PRISMTECH

Security Interceptors & Console (SI&C)

V 1.5

Administrator's Guide

USA Corporate Headquarters	European Headquarters	Central Europe
PrismTech Corporation	**PrismTech Limited**	**PrismTech (Germany) GmbH**
6 Lincoln Knoll Lane, Suite 100	5th Avenue Business Park, Team Valley	Schoenhauser Allee 6-7
Burlington, MA, 01803	Gateshead, Tyne & Wear, NE11 0NG	D-10119 Berlin
USA	United Kingdom	Germany
Tel: (1) 781-270-1177	Tel: +44 (0) 191-4979900	Tel: +49 (0) 30-44030660
Fax: (1) 781-238-1700	FAX: +44 (0) 191-4979901	Fax: +49 (0) 30-44030678

1st edition
Issue Date: 21 September 2006

Contents

CHAPTER 1 Introduction

This chapter provides a general overview of the functionality and architecture of the Security Interceptors and Console (SI&C).

1.1 Overview

The Security Interceptors and Console (SI&C) is a server-side CORBA AAA security mechanism that:

- authenticates clients,
- authorizes incoming client requests, and
- audits security-relevant system events.

The Security Interceptors are implemented in a Java run-time library that is linked to the server-side ORB implementations and enforces a security policy, which is specified in a security policy file. No application changes are required to use this mechanism, CORBA resources are protected by simply configuring the ORB to use the library. No on-line connections to a management console or policy server are necessary, so a process protected with SI&C is essentially self-contained.

Security policies are specified using a graphical console, the Policy Editor. Policies define the security requirements for CORBA objects, e.g., which authentication mechanism must be used, and which users may invoke a given operation. As just explained, the SI&C works in an off-line mode, i.e., the Policy Editor does not connect to the SI&C interceptors but rather writes policy files to storage. The SI&C runtime can be configured to check for updates of the policy file to dynamically update with the new policy data.

As an example, consider a CORBA Naming Service that you want to protect from malicious modifications. An example of such a modification would be an attacker removing name bindings, so that client applications cannot find required resources using names.

Even worse, an attacker could modify existing bindings so that clients will unknowingly look up mock objects, which may relay or even modify application traffic. The authentication and authorization mechanisms provided by SI&C can prevent security breaches like these granting access to operations like `bind()`, `unbind()` to trusted clients only.

1.2 Security Functions

The remainder of this chapter introduces the three A's in AAA security: authentication, authorization, and auditing, as provided by the SI&C.

1.2.1 Authentication

Authentication is the process of establishing an identity for the communication peer. Authentication is performed by verifying the authenticity of client credentials, e.g., a password, against data in a user database. Authenticated identities are the basis for determining privileges such as role memberships, and hence for making access decisions, as described in the next section. Authenticated identities are also useful for correlating audit events with clients.

The SI&C supports the following mechanisms for authenticating CORBA clients (listed from the weakest to the strongest):

- anonymous, i.e. unauthenticated clients, without credentials,
- IP-based, i.e. clients that only present source IP addresses,
- Password-based, i.e. clients that present user IDs and passwords. This is also referred to as Basic Authentication.
- X.509-based, i.e. clients engaged in an SSL handshake present public key certificates.

The SI&C lets you define a minimum authentication level for a CORBA object, e.g., password-based. With such a policy, the SI&C would not accept clients that authenticate using only source IP addresses, but both password-based and X.509-based authentication processes would be accepted.

It also possible to completely block authentication for a given resource by defining a minimum authentication level of BLOCKED. This means that no accesses are possible and can be useful as a means of deactivating a resource temporarily.

1.2.2 Authorization

Authorization is the process of determining if a given client may perform a specific operation on a target object. For the SI&C, this means finding out if a client has the permission to invoke an operation for a given object.

Permissions can be assigned to users directly in a security policy, but in larger systems with many permissions and with potentially changing user populations it is more convenient to manage and assign permissions based on roles. The SI&C fully supports role-based access control (RBAC), including role hierarchies.

1.2.3 Auditing

Auditing means recording security-relevant system evens. Examples for such events are a client failing to authenticate, or a rejected access request. The SI&C supports logging mechanisms for security events in the same way that the JacORB ORB logging mechanisms works. In addition to a well-defined set of security audit events a set of debug logging messages is available. Using the Policy Editor, it is possible to enable and disable each security audit event separately.

CHAPTER 2 Installation & Configuration

This chapter describes the installation and configuration of the Security Interceptors & Console (SI&C).

2.1 Prerequisites

To install the Security Interceptors for JacORB, please verify that the following software is properly installed on your target platform:

- Sun Java J2ME, CDC foundation profile 1.0.1 with Sun JSSE 1.0.3 (`jsse.jar`, `jcert.jar`, `jnet.jar`),
- JacORB 2.1.4 (`jacorb.jar`), built with JSSE security

The installer for the Policy Editor requires a JDK 1.4 installation on the target platform and will automatically verify that a suitable Sun Java installation exists. If not, it will install one.

2.1.1 What's on the CD

The CD-ROM contains the directory `sic/` with the following subdirectories:

- `lib/`: contains the Security Interceptors' libraries
- `doc/`: contains this Administrator's Guide and the Release Notes
- `policyeditor/`: contains the Policy Editor (the configuration console for the Security Interceptors)
- `examples/`: contains sample applications

Mounting the CD ROM on Linux and Solaris

Mounting on Linux

On Linux, you can usually mount the CD to make it available to the system by typing:

```
mount /cdrom
```

If this does not work, consult your operating system manuals. After mounting, the files contained on the CD will be available in the directory `/cdrom`.

Mounting on Solaris

On Solaris the CD is usually mounted automatically under `/cdrom`.

If not, use the command `volcheck`. If the Volume Manager is not running, determine the device name of the CD drive, and enter the following commands to mount the CD:

```
mkdir /cdrom/idbc_cd
/usr/sbin/mount -f hsfs -r /dev/dsk/cddevice /cdrom/idbc_cd
```

2.2 Installing the Security Interceptors

To install the Security Interceptors, copy the following files from the `sic/lib/` directory of the CD-ROM into the `lib/` subdirectory of your JacORB installation directory:

- `sic.jar`
- `commons-digester.jar`

2.3 Installing the Policy Editor

The Policy Editor can be installed in a Windows, Linux, or Solaris environment. On the CD ROM you will find the directories `linux/`, `solaris/`, and `windows/` below the `policyeditor/`directory. They contain the executable that will install the Policy Editor on the corresponding platform.

2.3.1 Installation Steps

Execute the installer (`PolicyEditor.bin` for Linux and Solaris and `PolicyEditor.exe` for Windows) to start the installation. The installer will lead you through the set up process:

1. Choose a language
2. Introduction
3. Choose an Install Folder. The default destination is:
   - on Windows: `<Program Files>\xtradyne\`
   - on Linux: `/usr/xtradyne/`
   - on Solaris: `/opt/xtradyne/`
4. Choose a Java VM or install one: The Policy Editor requires a JDK 1.4 installation on the target platform. If no suitable installation exists, please choose to install one.
5. Installation Summary

Installation Overview

After the installation is complete you will find the following directories in the destination install directory (if you choose to install all available install sets): `policyeditor`, `doc`, and `jre`.

`policyeditor/`

The `policyeditor` directory contains the Policy Editor: You will find the executables `PolicyEditor` in the `bin` directory. The `lib` directory contains the archives of the Policy Editor and other third party libraries. You can uninstall the Policy Editor by executing the file `UninstallPE`.

`doc/`

In the `doc` directory you will find the file `AdminGuide.pdf` containing this Administrator's Guide.

`jre/`

The directory `jre` contains the Java runtime environment required for the Policy Editor. This is an optional component.

2.4 Configuration

To protect CORBA servers with the SI&C, JacORB servers require some additional ORB configuration properties that are not required for standard JacORB servers.

Optionally, clients may also need configuration, depending on the security requirements. Clients that need to use SSL require some standard JacORB SSL configurations, and clients that want to pass a username and password require specific SI&C properties.

2.4.1 Configuration Files

The most convenient mechanism to apply the required configuration settings is by using an additional configuration property file, which is also known as a "custom properties" file in the JacORB documentation.

This is the recommended way to configure the ORB, but it is also possible to enter the properties directly into the standard `jacorb.properties` file, or to pass them through the Java system property mechanism on the command line, using the `-Dpropname=value` syntax.

2.5 Basic Server Configuration

The following section explains the properties used to configure SI&C.

2.5.1 Required Properties

The following ORB configuration properties are minimally required to use the SI&C runtime with JacORB. These properties are always required with the values as given below and must be present exactly as shown:

```
# PrismTech initializer to set up server-side interceptors
# REQUIRED SETTING, DO NOT EDIT!
# NOTE: this property has no value, just a key
org.omg.PortableInterceptor.ORBInitializerClass.com.pris-
mtech.corba.security.ServerInitializer=
```

2.5.2 Optional Properties

Enabling Access Control

The following property allows to disable SI&C access control completely. Defaults to "on".

```
prismtech.sec.enable_access_control=on
```

Policy File

The following property points to a custom access policy file that was created using the Policy Editor:

```
prismtech.sec.access_policy_file=POLICY_FILENAME
```

Dynamic Policy Updates

To allow the SI&C runtime to detect dynamic updates to the policy file and reconfigure the server security with the new policy, set the following property to “on”:

```
# set to on if servers should detect updates to the policy file
# and reconfigure automatically
prismtech.sec.configure.detect_updates=on
```

If this property is not present or set to “off”, no dynamic updates will be respected.

