DB2 Security and Audit

Kevin Street
street@ca.ibm.com
Disclaimer

This presentation is intended to provide general background information, not regulatory, legal or other advice. IBM cannot and does not provide such advice. Readers are advised to seek competent assistance from qualified professionals in the applicable jurisdictions for the types of services needed, including regulatory, legal or other advice.
The Importance of Data Security

- Historically focus on physical, network, and host security
- But database is where the valuables are kept!
- Data security has now moved to forefront, mostly due to rash of large breaches

Source: Flowingdata.com
Why Aren't all Databases Secure?

- Performance, features and scalability are often the requirements - security is usually an afterthought.
- Performance vs. Security - guess who wins?
- DBA’s are not usually security people, and vice versa.
- Lack of understanding of the threats.
- Vendors usually present security as feature / function without context or what it protects against.
- Security is taken care of in other layers – so why worry?
Counter Measures: Secure Data Server Features

- Ties DB2’s extensive security features to the threats they help protect against
  - Authentication
  - Authorization
  - Database Roles
  - Trusted Contexts
  - Label-Based Access Control
  - Auditing
  - Encryption
  - Static SQL / pureQuery
Key Security Features in DB2
Key Security Features in DB2

**Authentication**
- Authorization
- Database Roles
- Trusted Contexts
- Label-Based Access Control
- Auditing
- Encryption
- Static SQL / pureQuery
Authentication vs. Authorization

Authentication:
- Is this correct password for Mary?

Authorization:
- Does mary have an authorization to perform SELECT FROM mytable?

Connect to sample user mary using pwd.
Authentication

- **The process by which a user is validated**
  - Performed outside of DB2 via authentication security plug-in
  - Default is OS based authentication (Shipped with DB2)

- **Authentication Types**
  - SERVER
  - SERVER_ENCRYPT
    - User ID and password encrypted
  - DATA_ENCRYPT
    - Data and User ID and password encrypted
  - KERBEROS
  - GSSPLUGIN
  - LDAP
  - CLIENT
Key Security Features in DB2

- Authentication
- **Authorization**
  - Database Roles
  - Trusted Contexts
  - Label-Based Access Control
  - Auditing
  - Encryption
  - Static SQL / pureQuery
Authorization

- The process of checking whether a user is allowed to execute a statement or command
- Involves granting a set of permissions available to the authorization ID

Permissions can be obtained from 4 sources:
- Permissions held by the authorization ID itself
- Permissions held by the authorization ID’s groups
- Permissions held by the authorization ID’s roles
- Permissions held by PUBLIC
Authorization (Cont.)

- Permissions are divided into authorities and privileges

- Authorities
  - System level authorities (eg. SYSADM, SYSCTRL)
  - Database level authorities (eg. DBADM, SECADM)

- Privileges
  - Required to perform specific actions on database objects
  - Database, Table, View, Indexes, Schema, etc.
Separation of Duties - DB2 9.7

- **Separation of DBADM and SECADM**
  - Extend the scope of the current SECADM authority to be able to fully manage security and be grantable to roles and groups
  - Remove the capability to grant and revoke the DBADM and SECADM authorities from the SYSADM authority
  - Provide an option whereby a customer can set up a DBADM that does not have the capability to grant and revoke privileges

  ```sql
  GRANT DBADM ON DATABASE WITHOUT ACCESSCTRL TO USER JOE
  ```
Separation of Duties – DB2 9.7 (Cont.)

- **Data access**
  - Remove the implicit DBADM authority from SYSADM authority
  - Provide option whereby a customer can set up a DBADM that does not have any inherent capabilities to access table data
    
    `GRANT DBADM ON DATABASE WITHOUT DATAACCESS TO USER JOE`

  - Provide an option whereby a customer can set up a user to perform query plan analysis without having to give them the privilege to access and modify table data
    
    `GRANT EXPLAIN ON DATABASE TO USER JOE`
Separation of Duties – DB2 9.7 (Cont.)

