Security information management challenges and solutions

Manage information security in DB2 and Informix Dynamic Server

Samantha Tran
Manoj Mohan

July 13, 2006

Learn to identify the specific challenges in your own environment, and get on the right track with the task of designing and administering a strong security solution using either DB2® or Informix® Dynamic Server (IDS). In addition, a handy chart lays out the problems and the solutions, and lists how the solutions are implemented in both Informix and DB2. Validating a user's identity, controlling access, maintaining integrity and privacy -- these are all hot issues in information management today. Every company is faced with the task of providing a comprehensive plan for data security that addresses all these issues.

Secure information management

Managing secure information is one of the most difficult tasks to implement and maintain effectively. In the current network-centric business model it is becoming increasingly difficult to validate a person's identity, control access, and maintain integrity and privacy of data. Security is a multi-faceted problem that requires close analysis of all the vulnerable factors in a business infrastructure. While authentication, authorization, and encryption do not encompass all facets of information management, they are the three main areas of concern and are the areas that are examined in this article.

Authentication

Authentication methods seek to guarantee the identities of system users. The traditional user ID and password method of authentication is no longer effective or sufficient in this day and age where security is a major concern for large infrastructures. An additional challenge is that applications frequently need authentication models that segregate the security policy from the application, such as Lightweight Directory Access Protocol (LDAP), or Kerberos.

Businesses require that an authentication framework be easily maintained and updated. Its components may have to be changed due to deficiencies found in the authentication algorithms or to changes in requirements for system authentication. The solution to this problem must take into
account how the applications can use the new authentication framework in a generic way and how the users are authenticated by the framework in a generic way.

The two major frameworks that exist currently to enable applications to plug-in different authentication models are Generic Security Services Application Programming Interface (GSS-API) and Pluggable Authentication Module (PAM). DB2 UDB supports GSS-API and IBM IDS supports PAM.

### GSS-API

**More about GSS-API on DB2 UDB**

Learn how to use GSS-API security mechanisms to customize the IBM DB2 Universal Database (DB2 UDB) security plug-ins to achieve authentication based on public key technology. "Security plug-ins using the GSS-API security mechanisms (SPKM / LIPKEY)” (developerWorks, December 2005).

GSS-API enables application control over security. A programmer using GSS-API can write an application that is ignorant of the details of protecting network data. GSS-API also provides a framework that enables security services to callers in a generic fashion, which supports a variety of technologies like Kerberos or Public Key Mechanism.

### PAM

PAM is an authentication mechanism that enables system entry services such as login, rlogin, and telnet to be customized to incorporate changes required for an application. With the PAM framework, multiple authentication technologies can be added without changing any of the login services, thereby preserving existing system environments. PAM can be used to integrate login services with different authentication technologies, such as RSA, DCE, Kerberos, S/Key, and smart card-based authentication systems. Thus, PAM enables networked machines to exist peacefully in a heterogeneous environment, where multiple security mechanisms are in place.

### Trusted context

In a multi-tier environment, such as a transaction processing environment, it is sometimes necessary to control the security of middle-tier applications by preserving each end-user's identity and privileges through all tiers, and auditing actions. Many problems can arise from this scenario, such as loss of end-user identity, diminished end-user accountability, over-granting privileges to middle-tier’s authorization ID, and weakened security due to the fact that the middle-tier authorization ID must acquire all privileges of the end-users that might establish a connection.

DB2 has introduced **trusted context** authentication to permit the middle-tier to do this type of authentication. A trusted context is an object that gives users a specific set of privileges and is available when the user connects to the database through a trusted connection. For example, in a Web application environment, the middle-tier application establishes a trusted connection through a trusted context, thus enabling the end-user’s identity to be passed to the database server.

A trusted context allows for the definition of a unique set of interactions between DB2 and the external entity. This includes the ability for the external entity, such as a middleware server, to...
use the existing database connection under a different user without the need to authenticate the new connection user. It also provides the ability for a DB2 authorization ID to acquire a special set of privileges within a specific trusted context that are not available to it outside that trusted context, by defining roles. Websphere Application Server, PeopleSoft V7, Domino, and SAP R/3 are among the software products that support this three-tiered application model. When the middle tier establishes a connection with DB2, the middle-ware’s user ID and password are used for authentication purpose. Furthermore, the database privileges associated with the middle tier’s authorization ID is used for all authorization checking that must occur for any database access, including those accesses performed by the middle tier on behalf of an end-user. After the application has established a trusted connection with the DB2 server, the application may switch users associated with the trusted context object in the backend.

