eTrust™ Single Sign-On

Administrator Guide

r8

Second Edition

Contents

Chapter 1: Introduction to eTrust Single Sign-On

Chapter 2: The SSO Client

Chapter 3: The Policy Server

Chapter 4: The Policy Manager

Chapter 5: Managing Users and User Groups

Chapter 6: Managing Resources

Chapter 7: Managing Passwords

Chapter 8: Managing User Sessions

Chapter 9: Managing Services

Chapter 10: Authenticating Users to eTrust SSO

Chapter 11: Launching Applications with eTrust SSO

Chapter 12: Authenticating Users to Web Applications

Chapter 13: Working with the SSO Client

Chapter 14: Working with the User Data Store

Chapter 15: Working with the Policy Data Store

Chapter 16: Working with the Token Data Store

Chapter 17 : Working with Server Farms

Chapter 18: Maintenance

Chapter 19: Auditing and Logging

Appendix A: Configuring the SSO Client: SsoClnt.ini

Appendix B: Configuring the Policy Server

Appendix C: Configuring the One-Time Password Agent: seotp.ini

Appendix D: Using Selang

Appendix E: Interpreting Error Messages

Appendix F: Password Exits

Chapter 1

Introduction to eTrust Single Sign-On

eTrust™ Single Sign On (eTrust SSO) is a system that you can configure so that end users only have to authenticate (log on and identify themselves) once to gain access to all of their secure desktop applications. This includes some web browser-based applications.

The purpose of eTrust Single Sign-On is to:

- Simplify the logon and authentication process for end users
- Restrict access to specific data and applications on the network
- Create a more secure network environment
- Give administrators efficient and secure control over resources

The purpose of this guide is to describe how eTrust Single Sign-On works, and how to adjust and maintain it. This guide is aimed at administrators who are responsible for keeping an eTrust SSO system working. For information about implementing an eTrust SSO system, see the *Implementation Guide*.

Basic Concepts

This section gives you an overview of eTrust Single Sign-On concepts, including:

- End users
- Primary authentication
- Scripts
- Applications (accessed through eTrust SSO)
- Session profiles
- Resources
- Policies

End Users

End users are individuals (usually employees of your company) who are using eTrust SSO. The eTrust SSO system must identify each individual as unique. Each user has specific information recorded against them including:

- How they should be authenticated
- What applications they need access to
- What groups they belong to
- Application logon credentials

End-User Experience

From an end user's perspective, eTrust SSO is designed to help them log on to multiple software applications without having to identify themselves every time. However, end users must identify themselves when they first log on to eTrust SSO. This is called primary authentication. For more information, see the Primary Authentication section in this chapter.

Primary authentication usually uses a combination of a username and either a password, a smart card, an ID card or a biometric device. The term "biometric" refers to technology that uses biology such as a fingerprint or iris scanner.

Once an end user successfully authenticates and logs on to eTrust SSO, they see a list of applications that they can access. Users can select these applications either from the Start menu in Windows, or from an eTrust SSO application list. These applications typically require the user to provide identification. These applications can be from any company.

Ways to Upload End Users

When implementing eTrust SSO in your organization, the implementation team typically uploads a large number of users at once. The implementation team uses commands in the Policy Server to accomplish this task. This is not usually a regular administration task.

After eTrust SSO has been implemented in your organization you usually only need to add or remove users individually as staff are hired or leave the company. This is done through the Policy Manager or using eTrust Admin, a role-based, user- provisioning product. This is a regular administration task. For more information about the Policy Manager, see the Policy Manager section in the chapter "Tour of eTrust Components".

Groups of End Users

Assigning users to groups and setting the access permissions by group eliminates the need to create and remove access rules for each user. Administrators usually put users in groups to facilitate user management.

For example, you can create a user group for the Payroll department and grant only members of that group access to sensitive payroll information. As employees leave or join the Payroll department, you can add or remove them from the group to automatically grant or deny access to the correct resources.

Primary Authentication

Primary authentication is the method by which users identify themselves to the eTrust SSO system. For primary authentication to occur, a user must enter unique credentials (such as a user name and password) and the system must verify those credentials.

The eTrust SSO system is designed to use several different systems to verify those credentials, including third-party authentication methods.

Authentication Methods Developed and Supported by CA

eTrust SSO comes with two "native" authentication methods that you can use immediately out-of-the-box. The eTrust SSO native authentication methods are:

Authentication Software	Authentication Method
eTrust SSO	Username and Password
LDAP	Username and Password

Authentication Methods Supported by CA

eTrust SSO also comes with the ability to integrate quickly and easily with several third-party authentication vendors, using authentication agents. For more information about authentication agents see the Authenticating Users section in the "Common eTrust SSO Processes" chapter in this guide. You may already have this software implemented within your company. These are:

Authentication Software	Authentication Method
Cert	Digital certificates
Entrust	Digital certificates
Novell	Username and password
NT	Username and password
RSA SecurID	Secure ID card + PIN

Authentication Methods Supported by External Vendors

CA works with third-party vendors to help them to integrate with eTrust SSO. The software vendors providing authentication agents that integrate with eTrust SSO are:

Authentication Software	Authentication Method
Politec	Biometric devices (including iris, and fingerprint scan)
SAFLINK	Biometric devices (including iris, and fingerprint scan)

Additional Authentication Methods

Customers and third-party vendors can also integrate additional authentication methods using the eTrust SSO API.

Applications in eTrust SSO

What are Applications?

In eTrust SSO, *applications* are any software applications that have been added to eTrust SSO and are ready to allocate to users. These can be Windows, UNIX, mainframe, or web-based applications that you want your users to have access to after they have authenticated to eTrust SSO.

The eTrust SSO applications can be located on either the user's computer or on a computer connected to the network.

What are Application Lists?

Every user has an application list. This list contains all the applications that the user is authorized to use that are started by eTrust SSO. If the user is a member of a group, all the applications that the group can access appear on the user's eTrust SSO application list when they log on to eTrust SSO.

the application list can be displayed in:

- The Start menu
- An eTrust SSO toolbar
- An eTrust SSO window (launched from an icon)

The eTrust SSO administrator or implementation team can also customize how users access their application lists.

For an application to appear in a user's eTrust SSO application list, the administrator must write an eTrust SSO login script and must assign it to the user (or user group). For more information about scripts, see the Scripts section in this chapter.

Scripts

In eTrust SSO, *scripts* are Tcl scripts that perform tasks for the user. Scripts can be used for a wide variety of tasks. A *logon script*, for example, automatically logs a user in to an application. It automatically inserts the user's name and password in the relevant fields of the logon screens.

eTrust SSO scripts are written in an extended version of the Tcl scripting language. 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 eTrust 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 eTrust SSO.

You may also need to use JavaScript to launch Web applications using eTrust SSO.

Scripts to Launch Applications

The most obvious function of a script in the single sign-on environment is to launch applications and insert the user's logon credentials so that the user does not have to remember passwords or enter any data.

You must write a logon script for each application you add to the eTrust SSO system. The login scripts must conform exactly to the specific logon requirements of each application in your environment.

Other Functions for Scripts

Scripts can do more than simply launch applications and enter user credentials. Scripts can also be used to do the following:

- Close applications
- Change and synchronize passwords
- Automate repetitive tasks
- Automate long navigation trails

Example

Ken, a busy doctor, must access an application that permits him to enter patient data. Each time Ken logs in to this application, he must navigate through four windows and enter default data before reaching the screen he actually needs. A Tcl script can be written to automate this process, which improves productivity.

Session Profiles

You can define a session as the period of time a user is logged in to the eTrust SSO Client.

By default, eTrust SSO lets users have multiple concurrent sessions. Using eTrust SSO, you can set automatic session management rules to limit the number of concurrent sessions a user has open sessions. You can also work with sessions manually using the Session Administrator.

You can use eTrust SSO to:

- Limit the maximum number of sessions a user can have open simultaneously
- Define what happens when a user attempts to exceed this number of sessions
- Manually terminate any sessions

To protect sensitive information, you can use the Policy Manager to set the following:

- An idle time-out for locking the session
- An idle time-out for logging the user out of the session

What is a Session Profile?

Using the eTrust SSO management GUI, the Policy Manager, you can set **profiles** that define how user sessions work. Profiles are groups of settings applied to users or groups of users.

Profiles include the following settings:

- How many sessions the user can have open simultaneously
- What happens when the user reaches their maximum number of sessions
- What happens when the system is not used for a length of time

Resources

In eTrust SSO, a *resource* is software, hardware, and settings managed by the Policy Manager. Users and authentication agents are not included in the resources category.

Different Types of Resources

The Policy Manager can manage different types of resources.

Resources	Explanation
Data Stores	**User Data Stores:** Computers that store user details **User attributes:** Categories of extra information that can be recorded for each user, such as the country they work in.
Configuration Resources	**Terminal:** All computers that run the Policy Manager. **Authentication Host:** Computers that run the authentication agent. **Authentication Method:** The method used to authenticate users. **Password policies:** The rules that apply to the strength of the password. **Response Table:** Table that defines additional information returned with an authorization request. **Token Directory:** An LDAP directory in which the Policy Server stores the users' session information.
Web/Generic Resources	**URLs:** Web addresses of secure web pages that you can log on to using eTrust SSO. **EJBs:** Enterprise Java Beans (EJB) are not relevant to eTrust SSO. For more information about EJBs, see eTrust Web AC documentation.
Application Resources	Applications that users can log on to using eTrust SSO.
Session Resources	Details about how to configure each user's application session. For more information, see the Session Profiles section in this chapter.

Policies

A policy is a set of rules that defines the behavior of something. In eTrust SSO, you can define a policy for passwords.

This password policy controls the password that users enter to log on to the eTrust SSO system.

You can set the:

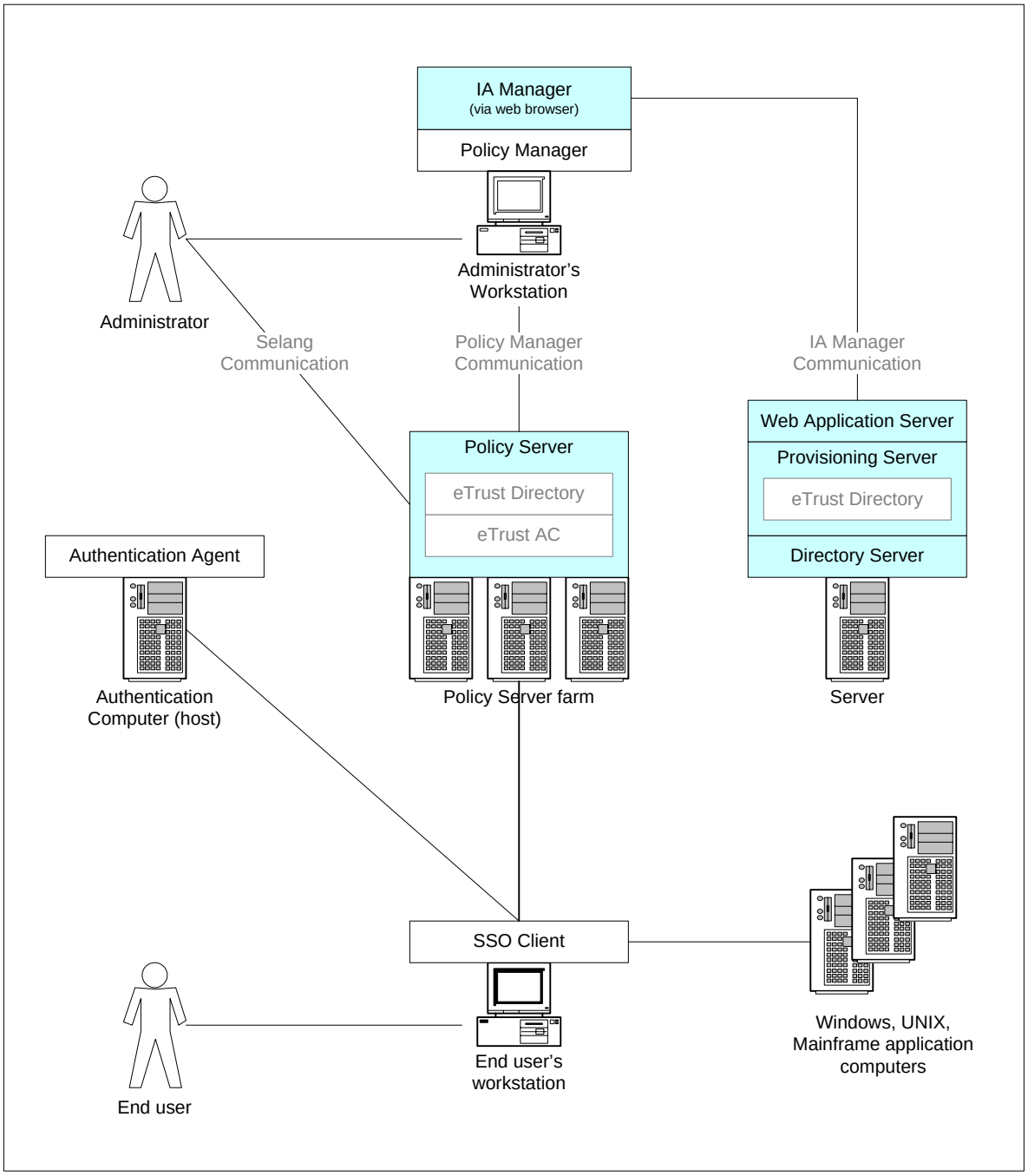
- Minimum and maximum length
- Alphanumeric combination
- Upper and lower case combination
- Password change interval
- Password history (how long before you can reuse an old password)
- Grace logons

eTrust SSO Components

This section introduces the main components of the eTrust SSO architecture. In this chapter, we look at:

- Policy Server
- Policy Manager
- Data Stores
  - eTrust Access Control
  - eTrust Directory (LDAP)
- eTrust SSO Client

The shaded boxes in the diagrams represent the software components.


IA Manager
(via web browser)
Policy Manager
Administrator's
Workstation
Administrator
Selang
Communication
Policy Manager
Communication
IA Manager
Communication
Web Application Server
Provisioning Server
eTrust Directory
Directory Server
Policy Server
eTrust Directory
eTrust AC
Authentication Agent
Authentication
Computer (host)
Policy Server farm
Server
SSO Client
Windows, UNIX,
Mainframe application
computers
End user's
workstation
End user

Policy Server

The Policy Server is the heart of eTrust Single Sign-On. It resides on a central UNIX or Windows server, and completely manages eTrust SSO. You can control the Policy Server from the command line using selang commands or by using the Policy Manager. For more information about selang, see the *selang Command Reference Guide*.

The Policy Server performs the following functions:

- Provides authentication
- Manages resources in the eTrust Access Control database
- Builds the list of applications that a user is allowed to access and sends it to the SSO Client
- Retrieves the logon scripts and the user-specific logon data for each application
- Determines which web resources users can access

Policy Manager

The Policy Manager is a Windows GUI for managing the Policy Server and the data stores. It is usually installed on an administrator's workstation with TCP/IP communication to the Policy Server. You can use the Policy Manager to communicate with both UNIX and Windows Policy Server computers.

As an administrator, you must perform a number of tasks regularly using the Policy Manager. These include:

- Configuring and connecting all the eTrust SSO components
- Adding and grouping users
- Allocating applications and web resources to users and user groups
- Establishing access rights

The Policy Manager also controls other eTrust products including eTrust Access Control and eTrust Web Access Control.

Data Stores

The *data stores* are the directories or databases that store the data associated with eTrust SSO. eTrust SSO comes with two data stores, eTrust Access Control and eTrust Directory, that each give slightly different benefits. You can also integrate third-party LDAP data stores.

eTrust Access Control (Data Store)

eTrust SSO comes with eTrust Access Control. The eTrust Access Control is a database that stores all information about:

- Resources
- Applications
- Access control rules

You can use either eTrust Access Control, eTrust Directory, or another LDAP directory to store information about:

- Users
- User groups
- Logon information

You can populate this database with user and group information from existing databases in your organization, during or after product installation. You can conveniently import user and group information by running a utility, or by using the command line interface.

Other eTrust products also use the eTrust Access Control database. Once you load information in the database, these products can all read and update the shared database for their separate and common purposes.

eTrust Directory (LDAP Data Store)

eTrust SSO comes with eTrust Directory. eTrust Directory is designed to efficiently manage thousands of users, which significantly enhances the performance and scalability of eTrust SSO. The eTrust Directory data store is perfect for large enterprise installations.

You can use eTrust Directory to store information previously stored on eTrust Access Control. eTrust Directory can store information about:

- Users
- User groups
- Logon information

Other eTrust products also use eTrust Directory. Once you load information in the data store, these products can all read and update the shared database for their separate and common purposes.

SSO Client

The SSO Client runs on every workstation that uses eTrust SSO desktop services. You can run the SSO Client software on the user workstation, or it can be run on the workstation from a network file server.

The eTrust SSO Client is a small application that allows users in your enterprise to work with eTrust Single Sign-On (eTrust SSO). This is the only eTrust SSO component that the end user sees and works with.

The SSO Client software handles the following tasks:

- Communicates with primary authentication agents to verify the user's primary authentication, and 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 Policy Server to gain access to applications
- Executes a logon script and logs the user in to the selected application
- Sends the results of the logon attempt to the Policy Server (when instructed by the logon script)

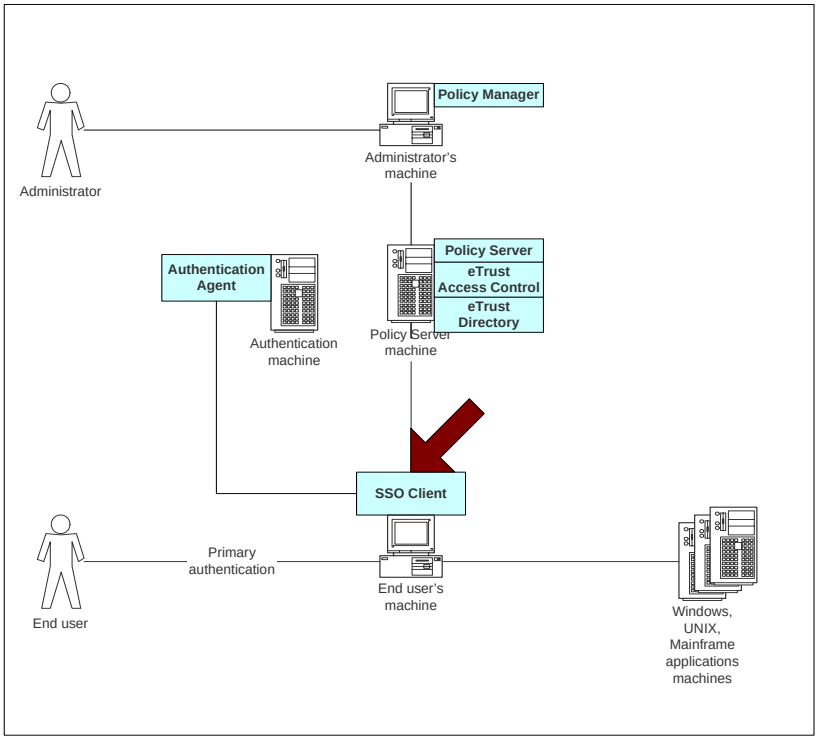
Chapter

2

The SSO Client

The SSO Client is a small application that allows users in your enterprise to work with eTrust SSO. This is the only eTrust Single Sign-On (eTrust SSO) component that the user sees and works with.

The client runs on every workstation that uses eTrust SSO services. The client can be installed on each workstation, or it can be run on the workstations from a networked server.



The SSO Client software 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 Policy 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 Policy Server if instructed to do so by the logon script

Running the SSO Client

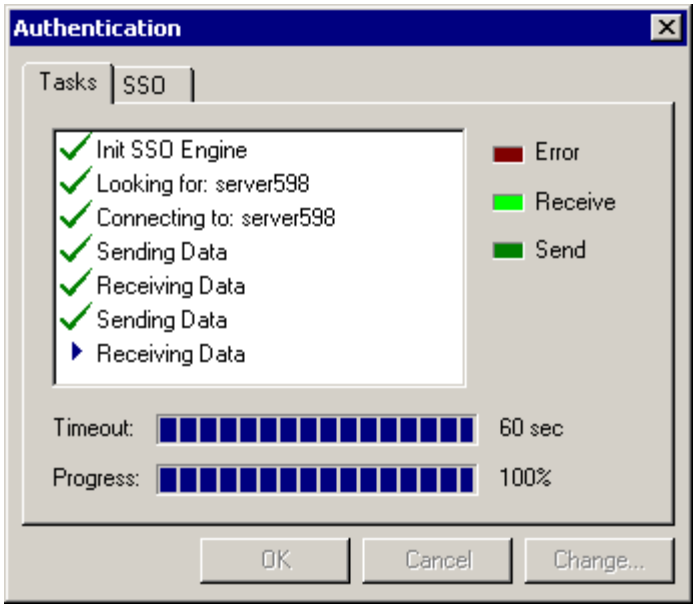
At installation, a shortcut for the SSO Client is placed in the system Start menu folder.

When an SSO-supported application is invoked with a desktop icon (which can be created with the SSO Tools), the SSO Client is automatically started, as well.

When the SSO Client for Windows starts up, it shows a window for primary authentication like this one:

The Authentication window contains a tabbed page for each method of primary authentication the user is authorized to use according to the settings in the SsoClnt.ini file. The user chooses a tabbed page for the type of primary authentication, then enters the credentials and, optionally, an authentication server.

The Tasks tabbed page shows the progress of the primary authentication process:



The SSO Tools Dialog and the SSO Toolbar Dialog

The SSO Client contains the SSO Tools and the SSO Toolbar, two different user interfaces that enable the user to manage passwords and to launch applications.

These tools allow the user to:

- Obtain user information
- Change application passwords using the Advanced button
- Refresh the application list
- Change the password for SSO native primary authentication
- Display the application list

The SSO Tools and the SSO Toolbar are transparently installed during the SSO Client installation. During the installation process you will be prompted to choose which user interface you want to use: SSO Tools or SSO Toolbar. When you run the SSO Client, the user interface you chose is shown.

Suppose you chose SSO Toolbar and, at any time, you decide to stop using it and start using SSO Tools. With a single modification to the SsoClnt.ini file you will be able to switch between these user interfaces.

The token Enabled in the section [Toolbar] in the SsoClnt.ini file determines if the SSO Toolbar option will be used as the user interface. This token can have two values: **yes** or **no**. If you specify Enabled=no, the SSO Toolbar option is disabled and the SSO Tools option becomes automatically enabled. Changing the value to Enabled=yes returns the option SSO Toolbar.

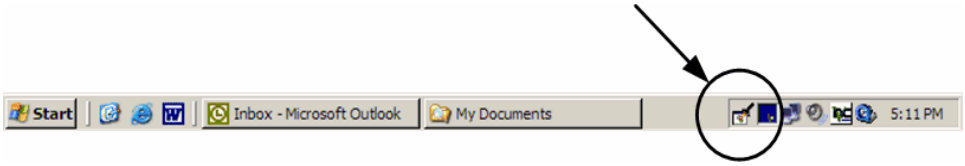
Important! *After modifying the token value in the SsoClnt.ini file, you must right-click the eTrust SSO Agent icon, select Exit, and re-run the SSO Client for the modification to take effect.*

For more information about the SsoClnt.ini file, see the Appendix section.

The following sections describe each interface in detail.

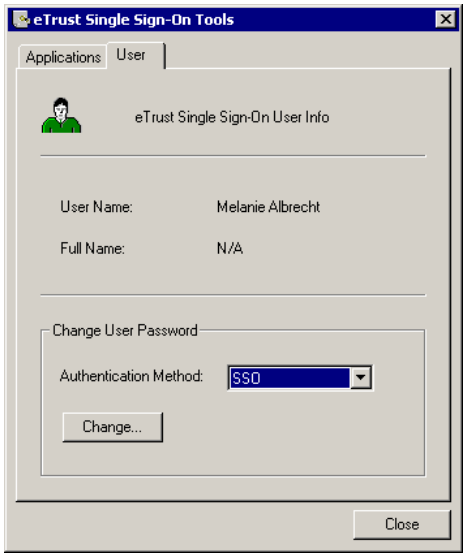
The SSO Tools Dialog

To open the eTrust SSO Tools dialog, right-click the eTrust SSO icon in the system tray, then select **Open SSO Tools**.



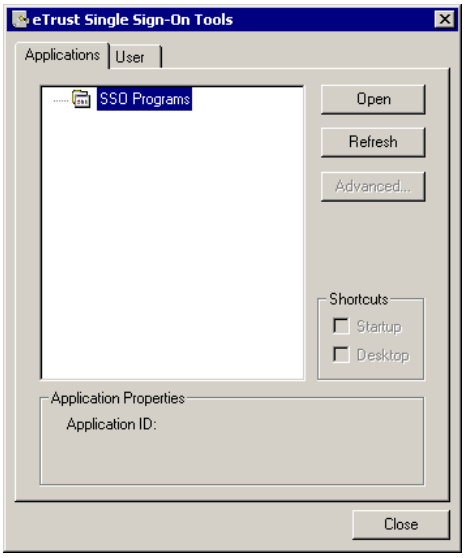
The eTrust SSO Tools dialog has two tabs: **User** and **Applications**.

User Tab—This page allows the user to see some user information and to change their primary authentication password.



To change the password for primary authentication, the user clicks the Change User Password button and fills in the boxes in the Change Password dialog that is displayed.

Applications Tab—This page allows the user to see and refresh their application list, and to launch applications.



The Application tab allows users to:

- View application properties for the selected application.
- Change the application password, by clicking the Advanced button to receive an Advanced Settings dialog.
- 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.

The SSO Toolbar

Once the user authenticates, the SSO Toolbar is automatically displayed as a floating toolbar on the screen.

The first display of the SSO Toolbar looks like this:

In this toolbar there are two icons: Logon and Lock.

The Logon icon— Allows the user to log onto the system and authenticate. After clicking the icon, an authentication tabbed window appears for the user to authenticate.

The Lock icon— Locks the station.

Users log on to SSO by clicking the Logon icon. Then, the SSO Toolbar shows the application list specific to the user. This is an example of a user's application list:

In this toolbar there are several icons:

Application icon—The user can access any application by clicking the corresponding application icon.

The Lock icon—With a single click the user locks the workstation.

The Logoff icon— With a single click the user logs off the workstation. Logging off automatically closes all the applications.

Note: The SSO Toolbar does not support container applications.

eTrust SSO Agent Pop-Up Menu

In the system tray, right-click the eTrust SSO agent icon to display a menu with the following items:

- Refresh Application List
- Open the SSO Tools, which opens the SSO Tools window. This is displayed only if the SSO Tools option was enabled.
- Remove Application List, which removes the application list from the Start menu. The application list can be restored by selecting Refresh Application List.
- Lock station (only in Windows 98SE).
- About SSO Client, which displays a message box with the SSO Client version number and information.
- Exit, which closes the SSO Client. When you select Exit, eTrust SSO displays a warning message and asks for confirmation.

Displaying the List of Applications

The SSO Client gets the application list from the Policy Server and displays it to the user.

To receive the application list, the user must already be authenticated. If the user's primary authentication was not successful, or if the user is marked as disabled in the data stores, the Policy Server does not send the application list to the user's workstation.

The Application List

The application list for each user contains all the SSO-supported applications that the user is allowed to use. The Policy Server dynamically builds the list every time it is needed.

In addition, the Policy Server can now rebuild the users' application lists periodically. For more information, see the section The Application List Background Calculation Utility in the 'Managing Services' chapter.

Changes made to the data stores affect the application list displayed. For example, if the user is joined to a group, all the applications that the group can access will appear on the user's application menu the next time the user logs into eTrust SSO or when the user refreshes the display.

The application list includes the following types of applications:

- Applications that are explicitly allowed to the user
- Applications that are part of application groups that are explicitly allowed to the user
- Applications that are allowed to one of the user groups to which the user belongs
- Applications that are part of application groups that are allowed to one of the user groups to which the user belongs
- Common applications—applications that have their default access property set to Execute or All so that any user authorized to use eTrust SSO can use them
- Container applications—applications used to join related applications in logical groups in the application list. Container applications are defined in the data stores. Container applications are not visible in the toolbar, but the applications that they contain are visible.

Note: The application list can be displayed in two different user interfaces: the SSO Tools and the SSO Toolbar.

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.

eTrust 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.

Note: You can change the Start menu item name SSO Programs by changing the value of the SSOFolderName token in the SsoClnt.ini file. However, you must be careful when selecting a new value for SSOFolderName. If, for example, you set the token SSOFolderName=Programs, you must set also the token MergeLinks=YES to allow merging the SSO programs with the start menu programs.

Similarly, you can change the default icon for applications by editing the following entries in the SsoClnt.ini file:

```
[sso]
DefaultAppIcon = C:\Win95\system\shell32.dll
DefaultIconOrder = 13
```

DefaultAppIcon provides a path to an EXE or DLL file that contains one or more icons. DefaultIconOrder defines which icon to use (the numbering starts from 0).

The default icon can be changed for applications only, not for containers, since the folder icon is selected automatically by the Windows operating system.

The application list can also be displayed using the SSO Tools or the SSO Toolbar.

Icons and Icon Names

The data stores contain the name of the file in which the application icon is located. The system administrator specifies the name of this file when defining a new application.

The icon itself is not kept in the data stores. The file containing the icon can be an ICO, EXE, or DLL file and must be available to the SSO Client when the latter is started. The administrator can specify the full path of this file or just the file name. In the latter case, the file must be available to the SSO Client according to the module search rules of Windows.

The data stores also contain the caption—the icon name—that appears under the icon of each application. The system administrator specifies this caption when defining a new application. If the administrator does not specify the caption for an application, the name of the application from the data stores is displayed by default.

Message of the Day

The system administrator can define a Message of the Day (MOTD). MOTDs enable the administrator to notify users about changes in working procedures, upcoming events, and so forth. There are two types of MOTDs:

Global MOTD—Messages for all SSO users. They will be displayed on the end-user workstation when the SSO Client is started.

Application MOTD—Messages for the users of a specific SSO-supported application. These will be displayed on the end-user workstation when the application is invoked by the user. A different message can be attached to each application.

The text of these messages resides on the Policy Server and is sent to the user's workstation when needed. Each message resides in a separate file, in the motd directory in the Policy Server installation area. You can change the directory by specifying a value for the MotdPath token in the ssod.ini file for UNIX servers, or in the Windows Registry, for Windowsservers.

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, eTrust 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:

SsoAgent.exe—This executable handles the following functions:

- Primary authentication of the user
- Building the links to the applications
- Handling the SSO Tools calls

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

- Communicating with the Policy 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

SsoTools.exe—This is a GUI for executing end–user activities such as password change and application list refresh

SsoClnt.ini—This is a file containing SSO Client configuration information; it is placed in the same directory as the SSO Client executables. For further information about this file, see the appendix 'Configuring the SSO Client.'

Authentication DLLs—These DLLs are located in the Auth subdirectory. Examples of dlls are: sso.dll, sso00.dll, NT.dll, NT00.dll, and so forth.

Additional files—Files that the SSO Client requires, including DLLs, are installed in the same folder as the executables.

In general, eTrust 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.
- In a batch installation of the SSO Client the StationLock feature is installed and the original password.cpl file Windows SYSTEM folder is renamed and replaced with an eTrust SSO password.cpl file. The SSO Client can be installed without StationLock and the password.cpl file will not be changed.
- 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.

Chapter

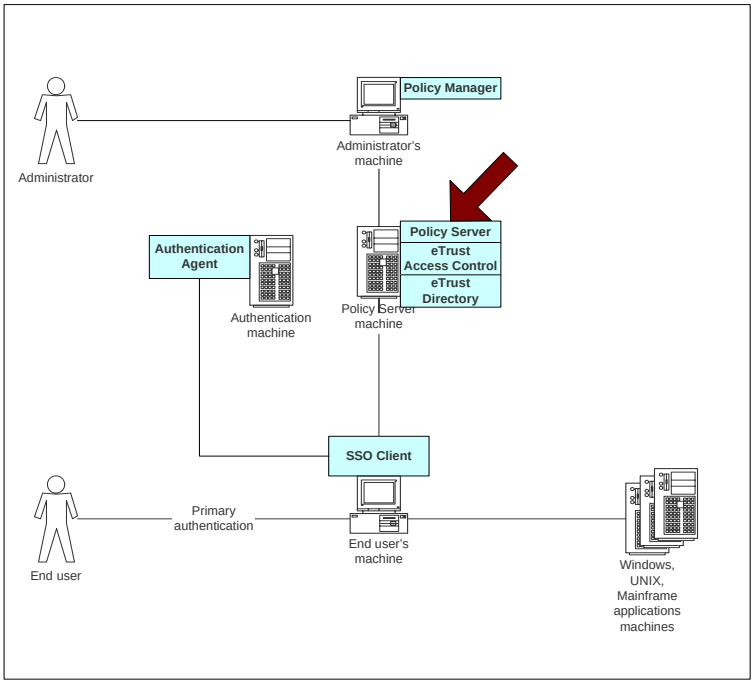
3

The Policy Server

The Policy Server is the center of eTrust Single Sign On. It resides on a UNIX or Windows server, and manages resources and provides services to the SSO Client.

The Policy Server performs the following functions:

- Authentication and authorization:
  - Builds the list of applications that a user is allowed to access and sends it to the SSO Client
  - Controls who can access which web resources
  - Controls who can access the data held in the eTrust SSO data stores
- Policy and user management:
  - Manages users and resources in the eTrust Access Control database
  - Retrieves the logon scripts and the user-specific logon data for each application
- Auditing, logging and tracing



Working with the Policy Server

You can control the Policy Server by using the Policy Manager, or from the command line using selang commands:

The *Policy Manager* is a Windows GUI that lets you work with users, groups, and other resources. For more information, see the chapter The Policy Manager.

Selang is a Computer Associates proprietary command language that allows you to interact with the eTrust Access Control database via the Policy Server.

Selang commands are used in the initial stages of eTrust SSO implementation and for batch operations. If batch operations are needed for changes in many user or application definitions, you can write commands and run them as a script in the background.

Later, when eTrust SSO has been installed and is running smoothly, you can use the Policy Manager instead of selang for most tasks.

For more information about selang see the Selang Command Reference Guide.

eTrust Components Installed on the Policy Server Computer

The server on which the Policy Server is installed also includes other eTrust SSO components:

- Data stores
- Logon scripts
- Utilities:
  - Application List Background Calculation Utility (psbgc)
  - Automatic Password Generation Utility (genkeypair)
- Initialization files (UNIX only)
- 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

eTrust SSO is installed with a copy of eTrust Access Control, and eTrust Directory. These can store all of the data that eTrust SSO requires.

The data that eTrust SSO requires is stored in three data stores:

Name of Data Store	Data	Stored In...
User data store	Users and user groups	Either one of: ▪ An LDAP directory (such as eTrust Directory) ▪ eTrust Access Control
Policy data store	▪ Applications & groups, authentication hosts & groups, password policies ▪ Rules of user access to applications ▪ Agents ▪ Administrators	eTrust Access Control
Token data store	The user's session details: ▪ Session ID ▪ Client IP number ▪ User name ▪ Last heartbeat time	An LDAP directory (such as eTrust Directory)

You can populate the eTrust Access Control database with user and group information from existing databases in your organization, during or after product installation. You can conveniently import user and group information by running a utility, or by using selang commands.

eTrust Directory is designed to efficiently manage thousands of users, which significantly enhances the performance and scalability of eTrust SSO. The eTrust Directory data store is perfect for large enterprise installations.

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

For more information about the data stores, see the chapters The User Data Store, and The Policy Data Store, and The Token Data Store.

Note: By default the Policy Server only recognizes English characters. If you are want to enter data with non-English characters you must set the Policy Server to recognize those characters. For more information about these settings, see the "Policy Server Settings" appendix in this guide and refer to the "DefaultLocate" setting.

Logon Scripts

Logon scripts are also filed on the same server host as the Policy Server.

These provide the instructions that the SSO Client executes to log an end user into an application.

For more information about logon scripts, see the Running Logon Scripts section in the "SSO Client" chapter.

Utilities

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

The Application List Background Calculation Utility (psbgc)

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

The utility psbgc compiles application lists as a background task, which reduces the Policy Server load and improves the 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 eTrust SSO. This reduces help desk calls from users who forget passwords and can improve the system administrators' tracking of application use.

Initialization File (UNIX Only)

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

```
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\eTrust\Shared\Policy Server\8.0
```

On UNIX, the Policy Server is configured in the policyserver.ini file.

For a description of the Policy Server settings that you can adjust, see the appendix 'Configuring the Policy Server (policyserver.ini or the Windows Registry).'

Message of the Day Files

The system administrator can define a Message of the Day (MOTD). MOTDs enable the administrator to notify users about changes in working procedures, upcoming events, and so forth.

The text of these messages resides on the Policy Server and is sent to the user's workstation when needed.

Using 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.

If you can create a network connection between two or more Policy Servers, each one can function as the main server for a number of workstations running SSO Client, while at the same time serving as a backup Policy Server for all the other Policy Servers.

When the replication mechanism is configured and running, all Policy Servers using it will simultaneously update the local data stores and all the data stores of the other Policy 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.

For more information about server farms, backups and failover, see the 'Working with Server Farms' chapter.

Directory Structure on Windows and UNIX

The following tables show the directory structure of the Policy Server on a UNIX server and on a Windows server.

The following table explains the directory structure of the Policy Server on a UNIX server relative to a base directory. This base directory is usually /opt/CA/eTrustSingleSignOn/PolicyServer.

UNIX Directory or File	Description
applica.ini	
appticket.key	
bin/	Policy Server daemon, startserver, stopserver, deinstall and GenKeyPair
config/	eTrust Directory database configuration files. This database was installed when you installed the Policy Server.
exits/	Password exits directory
lib/	
lic98.tar	
lic_98_install	
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.
samples/	Sample directory
scripts/	Script directory
wac.msg	

The following table explains the directory structure of the Policy Server on a Windows server relative to a base directory. This base directory is usually <System-drive>: \Program Files\CA\eTrust PolicyServer.

Windows Directory or File	Description
bin\	Policy Server process, GenKeyPair
Doc\	User documentation
Exits\	Password exits directory
Log\	Log directory
Motd\	"Message of the Day" for applications, one for each application and general Message of the Day. These files can be stored in a different directory.
polsrv.key	File of keys generated.
PsBgc\	Application lists calculated by the psbgc utility.
Samples\	Sample directory
Schemas\	eTrust Directory and Active Directory database configuration files The eTrust Directory database was automatically installed when you installed the Policy Server unless you specifically chose not to install it.
Scripts\	Script directory
pslog.ini	Log configuration file for the Web Agent

The features and functions for administration of eTrust Identity and Access Management (eTrust IAM) are built into the IA Manager. This chapter covers some of the tasks and activities you might need to perform after eTrust IAM is installed.

Starting the eTrust IAM Web Applications

The eTrust IAM web applications are installed by the Common Components installer. These applications are:

- eTrust Identity and Access Manager
- eTrust IAM Configuration
- eTrust IAM Self Service
- eTrust IAM SPML Service Configuration

Each of these web applications starts with a login screen which requires a username and password to access them. Most of these applications are typically accessed only by help desk staff and network administrators although the Self Service application may be configured for a wider group of users.

Each of the installed web applications are available via a program shortcut created on the computer where the IAM common components are installed. The shortcuts are located in the Windows Start menu under Programs, Computer Associates, eTrust, eTrust Identity and Access Management.

During rollout of eTrust IAM to your organization, you should implement a method for authorized users to easily access these web applications such as providing a desktop shortcut, browser bookmark, intranet link or Start Menu item on all desktops.

To start one of the web applications, follow these steps:

1. Manually enter the relevant URL into your internet browser:
2. IA Manager –

   `https://<hostname.company.com>:<sslport>/CA/IAM/Manager/`
3. IAM Configuration –

   `https://<hostname.company.com>:<sslport>/CA/IAM/Config/`
4. IAM Self Service –

   `https://<hostname.company.com>:<sslport>/CA/IAM/SelfService/`
5. IAM SPML Service Configuration -

   `http://<hostname.company.com>:<port>/iamspml`

   where:

   - <*hostname.company.com*> is the fully qualified hostname of the computer acting as the Web Application Server.
   - <*sslport*> is the Apache Tomcat SSL HTTP port number (the default value is 8443).

– <*port*> is the Apache Tomcat HTTP port number (the default value is 8080).

The login prompt appears.

6. Log in by entering the required fields and clicking the Log In button.

User Name
Enter an authorized username, such as etaadmin.

Password
Enter the password for the specified user. The initial password for etaadmin will have been provided when installing the IAM common components.

Admin Server
Required by SPML Service Configuration application only. Enter the name of the Admin Server for the specified user.

Domain
Required by SPML Service Configuration application only. Enter the name of the domain.

The relevant web application interface opens.

IA Manager Manual Configuration

During installation you configure some of the functionality of IA Manager however some parameters must be changed manually after installation.

Note: Always make a backup copy of the configuration file before you modify it.

To manually configure the parameters of IA Manager, follow these steps:

1. Using Windows Explorer, browse to the IA Manager directory. By default this directory is Program Files\CA\eTrust Identity and Access Manager\WEB-INF.

   The contents of this directory appear in the right pane.

2. Right-click the etwebadmin.properties file and choose Open With.

   The Open With dialog appears.

3. Select a text editor such as Notepad and click OK.

   The etwebadmin.properties file opens.

4. Change the appropriate configuration values, and then save and close the file.

   Note: Modify only those values that display in the Configuration Parameters list.

   The text editor closes.

5. Restart the Apache Tomcat service in Windows Services.

   The manual changes are applied.

Configuration Parameters

You can modify many default configuration values during the installation of IA Manager. After installation, these configuration values can be changed by editing the configuration file manually.

Note: Parameters, paths, or text strings that contain spaces must be entered in quotation marks ("xxx") preceded by a back slash (\). For example, you could enter \"C:\Program Files\CA\".

During silent mode installation, configuration values with a command line parameter can be set from the command line. However, not all configuration parameters have a command line parameter.

def_search_results
Defines the maximum number of search results to return in the search pane.
Default: 100

eta_domain
Command Line Parameter: ETASERVERDOMAIN
Defines the eTrust Admin Server domain name. This is usually the name of the computer on which eTrust Admin Server is installed, unless changes were made during that installation.
Default: localhost_name

ldap_host
Command Line Parameter: LDAPHOSTNAME
Identifies the computer that serves as the LDAP host. This is the computer on which eTrust Admin Server is installed.
Default: localhost_name

ldap_port
Command Line Parameter: LDAPPORT
Defines the LDAP port number.
Default: 20389

ldap_tlsPort
Command Line Parameter: LDAPTLSPORT
Defines the LDAP port number using TLS/SSL encryption.
Default: 20390

ldap_useTls
Indicates whether to use TLS/SSL encryption. 1 indicates to use TSL/SSL encryption; 0 indicates not to use the encryption.
Default: 1

log_error_detail
Indicates the level of errors logged. Enter a level from 1 to 4.
Default: 4

log_enabled
Indicates whether to enable logging. 1 indicates that logging is enabled; 0 indicates that it is disabled.
Default: 1

Max_Picture_Size_Kilobytes
Defines the maximum file size (in kb) that can be uploaded as a user photograph.
Default: 25

pass_rst_req_enabled
Indicates whether a user can request a password reset. 1 indicates that this option is enabled; 0 indicates that this option is disabled.
Default: 1

pass_rst_req_smtp
Command Line Parameter: SMTPSERVER
Defines the SMTP mail server.
Default: your_smtp_server

pass_rst_req_user
Command Line Parameter: ADMINEMAIL
Defines the administrator's email address.
Default: admin@your_company.com

self_admin_unlock_account_enabled
Indicates whether the administrator can unlock an account. 1 enables this option; 0 disables this option.
Default: 1

web_admin
Defines the user name of the Web application administrator.
Default: etawebad

Note: You can modify the web_admin attribute only if the password for the web_admin user is **changeoninstall**.

Troubleshooting IA Manager

IA Manager has many components which may require separate administrative tasks. This section describes some of the more common administrative tasks that you may perform.

Start or Restart a Windows Service

The Windows services required by IA Manager include:

- Apache Tomcat Service – The default service name is "CA Tomcat 4.1.29 eTrustIAMWebServer".
- Advantage Ingres – The default service name is "Ingres Intelligent Database [ET]".
- Directory Web Server – The default service name is "eTrust Directory Web Server".
- SSO Session Administrator – The default service name is "eTrust SSO Session Administrator".

To start a service running in Microsoft Windows Services, follow these steps:

1. Open the Windows Control Panel from your Start menu. Double-click Administrative Tools and then Services.

   The Services dialog appears.

2. Locate the appropriate service and right-click it.

   A pop-up menu appears.

3. If it is not already started, choose Start from the pop-up menu. If it is already started, choose Restart.

   The service will be listed as Started.

   Note: This service should start automatically after you have installed the IA Manager and also every time you start your machine. Verify that the service is listed as Automatic to ensure that it starts on a computer reboot.

4. Repeat this procedure for any other services, as necessary.

Stop a Windows Service

If a Windows service conflicts with services required by IA Manager, it may be necessary to stop the Windows service.

To stop a service in Microsoft Windows Services, follow these steps:

1. Open the Windows Control Panel from your Start menu. Double-click Administrative Tools and then Services.

   The Services dialog appears.

2. Locate the appropriate service and right-click it.

   A pop-up menu appears.

3. Choose Stop from the pop-up menu.

   The selected service stops.

Note: After you stop a conflicting process, you must start/restart the service that was in conflict. For more information, see Start a Windows Service.

Securing your eTrust Web Applications

The eTrust IAM suite of products rely on web applications which are secured using the SSL protocols with self-signed certificates. In order to fully secure your product suite you should replace the self-signed certificates with certificates signed by a certifying authority.

If your organization does not have an internal certifying authority:

- Use Self Signed host certificates. This is how the default IAM installation is created as it is impossible for the installer to create widely trusted certificates. Although this installation is not optimal, it can be useful for testing. If you wish to install self signed certificates on a web server not installed by the IAM installation see Replace the Tomcat SSL Certificate.

  Using Self Signed certificates has the following drawbacks:

  - Client computers cannot be assured of the identity of the web servers.
  - Each time a browser connects to a web server a warning about invalid certificates will be presented. This can confuse users and be a potential source of help desk calls.

- Use host certificates from a trusted Certifying Authority. This is the recommended option.

  1. Obtain host certificates and private keys for all web server hosts from a trusted Certifying Authority such as Verisign or Thawte.
  2. Install these host certificates and keys in your Tomcat keystore on each web server host. See Replace the Tomcat SSL Certificate.

If your organization already has an internal certifying authority:

1. Issue host certificates and private keys for all web server hosts.
2. Install these host certificates and keys in your Tomcat keystore on each web server host. See Replace the Tomcat SSL Certificate.
3. Install the host certificates in the browser keystore on all client computers.
4. Install the host certificates in the Tomcat keystore on all server computers.

Configure SSL Support for Tomcat

All computers serving web applications for eTrust IAM must have Tomcat configured to use SSL protocol. This is the default when installing Tomcat with the IA Manager software; however, if you installed Tomcat independently, it may not be configured with SSL support.

To install and configure SSL support for Tomcat using a self-signed certificate, follow these steps:

1. Verify that JDK version 1.4.2_04 is installed by checking the Add/Remove Programs list in your Control Panel for the program Java 2 SDK, SE v 1.4.2_04.
2. Create a new keystore containing one self-signed certificate by entering the appropriate command from the command prompt.

   On Windows systems, you should enter:

   ```
   %JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA -keystore
   \path\keystore_filename
   ```

   On UNIX systems, you should enter:

   ```
   %JAVA_HOME%/bin/keytool -genkey -alias tomcat -keyalg RSA -keystore
   \path\keystore_filename
   ```

   The keystore creation process begins.
3. Enter the keystore password when prompted.

   Note: The default password used by Tomcat is **changeit** (all lower case). If preferred, you can specify a custom password, but you must then specify the custom password in the server.xml configuration file also (see Step 8).

   The keystore creation process continues.
4. Enter general information for the certificate when prompted. The general information includes company, contact name, and so on. This information displays to users who attempt to access a secure page in your application, so make sure that the information provided here is appropriate.

   The keystore creation process continues.
5. Enter the key password when prompted. This password is created specifically for this certificate (as opposed to any other certificates stored in the same keystore file). You must use the same password for this and the keystore password.

   A keystore file with a certificate that your server can use is created.
6. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default location for the <*Tomcat_installation_directory*> when installed from the IA Manager software is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29.

7. Ensure that the SSL Coyote HTTP/1.1 Connector entry is not commented out in the file. The connector information looks similar to the following:

```
<!-- Define an SSL HTTP/1.1 Connector on port 8443 -->
<!--
<Connector className="org.apache.catalina.connector.http.HttpConnector"
      port="8443" minProcessors="5" maxProcessors="75"
      enableLookups="true" acceptCount="10" debug="0" scheme="https"
      secure="true">
 <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
      clientAuth="false" protocol="TLS"/>
</Connector>
-->
```

   If the Connector element is commented out, you must remove the comment tags (<!-- and -->) around it.

8. Configure the SSL Coyote HTTP/1.1 Connector entry to include the keystoreFile and keystorePass attributes for the Factory element.

   keystoreFile
   Specifies the location where the keystore file is located

   keystorePass
   Specifies the keystore (and certificate) password

   The connector information should look similar to the following:

```
<Connector className="org.apache.catalina.connector.http.HttpConnector"
      port="8443" minProcessors="5" maxProcessors="75"
      enableLookups="true" acceptCount="10" debug="0" scheme="https"
      secure="true">
 <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
      keystoreFile="your_keystore_full_path"
      keystorePass="your_keystore_password"
      clientAuth="false" protocol="TLS"/>
</Connector>
```

9. Save the file and close it.

   SSL support and self-signed certificates are configured for Tomcat.

For more information or to obtain and install a certificate from a certificate authority (such as verisign.com, thawte.com or trustcenter.de), see the Apache Tomcat document “SSL Config HOW-TO”, available at http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html (http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html).

Replace the Tomcat SSL Certificate

Your Tomcat SSL Certificate enables users browsing to your web pages to transmit and receive secure information. Certificates may be self-signed or provided by an independent third party. You can update your Tomcat SSL Certificate to improve the security of the information shared via web browsers.

To replace the Certificate for Tomcat SSL support, follow these steps:

1. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default location for this file is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29\conf\server.xml.

   The text file opens.

2. Search for the following text: keystoreFile. Change the attributes to present the replacement information.

   keystoreFile
   Specifies the location in which the keystore file is located.

   keystorePass
   Specifies the keystore (and certificate) password.

   The file reflects these changes.

3. Save the file and close it.

   The information is stored.

   Note: For more information, see the Apache Tomcat document "SSL Config HOW-TO", available at http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html (http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html).

Resolving Port Conflicts Manually

This topic is included for resolving Apache Tomcat port conflicts. For information about how to resolve Apache Tomcat port conflicts not covered here, see your Apache Tomcat documentation or the Apache Tomcat http://www.jakarta.apache.org/tomcat web site.

If you cannot run a particular installation of Apache Tomcat, one or more of the ports for which it is configured may currently be in use by another program (for example, another installation of Tomcat or some other web server). To rectify this problem, you must reconfigure the "broken" Tomcat to use different values for the Shutdown port, the Non-SSL HTTP port, and the SSL HTTP port.

To reconfigure the port values, follow these steps:

1. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default full path for this file is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29\conf\server.xml.

2. Search for the following text: Server port=.

   The text is highlighted.

3. Change the value that appears after Server port =. For example, if the number is 8005, change it to 9005.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's Shutdown port.

4. Now, search for the following text: Define a non-SSL Coyote HTTP.

   The text is highlighted.

5. From this position in the file, search for the following text: port=.

   The text is highlighted.

6. Change the value that appears after port=. For example, if the number is 8080, change it to 9080.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's non-SSL HTTP port.

7. Now, search for the following text: Define a SSL Coyote HTTP.

   The text is highlighted.

8. From this position in the file, search for the following text: port=.

The text is highlighted.

9. Change the value that appears after port=. For example, if the number is 8443, change it to 9443.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's SSL HTTP port.

10. Return to the beginning of the file and search for all instances of the following text: redirectPort=. For each instance, change the value that appears after redirectPort= to the same number that you entered for the new port number for Tomcat's SSL HTTP port.

    Note: Tomcat's SSL HTTP Port and Redirect port must use the same number.

11. Save the file, close it, and try to run this particular installation of Apache Tomcat again. Verify that Tomcat started correctly by entering the following URL in your web browser: http://localhost:8080/index.jsp. If you see the Apache Tomcat web page, it has started correctly. If not, then Tomcat has not started.

    If Tomcat starts correctly, you have successfully resolved your Tomcat port conflicts.

    Note: For more information about how to start Tomcat, see Start Apache Tomcat or see your Apache Tomcat documentation.

Chapter

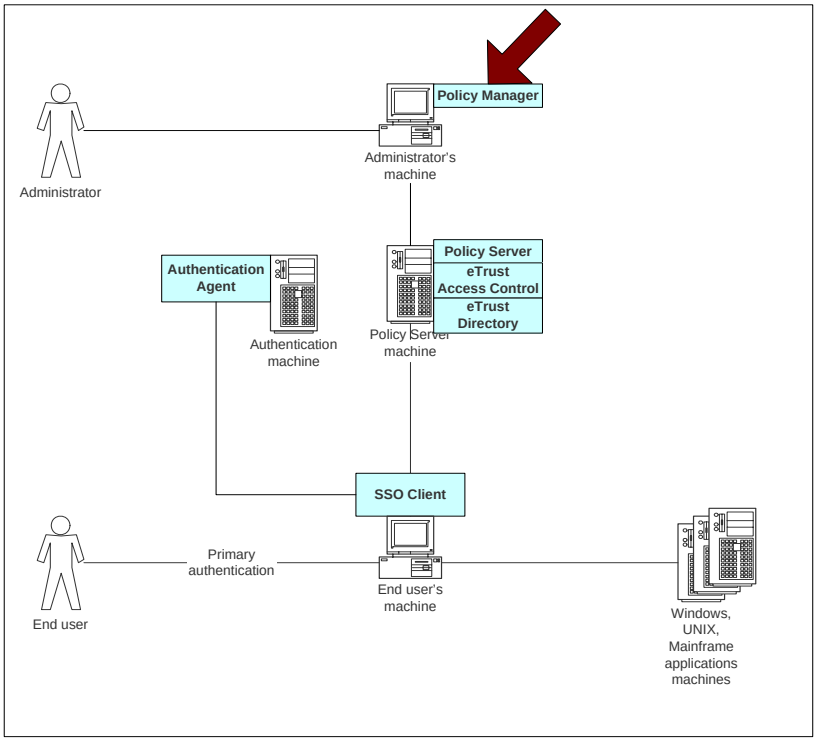
4

The Policy Manager

This chapter describes the Policy Manager. The Managing Users chapter and the Managing Resources chapter describe how to use the Policy Manager to work with eTrust SSO data.

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

- Users and user groups
- Applications and application groups
- Authentication hosts and authentication host groups
- Password policies
- Terminals on which eTrust SSO software is installed
- Other eTrust SSO resources



The Policy Manager generates commands and sends them to the Policy Server, which updates the data stores (eTrust Directory, eTrust Access Control, and others).

The Policy Manager can run on Windows NT/2000/XP/2003 workstations that have a TCP/IP connection to the Policy Server. You can use the Policy Manager to communicate with both UNIX and Windows Policy Server computers.

Almost everything that can be done with selang commands in UNIX can be done with the Policy Manager.

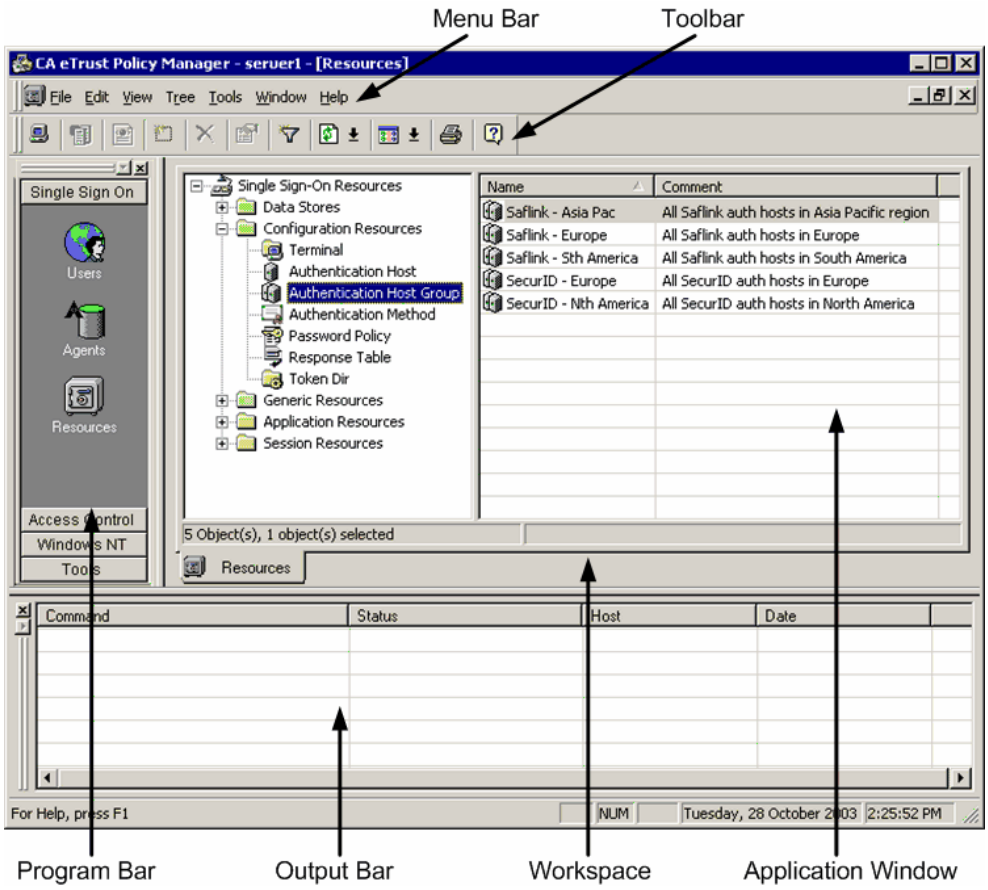
You can also use the Policy Manager to manage data in the following other eTrust applications, by changing the operation mode:

- eTrust Access Control
- eTrust Web Access Control

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.