Logging

SI&C uses the JacORB logging facilities. Please consult the JacORB documentation for details on logging configuration like logfiles, output format, etc. The log verbosity is a value between 0 and 4, with 0 meaning no output and 4 meaning debug-level verbosity. The following different loggers are used by SI&C:

```
# default verbosity, applies e.g., to configuration output
prismtech.sec.log.verbosity=1
# output log verbosity for credential-processing
prismtech.sec.creds.log.verbosity=1
# output log verbosity for the CSIv2 TSS
prismtech.sec.tss.log.verbosity=1
# output log verbosity for the CSIv2 CSS
prismtech.sec.css.log.verbosity=1
# output log verbosity for the SSL layer
prismtech.sec.jsse.log.verbosity=1
# output log verbosity for configuration file processing
prismtech.sec.config
# output log verbosity for authentication information
prismtech.sec.auth
# output log verbosity for access control information
prismtech.sec.access
```

SSL Support

SSL is configured using the standard JacORB SSL properties. The main SSL property that must be set is the following:

```
jacorb.security.support_ssl=on
```

On the server-side, the following properties specify the SSL options that a server supports and requires. Note that the client settings are required as well.

```
jacorb.security.ssl.server.supported_options=60
jacorb.security.ssl.server.required_options=20
jacorb.security.ssl.client.supported_options=20
jacorb.security.ssl.client.required_options=20
```

The values for these properties are hex values (without a leading 0x) and are calculated by summing up the values for individual options from the following set:

NoProtection = 1

EstablishTrustInClient = 40

EstablishTrustInTarget = 20

I.e. setting the value 60 means EstablishTrustInClient and EstablishTrustInTarget, in other words mutual authentication. Please refer to the JacORB documentation for more details on theses settings.

In addition to these SSL settings, the following properties MUST be used whenever SSL support is switched on:

```
# the qualified classname of the ssl server socket factory class
# REQUIRED SETTING, DO NOT EDIT
jacorb.ssl.server_socket_factory=com.prismtech.corba.jacorb.SSL-
ServerSocketFactory

# qualified classname of the ssl socket factory class
# REQUIRED SETTING, DO NOT EDIT
jacorb.ssl.socket_factory=com.prismtech.corba.jacorb.SSLSocket-
Factory

# ORBInitializer for the SSLClientCurrent
# REQUIRED SETTING, DO NOT EDIT
# NOTE: this property has no value, just a key
org.omg.PortableInterceptor.ORBInitializer-
Class.com.prismtech.corba.security.SSLClientCurrentInitializer=
```

For JSSE SSL-level debug output, it is necessary to pass the following property explicitly (not in the properties file!) to the JVM (or CVM):

```
 -Djavax.net.debug=ssl
```

CSIv2 Identity Assertions

```
prismtech.sec.csiv2.enable_identity_assertions=on
```

Set this property to "on" in order to enable support for CSIv2 identity assertions in the SI&C. If set to "off" (the default), all IdentityTokens sent by the client will be ignored. Apart from changes in the message processing behaviour, this property additionally controls specific settings in the server's IOR. That means after changing this property and restarting the server, the IOR needs to be redeployed to clients to properly reflect the server's settings.

If enabled, IdentityTokens will be dealt with in the following manner:

- Only tokens of type ITTAnonymous and ITTPrincipalName are processed. All other token types are ignored.
- If the client can be successfully authenticated, its identities established through authentication are replaced by the asserted identity. For ITTAnonymous, the new identity will be the PUBLIC role.
- If the authentication fails, the asserted identity is treated as absent.

SSLSessionCache

```
prismtech.sec.ssl_session_cache_size=25
```

This property controls the size of the cache for credentials associated with an SSLSession. If set to "0", the cache size is unlimited. Note that there is no automatic ageing and cache purging, i.e. entries will only be removed when the limit is reached and a new element is to be added. The default size is 25.

2.6 JacORB Client Configuration

Clients require much less configuration effort and can even be left unconfigured in some cases. In general, clients do need configuration for SSL und username/password usage. For client ORBs other than JacORB, please see the vendor documentation on setting up SSL and CSIv2 GSSUP tokens.

2.6.1 Username/Password Authentication

To allow JacORB clients to send user names and passwords at all, the following property setting is required:

```
# This initializer installs the CSS interceptor
org.omg.PortableInterceptor.ORBInitializerClass.com.prismt
ech.corba.security.csiv2.CSSInitializer=
```

2.6.2 SSL Configuration

For client authentication, set up a Java keystore with a valid X.509 certificate chain. The first certificate must contain the client's DN, the final certificate must be a trusted certificate.

The following three properties must be present exactly as shown below to enable JacORB SSL clients:

```
jacorb.security.support_ssl=on

# qualified classname of the ssl socket factory class
# REQUIRED SETTING, DO NOT EDIT
jacorb.ssl.socket_factory=com.prismtech.corba.jacorb.SSLSocket-
Factory

# ORBInitializer for the SSLClientCurrent
# REQUIRED SETTING, DO NOT EDIT
# NOTE: this property has no value, just a key
org.omg.PortableInterceptor.ORBInitializer-
Class.com.prismtech.corba.security.SSLClientCurrentInitializer=
```

In addition, the SSL client options are configured similarly to the server options explained above, e.g.:

```
jacorb.security.ssl.client.supported_options=60
jacorb.security.ssl.client.required_options=60
```

Alternatively you can use a socket factory that allows you to configure the range of ports it should use. Just set/modify the following properties on the client side to enable it:

```
# REQUIRED SETTING, DO NOT EDIT
jacorb.ssl.socket_factory=com.prismtech.corba.jacorb.PortRangeSoc
ketFactory
# REQUIRED SETTING, set to appropiate value
jacorb.ssl.socket_factory.port_min=<min>
```

```
#REQUIRED SETTING, set to appropiate value
jacorb.ssl.socket_factory.port_max=<max>
```

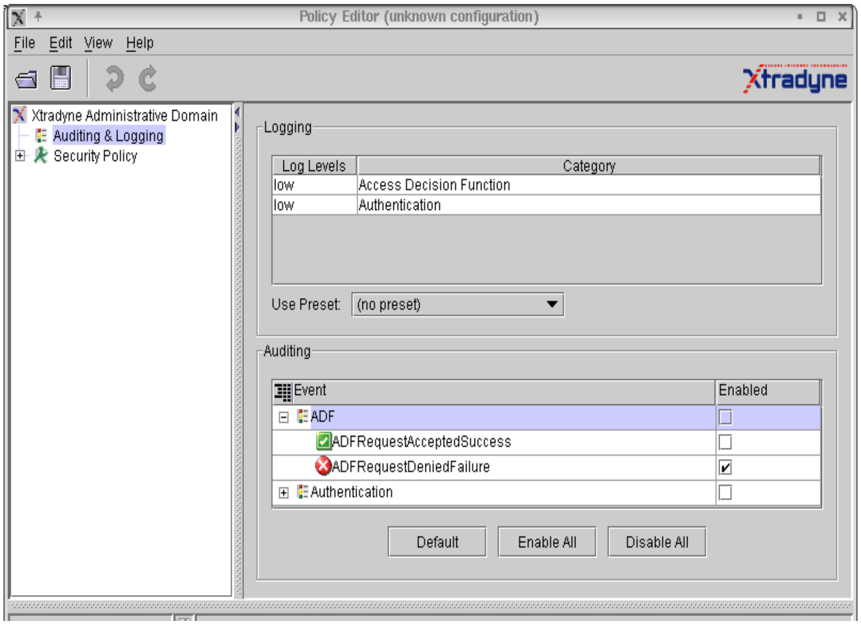
CHAPTER

3 Policy Editor

The Policy Editor is a tool to conveniently create and manage policies for the SI&C. The first section of this chapter gives detailed installation instructions. The subsequent chapters describe the panels of the Policy Editor.

3.1 Running the Policy Editor

To run the Policy Editor execute `<INSTALLDIR>/bin/PolicyEditor`.



Log Levels	Category
low	Access Decision Function
low	Authentication

Use Preset: (no preset)

Event	Enabled
ADF	☐
ADFRequestAcceptedSuccess	☐
ADFRequestDeniedFailure	☑
Authentication	☐

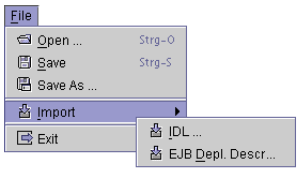
Fig. 1. Policy Editor

3.1.1 General Navigation

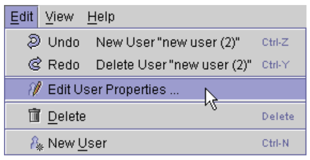
Tool Bar Icons

	Open a configuration from a local file.(**File➜Open**).
	Save the configuration data to a local file (**File➜Save**).
	Undo and Redo the last local modification. The undo stack is unlimited, which means you can undo as many changes as you wish. (**Edit➜Undo**, **Edit➜Redo**)

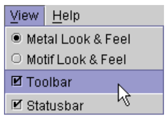
Policy Editor Menus



The Policy Editor can store and retrieve configuration data from files (**Open ...**, **Save ...**, **Save As ...**). You can import security policies from IDL files and from EJB Deployment Descriptors.


