

Using Open Database Connectivity

Revised: March 20, 2009, OL-17222-03

Cisco Access Registrar (CAR) supports Open Database Connectivity (ODBC), an open specification that provides application developers a vendor-independent API with which to access data sources. CAR provides a new type of RemoteServer object and a new service to support ODBC. You can use CAR to authenticate and authorize access requests by querying user information through ODBC.

ODBC is an application program interface (API). Real data exchange between an application and data store is still carried out by SQL through ODBC. To achieve the most flexibility, you are required to define your own SQL using **aregcmd**. CAR will register the SQL statements and send them to the data store through ODBC when required. Because you can define your own SQL, CAR supports sites that have their own data stores.

ODBC is configured using **.ini** files, specifically **odbc.ini** and **odbcinst.ini**. However, you cannot create or modify these files directly. CAR creates the **.ini** files after you use **aregcmd** to configure the ODBC connection. The SQL is stored in the local database (MCD). During execution, the CAR server reads the local database, prepares the SQL statements, and sends the SQL to the data source.



Note

CAR uses its own ODBC driver manager and does not share existing ODBC drivers (if you already have ODBC installed). If you are already using ODBC, you will have to maintain two separate ODBC installations.

The ODBC memory requirement depends on your configuration. The more datasources you configure, the more memory is required. Packet processing time might increase if you configure a large number of SQL statements under SQLDefinition.

The CAR 4.2 package includes some ODBC Drivers, and you should use the included driver whenever possible. If a data store's ODBC driver is not included with CAR, you are required to install it. You configure the driver library using **aregcmd** to modify the associated **ini** file.

This chapter has the following sections:

- [Oracle Software Requirements](#)
- [Configuring ODBC, page 20-2](#)
- [MySQL Support, page 20-7](#)

Oracle Software Requirements

The CAR 4.2 ODBC feature requires that you have Oracle 9i and/or 10g client software installed. All Oracle client software library files are expected under **\$ORACLE_HOME/lib**.

When you install CAR 4.2 software, the installation process prompts you for ORACLE_HOME variable and sets it in the CAR start-up script, **/etc/init.d/arserver**. Two other environment variables (ODBCINI and ODBCYSINI) are also set in the **arserver** script. To change any of these variables, modify the **/etc/init.d/arserver** script and restart the CAR server.

The following changes have been made to support Oracle 9:

- The file **liboraodbc.so** has been renamed to **liboraodbc8.so**.
- The file **liboraodbc9.so** has been added.

Configuring ODBC

You use **aregcmd** to define your ODBC configuration and SQL statements. The CAR server automatically creates the **ODBC.ini** file for your driver manager and driver based on how you configure ODBC.

To use ODBC in CAR, you must do the following:

1. Configure an ODBC Service
2. Configure an ODBC RemoteServer object
3. Configure an ODBC DataSource
4. Set ODBC service as the default AA service
5. Save your configuration

After you **save** and validate your configuration, it is saved in the MCD database. If you have configured an ODBC service, CAR will query the MCD database and create or modify the **odbc.ini** file before it builds a connection to the database. When you reload your configuration, CAR shuts down any existing ODBC connections, then queries the MCD database to create or modify the **odbc.ini** file and build a new connection for any configured ODBC Data Sources.

Configuring an ODBC Service

You configure an ODBC service under **/Radius/Services**. When you define an ODBC service under **/Radius/Services**, you must set its type to ODBC and provide the following configuration options:



Note

We will use ODBC as the ODBC service name in the following examples.

```
[ //localhost/Radius/Services/ODBC ]
  Name = ODBC
  Description =
  Type = odbc
  IncomingScript~ =
  OutgoingScript~ =
  OutagePolicy~ = RejectAll
  OutageScript~ =
  MultipleServersPolicy = Failover
```

RemoteServers/

Table 20-1 describes the ODBC service parameters.

Table 20-1 ODBC Service Parameters

Parameter	Description
Name	Required; inherited from the upper directory
Description	An optional description of the service
Type	Must be set to ODBC for ODBC service
IncomingScript	Optional
OutgoingScript	Optional
OutagePolicy	Required; must be set to AcceptAll or Drop Packet, or defaults to RejectAll
OutageScript	Optional
MultipleServersPolicy	Required; must be set to RoundRobin or defaults to Failover. When set to Failover, CAR directs requests to the first server in the list until it determines the server is off-line. If so, CAR redirects all requests to the next server in the list until it finds an on-line server. When set to RoundRobin, CAR directs each request to the next server in the RemoteServers list in order to share the resource load across all servers in the RemoteServers list.
RemoteServers	Required list of remote servers defined under /Radius/Services/ODBC/RemoteServers such as ODBC-Primary and ODBC-Secondary

Configuring an ODBC RemoteServer

You must configure an ODBC RemoteServer object for each RemoteServer object you list under **/Radius/Services/ODBC/RemoteServers**. Use the **aregcmd** command **add** to add ODBC servers under **/Radius/RemoteServers**.

Table 20-2 describes the ODBC service parameters.

Table 20-2 ODBC Remote Server Parameters

Parameter	Description
Name	Required; inherited from the upper directory
Description	An optional description of the server
Protocol	Required and must be set to ODBC; no default value
ReactivateTimerInterval	Required; default is 300000 (ms)
Timeout	Required; default is 15 (seconds)
DataSourceConnections	Required; number of concurrent connections to data source (default is 8)
ODBCDataSource	Required; no default value

Table 20-2 ODBC Remote Server Parameters (continued)

Parameter	Description
SQLDefinition	SQLDefinition/ (mandatory, no default); UserPasswordAttribute = (mandatory, no default; data store field for user password) SQLStatements/ SQLStatement1/ SQLStatement2/
ODBCToRadiusMappings	(optional) A list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the data store attribute retrieved. The data store attributes must match those defined in the external SQL file.
ODBCToEnvironmentMappings	(optional) A list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ODBC attribute retrieved.
ODBCToCheckItemMappings	(optional) A list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to be checked against the value of the data store attribute retrieved.

ODBC Data Source

ODBCDataSource is the name of the datasource to be used by the remote server. An ODBCDataSource name can be reused by multiple remote servers. You configure ODBCDataSources under **/Radius/Advanced/ODBCDataSources**. See [Configuring an ODBC DataSource, page 20-6](#), for more information.

SQL Definitions

SQLDefinitions lists the UserPasswordAttribute and one or more SQL statements, listed numerically in the order to be run. The UserPasswordAttribute represents a column in the database that contains users' password information. Individual SQLStatements are numbered SQL1 through SQL n under SQLStatements, as shown in the following example:

```
SQLDefinition/
  UserPasswordAttribute = asdfjkl
  SQLStatements/
    SQL1/
    SQL2/
    SQL3/
    ...
```

The following example is an SQL statement used for Authentication and Authorization:

```
SQLStatements/
  SQL1
    Name = SQL1
    Type = query (mandatory, no default; must be query)
    SQL = SQL statement (mandatory, no default)
```

ExecutionSequenceNumber = Sequence number for SQLStatement execution. (mandatory, no default and must be greater than zero).
 MarkerList = UserName/SQL_DATA_TYPE (mandatory, UserName must be defined)

Table 20-3 describes the SQL Statement parameters.

Table 20-3 SQL Statement Parameters

Parameter	Description
Name	Name/number of SQL statement
Type	Query (mandatory, no default value)
SQL	SQL query statement
ExecutionSequenceNumber	Sequence number for SQLStatement execution, must be greater than zero (mandatory, no default)
MarkerList	Defines all markers for the query. MarkerList uses the format <i>UserName/SQL_DATA_TYPE</i> .

SQL Syntax Restrictions

You must observe the following SQL syntax restrictions in SQL queries for CAR 4.2.

1. The SQL statement must be in the format of SELECT ... FROM ... WHERE ..." (Statements might be in lower-case.)



Note 'WHERE' is compulsory in the SQL statement.

2. Any arguments to Oracle functions like *distinct*, *count* must be given within braces, as shown in the following example:

```
select distinct(attribute),password from profiles where username=?
```

The resulted column from *distinct(attribute)* will be put into *attribute* which can be used for ODBC Mappings. The actual result set from Oracle for this column would be named *distinct(attribute)*.

3. The column list in the SQL statement must be delimited with a comma (,) and any extra spaces between statements are ignored. Aliasing for column names in SQL is not allowed. SQLDefinition properties define the SQL you want to execute, as shown in the following example.