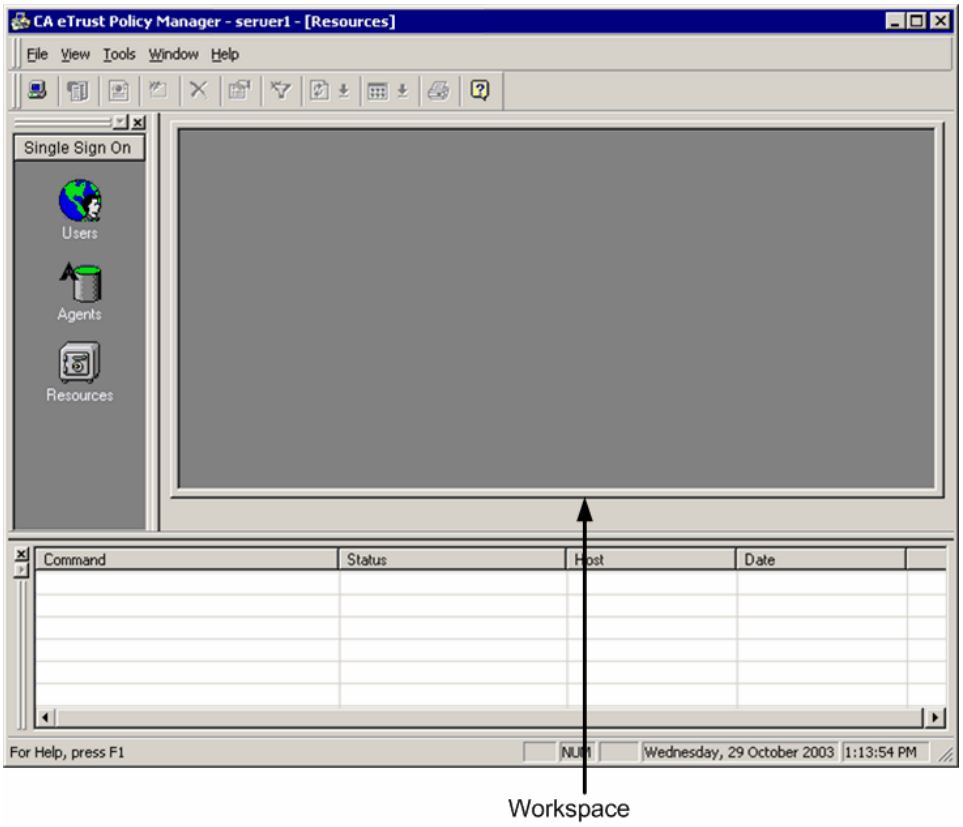
This is a sample Policy Manager window:



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

Workspace

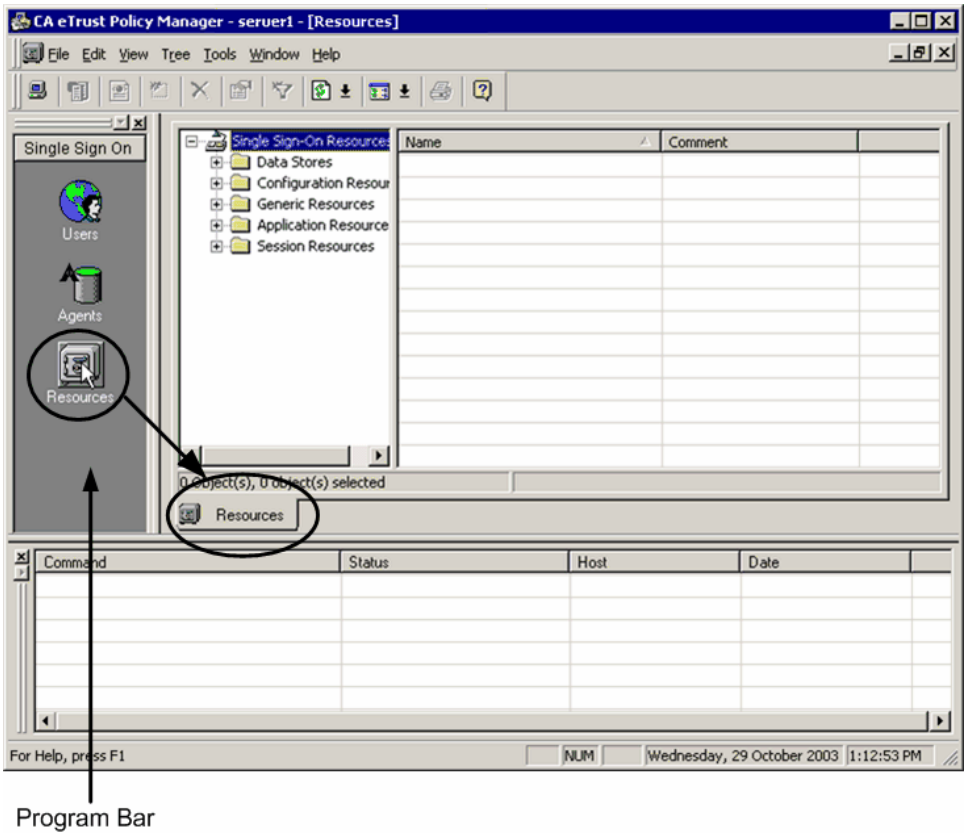
When the main window of the Policy Manager first appears, there is no application window section—only an empty workspace:



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:



To display the panels on the program bar, click the buttons labeled SSO, Access Control, Windows NT, and Tools.

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

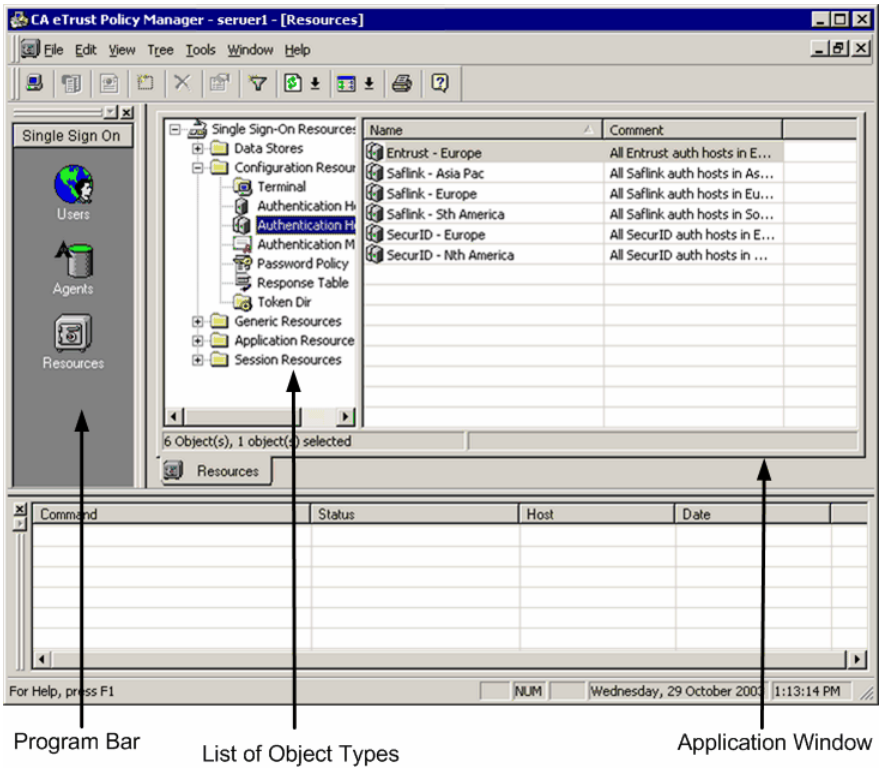
Agents—Lets you create and delete agents and agent types.

Resources—Lets you manage the data stores, configuration resources, generic resources, application resources, and agents in your policy data store.

Application Windows

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

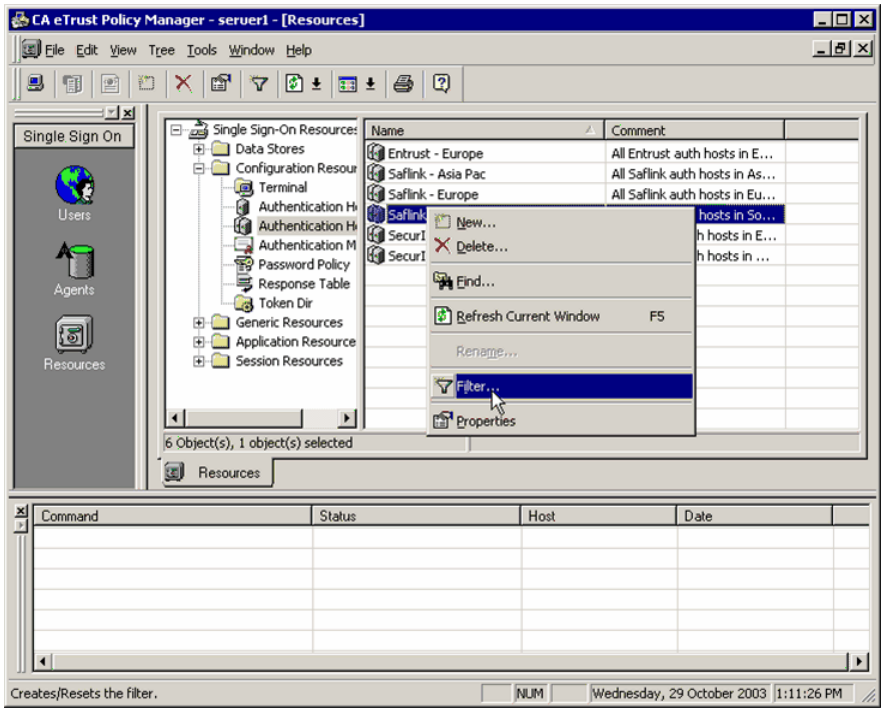
To open an application window, click an icon in the program bar, or click an option in the File, Open menu.



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.

Working 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
- Find a specific entry in the data store
- Rename the selected entry
- 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)

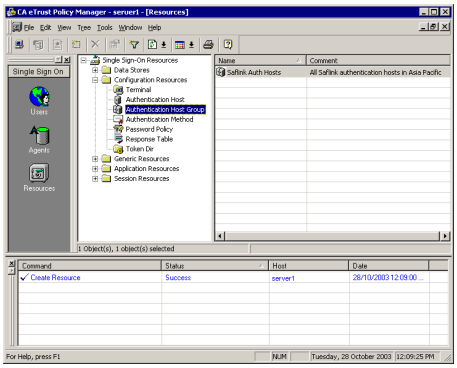
Workbook Mode

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

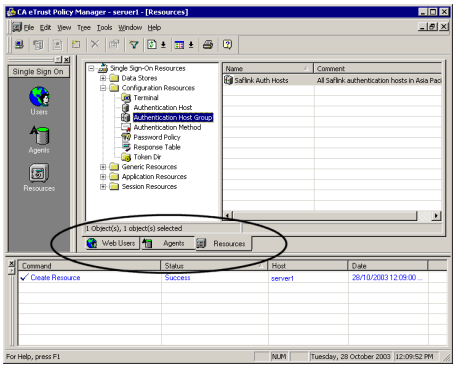
If Workbook Mode is switched on, each application window remains open, and you can quickly return to it by clicking the tab for that window.

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: SSO Users and Resources.



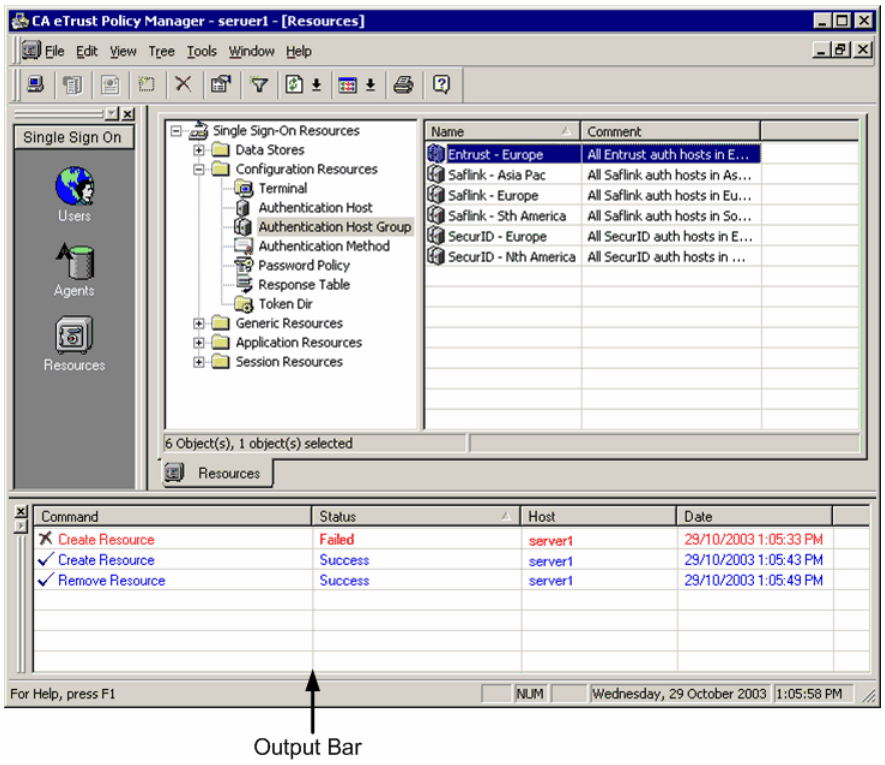
Policy Manager with Workbook mode off



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

Output Bar

Whenever you use the Policy Manager to make a change in a data store, the GUI generates selang commands, which it sends to the data store. The output bar displays a condensed version of the selang messages that are returned when you make these changes to a data store. For more information about selang, see the appendix 'Using Selang.'.

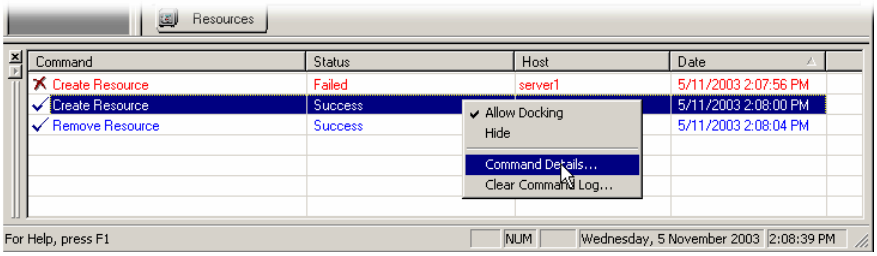


The commands that are displayed in the output bar are also recorded in the command log.

To limit which commands should 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 should 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 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.

Wizard Manager

The Wizard Manager button lets you activate the most commonly used wizards. You can click the button to open a window for selecting wizards, or click the arrow to select a wizard to activate. You can also activate wizards by using the Tools menu.

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.

Working with the Policy Manager Window

Starting the Policy Manager

To start the Policy Manager and work with it, you must be logged in as an eTrust SSO administrator.

1. Find out the following information:
   - The name of the machine on which the Policy Server is running

     If it is not running, start the Policy Server from the Services dialog on the Control Panel.
   - The user name and password of an eTrust SSO administrator

     If you are signing on for the first time after eTrust SSO was installed, use the LDAP administrator name and the SSO authentication method. The default LDAP administrator name is ldap-admin. This is defined during the IAM Common Components installation when you configure the Policy server.
   - Host name (DNS name or IP address) of the computer on which the Policy Server is running

     If you are logging on from the computer where the Policy Server is installed, use localhost as the name of the host.
2. On the Policy Manager computer, from the Start button on the taskbar, choose Programs, Computer Associates, eTrust, Policy Manager. The Login dialog appears.
3. Fill in the Login dialog with the following authentication information:
   - User name of an eTrust SSO administrator
   - Password for the administrator, which was specified during installation
   - Host name of the computer on which the Policy Server is running

     You can use the Browse button to locate the name of the host.

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

Moving, Hiding, and Displaying 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.

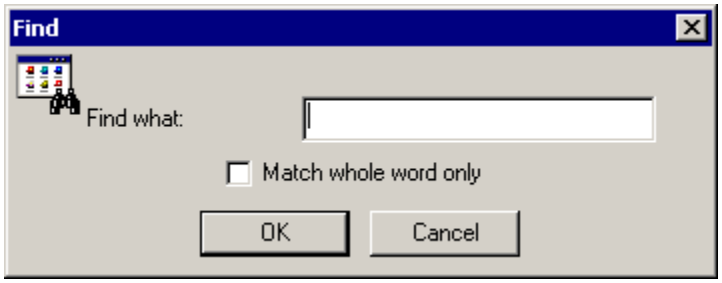
Finding, Filtering, and Sorting Entries

As you continue to use eTrust 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.

Finding an Entry

To find a particular entry in a list:

1. Navigate to the list in which you want to search.
2. Choose the Find command from the Edit menu. This displays the Find dialog.



3. 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.

Filtering 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:

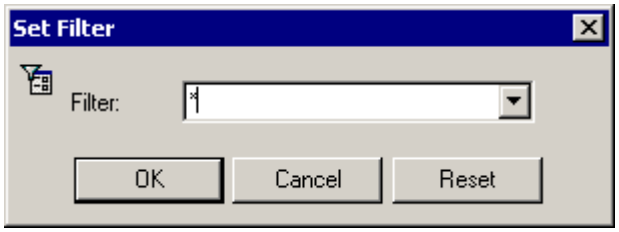
```
ber*
```

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

```
*USER*
```

To filter a list:

1. Choose the Edit, Filter menu option to display the Filter dialog:

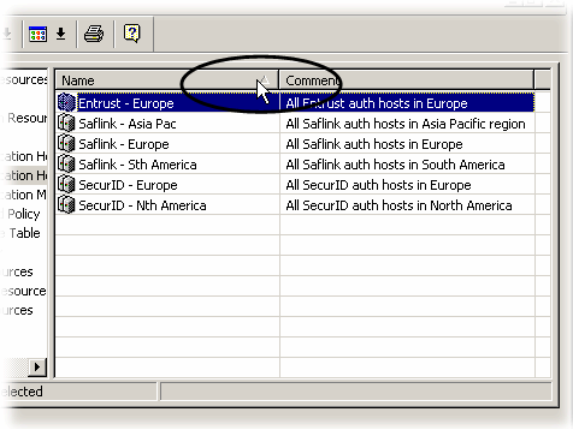


2. In the Filter field enter a string of characters that is found in all of the names of the entries that you want to display.

To redisplay the entire list of entries, click the Reset button, and then click OK.

Sorting 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.

Refreshing 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.

Customizing the Policy Manager

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

To change the default settings, choose Options from the Tools menu to display the Options dialog. 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 eTrust startup options and custom splash screen.
Create	Defines whether users and groups are created in the native operating system or in the eTrust SSO environment, or both. Also defines whether resources are always created with the wizard.
Accounts & Resources	Enables B1 security features, sets whether UNIX account details are shown on Windows, and defines whether Windows and UNIX resources are displayed in the resource tree when connected to a policy model
Command Log	Defines what types of commands appear in the Policy Manager command log window.
Format	Defines colors for different kinds of resources and users.
Password	Defines options for automatically generated passwords
Mail Contents	Defines an email message to be sent automatically when a user's password has been changed.
Mail Configuration	Sets whether emails from the Policy Manager are sent by SMTP or MAPI.
Transaction Mgr	Sets and modifies options for the transaction manager
Connections	Specifies the connection timeouts between the Policy Manager and the Policy Server.

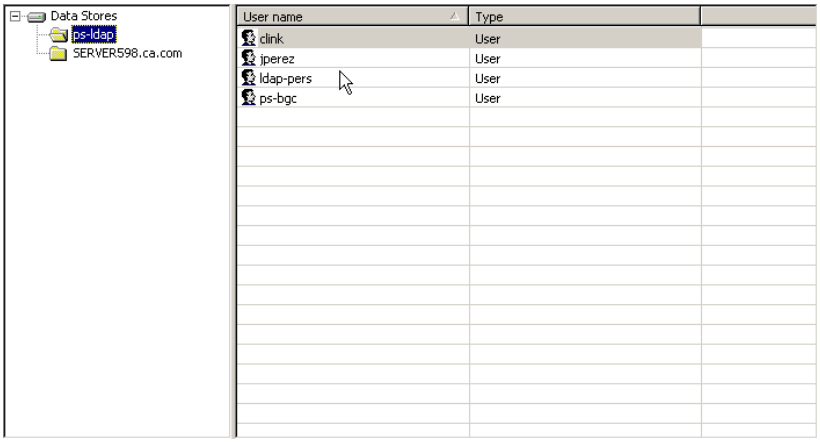
Chapter

5

Managing Users and User Groups

You can define and maintain all of the users and user groups in the user data store from the Users application window:

To display this window, click the Users icon in the program bar.



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

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 should include all the users who will use eTrust SSO. This includes system administrators as well as end users.

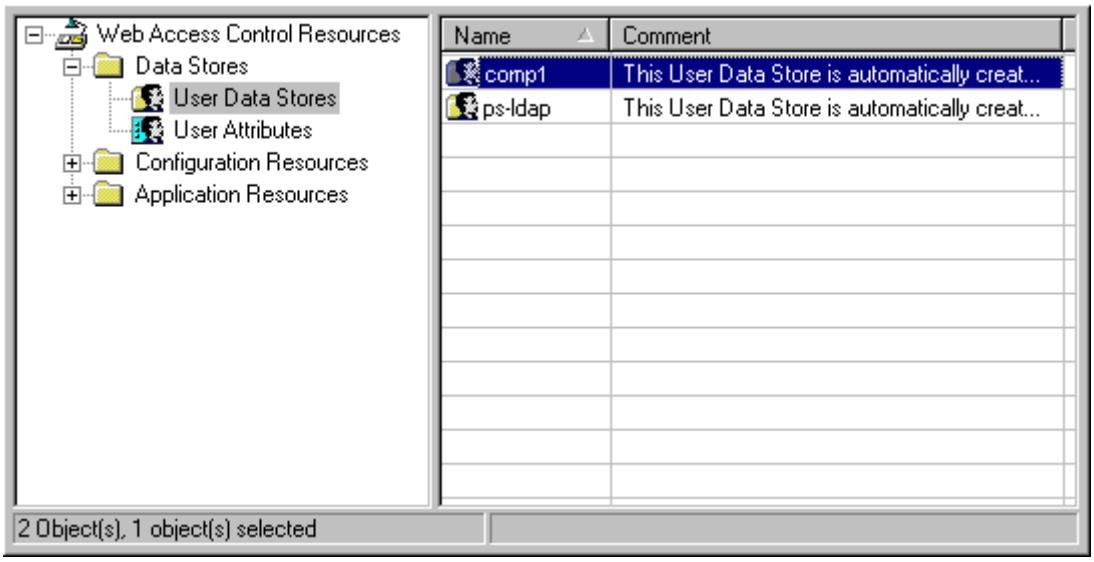
To simplify the task of assigning and removing authorizations, you should create groups of users who share the same functions or responsibilities. It is much easier to create a group, add users to the group, and then enable the group to use a particular host for initial authorization or to invoke an 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 or using selang on a command line.

Defining and Maintaining User Data Stores

From the Data Stores folder in the Policy Manager you can define:

- User data stores
- User attributes

To display the list of user data stores that are already defined, open the Data Stores folder and click the User Data Stores object. The following window displays in the workspace with the defined user data stores listed in the Name column.



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

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. You can change LDAP query limit by following this procedure.

1. Go to the eTrust Directory configuration file:

   `$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`

Creating a User Data Store

To define a new user data store:

1. In the Policy Manager, navigate to the SSO Resources, Data Stores section.
2. Right-click on User Data Stores, and select New.
3. Enter a name for the new user data store.
4. In the Data Store Type list, select one of the following data store types:

   LDAP—eTrust Directory, or any other LDAP-enabled directory

   AD—Microsoft Active Directory

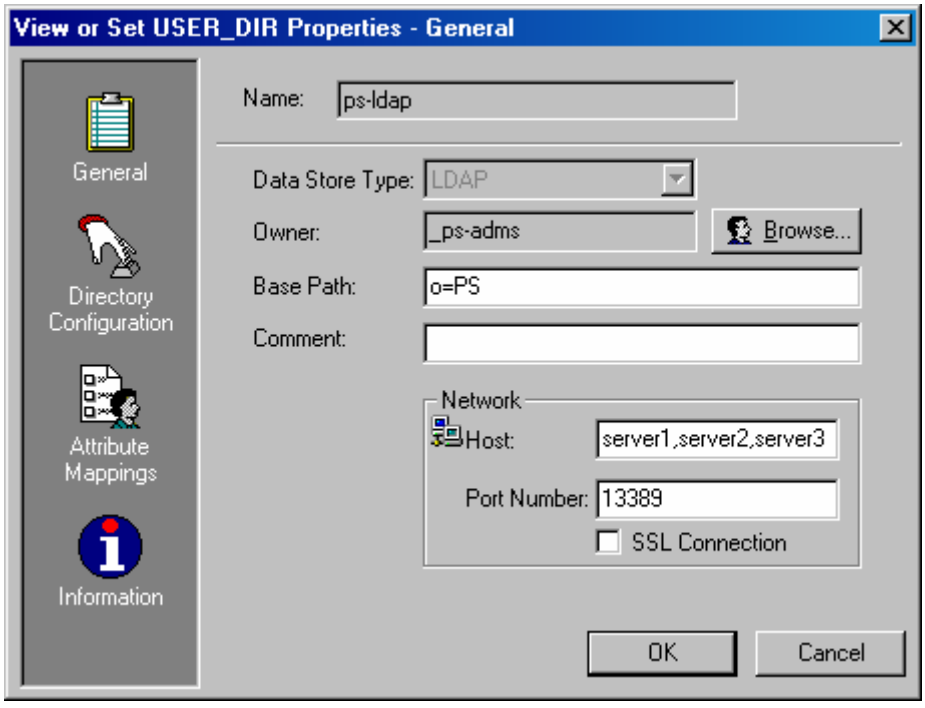
   eTrust_AC—eTrust Access Control

   TSS—eTrust CA Top Secret

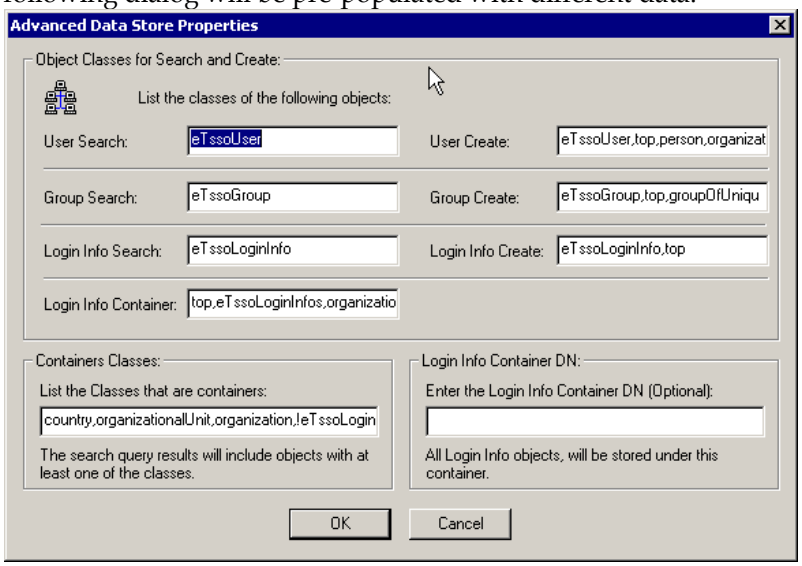
   ACF2—eTrust CA-ACF2

   RACF—OS/390 Security SErver
5. Fill out the other field on the General tab.

   Note: You can enter a comma separated list of hosts in the Host field to set up a server farm for the User Data Store as shown in the following example. If the Policy Server cannot contact the first host in the list, it tries to contact the next host in the list.



6. Select the Directory Configuration tab, then click the Advanced button to set the advanced options. Depending on the Data Store Type you selected, the following dialog will be pre-populated with different data:



7. Click OK to create the data store in the Policy Manager.

Populating the User Data Store

For eTrust SSO to function properly, the user data store must contain the required information on users and user groups.

Note: By default the Policy Server only recognizes English characters. If you are entering users with non-English characters you must set the Policy Server to recognize those characters. For more information about these settings, see the "Policy Server Settings" appendix in this guide and refer to the "DefaultLocate" setting.

Once the user data store contains the user and group definitions, eTrust SSO can begin to provide single sign-on functionality to the users.

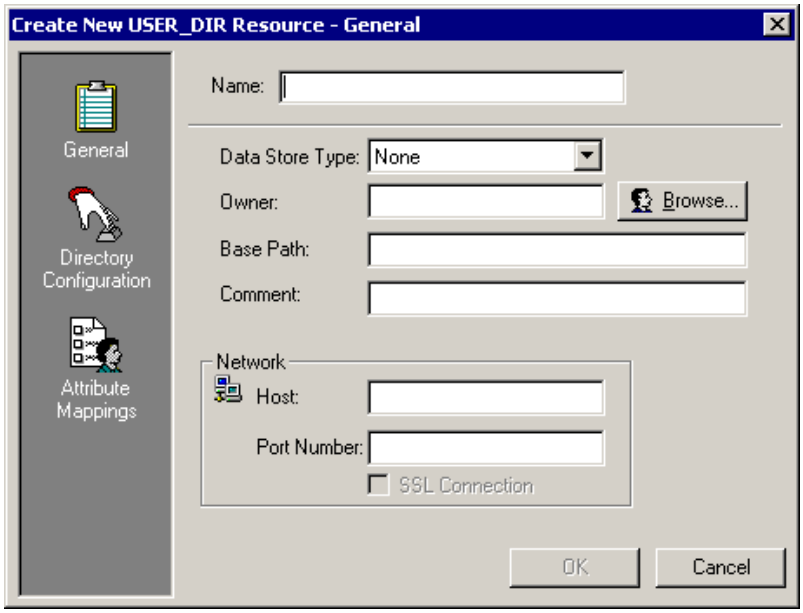
You can populate a user data store with the required information by using the following management tools:

Kind of Data Store	Tool for Populating the Data Store
eTrust Access Control	■ The Policy Manager ■ selang commands selang syntax is covered in the appendix "Using Selang".
eTrust Directory	■ The Policy Manager ■ JXplorer ■ IA Manager
Microsoft Active Directory	■ Microsoft Management Console
Other Third-Party Directories	■ The Policy Manager ■ Native management tools provided with the directory

Note: If you are using the eTrust Access Control data store as a user data store, you can populate this user data store with the users and groups from your current system data store by using the **ntimport** utility in Windows or the **UxImport** utility on UNIX.

Defining and Updating 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 Policy Server.

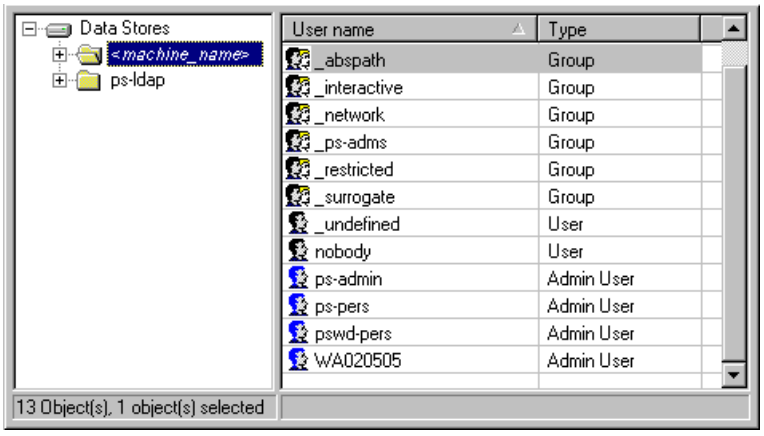
To change the properties of an existing user data store, you must first display the list of user data stores. Locate the one you want to change and double-click its entry in the list. This displays the View or Set USER_DIR Properties dialog. Make any changes that are necessary and click OK when you are finished.

Defining and Maintaining Users

The user data store should include everyone who uses the defined resources. This includes system administrators as well as end users.

You can define and maintain all of the users and user groups in the user data store from the Users application window.

To display this window, click the Users icon in the program bar.



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

The following sections explain how to define users, change the properties assigned to them, and remove them from the user data store.

To add applications to a newly created user, those applications must already exist. This means that you should create application entries in the Resources section before you create users.

Creating a User

To create a new user, follow these steps:

1. Click the Users icon.

   A list of data stores appears.

   - The LDAP (eTrust Directory) data store is listed as "ps-ldap". We recommend that you use the LDAP data store for user data.
   - The eTrust Access Control data store is listed as the name of the machine that the Policy Server is installed on.

2. Select the data store in which you want to create the new user.

   A list of any existing users in that data store appears.

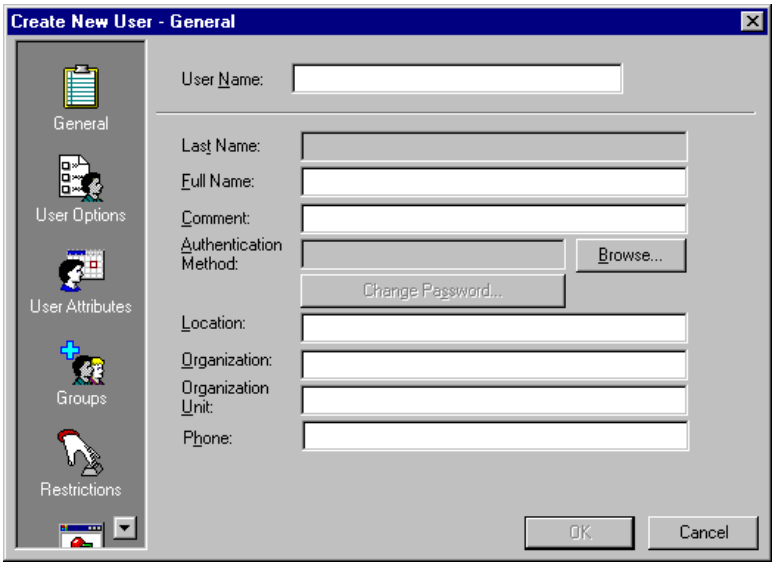
3. From the Edit menu, select New, User. You can also right-click and use the pop-up menu.

   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 dialogs:



In the left pane you can select icons that link to other information that can be defined for the user. These options are:

- **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 and indicates whether this user is an eTrust SSO administrator.
- **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 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 *eTrust Policy Manager Help*.

Changing a User's Properties

To change the properties of a user:

1. Display the list of users.
2. Double-click on the entry whose properties you want to change.

   This displays the View or Set User Properties dialog.
3. Make any changes, then click OK when you are finished.

The View or Set User Properties dialog is the same as the Create New User dialog, with the following additional dialogs:

Applications—Specifies which applications the user can access.

Application Groups—Specifies which groups of applications the user can access.

Application List—Lists the applications that this user has access to and sees on a personalized application list. From this dialog you can specify instructions for how this user logs on to an application. This list is generated when you click the Application List icon. This means that it might show more applications than the Application dialog.

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

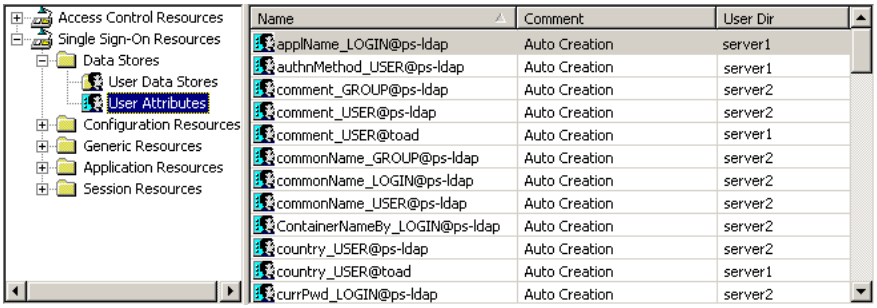
Creating User Attributes

Attributes are objects in the Policy Server data store that map between users, groups, logon information objects, class properties, and their equivalents on the external data store. Attributes hold user, group, and logon information in the user data store. User attributes can be used to grant a user access to a particular resource.

To work with the user attributes:

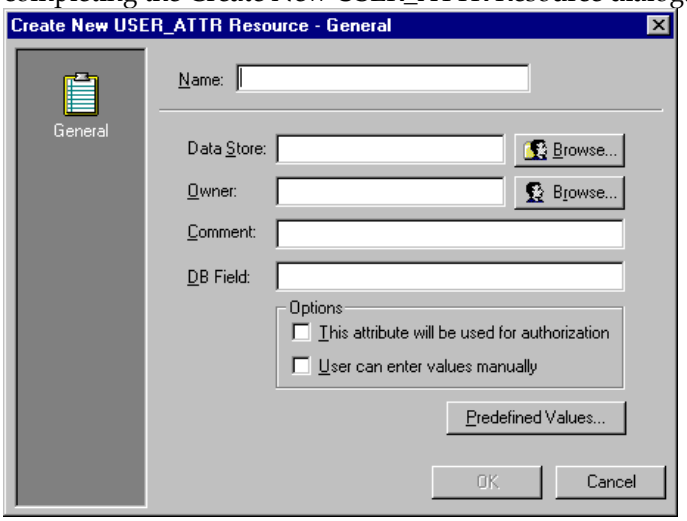
1. In the Policy Manager, select the Resources icon in the program bar on the left.
2. In the left pane of the application window, select Data Stores, User Attributes.

   The list of user attributes appears in the workspace:



3. Use the Edit menu or the pop-up menu to do any of the following:
   - Add and remove user attributes
   - Find a user attribute in the list
   - Change the properties of a user attribute

When you define a new user attribute, you must specify its properties by completing the Create New USER_ATTR Resource dialog:



Deleting a User

Follow this procedure to delete a user from the data store using the Policy Manager.

1. Click the Users icon.

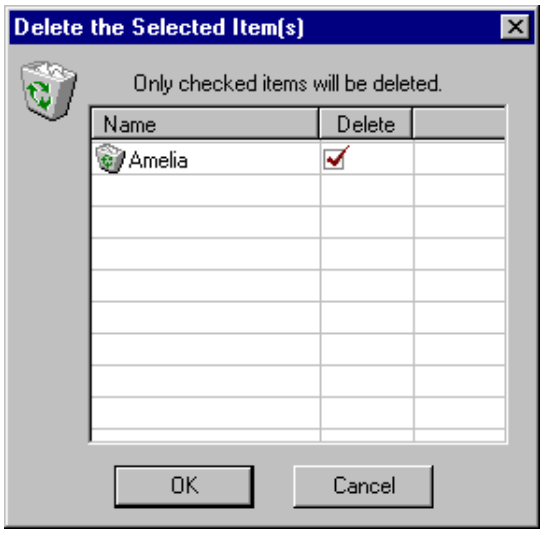
   A list of data stores appears.

2. Select the data store from which you want to delete the user.

   A list of any existing users in that 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.

Defining and Maintaining User Groups

The following sections explain how to create and maintain groups of users. For a discussion of how to assign access rights to resources using groups, see Using Groups to Assign Access Permissions in the chapter "Managing Resources with Policy Manager".

A group of users 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 eTrust 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 should 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.

Note: The Policy Server considers user groups to be accessors—just like users.

Creating User Groups

Follow this procedure to create a user group in the data store using the Policy Manager.

1. Click the Users icon.

   A list of data stores appears.

2. Select the data store in which you want to create the new user group.

   A list of any existing users in that data store appears.

3. From the Edit menu, choose New, Group.

   The Create New Group - General dialog appears.

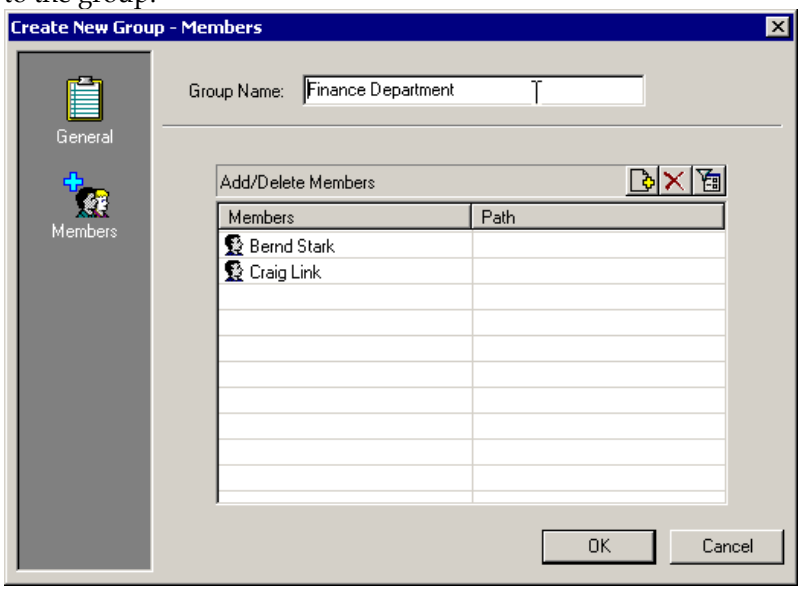
4. Enter the group name in the Group Name field.

   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.



7. Once you had 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 slightly different icon from individual users. 

Tip: 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".

Adding and Removing Group Members

There are three ways you can change the membership of a group:

- **Add members while creating a group**—Using the Members tab of the Create New Group dialog
- **Add or delete members from an existing group**—Using the Members tab of the View or Set Group Properties dialog
- **Add or remove group name from a user's record**—Using the Groups tab of the View or Set User Properties dialog

  When defining users, you can only add members to an existing group; if the group does not exist, you must create the group before you can add members to it.

Changing a User Group's Properties

To change the properties of a 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 Group Properties dialog.
3. Make any changes that are necessary and click OK when you are finished.

Notice that these additional dialogs are available when you edit a group's properties:

Applications dialog—Specify which applications this user group can access.

Application Groups dialog—Specify which application groups this user group can access.

Deleting a User Group

To delete a user group from the Policy Manager follow these steps.

1. Click the Users icon.

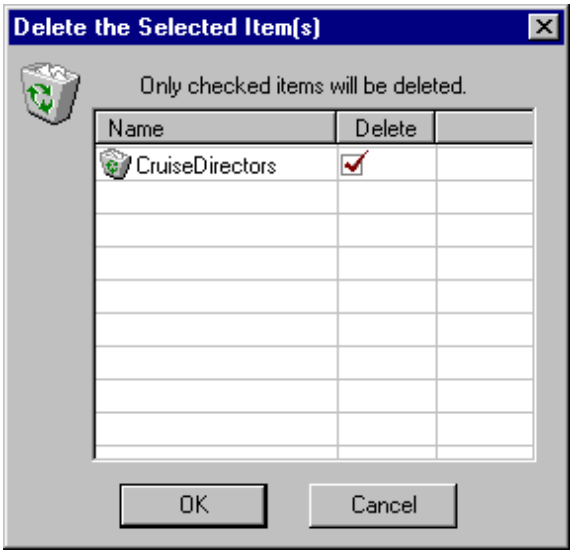
   A list of data stores appears.

2. Select the data store from which you want to delete the user group.

   A list of any existing users in that data store appears.

3. Right-click on the user group you want to remove, and select Delete from the pop-up menu.

   The Delete the Selected Items dialog appears.



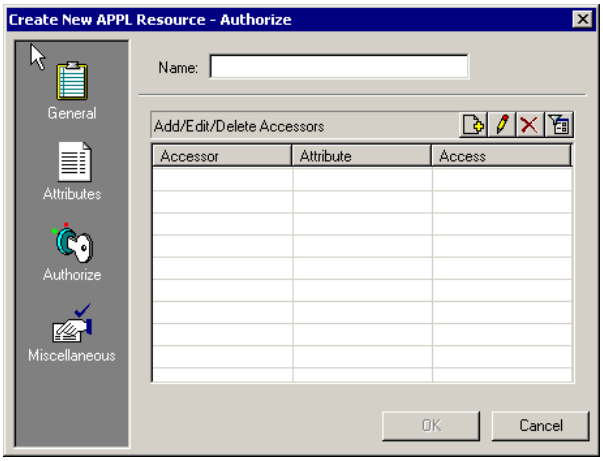
4. Be sure a checkmark appears in the check box in the Delete column beside the user, and then click OK to delete the user group.

Authorizing Users and Groups to Hosts and Applications

Users and groups must be authorized to log on to specific authentication hosts or groups of hosts. If users are to access applications through eTrust SSO, they must also be linked to the applications and groups of applications.

Users are linked to applications and application groups in the same way that applications are linked to application groups.

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.



The following selang commands authorize users and groups: **authorize** and **allow APPL**.

If you give an application **All** access, all users will be able to use that application.

Setting Up Administrators

Routine administration of eTrust 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

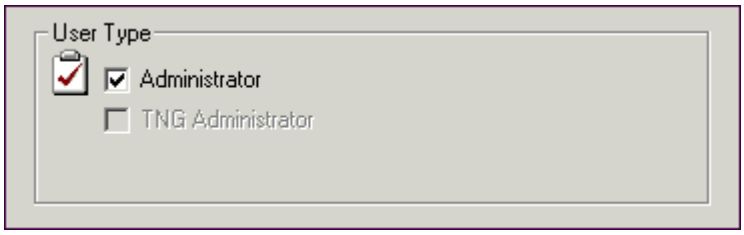
When you installed the Policy Server eTrust SSO, you defined two administrators. The default administrator names for:

- eTrust Access Control is ps-admin
- eTrust 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.

Defining 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 eTrust Access Control data store, but normal users should be created in the eTrust 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 follow these steps.

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 should now grant the new administrator access rights for the computer on which they will use the Policy Manager to administer the Policy Server. For more information about granting computer access rights, see Setting the Administrator Computer Access Rights.

Setting the Administrator Computer Access Rights

After you have created an eTrust SSO administrator, you should grant the new administrator access rights for the machine on which they will use the Policy Manager to administer the Policy Server.

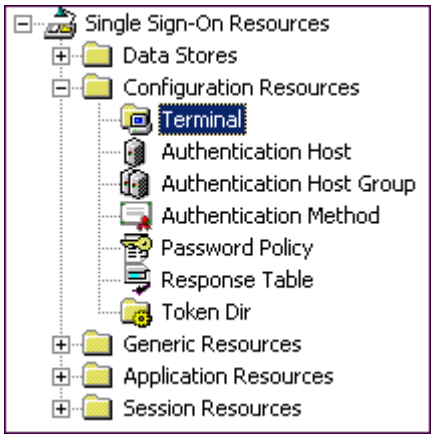
1. Select the Resources icon from the left hand pane.

The Single Sign-On Resources folders appear.

2. Expand the Configuration Resources folder, if it is not already expanded.

   A list of the configuration resources is displayed.

3. Select the Terminal configuration resource.



A list of computer names appears in the main window.

These computers can all be configured to give the administrators access to the Policy Server, using the Policy Manager. You can configure each computer to either give all administrators who log on to that computer access to the Policy Server, or you can define specific administrators to have access to the Policy Server, and block all other administrators.

4. To add a computer to this list, right-click and select New.

   The Create New Terminal Resource – General dialog appears.

5. Enter the full name of the computer and click OK.

   The new computer or "terminal" has been added to the list.

6. To set access rights to the Policy Server, right-click the computer name that you want to configure. This computer must have the Policy Manager software installed.

   The View or Set Terminal Properties – General dialog appears.

To give *all* administrators who log onto the computer access to the Policy Server:

a. Click the Set Default Access button.

   The Set Default Access dialog appears.

b. Click the All button then click OK.

To give specific administrators who log onto the computer access to the Policy Server:

a. Select the Authorize icon in the left pane.

   The View or Set Terminal Properties – Authorize dialog appears.

b. Use the Add, Edit and Delete buttons  to select administrators that you want to have access rights to the Policy Server from this computer.

Deleting an Administrator

eTrust SSO requires at least one administrator-level user.

This means that you cannot delete the last user who is defined as an Administrator.

You also cannot clear the Administrator check box on the User Attribute dialog for the last user.

Security Administrative Privileges

In order to carry out their duties, eTrust 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 *eTrust 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**—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 eTrust Access Control attribute. For information, see the *eTrust Access Control* documentation.

Ownership

Every record in the data store must have an owner. When a record is added to the data store, either its owner is explicitly assigned (by using the owner parameter), or ownership is assigned to the user who defines the record.

An owner can be a user or a group of users. If a user, or a group that owns a record is removed from the data store, the record no longer has an owner.

If neither a user nor a group is to be granted ownership of a record, the owner **nobody** must be assigned to the record. The user **nobody** is automatically included in the data store as a user without privileges.

The owner of a record has the following access privileges on the owned record:

Access	Description	Selang Commands
Read	Show the properties of the record.	showuser, showgrp, showres, showfile
Modify	Change the properties of the record.	chusr, chgrp, chres, chfile.
Delete	Remove the record from the data store.	rmusr, rmgrp, rmres, rmfile
Connect	Join a user to a group or disjoin a user from a group.	join, join-

The owner of a record is limited in the following ways:

- The owner of the only ADMIN record cannot delete it.
- Owners who do not have the AUDITOR attribute cannot update the audit mode. Only a user with the AUDITOR attribute can update the audit mode.
- Owners that do not have the ADMIN attribute cannot set the global authorization attributes—ADMIN, AUDITOR, OPERATOR, and PWMANAGER—for the users they own.
- Owners cannot make resources inaccessible to themselves.

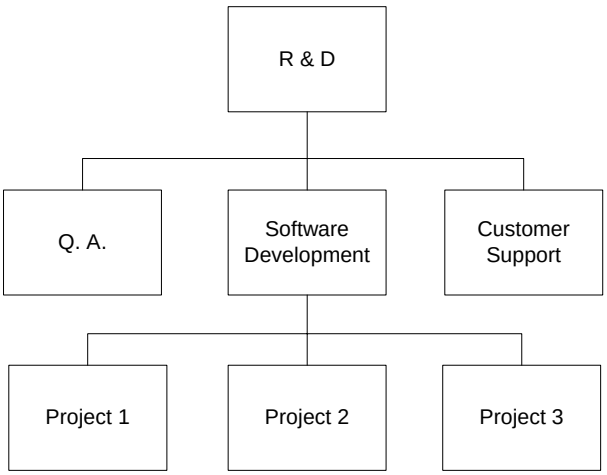
Group Authorization

Group authorization attributes use the parentage model described in this section.

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.

Group Authorization Attributes

A user or a group owns every record in the data store. The owner of a record has authorization to view, edit, and remove it. When a group owns a record, only certain privileged users within the group can manage the owned record. These users can also manage all the records owned by any groups that are subordinate to the group that they belong.

These privileged users have a *group authorization attribute* set in their own user records. The group authorization attributes are GROUP-ADMIN and GROUP-PWMANAGER

These attributes are set by the join command.

Note: A user becomes a member of a group when an authorized user executes the join command. Users with a group authorization attribute cannot manage members of a group unless the group also owns the members.

Group Scope

Users with a group authorization attribute can manage a limited set of records, which includes all records owned by the group in which the user has a group authorization attribute, and records owned by the subgroups of the group. This limited set of records is called the *group scope*.

Users with the GROUP-ADMIN attribute have the following access authorities to the records within their group:

Access	Description	Commands
Read	Show the properties of the record.	showuser, showgrp, showres, showfile
Modify	Change the properties of the record.	chusr, chgrp, chres, chfile
Delete	Remove the record from the data store.	rmusr, rmgrp, rmres, rmfile
Connect	Join a user to a group or disjoin a user from a group.	join, join-

Users with the GROUP-ADMIN attribute have the following restrictions:

- GROUP-ADMIN users cannot make resources inaccessible to themselves.
- GROUP-ADMIN users cannot delete the only ADMIN user record in the data store.
- GROUP-ADMIN users cannot remove the ADMIN attribute from the record of the only ADMIN user in the data store.
- GROUP-ADMIN users who are not ADMIN users cannot set the global authorization attributes—ADMIN, AUDITOR, OPERATOR, PWMANAGER, and SERVER—for any user.
- Without the AUDITOR attribute, even users with the GROUP-ADMIN attribute cannot update the audit mode.

Granting Security Administration Privileges

Staff can be assigned as eTrust SSO administrators and SSO password managers with the Policy Manager.

Select the required authorization with the Administration page of the User details dialog.

Users can be joined to GROUP-ADMIN or GROUP-PWMANAGER with one of the following selang commands:

```
join userName group (groupName) admin
```

or:

```
join userName group (groupName) pwmanager
```

Chapter

6

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:

Data Stores
Defines user data stores and user attributes.

Configuration Resources
Defines authentication hosts, authentication host groups, authentication methods, terminals, responses, password policies, token directories, and Policy Server settings.

Generic Resources
Defines generic resources for custom classes.

Application Resources
Defines applications and application groups.

Session Resources
Defines Policies for managing user sessions.

Populating 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. For an explanation of how to use selang and a list of selang commands and syntax, see the selang Command Reference Guide.

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 Data Stores

The Data Stores resource folder includes the user data stores and user attributes. For information about the user data stores and user attributes, see the chapter "Managing Users and User Groups " in this guide.

Managing Configuration Resources

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

- Terminals
- Authentication hosts
- Authentication host groups
- Authentication methods
- Password policies
- Response tables
- Token directories
- Policy Server settings

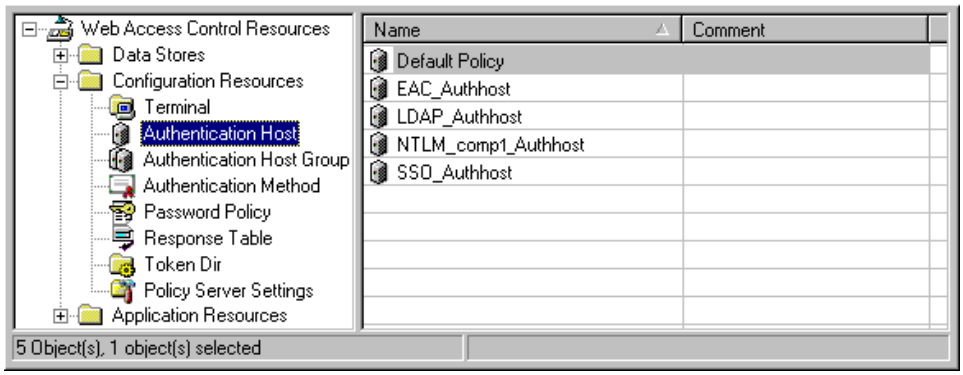
Defining Authentication Hosts

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, expand the Configuration Resources folder and 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 Policy 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 should not delete these as you will not be able to access the system without them.*

To Define an Authentication Host

To define an authentication host, follow these steps:

1. Define an authentication method. For more information, see Defining Authentication Methods in this chapter.
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 NT 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 *eTrust Policy Manager Help*. The online help also contains a complete explanation of all the dialogs and the fields they contain.

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.
NT	Specify the domain name or the name of the stand-alone Windows server that the NT 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 Policy Server does not accept SSO tickets generated with default or empty encryption keys by default. To change this behavior, change the AllowDefaultEncKey General Policy Server setting to Allowed. For more information, see the appendix "Configuring the Policy Server".

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 Policy Server maps the authenticated user to a user that is defined in the user data store.

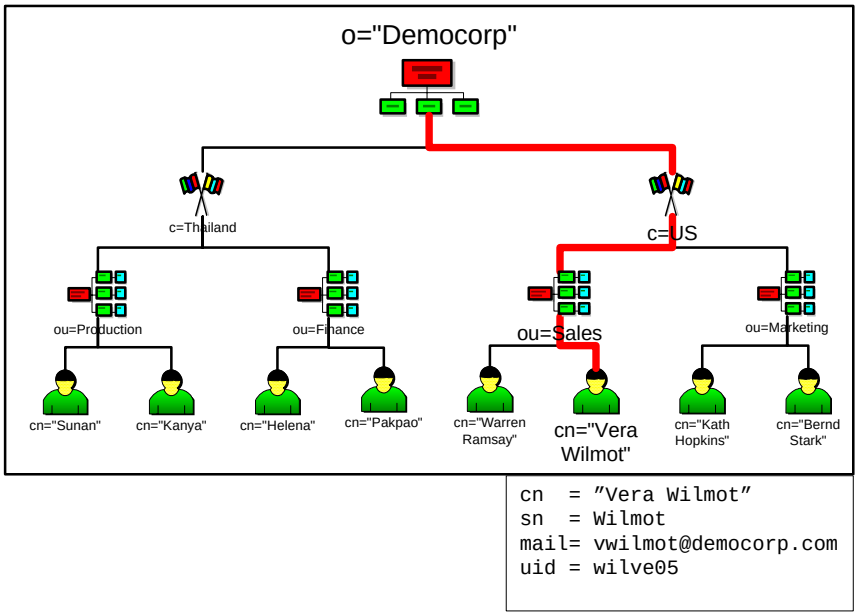
  This method of user mapping lets you use a different user data store to check the authorization level of the user. For example, the Policy Server checks if the user is permitted to use the selected authentication method.