Edit View Help
Undo New User "new user (2)" Ctrl-Z
Redo Delete User "new user (2)" Ctrl-Y
Edit User Properties ...
Delete Delete
New User Ctrl-N

The Edit Menu is also context sensitive. Depending on the selected item in the navigation tree the Edit Menu will adapt. This example shows the entries of the Edit Menu when a single user in your security policy is selected.



You can change the Java Look & Feel and hide the tool bar and status bar from view.

3.2 Auditing

3.2.1 Auditing

An audit policy specifies which auditable events are logged. Event notifications are messages created by the SI&C.

Introduction

recording security relevant events

In addition to specifying security policies for the SI&C, it is also necessary to monitor the system behavior. In order to determine exactly what went wrong and why, for example, a perceived breach of security was not prevented by the mechanisms and policies in place, an *audit log* is required. An audit log is a record of security-relevant events that the system observed and that can be analyzed to determine the effectiveness of security services as well as the reasons and circumstances of system failures. The SI&C supports recording events in audit logs.

Audit Events

Event notifications can be customized using the log event pattern string as described in section “Logging” on page 13. Typically, they contain at least name denoting the event and potentially a time stamp. Most notifications carry additional event-specific information.

Audit Event Types

Events are separated into the following types:

- **Success Audit** events: are security events that occur when an audited access attempt is successful, for example, a successful logon attempt.
- **Failure Audit** events: are security events that occur when an audited access attempt fails, for example, a failed logon attempt.

Event Flow

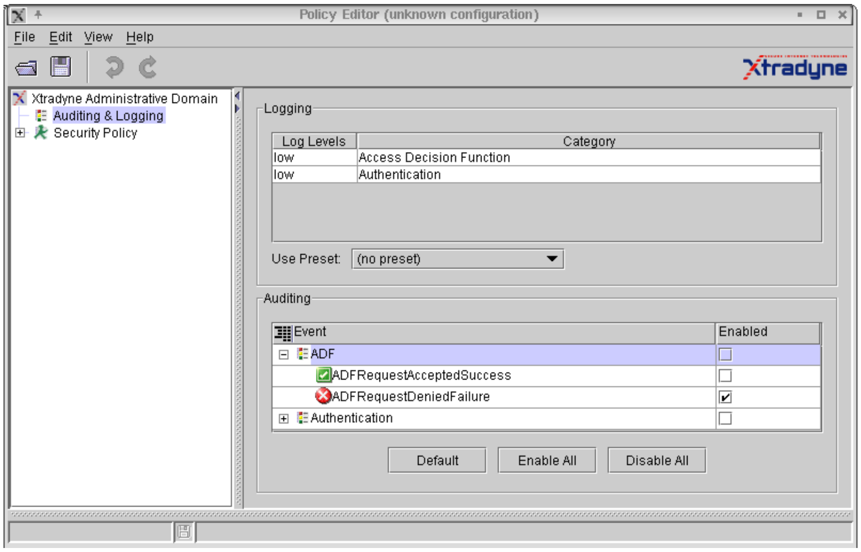
The SI&C uses the normal JacORB logging mechanisms but writes log messages to a separate file, i.e., it does not use the standard ORB output log file. The name and maximal size of the SI&C log file is configured using configuration properties, please see “Logging” on page 13 for details. If a maximal file size is set and the log file has reached this size, the log will rotate to another file.

3.2.2 Audit Policy

The set of events that are considered relevant are specified in an *audit policy*. The SI&C will only generate notifications for events that are selected from the set of auditable events. Skipping the creation of notifications that are not relevant improves the overall system performance.

enabling and disabling events

On the “Audit Policy” panel the audit policy can be managed and adjusted at run-time to select or deselect events as required. Administrators might, for example, want to see all events until they are reasonably confident that the system works as expected, then disable all notifications which do not indicate relevant failures. In the reverse case, they can simply “switch on” previously disabled events if they need to diagnose specific issues.



Log Levels	Category
low	Access Decision Function
low	Authentication

Use Preset: (no preset)

Event	Enabled
ADF	☐
ADFRequestAcceptedSuccess	☐
ADFRequestDeniedFailure	☑
Authentication	☐

Fig. 2. Configuring Audit Events

Auditing

In the lower part of the panel a list of available events is displayed. You can enable or disable the delivery of the corresponding notification.

Per default all events that indicate failure are enabled. To diagnose problems with a SI&C configuration, it might be useful to enable more notifications. Below the event list there are some buttons for your convenience to:

default events

- reset to the default setting,
- enable all notifications,
- disable all notifications.

Audit Event Categories

There are two event categories: ADF (Access Decision Function) and Authentication.

- Authentication Events: mechanism, success and outcome, failure
- Authorization Events: access allowed/denied

ADFRequestDeniedFailure, ADFRequestAcceptedSuccess

A request has been accepted or rejected by the access decision function (ADF). Possible reasons for a failure are:

- The client's authentication level is insufficient: Check the value of the authentication level ("ALA") in the event details. E.g., the client's ALA is "UserId/Password" but the resource's incoming policy has been configured to require SSL authentication. Adapt the client to use the required ALA or change the resource's authentication policy with the Policy Editor.
- The client has insufficient access rights: Access is denied following the rules in the resource's access control policy, see "Resource Properties – Accessors" on page 43 on how to assign the user access rights to a resource.
- There is no policy granting permissions for the requested resource at all.

AuthenticationBasicAuthenticationFailure, AuthenticationBasicAuthenticationSuccess

The success event indicates that the SI&C successfully authenticated a user with Basic Authentication.

The failure event indicates that the SI&C failed to authenticate user with Basic Authentication. This can have the following reasons:

- the client provided a user name which is unknown to the SI&C,
- the client provided the wrong password.

SSLAuthenticationCertificateFailure, SSLAuthenticationCertificateSuccess

The success event indicates that the SI&C successfully authenticated a user with an SSL certificate.

The failure event indicates that the client's SSL certificate is trusted and valid, but the SI&C failed to recognize it because the user is not known to the SI&C. Start the Policy Editor, go to the "Security Policy – Users" panel, and check that the user is included in the list (section "User Properties – General" on page 28 for details). Also check if the DN in the certificate equals the one given for this user (consult the "Certificate DN" property of this event).

Note that the failure event does not necessarily indicate a fatal failure, The SI&C tries all configured authentication methods for a user until it finds the proper one for this client. For example: There are two authentication methods defined for a user in the Policy Editor: SSL authentication and authentication by IP-address. The user contacts the SI&C authenticating by his IP-address. The SI&C will first try to authenticate the user with SSL. This will fail and an *SSLAuthenticationCertificateFailure* event will be logged, if configured. The SI&C will then try to authenticate the user by his IP-address. This will succeed.

IPbasedAuthenticationFailure, IPbasedAuthenticationSuccess

The failure event indicates that the SI&C failed to authenticate a user by IP address. The success event states that authenticating a user by IP address was successful.

CHAPTER 4 Security Policies

The SI&C relies on the concept of Role-Based Access Control (RBAC). The first section of this chapter gives a general introduction to this concept. The subsequent sections explain how to use the Policy Editor to define security policies.

4.1 Access Control

Access Control must consider both an application and its users. We divide the description of Access Control information in two parts:

- the application side, where the rights to send operations to Services are defined and combined into roles, and
- the user management side, where users can be assigned to roles.

This distinction is similar to the Enterprise Java Beans (EJB) separation of administrative concerns between the application developing domain (Bean Provider, Application Assembler), the production environment (System Administration), and the Deployer who links the two domains.

4.1.1 Access Control Policy

The access control policy comprises three components: Users, Roles, and Resources. Figure 1, “Components of the access control policy”, depicts the relationship between Users, Roles, and Resources.

Resources, Roles, Users

Resources represent CORBA Services that are identified by an Object Reference (IOR). A Resource has accessors, i.e. one or more Roles (see association ③ in figure 1). Generally a Role can access a Resource by sending operations. Roles represent application tasks and receive the necessary *permissions* to invoke *operations* on Resources to fulfill these tasks. Roles propagate these permissions to other Roles or Users. If a role Y has another role X as actor (association ② in figure 1) role X inherits Y's permissions. Hence, Y can be viewed as "junior" to X as X has all permissions Y has, but may have additional permissions of its own.

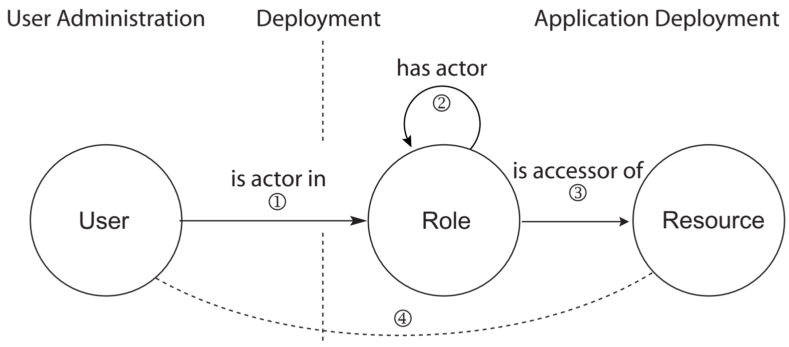


Fig. 1. Components of the access control policy

Role Engineering

We recommend that Users are not directly granted access to a Resource ④, but are assigned to Roles ①. This approach encourages you – prior to assigning permissions with the Policy Editor – to think about the abstract structure behind your policy. The full advantages of Role-Based Access Control (especially the reduced maintenance cost) can only be achieved by careful role-engineering. For flexibility, "short-cuts" are allowed (dotted line in figure 1), but we recommend that they are used only sparingly!