- **Least privilege**
  - WLMADM
    - Manages WLM objects
    - No need to grant users DBADM to manage WLM objects
  - SQLADM
    - Responsible for SQL tuning
    - No need to grant users DBADM to perform SQL tuning
  - EXPLAIN
    - Grants the ability to explain and prepare SQL statements
    - No need to grant users actual privileges on base tables
  - DBADM Grants
    - No longer grant the extra CONNECT, LOAD, CREATETAB, BINDADD, LOAD, CREATE_EXTERNAL_ROUTINE, IMPLICIT_SCHEMA, QUIESCE_CONNECT
    - The above used to remain in effect when DBADM is revoked
Key Security Features in DB2

- Authentication
- Authorization

**Database Roles**
- Trusted Contexts
- Label-Based Access Control
- Auditing
- Encryption
- Static SQL / pureQuery
Database Roles

- **What is a database role?**
  - A database object that groups together one or more privileges, authorities, security labels, or exemptions
  - Can be granted to users, groups, PUBLIC, or other roles

- **What is the advantage of database roles?**
  - Simplify the management of privileges in a database
  - SECADMs control access to their databases using the structure of their organizations
  - Control of specific roles can be delegated to others
Database Roles (Cont.)

- **Assign users to roles not groups**
  - Roles are controlled inside the database
  - Group privileges and authorities are not considered by DB2 when creating views, triggers, MQTs, static SQL, and SQL routines
  - DB2 cannot know when membership in groups change so that it can invalidate the database objects (for example, a view) created by users who relied on their group privileges to succeed
Database Roles – Example

- Without database roles:

  GRANT SELECT ON TABLE SERVER TO USER BOB, USER ALICE
  GRANT SELECT ON TABLE CLIENT TO USER BOB, USER ALICE
  GRANT SELECT ON TABLE TOOLS TO USER BOB, USER ALICE

  REVOKE SELECT ON TABLE SERVER FROM USER BOB, USER ALICE
  REVOKE SELECT ON TABLE CLIENT FROM USER BOB, USER ALICE
  REVOKE SELECT ON TABLE TOOLS FROM USER BOB, USER ALICE

  GRANT SELECT ON TABLE SERVER TO USER TOM
  GRANT SELECT ON TABLE CLIENT TO USER TOM
  GRANT SELECT ON TABLE TOOLS TO USER TOM
Database Roles – Example

- **With database roles:**

  CREATE ROLE developer

  GRANT SELECT ON TABLE SERVER TO ROLE developer
  GRANT SELECT ON TABLE CLIENT TO ROLE developer
  GRANT SELECT ON TABLE TOOLS TO ROLE developer

  GRANT ROLE developer TO USER BOB, USER ALICE

  REVOKE ROLE developer FROM USER BOB, USER ALICE

  GRANT ROLE developer TO USER TOM
Key Security Features in DB2

- Authentication
- Authorization
- Database Roles

**Trusted Contexts**
- Label-Based Access Control
- Auditing
- Encryption
- Static SQL / pureQuery
Trusted Context

- Helps solves two important security challenges

1. Application servers use of a single user ID
   - Loss of end user identity within the database server
   - Diminished user accountability
   - Over granting of privileges to a single authorization ID

2. Lack of control on when privileges are applied
   - Lack of control of when privilege can be applied can weaken overall security
   - Today all connections are treated the same
Trusted Context Example

CREATE TRUSTED CONTEXT appServer
BASED UPON CONNECTION USING SYSTEM AUTHID appServerID
ATTRIBUTES (ADDRESS ‘host-name1.dept.organization.com’,
ADDRESS ‘host-name2.dept.organization.com’
ENCRIPTION ‘HIGH’)

DEFAULT ROLE appServerRole
WITH USE FOR PUBLIC WITHOUT AUTHENTICATION,
Alice WITH AUTHENTICATION ROLE mgrRole

ENABLE
Advantages of Trusted Contexts

- **User accountability and compliance**
  - Eliminates shared user id with each person audited and accountable with their own user id
  - Allows switching of user id without having to tear down and re-establish connection

- **Improved security**
  - More control on when privileges are available to users
  - Helps alleviate concern of misusing the system authid credentials to access the DB2 server
  - Enforce the least privilege security principle
Key Security Features in DB2

- Authentication
- Authorization
- Database Roles
- Trusted Contexts
- **Label-Based Access Control**
- Auditing
- Encryption
- Static SQL / pureQuery
Label Based Access Control (LBAC)

- A flexible implementation of Mandatory Access Control (MAC)
- A security label is associated with both users and data objects

```
select serial_number, title from artifact
```
What Does LBAC Add to Table Protection?

