Adaptive Server® Anywhere Authenticated Edition Read Me First

Part number: 03988-01-0900-01

Last modified: June 2003

Contents

CHAPTER 1

OEM Edition Read Me First

About this booklet

This booklet describes how to use the OEM Edition of Adaptive Server Anywhere.

Contents

Authenticated Adaptive Server Anywhere applications

The OEM Edition of Adaptive Server Anywhere is provided for Sybase Commercial Application Partners. With the OEM Edition, applications that are not **authenticated** are limited in the functions they can carry out.

Database applications carry out a well-defined set of operations against a database. With the OEM Edition of Adaptive Server Anywhere, applications other than those you supply are restricted in the operations they can perform. These limitations are in addition to any imposed through the SQL permission scheme. An authenticated application can carry out any operation on the database, subject to the permissions granted to the user ID.

Note that unauthenticated connections can perform inserts, updates, and deletes on temporary tables. This allows complex reports to be created using stored procedures, and accessed using reporting tools, such as Crystal Reports.

The authentication mechanism is independent of application programming language or tool, and is carried out on every connection. This means you can use both authenticated connections and more restricted unauthenticated connections in your application.

Authentication is not a security device
Authentication is not a security mechanism. For example, anyone running an unauthenticated database server against the database can carry out any operation, subject to the usual SQL permissions scheme.

What you need for authentication

Developing an authenticated application is a simple process. A special **authentication signature** is incorporated into the database. A second signature is incorporated into your application. When the application connects to the database, the signatures are compared in order to authenticate the application.

Working with the OEM Edition requires the following steps:

1. **Use the authenticated database server** The authenticated database server is the version included in this package.
2. **Obtain authentication signatures** The authentication mechanism relies on signatures obtained from Sybase.
3. **Create an authenticated database** Authenticated databases are

protected from unauthenticated applications.

4. **Create an authenticated application** You must add instructions to your application that authenticate it whenever it connects to the database.

☞ For a detailed description of each of these steps, see "Developing an authenticated application" on page 4.

Database tools are self-authenticating

All the database tools included with SQL Anywhere Studio, including Sybase Central, Interactive SQL, and the command-line utilities, are self-authenticating.

This means that they are unrestricted in their operations against any authenticated database. If the database itself is not authenticated, the tools act in a restricted, read-only fashion.

Self-authentication enables you to use *dbbackup*, for example, to back up an authenticated database running on an authenticated database server.

Upgrading authenticated databases

If you apply the Upgrade utility to an authenticated database, the authentication information is lost. If you store the authentication statement in the file *saopts.sql*, as described in "Step 3: Authenticate your database" on page 4, the authentication information is preserved on upgrading.

If you want access to all the new features of SQL Anywhere Studio 9, [including those such as new indexing features, procedure profiling, and Java 2 suppo you must unload and reload your database. Again, adding the authentication statement to *saopts.sql* ensures that the authentication information is maintained.

Storing the authentication string in the saopts.sql file

The *saopts.sql* file is found in the following location: *C:\Program Files\Sybase\SQL Anywhere 9\scripts.* In the *saopts.sql* file, the section you edit is:

```
if not exists( SELECT * FROM SYS.SYSOPTION
   WHERE ucase( "option" ) = ucase( 'Database_authentication' ) ) THEN
      SET OPTION PUBLIC.Database_authentication = '';
```

Once you add the authentication statement, your *saopts.sql* file should look like the following:

```
if not exists( SELECT * FROM SYS.SYSOPTION
   WHERE ucase( "option" ) = ucase( 'Database_authentication' ) ) THEN
      SET OPTION PUBLIC.Database_authentication = 'your authentication statement
                    here';
```

Developing an authenticated application

This section describes the steps involved in creating an authenticated application.

Step 1: Use the authenticated database server

The edition of the database server provided in this package is the OEM Edition. This edition differs from the usual database server only in that it processes authentication instructions. The authentication instructions are ignored by other editions of the database server. If you do not use the authenticated database server, no restrictions are placed on unauthenticated applications.

Step 2: Obtain authentication signatures

❖ **To obtain your authentication signatures**

1. Open the following URL in your web browser:

   http://www.sybase.com/products/anywhere/authentication

2. Click Go To The Online Form Now.

3. Complete the form to obtain your authentication signatures. The following information is incorporated into your authentication mechanism:

   - **Company name** The name of your company.
   - **Application name** The name of your application.

     ☞ For information about how the company name and application name are incorporated into the authentication mechanism, see "Step 3: Authenticate your database" on page 4.

   When you complete the form, you will be provided (by e-mail) with a database signature and an application signature within 48 hours. These signatures are long (81 character) strings of characters and digits.