There are several implications from introducing trusted context in an application environment:

- The application is able to validate the credentials of a user by passing them directly to a database server. It is also possible to authenticate each user at the backend server. This makes it easier to audit the actions of each user and improves accountability.
- The database administrator is able to monitor which end-users are allowed to access the database server through specific middle-tier applications.
- The database administrator is able to audit actions of the middle-tier application acting on behalf of a given set of users. Since each middle tier can be delegated the ability to authenticate and act on behalf of a specific set of users, and with a specific set of roles, trusted context supports a limited trust model for the middle-tier application, and avoids the problem of an super user in the middle tier with all the privileges.
- The performance overhead is significantly reduced since there is no cost for establishing a new physical connection for each end-user. The middle-tier establishes a trusted connection and has the ability to switch end-users through the trusted context.

**Authorization**

In addition to authentication issues, threats to the security of a database server involve unauthorized access to sensitive information. After a user is authenticated, the database server authorizes that user to perform different types of operations. Authorization is necessary to ensure that each user has the appropriate level of access privileges. For example, a CFO of a company may have a need to access the financial records at the corporate level, while a first line manager has more limited access and may only be able to see his or her employees' payroll records. Fine-grained, privilege-based authorization allows organizations to manage access control by honing in on specific privileges and permissions for a user based on access requirements.

DB2 and IDS have implemented Role-Based Access Control (RBAC), which is a solution to restricting system access to authorized users. It combines the approach to Mandatory Access Control (MAC) and Discretionary Access Control (DAC). Within an organization, roles are created for various job functions performed by users. The permission to perform certain operations are assigned to a role. The system users are then assigned particular role(s), and through these role(s) assignment(s) the users acquire appropriate permissions to perform system functions. However, in environments where multiple levels of security are required, for instance...
Department of Defense (DoD). The system data is available to a user based upon the user's level of security clearance. Each user should have access to a security level based upon the level of data sensitivity he or she can see. To address this requirement, DB2 has implemented Label Based Access Control (LBAC). Each row or column can be assigned a security label that stores information about the classification, or sensitivity, of the data. Similarly, each database user is assigned a security label that determines which labeled data rows or columns he or she can access.

**RBAC**

The central notion of RBAC is that users do not have discretionary access to enterprise objects. Instead, access permissions are associated with roles, and users are made members of appropriate roles. RBAC greatly simplifies the management of authorization while providing an opportunity for system administrators to control access to enterprise objects at a level of abstraction that is close to the structure of their enterprise. It has quickly been adopted by a large number of software products, and in particular by Relational Database Management Systems (RDBMS).

**LBAC**

Label-Based Access Control is a means by which a database system controls access to a database object based on a security label contained in that object and the security label granted to the user attempting to access that object. The DB2 LBAC approach is to allow users to define the set of components that make up a security label and to specify the access rules. It allows users to define the structure of the security label to be used. LBAC lets users decide exactly who has write access and who has read access to individual rows and individual columns. A security label component is a new database entity that can be created, dropped, and altered. It is introduced as a building block for security labels. A security label is composed of one or more security label components. There are three types of security label components: arrays, sets, and trees. Once a user has defined the security label components, the user can then define the security labels and associate that security label to a user or to a security row(s).

**Encryption**

Encryption is not directly related to authentication and authorization but is an important aspect of protecting data, during transit or at rest, from unauthorized users. Network protocols such as HTTP, SMTP, and FTP do not provide adequate protection for sensitive data sent across the networks. The data transmitted over the network is susceptible to network attacks like snooping the network traffic, non-repudiation, tampering the data, spoofing, hijacking, and capture-replay.
Secure Socket Layer (SSL) is a great advancement over the traditional protocols. SSL ensures confidentiality and integrity of data transmission over the network. IDS supports encryption over the wire through OpenSSL library.

For certain applications you may decide to encrypt data as an additional measure of security. Most issues of data security can be handled by appropriate authentication and access control, ensuring that only properly identified and authorized users can access data. However, data in the database cannot normally be secured against the database administrators, since DBAs have unlimited privileges. Likewise, organizations may have concerns about securing sensitive data stored off-line, such as backup files stored with a third party. They may want to guard against intruders accessing the data where it is physically stored on the database. In order to protect data at rest, DB2 and IDS both support column level encryption (CLE). The database server can use CLE to store data in an encrypted format and provide password based access.

Managing secure information is one of the most difficult tasks to implement and maintain effectively. In this article, we have attempted to present the solutions to security problems that might exist in the business infrastructure. We have presented the solutions to security problems in authentication, authorization, and protecting data through encryption.

