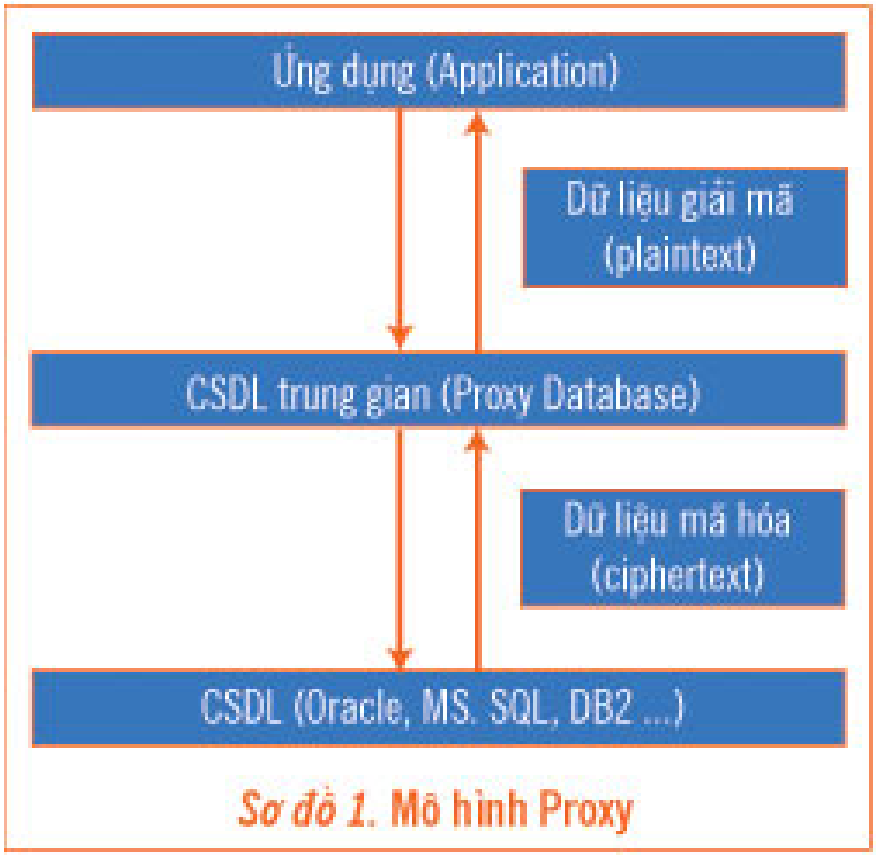
Database security method

In the data security strategy, most companies now focus resources on data protection on the transmission line. Meanwhile, the data protection issue in the database (database, database) has not been properly concerned.

In the data security strategy, most companies now focus resources on data protection on the transmission line. Meanwhile, the data protection issue in the database (database, database) has not been properly concerned.

In fact, security problems with the database can seriously affect the company's reputation and customer relations. Security incidents of stealing 40 million credit cards of recent customers with Master Card and Visa Card have partly increased attention to database security solutions.

Within the scope of this article, the writer wants to present database security solutions by constructing the encryption layer.



Sơ đồ 1. Mô hình Proxy

The simplest solution to protect data in databases at the file level, against unauthorized access to database files is encryption. However, data encryption at this level is a 'get eaten, fall down' solution, this solution does not

provide a level of security to access the database at table level (table), column (column) and row (row). Another weakness of this solution is that anyone with database access can access all data in the database. This creates a serious risk, allowing objects with administrative rights (admin) to access all sensitive data. In addition, this solution is limited because it does not allow different permissions for database users.

The second solution, as opposed to the file-level encryption solution mentioned above, solves the problem of encryption at the application level. This solution handles data encryption before transferring data to the database. Key management and access issues are supported by the application. Querying data to the database will return data results in encrypted form and this data will be decoded by the application. This solution solves the problem of security separation and supports role-based security policies (Role Based Access Control - RBAC). However, coding on the application layer requires a comprehensive change of the application architecture, even requiring the application to be rewritten. This is a significant issue for companies with many applications running on many different database platforms.

From the analysis of the two solutions mentioned above, it is easy to see that an optimal database security solution should support the following main factors:

1. Support encryption at table, column, and row level data.

2. Support security policy to assign access to column data level, support RBAC.

3. Encryption mechanism does not affect current applications.

Here are two models that meet the above requirements, especially the third one.

1. Build intermediate database layer

In this model, an intermediate database (proxy) is built between the application and the original database (Diagram 1). This intermediate database plays the role of encrypting data before updating to the original database, and decoding the data before providing it to the application. The intermediate database also provides additional key management functions, user authentication and access licensing.

