
PC-DCE™

# Guide to CAS

**Software Version 5.0**

# Notices

*PC-DCE Guide to CAS* - Software Version 5.0 - Revised March 2003



Entegrity Solutions Corporation, 410 Amherst St., Suite 150, Nahua, NH 03063 USA

# Contents


## Notices 2

## Preface 5

Intended Audience 5
Documentation 5
    PC-DCE Documentation Set 5
    The Open Group Documentation 6
Obtaining Technical Support 6
Contacting Entegrity Solutions 7
Obtaining Additional Technical Information 7

## Chapter 1 Overview 9

## Chapter 2 Technical Overview 11

2.1 Components 11
2.2 Co-Authentication Process 12
    2.2.1 Solicit Phase 12
    2.2.2 Challenge/Response Phase 13
    2.2.3 Cleanup 14

## Chapter 3 Developing CAS Applications 15

3.1 Introduction to Developing CAS Applications 15
3.2 Header File Summary 15
3.3 API Summary 15
    3.3.1 Server CASAPI Functions 15
    3.3.2 Client CASAPI Functions 16
    3.3.3 Client CASCALLBACK Function Types 16
    3.3.4 CASAPI and CASCALLBACK 16

## Chapter 4 Deploying CAS Applications 17

4.1 Full PC-DCE Client Requirement 17
4.2 ERAs Supporting CAS 17
    4.2.1 CASAUTHSVCS ERA 17
        4.2.1.1 Creating the CASAUTHSVCS ERA Schema 17
    4.2.2 pre_auth_req ERA 18
4.3 Specifying Authentication Methods Using ERAs 18
4.4 Installing Co-Authentication DLLs 18
4.5 Setting up the cass_handlers File 19
4.6 Using the Windows Event Log to Verify CAS Operation 19

Chapter 5 CAS Samples 21

5.1 Samples Overview 21
5.2 Setting Up the Samples 21
5.3 Running the Samples 22

Chapter 6 SecurID Co-Authentication 23

6.1 Security Dynamics SecurID Co-Authentication Overview 23
6.2 Setting Up SecurID Co-Authentication 23
6.3 Logging into DCE Using SecurID Co-Authentication 24
6.4 Logging In Using SecurID Tokens 24

Index 25

# Preface

## Intended Audience

This guide is for programmers who are developing distributed applications for the Entegrity® DCE implementation for Windows® called PC-DCE. This guide assumes a general knowledge of DCE (Distributed Computing Environment).

## Documentation

This section describes the documentation that Entegrity provides with PC-DCE on both the product CD and on the Entegrity web site (**www.entegrity.com**) under the Support link:

- *PC-DCE Documentation Set*
- *The Open Group Documentation*

Documentation on other Entegrity products, such as NetCrusader/Web, is also available on the Entegrity web site.

We are always trying to improve our documentation. If you notice any inaccuracies or cannot find information, please send email to **docs@entegrity.com**. We welcome any comments or suggestions.

## PC-DCE Documentation Set

The following documents are provided with PC-DCE:

- *PC-DCE Installation and Release Notes*
- *PC-DCE Overview Guide*
- *PC-DCE Administrator's Guide*
- *PC-DCE Developer's Notes*
- *PC-DCE Guide to CAS* (this book)

PC-DCE also provides online help with the following programs:

- PC-DCE Service Panel
- PC-DCE Configuration Panel
- DCE Director
- Visual DCE ACL Editor
- DCEsetup

Entegrity also provides OSF DCE Version 1.2.2 documentation on the product CD and the Entegrity Support web site.