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

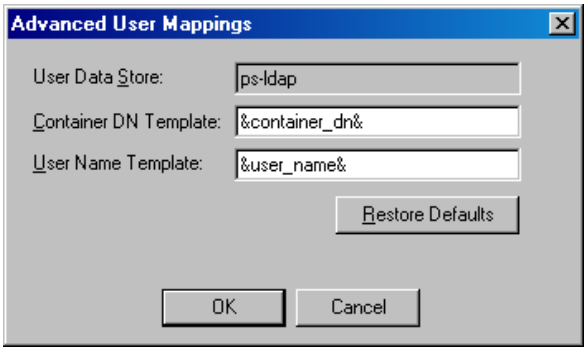
`cn="Vera Wilmot",ou=Sales,c-US,o="Company A"`

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.

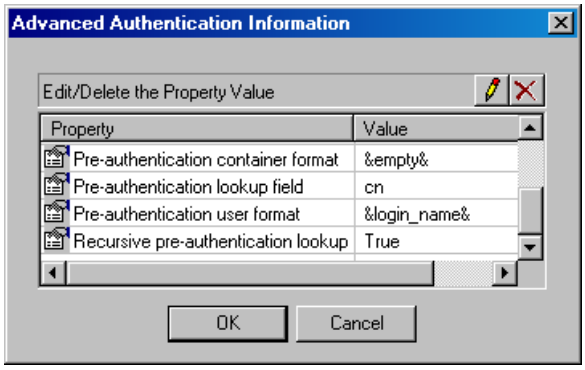
To configure post-authentication user mapping, follow these steps:

1. 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, then click the Advanced Authentication Information button.



2. 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 Policy Server.
&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.

Case Study – Post-Authentication User Mapping

Barry, the administrator for Company1, sets up a TSS directory for authentication and an eTrust Directory user data store. To let users (for example, Sherry) authenticate with the TSS directory and be mapped to the eTrust Directory user data store, Barry creates a user directory pointing to the TSS server (tss-dir) and a user directory pointing to the eTrust 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 eTrust 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. Database lookup is only supported by LDAP authentication providers.

Case Study – Pre-Authentication Lookup

Rita, the administrator of Company2, sets up an Active Directory for user authentication and an eTrust 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 eTrust 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 eTrust 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 name, 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&

Dialog	Field	Value
	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 will be mapped to "cn=Daniel, ou=comp, dc=domain1, dc=company2, dc=com" in the Active Directory.

Case Study – Post-Authentication Lookup

John, the administrator of Company3, sets up an Active Directory for user authentication and an eTrust 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 eTrust Directory data store (etr-dir). The existing Active Directory has users defined using their email address (for example, Amelia@Company3.com).

John sets the eTrust 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 eTrust 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 eTrust Directory user data store.

Backward Compatibility Information

eTrust SSO no longer requires users to know the authentication host parameters in order to authenticate. Instead, eTrust 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, eTrust 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 NT authentication method that is verified against a domain called Domain1, select NT 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 Policy 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

Defining 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.

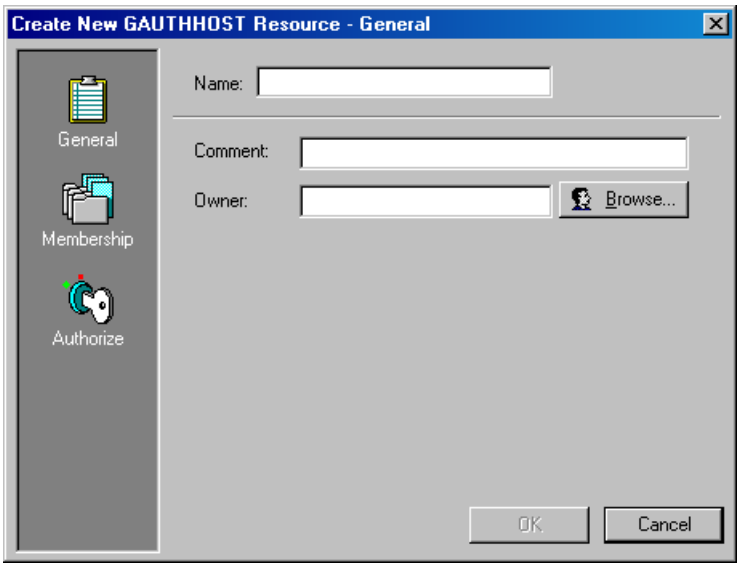
From the list of groups you can do the following tasks:

- 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.

Defining 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 *eTrust Policy Manager Help*.

Updating the Properties of a Groups of Authentication Hosts

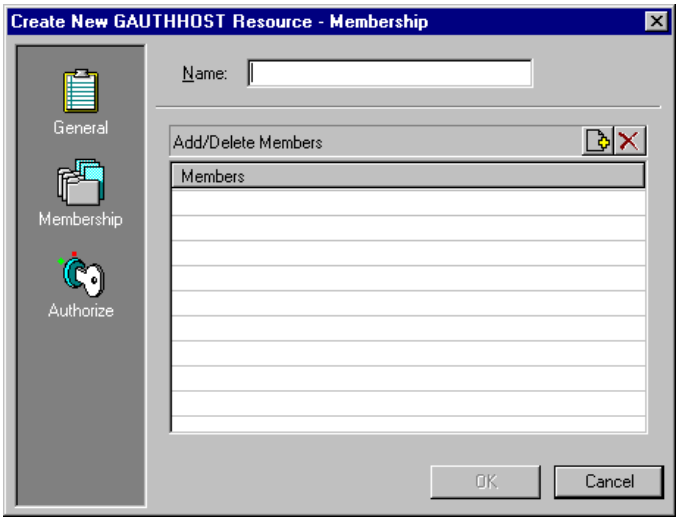
To change the properties of an existing authentication host group:

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.

Adding Members to a Groups 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.

   If this is an existing group, click OK to update the list of members.

Removing Members from a Group of Authentication Hosts

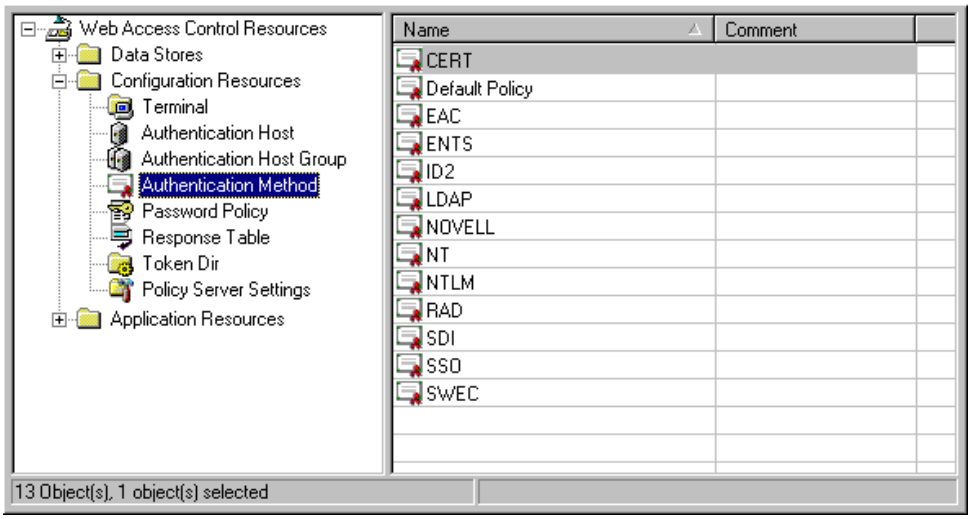
To remove members from the group:

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.

Defining Authentication Methods

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, open the Configuration Resources folder and click the Authentication Methods object. The following window displays in the workspace with the defined authentication methods listed in the Name column.

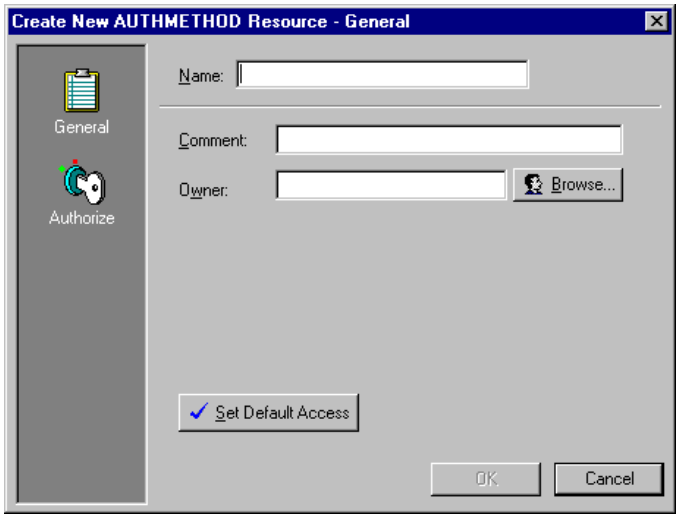


From the list of authentication methods you can locate an authentication method in the list or change the properties of an authentication method.

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.

Defining 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 *eTrust Policy Manager Help*.

Updating Authentication Method Properties

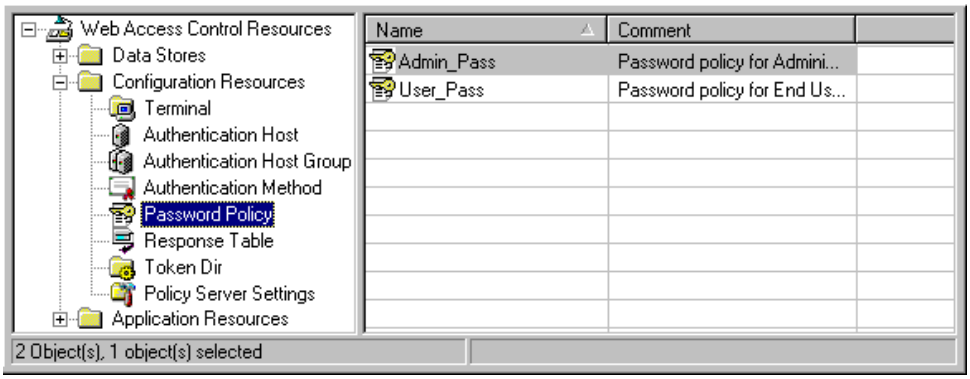
To change the properties of an existing authentication method:

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.

Defining Password Policies

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, open the Configuration Resources folder and click the Password Policy object. The following window displays in the workspace with the defined password policies listed in the Name column.



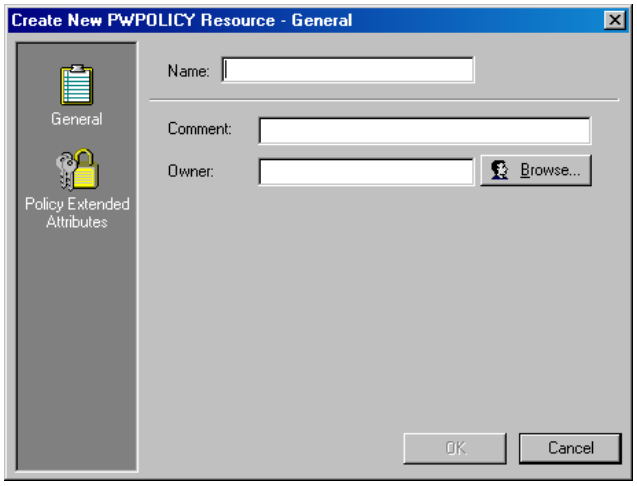
From the list of password policies you can do the following tasks:

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

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

Defining 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 *eTrust Policy Manager Help*.

Defining a Generic Password Policy

Generic password policies apply to all applications. For more information about generic and specific password policies, see the "Managing Passwords" chapter in this guide.

The following steps describe how 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.

Defining an Application-Specific Password Policy

Application-specific password policies are applied to particular applications. For more information about generic and specific password policies, see the "Managing Passwords" chapter in this guide.

To define a password policy that will apply to specific applications, follow these steps:

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.

Updating Password Policy Properties

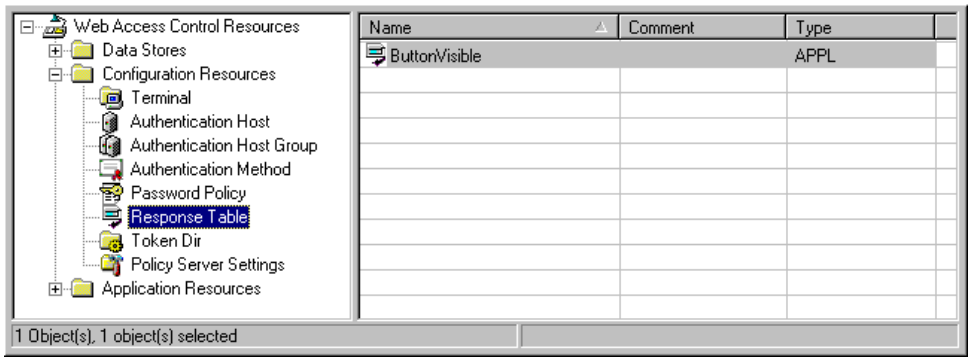
To change the properties of an existing password policy, follow these steps:

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.

Defining Response Tables

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, open the Configuration Resources folder and 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. For further information about defining response tables, see eTrust Web Access Control documentation, *Administrator Guide* chapter "Using the Web Agent".

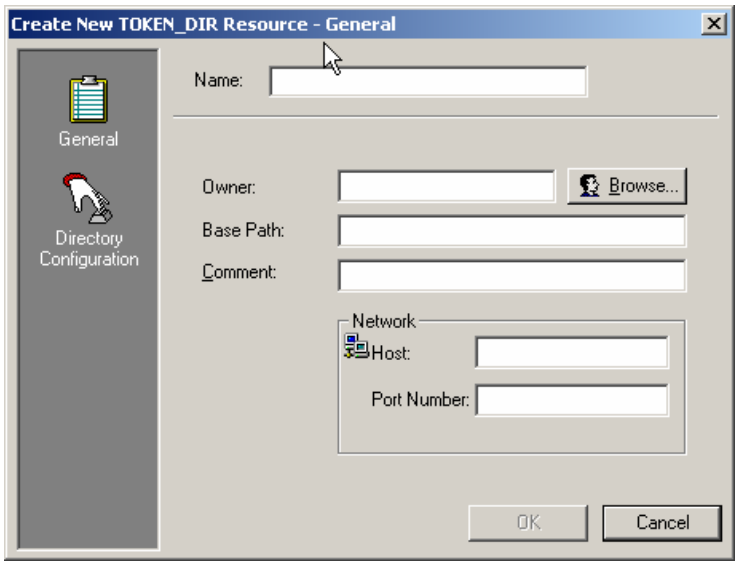
Defining a Token Directory

A *token directory* is an LDAP directory in which the Policy 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 Policy Server uses the token directory to save token and user information. If a Policy 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 Policy Servers. If the Policy 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 Policy Servers.

Defining 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 *eTrust Policy Manager Help*.

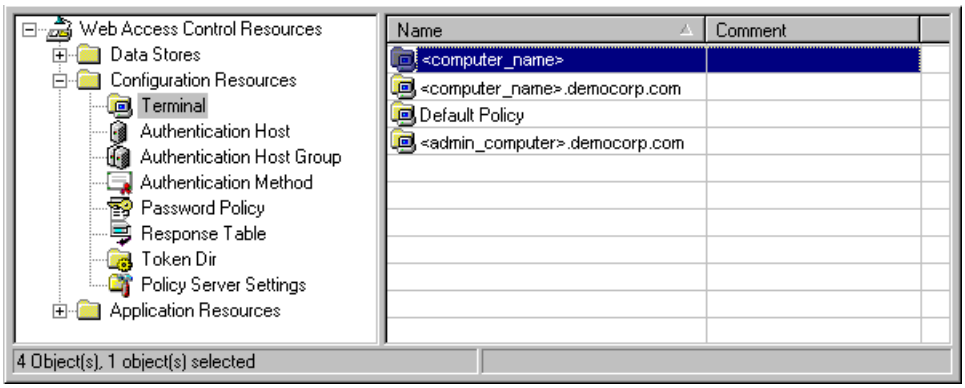
Updating Token Directory Properties

To change the properties of an existing token directory locate the token directory you want to change and double-click its entry in the list. This displays the properties dialog. Make any changes that are necessary and click OK when you are finished.

Defining Terminals

Administrators can perform administrative tasks from a workstation computer only if they have read and write permission on the Policy Server to their computer. Terminal objects are used to define which administrators can use a particular computer to manage the Policy Server (using the Policy Manager or selang). For example, in order to let administrator Gavin manage users and resources from his computer (Gavin01), you need to create a terminal object named Gavin01 and give Gavin read and write permissions.

To display a list of defined terminals, expand the Configuration Resources folder and click the Terminal folder. The following window displays in the workspace with the defined terminals listed in the Name column:



Note: The computer that was used to install the Policy Server is automatically added to this list.

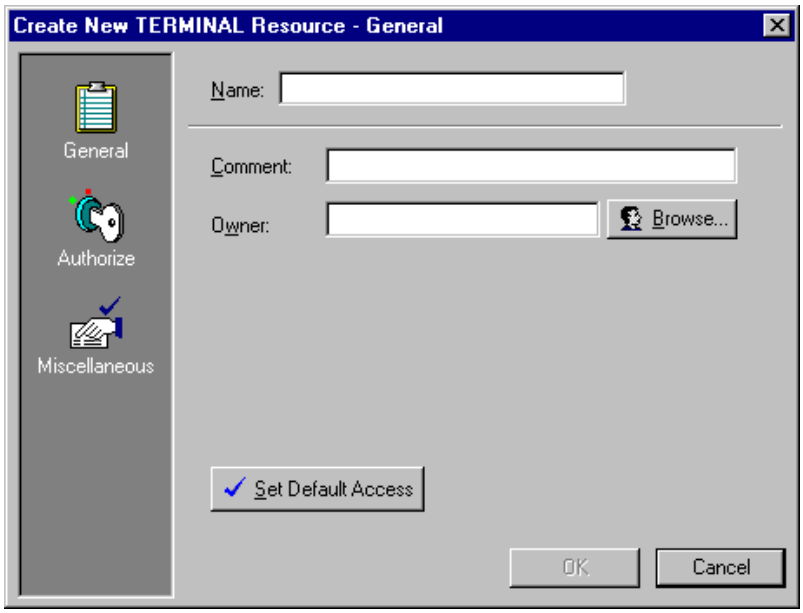
From the list of terminals you can do the following tasks:

- Add or remove a terminal
- Locate a terminal in the list
- Change the properties of a terminal.

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

Defining Terminal Properties

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



The dialogs used to define the properties of a terminal are:

- **General**—Defines the terminal's name and owner.
- **Authorize**—Lists who can use this terminal and what permissions they have.

  For more information see Assigning Access Permissions in this chapter.
- **Miscellaneous**—Sets day and time restrictions for this resource.

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

Updating Terminal Properties

To change the properties of an existing terminal resource:

1. Display the list of terminals.
2. Locate the terminal you want to change and double-click its entry in the list.

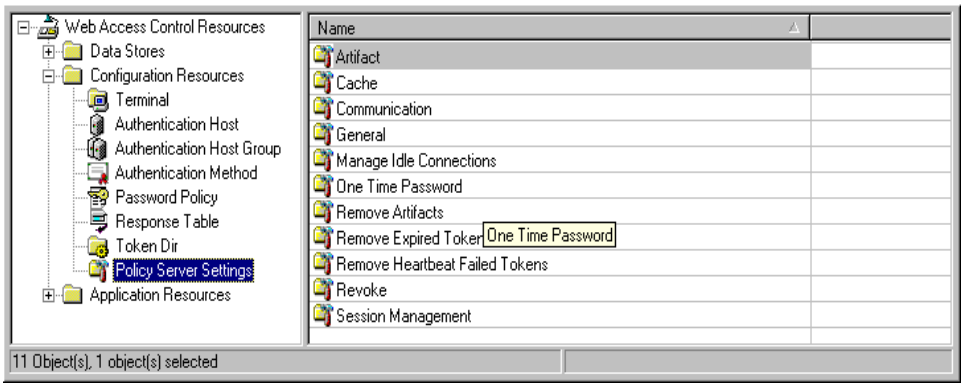
   The View or Set TERMINAL Properties dialog appears.
3. Make any changes that are necessary and click OK when you are finished.

Tip: Use the Information dialog to check when the last changes were made and who made them.

Defining Policy Server Settings

Policy Server Settings control the various aspects of the Policy Server configuration. For information about the settings you can modify and how they affect the Policy Server, see the appendix "Configuring the Policy Server".

To display the list of the Policy Server settings, expand the Configuration Resources folder and click the Policy Server Settings folder. The following window displays in the workspace with the subfolders containing the Policy Server settings listed in the Name column:



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

Updating Policy Server Settings

To change existing Policy Server settings:

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

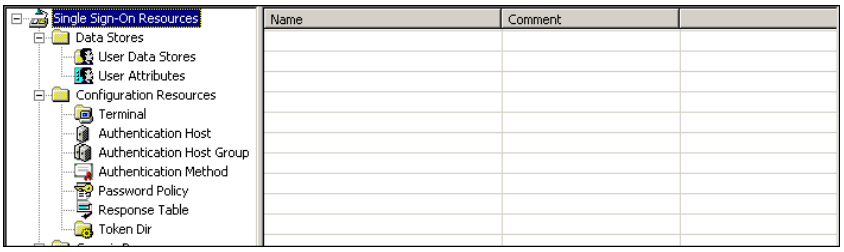
A *resource* is an entity that users and groups can access.

Resources are grouped by *class*, which is a name for the type of resource. For example, the TERMINAL class contains all terminals. For more information about the eTrust SSO classes for resources, see the chapter "The Policy Data Store".

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:

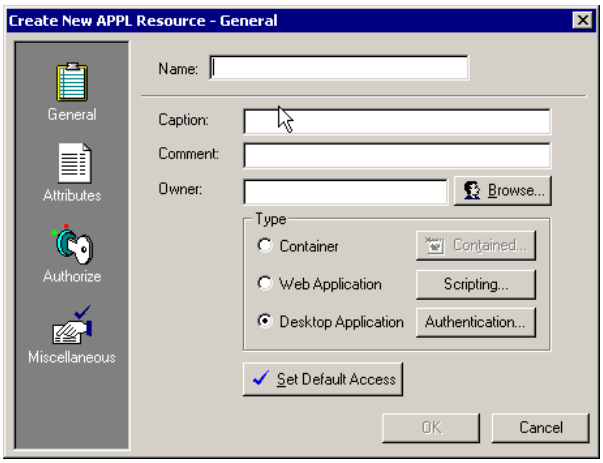
Resources Group	Resources
Data Stores	**User Data Stores**—Data store in which the Policy Server stores user information **User attributes**—Categories of extra information that can be recorded for each user, such as the country they work in, or their children's names.
Configuration Resources	**Terminal**—All machines that run the Policy Manager. **Authentication Host**—Machines 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 Policy Server stores the users' session information.
Generic Resources	**URLs**—Web Addresses of secure web pages that can be opened and logged on to using eTrust SSO. **EJBs**—Enterprise Java Beans (EJB) are not relevant to eTrust SSO. For more information about EJBs, see eTrust Web AC documentation.
Application Resources	Applications that users can log on to using eTrust 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 terminals and terminal groups from which users can log on
- 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.

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.

Managing Application Resources

Every application to be available through eTrust 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

For more information about application types, see The Application Class (APPL) in the "Working with the Policy Data Store" chapter.

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 on to 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. eTrust 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, eTrust 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 on to 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 Policy Server checks the normal expiration time of the eTrust SSO security token where the default expiration time is eight hours. For a sensitive application, the Policy 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

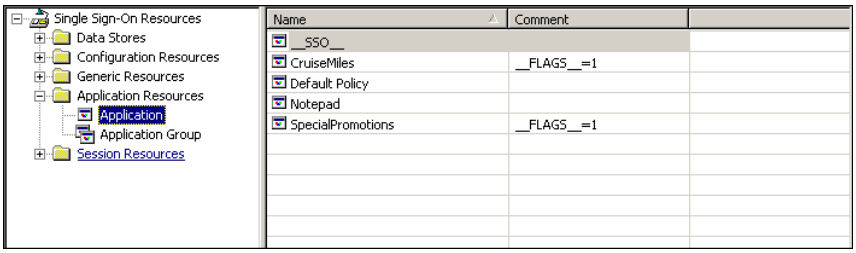
The password a user must enter to log on to a sensitive application is **the password used for primary authentication,** not the password of the specific application. For example, if primary authentication is set to NetWare and the Telnet application is marked as sensitive, the user is asked to provide the NetWare 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 by the SensitiveExpiration token in the ssod section of the ssod.ini file in the UNIX operating system or in the Policy Server registry key in the Windows operating system.

Defining Applications

Unlike other resources, applications have user name and password information and an HTML script associated with them. These associations allow eTrust 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.

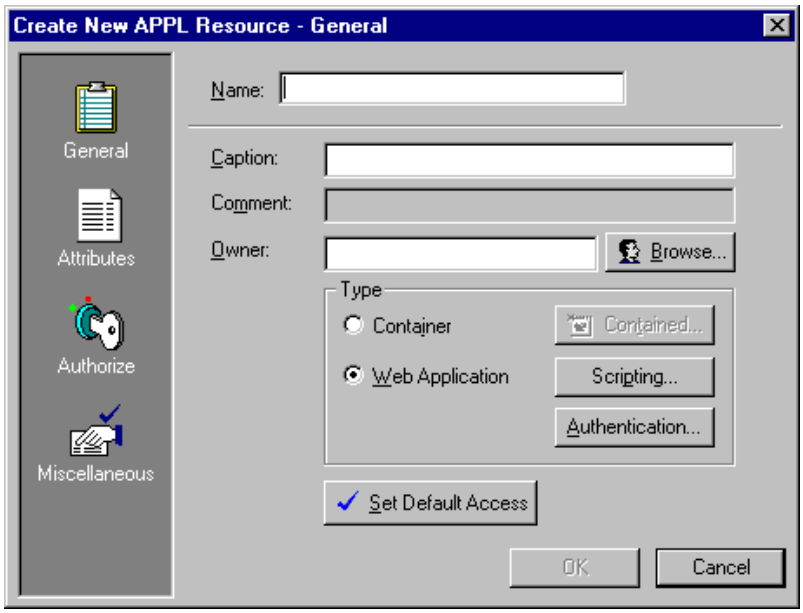
To display a list of defined applications, open the Application Resources folder and click the Application entry. The following window displays in the workspace with the defined applications listed in the Name column.



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.

Defining 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 example 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 or disabled application.

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

Miscellaneous—Sets day and time restrictions for using this application.

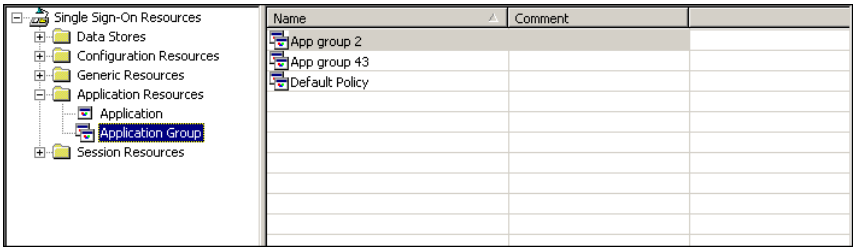
Updating an Application's Properties

To change the properties of an existing application, you must first display the list of applications. Locate the application you want to change and double-click its entry in the list. This displays the View or Set APPL Properties dialog. Make any changes that are necessary and click OK when you are finished.

Defining Application Groups

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).

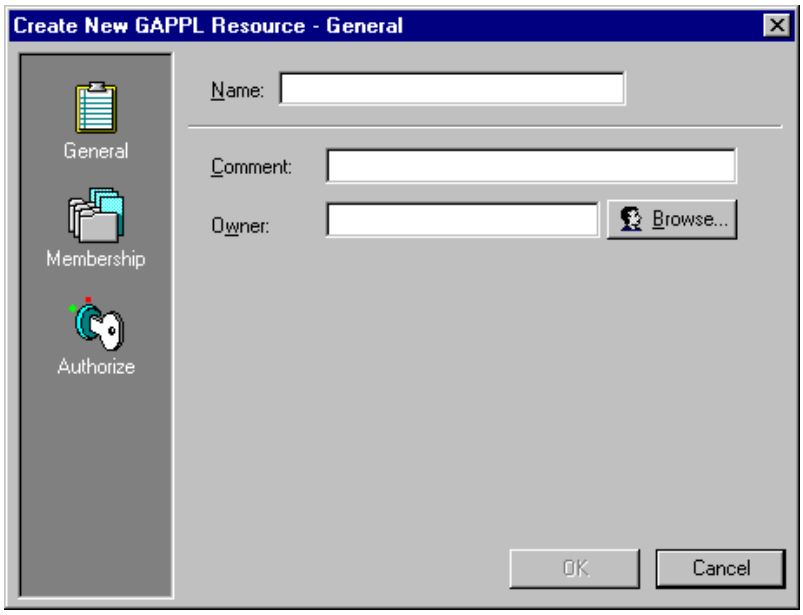
To display a list of defined application groups, open the Application Resources folder and click the Application Group entry. The following window displays in the workspace with the defined application groups listed in the Name column.



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.

Defining 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.

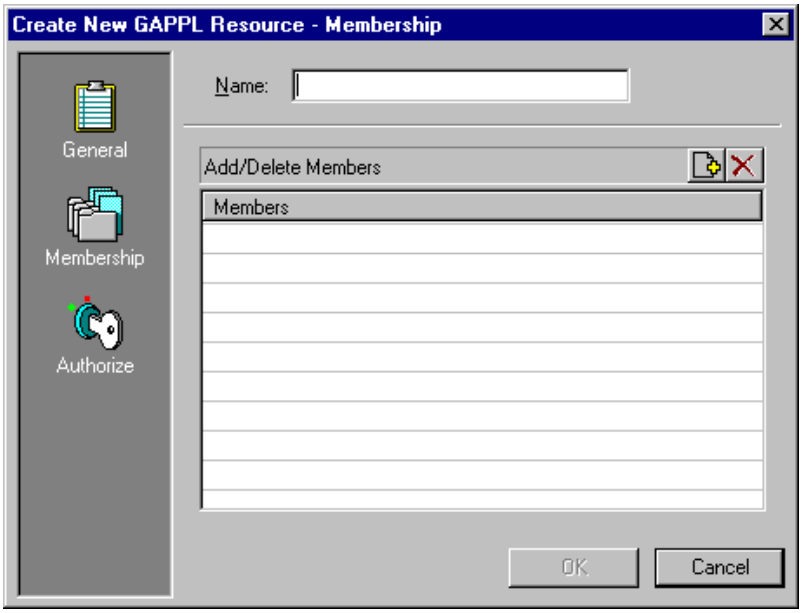
Updating an Application Group's Properties

To change the properties of an existing application group, you must first display the list of application groups. Locate the group you want to change and double-click its entry in the list. This displays the View or Set GAPPL Properties dialog. Make any changes that are necessary and click OK when you are finished.

Adding 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.



Click the Add icon in the Add/Delete Members section to display a list of defined applications. 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, you must first display the Membership dialog. Then highlight the group you want to remove and click the Delete icon in the Add/Delete Members section to remove the member.

Adding an Application Record

To add an application to 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. Select New from the main menu or from the entry list pop-up menu. If there is an application with similar properties to the new application, you can select Add Like from the pop-up menu.
3. In the New Application window that appears, enter all the parameters of the new application. Use the tab headings to page between the different sections. When you are sure that the information is correct and complete, push the Apply button.
4. Policy Manager adds the application to the data store and displays it in the application list. The New Application dialog remains open so that you can add another application, though the fields are blank.

Updating 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, push the Apply button. If you have not made any changes, the Add button remains dimmed.

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.

Linking and Unlinking Applications and Application Groups

To link or unlink an application group to one or more applications:

1. Display the entry list of application groups and select the application group that you want to update.
2. Press the right mouse button on the name of the application to display the pop-up menu. Select GAPPL_Applications from the right-button menu. The dialog that appears displays all the applications that the gappl has already been linked to; in this case, gappl Unit One is linked to application dictionary.
3. To link the application group to one or more applications, click on the application or applications you want to link the application group to in the right box—Not Linked—and press the arrow pointing left. To unlink an application group from an application, click on the application or applications in the left box—Linked—and press the arrow pointing right. Double-clicking on the name of the application will move the application to the opposite box.

   If you want to link all applications to the gappl at the same time, click the Link All button. You can unlink all the applications from the gappl in one operation by using the Unlink All button.
4. Select *Apply* to link the gappl to all the applications on the left and keep the dialog open. Select OK to link the gappl to all the applications on the left and close the dialog.

   The display in the dialog is updated. The dialog remains open and you can continue editing the assignment of gappls and applications.

Selang commands

The commands newappl and editappl add applications to the data store.

The commands newres GAPPL and editres GAPPL add an application group to the data store. When the parameter mem+ is used, the newres GAPPL command adds the new application to the application group.

Assigning 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 Policy 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: The TERMINAL class also has an Owner access rule type. Owner rules override access permissions defined in an ACL.

Setting 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:

- Terminals (TERMINAL)
- Authentication hosts (AUTHHOST)
- Authentication methods (AUTHMETHOD)
- Applications (APPL)
- Resources listed in the Generic Resources folder (URL)

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.

Setting 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. For information about creating an access control list for an individual resource, see Defining an Access Control List in this chapter.

You can also ease administration by creating groups and assigning access permissions by group. For information about using groups to assign access permissions, see Using Groups to Assign Access Permission in this chapter.

Defining an 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.

Defining 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:

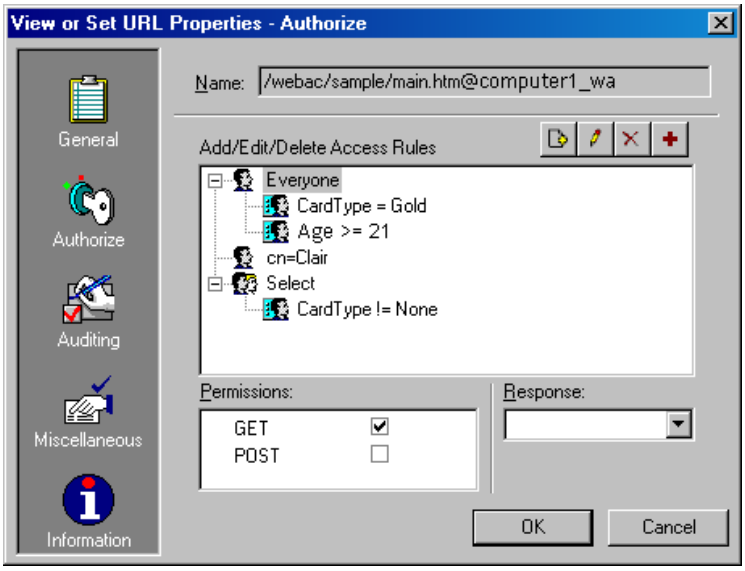
- a user or group name from a user data store, or everyone
- additional conditions that further narrow the accessor according to its attributes or the groups it belongs to.

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.

Defining 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 eTrust 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.

Defining Terminal Resource ACL

The access control list for terminal resources holds a list of accessors paired with access types. It lists the accessors authorized to access the terminal and the exact access they can have. The users listed in this ACL can only be from the eTrust Access Control database from which this terminal is defined.

The following table shows a sample access control list for the TERMINAL a12.

Accessor	Access
Payroll (GROUP)	R
Joe Stevens (USER)	R, W

From this table you can see that the Payroll group has read access to the TERMINAL a12, while the user Joe Stevens has read and write access.

Adding an Access Rule

To add an access rule, click the Add Rule icon in the Add/Edit/Delete Access Rules area and 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, highlight the entry you want to remove and click the Delete icon in the Add/Edit/Delete Access Rules area. The selected access rule will be removed from the list immediately.

Using Regular Expressions to Define Access Rules

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.

Using Groups to Assign Access Permission

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

7

Managing 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.

eTrust 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 password 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. eTrust 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 (e.g. a smartcard) or for their end applications.

This chapter discusses password protection policies and shows you how to set, change, and control passwords and their rules.

Password Management Tasks

Password management in eTrust SSO is controlled by the eTrust SSO administrator. eTrust SSO administrators perform these password actions:

- Set password policies for the whole eTrust 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

Password Management Tools

You can use the following tools to manage user passwords:

Software	Description
Policy Manager	Each time you create or update users defined in Windows, you can set or replace user passwords. For more information, see the chapter "Using the Administrator Interface." You also use Policy Manager to set password policies.
IA Manager	You can administer user passwords using the IA Manager which is a web-based administration tool. The IA Manager also provides some end user self-administration functions.

Managing User Passwords

This section describes some common tasks for managing passwords.

Specifying a User's Primary Authentication Password

Follow these steps to specify a user's primary authentication password:

1. Select the Users icon in the eTrust SSO program bar to display the Data Stores window.
2. Select the data stores folder to display the list of available users and either double-click the user in which you are interested or right-click the user and choose Properties. The View or Set User Properties - General dialog displays.
3. Select an authentication method if you have not already done so. For detailed instructions on selecting an authentication method, refer to the topic, Specifying a User's Authentication Method.
4. If the authentication method you choose supports passwords, such as SSO, the Change Password button will be active. Click Change Password and select an authentication method to display the Change Password dialog.
5. 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.
6. Click OK to return to the General dialog.
7. Click OK to finish.

Setting Up a Reminder for Users to Change Their Password

In the Policy Manager, you can set up a facility to remind users to change their passwords.

When the user logs on to the Policy Manager, a dialog appears reminding them to change their password. If the user clicks Yes, the Change Password dialog appears.

To set up the password change reminder, complete the following steps:

1. Click the Resources icon in the SSO program bar, and then choose Tools, Activates eTrust Classes from the menu.
2. On the Activate eTrust Classes dialog, check the Password box in the User Identity Control section.
3. Check the Change Own Password box in the Options section.
4. Click OK to close the dialog.

Changing the Primary Authentication Password

When the primary authentication method is eTrust SSO native authentication, the user password can be changed by either an administrator using the Policy Manager or the user using SSO tools.

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 eTrust SSO tools or the application to change their application password.
- **eTrust SSO**—When the current password for an application in eTrust SSO expires, eTrust SSO prompts the user for a new password. This password is updated in eTrust SSO automatically.
- **eTrust SSO administrators**— The administrator can change an application password using the Policy Manager if a user has forgotten their password.
- **The application**—The application may prompt for a password change, but this should be avoided. This is because eTrust SSO will not have been notified of this change, and there is a danger that the stored passwords could be out of sync.

How the Application Password is Stored

The logon info section of the user record in the data store contains two password fields for each application allowed to the user: 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 will use 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 is run, it accesses the values of CURRPWD and NEXTPWD, which the SSO Client has received from the Policy Server. Then, the script logs into the application using the current password CURRPWD, and checks if there is a value for NEXTPWD field. If there is, the script changes the password in the application from the current password to the next password (the value of NEXTPWD), and tells the Policy Server to set the value of CURRPWD to that of NEXTPWD and to null the value of NEXTPWD.

Logon scripts for password-based applications usually have a section that looks like this:

```
. . .
. . .
if     {$_NEXTPWD != "" } {
       # do password change
       . . .
       . . .
       }
```

For more information on the password fields in the user data store, see the chapter "The User Data Store" in this guide.

For more information on logon scripts, see the *Scripting: Guide and Reference*.

Password Change Initiated by the User

Users can change a password by using eTrust SSO tools or by changing the password directly in the application.

By using eTrust SSO tools, eTrust SSO will set NEXTPWD to the value of the password entered.

Changing the password directly in the application creates a potential problem, since now the value of CURRPWD does not match the password expected by the application.

The user can correct the situation either by using eTrust SSO tools to update the SSO profile with the new application password to set CURRPWD to the value of the new password, or by informing an eTrust SSO administrator, who then should use the Policy Manager to update the value of CURRPWD.

If the value of CURRPWD is not changed, the next 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 on to the application, and update the value of CURRPWD.

Password Change Initiated by eTrust SSO

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

If eTrust SSO's password auto-gen feature is enabled, the Policy 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 eTrust SSO system, administrators will only have to initiate application password change 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. Display the list of users in the data store and right-click the user whose application list you want to see.
2. Select Application List from the pop-up menu to get the application list window.
3, Right-click the name of the appropriate application and choose Update Login Info from the pop-up menu.
4. Select the Login Information tabbed page, type in the new password, and retype the new password in the Verify Password field.

   If the password has been changed directly by the user in the relevant application, select Update out-of-sync Password.

   If the password has not been changed directly by the user in the relevant application, do not select Update out-of-sync password.

Password Change Initiated by the Application

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

In order to avoid application-initiated password changes, eTrust SSO administrators should set the password interval in eTrust 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 is 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 eTrust SSO password policies).

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

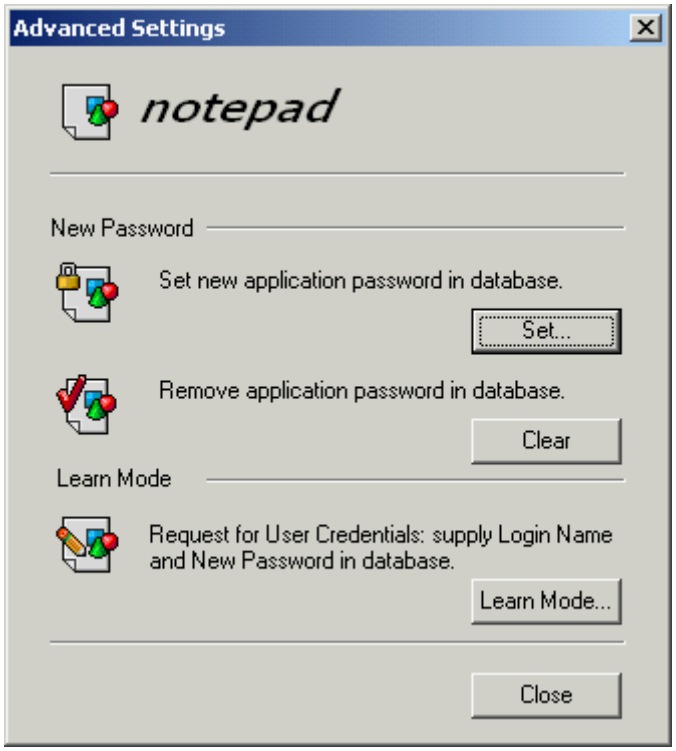
Letting Users to Update Their Own Credentials

The SSO Client can be set to let users to update two kinds of credential information:

- **Learn mode**—Allows users to enter a user name and password the first time they log on to an application through eTrust SSO
- **Change password**—Allows users to change a password

To access the Learn Mode and Set Password functions, in the Applications tabbed page click the Advanced button, from here you can access the Advanced Settings dialog

Here is an example of the Advanced Settings dialog for the application notepad :

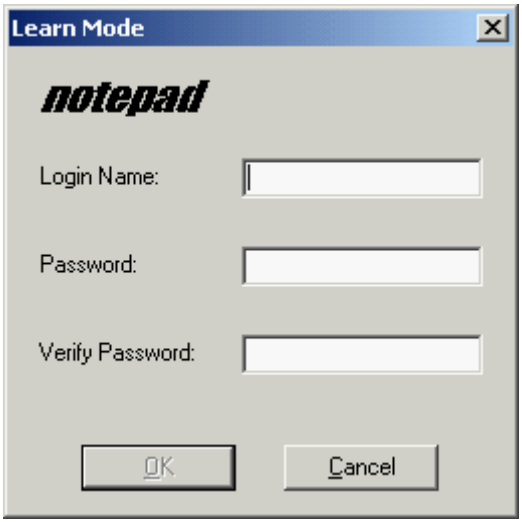


Learn Mode (First Logon Situation)

When you are populating the data stores, you can enter user credentials for every user, but to do so for every application that every user can access is a tremendous amount of work. eTrust SSO allows users to enter this information themselves using Learn Mode. For more information about Learn Mode, see the "About Authentication" chapter.

When a user invokes an application through eTrust SSO for the first time, and there are no user credentials in the data stores, eTrust SSO displays a dialog enabling the user to enter the logon name and password for the application.

Here is an example of the dialog for the application **notepad**:



After the user has entered the information and clicked OK, eTrust SSO updates the data stores with the logon name and current password, and logs the user into the application.

The Policy Server allows the user to enter user credentials only for password-based applications. If the logon type of an application is **ticket** and user credentials do not exist, the server sends a message to the SSO Client that no logon information is available and the user cannot log on to the application.

Setting and Changing Passwords

If there are user credentials (logon name and password) in the data stores, but they have expired or need to be changed, you use the Change Password dialog. For more information about setting and changing passwords, see the 'Managing Passwords' chapter.

The user can manually change the password:

1. In the Advanced Settings dialog the user clicks the button Set to receive the Learn Mode dialog.
2. The user enters the information and clicks OK.
3. eTrust SSO checks whether password quality is being monitored, and, if so, it checks whether the password entered by the user meets the criteria.
   - If the password meets the criteria, eTrust SSO updates the data stores.
   - If the password does not meet the criteria, the user is prompted for a new password.
4. The button Clear removes the application password from the database.

Calling the ssointrp Executable Directly

The SSO interpreter (ssointrp.exe or ssointrp) can be run directly from the Windows Run box or from the command line. Whenever a user invokes an application from an icon that has been dragged and dropped onto the desktop, the interpreter is called directly.

In this situation, the SSO Client will not update the icons on the desktop. For example, if the administrator changes the name of an application in the data stores or removes it altogether, users who have dragged the icon representing the application to the desktop will not succeed with invoking the application and will receive an error message.

Run the script interpreter executable from a Run dialog with the file name as an argument:

```
path/ssointrp.exe -file path/filename
```

To issue a single Tcl\sso command you can run from Start, Run in your taskbar:

```
"path\ssointrp.exe" –cmd sso_command
```

The following example brings up a message box saying "Hi John":

```
"D:\Program Files\eTrust\SSO\Client\ssointrp.exe" -cmd sso msgbox -msg "Hi John"
```

Managing Password Policies

The most important password 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. eTrust SSO cannot replace education, but it can enforce rules and policies that force users to use strong passwords. The company must emphasise that end-users are ultimately responsible for protecting their own authentication data.

Password policies define the qualities of an acceptable password. eTrust 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, eTrust 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 eTrust SSO also conform to the relevant password 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.

Note: Each password policy is represented by a record in the PWPOLICY class.

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. eTrust 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 authentication and application passwords:

- The 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 and minimum 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 21 characters.

Built-In Password Rules

The following policies are always used by the Policy Server:

- A new password cannot match previous passwords. The number of previous passwords that eTrust SSO stores is specified in the password policy.
- A new password must have at least the minimum number of alphanumeric characters, special characters, digits, lowercase characters, and uppercase characters specified in the password policy.
- A new password must not have more repetitive characters than is specified in the password policy.
- Each password must have a maximum lifetime; that is, it must expire, forcing the user to choose a new password after a certain interval.
- Each password must have a minimum lifetime. By specifying a minimum lifetime, you can prevent users from quickly and repeatedly changing passwords. With frequently changed passwords, they could overflow the password history stack and then re-use a previous password.

How the Policy Server Checks Passwords

The algorithm for the password policy check depends on the token PropDefaultMode that resides in the ssod section of the policyserver.ini file in a UNIX operating system and in the ssod section of the Policy Server registry key in a Windows operating system. The PropDefaultMode token can be set to 0 or 1:

0—Use the Pwpolicy default mode

1—Use the Property default mode (This is the default value)

After the check:

1. If there is not a specific PWPOLICY or _default PWPOLICY, then the Policy Server has no policy to check.
2. If a specific PWPOLICY does not exist but _default PWPOLICY exists, then the Policy Server checks the policy according to _default PWPOLICY.
3. If a specific PWPOLICY exists but a _default PWPOLICY does not exist, then the Policy Server checks the policy according to the specific PWPOLICY.
4. If both the specific PWPOLICY and _default PWPOLICY exist, since a PWPOLICY record consists of more then one rule (property), then the Policy Server can use _default PWPOLICY in two ways:
   a. If the password policy mode is Property Default Mode (PropDefaultMode = 1) in either the policyserver.ini file token or the Registry string, then the Policy Server checks, one by one, each rule of the specific PWPOLICY in the following way:
      - If the Policy Server encounters a nonzero value, it takes this value.
      - If the Policy Server encounters a zero value, it takes the parallel value from _default PWPOLICY.
   b. If the Password Policy mode is Pwpolicy Default Mode (PropDefaultMode = 0) in either the policyserver.ini file token or the Registry string, then the Policy Server uses only the specific policy as a whole.

Note: To define a specific policy for eTrust SSO password-based authentication, link this policy to the __SSO__ application.

Creating a New Password Policy

To create a password policy, do the following:

1. Select the Resources icon in the eTrust SSO program bar to display the Resources window.
2. Expand the Configuration Resources folder.
3. Right-click Password Policy and choose 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 the PWPOLICY Resource - General Dialog section.
5. Select other icons to define additional password policy information.
6. Click OK to finish.

Removing a Password Policy

To remove a password policy, follow these steps:

1. Select the Resources icon in the eTrust SSO 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.

Linking Policies and Applications

To link an application to a password policy, do the following:

1. In the Policy Manager, open an application entry.
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.

Defining the 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.

Setting 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 PWPOLICY _default record. However, this value is overridden for an application by the specific value in the password policy record linked to that application.

You should set the password interval value in eTrust SSO to a shorter time than the value set in the application. In this way, if logon is performed through eTrust SSO, the user is prompted by eTrust SSO for a new password before the application itself requires a new password.

Each time the Policy 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 times that the password can be used before a new password is required.

User Password Never Expires

You can configure the data store so that a password of a specific user will never expire. This is relevant 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 Expirer checkbox.
4. Click OK to close the dialog.

Automatically Generated Passwords

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

To use the feature:

1. Enable this option in the appropriate applications
2. Create password policies and attach them to the applications

   Note: You should 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.

When using the Policy Manager, to enable password auto-gen, the "Password generation enabled" checkbox should be checked in the application and in the user definitions. To attach a password policy to the application, choose a policy that appears in the authentication tabbed page of the application properties. If you want to attach a new policy, first define it.

When using Selang, to enable password auto-gen, use the pwd_autogen keyword:

```
editappl applname pwd_autogen
edituser username pwd_autogen
```

When using Selang, to attach a password policy to the application, use the command:

```
editappl applname pwpolicy(policy-name)
```

After enabling the password auto generation feature, no additional changes are required. There is no need to change the logon scripts. eTrust SSO will continue working as before, the only difference being that when the password expires (because of the password rules attached to the application), eTrust 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 *can no*t be used together.

Grace Logons, Revoke, Forced Change and Password History

eTrust SSO provides a number of related optional features that can be used with password application 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**—eTrust 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 will not be accepted and the user will be prompted for another new password.
- **Forced change**—When an administrator changes a user's password, the administrator can specify that eTrust SSO will force the user to change that password during the first logon for which it is used.
- **Revoke**—eTrust 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 an 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 eTrust 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.

On UNIX, the revoke feature is set in the revoke section of the Policy Server configuration file policyserver.ini. On Windows, this is set in the registry. For more information, see the 'Configuring the Policy Server' appendix.

Synchronizing Passwords Between Applications

It is possible to synchronize all of a user's application passwords, including the primary authentication password if eTrust 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, eTrust 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 has only one password to remember. However, this arrangement is relatively insecure: as soon as someone acquires a user's password, that someone is able to 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.

Enabling Password Synchronization

If all the properties in the data store records are properly set, then when a user selects a new password, eTrust SSO will replace the user's primary authentication password and all the user's application passwords with the new password selected by the user.

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 should only be synchronized among a certain subset of applications accessible to the user, enable synchronization for only the applications that should be synchronized.

To enable password synchronization for a user:

1. In the Policy Manager, open a user entry.
2. Select the User Attributes 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. Check the Passwd Synch Enabled checkbox.
4. Click OK to close the dialog.

Password Policy Algorithm for Synchronized Applications

When a user changes a password of a synchronized application, the Policy Server consolidates all of the password policies of the synchronized applications to one policy. The Policy 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 Policy Server. This consolidation is computed by taking the most secured rule for every rule in a PWPOLICY record.

Password Synchronization and the eTrust SSO Native Password

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

There are three ways to define how an eTrust SSO native password works with password synchronization:

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

These three ways are described in the sections below.

eTrust SSO Native Password Not Synchronized

When the user changes a password of the synchronized application, the password will be synchronized with all the synchronized applications, but the SSO native password will not be changed.

When the user changes the SSO native password, the passwords of the synchronized applications will not be synchronized with this password.

To make sure that the eTrust 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 Passwd Synch Enabled checkbox is not checked.
4. Click OK to close the dialog.

eTrust SSO Native Password Partly Synchronized

Changing the SSO native password makes the passwords of the synchronized applications to be synchronized with it.

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

To partly synchronize the eTrust SSO native password:

1. On the Policy 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 Passwd Synch Enabled checkbox.
5. On the Policy Server, set the Auto_RollOut keyname to 0:
   - On Windows, use the following registry key:

     ```
     HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\eTRust\Shared\Policy
     Server\8.0\ssod\Auto_RollOut
     ```

   - On UNIX, use the ssod section on the policyserver.ini file.

     See the appendix ‘Configuring the Policy Server’ for further information.

eTrust SSO Native Password Fully Synchronized

Changing the SSO native password makes all the passwords of the synchronized applications to be synchronized with it.

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

To fully synchronize the eTrust SSO native password:

1. On the Policy 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 Passwd Synch Enabled checkbox.
5. On the Policy Server, set the Auto_RollOut keyname to 1:
   - On Windows, use the following registry key:

     ```
     HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\eTRust\Shared\Policy
     Server\8.0\ssod\Auto_RollOut
     ```

   - On UNIX, use the ssod section on the policyserver.ini file.

     See the appendix 'Configuring the Policy Server' for further information.

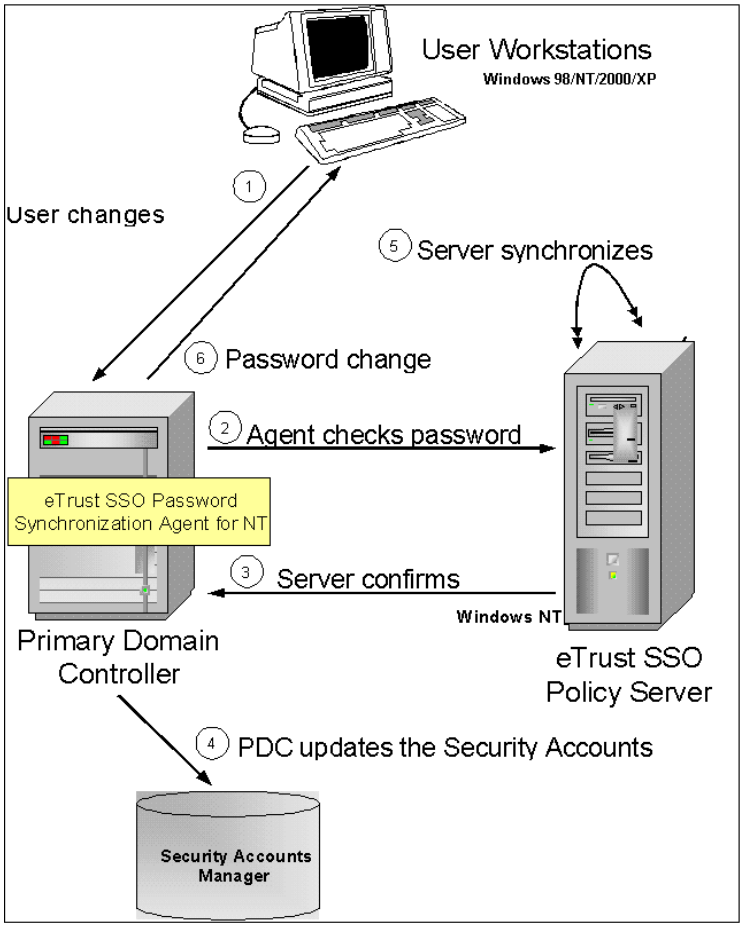
Password Synchronization Agents

Password Synchronization Agents let you keep passwords synchronized between platforms. If a user changes an SSO supported application outside of the SSO system, for example if they change their domain password on a domain-connected computer that has a password synchronization agent installed on it, then the password is automatically updated within the Policy Server.

How Password Synchronization Works on Windows

This section describes how password synchronization works on Windows. The Password Synchronization Agent for Windows works like a Windows NT password filter.

The following diagram shows how password synchronization works on Windows:



On Windows systems, when a user changes their domain password, the following steps take place (the step numbers correspond to the numbers on the previous diagram):

1. The user changes the password.
2. The password synchronization agent communicates with the Policy Server to check the password policy.
3. The Policy Server confirms the request.
4. The primary domain controller (PDC) updates the Security Accounts Manager.
5. The agent requests the Policy Server to synchronize passwords.
6. The PDC sends a positive reply to the user.

If there is an error, a message is sent to the Windows event logger on the primary domain controller. The error message may be displayed on the Event Viewer.

Setting up the Password Synchronization Agent for Windows

The Password Synchronization Agent for Windows must be installed on the primary domain controller (PDC) to enable password policy and password synchronization for Windows NT domain users.

The agent should also be installed on the backup domain controller that is designated to be the PDC.

The Password Synchronization Agent for Windows provides installation for all components needed for normal operation on the PDC.

In addition, the Password Synchronization Agent for Windows and the Policy Server must communicate with each other.

Registry Entries for the Password Synchronization Agent for Windows

The Password Synchronization Agent stores the parameters it uses in the registry.

Usually, there is no need to change these parameters after installation. In some cases, however, you might want to change some of them.

