NOTES: This session will discuss features of DB2 Native Encryption which became available in DB2 in DB2 10.5 FP5. The need for data encryption and current threat environment will be discussed. How DB2 Native Encryption meets these threats and an overview of the DB2 Native Encryption architecture will be presented along with key implementation elements. Finally, securing Client and Server network transmissions will be discussed using SSL/TLS. When used together, these capabilities provide you with a secure database environment.
Objectives

- Threat Environment
- Security Standards
- Overview of DB2 Native Encryption
- IBM Global Security Kit
- Keystore, Database Encryption Key and Master Encryption Key
- Backup and Restore considerations
  - DB CFG Parameters
  - Moving database to a different server
- Configuring SSL/TLS
  - Client
  - Server
- Summary
Threat Environment (Data and Network)

- **SECURITY** is a DBA’s Inherent Responsibility!
  - After all it is all about the DATA and the security and integrity of the DATA has always been a DBAs prime responsibility!!
  - Get involved!
- Today’s information system threats are far and wide
- Denial of Service Attacks (DOS)
- Distributed Denial of Service Attacks (DDOS)
- Brute Force
- Social Engineering (#1 way systems are hacked*)
  - Phishing
  - Malware contained in email attachments*(Target)
- Inherent (software flaws)
- Global
Security Vulnerabilities

- Database Defaults
- Default Userids
- Failure to follow least privilege
- Contractors and vendors with elevated privileges
- Outsourcers
- Mechanical Systems (HVAC) access can be exploited (Target Inc)
Standards and Regulations

- Payment Card Industry Data Security Standard (PCI DSS)
- Health Insurance Portability and Accountability Act (HIPAA)
- NIST
- FIPS
- Federal Privacy Act
- Graham-Leach-Bliley Act (GLBA)
- Data Protection Act (DPA) in UK
Standards and Regulations

- Health Information Technology for Economic and Clinical Health Act (HITECH)
- International Standards Organization (ISO)
- Import/Export Controls
- Trans-border data flow
- Data Breaches
- Privacy
Standards and Regulations

- Public Key Cryptography Standard #12 (PKCS#12) – DB2 10.5
- Secure Local Key management with Native Encryption
- Advanced Encryption Standard
DB2 10.5 FP5 Compliance

- Federal Information Processing Standards (FIPS)
  - Publicly announced US Government standards
  - ANSI
  - IEEE
  - ISO
- PCI DSS
- NIST
- AES
- TLS
NOTES: This slide provides an overview of the operation and components involved in implementing DB2 Native Encryption. OS file permissions are used to protect the keystore along with a stashed (or not stashed) password that is encrypted. There is one keystore per instance. The IBM GSKit is used to setup the keystore and password. DB2 generates the Database encryption key (DEK) based on the password and an encryption algorithm. The DEK is what is used to encrypt the database. The DEK is stored in the database and database backup. The Master Encryption key is stored outside the database. This is referred to as envelope encryption in the industry. The DBM CFG keystore_type and keystore_location are used to specify information about the keystore. If the password is stashed DB2 can start without user intervention. If not stashed, it must be provided as an parameter on the db2start command via the OPEN KEYSTORE option. The DB2 Native encryption is implemented deep within the DB2 code in the DB2 kernel. The kernel encrypts the data that is written to disk, thus the lowest possible overhead is realized. For AES encryption, DB2 can take advantage of processor improvements. To create an encrypted database, specify ENCRYPT on the CREATE DATABASE command. The encrlib and encropts DB CFG parameters will be automatically set for a newly created database. To encrypt and existing database, create the keystore and password, and update the location and type in the DBM CFG. Backup the database to be encrypted. Drop the database after it is backed up. Restore the previously backed up database and specify ENCRYPT on the
RESTORE DATABASE command. The restored database will be encrypted using the default settings. To determine if a database is encrypted, DB2 provides stored procedures and table functions described later. When both the encrlib and encropts DB CFG parameters are set, the backup will be automatically encrypted. It is possible to override these settings. DB2 Native Encryption does not require any table or schema changes.
To check if an instance has been created with Native Encryption, check the following DBM CFG