- Does not replace the traditional discretionary access control – instead complements it at the row and/or column level

- The content of a table appears different depending on the user accessing that table

- No user has any inherent privileges to access the content of LBAC protected data even if they are DBADM!
LBAC Label Components

Set

Development  Sales  Support

Array

Highest

Top Secret  Secret  Employee  Public

Lowest

Tree

Corporate

Publishing  Software

Development

Sales  Support

Business Sales  Home Sales
Key Security Features in DB2

- Authentication
- Authorization
- Database Roles
- Trusted Contexts
- Label-Based Access Control

Auditing
- Encryption
- Static SQL / pureQuery
DB2 Audit Facility

- Very configurable, low overhead

- **Audit Policy**
  - A database object that specifies what categories of events are to be audited

- **An audit policy can be applied to:**
  - A database
  - A table
  - A trusted context
  - An authorization id (user, role, group)
  - An authority (SYSADM, SYSCTRL, DBADM, SECADM, etc.)
Audit Policies

- An audit policy is a database object that holds an instance of the audit configuration
  - Ex. `CREATE AUDIT POLICY SENSITIVE_DATA_POL CATEGORIES OBJMAINT STATUS BOTH, SECMAINT STATUS BOTH, ERROR TYPE NORMAL;`

- Can be used within a database to provide granular audit capabilities
Management of the audit log files

- **DB2 Agent**
- **Active Audit Log File**
- **Extract Command**
- **Archived Audit Log Files**
- **Archive Command**
- **DB2 Server Audit Tables**
- **Report (flat) Audit File**
- **Delimited Audit File**
- **Load**
DB2 Audit Categories

- What audit Categories can be specified?
  - AUDIT – any access or configuration of the auditing system
  - CHECKING – any authorization checks done by DB2
  - CONTEXT – the big picture, lots of miscellaneous events
  - EXECUTE – execution of SQL statements
  - OBJMAINT – create/drop of objects, some alter
  - SECMAINT – grant/revoke
  - SYSADMIN – actions only SYSADM/DBADM can do
  - VALIDATE - authentication
Category - CHECKING

- **CHECKING** events are generated anytime DB2 checks the authority of a user attempting to access or manipulate database objects or functions.

- **Notable Fields in a CHECKING record**
  
  ```
  object schema=GSTAGER;
  object name=T1;
  object type=TABLE;
  access approval reason=DBADM;
  access attempted=SELECT;
  ```
Category - CONTEXT

- CONTEXT events are generated to show the operation context when a database operation is performed.

- Provides a high level view of what requests are being submitted to the server

- Notable Events:
  - EXECUTE, EXECUTE_IMMEDIATE
  - PREPARE
  - CONNECT/ATTACH
  - GET_SNAPSHOT

- Notable Fields in a CONTEXT record
  
  ```text
  select * from t1;
  ```
Category - EXECUTE

- EXECUTE events are generated at the completion of execution of SQL statements

- Notable Events
  - STATEMENT
  - CONNECT/CONNECT RESET
  - COMMIT/ROLLBACK

- Notable Fields in an EXECUTE record
  - Statement text
  - Input data values (optional) – host variables and parameter markers
    > Does not include LOBS, XML and structured types
  - Rows read and rows modified
The EXECUTE category also contains the compilation environment at the time of statement execution:
- Contains the SCHEMA in use at the time
- “select * from t1” – which t1 was used?