   The e-mail message containing your authentication information includes some examples of how to use the information. Some e-mail systems force line breaks in these instructions. You should be sure to rejoin lines broken in the e-mail message if the instructions are to work.

Step 3: Authenticate your database

The OEM Edition of Adaptive Server Anywhere does not permit any operations to be carried out on an unauthenticated database.

❖ **To authenticate a database**

1. Set the DATABASE_AUTHENTICATION option for the database, using the following SQL authentication statement:

   ```
   SET OPTION PUBLIC.DATABASE_AUTHENTICATION
     = 'company = company_name;
        application = application_name;
        signature = database_signature'
   ```

   The *company_name* and *application_name* arguments are the values you supplied to Sybase when obtaining your signature, and *database_signature* is the database signature that you received from Sybase.

2. Restart the database for the option to take effect. When the database server loads an authenticated database, it displays a message in the database server window describing the authenticated company and application. You can check that this message is present to verify that the DATABASE_AUTHENTICATION option has taken effect. The message has the following form:

   `This database is licensed for use with: Application: application_name Company: company_name`

You can store the authentication statement in a SQL script file to avoid having to type in the long signature. You can run the SQL script from Interactive SQL by choosing Run Script from the File menu.

If you store the authentication statement in the file *saopts.sql*, in the *scripts* subdirectory of your Adaptive Server Anywhere directory, it is applied whenever you create a database or upgrade a database.

☞ For instructions on storing the authentication statement in the *saopts.sql* file, see "Storing the authentication string in the saopts.sql file" on page 3.

Step 4: Authenticate your application

An authenticated application must set the CONNECTION_AUTHENTICATION database option immediately after connecting. The option must be set on every connection immediately after the connection is established.

ODBC or JDBC applications query the database about its capabilities, and the developer may not have control over these actions. For this reason, every connection has a thirty second grace period before the restrictions apply. The grace period allows an application to authenticate regardless of which development tool is being used.

The following SQL statement authenticates the connection:

```
SET TEMPORARY OPTION CONNECTION_AUTHENTICATION
  = 'company = company_name;
     application = application_name;
     signature = application_signature'
```

The option must be set for the duration of the connection only using the TEMPORARY keyword.

The company name and application name must match those in the database authentication statement. The signature is the application signature that you have obtained from Sybase.

The database server verifies the application signature against the database signature. If the signature is verified, the connection is authenticated and has no restrictions on its activities beyond those assigned to its user ID. If the signature is not verified, the connection is limited to those actions permitted by unauthenticated applications.

Executing the authentication statement

The way you execute the SET TEMPORARY OPTION statement that sets the authentication option depends on the programming interface you are using. This section describes how to set the option using the ODBC, PowerBuilder, JDBC, and Embedded SQL interfaces. The signatures listed here are not valid signatures.

- **ODBC** Use the following statement:

```
SQLExecDirect(hstmt,"SET TEMPORARY OPTION CONNECTION_AUTHENTICATION ='Company=MyCo;
               Application=MyApp;Signature=0fa55159999e14d818eb4fe3db41447146f1571g0
               c4429e41262995f8cdac5falkd3729c4a9afdaf'",SQL_NTS);
```

  The string must be entered on a single line or you must build it up by concatenation.

- **PowerBuilder** Use the following PowerScript statement:

```
EXECUTE IMMEDIATE "set temporary option ... "
USING SQLCA
```

- **JDBC** Use the following statement:

```
Statement Stmt1 = con.createStatement();
Stmt1.executeUpdate("SET TEMPORARY OPTION CONNECTION_AUTHENTICATION='Company=MyCo;A
               pplication=MyApp;Signature=0fa55159999e14d818eb4fe3db41447146f1571g0c
               4429e41262995f8cdac5falkd3729c4a9afdaf'");
```

  The string must be entered on a single line or you must build it up by concatenation.

- **Embedded SQL** Use the following statement:

```
EXEC SQL SET TEMPORARY OPTION CONNECTION_AUTHENTICATION ='Company=MyCo;Application=
              MyApp;Signature=0fa551599998e14d818eb4fe3db41447146f1571g0c4429e41262
              995f8cdac5falkd3729c4a9afdaf'
```

The string must be entered on a single line or you must build it up by concatenation.

Using special characters in the authentication option

If your company name has quotation marks, apostrophes, or other special characters (for example, Joe's Garage) you need to be careful about how you construct the authentication statement.

The entire set of authentication options (Company=...,Application=...,Signature=...) is a SQL string. The rules for strings in SQL dictate that if you include a quotation mark inside the string, it must be doubled in order to be accepted. For example:

```
SET TEMPORARY OPTION CONNECTION_AUTHENTICATION
  = 'company = Joe''s Garage;
  application = Joe''s Program;
  signature = 0fa55157edb8e14d818e...'
```