Parameters: Keystore type \( (\text{KEYSTORE\_TYPE}) = \text{NONE} \)
Keystore location \( (\text{KEYSTORE\_LOCATION}) = \)
DB2 Native Encryption Overview

- DB2 Native Encryption is **transparent** to applications and database schemas
- DB2 uses symmetric encryption for encrypting the database and backups
- Both Advanced Encryption Standard (AES) and 3DES are supported
  - AES with 256 bit key length is the default
- Can be seen via DB CFG parameter -- Encryption Options for Backup (ENCROPTS) = CIIPHER=AES:MODE=CBC:KEY LENGTH=256
DB2 Native Encryption Overview

- DB2 Native Encryption is implemented by the DB2 Kernel which encrypts the data before calling the underlying file system to write the data to disk
  - Via encryption algorithm and encryption key
- Current data is protected, any new tablespaces or any that are added in the future
- DB2 Native encryption exploits processor technology improvements
  - Intel AES-NI
  - Automatically detected and exploited
  - POWER8 Hardware Assisted Encryption

NOTES:
NOTES: A new master key is automatically added when you create an encrypted database without specifying the MASTER KEY LABEL option on the CREATE DATABASE command. The database uses this master key by default, but you can optionally add a different master key.

Encrypted master keys are stored in a PKCS#12-compliant keystore, which is a storage object for encryption keys that exists at the operating system level. In partitioned database environments or DB2 pureScale® environments, the keystore location must be accessible to all members.
Notes: Encrypting Data “in-place” is not supported, backup a database and restore backup into a new database with encryption enabled. To verify that the Global Security kit is installed, type PATH from a Windows command line. On UNIX or Linux the library needs to be included in the appropriate environmental variable, “Include $INSTHOME/sqlib/lib64/gskit in the LD_LIBRARY_PATH, LIBPATH, or SHLIB_PATH environment variable.”
NOTES: If you try and create an encrypted database without first creating a keystore, the following error will be

Returned: C:\Program Files\IBM\gsk8\bin>db2 create db gts1 encrypt

SQL1728N The command or operation failed because the keystore could not be accessed. Reason code "1".
Key Management (Another story)

- The keystore password can be automatically stashed to a file that automatically provides the password when required !!!!!
- CAUTION !!!!!!
- If not stored, you cannot start the database without providing the password (Key Management Required) (DB2 will start but database will not activate)
- DB2 11.1 supports Key Management Interoperability Management Protocol (KMIP 1.1)
- Initially validated on IBM Security Key Lifecycle Manager (ISKLM)
NOTES: This slide provides an overview of the operation and components involved in implementing DB2 Native Encryption.
NOTES: The keystore is required with one keystore per DB2 instance. Using the IBM Global Security kit, create the keystore by running the gsk8capicmd_64 command with the parameters specified.

Implementation

- Step 1. Create the keystore and password
- From a DB2 command line create the keystore and stash the password as follows:

  gsk8capicmd -keydb -create -db gtskeystore.p12
  -pw Str0ngPassw0rd –strong -type pkcs12 –stash;

```
C:\Program Files\IBM\gsk8\bin>gsk8capicmd_64 -keydb -create -db gtskeystore.p12
  -pw Str0ngPassw0rd -strong -type pkcs12 -stash
```
Keystore Setup and Usage

- The MK is stored outside the database in a Public-Key Cryptography Standards (PKCS#12) compliant keystore.
- Keystore is protected by file permissions and password protection.
- Make sure that only the DB2 instance owner has read/write access to the keystore.
- Make sure you create the keystore using the password option.
- The contents of the keystore (master key(s)) are encrypted from using a symmetric key derived from the password using a hashing algorithm.
- Without the password, the content of the keystore cannot be decrypted.
Implementation

- Step 1. Create the keystore and password
- Step 2. Update the DBM CFG parameters KEystore_TYPE and KEystore_LOCATION in same update command
- Step 3. Specify the type of key and location stored
- Step 4. Create new database and specify ENCRYPT on the CREATE DB command
- Check the DB CFG parameters of the new encrypted database

Encryption Library for Backup
(ENCRLIB) = db2encr.dll
Encryption Options for Backup (ENCROPTS) =
CIPHER=AES:MODE=CBC:K
EY LENGTH=256
Encrypted database = YES

NOTES: C:\Program Files\IBM\gsk8\bin>db2 update dbm cfg using keystore_type pkcs12 keys
tore_location c:\Program Files\IBM\gsk8\bin\gtskeystore.p12
SQL0104N An unexpected token "Files\IBM\gsk8\bin\gtskeystore.p12" was found
following "<identifier>". Expected tokens may include: "AGENTPRI".
SQLSTATE=42601
C:\Program Files\IBM\gsk8\bin>db2 update dbm cfg using keystore_type pkcs12 keys
tore_location 'c:\Program Files\IBM\gsk8\bin\gtskeystore.p12'
DB20000I The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
Keystore type (KEYSTORE_TYPE) = PKCS12
Keystore location (KEYSTORE_LOCATION) = c:\Program
Files\IBM\gsk8\bin\gtskeystore.p12
C:\Program Files\IBM\gsk8\bin>db2 create db gts1 encrypt
DB20000I  The CREATE DATABASE command completed successfully.
Implementation

- To Encrypt an existing database
  1. Backup the existing database
  2. Drop the existing database
  3. Restore the database you backed up and specify ENCRYPT on the RESORE DATABASE command

Example: “RESTORE DB SAMPLE ENCRYPT”

```
C:\Program Files\IB\SOL\BIN\db2 backup db sample
Backup successful. The timestamp for this backup image is : 2016042130615

C:\Program Files\IB\SOL\BIN\db2 restore db sample encrypt SOL\BIN\ The RESTORE command failed because the target database already exists and encryption options were specified. SOL\BIN\STATE=00000

C:\Program Files\IB\SOL\BIN\db2 drop db sample
DB20000I The DROP DATABASE command completed successfully.

C:\Program Files\IB\SOL\BIN\db2 restore db sample encrypt SOL\BIN\ The RESTORE DATABASE command completed successfully.
```
Database Backups

- After a database is created as encrypted, the aforementioned DB CFG parameters are automatically set and all subsequent database backups are encrypted using the DB CFG options and it is not necessary to specify ENCRYPT on the BACKUP Command.
Rotating the Master Key

- Rotating the Master Key is good practice and can be done using the DB2 provided ADMIN ROTATE MASTER KEY stored procedure.
- `db2 "CALL SYSPROC.ADMIN_ROTATE_MASTER_KEY (NULL)"
  - Output from the proc: Value of output parameters
  - Parameter Name: LABEL
  - Parameter Value: DB2_SYSGEN_DB2_GTS1_2016-04-23-13.51.43
  - Return Status = 0
- Rotation of the master key should be in accordance with your security policy!
- Rotation of the master key is logged to the db2diag.log file (see below in notes)

NOTES: Requires EXECUTE authority for the routine. DB2 native encryption also allows you to rotate your database MK to comply with your corporate security policies. You rotate your database MK by calling the new ADMIN ROTATE MASTER KEY procedure. The procedure decrypts your database DEK with the old MK and then re-encrypts it with the new MK. You have 2 options when calling the ADMIN ROTATE MASTER KEY procedure. You can either provide a label for the desired new MK or use the default. When using the default, DB2 automatically generates a new master key and adds it to the keystore on your behalf. Then, it rotates the current database MK to this newly generated MK.

2016-04-24-19.23.55.582000-240 I5874F579 LEVEL: Event
PID : 9584 TID : 3564 PROC : db2syscs.exe
INSTANCE: DB2 NODE : 000 DB : GTS1
APPHDL : 0-8464 APPID: *LOCAL.DB2.160424232247
AUTHID : PGUNNING HOSTNAME: GTS-LT1A
EDUID : 3564 EDUNAME: db2agent (GTS1) 0
FUNCTION: DB2 UDB, bsu security, sqlexRotateMasterKey, probe:1052
DATA #1 : String, 36 bytes
Key Rotation successful using label:
DATA #2 : String, 39 bytes
DB2_SYSGEN_DB2_GTS1_2016-04-24-19.23.54
RESTORE Considerations for An Encrypted Database

- By default RESTORE will use the previously configured settings
- To restore an encrypted database to a non-encrypted database backup and then drop the existing database and the restore the database using the NO ENCRYPT option on the RESTORE command

NOTES:
Copying and RESTORING an Encrypted Database from Server 1 to Server 2

- Backup the encrypted database on Server 1
- Securely copy the database to Server 2
- Securely copy the keystore and associated stash file from Server 1 to Server 2
- Update the DBM CFG KEYSTORE_LOCATION parameter on the Server 2 instance where the database will be restored
- Restore the encrypted database from Server 1 and specify the ENCRYPT option on the RESORE Database command

NOTES: If server 1 and server 2 do not implement the same security requirements, the keystore cannot be copied from server 1 to server 2. In this case, use a new master key for the backup image and securely send that key to the administrator for server 2.
Support Considerations

- When working with ENCRIPTED databases, Database logs and dump files are encrypted
- IBM Support will provide necessary instructions and tools for providing needed data to IBM
DB2 Native Encryption and HADR

- HADR and DB2 Native Encryption is fully supported
- STANDBY Instance requires a keystore and same encryption key as PRIMARY database
- Copy keystore to the STANDBY server or export the master key and import to STANDBY
- Update STANDBY Instance with keystore location and type
- Copy backup of PRIMARY Encrypted database to the STANDBY Server
- RESTORE the copied database to the STANDBY INSTANCE and specify the ENCRYPT option on the RESTORE command
DB2 Native Encryption and HADR

- Configure HADR on the STANDBY Database and Start HADR and ACTIVATE the STANDBY Database (same HADR steps as normally used on an unencrypted database)
Managing DB2 Native Encryption

- DB2 provided Stored Procedures and table functions
- Check encryption status of a database
  - `SELECT * FROM TABLE (SYSPROC.ADMIN_GET_ENCRYPTION_INFO())`
- Rotate the Master Key
  - `CALL SYSPROC.ADMIN.Rotate_master_key('newlabel')`
- Call the SYSPROC.ADMIN_GET_ENCRYPTION_INFO stored procedure to determine what master key label is being used along with other encryption options
  - "SELECT OBJECT_NAME, OBJECT_TYPE, ALGORITHM, ALGORITHM_MODE, KEY_LENGTH, MASTER_KEY_LABEL FROM TABLE(SYSPROC.ADMIN_GET_ENCRYPTION_INFO())"
DB2 Utility and Tool Support for DB2 Native Encryption

- Db2cklog
- Db2flsn
- db2LogsForRfwd
- Db2ckbkup
- Db2dart
- Db2adult
- Tools and utilities will use the keystore specified in the DBM CFG
SSL/TLS Setup and Configuration
Secure Socket Layer (SSL) Superseded by Transport Layer Security (TLS 1.2)

- SSL has been superseded by TLS but is often referred to as SSL/TLS
- DB2 provides for support of TLS 1.2
- TLS encrypts the connection between DB2 clients and the database server
  - Encrypts data over the wire whereas DB2 Native Encryption encrypts data at rest
  - Both are required for a secure solution
- Enabled at the DB2 Instance level
Configuring DB2 and Clients for SSL/TLS Support

- The IBM Data Server Driver for JDBC and SQLJ provides support for the SSL/TLS through the Java Secure Socket Extension (JSSE)
- 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/TLS
- The SSL communication will always be in Federal Information Process Standards (FIPS) mode
Setup the DB2 Instance for SSL/TLS Support

- Using the IBM Global Security Kit to create a key database and setup your digital certificates and a stash file
  
gsk8capicmd_64 -keydb -create -db "mygtsserver.kdb" -pw "myGtsServerPassw0rdpw0" -stash

- The key database is automatically populated with signer certificates from a few certificate authorities (CAs), such as Verisign
Setup the DB2 Instance for SSL/TLS Support

- Add a certificate to the key database
  - Server sends this to clients during SSL/TLS handshake to provide authentication for the server
- Use the GSKCapiCmd to create a new certificate request and send to a CA to be signed or create a self-signed certificate
- Create a self-signed certificate as follows:
  gsk8capicmd_64 -cert -create -db "myGTSServer.kdb" -pw "myGTSServerPassw0rdpw0"
  -label "myselfsigned" -dn "CN=www.gts1consulting.com,O=GTS1,
  OU=Consulting,L=Reading,ST=PA,C=US"
Setup the DB2 Instance for SSL/TLS Support

- Extract the certificate you just created to a file so it can be distributed to computers running clients that will establish SSL connections to the DB2 server

```
gsk8capicmd_64 -cert -extract -db "myGTSServer.kdb" -pw "myGTSServerPassw0rdp0" -label "myselfsigned" -target "myGTSServer.arm" -format ascii -fips
```
Setup the DB2 Instance for SSL/TLS Support

- Update the DBM CFG SSL_SVR_KEYDB parameter to the fully qualified path of the key database file
- Update the DBM CFG SSL_SVR_STASH parameter to the fully qualified path of the stash file
- Update the DBM CFG SSL_SVR_LABEL to the label of the digital certificate of the server
- Set the DBM CFG SSL_SVCENAME parameter to port that the DB2 instance will listen on for SSL connections
  - If TCP/IP and SSL are both enabled (DB2COMM registry variable set to ‘TCIP,SSL’, the SSL_SVCENAME must be set to a different port than the port to which SVCENAME is set
- To specify which cipher suites you want to use set the DBM CFG SSL_CIPHERSPECS to those cipher suites or leave unset (NULL) to allow the strongest available cipher suite to be used that is supported by both client and server
Setup the DB2 Instance for SSL/TLS Support

- Add the SSL value to the DB2COMM registry variable to enable SSL or support or set to both SSL and TCPIP if multiple protocol support

- Stop and Start the DB2 instance for all changes to take effect

- NOTE that SSL is supported for clients and the PRIMARY database but it is not supported between the Primary and Standby databases NOTE: DB2 11.1 will support SSL between primary and standby DB2 11.1
NOTES: Additionally the Java Runtime Environment (JRE) has to be configured. The JRE uses a Java security provider. The IBM JSSE provider is automatically installed with the IBM SDK for JAVA. I’ve included the link to the DB2 Knowledge Center page containing the details to complete the JRE configuration.

https://www.ibm.com/support/knowledgecenter/mobile/#!/SSEP00G_10.5.0/com.ibm.db2.luw.apdv.java.doc/src/tpc/imjcc_t0054066.html
NOTES: From DB2 V10.5 FP5 onwards, the SSLClientKeystoredb and SSLClientKeystash keywords are not needed in the connection string, db2cli.ini file, FileDSN, or db2dsdriver.cfg file. If you have not set or passed values for the SSLClientKeystoredb and SSLClientKeystash keywords, the CLI/ODBC client driver will create a default key database internally during the first SSL connection. The Client driver will call GSKit API's to create a key database populated with the default root certificates. If the application has passed the signer certificate of the server (*.arm file) using the SSLServerCertificate keyword, the client driver will add this certificate to this default key database and proceed for the SSL connection. In this case, the application needs to use the Security=SSL and SSLServerCertificate=<certificate file name> as in below connection string.

"Database=sampledb;Protocol=tcpip;Hostname=myhost;Servicename=50001; Security=ssl;SSLServerCertificate=server.arm;"
Configuring SSL Support in non-Java Clients

- If you are using the IBM Data Server Driver for ODBC and CLI, you use connection string parameters
- Call SQLDriverConnect with a connection string that contains the SECURITY=SSL keyword as indicated below:

"Database=sampledb; Protocol=tcpip; Hostname= myhost;
ServiceName=50001;
Security=ssl;
SSLClientKeystoredb=/home/test1/keystore/clientstore.kdb;
SSLClientKeystash=/home/test1/keystore/clientstore.sth;"

- If you are using the IBM data server client or IBM Data Server Runtime Client, you can use either connection string parameters or DB2 configuration parameters to set the path for the client key database and for the stash file
Configuring SSL Support in non-Java Clients

- If you are using the IBM data server client or IBM Data Server Runtime Client, you can use either connection string parameters or DB2 configuration parameters to set the path for the client key database and for the stash file.
- This example uses the db2cli.ini file to set connection string parameters:

```
[sampledb]
Database=sampledb
Protocol=tcpip
Hostname=myhost
Servicename=50001
Security=ssl
SSLClientKeystoredb=/home/test1/keystore/clientstore.kdb
SSLClientKeystash=/home/test1/keystore/clientstore.sth
```
Configuring SSL Support in non-Java Clients

- For an example of how to use a FileDSN keyword to identify DSN file that contains connection information refer to the DB2 Knowledge Center as follows:

NOTES:
Summary

- Threat Environment
- DB2 compliance with Security Standards
- DB2 Native Encryption Operation
- Key management
- SSL/TLS Client and Database Server configuration
- DB2 Native Encryption and SSL/TLS – A total solution
  - Secure Environment
References

- DB2 Native Encryption Primer --
  http://www.ibmbluhub.com/walid-encryption-primer/

- IBM DB2 Redbooks, 5 Tips series --
  https://www.ibm.com/developerworks/community/blogs/5things/entry/5_things_to_know_about_db2_native_encryption?lang=en

- SSL Setup:

- SSL Instance and Client Setup:
References

- Overview of DB2 Native Encryption, Greg Stager, IBM. DB2 Tech Talk on Developer Works (PDF) --
  https://www.ibm.com/developerworks/community/files/form/anonymous/api/library/3c0ed2b6-1b87-49b8-8ba6-f0d4437cec45/document/a39bb3eb-5ea5-4c43-a0da-48e3eef294ed/media/An

- Configuring SSL for non-Java Clients --

- Configure IBM Data Server Drive for JDBC and SQLJ to use SSL:

NOTES:
References

- Overview of DB2 Native Encryption, Greg Stager, IBM. DB2 Tech Talk on Developer Works (PDF) --
  https://www.ibm.com/developerworks/community/files/form/anonymous/api/library/3c0ed2b6-1b87-49b8-8ba6-f0d4437cec45/document/a39bb3eb-5ea5-4c43-a0da-48e3eef294ed/media/An

- PKCS12 Keystores --
References

- Developer Works Article on Managing Master Keys --

- Matt Huras’s DB2 11.1 Blog --
  http://www.ibmbigdatahub.com/blog/new-ibm-db2-release-simplifies-deployment-and-key-management

NOTES:
THANK YOU!

NOTES:
NOTES: Phil Gunning is an IBM Champion for Analytics with over 20 years of DB2 LUW experience. Phil is the principal of Gunning Technology Solutions, LLC specializing in DB2 LUW Performance and Tuning, High Availability, SQL Tuning, Disaster Recovery, Capacity Planning, Information Security, DB2 Migrations and DB2 Implementation planning. Phil has presented at IDUG North America, IDUG EMEA, local DB2 Regional User groups and is a major sponsor of the Central PA DB2 Users Group. Phil is the author or co-author of 5 books on DB2 LUW. Phil has a BS in Computer Science and an MBA. Phil has consulted at Fortune 500 companies around the world and implemented or tuned hundreds of DB2 LUW databases. Phil has taught Relational Database Design, JAVA Programming, Network Essentials, Operating Systems, Computer Architecture and Systems Analysis and Design at several universities. He used DB2 LUW for the Database class as far back as DB2 V5 Universal Database. Phil has also served as a technical reviewer for many DB2 LUW books and was on the editorial board of the IDUG Solutions Journal. Phil is a previous IDUG CPC Co-chair and two time presentations team lead.