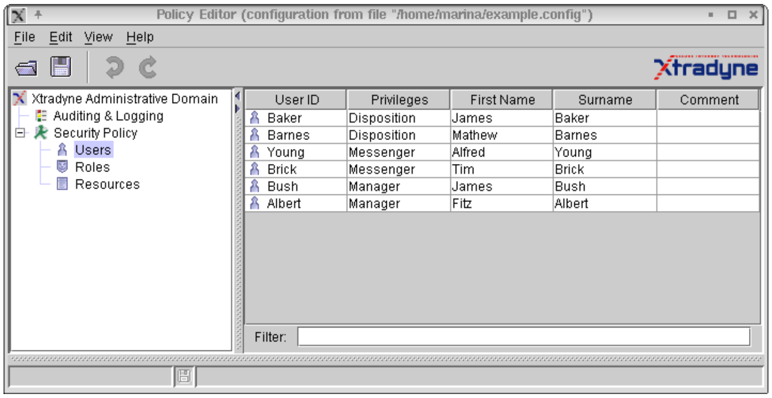
4.2 Defining Security Policies

The following sections describe how to use the Policy Editor to define the following policies:

- The *access control policy* defines which user has access to which resource (either for the whole resource interface or, optionally, for individual operations offered by the Service). Access control in the SI&C uses the concept described in the previous section.
- The *authentication policy* defines how the client must authenticate by setting a minimum authentication level for a resource.

4.3 General Navigation

The Security Policy offers three views, one for each kind of entity in the model: a list of users, roles, and resources (click on the ⊞ next to the security policy icon on the left side of the Policy Editor). You can switch between these views by clicking on the entries in the navigation tree (left side of the panel). This displays a list of the respective entities. To view or change the entities, double-click on the entity, or use the menu **Edit➜Edit Properties** or the context menu (via the right mouse button). Multiple entities can be selected by pressing the CTRL-key (selecting one by one) or the SHIFT-key (selecting a range of entities).


Policy Editor (configuration from file "/home/marina/example.config")
File
Edit
View
Help
Xtradyne
Xtradyne Administrative Domain
Auditing & Logging
Security Policy
Users
Roles
Resources
Filter:

User ID	Privileges	First Name	Surname	Comment
Baker	Disposition	James	Baker	
Barnes	Disposition	Mathew	Barnes	
Young	Messenger	Alfred	Young	
Brick	Messenger	Tim	Brick	
Bush	Manager	James	Bush	
Albert	Manager	Fitz	Albert	

Fig. 2. Screenshot of the Policy Editor – Example User List

using the filter

If you are looking for a specific entity, you can use the filter. Type the first few characters of the entity's ID in the "Filter" text field, between the panes and the entity list. The list shortens immediately, and only matching entities are displayed. Note that the filter is case insensitive and applies only to the ID field of the "User", "Roles", and "Resources" panel.

sorting entries

Entries can be sorted for each column in increasing or decreasing order. Just click on the heading of the appropriate column once or twice. The filter will then apply to the sorted column.

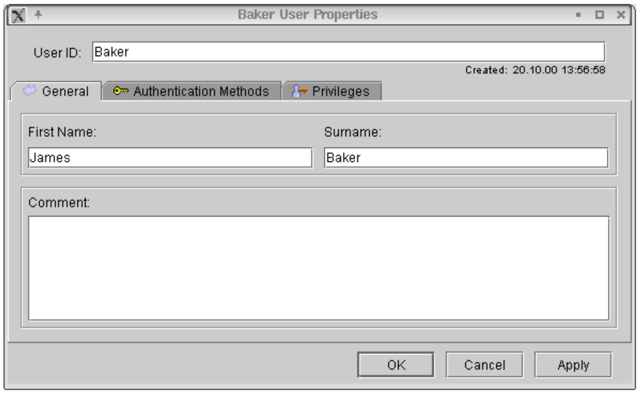
The following sections describe the properties that can be defined for each of the entities. Pictograms are often used in the text, for example denotes the entity user, roles, and resources.

4.4 Users

In general, each entity stores two kinds of information: mandatory data, which is essential for the functioning of the SI&C, and optional descriptive information. For example, a user's first name, surname, and a comment are not vital for the SI&C, but convenient for you to remember who that user was. An entity's name can include alphanumeric characters and additionally the following characters: "=", ".", ":", " ".

4.4.1 User Properties – General

To create or edit a new user, double-click on the entity, or use the menu **Edit➜Edit Properties** or the context menu (via the right mouse button). A window pops up which comprises three tabbing panes (see Figure 3, "General User Properties"): The **General** tabbing pane defines the personal data of a user and the activation and expiration date (optional). The **Authentication Methods** tabbing pane defines by which means the user must authenticate to the system. The **Privileges** pane defines the user's relationship with other entities, for example the user's memberships in certain roles.



User ID: Baker
Created: 20.10.00 13:56:58
First Name: James
Surname: Baker
Comment:

Fig. 3. General User Properties

Each user has a unique *User ID*, a *First Name*, *Surname*, and a *Comment*. First name, surname, and comment are optional information. The User ID identifies a user in the whole system.

Discard your changes by clicking the **Cancel** button, accept your changes with **OK** (the window will disappear) or **Apply** (the window will stay open).

4.4.2 User Properties – Authentication Mechanisms

For the system to realize who a user is, he or she must authenticate. The SI&C offers several ways of authenticating a user: User ID/Password authentication (also called Basic Authentication), IP-based authentication, and SSL authentication. In the following we will explain these methods and the corresponding configuration panel.

authentication policy

To configure the available authentication methods for a particular user, go to the Authentication Methods panel in the "User Properties" window. Right-click into the field "Applied mechanisms" and choose an appropriate authentication mechanism from the context menu. The properties of each authentication mechanism can be defined on the right side of the panel.

SSL X.509 Certificate Authentication

X.509 is a standard for digital certificates used by SSL. Certificates include among other things information about the identity of the certificate holder (i.e. the user). Chosing this authentication method implies that the client application has to use SSL, which additionally provides integrity and, optionally, confidentiality protection for messages in transit. Please use the JacORB SSL configuration properties for defining exactly how SSL is to be used (cf. section "SSL Support" on page 14).

SSL authentication and HTTPS

On the right side of the pane you give the distinguished name (DN) of the certificate the client uses to authenticate to the SI&C (single elements of the distinguished name have to be separated by a comma).

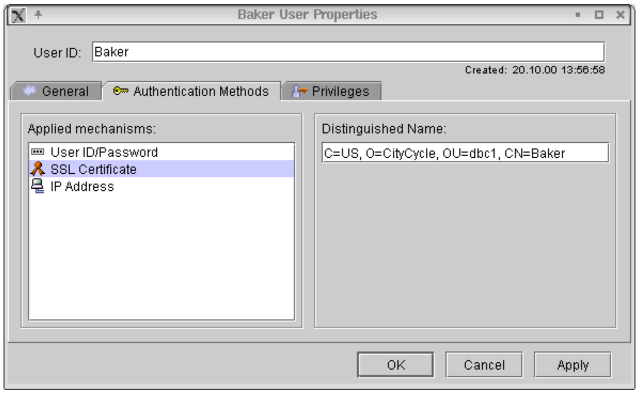


Fig. 4. Authentication Methods: X.509 Certificate authentication

Note that the client certificate DN must be unique, please ensure that multiple users do not share the same client certificate DN.

User ID/Password Authentication

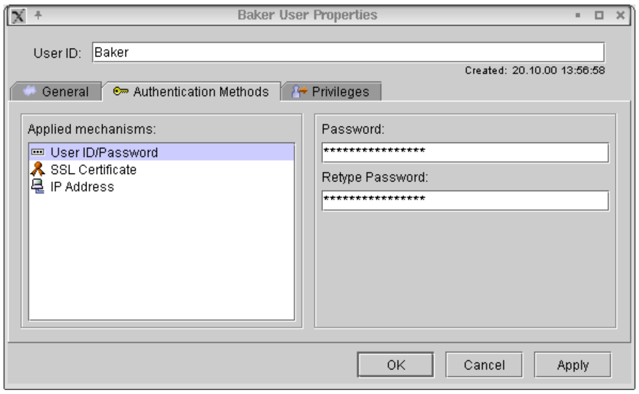
user ID/password authentication

A user can authenticate via a user ID/password scheme. The user ID to be supplied during the authentication process must be the same as the user ID defined here.

Configure the user ID and password here for the respective user. The user ID is displayed in the upper part of the panel. Provide the password (at least 4 characters long) on the right hand side of the panel.

Note that when a user forgets his password, there is no way of reconstructing it because only a SHA-1 hash is stored in the policy. In such a case the administrator has to assign the user a new password with the Policy Editor.

Also note that the number of asterisks displayed in the Policy Editor does *not* correspond to the number of letters contained in the password so that the actual length of an assigned password cannot be observed by an onlooker.