## The Open Group Documentation

The PC-DCE product CD and the Entegrity web site also provide The Open Group (formerly OSF) DCE Version 1.2.2 documentation, including the following guides:

- *OSF DCE Administration Guide — Core Components*
- *OSF DCE Administration Guide — Introduction*
- *OSF DCE Application Development Guide — Core Components*
- *OSF DCE Application Development Guide — Directory Services*
- *OSF DCE Application Development Guide — Introduction and Style Guide*
- *OSF DCE Application Development Reference*
- *OSF DCE Command Reference*
- *Introduction to OSF DCE*
- *OSF DCE Problem Determination Guide*
- *OSF DCE/File-Access Administration Guide and Reference*
- *OSF DCE/File-Access Users' Guide*
- *OSF DFS Administration Guide and Reference*
- *OSF GDS Administration Guide and Reference*

# Obtaining Technical Support

If you purchased PC-DCE directly from Entegrity Solutions or Gradient Technologies you are entitled to 30 days of limited technical support beginning on the day the product is expected to arrive.

You may also purchase a support plan that entitles you to additional services. You *must* register prior to receiving this support. For details, refer to the customer support information package that accompanied your shipment or refer to **http://support.entegrity.com**. The web site also contains online forms for easy registration.

If you purchased PC-DCE from a reseller, please contact the reseller for information on obtaining technical support.

# Contacting Entegrity Solutions

| Contact | Address | Phone/Fax/Email |
|---|---|---|
| **DCE Product and Sales Information** | Entegrity Solutions Corporation<br>410 Amherst Street, Suite 150<br>Nashua, NH 03063 USA | Email: DCESales@entegrity.com<br>Web: www.entegrity.com<br><br>Tel: +1-603-882-1306 ext.2700<br>Toll Free (US): 1-800-525-4343 ext. 2700<br>Fax: +1-603-882-6092 |
| **All Other Product and Sales Information Requests** | Entegrity Solutions Corporation<br>2001 Gateway Place, Suite 420W<br>San Jose, CA 95110 USA | Email: info@entegrity.com<br>Web: www.entegrity.com<br><br>Tel: +1-408-487-8600 ext. 123<br>Fax: +1-408-487-8610 |
| **Technical Support** | Entegrity Solutions Corporation<br>410 Amherst Street, Suite 150<br>Nashua, NH 03063 USA | Email: support@entegrity.com<br>Web: support.entegrity.com<br><br>Tel: +1-603-882-1306 ext. 2702<br>Toll Free (US): 1-888-368-3555 ext. 2702<br>Fax: +1-603-882-6092 |
| **Documentation Comments and Suggestions** | | Email: docs@entegrity.com |
| **Other Inquiries** | Entegrity Solutions Corporation<br>2001 Gateway Place, Suite 420W<br>San Jose, CA 95110 USA | Email: info@entegrity.com<br>Web: www.entegrity.com<br><br>Tel: +1-408-487-8600<br>Fax: +1-408-487-8610 |

For a complete listing of Entegrity Solutions Corporation sales, research and development, and solutions centers worldwide, please see the Entegrity web site at http://www.entegrity.com.

# Obtaining Additional Technical Information

| Contact | Address | Phone/Fax/Email |
|---|---|---|
| **The Open Group™**<br>Developer of DCE (Distributed Computing Architecture) software and standards. | The Open Group<br>29B Montvale Ave<br>Woburn<br>MA 01801<br>U. S. A. | Tel: +1 781-376-8200<br>Fax: +1 781-376-935811<br>**http://www.opengroup.org** |

CHAPTER 1

# Overview


The PC-DCE™ Co-Authentication Service (CAS) provides developers with the ability to plug alternative authentication methods into PC-DCE. A user logging in through CAS uses an alternative authentication method, for example a biometric device such as a fingerprint scanner, to obtain DCE login credentials.

To use a specific authentication method, you must install both a co-authentication DLL and a login program that implement the authentication method functionality. Currently, PC-DCE includes a co-authentication DLL and a login program for Security Dynamics SecurID® authentication. If you wish to use a different authentication method, you can develop your own DLL and login program. This document provides information you will need to write these programs.

Entegrity® Solutions Professional Services can assist customers who wish to use another authentication method but do not want to develop the co-authentication DLL and login program.

CHAPTER 2

# Technical Overview

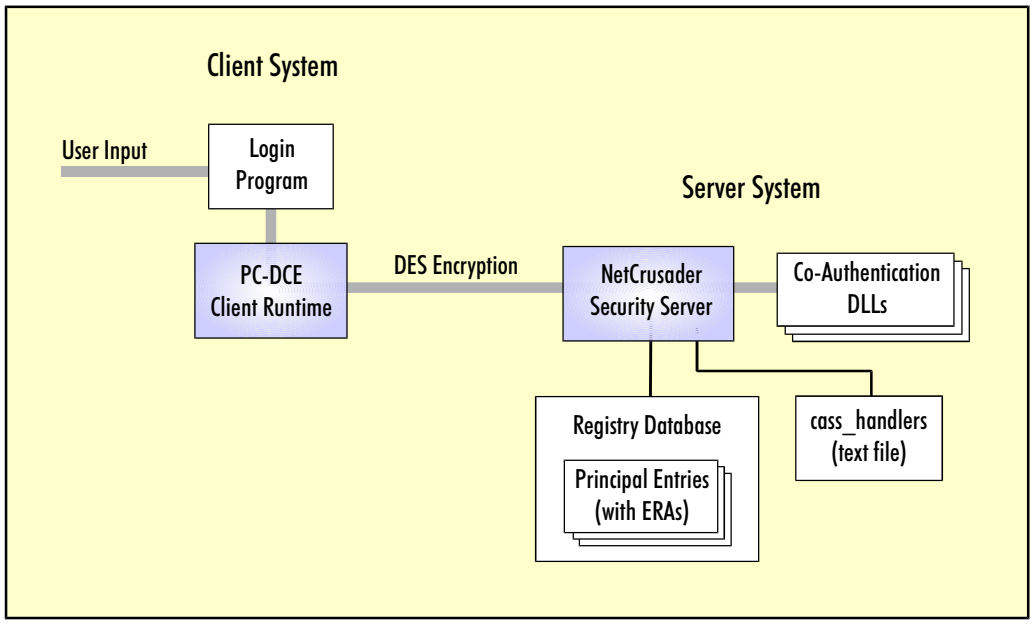

This chapter provides a technical overview of CAS, and contains the following sections:

2.1 Components
2.2 Co-Authentication Process

## 2.1 Components

*Figure 2-1* shows the general client and server components required by CAS:

Figure 2-1: Client and Server Components



The client system requires:

- **PC-DCE Client** — This is the full client (daemons configured).
- **Login Program** — Provided by the developer. The login program implements certain functions required by CAS and provides client support for each desired authentication method.

The server system requires:

- **Gradient Security Server** — The Gradient Security Server provides a generalized CAS interface that supports user-developed authentication methods implemented as Co-authentication DLLs (see the following item). The Gradient Security Server is the PC-DCE Security Server with NetCrusader extensions installed.
- **Co-Authentication DLLs** — The developer implements each desired authentication method as a DLL written to the CAS interface.
- **Registry Database** — The registry database provided by the PC-DCE security server includes entries for all principals (users) to be authenticated through CAS. Each principal entry that will use CAS requires an extended registry attribute (ERA) that lists the authentication methods allowed for this principal.
- **cass_handlers** — Text file that the security server reads to identify the authentication methods supported on this server and the associated DLLs.

## 2.2 Co-Authentication Process

For each login session, CAS uses an initial solicit phase to determine the authentication method to use. The solicit phase is followed by a challenge/response phase during which the user is prompted to supply authentication data, and the appropriate co-authentication DLL verifies the data.

### 2.2.1 Solicit Phase

To prepare for the solicit phase:

- The client login program obtains the user's principal name and builds a client-supported method list. To build the list, the login program calls the CASAPI function **casc_prompt_register()** for each supported authentication method. This function identifies the authentication method (by UUID) and registers required callback functions that the PC-DCE client runtime calls during the challenge/response phase.
- The Gradient Security Server identifies all installed co-authentication DLLs and builds a list of server-supported authentication methods. It identifies the installed co-authentication DLLs by reading the **cass_handlers** file, which contains a list of DLL names. The Gradient Security Server builds the server-supported method list by calling the

**cass_get_type()** function exported by each co-authentication DLL. This call returns a UUID that identifies the authentication method that the DLL supports.

To begin the solicit phase:

- The client login program calls **sec_login_validate_identity()**. This starts CAS within the PC-DCE client runtime. The PC-DCE client sends the Gradient Security Server a solicit message containing the principal's user name and the client-supported method list.
- The Gradient Security Server determines which method it should use by examing the principal's CASAUTHSVCS ERA, which is a prioritized list of co-authentication DLL names. It chooses the first method (DLL name) in the ERA list that is supported by both the server and the client.

## 2.2.2 Challenge/Response Phase

In the challenge/response phase, the Gradient Security Server calls the following CASAPI functions exported by the selected co-authentication DLL:

- **cass_initialize()** allows the DLL to load whatever it needs (such as another DLL) and set up any required environment parameters.
- **cass_response()** returns the initial challenge data. The Gradient Security Server can then send the client a reply to the initial solicit request. This reply identifies the selected method and includes the initial challenge data for that method. This begins the challenge/response phase. Note that it is possible for **cass_response()** to return a value of login complete (see below) which would automatically log in the user without verification.
- **cass_response_free()** allows the DLL to clean up the challenge data (free memory) once the security server is finished with it.

The PC-DCE client runtime calls the callback function (type **casc_routine_t** defined in **casc_proto.h**) that handles challenges and responses for the selected authentication method. The PC-DCE client runtime passes the challenge data to this function and collects the user response as an output. The PC-DCE client runtime sends the user's response back to the server and calls the callback function (type **casc_routine_free_t** defined in **casc_proto.h**) to clean up the response data.

The Gradient Security Server calls **cass_response()**, this time supplying the response data from the user. This returns one of three values:

- **challenge** — The DLL wants to send another challenge to the user and has provided the challenge data. The server sends the challenge to the client. When it receives the response, it calls **cass_response()** again. The challenge/response cycle continues either until login completes or login fails. After each cycle the server calls **cass_response_free()**.
- **login complete** — The DLL successfully authenticated the user. In the client login program, **sec_login_validate_identity()** returns TRUE.

- **login fail** — The authentication failed. In the client login program, **sec_login_validate_identity()** returns FALSE.

### 2.2.3 Cleanup

When the challenge/response phase is complete, the PC-DCE client calls the callback function (type **casc_routine_session_free_t** defined in **casc_proto.h**) to clean up any session data.

When the Gradient Security Server is shutting down, it calls the **cass_terminate()** function, which allows the DLL to perform any final cleanup.

CHAPTER 3

# Developing CAS Applications


This chapter provides information on developing CAS applications, and contains the following sections:

3.1 Introduction to Developing CAS Applications
3.2 Header File Summary
3.3 API Summary

## 3.1 Introduction to Developing CAS Applications

Developers must develop a client login program and a co-authentication DLL. If you haven't already, refer to *Chapter 2 on page 11*. Then refer to the header files for API definitions and to the samples for working prototypes. Refer to the rest of this chapter for additional information:

## 3.2 Header File Summary

The following header files are installed in *install_directory***include\dce** when you install the PC-DCE Application Developer's Kit:

- **cass_proto.h** — header file for co-authentication DLL functions
- **casc_proto.h** — header file for client functions
- **cas_types.h** — general cas types
- **cas.h** — more required definitions. Includes **cas_types.h**

## 3.3 API Summary

This section describes API functions.