Specifying More Than One Search Key

You can specify more than one search key for a table in the SQL SELECT. To do so, add another search criteria to the SQL statement and add the environment variable name to the MarkerList. For example, the following query and MarkerList can be used to look up a username and CLID match.

```
select password from user_table where username = ? and clid = ?
```

In this case, the marker list would look like this:

```
UserName/SQL_CHAR clid/SQL_CHAR
```

To configure the multiple entries in the MarkerList list, surround the entire string in double quotes like the following:

```
set MarkerList "UserName/SQL_CHAR CLID/SQL_CHAR"
```

To make this work, a variable called CLID must be in the environment dictionary. You can use a script to copy the appropriate value into the variable.

ODBCToRadiusMappings

You configure ODBCToRadiusMappings with a list of *name/value* pairs where name is the name of the data store attribute to retrieve from the user record and the value is the name of the RADIUS attribute to set to the value of the data store attribute retrieved.

For example, use the following **aregcmd** command to set a value for the variable *Framed-IP-Address*:

```
set FramedIPAddress Framed-IP-Address
```

When the ODBCToRadiusMappings has this entry, the RemoteServer retrieves the attribute from the data store user entry for the specified user, uses the value returned, and sets the response variable *Framed-IP-Address* to that value.

When an SQL select statement returns more than one row for a column mapped under ODBCToRadiusMappings, multiple Radius attributes are created.

For example, consider the following SQL *select* statement with ciscoavpair configured to Cisco-AVPair under ODBCToRadiusMappings. The table.column syntax requires an SQL alias for the mapping to work, as shown in the following example:

```
SQLStatements/
  SQL1/
    select table1.abc as t1abc, password from table2 where username = ?
    Mapping: t1abc = my_mapping
```

If two rows are returned for ciscoavpair column, two Cisco-AVPair attributes will be created.

ODBCToEnvironmentMappings

Under ODBCToEnvironmentMappings there is a list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ODBC attribute retrieved.

For example, when the ODBCToEnvironmentMappings has the entry: group =User-Group, the RemoteServer retrieves the attribute from the ODBC user entry for the specified user, uses the value returned, and sets the environment variable User-Group to that value. When an SQL select statement returns more than one row for a column mapped under ODBCToEnvironmentMappings, the value for all rows is concatenated and assigned to the environment variable.

Configuring an ODBC DataSource

ODBCDataSource is the name of the datasource to be used by the remote server. You configure ODBCDataSources under **/Radius/Advanced/ODBCDataSources**. Multiple remote servers can use the same ODBCDataSource.

Under the ODBCDataSource object definition, a list defines **ODBC.ini** filename/value pairs for a connection. The list includes a Type field and a Driver field, different for each Driver and Data Source, to indicate its Driver and Data Source. CAR 4.2 currently supports only the Easysoft Open Source Oracle Driver.

Table 20-4 describes the Easysoft Open Source Oracle Driver options.

Table 20-4 Easysoft Open Source Oracle Driver Options

Parameter	Description
Name	Name of the ODBCDataSource
Type	Required; must be Oracle_es
Driver	Required; liboarodbc.so (default value)
Database	Required; Oracle Client configuration database name (no default value)
UserID	Required; database user name (no default value)
Password	Optional user password; shown encrypted

Setting ODBC As Authentication and Authorization Service

Use **aregcmd** to configure the ODBC Service as the default authentication and authorization service under **//localhost /Radius** as in the following:

```
set DefaultAuthenticationService odbc-service
```

```
set DefaultAuthorizationService odbc-service
```



Note

When you use an ODBC service, configure the **BackingStoreDiscThreshold** property under **/Radius/Advanced** to ensure that the data generated by log files do not exceed the size limit configured.

Saving Your Configuration

When you use **aregcmd** to **save** your configuration, CAR attempts to validate the configuration, checks for all required parameters, and ensures there is no logic error. If the validation is successful, the configuration is saved to the MCD database. When you **reload**, CAR shuts down any current ODBC connections and builds new connections for the configured ODBC Data Sources.

MySQL Support

CAR 4.2 provides support for MySQL to query user records from a MySQL database and enables you to write accounting records into MySQL when using Oracle accounting. CAR 4.2 has been tested with MySQL 4.0.18 and MyODBC 3.51.06 (reentrant).

MySQL Driver

You can download the MySQL driver from the MySQL website at **<http://mysql.com>**. You can go directly to the driver download page using the following URL:

<http://dev.mysql.com/downloads/connector/odbc/3.51.html>

Save the downloaded file to a temporary location such as **/tmp**. Use commands like the following to unzip and install the driver:

```
gunzip -c MyODBC-3.51.06-sun-solaris2.8-sparc.tar.gz | tar xvf -
ln -s MyODBC-3.51.06-sun-solaris2.8-sparc myodbc
```

Configuring a MySQL Datasource

To configure the CAR server to query records from a MySQL database, configure the following:

- ODBCDataSource object
- RemoteServer object
- ODBC service
- Default AA services

Step 1 Log in to the CAR server and launch **aregcmd**.

Log in as a user with administrative rights such as user **admin**.

Step 2 Change directory to the **/Radius/Advanced/ODBCDataSources** and add a new ODBCDataSource.

```
cd /Radius/Advanced/ODBCDataSources
add mysql
```

Step 3 Set the new ODBCDataSource type to myodbc.

```
cd mysql

[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
  Name = mysql
  Description =
  Type =
```

```
set type myodbc
```

The following is the default configuration for an ODBCDataSource object of type myodbc:

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
  Name = mysql
  Description =
  Type = myodbc
  Driver =
  UserID =
  Password =
  DataBase =
  Server =
  Port = 3306
```

Step 4 Set the Driver property to the path of the MyODBC library. Use a command like the following:

```
set driver /scratch/myodbc/libmyodbc3_r.so
```

Step 5 Set the UserID property to a valid username for the MyODBC database and provide a valid password for this user.

```
set userid ar-mysql-user
```

```
set password biscuit
```

- Step 6** Provide a DataBase name and the name of the CAR RemoteServer object to associate with the ODBCDataSource.

```
set database database_name
```

```
set server remote_server_name
```

- Step 7** Change directory to **/Radius/RemoteServers** and add a RemoteServer object to associate with the new ODBCDataSource.

```
cd /Radius/RemoteServers
```

```
add mysql
```

- Step 8** Change directory to the new RemoteServer and set its protocol to odbc.

```
cd mysql
```

```
set protocol odbc
```

- Step 9** Set the ODBCDataSource property to the name of the ODBCDataSource to associate with this RemoteServer object.

```
set ODBCDataSource mysql
```

- Step 10** Change directory to **/Radius/Services** and add an ODBC service as described in [Configuring an ODBC Service, page 20-2](#).

- Step 11** Change directory to **/Radius** and set the DefaultAuthenticationService and DefaultAuthorizationService properties to the ODBC service added in the previous step.

Example Configuration

The following shows an example configuration for a MySQL ODBC data source. See [Configuring an ODBC DataSource, page 20-6](#) for more information.

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
  Name = mysql
  Type = myodbc
  Driver = /tmp/libmyodbc3_r.so
  UserID = mysql
  Password = <encrypted>
  DataBase = test
  Server = mysql-a
  Port = 3306
```

The following shows an example configuration for a RemoteServer. See [Configuring an ODBC RemoteServer, page 20-3](#) for more information.

```
[ //localhost/Radius/RemoteServers/mysql-a ]
  Name = mysql
  Description =
  Protocol = odbc
```

```

ReactivateTimerInterval = 300000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource = mysql
KeepAliveTimerInterval = 0
SQLDefinition/
UserPasswordAttribute = asdfjkl
SQLStatements/
  SQL1/
    Name = SQL1
    Type = query (mandatory, no default; must be query)
    SQL = SQL statement (mandatory, no default)
    ExecutionSequenceNumber = Sequence number for SQLStatement
    execution.(mandatory, no default and must be greater than zero).
    MarkerList = UserName/SQL_DATA_TYPE ..... (mandatory, UserName must be defined)
  SQL2/
  SQL3/
ODBCToRadiusMappings/
ODBCToEnvironmentMappings/
ODBCToCheckItemMappings/

```

The following shows an example configuration for an ODBC service. See [Configuring an ODBC Service, page 20-2](#) for more information.

```

[ //localhost/Radius/Services/ODBC ]
  Name = ODBC
  Description =
  Type = ODBC
  IncomingScript~ =
  OutgoingScript~ =
  OutagePolicy~ = RejectAll
  OutageScript~ =
  MultipleServersPolicy = Failover
  RemoteServers/
    1. mysql-a

```

The following shows an example configuration where the DefaultAuthenticationService and DefaultAuthorizationService properties have been set to the ODBC service.

```

[ //localhost/Radius ]
  Name = Radius
  Description =
  Version = 4.2
  IncomingScript~ =
  OutgoingScript~ =
  DefaultAuthenticationService~ = ODBC
  DefaultAuthorizationService~ = ODBC

```