The root key is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\eTrust SSO\Password Agent NT
```

The EventMessageFile registry value should point to the full path of SSOPwdFilterMsg.dll:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog\Application\Password Agent NT\EventMessageFile
```

The Notification Packages registry value should include the word SSOPwdFilter separated by binary 0 from its original value:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Notification Packages
```

These are the two sections under the ROOT key:

- **COMM** - Communication parameters
- **SSO** - Policy Server related parameters

Note: When changing the registry, unless specified otherwise, you will have to reboot the computer for the changes to take effect.

COMM

The COMM section contains tokens that define communication between the agent and the server.

Token	Definition	Default Value
UseBlockingSockets	Use blocking sockets.	No
TimeOutConnect	The timeout period in seconds to connect to the Policy Server.	120 (seconds)
TimeOutRecv	The timeout period in seconds to receive data from the Policy Server.	60 (seconds)
TimeOutSend	The timeout period in seconds to send data to the Policy Server.	60 (seconds)

SSO

The SSO section contains tokens related to the Policy Servers.

Token	Description	Default Value
AdminPwd	Password of AdminUser.	Set at installation.
AdminUser	User name with admin permission on the Policy Server.	Set at installation.
KeyPath	Path of the public keys used for SSO authentication of AdminUser.	{ROOT}\SSO\PubKey
LogLevel	Used to set the event logging options. You can set the following values: ▪ None (do not log anything) ▪ Error (log error messages) ▪ Info (log error and information messages) ▪ Debug (log error, information and debug messages) If you change a value in the registry, you need to restart the computer for it to take effect.	Error
MsgFilePath	Location of SSO message file.	%SystemRoot%\System32

Token	Description	Default Value
Servers	If a server list contains more than one server, the server names must be delimited with a comma, a space, or a tab. The following are examples: ▪ `srv1,srv2,losangeles,Chicago` ▪ `srv1 srv2 losangeles Chicago`	Set at installation.
SyncAppl	Name of the application used for password synchronization.	Set at installation. The PDC domain name recommended

Configuring Data on the Policy Server

The Password Synchronization Agent for Windows communicates with the Policy Server in order to implement password policy and password synchronization.

The SSO authentication method is the authentication method implemented with the server.

You must define the following in the Policy Server:

- AdminUser
- SSO authentication method
- Synchronization application
- Specific user declarations
- Applications settings
- Password policy

AdminUser

The user named AdminUser is a virtual user and does not need to be defined as a Windows NT user. The user actually represents the Password Synchronization Agent for Windows.

To define the AdminUser:

- Define a user with ADMIN permissions. The user name must be the same as the AdminUser in the agent registry (set during installation). This user will communicate with the server on behalf of all other users.

SSO Authentication Method

The agent communicates with the Policy Server via the SSO authentication method.

To ensure successful communication with the server, follow these steps in the Policy Manager:

1. Create a user and assign them to use SSO authentication.
2. Link the __SSO__ application or synchronization application to the user.
3. Ensure that the Authhost permissions are allowed for the user (by default this is access to all).

For more information, see the "Managing Users and User Groups" chapter in this guide.

Synchronization Application

To define the Synchronization Application you must define the SyncAppl, which is set during installation time, in the agent registry as an application using the pwd_sync flag. To define the Synchronization Application follow these steps in the Policy Manager:

1. Go to the Application Properties
2. Select the 'Password Sync Enabled' checkbox.

Specific User Declarations

For every user in the domain, you must set up specific user declarations. To set these in the Policy Manager, follow these steps:

1. Go to User Properties.
2. Select User Options.
3. Select the Password Synchronization Enabled checkbox.

For more information about how to create a user, see the "Managing Users and User Groups" chapter in this guide.

Applications Settings

For an application to be synchronized with other applications you must set the pwd_sync flag. This flag will cause the application password of the user to be synchronized with the SyncAppl application password when the user's application password changes. It will affect the application only if the user also has the pwd_sync flag. For example:

1. Go to Application Properties
2. Select the Authentication button in the General Properties tab
3. Select the Password Sync Enabled checkbox.

Note: Make sure that the application script has the following line in it:

```
sso notify –event pwdchange –status 0 –appname ApplName
```

This command tells SSO that the script changed the application password successfully. This will take effect only if the SyncAppl password changes.

Password Policy

If you want SSO to manage a password policy for the application password you can declare a pwpolicy resource and attach it to the __SSO__ application. For example:

```
editres pwpolicy PolicyName password(interval(30) history- \
                rules (grace (1)))
editappl __SSO__ pwpolicy (PolicyName)
```

Important! *Make sure that the policy defines at least one grace logon in order not to block the agent from changing the AdminUser password when it expires. The password is not changed; it is reset to the old one. Therefore, do not declare history check either.*

How Password Synchronization Works on Mainframe

This section explains how the Password Synchronization Agent for Mainframe (MF) works.

When you give the mainframe administrator Access Control authorization to make password changes, any user password change, action on the mainframe is propagated through the password policy model hierarchy into the local eTrust Access Control user data store. The password change will be propagated to the Policy Server, changing the password of the synchronizing application of the user on the Policy Server.

The Password Synchronization Agent for MF will change the user's current password of the chosen synchronization application (the default application is __SSO__) in the data store (current password field) when the user's Mainframe password is changed.

Each time a request is made, Password Synchronization Agent for MF communicates with the Policy Server to get the information needed and to set the new information for the user.

On mainframe systems, when a user changes their domain password, the following steps take place:

1. The user changes the password in the Mainframe environment.
2. Password Synchronization Agent for MF communicates with the Policy Server to check the password policy.
3. The Policy Server replies to the request.
4 The agent requests the Policy Server to synchronize the passwords.
5 The Policy Server synchronizes passwords.
6 The Policy Server confirms the password change to the user.

Setting up the Mainframe Password Synchronization Agent

Note: The Mainframe Password Synchronization Agent must be installed on a separate computer to the Policy Server.

Installation Requirements on the Mainframe

On each mainframe that you want to add to the password Policy Model hierarchy, you must install CA Common Services. You can find instructions for this installation and for configuring the mainframe for password synchronization in the following locations:

- For eTrust CA-ACF2 Security, in the eTrust CA-ACF2 Security Administrator Guide

- For eTrust CA-Top-Secret Security, in the eTrust CA-Top-Secret Security Administrator Guide
- For RACF, in the OS/390 Administrator Guide

Installation Requirements on Windows

To implement password synchronization between mainframes and eTrust Policy Server, choose a Windows machine running eTrust Access Control to serve as a parent to the mainframe and to propagate the password changes to the eTrust Policy Server. Ensure that the mainframe password synchronization option is installed on this machine.

To install the Mainframe Password Synchronization option, select the Mainframe Password Synchronization Agent from the eTrust SSO CD.

The following components are installed:

- Mainframe Password Synchronization Agent
- CA-Common Services (This is a stand-alone version of the Computer Associates Common Communication Interface.)
- Access Control Exit (This exit is responsible for synchronizing the password change requests from the Mainframe Password Synchronization Agent to the Policy Server)

If you already have Unicenter TNG, the mfsd service uses the CAICCI communication package. In this case, the installation does not install CAICCI. It instead proposes shutting down Unicenter TNG (if it is running) to upgrade its security option file *UniDir*/secopts (where *UniDir* is the directory in which Unicenter TNG is installed.) Remember to restart Unicenter TNG after eTrust SSO installs. If you do not have Unicenter TNG, the CAICCI package is installed in [C:\CA_APPSW.]

The installation lets you subscribe hosts to the Policy Model. If you have the host names and SYSIDs for the mainframes, you can subscribe them now. Otherwise, you can skip this step and subscribe these hosts later. If you choose to enter host names later, you can update the CAICCI configuration file and restart the CAICCI processes to establish a CCI connection between the newly subscribed mainframe and your Mainframe Password Synchronization Agent.

Checking the Installation

1. After installation of the MF Password Syncronization Agent, open the Windows Task Manager and choose the process Tab. The following processes should be running:
   - Mfsd.exe
   - Mfscpfd.exe
   - Eacmfs.exe

2. Open Windows Services. The following services should be running:
   - CA-Unicenter(NR-Server)
   - CA-Unicenter(Remote)
   - CA-Unicenter(Transport)
   - eTrust Access Control MF Synchronization Agent

Configuring the MF_PMDB Policy Model

The Mainframe Password Synchronization Agent installation creates a Policy Model MF_PMDB which will be used by the Mainframe Password Synchronization Agent. The mainframe subscribed during the installation will automatically be subscribed to the MF_PMDB Policy Model.

To subscribe other Mainframes to the Policy Model , please follow the steps below for **each** Mainframe :

1. Open a command prompt.
2. Navigate to the eTrust Access Control installation directory.
3. Type the following:

   ```
   Sepmd –sm MF_PMDB <mf_hostname> <mf type> <mf_sysid> <mf_admin_user> 0
   ```

   Where

   mf_hostname is the hostname of the mainframe

   mf_type is the type of mainframe. This could be RACF,ACF2 or TSS for Top Secret.

   mf_sysid is the system id for the mainframe. Note that this id cannot exceed 8 characters.

   mf_admin_user is the Mainframe administrator user.
4. To verify that the mainframe has been successfully subscribed to the Policy Model, type `sepmd –L MF_PMDB.` You should be able to see all the subscribed Mainframes as well as the local hostname.

For each mainframe you subscribed to the Policy Model, you must perform the following steps in selang:

1. Connect to the MF_PMDB policy model

   ```
   eTrustAC>hosts (MF_PMDB@localhost) uid(mf-admin) password(<mf-admin's password)
   ```

2. Create a MFTERMINAL record in the MF_PMDB policy model

   ```
   eTrustAC> nr MFTERMINAL <mf_sysid> defacc(none) owner(nobody)
   ```

   mf_sysid is the Mainframe's system id. The system id should not exceed 8 characters.

3. Create a USER record for each Mainframe adminstrator user

   ```
   eTrustAC>nu (<mf_admin_user>) admin auditor audit(a) native
   ```

4. Set the password of the Mainframe Administrator user

   ```
   eTrustAC>eu (<mf_admin_user>) password(<mf_admin_password>)
   ```

5. Authorize the administrator user created in Step 2 to the TERMINAL record of the local machine.

   ```
   eTrustAC>authorize TERMINAL (<hostname>) uid( <mf_admin_user>) access(All)
   ```

   hostname is the hostname of the local machine.

6. Authorize the administrator user created in Step 2 to MFTERMINAL record.

   ```
   eTrustAC> authorize MFTERMINAL <mf_sysid> uid( <mf_admin_user>) acc(r)
   ```

Configuring CAICCI

The Mainframe Synchronization Agent uses CAICCI to communicate with the Mainframes. This section details the CCI configuration on the Windows Machine with the Mainframe Password Synchronization Agent installed. You need to configure the CAICCI on the Mainframes to recognize the Windows node. Please refer to the respective Mainframes guide for more information on how to configure CAICCI on Mainframes.

During the installation of the Mainframe Password Synchronization Agent, the CAICCI configuration file is automatically updated to include the subscribed Mainframe. For each subsequent Mainframe subscribed after the installation, you need to manually update the CAICCI configuration file to include the new Mainframe subscribers and the CAICCI processes need to be restarted.

The CAICCI configuration file is located at c:\Program Files\CA\CCI_SA\cci\CAIUSER\ccirmtd.rc.

To configure CCI on a Windows machine follow these steps:

1. Ensure that you can `PING` the mainframe system. If you cannot, you might need to modify the TCP/IP settings on the Windows machine as well as the one on the mainframe.
2. Edit the ccirmtd.rc file. The following is the format for the CCI configuration file:

   a. The `LOCAL` statement applies to the local computer. The format is:

   ```
   LOCAL <TCP/IP Name> <CCI Name> <buffer size>
   <startup option> <port options> <retry interval>
   ```

   b. The `REMOTE` statement applies to the remote nodes – these nodes can be mainframes, UNIX or Windows machine. The format is:

   ```
   REMOTE <TCP/IP Name> < CCI Name> <buffer size>
   <startup option> <port options> <retry interval>
   ```

   The following options apply for both the LOCAL and REMOTE statements:

   - *TCP/IP Name*: Either an IP address or a name that is used as input to a name service to retrieve an IP address.
   - *CCI name*: The default is the hostname.
   - *buffer size*: A value between 1024 and 32768 (default) used for segmenting the data transfer. It is generally not necessary to alter this field.
   - *start-up options*: Either `STARTUP` (default) or `NOSTART`. `STARTUP` tells CAICCI to attempt a remote connection when activated. `NOSTART` implies that the remote system will be initiating the connection on the node.
   - *alias option*: Optional alias name used to differentiate multiple remote computers having exactly the same first eight characters or when their hostname exceeds eight characters. The format is `<ALIAS=aliasname>.`
   - *port options*: Optional numeric value set by default to 1721.
   - *retry interval*: Number of seconds berween retry connect attempts.

   Examples:

   The following statement tells CAICCI that the TCP/IP name for the local machine is `NTSRVR1` and that any remote system wishing to communicate with this system can do so by referencing `NTSRV1` as the name.

   ```
   LOCAL=NTSRV1 NTSRV1 32768 STARTUP
   ```

   The following statement tells CAICCI to attempt a connection to 141.222.111.121 and to internally register MF01 as the CCI name.

   ```
   REMOTE=141.222.111.121 MF01 32768 STARTUP PORT=1721
   ```

Determine if the CCI Remote Server is running by issuing the following command:

```
ccicntrl
```

```
%ccicntrl
Service "CA-Unicenter (Transport)", STATUS is "Running"
Service "CA-Unicenter (NR-Client)", STATUS is "Not Installed"
Service "CA-Unicenter (NR-Server)", STATUS is "Running"
Service "CA-Unicenter (Remote)", STATUS is "Running"
```

3. If you have modified the `ccirmtd.rc` file and the Remote Server is active, issue the following commands:

   ```
   ccicntrl stop
   ccicntrl start
   ```

```
%ccicntrl stop
Stopping Service "CA-Unicenter (Transport)"
Stopped Service "CA-Unicenter (Transport)"
Stopping Service "CA-Unicenter (NR-Server)"
Stopped Service "CA-Unicenter (NR-Server)"
Stopping Service "CA-Unicenter (Remote)"
Stopped Service "CA-Unicenter (Remote)"

%ccicntrl start
Starting Service "CA-Unicenter (Transport)"
Started Service "CA-Unicenter (Transport)"
Starting Service "CA-Unicenter (NR-Server)"
Started Service "CA-Unicenter (NR-Server)"
Starting Service "CA-Unicenter (Remote)"
Started Service "CA-Unicenter (Remote)"
```

Creating Users

For every Mainframe user, you must create a USER record in both the MF_PMDB and the Policy Server.

Note: The Mainframe Password Synchronization Agent (*MF_PMDB)* must installed on a separate computer to the Policy Server.

To create a user in the MF_PMDB Policy Model:

1. Open a command prompt.
2. Type: selang.
3. Connect to the MF_PMDB:

   ```
   eTrustAC>hosts (MF_PMDB@localhost) uid(mf-admin)  password(<mf-admin's password>)
   ```

4. Create a user:

   ```
   eTrustAC>nu <user-id> native
   ```

5. Authorize the user to the localhost TERMINAL record:

```
eTrustAC> authorize TERMINAL (<hostname>) uid(<user-id>) access(All)
```

6. Authorize the user to the Mainframe MFTERMINAL record. If the user resides in all mainframes, they must be authorized to all Mainframes' MFTERMINAL records.

   ```
   eTrustAC>authorize MFTERMINAL(<mf_sysid>) uid(<user-id>)  access(r)
   ```

To create a user in eTrust Policy Server:

1. Log onto the machine that the eTrust Policy Server resides on.
2. Using the Policy Manager, create a user in the default user data store.
3. Authorize the user to the synchronizing application.
   This is typically __SSO__.

Registry Entries for the Mainframe Password Synchronization Agent

The Mainframe Password Synchronization Agent stores the parameters it uses in the registry.

Usually, there is no need to change these parameters after installation. In some cases, however, you might want to change some of them.

The root key is:

```
HKEY_LOCAL_MACHINE\SOFTWARE\ComputerAssociates\eTrust SSO\Password Agent MF
```

These are the two sections under the ROOT key:

- **COMM** - Communication parameters
- **SSO** - Policy Server related parameters

Note: When changing the registry, you will have to reboot the computer for the changes to take effect.

COMM

The COMM section contains tokens that define communication between the agent and the server.

Token	Definition	Default Value
TimeOutConnect	The timeout period in seconds to connect to the Policy Server.	120 (seconds)
TimeOutRecv	The timeout period in seconds to receive data from the Policy Server.	60 (seconds)
TimeOutSend	The timeout period in seconds to send data to the Policy Server.	60 (seconds)

SSO

The SSO section contains tokens related to the Policy Servers.

Token	Description	Default Value
Servers	If a server list contains more than one server, the server names must be delimited with a comma, a space, or a tab. The following are examples: ▪ `srv1,srv2,losangeles,Chicago` ▪ `srv1 srv2 losangeles Chicago`	Set at installation.
AdminUser	User name with admin permission on the Policy Server.	Set at installation.
AdminPwd	Password of AdminUser.	Set at installation.
SyncAppl	Name of the application used for password synchronization.	Set at installation. Defaults to __SSO__
KeyPath	Path of the public keys used for SSO authentication of AdminUser.	{ROOT}\SSO\ PubKey
MsgFilePath	Location of SSO message file.	%SystemRoot %\System32

Token	Description	Default Value
ContinueToAccessControl	Specify whether the Mainframe Password Synchronization Agent should use Access Control to verify the user's password. Note that if this is set to yes, the verification is done when the password option is set to true in Access Control.	Yes
UserDataStore	The user data store on the Policy Server where the user resides	Set at installation.
SearchFilter	Search filter used when looking for users in a LDAP user data store.	Set at installation.
LogCfg	Full path to the log config file	Default: <Mainframe Password Sync Agent install_path>/eTrustSsoMf_log.cfg

Configuring Data on the Policy Server

The Mainframe Password Synchronization Agent communicates with the Policy Server in order to implement password policy and password synchronization.

The SSO authentication method is the authentication method implemented with the server.

You must define the following in the Policy Server:

- AdminUser
- SSO authentication method
- Synchronization application
- Specific user declarations
- Applications settings
- Password policy

AdminUser

The Administrator User is the user used by the Mainframe Password Synchronization Agent when it is changing password on the Policy Server. The user name must be the same as the AdminUser in the agent registry (set during installation). This administrator user should reside in the same user data store as the one specified in the UserDataStore registry setting.

To define the AdminUser:

- Define a user with ADMIN permissions. The user name must be the same as the AdminUser in the agent registry (set during installation). This user will communicate with the server on behalf of all other users.

Synchronization Application

To define the Synchronization Application:

Define the SyncAppl (set at installation time) in the agent registry as an application using the pwd_sync flag. To define the Synchronization Application follow these steps in the Policy Manager:

1. Go to the Application Properties
2. Select the 'Password Sync Enabled' checkbox.

Specific User Declarations

For every user in the domain, you must set up specific user declarations. To set these in the Policy Manager, follow these steps:

1. Go to User Properties.
2. Select User Options.
3. Select the Password Synchronization Enabled checkbox.

For more information about how to create a user, see the "Managing Users and User Groups" chapter in this guide.

Applications Settings

For an application to be synchronized with other applications you must set the pwd_sync flag. This flag will cause the application password of the user to be synchronized with the SyncAppl application password when the user's application password changes. It will affect the application only if the user also has the pwd_sync flag. For example:

1. Go to Application Properties
2. Select the Authentication button in the General Properties tab
3. Select the Password Sync Enabled checkbox.

Note: Make sure that the application script has the following line in it:

```
sso notify –event pwdchange –status 0 –appname ApplName
```

This command tells SSO that the script changed the application password successfully. This will take effect only if the SyncAppl password changes.

Password Policy

If you want SSO to manage a password policy for the application password you can declare a pwpolicy resource and attach it to the __SSO__ application. For example:

```
editres pwpolicy PolicyName password(interval(30) history- \
    rules (grace (1)))
editappl __SSO__ pwpolicy (PolicyName)
```

Important! *Make sure that the policy defines at least one grace logon in order not to block the agent from changing the AdminUser password when it expires. The password is not changed; it is reset to the old one. Therefore, do not declare history check either.*

One-Time Passwords

The most common way to authenticate a user is by requesting a password, and checking whether it matches the password recorded for that user.

The Problems with Passwords

Passwords suffer from two potential problems:

- The password is stored on a server. If the server's security is compromised, the password may become accessible, rendering it public, weakening the whole authentication process. The solution used in UNIX is to store a derived form of it, using a one-way encryption function.
  The problems are: The encryption function is known; and weak passwords are crackable, by using a dictionary.
- Systems connected to a network (including the Internet) are vulnerable to eavesdropping on network connections. Eavesdroppers obtain logon IDs and passwords of legitimate users. The captured logon ID and password can later be used to gain access to the system, in a replay attack.

Solving Password Problems by Using One-Time Passwords

The eTrust SSO one-time password agent is designed to counter these types of attacks. The OTP authentication type can eliminate the security risk of sending passwords across a network in clear text, and is recommended where applications cannot use a ticket-based system.

With one-time passwords (OTP), passwords are still sent across the network, but they cannot be used to log on a second time, so they are useless to whoever intercepts them. Also, the OTP cannot be used to discover the seed password stored in the policy data store.

On UNIX, the one-time password agent is built in to eTrust SSO. This agent can secure applications such as telnet, ftp, or any other UNIX application that uses the user database /etc/passwd.

For eTrust SSO, this functionality is useful for customers that are not interested in implementing Kerberos (CyberSAFE) and do not have any other strong authentication method (such as DCE).

Note: The one-time password logon method does not prevent a network eavesdropper from gaining access to private information, and does not provide protection against "inside jobs" or against active attacks where the potential intruder is able to intercept and modify the packet stream.

When eTrust SSO uses the one-time password authentication type, the Policy Server generates a unique password each time the user selects an OTP-authenticated application.

This OTP is a unique derivative of a seed password that is stored in the policy data store.

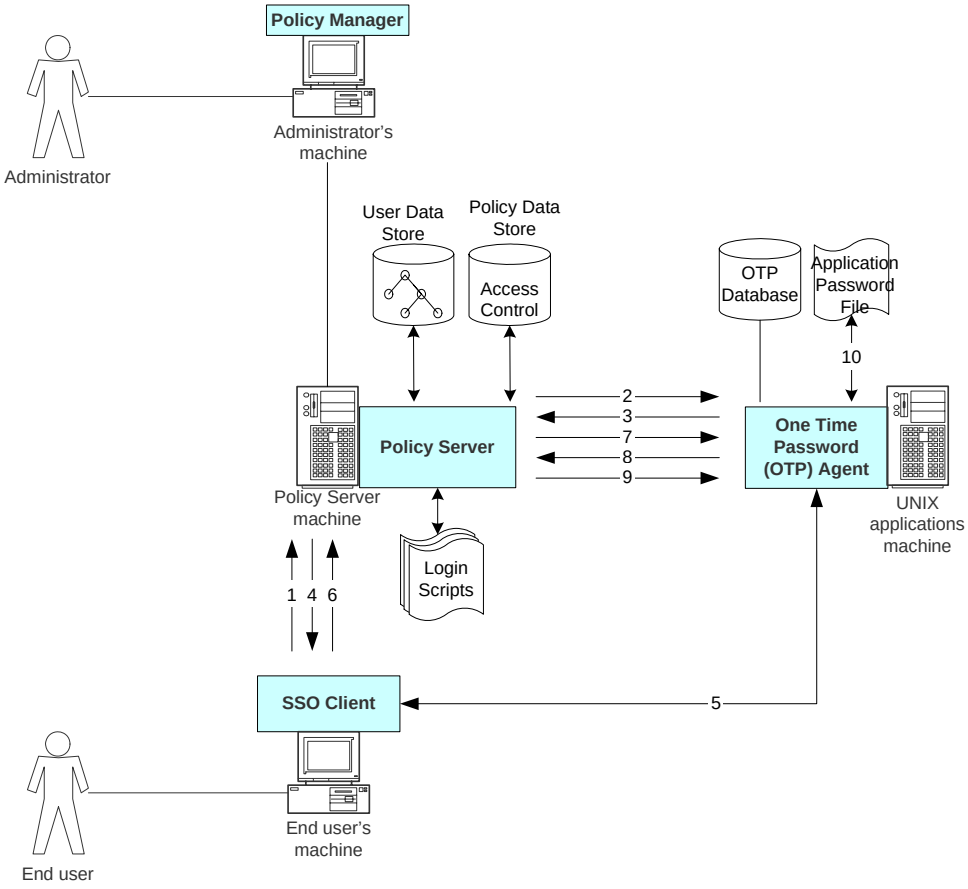
To run applications using OTP authentication, the following items need to be installed on the application host:

- The OTP agent
- A small OTP database

Only a single use-password (derived from the user's secret password) ever crosses the network. The user's secret password never crosses the network at any time, during logon or when changing the user's password. Thus, it is not vulnerable to eavesdropping/replay attacks. Added security is provided by the fact that no secret information need be stored on any system, including the host being protected.

The OTP agent allows user's secret password of any length, that is combined with a non-secret seed that can be set for each machine. This non-secret seed allows a user to use the same secret password on multiple machines (using different seeds) and to safely recycle secret passwords by changing the seed.

How OTP Authentication Works



1. From the application list, the end user selects an application that is defined as requiring an OTP logon. The SSO Client sends the user authentication details (from the primary authentication) and the application identifier to the Policy Server.
2. The Policy Server identifies the application requested as an OTP application, and asks the OTP agent on the application host for a challenge. This challenge is a value that the Policy Server needs to prepare the proper password for this logon. The challenge is specific to a particular pair of application and end user.
3. The OTP agent retrieves the challenge from the OTP database and sends it to the Policy Server.
4. The Policy Server creates a one-time password for the requested application, and sends it to the SSO Client.
   It does this by running a one-way function whose parameters include the challenge and the seed password that is stored in the application's logon info in the Policy data store.

5. The SSO Client runs the logon script, which starts the application and enters the user ID and the newly created one-time password. The application checks the password against the password stored in the application's password file, and if the password is correct, lets the end user work in the application.
6. The SSO Client monitors the running of the script, checks the results of the logon attempt, and notifies the Policy Server.
7. If the logon attempt was successful the Policy Server tells the OTP agent to update the challenge.
8. The SSO Agent updates the challenge, returns the challenge value to the Policy Server, and updates the challenge value in the OTP database.
9. The Policy Server prepares a new password for the application password file, using the one-way function with the new challenge value. The Policy Server encrypts the new one-time password in a system-compatible format and sends it to the ORP agent.
10. The OTP agent updates the password in the application password file.

The OTP Database

The OTP agent keeps a small database on the host on which it runs. This database has an entry for each OTP use, with the parameters used to generate a challenge.

The OTP database is created when the OTP agent is installed. In most cases, you do not have to deal with this database directly.

If the Policy Server asks for a challenge for a user who is not in the OTP database, the OTP agent replies that the user is not found. The Policy Server assumes that this is the first time the user has attempted to log on to the application, and instructs the OTP agent to create an entry for the user in the OTP database. This is automatic.

The user is added to the OTP database only if the user already appears in the operating system user data store. For example, this might be /etc/passwd on UNIX systems.

Chapter

8

Managing User Sessions

This chapter describes what session management is, and how to set it up.

eTrust SSO Sessions

An eTrust SSO session is the period of time that a user is logged on to eTrust SSO. During an eTrust SSO session, the user may be logged on to other eTrust SSO-enabled applications.

By default, eTrust SSO lets users have multiple concurrent sessions on different machines. Using eTrust 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 work with sessions manually using the Session Administrator.

Benefits of Session Management

Using the Policy Manager and Session Administration in the IA Manager 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 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 any sessions

To protect sensitive information, you can do the following:

- Set an idle time-out for locking the machine
- Set an idle time-out for logging the user off the eTrust SSO session as well as logging out the underlying Windows user
- Manually terminate any sessions

How eTrust SSO Controls User Sessions

eTrust SSO enables you to manage user sessions in the following ways:

- Set up automatic session management rules
- Work with sessions manually

Automatically Control User Sessions with the Policy Server

Using the Policy Manager, you can set up session profiles that define how the Policy 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 eTrust SSO. This timeout must be greater than the screen-lock timeout otherwise it will not be taken into account. eTrust SSO as well as the underlying Windows user is logged out after (logoff timeout - screen-lock timeout) time.

Manually Control User Sessions with the Session Administrator

In addition to storing automatic session profiles on the Policy Server, you can also manually track and terminate sessions using Session Administration in the IA Manager. 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

See The Session Administrator section for more information.

How Session Management Works

This section describes how session management works, and how you can change the session management settings to make it work best for your organization.

Terminating User Sessions

You can configure the Policy Server to terminate SSO Client sessions in two ways:

Direct Notification
This is the default method. It is quicker, but it only works if there is no network address translation, such as a firewall or IP chaining, between the SSO Client and the Policy Server.

Heartbeat Response
This method relies on the SSO Client sending a heartbeat to the Policy Server. You should use the Heartbeat Response method if your system uses network address translation between the SSO Client and the Policy Server.

Method 1: Direct Notification (Default)

This is the faster method of session termination.

When the user attempts to start a session, the Policy 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 Policy Server sends a notification to the relevant SSO Client, instructing the client to take action (terminate one or all sessions, ask the user which session to terminate, or deny access).

For the Direct Notification method to work, the eTrust SSO system must have the following:

- The SSO Client must be listening for notification messages from the Policy Server on a particular port.

  You can change this port in the SsoClnt.ini file. For more information, see the appendix "Configuring the SSO Client: SsoClnt.ini".

- The network must permit the Policy 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 SSO Client sends a heartbeat to the Policy Server at a regular interval.

The Direct Notification method may not be suitable for your system because the Policy Server cannot send messages directly to the SSO Client. If this is the case, you can configure the Policy Server to use the routine heartbeat response to send a termination message to an SSO Client.

This method causes a delay between the time at which the Policy Server decides to terminate a session, and the time at which the terminate session message is actually sent to the SSO Client. This is because the Policy Server must wait for a heartbeat from the SSO Client before it can notify the client.

Heartbeats from the SSO Client to the Policy Server

By default, the SSO Client sends a heartbeat to the Policy Server.

The SSO Client is set to terminate a user session if the Policy 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 an eTrust SSO session, and then stopping the SSO Client heartbeat reaching the Policy Server (for example, by disconnecting their machine from the network). Permitting this can enable the user to start a subsequent malicious session.

Applying 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 Policy Server Settings section lists all session management settings, and the parameters in order of increasing restrictiveness.

Example of 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.

For a list of all session management settings, see the table in the Policy Server Settings section.

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	ADMIN	ADMIN	-
Limit Choice	Close Oldest Session	Select Specific Session	Close Oldest Session
Heartbeat Fail Behavior	None	Logoff from SSO	Logoff from SSO
Logoff Timeout	5 minutes	10 minutes	5 minute
Screen Lock Timeout	1 minutes	2 minutes	1 minutes
User Max Sessions	1 session	3 sessions	1 session

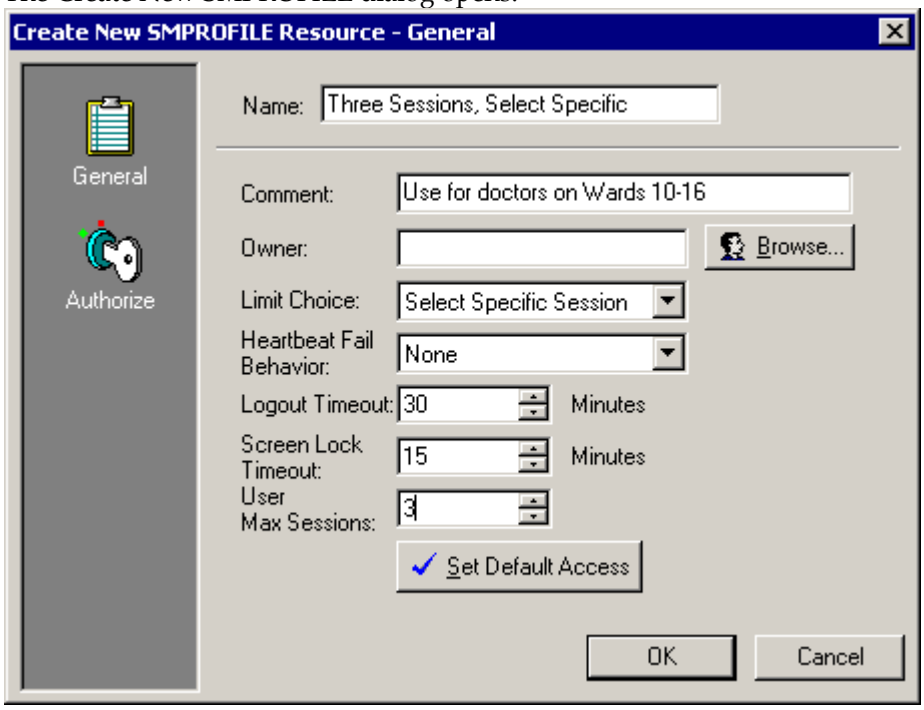
Work with Session Profiles

To manage user sessions, use the Policy Manager to create one or more session profiles. Then, apply the session profiles to individual users or whole groups.

Create a Session Profile

To create a session profile that can be applied to users to define their eTrust SSO session behavior, follow these steps.

1. Launch the Policy Manager.
2. Click the Resources icon, then select Single Sign-On Resources, Session Resources, Session Profile in the tree.
   The list of existing session profiles appears.
3. Right-click anywhere in the list area, and select New.
   The Create New SMPROFILE dialog opens.



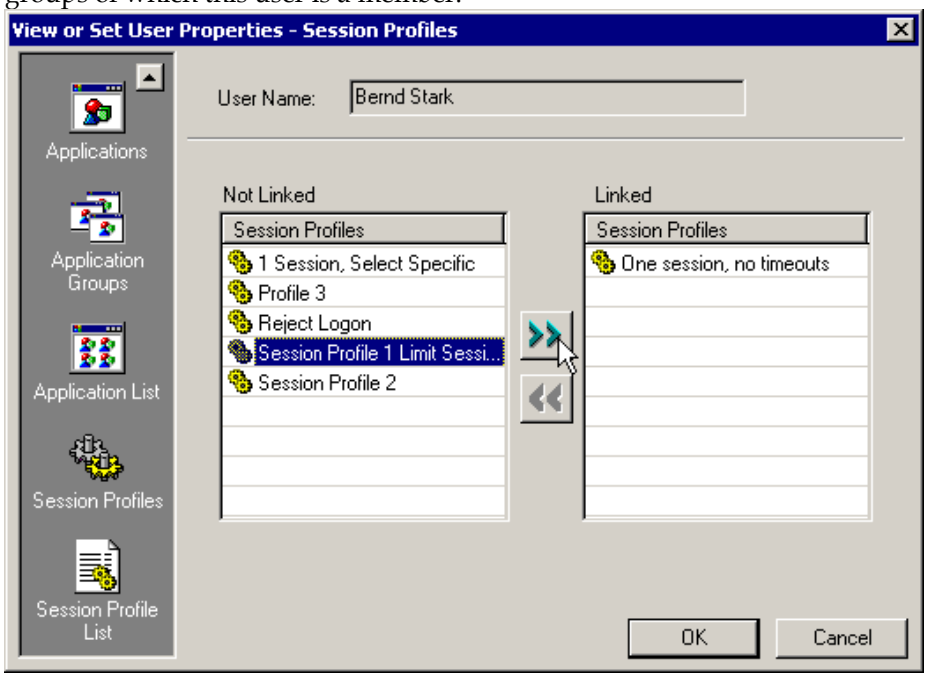
4. In the General dialog, set the behavior for the profile.
   Use the online help to get extra information about the options.
5. In the Authorize dialog, set permissions for users or groups to access the new session profile.
6. Click OK to save the new profile.

Apply a Session Profile to a Single User

To apply a session profile to a user to control their eTrust SSO session behavior, follow these steps.

To apply a session profile to a single user, follow these steps:

1. In the Policy Manager, open the Users section, and double-click a group or user name to open the User Properties dialog.
2. Select the Session Profiles section on the left of the dialog. The list shows the groups of which this user is a member.

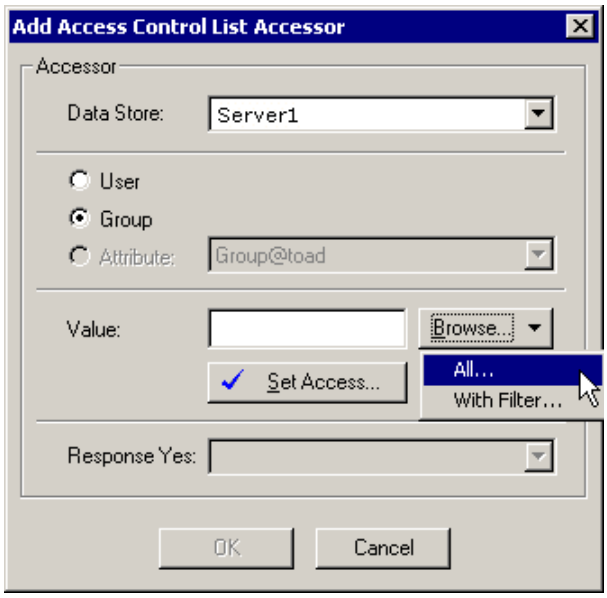


3. Select one or more session profile names in the list on the left, and click the  icon to apply the session profiles.
4. Click OK to close the dialog. The session profiles are assigned to the user.

Apply a Session Profile to a Group

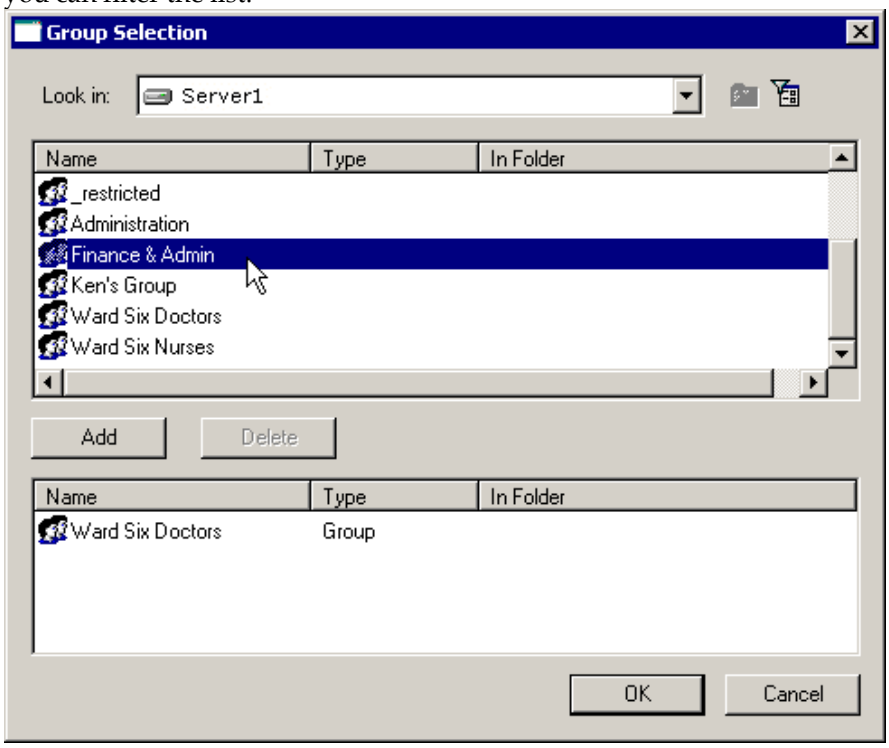
To apply a session profile to a user group to control their eTrust SSO session behavior, follow these steps.

1. In the Policy Manager, navigate to the Resources, Session Profiles section.
2. Double-click a session profile name to open the View or Set SMPROFILE Properties dialog.
3. Click the Authorize icon in the bar on the left.
4. Click the Add icon. The Add Access Control List Accessor dialog appears.



5. In the Add Access Control List Accessor dialog, make the following selections:
   a. Select the data store that stores the group.
   b. Select the Group option.
   c. If you know the exact name of the group, enter the group name in the Value field, and click OK.

Otherwise, 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.



d. In the Group Selection dialog, select one or more group names on the top pane, and click Add.

6. Click OK on all open dialogs to return to the main Policy Manager window. The session profile is applied to the groups you selected.

Session Management Settings

You can configure Session Management in two place; on the Policy Server via the Policy Manager, and on the SSO Client using the SsoClnt.ini file. This section explains how to adjust the Session Management settings in both locations.

Policy Server Settings

The following table describes each Session Management settings that you can adjust.

The parameters are listed in order of least restrictive to most restrictive. This is important if you plan to apply more than one session profile to a user. For more information, see the Applying Multiple Session Profiles section.

To Alter these settings follow these steps:

1. Launch the Policy Manager.
2. Go to Resources, Configuration Resources, Policy Server Settings
3. Double-click on Session Management

   The View or Set GPSCONFIGPROPERTY Properties – Settings dialog appears.
4. Double slick on any setting to change it. Refer to the table below for a description of each setting.

Setting	Description	Parameters
AllowTicketFromMultipleStations	Lets two machines use the same SSO ticket. Required for Citrix MetaFrame environments.	**Disabled** (default) **Enabled**
HeartbeatFailAfter	Maximum number of heartbeats missed before the client session terminates. For example, if set to 3, the 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)
HeartbeatInterval	Period between heartbeats	**0**—Disabled (no heartbeat is sent) **Integer**—Period in seconds between heartbeats (the default is 30 seconds)

Setting	Description	Parameters
HeartbeatProtocol	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)
SendSessionTermination	Whether or not the Policy Server sends session messages directly to SSO Client	**Disabled** — The Direct Notification method is disabled **Enabled** — The Direct Notification method is enabled (default)
SessionlessTerminals	List of terminals exempt from session management	**Blank**—All terminals are subject to session management (default) **IP address**—Terminal is exempt from session management (list is comma-delimited)
SessionTerminationTimeout	How long the Policy 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 40 seconds)
SessMgmtEnable	Whether Session Management is enabled or not. **Note:** If you set Session Management to Required then you must enter the IA Manager IP address as a Sessionless Terminal.	**Disabled** — (default) **Enabled** — Lets SSO Clients from eTrust SSO 6.5 work **without** Session Management, and SSO Clients from eTrust SSO 7.0 and 8.0 work **with** Session Management **Required** — If an eTrust SSO 6.5 Client (or earlier) starts, it attempts to connect to the Policy Server and then closes immediately.

Sample Policy Server Settings

To use any of the Session Management methods described in the How Session Management Works section, you must configure the Policy Server correctly.

Method	Setting	Required Value
No Heartbeat Heard	SessMgmtEnable	Any setting
	SendSessionTermination	Any setting
	HeartbeatInterval	An integer
	HeartbeatFailAfter	An integer
Direct Notification	SessMgmtEnable	Enabled
	SendSessionTermination	Enabled
Heartbeat Response	SessMgmtEnable	Enabled or Required
	SendSessionTermination	0
	HeartbeatInterval	An integer

SsoClnt.ini Settings

To use the Direct Notification method, the SSO Client must be listening for notification messages from the Policy Server on a particular port.

You may need to change the range of port numbers if you know that another application on your network routinely uses a port in this range.

You can change this port in the Session Management section of the SsoClnt.ini file:

```
[Session Management]
ClientPortRange=20001-20201
```

The ClientPortRange value specifies the range of port numbers to use (a single port number cannot be guaranteed). Port numbers are tried consecutively, from the lowest to the highest inclusively.

You must also configure the following values when you configure Session Management.

```
[SSO]
CleanTicketOnStart=no
```

For more information, see the appendix "Configuring the SSO Client: SsoClnt.ini."

The Session Administrator

The Session Administration is a web-based application that lets you view and terminate eTrust SSO sessions. Session Administration is part of the IA Manager.

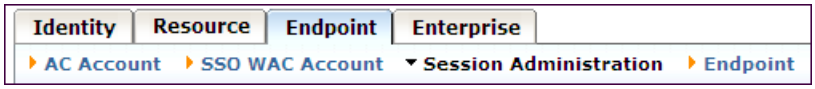
Launch Session Administrator

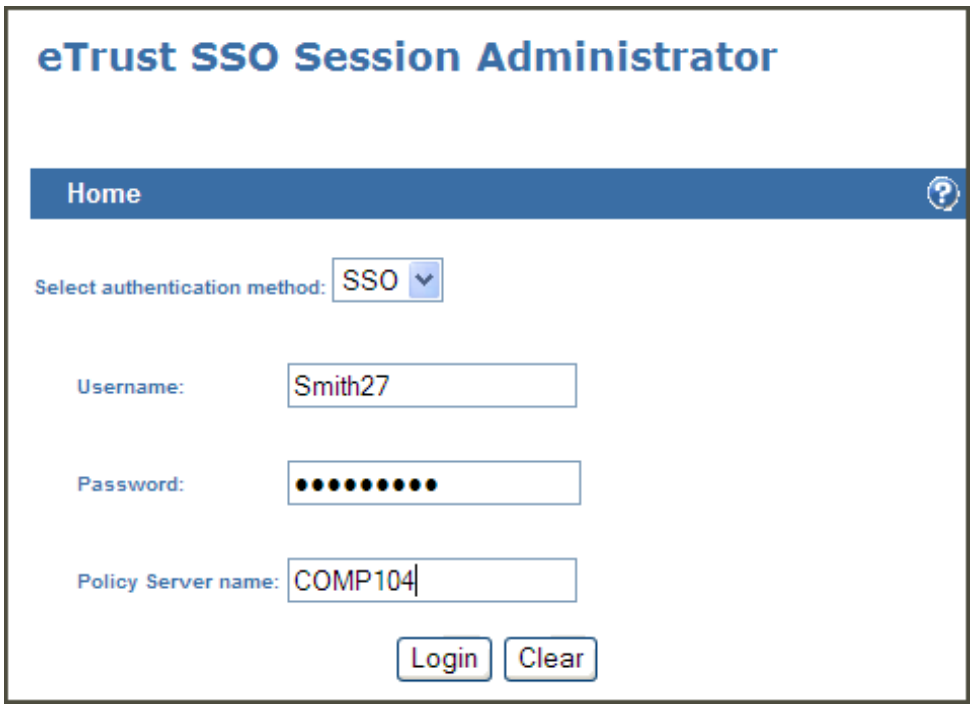
To log on to the Session Administrator, you must be a Session Administrator eTrust SSO user.

1. Launch the IA Manager.

   For information about installing the IA Manager, see the "Implementing the IAM Common Components" chapter in the *Administrator Guide.*

2. Go to the Endpoint tab, and select Session Administration





Create a Session Administrator User

You should assign administrator privileges to a user in both the LDAP data store (eTrust Directory) and the eTrust AC data store.

To create a new Session Administrator user, follow these steps:

1. In the Policy Manager, create a new user in the LDAP user data store and a new user in the eTrust AC data store with the same name and password.
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)
   ```

   To assign a user administrator rights in eTrust AC, run this command:

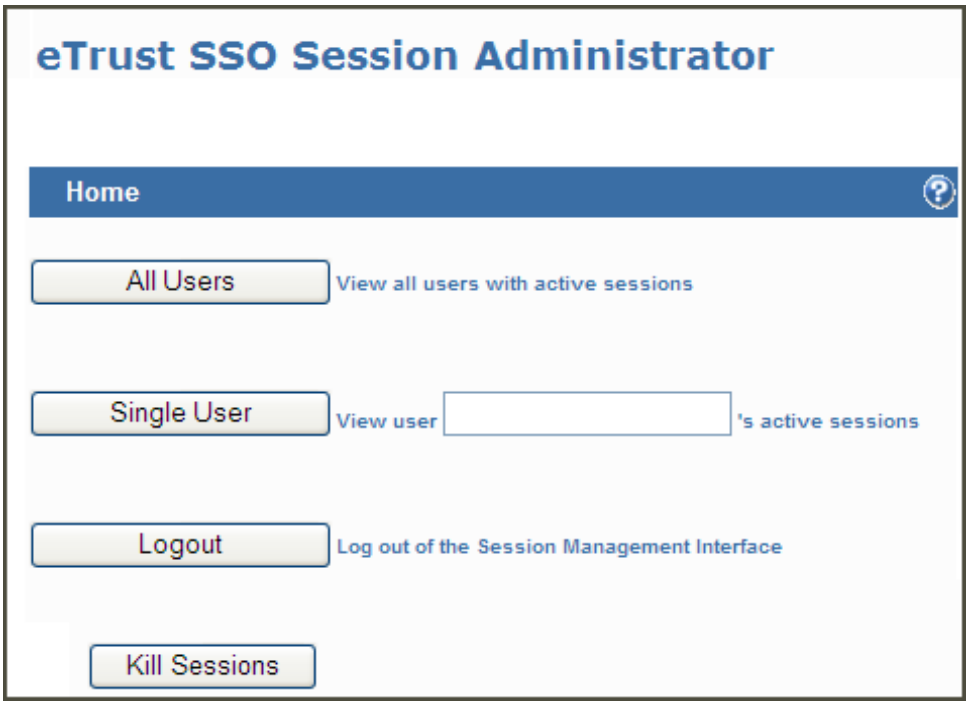
   ```
   authorize ROLE ADMIN user_attr("User@<eTrust AC computer name>") \
   attr_val("<username>") user_dir("<eTrust AC computer name>") access(Read)
   ```

   The new users will now be Session Administrator users. This means that they will be able to view and shut down sessions in Session Administration in the IA Manager.

Work with the Session Administrator

Using the Session Administrator you can:

- View all users who have one or more eTrust sessions open
- View all eTrust SSO sessions that a single user has open
- Terminate some or all sessions.

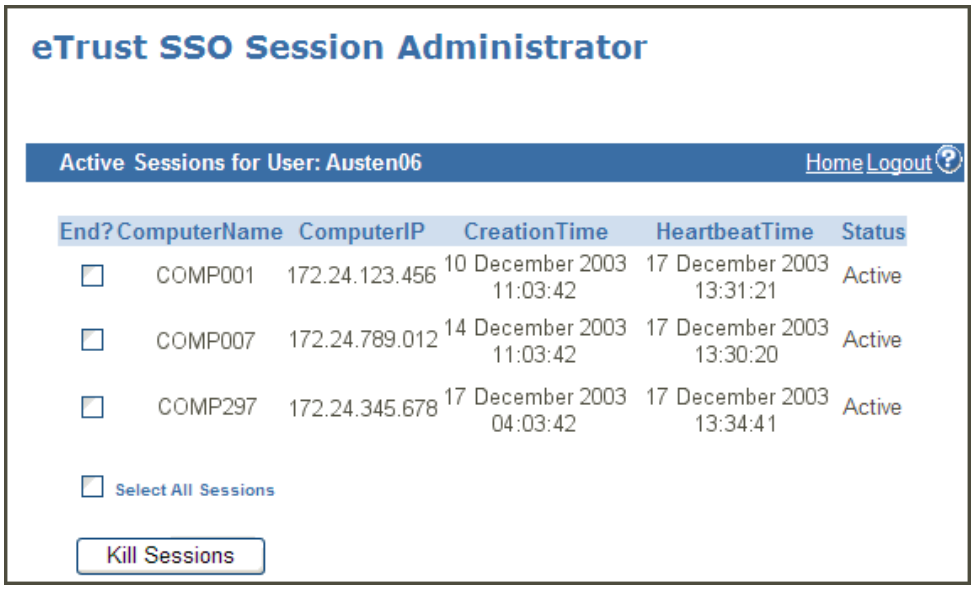


If you use the Session Administrator to end a user's active session, it only shuts down their eTrust SSO session(s).

To close all open applications, you should write a logoff script to use with Session Administrator.

View Sessions

When you view all sessions associated with a single user, you see a list of details about each session.



Session Detail	Description
ComputerName	What computer the active session is running on.
ComputerIP	The IP address of the computer that the session is running on.
CreationTime	The time that the user logged on to the SSO Client on that computer.
HeartbeatTime	The last time that the Policy Manager confirmed the session is active.
Status	Whether the session is active.

Chapter 9

Managing Services

This chapter describes some of the tasks and services for administering eTrust SSO.

Updating Users' Application Lists

This section explains how to keep users' application lists up to date.

Each time a user logs onto eTrust SSO they are presented with a list of applications that they can access via SSO. This application list is retrieved from an application list cache stored on the Policy 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 should be updated periodically. This is essential to avoid the users' application lists becoming out of date.

How to Update Users' Application Lists

There are two ways an administrator can update users' application lists cache:

- To update groups of users, you must run the psbgc utility. You should schedule this utility to run during non-peak times on the network.
- To update a single user, you must log onto the Policy Manager and open that individual's record and select Application List. Alternatively, you can run psbgc which also supports a single user update option

There is one way a user can update their own application list:

- To update their own application list, a user should select the Refresh button on the SSO Client Tools window. This operation will also update app list cache on server side.

How the Application Lists Cache Works

The application list cache is created by the Policy Server in the following way:

1. The SSO Client requests the application list from the Policy Server.
2. The Policy Server gets the application list information from the application list cache.

   If the application list cache does not exist, the Policy Server generates the cache.
3. The Policy Server sends the application list to the 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 Policy Server load and improves the SSO Client performance during peak times.

The utility should 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 eTrust SSO administrator should ensure that the Application List Background Calculation utility runs often enough 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

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.

How to Run the Application List Background Calculation (psbgc) Utility

This section explains how to run the psbgc utility and what commands you can use.

To run the psbgc utility

1. Log onto the Policy Server computer as an administrator and open a command line prompt.
2. Enter the psbgc command.

   For example, if you wanted to update all users within a particular LDAP container, you would enter the following command:

   ```
   psbgc -a admin_user -p admin_password -c LDAP_user_container
   ```

   For a full list of all options you can use in a psbgc command, see the following psbgc Utility Commands section.

psbgc Utility Commands

The utility can be run with the psbgc command, using the following options:

Option	Parameter	Description
-a[dministrator]	admin name	Specifies the administrator's user name. This user must have administrative rights to the Policy Server. When you install the Policy 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 and corresponding password.
-c[ontainer]	container name	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 will update all users under user data store's base container (base path).
-d[atastore]	data store name	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 will operate on all data stores
-g[roup]	group name	Specifies the name of the group of users for whom you want to calculate an application list.
-h[elp]		Provides help which explains all the psbgc options.
-i[ni]	path to configuration file	Specifies the path to psbgc.ini configuration file. By default this is stored in the bin directory on the Policy Server computer. If you are in the bin directory you do not need to specify where the psbgc.ini file is located.

Option	Parameter	Description
-p[assword]	Password	Specifies the administrator's password. This user must have administrative rights to the Policy Server. When you install the Policy 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 and corresponding password.
-r[ecursive]		Specifies that the psbgc utility should calculate application lists recursively for all users within a specified container. You must use this in conjunction with the –c option to specify which user container to search. This means that you can tell the psbgc utility to update all users' application lists within a hierarchy.
-u[ser]	User names	Specifies a single user for when you want to calculate an single user's application list.
-x		Specifies the number of users you want returned in pre-set portions. This is highly recommended if you have more than 200 users to update and you are using a data store that supports paging.

Example psbgc Commands

To calculate application lists for all users in all data stores, run this command:

```
psbgc –a ps-bgc –p password
```

To calculate application list for a single user, run this command:

```
psbgc –a ps-bgc –p password –d ps-ldap –c "ou=QA" –u "John Smith"
```

To calculate application list for all users under specified container in AD data store, run this command:

```
psbgc –a ps-bgc –p password –d AD –c "ou=QA"
```

psbgc.ini configuration file

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 Policy 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.

Keyname	Description	Values
ServerHost	Specifies the name of the machine (or IP address) where Policy Server is installed.	Define: 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).	Define: Computer name Default: [None]
AuthMethod	Specifies the authentication method used to authenticate the administrative user. Possible valued include: [SSO | LDAP | EAC]	Define: Abbreviated authentication method Default: SSO
NameAttribute	Specifies the name user name attribute for authentication.	Define: User name attribute Default: USERNAME
PasswordAttribute	Specifies the name of password attribute for authentication.	Define: User password attribute Default: PASSWORD
HostAttribute	Specifies the name of the auth attribute containing the name of the authentication host used for authentication.	Define: Authentication attribute Default: AUTHHOST
MsgFilePath	Specifies the path to the message file.	Define: Pathway Default: C:\Program Files\CA\eTrust Policy Server\Lang

Keyname	Description	Values
MsgFileName	Specifies the name of the message file. The message file contains error descriptions that are presented to the end user.	Define: File name Default: ENU.msg
KeyFileName	Specifies the name of the temporary file to cache the Policy Server's public keys used for encryption of the communication channel between psbgc and the Policy Server.	Define: 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 policy server.	Define: 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 the user names from the server to psbgc. You must also set the SendRecvSize parameter, in the Policy Server, to be the same value.	Define: File size in KB Default: 128KB

Managing Keys for Session Encryption

Key management is one of the essential tasks of eTrust SSO management.

Key pair generation for session encryption is performed with a utility named Sso_genkeypair supplied with eTrust 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 eTrust SSO is re-installed. Because key pair generation is a resource intensive process, it is not practical to run the process frequently.

eTrust SSO components are installed using a default public/secret key pair.

When a new key is set at the Policy Server, the server sends the new public key to the SSO Clients connected to it.

Running Sso_genkeypair

Run the Sso_genkeypair utility on the Policy Server host with the command:

```
Sso_genkeypair [-cfg pathname-to-ssod.ini]
```

The utility can be run at any time, but for the new keys to take effect you have to shut down ssod and then start it up again.

For a full list of Sso_genkeypair command-line options, see the *eTrust Access Control* documentation.

The Automatic Password Generation Utility

The Policy 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 eTrust SSO. This should reduce help desk calls from users who forget passwords and can improve the system administrators' tracking of application use.

Backing Up and Restoring a Data Store

Regular maintenance includes backing up your data stores regularly.

The following sections explain how to back up and restore an eTrust Access Control policy data store or user data store.

If you are using an external LDAP-type directory as your user data store (for example, eTrust Directory, or Active Directory from Microsoft), see the documentation that came with the product for instructions on backing up and restoring the directory.