### 3.3.1 Server CASAPI Functions

The following functions are defined in **cass_proto.h**:

- **cass_initialize()**
- **cass_get_type()**
- **cass_response_handler()**
- **cass_terminate()**

### 3.3.2 Client CASAPI Functions

The following function is defined in **casc_proto.h**:

- **casc_prompt_register()**

### 3.3.3 Client CASCALLBACK Function Types

The following function types are defined in **casc_proto.h**:

- **casc_routine_t**
- **casc_routine_free_t**
- **casc_routine_session_free_t**

### 3.3.4 CASAPI and CASCALLBACK

Notice that in **cas_types.h**, CASAPI is defined as __stdcall and CASCALLBACK is defined as __cdecl.

CHAPTER 4

# Deploying CAS Applications


This chapter provides information on deploying CAS applications, and contains the following sections:

4.1 Full PC-DCE Client Requirement
4.2 ERAs Supporting CAS
4.3 Specifying Authentication Methods Using ERAs
4.4 Installing Co-Authentication DLLs
4.5 Setting up the cass_handlers File
4.6 Using the Windows Event Log to Verify CAS Operation

## 4.1 Full PC-DCE Client Requirement

For this release of PC-DCE, the CAS client functions require the full PC-DCE client. You configure the full PC-DCE client by selecting the Configure Client Daemons setting in the PC-DCE configuration panel.

## 4.2 ERAs Supporting CAS

The ERAs that support CAS are CASAUTHSVCS and pre_auth_req.

### 4.2.1 CASAUTHSVCS ERA

CASAUTHSVCS is a multipart ERA that allows you to specify authentication methods on a per-principal basis. Refer to *Chapter 2 on page 11* for a discussion of how the security server uses this ERA to select the authentication method to be used for a principal.

#### 4.2.1.1 Creating the CASAUTHSVCS ERA Schema

The CASAUTHSVCS ERA schema is normally created by the PC-DCE configuration program when you configure PC-DCE on your system. If you do not wish to reconfigure PC-DCE (for example, if you are installing PC-DCE as an upgrade), you can create the schema using **dcecp** as follows:

```
dcecp> xattrschema create  /.:/sec/xattrschema/CASAUTHSVCS
-encoding stringarray
-aclmgr {principal {query r}
                   {update m}
                   {test r}
                   {delete m}} -multivalued yes
```

### 4.2.2 pre_auth_req ERA

You can use the pre_auth_req ERA (with a value of 11) to specify that the principal must log in using a CAS method; no fallback to a straight DCE name/password login is allowed.

## 4.3 Specifying Authentication Methods Using ERAs

To associate a principal with one or more authentication methods, use **dcecp** to attach the CASAUTHSVCS ERA to the principal. This ERA is a prioritized list of DLLs: if more than one method is supported by both the security server and the client, the security server selects the first that appears in the ERA.

In addition, you can use the pre_auth_req ERA to specify that the principal must log in using a CAS method; no fallback to a straight DCE name/password login is allowed.

The following example adds two methods (DLLs) to principal groucho:

```
dcecp> principal modify groucho -add {CASAUTHSVCS
{securidcass.dll}{examplecass.dll}}
```

In this list, SecurID authentication is the highest priority.

The following example adds the pre_auth_req ERA:

```
dcecp> principal modify groucho -add {pre_auth_req 11}
```

To verify that the ERA was added correctly use the **dcecp principal show** command :

```
dcecp> principal show groucho -all|{fullname {}}|{uid 105}|{uuid
00000069-e706-259c-b400-00802964ff95}
{alias no}
{quota unlimited}
{groups none}
{CASAUTHSVCS
 securidcass.dll
 examplecass.dll}
 {pre_auth_req 11}
```

## 4.4 Installing Co-Authentication DLLs

Install co-authentication DLLs in the *install_directory***bin** directory (for example: **pcdce32\bin**).