**Data-at-rest encryption**

IBM Informix Server and DB2 support CLE. It is used to set encryption passwords for columns containing sensitive data, such as credit card numbers. There are available built-in encryption functions like ENCRYPT_AES() and ENCRYPT_TDES() to encrypt data in columns containing the following character data types or smart large object data types: CHAR, NCHAR, VARCHAR, NVARCHAR, LVARCHAR, BLOB, CLOB. Once the data is encrypted, only users who can provide a secret password can decrypt the data. All values in a specific column of a database table are encrypted with the same password provided by the user, the same encryption algorithm, and the same cipher mode. Users can also use the SET ENCRYPTION PASSWORD statement to set an encryption password for a session. Only the users who can provide a secret password can view, copy, or modify encrypted data.

**Data in transit**

SSL developed by the Netscape Corporation, is an industry-accepted standard for network transport of data over a secure channel. SSL is supported by all currently available Web servers and Web browsers. The SSL protocol provides authentication, data encryption, and data integrity, in a public-key infrastructure (PKI). SSL addresses the problem of protecting user data exchanged between tiers in a three-tier system. By providing strong, standards-based encryption and integrity algorithms, SSL provides system developers and users with confidence that data will not be compromised in the Internet. Unlike password-based authentication, which authenticates client
to server only, SSL can authenticate server to client as well as client to server. This is a useful feature when building a Web-based three-tier system, since users often insist on authenticating the identity of an application Web server before they provide the server with sensitive information, such as credit card numbers. IDS enables encryption of data transmitted over the network using the encryption communication support module (ENCCSM).

**Security challenges and solution matrix**

*Table 1* shows the challenges in keeping information secure and provides solutions available in DB2 and IDS. It also compares the solutions available in Oracle.

**Table 1. Comparison of solutions provided by IBM and Oracle®**

<table>
<thead>
<tr>
<th>Category</th>
<th>Problems</th>
<th>Solutions</th>
<th>Security Technology</th>
<th>IBM DB2</th>
<th>IBM IDS</th>
<th>Oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authentication</strong></td>
<td>Maintainability of authentication infrastructure</td>
<td>Segregate security policy through generic authentication mechanism</td>
<td>PAM</td>
<td>GSS-API addresses how applications use the new authentication mechanisms in a generic way</td>
<td>PAM addresses how the user is authenticated by these mechanisms in a generic way</td>
<td>Strong Authentication: Supports Kerberos, DCE, RADIUS</td>
</tr>
<tr>
<td></td>
<td>Context sensitive authentication</td>
<td>Establish trust relation with server and application</td>
<td>Trust relationships</td>
<td>Trusted context</td>
<td></td>
<td>Proxy authentication</td>
</tr>
<tr>
<td><strong>Authorization</strong></td>
<td>Unauthorized access to data</td>
<td>Limit access to data rows and columns</td>
<td>LBAC</td>
<td>LBAC for DB2 9 limits access to both table rows and columns</td>
<td>IDS has implemented roles</td>
<td>Oracle Label Security controls access to table rows based on security labels.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Limit the privileges a user acquires</td>
<td>RBAC</td>
<td>RBAC for DB2 9</td>
<td></td>
<td>Oracle has implemented roles</td>
</tr>
<tr>
<td><strong>Encryption</strong></td>
<td>Eavesdropping on communication</td>
<td>Protect the network</td>
<td>SSL</td>
<td>No support currently. DB2 supports encryption through Distributed Relational Database Architecture (DRDA) encryption, and password encryption.</td>
<td>Supports with implementation of OpenSSL</td>
<td>Oracle implements SSL protocol</td>
</tr>
<tr>
<td></td>
<td>Unauthorized access through physical data</td>
<td>Protect data on disk</td>
<td>Data encryption</td>
<td>DB2 supports data encryption through the following built-in functions: ENCRYPT, DECRYPT_BIN, DECRYPT_CHAR, and GETHINTNo support currently</td>
<td>Column-level encryption</td>
<td>Oracle provides a PL/SQL package to encrypt and decrypt stored data with Obfuscation Toolkit.</td>
</tr>
</tbody>
</table>
Related topics

- Download a free trial version of IBM Informix Dynamic Server and IBM DB2 UDB at Information Management Downloads.
- Build your next development project with IBM trial software, available for download directly from developerWorks.
- "DB2 UDB security, Part 3: Security plug-ins using the GSS-API security mechanisms (SPKM / LIPKEY)" (developerWorks, December 2005): Learn how to use GSS-API security mechanisms to customize the IBM DB2 UDB security plug-ins to achieve authentication based on public key technology.
- "Encryption over the wire with IDS 9.40" (developerWorks, January 2004): Discover how the encryption technology has been incorporated in IDS 9.4.

© Copyright IBM Corporation 2006
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)