Baker User Properties

User ID: Baker

Created: 20.10.00 13:56:58

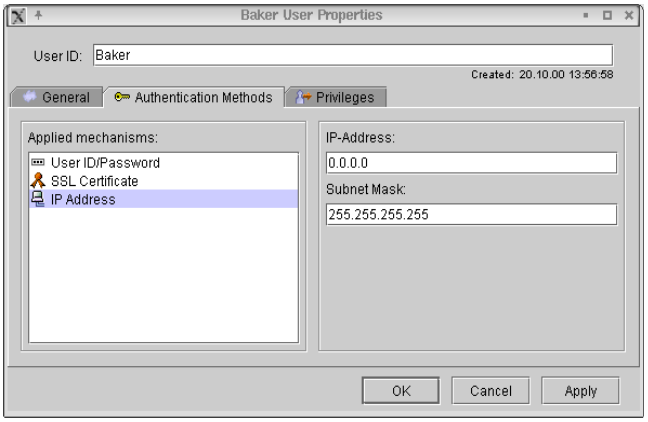
Password: ****************

Retype Password: ****************

Fig. 5. Authentication Methods: User ID / Password Authentication

IP Address

In case a particular user always connects from a single IP address or a range of IP addresses, you can map the IP address directly to the user ID. In this case, the user ID really corresponds to a name for a host or a subnet. A subnet mask can be provided which *identifies* – rather than masks out – the relevant bits of the *policy IP* address that are compared to the *source IP* address. The policy IP address is the one defined in the panel. The source IP address is the one from where the client connects.


Baker User Properties

User ID: Baker

Created: 20.10.00 13:56:58

Applied mechanisms:
- User ID/Password
- SSL Certificate
- IP Address

IP-Address: 0.0.0.0

Subnet Mask: 255.255.255.255

Fig. 6. Authentication Methods: IP Address

Using the Subnet Mask

The comparison works as follows: First the source IP is bit-wise AND-combined with the subnet mask. The result is then compared to the policy IP address, also AND-combined with the netmask. If the result is equal the IP address matches. As a formula: *(source IP & subnet mask) =(policy IP & subnet mask)*. For examples, see the table below.

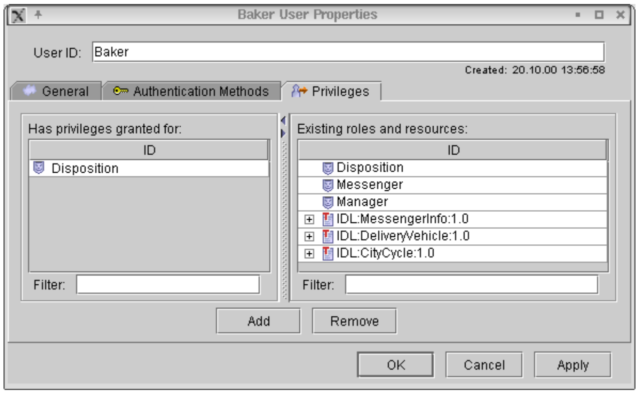
policy IP Address	**Subnet Mask**	**Description**
144.10.210.16	255.255.255.255	match exactly that IP address (default value)
144.10.210.0	255.255.255.0	matches the class C net, i.e. all host IP addresses starting with 144.10.210
144.10.210.176	255.255.255.240	matches a 4-bit subnet 144.10.210.176 – 144.10.210.255
144.10.210.7	255.255.255.0	does not match as zero bits in the subnet mask are not zero in the originator IP address

Table 1. policy IP address and subnet matching

You should be aware that this kind of authentication is vulnerable to IP spoofing attacks but may be sufficient within a corporate network. It can be useful, for example, if a partner organization has a fixed IP address range and the whole partner organization should be mapped onto the same user ID.

4.4.3 User Properties – Privileges

On the **Privileges** tabbing pane you can choose those entities from the list of existing roles and resources (on the right side of the panel) for which the user has privileges (left side of the panel). Privileges subsume role memberships and access permissions.



Baker User Properties

User ID: Baker

Created: 20.10.00 13:56:58

Has privileges granted for:

ID
Disposition

Existing roles and resources:

ID
Disposition
Messenger
Manager
⊞ IDL:MessengerInfo:1.0
⊞ IDL:DeliveryVehicle:1.0
⊞ IDL:CityCycle:1.0

Fig. 7. User Properties: Privileges

If a resource accepts specific operations, you can view these operations by clicking on the ⊞ next to the resource. You can grant privileges to invoke operations on a resource or for single operations. When double clicking on an operation the “Resources – Accessors” panel for that operation will pop up (see also page 43).

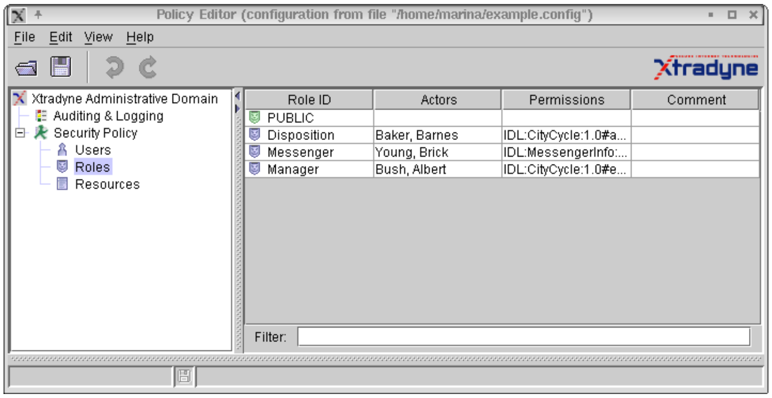
All memberships of the user are displayed in the left window and all known roles and resources are listed in the right window.

granting privileges for the selected user

One or more entities can be selected by pressing the CTRL-key (selecting one by one) or the SHIFT-key (selecting a range of entities). Use the filter to shorten the displayed list. While you are typing, the list is reduced to the entities with names that begin with the letters you typed. In order to grant a user privileges for other entities, select the entities in the window on the right side and press the “Add” button. The entities will now appear on the left side of the **Privileges** panel.

4.5 Roles

The “Roles” panel shows the list of roles, with the role ID, the role’s actors, their permissions, an expiration date, and a comment. A role’s actors can be an arbitrary number of users and other roles. A role’s permissions are the resources that a role is granted to access.


Policy Editor (configuration from file "/home/marina/example.config")
File
Edit
View
Help
Xtradyne
Xtradyne Administrative Domain
Auditing & Logging
Security Policy
Users
Roles
Resources
Filter:

Role ID	Actors	Permissions	Comment
PUBLIC			
Disposition	Baker, Barnes	IDL:CityCycle:1.0#a...	
Messenger	Young, Brick	IDL:MessengerInfo:...	
Manager	Bush, Albert	IDL:CityCycle:1.0#e...	

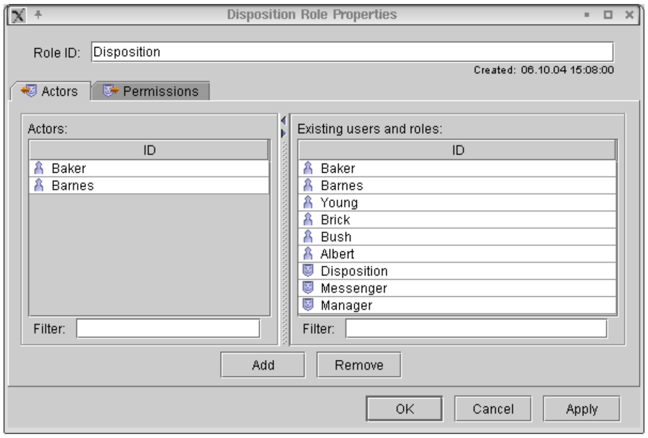
Fig. 8. Policy Editor – Role List

PUBLIC Role

Note the special role “PUBLIC”. This is a predefined role which cannot be deleted or created. The “PUBLIC” role cannot grant permissions to other entities explicitly. It is designed for granting general access to public resources or to public operations of a resource (for more details see “Public Access” on page 45).

4.5.1 Role Properties – Actors

To create or edit a new role, double-click on the entity, use the menu **Edit➔Edit Properties**, or the context menu (via the right mouse button).



Role ID: Disposition

Created: 06.10.04 15:08:00

Actors:

ID
Baker
Barnes

Filter:

Existing users and roles:

ID
Baker
Barnes
Young
Brick
Bush
Albert
Disposition
Messenger
Manager

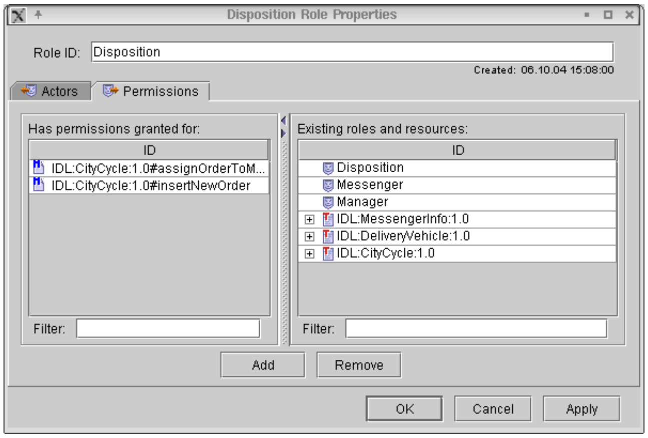
Filter:

Fig. 9. Role Properties: Actors

Roles have **Actors** to whom they pass on the permissions received from resources or roles. Actors are roles and users (see also Figure 1, “Components of the access control policy,” on page 26).

