

Data Administration  
**Authentication and Authorization Methods**  
**for Applications using SQL Server Databases**

Ver 1.3

August 7, 2003

**General Guidelines:**

- Database analysts, security personnel, and application developers must work closely together on database management system (DBMS) security issues.
- Application security issues and needs should be identified early in a project.  
Applications are a portal into the data of the DBMS. [You] cannot fully protect against a poorly designed application security architecture. - Giga
- **Authentication** is typically handled via the OS (operating system) and not the DBMS.
- **Authorization** is handled through the DBMS unless external security procedures are available.
- DBAs will physically implement security via the DBMS. Security personnel should be aware of and authorize changes in security levels and changes in access at the user level.
- Internet access to a DBMS should always be through an application server using filtered access and behind a firewall.
- Encryption of data stored in a database is used only in rare cases.
- Application Developers typically don't have access to alter production data or production database schemas.

**Authentication:**

(Who you are)

1. **Database User Level Security** – This method authenticates users through either a SQL Server login or their network account. It can use various means of SQL Server authorization to control which users can gain access to which data resources and at what permission level.

This is the only authentication method which allows auditing (within the database) of user actions.

2. **Single “Application” Account Access** – This is a common method which allows all users to connect to a database using a single SQL account. By itself, this method provides little or no control over individual user authorization within the database. By itself, this method is not recommended and is only applicable where data security is not a concern. This method may be combined with “application based permissions” (see below) for added control.

Often, if a single account is used, the account will have DBO (database owner) rights.

**Authorization:**

(What you can do)

\* The following assume authentication through one of the above methods.

1. **Individual (or Standard Role) Database Permissions** – Rights may be assigned to the user either individually or through membership in a SQL Server standard role. (SQL Server standard roles simplify the administration of user rights by grouping users into a “role” and granting specific database permissions to the role.)

The drawback to this method is that the user retains all rights in the database regardless of the access path. In other words, if a user bypasses the application and connects to the database via MS Access, they would still have the same insert, update, and delete rights that they have with the application, only without the added verifications and controls the application provides.

"Public" roles can be created for read-only access to data requiring public availability.

2. **Application Role Database Permissions** – SQL Server application roles provide a method to assign database permissions that can be enabled only through an application. This method allows the DBA to restrict a user's assigned rights while providing elevated rights when the database is accessed through the application. The application sets the application role and password. Without the application role, the user does not gain the elevated permissions.

Data Administration  
**Authentication and Authorization Methods  
for Applications using SQL Server Databases**

Ver 1.3

August 7, 2003

Multiple application roles may be used within a database or application system. Only one application role may be active for a session at any given time. Additional security must be built into the application to control a user's permissions if required.

3. **Application-Based Permissions** – This method may or may not be combined with either or both of the *database* authorization methods. With this method an additional security layer is built into the application to control which application operations a user may make use of.

This method typically will utilize a “user” table within the database or a secondary “security” database.

Passwords stored in the “user” table must be encrypted using MS SQL Server's PWDEncrypt() function. They may then be evaluated using SQL's PWDCmpare() function. The column used to store encrypted passwords should be defined as VarBinary(256).

Minimally, permissions should be set to ensure that users do not have select rights on the “user” table. In order to enforce this restriction, a stored procedure or user defined function must be created which will allow comparison of the entered password with the user's stored password. The procedure would only return a True/False, Yes/No, Go/NoGo result.

### **Special Access**

Special security arrangements are maintained for emergency access to production databases by application development and support personnel. This uses special accounts called Emergency Access Logon IDs (EALIDS).

Please ensure that you have the proper EALIDs set up for the databases which serve the applications you support. Contact a member of Data Administration if you have any questions.