DB2 LUW V11.1 CERTIFICATION TRAINING PART #3

Kent Collins @rcollins963
Mohankumar Saraswatipura @mohankumarsp

Agenda

1. DB2 Security
2. DB2 Native Encryption
3. Data Movement Utilities
4. Sample Questions

LDAP Authentication

Central management of user authentication and group memberships can be done using transparent LDAP authentication. This is done by configuring instances to use the operating system to authenticate users and acquire their assigned groups; the operating system authentication is done by using a LDAP server.

Transparent LDAP is enabled by setting the registry parameter DB2AUTH to OSAUTHDB or installing the LDAP security plug-ins. This allows the DB2 database manage to perform authentication of users against a LDAP directory instead of requiring that users and groups be defined on the local server.

When using a LDAP plugin module, all users associated with the database, including DB2 instance owner ID and fenced user must be defined on the LDAP server.

DB2 Security Plug-in Implementation

Use these steps when implementing DB2 security plug-in modules.

Decide what types of modules are needed. DB2 supports server, client and group plug-in modules.

	Usage
Server	When DB2 database manage to authenticate using LDAP ID and password. CONNECT and ATTACH statements use this plug-in.
Client	When user and password authenticate occurs on client system. DB2 server is configured with SRVCON_AUTH or AUTHENICATION settings of CLIENT. This is not a recommended type because it is difficult to secure.
Group	When group definitions are stored on the LDAP server.

The most common scenario would include plug-ins for Server and Group with DBM configurations for AUTHENICATION or SRVCON_AUTH set to SERVER, SERVER_ENCRYPT or DATA_ENCRYPT.

DB2 Security Plug-in Implementation

- Configure the plug-in modules by setting correct values in IBM LDAP security plug-in configuration file. (default name is IBMLDAPSecurity.ini)

  On Unix: INSTHOME/sqllib/cfg/IBMLDAPSeciruty.ini

  On Windows: %DB2PATH%\cfg\IBMLDAPSecurity.ini

- Enable the selected plug-in modules

  Server: UPDATE DBM CFG USING SRVCON_PW_PLUGIN IBMLDAPauthserver

  Client: UPDATE DBM CFG USING CLNT_PW_PLUGIN IBMLDAPauthclient

  Group: UPDATE DBM CFG USING GROUP_PLUGIN IBMLDAPgroups

  Note: DB2 instance must be stopped and started.

- Test connecting with several of the LDAP user IDs

Security Concerns Using LDAP

LDAP authentication process is called binding to the LDAP server and it is performed before accessing information in the LDAP directory. The use of access controls limits what information in the LDAP directory can be read or updated. Default access control is defined as follows:

- Database/Node entries are read everyone and update only directory administrator or creator of the object
- User profiles can be read/updated by that user or the directory administrator. Users cannot read other users profiles unless they have directory administrator authority.

Searching LDAP Servers

DB2 searches the entire LDAP server but it is possible to restrict searches of other LDAP servers in the enterprise.

If information is not found in the current LDAP server it is possible to search all LDAP servers or to restrict the search scope to the current LDAP server and local DB2 database catalog. When the search scope is set it applies to the entire enterprise.

The search scope is controlled using the DB2 database profile registry variable DB2LDAP_SEARCH_SCOPE. The default value if not set is domain which limits the search to the directory of the current LDAP server. Valid values are local, domain or global.

DB2LDAP_SEARCH_SCOPE and DB2LDAP_KEEP_CONNECTION are the only variables that can be set at the global level.

Create and Use Trusted Context

When defining a trusted relationship between an application server and a database create a trusted context object. The object consist of a authorization ID, IP address or domain name and data stream encryption. The connection process includes a check whether the connection matches the definition of the trusted context object in the database. If there is a match the connection is trusted. IPC protocol cannot be used for a trusted connection and authentication type of client does not support explicit connections

Trusted connections have additional capabilities depending on whether they are implicit or explicit. The initiator of a explicit connection can switch current user ID on the connection with or without authentication and can acquire additional privileges via role inheritance. Implicit trusted connection is any trusted connection that is not explicitly created. An implicit trusted connection cannot perform a user switch

Create and Use Trusted Context

- ✓ CREATE TRUSTED CONTEXT APP1 BASED UPON CONNECTION USING SYSTEM AUTHID APP1USR
  ATTRIBUTES(ADDRESS '135.1.65.2')
  DEFAULT ROLE USERROLE
  ENABLE ;
- ✓ CONNECT TO MYDB USER APP2USR USING XXX;

SECADM is required to create, alter or drop a trusted context.

The connection requests gets a SQLSTATE 01679 warning to indicate the connection was established but is not trusted.

- ✓ CONNECT TO MYDB USER APP1USR USING XIS XX;

The connection request is successful and trusted because of the APP1 context. All privileges defined to role userRole are given to user APP1USR. User APP1USR may only have access to role userRole when using this context so that means only from server '135.1.65.2'.

Create Explicit Connection and Switching User

- ✓ CREATE TRUSTED CONTEXT SW1 BASED UPON CONNECTION USING SYSTEM AUTHID SW1USR

ATTRIBUTES(ADDRESS '135.12.15.2')
WITH USE FOR SW2USR WITHOUT AUTHENTICATION,
SW3USR WITH AUTHENTICATION
Enable ;

Java Example:
Get connection using API getDB2TrustedPooledConnection
Switch with or without authentication using
getDB2Connection or reuseDB2Connection

SW2USR can switch without providing any authentication but SW3USR will be prevented from switching unless authentication is successful.

Information About Trusted Context

Information about trusted context objects can be found using SYSCAT.CONTEXTS and SYSCAT.CONTEXTATTRIBUTES catalog views.
Trusted context privileges acquired through a role are effective only for dynamic DML operations. The scalar function VERIFY_TRUSTED_CONTEXT_ROLE_FOR_USER in schema SYSIBM can be used to verify the role status of a SESSION_USER.

✓
```
CREATE PERMISSION DIVISION_ACCESS
ON TRAIN_STATION FOR ROWS
WHERE VERIFY_TRUSTED_CONTEXT_ROLE_FOR_USER
(SESSION_USER,'YARD_MGR') = 1
AND BRANCH = (SELECT HOME_BRANCH FROM INTERNAL_INFO
WHERE EMP_ID = USER)
ENFORCED FOR ALL ACCESS ENABLE;
```

The name of the trusted context object used to establish a trusted connection is stored in the DB2 global variable SYSIBM.TRUSTED_CONTEXT.

Restrict Access to Sensitive Data

DB2 provides a number of ways to restrict user access to sensitive data.

Views - Views are easy to setup and manage. They also provide a layer of separation between the data structure and code even when access restriction is not the goal.

LBAC - is configured by a security administrator and provides custom row and column level access through the definition of security labels. A security policy describes the criteria that will decide who has access to what data. Each policy can have one or more labels.

Introduced in V10.1 RCAC - is sometime referred to as fine-grained access control or FGAC. No database user is exempted. Only users with SECADM authority can manage RCAC. The enforcement is data-centric. No application changes are required to implement RCAC.

Views Used to Restrict Access

Views can be used to limit the columns and rows accessed by users or groups. Functions can be used to redact some or all characters in a column. For example only the last four numbers of employee social security numbers could be returned for a specific group of users while others see the values completely.

Most database organizations have significant understanding of views because they have been supported objects in most database software for years,

```
✓  CREATE VIEW BYDIV AS
SELECT NAME, LOCATION FROM DIV, ORG
WHERE DIV.DEPT=ORG.DEPTNUMB
  AND ORG.EMPID = SESSION_USER ;
```

RCAC Access to Sensitive Data

RCAC introduces us to a different direction in data access. Instead of being based on "what" is asked, access is determined based on "who" is asking "what". Result sets of the same query will change based on the context in which the query was asked.

Statements used to manage RCAC

- ❑ Create Permission
- ❑ Create Mask
- ❑ Create Table…. Security policy
- ❑ Create Trigger
- ❑ Create Function
- ❑ Rename
- ❑ Grant/Revoke/Comment
- ❑ Alter Permission…. enable/disable
- ❑ Alter Mask ….enable/disable
- ❑ Alter Table ….activate column access control
- ❑ Alter Trigger
- ❑ Alter Function

RCAC Access to Sensitive Data

Row Controls

Row Permission → Row Access Control Rule → SQL Search Condition

Column Masks

Column Mask → Column Access Control Rule → SQL Case Expression

- No users are exempted
- Does not matter how data is accessed
- No application changes are required

VERIFY_ROLE_FOR_USER
VERIFY_GROUP_FOR_USER
VERIFY_TRUSTED_CONTEXT_ROLE_FOR_USER

Using Roles

Roles help simplify administration and management of database privileges. Roles are database objects which group together privileges making it easy to classify users. They are different from groups because they are internal to the database.

- Roles can mirror organizations
- Users get role memberships that reflect their job responsibilities. As users change jobs there role memberships can easily change.
- Privileges are assigned to roles instead of users or groups which are controlled by external services.
- When privileges within a role are changed every user with membership gets it.
- Roles cannot be enabled or disabled explicitly.
- Security Administrator can delegate management of roles to others.

Role Example

Create a ROLE for the Developer Class. Initially give the Developers in the organization access to the database with read access to the DB2 system catalog tables and views with the exception of all tables and views containing authorization information. Give all developers DB2/JSON access.

Revoking privileges from roles can cause objects to become invalid or inoperative.

```
CREATE ROLE DEVELOPER @
CREATE ROLE READ_CATLG @

select 'grant select on table ',TRIM(tabschema)||'.'||TRIM(tabname),' to role READ_CATLG @'
from SYSCAT.TABLES where type = 'V' and tabschema = 'SYSCAT'
  and not tabname like '%AUTH' with ur @

Grant insert, update, delete, select on table systools.sysjson_index to role developer @

GRANT ROLE "READ_CATLG" TO ROLE "DEVELOPER"@

select 'grant select on table ',TRIM(tabschema)||'.'||TRIM(tabname),' to role READ_CATLG @'
from SYSCAT.TABLES where type = 'T' and tabschema = 'SYSIBM' and not tabname like '%AUTH'
with ur @
```

DB2 Native Encryption

Databases and backups can be encrypted. It requires no changes to hardware, software, applications or schemas. Native Encryption can be used to manage and secure keys.

To use Native Encryption you need GSKIT installed and the proper license. DB2 installer installs GSKIT locally in sqllib/libxx/gskit. If GSKIT is installed in a central Unix location like /usr/lib make sure this version and the one installed by DB2 are the same.

- ❑ Create local keystore
- ❑ Configure Instance
- ❑ Generating a Master Key

Central KeyStore set KEYSTORE_TYPE = 'KMIP'
Local KeyStore set KEYSTORE_TYPE = 'PKCS12'

```
gsk8capicmd_64 -keydb -create -db "<file-name>"
-pw "<password>" -type pkcs12 -stash
```

```
update dbm cfg using keystore_location <file-name>
update dbm cfg using keystore_type pkcs12
```

```
gsk8capicmd_64 -secretkey -create -db "<keystore-file-
name>" [-pw "<password>" | -stashed ] -label "<label>" -
size "<key-length-in-bytes>"
```

DB2 Native Encryption

CREATE DATABASE <DBNAME> ENCRYPT — Defaults
CREATE DATABASE <DBNAME> ENCRYPT — Custom
cipher aes key length <length> master key label <label>

Encrypting a current Database

- Create a keystore
- Configure Instance
- Backup DB
- Drop un-encrypted DB
- Restore db <dbname> into <new dbname> encrypt

A stash file should be used to allow normal instance operations like db2start. Without a stash file the password to the keystore must be entered on the db2start command or when prompted.

Good Reason to never let apps connect to physical DB name; Use aliases

DB2 Native Encryption – Best Practices

- ✓ Use good password
- ✓ Secure files with native OS protection
- ✓ Rotate Master Keys using ADMIN_ROTATE_MASTER_KEY
- ✓ Change keystore passwords regularly gsk8capicmd_64 <changepw>
- ✓ Backup files regularly and everytime keystore changes
- ✓ Never delete keys from keystore

You can verify the encryption status of a database using ADMIN_GET_ENCRYPTION_INFO.
Also by db2 “get db cfg for <dbname>” | grep –i encrypted

For Encryption Options run this

```
SELECT * FROM
TABLE("SYSPROC"."ADMIN_GET_ENCRYPTION_INFO"()) AS
UDF FOR FETCH ONLY;
```

Encrypting Data On the Wire

Encrypt data between client and server using DATA_ENCRYPT authentication type or Secure Sockets Layer (SSL) also sometimes referred to as TLS.

- Make sure GSKIT is in LIBPATH
- Connection Concentrator is off max_connections > max_coordagents
- Create Key Database for certificates gskcapicmd command

- Add certificate

```
gsk8capicmd_64 -keydb -create -db "mydb2server.kdb" -pw "myDB2ServerPassw0rdpw0" -stash
```

- Extract certificate to a file and place on client

```
gsk8capicmd_64 -cert -create -db "mydb2server.kdb" -pw "myDB2ServerPassw0rdpw0" -label
"myselfsigned" -dn "CN=myhost.mycompany.com,O=myOrganization,
OU=myOrganizationUnit,L=myLocation,ST=ON,C=CA" -size 2048 -sigalg SHA256_WITH_RSA
```

```
gsk8capicmd_64 -cert -extract -db "mydb2server.kdb" -pw "myDB2ServerPassw0rdpw0" -
label "myselfsigned" -target "mydbserver.arm" -format ascii -fips
```

Encrypting Data On the Wire

- DB2COMM set to (TCPIP,SSL)
- Set SSL_SVR_KEYDB to the fully qualified path of key database file REQUIRED
- Set SSL_SVR_STASH to the full qualified path of the stash file REQUIRED
- Set SSL_SVR_LABEL to the digital certificate of the server
- Set SSL_SVCENAME to the SSL Port. Not the same as SVCENAME REQUIRED
- Stop and Start Instance

DB2 Data Maintenance Utilities

Some of the data maintenance utilities available within DB2 are:

- REORGCHK
- REORG
- REBIND
- RUNSTATS
- FLUSH PACKAGE CACHE
- ADMIN_CMD

The REORGCHK Utility

The way in which data is physically distributed across table space containers can have a significant impact on how applications that access the data perform. And the way data is distributed is controlled primarily by the insert, update, and delete operations that are executed against tables.

When REORGCHK is executed, it generates statistics on a database and analyzes those statistics to determine whether one or more tables need to be reorganized (which will cause any existing internal gaps to be removed).

You invoke the REORGCHK utility by executing the REORGCHK command, The basic syntax for this command is:

REORGCHK <UPDATE STATISTICS | CURRENT STATISTICS>

<ON TABLE USER | ON SCHEMA [*SchemaName*] |

ON TABLE [USER | SYSTEM | ALL | [*TableName*]>

The REORG Utility

Upon careful evaluation of the REORGCHK utility's output, you may discover that one or more tables or indexes must be reorganized. If that is the case, you can reorganize them by using DB2's REORG utility. The REORG utility eliminates gaps in table space containers by retrieving the data stored in a table and one or more of its associated indexes and rewriting it onto defragmented, physically contiguous pages in storage.

The basic syntax for this command is:

REORG TABLE [*TableName*] <INDEX [*IndexName*]>

<ALLOW READ ACCESS | ALLOW NO ACCESS>

<USE [*TmpTSName*]> <INDEXSCAN>

<LONGLOBDATA <USE [*LongTSName*]>> <KEEPDICTIONARY | RESETDICTIONARY>

Online REORG

Online REORG performs reorganization allowing full access to data in the table. Starting Cancun Release 10.5.0.4 online table reorganization is supported in DB2 pureScale environment.

During an online table REORG operation, data is not copied to a temporary table space like offline reorg; instead, rows are moved within the existing table object to reestablish clustering, reclaim free space, and eliminate overflow rows.

Phases of online REORG operation:

- SELECT n pages
- Vacate the range
- Fill the range
- Truncate the table

REORG Command Examples

REORG operation to rebuilding the dictionary allowing read only workload:

- ✓ REORG TABLE employee ALLOW READ ACCESS RESETDICTIONARY;

Online REORG operation options:

- ✓ REORG TABLE employee INPLACE CLEANUP OVERFLOWS;
- ✓ REORG TABLE employee INPLACE FULL;
- ✓ REORG TABLE employee INPLACE FULL TRUNCATE TABLE;

The RUNSTATS Utility

The RUNSTATS utility collects statistics on tables, indexes, and statistical views. The DB2 Optimizer uses such information when deciding on the best access plan to use to obtain data in response to a query.

The statistics update should be done when

- A table has been modified considerably i.e., 10% – 20% of the base data
- A table has been reorganized (REORG)
- When a table has been enabled for compression
- A new index is created for a table
- Before binding application programs whose performance is critical
- When you want to compare current and older statistics
- When a pre-fetch size is changed

The RUNSTATS Command

RUNSTATS ON TABLE [*TableName*]
FOR <<SAMPLED> DETAILED>
[INDEXES | INDEX] [[*IndexName*,...] | ALL]
<EXCLUDING XML COLUMNS> <ALLOW READ ACCESS | ALLOW WRITE ACCESS> <SET PROFILE NONE | SET PROFILE <ONLY> | UPDATE PROFILE <ONLY>>
<UTIL_IMPACT_PRIORITY [*Priority*]>

- ✓ RUNSTATS ON TABLE payroll.employee FOR INDEXES ALL ALLOW READ ACCESS
- ✓ RUNSTATS ON TABLE payroll.department ON ALL COLUMNS WITH DISTRIBUTION DEFAULT

The RUNSTATS Command (continued)

- RUNSTATS operation can be throttled using UTIL_IMPACT_PRIORITY
- RUNSTATS can be run on one or more individual columns
- ✓ RUNSTATS ON TABLE employee ON COLUMNS ((ID, NAME));
- Sampled RUNSTATS on TABLE and INDEX
- ✓ RUNSTATS ON TABLE payroll.employee AND SAMPLED DETAILED INDEXES payroll.idx1_employee TABLESAMPLE SYSTEM(30) INDEXSAMPLE SYSTEM (30);

The Rebind Utility

The REBIND utility allows a user to recreate a package stored in the database without needing the original bind file. The basic syntax for REBIND is:

REBIND <PACKAGE> [*PackageName*] <VERSION [*Version*]>
RESOLVE [ANY | CONSERVATIVE]
<REOPT NONE | REOPT ONCE | REOPT ALWAYS>

to rebind package EMP_MGMT, you execute a REBIND command:

- ✓ REBIND PACKAGE emp_mgmt

Flushing the Package Cache

The FLUSH PACKAGE CACHE SQL statement provides database administrators with the ability to remove cached dynamic SQL statement packages from memory (the package cache) by invalidating them.

The invalidation of a cached dynamic SQL statement package has no effect on current users of the statement; however, once you invalidate a package, any new requests for the statement with which the invalidated package was associated will cause the DB2 Optimizer to reprocess the statement, which in turn will produce a new cached package.

The basic syntax for the FLUSH PACKAGE CACHE statement is:

- ✓ FLUSH PACKAGE CACHE DYNAMIC

A Word About ADMIN_CMD()

The ADMIN_CMD () stored procedure is a special system built-in stored procedure that allows applications to run select administrative commands by using the CALL SQL statement. One can invoke the following commands by using the ADMIN_CMD () stored procedure:

ADD CONTACT
ADD CONTACTGROUP
AUTOCONFIGURE
BACKUP (online only)
DESCRIBE
DROP CONTACT
DROP CONTACTGROUP
EXPORT
FORCE APPLICATION
IMPORT
INITIALIZE TAPE
LOAD
PRUNE HISTORY/LOGFILE
QUIESCE DATABASE
QUIESCE TABLESPACES FOR TABLE
REDISTRIBUTE

A Word About ADMIN_CMD()

REORG INDEXES/TABLE
RESET ALERT CONFIGURATION
RESET DB CFG
RESET DBM CFG
REWIND TAPE
RUNSTATS
SET TAPE POSITION
UNQUIESCE DATABASE
UPDATE ALERT CONFIGURATION
UPDATE CONTACT
UPDATE CONTACTGROUP
UPDATE DB CFG
UPDATE HISTORY
UPDATE DBM CFG
UPDATE HEALTH NOTIFICATION CONTACT LIST
GET STMM TUNING
UPDATE STMM TUNING

- ✓ CALL ADMIN_CMD ('REORG TABLE employee INPLACE')
- ✓ CALL ADMIN_CMD ('UPDATE DB CFG FOR sample USING LOGSECOND 30')

Thank you!

Robert (Kent) Collins

@rollins963

Kent, founder of Shiloh Consulting, Inc., is currently a Data Solutions Architect with BNSF Railway. He is an IBM Champion (2010–2017) and a frequent speaker at the DB2Night Show and IDUG and IOD (IBM Insight) conferences. Kent has worked continually with DB2 from its introduction to the market in 1984, amassing a wealth of knowledge and experience. He graduated from the University of Texas at Dallas with majors in mathematics and computer science. He is an IBM Certified Solutions Expert and also holds certifications in DB2, AIX, Linux, .NET, Oracle, SQL Server, Windows, and z/OS. Kent is proficient in many programming languages and, as a Java Architect, specializes in Enterprise HA/Performance systems. He lives in Dallas with his wife, Vicki; together, they have three children and one grandchild.

Mohankumar Saraswatipura

@mohankumarsp

Mohan works as a Database Solutions Architect focusing on IBM DB2, Linux, UNIX, and Windows solutions. Prior to his current position, he worked as a Database Solutions Architect at Reckitt Benckiser Group, plc (UK) focusing on IBM Smart Analytics System 5600, Siebel, SQL Server, and SAP HANA solutions. He is an IBM Champion (2010–2017) and a *DB2's Got Talent* 2013 winner. Mohan has written dozens of technical papers for IBM developerWorks and *IBM Data* Magazine. He is an IBM Certified DB2 Advanced Database Administrator, DB2 Application Developer, and DB2 Problem Determination Master. Mohan holds a Master's of Technology (M Tech) degree in computer science and an Executive MBA (IT).

Please contact us if you have any questions @ www.EnterpriseDB2.com