4.5.2 Role Properties – Permissions

The role's Permissions tabbing pane is similar to the user's privileges tabbing pane. Roles receive permissions from resources or other roles (see screenshot below).


Disposition Role Properties

Role ID: Disposition

Created: 06.10.04 15:08:00

Actors | Permissions

Has permissions granted for:

ID
IDL:CityCycle:1.0#assignOrderToM...
IDL:CityCycle:1.0#insertNewOrder

Filter:

Existing roles and resources:

ID
Disposition
Messenger
Manager
⊞ IDL:MessengerInfo:1.0
⊞ IDL:DeliveryVehicle:1.0
⊞ IDL:CityCycle:1.0

Filter:

Add | Remove

OK | Cancel | Apply

Fig. 10. Role Properties: Permissions

As usual there is a ⊞ next to the resource if it accepts specific operations. In this case permission can be granted for the whole resource or for single operations.

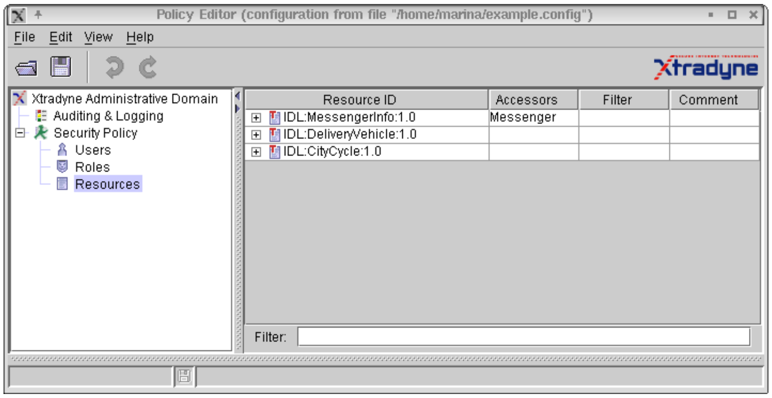
Again, you can choose those entities from the existing roles and resources on the right side for which the role "Has permissions granted for" (the panel on the left side).

granting permissions for the selected role

To grant a role permissions to access resources or memberships in other roles, select the entities in the window on the right side and press the "Add" button. The entities will now appear on the left side of the Permissions tabbing pane. As explained earlier, granting a role X membership to the role Y causes X to inherit Y's permissions. Hence, X can be viewed as "senior to" Y as it has all permissions that Y has, but may have additional permissions of its own.

4.6 Resources

Resources point to target services. These are CORBA Services specified by their IOR.


Policy Editor (configuration from file "/home/marina/example.config")
File
Edit
View
Help
Xtradyne
Xtradyne Administrative Domain
Auditing & Logging
Security Policy
Users
Roles
Resources
Filter:

Resource ID	Accessors	Filter	Comment
IDL:MessengerInfo:1.0	Messenger		
IDL:DeliveryVehicle:1.0			
IDL:CityCycle:1.0			

Fig. 11. Policy Editor: Resource List

Adding a Resource

To add a new resource, select **New IDL Type...** or **New Corba Object...** from the context menu (right-click in the resources table).

service operations

New operations for a resource can be added directly on the “Resources” panel using the context menu. If an EJB deployment descriptor or an IDL document describing the interface that you want to expose is available the exposure wizards may be used instead of defining operations manually (refer to section “Exposure Wizards” on page 37). Operations offered by a resource can be viewed by clicking the ⊞ next to the resource. When double-clicking on an operation the **Accessors** panel for this operation will pop up (see also “Resource Properties – Accessors” on page 43). Instead of defining a resource with all its operations manually, you can use the exposure wizard. This is explained in the next section.

Resource Types

instance- and type-based definition of resources

There are two types of CORBA resources – type-based and instance-based resources. Type-based resources are defined by their IDL Type (usually: `IDL:<pragma prefix/module-hierarchy>/<interface>:<version>`). In the above example, this is the top-level interface “CityCycle” in the default version “1.0”. In CORBA IDL, interfaces can inherit from other interfaces, but the Policy Editor does not support inheritance hierarchies. Only rules for IDL types that match exactly are applied.

Instances are defined by `<hostIP>:<port>/<objectkey>`. The host must be specified by its IP address. The object key must be given in URL encoding. In the example above the object runs on 192.168.47.11 on port 18011 and the object key is `CCImpl/CCPOA/Rees`.

Note that if both an IOR-based resource and an IDL-type-based resource are found for a run-time request, the policy for the IOR-based one takes precedence.

4.6.1 Exposure Wizards

The Policy Editor offers exposure wizards to import IDL descriptions of a resource. Furthermore EJB Security Policies can be imported into the security policy.

The IDL exposure wizard is explained in the following section. The EJB import wizard is explained on page 39.

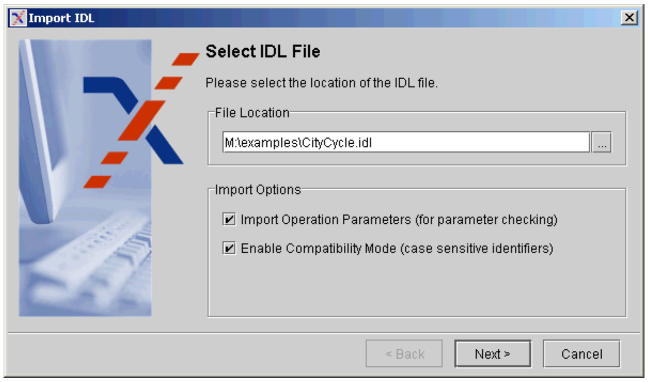
IDL Exposure Wizard

The IDL exposure wizard can be accessed via the menu item **File ➜ Import** or via the context menu on the "Security Policy – Resources" panel.

On the first panel of the IDL exposure wizard you select the IDL file.

import options

As import options you can choose to import operation parameters (in case you want to define rules for parameter checking). Furthermore, a compatibility mode is provided for IDL files which do not strictly follow the CORBA IDL standard. Use the compatibility mode only if the default mode yields an error. To proceed press the "Next" button.



Import IDL

Select IDL File

Please select the location of the IDL file.

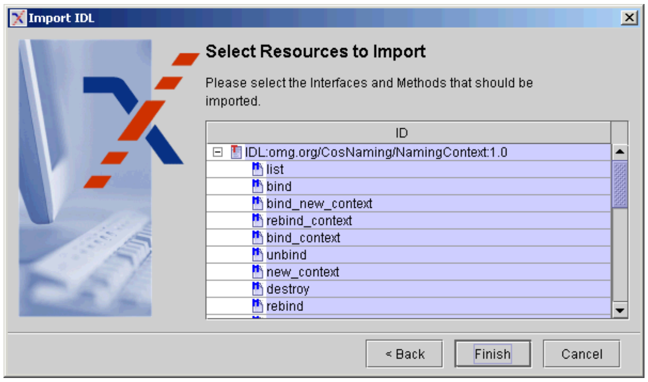
File Location: M:\examples\CityCycle.idl

Import Options

- [✓] Import Operation Parameters (for parameter checking)
- [✓] Enable Compatibility Mode (case sensitive identifiers)

Fig. 12. The IDL import wizard: Select IDL File

The wizard then extracts service descriptions from the IDL document. The next step is to choose the resources and operations that you want to import into the security policy.



Select Resources to Import

Please select the Interfaces and Methods that should be imported.

ID
IDL:omg.org/CosNaming/NamingContext:1.0
list
bind
bind_new_context
rebind_context
bind_context
unbind
new_context
destroy
rebind

Fig. 13. The IDL import wizard: Select Resources to Import

conflicts while importing

If the resource that you want to import already exists in the security policy the Policy Editor will show a panel and you can choose to “Update”, “Overwrite”, or “Skip” the conflicting resource. When choosing “Update” all policy definitions for this resource will be kept and only those entities that do not already exist in the policy will be imported. Note that if you choose “Overwrite” the resource properties will be reset with the default security policy settings!

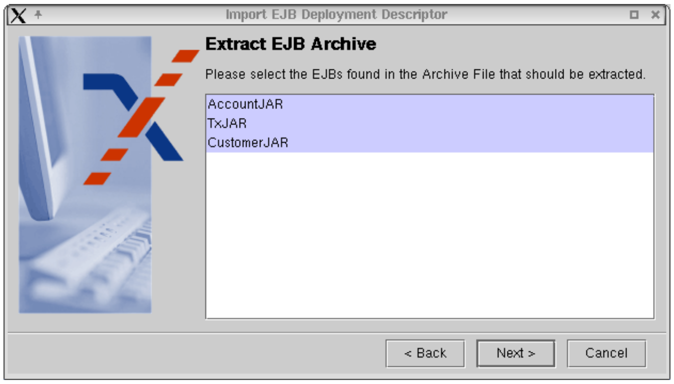
The IDL import is complete now, but note that the security policy for the imported resource should be refined: By default the imported resource has no accessors, so the SI&C will not allow any access (unless the resource already existed in the policy before importing and accessors have already been defined). Please set the security policy within the “Resource Properties” panel and associate the resource with its accessors as described in section “Resource Properties” on page 41 and following.

Importing EJB Security Policies into the Security Policy

The EJB deployment descriptor allows the application assembler to define a security policy to be enforced by the container. This policy consists of method permissions and security roles. A method permission can be a single method, a set of methods with the same name (overloaded methods), or all methods of a specified interface. Method permissions are grouped in security roles. In addition to the user defined security roles, the application assembler can declare method permissions as “unchecked”, i.e., access is granted without restrictions.

To import EJB security policies into the SI&C security policy, perform the following steps:

1. Select **File➔Import EJB Deployment Description...**. This will open a dialog where an enterprise archive containing multiple EJBs or a single EJB jar file can be selected. As import options you can choose to additionally import operations from Home- and Remote-Interfaces that are not explicitly declared in the deployment descriptor. After clicking "Open", the deployment descriptors of the EJBs contained in the selected archive will be parsed. This may take a few seconds to complete.
2. After successful parsing, a new dialog pops up. Select the EJBs that you want to import into the security policy from the available EJBs in the enterprise archive.



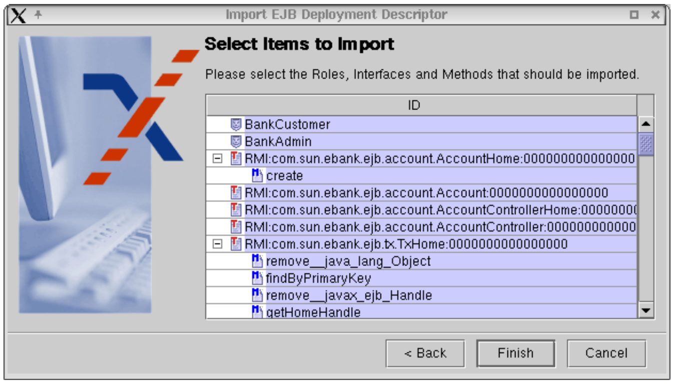
Extract EJB Archive

Please select the EJBs found in the Archive File that should be extracted.

AccountJAR
TxJAR
CustomerJAR

Fig. 14. Extract EJB Archive

3. Select roles, interfaces, and methods for import into the security policy (methods correspond to operations in the security policy).



Select Items to Import

Please select the Roles, Interfaces and Methods that should be imported.

```
ID
  BankCustomer
  BankAdmin
⊟ RMI:com.sun.ebank.ejb.account.AccountHome:000000000000000
    create
  RMI:com.sun.ebank.ejb.account.Account:0000000000000000
  RMI:com.sun.ebank.ejb.account.AccountControllerHome:0000000
  RMI:com.sun.ebank.ejb.account.AccountController:000000000000
⊟ RMI:com.sun.ebank.ejb.tx.TxHome:0000000000000000
    remove__java_lang_Object
    findByPrimaryKey
    remove__javax_ejb_Handle
    getHomeHandle
```

Fig. 15. Select Items to Import

During import some method names may have to be changed according to the Java-To-IDL Mapping. For example, IDL doesn't support method overloading, so overloaded Java method names have to be renamed in order to generate correct IDL.

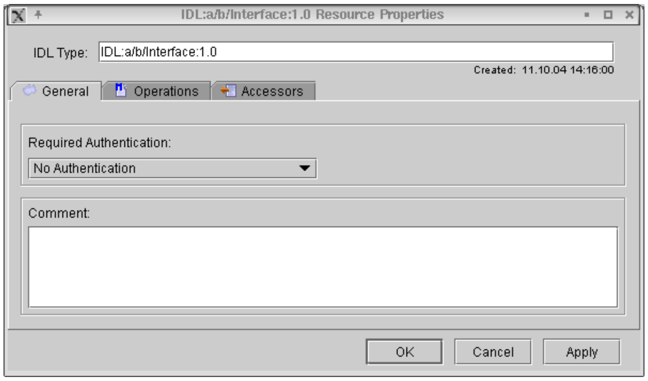
conflicts while importing

If the role or resource that you want to import already exists in the security policy the Policy Editor will show a panel and you can choose to “Update”, “Overwrite”, or “Skip” the conflicting resource. When choosing “Update” all policy definitions for this role or resource will be kept and only those entities that do not already exist in the policy will be imported. Note that if you choose “Overwrite” the resource properties will be reset with the default security policy settings!

The EJB import is complete now, but note that the security policy for the imported roles or resources should be refined. By default the imported resources have no accessors, so the SI&C will not allow any access (unless the resources already existed in the policy before importing and accessors have already been defined). Please set the security policy within the “Resource Properties” panel and associate the resources with its accessors as described in following sections.

4.6.2 Resource Properties

To edit a new resource, double-click on it or use the menu **Edit➜Edit Resource Properties** or the context menu (via the right mouse button). In either case, the “Resource Properties" panel pops up. This panel has several tabbing panes: General, Operations, and Accessors.



IDL Type: IDL:a/b/Interface:1.0

Created: 11.10.04 14:16:00

Required Authentication: No Authentication

Comment:

Fig. 16. General Resource Properties

4.6.3 Resource Properties – General

The General tabbing pane defines the security attributes of a resource: the required authentication and protection of the connection via which the resource is accessed, and time constraints.

Required Authentication

Resources can define the access mode by which they allow access to specific operations. In the General tabbing pane, you define the minimum *Required Authentication*. These levels apply to all operations of a resource. The SI&C supports the following authentication mechanisms (listed from the weakest to the strongest):

required authentication

- *No Authentication required* (anonymous access allowed)
- *Source IP Address*: authentication by IP address
- *User ID/Password*
- *SSL Client Authentication*
- *Blocked*: "emergency stop" to disallow any access

Note that the chosen authentication method is the minimum required method, i.e., if you allow access with at least *User ID/Password* authentication, access is also granted for clients using SSL X.509 credentials (if the user has permissions to access the resource this way). The SI&C will look first for authentication information for the stronger mechanism (e.g., SSL) and only consider other mechanisms if no authentication information is available for that level.

Selecting *No Authentication* means that the SI&C will accept unauthenticated messages, or any of the authentication mechanisms. Note that if a client offers an authentication method and the authentication fails, the SI&C will reject the authentication attempt even if the required authentication is only anonymous.

4.6.4 Resource Properties – Operations

The **Operations** panel allows to edit the operations of a resource. On the left side of the operations table you see the operation name. Click on the ⊞ next to an operation to view the associated parameters. To edit an operation use the context menu which pops up when right-clicking into the table. You can choose to “Edit” or “Delete” a chosen operation, or insert a “New Operation”.

Define New Operations

If no IDL document is available for import you can define the operations of a service here. Choose “New Operation” from the context menu to bring up the dialog.

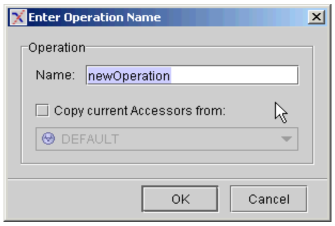


Fig. 17. Resource Properties: Enter a new operation

copy an accessor list

Enter the name of the operation for which you want to define permissions. When clicking the check box “Copy current Accessors from”, you can choose an operation from the drop down menu. The accessors of the chosen operation will be copied to the newly created operation rule (cf. “Resource Properties – Accessors” on page 43).

4.6.5 Resource Properties – Accessors

The previous sections explained how protection and authentication requirements for a resource are specified. On the **Accessors** tabbing pane you can define which entities (users or roles) are granted access to a resource. Access can be granted to an entire resource, i.e., to all its operations at once or in a more fine-grained fashion: different access rights can be defined for the different operations of a resource. These operation-specific rules override the resource default, i.e., you may define stricter or *less* (!) strict rules.

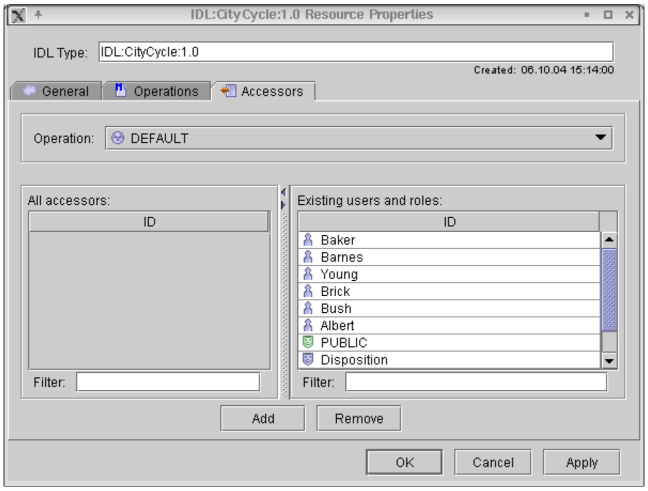
The **Accessors** tabbing pane is similar to the user's "Privileges" and the role's "Permissions" tabbing pane. Again, you can choose those entities on the right side, for which the resource grants access permissions (the panel on the left side).

Note that you can change access rights for multiple operations at the same time. Go to the "User – Privileges" panel, select the required operations and assign them to the user in one step.

4.6.6 *Configuring Accessor Lists*

access rights for a resource vs. operation-specific permissions

If you want to assign the same access rights for all operations of a resource no operation rules have to be defined. Choose the value "DEFAULT" in the operation drop down menu in the upper part of the **Accessors** tabbing pane and select those entities from the list of existing users and roles that you want to grant access to the resource (on the right side of the panel). When you press the "Add" button the selected entities will appear in the "All Accessors" list on the left side of the panel.


IDL:CityCycle:1.0 Resource Properties

IDL Type: IDL:CityCycle:1.0

Created: 06.10.04 15:14:00

Operation: DEFAULT

All accessors:

ID

Existing users and roles:

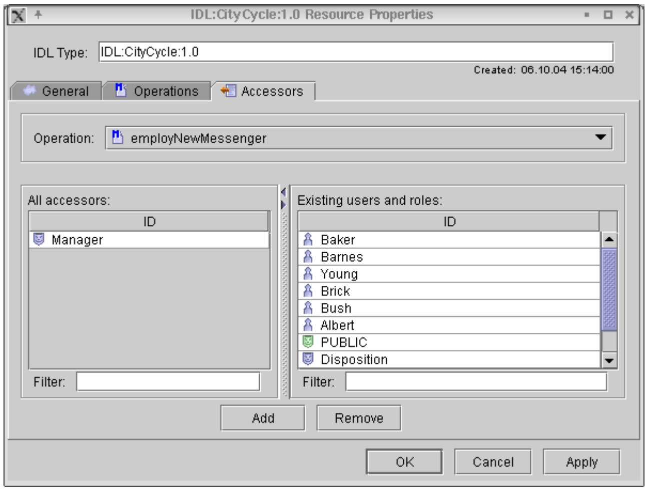
ID
Baker
Barnes
Young
Brick
Bush
Albert
PUBLIC
Disposition

Fig. 18. Resource Properties – Accessor list for the whole resource

Note that the default applies only to operations that are **not** explicitly defined in the list of operations (on the Operations tabbing pane). Delete those operations from the list of operations for which the default access rules shall apply. Note that if you delete operations for which parameter filters were defined, these filter rules will be lost.

Operation-Specific Access Rights

To define an accessor list for a specific operation, first select the operation in the drop down list (operations for a resource can be defined or imported via IDL) and then choose the users and roles from the window on the right side of the panel and press the “Add” button. The selected entities will now appear in the “All Accessors” list on the left side of the panel.



IDL:CityCycle:1.0 Resource Properties

IDL Type: IDL:CityCycle:1.0

Created: 06.10.04 15:14:00

General | Operations | Accessors

Operation: employNewMessenger

All accessors:

ID
Manager

Filter:

Existing users and roles:

ID
Baker
Barnes
Young
Brick
Bush
Albert
PUBLIC
Disposition

Filter:

Add | Remove

OK | Cancel | Apply

Fig. 19. Resource Properties – Accessor list for a specific operation

Note that the “DEFAULT” accessor list in the operations drop down menu will apply to every operation that is not explicitly defined.

Public Access

PUBLIC Role

To grant public access to a resource, you can add the role “PUBLIC” to the accessor list. This is usually useful for bootstrapping like access to a naming service, etc.

Note that adding the role “PUBLIC” to the “DEFAULT” accessor list implies that all operations which are not defined in the Policy Editor will be granted public access by the SI&C! It is not recommended to do this! Instead you should leave the “DEFAULT” accessor list empty, thus denying access for operations that are not explicitly mentioned.

APPENDIX

A *SSLCurrent*

This appendix gives a brief overview on the SSLClientCurrent and SSLServer-Current objects, that offer an API to set and retrieve SSL keys and certificates.

A.1 SSLClientCurrent

The `SSLClientCurrent` allows to set a keystore in PKCS12 format that contains the keys and certificates to be used for opening SSL connections. Additionally, it allows to retrieve the keys and certificates that are in use.

```
package com.prismtech.corba.security;
public interface SSLClientCurrent
    extends org.omg.CORBA.Current{
    /**
     * Get the local SSL credentials that are used when
     * authenticating to a server. No restrictions on calling
     * context.
     * @return the principal, if present, <code>null</code>
     * otherwise.
     */
    java.security.Principal getLocalPrincipal();

    /**
     * Set the PKCS12 keystore that contains the key, certificate
     * and trusted certificates to be used by the SSL layer when
     * connecting to servers.
     * This must be called in (via an ORBInitializer) or right
     * after ORB initialization (before the first connection is
     * opened), and may be called at any time again later on (for
     * runtime reconfiguration).
     *
     * @param keystore a keystore in PKCS12 format
     * @param store_pass the password to decrypt the keystore
     * @param key_pass the password to decrypt the key
     */
    void setPKCS12Keystore(byte[] keystore,
                           char[] store_pass,
                           char[] key_pass)
        throws java.io.IOException,
        java.security.NoSuchAlgorithmException,
        java.security.KeyStoreException,
        java.security.cert.CertificateException,
        java.security.UnrecoverableKeyException;

    /**
     * Set the SecureRandom object to be used by the JSSE SSL
     * Layer. If not set, the default JDK implementation will
     * be used.
     * NOTE: This must be called before setPKCS12Keystore()
     * to take effect.
     *
     * @throws org.omg.CORBA.BAD_INV_ORDER If called after
     * the JSSE SSL layer has been configured
     */
    void setSecureRandom(java.security.SecureRandom rnd)
        throws org.omg.CORBA.BAD_INV_ORDER;

    /**
     * Set the TrustManager[] object to be used by the JSSE SSL
     * Layer. If not set, the default JDK implementation will be
     * used. <br>
     * NOTE: This must be called before setPKCS12Keystore() to
     * take effect.
     *
     * @throws org.omg.CORBA.BAD_INV_ORDER If called after the
     * JSSE SSL layer has been configured
     */
    void setTrustManagers(com.sun.net.ssl.X509TrustManager[]
                          trustManagers)
        throws org.omg.CORBA.BAD_INV_ORDER;
}
```

A.2 SSLServerCurrent

The SSLServerCurrent offers the same functionality as the SSLClientCurrent, with the addition of a method to retrieve the certificate chain the client used to authenticate with.

```
package com.prismtech.corba.security;

public interface SSLServerCurrent
    extends org.omg.CORBA.Current{
    /**
     * Get the local SSL credentials that are used when
     * authenticating to a client. No restrictions on calling
     * context.
     * @return the principal, if present, <code>null</code>
     * otherwise.
     */
    java.security.Principal getLocalPrincipal();

    /**
     * Get the client SSL credentials that it used when
     * authenticating to this server. This may only be called
     * during a remote invocation on the thread that the ORB used
     * to invoke the servant code.
     * @return the principal, if present, <code>null</code>
     * otherwise.
     */
    java.security.Principal getClientPrincipal();

    /**
     * Set the PKCS12 keystore that contains the key, certificate
     * and trusted certificates to be used by the SSL layer when
     * accepting connections from clients. <br>
     * This must be called in (via an ORBInitializer) or right
     * after ORB initialization (before the first connection is
    * accepted), and may be called at any time again later on (for
     * runtime reconfiguration).
     *
     * @param keystore a keystore in PKCS12 format
     * @param store_pass the password to decrypt the keystore
     * @param key_pass the password to decrypt the key
     */
    void setPKCS12Keystore(byte[] keystore,
                           char[] store_pass,
                           char[] key_pass)
        throws java.io.IOException,
        java.security.NoSuchAlgorithmException,
        java.security.KeyStoreException,
        java.security.cert.CertificateException,
        java.security.UnrecoverableKeyException;

    /**
     * Set the SecureRandom object to be used by the JSSE SSL
     * Layer. If not set, the default JDK implementation will
     * be used. NOTE: This must be called before
     * setPKCS12Keystore() to take effect.
     *
     * @throws org.omg.CORBA.BAD_INV_ORDER If called after
     * the JSSE SSL layer has been configured
     */
    void setSecureRandom(java.security.SecureRandom rnd)
        throws org.omg.CORBA.BAD_INV_ORDER;

    /**
     * Set the TrustManager[] object to be used by the JSSE SSL
     * Layer. If not set, the default JDK implementation will be
     * used. NOTE: This must be called before setPKCS12Keystore()
     * to take effect.
     *
     * @throws org.omg.CORBA.BAD_INV_ORDER If called after the
     * JSSE SSL layer has been configured
     */
    void setTrustManagers(com.sun.net.ssl.X509TrustManager[]
                          trustManagers)
        throws org.omg.CORBA.BAD_INV_ORDER;
}
```

A.3 X509CertificateChainPrinicpal

This extension of java's Principal interface allows access to a certificate chain.

```
package com.prismtech.corba.security;
public interface X509CertificateChainPrincipal
    extends java.security.Principal{
    /**
     * Get the chain of certificates contained in this
     * principal.
     */
    java.security.cert.X509Certificate[] getCertificateChain();
}
```

APPENDIX B *CSIv2ClientCurrent*

The `CSIv2ClientCurrent` *is a* `Current` *object analogous to the* `SSLClientCurrent`*, that allows to set username, password and an identity to assert for use by the CSIv2 layer.*

```
package com.prismtech.corba.security.csiv2;
public interface CSIv2ClientCurrent
    extends org.omg.CORBA.Current{
    /**
     * Set username and password to be sent in a CSIv2 GSSUP
     * AuthenticationToken.
     */
    public void setUsernamePassword(
        String username, String password);

    /**
     * Reset username and password so no authentication token
     * is sent
     */
    public void removeUsernamePassword();

    /**
     * Set the identity to be sent in a CSIv2 IdentityToken
     */
    public void setAssertedIdentity(String identity);

    /**
     * Set the identity to be sent in a CSIv2 Identity Token
     * to be the anonymous identity.
     */
    public void setAssertedIdentityAnonymous();
    /**
     * Reset identity so no IdentityToken is sent.
     */
    public void removeAssertedIdentity();
}
```