**SET_COMPILATION_ENV** can change the current compilation environment to the one given as input.

Given a recreation of the data at the time of the event (from backups/rollforward), there is sufficient information logged in an EXECUTE record to replay the statement to see exactly what effect it had.
Category - SECMAINT

- SECMAINT events are generated whenever there is a change to any privileges, authorities or other security related configuration.
- Primarily to track any GRANT and REVOKE statements
- **Notable Events:**
  - GRANT, REVOKE
  - ALTER_SECURITY_POLICY
  - TRANSFER_OWNERSHIP
- **Notable Fields:**
  ```
  object schema=BOSS; object name=T1; object type=TABLE;
  grantor=BOSS;
  grantee=WORKER;
  grantee type=USER;
  privilege=SELECT;
  ```
Audit Performance

- The performance of audit is very dependent on the type of workload
  - OLTP has a higher overhead than warehousing

- The performance of audit is very dependent on the speed of the disks
  - Use fast disks
  - Isolate audit data from other database data on separate disks
  - Separate active audit log and archived audit logs
  - Use individual disks for each partition in DPF

- The performance of audit is very dependent on the number of events generated
  - Use Audit Policies to only audit what is absolutely necessary

- CONTEXT and EXECUTE are the heaviest categories, the others have little impact
Key Security Features in DB2

- Authentication
- Authorization
- Database Roles
- Trusted Contexts
- Label-Based Access Control
- Auditing
  - Encryption
  - Static SQL / pureQuery
Encryption

- Used to ensure data privacy for sensitive data
- Used in to main areas:

1. Data in Transit
   - DATA_ENCRYPT Option
   - Secure Socket Layer (SSL)

2. Data at Rest
   - Column level encryption
   - File level encryption
Encryption – Data in Transit

- Data in Transit
  - DATA_ENCRYPT Option
    - Very simple to set - AUTHENTICATION configuration parameter
    - Automatically encrypts user data during client to server communications as it travels over the network
    - Available since version V8.2
  - Secure Socket Layer (SSL)
    - Stronger cryptographic algorithm
    - CLI, CLP, and .Net Data Provider client applications and applications that use the IBM® Data Server Driver for JDBC and SQLJ (type 4 connections) support SSL.
    - Recommended Option
Why Secure Socket Layer (SSL)?

- **DATA_ENCRYPT uses DES with 56 bit encryption keys**
  - DES is no longer the industry standard encryption algorithm
  - Most auditors would require AES encryption

- **SSL is the industry standard for data in-transit encryption**
  - Provides AES encryption
  - Provides data integrity

- **Federal Information Processing Standards (FIPS) 140-2 compliance**
How DB2 9.7 SSL Works

JCC Client

JCC Client
SSL (JSSE)
TCP/IP

DB2 Server
SSL (GSKit)
TCP/IP

Digital certificates database

Signer certificate database

iKeyman tool

Encrypted communication
How DB2 9.7 SSL Works (Cont.)

- **DB2 server configuration**
  - Set the following DBM configuration parameters
    - SSL_SVR_KEYDB: Key store file
    - SSL_SVR_STASH: Stash file
    - SSL_SVCENAME: SSL port
  - Optionally, select a ciphers suite
    - SSL_CIPHERSPECS: Allowed ciphers suite
    - SSL_VERSIONS: Allowed SSL/TLS versions
  - Enable SSL communication for the instance
    - `db2set DB2COMM=SSL` or `db2set DB2COMM=SSL,TCPIP`
How DB2 9.7 SSL Works (Cont.)

- **DB2 client configuration**
  - Set connection string parameters or DB2 configuration parameters (depending on client type and language)

  - Eg. Modify a Java application to request an SSL connection

    ```java
    ... 
    properties.put("sslConnection", "true");
    System.setProperty("javax.net.ssl.trustStore", "/home/wrjaibi/client.jks");
    System.setProperty("javax.net.ssl.trustStorePassword", "xxxxxx");
    ...... 
    con = java.sql.DriverManager.getConnection(url, properties);
    ```
Encryption – Data at Rest