If you are backing up a data store because you plan to move the data store to another machine, make sure you also back up and move the following:

- TCL scripts
- Message of the Day files

Backing Up an eTrust AC Data Store

Any standard backup method can be used to back up the policy data store (seosdb) or the user data store. For example, in UNIX, you could use ADSM or **copy seosdb** to a tar file.

You cannot perform incremental backups on the policy or user data store.

It is recommended that you stop eTrust Access Control and shut down its daemons before performing the backup. However, it is possible to perform the backup while eTrust Access Control is running.

Backing Up on Windows

To back up an eTrust Access Control data store that runs on a Windows computer, use the backup utility **acbackup.bat**, which is installed in the **Policyserver\bin** directory. Use the following syntax:

```
acbackup backup targetfilename
```

where **targetfilename** is the name of the text file to which the data store will be backed up.

This backup utility prompts you for information about the backup, including whether the backup is to be performed online or offline. Run an online backup if Access Control is still running. Run an offline backup is Access Control is not running.

Backing Up on UNIX

To back up an eTrust Access Control data store that runs on a UNIX computer, use the backup utility **sedb2scr**.

This utility outputs the eTrust Access Control database contents as selang commands. Use the following syntax:

```
./sedb2scr -r > targetfilename.txt
```

where **targetfilename** is the name of the text file to which the data store will be backed up.

Backing Up a Replicated Data Store

If you are performing maintenance on a Policy Server farm that has full replication (mirror imaging) implemented, the best backup method is to do the following:

1. Stop the Policy Server and the eTrust Access Control services on any one of the server hosts.
2. Back up the database of that Policy Server.
3. Restart eTrust Access Control and the Policy Server.

   This server's database will be updated with all of the changes that took place while it was down as long as one other Policy Server in the server farm remains running.

Restoring an eTrust AC Data Store

To restore a eTrust Access Control data store, the Access Control service must be running.

Use the backup utility **acbackup.bat**, which is installed in the **Policyserver\bin** directory. Use the following syntax:

```
acbackup restore sourcefilename
```

where **sourcefilename** is the name of the text file from which the data will be restored.

Starting and Stopping the eTrust AC Services and Daemons

The steps you take to start and stop the service associated with eTrust Access Control differ depending on whether you are running Windows or UNIX.

If You Are Running Windows

The eTrust Access Control services are:

- SeOS Agent
- SeOS Engine
- SeOS TD
- SeOS Watchdog

To start and stop these services, use the Services dialog, which can be opened from the Control Panel.

You can also use the command line to start and stop the services.:

1. Run the Command Prompt from the eTrust Access Control file location in the **\bin** directory.
2. Use the following commands to stop and start all of the services:

```
secons –s
seosd –start
```

If You Are Running UNIX

The eTrust Access Control daemons are:

- seagent
- seosd
- seoswd

To verify the status of the services for eTrust Access Control run either of the following commands.

```
ps –aef
ps –aef | grep –i seos
```

Use the additional operand to verify whether all of the eTrust Access Control daemons are running.

To start the services for eTrust Access Control, go to the eTrust Access Control file location in the **/bin** directory and run the **seosd** command. To stop the services, run the **secons –s** command from the same location.

Communication Between Components

Communication Protocols Between Components

eTrust SSO requires the following communication protocols:

- Between Policy Server and SSO Clients—TCP/IP
- Between SSO Clients and primary authentication agents—the protocol required by the authentication host
- Between SSO Clients and application agents—the protocol required by the application host
- Between SSO Clients and application hosts—the protocol required by the application host

Encrypting Communications Between Components

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.

Encryption Flow

1. The SSO Client generates a session key.
2. The SSO Client then initiates a connection with the Policy Server.
3. Once the connection is established, the SSO Client encrypts the session key with the Policy Server's public key, and sends the encrypted session key to the Policy Server.
4. The Policy Server decrypts the session key using the server's secret key.
5. The Policy 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 Policy Server sends its public key to the SSO Client, which prompts the user to accept it. If the user accepts it, the public key sent is cached and the SSO Client resumes the process at step 3 above. Otherwise, the connection is terminated.

Ports Used in eTrust SSO

eTrust SSO uses the following default ports:

Component	Protocols	Port Number	Description and configuration
Policy Server	TCP	13980	This is the TCP port where the Policy Server will listen. El Gamel/3DES This is configured: ▪ WIN: Windows Registry ▪ UNIX: policyserver.ini
	UDP	13990	
Policy Server Token Directory "pstd"	LDAP	13390 Yes	The LDAP communication port. This uses SSL. This is configured in the eTrust Directory configuration files + Policy Manager: Configuration Resources, Token Dir, PSTD, General
	Telnet	19390 r8 GA 13380 post r8 GA	This is the console port for debugging. This is configured in the eTrust Directory Config files. This uses SSL.
Policy Server User Directory "ps"	LDAP	13389	This is the LDAP communication port. This uses SSL. This is configured in: eTrust Directory config files+ Policy Manager: Data Stores, ser Data Stores, ps-ldap, General
	Telnet	19389 r8 GA 13379 post r8 GA	This is the console port for debugging. This uses SSL. This is configured in: eTrust Directory Config files.
eTrust Access Control	TCP protocol	8891	This is the TCP port where Access Control will listen. 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 SsoClnt.ini file.

Component	Protocols	Port Number	Description and configuration
Certificate Authentication Agent	TCP/IP	13987 Not on UNIX	This is the port for the TGA ticket granting agent. This uses SSL. This is configured in: ▪ Windows Registry on the Cert auth. agent ▪ HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\sso_tga_cert_Agent1\Parameters\sso_tga_cert_Agent1\PortNumber ▪ SsoClnt.ini file
Entrust Authentication Agent	TCP/IP	13987 Not on UNIX	This is the port for the TGA ticket granting agent. This does not use SSL. Configure in both of: ▪ Windows Registry on the Entrust auth. agent ▪ SsoClnt.ini file
LDAP Authentication Agent	TCP/IP	17979 Not on UNIX	This is the port for the TGA ticket granting agent. This does not use SSL. Configure in both of: ▪ Windows Registry on the Entrust auth. agent ▪ SsoClnt.ini file
RSA/SecurID Authentication Agent	TCP/IP	13969 Same for both UNIX and Windows	This is the port for the TGA ticket granting agent. This does not use SSL. Configure in both of: ▪ Windows Registry on the Entrust auth. agent OR ▪ UNIX install directory tga_rsa.ini and ▪ SsoClnt.ini file
Password Synchronization Agent (Mainframe)	CCI	1721	This is the communication port for the mainframe. This does not use SSL. This is configured in: CCI\CCI\CCIUSER in the file CCIRMCTD.RC

Component	Protocols	Port Number	Description and configuration
	TCP/IP	9123	This is the communication port for the mainframe. This does not use SSL.
One Time Password Agent (OTP) (UNIX)	TCP/IP	UNIX ONLY 13967	This is the port the OTP agent listens for messages from the Policy Server. This does not use SSL. This is configured in: ssotp.ini (install directory for OTP)

Chapter 10

Authenticating Users to eTrust SSO

This chapter discusses how eTrust SSO authenticates users. It then describes the different authentication methods that eTrust SSO can work with.

For information about authenticating to applications, see the "Authenticating Users to Applications" chapter.

About Authentication

Primary authentication is the process by which users prove their identity to eTrust SSO.

To do this, the user might provide a valid user name and password, or biometric information such as a fingerprint or retina print. After users have proven their identity to eTrust SSO, they receive their individual application lists and are entitled to get eTrust SSO services.

Usually, users only go through primary authentication the first time they use eTrust SSO. However, users are sometimes asked to re-authenticate (do primary authentication again):

- When they unlock StationLock
- When they log on to a sensitive application
- When the time since the last primary authentication exceeds a designated value

To log on to 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.

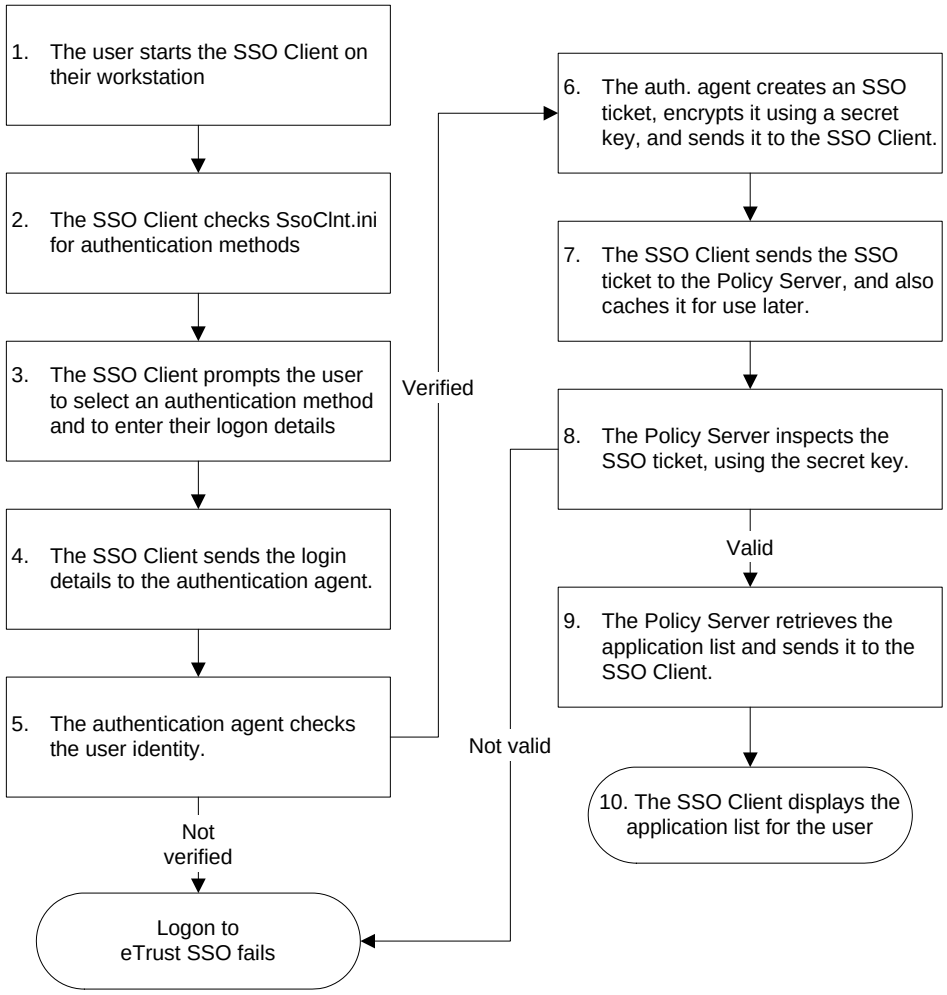
For more information about application authentication, see the 'Authenticating Users to Applications' chapter.

How Primary Authentication Works

The primary authentication process involves the Policy Server, the SSO Client, and the authentication agent. The following sections describe this process.

Flow Diagram of the Primary Authentication Process

The following flow diagram summarizes the eTrust authentication process. Each step in the diagram is described more fully in the following section, Description of the Primary Authentication Process.



Description of the Primary Authentication Process

This section gives you an overview of the primary authentication process. These steps correspond to the steps in the flow diagram in the previous section.

1. The user starts the SSO Client on their workstation.
2. The SSO Client checks the AuthMethods keyname in the ServerSet section of the SsoClnt.ini file.

   All authentication methods listed in this section will be available to the user. The first in the list will be displayed as the default.
3. The SSO Client opens the logon dialog, which prompts the user for the following:
   - Select an authentication method (the list of methods is taken from the AuthMethods keyname in the SsoClnt.ini file)
   - Enter credentials, such as a user name and password, biometric information, or a smart card.
4. The SSO Client sends the user's logon details and authentication method to the eTrust SSO authentication agent on the authentication host.
5. The authentication agent verifies that the credentials used to log on correspond to a valid user on the authentication host.
6. If the verification is successful, the authentication agent creates an SSO ticket, encrypts it using a secret key, and sends it to the 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 SSO Client does two things with the SSO ticket:
   - The SSO Client caches the SSO ticket. Later, it uses the same ticket in the application logon process.
   - The SSO Client sends the SSO ticket to the Policy Server.
8. The Policy Server does the following with the SSO ticket:
   a. Gets the secret key of the authentication host from its record in the AUTHHOST class and decrypts the ticket
   b. Checks that the authentication method is valid for the user
   c. Compares the details, such as name and serial number, of the authentication host kept in the AUTHHOST record with the details in the ticket
   d. Verifies that the current user is authorized to use the authentication host that originated the ticket
   e. Compares the name of the user with the name extracted from the ticket
   f Checks that the ticket has not expired

If any of the checks fails, the ticket is rejected by the Policy Server.

9. If the ticket is valid, the Policy Server retrieves from the user data store the list of the applications that the user is authorized to use, and sends the list to the SSO Client.
10. The SSO Client displays the list of applications. The user can now start work.

Primary Authentication Components

Primary authentication involves several components that should all be installed and communicating with each other.

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 application list

The Authentication Host

The authentication host is a computer on which the authentication software resides. The authentication software is the component that verifies the user credentials.

When a Policy Server is used for primary authentication, it is referred to as the authentication host.

The authentication host can be a general-purpose server running NetWare, UNIX, or Windows OS, or a specialized server such as RSA ACE, SAFLINK, or Entrust.

Each authentication host on which an eTrust SSO primary authentication agent is installed must be defined in the data stores with a record in the AUTHHOST class. This record contains details that identify the host and the key used for encryption.

The administrator must create an entry in the AUTHHOST class for each authentication host. The name of the AUTHHOST record in the data store must match the name of the host as it is known by native operating system. The DNS name by which the host is known is not relevant.

If required, an authentication host may belong to one or more authentication host groups (class GAUTHHOST). The administrator can use authentication host groups to specify the access rules for users to groups of hosts, rather than having to set up separate rules for each host.

For the Policy Server to accept the SSO ticket provided by the user, the user must be authorized to access the host that generated the ticket.

This authorization is the normal accessor-resource relationship that is defined using access controls (ACLs) in the data stores. The authorization might be a direct link between the user and the host, or it might be an indirect link via one of the user groups of which the user is a member.

The Authentication Agent

In order to provide maximum operational flexibility, a separate authentication agent handles the interactions between the SSO Client and the third-party authentication server.

Each of the authentication agents is a bridge for communication between the SSO Client and an authentication server.

eTrust SSO includes ready-made agents for various authentication systems.

The SSO Ticket

The SSO ticket is an encrypted string containing the information needed for authenticating the user to the Policy Server. This information includes:

- User identification (such as user name and password, or biometric information)
- Authentication method
- Time stamp

eTrust SSO limits the lifetime of the SSO ticket. When the ticket expires, the SSO Client has to request that the user re-authenticate (in effect, the user has to carry out the same primary authentication again).

How the SSO Ticket Is Used

The SSO ticket is used to carry authentication credentials between components of the eTrust SSO system.

An eTrust SSO ticket is created by the component that authenticates the user. This can be an authentication agent or the Policy Server.

When the SSO Client starts, it requests authentication from its designated primary authentication component (such as the Windows NT agent). The SSO primary authentication component verifies the credentials that the SSO Client provides, and if they are valid sends an SSO ticket to the SSO Client. The SSO Client then sends this ticket to the Policy Server as a proof that the user has been authenticated.

After the primary authentication component creates an SSO ticket, it sends the ticket to the SSO Client and the client caches it. An SSO Ticket is valid for a predetermined period of time set by the token TicketExpiration. This is defined on the Policy Server (in policyserver.ini on UNIX and in the registry on Windows).

When the user attempts to log on to an application, the SSO Client sends the ticket it has cached to the Policy Server. If the ticket has not expired, the server provides the service. If the ticket has expired, the server informs the SSO Client that the ticket is not valid and tells it to prepare a new ticket by re-authenticating the user. This means that the user goes through primary authentication again.

Encrypting the SSO Ticket

The ticket is encrypted using a secret key that both the primary authentication agent and the Policy Server know.

It is recommended that each authentication host have a different key.

The Policy 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.

If no key was specified for an authentication host, the server and the host use a default key. While the default key can be used for setup and testing, it should be changed for every authentication host, when beginning to work in a production environment.

The key value on the authentication host should 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.

For information about changing keys, see the section Managing eTrust SSO Keys in the 'Managing Services' chapter.

The Application List

Depending on the workstation operating system and desktop configuration, the list can be displayed as application icons in a floating toolbar, as a program group, or as a part of the Start menu.

After primary authentication has been carried out, the SSO Client requests an application list. To build the application list, the Policy Server uses the authorizations in the data stores. There are two options for handling 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.

For a description of how application list caching functions, see the section The Application List Background Calculation Utility in the 'Managing Services' chapter.

Primary Authentication Methods

Each eTrust SSO user is associated with one or more primary authentication methods. The primary authentication method determines which primary authentication method that the Policy Server accepts for that user.

Within the one eTrust 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.

eTrust 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 eTrust SSO	The Policy Server (serving as an authentication agent in addition to its other functions)
Novell	A primary authentication agent that resides on an operating-system server (such as a Windows NT server)
NT	
LDAP	A primary authentication agent that resides on a security server (such as a SecurID, SAFLINK, or Entrust server)
Entrust	
RSA SecurID	
Safeword	
SAFLINK	

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.

For more information about the AuthenticationMethod property, see the chapter 'Working with the User Data Store'.

Authenticating with Native eTrust SSO

If you use the native SSO authentication method, the Policy Server acts as the primary authentication host and checks the user ID and password against a password stored in the data stores for primary authentication.

All the components needed for eTrust SSO native authentication are installed automatically with the Policy Server software.

eTrust SSO native authentication is completely independent from the user's network sign-on.

When the SSO Client starts, it checks which primary authentication method should be used. If its AuthMethod token is set to SSO, the SSO Client displays to the user a standard SSO dialog asking for user ID and password. The SSO Client encrypts the user ID and password and sends them to the Policy Server. The Policy Server checks the user record in the data stores and if it finds the password in the SSO logon info, it returns an SSO ticket to the SSO Client.

The SSO Client caches the SSO ticket and uses it whenever it carries out application logon.

In order to use SSO native authentication, the user has to be both defined as using SSO native authentication and allowed to get services from the Policy Server (that is, the Policy Server also acts as the authentication host).

Note: SSO Authentication does not support hierarchical name spaces.

Authenticating with the Operating System Logon

Normally, users must log on through a primary operating system to get to a working environment. The primary logon can be through Windows or Novell.

eTrust SSO primary authentication can be integrated with the operating system logon, so that the end user performs only one logon. This logon carries out primary authentication and then logs the user onto the user's primary network operating system. This integrated authentication can also support third-party verification methods.

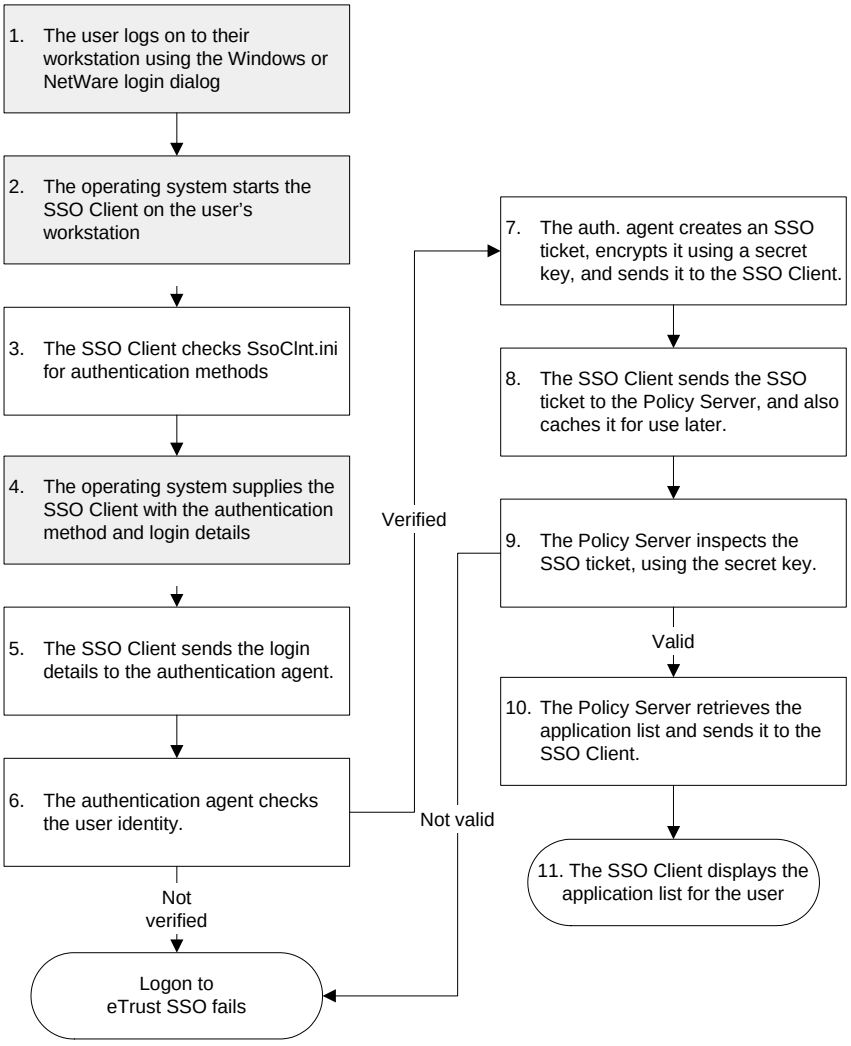
If you choose to use OS primary authentication, users do not have to change their work habits. Users log on exactly as before, and this is the only logon that is required of them. Using the information and authorization from their primary logon, eTrust SSO will log on users to other environments.

Since these operating systems do not support ticket mechanisms, eTrust SSO uses authentication agents to supplement their authentication. The authentication agent is installed on the authentication host and provides the Policy Server with an SSO ticket as proof that the specified user logged on to the required host.

Flowchart of the Operating System Authentication Process

This section describes the general flow of OS primary authentication, without regard to the differences in the OS platforms (NetWare and WinNT). For platform-specific details, see the section on each particular primary authentication method in the Implementation Guide.

The shaded boxes show the steps that are different from the standard authentication process described in the How Primary Authentication Works section.



Windows Authentication

The authentication host 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 SsoClnt.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.

Novell NetWare Authentication

The SSO Client communicates with the eTrust SSO Novell NetWare agent via NetWare's NCP Extensions. This mechanism lets the agent provide services to NetWare clients using their existing connection to the NetWare server. The agent is actually extending the services provided by native NetWare. The agent queries the NetWare operating system to find out the name of the user who is logged into the connection from which the authentication request arrived.

Once the NetWare server has been defined to eTrust SSO as an object in the AUTHHOST class, the NetWare server can function as an authentication host.

Authenticating with Third-Party Software

To allow eTrust SSO to work with a third-party authentication host, you need to install an authentication agent for that server.

The user's sign-on is taken to a security server, such as RSA SecurID, Entrust, or SAFLINK.

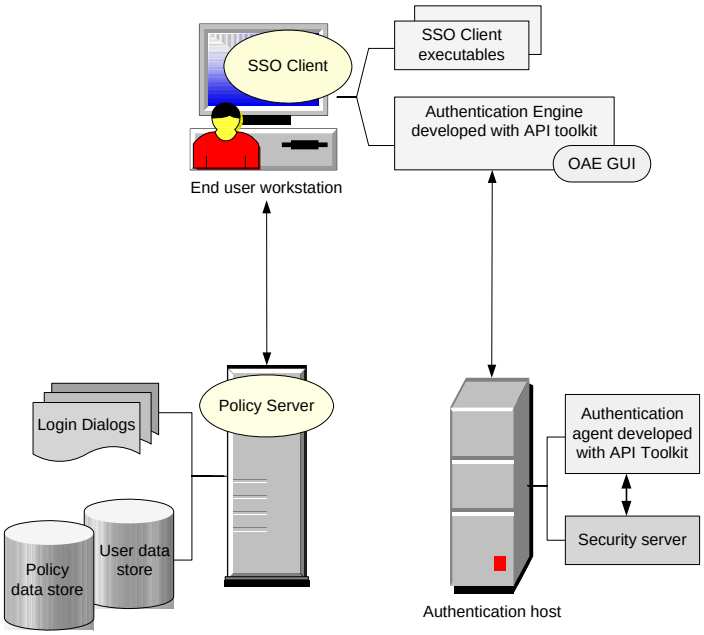
For more information, see the 'Implementing Authentication Agents' chapter in the Implementation Guide.

Integrating with Other Third-Party Authentication Methods

This integrates an initial primary authentication to eTrust SSO with a subsequent logon to the operating system and can also involve third-party verification processes. It is facilitated by supplied agents or developed with the use of the Open Authentication API Toolkit.

The Open Authentication Toolkit of eTrust SSO provides APIs and code modules that allow administrators, integrators, and authentication vendors to develop eTrust SSO integration to meet specific site needs and services on their own.

The following diagram shows a schematic view of primary authentication with an authentication agent developed with the Open Authentication Toolkit:



The Open Authentication Toolkit lets developers build authentication engines (AEs) that communicate between the SSO Client and an authentication agent or a device such as a fingerprint or card reader. These AEs can be added on-site by simply plugging in (file copying) DLLs.

The authentication architecture defines AEs that do not include platform-dependent code. This makes it easier to deploy the same AE in a number of different environments (UNIX, Windows, Mac, and web).

The Open Authentication Toolkit also provides it own GUI called Open Authentication Engine (OAE) GUI. AE standards require this GUI instead of user-interface specific code to allow easy deployment in various environments, giving identical functionality and a similar appearance. Interfaces supported include Windows native look-and-feel, Visual Basic, Java-AWT, UNIX Motif, Tcl/Tk, Macintosh look-and-feel, HTML, and others. This level of independence is achieved by having each AE define its relationship with the OAE GUI.

The toolkit provides several exit points (hooks) for site customizing. This enables administrators and integrators to set up IPC (Inter-Process Communication) between their in-house applications and the SSO Client infrastructure.

Please contact your CA representative if you a looking to develop your own primary authentication mechanism using the Open Authentication Toolkit. A sample authentication agent is available from CA eTrust SSO Development, which can be used as a tutorial to help you develop your own authentication agent.

Chapter 11

Launching Applications with eTrust SSO

This chapter describes how the eTrust SSO system launches eTrust SSO-supported applications and inserts the user's credentials. It specifically talks about logon scripts, which launch the applications and simulate the users actions, and application authentication, which is how users are identified to the application.

Here is a summary of what happens when a user selects an eTrust SSO-supported application.

Step 1

The user selects an application from their list of eTrust SSO-supported applications on their computer and the SSO Client sends the user authentication (the SSO ticket created in the primary authentication process) and the application identifier to the Policy Server.

Step 2

The Policy Server retrieves the appropriate logon script and the relevant logon variables (user ID and password or ticket), and then sends them back to the SSO Client.

Step 3

The SSO Client then runs the logon script, which launches the application and plugs in the logon variable when required. The logon script has two main responsibilities; one to start up the application, and two to enter the logon variables (user ID and password or ticket) as requested by the application's logon process.

Logon Scripts

The actual logon process is carried out by a logon script. These logon scripts are written in an extended version of Tcl, a scripting language that gives you the use of variables, conditions, loops, procedures, and other common programming devices with a minimum of complexity. A logon script starts up an application, responds to the application's prompts for user ID and password, and handles related logon tasks. It should notify the Policy Server of the results of the logon attempt.

Following is an example of the main portion of a logon script for a telnet client that comes with Windows NT:

```
# run the NT telnet client
sso run -path telnet.exe

# connect to the remote host
sso menu -item "Connect/Remote System"
sso setfield -label "Host Name" -value $_HOST
sso click -label Connect

# verify that the telnet window appears
sso window -title Telnet

# wait for the user ID; respond
sso waittext -text "logon:"
sso type -text "$_LOGINNAME{enter}"

# wait for the password prompt; respond
sso waittext -text "password:"
sso type -text "$_PASSWORD{enter}"

# wait for the system prompt
sso waittext -text ">"

...
```

The logon variables that appear in this logon script are $_HOST, $_LOGINNAME, and $_PASSWORD. The SSO Client on the user's workstation replaces these variables with the values received from the Policy Server.

Symbol	Meaning
$	Tcl variables
$_	SSO logon variables
#	Comment

For a full explanation of logon scripts, see the eTrust SSO *Tcl Scripting Reference Guide*.

Logon Variables

The logon variables include the logon script and the logon data sent to the SSO Client. These variables are fetched from the 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 in the LDAP or eTrust Access Control data store in the user's record as properties of the LOGONINFO section. 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
   - LOGON_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 LOGONNAME contains the value UTST021.

2. The Policy Server generates an AppTicket and stores the result in the Tcl variable _PASSWORD.
3. The Policy Server places the logon name UTST021 in the Tcl variable _LOGONNAME.
4. The server sends the CICS.TCL logon script and the two logon variables _PASSWORD, _LOGONNAME, and _HOST to the SSO Client.
5. The SSO Client executes the supplied script, entering the username (_LOGONNAME) and ticket (__PASSWORD) as required.

Learn Mode (First Logon Situation)

In order to reduce the amount of configuration needed, eTrust SSO has a *learn mode* that functions during the first logon to an application and lets the user provide the logon credentials for the application.

If the user credentials needed for an application are not found in the user record and the application logon uses password authentication, the Policy Server and SSO Client assume that this is the first time the user is logging into the application via eTrust SSO. eTrust SSO then enters learn mode (also called the *first logon situation*), as follows:

1. The Policy Server notifies the SSO Client that no credentials are available.
2. The SSO Client displays a Learn Mode dialog box that prompts the user for user credentials (logon name and password for the application requested).
3. After the user supplies the user credentials, the client sends the credentials to the server and the client repeats the logon process with the new logon credentials.

Note: Learn mode only functions for users who are authorized to use an application and who have carried out primary authentication.

Application Authentication

All application logons supported by eTrust 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

eTrust SSO offers two different methods of application authentication:

- Password authentication which can be used for applications on any platform (Windows, UNIX or Mainframe)
- Ticket authentication which is only used for Mainframe applications. Ticket authentication can be broken down into two subsections:

  - PassTickets

  - AppTickets

The application authentication method used for an SSO-supported application is specified in the LOGON_TYPE property of the application's record in the eTrust Access Control data store. If a value for the LOGON_TYPE property is not specified, the default method used is password.

Each SSO application record in the Access Control data store can have only one application authentication method associated with it (the value of the LOGON_TYPE property: pwd, AppTicket). However, this does not mean that all users working with eTrust SSO have to use the same method for a particular application. In the eTrust Access Control data store, you can define several records that represent the same application. Each of the records can be defined with a different application authentication method. Each user can be authorized to use the application with the appropriate authentication method. This method may also be used to provide different password policies to different groups of users for a given application.

The administrator can change the authentication method of an application at any time.

Application Authentication for Windows and UNIX

The most common way to implement SSO for an application is to use password-based authentication. When using the password authentication method, eTrust SSO supplies the application with a user ID and a password.

Logging in Using Passwords

The following steps describe the process of logging in using passwords:

1. The Policy Server gets logon variables from the LOGONINFO 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 LOGONINFO section that relates to the master application.

   Logon variables include the current password (CURRPWD) 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 nest password is non-null it indicates that the script should change the password after logging in to the application.

2. The Policy Server checks whether the user password has expired (an event controlled by eTrust SSO, not by the application itself). If the User Password Never Expires feature is enabled, then the Policy Server skips this check. If the password has expired, the Policy Server sends a message to the SSO Client that the password has expired; the SSO Client then prompts the user for a new password.

   If the password auto-gen feature is enabled, the Policy Server generates a new password and puts it in the NEXTPWD property in the database, and the process continues.

   For more information about the User Password Never Expires feature, see Managing User Passwords in the chapter "Administering eTrust SSO Users and Resources."

3. If the current password has not expired, the Policy Server sends the SSO Client the logon variables and the appropriate logon script.

4. When the SSO Client receives a logon script and logon variables from the Policy Server, it plugs the variables into the logon script and then begins to execute the logon script.

5. A properly written logon script for a password-authenticated application checks if a new password was sent by the server (that is, whether the SSO Client received a non-null value for the NEXTPWD variable). If a new password is received, then a properly written logon script changes the password during the logon process, in the way required by the application.

6. A properly written logon script also notifies the Policy Server about the password change after it is carried out. The Policy Server moves the value of the new password (NEXTPWD) to the current password field (CURRPWD), clears the NEXTPWD property, and updates the PWDCHANGE field value with the time that the password was changed.

If a new password is provided by the user, the SSO Client sends it to the Policy 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 LOGONINFO as the next password (NEXTPWD). If the new password does not meet the password policy's requirements, the password is rejected by the Policy Server and the user is prompted to supply a new password.

Application Authentication 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 therefore 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 LOGONINFO section of user records.

Two Methods of Ticket-Based Application

eTrust SSO offers built-in support for two ticket formats that work with OS/390 (MVS) mainframe applications and common OS/390 external security packages:

- The AppTicket ticket format, which is a proprietary eTrust SSO format. AppTickets are compatible with OS/390 Security Server (RACF) from v1.9.2 and CA-TOP SECRET from v4.4.
- The PassTicket format, which uses a ticket algorithm created by IBM. eTrust SSO support for PassTickets is compatible with all versions of external security packages that themselves support PassTickets: RACF (OS/390 Security Server) from v1.9.2, CA-TOP SECRET from v4.4, and CA-ACF2 from v6.2.

Ticket-based application authentication is recommended wherever one of the following situations exists:

- There is a way for eTrust SSO to intervene in the password validation process (like the pre-logon exit of RACF)
- The target system supports PassTickets

In addition to eTrust SSO's built-in solutions for OS/390 (MVS), 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.

Differences Between AppTicket and PassTicket Authentication

The main differences between AppTicket-based authentication and PassTicket-based authentication are summarized in the following table:

Feature	AppTicket	PassTicket
Support for applications	All OS/390 (MVS) applications that perform a standard authentication to the mainframe external security package.	Only TSO, IMS, CICS, and APPC applications.
Support for security packages	RACF, TSS.	RACF, TSS, CA-ACF2.
Control of ticket validation parameters	Increased control; for example, the administrator can change the expiration period of AppTickets.	Attributes fixed by the IBM algorithm and not modifiable by the system administrator.
Key handling	Allows a hierarchy of keys for different security requirements: ▪ Host Key ▪ Site Key ▪ Initial Key	Allows only one key level (Application Key).
Installation	Installation of AppTicket support customizes a security package exit and puts an SSO agent on the mainframe.	PassTicket support is built into eTrust SSO and no changes need to be make to the mainframe or to the security package.

AppTicket Authentication

eTrust SSO's AppTicket is a proprietary ticket generated and encrypted by the Policy Server and used with a Kerberos-like implementation of tickets.

It is a one-time-only password replacement for an OS/390 (MVS) external security package: RACF (OS/390 Security Server) or TSS (CA-Top SECRET). It can be used with a mainframe running OS/390 or MVS from version 4.3. The use of AppTickets does not prevent the use of the usual passwords of the security package.

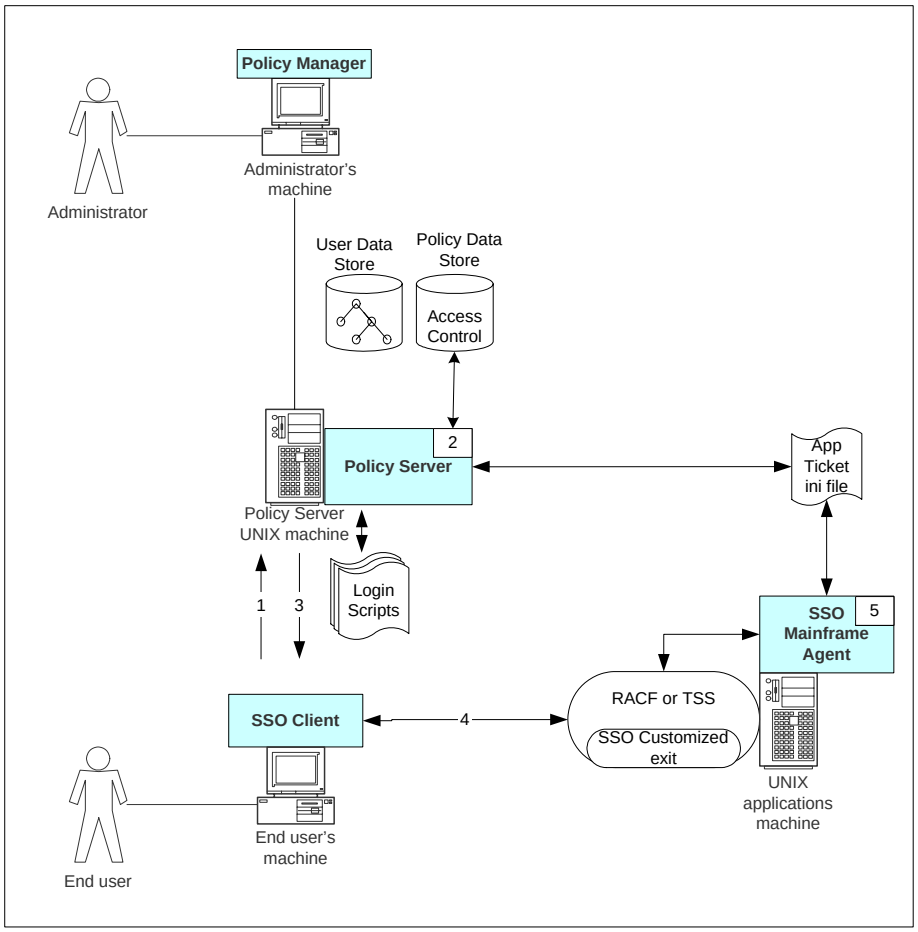
When eTrust SSO is set up to use AppTickets (generally during installation), two SSO components are installed on the mainframe:

- An SSO routine is added to the pre-logon exit of the external security package (RACF or TSS). This is the *SSO-customized exit* or the *SSO exit* (exits are provided by RACF and TSS to allow the enhancement of their normal processing). The SSO-customized exit allows the use of AppTickets for all OS/390 (MVS) applications that perform a standard authentication to the security package.
- A separate SSO process is installed as a started task (STC). This is the *SSO MF Agent* or the *SSO Agent*. Its functions include:
  - Providing the SSO exit with data from the mainframe ini file
  - Handling operator commands for controlling the SSO AppTicket processing, sent from the operator's console

AppTicket implementations can also be developed to work in other computing environments.

Logging in Using an AppTicket

The following diagram illustrates application authentication using an AppTicket:



1. The user selects an application and the SSO Client sends user authentication and the application identifier to the Policy Server.
2. If the request is valid and the application authentication method is AppTicket, the Policy Server generates and encrypts an AppTicket from data that includes user credentials, application identifier, encryption key, and timestamp.

   System clocks on SSO Client workstations do not have to be synchronized with either the Policy Server host or with the mainframe application host, since system time on the SSO Client workstation is not used in generating or checking the AppTicket.
3. The Policy Server adds the AppTicket as a _PASSWORD value to its response packet and sends the response packet to the SSO Client.

4. The SSO Client runs the logon script, which sends the user ID and the AppTicket to the mainframe application host as if the AppTicket were a regular password.
5. The mainframe security package (RACF or TSS), using the exit customized by eTrust SSO, checks whether the password is a valid AppTicket generated by eTrust SSO. One of the following actions occur:
   - If the password is a valid AppTicket and the user is authorized to log on through SSO to the mainframe application, the security package allows the logon. If it is a valid AppTicket, but the user is not authorized, logon is denied.
   - If the password is not a valid AppTicket, normal RACF or TSS processing continues to determine if it is a valid password; if it is not a valid password, logon is denied, as would be the case with any invalid password.

AppTicket authentication requires the following:

- Active SSO-customized exit
- Active SSO MF Agent
- Matching keys and other AppTicket tokens in Policy Server and mainframe ini files
- Synchronization of the system clocks on the Policy Servers, on the SSO backup servers, and on the mainframes where the AppTickets are checked.

Note: The SSO-customized exit only verifies the contents of the password field, that is, the AppTicket. It does not take any part in the normal evaluation of other logon information by the security package. For example, if the user is not allowed to log on at this time, the security package prevents the logon, even if a valid AppTicket was provided to the exit.

AppTicket Parameters

AppTicket parameters are stored in matching AppTicket ini files on Policy Server hosts and on mainframe application hosts. The files contain tokens for permitted time differences, for operating features, and for encryption keys. Because these files contain keys, they must be protected against any unauthorized read access.

On the Policy Server, the AppTktFile token in ssod.ini gives the full path to the AppTicket key file, whose default name is appticket.key.

For full details, see The AppTicket .ini File in the appendix “Initialization Files.”

Clock Synchronization

Two tokens are used by the SSO exit to control the latitude that eTrust SSO allows for the difference between the timestamp of the AppTicket and the time on the mainframe system clock:

- AppTicketExpPeriod specifies the maximum positive time difference allowed (that is, by how much can the AppTicket timestamp be earlier than the mainframe system clock). This is the AppTicket expiration period. The AppTicketExpPeriod value has to be specified in the ini file on the Policy Server as well, because the Policy Server uses the value in generating the AppTicket.
- MaxNegativeTimeDiff specifies the maximum negative time difference allowed (that is, by how much can the AppTicket timestamp be earlier than the time of the mainframe system clock).

The following table demonstrates how these values are used:

AppTicket Timestamp	Mainframe System Clock	AppTicketExpPeriod	MaxNegativeTimeDiff	AppTicket Valid or Invalid
12:00	12:00	0h15 (15 min)	0h5 (5 min)	Valid
11:50	12:00	0h15 (15 min)	0h5 (5 min)	Valid
11:40	12:00	0h15 (15 min)	0h5 (5 min)	Invalid
12:03	12:00	0h15 (15 min)	0h5 (5 min)	Valid
12:08	12:00	0h15 (15 min)	0h5 (5 min)	Invalid

A timestamp includes the date, as well as the time. The above illustration assumes that the date value is identical in the AppTicket and on the mainframe system clock.

If the timestamp on the AppTicket is outside the leeway specified in the mainframe initialization file, then the mainframe returns the normal “Wrong Password” error message, without specifying any more details.

Key Management

The AppTicket initialization files contain two kinds of keys:

- The Site Global Key—Which should be found in the AppTicket ini files on all the Policy Servers and mainframe hosts, and must have an identical value in all the files.
- Host keys—Each specific host key must have the same value in that host's AppTicket ini file and in the AppTicket ini file on the Policy Server. The Policy Server's AppTicket ini file has to contain all the host AppTicket keys, but each mainframe host can keep only its own host AppTicket key.

The Policy Server searches for a global site key if it can't find the required host key (for example, if there is no token or the host key token has no value). If the token has a value, but the value is not the appropriate one, the AppTicket being checked is treated as an invalid AppTicket, as explained in this chapter.

The order of search for keys is:

host key → GlobalSiteKey → Initial Key

Initial Key is a key that is coded into the eTrust SSO package and is intended for installation and debugging purposes only.

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 an OS/390 (MVS) external security package: RACF (OS/390 Security Server), TSS (CA-TOP SECRET), or CA-ACF2.

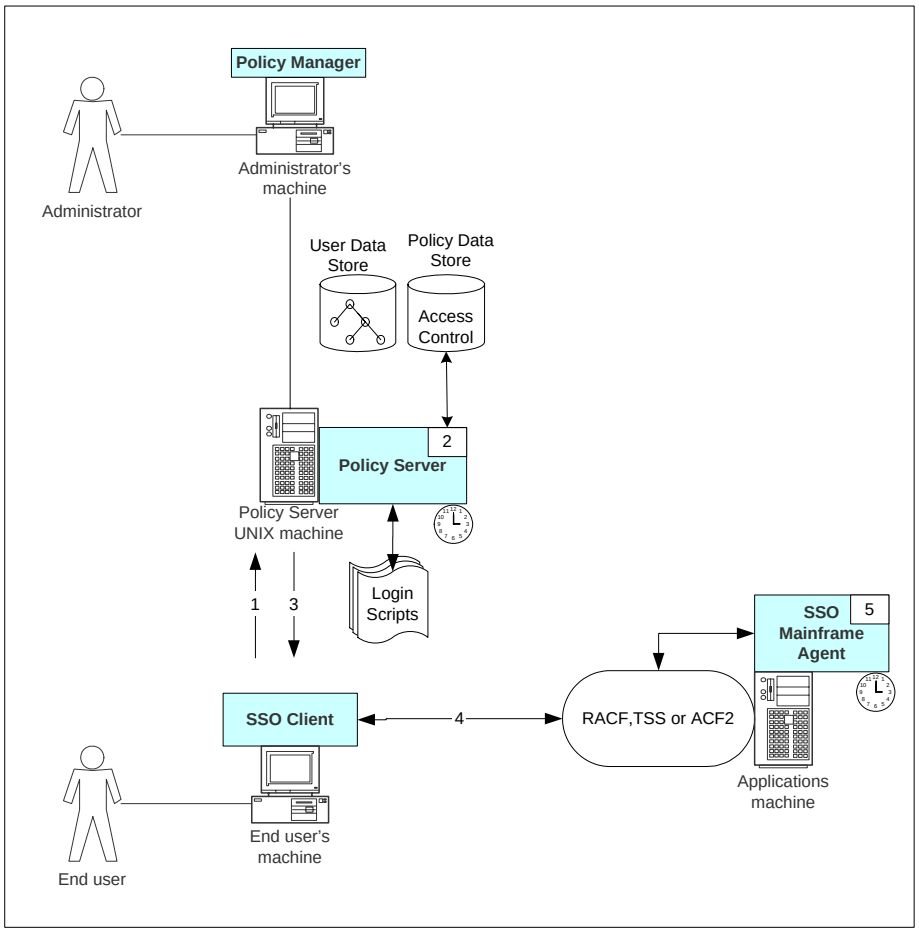
eTrust SSO support for PassTickets is compatible with all the versions of external security packages that themselves support PassTickets: RACF from v1.9.2, TSS from v5.0, and CA-ACF2 from v6.2.

eTrust SSO's implementation of PassTicket application authentication does not involve any changes to applications or to the mainframe security package.

In eTrust SSO's implementation of PassTicket authentication, the Policy Server generates PassTickets according to the IBM algorithm. This support is built into eTrust SSO and requires no installation or configuration on the Policy 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 eTrust 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 OS/390 (MVS) 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 Policy Server.
2. If the request is valid and the authentication method is PassTicket, the Policy 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 Policy Server and on the mainframe application host must be synchronized.

   However, system clocks on SSO Client workstations do not have to be synchronized with either the Policy Server host or the mainframe application host, since system time on the SSO Client workstation is not used in generating or checking the PassTicket.
3. The Policy 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: eTrust 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.

Sensitive Applications

When the SSO administrator marks an application as *sensitive*, the user is forced to reauthenticate more frequently to use the sensitive application. For a normal application, the Policy Server checks the normal expiration time of the SSO ticket (default expiration time is 8 hours). For a sensitive application, the Policy 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 in order 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 NetWare, and the telnet application is marked as sensitive, the user will be asked to provide the NetWare password when selecting the telnet application. When he enters it, the SSO Client runs the primary authentication process 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 by the SensitiveExpiration setting in the **sso** section of the ssod.ini file/registry.

Each application is represented by a record in the APPL class. A complete description of all the properties is given in the eTrust SSO *Tcl Command Reference Guide.*

Chapter 12

Authenticating Users to Web Applications

eTrust SSO can work with web applications in the same way that it works with other applications.

You can use the 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 Policy Server to authenticate the user and determine if access to the specific resource should 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.

For information, see the 'Adding Web Applications to SSO' chapter in the *Implementation Guide*.

Supported Authentication Methods

When you installed the Policy Server, you chose the authentication methods that you wanted to support. The authentication methods that can be selected when installing the Policy Server include:

- LDAP
- NT
- SDI
- X509
- SSO (proprietary password-based authentication)
- NTLM

The Web Agent can support all of the authentication methods supported by your Policy 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 eTrust SSO 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 eTrust 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 eTrust SSO system.

Client Logon

This logon method launches web applications in the same way as any other eTrust 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 SSO Client installed on every user's computer
- 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 SSO Client installation on each client machine, the htmlext.dll file contains a set of HTML extensions for the desktop SSO Client to provide script functions to help with writing the scripts for this function.

Example

You want to allow users to log on to the Hotmail web site using eTrust SSO. To do this you use the client logon method because you cannot install the eTrust SSO Web Agent on the Hotmail server, and because Hotmail has a simple repeatable logon process for which you can easily write a Tcl script.

For more information about the client logon method of using eTrust SSO with web applications, see the 'Authenticating Users to Applications' chapter.

Cookie Logon

The cookie logon method requires the 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 SSO Client installed on every user's computer
- The SSO Client configured to create a cookie when the user authenticates
- The eTrust 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 eTrust SSO desktop authentication.

To configure the client to create a cookie during authentication, you must modify SsoClnt.ini and specify the web servers URL in the DomainNameServer keyname. For example, if you were running a web server on a machine on zz.com, you might set the token as DomainNameServer=http://www.zz.com

Example

You want to log on to a restricted area on your Intranet using eTrust SSO. To do this you use the cookie logon method for two reasons. First, you can install the eTrust SSO web server on the intranet server. Second, you can reuse the eTrust SSO ticket that was created on your machine during the primary authentication to create a cookie.

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 SSO Client installed on users' computers, and no Tcl scripting is required.

To use this method you must have:

- The eTrust 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

The browser logon method uses the following process:

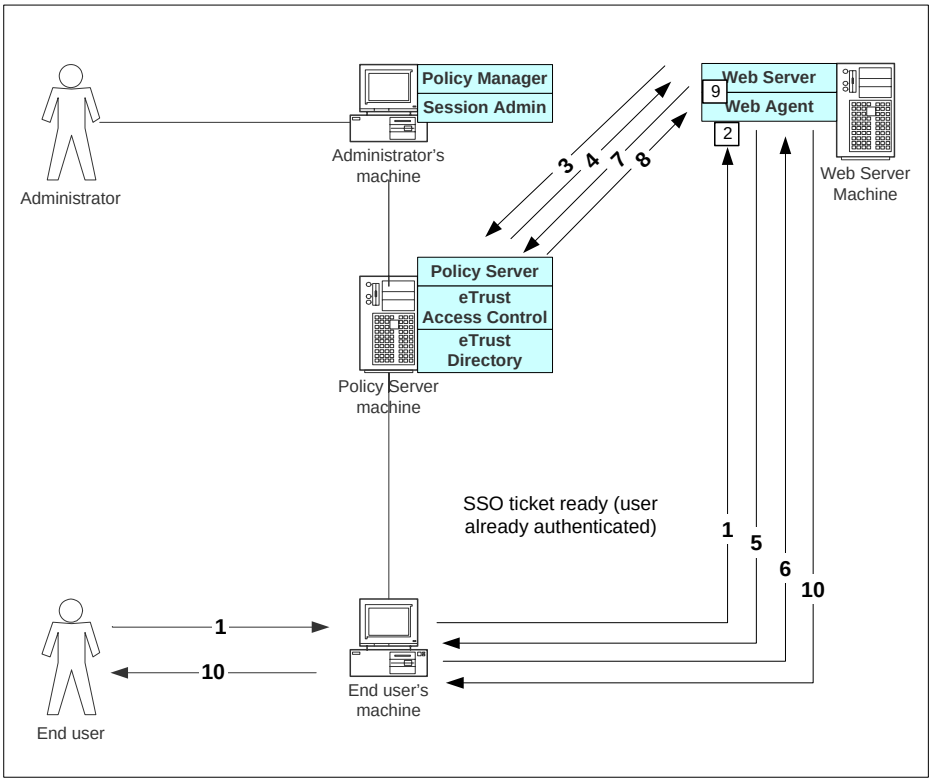
1. The user requests a web application through their web browser.
2. The Web Agent intercepts the request to access the web application
3. The Web Agent connects to the Policy Server to check if the web application has access rules.
4. The Policy Server sends a message to the Web Agent specifying any 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 Policy Server to authenticate the user.
6. The web agent sends the user's authentication credentials to the Policy Server to check whether this user has rights to access the web application.
7. The Policy 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

13

Working with the SSO Client

This chapter discussed different ways to customize and use the SSO Client. This chapter covers:

- SSO GINA functionality
- Workstation Modes, including Unlock Workstation/Shared Workstation
- Workstation Locking
- Automatic Application List Refresh

SSO GINA

The GINA performs all user identification and authentication interactions to the Windows operating system. GINA stands for Graphical Identification and Authentication DLL. The GINA has four dialogs that user will see according to their actions and the state of the workstation: Welcome, Authentication, Security, and Locked.

The GINA is a DLL component that is loaded by Winlogon. The Microsoft GINA (msgina.dll) can be replaced with the eTrust SSO GINA (ssogina.dll).

Some reasons you might want to replace the Windows GINA with the SSO GINA are:

- One-step authentication onto both the workstation and SSO
- A choice of all SSO-supported authentication methods for Windows logon
- Setting up Shared Workstation mode, if required (for more details about setting up Share Workstations, see the Workstation Modes section of this chapter).

If you want to use the SSO GINA, you must install it with the SSO Client. When the SSO Client is installed with the SSO GINA selected, it replaces the MS GINA.

If the SSO Client is uninstalled, the SSO GINA is uninstalled, and the Microsoft GINA is reinstated.

System Requirements

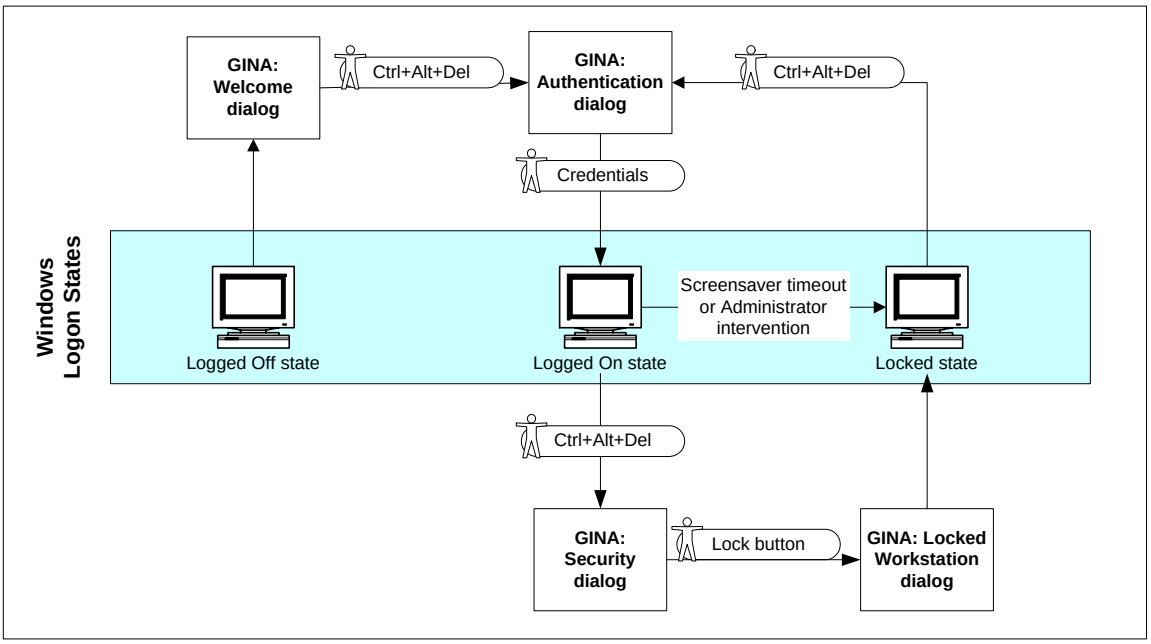
Platform	Version/System
Windows	NT 2000 XP

Windows Logon States

The GINA screens control user access to the computer and let the user change the Windows logon states. At any point in time, Windows is in one of three logon states:

- Logged-Off State
- Logged-On State
- Workstation-Locked State

The following diagram shows the three Windows Logon states and how the GINA screens and user actions change Windows from one state to another.

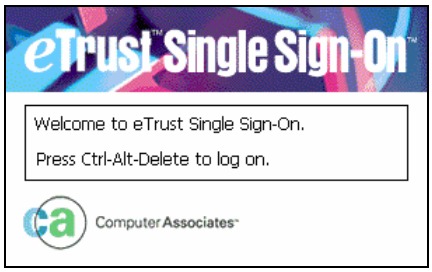


SSO GINA Dialogs

There are four different SSO GINA dialogs that are displayed according to the user action and the Windows logon state.

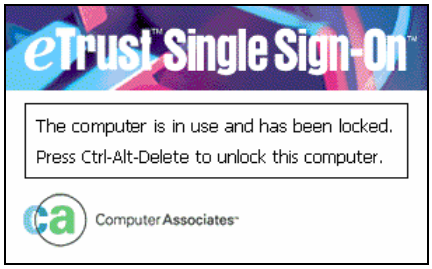
GINA: Welcome Dialog

Users will see this GINA dialog when they are not logged on to Windows. For example, when they have just started their computer.



GINA: Locked Workstation Dialog

Users will see this GINA dialog when the workstation has been locked, for example if they selected the “Lock Computer” button from the Security GINA dialog, or because they have an automatic screen lock set up on their computer. There are other ways to lock the workstation.



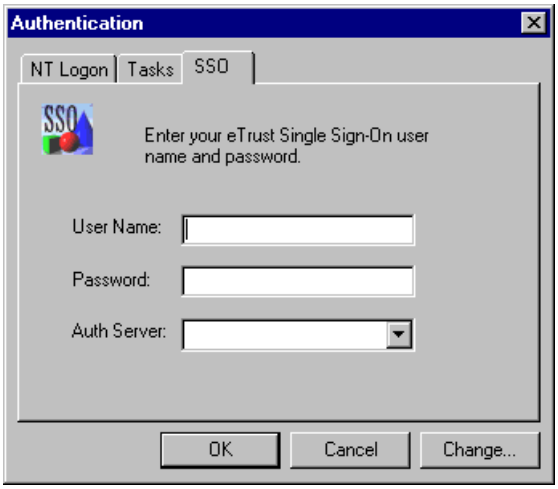
GINA: Security Dialog

Users will see this GINA dialog when they have pressed Ctrl + Alt + Del from an active Logged On Windows state, for example, if they are going to lunch and want to lock their workstation.