## 4.5 Setting up the cass_handlers File

The **cass_handlers** file is a text file that the Gradient Security Server reads to identify installed co-authentication DLLs. The full path is *install_directory***opt\dcelocal\var\security\cass\cass_handlers**.

The file is a list of DLL names, each terminated by a carriage return. Comments begin with a semicolon and blank lines are ignored. The following is an example **cass_handlers** file:

```
; Security Dynamics DLL
securidcass.dll
; Example Co-authentication DLL
examplecass.dll
```

## 4.6 Using the Windows Event Log to Verify CAS Operation

To verify that a particular CAS authentication service has been loaded by the Security Server, use the Windows Event Viewer. Each successfully-loaded authentication service adds a trace to the log with a Source label of *Entegrity DCE*. When you click the trace entry, an event detail dialog box appears with information similar to the following:

```
[CAS] securidcass.dll auth. service loaded.
uuid:00389c30-4341-13de-91b9-00802969679f
```

Each successfully-loaded authentication service adds an entry.

---

NOTE: The Gradient Security Server loads authentication services after the first login. Therefore a user may need to perform an initial login, CAS or normal, before the CAS traces appear in the log.

---

CHAPTER 5

# CAS Samples


This chapter provides information on deploying CAS applications and contains the following sections:

5.1 Samples Overview
5.2 Setting Up the Samples
5.3 Running the Samples

## 5.1 Samples Overview

The PC-DCE ADK includes compiled sample programs in *install_directory***samples\cas**:

- **examplecass.dll** — Sample co-authentication DLL
- **login_pgm.exe** — Sample login program that accesses **examplecass.dll**

The Application Developer's Kit (ADK) also includes the sample source files and a makefile in **samples\cas**, as well as header files in *install_directory***include**.

## 5.2 Setting Up the Samples

Before you run the samples:

1. Move **examplecass.dll** to *install_directory***bin**.
2. Add the following line to the cass_handlers file:

   ```
   examplecass.dll
   ```

   For more information, see *Section 4.5 on page 19*.
3. Add the CASAUTHSVCS ERA to the principal you will use to log into DCE through the sample. The ERA should specify **examplecass.dll**. For more information, see *Section 4.3 on page 18*.
4. Restart PC-DCE.

## 5.3 Running the Samples

To run the samples:

1. Make sure PC-DCE is running.
2. Start the sample login program either from the PC-DCE program group or by running **login_pgm.exe** from the command line.
3. The program prompts you for your username followed by three challenge prompts. Enter the name of the principal you set up with the CASAUTHSVCS ERA. When the program challenges you to "enter one", enter the word "one", and so on.

   After you respond to the third challenge ("enter three"), the login program logs you into DCE and exits. You can run **klist** from the command line to view your credentials.

CHAPTER 6

# SecurID Co-Authentication


This chapter provides information on deploying CAS applications, and contains the following sections:

6.1 Security Dynamics SecurID Co-Authentication Overview
6.2 Setting Up SecurID Co-Authentication
6.3 Logging into DCE Using SecurID Co-Authentication
6.4 Logging In Using SecurID Tokens

## 6.1 Security Dynamics SecurID Co-Authentication Overview

PC-DCE includes a co-authentication DLL that enables DCE login through Security Dynamics® SecurID Token and ACE/Server technology. If a principal's CASAUTHSVCS ERA specifies SecurID® co-authentication, the PC-DCE **dce_login** program automatically prompts for the SecurID passcode.

## 6.2 Setting Up SecurID Co-Authentication

Before principals can log into DCE using SecurID co-authentication:

1. Install the Security Dynamics ACE/Server® on a Windows or UNIX machine in your network.
2. Install the Security Dynamics ACE/Client® on the Gradient Security Server machine.
3. Log into the DCE cell as administrator:

   **dce_login cell_admin** *cell_password*
4. Use the Security Dynamics administration tool available in the ACE/Server admin program to create a Security Dynamics user with a token. You must also use this tool to enable the Gradient Security Server host to perform ACE/Client logins on behalf of Security Dynamics users.
5. Create a DCE principal account with the same name as the new Security Dynamics user.
6. Add the CASAUTHSVCS ERA to the principal you will use to log into DCE through the sample. The ERA should specify **securidcass.dll**. For more information, see *Section 4.3 on page 18*.
7. If the user will use PC-DCE integrated login, use the ACE/Client control panel to turn off the Security Dynamics SDGINA Windows login feature.