This solution allows to create more security functions for the database. However, the intermediate database model requires building a database application to replicate all functions of the original database.

Currently, on the market of database encryption products, Secure.Data of Protegrity (www. Protegrity.com) uses the above proxy model.

2. Use the mechanism available in the database

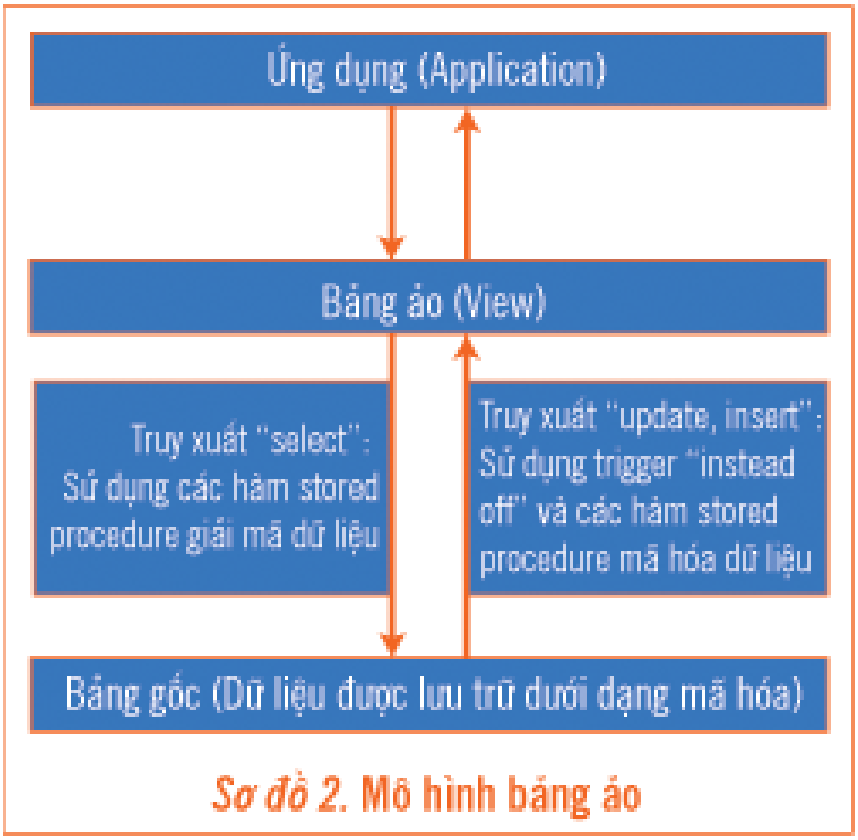
This model addresses column coding issues based on the following mechanisms:

a. The Stored Procedure functions in the database for coding and decoding functions

b. Using the View mechanism in the database creates virtual tables, replacing the actual encrypted tables.

c. The 'instead of' trigger mechanism is used to automate the encoding process from View to the original table.

In this model, the data in the original tables will be encrypted, the name of the original table is changed. A virtual table (View) is created with the name of the original table, the application will access this virtual table.



Sơ đồ 2. Mô hình bảng ảo

Retrieving data in this model can be summarized as follows (Diagram 2):

Data retrieval to the original table will be replaced by access to the virtual table.

Virtual tables are created to simulate data in the original table. When executing the 'select' command, the data will be decoded for the virtual table from the original (encrypted) table. When executing the 'Insert, Update' command, 'instead of' trigger will be executed and encrypted data to the original table.

Managing distribution of access to columns will be managed on virtual tables. In addition to the basic rights provided by the database, two new permissions are defined:

1. Users may only read data in encrypted form (ciphertext). This right is suitable for those who need to manage databases without reading the data content.

2. Users are allowed to read data in decoded form (plaintext).

The above solution has simple advantages, easy to develop. However, due to the limitations of view, trigger and data management mechanisms, this solution has the following limitations:

Index columns cannot be encrypted, thus restricting applications that need to support index

Encrypted data is large in size compared to the original data. This difference is not significant for text (text) data, but is significant for digital and binary data. For example, 1-byte numeric data will be increased by 2 bytes after encryption.

Database access speed decreases due to the implementation of encryption layer

Currently, on the market of database encryption products, DBEncrypt (www.appsecinc.com) and nCypher (www.ncypher.com) developed according to the above model.

CISSP - CISSP,
Project Manager, Global Cybersoft Vietnam

Refer

White Paper: Achieving Data Privacy Through Database Encryption.
White Paper: Protecting Oracle Database.
White Paper: Encryption of Data at Rest.