If the user has invoked the GINA Security dialog, but does not take any action for a period of minutes, the Windows Security dialog closes, and Windows returns to the previous state. The exception to this is when the SSO GINA is deployed in a Windows NT environment in which case, if the user does nothing for two minutes, the workstation automatically goes into station lock and shows the GINA locked dialog. The user must then press <Ctrl><Alt><Del> and then enter their credentials to unlock the workstation.

GINA: Authentication Dialog

Users will see this GINA dialog when they have pressed Ctrl + Alt + Del from either the Logon GINA dialog or the Unlock GINA dialog. The user chooses either to authenticate to SSO (and thus automatically to the Windows network or a local machine), or to log on to Windows without executing the SSO Client.



SSO—Request to authenticate to eTrust SSO using the SSO authentication method. After SSO authentication is completed, the user will be automatically authenticated to the Windows network or local machine, and the SSO Client will start. The user may need to specify the Domain on the NT Logon tab.

Tasks— Displays eTrust SSO status.

NT logon—Manual Windows logon only, without starting eTrust SSO, and without any SSO intervention. The user may need to specify the Domain on the NT Logon tab.

Change button – The user may choose which authentication method to use, if they have more than one method allocated to them, by clicking the Change button.

GINA Setup

For users to be able to log on to Windows using the SSO GINA you must create a domain application on the Policy Server. To setup a domain application for the SSO GINA on the Policy Server, use the Policy Manager, create an applications with the same name as your domain, and assign this "application" to the user(s). If you want users to be able to log on to the local machine via the SSO GINA, define an application called "NT_LOCAL_LOGON" (note all caps), and assign the users to this application. The user should ensure that the correct domain (or local logon application) is displayed in the NT tab of the SSO GINA login dialog.

SSO Client Settings for the SSO GINA

To configure the SSO GINA behavior and GINA dialogs, you must change the eTrust SsoClnt.ini file. The following describes how to change the SsoClnt.ini values for the SSO GINA.

[GINA]

LogonBitmap — This token lets you define the a bitmap image that can replace the default SSO GINA logon bitmap. If you do not specify any value here, the Client uses the default SSO GINA Welcome dialog.

LogonTitle — This token lets you define the title that will be used in the SSO GINA Logon Window. If this value is not defined, the Client uses the default value of "Windows Logon".

LockedBitmap — This token lets you define the a bitmap image that can replace the default SSO GINA locked bitmap. If you do not specify any value here, the Client uses the default SSO GINA Locked dialog.

LockedTitle — This token lets you define the title that will be used in the SSO GINA locked dialog. If this value is not defined, the Client uses a default value of "Windows Locked".

GinaPassThrough — This token lets you define whether to bypass the SSO GINA for the Microsoft GINA, assuming the SSO GINA is installed.

Yes = Microsoft GINA
(MS GINA Welcome dialog, MS GINA Authentication dialog, SSO GINA Locked dialog, SSO GINA Security dialog)

No = Use SSO GINA (all GINA dialogs)

For more information about this functionality, see the Define Workstation Modes section of this document.

logonCAD — Defines whether the CTRL-ALT_DEL bitmap (whether default or defined by the LogonBitmap property) is displayed. Valid values are:-

0 = ALWAYS show it

1 = NEVER show it

2 = Let OS decide (see the following Windows Registry Values)

It is also good practice to set the following values as shown when you use the SSO GINA.

[SSO]

CleanTicketOnStart	Specifies whether the SSO ticket is deleted from the cache when the SSO Client starts. You should leave this set to the default 'no' setting if you are using the SSO GINA.

Windows Registry Values

The following registry setting determines whether the CTRL-ALT-DEL screen is displayed within Windows:
`HKEY_LOCAL_MACHINE->SOFTWARE->Microsoft->Windows NT->WinLogon->DisableCAD`

If logonCAD is set to 0 or 1, SSO will override this value. If logonCAD is set to 2, then whatever value the DisableCAD registry setting is will determine if the CTRL-ALT-DEL screen is displayed).

The following registry setting determines the action that SSO GINA will take 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 will be displayed. If it is set to 1, the workstation will be locked. If it is set to 2, a windows logoff will be performed.

SSO Client Workstation Modes

There are several different ways that you can configure the workstation in an eTrust SSO environment. You should choose the workstation mode according to how your users access the workstations. These workstation modes affect how the workstation is "unlocked" and how the users of that machine access the SSO applications and the Windows desktop.

The four different workstation modes are:

- Single-user mode
- Multiple Users, Single Windows desktop mode
- Multiple users, multiple Windows desktops (full log off) mode
- Full shared workstation mode

System Requirements

The Shared Workstation functionality is compatible with the following platforms.

Platform	Version/System
Windows	NT 2000 XP

Workstation Modes

Single-User Mode

This is used in a normal non-shared workstation environment. The same person uses this computer all the time and only has to log on to their own session of Windows desktop and have access to their eTrust SSO applications. 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 eTrust SSO from this computer.

Multiple Users, Single Windows Desktop Mode

This is used when two or more people share a customized Windows setup, but need access to their own eTrust SSO applications on a workstation. All users work as different eTrust SSO users using the same Windows profile.

This is used when a workstation is used by more than one person who both share a customized Windows setup, but need access to their own eTrust SSO applications.

Scenario

Hilary and Mike both 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 eTrust SSO applications. When either of then unlocks the workstation in their own name they will see the same Windows setup, but their own specific eTrust SSO applications.

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

To set this up, in the data stores, define for all the SSO users the same user credential for the domain application (the application with the name of the domain to which the user will log on), or for the application `NT_LOCAL_LOGON` for local logon.

Multiple Users, Multiple Windows Desktops Mode

This is used when more than one person shares a workstation and each user wants to have their own customized Windows setup as well as their own eTrust SSO applications. All users work as different eTrust SSO users using different Windows profiles. This option also supports eTrust SSO users who use their own Windows profile. This method is slower, because it completely logs one user off Windows and then logs the next user on and is not recommended.

Scenario

Peter and Sally both share a workstation. They do very different jobs so they each want their own Windows desktop and their own eTrust 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 eTrust SSO applications.

Full Shared Workstation Mode

This is used when more than one person shares a workstation and each user is happy to share a generic Windows setup, but each user wants to have their own customized eTrust SSO applications. This is like option **Multiple Users, Multiple Windows Desktops Mode**, but is much faster and suits an environment where several people may have to use one workstation in quick succession.

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 workstation and see their own eTrust SSO applications. From here they can print to the printer right beside the computer because this Lock Option uses the local settings of the shared Windows setup.

If you use this Lock Option, you must set the GINAPassThrough to ‘yes’ in the SsoClnt.ini file. You may also want to configure the Windows Registry to automatically log the defined Windows user onto the computer.

SSO Client Settings for Workstation Modes

To configure the SSO Client for workstation modes, you must change the eTrust SsoClnt.ini file. The following describes how to change the SsoClnt.ini values for the different workstation modes.

[SystemLogon]

UnlockStationMode — This token lets you define the workstation mode.

- 0 Single SSO user
- 1 Multiple SSO users, single Windows desktop
- 2 Multiple SSO users, multiple Windows desktops (full Windows log off and log on).
- 3 Shared Workstation mode

[Logging]

ClientConfigFile — Specify the location of the SSO Client logging configuration files.

If the value is left blank, logging is disabled.

If the keyname is commented out, the default value SsoClientLog.cfg is used. Unless otherwise specified, this is created in the installation directory.

GinaConfigFile — Specify the location of the GINA configuration files.

If the value is left blank, logging is disabled.

If the keyname is commented out, the default value SsoGinaLog.cfg is used. Unless otherwise specified, this is created in the installation directory. For example: "c:\<client dir>\SsoGinaLog.cfg"

[ToolBar]

AutoLogon

This value should only be set to Yes if the SSO GINA is installed and GinaPassThrough is set to No.

This setting will cause the toolbar to automatically attempt a logon when the SSO Client is started, just as if the logon button had been pressed by the user.

If the CleanTicketOnStart item in the [SSO] section of the SsoClnt.ini file is set to No and the SSO Client has been set up to start automatically when the user logs on to the workstation, then setting AutoLogon to **Yes** will cause the SSO Client to startup and automatically log the user in as the same SSO user that was used to log into Windows.

[StationLock]

MultiUser

This determines whether the user name field in the Authentication Dialog will be enabled/disabled after an initial user has logged into SSO.

This should always be set to **yes** for Shared Workstation mode.

[EventCommands]

UserLogoffCmd

Specifies a command to run when a user logs off from SSO. In Shared Workstation mode, this will always be set to ensure that a user's applications are terminated when another user logs on. This is normally facilitated by the running of a TCL script which will close all open SSO applications and utilize the "hide desktop" utility referred to in the "Tips and Hints" section of the document to ensure no other users can see what applications the user had previously had open.

UserLogonCmd

Specifies a command to run when a user logs onto to SSO. In Shared Workstation mode, this may be set to download a global.tcl script and a logoff script from the Policy Server.

ClientShutdownCmd

Specifies a command to run when the SSO Client terminates.

ClientStartupCmd

Specifies a command to run when the SSO Client starts.

Tip: The EventCommands values can be used to provide any customized functionality that is required during these specific events – for example, Tcl scripts can be run to control screensaver behavior whilst SSO is running – it would be possible to define a TCL script to update the registry settings when SSO is started, to amend them when a user logs in, to further amend them when a user logs out and to re-instate the initial screensaver state when SSO is terminated. EventCommands should never rely on user input.

[TrayMenu]

ShowTrayIcon	Setting this value to No in a Shared Workstation environment where the Toolbar is being used, will prevent an end user from terminating the SSO Client by right-clicking on the icon in the System Tray. The only possible way they can force the termination of the SSO Client is via the Task Manager.

It is also good practice to set the following value as shown when you use the SSO shared workstation mode.

[SSO]

CleanTicketOnStart	Specifies whether the SSO ticket is deleted from the cache when the SSO Client starts. You should leave this set to the default 'no' setting if you are using the shared workstation functionality.

Tips and Hints for Shared Workstation Mode

Locking the Workstation When the Computer Starts

After auto-logging in to the machine, it is desirable that the SSO Client is invoked and the machine is placed into Station Lock automatically. This means that an end user has to pass SSO primary authentication before obtaining access to the desktop.

The automatic locking of the workstation can be achieved in several ways - two alternatives are listed below.

1. Define two shortcuts in the \Documents and Settings\All Users\Start Menu\Programs\Startup folder to invoke the SSO Client (%ProgramFiles%\CA\eTrust SSO\Client\ssointrp.exe) and the Station Lock process.
2. Add String value entries to the `HKLM\Software\Microsoft\Windows\Current Version\Run` registry key to invoke both the SSO Client and Station Lock processes.

Hide the Desktop When a User is Logging Out of SSO

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

eTrust SSO automatically comes with a utility that you can use to hide the desktop while a logoff script is running on the user's machine. If you invoke this utility at the very start of the logoff script, it will "cover" the screen, and all open windows, with either 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 SSO Client installation directory - please refer to the associated 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.

Application List Refresh

The Application List Refresh is a setting that you can configure to periodically check the Policy Server for any changes to the users application list and then automatically update the user's application list on the SSO Client, if any changes have occurred.

When to use Application List Refresh

Every time a user logs onto the SSO Client, the users application list will be updated from the Policy Server. This is the only time that the application list will be refreshed, unless you set up the automatic application list refresh to run periodically. The application list refresh functionality is therefore extremely important in an environment where users stay logged onto the SSO Client for extended periods of time, for example more than a day.

SSO Client Settings for Application List Refresh

To configure the SSO Client for automatic application list refresh, you must change the eTrust SsoClnt.ini file. The following describes how to change the SsoClnt.ini values for application list refresh.

[AppListRefresh]

EnableRefresh	To enable automatic application list refresh this value should be set to 'yes'. If this is set to 'no', the rest of the tokens in this section are ignored.
TimePeriod	Specify how many hours and minutes to elapse before the SSO Client checks for an updated application list. For example, if you set this value to 24 hours (24h0) then the application list refresh will occur once every day. If left as 0h0, then a periodic refresh does not occur, and you must a time-specific refresh (StartTime and EndTime) instead. If this value is set, then the specific-time refresh tokens are ignored.
StartTime	If you have not specified a period application list refresh using the **TimePeriod** token, you can specify a time each day that a refresh occurs (between the StartTime and EndTime). We recommend that you set a reasonable time window

	between the StartTime and EndTime beucase the refresh is then spread across the time space and this puts less strain on the network. We also recommend that you set this to run at low network traffic periods. If the values are left as 0h0 (default), the time specific refresh does not occur. The time is specified as a 24 hour clock: 21h31 indicates 9:31 pm. If these values are set, they are only used if the 'TimePeriod' token is not set. **Note:** If the StartTime and the EndTime have the same value, then a periodic refresh does not occur.
EndTime	See StartTime in this table.

Workstation Locking

When the eTrust SSO application menu is open on a user's station, anyone who walks by an unattended station can immediately access all the applications that are on the application menu.

You can eliminate this risk by using either a manual or an automatic process.

Manually

A user can manually lock their workstation by simultaneously pressing Ctrl, Alt and Delete on their keyboard and then selecting the Lock_Computer button on the GINA Security dialog.

Automatically

Another way to reduce the risk of unauthorized users accessing applications in an application menu is to define a screen saver to lock the workstation after a defined period of inactivity. This can be achieved on Windows 98 SE workstations using the properties defined in the StationLock section of the SsoClnt.ini file. For more information, see the 'Configuring the SSO Client: SsoClnt.ini' Appendix of this guide.

On Windows NT, 2000 and XP the same behavior can be achieved by defining Tcl scripts that manipulate the Windows Registry and then invoking these function scripts from appropriate SSO Client events as defined in the EventCommands section of the SSOClnt.ini file.

Chapter 14

Working with the User Data Store

eTrust SSO supports the following types of repositories to store user information:

- eTrust Access Control database
- LDAP-based directories

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 eTrust SSO system.

A *class* is the type of an object where the objects are the instances of that class. For example, the USER class defines all the fields and properties that the objects of that type (the users) will have.

The information in the objects is stored in *properties*, which are equivalent to entry fields.

Data Classes in eTrust SSO

eTrust SSO uses four 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 eTrust SSO user data store can reside on the local host (where the Policy 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 NetWare, eTrust SSO should use the same user ID as that used by NetWare or an alias name that is defined in the user entry.

The *logon name* is the name that is used to log on to an application. It can be different from the user ID. A user can have different logon names for different applications.

Entry Ownership

An owner can be a user or a group of users that are defined in the user data store. If a user or a group that owns a entry is removed from the data store, the entry no longer has an owner.

If ownership of an entry is not granted to a user or a group, the owner **nobody** must be assigned to the entry. The user **nobody** is automatically included in the user data store as a user without privileges.

Forbidden Characters in Property Values

For all classes, property values can contain most of the ASCII characters, including blanks. The following characters can **not** be used:

- [ ] Left and right square brackets
- \ Backslash
- : Colon
- ? Question mark
- | Pipe
- " Double quote marks
- * Asterisk
- , Comma

LDAP-enabled Directories

eTrust 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:

- eTrust Directory
- eTrust CA-Top Secret (TSS)
- eTrust CA-ACF2 (ACF2)
- Microsoft Active Directory
- 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 Policy 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:

eTrust Access Control	eTrust Directory	Active Directory	TSS, ACF2, RACF
USER	eTssoUser	User	-
GROUP	eTssoGroup	Group	-
LOGININFO	eTssoLoginInfo	eTssoLoginInfo Only available if the schema has been extended	-
CONTAINER	eTssoLoginInfos organization organizationalUnit country	container organizationalUnit	* The * causes the Policy Server to find the container class by the **nameby** attribute rather then the class name.

Using a Directory Schema

A *schema* is the definition of classes that are used in a directory. You can change the schemas that are supplied with eTrust 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. eTrust SSO will 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 Policy 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 Policy Server installation creates an eTrust Directory instance with an extended schema. It also provides a utility to modify 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.

eTrust Directory

The eTrust Directory user data store is an LDAP-enabled directory for holding user information. You can install this data store when you install the Policy Server. The eTrust Directory user data store can reside on the localhost (where the Policy Server is installed) or on a remote host.

eTrust Directory's schema is stored in a file. The default eTrust Directory user data store schema created by the Policy Server installation is in the file PolSrv.dxc located in the **%DXHOME%\config\schema** directory in your path.

Microsoft Active Directory

The Microsoft Active Directory user data store is an LDAP-based directory for holding user information. You can create this data store after you install the Policy Server. The Active Directory user data store can reside on the machine where the Policy Server is installed or on a remote host.

Note: You cannot use the Policy Manager or selang to add or delete users from an Active Directory user data store. Use the Microsoft Management Console to add and delete users.

Before creating a new Active Directory user data store, you must extend the schema of the Active Directory database.

The existing the schema is required to support some of the functionality of eTrust SSO (for example, saving logon information, day and time restrictions, and so on). when extending the schema, the following classes are created:

- **eTssoUser**—This class is created (based on the inetOrgPerson or User class) to include additional fields.
- **eTssoLoginInfo**—This class is created as a subclass of the top class, to save logon information.
- **eTssoLoginInfos**—This class is created as a subclass of the top class. It is used as a container to eTssoLoginInfo objects

OS/390 LDAP Directories

eTrust SSO provides strong and direct authentication for your applications to user repositories on OS/390. By using a direct access methodology, you can directly issue LDAP authentication requests to your eTrust CA-ACF2, eTrust CA-Top Secret, and RACF user repositories from the intercepts provided in eTrust SSO.

Note: To reduce the time you will wait to view the list of user entries, it is strongly recommended that filters be used when displaying users from an eTrust CA-ACF2, eTrust CA-Top Secret, or RACF user data store. For information on using filters, see the chapter 'Managing Data with the Policy Manager.'

Queries in ACF2, TSS, and RACF Data Stores

TSS, ACF2 and RACF do not support LDAP queries by class name. This means that eTrust 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 should be treated as containers, in order to get the server to find containers by nameby rather then class name. you should 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 Policy Server will disregard all containers in the data store.

The ACF2, TSS, and RACF data stores do not support the Logininfo class.

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	eTrust Access Control	LDAP-based Directories				
		LDAP (eTrust Directory)	Active Directory	TSS	ACF2	RACF
AuthenticationMethod	authnMethod	eTssoAuthnMethod	eTssoAuthnMethod	-	-	-
Comment	comment	description	description	User-Type	-	-
CommonName	commonName	cn	sAMAccountName	tssacid	acf2lid	racfid
DayRestrictions	dayRestrictions	eTssoDayRestriction	eTssoDayRestriction	-	-	-
ExpireAt	expireAt	eTssoExpireAt	eTssoExpireAt	-	-	-
FullName	fullName	displayName	displayName	Name	Name	-
IsDisabled	isDisabled	eTssoIsDisabled	eTssoIsDisabled	-	-	-
Location	location	l (lower-case L)	l (lower-case L)	-	-	-
MemberOf	memberOf	-	memberOf	memberOf	memberOf	racfConnectGroupName
Organization	organization	o	o	Zone	-	-
OrganizationalUnit	orgUnit	ou	ou	Department	-	-
Password	-	userPassword	-	-	-	-
PasswordAutoGeneration	isPwdAutoGen	eTssoIsPwdAutoGen	eTssoIsPwdAutoGen	-	-	-
PasswordInterval	pwdInterval	eTssoPwdInterval	eTssoPwdInterval	-	-	-

PasswordSynchronization	isPwdSync	eTssoIsPwdSync	eTssoIsPwdSync	-	-	-
Phone	phone	telephoneNumber	telephoneNumber	-	-	-
ResumeAt	resumeAt	eTssoResumeAt	eTssoResumeAt	-	-	-
RevokeCount	revokeCount	eTssoRevokeCount	eTssoRevokeCount	-	-	-
Surname	sn	sn	sn	tssacid	acf2lid	racfid
SuspendAt	suspendAt	eTssoSuspendAt	eTssoSuspendAt	-	-	-
TimeRestrictions	timeRestrictions	eTssoTimeRestriction	eTssoTimeRestrictions	-	-	-

Properties

Property	Description	Parameter
User	The name of the user. This is the key property. It defines if the Policy 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 Policy Server configuration file and registry key.	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
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

Property	Description	Parameter
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 Policy Server. This applies to the user's SSO password as well as passwords for applications that are managed by Policy 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
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)

Property	Description	Parameter
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 token in the Revoke section of the Policy Server initialization file and registry key.	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 Policy 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, Policy Server assumes that the directory cross-references information on the user and group objects.

  The Policy 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 that when updating in this mode, Policy Server will update 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	eTrust Access Control	LDAP-based Directories				
		LDAP (eTrust Directory)	Active Directory	TSS	ACF2	RACF
Comment	comment	eTssoComment	info	User-Type	-	-
CommonName	commenName	cn	sAMAccountName	tssprofile	acf2lidgrp	racfid
FullName	fullName	eTssoDisplayName	Description	Name	-	-
Member	member	uniqueMember	member	uniqueMember	uniqueMember	racfGroupUserIds

Properties

Property	Description	Parameter
Group	The name of the group. This is the key property.	String
Comment	A remark; not used during authorization.	String (256 character limit)
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 in to a specific application. The Policy Server sends application logon information to the application host after the Policy Server checks the user's request to log on to an application.

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	eTrust Access Control	LDAP-based Directories				
		LDAP (eTrust Directory)	Active Directory	TSS	ACF2	RACF
ApplicationName	applName	eTssoApplName	eTssoApplName	-	-	-
CommonName	commonName	cn	cn	-	-	-
CurrentPassword	currPwd	eTssoCurrPwd	eTssoCurrPwd	-	-	-
failedLoginTime	-	eTssoFailedLogin	-	-	-	-
FirstLoginTime	firstLoginTime	eTssoFirstLogin	eTssoFirstLogin	-	-	-
GraceCount	graceCount	eTssoGraceCount	eTssoGraceCount	-	-	-
LastLoginTime	lastLoginTime	eTssoLastLogin	eTssoLastLogin	-	-	-
LoginCount	loginCount	eTssoLoginCount	eTssoLoginCount	-	-	-
LoginID	loginID	eTssoLoginID	eTssoLoginID	-	-	-
NextPassword	nextPwd	eTssoNextPwd	eTssoNextPwd	-	-	-
oldPasswords	-	eTssoOldPasswords	-	-	-	-
PasswordChangedTime	pwdChangedTime	eTssoPwdChangedAt	eTssoPwdChangedAt	-	-	-
pwdHistCount	-	eTssoPwdHistCount	-	-	-	-
UserDN	-	eTssoUserDN	eTssoUserDN	-	-	-

Properties

Property	Description	Parameter
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 *userID@applicationName*.	String
CurrentPassword	The current password of the user.	String
FirstLoginTime	Date and time the user first logged in to the application through eTrust SSO.	**Date**—time_t (sec since 1970)
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 till it reaches 0, at which time the password is no longer valid.	Integer
lastFailedLoginTime	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 eTrust SSO.	**Date**—time_t (sec since 1970)
LoginCount	The number of times the user has logged in to the application through eTrust 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 eTrust SSO.	**Date**—time_t (sec since 1970)
UserDN	The distinguished name of the user that this logon information relates to.	DN

Chapter

15

Working with the Policy Data Store

eTrust SSO uses the eTrust Access Control (eTrust AC) database as the policy data store. Even though the complete eTrust AC database is installed on the Policy Server host, eTrust SSO uses only a subset of this database for its operations. This subset is called the *policy data store*.

The eTrust Access Control database information used by eTrust SSO consists of:

- Definitions for resources, applications, application groups, authentication hosts, authentication host groups, terminals, and other classes
- Rules of user access to applications and other resources and responses for those rules
- Agent definitions for the Web Agent, Application Server Agents, and any custom agents you have defined

You can populate the policy data store by using either the Policy Manager or selang commands.

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
TERMINAL	Terminals
USER_ATTR	User attributes
USER_DIR	User data stores

For instance, you may use a database to store and display your data. Each database view (record) can be defined as a member of a user-defined class that specifies what type of authority is required to create each database view. Before users are permitted to create a database view, eTrust SSO checks the authorization level of the user.

Note: Names of user-defined classes **cannot** be all uppercase.

The Agent Class (AGENT)

The AGENT class contains a variety of agent object types.

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
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.	**1** – Yes **2** – No
AGENT	The name of the agent; this is the key value.	Cannot be modified
AGENT_TYPE	The type of agent.	
CACHE_TO	Specifies the number of seconds for which the authorization decision is valid.	
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 value is the creator of the record.
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 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_TYPE	The name of the agent type; this is the key value.	Cannot be modified
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 resources in the policy data store. The application record contains information that is used when displaying the application on the user's workstation 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 should be accessed through eTrust 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

For a full description of these application types, see Managing Application Resources in the "Working with the Policy Data Store" chapter.

Each record in the APPL class contains the following properties.

Property	Description	Parameter
APPL	The application name; this is the key value.	Cannot be modified
AZNACL	The accessors (users and groups) who are permitted 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. ▪ **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 accessor has to the class. The valid access types are: all, chmod, chown, chdir, create, delete, execute, join, modify, none, password, read, rename, sec, utime, and write.	
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 record names of the contained applications if the current record describes a container.	
CREATE_TIME	The time the record 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)

Property	Description	Parameter
ICONFILE	The file name or full path of the file 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: 1. Current directory 2. Windows system directory 3. Windows directory 4. Directories listed in the PATH environment variable	The default is the default icon of each user's workstation
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 eTrust 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)
LOGIN_TYPE	How user passwords are to be provided.	▪ **pwd**—Plain password (default) ▪ **none**—No password required. ▪ **appticket** or **passticket**—Generated by the Policy Server.
MASTER_APPL	The record name of the application supplying the password.	▪ No master (default)

Property	Description	Parameter
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.	▪ Blank ▪ The creator of the record (default)
PWD_AUTOGEN	Indicates whether the application's password is automatically generated by the Policy Server.	▪ 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)
PWPOLICY	The record name of the password policy 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 accessors who are not defined to eTrust 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.
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 record 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.

You create a record for each application in the APPL application class before linking specific applications to the relevant group in the GAPPL class.

For example, suppose the policy data store contains a record in the class 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 a record in the GAPPL class. Each record in the GAPPL class contains the following properties.

Property	Description	Parameter
GAPPL	The application group name; this is the key value.	Cannot be modified
AZNACL	The accessors (users and groups) who are permitted to access the application.	
COMMENT	A remark; not used during authorization.	
CREATE_TIME	The time the record was created.	Cannot be modified
MEMBERS	A list of applications that belong to the group.	
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
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.
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 accessors who are not defined to eTrust 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 **ALL**—Permission to invoke the application. You can use the abbreviation A. ▪ **None**—(default) No permission. You can use the abbreviation N
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 eTrust SSO users. You determine which hosts can authenticate which users by adding a record to the AUTHHOST class in the policy data store.

Note: The authentication host for Windows NT authentication cannot contain more than 15 characters since 15 is the maximum number of characters Windows NT allows for the name of an NT server or workstation.

Each authentication host is represented by a record in the AUTHHOST class. The name of an AUTHHOST record can be any logical name.

For Windows NT and NetWare authentication, the name of the host must be written in uppercase letters; for SSO and SecurID authentication the host name must be written exactly as it is written in the operating system. In UNIX, the norm is that host names are written in lowercase.

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.	
AUTHHOST	The authentication host name; this is the key value.	Cannot be modified
AZNACL	A list of accessors and whether they have permission to log on using the authentication host. Values are determined by the authorize and authorize- commands.	
AUTH_METHOD	The AUTHHOST object's method ID.	
COMMENT	A remark; not used during authentication.	
CONT_FORMAT	A format string that the Policy 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.	
CREATE_TIME	The date and time the record was created.	Cannot be modified

Property	Description	Parameter
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 pertains to the X.509 authentication method and is necessary for backward compatibility with SSO Clients.	
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)
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 eTrust 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 database where the user is defined.	Cannot be modified
USER_FORMAT	A format string that the Policy 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 Policy 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.	
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)

Each record in an authentication host group 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 a record 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 record representing the Accounting Department in the policy data store 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
GAUTHHOST	The authentication host group name; this is the key value.	Cannot be modified
AZNACL	A list of accessors and whether they have permission to use the authentication hosts. Values are determined by the authorize and authorize-commands.	
COMMENT	A remark, not used during authentication.	
CREATE_TIME	The date and time the record was created.	Cannot be modified
MEMBERS	A list of the authentication hosts that belong to the group.	
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)
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 a record in the PWPOLICY class. For more information about password policies, see the "Managing Passwords" chapter.

Each record in the PWPOLICY class contains the following properties.

Property	Description	Parameter
PWPOLICY	The password policy name; this is the key value.	Cannot be modified
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 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
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
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 record in the RESOURCE_DESC class contains the following properties.

Property	Description	Parameter
RESOURCE_DESC	The resource description name; this is the key value.	Cannot be modified The default is the name of the user-defined class
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
RESPONSE_TAB	The name of the object in the RESPONSE_TAB class.	Cannot be modified
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 Terminal Class (TERMINAL)

The TERMINAL class defines objects that represent the workstations used to run the Policy Manager or send selang commands. Administrators can perform administrative tasks from a workstation only if they have READ and WRITE permission on the Policy Server and to their workstations.

Each record in the TERMINAL class contains the following properties.

Property	Description	Parameter
TERMINAL	The name of the terminal; this is the key value.	Cannot be modified
ACL	The access control list that applies to the accessors requesting access to the terminal. Each element in the list contains the following information: ▪ **Accessor reference**—A reference to an accessor object (USER or GROUP). ▪ **Permitted access**—The access type that the accessor has for this resource. Valid values are NONE, READ, and WRITE.	The data type is Special.
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 (255 character limit)
CREATE_TIME	The date and time the record was created.	Cannot be modified
DAYTIME	The day and time restrictions that govern when the terminal can be accessed.	
GROUPS	The list of terminal groups that this terminal belongs to.	
OWNER	Controls who can manage the record. An owner can be a user or a group of users that are defined in the user data store. If a user or a group that owns a record is removed from the data store, the record no longer has an owner. If ownership of a record is not granted to a user or a group, the owner **nobody** must be assigned to the record. The user **nobody** is automatically included in the user data store as a user without privileges.	▪ The creator of the record (default) ▪ **nobody**—If no value entered

Property	Description	Parameter
UACC	The default access for the resource, which specifies the access granted to accessors who are not defined to eTrust SSO or who do not appear in the resource's access control list. Refer to the ACL property for a list of valid values.	▪ **NONE** (default) ▪ **READ** ▪ **WRITE**
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
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 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 userdir database. Since different databases can contain different attributes, the attribute fields should be synchronized.	
FIELDID	For internal use only; the internal number of the USER_DIR object attribute in the mapping table managed by the Policy Server.	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

Property	Description	Parameter
USERATTR_FLAGS	Contains information about the attribute.	▪ **aznchk**—Indicates that this attribute will be used for authorization. ▪ **predef** or **freetext** or **userdir** —The source of the user attributes. ▪ **user** or **group**—Whether the attribute (accessor) is a user or a group.
USER_DIR_PROP	The name of the user's directory.	Cannot be modified

The User Directory Class (USER_DIR)

The USER_DIR class contains information about the user data store. eTrust SSO can work with different types of user data stores, and the authentication mechanism can refer to users who are not defined in the policy data store. This means that you must store all necessary information about the database in this class.

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.	
CONTOBJ_CLS	A list of the object classes that are containers.	
CREATE_TIME	The date and time the record was created.	Cannot be modified
DIR_TYPE	The type of user directory.	▪ **ETRUST_AC**—(default) eTrust Access Control ▪ **LDAP**—Any LDAP-enabled directory, including eTrust Directory ▪ **AD**—Microsoft Active Directory ▪ **TSS**—eTrust CA-Top Secret ▪ **ACF2**—eTrust CA-ACF2 ▪ **RACF**
GRPOBJ_CLS	The name of the classes that the user object inherits from. This property is needed for the creation of new groups in an LDAP directory.	

Property	Description	Parameter
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.	
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.	
LOCATION	Where the host resides.	
MAX_RET_ITEMS	The maximum number of items retrieved per query.	
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)
PATH	The relative distinguished name in the LDAP tree where all queries begin.	
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 eTrust SSO should wait for a connection to the user directory.	
UACC	The default access for the resource, which specifies the access granted to accessors who are not defined to eTrust 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.

Property	Description	Parameter
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 was created with this USER_DIR object as the value for the user_dir parameter.	Cannot be modified The default is the list user's attribute
USERDIR_HOST	The DNS name of the host where the database 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.	
VERSION	The user directory's version number.	

Generic Classes

The installation of the Policy Server automatically creates the following predefined classes:

- EJB
- GEJB
- URL
- GURL

Note: EJBs and GEJBs are not relevant to eTrust SSO. For more information about EJBs, see eTrust Web AC documentation.

These predefined classes, with the exception of the GEJB and GURL classes, are created with the same properties as a user-defined class. See Creating User-Defined Classes for a description of each property.

The purpose of the GEJB class is to group the resources of the EJB class. The purpose of the GURL class is to group the resources of the URL class. Therefore, the GEJB and GURL classes have different properties than the other predefined classes since they are grouping classes.

You can create generic access rules that apply to these generic resource classes and user-defined classes. See your eTrust Access Control documentation for more information about creating and using generic rules.

Creating User-Defined Classes

When you define new agent types (AGENT_TYPE objects), the agent type you create may require you to create additional classes to the ones offered.

eTrust SSO gives you the ability to define classes to fit your company's needs. For instance, you may use a database to store and display your data. Each database view (object) can be defined as a member of a user-defined class that specifies what type of authority is required to create each database view. Before users are permitted to create a database view, eTrust SSO checks the authorization level of the user.

To create a user-defined resource class or group, choose User-Defined class manager from the Tools menu. Click the New icon to start the Create Class wizard. This wizard will help you define your new resource class or group.

Important! *To use the wizard, the Policy Server and Policy Manager must be installed on the same computer.*

While creating the class, remember that names of user-defined classes **cannot** be all uppercase. You can also specify different access rights for your classes (for example, the URL class has GET and POST rights and the APPL class has the EXECUTE right).

You can also create groups of classes (like URL and GURL). While creating the name of the group class, remember that the name must always start with a capital G. In addition, the group class must have the same access rights as the individual class (for example, the URL and GURL classes have the same access rights—GET and POST.)

Modifiable Properties

The properties that you can modify in a user-defined class object are listed and explained next.

Note: Only these properties apply when defining a resource grouping class: AZNACL, COMMENT, OWNER, MEMBERS, and UACC.

AZNACL

The list of accessors (based on user attributes) permitted to access the class and their access types. Each element in this list contains the following information:

User Attr
The user attribute on which the permission is defined.

Attr Value
The value of the user attribute.

User Dir
The user data store to which this user attribute refers.

Permitted access
The types of access the accessor has to the class. The valid access types depend on the class. For example, you can define the class DRIVE with access types of read, format, burn, and defragment. You can also select None if you do not want any of the available accesses used or select All to permit the use of all available accesses.

Note: Certain words are reserved and cannot be used as access types.

COMMENT	Additional information you want to include in the object. The alphanumeric string can contain up to 255 characters. eTrust SSO does not use this information for authorization. Use the comment or comment- parameter with the chres, editres, and newres commands to modify this property.
DAYTIME	The day-of-week and time-of-day restrictions that govern when the resource can be accessed. Use the restrictions(days and time) parameter with the chres, editres, and newres commands to modify this property.
GROUPS	The grouping class objects that this resource is a member of if this class is not a group (like the GURL class). This property is defined to the group resource object by the mem(members) parameter and the chres, editres, and newres commands.
MEMBERS	If this class is a group class for a resource, then this property contains the members of the resource. Use the mem(members) parameter with the chres, editres, and newres commands to modify this property.
OWNER	The user or group that is the owner of the object. Use the owner parameter with the chres, editres, or newres command to modify this property.
UACC	The default access for the resource, which specifies the access granted to accessors who are not defined to eTrust Web AC or who do not appear in the resource's access control list. Refer to the ACL property for a list of valid values. Use the defaccess parameter with the chres, editres, or newres command to add or change this property.

Non-modifiable Properties

The properties that you cannot change in a user-defined class object are listed and explained next..

CREATE_TIME	The date and time the object was created.
UPDATE_TIME	The date and time the object was last modified.
UPDATE_WHO	The person who performed the update.

Chapter 16

Working with the Token Data Store

The token directory is an LDAP directory in which the Policy Server stores the user's session information. This includes the session ID, client IP number, username, and the last heartbeat time.

During the authentication process, the Policy Server uses the token directory to save token and user information. If a Policy Server fails, the backup server can use the token directory to fetch token information for unknown tokens.

The token directory solution supports configurations with very large number of Policy Servers. If the Policy 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 Policy Servers.

Policy Server Background Processes

Policy Server uses two background processes:

- **Remove Expired Token**—searches for tokens that have expired and removes them.
- **Remove HBF Tokens**—searches for tokens that have not sent a heartbeat for longer than the grace period, and removes them. HBF stands for Heartbeat Failed.

On UNIX, adjust the following settings in the policyserver.ini file. On Windows, use the following registry keys:

```
HKLM\SOFTWARE\ComputerAssociates\eTrust\Shared\Policy Server\8.0\bgc.RET
HKLM\SOFTWARE\ComputerAssociates\eTrust\Shared\Policy Server\8.0\bgc.RHFT
```

Keyname	Description	Parameters
Enabled		
IdlePeriod	A cycle means an entire job the background process should 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 Policy Server starts.	Either: ▪ The delay (in seconds) between when the background process started and when the Policy Server should start. ▪ The time of a day it should start (in xxhxx format)

The eTssoConnectedUser Class

The eTssoConnectedUser class represents a user identity. A user can have more then one session with the Policy Server, thus every eTssoConnectedUser may have more than one eTssoSession.

This means that the user can have one ConnectedUser object but more then one concurrent session.

This class contains the following fields:

Property	Description	Parameter
commonName	The name of the object This is a combination of user name container and user dir to create a uniquely distinguished name. This naming convention is called Ename or Enterprise Name.	The Ename format is: <*userDB_name*>://Cont:<*container_dn*>_Uid:<*user_name*> (where the container_dn and the user name have _ instead of every =
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 should log off all of their active sessions (the registered ones).

Property	Description	Parameter
eTssoScreenLockTimeout	The timeout (on idle) used before activating screen lock.	Integer (seconds)
eTssoLogoutTimeout	The timeout (on idle) used before logging a user out.	Integer (seconds)
eTssoMaxUserSessions	The maximum number of concurrent sessions allowed for a user.	Integer
eTssoSessionLimitChoice	A code specifying how Policy Server should handle a session limit.	
eTssoHeartbeatFailBehaviour	A code specifying how Policy Server should 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 should 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 should 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 client station	String
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 created the ticket 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.

Property	Description	Parameter
eTssoMissedHeartbeatAllowed	The number of heartbeats that the Policy 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 Policy Server expects the client to send	Integer (seconds)
eTssoHBEncKey	The encryption key used by the client to encrypt the heartbeat message	
eTssoHBEncKeyLen	The encryption key length used by the client to encrypt the heartbeat message	Integer
eTssoLoginTime	The time that the ticket used to register this session was created	**Date**—time_t (sec since 1970)
eTssoClientHost	The client machine	Machine name or IP address
eTssoClientPort	The port of the client machine	
eTssoClientEncKey	The encryption key used by the Policy Server to connect and communicate with the client for kill sessions When Policy Server wants to kill a client connection it should encrypt the request with the client encryption key.	
eTssoClientEncKeyLen	The length of encryption key used by the Policy Server to connect and communicate with the 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 client used this session in.	Multi-valued

Chapter

17

Working with Server Farms

This chapter discusses how to use a server farm to provide reliable and rapid services.

A *server farm* is a system of multiple networked Policy Server computers. The data on each server is replicated to all the other servers.

In a server farm, you can set up a ***failover*** system that automatically switches from a failed server to a running server. This allows users to keep working without interruption.

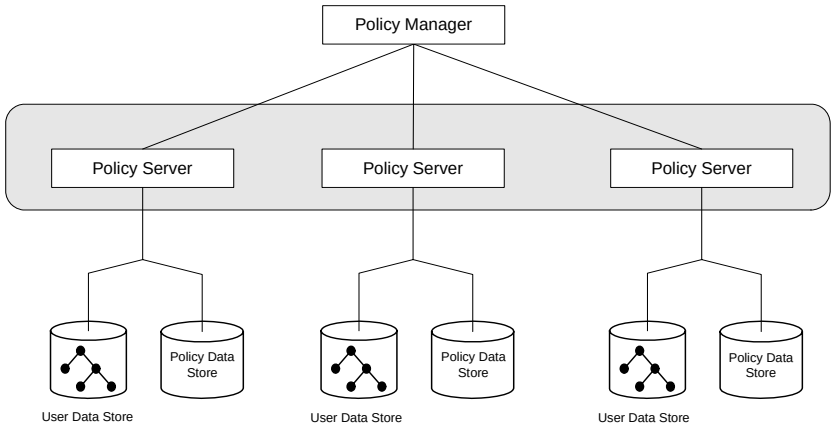
A server farm can also work with a ***load-balancing*** system to distribute communications and processing between the Policy Servers in the farm.

Replicating Data Within a Server Farm

When eTrust SSO uses a server farm, each of the Policy Servers must replicate their data to the other Policy Servers. This means that each Policy Server simultaneously updates their local data store and all the data stores of the other Policy Servers in the replication group.

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.

You can manage the entire server farm through a single Policy Manager, as shown in the following illustration:

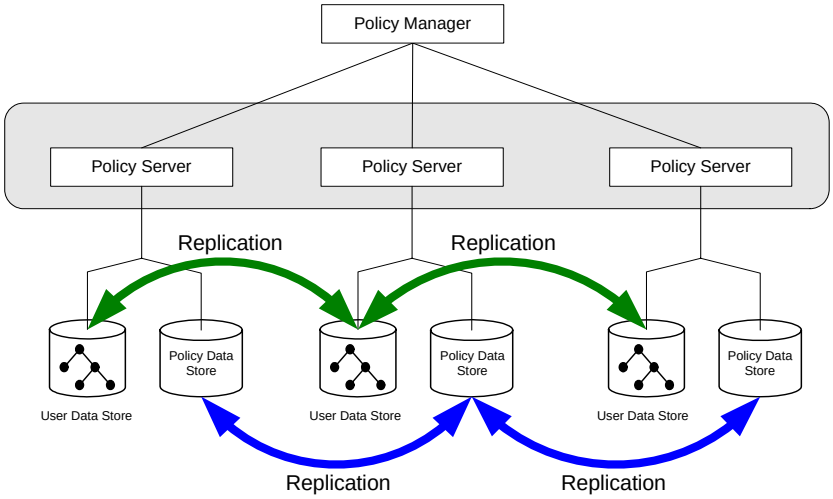


Server farms are good for many reasons:

- If one server fails and other servers keep working, users will notice no interruption to their work.
- If one server fails, no data will be lost because all data stores are replicated in real time
- A server farm allows a load balancer to be used for increased throughput and performance

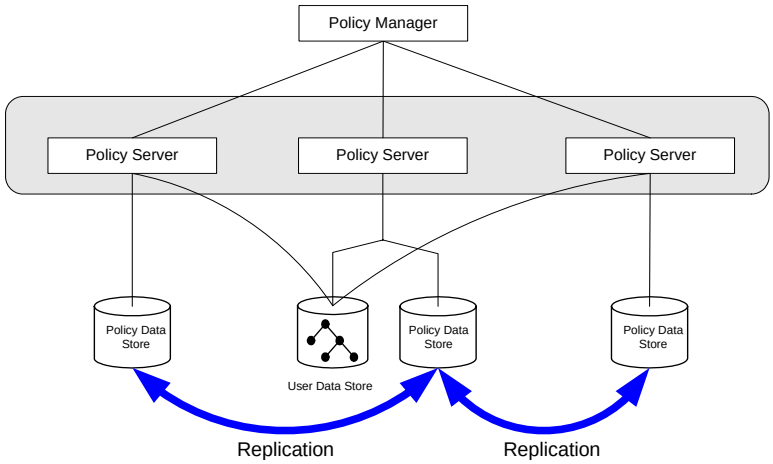
Designing a Policy Server Farm

When using a server farm, you need to configure the user data store and policy data store to accurately replicate all of their data to the other members of the server farm. For more information about setting up a server farm and replicating information, see the "Implementing A Server Farm" chapter of the *eTrust SSO Implementation Guide*. Replication duplicates any additions, modifications, or deletions to any of the data stores in the other data stores. The following illustration shows how the replication mechanism works.



The replication mechanism for the policy data store is based on the proven eTrust Access Control PMDB (Policy Model Database) technology. The replication mechanism for the built-in LDAP user data store (eTrust Directory) is based on the highly scalable, X.500-based architecture of this directory.

As an alternative, you may decide to use only one user data store and multiple replicated policy data stores. If you do this, configure each of the Policy Servers in the server farm to point to the user data store computer, as shown in the following illustration:



Each Policy Server can function as the main server for a number of workstations running SSO Client, while at the same time serving as a backup Policy Server for all the other Policy Servers.

The data stores are fully replicated, which means that user information and access control policies are defined centrally and replicated to the other Policy Servers in real time. You can further augment this type of highly scalable architecture by installing hardware or software load balancers.

Maintaining a Server Farm

If you are performing maintenance on a Policy Server farm that has full replication implemented, the best backup method is to do the following:

1. Stop the Policy Server and the eTrust Access Control services on **one** of the server hosts.
2. Back up the data stores of that Policy Server.
3. Restart the Policy Server and eTrust Access Control services.

   This server's database will then be updated with all of the changes that took place while it was down, as long as at least one other Policy Server in the server farm remained running.

Failover

In an eTrust SSO system with server failover, if one server stops working, the users who were authenticated to the failed server can keep working without interruption. This is because all data is replicated to other servers, and one of those other servers can immediately take over the failed server's functions.

How Server Failover Works

To implement server failover, you need:

- The Servers keyname in the SsoClnt.ini file must list two or more active Policy Servers on the network. For example:

  ```
  [ServerSet0]

  PolicyServers = server1 server2
  ```

- All authentication hosts (AUTHHOSTs) must be defined in all eTrust SSO data stores on all Policy Server hosts
- Each AUTHHOST must have a key. The key value must be the same on each Policy Server in the failover group. This is done automatically by the replication mechanism.

  For example, in the server1 data store, there are two AUTHHOST records:

  ```
  Authhost server1 key(1111)
  Authhost server2 key(2222)
  ```

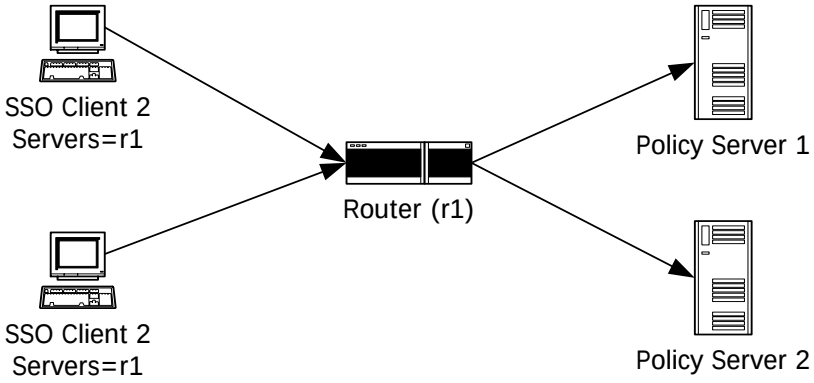
  In the server2 data store, there are also two AUTHHOST records:

  ```
  Authhost server1 key(1111)
  Authhost server2 key(2222)
  ```

- The data stores must be replicated. The eTrust Directory and eTrust Access Control data stores can each be implemented to replicate. For more information about setting up a server farm and replicating information, see the "Implementing A Server Farm" chapter of the *eTrust SSO Implementation Guide*.

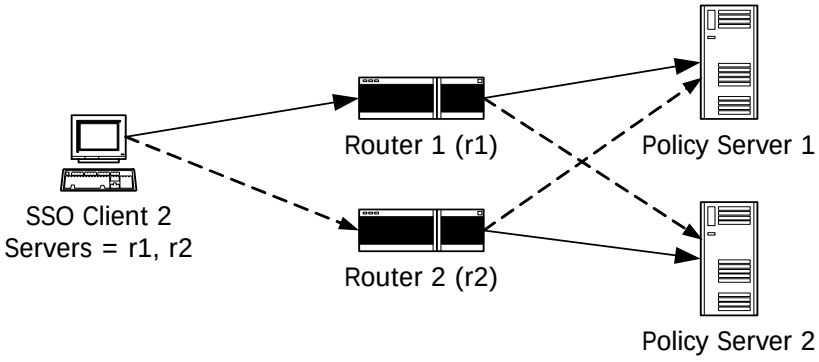
Failover Using Routing Hardware

You can install a load balancing router between SSO Clients and the Policy Servers. The router is then responsible for identifying failed servers and routing requests to others servers in the farm. When the failed server resumes its operation, the router automatically detects it and starts routing requests to the server again.



To implement this, the Servers keyname in the SSO Client configuration file is set to the IP address of the router, and the router has a list of the IP addresses of all the servers in the farm.

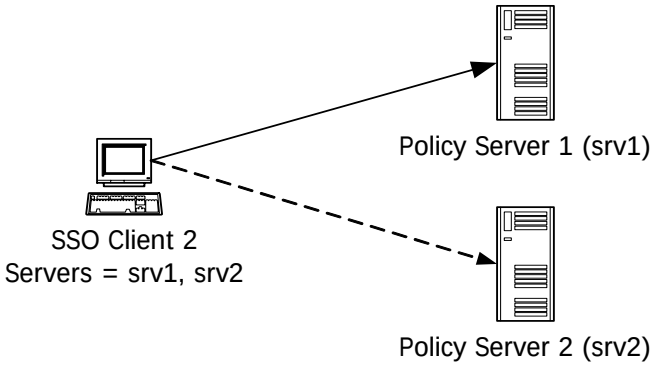
It is possible to use more that one hardware router to provide a backup in case primary router fails.



Policy Server Failover Using the SSO Client

To enable failover using only the SSO Client, include more than one server in the [ServerSet0] keyname in the SsoClnt.ini file on the SSO Client computer.

The client tries to work with the server that appears first on the list. If the first server on the list is not working, the client will request services from the next server on the list, and so on.



The assignment of clients to servers will have to be planned based on the following factors:

- Number of users at each local site
- Expected activity level
- Geographical considerations
- Survivability and physical security considerations
- Load balancing

Setting up a replication data store involves creating the replication data store and defining host data stores as subscribers of the replication data store.

Authentication Host Failover Using the SSO Client

To enable authentication host failover using only the SSO Client, you must change the SsoClnt.ini file on the SSO Client computer. You should list multiple authentication host servers that correspond to the authentication method that you have specified. For example,

[ServerSet0]
AuthMethods=LDAP SSO
AuthLDAP=AuthServer1 AuthServer2

The client tries to work with the server that appears first on the list. If the first server on the list is not working, the client will request services from the next server on the list, and so on.

The Replication Data Store

The data stores contain all the eTrust SSO data for the enterprise. They are updated by the local Policy Server with the data that arrives from users and from eTrust SSO administrators.

The replication data store is a data store that is configured to replicate to other data stores. This means that it includes a list of subscriber data stores that reside on separate computers. Whenever a change is made to the replication data store, the subscriber data stores are automatically updated.

For example, if a user is deleted from the replication data store, a command to delete that user is sent to all the subscriber data stores. In this way, a single command removes a user from many data stores on a variety of computers.

If a subscriber data store does not respond, the replication data store sends the command every 30 minutes until the subscriber data store has been updated. If a subscriber data store is responding, but refuses to apply the transaction, the replication data store places the transaction in an error log.

In a server farm with universal replication, all data stores are configured to be replication data stores.

The Token Data Store

When a user logs on, eTrust SSO stores information about the user's session in the token data store and creates a handle to the information, the token.

To allow any server in a server farm to process a request from any client, the information in the token data store needs to be available to all members of the server farm at all times. In a server farm or failover environment, the token data store should be replicated to remove a possible single point of failure.

This eliminates the need for the user to re-authenticate when a server fails and for server requests to be evenly distributed (loaded) across all members of the server farm.

Keeping Databases Synchronized

When you make changes with the Policy Manager in the policy data store or in an eTrust Access Control user data store, the changes are automatically replicated and all of the databases remain synchronized. However, if you make changes with or from a third-party manager that issues commands, the changes are not automatically replicated and the databases are no longer synchronized.

If you make changes using selang, enter the following command to keep the databases synchronized:

```
eTrust> hosts (localhost PS_PMDB@localhost) uid(ps-admin) password
(<ps-admin's password>)
```

For example:

```
eTrust> hosts (localhost PS_PMDB@localhost) uid(ps-admin) password
(<ps-admin's password>)
(localhost)
Successfully connected
INFO: Target hosts' version is <XXX>
(PS_PMDB@localhost)
Successfully connected
INFO: Target hosts' version is <XXX>
eTrust>
```

Where <XXX> is the version of eTrust AC.

If you are using a third-party manager, refer to its documentation to find out the way it updates the database.

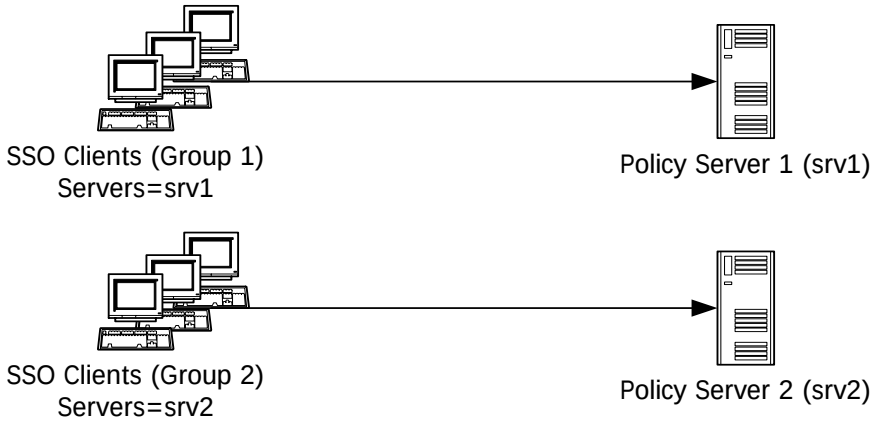
This is currently required because at present when an administrator is defined on one host, his definitions are replicated to all other local data stores. However, he will not be defined in the other PMDBs. As a result, when a user `who is newly defined as an administrator on only one host tries to make data store changes from a second Policy Server, the changes will not be replicated because this user is not defined as an administrator in the PMDB of the second Policy Server.

Load Balancing

The following options are available for SSO Clients working with the Policy Server.

The SSO Client can be configured to work with a particular Policy Server by defining the Servers keyname in the SsoClnt.ini file on the SSO Client computer. Although it is possible to define more than one server in this keyname, this can only be used to enable failover. It does not enable load balancing.

The simplest load balancing solution is to divide all SSO Clients installed on a site into groups and assign different primary servers for every group:



Load Balancing with a Hardware Load Balancer

If you have set up a server farm, you can use a hardware load balancer or router to balance the load between the Policy Servers in the farm.

To do this, set each SSO Client to use the router instead of a Policy Server. Set this in the Servers keyname of the SsoClnt.ini file.

The load balancer measures the load on the Policy Servers in the farm and routes requests to different servers according to one of the algorithms supported.

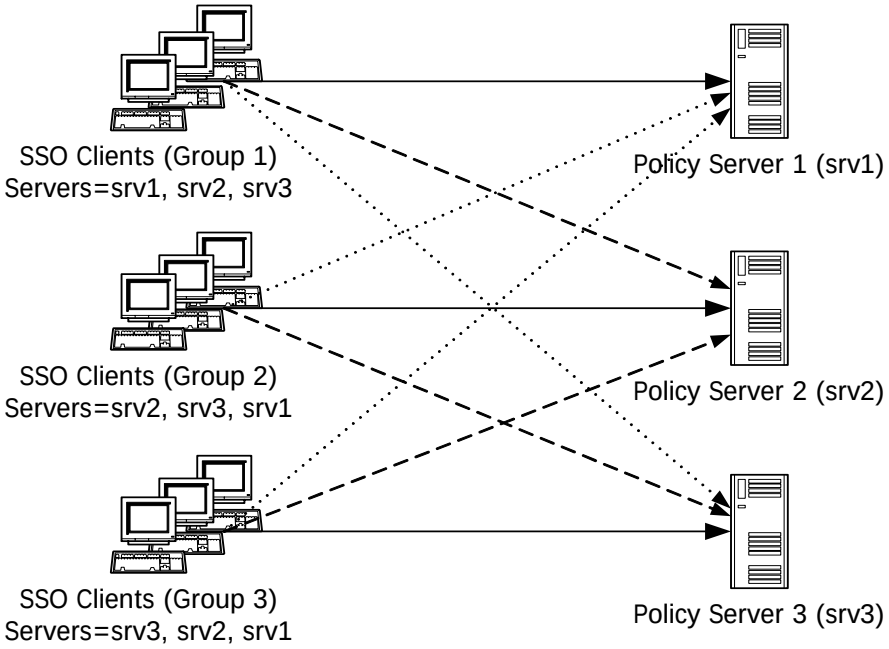
This architecture can also support failover. See the Failover Using Routing Hardware section for an illustration.

Load Balancing and Failover

In large SSO implementations, for the load balancing purposes different groups of SSO Clients will be configured to work with different primary servers. In this case, other servers can play a role of backup servers.