## 6.3 Logging into DCE Using SecurID Co-Authentication

To log into DCE using SecurID co-authentication:

1. Make sure PC-DCE and the ACE/Server are running.
2. *If you have integrated login configured*, log into the operating system and enter your user name. A dialog box prompts you for your passcode.

   *If you do not have integrated login configured*, run **dce_login** from the command line and enter your user name. **dce_login** prompts you for your passcode.
3. Enter your passcode exactly as you would for normal SecurID authentication. If you need more information on logging in using SecurID tokens, see *Section 6.4*. After you enter your correct PIN number and passcode, the Gradient Security Server logs you into DCE.
4. You can verify the login by running **klist** from the command line.

## 6.4 Logging In Using SecurID Tokens

Logging into a protected network using a SecurID token will vary depending upon your token type, as described below:

- For Standard SecurID or Key Fob tokens, log in by entering both your Personal Identification Number (PIN) and the passcode currently displayed by your token:

  **dce_login** *username PIN passcode*

- For a SecurID PINPAD token, you must first enter your PIN into the token, then log in by entering the token-generated passcode:

  **dce_login** *username passcode*

If this is your first login using your SecurID token, the screen may display an assigned PIN immediately after you enter the passcode. Record this PIN for use in subsequent logins or, if your system allows changes, enter a new PIN.

If you enter your passcode incorrectly three consecutive times followed by a correct entry, you will be prompted for a new SecurID code.

# Index


## A

ACE/Server 23
Authentication method 12, 18

## C

Callback functions 12
cas.h 15
cas_types.h 15, 16
CASAPI 13, 15 to 16
CASAUTHSVCS, *see* ERA
casc_get_type() 13
casc_prompt_register() 12, 16
casc_proto.h 13, 14, 15, 16
casc_routine_free_t 13, 16
casc_routine_session_free_t 14, 16
casc_routine_t 13, 16
cass_get_type() 15
cass_handlers 12, 19
cass_initialize() 13, 15
cass_proto.h 15
cass_response() 13
cass_response_free( 13
cass_response_handler() 15
cass_terminate() 14, 15
__cdecl 16
Challenge/Response phase 13
    see also Co-authentication process 13
client, full 17
Co-authentication
    DLLs 12, ?? to 14, 18
    process 12 to 14
Co-authentication DLL
    SecurID 23
Components 11
Contacting Entegrity Solutions 7

## D

DCE
    The Open Group documentation 6
DLLs, see Co-authentication DLLs 12
Documentation 5, 6, 7

## E

ERA 12
    CASAUTHSVCS 13, 17 to 18
    pre_auth_req 17 to 18
Events 19
examplecass.dll 21
Extended registry attribute, *see* ERA

## F

full client 17

## G

Gradient Security Server 19

## H

Header files 15

## L

Log 19
Login program 12, 21
login_pgm.exe 21

## M

makefile 21

## O

OSF documentation 6
Overview 9

## P

PC-DCE 17
    additional documentation 5
pre_auth_req, *see* ERA
Process, *see* Co-authentication process

# R

Registry Database 12

# S

Samples 21, 22
SDGINA Windows login 23
sec_login_validate_identity() 13
SecurID authentication 23, 24
securidcass.dll 23
Security Server 12
Solicit phase 12
    see also Co-authentication process 12
__stdcall 16
Support 6, 7

# T

Technical support 6, 7

# W

Windows NT Event Log 19