinguished name of the container in the LDAP tree where all queries begin and all containers are relative to it.	
PORT_NUM	The TCP/IP port number used to connect to the user directory.	
RAUDIT	Determines whether to perform auditing on the resource.	■ **ALL**—Audits all access requests whether successful or not. ■ **ALLOW**—Audits all granted access requests. ■ **DENY**—Audits only denied access requests. ■ **NONE**—Audits no access requests.
TIMEOUT_CON	The number of seconds that CA SSO must wait for a response from the user directory.	

Property	Description	Parameter
UACC	The default access for the resource, which specifies the access granted to accessors who are not defined to CA SSO or who do not appear in the resource's access control list. Refer to the ACL property for a list of valid values.	■ **READ** or **ALL**—Permission to use the host. You can use the abbreviation A. ■ **None**—No permission. You can use the abbreviation N.
UPDATE_TIME	The time the record was last modified.	Cannot be modified
UPDATE_WHO	The name of the user or group who last updated the record	Cannot be modified
USERATTR_LIST	A list of objects in the USER_ATTR class that are associated with this USER_DIR object.	Cannot be modified The default is the list user's attribute
USERDIR_HOST	The DNS name of the host where the user directory resides.	
USROBJ_CLS	The name of the classes that the user object inherits from. This property is needed when creating new users in an LDAP directory.	Object classes that are wrapped in <> will be used for search operations. Object classes that are not wrapped in <> will be used for create operations. ! is used as not Note that the search object classes are ANDed together in the search query - object must have all the object classes names to be considered as a user
VERSION	Used to specified communication protocol version (LDAP version).	3 (default)

Chapter 13: Working with the Token Data Store

This section contains the following topics:

About the Token Data Store

The token data store is an LDAP directory in which the CA SSO Server stores the user's session information. This includes the session ID, client IP address, username, and the last heartbeat time. The token directory solution supports configurations with a large number of CA SSO Servers.

During the authentication process, the CA SSO Server uses the token directory to save token and user information. If an CA SSO Server fails, the backup server can use the token directory failover/replication to fetch token information for unknown tokens.

CA SSO Server Background Processes

The CA SSO Server uses two background processes:

Remove Expired Token

Searches for tokens that have expired and removes them.

Remove Heartbeat Failed (HBF) Tokens

Searches for tokens that have not sent a heartbeat for longer than the grace period, and removes them.

To adjust these settings in the Policy Manager, go to Resources, Configuration Resources, Policy Server Settings.

Keyname	Description	Parameters
Enabled	Enable this background process.	

Keyname	Description	Parameters
IdlePeriod	A cycle means an entire job the background process must do. To ensure the background process will be active on background only, the job is divided to small operations, this key holds the number of seconds the background process would wait between operations.	
Interval	The number of seconds the background process waits, after finishing an entire cycle (entire job), before starting over	
StartTime	The time that the CA SSO Server starts.	Either: ■ The delay (in seconds) between when the CA SSO Server started and when the background process must start. ■ The time of a day it must start (in xxhxx format)

The eTssoConnectedUser Class

The eTssoConnectedUser class of the token directory used to represent a user identity. A user can have more than one session with the CA SSO Server, thus every eTssoConnectedUser may have more than one eTssoSession.

This means that the user can have one ConnectedUser object but more than one concurrent session. This class contains the following fields:

Property	Description	Parameter
commonName	The name of the object.	The connected user will be created in a container corresponding to the container where the user object lives. With the name of the object attached to the object name. That is, cn=username:cn, ou=develop:ou, ou=users:cn, ou=<userDB_name>
eTssoTokenId	The TokenId field is a multi valued field that lists all the user's tokens (sessions) names, as a reference to their objects	

The following information is calculated by merging the settings taken from the SessionProfiles granted to the user by Authorization. This information is stored per ConnectedUser identity and can be refreshed only by recreating the ConnectedUser object. To cause this, a user must log off all of their active sessions (the registered ones).

Property	Description	Parameter
eTssoScreenLockTimeout	The timeout (on idle) used before activating screen lock.	Integer (minutes)
eTssoLogoutTimeout	The timeout (on idle) used before logging a user out.	Integer (minutes)
eTssoMaxUserSessions	The maximum number of concurrent sessions allowed for a user.	Integer
eTssoSessionLimitChoice	A code specifying how CA SSO Server must handle a session limit.	
eTssoHeartbeatFailbehavior	A code specifying how CA SSO Server must handle a situation where a client didn't send an heartbeat for more then the maximum number of grace times.	

The eTssoSession Class

The eTssoSession class represents a session. Every connected user must have at least one session object. This class contains the following fields:

Property	Description	Parameter
commonName	The TokenId in a form of a UUID	
eTssoLastKeepAliveTime	■ The last time that SSO Client sent a heartbeat on behalf of this session	■ **0—**A session-less token, meaning that this session was never registered and must not be checked for a heartbeat ■ **Date—**time_t (sec since 1970)
eTssoWorkstationId	The client's station.	IP address
eTssoWorkstationName	The display name for the SSO Client workstation	String

Property	Description	Parameter
eTssoAuthnMethod	The authentication method used to get the ticket for this session	■ SSO ■ LDAP ■ EAC ■ SecurID ■ NT ■ CERT ■ NTLM
eTssoAuthHost	The object name of the authentication host that performed the authentication for this session.	
eTssoSessionStatus	The status of this session	■ **Active—**the session is alive and valid ■ **DeletePending—**the session was terminated but the user was not notified. The session will stay in this state until the client accesses the server again or the background process cleans it after expiration. ■ Other SSO Server transitional/internal states.
eTssoMissedHeartbeatAllowed	The number of heartbeats that the SSO Server will allow a client to miss. Beyond that, this session would be considered as a Heartbeat Failed session.	Integer
eTssoHeartbeatInterval	This field stores the interval between heartbeats that SSO Server expects the client to send	Integer (seconds)
eTssoHBEncKey	The encryption key used by the SSO Client to encrypt the heartbeat message	
eTssoHBEncKeyLen	The encryption key length used by the SSO Client to encrypt the heartbeat message	Integer
eTssoLoginTime	The authentication time of this session.	**Date—**time_t (sec since 1970)
eTssoClientHost	The SSO Client machine	Machine name or IP address
eTssoClientPort	The port of the SSO Client machine	

Property	Description	Parameter
eTssoClientEncKey	The encryption key used by the SSO Server to connect and communicate with the client for kill sessions When SSO Server wants to kill an SSO Client connection it must encrypt the request with the SSO Client encryption key.	
eTssoClientEncKeyLen	The length of encryption key used by the SSO Server to connect and communicate with the SSO Client.	Integer (characters)
eTssoUserEName	The name of the ConnectedUser object. The user name container and user dir are combined to create a unique distinguished name. This naming convention is called Ename or Enterprise Name.	String The Ename format is: *<userDB_name>*://Cont:*<container_dn>*_ Uid:*<user_name>* (where the container_dn and the user name have _ instead of every =
eTssoUserDirName	The name of the user data store that contains the user's data	String
eTssoContainerDN	The distinguished name of the user's container	String
eTssoUserName	The name of the user	String
eTssoSecondaryStation	Lists the IP addresses of all the station that the SSO Client used this session in.	Multi-valued

Chapter 14: Maintenance

This section contains the following topics:

SSO Server

You can stop and start the SSO Server on Windows using Windows Services. You access Windows Services from the Start menu by selecting Control Panel, Administrative Tools, Services. The SSO Server is listed amongst the other services as CA SSO Server.

You can also start and stop the SSO Server using the command prompt.

Note: You must be a user with administrative privileges to run these commands, for example, in the Administrative Users group on Windows, or a user with root privileges on UNIX.

Start the SSO Server

To start the SSO Server on Windows

1. Open a command line prompt.
2. Type:

```
net start ssod
```

To start the SSO Server on UNIX

1. Navigate to <install path for SSO Server>/bin
2. Type:

```
./PolicyServer -start
```

Stop the SSO Server

For more information about stopping the SSO Server, see the Scheduling Maintenance Tasks chapter in the *CA SSO Implementation Guide.*

To stop the SSO Server on Windows

1. Open a command line prompt.
2. Type the following:

   ```
   net stop ssod
   ```

To stop the SSO Server on UNIX

1. Navigate to <install path for SSO Server>/bin
2. Type the following:

   ```
   ./PolicyServer -stop
   ```

Check the Status of the SSO Server

To check the status of the SSO Server on Windows

1. Go to the Windows Services tool available from the Windows Start menu.
2. Find the CA SSO Server service, and check the status in the third column.

To check the status of the SSO Server on UNIX

- Use the following command

  ```
  ps –aef | grep PolicyServer
  ```

CA Access Control

You can stop and start CA Access Control on Windows using Windows Services. You can access Windows Services from the Start menu by selecting Control Panel, Administrative Tools, Services.

You can also start and stop CA Access Control using the command prompt. See below for further details.

Note: You must be a user with administrative privileges to run these commands, for example, in the Administrative Users group on Windows, or a user with root privileges on UNIX.

Start CA Access Control

To start CA Access Control on Windows

1. Open a command line prompt.
2. Type: seosd –start

To start CA Access Control on UNIX

1. Navigate to <install path for CA Access Control>/bin
2. Type: ./seload

Stop CA Access Control

To stop CA Access Control on Windows

1. Open a command line prompt.
2. Type: secons –s

To stop CA Access Control on UNIX

1. Navigate to <install path for CA Access Control>/bin
2. Type: ./secons –s

Check the Status of CA Access Control

To check the status of CA Access Control on Windows

1. From the Start menu, select Control Panel, Administrative Tools, Services.
2. Find the CA Access Control services, and check the status in the third column.

To check the status of CA Access Control on UNIX

- Use the ps command, for example:

```
ps –aef | grep seosd
ps –aef | grep seagent
ps –aef | grep seoswd
```

CA Directory

You can stop and start CA Directory on Windows using Windows Services. You can access Windows Services from the Start menu by selecting Control Panel, Administrative Tools, Services.

You can also start and stop CA Directory using the command prompt. See below for further details.

Before You Stop or Start CA Directory

You must:

- Be a DSA user to perform these functions on UNIX. You can log in as a DSA user by typing su – dsa
- Stop the CA SSO Server service before you stop CA Directory.

Start CA Directory

You can stop CA Directory on both UNIX and Windows using the same command. On UNIX you must be logged in at the DSA user.

You do not have to be in a particular directory to run these commands.

To start CA Directory on Windows or UNIX

1. Open a command line prompt
2. To start a specific DSA, type: dxserver start <dsa_name>

   To start all DSAs, type: dxserver start all

Stop CA Directory

You can stop CA Directory on both UNIX and Windows using the same command. On UNIX you must be logged in at the DSA user.

To stop CA Directory on Windows or UNIX

1. Open the /opt/CA/Directory/dxserver/bin folder and login as DSA user as follows:

   ```
   su -dsa
   ```

2. To stop a specific DSA, type: dxserver stop <dsa_name>

   To stop all DSAs, type: dxserver stop all

Check the Status of CA Directory

To check the status of CA Directory

1. Open a command line prompt
2. Type: dxserver status

Authentication Agents

Each of the authentication agents can be started either as a Windows service or as a stand-alone executable from the command line.

The command line usage is as follows:

```
methodtga.exe <option>
```

Where “methodtga.exe” is one of the authentication methods, for example, ldaptga.exe, and “<option>” is one of the following:

install [name]

Install the service using the specified instance name. Once it is installed it can be stopped and started using the Windows Service Control Manager or from the command line. The instance name chosen will affect the service name and the names of any associated log files.

For example, to install the certificate authentication agent with the name “foo” type:

```
certtga.exe install foo
```

start [name|all]

Start the named instance of the service or start all installed instances of the service if the key word ‘all’ is used. When running multiple instances of the same authentication agent on the one machine, multiple configuration files may be required.

For example, to start an already installed Windows authentication agent with the name “bar” type:

```
wintga.exe start bar
```

stop [name|all]

Stop the named instance of the service or stop all installed instances of the service if the key word ‘all’ is used.

For example, to stop any installed instances of the LDAP authentication agent type:

```
ldaptga.exe stop all
```

remove [name|all]

Remove the named instance of the service or remove all installed instances of the service if the key word ‘all’ is used. Note that instances that are running will not be removed until they are stopped.

For example, to remove any installed instances of the RSA authentication agent type:

```
rsatga.exe remove all
```

status [name|all]

Display the status of the named instance of the service or the status of all installed instances of the service if the key word ‘all’ is used.

For example, to display the current status of any installed instances of the LDAP authentication agent type:

```
ldaptga.exe status all
```

standalone

Run an authentication agent as a standalone executable. This option is primarily intended for testing purposes.

For example, to run the certificate authentication agent in standalone mode type:

```
certtga.exe standalone
```

SSO Watchdog Service

The SSO Watchdog service helps you monitor the status of the CA SSO Server. When you start the Watchdog it runs constantly in the background and if it detects that the CA SSO Server is no longer responding, it triggers a reboot of the CA SSO Server.

You can configure the Watchdog to automatically:

- Send you a message before it reboots the CA SSO Server
- Perform commands before or after it reboots the CA SSO Server

You can also manually check the status of the CA SSO Server by connecting to the Watchdog using a web browser.

Start the Watchdog

The Watchdog is installed as part of the SSO Server. You must start the Watchdog service manually.

To start the Watchdog on Windows

1. From the Start menu, select Control Panel, Administrative Tools, Services
2. Double-click on the Watchdog Service and click Start.

To start the Watchdog on UNIX

Navigate to the /opt/CA/SingleSign-On/Server/bin folder and run the following commands:

```
./PSWD –start
```

Note: If you cannot start the Watchdog, it may be disabled. To change this, you must change the Watchdog settings.

Stop the Watchdog

The SSO Watchdog is installed as a service that needs to be manually stopped and started.

To stop the SSO Watchdog on Windows

1. From the Start menu, select Control Panel, Administrative Tools, Services
2. Double-click on the Watchdog Service and click Stop.

To stop the SSO Watchdog on UNIX

Navigate to the /opt/CA/SingleSign-On/Server/bin folder and run the following commands:

```
./PSWD –stop
```

Check the CA SSO Server Using the Watchdog

The Watchdog service reports on the status of the CA SSO Server. To check on the CA SSO Server status, connect to the Watchdog service using a browser window to access the CA SSO Server computer on port 13391. You can also write an application that connects to this service directly - in this case you must use the port number 13392.

To check the CA SSO Server status using a browser interface

1. Ensure the Watchdog service is running.
2. Open a web browser and type the following address:

   `http://<<SSOServer>>:13391/default`

   The page will return a status message, such as: SYSTEM TEST SUCCESS.

Change the Watchdog Settings

The Watchdog settings are stored in two locations, bg.cia and ssod. Each location contains different settings:

The bg.cia settings control:

- WatchDog enabled/disabled
- Number of connection failures before the WatchDog reboots the server
- How often the Watchdog checks the SSO Server

The ssod settings control:

- Connections between the SSO Server and the WatchDog
- Connections between the Watchdog and the client you use to monitor it
- Notification messages sent to administrators

- Automatic commands to run before or after a server reboot
- Times between connection attempts
- Log file details
- Port numbers

To change Watchdog settings on Windows

1. Navigate to the following Windows registry on the SSO Server computer.

   HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server
2. Open either ssod or bg.cia sections and edit the settings.

To change Watchdog settings on UNIX

1. Navigate to the following directory on the SSO Server computer.

   `/opt/CA/SingleSign-On/Server/policyserver.ini`
2. Edit either ssod or bg.cia settings and save changes.

To change Watchdog trace

1. Navigate to the following directory on the SSO Server computer.

   `/opt/CA/SingleSign-On/Server/pswd.ini`
2. Uncomment the following line and save changes:

   `MST Trace`

Change the Watchdog Port Number

You may need to change the Watchdog port number if you have a port number conflict.

To change the Watchdog port number on Windows

1. Navigate to the following registry key on the SSO Server computer.

   HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server\ssod
2. Change the number of the port setting.
   - For http requests use WDHTTPort
   - For direct requests use WDTCPPort

To change the Watchdog port number on UNIX

1. Navigate to the Policyserver.ini file on the SSO Server computer.

   `/opt/CA/SingleSign-On/Server/policyserver.ini`

2. Change the number of port setting.
   - For http requests use WDHTTPort
   - For direct requests use WDTCPPort

Change Watchdog Checking Frequency

You may need to change how often the Watchdog checks the status of the SSO Server. By default this is set to every 10 seconds. The main reasons you might do this are to:

- Reduce the load on the SSO Server by reducing the frequency of checks
- Reduce the time it takes to identify a problem by increasing the frequency of checks

To change the frequency of the Watchdog checks on Windows

1. Navigate to the following Windows registry key on the SSO Server computer.

   HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server\bg.CIA

2. Edit the Interval setting.
3. Change the time in seconds.

To change the frequency of the Watchdog checks on UNIX

1. Navigate to the Policyserver.ini file on the SSO Server computer.

   `/opt/CA/SingleSign-On/Server/policyserver.ini`

2. Find the [bg.CIA] section and edit the Interval setting.
3. Change the time in seconds.

Change the Watchdog User Password

The pswd-pers user password is kept in the pswd.dat file on the CA SSO Server.

To change the Watchdog user password

1. Navigate to the bin directory.
2. Type the following command:

   ```
   PSWD –s <<username>> <<password>>
   ```

Note: You must update Access Control datastore using the Policy Manager as well. The user that Watchdog Service uses must be given the EAC authentication method.

Change the Watchdog Failure or Success Message

You may want to change the message the Watchdog sends to the listening client when there is a problem with the SSO Server. This message might include a specific instruction for the administrator. By default these messages are "System test success" and "System test failure".

To change the Watchdog failure or success message on Windows

1. Navigate to the following registry key on the SSO Server computer.

   HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server\ssod

2. Edit the WDOnSuccessString or WDOnFailureString setting.
3. Specify the message.

To change the Watchdog failure or success message on UNIX

1. Navigate to the Policyserver.ini file on the SSO Server computer.

   ```
   /opt/CA/SingleSign-On/Server/policyserver.ini
   ```

2. Edit the WDOnSuccessString or WDOnFailureString setting.
3. Specify the message.

Run a Command before the Watchdog Reboots the SSO Server

You can configure the Watchdog to run a command before it triggers a reboot of the SSO Server, if the Watchdog detects that the SSO Server has stopped running. You might choose to run a command that backs up information, switches routing to another server or notifies an administrator.

To configure a command to be run before an automatic server reboot on Windows

1. Navigate to the following Windows registry key on the SSO Server computer.

   HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\Server\ssod

2. Edit the WDRestartNotificationCommand setting.
3. Specify the command.

To configure a command to be run before a automatic server reboot on UNIX

1. Navigate to the Policyserver.ini file on the SSO Server computer.

   `/opt/CA/SingleSign-On/Server/policyserver.ini`

2. Edit the WDRestartNotificationCommand setting.
3. Specify the command.

Obfuscation Tool

The Obfuscation Tool called, ssoencconf, is a command line tool designed to obfuscate sensitive configuration information stored in the Windows registry or ini files for various SSO components. It is automatically installed with each SSO component that requires it. It is always installed in the 'bin' directory of the component. You will use this tool periodically to maintain sensitive configuration information.

You can use this tool to obscure text in a plain text file to keep it secure, such as password information.

For example, if the sample PKCS#12 identity file installed for the Windows Authentication Agent is replaced by a custom PKCS#12 file, the password for this new file must also be configured in the appropriate .ini file. The value of the "IdentityPassword" setting in the ini file is not the plaintext PKCS#12 file's password, but the obfuscated version of this which is output by the ssoencconf tool.

ssoencconf Command - Obscure Sensitive Information in Text Files

Certain sensitive information must be obfuscated to make it secure. This sensitive information must never be visible in plain text. Use the ssoencconf command to obscure this information stored in either the Windows Registry or INI files. This utility is installed in the bin folder for each component that requires information to be obfuscated.

You can either direct the obfuscation tool to a specific INI file or Windows Registry Key to perform the obfuscation in the correct location, or you can run it in the current command window so that you can cut and paste the encrypted output manually.

Note: You must run the ssoenconf commands on the same operating system as the item you want to obfuscate. The obfuscation tool will create a different encryption for the same word if they are created on different platforms.

This command has the following format:

`ssoencconf [-r registry-key|-i path-and-file] [-v name] -d data`

-r registry-key

Specifies the registry key that contains the sensitive information. You cannot use this in conjunction with -i.

-i path-and-file

Specifies the configuration file (and location) which contains the sensitive information. You cannot use this in conjunction with -r.

-v name

Specifies the name of the sensitive configuration item.

-d data

Defines the data that you want to obscure, such as a password. To obscure a blank field, use "".

Example: Obfuscate a password in a Windows Registry key

This example shows how to replace a password in the Windows registry with the encryption of the string "newpassword" to make it more secure.

```
ssoencconf -r
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\PSA\PolicyServer -v
AdminPassword -d newpassword
```

Example: Obfuscate a password in an INI file

This example shows how to replace a password in an INI file with the encryption of the string “newpassword” to make it more secure.

```
ssoencconf -i "C:\Program Files\CA\ Single Sign-On\Client\cfg\Auth.ini" -v
IdentityPassword -d newpassword
```

```
Example: Obfuscate a blank password in an INI file
```

This example shows how to replace a password field in the INI file with the encryption of a blank string.

```
ssoencconf -i "C:\Program Files\CA\Single Sign-On\Client\cfg\Auth.ini" -v
IdentityPassword -d ""
```

Example: Obfuscate a word to your current command window

This example shows how to encrypt the string “newpassword” and show the result in your current command window. You might do this if you want to cut and paste the encrypted output manually.

```
ssoencconf -d newpassword
```

Chapter 15: Auditing and Logging

This section contains the following topics:

About Auditing and Logging

Auditing and logging is useful for:

- Verifying correct installation and operation of CA SSO
- Diagnosing problems with CA SSO
- Reporting on CA SSO events

Note: You must only use logging for troubleshooting. If you leave logging on all the time, it will slow the system down, and use up disk space.

Installation Logging for Windows MSI Installers

To set up logging for installations on MSI installers, you must use the Windows Registry on the target computer.

If you enable logging for Windows installations, four log files are created for each CA SSO component that is installed. Each of the four files contains a different level of logging.

To enable logging for installations on Windows

1. Open the Registry Editor.
2. Navigate to the following registry key:

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer
   Reg_SZ: Logging
   ```

   If you do not have a Logging string value, create one by right-clicking and selecting New, String Value and naming it Logging.

3. Double-click on the Logging file.

   The Edit String dialog appears.

4. Type in the letter(s) corresponding to the logging mode you want and click OK.

5. To check the results of the logging to find out where your Temp directory is, go to the command line and type echo %temp%.

   Note: This applies to the following installers: AD Listener and Policy Manager.

Logging Mode Settings

The letters in the value field can be in any order. Each letter turns on a different logging mode. These values apply to MSI version 1.1 and above:

Logging Mode	Description
v	Verbose output
o	Out-of-disk-space messages
i	Status messages
c	Initial UI parameters
e	All error messages
w	Non-fatal warnings
a	Start up of actions
r	Action-specific records
m	Out-of-memory or fatal exit information
u	User requests
p	Terminal properties
+	Append to existing file
!	Flush each line to the log
"*"	Wildcard, log all information except for the v option. To include the v option, specify "/l*v".

Installation Logging for (ISMP) Installers

To set up logging for installations on Install Shield Multi Platform (ISMP) installers, you must use the following command line:

`-log # !{filename} {parameters}`

where

#

Specifies that logging information is echoed to the standard output.

filename

Defines the file name for the generated log output. If ! is specified without a file name, then the default log file is used.

parameters

Defines the list of log event types to be logged. You can log:

- Log all events (@ALL)
- Error events (@err)
- Warning events (@wrn)
- Primary events (@msg1)
- Secondary events (@msg2)
- Debug events (@dbg)
- Disable all logging (@NONE)

Note: This applies to the following installers: CA SSO Client, CA SSO Server, Session Administrator, SSO Documentation, LDAP Authentication Agent, Certificate Authentication Agent, Windows Authentication Agent, RSA Authentication Agent and Password Synchronization Agent.

Example: Installation Logging for ISMP Installers on UNIX

In most UNIX shells, the semicolon character (";") has a specific meaning to the OS, and results in an error when used on a command line. This problem can be avoided if the option is surrounded in single quotes (') or another escape character.

To log all warning and primary events to the file mylog.txt, type the following:

```
./install.bin -log '!/home/myprojects/mylog.txt' '@wrn;msg1'
```

Note: This command line option can have a maximum of three arguments.

Example: Installation Logging for ISMP Installers on Windows

To log all event types to the file logfile.txt, type the following:

```
setup.exe -log !logfile @ALL
```

Note: This command line option can have a maximum of three arguments.

Logging for CA SSO Components

CA SSO log files record:

- Date, time and level of each eligible event

In addition, the CA SSO Client and Server record:

- The IP address of the machine (if applicable) that initiated the request
- The user's details, such as DN and user data store name

Configure Logging

By default, logging is enabled for all CA SSO components. You can configure logging options including output location and level of logging using the component's logging configuration file. You can configure logging options for:

- CA SSO Client
- CA SSO Server
- Session Administrator
- GINA
- SSO Interpreter
- Authentication agents

Levels of Logging

There are five levels of severity that can be logged for all CA SSO components. Each level includes the logging of the level below it, as well as extra information:

TRACE

Logs entry and exit functions to provide a record of program execution.

DEBUG

Logs information used by CA SSO developers to track problems.

INFO

Logs information about the normal operation of the client.

WARN

Logs warning level events that generally indicate that something unexpected has happened, however the CA SSO Client is able to continue.

ERROR

Logs error level events that indicate a serious problem has occurred and the operation cannot be continued, however the CA SSO Client is able to keep running.

CA SSO Client Log Files

For the CA SSO Client, the logging.properties file specifies the location of the logging configuration file. By default, the logs are created in the log folder within the installation directory. You can change this location by editing the *log4cplus.appender.sso.File* line in the logging.properties file.

For CA SSO Client logging to be successful, the user must have write access to the area that the log file is being written to.

CA SSO Server Log Files

CA SSO Server logging is enabled by default post-installation.

You can configure log file output location and level of logging on Windows by using the pslog.ini file, which is installed in the installation directory.

The pslog.ini file contains the following lines:

Output

The output source, for example, log files, Windows Event Viewer or CA Audit.

Filename

The name of the file.

MaxFileSize

The maximum size of the file before it rolls-over to a new log.. When the log file reaches its maximum size, it is copied to a new file named <log>.000, and a new empty <log>.log file is created. This process is repeated each time the log file reaches its maximum size. When a new file is created, the previous <log.log> is renamed <log>.000 and the previous <log>.000 is renamed one increment number higher to create a chain. For example, <log>.000, will be renamed <log>.001, <log>.log will be renamed <log>.000 and a new <log>.log will be created.

MaxNumberOfHistFiles

Specifies the maximum number of history files that will be saved. When the maximum number of history files is reached, the oldest file is deleted.

MST_LOG

Logs all user events, such as creating a new user or logging on. This is enabled by default.

MST_TRACE

Logs all events. This is commented out by default, because the log file quickly becomes very large.

Session Management Log files

There are two kinds of log files for the Session Administrator:

- The Session Administrator's communications with the CA SSO Server
- The Session Administrator's inner workings

Also, you can read the logs of the Tomcat server. These logs are written to the %SESSION_ADMIN_INSTALLED_DIR%\logs directory.

Location of the Session Administrator/CA SSO Server Communication Log File

The Session Administrator reports most communication issues (if any) directly into your web browser page, however, if there are some unexpected problems, you can look in the communication log files etWACJavaSDK_C.log and etWACJavaSDK_J.log).

Both files are located in:

```
%SESSION_ADMIN_INSTALLED_DIR%\webapps\SessionAdministrator\log\
```

where

%SESSION_ADMIN_INSTALLED_DIR%

Specifies the installed directory, for example C:\Program Files\CA\Single Sign-On\Session Administrator.

Note: The location of the communication log files is predefined and cannot be changed.

Change the Location of the Session Administrator Log File

Use the following procedure to change the location of the session administrator log file.

To change the log file location

1. Open the following file:

   ```
   %SESSION_ADMIN_INSTALLED_DIR%\log\log4j_config.lcf
   ```

   where

   %SESSION_ADMIN_INSTALLED_DIR%

   Specifies the installed directory, for example C:\Program Files\CA\Single Sign-On\Session Administrator.

2. Find the following line:

   ```
   log4j.appender.session_admin.File=C:/Program Files/CA/Single Sign-On/Session Administrator/webapps/SessionAdministrator/log/SessionAdministrator.log
   ```

3. Change the line to refer to a different file location. For example:

   ```
   log4j.appender.session_admin.File=c:\\mylogdir\\mylogfile.txt
   ```

Password Synchronization Agent Log Files

The Password Synchronization Agent writes to the Windows Event Viewer. This is not configurable.

CA Directory Log Files

CA Directory includes a number of log files, most of which are very detailed. These files are written to the dxserver\logs directory.

For more information about logging for CA Directory, see Monitoring the Directory and Messages and Logs in the CA Directory *Administration Guide*.

Auditing CA SSO Components

Auditing lets you track the everyday events in CA SSO. It lets you examine the events that have occurred in CA SSO for review and assessment of previous and potential threats to or violations of, your environment. With CA SSO you can log significant events at varying levels of detail to several different destinations. You can also filter the output before it is written to control exactly which events you want to track.

The type of audit event produced varies depending on the CA SSO component that produced it. The following sections describe:

- The types of audit events produced by each CA SSO component
- How to interpret the audit events produced
- How to work with the audit output

Audit Events Produced by the CA SSO Server

The audit events produced by the CA SSO Server can be divided into the following categories:

System events

System events include successful or unsuccessful starting of the CA SSO Server, stopping the CA SSO Server, and token cleanup when tokens reach their limit.

Administration events

Administration events include creating and deleting users and assigning a user to a group, and password change requests.

Authentication events

Authentication events include successful and unsuccessful authentication. A successful authentication event looks like this: “User X successfully authenticated using method Y.” A failed authentication event looks like this: “User X failed authentication using the Authentication plug-in Y.”

Application execution

These events will include date, time, username, workstation IP address of a user who has attempted to run an application.

Authorization events

Authorization events indicate whether access to a resource was granted or denied for a user. When access is granted, the event looks like this: “User X from Agent Z was granted A access to resource R of class C.” When access is denied, the event looks like this: “User X from Agent Z was denied A access to resource R of class C.”

Common services

Events produced by common services include the user logging out of the system, an invalid or expired token, or the token exchange in a server farm.

Note: All audit events include time stamp, IP address of a workstation initiating the request (if applicable) and user name details. CA SSO Server audits also support the audit destinations: files, debug monitors, Windows Event Viewer and CA Audit.

Collecting Events and Controlling Output

The CA SSO Server has a configuration file that controls which events are collected and the output format and destination of the collected events. Using the configuration file, you can determine how much information is collected, which components of CA SSO the information is collected from, and where the collected information is saved.

CA SSO Server Auditing Configuration File

The CA SSO Server auditing configuration file pslog.ini contains parameters that CA SSO needs to process log messages generated by the CA SSO Server. The pslog.ini file is stored in the installation path of the CA SSO Server; the default path is:

```
c:\Program Files\CA\Single Sign-On\Server\pslog.ini.
```

Parameters and their values occupy a line in the pslog.ini file in the format:

```
parameter=value
```

Important! Before you start editing the pslog.ini file, stop the CA SSO Server. After you finish editing, start the CA SSO Server.

The lines containing parameters for a particular process or utility are grouped together as a section. The pslog.ini files contain the following sections:

- The Clog Settings section
- The Audit section
- The Log section
- The Trace section
- The filters section

Each section in the pslog.ini file starts with a header line that gives the section name in square brackets.

Parameter values depend on which output destination you are using. Some output destinations may also have additional parameters.

The Clog Settings Section

The [Clog Settings] section is used to gather information about events that occur on the CA SSO Server or identify problems a CA SSO Server may be having. Either situation may require redirecting log messages to a logging utility. This section redirects logs generated by the CA SSO Server to any logging utility. Each parameter for this section is explained next.

MsgFilePath

Contains the path to the directory where the message file is stored.

MsgFileName

Specifies the name of the message file. The default name is wac.msg.

eTAuditDLL

Identifies the path to this DLL file is set during installation.

DBMontiorDLL

Identifies the path to this DLL file is set during installation.

NTEventLogDLL

Identifies the path to this DLL file is set during installation.

LogFileDLL

Identifies the path to this DLL file is set during installation.

The Audit Section

Identifies the destination for audit messages and the name of the file that contains them.

The Log Section

Identifies the destination for log messages and the name of the file that contains them.

The Trace Section

Identifies the destination for trace messages and the name of the file that contains them.

The Filters Section

Identifies which audit events you want to receive. All LOG events are enabled by default. Setting a filter enables the specified events to be written to the destination specified in your configuration file.

Audit Events Produced by the Web Agent

The audit events produced by the Web Agent can be divided into the following categories:

System events

System events include successful or unsuccessful starting of the Web Agent and stopping the Web Agent.

Administration events

Administration events include whether user self-registration succeeded or failed and the refreshing of the cached application list.

Common services

Events produced by common services include the secondary server's request for a user's token and the execution of an external command.

Collecting Events and Controlling Output

The Web Agent component has a configuration file that controls which events are collected and the output format and destination of the collected events. Using the configuration file, you can determine how much information is collected, which components of CA SSO the information is collected from, and where the collected information is saved.

The name of the Web Agent auditing configuration file is webagentlog.ini. It is located in the installation path of its component.

Web Agent Auditing Configuration File

The Web Agent auditing configuration file, called webagentlog.ini, contains parameters that CA SSO needs to process log messages. Parameters and their values occupy a line in the webagentlog.ini file in the format:

```
parameter=value
```

Important! Before you start editing the webagentlog.ini file, stop the web server. After you finish editing, start the web server.

The lines containing parameters for a particular process or utility are grouped together as a section. The webagentlog.ini files contain the following sections:

- The Clog Settings section
- The Audit section
- The Log section
- The Trace section
- The filters section

Each section in the webagentlog.ini file starts with a header line that gives the section name inside square brackets.

The Clog Settings Section

The [Clog Settings] section is used to gather information about events that occur on the CA SSO Server or identify problems a CA SSO Server may be having. Either situation may require redirecting log messages to a logging utility. This section redirects logs generated by the CA SSO Server to any logging utility. Each parameter for this section is explained next.

MsgFilePath

Contains the path to the directory where the message file is stored.

MsgFileName

Specifies the name of the message file. The default name is wac.msg.

eTAuditDLL

Identifies the path to this DLL file is set during installation.

DBMontiorDLL

Identifies the path to this DLL file is set during installation.

NTEventLogDLL

Identifies the path to this DLL file is set during installation.

LogFileDLL

Identifies the path to this DLL file is set during installation.

The Audit Section

Identifies the destination for audit messages and the name of the file that contains them.

The Log Section

Identifies the destination for log messages and the name of the file that contains them.

The Trace Section

Identifies the destination for trace messages and the name of the file that contains them.

The Filters Section

Identifies which audit events you want to receive. All LOG events are enabled by default. Setting a filter enables the specified events to be written to the destination specified in your configuration file.

Audit Events Produced by CA Access Control

CA Access Control logs two types of audit events:

Administration events

An audit record is created for each operation resulting from Policy Manager or selang commands.

SSO events

CA SSO creates audit records for events in which a user has used the CA SSO Client.

Administrative Events

Every event record includes:

- The user issuing the request
- The type of event
- Successful or unsuccessful and, in some cases, details of the event, such as application name (when logon variables are requested)
- Date and time
- The target user (when an administrative request is issued)

CA SSO Events

Every event record includes:

- The user issuing the request
- The type of event
- Successful or unsuccessful and, in some cases, details of the event, such as application name (when logon variables are requested)
- Date and time

There are three types of CA SSO audit records:

- Logon to CA SSO
- Request for an application list
- Request for logon variables

Logon to CA SSO (SSO_LOGIN)

This record reports a logon to SSO event. For example: A user makes SSO authentication, and the audit shows:

```
21 Mar 2004 18:43 P SSO_LOGIN ester Read 0 0 ssod
```

Request for an Application List (SSO_GAL)

This record reports on a user request for an application list. The Get Application List request can be committed by fetching the list from cache, or can be a full calculation of the current application list from the CA SSO Server database.

Examples:

The end user starts the CA SSO Client and gets his cached application list:

```
21 Mar 2004 18:43 P SSO_GAL ester Read 0 0 Cached ssod
```

The end user clicks Refresh button on the CA SSO Tools window, which leads the CA SSO Server to make GAL with full calculation:

```
21 Mar 2004 19:35 P SSO_GAL (ester) Read 0 0 ssod
```

Request for Logon Variables (SSO_GLV)

This record reports on a user request for logon variable. For example: User runs an application LotusNotes:

```
21 Mar 2004 19:35 P SSO_GLV (ester) Read 0 0 LotusNotes
```

Configure Audit Output

CA SSO gives you the ability to configure the audit output that was generated, and then specify where the output must be sent. Audit output consists of three types of messages generated by CA SSO:

Audit messages

This type of message provides the most basic notification, which is that a significant event occurred. Use audit messages for production logging of normal traffic.

Log messages

This type of message gives you more information about each event.

Trace messages

This type of message provides very detailed data about each event. Use trace messages only when debugging a specific problem.

Note: The following sections use audit as the type of message, but the same concepts apply to the log and trace message types.

The following sections explain how to specify a destination for audit output and how to filter events.

Specify an Output Destination

CA SSO supports several destinations where you can have your audit output delivered:

- CA Audit
- Log File
- Event Viewer

You can control the destination of audit output by the value you specify for the Output key in the Audit section of the configuration file. Use the following values to direct audit output to a particular destination:

eTAudit

Directs audit output to CA Audit.

LogFile

Directs audit output to the Log File.

NTEventLog

Directs audit output to the Event Viewer (for Windows only).

For example, to direct audit output to the Log File, specify the following statement in the Audit section of the configuration file:

```
[Audit]
 Output=LogFile
```

Each output destination requires additional keys to be specified in the Audit section of the configuration file. The following sections explain the keys required for each destination.

Configure the Configuration File to Use CA Audit

CA Audit collects enterprise-wide security and system audit information without the reduced performance and overwhelming network traffic caused by other auditing products. It consolidates data from UNIX and Windows NT servers and other CA products and stores it in a central database for easy access and reporting. Administrators use CA Audit for monitoring, alerting, and reporting information about user activity across platforms.

To write events as CA Audit output, you need to have an CA Audit client installed on your network. You also need an CA Audit server installed either on the same machine where CA SSO is installed, or in the network where your CA Audit client can communicate with it. Then, to set CA Audit as the output destination of your audit events, specify eTAudit as the value for the Output key in the Audit section of the configuration file. After specifying eTAudit for the Output key, you need to add the following keys to the Audit section:

Router

The name or IP address of the CA Audit Router. This key is required.

Port

The port number the CA Audit Router is listening to. This key is optional.

Timeout

How long to wait for CA Audit Router to reply before timing out, specified in seconds. This key is optional.

Here is an example of the Audit section that sets CA Audit as the output destination of your auditing events:

```
[Audit]
Output = eTAudit
Router = 123.232.233.232
Port = 8240
Timeout = 5
```

Refer to the *CA Audit Administrator Guide* to set up the communication for the CA Audit client and server. You can then use the CA Audit Security Monitor to view your audit output.

Configure the Configuration File to Use the Log File

To set the Log File as the output destination of your audit events, specify *LogFile* as the value for the Output key in the Audit section of the configuration file. After specifying a value for the Output key, you need to add the following keys to the Audit section:

FileName

The full name of the file (including the path) where you want to store the logs. This key is required. On Windows, the user ps-pers must also have full access to the specified directory.

OpenMode

This key is optional. If enabled (the value equals 1), keeps the destination file open between writes, thus increasing audit performance. The default is 0 (not enabled).

MaxFileSize

This key is optional. Specifies the maximum size of the output file in bytes. The default is 700000 bytes.

MaxNumberOfHistFiles

This key is optional. Specifies the maximum number of history files that will be saved. The default is 5.

MessageFormat

Specifies the format of the output message. This key is optional.

Note: The reporting facility of CA SSO requires a specific message format to correctly interpret audit data. This format is set during the installation of reporting and is described later in this chapter.

You can use the following format variables in your message:

Format Variable	Stands For
%D	Date
%A	Application
%G	Category
%U	User
%T	Time
%C	Component
%L	Level
%M	Message

Here is an example of the Audit section that sets Log File as the output destination of your auditing events:

```
[Audit]
Output = LogFile
FileName = C:\webac\audit.log
OpenMode = 1
MaxFileSize = 2048000
MaxNumberOfHistFiles = 20
MessageFormat = Event was collected on %D %T.\n Message: %M
```

Configure the Configuration File to Use the Event Viewer

The Event Viewer is a tool you can use to monitor events in your Windows system. You can use the Event Viewer to view and manage System, Security, and Application event logs. You can also archive event logs.

Note: The Event Viewer destination is not supported on UNIX platforms.

Note: The event logging service starts automatically when you run Windows. You can stop and start event logging with the Services tool in Control Panel.

To set Event Viewer as the output destination of your audit events, specify NTEventLog as the value for the Output key in the Audit section of the configuration file. There are no extra keys you need to set.

Here is an example of the Audit section that sets the Event Viewer as an output destination of your auditing events:

```
[Audit]
Output = NTEventLog
```

Filter Events

CA SSO lets you filter the audit events you want to receive. Log events are enabled by default. Setting a filter enables the specified events to be written to the destination specified in your configuration file.

You can filter events based on source component, message type, and message level. Each filter is an entry in the Filters section of the configuration file. Each entry consists of three fields in the following format:

[*Source Component*. *Message Type*. *Message Level*]

Source Component is the name of the source component that issued the event. Replace *Source Component* with one of the following values:

WebAgent

To enable Web Agent events.

CA SSO Server

To enable CA SSO Server events.

*****

To enable events from all source components

Message Type is the message type for the event. Replace *Message Type* with one of the following values:

MST_LOG

To enable log events.

MST_TRACE

To enable trace events.

MST_AUDIT

To enable audit events.

*****

To enable events for all message types.

Message Level is the event severity level. Replace *Message Level* with one of the following values:

MSL_INFORMATION

To enable information events.

MSL_WARNINGS

To enable warning events.

MSL_CRITICAL

To enable critical events.

MSL_FATAL

To enable fatal events.

*****

To enable events for all severity levels.

For example, to enable critical audit events from the Web Agent, specify:

```
[filters]
 [WebAgent.MST_AUDIT.MSL_CRITICAL]
```

To enable critical and fatal audit events from all components, specify:

```
[filters]
 [*.MST_AUDIT.MSL_CRITICAL]
 [*.MST_AUDIT.MSL_FATAL]
```

To enable all events, specify:

```
[filters]
 [*.*.*]
```

Audit Tools

A number of audit tools can be used for auditing the events, including CA Audit and seaudit.

Security Audit (seaudit)

The seaudit module is the CA Access Controll audit file viewer. It is a command-line interface that allows you to filter out audit events.

The seaudit module is supplied with CA SSO. For more information about seaudit, see the *CA Access Control Administration Guide*.

CA Audit for CA SSO

CA Audit is an audit collection and alerting tool. This tool can be used to process events and information generated by CA SSO. CA Audit is not supplied with CA SSO. You can purchase it separately.

CA Enterprise Log Manager

CA Enterprise Log Manager is an auditing product that lets you collect, normalize, aggregate, and report on IT activity, and generate alerts requiring action when possible compliance violations occur. This product can be used to process events and information generated by CA SSO. CA Enterprise Log Manager is not a component of CA SSO. You must purchase it separately.

Note: For more information about how to configure CA Enterprise Log Manager to receive events from CA SSO, see the CA Single Sign-on Connector Guide packaged with CA Enterprise Log Manager.

Chapter 16: FIPS 140-2

This section contains the following topics:

FIPS Overview

The Federal Information Processing Standards (FIPS) 140-2 publication is a security standard for the cryptographic libraries and algorithms a product and all its components must use for encryption. Encryption affects the storage and verification of passwords, as well as the communication of all sensitive data between components of CA products and between CA products and third-party products. FIPS 140-2 specifies the requirements for using cryptographic algorithms within a security system protecting sensitive but unclassified data.

CA SSO uses the Advanced Encryption Standard (AES) adapted by the US government. CA SSO incorporates the RSA BSafe and Crypto-C ME v2.0 cryptographic libraries, which have been validated as meeting the FIPS 140-2 Security Requirements for Cryptographic Modules.

Communication

As a part of FIPS 140-2 compliance, all communication between various SSO components is encrypted using TLS/SSL protocol. The following are the SSO components that support this functionality:

- SSO Desktop Sever
- CA SSO Client
- SSO Authentication Agents (WIN, CERT, LDAP and SSO)
- Policy Manager
- PSA
- ADS Listener

Communication Modes

CA SSO is enabled to communicate in the following three modes between its components:

- Compatible
- TLS
- Mixed

For more information on these modes, see Communication Modes in the *CA SSO Implementation Guide*.

Configuring the CA SSO Client in Different Modes

In the CA SSO Client a DWORD registry value called "CommMode" is provided to configure the communication mode. The key is available under the following branch on the corresponding machines:

```
HKLM\Software\ComputerAssociates\SingleSignOn\Client
```

For communication in FIPS-only or dual mode (supports FIPS compatible modes of communication) you must provide the certificate file for the communication. The IdentityFile entry in the auth.ini and client.ini files points to the location of the certificate files:

The values in the client.ini file are related to the communication with the CA SSO Server and the values in auth.ini file are related to the respective authentication agents. The default certificate files are placed in the cfg folder.

Note: If you use custom certificates all instances of these attributes in the auth.ini file and client.ini file have to replaced with the correct path of the certificates.

Configure the CA SSO Client in Compatible Mode

For this mode, the certificate file is used for encryption, mentioned during installation, are required.

To configure the CA SSO Client in compatible mode

1. Set the "CommMode" registry entry to 0 which indicates compatible mode.
2. Exit all the CA SSO client agents such as SSO Tools, status icon, and launch bar.
3. Restart the Client service on CA SSO Client machine.

Configure the CA SSO Client in FIPS-only Mode

For this mode, the certificate file is used for encryption, mentioned during installation is required.

To configure the CA SSO Client in FIPS-only mode

1. Set the "CommMode" registry entry to 1 which indicates FIPS only mode.
2. Set the path of Identity File in the client.ini and auth.ini files.
3. Exit all the CA SSO client agents such as SSO Tools, status icon, and launch bar.
4. Restart the Client service on CA SSO Client machine.

Configure the CA SSO Client in Dual Mode

Dual mode supports FIPS compatible modes of communication. For this mode, the required certificate (in .pem format) must be available in the value %InstallDir%\cfg directory. In the case of Custom certificates, the absolute paths to the Certificate file must be provided as value for IdentityFile in the auth.ini and client.ini files

To configure the CA SSO Client in Dual mode

1. Set the "CommMode registry" entry to 2 which indicates TLS mode.
2. Set the paths of Identity File in the client.ini and auth.ini files.
3. Exit all the CA SSO client agents such as SSO Tools, status icon, and launch bar.
4. Restart the CA SSO Client service on CA SSO Client machine.

   Note: If this configuration is done after installation, you must edit the following entries in the corresponding files %InstallDir%/cfg/Client.ini and Auth.ini.
   `IdentityFile=<FULL_PATH_TO_THE_IDENTITY_FILE_IN_PEM_FORMAT>`

For example:

```
IdentityFile=c:\Program files\CA\Single Sign-on\cfg\sso_rootcert.pem
```

Configuring the CA SSO Authentication Agents in Different Modes

In the CA SSO authentication agents a DWORD registry value called "CommMode" is provided to configure the communication mode. The key is available under the following branch on the corresponding machines:

`HKLM\Software\ComputerAssociates\SingleSignOn\<CA_<AuthMethod>tga>`

For communication in TLS or mixed mode (supports FIPS compatible modes of communication) you must provide the certificate file for the communication. The IdentityFile and PrivateKeyPath entries in the authentication agent configuration file points to the location of the certificate files.

Configure Authentication Agents in Compatible Mode

To configure SSO Auth agents to use compatible mode

1. Set CommMode registry entry to 0.
2. Restart the agent TGA service.

Configure Authentication Agents in TLS Mode

To configure SSO Auth agents to use TLS mode

1. Set CommMode registry entry to 1.
2. Set the paths of IdentityFile and PrivateKeyPath in the authentication agent configuration files.
3. Restart the agent TGA service.

Configure Authentication Agents in Mixed Mode

To configure authentication agents in mixed mode

1. Set CommMode registry entry to 2.
2. Set the paths of IdentityFile and PrivateKeyPath in the authentication agent configuration files.
3. Restart the agent TGA service.

Configuring the CA SSO Server in Different Modes

A new Configuration Property called "CommMode" has been added to the server. "CommMode" contains the configured communication mode value (0,1,2) and the property is located in Access Control. Issue the following command at the selang prompt to show the current value of "CommMode":

```
showRes PSCONFIGPROPERTY CommMode@ssod
```

or

```
sr PSCONFIGPROPERTY CommMode@ssod
```

To change the mode, issue the following command at the selang prompt:

```
er PSCONFIGPROPERTY CommMode@ssod gen_prop(value) gen_val(0 or 1 or  2)
```

Alternately, this can be changed from the Policy Manager by navigating to PolicyServerSettings > Communication and editing the CommMode setting.

The CA SSO Server listens on default port 13980. In FIPS mode, the default SSL port is 13981. In dual mode, the CA SSO Server supports both FIPS and non-FIPS modes of communication. In dual mode both ports are used. The ports configured can be seen using the following two commands:

```
showRes PSCONFIGPROPERTY PortNumber@ssod
showRes PSCONFIGPROPERTY SSLPortNumber@ssod
```

To change the mode, issue the following command at the selang prompt

```
er PSCONFIGPROPERTY PortNumber @ssod gen_prop(value) gen_val(<portnumber>)
er PSCONFIGPROPERTY SSLPortNumber @ssod gen_prop(value) gen_val(<sslportnumber>)
```

Alternately, this can be changed from the Policy Manager by navigating to PolicyServerSettings > Communication and editing the PortNumber and SSLPortnumber settings.

Configure the CA SSO Server in Compatible Mode

To configure the CA SSO Server to run in Compatible mode

1. Set the CommMode property to 0 which indicates Compatible mode. This can be done using one of the following methods:

   - Using selang

     Open Selang command prompt and execute the following command:

     ```
     editres PSCONFIGPROPERTY CommMode@ssod gen_prop(value) gen_val(0)
     ```

     or alternately,

     ```
     er PSCONFIGPROPERTY COmmMode@ssod gen_prop(value) gen_val(0)
     ```

   - Using Policy Manager

     In Policy Manager, go to Communication Settings in Policy Server settings and change the CommMode property to compatible mode.

2. Change the communication mode of CA Access Control to non-FIPS by editing the following entries in the registry key HKLM\Software\ComputerAssociates\AccessControl\Crypto:

   Communication mode

   Set the value of this key to non_ssl

   FIPS_only

   Set the value to 0.

3. Restart the CA Access Control services and then restart the CA SSO Server service.

   The CA SSO Server is configured in compatible mode.

Configure the CA SSO Server in TLS Mode

To configure the CA SSO Server to run in TLS mode

1. Set the "CommMode" property to 1 which indicates FIPS only mode. This can be done using one of the following methods:

   - Using selang

     Open Selang command prompt and issue the following command:

     ```
     editres PSCONFIGPROPERTY CommMode@ssod gen_prop(value) gen_val(1)
     ```

   - Using Policy Manager

     In Policy Manager, go to Communication Settings in Policy Server settings and change the "CommMode" property to TLS.

2. Change the communication mode of CA Access Control to FIPS by editing the following entries in the registry key HKLM\Software\ComputerAssociates\AccessControl\Crypto:

   Communication mode

   Set the value of this key to fips_only

   FIPS_only

   Set the value to 1.

3. Restart CA Access Control services and then restart the CA SSO Server service.

Configure the CA SSO Server in Mixed Mode

To configure the CA SSO Server to run in mixed mode

1. Set the "CommMode" property to 2 which indicates compatible and TLS communication mode. This can be done using one of the following two methods:

   - Using selang

     Open selang command prompt and execute the following command:

     ```
     editres PSCONFIGPROPERTY CommMode@ssod gen_prop(value) gen_val(2)
     ```

   - Using Policy Manager

     In Policy Manager, go to Communication Settings in Policy Server settings and change the CommMode field to mixed.

2. Restart the CA SSO Server service.

Configure the Policy Manager in Different Modes

The Policy Manager contains Communication modes information in registry keys. Set the following keys as indicated below:

- CA SSO Server Communication

  `HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\AccessControl\Client\ClientType`

  Key: AZN_CommMode

  Value: [0 | 1] For Compatible mode and TLS mode respectively

- Access Control communication

  - Copy the default certificates of CA Access Control from the DVD to the path pointed to by HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\AccessControl\crypto private_key and subject_certificate values.
  - The two files (sub.key and sub.pem) reside in the sample_certs directory on the product DVD.

  `HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\AccessControl\Crypto`

  Key: communication_mode (set to fips_only or non_ssl)

  Value: [fips_only | non_ssl for Compatible mode and FIPS only mode respectively

  Key: fips_only

  Value: [0 | 1] for Compatible mode and TLS mode respectively

Note: Mode 2 is not supported in the Policy Manager as CA Access Control has only 2 modes (FIPS or FIPS and non-FIPS).

The PwdEncUtil Command Line Utility

The CA SSO administrators use the password encryption command-line utility to perform the following tasks:

- Generate new Key Encryption and Password encryption keys and expire or replace active keys. (The KEK and PEK that are currently used by the CA SSO Server.)
- Re-encryption of all existing passwords after changing the PEK.
- Re-encryption of legacy passwords (After an upgrade from r8.1.)
- Verify and retain the integrity of the existing passwords if an earlier operation (such as changing the PEK or re-encrypting the legacy passwords) fails intermittently.

The password encryption utility is placed in *%Installdir%\bin* folder. This utility operates in the following modes:

- PEK Operations Mode
- KEK Operations Mode
- Data Check Mode

Note: We strongly recommend that the PEK and KEK files be backed up at a safe location all the time. The keys must also be backed up immediately whenever either of the PEK and KEK is updated using the "PwdEncUtil.exe" utility. Backing up is useful for restoring the old passwords when the keys are accidentally deleted or corrupted in the server installation directory.

If CA Directory is the User Data Store with greater than 200 users, verify that the CA Directory limits are configured to return maximum number of users. To run this utility on all the users in the database, set the max-op-size of the directory to **0** or to the maximum users in the database and restart the directory services. Setting the limit helps ensure that all user information is encrypted with the same set of encryption keys.

Note: For more information about setting max-op-size, see the CA Directory documentation set.

Important! If the CA Directory max-op-size limit is not configured on databases greater than 200 users before running the password encryption utility, you cannot retrieve the user information for certain users.

PEK Operations Mode

PEK Operations mode focuses on creation of a new password encryption key and maintenance of application passwords.

The following is the syntax for the PEK mode:

```
PwdEncUtil -p -n
```

-p

Indicates PEK operations mode.

-n

Generates a new PEK key and re-encrypts all application password encrypted with old PEK with new PEK.

Note: When you generate a new PEK key, the utility performs a data integrity check that verifies that all user and application login information is re-encrypted with the latest encryption key.

KEK Operations Mode

In KEK Operations mode, a new KEK is generated and used to re-encrypt the Password Encryption Key.

The following is the syntax for the KEK mode:

```
PwsEncUtil.exe -k -n
```

-k

Indicates KEK operations mode.

-n

Generates a new KEK and decrypts the password encryption with old KEK and re-encrypts with new KEK.

Data Integrity Check Mode

In Data Integrity Check mode, the utility is used for two reasons:

- Re-encrypts all user passwords that are encrypted with the legacy SSO algorithm, with AES-256 algorithm.
- Check and retain the integrity of the existing passwords when an earlier operation (such as a PEK change or re-encryption of legacy passwords) failed intermittently or was interrupted.

The following is the syntax for the data integrity check mode:

```
PwdEncUtil.exe -c
```

-c

Indicates data check mode.

Configuration Parameters

The SSO administrator must regularly generate new KEK and PEK keys to ensure security of the system. The following configuration parameter is stored in the Config DSA:

MaxPasswordHistoryCount

Indicates the maximum number of old passwords to be maintained in the history list. A value of 0 or a negative value is interpreted as having no limit.

Default Value: 8

Migration Considerations

The Password Encryption Utility must be run by the SSO administrator in data check mode to migrate existing passwords which are encrypted with legacy SSO algorithm to AES-256 encrypted format.

The sample command for migration is:

```
PwdEncUtil.exe -c
```

Important! You must stop the CA SSO Server services before migrating the existing passwords. When you stop the CA SSO Server services, the CA SSO Server is not available to your users, so you must plan accordingly before running the PwdEnc utility.

Appendix A: Configuring the CA SSO Client

The behavior of the CA SSO Client is determined by the contents of two configuration files: Client.ini and Auth.ini. These configuration files are divided into sections. Each section has one or more keynames (also called tokens or settings) that you can configure to affect the behavior of the CA SSO Client. This chapter describes each of the keynames in the Client.ini and Auth.ini files.

This section contains the following topics:

Location of the Configuration Files

When you install the CA SSO Client, the Client.ini and Auth.ini files are installed in the following location:

```
C:\Program Files\CA\Single Sign-On\Client\cfg\
```

Formatting Rules for the Configuration Files

Certain formatting rules apply to all values in the configuration files.

Separating values

Separate values with spaces or tabs.

Specifying time periods

Time periods are specified using the following syntax:

days d *hours* h *minutes* m *seconds* s

You can specify exact times, for example:

```
1d 5h 30m 0s
```

You can use any of these times. For example:

```
3h
```

A number without an annotation defaults to seconds. For example, the following is parsed as 180 seconds:

```
180
```

Paths or strings with spaces

Quotation marks are not required for paths or strings that contain spaces.

Multiple values

If the first value fails, the second value is used instead and so on through the list.

Edit Configuration Files in Windows Vista, Windows 2008, or Windows 7

To edit configuration files in Vista, you must open the configuration files in an elevated mode.

To edit configuration files

1. Locate a text editor, right-click and select Run as Administrator.
2. Enter the administrator credentials.

   The text editor runs in an elevated mode.
3. Locate and open a configuration file in the text editor.
4. Edit the configuration file and save the file.

   The changes are updated in the configuration files.

Sections in the Configuration Files

This table lists the sections of the configuration files (Client.ini and Auth.ini) with a brief description about each section. Each section of the configuration files contains one or more keynames (also called tokens or settings) that affect the behavior of the CA SSO Client.

These two files are configured using the options you selected during installation, but also contain some default information.

This table describes the end-user experience settings that affect the CA SSO Client.

Client.ini File Section	Description
Localization	Localization information and SSO messages.
Launchbar	Appearance and behavior of the Launchbar.
Lanchbar/OptionsMenu	What options the user sees on the Options Menu of the Launchbar.
Launchbar/AppMenu	What the user sees on the Application Menu of the Launchbar. This appears when the user right-clicks the application buttons on the Launchbar.
GINA	Appearance and behavior of the SSO GINA.
GINA/SystemLogon	Behavior of the SSO GINA in a NetWare environment.
GINA/StationLock	Behavior of the SSO GINA in a shared computer environment.
Credential Providers	SSO Credential Providers Configuration
Credential Provider Tile Images	Appearance of credential provider tiles
SSO Tools	Behavior of the SSO Tools window.
StatusIcon	Behavior of the taskbar SSO status icon.
StatusIcon/Menu	What options the user sees on the taskbar SSO status icon menu.
ApplicationLinks	Whether SSO-enabled applications appear in the Windows Start menu.
UserPage	Whether a user can change their primary credentials from the SSO Tools User tab.
AppListRefresh	How often the user's application list is updated.
Cache	CA SSO Client cache information.
EventCommands	Commands to be run upon specific SSO events.
Logging	Location of the CA SSO Client logging configuration file.
NetworkCommunication	Behavior of the CA SSO Client in relation to the network.
OfflineOperation	Whether offline operation is available on the computer.
ScriptInterpreter	Behavior of the Script Interpreter which runs the SSO scripts.
HLLAPI	HLLAPI settings.
WebAgentIntegration	Fully-qualified URL of the site on which the web server and SSO web agent are running.
ConfigurationSource	Source of the configuration file (Client.ini and Auth.ini). This defines whether you have a common CA SSO Client configuration file that can be automatically updated on all end user's computers.

Client.ini File Section	Description
WindowWatcher	Controls the behavior of Window Watcher. Using Window Watcher, CA SSO Client lets users launch applications from outside CA SSO Client.

This tables describes the settings that affect SSO authentication

Section of Auth.ini File	Description
AuthOptions	Defines how the authentication dialog handles server sets.
ServerSet(n)	Defines the behavior of each of the server sets. You may have multiple ServerSet sections, for example ServerSet1, ServerSet2…)
Auth.WIN	Defines Windows-specific authentication behavior.
Auth.CERT	Defines Certificate-specific authentication behavior.
Auth.RSA	Defines RSA-specific authentication behavior. This is also called Secure ID authentication.
Auth.CITRIX	Defines how the CA SSO Client behaves in on a Citrix Metaframe computer when you are configuring Session Migration.

Client.ini Configuration

This section of the Client.ini file defines settings that affect the user experience.

[Localization]

Locale

Defines which language alphabet the SSO Client interface can display. A three letter language code represents Locale. CA SSO supports the following non-English characters.

Value: *three character language code*

DEU = German

ENU = English

ESP = Spanish

FRA = French

ITA = Italian

SVE = Swedish

Default: *ENU*

ServerMessageFile

Defines the file that contains error messages that the user sees when the Client encounters problems connecting to the SSO Server.

Value: *path and file name*

Default: %SSOINSTALLDIR%\res\enu.msg

HelpDeskMessage

Defines an additional message shown to users when the Client encounters an unexpected error. If this value is left blank, no additional message will be displayed to the user.

Value: *text*

Default: If problems persist, contact the person that manages your CA SSO account.

MessageOfTheDayFile

Defines the location of the message of the day file. This is a message that the user sees when they log onto CA SSO.

Value: *path and file*

Default: [no default]

[LaunchBar]

StartDocked

Defines whether the Launchbar starts docked to one edge of the screen, or whether it is free-floating. The dock location is defined by the DockedEdge token.

Value: [yes|no]

Default: no

DockedEdge

Defines which edge of the screen the Launchbar will start docked to if you specified StartDocked=yes.

Value: [top|bottom|left|right]

Default: top

OffSetX

Defines the X coordinate (horizontal plane) of the Launchbar from the top left corner of the dialog to the top left corner of the screen. This value only applies to the SSO Launchbar if it is in "floating" mode. If StartDocked=yes, this value is ignored.

Value: *The number of pixels*

Default: the window is centred

OffSetY

Defines the Y coordinate (vertical plane) of Launchbar from the top left corner of the dialog to the top left corner of the screen. This value only applies to the SSO Launchbar if it is in "floating" mode. If StartDocked=yes, this value is ignored.

Value: *number of pixels*

Default: the window is centred

AlwaysOnTop

Defines whether the Launchbar should always be visible and stay on top of all other windows. This value is only valid if StartDocked=no.

Value: [yes|no]

Default: no

AutoHide

Defines whether the Launchbar should automatically hide until the mouse moves over it. This token is only valid when the Launchbar=yes.

Value: [yes|no]

Default: no

AutoLogon

Defines whether to initiate the authentication sequence as soon as the Launchbar is started, as opposed to letting the user click the "Logon" button on the Launchbar.

Value: [yes|no]

Default: no

AppLineCount

Defines the default number of application icons that appear in each row of the Launchbar window.

Value: *Number of applications*

Default: 3

DisplayUserName

Defines whether the name of the user must appear in the window title.

Value: [yes|no]

Default: yes

RestorePosition

Defines whether the Launchbar must revert back to its original position when the user logs off.

Value: [yes|no]

Default: no

ConfirmLogoff

Defines whether the user sees a dialog asking them to confirm if they want to log off CA SSO.

Value: [yes|no]

Default: no

ButtonSize

Defines the size of the buttons on the Launchbar.

- Small (16x16 pixels)
- Medium (32x32 pixels)
- Large (48x48 pixels)
- Auto (Median size, but adjusts to the width of the icon to fit the longest caption. You can limit this using MaxCaptionLength)

Values: [small|medium|large|auto]

Default: auto

MaxCaptionLength

Defines the character limit on the width of the caption of any button that appears on the launchbar. This token is only valid when ButtonSize=auto.

Value: *number of characters*

Default: 12

DisplayLogonButton

Defines whether the user sees a Logon button on the Launchbar. When the user clicks this button they will be prompted to authenticate. If you set this to no, to remove the user's control over Logon/Logoff, you may also want to remove the Logon/Logoff buttons on all the user interfaces: the Status Icon and SSO Tools.

Value: [yes|no]

Default: yes

DisplayLockComputerButton

Defines whether the user sees a Lock Computer button on the Launchbar.

Value: [yes|no]

Default: yes

DisplayOptionsButton

Defines whether the user sees an Options button on the Launchbar. When the user clicks the Options button they will see options to:

- Exit the Launchbar
- Refresh their application list
- Display their details
- Auto hide the LaunchBar, if docked
- Display the LaunchBar on top of other windows

Value: [yes|no]

Default: yes

DefaultApplicationIconFile

Defines what icon the user sees for an application in the user interfaces. This is only used if you do not assign a custom icon for an application. If left blank, this will automatically use the default generic SSO application icon.

Value: *path and file name (.ico extension)*

Default: [no default] (therefore the SSO default icon is used)

DefaultApplicationIconIndex

Defines which icon to display to the user if there are multiple icons in the file defined by DefaultApplicationIconFile. If you set this to 0 the first icon will be used. If you set this to 1 the second icon will be used and so on.

Value: see description

Default: 0

DisplayDefaultIconForDisabledApp

Specifies that the default icon is displayed for disabled applications.

Value: [yes|no]

DoubleClickInterval

Specifies whether the user wants to launch applications on double click from the SSO Launch bar window.A delay (milliseconds) occurs when launching an application on any Windows computer (Windows 2008, Vista,and Windows 7). The value can be anything between 150 through 500, for double click to be enabled on the Launch bar. If users do not want Double click functionality or do not want a delay in launching applications on single click, then set the value to 0.

Value: [150 - 500]

Default: 0

[LaunchBar/OptionsMenu]

AlwaysOnTop

Defines whether the "Always on top" item appears in the Options menu on the Launchbar. This lets the user control the behavior of the SSO Launchbar and overrides the AlwaysOnTop token in the Launchbar section.

Value: [yes|no]

Default: yes

AutoHide

Defines whether the "Auto Hide" item appears in the Options menu on the Launchbar. This lets the user control the behavior of the SSO Launchbar and overwrites the AutoHide token in the Launchbar section.

Value: [yes|no]

Default: yes

RefreshApplist

Defines whether the "Refresh Application List" item appears in the Options menu on the Launchbar. This lets the user refresh their application list. You might set this value to no if you wanted to reduce the load on the servers in a large-scale implementation.

Value: [yes|no]

Default: yes

UserConfiguration

Defines whether the "My Details" item appears in the Options menu on the Launchbar. This lets the user change their authentication credentials, if the authentication method supports this.

Value: [yes|no]

Default: yes

Exit

Defines whether the Exit Launchbar item appears in the Options menu on the Launchbar. This lets the user exit from CA SSO.

Value: [yes|no]

Default: yes

[LaunchBar/AppMenu]

EnableAdvancedApplication

Defines whether the user sees a menu when they right-click on an application. This menu lets the user:

- Change their password
- Create a shortcut for that application on their desktop
- Configure the application to run as soon as they log onto Windows

Value: [yes|no]

Default: yes

[GINA]

LogonBitmap

Defines the name (including the path) of an image to use for the SSO GINA's logon window.

If this value is omitted, or that image cannot be loaded, the GINA uses a default bitmap which says “Welcome to CA Single Sign-On". You can customize this text using LogonText value.

Value: *path and image name*

Default: [no default]

LogonTitle

Defines the title for the SSO GINA's logon window. If not specified, the default value is “Windows Logon”.

Value: *dialog title*

Default: Windows Logon

LogonText

Defines the text the user will see in the SSO GINA Logon dialog.

Value: *Text*

Default: Welcome to CA Single Sign-On

LockedBitmap

Defines the name (including the path) of an image to use for the SSO GINA's 'Station locked' window.

If this value is omitted or that image cannot be loaded, the GINA uses a default SSO bitmap which says "This computer is in use and has been locked".

Value: *path and image*

Default: [no default]

LockedTitle

Defines the title for the SSO GINA's 'Station locked' window. If not specified the default value of “Computer Locked” is used.

Value: *dialog title*

Default: Computer Locked

LockedText

Defines the text that the user sees on the SSO GINA when the computer is locked.

Value: *text*

Default: This computer is in use and has been locked.

Font

Defines the font used for LogonText and LockedText on the GINA windows.

Value: *any system font by name*

Default: Arial

FontSize

Defines the size of the font specified in the Font value.

Value: *font size*

Default: 13

GinaPassThrough

Defines whether to bypass the SSO GINA and go to the Microsoft GINA (MSGINA), even if the SSO GINA is installed.

If GinaPassThrough is set to no, the user sees the SSO GINA in all cases (welcome screen, logon screen, Ctrl+Alt+Del options screen and locked screen).

If GinsPassThrough is set to yes, the user sees the MSGINA screen for the logon screen, but sees the SSO GINA for all other GINA screens.

You need to use this setting if you plan to use Full Shared Computer mode. This is used when more than one person shares a computer and each user can share a generic Windows setup, but each user needs to have their own customized SSO applications.

Value: [yes|no]

Default: **no**

LogonCAD

Specify whether to display the Ctrl + Alt + Del dialog at logon.

Value: [yes|no]

Default: yes

FetchDomainsFromSystem

Defines from where the Domains are fetched. If yes is specified, the Domains are fetched from the network. If set to no, the Domains are fetched from the Key Domains INI file.

Value: [yes|no]

Default: yes

Domains

This key is used only if the FetchDomainsFromSystem key is set to no.

Value: *list of domains separated by spaces.*

Default: [no default]

DisplayCustomName

Specifies if the CA SSO Client must display the custom name for the user.

Value: [yes|no]

Default: no

[GINA/SystemLogon]

NetWareLogon

Defines whether to use different Windows and NetWare logon credentials. If you set this value to 'yes' you will need to provide an SSO logon script for NetWare logon. This will perform netware/NDS logon when a user performs sign-on from the GINA.

Value: [yes|no]

Default: no

NetWareServer

Defines the default NetWare server name. This is only necessary if you set NetWareLogon token to 'yes' and the NetWare client does not already specify a preferred server in the following location: HKLM\System\CurrentControlSet\Services\NetwareWorkstation\Parameters\Preferred Server

Value: *Name of NetWare Server*

Default: [no default]

[GINA/StationLock]

EnableOsUnlock

Defines whether you want users to be able to unlock their computer using the Windows Logon option only on the SSO GINA. If this is set to no and the user cannot unlock the computer using their SSO credentials then they will not be able to revert to unlocking the computer using the Windows logon. (Users can override this setting and get the option to bypass SSO by pressing the CTRL + ALT + Z for a dialog instance).

Value: [yes|no]

Default: yes

DisableShutdown

Defines whether you want to disable the Shutdown button. The user sees the Shutdown button on the Windows Authentication dialog when they log on using the SSO GINA and on the Secure Information dialog when they press presses Ctrl + Alt + Del using the SSO GINA.

Value: [yes|no]

Default: no

DisableLogoff

Defines whether you want to disable the logoff button. The user sees the Logoff button on the Secure Information dialog when they press Ctrl + Alt + Del using the SSO GINA.

Value: [yes|no]

Default: no

UnlockStationMode

Defines the workstation mode. These values only apply when the SSO GINA and Credential Providers are in use. When GINA is in use, the valid values are zero through three. For Credential Providers the the valid values are zero through five.

- Single user lock option

  Select option 0. This is used in a regular non-shared computer environment (one Windows user, one SSO user).

- Multiple SSO user lock option
  Select option 1. This is used when two or more people share a customized Windows setup, but need access to their own SSO applications on a computer (one Windows user, multiple SSO users).

- Multiple Windows user lock option
  Select option 2. This is used when more than one person shares a computer and each user needs to have their customized Windows setup as well as their own SSO applications (multiple Windows users, multiple SSO users).

- Kiosk mode lock option
  Select option 3. This is used when more than one person shares a computer and shares a generic Windows setup, but each user needs to have their customized SSO applications. This is like option Multiple Windows user lock mode, but is much faster and suits an environment where several people may have to use one computer in quick succession or may not have a Windows user account.

- Single user Windows session lock option
  Select option 4. This is used in a non-shared Windows session environment (one CA SSO user for one Windows user, but multiple Windows sessions (accounts) are allowed)

- Multiple users Windows session lock option
  Select option 5. This is used when two or more people share the same Windows account but need access to their own SSO applications on a computer (multiple SSO users for one Windows users, but multiple sessions are allowed).

Value: [0|1|2|3|4|5]

Default: **1**

ShowLockedUsername

Defines whether you want the name of the user who locked the computer to appear on the title bar while the computer is locked.

Value: [yes|no]

Default: yes

LogoffWindowsOnLogoutTimeout

Specifies that the CA SSO Client logs off the Windows session if the session timeout parameter is configured in the CA SSO Session Administrator. If the session timeout time expires, the CA SSO Session Administrator logs off users from the Session Administrator. In addition to session timeout, if the LogoffWindowsOnLogoutTimeout is configured, the users are logged off the Windows session too.

Value: [yes|no]

Note: For more information about the session timeout parameter, see the Manually Configure Session Timeout Settings section in this guide.

[SSOTools]

DisplayLogonButton

Define whether the user sees the Logon/Logoff button on the SSO Tools dialog. If you set this to no, to remove the user's control over Logon/Logoff, you may also want to remove the Logon/Logoff buttons on all the user interfaces: the Launchbar and the Status Icon.

Value: [yes|no]

Default: yes

EnableChangePasswordButton

Define whether the "Change Password" button is enabled. This button lets users change their logon credentials for a specific application.

Value: [yes|no]

Default: yes

EnableRefreshButton

Defines whether the Refresh List button is enabled on the SSO Tools interface. This button lets users update their application list from the SSO Server. You might set this value to 'no' if you wanted to reduce the load on the servers in a large-scale implementation.

Value: [yes|no]

Default: yes

ExplorerOpenFolder

Defines whether the Open button is enabled on the SSO Tools dialog when an application folder is selected. If the user clicks this button it launches Windows Explorer and opens to the respective folder in their Start menu if the corresponding folder is created. This is related to the StartMenu_CreateLinksAutomatically value.

Value: [yes|no]

Default: no

[StatusIcon]

DisplayNotifications

Defines whether the user sees the status balloons that pop up from the SSO Status Icon. The two messages the user might see are "Connection to the SSO Server has been lost" and "Connection to the SSO Server has been restored".

Value: [yes|no]

Default: yes

[StatusIcon/Menu]

Logon

Define whether to include the Logon option in the Status Icon menu. If you set this to no, to remove the user's control over Logon/Logoff, you may also want to remove the Logon/Logoff buttons on all the user interfaces: the Launchbar and SSO Tools.

Value: [yes|no]

Default: yes

Logoff

Define whether to include the Logoff option in the Status Icon menu. If you set this to no, to remove the user's control over Logon/Logoff, you may also want to remove the Logon/Logoff buttons on all the user interfaces: the Launchbar and SSO Tools.

Value: [yes|no]

Default: yes

RefreshApplicationList

Define whether to include the Refresh Application List option in the Status Icon menu. When a user clicks this the SSO Server recalculates that user's application list and sends it to the SSO Client. You might set this value to ‘no’ if you wanted to reduce the load on the servers in a large-scale implementation.

Value: [yes|no]

Default: yes

OpenSSOTools

Define whether to include the Open SSO Tools option in the Status Icon menu. When a user clicks this, the SSO Tools window opens.

Value: [yes|no]

Default: yes

OpenLaunchBar

Define whether to include the OpenLaunchBar option in the Status Icon menu. When a user clicks this the SSO Launchbar opens.

Value: [yes|no]

Default: yes

LockComputer

Define whether to include the Lock Computer option in the Status Icon menu. When a user clicks this the workstation is locked.

Value: [yes|no]

Default: yes

ApplicationList

Define whether to include the ApplicationList option in the StatusIcon menu. When a user clicks this they will see their application list as a sub-menu in the StatusIcon menu.

Value: [yes|no]

Default: yes

AboutSSO

Define whether to include the About option in the Status Icon menu. When a user clicks this they will see system information about SSO such as the version number.

Value: [yes|no]

Default: yes

Exit

Define whether to include the Exit option in the Status Icon menu. When a user clicks this the SSO Client Status Icon closes. If the Status Icon is closed, restarting the workstation will restart it.

Value: [yes|no]

Default: yes

[ApplicationLinks]

StartMenu_Folder

Specify the name of the folder in the Start menu that displays the shortcuts to the SSO-supported applications.

Value: *item name in the Start menu*

Default: SSO Programs

StartMenu_Hierarchy

Defines whether the user sees their SSO-enabled applications in their application containers on their Windows Start menu. If this is set to 'no', the user will see all their SSO-enabled applications in a flat list.

Value: [yes|no]

Default: yes

StartMenu_CreateLinksAutomatically

Defines whether the user sees an SSO submenu in the Windows Start menu.

Value: [yes|no]

Default: yes

StartMenu_RemoveLinksAutomatically

Defines whether to delete the SSO submenu from the Windows Start menu (and all the links it contains) when the SSO Client is shut down. You would set this value to 'yes' if you had SSO set up in a Shared Computer environment in order to remove the previous user's shortcuts when a new user logs on.

Value: [yes|no]

Default: yes

AllowUserToCreate

Defines whether the user has the ability to create a shortcut on their Windows desktop or in their Startup Group (to start automatically when Windows starts). The user can configure these options using the check boxes on SSO Tools or by right-clicking an application in the Launchbar and selecting these options from the pop-up menu. You might choose to disable this option if you had SSO set up in a full or partially shared computer environment to avoid a large number of applications being started at Windows startup.

Value: [yes|no]

Default: yes

[UserPage]

EnableChangeAuthCredentials

Defines whether a user can change their primary credentials from the User window (accessed from the SSO Tools, My Details tab or from Launchbar, Options, My Details).

Value: [yes|no]

Default: yes

This section of the Client.ini file defines the settings that affect the core functionality.

[AppListRefresh]

Enabled

Defines whether you want to turn the SSO Client's automatic application list refresh on.

If this is set to 'no', the rest of the tokens in this section are ignored.

Value: [yes|no]

Default: no

Interval

Defines the time between checks for an updated application list.

Note: If this value is set, then the EarliestStartTime and LatestStartTime values are ignored.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: [no default]

EarliestStartTime

In conjunction with LatestStartTime, defines the time period within which a daily refresh occurs (random point between EarliestStartTime and LatestStartTime). You might want to schedule this during low-network traffic periods.

If these values are set, they are only used if the 'Interval' token is not set.

The time is specified as a 24 hour clock: for example, 21:31 indicates 9:31 pm.

Value: *time in [hours]:[minutes]*

Default: 09:00

LatestStartTime

For a full description, see EarliestStartTime in this section.

Value: *Time in [hours]:[minutes]*

Default: 17:00

[Cache]

CacheDirectory

Defines the location of the cache directory that stores the SSO scripts and other information on the Client computer.

Value: *path*

Default: %SSOINSTALLDIRE%\data

UserCacheDirectory

Defines the location of the cache directory that stores the user-specific logon credentials on the Client computer.

Value: *path*

Default: %SSOINSTALLDIRE%\data

ApplicationScriptCachePeriod

Defines how long the SSO Client stores SSO Scripts before refreshing them. You may want to use this reduce network traffic.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 0d

MessageOfTheDayCachePeriod

Defines how long the SSO Client will store SSO application messages of the day. You may want to use this to reduce network traffic.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 0d

ServerPublicKeyFile

Defines where the SSO Server's public key is stored.

Value: *path and file name*

Default: P%SSOINSTALLDIRE%\data\server_key.ini

CacheFileMaxAge

Defines the maximum age of the files in the cache directory. Any data file older than this will be deleted.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 30d

[EventCommands]

SsoOnline

Defines the Windows command line program or script to run when the SSO Client detects Offline-to-Online transition.

Value: *path and script or command name*

Default: [no default]

SsoOffline

Defines the Windows command line program or script to run when the SSO Client detects Online-to-Offline transition.

Value: *path and script or command name*

Default: [no default]

SsoSignOn

Defines the Windows command-line program or script to run when a user logs on to the SSO Client. In versions earlier than SSO r8.1 this token was called UserLogonCmd.

Value: *path and script or command name*

Default: [no default]

SsoSignOff

Defines the Windows command-line program or script to run when the user logs off (or is forced to log off) the SSO Client. In versions earlier than SSO r8.1 this token was called UserLogoffCmd.

Value: *path and script or command name*

Default: [no default]

WindowsLogon

Defines the Windows command-line program or script to run when a user logs on to the computer and authenticates using the SSO GINA or Windows GINA. This coincides with the Windows "Loading personal settings..." dialog when logging-in.

Value: *script name*

Default: [no default]

StartShell

Defines the Windows command-line program or script to run after the Windows "Applying personal settings..." dialog has disappeared, but before the user's shell has become visible.

Value: *path and script or command name*

Default: [no default]

WindowsLogoff

Defines the Windows command-line program or script to run when a user has selected logoff and after their Windows Shell has exited -- it cannot be used to clean-up applications. Its execution coincides with the Windows "Logging off..." dialog.

Value: *path and script or command name*

Default: [no default]

Lock

Defines the Windows command-line program or script to run when Windows is locked. This command is executed once the interactive desktop has changed to Winlogon but before the Ctrl-Alt-Del dialog has appeared.

Value: *path and script or command name*

Default: [no default]

Unlock

Defines the Windows command-line program or script to run after the user has been authenticated by the Windows GINA, but before the interactive desktop is switched from Winlogon to Default.

Value: *path and script name*

Default: [no default]

MaxWait

Defines the time in seconds the CA SSO Client waits for an event command to finish before executing. In versions earlier than SSO r8.1 this token was called EventTimeout. If MaxWait is set to 0, the system returns immediately. If MaxWait is left blank, the CA SSO client will default the value to 60 seconds.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 60s

[Logging]

ConfigFile

Defines the location of the SSO Client logging configuration file. If the value is left blank, logging is disabled.

Value: *logging.properties file and path*

Default: %SSOINSTALLDIRE%\cfg\logging.properties

[NetworkCommunication]

ConnectTimeout

Defines the time in seconds the Client will try to connect to the SSO Server before it gives up.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 120s

ReceiveBufferSize

Defines the size (in bytes) of the largest file that can be downloaded by the Client. You must specify a value higher than the largest SSO application script (in bytes). This value must correspond with the SendBuffSize which is set on the SSO Server.

To set the SendBuffSize on a SSO Server use the Policy Manager and navigate to Configuration Resources, SSO Server Settings, Communication, SendBuffSize and edit this value.

Value: *number of bytes*

Default: 262144

ListenPort

Defines on which port the Client listens for messages from the SSO Server. This affects Session Management. You may need to allow this port number on Client firewalls.

Value: *port number*

Default: 20001

Disconnect

Defines whether the Client disconnects from the SSO Server after every major operation. This frees up server resources and may improve performance in large installations.

Value: [yes|no]

Default: yes

IdentityFile

Defines the path to the certificate file (.pem file) which is used for TLS communication.

Value: *pathname*

[OfflineOperation]

Enabled

Defines whether you want to enable Offline Operation. This lets users connect to CA SSO when the SSO Client cannot establish connection to the SSO Sever and/or the authentication agent.

Value: [yes|no]

Default: yes

TimeLimit

Defines the maximum time that a user may continue to use CA SSO while offline before they must connect to the network and re-authenticate.

Value: *time in seconds (s), minutes (m), hours (h), or days (d)*

Default: 5d

RetryServer

Defines how frequently the SSO Client attempts to reconnect to the SSO Server if the SSO Client is offline. This should be a short enough time that the SSO Client can restore the connection in a timely manner, but long enough so it does not create too much network traffic.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 60s

DisplayOfflineModeConfirmation

If it is set to yes, user is prompted that server is offline and waits for user input to continue or abort the login process. If it is set to no, there will not be any prompt and user will work offline.

Value: [Yes|No]

Default: Yes

[ScriptInterpreter]

Plugins

Defines which additional Tcl extensions (DLLs) the SSO Interpreter must load. The Plugins DLLs must live in the SSO Client directory, or in a directory listed in the PATH environment variable. The DLLs are separated by spaces.

Value: *DLL names*

Default: [no default]

VarsOverwrite

Defines the default values for some of the SSO Interpreter values. For example:

VarsOverwrite=_TIMEOUT=10 _WINTITLE="Windowtitle"

The maximum number of name/value pairs that will be read when parsing is 32. The total string length for VarsOverwrite that will be read in is 254. For more details about the variables, see the *Tcl Scripting Reference Guide*.

Value: *variable name and value*

Default: [no default]

SsoSignOffOnExit

Specifies if the CA SSO user is logged off after the CA SSO interpreter completes executing the script for an application.

Value: Yes|No

Yes

CA SSO users are logged off after the CA SSO interpreter completes executing the script for an application.

No

CA SSO users are not logged off after the CA SSO interpreter completes executing the script for an application.

Default: No

[HLLAPI]

Hllapi

Defines the name of the HLLAPI DLL provided with the emulation software (excluding the path).

For example: Whllapi.dll

Value: *DLL name*

Default: [no default]

HllapiFunc

Defines the name of the function that SSO calls to execute HLLAPI services.

Consult the emulation software documentation or vendor for the function name.

For example: WinHLLAPI

Value: *function name*

Default: [no default]

HllapiDllPath

Defines the location of the HLLAPIDLL folder (do not include the file name).

For example: “C:\Program Files\QWS370 PLUS”

Value: *path*

Default: [no default]

[WebAgentIntegration]

CookieURLs

Defines the fully-qualified URL of the site on which the web server and SSO web agent are running. For example: http://computer1.ca.com

Value: *HTTP URL*

Default: [no default]

SSOTokenVersion

Defines the SSO web token version supported on the web server. This is dependent on the version of SiteMinder that you integrate with SSO Client. For SiteMinder r12 and above, the SSO web token version must be 12. For SiteMinder versions earlier than r12, the SSO web token must be 6. Cookies for target SiteMinder versions are created based on this value.

Value: [6|12]

Default: 6

[ConfigurationSource]

ClientIniFile

Defines the location on the network of the central Client.ini file.

Value: *path and file name*

Default: [no default]

AuthIniFile

Defines the location on the network of the central Auth.ini file.

Value: *path and file name*

Default: [no default]

CachePeriod

Defines how frequently the SSO Client should download the central SSO Client configuration files (Client.ini and Auth.ini) from the central network location.

Value: *time in [days]d[hours]h[minutes]m[seconds]s*

Default: 1d

Note: If you enter the Value 0 for the CachePeriod, then it disables the download configuration feature.

[SessionCleanup]

SessionCleanupIntervalInMins

Specifies the interval between successive cleanups of memory used by tokens. The interval must be in minutes.

Default: 0

[Reauthentication]

SuppressExpirationNotification

Controls the display of the dialog that states "Please re-authenticate to continue this SSO operation" to users.

Yes

Suppresses the dialog that states Please re-authenticate to continue this SSO operation.

No

Displays the dialog to users.

[CredentialProvider]

EnableSSOCredentialProvider

Defines how the Credential Providers will be loaded based on this key. If yes is specified, Credential Providers are loaded. If set to no, the Credential Providers are not loaded.

Value: *[yes/no]*

Default: yes

AddLocalMachineToDomainSelectionOptions

Defines how the local computer name is added to the Domain list box for selection. If yes is specified, the local computer name is added to the Domain list box for selection. If set to no, only the domain is listed in the list box.

Value: *[yes/no]*

Default: yes

FetchDomainsFromSystem

Defines from where the Domains are fetched. If yes is specified, the Domains are fetched from the network. If set to no, the Domains are fetched from the Key Domains INI file.

Value: *[yes\no]*

Default: yes

FilterMSProvider=

Default value is yes. If yes, the Microsoft Credential Provider is filtered and SSO Microsoft Credential provider wrapper is shown. If no, then the Microsoft Credential Provider if available for logon to the computer. This parameter is used only if EnableSSOCrdentialProvider parameter is set to Yes; else, this parameter is ignored.

Value: *[yes/no]*

Default: yes

Domains

This key is used only if the FetchDomainsFromSystem key is set to no.

Value: *list of domains separated by spaces.*

Default: [no default]

LoginBitmap

Defines the path (including the file name) of an image to use for the SSO credential provider logon window.

If this value is omitted, or that the image cannot be loaded, the credential provider uses a default bitmap embedded in the resource file.

Value: *path and image name*

Default: [no default]

MaxConcurrentSessions

Specifies the maximum number of concurrent sessions allowed on a workstation. To enable this setting, enter a positive value. The positive value enables this setting and also indicates the limit on the number of concurrent sessions allowed on a workstation. This setting is disabled by default.

Default:0

LimitChoice

Specifies the behavior of the CA SSO Client when the concurrent session limit is reached and user requests a new session.

Note: This setting is enabled only if the MaxConcurentSessions setting is enabled.
Values: [0-1]

0 - Close Oldest Session

1 – Reject Logon

Default: 0

MonitorAppExes

Specifies that the CA SSO Client verifies each of the existing sessions for applications mentioned in this entry. The CA SSO Client uses this entry only when you have set a limitation on the number of concurrent sessions and a user requests a new session. If the MaxConcurrentSessions limit is reached, the CA SSO Client logs off a session that does not have any of the applications mentioned in this entry. If all the existing sessions have any of the applications from this entry, the CA SSO Client does not create a session.
Example: MonitorAppExes=C:\windows\system32\calc.exe,D:\Program Files\Apps\def.exe

[CredentialProviderTileImages]

ServerSetTile

Defines the path (including the file name) of an image to use for the server set tile. The image must be a .bmp file.

Value: path and image name

Default: [no default]

MSCPTile

Defines the path (including the file name) of an image for the Windows only logon dialog. If this value is omitted, or the image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

SSOCPTile

Defines the path (including the file name) to use for the CA SSO credential provider authentication dialog. If this value is omitted, or the image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

WINCPTile

Defines the path (including the file name) of an image to use for the Windows authentication dialog. If this value is omitted, or the image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

LDAPCPTile

Defines the path (including the file name) of an image to use for the LDAP credential provider authentication dialog. If this value is omitted, or that image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

RSACPTile

Defines the path (including the file name) of an image to use for the RSA credential provider authentication dialog. If this value is omitted, or that image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

CERTCPTile

Defines the path (including the file name) of an image to use for the Certificate credential provider authentication dialog. If this value is omitted, or that image cannot be loaded, the credential provider uses the bitmap specified in the LoginBitmap key. If LoginBitmap key is also empty or cannot be loaded, a default image embedded in the resource file is loaded. The image must be a .bmp file.

Value: path and image name

Default: [no default]

[WindowWatcher]

EnableWindowWatcher

Specifies if Window Watcher is enabled for the CA SSO Client.

Value: Yes|No

Yes

Specifies that Window Watcher is enabled for CA SSO Client.

No

Specifies that Window Watcher is disabled for CA SSO Client.

Default: No

EnableCaseInsensitiveMatch

Specifies if the matching of watchlist attributes and application attributes is case-sensitive.

Value: Yes|No

Yes

Specifies that the matching is case-insensitive.

No

Specifies that the matching is case-sensitive.

Default: No

PollInterval

Specifies the frequency in seconds at which Window Watcher monitors the applications launched by users.

Default: 5 seconds

[PasswordDialogLabels]

PasswordFieldLabel

Specifies the label for the Password field.

Default: Password

Limit: 14 characters

VerifyPasswordFieldLabel

Specifies the label for the Verify Password Field.

Default: Verify Password

Auth.ini Configuration

This section of the Auth.ini file defines settings that affect the authentication options.

[AuthOptions]

ServerSetSelection

Defines whether the user can select a server set when they click the Change button on their authentication dialog.

0 = Always show the server set selection to the user

1 = Only show the server set selection if more than one server set or authentication method is assigned to the user.

2 = Automatically selects the server set and authentication method that was selected by the pervious user, if the RememberLastUser token is activated in this file. If there is only one server set and authentication method assigned to the user, they are not prompted to select these.

3 = Never show server set selection to the user

Value: [0|1|2|3]

Default: 0

Notes:

Login Process

- The ServerSetSelection parameter controls the display of login, lock, and unlock screens that are displayed to users. If ServerSetSelection is set to 0, 1 or 2, the Server Set dialog is displayed to users and users can change the authentication method and domain used for authentication.
- If ServerSet Selection is set to 3, the login screen populates the default authentication method and domain and prompts the user for username and password. Users can change the authentication method and domain by pressing ESC.

Lock and unlock

- If ServerSetSelection is set to 0, 1 or 2, the Server Set dialog is displayed to users and users can change the authentication method and domain used for authentication.
- If ServerSetSelection is set to 3, displays the locked tile of the previous user, to changed username you must press ESC. The user tile then populates the authentication methods and domain of the previous user and prompts for username and password. Users can change the authentication method and domain by pressing ESC.

RememberLastUser

Defines whether the Client remembers the previous user who logged on and uses those details to pre-populate the server set dialog during the SSO logon.

Value: [yes|no]

Default: yes

ServerSetAlphabetic

Defines whether to list the server sets alphabetically. If set to no, the Client will order these according to server set number.

Value: [yes|no]

Default: no

RestrictLogonDomain

Defines whether to restrict the user's access to the domain. If this is set to yes, the user cannot select the domain from the SSO GINA window. They will only be able to select to logon to their own computer, without access to the domain.

Value: [yes|no]

Default: [no default]

This section of the Auth.ini file defines settings that affect the server sets.

[ServerSet1]

Name

Defines the name that you want users to see in their Server Set drop-down list on the authentication screen. For example, “Home Logon” or “Work Logon”.

Value: *server set name*

Default: [None]

PolicyServers

Defines the list of SSO Servers in a set. Use spaces to separate. When an SSO Client attempts to connect to a SSO Server, it tries to use the first SSO Server in the list. If that SSO Server is not available, the second SSO Server in the list is used. For example: server1 server2. You must never leave this value empty.

Value: *server name(s)*

Default: [no default]

AuthMethods

Defines the list of authentication methods available to users. Use spaces to separate. The first value listed here appears in order in the Authentication dialog and the first value is the default. If the first value fails to load, the next value is used.

You must specify an authentication host in the AuthXXX section below that corresponds to the authentication method(s) defined here or the Server Set is deemed invalid and will not be displayed.

The exceptions to this rule are the AuthWIN and AuthCITRIX methods that will allow the use of the keyword <auto>. The <auto> keyword signifies to locate the nearest domain controller, or nearest Citrix server respectively.

For example:

```
AuthMethods=LDAP
AuthLDAP=AuthServer1 AuthServer2
```

Value: [CERT|LDAP|RSA|SSO|WIN|

Default: [See description]

OfflineTimeout

Defines the time in seconds that elapses before an authentication host that has been marked as offline can be retried. If all the hosts specified for a given ServerSet/authentication method are marked as offline, rather than making the authentication attempt fail, all associated hosts will be retried again.

The minimum value is 300 seconds (5 minutes). If you set this value to less than 300, it will automatically adjust this value to 300.

Value: *time in seconds*

Default: 1800

LogicalAuthSSO

Defines the logical authentication hosts to use for SSO authentication.The default value is SSO_Authhost which is automatically configured during installation.

Note: If you want to edit the Auth.ini file post installation, the LogicalAuthSSO setting must always be set to SSO_Authhost. Any other value could cause SSO authentication to fail.

Value: SSO_Authhost

Default: SSO_Authhost

AuthLDAP

Specify the list of the authentication hosts to use for LDAP authentication. Use spaces to separate.

There is no default value for AuthLDAP unless specified during the installation.

When an SSO Client attempts to authenticate, it tries to use the first host in the list. If that host is not available, the next host in the list is used.

Value: *server name(s)*

Default: [See description]

AuthWIN

Specify the list of the authentication hosts to use for Windows authentication. Use spaces to separate.

There is no default value for AuthWIN unless specified during the installation.

When an SSO Client attempts to authenticate, it tries to use the first host in the list. If that host is not available, the next host in the list is used.

Value: *server name(s)*

Default: [See description]

AuthCERT

Specify the list of the authentication hosts to use for Certificate authentication. When an SSO Client attempts to authenticate, it tries to use the first host in the list. If that host is not available, it uses the second host in the list and so on.

There is no default value for AuthCERT unless specified during the installation.

Value: *server name(s)*

Default: [See Description]

AuthRSA

Specify the list of the authentication hosts to use for RSA SecureID authentication (previously referred to as SDI authentication). Use spaces to separate.

There is no default value for AuthRSA unless specified during the installation.

When an SSO Client attempts to authenticate, it tries to use the first host in the list. If that host is not available, the next host in the list is used.

Value: *server name(s)*

Default: [See description]

AuthCITRIX

Specify the Citrix Server to use.

The <auto> value signifies that the Citrix virtual channel will be used to retrieve the user's SSO ticket from the user's local computer and for that ticket to be used automatically by the SSO Client on the Citrix Server.

Value: *Citrix server name(s)*

Default: <auto>

[Auth.WIN]

ConnectionTimeout

Defines how long in seconds the SSO Client tries to connect to the Windows authentication server before it times out.

Value: *number of seconds*

Default: [no default]

IdentityFile

Defines the certificate file on the SSO Client computer. This certificate file and password are supplied during the SSO Client installation. This ensures an SSL-secured connection between the SSO Client computer and the Windows authentication agent computer.

Value: *file name*

Default: Supplied during SSO Client installation

IdentityPassword

Defines the password associated with the certificate file (defined by the IdentityFile token). This certificate file and password are supplied during the SSO Client installation. This ensures an SSL-secured connection between the SSO Client computer and the Windows authentication agent computer. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component.

Value: *password*

Default**: Supplied during SSO Client installation**

TrustFile

Defines the .pem file that establishes trusted communications between the SSO Client computer and the Windows authentication computer. This is mandatory. This value is entered during installation of the SSO Client.

Value: *file with .pem extension*

Default: Supplied during SSO Client installation

FIPSIdentityFile

Defines the certificate .pem file used for TLS communications between the CA SSO Client computer and the Windows authentication computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .pem extension*

Default: Supplied during CA SSO Client installation in FIPS or Mixed Mode.

FIPSTrustFile

Defines the .pem file that establishes trusted communications between the CA SSO Client computer and the Windows authentication computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .pem extension*

Default: Supplied during CA SSO Client installation in FIPS or Mixed Mode.

PrivateKeyPath

Defines the certificate .key file used for TLS communications between the CA SSO Client computer and the Windows authentication computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .key extension*

Default: Supplied during CA SSO Client installation in FIPS or Mixed Mode.

AutoNetworkAuth

Defines whether to let the Windows authentication method use the user's network credentials to log them on to the SSO Client.

If no, the user must enter their credentials manually.

If yes, we recommend that the SSO administrator adds a logoff script to log the authenticated user off the network.

Value: [yes|no]

Default: No

PDCFallback

Defines whether Windows authentication attempts to find an authentication host on the domain if none of the specified hosts are online or the specified host is '<auto>'.

Setting the host to '<auto>' and PDCFallback to 'no' is invalid and causes Windows authentication to fail.

Value: [yes|no]

Default: yes

NearestDomainController

If none of the specified hosts are online, or the specified host is '<auto>', Windows authentication attempts to find an authentication host on the domain.

In this case, this value is used to define whether the SSO Client must try to authenticate to the nearest Domain Controller (DC).

If yes, the SSO Client tries to connect to the nearest DC on the network regardless of the target OS (Active Directory architecture).

If not, the SSO Client tries to connect to the Primary Domain Controller (PDC) specified.

If <auto>, the SSO Client checks the operating system and behave accordingly. It tries to connect to the nearest DC (presumes Active Directory architecture) and if this fails, connects to the PDC (presumes NT4 architecture).

This value does not have any effect if PDCFallback is set to no.

Value: [yes|no|<auto>]

Default: <auto>

[Auth.CERT]

CertStore

Defines where the user certificate is stored.

- FILE = Stores user certificate in a local disk file (PKCS#12 format)
- PKCS11 = Stores the user certificate on a PKCS#11 token (smart card in most cases).
- MSCAPI = Uses the MSCAPI certificate store or smart card

You can specify all storage methods, for example:

```
certStore=PKCS11 FILE
```

Value: [FILE|PKCS11|MSCAPI]

Default: FILE

Pkcs11LibraryPath

Defines the full path name of the PKCS#11 library that you want to use with the type of PKCS#11 token you have selected.

This must be defined otherwise you will not be able to use the PKCS#11 token. This property is only relevant if CertStore=PKCS11.

If the CertStore attribute is set to FILE, then the attribute can be left empty.

Value: *path*

Default: [no default]

Pkcs11PromptText

Defines the prompt the user sees when the PKCS#11 token radio button is selected on the certificate authentication page.

If this is left empty, the default text will be used. The default text is: “Insert your token into reader and enter your password.”

Value: *text*

Pkcs11TokenAbsenceBehavior

Specifies how the SSO Client behaves upon removal of the SmartCard while it is configured with CertStore=PKCS11

Value: 0|1|2

0

Do nothing

1

Workstation lock

2

SSO logoff

Default: 0

DisablePasswordField

Defines whether to disable the password field. You might do this when you want to force an authentication method other than password or PIN.This is only related to authentication using PKCS#11 token.

If the password field is disabled, the system forces the user to use a third-party authentication method (such as user's fingerprints) to authenticate to the smart card.

Value: yes|no

Default: no

AutoAuthenticate

Defines whether to automatically authenticate the user using the certificate authentication method with the certificate stored in the Windows user's personal certificate store. To view the certificates in the Windows user's personal store, open Internet Explorer and click Tools, Internet Options, Content, Certificates, Personal. This applies to the MS CAPI certificate store only. (CertStore=MSCAPI)

Value: [yes|no]

Default: no

CertThumbPrint

Defines the thumb print (SHA1 hash) of the certificate in the Windows user's personal certificate store to use when carrying out automatic authentication (AutoAuthenticate=yes). This is only related to automatic authentication using MS CAPI (CertStore=MSCAPI). This setting is frequently used in Shared Computer environments. To get the thumb print of the certificate you want to use, open Internet Explorer and go to Tools, Internet Options, Content, Certificates, Personal. Select the certificate you want to use and click View. On the Details page of the dialog that pops up, there is a field named Thumbprint. Copy the value of this field and use it as the value of this configuration setting. Remember to remove any spaces among the text.

Value: text

Default: [no default]

DefaultServerSet

Defines which Server Set to use with Certificate authentication if the user starts the authentication process by inserting a PKCS#11 token or an MS CAPI-enabled smart card into one of the PKCS#11 slots or smart card readers that are connected to the end user workstation. This is only relevant when the end user is using SSO GINA to log on and the user is facing the GINA welcome dialog.

Value: *server set name*

Default: [no default]

IdentityFile

Defines the certificate .pem file used for TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .pem extension*

Default: Supplied during the CA SSO Client installation in FIPS or Mixed Modes.

VSlotsPerReader

Specifies the number of virtual slots NetID software configures for a Reader. This value must be equal to number of pins you configure.

Default: 1

VSlotForAuth

The Virtual Slot that is configured for Authentication.

Default: 0

Range: 0 to <VSlotsPerReader>

AutoCertSelection

Specifies if CA SSO must filter certificates and display only the filtered certificates to the users.

Value: Yes|No.

Yes

Specifies that CA SSO filters and displays only the filtered certificates to the users. The other filtering related parameters are activated only if this option is set to Yes.

No

Specifies that CA SSO does not filter the certificates, but displays all the certificates to the users.

Default: No

FilterDLLPath

Specifies the path to the namemapping DLL that implements the filter.

Value: path

MappingMethod

Specifies the certificate parameter that the CA SSO Client uses for certificate filtering.

Value: [C|CN|DN|DNS|EMAIL|IP|L|O|OU|URI|UPN]

Default: CN

ExpectedValue

Specifies the value that is matched with the value of the parameter in MappingMethod. If this parameter matches the value for MappingMethod parameter, CA SSO Client displays only those certificates to the users.

Example: SHA1 value of thumb print

ShowFilteredCertificates

Specifies if CA SSO Client displays the list of filtered certificates to the users.

Value: Yes|No.

Yes

Specifies that CA SSO Client displays the list of filtered certificates to the users.

No

Specifies that CA SSO Client does not display the list of filtered certificates to the users. If you select this option, CA SSO Client uses the first certificate that matches the filter criteria to authenticate users.

Default: No

FilteringPattern

Specifies the filtering pattern that CA SSO Client uses to match the ExpectedValue parameter with the MappingMethod parameter.

Value: 0|1|2

0

Specifies that the entire string from the ExpectedValue is matched with the value of the certificate parameter specified in MappingMethod.

1

Specifies that the start of the string from the ExpectedValue is matched with the value of the certificate parameter specified in MappingMethod.

2

Specifies that ExpectedValue is treated as a substring and is matched with substrings of value of the certificate parameter specified in MappingMethod.

Default: 0

[Auth.LDAP]

IdentityFile

Defines the certificate .pem file used for TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .pem extension*

Default: Supplied during the CA SSO Client installation in FIPS or Mixed Modes.

[Auth.RSA]

IdentityFile

Defines the certificate .pem file used for TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation of the CA SSO Client in FIPS or Mixed Mode of installation.

Value: *file with .pem extension*

Default: Supplied during the CA SSO Client installation in FIPS or Mixed Modes.

HidePinInputField

Specifies if the CA SSO Client displays the PIN field in the RSA authentication dialog. Set the value to Yes to hide the PIN field. Set the value to No to display the PIN field.

Value: [Yes|No]

Default: No

Note: If you set this field value to Yes, the value of the Token code field in the RSA authentication dialog is masked.

[Auth.CITRIX]

ReadTimeOut

Defines the length of time the SSO Client on the Citirix Server waits for a response from the SSO Client on the end user workstation.

Value: *time*

Default: 60s

Registry Configuration

Use the following registry file to define settings that affect the user experience.

Client registry Key

`HKLM\Sofware\ComputerAssociated\SingleSignOn\Client`

CommMode

Specifies a dword value with takes 0, 1 or 2

- 0 - Compatible Mode
- 1 - TLS Communication Mode
- 2 - FIPS compatible TLS Communication Mode

After the SSO Client has been installed, if the communication mode needs to be changed, this registry key must be changed accordingly, and the CA SSO Client service restarted. Also, the path for the IdentityFile key in the client.ini and the chosen authentication agents must be set in case of mode 1 and 2 mentioned above.

Appendix B: Configuring the CA SSO Server

CA SSO Server configuration is divided into two sections.

- The global server parameters
- Machine specific parameters (for example file paths).

The machine-specific parameters are stored in the machine registry (Windows) and override the global ones (which are stored in CA Access Control and can be modified using the CA SSO Policy Manager).

Each configuration parameter is divided into sections. Each section has one or more settings/parameters that you can configure to affect the behavior of the CA SSO Server. This appendix describes each of those configuration parameters.

This section contains the following topics:

Where the SSO Server Settings are Located

This section describes the SSO Server settings that you can adjust. You can adjust most of the SSO Server settings using the SSO Server Settings configuration resource in the Policy Manager. Other settings are configured using ini files on UNIX or the system registry on Windows.

CA SSO Server Settings

The following list describes the CA SSO Server Settings subfolders that contain the properties you can configure using the Policy Manager:

Subfolder	Defines
AD Authentication Provider	Settings that control the changing of a user's password in Active Directory.
Cache	Client side authorization cache settings
Communication	CA SSO Server communication settings
General	General CA SSO Server settings

Subfolder	Defines
Manage Idle Connections	Settings that control the background process that closes inactive connections to LDAP user data stores
One Time Password	One-time password settings
Refresh Applications Cache	Settings that control the refreshing of the Applications Cache
Refresh Authentication Host Cache	Settings that control the refreshing of the Authentication Host Cache
Refresh Password Policy Cache	Settings that control the refreshing of the Password Policy Cache
Refresh Session Management Profile Cache	Settings that control the refreshing of the Session Management Profile Cache
Refresh User Directory Cache	Settings that control the refreshing of the User Directory Cache
Remove Expired Tokens	Controls the background process that removes expired tokens from the Token Directory
Remove Heartbeat Failed Tokens	Controls the background process that removes expired heartbeat failed tokens from the Token Directory
Revoke	Configuration parameters for the revoke feature. This feature suspends the end user after a specified number of logon attempts that failed because of improper logon credentials.
Session Management	Session management settings

One Time Password

GenLowWater

Specifies the generation number at which the password is expired.

Default: 4

MaxGenerations

Specifies the initial generation number for one-time passwords.

Default: 1000

MaxSeedLength

Specifies the maximum seed length.

Default: 10

PortNumber

Specifies the port number of the TCP port where the one-time password agent will listen.

Value: *port number*

Default: 13967

Session Management

AllowTicketFromMultipleStations

Specifies whether two machines can use the same SSO ticket. For Citrix MetaFrame environments, set this to *Enabled*.

Value: Disabled | Enabled

Default: Disabled

NonActivatedSessionsExpiration

Specifies how long would a non activated session would be valid before expiring or before activated.
a non-activated session is created when the maximum number of sessions for a user has reached, and the user is allowed to choose a session to kill (the limit choice is set to Select Specific Session)

Value: number of seconds before expiration

Default: 900

SessionlessTerminals

Specifies the terminals that are exempt from session management. If you leave this blank, all terminals are subject to session management.

Value: [Blank] | *Comma-separated list of IP addresses*

Default: [no default = blank]

SessionTerminationTimeout

Specifies how long the CA SSO Server waits for an SSO Client to shut down before continuing to log a user on the new session.

Only relevant if using the Direct Notification method.

Value: *Value: Number of seconds*

Default: 40

AD Authentication Provider

UseLDAPChangePassword

Specify the method used to change a user's password in Active Directory.

If you specify yes, LDAP is used. If you specify no, NetAPI is used.

Value: yes|no

Default: yes

MaxConnections

The maximum numbers of users allowed to be logged in simultaneously.

Value: Number of connections

Default: 1000

Remove Heartbeat Failed Tokens

Enabled

Specifies whether the background process that removes expired heartbeat failed tokens from the Token Directory is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between each section of a background process job. This idle period prevents the CA SSO Server and CA Directory from being flooded during a cycle.

Value: *Number of seconds*

Default: 5

Interval

The time between background process jobs.

Value: *Number of seconds*

Default: 3600

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of minutes after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of minutes* | *hours*h*minutes*

Default: 60

Manage Idle Connections

Enabled

Specifies whether the Manage Idle Connection background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Specifies the time between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Specifies the time between background process jobs.

Value: *Number of seconds*

Default: 36

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 120

Remove Expired Tokens

Enabled

Specifies whether the background process that removes expired tokens from the Token Directory is enabled.

Value: yes|no

Default: yes

IdlePeriod

Defines the length of the pause between each iteration of a background process job. This idle period prevents the CA SSO Server and CA Directory from being flooded during a cycle.

Value: *Number of seconds*

Default: 5

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 3600

StartTime

Defines the start time of the background process job. This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds | hourshminutes*

Default: 300

Refresh Applications Cache

Enabled

Specifies whether the “Refresh Applications Cache” background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 28800

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 28800

Refresh Authentication Host Cache

Enabled

Specifies whether the “Refresh Authentication Host Cache” background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 28800

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 28800

Refresh Password Policy Cache

Enabled

Specifies whether the “Refresh Password Policy Cache” background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 28800

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 28800

Refresh Session Management Profile Cache

Enabled

Specifies whether the “Refresh Session Management Profile Cache” background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 28800

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 28800

Refresh User Directory Cache

Enabled

Specifies whether the “Refresh User Directory Cache" background process is enabled.

Value: [yes | no]

Default: yes

IdlePeriod

Defines the length of the pause between two sequential operations within the process job.

Value: *Number of seconds*

Default: 1

Interval

Defines the length of the pause between background process jobs.

Value: *Number of seconds*

Default: 28800

StartTime

Defines the start time of the background process job.

This can be in either of two formats:

- The number of seconds after the CA SSO Server starts
- The time of day at which the process must run, in 0h0 format. For example, 22h0 is 10 p.m.

Value: *Number of seconds* | *hours*h*minutes*

Default: 28800

Cache

ClientAZNLifeTime

Defines the time that the authorization cache entry is valid on SSO Client side.

Value: *Number of seconds*

Default: 300

Revoke

def_disable_time

Specifies the period of time a user account will be suspended after providing the wrong password while using SSO (primary) authentication a certain number of times (defined by def_fail_count).

Value:

Default: [no default]

def_fail_count

Specifies the number of times a user is allowed to put in a wrong password in SSO authentication before the account gets suspended.

Value:

Default: [no default]

Communication

Auto_RollOut

Automatically moves the next password to current password, if the next password exists.

If an administrator changes the value of Auto_RollOut from 0 to 1, and the user has a nextpwd value in the logon record for the __SSO__ application, then the administrator needs to nullify the nextpwd value. This is because users may forget the last change they made to the synchronized password application.

If the CA SSO Server and the SSO authentication host are not on the same computer, you must configure both computer as a server farm to enable password synchronization between the SSO password and the synchronized applications.

Value: 0 | 1

Default: 1

ForkLimit

Defines the maximum number of concurrent connections (communications) to handle.

Default: 50

MaxAppQty

Defines the maximum number of applications per user.

Default: 200

PortNumber

Specifies the port number of the TCP port that the CA SSO Server listens on.

Default: 13980

SSLPortNumber

Specifies the port number of the TLS port that the CA SSO Server listens on.

Default: 13981

CommMode

Specifies the Communication mode that the server is running on.

Value: = Compatible | TLS | Mixed

PropDefaultMode

Specifies whether to use _default PWPOLICY as the property default (enabled) or as the pwpolicy default (disabled).

Value: Enabled | Disabled

Default: Enabled

RecvBuffSize

Specifies the length of the buffer used when receiving the data (in bytes).

Value: *number of bytes*

Default: 262144

ReverseIpLookup

Internal server parameter. This value must be disabled.

Value: Enabled | Disabled

Default: Disabled

SendBuffSize

Defines the send buffer size for communication in bytes.

Value: *number of bytes*

Default: 262144

SensitiveExpiration

Defines the lifetime of the ticket for sensitive applications in hours and minutes, using 0h0 format. The minimum time is one minute (0h1).

Value: *hours*h*minutes*

Default: 0h5 (5 minutes)

TicketExpiration

Defines the lifetime of the ticket in hours and minutes, using 0h0 format. The minimum time is one minute (0h1).

Value: *hours*h*minutes*

Default: 8h0 (8 hours, 0 minutes)

TimeOutConnect

Defines the timeout period used during connection.

Value: *Number of seconds*

Default: 120

TimeOutRecv

Defines the timeout period used during receiving.

Value: *Number of seconds*

Default: 60

TimeOutSend

Defines the timeout period used during sending.

Value: *Number of seconds*

Default: 60

UdpInUse

Enables or disables CA SSO Server's UDP listener.

Value: [Enabled | Disabled]

Default: Enabled

UdpPortNumber

Specifies the port number of the UDP port where CA SSO Server will listen.

Value: *port number*

Default: 13990

General

AllowDefaultEncKey

Specifies whether CA SSO Server can accept SSO tickets generated with default or empty encryption keys.

Value: [Allowed | Not allowed]

Default: Not allowed

CommListenQueueSize

Defines the maximum length of the queue of pending connections. This is the backlog of incoming connections.

Value: *number of connections in queue*

Default: 20

CommMaxIdle

Specifies the maximum period of time the information about a connected user remains in CA SSO Server cache when the user is not using the server.
In case the user starts using the server again, and his authentication has not expired, the server retrieves this information again from the token directory.

When no value is specified (default) the ticket expiration time is used.

Value: *hours*h*minutes*

Default: [no default]

DefaultAdminGroup

Defines the name of the group whose members are administrators. This is used only for a quicker browsing of the user names. No authorization decision is based upon this, and users must still be assigned the admin role.

Value: *administrator group name*

Default: _ps-adms

DefaultContext

Defines the default container DN used for SSO authentication. If no DN is defined, the users are located in the base container.

Value: *container name*

Default: [no default]

DefaultLocale

Specifies the locale of old (non-UTF8) SSO clients.

By default the CA SSO Server is set to use English as the default language. If you want the CA SSO Server to use characters other than English then you must change this setting.

For example, if you have users with French characters in their names, you must change this setting to FRA.

- DEU = German
- ENU = English
- ESP = Spanish
- FRA = French
- ITA = Italian
- SVE = Swedish

Value: *thee digit language code*

Default: ENU

DefaultMethod

Specifies the default authentication method. This method is used when the server gets an authentication request with no authentication method specified.

Value: [CERT | LDAP | RSA | SSO | WIN]

Default: SSO

DefaultScriptFormat

Specifies whether a script file must be treated as UTF8 or Multibyte. If the script file header specifies that the file is UTF8 (windows), it overrides this setting.

Value: [Multibyte | UTF8]

Default: Multibyte

DefaultTOKEN_DIR

Specifies the name of the Token Dir object in CA Access Control, which stores the token dir connection and general information.

Value: *name of the token directory*

Default: PSTD

DefaultUSER_DIR

Specifies the name of the User_Dir object in CA Access Control, which stores the connection and general information of the default user data store. This is used when using SSO authentication.

Value: *name of the user directory*

Default: ps-ldap

EnablePerfMon

Defines whether the CA SSO Server statistics and monitor collector is running. This collector is used by a performance monitor (refer to Microsoft performance monitor for more information) and psperfmon utility.

Value: [yes | no]

Default: Yes

EnforceStationId

Defines how the CA SSO Server handles security tokens. If this is marked Enabled, the CA SSO Server restricts security tokens to the single computer to which these tokens were issued to. If this is marked Required, the CA SSO Server does not support agents that do not support this feature.

Value: [Disabled | Enabled | Required]

Default: Disabled

MaxConnections

Defines the maximum number of connected users the CA SSO Server will store in its internal cache.

Value: *Number of users*

Default: 10,000

RecieveQueueSize

Defines the maximum length of the queue of accepted connections waiting to be served.

When this is left empty (which is the default), the value is determined automatically by multiplying Fork Limit by 10.

Value: *Queue length*

Default: [no default]

SendBusyQueueSize

Determines the maximum length of the queue where the server collects all of the client requests for sending back a “busy” response.

Value: *number of busy requests in queue*

Default: 20

ServerFarmSupport

When disabled, it decreases the number of accesses to the Token Directory and thus increases performance in a single CA SSO Server.

Value: [Disabled | Enabled]

Default: Enabled

ServerIP

Sets the IP address of the CA SSO Server that can be used to access the CA SSO Server directly in configurations where the CA SSO Server has more than one IP address, when behind a firewall, a smart router or NAT.

Value: *IP address*

Default: [no default]

ServerVirtualIP

Defines the IP address of the CA SSO Server used by SSO Clients that connect to the CA SSO Server through the CA SSO Server API when the CA SSO Server is behind a firewall, a smart router or NAT.

In most cases, this is the IP address of a router or load balancer.

Default: [No default]

SSOAuthHostName

Defines the name of the AUTHHOST object in the administrative data store that is used by the CA SSO Server when generating tickets for SSO authentication.

Default: SSO_Authhost

TrustedPolicyServers

List of CA SSO Server computers in the farm and their aliases

Default: [no default]

Registry and INI File Settings

In addition to the SSO Server settings you can configure through the Policy Manager, you need to configure some settings through the Windows registry or INI files.

If your SSO Server is running on a Windows machine, use the following Windows Registry key to configure those settings:

```
HKEY_LOCAL_MACHINE\Software\ComputerAssociates\SingleSignOn\Server
```

If your SSO Server is running on a UNIX machine, use the policyserver.ini file to configure those settings. If you installed the SSO Server in the default location on UNIX, look for the INI file here:

```
/opt/CA/SingleSign-On/Server/policyserver.ini
```

The following list describes the sections in the INI file, and the keys in the Windows Registry:

Section or Key:

ssod

Defines SSO Server communication settings

bg.CIA

Defines information used by the Watchdog background task responsible for checking if the SSO Server is available.

exits

Defines password exit tokens

Main

Defines general SSO Server settings

auth.<method_name>

Defines authentication plug-in settings and internal parameters

AuthMap

Defines the authentication methods used

UserDBProvider.<provider_name>

Defines user data store provider settings and internal parameters

CacheSync

Defines the details of the internal queue where online updates sent by CA Access Control are stored before being processed by CA SSO Server.

ssod

The following settings control the connections between the Watchdog and the client you use to monitor the Watchdog:

WDAuthHostName

Defines the name of the authentication host that the Watchdog will use to authenticate to the SSO Server.

Default: EAC_Authhost

WDOnlineChecksMode

Defines whether the Watchdog reports a cached status of the SSO Server, or whether it checks the SSO Server status in real-time.

The frequency of the Watchdog checks on the SSO Server is determined by the settings in the bg.CIA.

Value:

- 0 - Watchdog checks the SSO Server status in the cache (faster)
- 1 - Watchdog checks the real-time SSO Server status (slower)

Default: 1

WDHTTPPort

Defines the port number the client should use to make HTTP requests to the Watchdog.

Default: 13391

WDTCPPort

Defines the port number the client should use to make direct requests to the Watchdog.

Default: 13392

WDCommListenQueueSize

Defines the maximum number of requests from the client to the Watchdog service that Windows will store.

Default: 10

WDCommRecvTimeout

Defines the time it takes for a client connection to the Watchdog to fail (through browser or telnet, or the smart router).

Note: This is not the Watchdog timeout for connections to the SSO Server.

Value: *Number of seconds*

Default: 5

WDOnFailureString

Defines the text sent to the client if the Watchdog detects a failure on the SSO Server.

Default: SYSTEM TEST FAILURE

WDOnSuccessString

Defines the text send to the client if the Watchdog test of the SSO Server is a success.

Default: SYSTEM TEST SUCCESS

WDOnInitString

Defines the text sent to the Client if the Watchdog test to the SSO Server indicates that the server is still in the start-up stage.

Default: WATCHDOG IS INITIALIZING

The following settings control the connections between the Watchdog and the SSO Server:

WDConnTimeout

Defines the time in seconds that it takes the Watchdog to connect to the SSO Server before it registers this as a fail.

Default: 2

WDMaxConnections

Defines the maximum number of connections the connection pool can use for the Watchdog to connect to the Server.

Default: 5

WDMaxThreads

Defines the maximum number of threads the thread pool can use.

Default: 10

WDWorkingDirectory

Defines the path to the directory which contains the credentials file. These credentials let the Watchdog log onto the SSO Server.

Default: <ps_path>

The following settings control the notifications and commands that occur when the Watchdog detects a particular state within the SSO Server.:

WDNotificationCommand

Defines a command that is run when the number of restart commands is exceeded.

Default: [no default]

WDNumRestartToNotify

Defines the number of times the restart command should fail before a notification is sent to the client.

The notification is set in the WDNotificationCommand setting.

Default: 5

WDRestartNotificationCommand

Defines a command to run before the SSO Server is restarted.

Default: [no default]

WDMsgFile

Defines the name of the Watchdog message file. This file should always end in an extension of .msg. This file contains the messages that are sent to the Watchdog log file (PSWDLOG.ini).

Default: ENU.msg

WDMsgPath

Defines the path to the Watchdog message file.

Default: <server_dir>lang

WDLoggerIniFile

Defines the name of the Watchdog log INI file.

Default: <sso_server_dir>PSWDLOG.ini

The following settings control the timeout settings of the Watchdog service:

WDNotificationLimit

Defines the minimum time between running the notification commands sent from the Watchdog to the client due to successive restart failures.

Value: *Number of seconds*

Default: 3600

WDMinTimeBetChecks

Defines the minimum time in seconds for which the watchdog stores the most recent CA SSO Server status. When you manually request the status of CA SSO Server, the watchdog service checks this interval. If this interval is lapsed, the watchdog service connects to the CA SSO Server and updates the status.If this interval is not lapsed, the watchdog service returns the stored status of the recent check. The CA SSO Server status is displayed in the browser.

Note: If the interval is set to 0, for every status request, the watchdog service connects to the CA SSO Server to update the status.

Value: 0 | *Number of seconds*

Default: 0

The following settings are general SSO Server settings, and do not relate to the Watchdog service:

AppTktFile

Defines the full path to the AppTicket key file.

Default: <ps_path>/appticket.key

BackCalcPath

Defines the path to the BackCalc (Server cache) files.

Default:

- On UNIX - /opt\CA\Single Sign-On\Server\psbgc
- On Windows - C:\Program Files\CA\eTrust SSO\Server\PsBgc

GlobalPreCmdPath

Defines the Global PreCommand SSO-Tcl script file name that is used. For example, Global.tcl.

Default: [no default]

MotdPath

Defines the path to the MOTD (Message of the Day) files.

Default: <ps_path>/motd

ScriptPath

Defines the path to the SSO Tcl scripts.

Default: <ps_path>/scripts

SsodKeyFile

Defines the ssod key file (and path).

Default: <ps_path>/polsrv.key

bg.CIA

The following bg.CIA settings control the Watchdog service:

Enabled

Defines whether service is enabled. Setting it to 0 disables the Watchdog, which means that even if the service is up, it will not do anything.

Value:

- 0 - Disabled
- 1 - Enabled

Default: 1

IdlePeriod

Defines the time that the Watchdog waits before it queries the CA SSO Server if the previous known state was down.

Value: *Number of seconds*

Default: 3

Interval

Defines the time that the Watchdog waits before it queries the CA SSO Server if the previous known state was up.

Value: *Number of seconds*

Default: 10

StartTime

Defines the time the Watchdog waits after it is up, before it starts to query the CA SSO Server.

Value: *Number of seconds*

Default: 1

WDMaximumFailed

Defines how many connections must fail before the WatchDog decides that the CA SSO Server is down.

Value: *Number of connections*

Default: 3

exits

ExitsPath

Specifies the path to the exit.dlls.

Default: [None]

PasswordExits

List of password exit names. Located under ExitsPath, it contains the list of DLLs.

Default: [None]

AutogenExit

List of auto generation exit names. It is located under ExitsPath.

Default: [None]

Main

PolicyDBProviderDll

Specifies the path to the Policy DB provider plug-in.

This is a DLL on Windows and a shared object on UNIX.

Default: <CA SSO Server Install Dir>\bin\EAC_Provider_API.dll

MsgFile

The name of the CA SSO Server message file.

Value: *File name*

Default: ENU.msg

MsgPath

The path for the CA SSO Server message file.

Value: *path*

Default:<CA SSO Server Install Dir>\Lang

LoggerIniFile

Points to the name of the CA SSO Server's log INI file.

Value: *path and file name*

Default: <CA SSO Server Install Dir>\pslog.ini

UserDBProviders

A list of user data store providers' sections. The sections contain the plug-in information and settings to be loaded by the CA SSO Server.

Default: LDAP, EAC,AD, TSS, ACF2, RACF

auth.<method_name>

DLL

Specifies the full path to the authentication provider library.

Default: <CA SSO Server Install Dir>\bin\Wac<method>Auth.dll

UserDBProvider.<provider_name>

ID

Specifies the internal ID of this provider inCA Access Control.

Value:

1 - LDAP

4 - AD

8 - Eac

32 - ACF2

64 - TSS

256 - RACF

Default: [Differs according to provider]

DLL

Specifies the path to the plug-in library. This is a DLL on Windows and a shared object on UNIX.

Value: *Path to the file extension .so*

Default:<SSO Server Install Dir>\bin\<provider dll name>

AuthMap

Methodx

Defines the authentication methods available. x is the method id (number).

For example: Method5=SSO, CA SSO Authentication

Value: <authentication name>[:<legacy authentication name>], Description [:<legacy authentication name>] means it is an optional part, only authentication methods that have been renamed in the past will have it.

Default: [no default]

CacheSync

EnableDebugPrints

For internal use only.

Default: 0

QueueSize

Specifies the size of the internal queue where online updates sent by CA Access Control are stored before being processed by the SSO Server.

Default: 50

Appendix C: Configuring Authentication Agents

This section contains the following topics:

About Authentication Agent Configuration

The SSO authentication agents are configured using configuration INI files. The name of the configuration file is defined by the name of the authentication method and is of the form 'CA_methodtga.ini'. For example, the LDAP authentication agent configuration file is called CA_ldaptga.ini.

Instance specific configuration files can also be specified. These configuration files contain any instance specific settings that need to override those specified in the base configuration file. The instance specific configuration files are of the form 'CA_method_instancetga.ini'.

Instance specific configuration files are primarily intended to allow multiple instances of an authentication agent to run on the same authentication host. For example, if an authentication agent was installed using the command 'ldaptga.exe install foo', this 'foo' instance of the ldap authentication agent would first look for a configuration file called 'CA_ldaptga.ini' and then for a configuration file called 'CA_ldaptga_foo.ini'. Any configuration values specified in the latter takes precedence over the same values specified in the former.

How to Configure Multiple Instances of a Given Authentication Agent

With the exception of the Windows authentication agent, you can configure an authentication agent to run multiple instances or services on the same machine. To do this, you need to configure instance specific configuration information. These step summarize how to create multiple instances of an authentication agent:

1. Create specific versions of the authentication agents configuration file
2. Install or create multiple instances
3. Start the instance or service.

Note: Authentication agents do not prevent socket re-use on their listen ports. Do not configure multiple instances of an authentication agent to listen at the same port.

Create Instance-Specific Versions of the Authentication Agents Configuration Files

Before you create and start multiple instances of a given authentication agent, you need to create instance specific configuration files.

To create an instance-specific version of an authentication agent configuration file

- Adding the service name to the default application configuration file. For example, the default LDAP authentication agent configuration file ca_ldaptga.ini could be copied and renamed to ca_ldaptga_service1.ini.

Note: As a minimum, all listening port information must be changed. The default logging configuration file will ensure that different instances are logged to different files, based on the instance name.

Install or Create Multiple Instances

To install or create multiple instances of a given authentication agent on the one machine, simply specify different instance names using the install commands.

The following procedure takes you through creating multiple ldap authentication agent instances.

To create multiple instance of the ldap authentication agent

1. Open the Command Prompt and enter

   `ldaptga install {instance_name}`

   instance_name

   Specifies the name of the LDAP authentication agent instance. For example ca_ldaptga_service1.ini.

2. Repeat Step 1 for all further instances.

Start the Instances or Services

To start multiple instances of a given authentication agent on the one machine, simply specify different instance names using the start commands.

Important! Do not start multiple instances of an authentication agent without first creating the required instance specific configuration file(s) with alternative port information.

The following procedure takes you through starting multiple ldap authentication agent instances.

To start multiple instance of the ldap authentication agent

1. Open the Command Prompt and enter

   `ldaptga start {instance_name}`

   instance_name

   Specifies the name of the LDAP authentication agent instance. For example ca_ldaptga_service1.ini'

2. Repeat Step 1 for all further instances.

More information:

Authentication Agents (see page 267)

How to Configure Communication Modes of a Given Authentication Agent

The following are the registry keys for the different Authentication Agents that can be configured:

- Windows Authentication Agent

  `HKLM\Sofware\ComputerAssociated\SingleSignOn\CA_wintga`

- LDAP Authentication Agent

  `HKLM\Sofware\ComputerAssociated\SingleSignOn\CA_ldaptga`

- RSA Authentication Agent

  `HKLM\Sofware\ComputerAssociated\SingleSignOn\CA_rsatga`

- Certiticate Authentication Agent

  `HKLM\Sofware\ComputerAssociated\SingleSignOn\CA_certtga`

- CommMode

  A dword value with takes 0 |1 | 2

  - 0 - Compatible Mode
  - 1 - FIPS Compliant Communication Mode
  - 2 - FIPS Compatible and non-FIPS Communication Mode

If the communication mode needs to be changed after installation of the CA SSO authentication agent, the registry key needs to be changed accordingly, and the authentication agent service restarted.

For information on configuring Authentication agents in different communication modes, see the Communication (see page 299) section in this guide.

Certificate Authentication

This section lists the settings you can configure in the Certificate authentication agent. All of these settings may be edited, but you must restart the CA SSO – Certificate authentication agent Windows service for the changes to take effect.

The settings for the Certificate authentication agent are found in the 'CA_certtga.ini' configuration file located in %SSOINSTALLDIR%/cfg directory.

[Logging]

This section specifies the logging properties file.

ConfigFile

Specifies the name and location of the logging properties file. This file is used to configure logging. If this parameter is not specified logging is not configured.

Value: path and filename

Default: %SSOINSTALLDIR%/cfg/logging.properties

[Ticket]

This section specifies the values required when constructing SSO Tickets.

TicketKey

Specifies the key used to encrypt the ticket that is created by the authentication agent (after successful authentication) and sent to the CA SSO Client. The value of this option must correspond to the "Ticket Encryption key" value associated with the authentication host entry on the CA SSO Server that corresponds to the value of the AuthHostName configuration option (see the following). You can navigate to the "Ticket Encryption Key" value in CA SSO Server, using Policy Manager as follows:

Resources-> Configuration Resources -> Authentication Host -> Cert Authhost -> Authentication information -> Advanced Authentication Information

Value: *key*

Default: [no default]

AuthHostName

Specifies the name of the authentication host that is included in the SSO ticket. If you leave this blank, the computer name of the authentication host is used.

A default authentication host entry is created for each supported authentication method by default during the SSO Server installation. Each of these default entries is associated with the ps-ldap user data store and has a randomly-generated encryption key (see the TicketKey configuration option) value assigned to it.

Value: [CERT_Authhost|LDAP_Authhost|RSA_Authhost]

Default: METHOD_Authhost

[NetworkCommunication]

This section specifies network communication between the CA SSO Client and the authentication agent.

NumberOfThreads

Specifies the number of worker threads that get created for dealing with incoming client requests.

Value: number of worker threads

Default: 20

ConnectTimeout

Specifies the connect time-out value (in seconds).

Value: time in seconds

Default: 60

ReceiveTimeout

Specifies the receive time-out value (in seconds).

Value: time in seconds

Default: 60

SendTimeout

Specifies the send time-out value (in seconds).

Value: time in seconds

Default: 30

SocketBlockingMode

Specifies the socket blocking mode, zero for non-blocking, one for blocking.

Value: blocking modes

Default: 0

ListenPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 13987

ListenSSLPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 13988

IdentityFile

Defines the certificate .pem file used for TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS or Mixed Mode of installation

Value: file with .pem extension

Default: Supplied during the CA SSO Client installation in FIPS or Mixed Mode.

PrivateKeyPath

Defines the certificate .key file used for TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS or Mixed Mode of installation

Value: file with .key extension

Default: Supplied during the CA SSO Client installation in FIPS or Mixed Mode.

PubKeyP

Specifies the 'P' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyY

Specifies the 'Y' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyG

Specifies the 'G' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

SecretKeyX

Specifies the secret portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

[Parameters]

This section specifies certificate-specific configuration values.

RevocationMeth

Specifies a revocation method used for certificate validation.

Value: [FIXED_OCSP | AIA_OCSP | CRL | FIXED_OCSP+CRL | AIA_OCSP+CRL | CRLDP | CRLDP+CRL] [FIXED_OCSP_FALLBACK_TO_CRL] [FIXED_OCSP_FALLBACK_TO_CRLDP] [AIA_OCSP_FALLBACK_TO_CRL] [AIA_OCSP_FALLBACK_TO_CRLDP] [CRLDP_FALLBACK_TO_CRL]

Default: [no default]

VerifyDepth

Specifies the maximum depth of the verification chain. If this is empty, it defaults to 2.

The value of this depth affects the checking of the certificates in the trusted set. If you want the verification and checking of expiration of all the certificates in the chain, you need to specify a big enough value here and include the self signed root certificate in the trusted set.

For example, if you have a certification chain comprised of ROOT, CA and END_ENTITY, you need to set this value to 2 or more to make the verification on CA and END_ENTITY, and the expiration checking on all certificates in the chain to happen.

Value: max depth

Default: 2

CrlPollInterval

Specifies the time interval in seconds between each poll for new updates of CRL.

If the CRL is up to date:

- and CrlPollInterval is 0, then no polling of CRL file will take place until the next update due attribute of the CRL file is reached.
- and CrlPollInterval is greater than 0, then polling will take place during that interval.

If the CRL is out of date:

- and the CrlPollInterval is 0 or greater than 15, then polling will take place at 15 second intervals.
- and the CrlPollInterval is between 1-14 inclusive, then polling will take place during that interval.

Value: time in seconds

Default: 0

CrlRetrievalTimeOut

Specifies the CRL retrieval time-out value.

Value: time in seconds

Default: 60

TrustedPath

Specifies the directory where trusted certificates are located.

Value: path

Default: [no default]

TrustedNames

Specifies a list of DER-encoded certificate file names that are in the directory defined by TrustedPath. The names must be comma separated.

Value: file names

Default: [no default]

CrlFileName

Specifies the path and name of the DER-encoded CRL file. It can be a HTTP web address or a LDAP directory entry both in URL format, or a local file or a file located on a network drive. When specifying a local file or a file on a network drive, both direct path and the URL format can be used.

Value: path and file

Default: [no default]

CrlIssuerCert

Specifies the path and name of the DER-encoded CRL issuer certificate file.

Value: path and file

Default: [no default]

CrlDPIssuerCertPath

Specifies the path to the directory where the CRL issuer certificates for CRLDP revocation checking are stored.

Value: path

Default: [no default]

CrlDPIssuerCerts

Specifies the comma-separated file name list of the CRL issuer certificates that are in the directory specified by the value of CrlDPIssuerCertPath.

Value: file name

Default: [no default]

OcspSignCert

Specifies the path and name of the certificate that is used to sign requests sent to OCSP Responder.

This must be in pkcs12 format.

To obfuscate a clear password, run the command:

```
ssoencconf.exe –d "<Password_To_Be_Obfuscated>"
```

For example, ssoencconf.exe –d TestMe1

Value: path and file name

Default: [no default]

OcspSignCertPass

Specifies the password for OcspSignCert.

This value is stored in the configuration file in an obfuscated form. If configuration was done by the installer, this obfuscation will be taken care of automatically. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component.

To obfuscate a clear password, run the following command:

```
Ssoencconf.exe -d "<Password_To_Be_Obfuscated>"
```

For example, ssoencconf.exe -d TestMe1.

Value: obfuscated password

Default: [no default]

OcspResponder

Specifies the URL of the OCSP responder.

Value: URL

Default: [no default]

HttpProxy

Specifies the proxy name through which the OCSP request is sent and/or the CRL is retrieved over HTTP.

Value: time in seconds

Default: [no default]

CrlExpiryMode

Specifies the CRL modes and grace period:

1 - ignore the next update time of CRL and always use the CRL at hand to check certificates.

2 - same as mode 1. Also log a error message after a Grace Period defined by CrlGracePeriod.

3 - before Grace Period passed, use the expired CRL to check certificates; After the Grace Period, fail authentication. This is the default mode.

Value: CRL mode

Default: 3

CrlGracePeriod

Specifies the CLR grace period required by some of the CRL modes defined by CrlExpiryMod.

Value: time period

Default: 0h0 [equals to 0 Hours 0 Seconds]

Time Period Format: 0d0h0m0s [must be in sequence D | H | M | S]

D/d: Day

H/h: Hour

M/m: Minutes

S:/s Seconds

If unit is not defined, the default is seconds.

[NameMapping]

This section specifies the certificate authentication agent name mapping configuration.

MappingMethod

Specifies the method used by the certificate authentication agent to map a certificate to a SSO user name.

Value: [C|CN|DN|DNS|EMAIL|IP|L|O|OU|URI|

Default: CN

NameMappingDLLPath

Specifies the path to the name mapping DLL.

Value: path

Default: C:\Program Files\CA\eTrust SSO\Cert Agent\bin\name_mapping.dll

[Service]

This section specifies the value used when creating or running the service or daemon.

LogDir

Specifies the directory used by the service/daemon for logging any instance specific internal messages sent to standard IO. If no value is specified it will default to '%SSOINSTALLDIR%/log'

Value: path

Default: %SSOINSTALLDIR%/log

InstallDir

Specifies the directory used for the instance specific install file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/install'.

Value: path

Default: [no default]

PidDir

Specifies the directory used for the instance specific PID file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/pid'.

Value: path

Default: [no default]

RSA Authentication

This section lists the settings you can configure in the RSA authentication agent. All of these settings may be edited, but you must restart the CA SSO – RSA authentication agent Windows service (or UNIX daemon) for the changes to take effect.

The settings for the Windows authentication agent are found in the CA_rsatga.ini configuration file.

[Logging]

This section specifies the logging properties file.

ConfigFile

Specifies the name and location of the logging properties file. This file is used to configure logging. If this parameter is not specified logging is not configured.

Value: path and filename

Default: %SSOINSTALLDIR%/cfg/logging.properties

[NetworkCommunication]

This section specifies network communication between the CA SSO Client and the authentication agent.

NumberOfThreads

Specifies the number of worker threads that get created for dealing with incoming client requests.

Value: number of worker threads

Default: 20

ConnectTimeout

Specifies the connect time-out value (in seconds).

Value: time in seconds

Default: 60

ReceiveTimeout

Specifies the receive time-out value (in seconds).

Value: time in seconds

Default: 60

SendTimeout

Specifies the send time-out value (in seconds).

Value: time in seconds

Default: 30

SocketBlockingMode

Specifies the socket blocking mode, zero for non-blocking, one for blocking.

Value: blocking modes

Default: 0

ListenPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 13969

ListenSSLPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 13988

IdentityFile

Defines the certificate .pem file used for FIPS-only/TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS mode of installation.

Value: file with .pem extension

Default: Supplied during the CA SSO Client installation in FIPS-only or TLS mode.

PrivateKeyPath

Defines the certificate .key file used for FIPS-only/TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS Mode of installation

Value: file with .key extension

Default: Supplied during the CA SSO Client installation in FIPS-only or TLS mode.

PubKeyP

Specifies the ‘P’ component of the public portion of the authentication agent’s ElGamal key. This value is set by the installer or can be generated using the ‘keygen’ utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyY

Specifies the 'Y' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyG

Specifies the 'G' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

SecretKeyX

Specifies the secret portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

[Ticket]

This section specifies the values required when constructing SSO Tickets.

TicketKey

Specifies the key used to encrypt the ticket that is created by the authentication agent (after successful authentication) and sent to the CA SSO Client. The value of this option must correspond to the encryption key value associated with the authentication host entry on the CA SSO Server that corresponds to the value of the AuthHostName configuration option.

You can navigate to "Ticket Encryption Key" value in CA SSO Server, using Policy Manager as follows:

Resources-> Configuration Resources -> Authentication Host -> RSA_Authhost -> Authentication information -> Advanced Authentication Information

Value: key

Default: [no default]

AuthHostName

Specifies the name of the authentication host that is included in the SSO ticket. If you leave this blank, the computer name of the authentication host is used.

A default authentication host entry is created for each supported authentication method by default during the SSO Server installation. Each of these default entries is associated with the ps-ldap user data store and has a randomly-generated encryption key (see the TicketKey configuration option) value assigned to it.

Value: [CERT_Authhost|LDAP_Authhost|RSA_Authhost]

Default: METHOD_Authhost

[Service]

This section specifies the value used when creating or running the service or daemon.

LogDir

Specifies the directory used by the service/daemon for logging any instance specific internal messages sent to standard IO. If no value is specified it defaults to '%SSOINSTALLDIR%/log'

Value: path

Default: %SSOINSTALLDIR%/log

InstallDir

Specifies the directory used for the instance specific install file when the authentication agent is being run as a unix daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/install'.

Value: path

Default: [no default]

PidDir

Specifies the directory used for the instance specific PID file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/pid'.

Value: path

Default: [no default]

[RSA]

This section is used to configure values specific to the RSA/ACE server connection.

VAR_ACE

Specifies the location of the RSA/ACE server's 'sdconf.rec' file. Specifying a value for this parameter overrides the value of the VAR_ACE environment variable.

This parameter is only used by UNIX installations of the RSA authentication host.

Value: path

Default: [no default]

Windows Authentication

This section lists the settings you can configure in the Windows authentication agent. All of these settings may be edited, but you must restart the CA SSO – Windows authentication agent Windows service for the changes to take effect.

The settings for the Windows authentication agent are found in the CA_wintga.ini configuration file.

[Logging]

This section specifies the logging properties file.

ConfigFile

Specifies the name and location of the logging properties file. This file is used to configure logging. If this parameter is not specified logging is not configured.

Value: path and filename

Default: %SSOINSTALLDIR%/cfg/logging.properties

[General]

This section specifies values required when constructing SSO Tickets.

AuthHostName

Specifies the name of the authentication host that is included in the SSO ticket. If you leave this blank, the computer name of the authentication host is used.

A default authentication host entry is created for each supported authentication method by default during the SSO Server installation. Each of these default entries is associated with the ps-ldap user data store and has a randomly-generated encryption key (see the TicketKey configuration option) value assigned to it.

Value: [CERT_Authhost|LDAP_Authhost|RSA_Authhost]

Default: METHOD_Authhost

TicketEncryptionKey

Specifies the key used to encrypt the ticket that is created by the authentication agent (after successful authentication) and sent to the CA SSO Client. The value of this option must correspond to the encryption key value associated with the authentication host entry on the CA SSO Server that corresponds to the value of the AuthHostName configuration option (see previously in this table).

You can navigate to "Ticket Encryption Key" value in CA SSO Server, using Policy Manager as follows:

Resources-> Configuration Resources -> Authentication Host -> WIN_Authhost -> Authentication information -> Advanced Authentication Information

Value: key

Default: [no default]

[Connection]

This section specifies communication with the Windows domain controller.

PipeInstances

Specifies the number of named pipes that are created. This value affects the number of CA SSO Client connections that can simultaneous be serviced.

The maximum allowed value is 64.

Value: number of pipes

Default: 4

[Security]

This section specifies values associated with the SSL context.

IdentityFile

Specifies the name and location of a PKCS12 file containing the server's identity certificate(s).

Value: path and filename

Default: [no default]

IdentityPassword

Specifies the password to use to access the contents of IdentityFile.

This value is stored in the configuration file in an obfuscated form. If configuration was done by the installer, this obfuscation will be taken care of automatically. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component.

Value: obfuscated password

Default: [no default]

TrustFile

Specifies the name and location of a PEM file containing trusted certificates

Value: path and filename

Default: [no default]

ListenSSLPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 13988

FIPSIdentityFile

Defines the certificate .pem file used for FIPS-only or TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS Mode of installation

Value: file with .pem extension

Default: Supplied during the CA SSO Client installation in FIPS-only or TLS mode.

PrivateKeyPath

Defines the certificate .key file used for FIPS-only or TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS Mode of installation

Value: file with .key extension

Default: Supplied during the CA SSO Client installation in FIPS-only or TLS mode.

FIPSTrustFile

Specifies the name and location of a PEM file containing trusted certificates. This is chosen in case of Installation in FIPS-only mode or TLS mode

Value: path and filename

Default: [no default]

[Service]

This section specifies the value used when creating or running the service or daemon.

LogDir

Specifies the directory used by the service/daemon for logging any instance specific internal messages sent to standard IO. If no value is specified it defaults to '%SSOINSTALLDIR%/log'

Value: path

Default: %SSOINSTALLDIR%/log

InstallDir

Specifies the directory used for the instance specific install file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/install'.

Value: path

Default: [no default]

PidDir

Specifies the directory used for the instance specific PID file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/pid'.

Value: path

Default: [no default]

LDAP Authentication

This section lists the settings you can configure in the LDAP authentication agent. All of these settings may be edited, but you must restart the CA SSO – LDAP authentication agent Windows service for the changes to take effect.

The settings for the LDAP authentication agent are found in the CA_ldaptga.ini configuration file.

[Logging]

This section specifies the logging properties file.

ConfigFile

Specifies the name and location of the logging properties file. This file is used to configure logging. If this parameter is not specified logging is not configured.

Value: path and filename

Default: %SSOINSTALLDIR%/cfg/logging.properties

[NetworkCommunication]

This section specifies network communication between the CA SSO Client and the authentication agent.

NumberOfThreads

Specifies the number of worker threads that get created for dealing with incoming client requests.

Value: number of worker threads

Default: 20

ConnectTimeout

Specifies the connect time-out value (in seconds).

Value: time in seconds

Default: 60

ReceiveTimeout

Specifies the receive time-out value (in seconds).

Value: time in seconds

Default: 60

SendTimeout

Specifies the send time-out value (in seconds).

Value: time in seconds

Default: 30

SocketBlockingMode

Specifies the socket blocking mode, zero for non-blocking, one for blocking.

Value: blocking modes

Default: 0

ListenPort

Specifies the port number on which the authentication agent listens for client requests.

Value: port number

Default: 17979

ListenSSLPort

Specifies the port number on which the authentication agent will listen for client requests.

Value: port number

Default: 17980

IdentityFile

Defines the certificate .pem file used for FIPS-only or TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS mode of installation

Value: file with .pem extension

Default: Supplied during CA SSO Client installation in FIPS-only or TLS mode

PrivateKeyPath

Defines the certificate .key file used for FIPS-only or TLS communications between the CA SSO Client computer and the Certificate authentication Agent computer. This is mandatory. This value is entered during installation in FIPS-only or TLS mode of installation

Value: file with .key extension

Default: Supplied during CA SSO Client installation in FIPS or Mixed Mode

PubKeyP

Specifies the 'P' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyY

Specifies the 'Y' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

PubKeyG

Specifies the 'G' component of the public portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

SecretKeyX

Specifies the secret portion of the authentication agent's ElGamal key. This value is set by the installer or can be generated using the 'keygen' utility provided with each authentication agent.

Value: an ElGamal key value

Default: [no default]

[Ticket]

This section specifies the values required when constructing SSO Tickets.

TicketKey

Specifies the key used to encrypt the ticket that is created by the authentication agent (after successful authentication) and sent to the CA SSO Client. The value of this option must correspond to the encryption key value associated with the authentication host entry on the CA SSO Server that corresponds to the value of the AuthHostName configuration option.

You can navigate to “Ticket Encryption Key” value in CA SSO Server, using Policy Manager as follows:

Resources, Configuration Resources > Authentication Host > LDAP_Authhost > Authentication information > Advanced Authentication Information

Value: key

Default: [no default]

AuthHostName

Specifies the name of the authentication host that is included in the SSO ticket. If you leave this blank, the computer name of the authentication host is used.

A default authentication host entry is created for each supported authentication method by default during the SSO Server installation. Each of these default entries is associated with the ps-ldap user data store and has a randomly-generated encryption key (see the TicketKey configuration option) value assigned to it.

Value: [CERT_Authhost|LDAP_Authhost|RSA_Authhost]

Default: METHOD_Authhost

[Service]

This section specifies the value used when creating or running the service or daemon.

LogDir

Specifies the directory used by the service/daemon for logging any instance specific internal messages sent to standard IO. If no value is specified it defaults to '%SSOINSTALLDIR%/log'

Value: path

Default: %SSOINSTALLDIR%/log

InstallDir

Specifies the directory used for the instance specific install file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/install'.

Value: path

Default: [no default]

PidDir

Specifies the directory used for the instance specific PID file when the authentication agent is being run as a UNIX daemon (currently only relevant for RSA). If no value is specified it defaults to '%SSOINSTALLDIR%/pid'.

Value: path

Default: [no default]

[LDAPConnection]

This section is used to configure the connection to the LDAP directory.

AuthMethod

Specifies the method used for LDAP authentication. Possible value are 'Compare' (see description of IdAttribute option further below, for details) and 'Bind' (default). When 'Bind' method is used, LDAP authentication agent verifies the validity of the provided credentials using bind attempt.

Bind can be used regardless of LDAP server platform but it can take much more time than Compare. In certain cases Bind will be a method of choice, simply because of the directory restrictions. For example, in Active Directory the 'userpassword' attribute is read-only, and is not suitable for use with 'Compare' LDAP authentication method.

Value: [compare | bind]

Default: Bind

IdAttribute

Specifies the name of the attribute in the directory schema whose value is compared to the username provided during the authentication process.

This parameter only has an effect when the AuthMethod parameter is 'Compare'.

Value: directory scheme attribute

Default: [no default]

PolicyName

Specifies the name of the LDAP policy file.

The LDAP policy file must be located in the '%SSOINSTALLDIR%/cfg' directory and must have a '.ini' file extension. Do not include the extension when specifying the policy file name for this parameter.

Value: filename

Default: ldapPolicy

OfflineTimeout

Specifies the time in seconds that the LDAP authentication server stays marked as offline after the LDAP authentication agent fails to communicate with it.

If not specified it defaults to 120 seconds.

Value: number of seconds

Default: 120

ConnectionLifetime

Specifies the maximum time in seconds that the connection made by the LDAP authentication agent to the LDAP authentication server is maintained. If the information needed by the agent is obtained before this period elapses, the agent terminates the connection.

Value: time in seconds

Default: 3600

StandbyConnections

Specifies the minimum number of connections to the group (pool) of LDAP authentication servers (defined in tga_ldapPolicy.ini)

The number of connections maintained in the pool is kept within the range of StandbyConnections and MaxConnections. A minimum number of standby connections is maintained, and increased to the maximum number as required. When reducing the number of connections (due to not having been used recently) the standby is used as the minimum.

Value: number of connections

Default: 5

MaxConnections

Specifies the maximum number of connections that can be opened to the group (pool) of LDAP authentication servers defined in ldapPolicy.ini.

See also StandbyConnections.

Value: number of seconds

Default: 10

UserNamePrefix

Specifies text before the user name that is not put in the SSO ticket during authentication. For example, set the UserNamePrefix to cn= to remove the first three characters from the following DN:

cn=Juanita Perez, ou=CompanyName

This option can be useful if the type of the user datastore on the CA SSO Server does not correspond with the user datastore used during authentication. For example, authenticating using credentials of a user from CA Directory and then verifying the ticket against a user in CA Access Control-type data store.

Value: text

Default: [no default]

UserNameSuffix

Specifies the text after the user name that will not be put in the SSO ticket during authentication. For example, set the UserNameSuffix to , ou=CompanyName to remove the last sixteen characters from the following DN:

cn=Juanita Perez, ou=CompanyName

See comment for UserNamePrefix option above for more detail.

UserNamePrefix and UserNameSuffix configuration parameters can both be omitted, in which case user's entire distinguished name (DN) will be stored in the ticket generated by the LDAP authentication agent.

Value: text

Default: [no default]

IsActiveDirectoryAware

Specifies whether the LDAP authentication agent supports extended Active Directory-specific functionality. This extended functionality provides support for Active Directory password policies, more detailed error messages, and the ability to change the user's password. Set this to yes only if the target directory is Active Directory and AuthMethod is set to bind. If no value is specified, this setting defaults to no.

Value: [yes|no]

Default: no

DisplayFailureMessageDetail

Specifies whether the LDAP authentication agent displays details as to why a given authentication or change password attempt failed. This setting is only used if IsActiveDirectoryAware is set to yes. If not specified, this setting defaults to yes.

Value: [yes|no]

Default: yes

LDAP Authentication Name Mapping and Failover

The following settings can be configured in ldapPolicy.ini file that gets installed to the same directory as the LDAP authentication agent Windows service executable.

[NameMapping]

One or more name mappings must be defined, contained in sections named [NameMapping<index>], with the first index value being 0 and consequent ones being in increments of 1. If you have sections [NameMapping0] and [NameMapping7], the latter won't be read in or used.

When specifying multiple name mappings it is the administrator's responsibility to ensure that no given SSO user name maps to multiple user's in the directory.

While selecting the method for *name mapping*, consider that *search,* although more flexible than *substitution,* can also be much more time consuming.

StaticName

Specifies the distinguished name (DN) of the user, given the value entered by the user in the authentication dialog. The format string must generally be of the form: <attribute name>=%s,<the remainder of the user distinguished name value>.

Note: Only if NameMapping=Substitution.

BaseDN

Specifies the Base DN for the LDAP search.

Note: Only if NameMapping=Search

Filter

Specifies the filter for the LDAP search, that usually takes the form of the format specifier string <attribute name>=%s. Filter needs to be designed in such a way so that it could be used to uniquely identify a user below a base DN (see BaseDN parameter above).

Note: Only if NameMapping=Search

Scope

Specifies the scope for the LDAP search if the NameMapping type 'Search' is used. Select one of three possible values:

1- Subtree

2 - OneLevel

3 - Object

Note: Only if NameMapping=Search

[Primaries]

The LDAP authentication agent can distribute processing between LDAP servers. You can define two groups of LDAP servers: Primary and Secondary. Primary servers can be configured using the installation wizard, and both Primary and Secondary servers can be configured in the ldapPolicy.ini file post-installation.

The LDAP authentication agent always tries to bind to the servers from the Primary group first and if it fails, it binds to the servers defined in the Secondary group.

Within each group of servers, LDAP server definitions consist of two parts:

- LDAPHost<index>=<hostname>[<port number>][/bias_value]

  The first index value being 0 and consequent ones in increments of 1.

  Port number and bias value are optional, with the default values being 389 and 1 respectively. Bias value can be used to configure loadbalancing between the servers within the group (for example, LDAP server with a bias value of 50 is fifty times as likely to be chosen when decision is made to which server to connect).

- [<group type>.LDAPHost<index>]

  A section containing the details required for the initial administrative bind, when a connection to the server is established for the first time.

Note: At least one LDAP server must be defined in the Primaries group, in order for the LDAP authentication agent to initialize successfully.

LDAPHost0

Specifies the name of the authentication host.

Value: *computer name*

[Primaries.LDAPHost0]

AuthenticationLevel

Specifies the level of authentication to use when connecting to the hosts LDAP directory:

- Anonymous
- Simple
- SSL

LoginName

Specifies the login name to use for simple authentication.

Note: Only if AuthenticationLevel= Anonymous.

Password

Specifies the password to use for simple authentication. This value is stored in the configuration file in an obfuscated form. If configuration was done by the installer, this obfuscation is taken care of automatically. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component.

Note: Only if AuthenticationLevel= Anonymous.

Keystore

Specifies he name and path of a PEM file key store to be used when SSL authentication is used.

Note: Only if AuthenticationLevel=SSL

For information about the Secondaries, see the settings descriptions for the Primaries.

If any of the parameters required for non-*Anonymous* authentication are missing, the LDAP authentication agent fails to start. If any of the parameters required for SSL communication are missing, the LDAP authentication agent fails to start.

Appendix D: Configuring the Password Synchronization Agent

The Password Synchronization Agent settings are stored in two places. The Password Filter settings are stored in the registry. The Password Exit settings are stored in the file WinPsaExit.ini.

This section contains the following topics:

Password Exit Configuration

The Password Exist is the Password Synchronization Agent that is installed on the CA SSO Server computer to monitor password changes.

Formatting Rules for the Configuration Files

Certain formatting rules apply to all values in the configuration files.

Separating values

Separate values with spaces or tabs.

Specifying time periods

Time periods are specified using the following syntax:

days d *hours* h *minutes* m *seconds* s

You can specify exact times, for example:

1d 5h 30m 0s

You can use any of these times. For example:

```
3h
```

A number without an annotation defaults to seconds. For example, the following is parsed as 180 seconds:

```
180
```

Paths or strings with spaces

Quotation marks are not required for paths or strings that contain spaces.

Multiple values

If the first value fails, the second value is used instead and so on through the list.

Sections of the Password Exit Configuration File WinPSAExit.ini

Usually, there is no need to change these parameters after installation. In some cases, however, you might want to change some of them.

These are the four main sections in the WinPSAExit.ini file:

Logging

This section lets you specify the location of the logging configuration file.

LDAPConnection

This section lets you configure communication parameters for the LDAP connections made to the DCs.

Synchronization

This section lets you configure the synchronization application and data store parameters.

LDAPSearch

This section lets you configure LDAP Search parameters used if the configured user data store is not an Active Directory data store.

WinPSAExit.ini Configuration

[Logging]

ConfigFile

Specifies the path to the location of the logging.properties file. This information is entered during installation.

Value: file and path

Default: %SSOINSTALLDIR%/cfg/logging.properties

[LDAPConnection]

DomainControllers

Specifies a comma separated list of DC hostnames and ports for LDAP connections over SSL. This information is entered during installation.

Note: The hostnames must match those in the machine certificates used by the DCs to authenticate SSL communication.

For example:

```
dc01.domain.com:636, dc02.domain.com:636
```

Value: *host and domain name:port number*

Default: [no default]

PSAUserDN

Specifies the fully qualified DN of the PSA user used by the Password Exit to perform password synchronization operations on AD. This user must have ResetPassword privileges. This information is entered during installation.

Value: *user distinguished name*

Default: [no default]

PSAUserPassword

Specifies the obfuscated password of the user identified by PSAUserDN. This information is entered during installation. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component.

Value: *obfuscated password*

Default: [no default]

NumberOfRetries

Specifies the number of times to retry connection to a DC.

Default: 0

Retry Interval

Specified the amount of time to wait between connection retries.

Default: 10 (seconds)

MaxConnections

Specifies the maximum number of LDAP connections in the connection pool.

Default: 10

StandbyConnections

Specifies the minimum number of LDAP connections in the connection pool.

Default: 2

ConnectionLIfetime

Specifies the maximum time that an LDAP connection is maintained.

Default: 1h

OfflineTimout

Specifies the maximum amount of time that the Password Exit waits for a successful connection to a DC before it marks it as offline.

Default: 2m

NetworkTimeout

Specifies the time before the Password Exit aborts while trying to connect to the LDAP directory.

Default: 5s

OperationTimeout

Specifies the time before the Password Exit aborts an operation such as search of password updates in the LDAP directory.

Default: 1m

Keystore

Specifies the path to a trust file (PEM format) to verify the authenticity of DC machine certificates when performing SSL handshakes. This is usually the domain certification authority's certificate, however this depends on your PKI implementation. This value is optional and can be entered during installation.

Default: %SSOINSTALLERDIR/cert

SslIgnoreVerifyErrors

Specifies whether to ignore SSL handshake verification errors.

Value: [yes|no]

Default: no

SslVerifyDepth

Specifies verification depth limit of certification chain.

Default: 255

SslCheckHostname

Specifies whether to force hostname and certification subject match.

Value: [yes|no]

Default: yes

SslMatchHostFQDN

Specifies whether to force matches of fully qualified domain names, for example, whether to allow foo versus foo.domain.com matches.

Value: [yes|no]

Default: no

[Synchronization]

UserDataStore

Specifies the name of the user data store to synchronize with AD. This information is entered during installation.

Default: [no default]

IsActiveDirectory

Specifies a Boolean value denoting whether the user data store identified by UserDataStore is an AD data store. This information is entered during installation.

Default: [no default]

SyncAppls

Specifies a comma separated list of applications to synchronize with Active Directory. This information is entered during installation.

Default: [no default]

[LDAPSearch]

Specifies the LDAPSearch section contains parameters for mapping users between a non-AD user data store used by the CA SSO Server and AD. If the Synchronization parameter IsActiveDirectory is set to 1, the following parameters are ignored by the PSA exit

BaseDN

Specifies the base DN for the LDAP search. This is set during installation.

Value:

Default: [no default]

Scope

Specifies the scope for the LDAP search. The value of this parameter must be one of Subtree, OneLevel, Object.

Set at installation.

Value:

Default: [no default]

More information:

Maintenance (see page 263)

Password Filter Configuration

The Password Filter is the Password Synchronization Agent that is installed on the Domain Controller to monitor password changes.

Windows Registry Key Names

Usually, there is no need to change these parameters after installation. In some cases, however, you might want to change some of them.

There are four Windows Registry Keys that let you configure the Password Exit:

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\PSA

This lets you configure the communication mode.

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\PSA\Logging

This lets you configure the logging parameters.

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\PSA\NetworkCommunication

This lets you configure the network communication parameters.

HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\SingleSignOn\PSA\PolicyServer

This lets you configure the CA SSO Server-related parameters.

Note: When changing the registry, unless specified otherwise, you will have to reboot the computer for the changes to take effect.

Windows Registry Keys for the Password Filter

[CommMode]

CommMode

Specifies a dword value which takes 0 or 1.

- 0 - Compatible Mode
- 2 - FIPS-only Mode

If the communication mode needs to change after the PSA Filter has been installed, this registry key must be changed accordingly, and the machine must be restarted.

[Logging]

ConfigFile

Specifies the location of the Password Filter configuration file.

Value: path and file

Default: %InstallDir%\cfg\logging.properties

[NetworkCommunication]

The networkCommunication section contains tokens that define communication between the agent and the server

ConnectTimeout

Specifies how long before the Password Filter stops trying to connect to the SSO Server.

Default: 120s

SendTimeout

Specifies how long before the Password Filter stops trying to send data to the SSO Server.

Default: 60s

ReceiveTimeout

Specifies how long before the Password Filter stops trying to receive data from the SSO Server.

Default: 60s

SendBufferSize

Specifies the maximum size of the send buffer.

Value: number of bytes

Default: 10240

ReceiveBufferSize

Specifies the maximum size of the receive buffer.

Value: number of bytes

Default: 262144

[PolicyServer]

The PolicyServer section contains tokens related to the SSO Servers.

Servers

Specifies the servers. If the list contains more than one server, the server names must be delimited with a comma, a space, or a tab. This value is set during installation. The following are examples:

srv1,srv2,losangeles,Chicago

srv1 srv2 losangeles Chicago

Default: [no default]

AdminUser

Specifies the user name with administrator permission on the SSO Server. This value is set during installation.

Default: [no default]

AdminPassword

Specifies the obfuscated password of AdminUser. To alter this value manually at a later stage, use the Obfuscation Tool 'ssoencconf.exe' supplied with the product. You can find this tool in the bin directory for this component. This value is set during installation.

Default: [no default]

UserDataStore

Specifies the name of the user data store to use on the SSO Server. This value is set during installation.

Default: [no default]

SearchFilter

Specifies the search filter used to map users from Active Directory to the user data store identified by the UserDataStore token. This value is set during installation.

Default: [no default]

SyncAppl

Specifies the name of the application used for password synchronization. For more information, see Define the Synchronization Application(s) section in this guide. This value is set during installation. We recommended you use the domain name.

Default: [no default].

PollConfiguration

Specifies whether the settings in [PolicyServer] and [NetworkCommunication] sections need to be refreshed during synchronization operations. Note: This may cause slight performance degradation if enabled.

yes: Poll the configuration settings

no: Don't poll the configuration settings

Note: If this value is changed from No to Yes you must reboot the computer before the setting takes effect.

Default: no

PollInterval

Specifies the amount of time that must elapse before the Password Filter reads the settings from the Windows Registry.

Default: 60s

KeyPath

Specifies the path of the public keys used for SSO authentication of AdminUser.

Default: %InstallDir%\bin

MsgFilePath

Specifies the location of SSO message file.

Default: %InstallDir%\enu.msg

Appendix E: Configuring Multiwrite Replication

This section contains the following topics:

About the Scenario

This appendix uses the following example information:

Server Farm 1

- Machine 1 Host Name: Farm1_Machine1 (hub)
- Machine 2 Host Name: Farm1_Machine2

Server Farm 2

- Machine 1 Host Name: Farm2_Machine1 (hub)
- Machine 2 Host Name: Farm2_Machine2

CA Directory Post-Installation State

You must configure the file on the CA SSO Server that that contains all DSAs that this machine knows about. This file is called (PS_Servers.dxg). A DSA is a CA Directory file.

After installation this is only the machines in the same server farm:

- Farm1_Machine1 knows about Server Farm 1 (Farm1_Machine1 & Farm1_Machine2)
- Farm1_Machine2 knows about Server Farm 1 (Farm1_Machine1 & Farm1_Machine2)
- Farm2_Machine1 knows about Server Farm 2 (Farm2_Machine1 & Farm2_Machine2)
- Farm2_Machine2 knows about Server Farm 2 (Farm2_Machine1 & Farm2_Machine2)

Example PS_Servers.dxg for Farm1_Machine1:

```
source "../knowledge/PS_Farm1_Machine2.dxc";
source "../knowledge/PSTD_Farm1_Machine2.dxc";

source "../knowledge/PS_Farm1_Machine1.dxc";
source "../knowledge/PSTD_Farm1_Machine1.dxc";
```

When you install the CA SSO Server, it has a knowledge file. Each knowledge file has multi-write enabled but no multi-write-group defined. This file is called PS_*computer name*.dxc

With this configuration, each configured DSA sends updates to all DSAs it knows about.

Example PS_Farm1_Machine1.dxc on Farm1_Machine1:

```
set dsa PS_Farm1_Machine1 =
{
prefix        = <o "PS">
dsa-name      = <o "PS"><cn PS_Farm1_Machine1>
dsa-password  = "secret"
address       = tcp "Farm1_Machine1.ca.com" port 13389,
                tcp "localhost" port 13389
disp-psap     = DISP
cmip-psap     = CMIP
snmp-port     = 13389
console-port  = 13379
ssld-port     = 1112
auth-levels   = anonymous, clear-password
max-idle-time = 60
dsa-flags     = multi-write
trust-flags   = allow-check-password, trust-conveyed-originator
};
```

Example PS_Farm1_Machine2.dxc on Farm1_Machine1:

```
set dsa PS_Farm1_Machine2 =
{
prefix        = <o "PS">
dsa-name      = <o "PS"><cn PS_Farm1_Machine2>
dsa-password  = "secret"
address       = tcp "Farm1_Machine2.ca.com" port 13389,
                tcp "localhost" port 13389
disp-psap     = DISP
cmip-psap     = CMIP
snmp-port     = 13389
console-port  = 13379
ssld-port     = 1112
auth-levels   = anonymous, clear-password
max-idle-time = 60
dsa-flags     = multi-write
trust-flags   = allow-check-password, trust-conveyed-originator
};
```

How to Set Up Multiwrite Replication

To set up Multiwrite replication, you must perform these steps:

1. Configure the Multiwrite groups
2. Copy the knowledge files to the other server farms
3. Add the knowledge files to the group knowledge files and assign the hub order
4. Verify the configuration

Configure Multiwrite Groups

Each PS knowledge file (PS_<HOSTNAME>.dxc) needs to have a multi-write group defined.

For example, all servers in "Server Farm 1" will be assigned to group "Farm1" and "Server Farm 2" to group "Farm2".

This change needs to be done to **all** PS knowledge files on **all** machines, because the same set of configuration files need to exist on all machines.

Note: The "multi-write-group" setting must occur immediately before the "dsa-flags" setting.

Here is an example change to PS_Farm1_Machine1.dxc on Farm1_Machine1:

```
set dsa PS_Farm1_Machine1 =
{
prefix        = <o "PS">
dsa-name      = <o "PS"><cn PS_Farm1_Machine1>
dsa-password  = "secret"
address       = tcp "Farm1_Machine1.ca.com" port 13389, tcp "localhost" port 13389
disp-psap     = DISP
cmip-psap     = CMIP
snmp-port     = 13389
console-port  = 13379
ssld-port     = 1112
auth-levels   = anonymous, clear-password
max-idle-time = 60
multi-write-group = "Farm1"
dsa-flags     = multi-write
trust-flags   = allow-check-password, trust-conveyed-originator
};
```

This change would be done to eight files in total:

- PS_Farm1_Machine1.dxc on Farm1_Machine1 (add multi-write-group = "Farm1")
- PS_Farm1_Machine2.dxc on Farm1_Machine1 (add multi-write-group = "Farm1")
- PS_Farm1_Machine1.dxc on Farm1_Machine2 (add multi-write-group = "Farm1")
- PS_Farm1_Machine2.dxc on Farm1_Machine2 (add multi-write-group = "Farm1")
- PS_Farm2_Machine1.dxc on Farm2_Machine1 (add multi-write-group = "Farm2")
- PS_Farm2_Machine2.dxc on Farm2_Machine1 (add multi-write-group = "Farm2")
- PS_Farm2_Machine1.dxc on Farm2_Machine1 (add multi-write-group = "Farm2")
- PS_Farm2_Machine2.dxc on Farm2_Machine1 (add multi-write-group = "Farm2")

Copy the Knowledge Files to other Server Farms

Copy all PS_<MACHINENAME>.dxc files from each server farm to all other server farms as shown below:

- PS_Farm1_Machine1.dxc to Farm2_Machine1
- PS_Farm1_Machine2.dxc to Farm2_Machine1
- PS_Farm1_Machine1.dxc to Farm2_Machine2
- PS_Farm1_Machine2.dxc to Farm2_Machine2
- PS_Farm2_Machine1.dxc to Farm1_Machine1
- PS_Farm2_Machine2.dxc to Farm1_Machine1
- PS_Farm2_Machine1.dxc to Farm2_Machine2
- PS_Farm2_Machine2.dxc to Farm2_Machine2

Now, all servers must have the four PS knowledge files.

Add the CA SSO Server Knowledge Files to the Group

The files have been copied to the machine, but they still need to be referenced in the PS group knowledge files.

The order that the PS knowledge files are listed defines which machine is the hub:

- The first PS knowledge file loaded for each multi-write group becomes the hub
- The second PS knowledge file loaded for each mulit-write becomes the hub when the first is offline (and so on)

Example change to PS_Servers.dxg on Farm1_Machine1:

```
source "../knowledge/PS_Farm1_Machine1.dxc";
source "../knowledge/PS_Farm2_Machine1.dxc";

source "../knowledge/PS_Farm1_Machine2.dxc";
source "../knowledge/PS_Farm2_Machine2.dxc";

source "../knowledge/PSTD_Farm1_Machine1.dxc";
source "../knowledge/PSTD_Farm1_Machine2.dxc";
```

In the above, the "hub" for each multi-write group is listed first. Then each "peer" is listed. The PSTD is not replicated between multi-write groups so it remains unchanged.

When updates occur on Farm1_Machine1 the following shows the order of the replication:

- Farm1_Machine1 to Farm2_Machine1 to Farm2_Machine2 to Farm1_Machine2

If Farm2_Machine1 goes offline, Farm2_Machine2 becomes the hub the following shows the order of the replication:

- Farm1_Machine1 to Farm2_Machine2 to (Farm2_Machine1 when online) to Farm1_Machine2

Verify the Multiwrite Configuration

To verity the configuration, you must follow these steps:

1. Run "ttermpro.exe" on each SSO Server machine.

   This file is available from <CA Directory installation>/ttermpro/ttermpro.exe
2. Connect to localhost, port 13379 (the local PS DSA)
3. On each machine type the following command to enable a detailed level of trace logging:

   ```
   trace x500;
   ```

4. Add a user to the "ps-ldap" data store on Farm1_Machine1, the following output must be observed:

Farm1_Machine1 (some events omitted):

LDAP ADD-ENTRY-REQ # the user is added locally

(Remote) [Farm1_Machine2] DSP BIND-REQ # bind to other member of this server farm

(Remote) [Farm2_Machine1] DSP BIND-REQ # bind to hub of the other server farm

(Remote) [Farm1_Machine2] DSP ADD-ENTRY-CONFIRM # the user is added

(Remote) [Farm2_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

Farm1_Machine2:

[Farm1_Machine1] DSP BIND-CONFIRM # incoming bind from other member of this server farm

[Farm1_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

Farm2_Machine1:

[Farm1_Machine1] DSP BIND-CONFIRM # incoming bind from the hub of the other server farm

[Farm1_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

(Remote) [Farm2_Machine2] DSP BIND-REQ # bind to other member of this server farm

(Remote) [Farm2_Machine2] DSP ADD-ENTRY-CONFIRM # the user is added

Farm2_Machine2:

[Farm2_Machine1] DSP BIND-CONFIRM # incoming bind from the hub of this server farm

[Farm2_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

In the above, Farm2_Machine2 does not receive the update from the originating machine, but the local hub instead.

The above can be repeated with the hub (Farm2_Machine1) offline, to demonstrate failover:

Farm1_Machine1 (some events omitted):

LDAP ADD-ENTRY-REQ # the user is added locally

(Remote) [Farm1_Machine2] DSP BIND-REQ # bind to other member of this server farm

(Remote) [Farm2_Machine1] DSP BIND-REQ # bind to hub of the other server farm

(Remote) [Farm1_Machine2] DSP ADD-ENTRY-CONFIRM # the user is added

(Remote) [Farm2_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

Farm1_Machine2:

[Farm1_Machine1] DSP BIND-CONFIRM # incoming bind from other member of this server farm

[Farm1_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

Farm2_Machine1

Farm2_Machine2:

[Farm1_Machine1] DSP BIND-CONFIRM # incoming bind from the hub of the other server farm

[Farm1_Machine1] DSP ADD-ENTRY-CONFIRM # the user is added

(Remote) [Farm2_Machine1] DSP BIND-REQ # bind to other member of this server farm

** ALARM **: MW: Hub DSA 'Farm2_Machine1' cannot be contacted

Appendix F: Interpreting Error Messages

Error messages are received in the context of the calling API function, so each error message does not have one explicit meaning, but must be interpreted. This appendix lists error messages, component codes, and the detailed error codes you may receive. Use these lists to interpret any error messages you may receive.

This section contains the following topics:

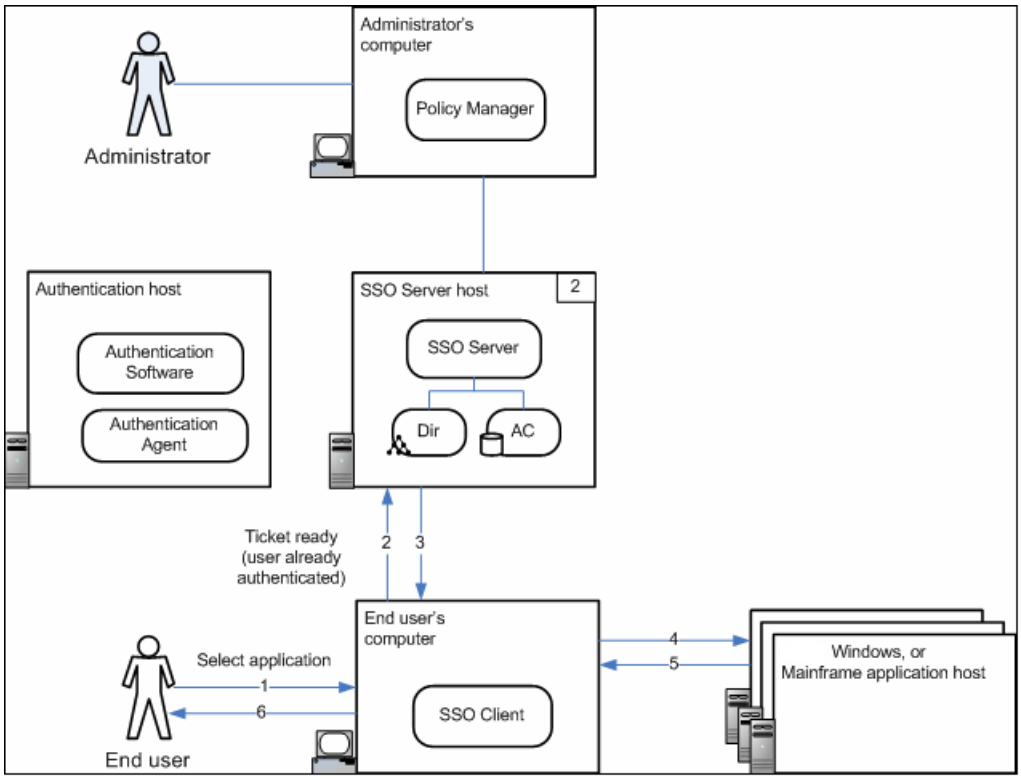
Error Message Flow Diagrams

Some of the error messages listed below include a reference to a step in one of the following flow diagrams. For example, a reference to flow diagram 2.3 means the third numbered step in the second flow diagram.

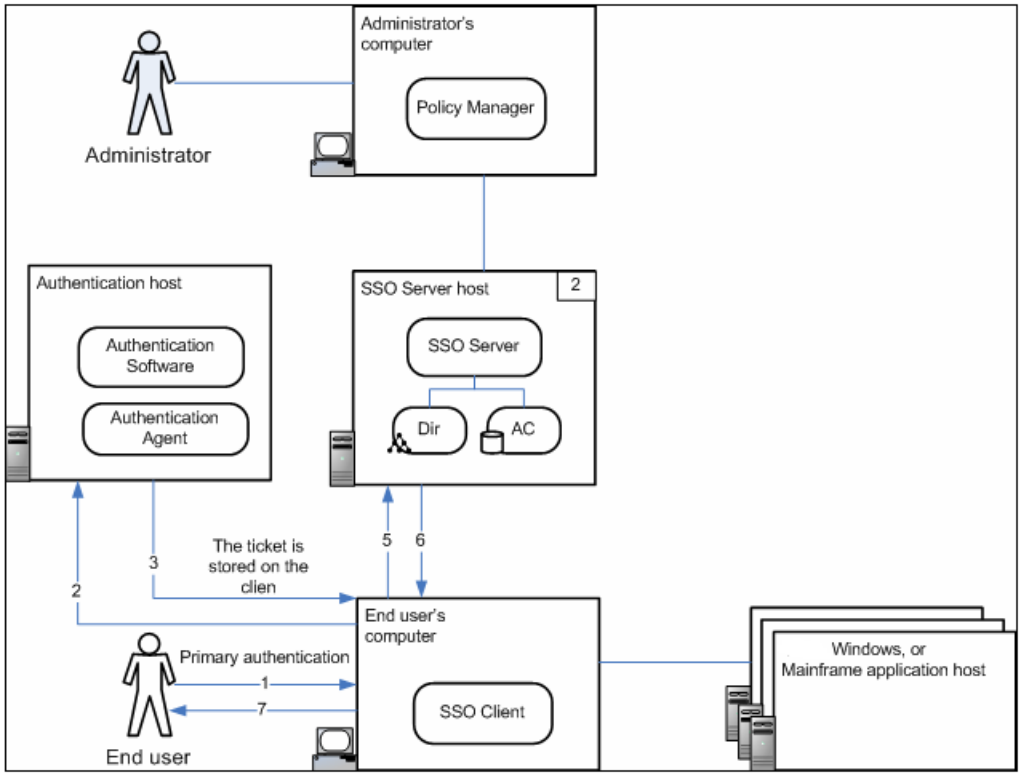
Error Message Flow Diagrams

This section lists the error messages you may receive, including a description of the error message and any corrective actions you can take.

Flow Diagram 1



Flow Diagram 2



Error Messages

Error Message: "Can't connect to host <host name>"

Number

0x010000

Component

CA SSO Client

Communication Flow diagram Reference

2.2, 2.4

Cause

CA SSO Server is down whilst either trying to logon or access something.

Resolution

Restart the CA SSO Server service on the Server computer

Cause

A valid Authentication host name is not entered into the Auth.ini file

Resolution

Enter the name of the server computer in the appropriate 'authhost' field

Cause

The authentication host computer is not running

Resolution

Start the authentication service on the authentication host

Error Message: "Host <hostname> is unknown"

Number

0x01

Component

CA SSO Client

Communication Flow diagram reference

2.2

Cause

The hostname specified in the Auth.ini file cannot be reached

Resolution

Verify that the hostname is correctly spelt and can be contacted via TCP/IP

Error Message: "Host <hostname> is unreachable"

Number

0x02

Component

CA SSO Client

Communication Flow diagram reference

2.2

Cause

The hostname specified in the Auth.ini file cannot be reached

Resolution

Verify that the host computer is spelt correctly and that the computer is booted and can be contacted via TCP/IP

Error Message: "Unable to connect to CA SSO Server Token Directory"

Number

Number not known

Component

Policy Manager

Cause

Unknown

Resolution

Restart the DSAs on the CA SSO Server computer

Error Message: "Communication failure with host <server name> <extended information>"

Component

CA SSO Client

Cause

Computer does not have TCP/IP installed

Resolution

Install TCP/IP networking on the Server computer

Cause

Port number is already in use

Resolution

Change the port number range being used by the CA SSO Client

Error Message: "Communication aborted with host <host name>"

Number

0x01010402

Component

CA SSO Client

Communication Flow diagram Reference

2

Cause

Communication timeout with CA SSO Server

Resolution

Error Message: "Application <application name> not found <server name>"

Number

0x01020301

Component

CA SSO Client

Communication Flow diagram Reference

1.2

Cause

Application list is outdated and the CA SSO Client is trying to access an application that no longer exists in the CA SSO Server database

Resolution

Refresh application list

Error Message: "User <user name> not found <server name>"

Number

0x01020401

Component

CA SSO Client

Communication Flow diagram Reference

2.2

Cause

CA Access Control not running while trying to logon

Resolution

Start the CA Access Control daemons

Cause

User does not exist

Resolution

Add the user to the database

Error Message: "Password has expired"

Number

0x0600

Component

CA SSO Client

Communication Flow diagram Reference

Cause

The user's password has expired

Resolution

Through either the Client or the Policy Manager, change the user's password

Error Message: "Can't change password for application <application name>"

Number

0x0701

Cause

Password policy states that the user cannot change the password.

Resolution

Use the Policy Manager to change the settings associated with that user on the CA SSO Server to allow them to change their password

Error Message: "Password Quality Check failed <extended information>"

Number

0x040000 (0x0101, 0x0201, 0x0301, 0x0401, 0x0501, 0x0601, 0x0701, 0x0801, 0x0901)

Component

CA SSO Client

Communication Flow diagram Reference

1.3

Cause

Password does not meet the password policy minimum requirements for the number of a particular type of character (numerical, alpha, special character, uppercase and lowercase)

Resolution

Check the minimum standards as specified in the password policy and re-enter a password

Cause

Too many repetitive characters in the password

Resolution

Check the minimum standards as specified in the password policy and re-enter a password

Cause

Password has been used previously, which does not meet the password policy requirements

Resolution

Check the minimum standards as specified in the password policy and re-enter a password

Number

0x01040a01

Error Message: "Cannot set an empty new password"

Component

CA SSO Server

Communication Flow diagram Reference

2.3

Cause

Empty password not allowed for application.

Resolution

Supply a password

Error Message: "Database Engine Failure"

Number

0x050000

Cause

Trying to access information stored in the database while CA Access Control is shutdown.

Resolution

Restart CA Access Control

Error Message: "Database Engine is not operational"

Number

0x0101

Cause

Trying to access information store in the database while CA Access Control is shutdown.

Resolution

Restart CA Access Control

Error Message: "Invalid Ticket" : <additional mistmatch>

Number

0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08

Cause

Ticket Checksum Mismatch - The key for ticket encryption on the the CA SSO Server does not match that of the authentication host

Resolution

Change either the encryption key of the authentication agent in its configuration .INI file, or use the Policy Manager to change the key for the authentication host record on the CA SSO Server.

Error Message: "User has no password yet"

Number

0x0300

Component

CA SSO Server

Communication Flow diagram Reference:

2.4, 2.5

Cause

User password has not been set for the authentication method used

Resolution

Create a user password via the Policy Manager for that user.

Error Message: "Password mismatch for user <user name>"

Number

0x01060400

Component

CA SSO Client

Communication Flow diagram Reference:

2.2

Cause

User has supplied an incorrect password

Resolution

Supply correct password or change password using Policy Manager if necessary

Error Message: "User not allowed to use this Authentication Method <authentication method>"

Number

0x01070201, 0x0101

Cause

The user has not been assigned rights to use this authentication method

Resolution

Within the Policy Manager assign the authentication method in question to the list of allowed authentication methods for this user.

Error Message: "Can't login now"

Number

Number not known

Cause

User's account has been barred on the Server

Resolution

On the CA SSO Server, uncheck the box that disables the user's account.

Cause

User does not exist when authenticating to a CA SSO Server

Resolution

Create the user on the CA SSO Server

Error Message: "Can't access application"

Number

0x01070401

Component

CA SSO Client (GINA)

Communication Flow diagram Reference:

1.2, 2.4

Cause

Application list is outdated and the CA SSO Client is trying to launch an application which they no longer have access rights for.

Resolution

Refresh application list

Cause

User has no access to an application that matches the logon domain

Resolution

Give user access rights to the appropriate application (<DOMAIN NAME> or NT_LOCAL_LOGON)

Error Message: "Unauthorized database update <additional information>"

Number

0x0601

Component

Communication Flow diagram Reference:

Cause

Resolution

Error Message: "User can't change loginid for application <application name>"

Number

0x0701

Cause

User does not have rights to change the application password

Resolution

Use the Policy Manager to change the user password policy assigned to this application on the CA SSO Server to allow the user to change the password.

Error Message: "Access Denied"

Number

0x070000

Component

Authentication Agent and its host

Communication Flow diagram Reference:

2.2 & 2.3

Cause

Appears when using LDAP authentication. It means either the username or password are incorrect.

Resolution

Ensure communications between the authentication agent and its host server are functioning as expected.

Cause

Occurs in all authentication agents if there is an error on the authentication server, please see the individual authentication server error logs for more information.

Resolution

Ensure communications between the authentication agent and its host server are functioning as expected.

Error Message: "User <user name> is disabled"

Number

0x505

Component

CA SSO Server

Communication Flow diagram Reference:

2.4, 2.5

Cause

The user's account has been disabled.

Resolution

If this was done in error, use the Policy Manager to enable the user account.

Error Message: "Resource is disabled"

Number

0x109

Component

CA SSO Server

Communication Flow diagram Reference:

2.5

Cause

The application that is being launched is disabled.

Resolution

If this was done in error, use the Policy Manager to enable the application.

Error Message: "Access denied"

Number

0x401

Component

CA SSO Server

Communication Flow diagram Reference:

2.5

Cause

The current user does not have rights to launch the application in question.

Resolution

Use the Policy Manager to grant the user permissions to use this application by adding it to their list of authorized applications.

Error Message: "User <user name> not allowed to login now"

Number

0x02

Component

CA SSO Server

Communication Flow diagram Reference:

2.5

Cause

The current user account has restrictions on it that prohibit logon at that particular time.

Resolution

Change the logon restrictions on the account or logon during an allowed time.

Error Message: "User not allowed to use <authentication host name?"

Number

0x0201

Component

CA SSO Client

Communication Flow diagram Reference:

2.4

Cause

When connecting to an authentication host that is not defined on the CA SSO Server listed in the Auth.ini file.

Resolution

Add the authhost to the CA SSO Server or change the CA SSO Server name in the Auth.ini file.

Error Message: "A communication failure occurred" "Details:" "Reported by CA SSO Server's Client API"

Number

Number not known

Component

Policy Manager

Communication Flow diagram reference:

None - occurs during start of Policy Manager

Other Symptoms:

When Policy Manager starts, there is no connection to the CA SSO Server.

Cause

Unknown

Resolution

Check that the CA SSO Server is running, and that you have entered the correct password for the Policy Manager administrator during logon to the Policy Manager.

Error Message:"Password has not been set yet"

Number

0x807

Component

CA SSO Client

Communication Flow diagram reference:

2.2

Cause

User does not have a current password for __SSO__ application.

Resolution

Set a password for the user through the Policy Manager.

Error Message: Communication error with host

Number

12296

Component

CA SSO Client

Communication Flow diagram reference:

Cause

CA SSO Client is unable to communicate with the Authentication Agent.

Resolution

Ensure the network to authentication agent is up and the authentication agent is running.

Error Message: 0x0 ETWAC_API_OK

Reason:

Indicates success.

Action

No action necessary.

Error Message: 0x100 ETWAC_API_FAIL

Reason

Operation failed.

Action

Enable Web Agent logging to obtain more detailed information about the failure.

Error Message: 0x200 ETWAC_API_INVALID_PARAM

Reason

The supplied parameters for an API are invalid or there is a problem with the system settings (for example, a required parameter is null).

Action

Verify that the supplied parameters and system settings are correct.

Error Message: 0x300 ETWAC_API_LIMIT_EXCEEDED

Reason

Too many objects are being retrieved and the limit was exceeded. This notification indicates that there are more objects than the ones in the list that you received. For example, if you call the Etwac_Adm_FindUsers() function and there are 1000 users to retrieve and the limit set in the user database settings or in the user data store object is 200, you only will receive 200 users.

Action

Increase the limit.

Error Message: 0x400 ETWAC_API_INSUFFICIENTS_RIGHTS

Reason

Insufficient rights. Results from either trying to access a resource and not having permission for this access, or having insufficient rights for administrative actions. For example, an administrator gets this error from administrative functions such as get_token_info.

Note: If you are an administrator, you can use get_token_info on other tokens.

Action

Determine whether the user must have this permission and grant the permission to the user if necessary.

Error Message: 0x500 ETWAC_API_CANT_LOGIN

Reason

The Authentication plug-in validated the user, but the logon restriction failed the authentication so the user cannot log on. This error can occur because:

- The user's account has expired or is disabled
- The requested authentication method is denied
- The user is not allowed to use authentication host

Action

Depending on the reason the error occurred, either:

- Enable the user's account
- Correct the requested authentication method
- Correct the problem with the authentication host

Error Message: 0x600 ETWAC_API_TOKEN_IS_NOT_VALID

Reason

Token is not valid. A token can be invalid because:

- The token cannot be extracted from the string
- The CA SSO Server has been restarted which restarts the Token Directory (restarting the Token Directory removes all existing tokens) and all tokens that exist on the CA SSO Client are not valid
- The CA SSO Server cannot recognize the token, or someone has tampered with the token
- The CA SSO Server crashes, the user cache is erased, and the CA SSO Server cannot tell what the old token was
- The token used has expired

Action

The CA SSO Server holds a table that contains all of the users who are logged in. When you restart the CA SSO Server, this table is cleared and all users that were previously connected to the CA SSO Server must reconnect.

Error Message: 0x700 ETWAC_API_ACCESSOR_TOKEN_IS_NOT_VALID

Reason

The token of the user that the operation is performed on is not valid.

Action

This error is similar to ETWAC_API_TOKEN_IS_NOT_VALID. It is returned from APIs that use two tokens only when the second token (the token that the operation is performed on) is invalid.

Error Message: 0x800 ETWAC_API_OBJECT_NOT_FOUND

Reason

A user, group, or other object cannot be found: the object is in the context of the API.

Action

Verify that the user, group, or other object exists and that the object is specified correctly in your program.

Error Message: 0x900 ETWAC_API_OBJECT_ALREADY_EXISTS

Reason

A user, group, or other object already exists; the object is in the context of the API.

Action

Verify that the user, group, or other object exists and that the object is specified correctly in your API call.

Error Message: 0xA00 ETWAC_API_COMM_ERROR

Reason

Communication error. This error can occur because:

- An error occurred during the packing or unpacking of information
- Incompatible versions of the client APIs and the CA SSO Server were used
- The host was not found
- A bad port number was used
- Another communication error occurred

Action

This error is common when the CA SSO Server is down, so verify that the CA SSO Server is up and running.

Component Codes

This section lists and describes the component codes.

0x00000001 CC_GEN

Component: Generic component.

0x00000002 CC_CLIENT_API

Component: Client-side component.

0x00000003 CC_USER_DB_LDAP

Component: LDAP user data store provider component.

0x00000005 CC_USER_DB

Component: User data store proxy component.

0x00000007 CC_POLICY_DB

Component: Policy data store proxy component.

0x00000008 CC_POLICY_SERVER

Component: CA SSO Server.

0x00000009 CC_TCPXDR

Component: TCP XDR component.

0x0000000A CC_TCPCOMM

Component: TcpComm communication component.

0x0000000E CC_AUTH_ENGINE

Component: Authentication engine component.

0x0000000F CC_ATZN_ENGINE

Component: Authorization engine component.

0x00000011 CC_AUTH_PLUGIN

Component: Authentication plug-in component.

Detailed Error Codes

This section lists the detailed error codes.

0x00000100 ETWAC_FAIL

Reason:

Operation failed.

Action:

Enable Web Agent logging to obtain more detailed information about the failure.

0x00000103 ETWAC_AUTH_HOST_NOT_FOUND

Reason:

Authentication host was not found.

Action:

Verify that this authentication host is defined for the SSO Server. Define it if necessary.

0x00000104 ETWAC_AUTH_HOST_INVALID_PARAM

Reason:

Authentication host is not configured properly.

Action:

Contact your administrator.

0x00000105 ETWAC_AUTH_ERROR

Reason:

An authentication error has occurred.

Action:

Enable Web Agent logging to obtain more information about the authentication error.

0x00000106 ETWAC_NO_MEMORY

Reason:

Not enough memory available to process this command.

Action:

Increase the amount of available memory, and then reissue the command.

0x00000107 ETWAC_BUFFER_TOO_SHORT

Reason:

The buffer is too small.

Action:

Increase the size of the buffer.

0x00000108 ETWAC_TOKEN_EXPIRED

Reason:

The token has expired.

Action:

Reauthenticate.

0x00000109 ETWAC_OBJECT_DISABLED

Reason:

The resource is disabled.

Action:

Contact the SSO administrator to enable the resource.

0x0000010A ETWAC_CANT_CHANGE_LOGINID

Reason:

Unable to update logon name.

Action:

Contact the SSO administrator.

0x0000010B ETWAC_CANT_CHANGE_PASSWORD

Reason:

Cannot change password for the indicated application. The application type is NONE, APPTICKET, or PASSTICKET.

Action:

Contact the SSO administrator to change the application's authentication type.

0x0000010C ETWAC_PASSWORD_EXPIRED

Reason:

The specified account has expired.

Action:

Reactivate the account.

0x0000010D ETWAC_PASSWORD_EXPIRED_AUTOGEN

Reason:

The password has expired and a new password was generated.

Action:

No action necessary.

0x0000010E ETWAC_PASSWORD_MISMATCH

Reason:

Password mismatch for the indicated user.

Action:

Verify that the supplied password is correct for the indicated user.

0x0000010F ETWAC_USER_DATASTORE_INIT_FAILED

Reason:

User data store initialization failed.

Action:

Enable SSO Server logging to obtain more detailed information about this failure.

0x00000110 ETWAC_POLICY_DATASTORE_INIT_FAILED

Reason:

Administrative data store initialization failed.

Action:

Enable SSO Server logging to obtain more detailed information about this failure.

0x00000111 ETWAC_QUERY_FAILED

Reason:

Query failed.

Action:

Restate the query and try again.

0x00000112 ETWAC_UPDATE_QUERY_FAILED

Reason:

Update query failed.

Action:

Restate the query and try again.

0x00000113 ETWAC_AUTOGEN_CHARS_VS_MAX

Reason:

Password auto-generation failed as a result of the following rules contradiction: the minimum length is greater than the maximum length in the consolidated password policies.

Action:

Change the values so that the specified minimum length of a password is less than the specified maximum length.

0x00000114 ETWAC_AUTOGEN_MIN_VS_MAX

Reason:

Password auto-generation failed as a result of the following rules contradiction: the calculated minimum length is greater than the maximum length in the consolidated password policies.

Action:

Change the values so that the calculated minimum length of a password is less than the maximum length.

0x00000115 ETWAC_PWDQC_ALREADY_USED

Reason:

According to the consolidated password policies, the password was used previously.

Action:

Supply a different password.

0x00000116 ETWAC_PWDQC_EMPTY_NOT_ALLOWED

Reason:

Cannot set an empty new password.

Action:

Supply a different password.

0x00000118 ETWAC_PWDQC_LACKS_ALUMN

Reason:

According to the consolidated password policies, the password lacks alphanumeric characters (minimum $D characters).

Action:

Supply a password containing letters and numbers.

0x00000119 ETWAC_PWDQC_LACKS_ALPHA

Reason:

According to the consolidated password policies, the password lacks letters (minimum $D characters).

Action:

Supply a password containing letters.

0x0000011A ETWAC_PWDQC_LACKS_DIGITS

Reason:

According to the consolidated password policies, the password lacks digits (minimum $D characters).

Action:

Supply a password containing numbers.

0x0000011B ETWAC_PWDQC_LACKS_LOWER

Reason:

According to the consolidated password policies, the password lacks lowercase characters (minimum $D characters).

Action:

Supply a password containing lowercase letters.

0x0000011C ETWAC_PWDQC_LACKS_UPPER

Reason:

According to the consolidated password policies, the password lacks uppercase characters (minimum $D characters).

Action:

Supply a password containing uppercase letters.

0x0000011D ETWAC_PWDQC_PWDTOOLONG

Reason:

According to the consolidated password policies, the password is too long (maximum $D characters).

Action:

Supply a shorter password.

0x0000011E ETWAC_PWDQC_PWDTOOSHORT

Reason:

According to the consolidated password policies, the password is too short (minimum $D characters).

Action:

Supply a longer password.

0x0000011F ETWAC_PWDQC_REPCHARS

Reason:

According to the consolidated password policies, the password has repetitive characters (maximum $D characters).

Action:

Supply a password that has no repeating characters.

0x0000012C ETWAC_PWDQC_LACKS_OTHERS

Reason:

According to the consolidated password policies, the password lacks “other characters” (minimum $D characters).

Action:

Supply a password with “other characters.”

0x00000121 ETWAC_SETPWD_CHARS_VS_MAX

Reason:

Indicates a password rules contradiction where the calculated minimum length is greater than the maximum length in the consolidated password policies ($S).

Action:

Change the values so that the calculated minimum length of a password is less than the maximum length.

0x00000122 ETWAC_SETPWD_MIN_VS_MAX

Reason:

Indicates a password rules contradiction where the minimum length is greater than the maximum length in the consolidated password policies ($S).

Action:

Change the values so that the minimum length of a password is less than the maximum length.

0x00000123 ETWAC_AUTH_PLUGIN_INIT_FAILED

Reason:

Authentication plug-in initialization failed.

Action:

Enable SSO Server logging to obtain more information about this failure.

0x00000125 ETWAC_INVALID_DN_SYNTAX

Reason:

The distinguished name has an invalid syntax.

Action:

Correct the syntax of the distinguished name and try again.

0x00000200 ETWAC_INVALID_PARAM

Reason:

The supplied parameters are invalid.

Action:

Correct the supplied parameters and try again.

0x00000201 ETWAC_INVALID_AUTH_METHOD

Reason:

Unknown authentication method.

Action:

Determine whether this authentication method must be supported and define it if necessary.

0x00000202 ETWAC_NAMING_VIOLATION

Reason:

There was a naming violation.

Action:

Correct the name and try again.

0x00000400 ETWAC_INSUFFICIENT_RIGHTS

Reason:

Insufficient rights for this particular request.

Action:

Determine whether the user must have this permission, and then contact the SSO administrator to grant permission to the user if necessary.

0x00000500 ETWAC_CANT_LOGIN

Reason:

Logon failure; unable to log on.

Action:

Enable Web Agent logging to obtain more information about the failure.

0x00000501 ETWAC_AUTHMETHOD_NOT_ALLOWED

Reason:

The user is not allowed to use the indicated authentication method.

Action:

Contact the SSO administrator.

0x00000502 ETWAC_AUTHHOST_NOT_ALLOWED

Reason:

The user was not allowed to use the indicated authentication host.

Action:

Contact the SSO administrator.

0x00000503 ETWAC_TIMEDAY_RESTRICTION

Reason:

The specified user is not allowed to log on now.

Action:

Log on during the time of the day that access is allowed.

0x00000505 ETWAC_USER_DISABLED

Reason:

The specified user is disabled.

Action:

Contact the SSO administrator to enable the user, if appropriate.

0x00000506 ETWAC_USER_EXPIRED

Reason:

The user's account has expired.

Action:

Reactivate the account, if appropriate.

0x00000600 ETWAC_TOKEN_NOT_VALID

Reason:

The token is no longer valid.

Action:

Reauthenticate.

0x00000601 ETWAC_HAS_TO_LOGIN_FIRST

Reason:

The user has to authenticate first.

Action:

Authenticate, and then try again.

0x00000602 ETWAC_UNTRUSTED_TICKET_SOURCE

Reason:

The ticket was issued by an untrusted server.

Action:

Contact the SSO administrator to authorize the issuing SSO Server, if appropriate.

0x00000603 ETWAC_TOKEN_EXPIRED

Reason:

The token has expired.

Action:

Reauthenticate.

0x00000700 ETWAC_ACCESSOR_TOKEN_NOT_VALID

Reason:

The token is no longer valid.

Action:

Reauthenticate.

0x00000701 ETWAC_ACCESSOR_TOKEN_EXPIRED

Reason:

The token has expired.

Action:

Reauthenticate.

0x00000800 ETWAC_OBJECT_NOT_FOUND

Reason:

The specified object was not found.

Action:

Enable SSO Server logging to obtain more information about the error.

0x00000801 ETWAC_USER_NOT_FOUND

Reason:

The specified user was not found.

Action:

Verify that the user exists in the user data store.

0x00000802 ETWAC_GROUP_NOT_FOUND

Reason:

The specified group was not found.

Action:

Verify that the specified group is defined in the data store.

0x00000803 ETWAC_CLASS_NOT_FOUND

Reason:

The class was not found.

Action:

Determine whether the class exists, and then verify that the class is specified correctly in the API call.

0x00000804 ETWAC_TARGET_USER_NOT_FOUND

Reason:

The target user was not found.

Action:

Determine whether the user exists, and then verify that the user is specified correctly in the API call.

0x00000805 ETWAC_ATTRIBUTE_NOT_FOUND

Reason:

The attribute was not found.

Action:

Determine whether the attribute exists, and then verify that the attribute is specified correctly in the API call.

0x00000806 ETWAC_LOGIN_INFO_NOT_FOUND

Reason:

Unable to find logon information for the application.

Action:

Enable Web Agent logging to obtain more information about this failure.

0x00000807 ETWAC_PASSWORD_NOT_FOUND

Reason:

The password has not been set yet.

Action:

Set the password and try again.

0x00000900 ETWAC_OBJECT_ALREADY_EXISTS

Reason:

The object already exists.

Action:

No action is necessary.

0x00000901 ETWAC_VALUE_ALREADY_EXISTS

Reason:

The entry already exists.

Action:

No action is necessary.

0x00000902 ETWAC_USERS_ALREADY_EXISTS

Reason:

The user already exists

Action:

No action is necessary.

0x00000903 ETWAC_GROUP_ALREADY_EXISTS

Reason:

The group already exists.

Action:

No action is necessary.

0x00000904 ETWAC_CLASS_ALREADY_EXISTS

Reason:

The class already exists.

Action:

No action is necessary.

0x00000A00 ETWAC_COMM_ERROR

Reason:

A communication failure occurred.

Action:

Enable Web Agent logging to obtain more information about this failure.

0x00000A01 ETWAC_PACK_FAILED

Reason:

Failed to pack data.

Action:

Enable SSO Server logging to obtain more information about this failure.

0x00000A02 ETWAC_UNPACK_FAILED

Reason:

Failed to unpack data.

Action:

Enable SSO Server logging to obtain more information about this failure.

0x00000A03 ETWAC_INCOMPATIBLE_VERSION

Reason:

Incompatible version.

Action:

Enable Web Agent logging to obtain more information about this problem.

0x00000A04 ETWAC_COMM_TIMEOUT

Reason:

The communication time-out period expired.

Action:

Try again later.

0x00000A05 ETWAC_HOST_NOT_FOUND

Reason:

The host was not found.

Action:

Determine whether the host exists, and then verify that the host is specified correctly in the API call.

0x00000A06 ETWAC_SERVER_DOWN

Reason:

The SSO Server is down.

Action:

Contact the SSO administrator to determine the status of the SSO Server.

0x00000A07 ETWAC_SERVER_BUSY

Reason:

The SSO Server is busy.

Action:

Try again later, or contact the SSO administrator to reevaluate the SSO Server settings to improve performance, if possible.

0x00000A08 ETWAC_INVALID_COMM_PARAM

Reason:

Invalid communication parameters.

Action:

Correct the parameters and try again.

0x00000A09 ETWAC_COMM_INPUTEXHAUSTED

Reason:

The connection was unexpectedly closed by the SSO Client or the SSO Server.

Action:

Enable SSO Server logging to obtain more information about this situation.

Appendix G: Password Exits

This section contains the following topics:

About Password Exits

CA SSO has password rules that have been defined to produce a wide range of password possibilities to meet every site's requirements for passwords. However, there are times when a site uses password rules that go beyond the rules that have been created in CA SSO. To create password rules that are unique to a site, use password exits.

There are two types of password exits: password change exits and password auto gen exits. If you want to implement customized password functionality using password exits, you must write a DLL (dynamic-link library).

Password Change Exits

There are two kinds of password change exits: pre-exit and post-exit. The pre-exit is a flag to indicate that the exit must perform a quality check and may deny a certain password, before the password is actually changed. The post-exit is flag used as notification of a successful password change. The process that changes application passwords is executed in the following order:

- SSO password validation
- SSO pre exit
- CA SSO Server update password
- SSO post exit

Password Auto-Gen Exit

If the password auto-gen exit parameter is defined, you can write your own auto-gen exit instead of using SSO's auto-gen exit. Auto-gen is an SSO function that creates a new password for the user when a password expires.

Password Exit Tokens

You need to create a run-time library (DLL/shared library). There is a configuration section called exits, which has three tokens. The three tokens are named ExitsPath, PasswordExits, and AutogenExit. The ExitsPath token defines the directory path where any exit DLLs are located (by default, a folder named Exits is found in the CA SSO Server install path). PasswordExits contains the list of DLLs.

Note: You can write more than one DLL. The DLLs are executed in the order that they appear in the PasswordExits token. AutogenExit is the name of the autogen DLL that is located in ExitsPath. The include files (for example, Pwd_Exits.h) are located in SSO Server \SDK\Include directory.

Password Exit Functions

The following table summarizes the password exit functions that need to be defined in a password exit DLL:

Function	Description	Exit Type
ssod_Exit_Pwd_InitEx	PasswordExit notification. Called from various locations (the context of the call can be determined by the Exit Flag variable (see below).	Password exit
ssod_Exit_Pwd_Term	Performs termination and cleanup of the password exit DLL.	Password exit
ssod_Exit_SetPwdEx	PasswordExit notification. Called from various locations (the context of the call can be determined by the Exit Flag variable (see below).	Password exit
Ssod_Exit_Free_ErrorString	If you have defined an error string for the CA SSO Server, this frees the memory allocated to this string. **Note:** This function is also used to free the other string information returned by the init function (user_attr_to_fetch, relevant_user_dirs, relevant_applications)	Password exit
ssod_Exit_Auto_Init	Initializes the auto-gen exit DLL.	Auto-gen exit
ssod_Exit_Auto_Term	Performs termination and cleanup of the auto-gen exit DLL.	Auto-gen exit
ssod_Exit_PwdAutoGen	Generates a password that is used instead of the SSO password auto-gen mechanism.	Auto-gen exit

Password Exit Flags

To determine whether the exit is being called before or after the Server changes the password, the exit is called with one of the following flags set:

Flag	Description	Exit Type
ETSSO_SP_EXIT_FLAG_PRE_SETPWD	The Server is calling the exit to notify it that the change is made to one of the following: ■ szCurrentPassword ■ szNextPassword ■ Both current and next passwords The password exit can choose to cancel this operation by returning a non zero value	Password exit
ETSSO_SP_EXIT_FLAG_POST_SETPWD	This notifies the exit that the password has been successfully changed.	Password exit
ETSSO_SP_EXIT_FLAG_PWD_AUTOGEN	This notifies the exit that the password has been successfully changed due to an auto gen operation (so pre_setpwd was called for this change).	Password exit
ETSSO_SP_EXIT_FLAG_PWD_CHANGE_NOTIFY	This notifies the exit that the password that was in NEXTPWD has been moved to CURRPWD.	Password exit
ETSSO_SP_EXIT_FLAG_CLEAR_LOGININFO	This notifies the exit that the login information for this application has been successfully cleared from the Server's database.	Password exit
ETSSO_SP_EXIT_FLAG_LOGIN_NOTIFY	This notifies the password exit that the user was able to successfully log in using the current password (CURRPWD), and allows the exit to specify whether PwdChangeNotify is needed or if the value in NEXTPWD fails the password policy requirements.	Password exit
ETSSO_SP_EXIT_FLAG_SYNC_APPL	This notifies the exit that the password change has come due to a sync application's password being changed.	Password exit

Password Exit Returns

The following return values may be specified by the password exit when a ETSSO_SP_EXIT_FLAG_LOGIN_NOTIFY flag is passed as a parameter.

Return Value	Description	Exit Type
ETSSO_SP_EXIT_RET_PWD_CHANGE_NOTIFY_REQUIRED	Signals the Server that a password change is required - This will trigger the server to perform PwdChangeNotify.	Password Exit
ETSSO_SP_EXIT_RET_NEXT_PWD_DOES_NOT_QUALIFY	Signals the Server to clear the value stored in NEXTPWD and set the 'User must change password on next login' flag.	Password Exit

ssod_Exit_Pwd_InitEx

The ssod_Exit_Pwd_InitEx function initializes the password exit DLL.

Syntax

```
unsigned long ssod_Exit_Pwd_InitEx(
   void     **ppCtxExit,

SSODServerInfo *pServerInfo
   unsigned long   *pulExitVersion

unsigned long *required_data_flags

const char **user_attr_to_fetch

const char **relevant_user_dirs

const char **relevant_applications
  );
```

Parameter	Description	Type
ppCtxExit	Provides the address of a pointer variable to be associated with the exit. This allows the exit to register a pointer to exit-specific context. This context can be used to store any exit-specific data. This pointer is provided to the exit on all the following calls to the exit (including on ssod_Exit_Pwd_Term where it can be freed by the exit before the server shuts down).	Input/Output

Parameter	Description	Type
pServerInfo	An CA SSO Server structure containing version information. Review the Pwd_Exits.h file for further information.	Input
pulExitVersion	Indicates the exit version.	Input
required_data_flags	Mask of flags that specifies the additional information needed by the password exit (see table below). This information is provided for every SetPwdEx exit call in the vars_list array. Review the Pwd_Exits.h file for further information.	Input/Output
user_attr_to_fetch	When required_data_flags has the SP_VAR_USER_ATTR_VAL bit turned on, this parameter must return to the CA SSO Server a zero terminated string with the name of the user attribute to query. The name of user attr will not have its @<user_dir> postfix, and is freed by the function Ssod_Exit_Free_ErrorString provided by the exit dll.	Input/Output
relevant_user_dirs	This parameter is provided to improve CA SSO Server's performance and allow the exit owner to specify a comma separated list of user data store names that the exit is called for. When a user from a user data store that is not mentioned in the list, is served, the exit is not called. Setting this parameter disables the filter. The zero terminated string of comma separated user data store names returned by this parameter is freed by the function Ssod_Exit_Free_ErrorString provided by the exit dll.	Input/Output

Parameter	Description	Type
relevant_applications	This parameter is provided to improve the CA SSO Server's performance and allow the exit owner to specify a comma separated list of applications that the exit is called for. When the application being processed is not one of the applications mentioned in that list, the exit is not called. Setting this parameter to NULL disables the filter. The zero terminated string of comma separated application names returned by this parameter is freed by the function Ssod_Exit_Free_ErrorString provided by the exit dll.	Input/Output

ssod_Exit_Pwd_InitEx Flags

The following table describes the flags:

required_data_flags	Description
SP_VAR_EXISTING_ PWD	Setting this flag includes the application's current password (prior to performing any modifications) in the vars_list array.
SP_VAR_CURR_LOGIN_ID	Setting this flag includes the login id in the vars_list array.
SP_VAR_USER_DN	Setting this flag includes the usr's DN relative to the user data store's base path in the vars_list array.
SP_VAR_USER_FULL_DN	Setting this flag includes the user's full DN (including the user data store's base path) in the vars_list array.
SP_VAR_USER_ATTR_VAL	Setting this flag causes the server to query the directory and retrieve the user's value of the field specified in the userattr object. (the user attr name is provided by the "user_attr_to_fetch" parameter) This value will be included in the vars_list array.

Return values

A return value of 0 indicates success.

A non-zero return value indicates that exit initialization failed.

ssod_Exit_Pwd_Term

The ssod_Exit_Pwd_Term function terminates and cleans up the password exit DLL.

Syntax

```
unsigned long   ssod_Exit_Pwd_Term (
 void        **ppCtxExit
 );
```

Parameter	Description	Type
ppCtxExit	Provides the pointer to exit-specific context. **Note:** This call allows you to free any information stored in this context that has to be freed.	Input/Output

Return values

A return value of 0 indicates success.

A non-zero return value indicates an error.

ssod_Exit_SetPwdExe

The ssod_Exit_SetPwdEx function is called after the CA SSO Server checks the quality of the new password using the password policies of SSO, and after CA SSO Server sets the password.

Syntax

```
unsigned long  ssod_Exit_SetPwdEx(
   void        *pCtxExit,
   const char *szUserDirName,
   const char *szContainerDN,
   const char      *szSSOUserName,
   const char      *szApplName,
   const char      *szLoginName,
   const char      *szCurrentPassword,
   const char      *szNextPassword,
   unsigned long sp_flags,
   unsigned long exit_flags,
   SetPwdVar vars_list[],
   unsigned long vars_count,
   char       **pszExtra
  );
```

Parameter	Description	Type
pCtxExit	Exit specific context.	Input
szUserDirName	The name of the user's User Data Store	Input
szContainerDN	The user's container dn (relative to the user data store's base path)	
szSSOUserName	The user name as it appears in the CA SSO database.	Input
szApplName	The application name.	Input
szLoginName	The application’s login ID.	Input
szCurrentPassword	The current value of the application password.	Input
szNextPassword	The next value of the application password	Input
sp_flags	The eTsso_SetPwd request flags, as sent by the user (i.e. by SSO Client/PM)	Input

Parameter	Description	Type
exit_flags	The flags used by the exit determine the reason for calling the exit (see flags table above).	Input
vars_list[]	The array of variables that we requested in the ssod_Exit_Pwd_InitEx function call.	Input
vars_count	The number of items in the array of vars_list (above).	Input
pszExtra	A pointer to a string that contains an error message if the password is rejected.	Output

Return Values

A return value of 0 indicates that the password is approved.

A non-zero return value indicates that the password is rejected.

The values of this function may vary according to the location of the call (the location of the call can be determined by the exit_flag parameter). The following describe those locations:

Flag	Location
ETSSO_SP_EXIT_FLAG_PRE_SETPWD	For master application: ■ ApplName - Master Application name (might be different from the application name in original SP request). ■ LoginID - as received from client ■ CurrPwd - as received from client ■ NextPwd - as received from client ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_PRE_SETPWD

Flag	Location
	For Sync application: ■ ApplName - Sync Application name. ■ LoginID - NULL ■ CurrPwd - NULL ■ NextPwd - NextPwd or if not specified, CurrPwd ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_PRE_SETPWD| ETSSO_SP_EXIT_FLAG_SYNC_APPL
ETSSO_SP_EXIT_FLAG_POST_SETPWD	For master application: ■ ApplName - Master Application name (might be different from the application name in original SP request). ■ LoginID - as received from client ■ CurrPwd - as received from client ■ NextPwd - as received from client ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_POST_SETPWD
	For Sync application: ■ ApplName - Sync Application name. ■ LoginID - NULL ■ CurrPwd - NULL ■ NextPwd - NextPwd or if not specified, CurrPwd ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_POST_SETPWD| ETSSO_SP_EXIT_FLAG_SYNC_APPL
ETSSO_SP_EXIT_FLAG_PWD_AUTOGEN	■ ApplName - Master Application name (might be different from the application name in original SP request). ■ LoginID - NULL ■ CurrPwd - NULL ■ NextPwd - auto generated password. ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_PWD_AUTOGEN
	■ ApplName - Master Application name (might be different from the application name in original SP request). ■ LoginID - NULL ■ CurrPwd - the current value of next pwd ■ NextPwd - NULL// maybe empty string? ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_PWD_CHANGE_NOTIFY

Flag	Location
ETSSO_SP_EXIT_FLAG_CLEAR_LOGININFO	■ ApplName - Master Application name (might be different from the application name in original SP request). ■ LoginID - NULL. ■ CurrPwd - NULL. ■ NextPwd - NULL. ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_CLEAR_LOGININFO
ETSSO_SP_EXIT_FLAG_LOGIN_NOTIFY	■ ApplName -Application name. ■ LoginID - NULL. ■ CurrPwd - NULL. ■ NextPwd - current next pwd. ■ SetPwd Exit Flag - ETSSO_SP_EXIT_FLAG_LOGIN_NOTIFY

Return Values

A return value of 0 indicates that the password is approved.

A value of ETSSO_SP_EXIT_RET_PWD_CHANGE_NOTIFY_REQUIRED indicates that the Server must perform a password change notify.

A value of ETSSO_SP_EXIT_RET_NEXT_PWD_DOES_NOT_QUALIFY indicates that the value in NEXTPWD fails the password quality check.

A non-zero return value indicates that the password is rejected.

ssod_Exit_Free_ErrorString

The ssod_Exit_Free_ErrorString function is called if the CA SSO Server is passed an error string in the ssod_Exit_SetPwdEx function pszExtra parameter.

Syntax

```
unsigned long  ssod_Exit_Free_ErrorString(
   void      *pCtxExit,
   char      *pszExtra
);
```

Parameter	Description	Type
ppCtxExit	Exit specific context.	Input
pszExtra	The string to be freed.	Input

ssod_Exit_Auto_Init

The ssod_Exit_Auto_Init function initializes the auto-gen exit DLL.

Syntax

```
unsigned long ssod_Exit_Auto_Init(
   void     **ppCtxExit,
   unsigned long   *pulExitVersion
  );
```

Parameter	Description	Type
ppCtxExit	Indicates the pointer to exit-specific context. This is a placeholder for any static data the auto-gen exit needs.	Input/Output
pulExitVersion	Indicates the exit version, which is defined by SSOD_EXIT_VERSION.	Output

Return Values

A return value of 0 indicates success.

A non-zero return value indicates that exit initialization failed.

ssod_Exit_Auto_Term

The ssod_Exit_Auto_Term function terminates and cleans up the auto-gen exit DLL.

Syntax

```
unsigned long ssod_Exit_Auto_Term(
   void     **ppCtxExit,
);
```

Parameter	Description	Type
ppCtxExit	Indicates the pointer to exit-specific context. This is a placeholder for any static data the auto-gen exit needs.	Input/Output

Return Values

A return value of 0 indicates success.

A non-zero return value indicates an error.

ssod_Exit_PwdAutoGen

The ssod_Exit_PwdAutoGen function generates a password, which is used instead of the SSO default password auto-gen mechanism.

Syntax

```
unsigned long  ssod_Exit_PwdAutoGen(
   void      *pCtxExit,
   const char     *szSSOUserName
   const SEOS_PASSWDRULES  *pPwdRules
   const char     *szApplName,
   const char     *szLoginName,
   char      *szPassword,
   int       iPwdSize
   char      **pszExtra
);
```

Parameter	Description	Type
pCtxExit	Exit specific context.	Input
szSSOUserName	The user name as it appears in the SSO database.	Input
pPwdRules	A pointer to the SEOS_PASSWRDRULES structure.	Input
szApplName	The application name.	Input
szLoginName	The application's login ID.	Input
szPassword	The generated password that the exit created.	Input
iPwdSize	Defines the password buffer length.	Output
pszExtra	A pointer to a string that contains an error message.	Output

Return Values

A return value of 0 indicates success.

A non-zero return value indicates that a password could not be generated.