For example, in an organization with three hundred SSO Clients and three servers, the Servers keyname definitions for SSO Clients could be as follows:

- First group of one hundred SSO Clients: srv1, srv2, srv3
- Second group of one hundred SSO Clients: srv2, srv3, srv1
- Third group of one hundred SSO Clients: srv3, srv1, srv2



Chapter

18

Maintenance

Policy Server

You can stop and start the Policy Server on Windows using Windows Services. You access Windows Services from the Start menu by selecting Control Panel, Administrative Tools, Services.

You can also start and stop the Policy Server using the command prompt. See below for further details.

Note: You must be a root user to perform these functions on a UNIX computer.

Stopping the Policy Server

To stop the Policy Server on Windows follow these steps:

1. Open a command line prompt.
2. Navigate to <install path for Policy Server>\bin
3. Type: `policyserver –stop`

To stop the Policy Server on UNIX follow these steps:

1. Navigate to <install path for Policy Server>/bin
2. Type: ./`stopserver`

Starting the Policy Server

To start the Policy Server on Windows follow these steps:

1. Open a command line prompt.
2. Navigate to <install path for Policy Server>\bin
3. Type: `policyserver -start`

To start the Policy Server on UNIX follow these steps:

1. Navigate to <install path for Policy Server>/bin
2. Type: `./startserver`

Checking The Status of the Policy Server

To check the status of the Policy Server on Windows go to the Windows Services tool available from the Windows Start menu.

To check the status of the Policy Server on UNIX use the ps command. For example:

```
ps -aef | grep PolicyServer
```

eTrust Access Control

You can stop and start eTrust 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 eTrust Access Control using the command prompt. See below for further details.

Stopping eTrust Access Control

To stop eTrust Access Control on Windows follow these steps:

1. Open a command line prompt.
2. Type: `secons -s`

To stop eTrust Access Control on UNIX follow these steps:

1. Navigate to <install path for eTrust Access Control>`/bin`
2. Type: `secons -s`

Starting eTrust Access Control

To start eTrust Access Control on Windows follow these steps:

1. Open a command line prompt.
2. Type: `seosd -start`

To start eTrust Access Control on UNIX follow these steps:

1. Navigate to <install path for eTrust Access Control>/bin
2. Type: `./seload`

Checking The Status of eTrust Access Control

To check the status of eTrust Access Control on Windows, go to the Windows Services tool available from the Windows Start menu.

To check the status of eTrust Access Control on UNIX, use the ps command, for example:

```
ps –aef | grep seosd
ps –aef | grep seagent
ps –aef | grep seoswd
```

eTrust Directory

You can stop and start eTrust 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 eTrust Directory using the command prompt. See below for further details.

Before You Stop or Start 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 Policy Server service before you stop or start eTrust Directory.

Stopping eTrust Directory

You can stop eTrust Directory on both UNIX and Windows using the same command. You do not have to be in a particular directory to run these commands.

To stop eTrust Directory on Windows or UNIX follow these steps:

1. Open a command line prompt
2. To stop a specific DSA type: `dxserver stop <dsa_name>`

   To stop all DSAs type: `dxserver stop all`

Starting eTrust Directory

You can stop eTrust Directory on both UNIX and Windows using the same command. You do not have to be in a particular directory to run these commands.

To start eTrust Directory on Windows or UNIX follow these steps:

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`

Checking the Status of eTrust Directory

To check the status of eTrust Directory on either Windows or UNIX follow these steps:

1. Open a command line prompt
2. Type: `dxserver status`

Authentication Agents

Certificate, Entrust, LDAP, and RSA SecurID (Windows)

For the following authentication agents, the ticket granting agent (TGA) can be started either as a Windows service or as a stand-alone executable from the command line:

- Certificate
- Entrust
- LDAP
- RSA SecurID (Windows)

The syntax for the authentication agents commands is:

```
tga<auth-method>.exe <option>
```

where <option> is one of the following:

-start [name]
Start the service corresponding to the specified name (or the default service, if the name has not been specified)

-stop [name]
Stop the service corresponding to the specified name (or the default service, if the name has not been specified)

-i[nstall] [name]
Create a service object using information based on the specified name (or the default set of information, if the name has not been provided) and add it to the Service Control Manager database

-r[emove] [name]
Mark the specified service for deletion from the Service Control Manager database

-d [name]
Run the application from the command line, using configuration settings of the service corresponding to the specified name (or the default service, if the name has not been specified)

For example:

```
tgaCertificate.exe -start
```

Windows

The syntax for the Windows authentication agent commands is:

```
SSOAuthNT.exe <option>
```

where <option> is one of the following:

-start
Start the Windows authentication agent service

-stop
Stop the Windows authentication agent service

-i[nstall]
Create a service object and add it to the Service Control Manager database

-r[emove]
Mark the Windows authentication agent service for deletion from the Service Control Manager database

For example:

```
SSOAuthNT.exe -start
```

Novell

To start the NetWare authentication agent manually, use the load command:

```
NW_server : load ssoauth
```

To disable the NetWare authentication agent, use the Esc key in the NetWare authentication agent main console, or enter the unload command:

```
NW_server : unload ssoauth
```

To run the agent without displaying the console, use the load command with the -NOSCREEN option:

```
NW_server : load ssoauth -NOSCREEN
```

RSA SecurID (UNIX)

The syntax for the command to start the UNIX authentication agent is:

```
ssorsasd [options]
```

Where options is one or more of the following:

-altpwd
Change the password. cd to '/usr/tmp' (Inactive if -nodaemon specified). The default password is '/'.

-c(fg) <path>
Specify the path to a configuration file. By default, the agent attempts to locate tga_rsa.ini file in predefined locations, starting with current directory.

-h(elp)
Show the Help menu.

-n(odaemon)
Do not start the daemon automatically when the executable is launched.

-port <PortNumber>
Override the default port (13969).

SafeWord

The syntax for the SafeWord authentication agent commands is:

```
ssoswd [-c <ConfigFile>] [-p <PortNumber>] [-h] [-altpwd] [-nodaemon]
```

-altpwd
Change the password. cd to '/usr/tmp' (Inactive if -nodaemon specified). The default password is '/'.

-c <ConfigFile>
Specify the path to a configuration file. By default, the agent attempts to locate swec.cfg file in predefined locations, starting with current directory.

-h
Show the Help menu.

-nodaemon
Do not start the daemon automatically when the executable is launched.

-p <PortNumber>
Override the default port (13970).

One Time Password (UNIX)

The syntax for the UNIX One Time Password (OTP) authentication agent commands is:

```
startserver [-p <PortNumber>] [-c(fg) <file path>] [-v] [-h] [-nodaemon]
```

-c[fg] <path>
Specify the path to a configuration file. By default, the agent attempts to locate seotp.ini file in predefined locations, starting with current directory.

-h
Show the Help menu.

-nodaemon
Do not start the daemon automatically when the executable is launched.

-p <PortNumber>
Override the default port (13970).

-v
Shows the options being used ("Verbose").

To stop the OTP authentication agent type:

```
stopserver
```

PS Watchdog

There is a new process installed with the Policy Server, called eTrust Policy Server Watchdog (PS Watchdog).

This process monitors Policy Servers and supplies status information such as whether the Policy Server is running or not.

Start and Stop the PS WatchDog

The PS Watchdog is not turned on by default when the Policy Server is installed. It is installed as a service that needs to be manually started.

This section explains how to turn the PS Watchdog on.

Windows

An administrator can turn this process on by going to the Windows Services Manager and starting “eTrust Policy Server Watchdog Service”. You can access Windows Services from the Start menu by selecting Control Panel, Administrative Tools, Services.

Unix

An administrator can turn this process on by typing the following commands.

To start the PS Watchdog

```
PSWD –start
```

To stop the PS Watchdog:

```
PSWD -stop
```

Using the PS Watchdog

The PS Watchdog process periodically checks the Policy Server (by running a basic functionality server check), and can report the current status if desired.

To check the Policy Server status, follow these steps:

1. Activate the service on a Policy Server machine. For more information see, "Starting the PS Watchdog".
2. Open a web browser and type the following address:

   ```
   http://<<machine name>>:13391
   ```

   The page returned should contain: SYSTEM TEST SUCCESS
3. Stop the Policy Server, but keep the PS Watchdog running.
4. Repeat step 2.

   The page returned should contain a failure response.

Note: If you do not see a failure immediately, wait few seconds and try again. The timeframes and numbers of tries until PS watchdog decides the Policy Server is down are configurable. For more information, see the Where to Configure the PS Watchdog section in this chapter.

How the PS Watchdog works

This section explains how the watchdog works.

The PS Watchdog does not just ping the Policy Server, it performs the following steps:

1. Connects to Policy Server and perform EAC authentication
   (uses the pswd-pers user to it, can be found in the AC local repository)
2. Gets user information (above user)
3. Logs out
4. Closes the connection

Using the above command sequence we make sure that:

- Policy Server is up and listening.
- Policy Server accepts new client requests.
- Policy Server can serve requests
- AC is up and running and can serve requests

How To Change the User Password

The pswd-pers user password is kept in the pswd.dat file on the PS. It can be changed by using the following command: (from bin)

```
PSWD –s <<username>> <<password>>
```

Note: You must update the datastore using the Policy Manager as well. The user that PS Watchdog Service uses must be given the EAC auth method.

eTrust IAM

The features and functions for administration of eTrust Identity and Access Management (eTrust IAM) are built into the IA Manager. This chapter covers some of the tasks and activities you might need to perform after eTrust IAM is installed.

Starting the eTrust IAM Web Applications

The eTrust IAM web applications are installed by the Common Components installer. These applications are:

- eTrust Identity and Access Manager
- eTrust IAM Configuration
- eTrust IAM Self Service
- eTrust IAM SPML Service Configuration

Each of these web applications starts with a login screen which requires a username and password to access them. Most of these applications are typically accessed only by help desk staff and network administrators although the Self Service application may be configured for a wider group of users.

Each of the installed web applications are available via a program shortcut created on the computer where the IAM common components are installed. The shortcuts are located in the Windows Start menu under Programs, Computer Associates, eTrust, eTrust Identity and Access Management.

During rollout of eTrust IAM to your organization, you should implement a method for authorized users to easily access these web applications such as providing a desktop shortcut, browser bookmark, intranet link or Start Menu item on all desktops.

To start one of the web applications, follow these steps:

1. Manually enter the relevant URL into your internet browser:
    1. IA Manager -
    https://<*hostname.company.com*>:<*sslport*>/CA/IAM/Manager/
    2. IAM Configuration -
    https://<*hostname.company.com*>:<*sslport*>/CA/IAM/Config/
    3. IAM Self Service -
    https://<*hostname.company.com*>:<*sslport*>/CA/IAM/SelfService/
    4. IAM SPML Service Configuration -
    http://<*hostname.company.com*>:<*port*>/iamspml

    where:

- <*hostname.company.com*> is the fully qualified hostname of the computer acting as the Web Application Server.
- <*sslport*> is the Apache Tomcat SSL HTTP port number (the default value is 8443).
- <*port*> is the Apache Tomcat HTTP port number (the default value is 8080).

The login prompt appears.

2. Log in by entering the required fields and clicking the Log In button.

User Name
Enter an authorized username, such as etaadmin.

Password
Enter the password for the specified user. The initial password for etaadmin will have been provided when installing the IAM common components.

Admin Server
Required by SPML Service Configuration application only. Enter the name of the Admin Server for the specified user.

Domain
Required by SPML Service Configuration application only. Enter the name of the domain.

The relevant web application interface opens.

IA Manager Manual Configuration

During installation you configure some of the functionality of IA Manager however some parameters must be changed manually after installation.

Note: Always make a backup copy of the configuration file before you modify it.

To manually configure the parameters of IA Manager, follow these steps:

1. Using Windows Explorer, browse to the IA Manager directory. By default this directory is Program Files\CA\eTrust Identity and Access Manager\WEB-INF.

   The contents of this directory appear in the right pane.

   Right-click the etwebadmin.properties file and choose Open With.

   The Open With dialog appears.

2. Select a text editor such as Notepad and click OK.

   The etwebadmin.properties file opens.

3. Change the appropriate configuration values, and then save and close the file.

   Note: Modify only those values that display in the Configuration Parameters list.

   The text editor closes.

4. Restart the Apache Tomcat service in Windows Services.

   The manual changes are applied.

Configuration Parameters

You can modify many default configuration values during the installation of IA Manager. After installation, these configuration values can be changed by editing the configuration file manually.

Note: Parameters, paths, or text strings that contain spaces must be entered in quotation marks ("xxx") preceded by a back slash (\). For example, you could enter \"C:\Program Files\CA\".

During silent mode installation, configuration values with a command line parameter can be set from the command line. However, not all configuration parameters have a command line parameter.

def_search_results
Defines the maximum number of search results to return in the search pane.
Default: 100

eta_domain
Command Line Parameter: ETASERVERDOMAIN
Defines the eTrust Admin Server domain name. This is usually the name of the computer on which eTrust Admin Server is installed, unless changes were made during that installation.
Default: localhost_name

ldap_host
Command Line Parameter: LDAPHOSTNAME
Identifies the computer that serves as the LDAP host. This is the computer on which eTrust Admin Server is installed.
Default: localhost_name

ldap_port
Command Line Parameter: LDAPPORT
Defines the LDAP port number.
Default: 20389

ldap_tlsPort
Command Line Parameter: LDAPTLSPORT
Defines the LDAP port number using TLS/SSL encryption.
Default: 20390

ldap_useTls
Indicates whether to use TLS/SSL encryption. 1 indicates to use TSL/SSL encryption; 0 indicates not to use the encryption.
Default: 1

log_error_detail
Indicates the level of errors logged. Enter a level from 1 to 4.
Default: 4

log_enabled
Indicates whether to enable logging. 1 indicates that logging is enabled; 0 indicates that it is disabled.
Default: 1

Max_Picture_Size_Kilobytes
Defines the maximum file size (in kb) that can be uploaded as a user photograph.
Default: 25

pass_rst_req_enabled
Indicates whether a user can request a password reset. 1 indicates that this option is enabled; 0 indicates that this option is disabled.
Default: 1

pass_rst_req_smtp
Command Line Parameter: SMTPSERVER
Defines the SMTP mail server.
Default: your_smtp_server

pass_rst_req_user
Command Line Parameter: ADMINEMAIL
Defines the administrator's email address.
Default: admin@your_company.com

self_admin_unlock_account_enabled
Indicates whether the administrator can unlock an account. 1 enables this option; 0 disables this option.
Default: 1

web_admin
Defines the user name of the Web application administrator.
Default: etawebad

Note: You can modify the web_admin attribute only if the password for the web_admin user is **changeoninstall**.

Troubleshooting IA Manager

IA Manager has many components which may require separate administrative tasks. This section describes some of the more common administrative tasks that you may perform.

Start or Restart a Windows Service

The Windows services required by IA Manager include:

- Apache Tomcat Service – The default service name is "CA Tomcat 4.1.29 eTrustIAMWebServer".
- Advantage Ingres – The default service name is "Ingres Intelligent Database [ET]".
- Directory Web Server – The default service name is "eTrust Directory Web Server".
- SSO Session Administrator – The default service name is "eTrust SSO Session Administrator".

To start a service running in Microsoft Windows Services, follow these steps:

1. Open the Windows Control Panel from your Start menu. Double-click Administrative Tools and then Services.

   The Services dialog appears.

2. Locate the appropriate service and right-click it.

   A pop-up menu appears.

3. If it is not already started, choose Start from the pop-up menu. If it is already started, choose Restart.

   The service will be listed as Started.

   Note: This service should start automatically after you have installed the IA Manager and also every time you start your machine. Verify that the service is listed as Automatic to ensure that it starts on a computer reboot.

4. Repeat this procedure for any other services, as necessary.

Stop a Windows Service

If a Windows service conflicts with services required by IA Manager, it may be necessary to stop the Windows service.

To stop a service in Microsoft Windows Services, follow these steps:

1. Open the Windows Control Panel from your Start menu. Double-click Administrative Tools and then Services.

   The Services dialog appears.

2. Locate the appropriate service and right-click it.

   A pop-up menu appears.

3. Choose Stop from the pop-up menu.

The selected service stops.

Note: After you stop a conflicting process, you must start/restart the service that was in conflict. For more information, see Start a Windows Service.

Securing your eTrust Web Applications

The eTrust IAM suite of products rely on web applications which are secured using the SSL protocols with self-signed certificates. In order to fully secure your product suite you should replace the self-signed certificates with certificates signed by a certifying authority.

If your organization does not have an internal certifying authority:

1. Use Self Signed host certificates. This is how the default IAM installation is created as it is impossible for the installer to create widely trusted certificates. Although this installation is not optimal, it can be useful for testing. If you wish to install self signed certificates on a web server not installed by the IAM installation see Replace the Tomcat SSL Certificate.

   Using Self Signed certificates has the following drawbacks:

   a. Client computers cannot be assured of the identity of the web servers.

   b. Each time a browser connects to a web server a warning about invalid certificates will be presented. This can confuse users and be a potential source of help desk calls.

2. Use host certificates from a trusted Certifying Authority. This is the recommended option.

   a. Obtain host certificates and private keys for all web server hosts from a trusted Certifying Authority such as Verisign or Thawte.

   b. Install these host certificates and keys in your Tomcat keystore on each web server host. See Replace the Tomcat SSL Certificate.

If your organization already has an internal certifying authority:

1. Issue host certificates and private keys for all web server hosts.
2. Install these host certificates and keys in your Tomcat keystore on each web server host. See Replace the Tomcat SSL Certificate.
3. Install the host certificates in the browser keystore on all client computers.
4. Install the host certificates in the Tomcat keystore on all server computers.

Configure SSL Support for Tomcat

All computers serving web applications for eTrust IAM must have Tomcat configured to use SSL protocol. This is the default when installing Tomcat with the IA Manager software; however, if you installed Tomcat independently, it may not be configured with SSL support.

To install and configure SSL support for Tomcat using a self-signed certificate, follow these steps:

1. Verify that JDK version 1.4.2_04 is installed by checking the Add/Remove Programs list in your Control Panel for the program Java 2 SDK, SE v 1.4.2_04.
2. Create a new keystore containing one self-signed certificate by entering the appropriate command from the command prompt.

   On Windows systems, you should enter:
   ```
   %JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA -keystore
   \path\keystore_filename
   ```

   The keystore creation process begins.
3. Enter the keystore password when prompted.

   Note: The default password used by Tomcat is **changeit** (all lower case). If preferred, you can specify a custom password, but you must then specify the custom password in the server.xml configuration file also (see Step 8).

   The keystore creation process continues.
4. Enter general information for the certificate when prompted. The general information includes company, contact name, and so on. This information displays to users who attempt to access a secure page in your application, so make sure that the information provided here is appropriate.

   The keystore creation process continues.
5. Enter the key password when prompted. This password is created specifically for this certificate (as opposed to any other certificates stored in the same keystore file). You must use the same password for this and the keystore password.

   A keystore file with a certificate that your server can use is created.
6. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default location for the <*Tomcat_installation_directory*> when installed from the IA Manager software is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29.
7. Ensure that the SSL Coyote HTTP/1.1 Connector entry is not commented out in the file. The connector information looks similar to the following:
   ```
   <!-- Define an SSL HTTP/1.1 Connector on port 8443 -->
   <!--
   ```

```
<Connector className="org.apache.catalina.connector.http.HttpConnector"
      port="8443" minProcessors="5" maxProcessors="75"
      enableLookups="true" acceptCount="10" debug="0" scheme="https"
      secure="true">
 <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
      clientAuth="false" protocol="TLS"/>
</Connector>
-->
```

If the Connector element is commented out, you must remove the comment tags (<!-- and -->) around it.

8. Configure the SSL Coyote HTTP/1.1 Connector entry to include the keystoreFile and keystorePass attributes for the Factory element.

   keystoreFile
   Specifies the location where the keystore file is located

   keystorePass
   Specifies the keystore (and certificate) password

   The connector information should look similar to the following:

```
<Connector className="org.apache.catalina.connector.http.HttpConnector"
      port="8443" minProcessors="5" maxProcessors="75"
      enableLookups="true" acceptCount="10" debug="0" scheme="https"
      secure="true">
 <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
      keystoreFile="your_keystore_full_path"
      keystorePass="your_keystore_password"
      clientAuth="false" protocol="TLS"/>
</Connector>
```

9. Save the file and close it.

   SSL support and self-signed certificates are configured for Tomcat.

For more information or to obtain and install a certificate from a certificate authority (such as verisign.com, thawte.com or trustcenter.de), see the Apache Tomcat document "SSL Config HOW-TO", available at http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html (http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html).

Replace the Tomcat SSL Certificate

Your Tomcat SSL Certificate enables users browsing to your web pages to transmit and receive secure information. Certificates may be self-signed or provided by an independent third party. You can update your Tomcat SSL Certificate to improve the security of the information shared via web browsers.

To replace the Certificate for Tomcat SSL support, follow these steps:

1. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default location for this file is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29\conf\server.xml.

The text file opens.

2. Search for the following text: keystoreFile. Change the attributes to present the replacement information.

   keystoreFile
   Specifies the location in which the keystore file is located.

   keystorePass
   Specifies the keystore (and certificate) password.

   The file reflects these changes.

3. Save the file and close it.

   The information is stored.

Note: For more information, see the Apache Tomcat document "SSL Config HOW-TO", available at http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html (http://jakarta.apache.org/tomcat/tomcat-4.1-doc/ssl-howto.html).

Resolving Port Conflicts Manually

This topic is included for resolving Apache Tomcat port conflicts. For information about how to resolve Apache Tomcat port conflicts not covered here, see your Apache Tomcat documentation or the Apache Tomcat http://www.jakarta.apache.org/tomcat web site.

If you cannot run a particular installation of Apache Tomcat, one or more of the ports for which it is configured may currently be in use by another program (for example, another installation of Tomcat or some other web server). To rectify this problem, you must reconfigure the "broken" Tomcat to use different values for the Shutdown port, the Non-SSL HTTP port, and the SSL HTTP port.

To reconfigure the port values, follow these steps:

1. Browse to the <*Tomcat_installation_directory*>\conf\ directory and open the server.xml file in a text editor.

   The default full path for this file is C:\Program Files\CA\SharedComponents\Tomcat\4.1.29\conf\server.xml.

2. Search for the following text: Server port=.

   The text is highlighted.

3. Change the value that appears after Server port =. For example, if the number is 8005, change it to 9005.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's Shutdown port.

4. Now, search for the following text: Define a non-SSL Coyote HTTP.

   The text is highlighted.

5. From this position in the file, search for the following text: port=.

   The text is highlighted.

6. Change the value that appears after port=. For example, if the number is 8080, change it to 9080.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's non-SSL HTTP port.

7. Now, search for the following text: Define a SSL Coyote HTTP.

   The text is highlighted.

8. From this position in the file, search for the following text: port=.

   The text is highlighted.

9. Change the value that appears after port=. For example, if the number is 8443, change it to 9443.

   Note: The new port number must be greater than 1024 and less than 41152, and must not be in use by any other program on your computer.

   The number you entered now appears as the new port number for Tomcat's SSL HTTP port.

10. Return to the beginning of the file and search for all instances of the following text: redirectPort=. For each instance, change the value that appears after redirectPort= to the same number that you entered for the new port number for Tomcat's SSL HTTP port.

    Note: Tomcat's SSL HTTP Port and Redirect port must use the same number.

11. Save the file, close it, and try to run this particular installation of Apache Tomcat again. Verify that Tomcat started correctly by entering the following URL in your web browser: http://localhost:8080/index.jsp. If you see the Apache Tomcat web page, it has started correctly. If not, then Tomcat has not started.

    If Tomcat starts correctly, you have successfully resolved your Tomcat port conflicts.

Note: For more information about how to start Tomcat, see Start Apache Tomcat or see your Apache Tomcat documentation

Back Up and Restore Data

Backing Up eTrust Access Control Data

The dbmgr -backup function creates an online backup of the eTrust AC database in the specified directory. This function is available whether the eTrust AC services are running or not.

The backup directory cannot be located on a remote machine; if the directory does not exist, this dbmgr -backup option creates it.

To backup the eTrust AC data on Windows, follow these steps:

1. Open a command line prompt.
2. Type: dbmgr -backup <backup directory>

Backup eTrust Directory

Back Up eTrust Directory Data

To back up the eTrust Directory data on either **Windows or UNIX**, follow these steps:

Note: You must be a DSA user to perform these functions on a UNIX computer. You can log in as a DSA user by typing `su - dsa`

1. Open a command line
2. Backup all data by typing:

   ```
   dxdumpdb -p "o=PS" ps > <FILENAME>
   ```

   This will create an LDIF format record of the eTrust Directory information <FILENAME> which can be loaded into the other eTrust Directory data stores.

   Note: Remember where you saved the file and what you called it.

Restore eTrust Access Control

Restoring and Replicating eTrust Access Control Data

To restore eTrust AC data on Windows after you have backed it up using dbmgr -backup follow these steps:

1. Stop eTrust AC services by typing: secons -s

2. Copy backed up data by typing: copy <backup directory> <eTrust AC install directory>/data/seosdb
3. Re-start the eTrust AC services by typing: seosd –start

Restore eTrust Directory.

Restoring and Replicating eTrust Directory Data

To restore eTrust Directory data after you have backed it up, follow these steps:

Note: You must be a DSA user to perform these functions on a UNIX computer. You can log in as a DSA user by typing `su – dsa`

1. From the command line, make sure eTrust Directory is running by typing: `dxserver status`
2. Execute the command:

   ```
   %dxmodify -a -c -h localhost:<PORT> -D "cn=<eTrust Directory
   administrator>,o=PS" -w <User_Password> -f <FILENAME>
   ```

   The default port is 13389.

   You must use the user and password of the eTrust Directory administrator user (ldap-admin by default). You entered this username and password during installation.

   Note: You may see errors for records that already exist in the database. You must manually reconcile these duplications.
3. Restart eTrust Directory.

WARNING! *Make sure that all servers are started and reachable, as the Directory queue may overflow if some servers are offline. This is because replication updates need to be stored in memory.*

Chapter 19

Auditing and Logging

This chapter describes how logging and auditing work with eTrust SSO.

Logging

Logging is useful for two reasons:

- Some logging is required in order to use auditing.
- Logging can help you diagnose problems with the eTrust SSO system.

Note: You should only use logging for troubleshooting. If you leave logging on all the time, it will slow down the system, and use up disk space.

Logging for Windows Installations

To set up logging for installations you must use the Windows Registry on the target machine.

If you enable logging for Windows installations, four log files will be created for each eTrust 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**. Use the Logging Mode Settings table.

5. To check the results of the logging, go to the Temp directory.

   To find out where your Temp directory is, go to the command line and type **echo %temp%.**

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".

Logging for the SSO Client, the GINA, and the Authentication Agents

Logging records the date, time and level of each eligible event. The logs can be used to work out what is going wrong.

Configuring Logging

You can set up logging for the SSO Client, the GINA, the SSO Interpreter and the authentication agents in the same way. Each of these components includes a logging configuration file (*.cfg) that specifies the location of the log file, and the level of logging.

For the SSO Client, the SsoClnt.ini file specifies the location of the logging configuration file. By default, the logs are created in the installation directory. You can change this location by editing the ClientConfigFile keyname in the **Logging** section of the SsoClnt.ini file.

For logging to a file to be successful for the SSO Client, the user that is logged on must have write access to the area that the log file is being written to. If the user doesn't have the correct access, move the logfile to a dir that they do have access to or change their access privileges.

Levels of Logging

There are five levels severity that can be logged for the SSO Client, the GINA, and the authentication agents. Each level includes the logging of the level below it, as well as extra information:

DEBUG – Logs information used by eTrust 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 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 SSO Client is able to keep running.

FATAL – Logs fatal level events which cause the SSO Client to shut down.

Logging for the Policy Server

By default, Policy Server logging is enabled.

On Windows, the Policy Server logging is enabled in the log.ini file, which installed in the root directory.

On UNIX, the Policy Server logging is enabled in the policyserver.ini file, which installed in the root directory.

The Windows and UNIX configuration files each contain the following two lines:

MST_AUDIT—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.

The location of the Policy server log file is set in LoggerIniFile keyname, in the main section of the policyserver.ini file. By default, the log files are written to the **logs** directory.

Logging for Session Management

There are two kinds of log file for the Session Administrator:

- The Session Administrator's communications with the Policy Server
- The Session Administrator's inner workings

Also, you can read the logs of the Tomcat server. These logs are written to the **CATALINA_HOME\logs** directory.

Note the use of the double back-slashes in the following instructions.

For more information, see the "Session Management" chapter in the eTrust SSO Implementation Guide.

To Change the Location of the Communication Log File

1. Find the following file:

   ```
   %CATALINA_HOME%\webapps\SessionAdministrator\log\log4c_config.cfg:
   ```

2. In the log4c_config.cfg file, find the following line:

   ```
   appender root pattern file \
   ${CATALINA_HOME}\\webapps\\SessionAdministrator\\log\\SessionMgtGUI_C.log \
   %d %p %c [%x] - %m%n
   ```

3. Change the path. For example, you could change it to:

   ```
   c:\\mydir\\logfiles\\mylogfile.txt
   ```

To Change the Location of the Session Administrator Log File

1. Open the following file:

   ```
   %CATALINA_HOME%\webapps\SessionAdministrator\log\log4j_config.lcf
   ```

2. Find the following line:

   ```
   log4j.appender.R.File=${catalina.home}\\webapps\\SessionAdministrator\\log\\Sess
   ionMgtGUI_J.log
   ```

3. Change the line to refer to a different file location. For example:

   ```
   log4j.appender.R.File=c:\\mylogdir\\mylogfile.txt
   ```

Logging for the Password Synchronization Agent

The Password Synchronization Agent writes to the Windows Event Viewer. This is not configurable.

Logging for eTrust Directory

eTrust Directory includes a number of log files, most of which are very detailed. These files are written to the dxserver\logs directory.

Check the following two log files in particular, where *machinename* is the name of the Policy Server computer:

- pstd_*machinename*.log
- ps_*machinename*.log

For more information about logging for eTrust Directory, see the "Monitoring the Directory" chapter, and the "Messages and Logs" appendix in the eTrust Directory Administrator Guide.

Auditing Access Control

Auditing allows you to track the everyday events in eTrust SSO.

Security auditing allows you to examine the events that have occurred in eTrust SSO for review and assessment of previous and potential threats to, or violations of, your eCommerce environment. With eTrust 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.

This section explains how to interpret the audit events produced by the different components of eTrust SSO and how to work with the audit output.

Access Control Auditing Events

eTrust Access Control logs two types of audit events :

- **Administration events**—An audit record is created for each operation done from the Policy Manager or selang commands.
- **SSO events**—eTrust SSO creates audit records for events in which a user has used the 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)

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 eTrust SSO audit records:

- Logon to eTrust SSO
- Request for an application list
- Request for logon variables

Logon to eTrust SSO (SSO_LOGIN)

This record reports a logon to SSO event. For example: A user makes SSO Authentication, and 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 Policy Server database.

Examples:

The end user starts the 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 eTrust SSO Tools window, which leads the Policy 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
```

Audit Tools

A number of audit tools can be used for auditing the events, including eTrust Audit, and seaudit.

Security Auditing (seaudit)

The seaudit module is the eTrust Access Control audit file viewer. It is a command-line interface that allows you to filter out audit events.

The seaudit module is supplied with eTrust SSO. For more information about seaudit, see *eTrust Access Control Administrator Guide*.

eTrust Audit for eTrust SSO

eTrust Audit is an audit collection and alerting tool. This tool can be used to process events and information generated by eTrust SSO. eTrust Audit is not supplied with eTrust SSO. You can purchase it separately.

Auditing the Policy Server and the Web Agent

The type of audit event produced varies depending on the eTrust SSO component that produced it. The following sections describe the types of audit events produced by each eTrust SSO component.

Audit Events Produced by the Policy Server

The audit events produced by the Policy Server can be divided into the following categories:

System events—System events include successful or unsuccessful starting of the Policy Server, stopping the Policy 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.

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."

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.

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 Policy Server and Web Agent components each have a configuration file that controls which events are collected and the output format and destination of the collected events. Using the configuration files, you can determine how much information is collected, which components of eTrust SSO the information is collected from, and where the collected information is saved.

The name of the Policy Server auditing configuration file is pslog.ini in Windows and log.ini in UNIX. The name of the Web Agent auditing configuration file is webagentlog.ini. Each configuration file is located in the installation path of its component. The following sections discuss the contents of each configuration file.

Using the Policy Server Auditing Configuration File

The Policy Server auditing configuration file (called pslog.ini in Windows and log.ini in UNIX) contains parameters that eTrust SSO needs to process log messages generated by the Policy Server. The pslog.ini file is stored in the installation path of the Policy Server; the default path is:

```
Program Files/CA/eTrustPolicyServer/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 Policy Server. After you finish editing, start the Policy Server.*

The lines containing parameters for a particular process or utility are grouped together as a section. The pslog.ini files contains 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. See the Working with Audit Output section for a list of parameter values and additional parameters for each output destination.

The Clog Settings Section

The [Clog Settings] section is used to gather information about events that occur on the Policy Server or identify problems a Policy Server may be having. Either situation may require redirecting log messages to a logging utility. This section redirects logs generated by the Policy 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 DBMontiorDLL NTEventLogDLL LogFileDLL	The paths to these DLL files are set during installation.

The Audit Section

The [Audit] section identifies the destination for audit messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output.

The Log Section

The [Log] section identifies the destination for log messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output.

The Trace Section

The [Trace] section identifies the destination for trace messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output.

The Filters Section

The [filter] section specifies 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. For information about setting filters, see Filtering Events later in this chapter.

Using the Web Agent Auditing Configuration File

The Web Agent auditing configuration file, called webagentlog.ini, contains parameters that eTrust 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 contains 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 Web Agent or identify problems a Web Agent may be having. Either situation may require redirecting log messages to a logging utility. This section redirects logs generated by the Web Agent 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 DBMontiorDLL NTEventLogDLL LogFileDLL	The paths to these DLL files are set during installation.

The Audit Section

The [Audit] section identifies the destination for audit messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output.

The Log Section

The [Log] section identifies the destination for log messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output

The Trace Section

The [Trace] section identifies the destination for trace messages and the name of the file that contains them. For a description of each parameter, see the section Working with Audit Output

The Filters Section

The [filter] section specifies 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. For information about setting filters, see Filtering Events later in this chapter.

Audit Output

eTrust SSO gives you the ability to configure the audit output that was generated, and then specify where the output should be sent. Audit output consists of three types of messages generated by eTrust 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.

Specifying an Output Destination

eTrust SSO supports several destinations where you can have your audit output delivered:

- eTrust 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 eTrust 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.

Configuring the Configuration File to Use eTrust Audit

eTrust Audit, part of the Computer Associate eTrust Defense, Access, and Management solutions, 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 eTrust products and stores it in a central database for easy access and reporting. Administrators use eTrust Audit for monitoring, alerting, and reporting information about user activity across platforms.

To write events as eTrust Audit output, you need to have an eTrust Audit client installed on your network. You also need an eTrust Audit server installed either on the same machine where eTrust SSO is installed, or in the network where your eTrust Audit client can communicate with it. Then, to set eTrust 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 eTrust Audit Router. This key is required.

Port— The port number the eTrust Audit Router is listening to. This key is optional.

Timeout— How long to wait for eTrust Audit Router to reply before timing out, specified in seconds. This key is optional.

Here is an example of the Audit section that sets eTrust Audit as the output destination of your auditing events:

```
[Audit]
Output = eTAudit
Router = 123.232.233.232
Port = 8240
Timeout = 5
```

Refer to the *eTrust Audit Administrator Guide* to set up the communication for the eTrust Audit client and server. You can then use the eTrust Audit Security Monitor to view your audit output.

Configuring 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 eTrust 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
```

Configuring the Configuration File to Use the Event Viewer

Event Viewer is a tool you can use to monitor events in your Windows NT or Windows 2000 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
```

The Policy Server provides logging facilities that can be configured using settings in the pslog.ini file located in the Policy Server install directory.

The maximum size in KB of any log files is defined by a configuration property in the pslog.ini file called MaxFileSize. When the log file gets to this 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 that <log>.log files reaches the maximum size. When a new file is created is the previous <log.log> is renamed <log>.000 and the previous <log>.000 is renamed one increment 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.

The maximum number of "historical" files is defined by a configuration property in the pslog.ini file called MaxNumberOfHistFiles. When MaxNumberOfHistFiles is reached, the oldest file (i.e. the last one in the "chain") is deleted and the sequential numbering defined previously is maintained.

Filtering Events

eTrust 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.

Policy Server—To enable Policy 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]
[*.*.*]
```

Generating Reports

If you installed the reporting facility provided with eTrust SSO, you can generate reports from the events collected in the log file.

Note: There are certain modifications you must make to support the reporting facility when setting up the configuration file to send events to the log file. These modifications are explained in Configuring the Configuration File to Use the Log File earlier in this chapter.

The reporting facility of eTrust SSO extracts event information from audit message types and generates summary reports of statistics. The reporting facility requires the following settings in the Audit section of the configuration file:

```
Output = LogFile
MessageFormat = %T, %D - %M\n
```

Specify the FileName of your choice. It is also recommended that you set the MaxNumberOfHistFiles and MaxFileSize parameters to capture events over a period that is long enough to let you regularly assess whether any violations or threats have occurred. You need to enable AUDIT events in your filter by specifying either [*.MST_AUDIT.*] or [*.*.*].

Starting and Stopping the Reporting Facility

To start the reporting facility when using **Windows**, first verify that Tomcat is running. If Tomcat is not running, take **one** of the following actions:

- If you do not have Tomcat installed as a service, start Tomcat from the Start button by choosing Programs, Apache Tomcat 4.0, Start Tomcat.
- If you have Tomcat installed as a service, open Control Panel and start the Tomcat service.

After Tomcat is running, you can start the reporting facility from the Start button by choosing Programs, Computer Associates, eTrust, Single Sign-On, Reporting. The report containing summary statistics displays by default. To stop the reporting facility, close the web browser containing the report, and then stop Tomcat.

- If you do not have Tomcat installed as a service, stop Tomcat from the Start button by choosing Programs, Apache Tomcat 4.0, Stop Tomcat.
- If you have Tomcat installed as a service, open Control Panel and stop the Tomcat service.

To start the reporting facility when using **UNIX**, run the start_report.sh script located in the bin directory where reporting is installed. After reporting starts, you will be shown the URL used for reporting and asked to use your web browser to view reports at that URL. To stop the reporting facility, run the stop_report.sh script located in the bin directory where reporting is installed. You will be notified when the report facility stops.

Important! *You must restart the reporting facility each time you restart your system.*

Accessing Reports Without a Web Server

If you do not have a web server or need to access reports from another machine, you can access the reports through Tomcat Version 4.0 using this URL:

```
http:your_machine_name:8080/sso/servlet/SSOIndex?waclogIniPath=your_pslog.ini_path
```

This URL assumes that you are using port 8080 for Tomcat.

Generating Reports

You can generate reports by selecting them from the drop-down list. The reports you can generate are:

- Summary of Authentication and Authorization
- List of Users Created
- List of Users Deleted
- List of Users Updated
- List of Resources Denied Authorization
- List of Users Passed Authentication
- List of Users Failed Authentication

Reports are displayed using your web browser.

You can generate any report for a specific time period. The time period begins with the month, day, year, hour, minute, and second you enter at the top of the report and ends with the current date and time. You can update the data on the report at any time by clicking the Refresh Report button.

Note: The content and format of the report cannot be customized.

The amount of data kept and how long it is kept are controlled by the MaxFileSize and MaxNumberOfHistFiles parameters in the pslog.ini file. Be sure to assign values to these parameters that will accommodate your needs.

Troubleshooting Report Problems

Occasionally you may have a problem displaying a report or displaying particular information within a report. If you happen to encounter these problems, try the following solutions.

- If you cannot get a report to display in your browser after clicking on the reporting link, first verify that Tomcat is successfully installed on your machine. You can do this by starting Tomcat.
  - If you do not have Tomcat installed as a service, start Tomcat from the Start button by choosing Programs, Apache Tomcat 4.0, Start Tomcat.
  - If you have Tomcat installed as a service, open Control Panel and start the Tomcat service.
- If Tomcat is started, then verify that the webac directory and webac.war file exist in the webapps directory located in your Tomcat installation path. If the webac directory does not exist, stop and restart the Tomcat server to create the webac directory. If the webac.war file is missing, reinstall the report facility.
- **Note:** Advanced users can copy the webac.war file from the installation CD to the webapp directory.
- Finally, verify that the port number of the report URL matches the Tomcat port number. The default port number is 8080; you may have changed it when you installed Tomcat.
- If the report displays after clicking on the reporting link but no detail information about events appears within the report, try the following solutions.

Solution 1

- Locate the pslog.ini or log.ini file in the PolicyServer directory in the installation path for eTrust SSO and verify that the appropriate modifications have been made to the Audit and filters sections. In the Audit section, you must see the following parameters and values:

```
Output=Logfile
FileName=<Your_Logfile_Name>
MessageFormat=%T, %D - %M\n
```

- In the filters section, you must see the following line:

```
[*.MST_AUDIT.*]
```

Solution 2

- Stop and restart the Policy Server service.

Solution 3

- Locate the eSSOReport.url file in the PolicyServer directory in the installation path for eTrust SSO and verify that the correct port is listed. The port listed after localhost in the URL must be the port that the Tomcat server is using. The default port is 8080.

Appendix

A

Configuring the SSO Client: SsoClnt.ini

The SsoClnt.ini file contains values used by the SSO Client. The SSO Client is 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 SSO Client.

This chapter describes the sections and keynames within the SsoClnt.ini file. The sections and keynames in this chapter correspond to the sections within the SsoClnt.ini file.

Location of the SsoClnt.ini File

For local installations, place the SsoClnt.ini file in the same directory as the SSO Client executables, on the user's machine.

For network installations, place the SsoClnt.ini file in the same directory as the SSO Client executable. You can install more than one SSO Client executable on the same workstation, but each must be in a separate directory with its own SsoClnt.ini file.

Formatting Rules

Certain formatting rules apply to all values in the SsoClnt.ini file. This table explains those rules.

Circumstance	Formatting Rule
Separating values	Separate values with commas, spaces, or tabs.
Pathways or strings with spaces	Use quotation marks for any path or string that contains spaces. For example, "Press Ctrl + Alt + Del to logon."
Multiple values	If the first value fails, the second value is used instead and so on through the list.

Sections in the SsoClnt.ini File

This table lists all the sections of the SsoClnt.ini file together with a brief description about each section. Each section of the SsoClnt.ini file contains one or more keynames (also called tokens or settings) that affect the behavior of the SSO Client.

Section of INI File	Description
ServerSet0	Values for the servers sets (you may have multiple ServerSet sections, for example ServerSet0, ServerSet1, ServerSet2…)
sso	Values for Policy Server and SSO Client configuration information
GINA	Values for the eTrust SSO GINA
Logging	Values for Information about logging functionality
GlobalIni	Values for a centralized SsoClnt.ini file
auth.NT	Values for Windows primary authentication
auth.NOVELL	Values for Novell NetWare primary authentication
auth.ENTS	Values for Entrust primary authentication
auth.CERT	Values for Cert primary authentication
comm.	Values for Communication parameters
SessionManagement	Values for SSO session configuration
StationLock	Values that configure the station lock option
TrayMenu	Values that show or remove specific menu items
Tools	Values that manage the SSO tools options
SystemLogon	Values for SSO authentication (for Windows using the Network Provider) and for Windows NT GINA upgrade
SSO Interpreter	Values for configuring the appearance and operation of the SSO Client
HLLAPI	Values that control HLLAPI emulation options (Win clients)
ToolBar	Values for configuring the SSO toolbar
MetaframeMigration	Values for configuring Citrix Metaframe Application migration.
EventCommands	Values for scripts that run during the SSO Client events
AppListRefresh	Values for configuring automatic application refresh

ServerSet0

ServerSets extend the fault tolerance and failover of eTrust SSO Client during its interaction with the Policy Server and the authentication hosts. Server sets group related server data together. You can create multiple ServerSets starting with ServerSet0, then ServerSet1, ServerSet2 etc. ServerSet0 is mandatory, subsequent ServerSets are optional.

Note: For ServerSets to function correctly, you must have all Policy Servers set up in a server farm configuration and the token directory data must be replicated across all servers. For more information about setting up a server farm, see the "Implementing a Server Farm" chapter in the *eTrust SSO Implementation Guide*.

Keyname	Description	Values
Name	Specify 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".	Define: Server Set name Default: [None]
FailoverInterval	Specify the time in minutes that should elapse before the system should try to reconnect to a server or authentication host that was marked as 'offline' on a previous try. **Note:** You can not set this value to less than five minutes. If you set the value to less than five, it will automatically default to five minutes.	Define: Time in minutes Default: 30
PolicyServers	Specify the list of SSO Policy Servers. Use spaces to separate. When an SSO Client attempts to connect to a Policy Server, it tries to use the first server in the list. If that server is not available, the second server in the list is used. For example: server1 server2	Define: Server name(s) Default: Name entered during installation

Keyname	Description	Values
AuthMethods	Specify the list of authentication methods to be available to users. The first value in the Authentication dialog 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 AuthNT, AuthNOVELL and AuthCITRIX methods that will allow the use of the keyword <auto>. The <auto> keyword signifies to locate the nearest domain controller, locate the nearest NOVELL Netware server or nearest Citrix server respectively. For example: AuthMethods=LDAP AuthLDAP=AuthServer1 AuthServer2	Define: Authentication name Default: [See description]
AuthSSO	Specify the list of the authentication hosts to use for eTrust SSO native authentication. Space separated. There is no default value for AuthSSO 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.	Define: Server name(s) Default: [See description]
AuthCERT	Specify the list of the authentication hosts to use for Cert authentication. Space separated. There is no default value for AuthCERT 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.	Define: Server name(s) Default: [See description]
AuthLDAP	Specify the list of the authentication hosts to use for LDAP authentication. Space separated. 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.	Define: Server name(s) Default: [See description]

Keyname	Description	Values
AuthRSA	Specify the list of the authentication hosts to use for RSA SecureID authentication (previously referred to as SDI authentication). Space separated. 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.	Define: Server name(s) Default: [See description]
AuthNT	Specify the list of the authentication hosts to use for Windows authentication. Space separated. There is no default value for AuthNT 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.	Define: Server name(s) Default: [See description]
AuthNOVELL	Specify the list of the authentication hosts to use for Novell authentication. Space separated. There is no default value for AuthNOVELL 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.	Define: Server name(s) Default: [See description]
AuthENTS	Specify the list of the authentication hosts to use for ENTS authentication. Space separated. There is no default value for AuthENTS 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.	Define: Server name(s) Default: [See description]
AuthSWEC	Specify the list of the authentication hosts to use for SWEC authentication. Space separated. There is no default value for AuthSWEC 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.	Define: Server name(s) Default: [See description]

Keyname	Description	Values
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 workstation and for that ticket to be used automatically by the SSO Client on the Citrix Server.	Define: Citrix server name(s) Default: <auto>

SSO

Keyname	Description	Values
SSOFolderName	Specify the name of the folder in the Start menu that displays the shortcuts to the SSO-supported applications.	Define: Item name in the Start menu Default: "SSO Programs"
MergeLinks	If the folder specified by the SSOFolderName already exists in the Start menu, do you want to add the SSO shortcuts to the existing folder, or do you want to replace the contents of the folder? For example, if you want the SSO shortcuts to appear in the "Programs" folder on the user's Start menu, set: SSOFolderName = Programs MergeLinks = yes **Note:** If you rename the SSOFolderName an existing folder name, and **do not** set MergeLinks to yes the SSO shortcuts will **replace** what is in that folder.	Define: [yes|no] Default: No
BuildLinks	Do you want to build an SSO submenu in the Windows Start menu?	Define: [yes|no] Default: Yes
CleanLinksOnExit	Do you want to delete the SSO Programs folder in the Windows Start menu (and all the links it contains) when the SSO Client is shut down?	Define: [yes|no] Default: No
CleanTicketOnStart	Do you want to delete the SSO ticket from the cache when the SSO Client starts? **Note:** This must be set to "no" if you are using Session Management and the SSO GINA.	Define: [yes|no] Default: no

Keyname	Description	Values
CleanTicketOnExit	Do you want to delete the SSO ticket from the cache for all users when the SSO Client exits?	Define: [yes|no] Default: Yes
ExplorerOpenFolder	Do you want to be able to open container applications in Windows Explorer?	Define: [yes|no] Default: Yes
ShowLastName	Do you want the last logon name to display in the Authentication dialog?	Define: [yes|no] Default: Yes
AuthTimeOut	Specify the time in seconds that the authentication dialog remains open.	Define: Time in seconds Default: 250
DefaultAppIcon	Specify the path to the executable or DLL containing the default icon to use for SSO applications.	Define: Pathway Default: [None]
DefaultIconOrder	Specify the reference for the location of the default icon in the file referenced in the DefaultAppIcon token.	Define: Ordinal value of the icon Default: 0
DomainNameServer	Specify the internet domain name of the site on which the web server and SSO web agent is running. For example: http://machine1.ca.com	Define: Pathway Default: [None]
DisableShutdown	Do you want to disable the Shutdown button? The Shutdown button appears on the 'NT Logon' page of the SSO GINA 'Windows Logon' dialog and on the Secure Information dialog (when the user logs on and presses Ctrl + Alt + Del using the SSO GINA only).	Define: [yes|no] Default: No
EngineRegisterTimeOut	Specify the time in seconds the SSO Client should try to register the Client Engine. If time expires, the SSO Client closes.	Define: Time in seconds Default: 60
SysErrorMessage	Specify the message that displays when errors occur. This message appends to the system generated error message.	Define: Error message Default: "Contact your eTrust SSO Administrator."
DataDirectory	Specify the directory where the SSO Client can store user data files. This directory location is always "\data" in the SSO Client install directory. This can be changed manually after installation.	Define: Pathway Default: "[client installation directory]\data"

GINA

Keyname	Description	Values
LogonBitmap	Specify 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 will look for the image SsoLogon.bmp in the system32 directory. If this second value is not found the GINA will use a default bitmap which says "Welcome to eTrust SSO. Press Ctrl +Alt + Del to logon."	Define: pathway and image Default: [See description]
LogonTitle	Specify the title for the SSO GINA's logon window. If not specified, the default value is "Windows Logon".	Define: Dialog title Default: "Windows Logon"
LockedBitmap	Specify the name (including 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 will look for the image SsoLocked.bmp in the system32 directory. If the GINA cannot load the SsoLocked.bmp, it uses a default bitmap.	Define: pathway and image Default: [None]
LockedTitle	Specify the title for the SSO GINA's 'Station locked' window. If not specified the default value of "Workstation Locked" will be used.	Define: Dialog title Default: "Workstation Locked"
GinaPassThrough	Do you want to chain to the next GINA during logon? **Note:** GinaPassThrough is currently only supported for UnlockStationMode 3.	Value: [yes|no] Default: No
LogonCAD	Specify whether to display the Ctrl + Alt + Del bitmap at logon. 0 Always show bitmap 1 Never show bitmap 2 Let Operating System decide	Define: [0|1|2] Default: 2

Logging

Keyname	Description	Values
ClientConfigFile	Specify the location of the SSO Client logging configuration files. If the value is left blank, logging is disabled. If the keyname is commented out, the default value SsoClientLog.cfg is used. Unless otherwise specified, this is created in the installation directory. For example: "c:\<client dir>\SsoClientLog.cfg"	Define: Pathway Default: [See description]
GinaConfigFile	Specify the location of the GINA configuration files. If the value is left blank, logging is disabled. If the keyname is commented out, the default value SsoGinaLog.cfg is used. Unless otherwise specified, this is created in the installation directory. For example: "c:\<client dir>\SsoGinaLog.cfg"	Define: Pathway Default: [See description]

GlobalIni

This section allows you to define a centralized ssoclnt.ini file that can be used by many workstations. On startup of the SSO Client, if the UseGlobalIniFile keyname is set to yes, then a centralized ssoclnt.ini file will be "copied" down automatically to the local workstation when the SSO Client starts up if the local SsoClnt.ini file is deemed to have expired.

Keyname	Description	Values
UseGlobalIniFile	Do you want to copy the SsoClnt.ini file from the network to the local computer when the SSO Client is started? UseGlobalInifile must be set to yes in both the initial ssoclnt.ini on the local workstation and in the "copied" down ssoclnt.ini if you want to use a centralized ssoclnt.ini file. If UseGlobalInifile is set to no, the local SsoClnt.ini file is used, as per standard SSO Client operation, and all values are read from this local version.	Define: [yes|no] Default: No

Keyname	Description	Values
GlobalIniCachingTime	Specify the time after which the "copied" SsoClnt.ini file expires. This is checked each time the SSO Client is started. If it has expired, the SSO Client will try to download a new SsoClnt.ini file from the network location specified in the GlobalIniFile keyname. We recommend that you keep the initial workstation SsoClnt.ini file default of 00:00:00 so that the first time the SSO Client is started on the workstation it will "copy" down the centralized SsoClnt.ini file immediately. The centralized SsoClnt.ini files may have a different GlobalIniCachingTime from the initial workstation SsoClnt.ini file, to allow periodic checking of the centralized SsoClnt.ini file and not every time the SSO Client is started. For example: 06:00:00 indicates that the first time the SSO Client is started after six days from the last time the SsoClnt.ini file was downloaded, the new one will be copied from the network on to the local machine. If an attempted copy fails (for whatever reason), the previous local version will be used by the SsoClnt and an appropriate message will be appended to the SSO Client log file.	Define: [dd:hh:mm] [days:hours:minutes] Default: 00:00:00
GlobalIniFile	Specify the location of the centralized SsoClnt.ini file that will be "copied" to the local workstation. This must define the full path to the centralized file and include the name of the SsoClnt.ini file. The full path can include the names of mapped drives and also UNC names. For example: \\centralmachine\ssopath\ssoclnt.ini	Define: Pathway Default: [none]

auth.NT

Keyname	Description	Values
AutoNetworkAuth	Do you want 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 recommended that the administrator adds a logoff script to the authenticated user out of the network.	Define: [yes|no] Default: No
NearestDomainController	Do you want the SSO Client to try to authenticate to the nearest Domain Controller(DC)? Yes = Always try to connect to the nearest DC on the network regardless of the target OS (Active Directory architecture). No = Always try to connect to the Primary Domain Controller (PDC) specified. <auto>= Check the operating system and behave accordingly: ▪ Win2K or later workstation= try to connect to the nearest DC (presume Active Directory architecture) and if this fails, connect to the PDC (presume NT4 architecture). ▪ Win9x or NT4 = connect to the PDC, regardless of location (presume NT4 architecture).	Define: [yes|no|<auto>] Default: <auto>

auth.NOVELL

Keyname	Description	Values
ncpfromFile	Specify whether to use Novell Netware core protocol should be used for requesting information from Netware Server. 0 Novell Netware core protocol is not used 1 Novell Netware core protocol is used	Define: [0|1] Default: 0
AutoNetworkAuth	Do you want to let the Novell 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 administrator add a logoff script to log the authenticated user.	Define: [yes|no] Default: No

auth.ENTS

Keyname	Description	Values
EntrustInifile	Specify the path to the Entrust.ini file.	Define: Pathway Default: [None]
EntrustProfile	Specify the path to Entrust Profile (.epf file).	Define: Pathway Default: [None]

auth.CERT

Keyname	Description	Values
CertStore	Specify the type of storage for the user certificate. FILE = Stores user certificate in a local disk file PKS11 = Stores the user certificate on a PKCS#11 token (smart card in most cases). You can specify both storage methods, for example: certStore=PKCS11 FILE	Define: [FILE | PKCS11] Default: FILE
defaultPkcs	Specify the default smart card reader name when certificate authentication dialog is displayed. This can be left empty or set to the name of the reader you want to make the default	Define: Name of reader Default: [None]
Pkcs11LibraryPath	Specify the full path name of the PKCS#11 library that you want to use with the type of smart card/token you have selected. This must be defined otherwise you will not be able to use the smart card. This property is only relevant if CertStore=PKS11. If the CertStore attribute is set to FILE, then the attribute can be left empty.	Define: Pathway Default: [None]
Pkcs11PromptText	Specify the prompt text displayed when the token (smart card) radio button is selected on the certificate authentication page. If this is left empty, the default text will be used. The default text is: “Enter your Token to reader and type password.”	Define: Text message Default: [See description]
disablePasswordField	Specify whether the password field is disabled or not. 0 = the password field is not disabled. 1 = the password field is disabled If the password field is disabled, the system forces the user to use some kind of third party authentication (such as user’s fingerprints) to authenticate to the smart card.	Define: [1 | 0] Default: 0

Keyname	Description	Values
Pkcs11TokenAbsenceBehavior	Specify the action that SSO Client will take when the user removes the smart card from the reader after they have authenticated with the certificate authentication method and a smart card. 0 = no action will be taken 1 = the workstation will be locked 2 = the user will be logged off	Define: [0|1|2] Default: 0
SCWaitingTime	Specify the time you want the SSO Client to wait before it becomes active and lets users perform Smartcard authentication. If the user has an SSO GINA and a slow computer they might get an error because the user can submit their smartcard authentication before the Smartcard service has started up in Windows. By setting the SCWaitingTime to delay the SSO Client for a few seconds this error can be avoided.	Define: Time in seconds Default: 0

comm

Keyname	Description	Values
TimeOutConnect	Specify the period in seconds before the connection timeout.	Define: Time in seconds Default: 120

SessionManagement

Keyname	Description	Values
ClientPortRange	Specify the range of ports that the SSO Client can use to receive messages from the Policy Server. **Note:** You can only use one range. If another format is used, such as a comma separated list, the default will be used.	Define: Port numbers Default: 20001-20201

StationLock

Keyname	Description	Values
Enabled	Do you want to enable the StationLock keynames in this section? If No, then the other tokens in this section are ignored.	Define: [yes|no] Default: Yes
LockScreenSaver	Do you want to protect the screen saver details from change while an SSO user is logged on? This only applies to Win 98SE/ME. For NT/2K/XP, the operating system setting in *Properties* determines this setting.	Define: [yes|no] Default: Yes
TimeOut	Specify the time in minutes that elapses before the station lock is activated. This only applies to Win 98SE/ME. For NT/2K/XP, the operating system setting in *Properties* determines this time out.	Define: Time in minutes Default: 5
MultiUser	Do you want multiple users to be able to unlock the computer? Yes = The user name field can be modified regardless of the SSO Client state. No = The user name field cannot be modified if the client state is logged on.	Define: [yes|no] Default: No
EnableOsUnlock	Do you want to allow users to be able to unlock their computer using the Windows Logon tab 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 machine using the Windows logon.	Define: [yes|no] Default: Yes
PermanentProtection	Do you want StationLock enabled when the SSO Client is not started? This only applies to Win 98SE/ME.	Define: [yes|no] Default: No
EnableSSOLogoff	Do you want a Logoff button to appear in the unlock station window? **Note:** If UnlockStationMode is set to "0" EnableSSOLogoff setting is ignored.	Define: [yes|no] Default: No

Keyname	Description	Values
GINAUnlockTimeout	Specify the time in seconds for the unlock operation to process a user's logoff script before the unlock function will be automatically terminated, and return to the desktop. This is to avoid a situation where the GINA hangs waiting for a script to terminate.	Define: Time in seconds Default: 0

TrayMenu

Keyname	Description	Values
ItemRefreshApplication List	Do you want to show the Refresh Application List item in the tray menu?	Define: [yes|no] Default: Yes
ItemOpenTools	Do you want Open SSO Tools in the tray menu?	Define: [yes|no] Default: Yes
ItemRemoveApplication List	Do you want Remove Application List in the tray menu?	Define: [yes|no] Default: Yes
ItemAboutClient	Do you want About SSO Client in the tray menu?	Define: [yes|no] Default: Yes
ItemExit	Do you want Exit in the tray menu?	Define: [yes|no] Default: Yes
ItemLockStation	Do you want the LockStation in the tray menu?	Define: [yes|no] Default: Yes
EnableDoubleClick	Do you want to enable double-clicking of the tray icon to open the SSO Tools menu? For this value to work, you must set ShowTrayIcon to yes.	Define: [yes|no] Default: Yes
ShowTrayIcon	Do you want to show the SSO Client tray menu icon in the system tray? If the installation is on a Citrix Metaframe server or on a shared workstation, we recommend that you set the value to no. In all other situations, keep the value as yes.	Define: [yes|no] Default: Yes

Tools

Keyname	Description	Values
EnableChangeUserPassword	Do you want to enable the password change for eTrust SSO?	Define: [yes|no] Default: Yes
EnableAdvanced	Do you want to enable the Advanced button in the "SSO Tools" window?	Define: [yes|no] Default: Yes
EnableRefresh	Do you want to enable the Refresh button in the "SSO Tools" window?	Define: [yes|no] Default: Yes

System Logon

Keyname	Description	Values
MustBeValidated	Do you want the user to authenticate to SSO during logon? This token is for Windows 98SE using the SSO Network Provider.	Define: [yes|no] Default: No
NetWareLogon	Do you want to use different Novell and Windows credentials? This token is for Windows NT GINA upgrade. The User can log on to Novell during logon to Windows NT with different credentials. If this token is Yes, the SSO Client finds the application <Novell server name> in the SSO database and the SSO Client logs the user into Novell with the user name and password from the <Novell server name> application.	Define: [yes|no] Default: No
NetWareServer	Specify the NetWare server to log on to if NetWareLogon is set to 'yes'. This value is ignored if the Novell Client contains its own Preferred Server value in the registry.	Define: Server name Default: [blank]

Keyname	Description	Values
UnlockStationMode	Specify the user lock option. 0 Single user lock option 1 Multiple SSO user lock option 2 Multiple Windows user lock option 3 No Windows re-authentication lock option (multiple SSO users, underlying Windows user remains untouched). This token manages Station Unlock in Windows NT/2000/XP. These values only apply when the SSO GINA is in use. For options 0, 1, and 2, the SSO Unlock screen displays information about the user. The top line displays the current SSO user, and the second line displays the current Windows user. For option 3, the SSO Unlock only displays the Windows user.	[0, 1, 2, 3] Default: 1

SSO Interpreter

Keyname	Description	Values
DisplayTask	Do you want the SSO Client to display the 'Tasks' dialog while connecting to Policy Server during script execution by the SSO Interpreter?	Define: [yes|no] Default: No
CloseAgentOnExit	Do you want the SSO Interpreter to start without the SSO Client? If the token is Yes, when the Interpreter finishes running it closes the SSO Client.	Define: [yes|no] Default: No
ClientWaitTime	Specify the timeout period in seconds for SSO Client to start when the SSO Interpreter is launched.	Define: Time in seconds Default: 60

Keyname	Description	Values
Plugins	Specify which additional Tcl extensions (DLLs) the SSO Interpreter should 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.	Define: DLL names Default: [None]
VarsOverwrite	Specify default values for some of the SSO Interpreter values. For example: VarsOverwrite=_TIMEOUT=10, _WINTITLE="Windowtitle" For more details about the variables, see the *eTrust SSO Scripting Reference Guide*.	Define: Variable name and value. Default: [None]

HLLAPI

Keyname	Description	Values
Hllapi	Specify the name of the HLLAPI DLL provided with the emulation software (excluding the path). For example: Whllapi.dll	Define: DLL name Default: [None]
HllapiFunc	Specify 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	Define: Function name Default: [None]
HllapiDllPath	Specify the location of the HLLAPIDLL folder, (do not include the file name). For example: "C:\Program Files\QWS370 PLUS"	Define: Pathway Default: [None]

Toolbar

Keyname	Description	Values
Enabled	Do you want the SSO Toolbar option enabled? If no, the other tokens in this section are ignored.	Define: [yes|no] Default: No
BuildToolBar	Do you want applications to appear in the tool bar? If no, the AppLineCount is ignored.	Define: [yes|no] Default: Yes
ApplLineCount	Specify the maximum number of applications that can appear in each toolbar line.	Define: Number of applications Default: 4
OffSetX	Specify the X coordinate of toolbar. Number of pixels from the top left of the window.	Define: number of pixels Default: 200
OffSetY	Specify the Y coordinate of toolbar. Number of pixels from the top left of the window.	Define: number of pixels Default: 200
Message	Specify the text message to appear in the toolbar logon dialog.	Define: Text message Default: "Welcome"
EditWidth	Specify the width of the message edit box in the Toolbar Logon dialog.	Define: Width Default: 10
EditHeight	Specify the height of the message edit box in the Toolbar Logon dialog.	Define: Height Default: 60
Background	Specify the background color of the message edit box in the Toolbar Logon dialog.	Define: Color (RGB) Default: 255.255.255 (255.255.255 = white)
Foreground	Specify the foreground color of message edit box in the Toolbar Logon dialog.	Define: Color (RGB) Default: 0.0.0 (0.0.0 = black)
TopMost	Specify whether the toolbar displays on top of the Toolbar Logon dialog.	Define: [yes|no] Default: No
AutoLogon	Do you want the toolbar to automatically log a user on if a ticket already exists in the cache for that user? **Note:** If the CleanTicketOnStart is set to "yes", the tickets are deleted.	Define: [yes|no] Default: No

Keyname	Description	Values
showMOTD	Do you want to display the Message of the Day when the user logs on? The Message of the Day is set on the Policy Server in the MOTD file.	Define: [yes|no] Default: No

MetaframeMigration

Keyname	Description	Values
SecondaryClient	Is this SSO Client installed on a Citrix MetaFrame server?	Define: [yes|no] Default: No

EventCommands

You can use the event commands to execute any program or script that can be run from a Windows command line. The events or actions that you can use to trigger a program or script are as follows:

- The user logs on to the SSO Client
- The user logs off from the SSO Client
- The SSO Client starts up
- The SSO Client shuts down

When you use event commands, you should:

- Avoid commands that require user interactions because situations can occur where the user may not be able to access the desktop.
- Be aware that certain events can trigger more than one event command. This means that the first command may still be running when the second command starts. The EventTimeout and GINAUnlockTimeout are designed to help avoid these situations. Examples of the command overlap problem include:
  - A user closes the SSO Client while they are still logged in. This triggers a logoff then a shutdown.
  - In a shared workstation environment using the SSO GINA, an SSO user is logged onto a computer that is in station lock, then another SSO user logs on that computer. This triggers a logoff then a logon.

- Enclose all event command strings in quotation marks if they contain any spaces. For example, you must use quotation marks if a path statement within the command includes any spaces.

Here is an example of how you might use the UserLogoffCmd. This functionality is particularly relevant in shared workstation environments. For example, you could run a Tcl script to close all applications when the user logs off, or is forced to log off:

```
ssointrp.exe -standalone -file script_name
```

`ssointrp.exe`
Invokes the SSO Interpreter.

`-standalone`
Specifies that the command does not have to access the SSO Client to run the SSO Interpreter. This part of the syntax is optional.

`-file`
Specifies that there is a script to run.

`script_name`
Specifies the script of your choosing, such as a Tcl logoff script located in the same directory as the SSO Interpreter. In this example, the program or script is located in the same directory as the SSO Interpreter.

Note: If you specify a path that includes spaces, you need to include quotation marks:

```
UserLogoffCmd=""C:\Program Files\CA\eTrust SSO\Client\ssointrp.exe"
–standalone –file "C:\Program Files\CA\eTrust
SSO\Client\sso_logoff.tcl""
```

Keyname	Description	Values
UserLogoffCmd	Specify the Windows command-line program or script to run when the user logs off, or if forced to log off, the SSO Client.	Define: Command Default: [None]
UserLogonCmd	Specify the Windows command-line program or script to run when a user logs on to the SSO Client.	Define: Command Default: [None]
ClientShutdownCmd	Specify the Windows command-line program or script to run when the SSO Client shuts down.	Define: Command Default: [None]
ClientStartupCmd	Specify the Windows command-line program or script to run when the SSO Client starts up.	Define: Command Default: [None]

Keyname	Description	Values
EventTimeout	Specify the time in seconds the SSO Client waits for an event command to finish before executing. If EventTimeout=0, the system waits indefinitely for the command to finish.	Define: Time in seconds Default: [0]

AppListRefresh

Keyname	Description	Values
EnableRefresh	Do 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.	Define: [yes|no] Default: No
TimePeriod	Specify the time between checks for an updated application list. If set, a refresh occurs every <n> minutes. **Note:** If left as 0h0, then a periodic refresh does not occur. If this value is set, then the specific-time refresh tokens are ignored.	Define: Time in [hours]h[minutes] Default: 0h0 (0 hours, 0 minutes)
StartTime	Specify the time each day that a refresh occurs every day (between the StartTime and EndTime). You might want to schedule this during low-network traffic periods. **Note:** If the values are left as 0h0 (default), the refresh does not occur. **Note:** If the StartTime and the EndTime have the same value, then a periodic refresh does not occur. If these values are set, they are only used if the 'TimePeriod' token is not set. The time is specified as a 24 hour clock: 21h31 indicates 9:31 pm.	Define: Time in [hours]h[minutes] Default: 0h0 (0 hours, 0 minutes = midnight)

Keyname	Description	Values
EndTime	See StartTime in this table. **Note:** If the values are left as 0h0 (default), the refresh does not occur. **Note:** If the StartTime and the EndTime have the same value, then a periodic refresh does not occur. If these values are set, they are only used if the 'TimePeriod' token is not set. The time is specified as a 24 hour clock: 21h31 indicates 9:31 pm.	Define: Time in [hours]h[minutes] Default: 0h0 (0 hours, 0 minutes = midnight)

Appendix

B

Configuring the Policy Server

This appendix describes the Policy Server settings that you can adjust. You can adjust most of the Policy Server settings using the Policy Server Settings configuration resource in the Policy Manager. Other settings are configured using ini files on UNIX or the system registry on Windows.

Policy Server Settings

The following table lists the Policy Server Settings subfolders that contain the properties you can configure using the Policy Manager:

Setting Display Name	Description
Artifact	Artifact generation settings
Cache	Client side authorization cache settings
Communication	Policy Server communication settings
General	General Policy Server settings
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
Remove Artifacts	Settings that control the background process that removes expired artifacts for the Token Directory
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

Artifact

Property	Description	Value
Expiration	Artifact lifetime. The minimum time is one minute (0h1).	Define: Elapsed time in hours and minutes, 0h0 format Default: 0h2 (2 minutes)

AD Authentication Provider

Property	Description	Value
UseLDAPChangePassword		Default: Yes (Use LDAP)

Cache

Property	Description	Value
ClientAZNLifeTime	Amount of time (in seconds) authorization cache entry is valid on client side	Define: Time in seconds

Communication

Property	Description	Value

Property	Description	Value
Auto_RollOut	When the value is set to 1 (yes), if next password exists, it automatically moves it to current password. If the 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 Policy Server computer is not the same as the SSO Authhost computer, then both computers must be configured as a server farm to enable password synchronization between the SSO password and the synchronized applications.	Define: [0 | 1] Default: 1 (yes)
ForkLimit	Maximum number of concurrent connections to handle.	Define: Number of sessions Default: 20
MaxAppQty	Maximum number of applications per user	Define: Number of applications Default: 200
PortNumber	TCP Port where Policy Server will listen	Define: Port number Default: 13980
PropDefaultMode	Use _default PWPOLICY as property default (enabled) or as pwpolicy default (disabled)	Define: [Enabled | Disabled] Default: Enabled
ReverseIpLookup	Resolves the client IP address in order to determine whether the user is allowed to log on from that terminal. When the value is set to disabled, no terminal check is done. This means that the user may log on even if they do not have terminal authorization. This might be the case if the terminal is not defined in the data store and the defaccess of terminal _default is set to NONE. If the token is set to enabled, the check is done and logon is permitted accordingly.	Define: [Enabled | Disabled] Default: Disabled

Property	Description	Value
SendBuffSize	The send buffer size for communication (in bytes)	Define: Size in bytes Default: 600000
SensitiveExpiration	Ticket lifetime for sensitive applications. For example, 0h5 is five minutes. The minimum time is one minute (0h1).	Define: Time in 0h0 format Default: 0h5 (5 minutes)
TicketExpiration	Ticket lifetime. For example, 0h5 is five minutes. The minimum time is one minute (0h1).	Define: Time in 0h0 format Default: 8h0 (8 hours, 0 minutes)
TimeOutConnect	Timeout period for connection	Define: Time in seconds Default: 120
TimeOutRecv	Timeout period for receiving	Define: Time in seconds Default: 60
TimeOutSend	Timeout period for sending	Define: Time in seconds Default: 60
UdpInUse	Enables or disables Policy Server's UDP listener.	Define: [Enabled | Disabled] Default: Enabled
UdpPortNumber	The UDP port number on which Policy Server will listen.	Define: Port number Default: 13990

General

Property	Description	Value
AllowDefaultEncKey	Allows Policy Server to accept SSO tickets generated with default or empty encryption keys.	Define: [Allowed | Not allowed] Default: Not allowed
ComListenQueueSize	Determines the maximum length of the queue of pending connections (backlog of incoming connections).	Define: queue length Default: 10

Property	Description	Value
DefaultAdminGroup	Defines the name of the group that its members are administrators. This is used only for a quicker browsing of the user names. No authorization decision is based upon this, users must still be assigned the admin role.	Define: Administrator group name Default: _ps-adms
DefaultContext	The default container dn used for sso authentication. The default empty value locates users in the base container.	Define: User container name Default: [None]
DefaultLocale	Specifies the locale of SSO clients. By default the Policy Server is set to recognize English language characters. If you want the Policy Server to recognize characters other than English then you must change this settings. The available locales are: ■ English - ENU ■ Japanese - JPN ■ German - DEU ■ French - FRA ■ Spanish - ESP ■ Italian - ITA ■ Brazilian-Portuguese - PTB ■ Simplified Chinese - CHS ■ Korean - KOR ■ Traditional Chinese – CHT For example, if you have users with French characters in their name, you must change this settings to FRA.	Default: ENU
DefaultMethod	The default authentication method that is used when the server gets an authentication request with no authentication method specified.	Default: SSO
DefaultTOKEN_DIR	The name of the Token Dir object in eTrust AC used to store the token dir connection and general information.	Default: PSTD

Property	Description	Value
DefaultUSER_DIR	The name of the User_Dir object in eTrust AC used to store the connection and general information of the default user data store, used when using SSO authentication.	Default: ps-ldap
EnablePerfMon	Enables or disables the Policy Server statistics and monitor collector that is used by a performance monitor (refer to Microsoft performance monitor for more info) and psperfmon utility	Define [yes|no] Default: Yes
EnforceStationId	When enabled, the Policy Server will restrict security tokens to the single station to which these tokens were issued to. In *Required* mode, the Policy Server will not support agents that do not support this feature	Default: Disabled Enabled Required
MaxConnections	The maximum number of connected users the Policy Server will store in its internal cache	Integer Default: 1000
RecieveQueueSize	Determines the maximum length of the queue of accepted connections waiting to be served The empty default means that the value is determined automatically by multiplying ForkLimit by 10.	Integer
SendBusyQueueSize	Determines the maximum length of the queue of "busy" requests send to the clients	Default: 20
ServerFarmSupport	Decreases the number of accesses to the Token Directory. When enabled, it increases performance in a single Policy Server.	Default: Enabled
ServerIP	Sets the IP address of the Policy Server that can be used to access the Policy Server directly in configurations where the Policy Server is behind a firewall, a smart router or, NAT.	Default: [None]
ServerVirtualIP	Sets IP address of the server that is used by clients that connect to the Policy Server through the Policy Server API, when the Policy Server is behind a firewall, a smart router or NAT. In most cases, this is the IP address of a router/load balancer.	Default: [None]
SSOAuthHostName	The name of the AUTHHOST object in the policy data store that is used by the Policy Server when generating tickets for SSO authentication	SSO_Authhost

Property	Description	Value
TrustedPolicyServers	List of Policy Server computers in the farm and their aliases	Default: [None]

Manage Idle Connections

Property	Description	Value
Enabled	Specifies whether the manage idle connection background process is enabled	Yes No – default
IdlePeriod	The time between two sequential operations within the process job	Time in seconds Default: 1
Interval	The time between background process jobs	Time in seconds Default: 36
StartTime	The start time of the background process job	Default: 120 This can be in either of two formats: ▪ The number of minutes after the Policy Server starts ▪ The time of day at which the process should run, in 0h0 format. For example, 22h0 is 10 p.m.

One Time Password

Property	Description	Value
GenLowWater	Generation number at which password is expired	Default: 4
MaxGenerations	Initial generation number for One Time Passwords	Default:1000
MaxSeedLength	Maximum seed length	Default:10
PortNumber	TCP port OTP agent will listen to	Default:13967

Remove Artifacts

Property	Description	Value
Enabled	Specifies whether the remove artifacts background process is enabled	Define: [yes|no] Default: Yes
IdlePeriod	The time between two sequential operations within the process job.	Time in seconds Default: 20
Interval	The time between background process jobs	Time in seconds Default: 300
StartTime	The start time of the background process job	Default: 180 This can be in either of two formats: ▪ The number of minutes after the Policy Server starts ▪ The time of day at which the process should run, in 0h0 format. For example, 22h0 is 10 p.m.

Remove Expired Tokens

Property	Description	Value
Enabled	Whether the background process that removes expired tokens from the Token Directory is enabled	Yes - default No
IdlePeriod	The time between each section of a background process job This idle period is to prevent the Policy Server and eTrust Directory from being flooded during a cycle.	Time in seconds Default: 5
Interval	The time between background process jobs	Time in seconds Default: 28800

Property	Description	Value
StartTime	The start time of the background process job	Default: 300 This can be in either of two formats: ▪ The number of minutes after the Policy Server starts ▪ The time of day at which the process should run, in 0h0 format. For example, 22h0 is 10 p.m.

Remove Heartbeat Failed Tokens

Property	Description	Value
Enabled	Whether the background process that removes expired heartbeat failed tokens from the Token Directory is enabled	Yes –default No
IdlePeriod	The time between each section of a background process job This idle period is to prevent the Policy Server and eTrust Directory from being flooded during a cycle.	Time in seconds Default: 5
Interval	The time between background process jobs	Time in seconds Default: 3600
StartTime	The start time of the background process job	This can be in either of two formats: ▪ The number of minutes after the Policy Server starts ▪ The time of day at which the process should run, in 0h0 format. For example, 22h0 is 10 p.m. Default: 60

Revoke

Property	Description	Value
def_disable_time	Time for which the user will be denied logon. The value 0 specifies that the user will not be automatically reinstated and that the intervention of an administrator will be required.	Define: Time in minutes Default: [none]
def_fail_count	Number of failed logons permitted before the end user's authorization to continue to try to log on is suspended.	Define: Number of logons Default: [none]

Session Management

Property	Description	Value
AllowTicketFromMultiple Stations	Lets two machines use the same SSO ticket. Required for Citrix MetaFrame environments.	Disabled (default) Enabled
HeartbeatFailAfter	Maximum number of heartbeats missed before the client session terminates. For example, if set to 3, the 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 before the session terminates (the default is 3)
HeartbeatInterval	Period between heartbeats	0—Disabled (no heartbeat is sent) Integer—Period in seconds between heartbeats (the default is 30 seconds)
HeartbeatProtocol	The protocol that sends the heartbeat. This determines whether a message is sent with the heartbeat.	Enabled – TCP. This protocol can include messages with heartbeat responses (default) Disabled - UDP. This protocol cannot include messages with heartbeat responses

Property	Description	Value
SendSessionTermination	Whether the Policy Server sends session messages directly to the SSO Client	Disabled—Disable the Direct Notification method Enabled—Enable the Direct Notification method (default)
SessionlessTerminals	List of terminals that are exempt from session management. Blank: All terminals are subject to session management (default) IP address: Terminal is exempt from session management (list is comma-delimited)	Define: Terminal(s) sessions that are exemp
SessionTerminationTimeout	How long the Policy 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.	Define: Time in seconds Default: 40
SessMgmtEnable	Whether session management is enabled. When set to **enabled**, SSO Clients from eTrust SSO 6.5 can work **without** Session Management, and SSO Clients from eTrust SSO 7.0 work **with** Session Management When set to **required**, if an SSO Client from eTrust SSO 6.5 or earlier starts, it attempts to connect to the Policy Server and then closes immediately.	Define: [Disabled | Enabled | Required] Default: Disabled

Registry and INI File Settings

In addition to the Policy Server settings you can configure through the Policy Manager, you need to configure some settings through the Windows registry or ini files.

If your Policy Server is running on a Windows machine, use the following Windows Registry key to configure those settings:

```
HKEY_LOCAL_MACHINE\Software\ComputerAssociates\eTrust\Shared\Policy
Server\8.0
```

If your Policy Server is running on a UNIX machine, use the policyserver.ini file to configure those settings. If you installed the Policy Server in the default location on UNIX, look for the INI file here:

```
/opt/CA/eTrustPolicyServer/policyserver.ini
```

The following table lists the sections in the INI file, and the keys in the Windows Registry:

Section or Key	Description
ssod	Defines Policy Server communication settings
exits	Defines password exit tokens
Main	Defines general Policy 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
bg.CIA	Defines information used by the Watchdog background task responsible for checking if the Policy Server is available.

ssod

Setting	Description	Parameters
MotdPath	Path where MOTDs (Message of the Day files) reside	Define: Pathway to folder Default: <ps_path>/motd
ScriptPath	Path where SSO-Tcl scripts reside	Define: Pathway to folder Default: <ps_path>/scripts
SsodKeyFile	Path where the ssod.key file resides.	Define: Pathway to a file Default: <ps_path>/polsrv.key
AppTktFile	Full path to the AppTicket key file (The default name of the file is appticket.key)	Define: Pathway Default: <ps_path>/appticket.key
GlobalPreCmdPath	Path to the Global PreCommand SSO-Tcl script (relative to ScriptPath)	Define: Pathway Default: [None]
WDCommListenPort	Specifies the watchdog port	Default: 13391
WDCommListenQueueSize		Define: Default:

Setting	Description	Parameters
WDCommRecvTimeout	Specifies the watchdog timeout for connections to it (via browser or telnet, or the smart router). **Note:** This is not the watchdog timeout for connections to the Policy Server.	Define: Time in seconds Default:
WDHTTPMode	Specifies whether the watchdog talk http. In text mode, the browser cannot connect to it and get an html response. If you use telnet, the outcome will be simple text.	1 - uses http (default) 0 – uses text mode
WDLoggerIniFile	Specifies the name of the watchdog log INI file.	
WDOnFailureString	Specifies the failure string.	Default: "SYSTEM TEST FAILURE"
WDOnSuccessString	Specifies the success string.	Default: "SYSTEM TEST SUCCESS"
WDTargetPolicyServer	Specifies the Policy Server that the watchdog watches. The default, no value, is to watch the same Policy Server that the watchdog is installed with (localhost). **Note:** You should test a remote Policy Server to see that the reaction time is fast enough to be considered as "up". The above timeouts can be configured to specify that a Policy Server is down, even if it is up, but the communication to it is slower than needed.	Define: Default:
BackCalcPath	Path where BackCalc (Server cache) files reside	Define: Pathway Default (UNIX): /usr/sso/ssobgc <server_Home>\psbgc Default (Windows): C:\Program Files\CA\eTrust Policy Server\PsBgc

exits

Setting	Description	Parameters
ExitsPath	The directory path where the exit.dlls are located.	Define: Pathway

Setting	Description	Parameters
		Default: [None]
PasswordExits	List of password exit names. Located under ExitsPath, it contains the list of DLLs.	Define: Default: [None]
AutogenExit	List of auto generation exit names. It is located under ExitsPath.	Define: Default: [None]

Main

Setting	Description	Parameters
SSOMode	Enables Policy Server's SSO features. 1—SSO mode 0—not SSO mode	Define: [0|1] Default: 1
WACMode	Enables Policy Server's WAC features. 1—SSO mode 0—not SSO mode	Define: [0|1] Default: 1
MaxConnections	The maximum numbers of users allowed to be logged in simultaneously.	Define: Number of connections Default: 1000
PolicyDBProviderDll	The Policy DB provider plug-in This is a DLL on Windows and a shared object on UNIX.	Define: Pathway Default:
MsgFile	The name of the Policy Server message file	Define: File name Default:
MsgPath	The path for the Policy Server message file	Define: Pathway Default:
LoggerIniFile	Points to the name of the Policy Server's log INI file.	Define: Pathway Default:
UserDBProviders	A list of user data store provider's plug-ins to be loaded by Policy Server.	Define: Default:

auth.<method_name>

Setting	Description	Parameters
DLL	Full path to Authentication provider library	Define: Default:

AuthMap

Property	Description	Parameters
Methodx	A list of authentication names and descriptions	Define: Comma-separated list Default:

UserDBProvider.<provider_name>

Setting	Description	Parameters
ID	The internal ID of this provider in the eTrust Access Control. LDAP = 1 AD = 4 Eac = 8 ACF2 = 32 TSS = 64 RACF = 256	Define: Numeral representing provider Default: [Differs according to provider]
DLL	The plug-in library This is a DLL on Windows and a shared object on UNIX.	Define: Pathway to file extension .so Default:

bg.CIA

Setting	Description	Parameters
Enabled	Specifies whether service is enabled. Setting it to 0 disables its action, which means that even if the service is up, it will not do anything.	Define: [0|1] Default: 1

Setting	Description	Parameters
IdlePeriod	Specifies the interval, in seconds, between queries in failure mode (when the watchdog determines that the Policy Server is down).	Define: Elapsed time in seconds Default: 20
Interval	Specifies the interval, in seconds, that the watchdog uses to query the Policy Server.	Define: Elapsed time in seconds Default: 5
MsgFile	The name of the Policy Server WatchDog message file	Define: File name ending in .msg Default:
MsgPath	The path to the Policy Server WatchDog message file	Define: pathways to MsgFile Default:
StartTime	Specifies the number of seconds after the watchdog is up and before it starts to query the Policy Server.	Define: Time in seconds Default: 1
WDConnTimeout	Specifies the connection timeout when trying to connect to the Policy Server. If there is still no connection to Policy Server, it is considered as a failure to connect.	Define: Time in seconds Default: 2
WDMaximumFailed	Specifies how many connection failures are required for the watchdog to decide that the Policy Server is down.	Define: Number of connections Default: 3
WDTargetPolicyServer	Specifies which Policy Server the watchdog watches. An empty value means that the watchdog watches the Policy Server that the watchdog was installed with (localhost).	Define: Computer name or IP address Default: [None]

Appendix C

Configuring the One-Time Password Agent: seotp.ini

The seotp.ini file is installed when one-time password (OTP) authentication is used. It provides initialization parameters for the OTP agent.

For further information about the OTP agent, see the 'Authenticating Users to Applications chapter.'

Sections of the seotp.ini File

The seotp.ini file has one untitled section.

Keyname	Description	Values
OtpDatabase	The path to the OTP database	Default value: /usr/seotp/seopt.db
PasswdFile	The path to the password file	Default value: /etc/passwd
ShadowFile	The path to the shadow file, if it is used	**Blank** (Default) No shadow file is used **AIX** /etc/security/passwd **Solaris** /etc/shadow **SolarisX86** /etc/shadow **HP_UX 10** /etctcb/files/auth

Keyname	Description	Values
ShadowChar	If there is a shadow file, the ShadowChar token specifies the character that appears after every user name in every entry in the password file.	If a user entry is Sharon:x:253:1::/home/Sharon:/bin/sh, then the ShadowChar value should be x.
NisMakeDir	If this is an NIS Server installation, this token specifies the path to the NIS make directory.	Leave blank if no NIS server is used
NisMakeCmd	If this is an NIS Server installation, this token specifies the make command.	Leave blank if no NIS server is used
RouteTo	If this is an NIS Client installation, this token specifies the name of the NIS server.	Leave blank if no NIS server is used
UseBlockingSockets	Whether to use blocking sockets	Yes No (default)
TimeOutConnect	Timeout period for connecting	Time in seconds (default is 60 seconds)
TimeOutRecv	Timeout period for receiving	Time in seconds (default is 60 seconds)
TimeOutSent	Timeout period for sending	Time in seconds (default is 60 seconds)
IdleFreq	Frequency of the idle function	Time per second (default is 20 times per second)

Appendix

D

Using Selang

eTrust Single Sign On provides selang, a command language for entering and updating definitions in the data store. Generally, you use the Policy Manager to update the data store. However, you can also use Selang commands. The Selang commands are particularly useful in batch operations.

This appendix explains the syntax of those commands specific to eTrust SSO, as well as other commands that include parameters specific to eTrust SSO.

In addition, all other commands that appear in the *Command Reference Guide* are applicable and may be used in eTrust SSO. You should familiarize yourself with eTrust Access Control by reading the *eTrust Access Control* documentation.

Working with selang Commands

There are two ways to use selang:

- At the command prompt
- Within the program's console

Using selang Commands at a Command Prompt:

To work with selang commands in a command window;

1. Open a command window.
2. If selang is included in the path (which it should be if you've installed eTrust SSO), type **selang -h** at the dos prompt.
3. Else, go to the eTrust Access Control data store bin directory. Type "dir selang.exe" to confirm that the executable exists there; then type **selang -h** and you should see a list of selang commands.

For example, to run a file containing a selang script, type the following command:

```
selang -v -s -f <filename> -o <outputfilename>
```

This runs the script file 'filename' in verbose mode, with output piped to 'outputfilename'. You can include the path to the file, but make sure to surround it with double quotes (" ") to stop it failing at any spaces in the path.

Using selang Commands in the Selang console

To work with selang commands in a command window;

1. Open a command window.
2. Type **selang** at the dos prompt to enter the console. The command prompt will change to: **eTrust>**

Within the console, you can type 'help' to view the commands available. You can also type 'help' followed by the command name for more information about the syntax of a command.

Tips and Tricks for Using selang Commands

Whatever runs in the selang console, will also run in a script

This means that you can check a line that's giving you a problem by running it within the console first, rather than trying to debug an entire script

Selang is Case-Sensitive and Space-Sensitive

For example, the way to continue a line onto the next line is to use a '\' at the end of the line. However, if there is any space on the line after the '\', the entire script will fail .

If you have a couple of lines, continued on with '\', this literally joins the end of one line onto the start of another.

For example, the following script named Script1 will not work, because of the lack of space before the '\' on line 1, or at the start of the next line

Script 1

```
#define six new applications :
newapp ( App1 App2 App3\
App4 App5 App6)
```

Either of the scripts below will work: add a space before the start of the following line (as in Script2), or a space before the '\' at the end of a line (Script3):

Script 2

```
#define six new applications :
newapp ( App1 App2 App3\
  App4 App5 App6)
```

Script 3

```
#define six new applications :
newapp ( App1 App2 App3 \
App4 App5 App6)
```

Include Lists in Brackets ( )

When you have list of items to create, edit, or update, list them in brackets: ( ), with a space between each item.

You must have a space at the start of the list, that is, after the first bracket.

Sample Commands

This section provides you with examples of some of the commands related to data store administration.

- To define a new SSO user called janjones enter:

```
newusr janjones name (‘Jan Jones’) \
admin            \
auth_type (METHOD7)     \
pwd_autogen
```

  This newusr command defines a user ID of janjones for the user Jan Jones. Jan will have the admin authorization attribute, will be required to authenticate with Method 7 (Windows NT primary authentication), and will use passwords automatically generated by eTrust SSO (pwd_autogen).

- To define a user group called acctgp enter:

```
newgrp acctgp           \
comment (‘the payroll group in Accounting’)
```

  The comment notes that this group represents the payroll group in the Accounting department.

- To join janjones to the acctgp group enter:

```
join janjones group(acctgp)
```

- To define an authorization host (authhost) named athena enter:

```
newres AUTHHOST athena \
defaccess(read) \
audit(failure) \
key(inthebeginning1066)
```

  This newres command defines a new resource—an authorization host for primary authentication—called athena in the AUTHHOST class. Since its default access is defined as read, every user can authenticate with athena. An audit record will be written in the SSO audit file every time an attempt to authenticate with this host fails. The encryption key for the host is inthebeginning1066.

- To authorize members of acctgp to use the host athena for primary authentication enter:

```
authorize AUTHHOST athena \
access(all)   \
gid(acctgp)
```

- To define a new mainframe application called TSO_TEST enter:

```
newappl TSO_TEST host(MVSTEST) \
iconfile(emu.exe)   \
caption(‘TSO Test’)  \
login_type(ticket)
```

This newappl command defines a new application called TSO_TEST. The icon representing the application can be found in the emu.exe file and the caption that will appear underneath the icon will be TSO Test. The application resides in a host called MVSTEST and users must use tickets generated by the Policy Server to log on to the application.

- To allow acctgp to use the TSO_TEST application enter:

```
allow APPL TSO_TEST \
gid(acctgp)
```

  This allow command enables users who are members of acctgp to invoke the TSO_TEST application. Since no access is specified, the default access execute applies.

- Commands that add users and groups to the data store include **newusr** and **editusr** commands, which add a user to the data store, and **newgrp** and **editgrp** commands, which add a user group.

Scripts

A script contains a line for each command that is to be run in a batch job. Each command performs an operation on an object (record) in the data store. The operation can be creating, updating, or deleting the record. The syntax of a line in a script is:

```
command(objectName) parameter1Name(Value) \
parameter2Name(Value) ... \
parameternName(Value)
```

Generally, scripts are used in setting up a new data store and for carrying out an identical change in a large number of data store records. This is an example of a script that creates user records:

```
editusr (“JSmith”) fullname(“Jason Smith”) \
phone(“736-519-2526”) location(“Acme”) \
org_unit(“Loans”) auth_type(Method5) \
when(days(mon, tue, wed, thu, fri, sat, ) \
time (AnyTime))
editusr (“BBrown”) fullname(“Betty Brown”) \
phone(“736-519-2519”) location(“Acme”) \
org_unit(“MIS”) auth_type(Method5)
when(days(mon, tue, wed, thu, fri, sat, ) \
time (AnyTime))
editusr . . .
```

The simplest way to begin building a script is to use the Policy Manager to set up one data store record with the parameters you plan to load. For instance:

1. Run the Policy Manager, enable the Command Log feature.
2. Set all the record values you need in the relevant Policy Manager windows and click Ok or Apply.
3. The equivalent command will appear in the Command Log window.
4. To copy the command in the Command Log window, highlight the command with the mouse, right-click on the Command Log window to get a pop-up menu, and select Copy.

Note that this will not work for changes to logon information (editlogin commands).

To run a script, use the command:

```
-r filename
```

Conventions

This appendix uses several conventions to make locating and identifying information easier. When representing syntax and user input, the following conventions are used:

Convention	Usage
Italic type	Indicates a variable name or placeholder for which you must supply an actual value.
Case Sensitivity	System command and environment variable names may be case-sensitive, depending on the requirements of your operating system.
No braces nor brackets	Used with one mandatory item.
{ } curly braces	Used to enclose mandatory items that are used with OR (the vertical bar)
[ ] square brackets	Used to enclose an optional item.
| vertical bar (OR)	Used between items in a list to indicate that you must choose only one of the items.
\ backslash	Sometimes a command does not fit on a single line in the book. A backslash at the end of a line indicates that the command continues on the following line. The use of backslashes in the book does not necessarily indicate that you need backslashes in the same places.

Command Types

There are two different types of commands addressed in this section. The first, Selang commands, are specific to eTrust SSO. The other commands may be used to define other functions, but include parameters that are specific to eTrust SSO.

Selang Commands

This section identifies selang commands specific to eTrust SSO. These commands will allow you to:

- Add or remove accessors to a resource access control list (ACL).
- Change or add a new application setting to the data store.
- Create or change accessor logon information.

The Selang commands specific to eTrust SSO are:

Command	Description
allow	Add accessors to a resource ACL
allow-	Remove accessors from a resource ACL
chappl	Change an eTrust SSO application record
chlogin	Change logon information for a specific user/application combination
editappl	Create or change application settings in the data store (to create or change application groups, use editres)
editlogin	Create or change logon information for a specific user/application combination
newappl	Add one or more new applications to the data store (to add application groups, use newres)
newlogin	Add logon information for a specific user/application combination
rmappl	Remove application records from the data store (to remove application groups, use rmres)
rmlogin	Remove logon information
showappl	List the properties of application records in the data store

Other Commands

This section identifies other commands that include parameters specific to eTrust SSO that will allow you to:

- Manage the properties of a user record.
- Change or add a new group setting to the data store.
- Manage the properties of a resource record.

The commands with parameters specific to eTrust SSO are as follows:

Command	Description
authorize	Maintain the lists of users and groups authorized to access a particular resource.
authorize-	Deny access to a particular resource
chgrp	Change existing group settings in the eTrust Access Control data store
chres	Change the properties of an authentication host, group of authentication hosts, password policy, or other resource.
chusr	Change the properties of a user
editgrp	Add a new group to or change an existing group in the eTrust data store.
editres	Modify an existing resource record or define a new one
editusr	Modify an existing user record or define a new one
join	Join a user to a group
join-	Remove a user from a group
newgrp	Add new groups to the eTrust Access Control data store.
newres	Define a new resource record
newusr	Define a new user record

Further Information

The *Selang Scripting Guide* describes how to write and maintain scripts for eTrust SSO.

The *Getting Started* guide explains how to install eTrust SSO, and provides a quick tour of the implementation process.

The *eTrust Access Control* documentation provides a detailed reference to the Access Control commands that are not specific to eTrust SSO. It also discusses in detail the concepts used by eTrust Access Control, including the data store in particular. An implementation guide is included.

Appendix

E

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.

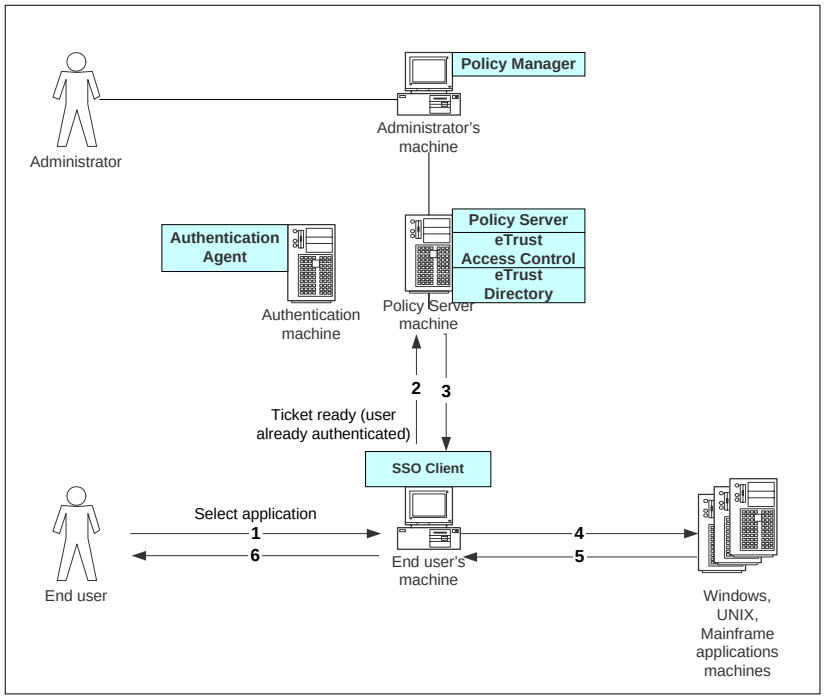
Error Message Flow Diagrams

This section lists the error messages you may receive, including a description of the error message and any corrective actions you can take.

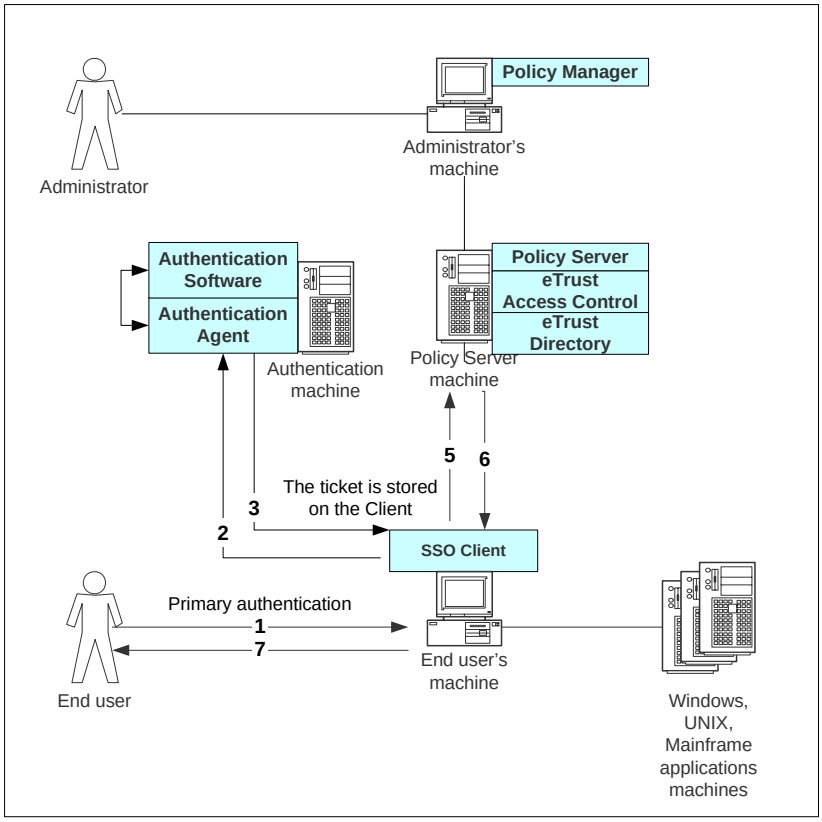
Error Message Flow Diagrams

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.

Flow Diagram 1



Flow Diagram 2



Error Messages

Error Message: "Can't connect to host <host name>"

Number: 0x010000

Component: SSO Client

Communication Flow diagram Reference: 2.2, 2.4

Cause: Policy Server is down whilst either trying to logon or access something.

Resolution: Restart the Policy Server service on the Server machine

Cause: A valid Authentication host name is not entered into the SsoClnt.ini file

Resolution: Enter the name of the server machine in the appropriate 'authhost' field

Cause: The authentication host machine is not running

Resolution: Start the authentication service on the authentication host

Error Message: "Host <hostname> is unknown"

Number: 0x01

Component: SSO Client

Communication Flow diagram reference: 2.2

Cause: The hostname specified in the SsoClnt.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: SSO Client

Communication Flow diagram reference: 2.2

Cause: The hostname specified in the SsoClnt.ini file cannot be reached

Resolution: Verify that the host machine is spelt correctly and that the machine is booted and can be contacted via TCP/IP

Error Message: "Unable to connect to Policy Server Token Directory"

Number: Number not known

Component: Policy Manager

Cause: Unknown

Resolution: Restart the DSAs on the Policy Server machine

Error Message: “<”Path name”> is not recognized as an internal or external command"

Component: SsoClnt.ini

Cause: A space appears in either the event command string or in the path statement of an Event Command in the SsoClnt.ini file.

Resolution: Enclose in quotation marks. See the following examples.

Example of correct syntax:

```
UserLogoffCmd=””C:\Program Files\CA\eTrust
SSO\Client\ssointrp.exe” –standalone –file “C:\Program
Files\CA\eTrust SSO\Client\sso_logoff.tcl””
```

Example of incorrect syntax:

```
UserLogoffCmd=”C:\Program Files\CA\eTrust
SSO\Client\ssointrp.exe –standalone –file C:\Program
Files\CA\eTrust SSO\Client\sso_logoff.tcl”
```

Error Message: "Low-Level communication error <additional information>"

Number: 0x0200

Component: SSO Client

Cause: Ingres and eTrust Directory not started on the Policy Server machine

Resolution: At the command line run 'ingstart' from the Ingres\bin directory followed by 'dxserver start all'

Number: 0x0300

Error Message: "Communication failure with host <server name> <extended information>"

Component: SSO Client

Cause: Machine does not have TCP/IP installed

Resolution: Install TCP/IP networking on the Server machine

Cause: Port number is already in use

Resolution: Change the port number range being used by the SSO Client

Error Message: "Communication aborted with host <host name>"

Number: 0x01010402

Component: SSO Client

Communication Flow diagram Reference: 2

Cause: Communication timeout with Policy Server

Resolution: ?

Error Message: "Application <application name> not found <server name>"

Number: 0x01020301

Component: SSO Client

Communication Flow diagram Reference: 1.2

Cause: Application list is outdated, client is trying to access an application that no longer exists in the database

Resolution: Refresh application list

Error Message: "User <user name> not found <server name>"

Number: 0x01020401

Component: SSO Client

Communication Flow diagram Reference: 2.2

Cause: Access Control not running whilst trying to logon

Resolution: Start the Access Control daemons

Cause: User does not exist

Resolution: Add the user to the database

Error Message: "Password has expired"

Number: 0x0600

Component: 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: Change the settings associated with that user on the Policy Server to allow them to change their password

Error Message: "Memory allocation failure in ssoapi <additional information>"

Number: 0x0101

Component:

Communication Flow diagram Reference:

Cause: Parameters supplied to the API are not the correct format

Resolution: Check the documentation and reformat the command using the SSOAPI accordingly

Error Message: "Password Quality Check failed <extended information>"

Number: 0x040000 (0x0101, 0x0201, 0x0301, 0x0401, 0x0501, 0x0601, 0x0701, 0x0801, 0x0901)

Component: SSO Client

Communication Flow diagram Reference: 1.3

Cause: Password does not meet the password policy minimum requirements for the number of of a particular type of character (numerical, aplha, special character, uppercase and lowercase)

Resolution: Check the minimum standards as specified in the password policy and re-enter a password

Cause: Too many repetative 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: **"Cannot set an empty new password"**

Component: Policy 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 store in the database if Access Control has shutdown whilst user logged in

Resolution: Restart Access Control

Error Message: "Database Engine is not operational"

Number: 0x0101

Cause: Trying to access information store in the database if Access Control has shutdown whilst user logged in

Resolution: Restart 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 policy server does not match that of the authentication host

Resolution: Change either the encryption key in the registry on the authentication agent machine or the key for the authentication host Policy Server.

Error Message: "User has no password yet"

Number: 0x0300

Component: Policy Server

Communication Flow diagram Reference: 2.4, 2.5

Cause: User password has not been set for the authentication method used

Resolution: Created a user password via the Policy Manager or through the selang command line interface,

Error Message: "Password mismatch for user <user name>"

Number: 0x01060400

Component: SSO Client

Communication Flow diagram Reference: 2.2

Cause: User has supplied an incorrect password

Resolution: Supply correct password or change password via 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 Policy Server, uncheck the box that disables the user's account.

Cause: User does not exist when authenticating to a Policy Server

Resolution: Create the user on the Policy Server

Error Message: "Can't access application"

Number: 0x01070401

Component: SSO Client (GINA)

Communication Flow diagram Reference: 1.2, 2.4

Cause: Application list is outdated, 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 doesn't have rights to change the application password

Resolution: Change the user password policy assigned to this application on the Policy Server to allow the user to change the password.

Error Message: "Access Denied"

Number: 0x070000

Component: Authentication Agent and it's 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 it's 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 it's host server are functioning as expected.

Error Message: "User <user name> is disabled"

Number: 0x01

Component: Policy Server

Communication Flow diagram Reference: 2.4, 2.5

Cause: The user's account has been disable.

Resolution: If this was done in error, use the Policy Manager or the selang command line interface to enable the user account.

Error Message: "Application <application name> is disabled"

Number: 0x02

Component: Policy 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 or the selang command line interface to enable the application.

Error Message: "User not allowed to use APPL <application name>"

Number: 0x01

Component: Policy Server

Communication Flow diagram Reference: 2.5

Cause: The current user does not have rights to launch the application is question.

Resolution: Using the Policy Manager or the selang command line interface, grant the user permissions to use this application by adding it to their list of authorised applications.

Error Message: "User <user name> not allowed to login now"

Number: 0x02

Component: Policy 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 login from terminal <terminal name>"

Number: Number not known

Component: Policy Server, Policy Manager

Communication Flow diagram Reference: N/A

Cause: The Policy Manager is attempting to logon from a terminal that has not been defined as an allowed terminal in the Policy Server's database.

Resolution: From an allowed terminal, using the Policy Manager or the selang command line interface, create a new terminal named after the machine that originally attempted to connect to the Policy Server.

Error Message: "User not allowed to use <authentication host name?"

Number: 0x0201

Component: SSO Client

Communication Flow diagram Reference: 2.4

Cause: When connecting to an authentication host that is not defined on the policy server listed in the client configuration file.

Resolution: Add the authhost to the policy server or change the server name in the client configuration file.

Error Message: "A communication failure occurred" "Details:" "Reported by Policy 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 "SSO" program bar (only Access Control, Windows NT and Tools).

Cause: Unknown

Resolution: Unknown

Error Message:"User <user name> has no password yet"

Number: 0x01060300

Component: 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 Policy Manager.

Error Message:Communication error with host

Number: 12296

Component: SSO Client

Communication Flow diagram reference:

Cause: SSO Client is unable to communicate with the Authentication Agent.

Resolution: Ensure network to auth agent is up and authentication agent is running.

0x0 ETWAC_API_OK

Reason:

Indicates success.

Action:

No action necessary.

0x100 ETWAC_API_FAIL

Reason:

Operation failed.

Action:

Enable Web Agent logging to obtain more detailed information about the failure.

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.

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.

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 will get 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 should have this permission and grant the permission to the user if necessary..

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

The user is not allowed to access the Policy Server machine (not permitted access to that terminal; eTrust Access Control authentication only)

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

Grant the user access to the Policy Server machine

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 server has been restarted and all tokens that exist in the client are not valid

The server cannot recognize the token, or someone has tampered with the token

The server crashes, the user cache is erased, and the Policy Server cannot tell what the old token was

The token used has expired

Action:

The Policy Server holds a table that contains all of the users who are logged in. When you restart the Policy Server, this table is cleared and all users that ere previously connected to the Policy Server will need to reconnect.

0x700 ETWAC_API_ACCESSOR_TOKEN_IS_NOT_VALID

Reason:

The token of the user that the operation will be 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 will be performed on) is invalid.

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.

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.

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 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 Policy Server is down, so verify that the Policy 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.

0x00000004 CC_USER_DB_EAC

Component:

eTrust Web AC user data store component.

0x00000005 CC_USER_DB

Component:

User data store proxy component.

0x00000006 CC_POLICY_DB_EAC

Component:

eTrust Web AC policy data store provider component.

0x00000007 CC_POLICY_DB

Component:

Policy data store proxy component.

0x00000008 CC_POLICY_SERVER

Component:

Policy 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.

0x00000101 ETWAC_UNKWOWN_USER_DB

Reason:

Unknown user data store

Action:

Check whether this type of user data store is supported by eTrust Web AC. If supported, define it to the Policy Server.

0x00000102 ETWAC_UNKNOWN_POLICY_DB

Reason:

Unknown policy data store

Action:

Check whether this type of policy data store is supported by eTrust Web AC. If supported, define it to the Policy Server.

0x00000103 ETWAC_AUTH_HOST_NOT_FOUND

Reason:

Authentication host was not found.

Action:

Verify that this authentication host is defined to the Policy 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 administrator to enable the resource.

0x0000010A ETWAC_CANT_CHANGE_LOGINID

Reason:

Unable to update logon name.

Action:

Contact the 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 administrator to change the application 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 Policy Server logging to obtain more detailed information about this failure.

0x00000110 ETWAC_POLICY_DATASTORE_INIT_FAILED

Reason:

Policy data store initialization failed.

Action:

Enable Policy 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 Policy 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 should 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 should have this permission, and then grant the 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 administrator.

0x00000502 ETWAC_AUTHHOST_NOT_ALLOWED

Reason:

The user was not allowed to use the indicated authentication host.

Action:

Contact the administrator.

0x00000503 ETWAC_TIMEDAY_RESTRICTION

Reason:

The specified user is not allowed to log on now.

Action:

Log on during the time on the day that access is allowed.

0x00000504 ETWAC_TERMINAL_RESTRICTION

Reason:

The user is not allowed to log on from the specified terminal.

Action:

Log on from an accepted terminal.

0x00000505 ETWAC_USER_DISABLED

Reason:

The specified user is disabled.

Action:

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 administrator to authorize the issuing Policy 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 Policy 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 Policy Server logging to obtain more information about this failure.

0x00000A02 ETWAC_UNPACK_FAILED

Reason:

Failed to unpack data.

Action:

Enable Policy 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 server is down.

Action:

Contact the administrator to determine the status of the server.

0x00000A07 ETWAC_SERVER_BUSY

Reason:

The server is busy.

Action:

Try again later.

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 client or the server.

Action:

Enable Policy Server logging to obtain more information about this situation.

Appendix

F

Password Exits

This chapter describes how to use password exits in eTrust SSO.

Password Exits

eTrust Single Sign-On 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 passwords rules that go beyond the rules that have been created in eTrust Single Sign-On. 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 exists, you must write a DLL (dynamic-link library).

Password Change Exits

There are two kinds of password change exits: *pre-exit* and *post-exit*. Pre-exit is essentially an exit to validate passwords. Post-exit is 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, sso db update, 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 eTrust Single Sign-On'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 have 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* found in the Policy Server install path). PasswordExits contains the list of DLLs. Note that 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, ssodExits.h) are located in ssodDir/include.

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_Init	Initializes the password exit DLL.	Password exit
ssod_Exit_Pwd_Term	Performs termination and cleanup of the password exit DLL.	Password exit
ssod_Exit_PreSetPwd	Is called after SSO checks the quality of the new password using the password policies of SSO.	Password exit
ssod_Exit_PostSetPwd	Is called after the SSO server successfully updates the SSO database with the new password.	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 default password auto-gen mechanism.	Auto-gen exit

These functions are described in detail in the following pages.

ssod_Exit_Pwd_Init

The ssod_Exit_Pwd_Init function initializes the password exit DLL.

Syntax

```
unsigned long ssod_Exit_Pwd_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.	Input/Output
pulExitVersion	Indicates the exit version.	Output

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	Indicates the pointer to exit-specific context. This is a placeholder for any static data.	Input/Output

Return values

A return value of 0 indicates success.

A non-zero return value indicates an error.

ssod_Exit_PreSetPwd

The ssod_Exit_PreSetPwd function is called after SSO checks the quality of the new password using the password policies of SSO.

Syntax

```
unsigned long  ssod_Exit_PreSetPwd(
   void        *pCtxExit,
   const char      *szSSOUserName,
   const char      *szApplName,
   const char      *szLoginName,
   const char      *szPassword,
   char       **pszExtra
  );
```

Parameter	Description	Type
pCtxExit	Exit specific context.	Input
szSSOUserName	The user name as it appears in the SSO database.	Input
szApplName	The application name.	Input
szLoginName	The application's login ID.	Input
szPassword	The password that is to be set.	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.

ssod_Exit_PostSetPwd

The ssod_Exit_PostSetPwd function is called after the SSO server successfully updated the SSO database with the new password.

Syntax

```
unsigned long  ssod_Exit_PostSetPwd(
   void      *pCtxExit,
   const char     *szSSOUserName,
   const char     *szApplName,
   const char     *szLoginName,
   const char     *szPassword,
   char      **pszExtra
);
```

Parameter	Description	Type
pCtxExit	Exit specific context.	Input
szSSOUserName	The user name as it appears in the SSO database.	Input
szApplName	The application name.	Input
szLoginName	The application's login ID.	Input
szPassword	The password that is to be set.	Input
pszExtra	A pointer to a string that contains an error message when this password is rejected.	Output

Return Values

A return value of 0 indicates success.

A non-zero return value indicates an error.

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.