- **Data at Rest**
  - **Column Level Encryption**
    - ENCRYPT / DECRYPT SQL Functions
    - Uses DES
    - Available since V7.2
  - **File Level Encryption**
    - Increased Performance
    - Transparent to Application
    - Comprehensive protection (encrypt audit logs, TXN logs, etc.)
    - IBM Database Encryption Expert
    - **Recommended Option**
IBM Database Encryption Expert

- **High Performance Encryption**
  - Transparent to application, database, and storage infrastructure
  - MetaClear™ encryption allows management without viewability

- **Context Aware Access Control**

- **Operational Controls**
  - Complete audit and reporting of all data access
  - Manage privileged users

- **Centralized Enterprise Key Management**
  - Governance of key usage
  - Control who/where restores can occur
  - Track and audit key usage
Key Security Features in DB2

- Authentication
- Authorization
- Database Roles
- Trusted Contexts
- Label-Based Access Control
- Auditing
- Encryption

Static SQL / pureQuery
What is Static SQL?

- All SQL known before execution
  - Application and DBMS have both prepared for handling the SQL ahead of time

- SQL is parsed, optimized and bound into a “package” during build/deployment

- Gives database an application perspective of the running programs.
# Static SQL: Major Advantages

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dynamic SQL</th>
<th>Static SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security</strong></td>
<td>• Privileges handled at object level.</td>
<td>• Privileges are package based.</td>
</tr>
<tr>
<td></td>
<td>• All users or groups must have direct table privileges</td>
<td>• Only administrator needs direct table access. Users only need execute privilege.</td>
</tr>
<tr>
<td></td>
<td>• Security exposure, and administrative burden</td>
<td>• Prevent non-authorized SQL execution.</td>
</tr>
<tr>
<td><strong>Predictable</strong></td>
<td>• Can approach static SQL performance with expert tuning to maximize</td>
<td>• All SQL parsing, catalog access, done at BIND time.</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>dynamic SQL cache hits</td>
<td>• Fully optimized during execution</td>
</tr>
<tr>
<td></td>
<td>• Cache misses costly</td>
<td></td>
</tr>
<tr>
<td><strong>Access path</strong></td>
<td>• Unpredictable – Any prepare can get a new access path as statistics or</td>
<td>• Guaranteed – locked in at BIND time.</td>
</tr>
<tr>
<td><strong>Reliability</strong></td>
<td>host variables change</td>
<td>• All SQL available ahead of time for analysis by EXPLAIN.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Static SQL: Secure Table Privilege Example

<table>
<thead>
<tr>
<th>Dynamic SQL</th>
<th>Static SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table privileges must be granted directly to the user, groups or role.</td>
<td>Users require no specific table privileges</td>
</tr>
<tr>
<td><strong>GRANT SELECT ON TABLE PAYROLL TO ROLE HR;</strong></td>
<td><strong>GRANT SELECT ON TABLE PAYROLL TO ROLE BIND_ADM;</strong></td>
</tr>
<tr>
<td><strong>GRANT EXECUTE ON PACKAGE PAY_PKG TO ROLE HR;</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>NAME</th>
<th>POSITION</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database Admin</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**User**
pureQuery Gives You The Best of Both Worlds

- pureQuery is a high-performance data access platform that simplifies the development and management of Java apps.

- pureQuery APIs let you build a Java application with much less code vs. using JDBC or SQLJ.

- Supports both Static SQL and Dynamic SQL.

- Allows you to code to dynamic SQL, and turn on static SQL at deployment time!

```
JDBC     pureQuery     SQLJ
Dynamic SQL  Static SQL
```

Runtime Control
Summary
Summary

- Data security has become critically important due to increase of severe data breaches and regulatory compliance requirements

- DB2 provides one of the industry's most secure data server environments

- The IBM Data Server Security Blueprint get you started and helps simplify the task of securing your data server
  - Threat oriented
  - Includes current recommendations of DB2 security team
Summary

- The blueprint leverages DB2’s extensive security features. Key security features include:
  - Authentication
  - Authorization
  - Database Roles
  - Trusted Contexts
  - Label-Based Access Control
  - Auditing
  - Encryption
  - Static SQL / pureQuery

- The IBM Data Security Blueprint and accompanying whitepaper can be downloaded from: