www.DB2NightShow.com

#250 – Dec. 2, 2022

A Db2 Security Primer

Greg Stager – gstager@ca.ibm.com

IBM Canada

CIS (Center for Internet Security)

- CIS is a community driven non-profit organization, that among other activities publishes "Benchmarks" for various products
  - Series of recommendations for security hardening an installation

- I have worked with CIS to have the Db2 11.x Benchmark published

- https://www.cisecurity.org/cis-benchmarks/

Database security landscape

Authentication

Authorization

Auditing

Encryption

Authentication

Proving you are who you say you are

- Authentication is the process used by Db2 to validate that the credentials presented for an external user identity are valid and meant for use with the given user identity
  - Db2 relies on external 3rd parties to provide this validation
- The mechanism used for this validation is defined by the AUTHENTICATION database manager configuration parameter
  - All databases under the same Db2 instance use the same authentication mechanism
- Results of a successful authentication:
  - External User ID is mapped to a Db2 authorization ID
  - Any externally defined groups associated with the user are mapped to Db2 authorization IDs
    - Group membership is also defined outside of Db2

Authentication options

- Operating system (default)
  - User validation using a password
  - Group membership

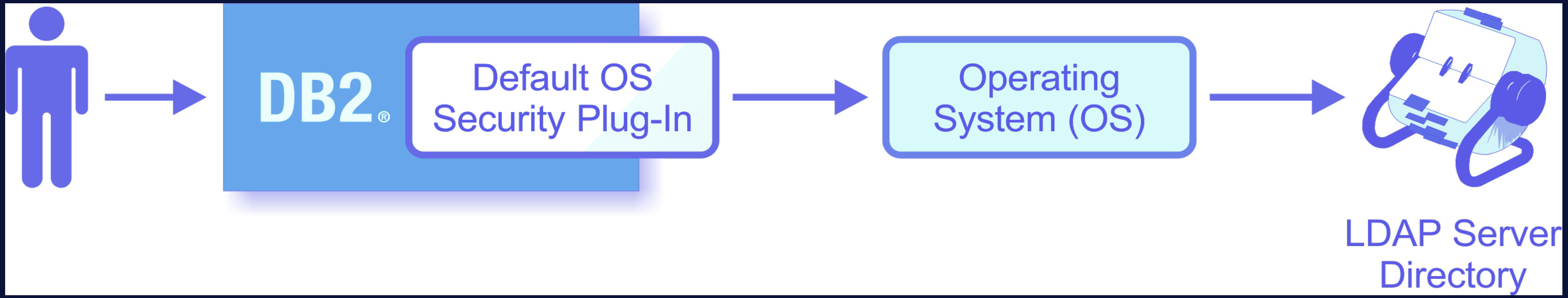
- Kerberos
  - User validation using a Kerberos ticket
  - Single sign-on
  - No group membership

- LDAP Plug-in
  - User validation using a password
  - Group membership

- Customized (via Plug-in)
  - Two types – userid/password or GSSAPI based
  - Group membership

Most popular? Transparent LDAP

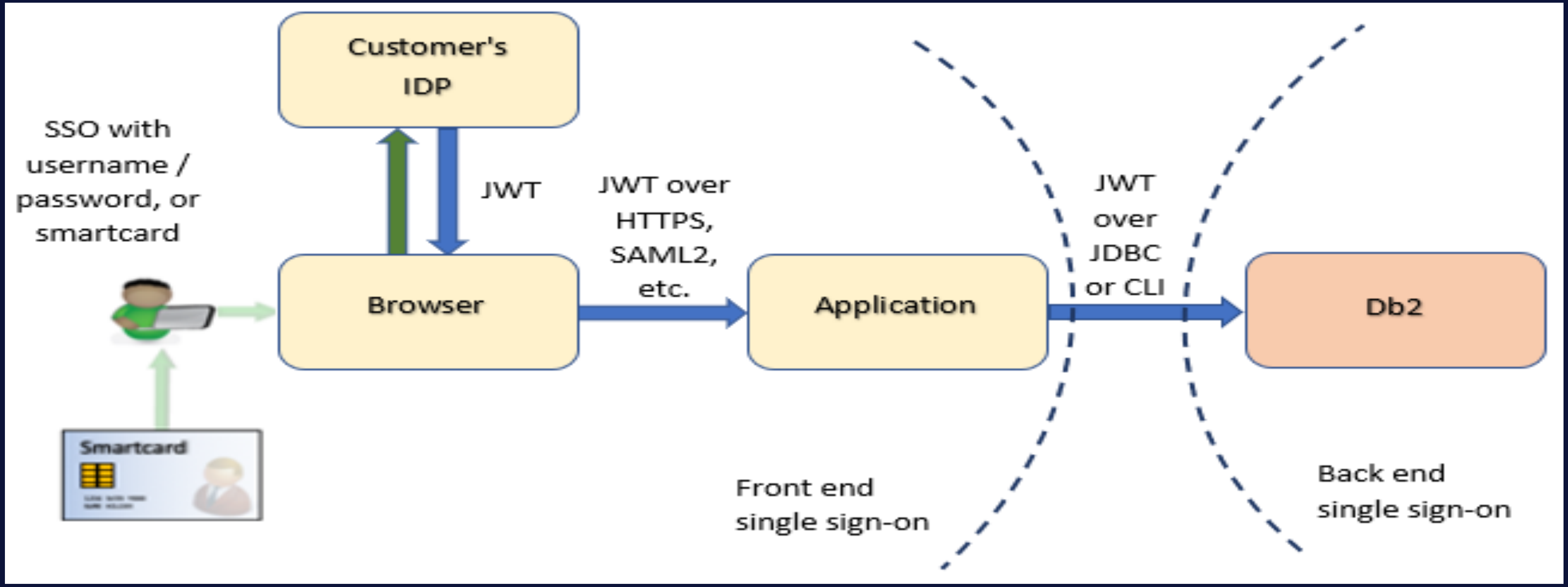
- LDAP = Lightweight Directory Access Protocol (LDAP)
  - Standard protocol for accessing information at a directory server
  - Enables centralized authentication services for use across enterprise



- The transparent LDAP approach integrates LDAP at the OS level which means both OS and Db2 authentication requests are satisfied by the same mechanism
  - Db2 authenticates users and acquires their groups via regular OS APIs
  - Enabled by setting DB2AUTH registry variable to “OSAUTHDB”

Single sign-on with JSON Web Token (11.5.4.0)

- Authentication without username/password via JSON Web Token (JWT)
- Identity Provider (IDP) signs the token that is later passed to Db2
- Db2 is configured to "trust" IDPs and use their public key to verify the token
- If the token is valid, Db2 uses the identity within the JWT for authentication



Example

- Sample JWT payload:

```
{
  "name": "John Doe",
  "issuer": "KNOXSSO",
  "username": "admin",
  "exp": 1516239022
}
```

- On CLP:

```
CONNECT TO dbname ACCESSTOKEN <token> ACCESSTOKENTYPE JWT
```

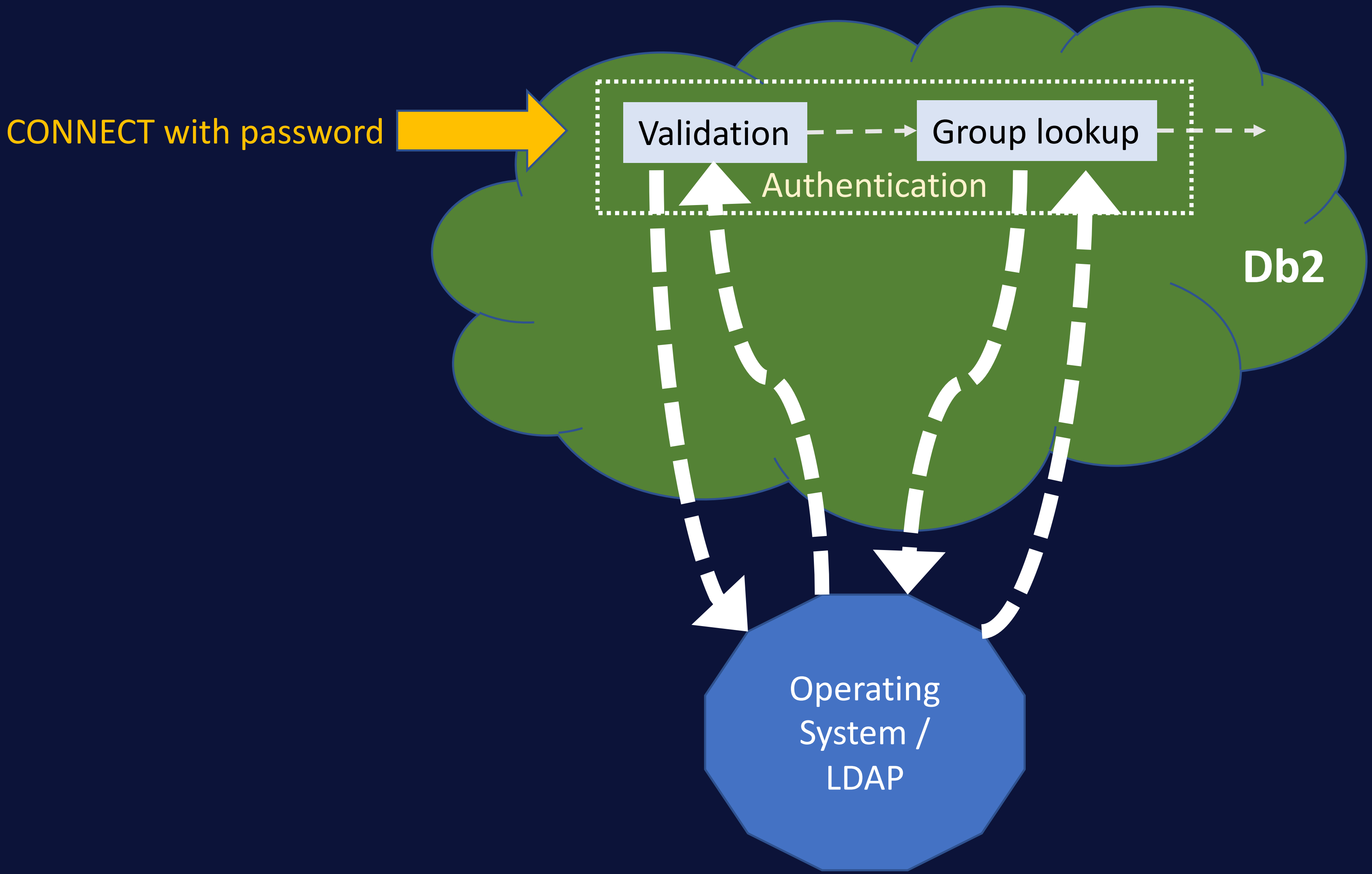
- In JDBC using API

```
dataSource.setAccessToken( "<token>" );
dataSource.setAccessTokenType( "JWT" );
...
Connection conn = dataSource.getConnection( );
```

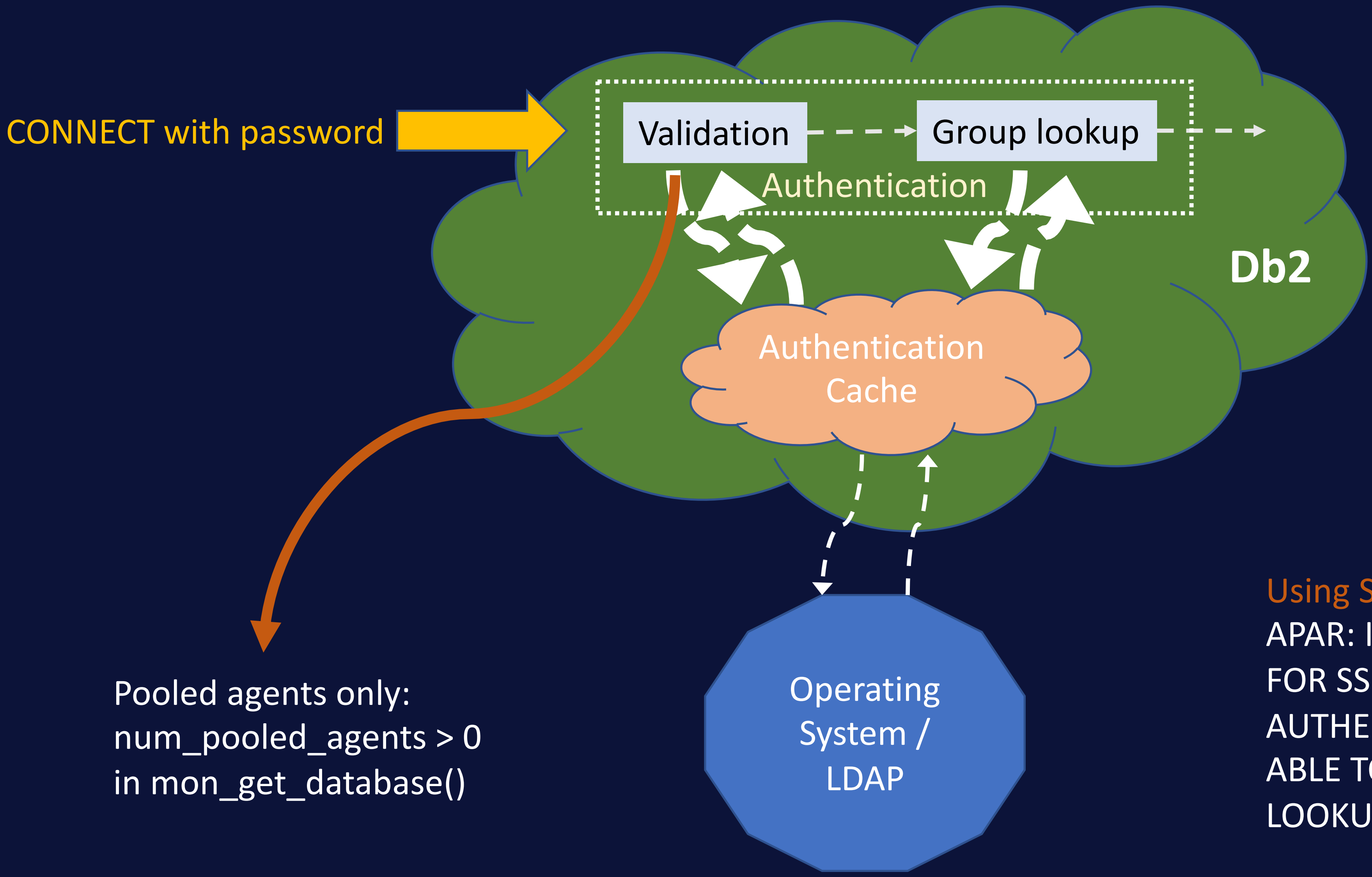
Authentication cache (11.5.3.0)

- Introduced to help improve performance for password based authentication in the following scenarios:
  - Connections are of extremely short duration
  - Authentication "pipeline" gets overwhelmed
- Cache contains the results of successful authentication efforts and group lookup results
  - Results are only kept for limited amount of time (configurable)
  - Cache size is configurable by number of unique user IDs to be cached
- When enabled, incoming requests (and associated credentials) are compared to cached entries and, if match found, further authentication processing is bypassed

Without the authentication cache



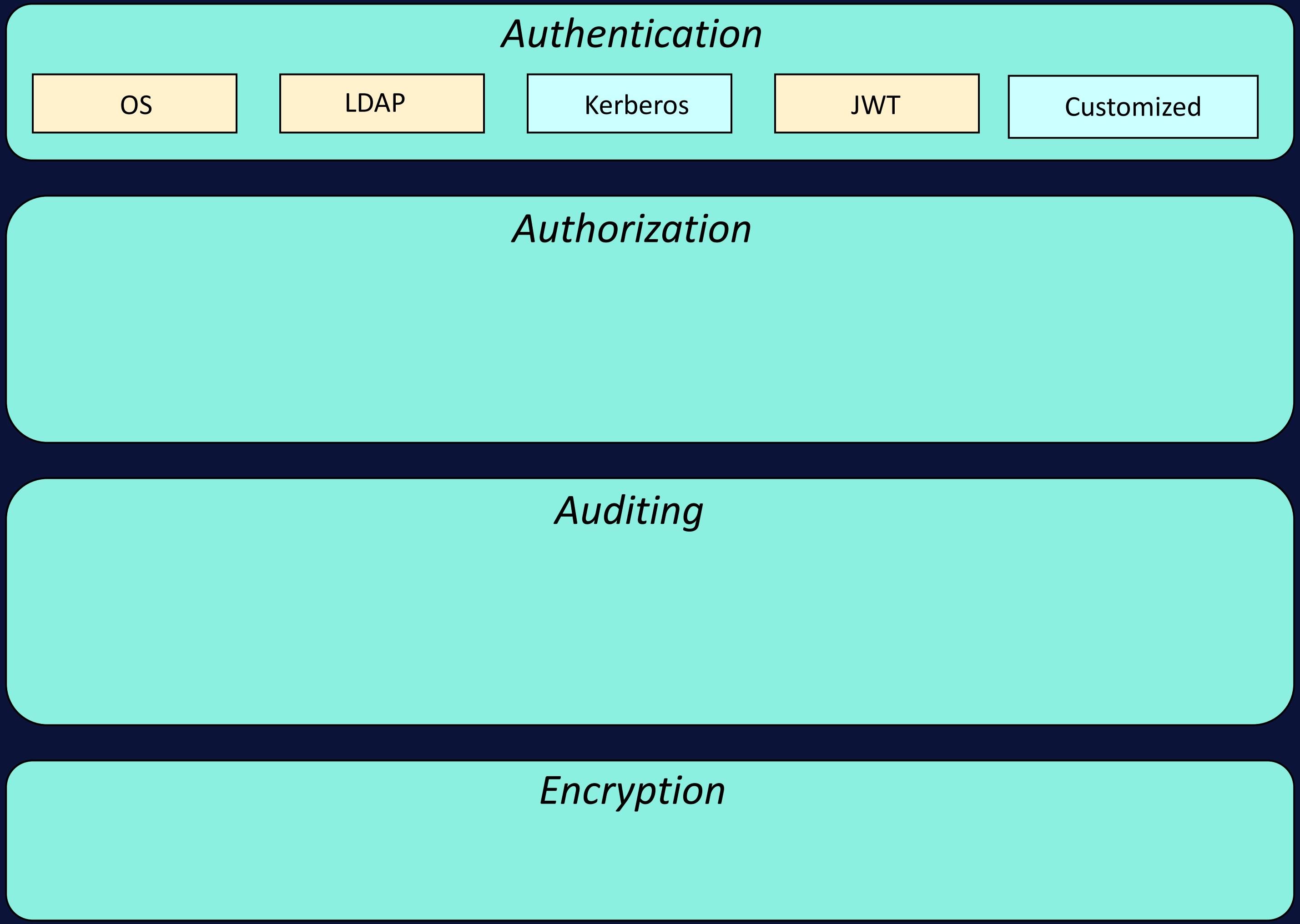
With the authentication cache



Pooled agents only:
num_pooled_agents > 0
in mon_get_database()

Using SSL? Get 11.5.7.0 or later
APAR: IT36961
FOR SSL CONNECTIONS ONLY, AUTHENTICATION CACHE IS NOT ABLE TO PERFORM USER OR GROUP LOOKUP

Database security landscape



Authorization

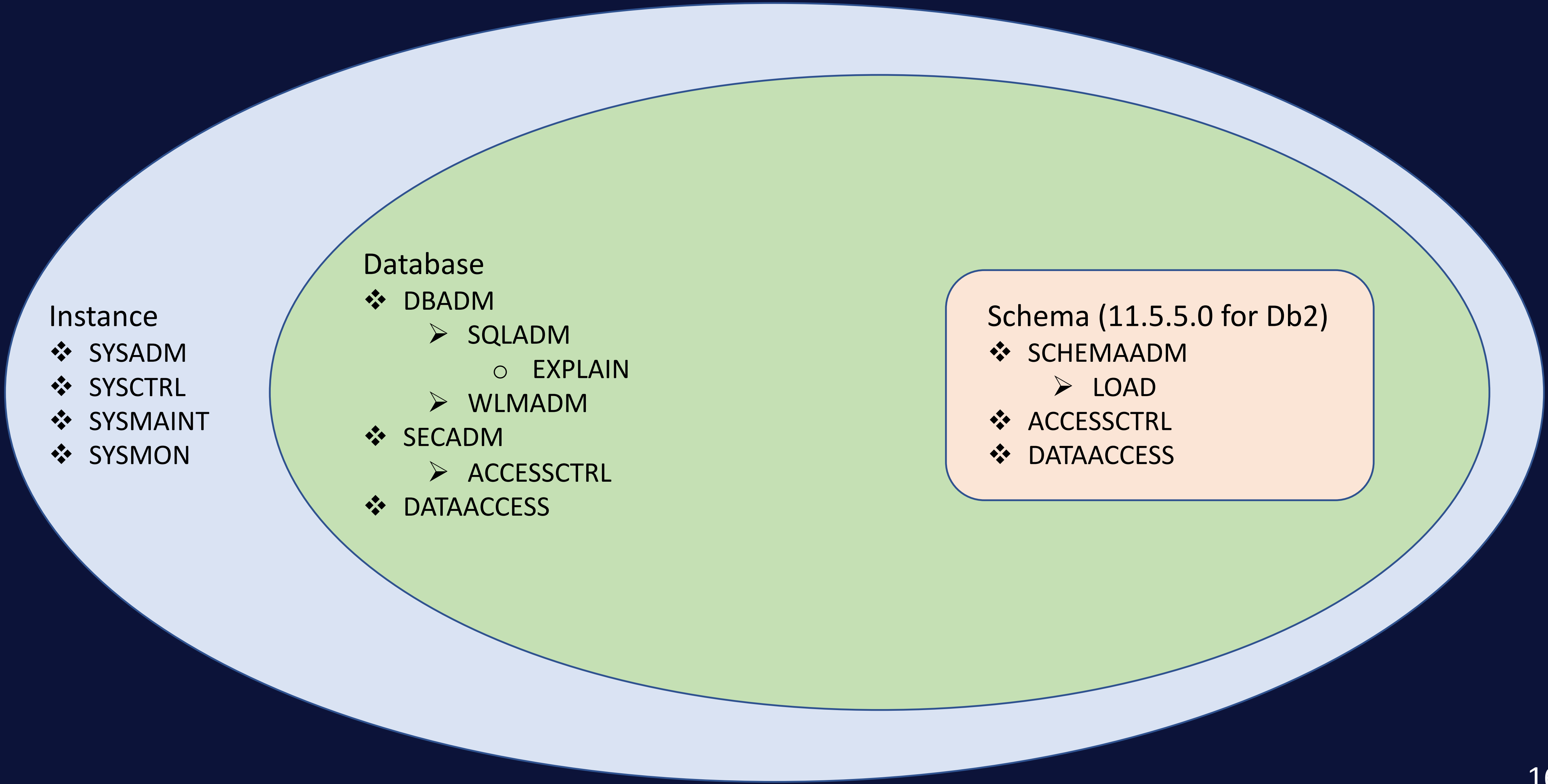
Authorities & privileges: What are they?

- Authorities and privileges are explicitly declared permissions within Db2 used to allow users to perform specific actions
    - ➢An action is authorized based on the collection of authorities and privileges held, directly or indirectly, by an authorization ID

- Authorities represent a predefined collection of Db2 permissions within a specific domain

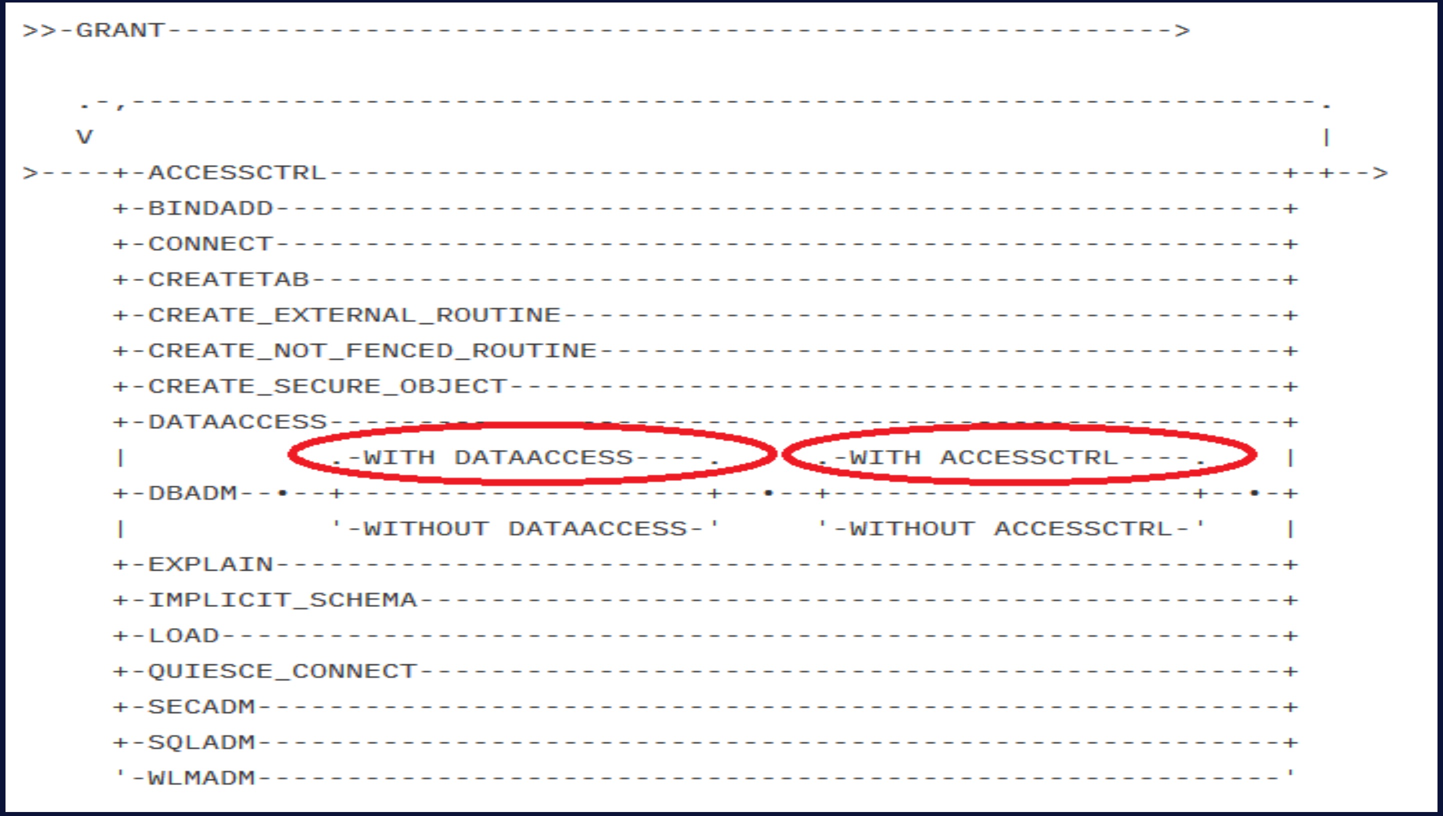
- Privileges represent Db2 permissions on a specific database object
    - Privilege is on a specific instance of an object (not a specific type of object)

Primary Db2 authorities



Caution: DBADM GRANT statements

- By default, GRANT DBADM also implicitly grants DATAACCESS and ACCESSCTRL authorities
    - ➢Only do this if they really need it!



```
>>-GRANT-------------------------------------------------------------->

      .-,------------------------------------------------------------------.
      V                                                                     |
>-----+-ACCESSCTRL---------------------------------------------------------+-+-->
      +-BINDADD------------------------------------------------------------+
      +-CONNECT------------------------------------------------------------+
      +-CREATETAB----------------------------------------------------------+
      +-CREATE_EXTERNAL_ROUTINE--------------------------------------------+
      +-CREATE_NOT_FENCED_ROUTINE------------------------------------------+
      +-CREATE_SECURE_OBJECT-----------------------------------------------+
      +-DATAACCESS---------------------------------------------------------+
      |                 .-WITH DATAACCESS----.     .-WITH ACCESSCTRL----.  |
      +-DBADM--•--+-----------------------+--•--+-----------------------+--•-+
      |                 '-WITHOUT DATAACCESS-'     '-WITHOUT ACCESSCTRL-'  |
      +-EXPLAIN------------------------------------------------------------+
      +-IMPLICIT_SCHEMA----------------------------------------------------+
      +-LOAD---------------------------------------------------------------+
      +-QUIESCE_CONNECT----------------------------------------------------+
      +-SECADM-------------------------------------------------------------+
      +-SQLADM-------------------------------------------------------------+
      '-WLMADM-------------------------------------------------------------'
```

Additional schema level privileges (11.5.5.0 for Db2)

SELECTIN	Gives the ability to retrieve rows from all existing and ***future*** tables or views defined in the schema
INSERTIN	Gives the ability to insert rows and to run the IMPORT utility on all existing and ***future*** tables or views defined in the schema
UPDATEIN	Gives the ability to use the UPDATE statement on all existing and ***future*** tables or updatable views defined in the schema
DELETEIN	Gives the ability to delete rows from all existing and ***future*** tables or updatable views defined in the schema
EXECUTEIN	Gives the ability run all existing and ***future*** user-defined functions, methods, procedures, packages, or modules defined in the schema

Authorities & privileges: Who can hold them?

- Authorities and privileges can be associated to a specific Db2 authorization ID

- A Db2 authorization ID consists of:
  - Authorization ID type
  - Unique authorization ID value (128-byte limit)

- Authorization ID types:
  - Individual user (‘U’)
  - Group (‘G’)
  - Role (‘R’)
  - PUBLIC (‘P’)
    - Represents all authorization IDs in the “universe”

How the different authorization ID types interact

- Authorization processing considers both a primary authorization ID and secondary authorization IDs associated with the primary ID
  - Specific IDs considered are determined by the context

- Primary authorization ID
  - Used to record “who” performed the action
  - Represents an individual user ID (‘U’)

- Secondary authorization ID
  - One or more authorization IDs associated with the primary authorization ID
  - Used to supplement the primary ID's privileges where allowed by Db2
  - Represents groups, database roles, and/or PUBLIC

Common Db2 authorization ID terminology

- SYSTEM AUTHORIZATION ID
  - Primary authorization ID (and associated secondary authorization IDs) used to establish the current session and is checked for CONNECT privilege

- SESSION AUTHORIZATION ID
  - Primary authorization ID (and associated secondary authorization IDs) used for any session authorization checking after CONNECT processing.
    - ➢ The value of the SESSION AUTHORIZATION ID is controlled by the authentication logic but is typically the same value as the SYSTEM AUTHORIZATION ID

- STATEMENT AUTHORIZATION ID
  - Primary authorization ID (and associated secondary authorization IDs) used for any authorization requirements of an SQL statement
    - ➢ Also used to determine object ownership (for DDL where appropriate).
  - Can vary depending on the type of SQL statement and the context in which the statement is issued
    - ➢ Different sources for dynamic SQL versus static SQL and different options available for routine or non-routine context

Authorities & privileges: How do you get them?

- Authorities and privileges can be acquired permanently through explicit and implicit mechanisms
  - GRANT and REVOKE SQL statements are the explicit mechanisms
    - Can be used for an authorization ID representing user, group, or role
  - Object creation and removal are examples of an implicit mechanism
    - E.g., Object owner granted some permissions by Db2 as a result of the creation
      - Note that other implicit mechanisms exist
- Authorities and privileges can also be acquired temporarily through different execution contexts
  - E.g., You have access to them only while you are in that execution context
  - Examples of different execution contexts include static SQL, views, routines, and trusted contexts

Temporary access to authorities and privileges

- Inheritance
  - A privilege on a view implicitly gives you the same access to the objects in the view definition when using the view
    - E.g., Inserting into a view, inserts into a table in the view definition
  - Execute privilege on a package gives you the right to execute any static SQL in that package and inherit the package owner's privileges through that SQL
    - E.g., Executing a package with static DELETE statement, lets you delete from that object

- Alternate authorization models
  - Package DYNAMICRULES bind option lets you specify which authorization ID is used for embedded dynamic SQL issued by that application
    - Possible options include able to have the primary authorization ID be the Package Definer or Executor, the (associated) Routine Definer or Invoker

- Trusted context

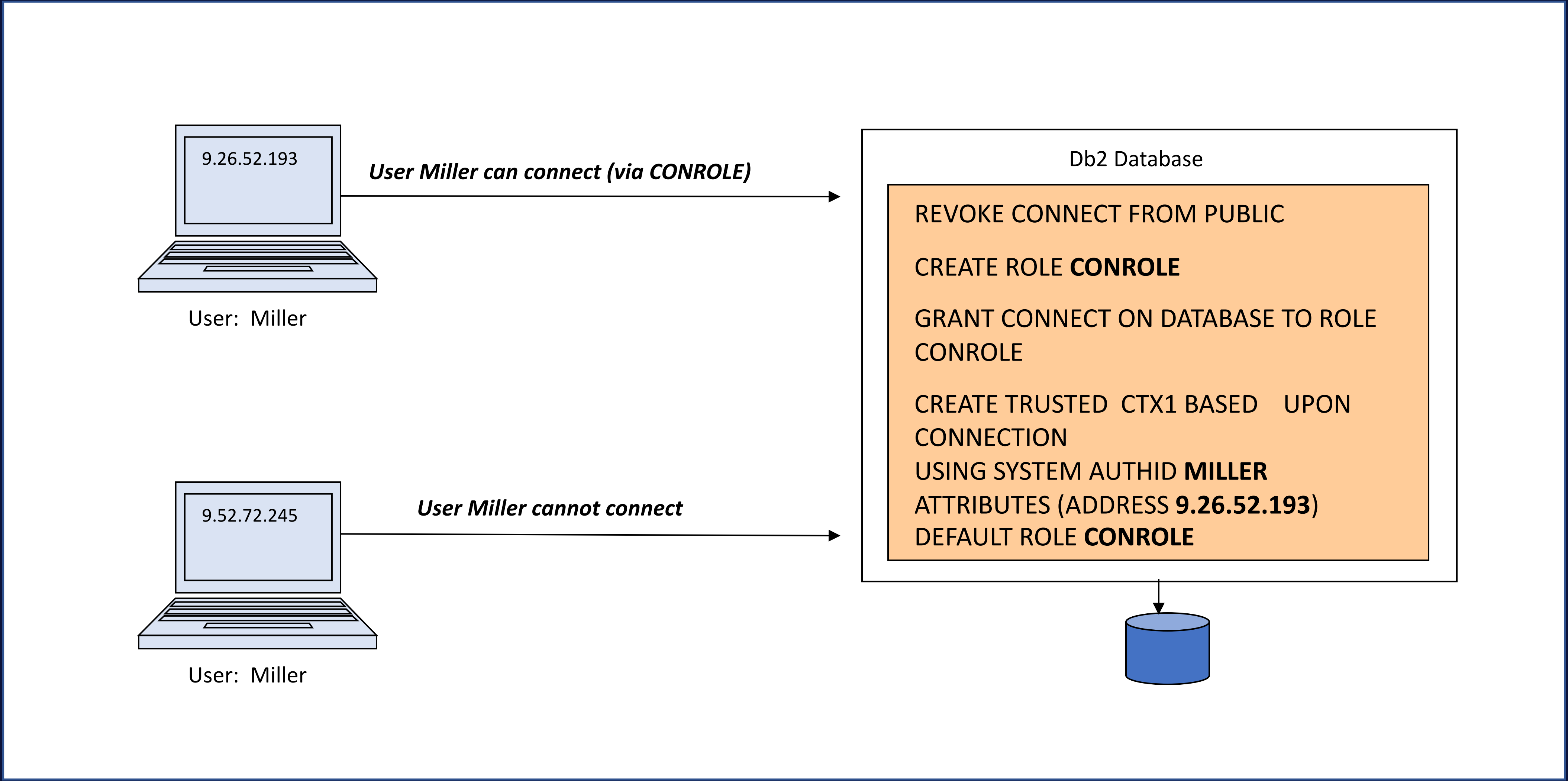
A trusted context is…

- A declaration of a “trust relationship” between the database and an external application based on a set of explicit of trust attributes:
  - System authorization ID
  - IP address
  - Level of communication security

- A connection that matches the trust attributes for a defined trusted context is called a trusted connection. There are 2 types:
  - An implicit trusted connection
  - An explicit trusted connection

- An trusted connection allows a user to inherit a role that is not available to them outside the scope of that trusted connection
  - ➢The session authorization ID of the connection is given “temporary” membership to a role declared in the trusted context definition

Implicit trusted connection: Role inheritance



9.26.52.193
User: Miller
User Miller can connect (via CONROLE)
9.52.72.245
User: Miller
User Miller cannot connect
Db2 Database

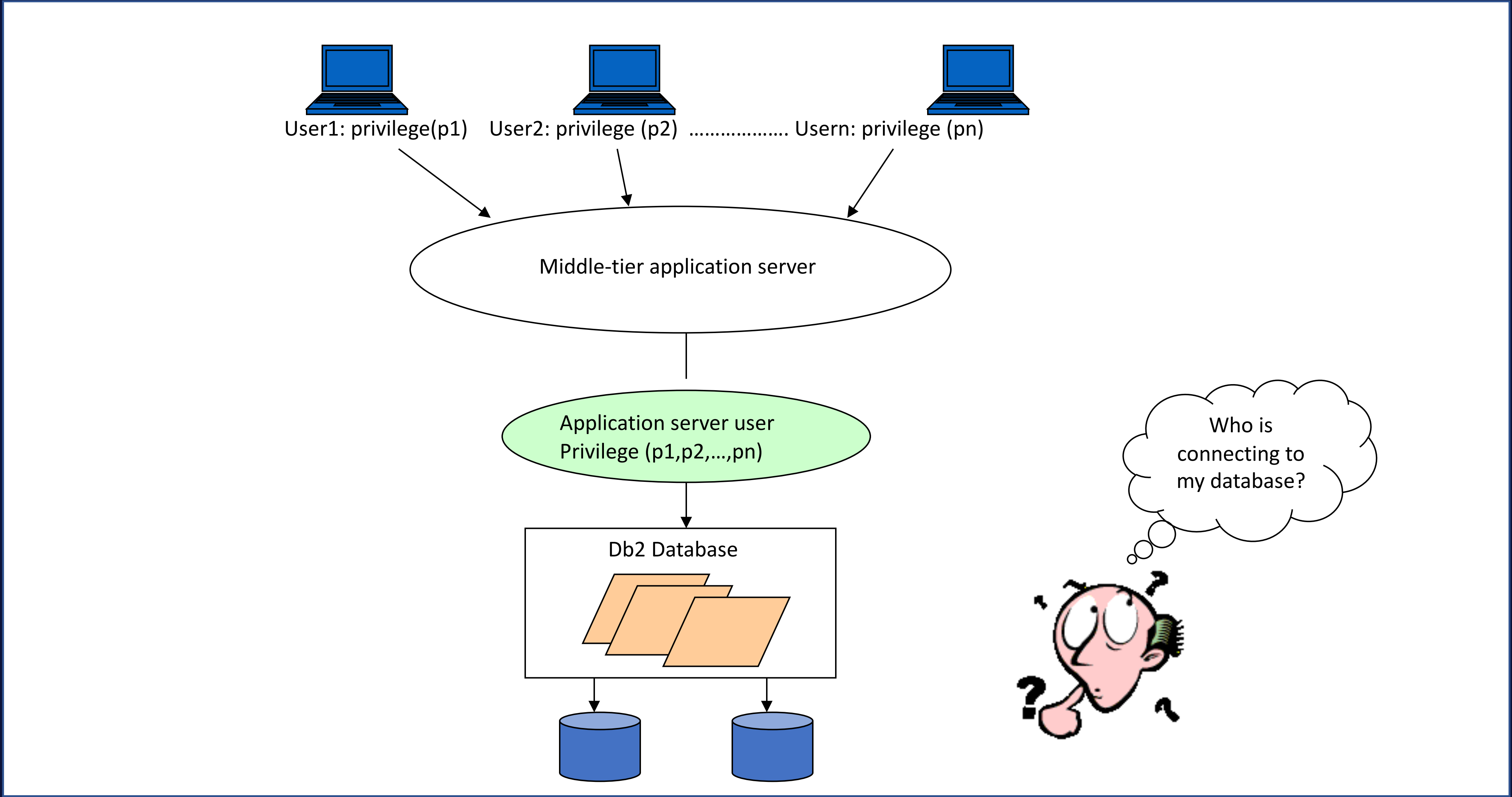
```
REVOKE CONNECT FROM PUBLIC

CREATE ROLE CONROLE

GRANT CONNECT ON DATABASE TO ROLE
CONROLE

CREATE TRUSTED  CTX1 BASED    UPON
CONNECTION
USING SYSTEM AUTHID MILLER
ATTRIBUTES (ADDRESS 9.26.52.193)
DEFAULT ROLE CONROLE
```

Explicit trusted connections

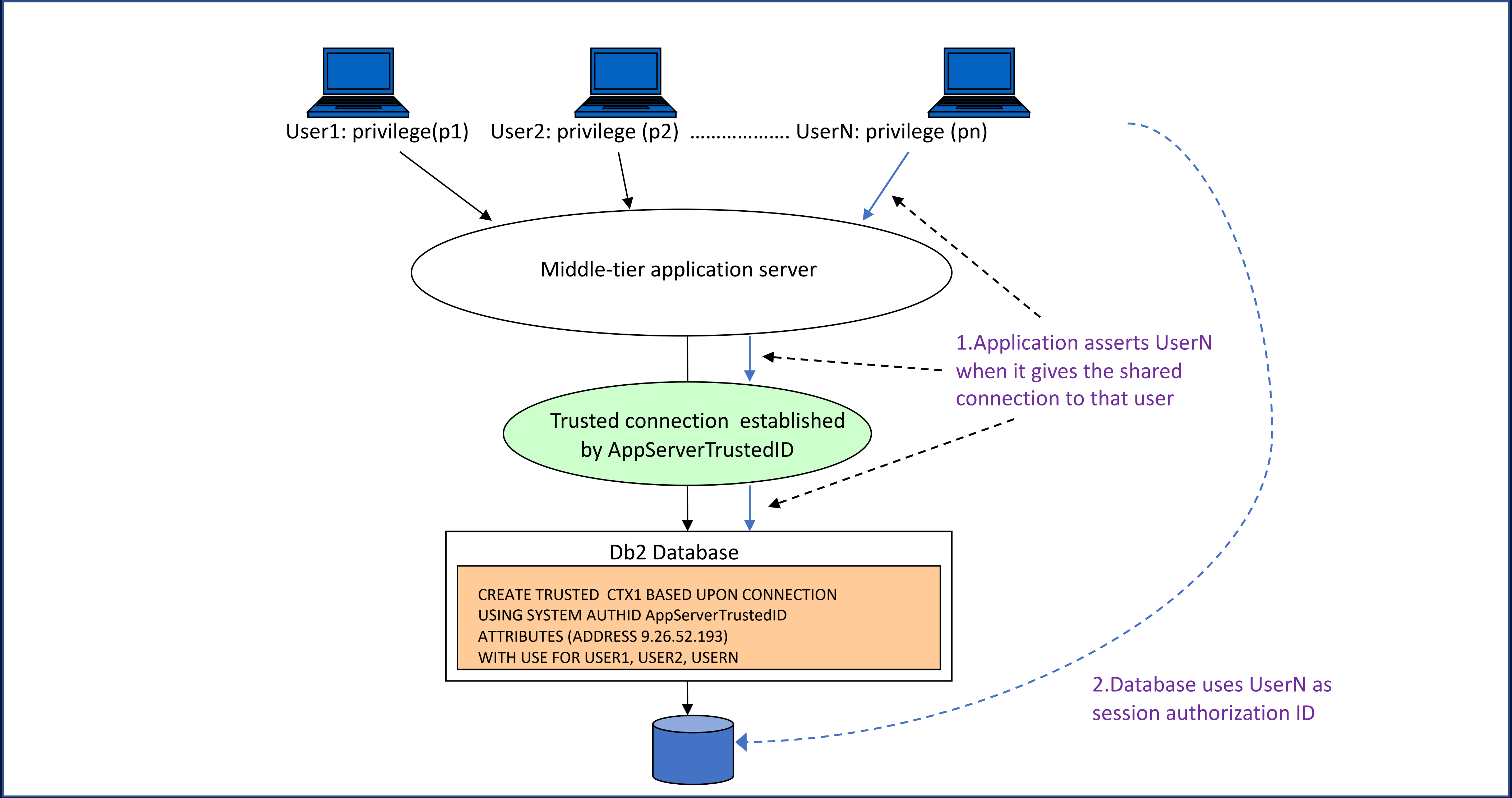
- An explicit trusted connection allows a trusted application server to switch, or assert, the current end-user ID on the existing connection in an efficient manner

- An application server establishes the original connection with an explicit request for trust and, once established, it can then issue requests to the database server to change the session authorization ID for any new unit of work
    - ➢The ID used to do the initial connect request for the application server only needs CONNECT privilege

Who is doing what ?



Identity assertion model


User1: privilege(p1)
User2: privilege (p2)
.... UserN: privilege (pn)
Middle-tier application server
1.Application asserts UserN when it gives the shared connection to that user
Trusted connection established by AppServerTrustedID
Db2 Database
2.Database uses UserN as session authorization ID

```
CREATE TRUSTED  CTX1 BASED UPON CONNECTION
USING SYSTEM AUTHID AppServerTrustedID
ATTRIBUTES (ADDRESS 9.26.52.193)
WITH USE FOR USER1, USER2, USERN
```

Advanced authorization controls

- Sometimes there are requirements to implement authorization controls that operate within a table object itself at the row, column, or cell level
  - Such controls are referred to as “fine grained access controls” (FGAC)

- FGAC supplement traditional authorization controls and allow security administrators to control the results sets seen by different people even when they run the same SQL statement
  - ➢Your privileges do not control what data you can see with FGAC

- Db2 offers two variations of FGAC
  - Label-Based Access Control (LBAC)
  - Row and Column Access Control (RCAC)

Label-Based Access Control (LBAC)

- LBAC is an implementation of a Mandatory Access Control (MAC) system
  - Both the users and the data itself are explicitly assigned a security label value
  - The intersection between the user security label and the data security label determines what rows and columns can be seen by each user
    - Based on a set of pre-defined rules on how different security labels interact

- A key prerequisite for LBAC is a clear definition of security labels and assignments
  - Change is very difficult to propagate as labels are part of the data itself

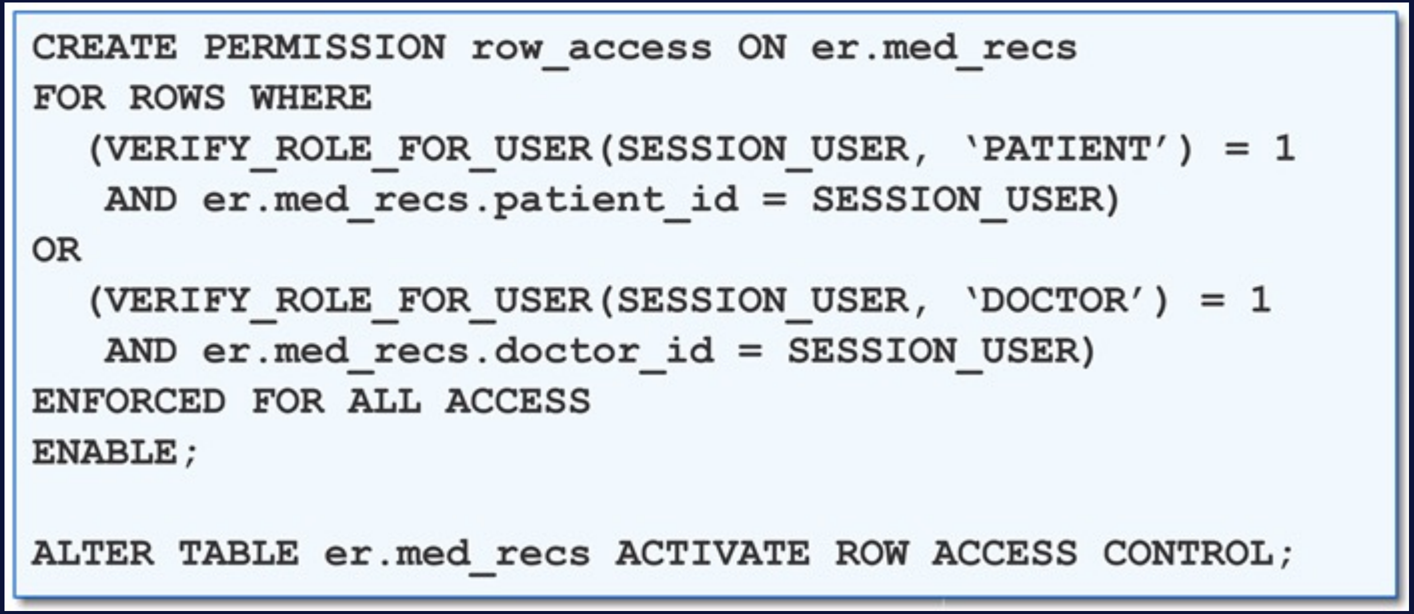
- Primary use case is in traditional military and intelligence domains
  - Is used in some commercial environments

Row and Column Access Control (RCAC)

- RCAC is based on the use of simple, flexible SQL to express customer supplied rules
  - Unlike LBAC, no need to define and assign security labels and no impact on the underlying data

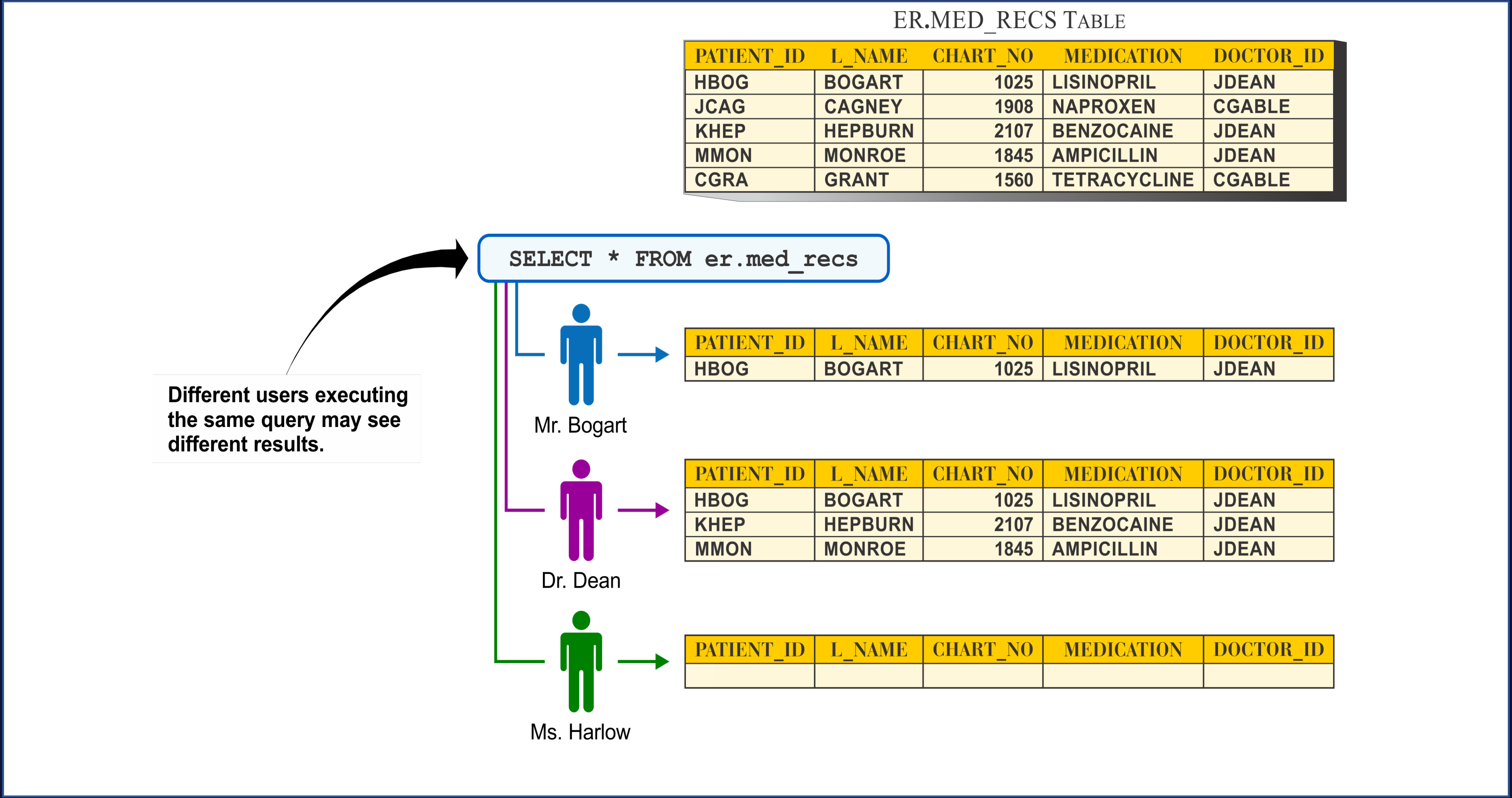
- RCAC consists of two components:
  - Row permissions
    - An SQL search condition that describes what set of rows can be accessed
  - Column masks
    - An SQL CASE expression that describes what column values are permitted to be seen and under what conditions

Row permission example



```
CREATE PERMISSION row_access ON er.med_recs
FOR ROWS WHERE
  (VERIFY_ROLE_FOR_USER(SESSION_USER, 'PATIENT') = 1
   AND er.med_recs.patient_id = SESSION_USER)
OR
  (VERIFY_ROLE_FOR_USER(SESSION_USER, 'DOCTOR') = 1
   AND er.med_recs.doctor_id = SESSION_USER)
ENFORCED FOR ALL ACCESS
ENABLE;

ALTER TABLE er.med_recs ACTIVATE ROW ACCESS CONTROL;
```

Row permissions in action



ER.MED_RECS TABLE

PATIENT_ID	L_NAME	CHART_NO	MEDICATION	DOCTOR_ID
HBOG	BOGART	1025	LISINOPRIL	JDEAN
JCAG	CAGNEY	1908	NAPROXEN	CGABLE
KHEP	HEPBURN	2107	BENZOCAINE	JDEAN
MMON	MONROE	1845	AMPICILLIN	JDEAN
CGRA	GRANT	1560	TETRACYCLINE	CGABLE

```
SELECT * FROM er.med_recs
```

Different users executing the same query may see different results.

Mr. Bogart

PATIENT_ID	L_NAME	CHART_NO	MEDICATION	DOCTOR_ID
HBOG	BOGART	1025	LISINOPRIL	JDEAN

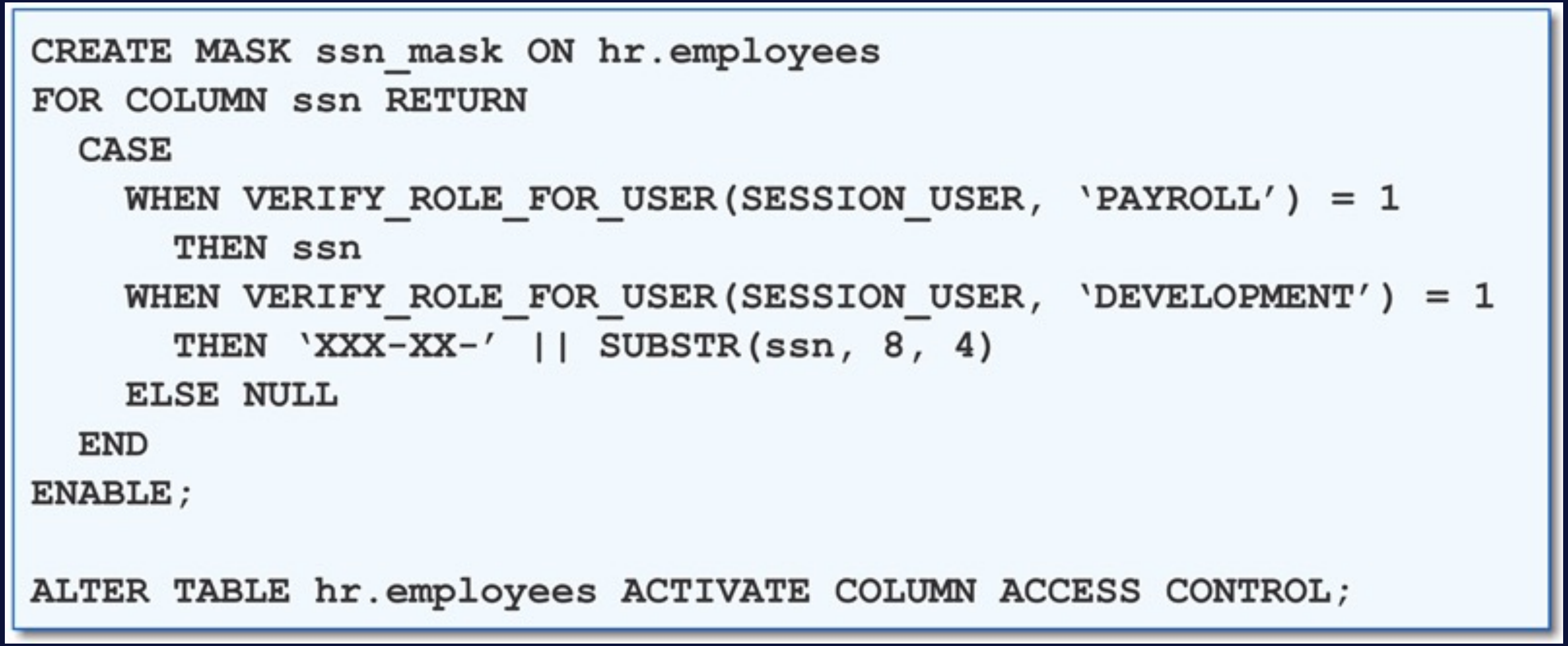
Dr. Dean

PATIENT_ID	L_NAME	CHART_NO	MEDICATION	DOCTOR_ID
HBOG	BOGART	1025	LISINOPRIL	JDEAN
KHEP	HEPBURN	2107	BENZOCAINE	JDEAN
MMON	MONROE	1845	AMPICILLIN	JDEAN

Ms. Harlow

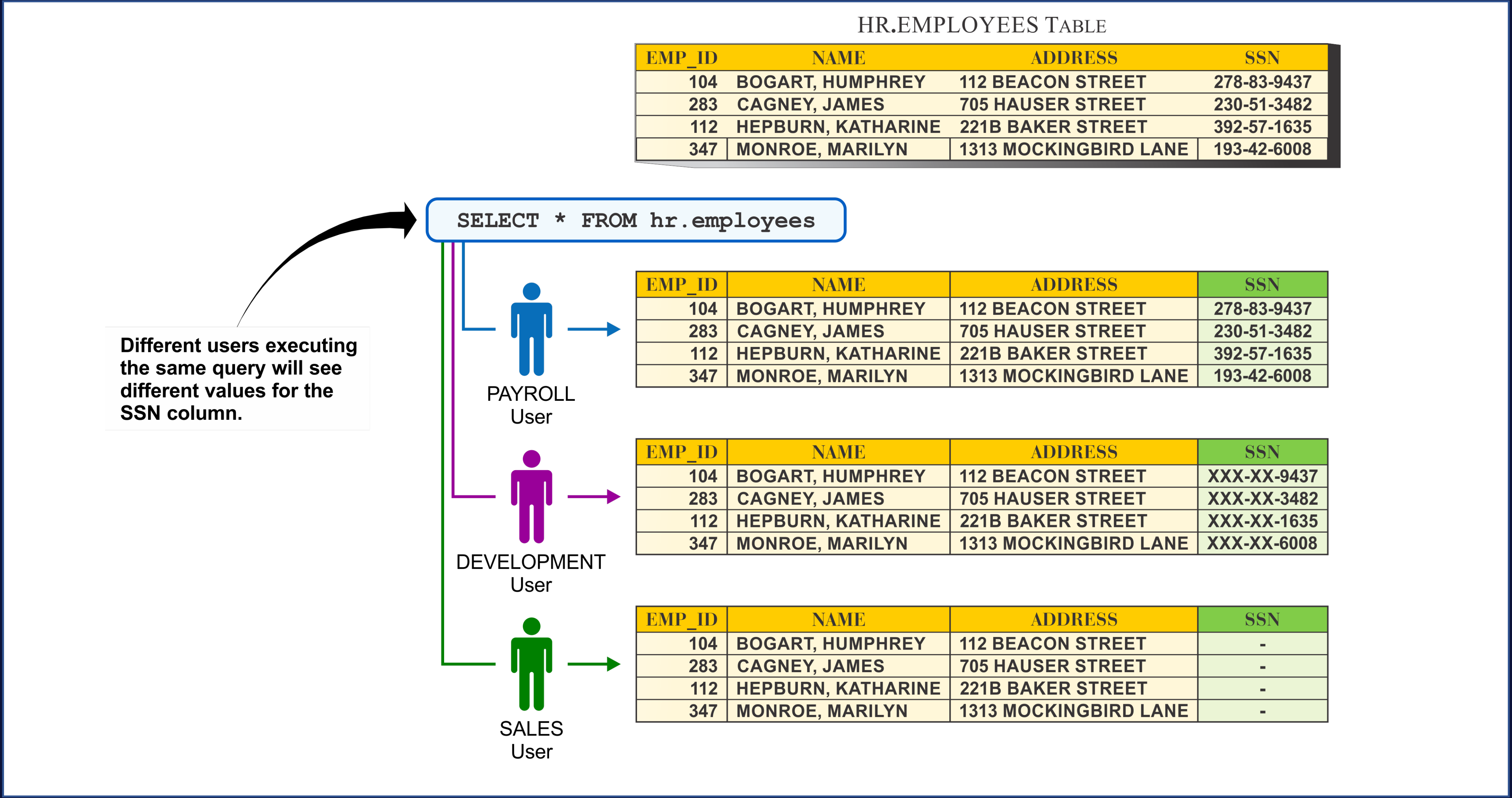
PATIENT_ID	L_NAME	CHART_NO	MEDICATION	DOCTOR_ID

Column mask example



```
CREATE MASK ssn_mask ON hr.employees
FOR COLUMN ssn RETURN
  CASE
    WHEN VERIFY_ROLE_FOR_USER(SESSION_USER, ‘PAYROLL’) = 1
      THEN ssn
    WHEN VERIFY_ROLE_FOR_USER(SESSION_USER, ‘DEVELOPMENT’) = 1
      THEN ‘XXX-XX-’ || SUBSTR(ssn, 8, 4)
    ELSE NULL
  END
ENABLE;

ALTER TABLE hr.employees ACTIVATE COLUMN ACCESS CONTROL;
```

Column masks at work



HR.EMPLOYEES TABLE

EMP_ID	NAME	ADDRESS	SSN
104	BOGART, HUMPHREY	112 BEACON STREET	278-83-9437
283	CAGNEY, JAMES	705 HAUSER STREET	230-51-3482
112	HEPBURN, KATHARINE	221B BAKER STREET	392-57-1635
347	MONROE, MARILYN	1313 MOCKINGBIRD LANE	193-42-6008

```
SELECT * FROM hr.employees
```

Different users executing the same query will see different values for the SSN column.

PAYROLL User

EMP_ID	NAME	ADDRESS	SSN
104	BOGART, HUMPHREY	112 BEACON STREET	278-83-9437
283	CAGNEY, JAMES	705 HAUSER STREET	230-51-3482
112	HEPBURN, KATHARINE	221B BAKER STREET	392-57-1635
347	MONROE, MARILYN	1313 MOCKINGBIRD LANE	193-42-6008

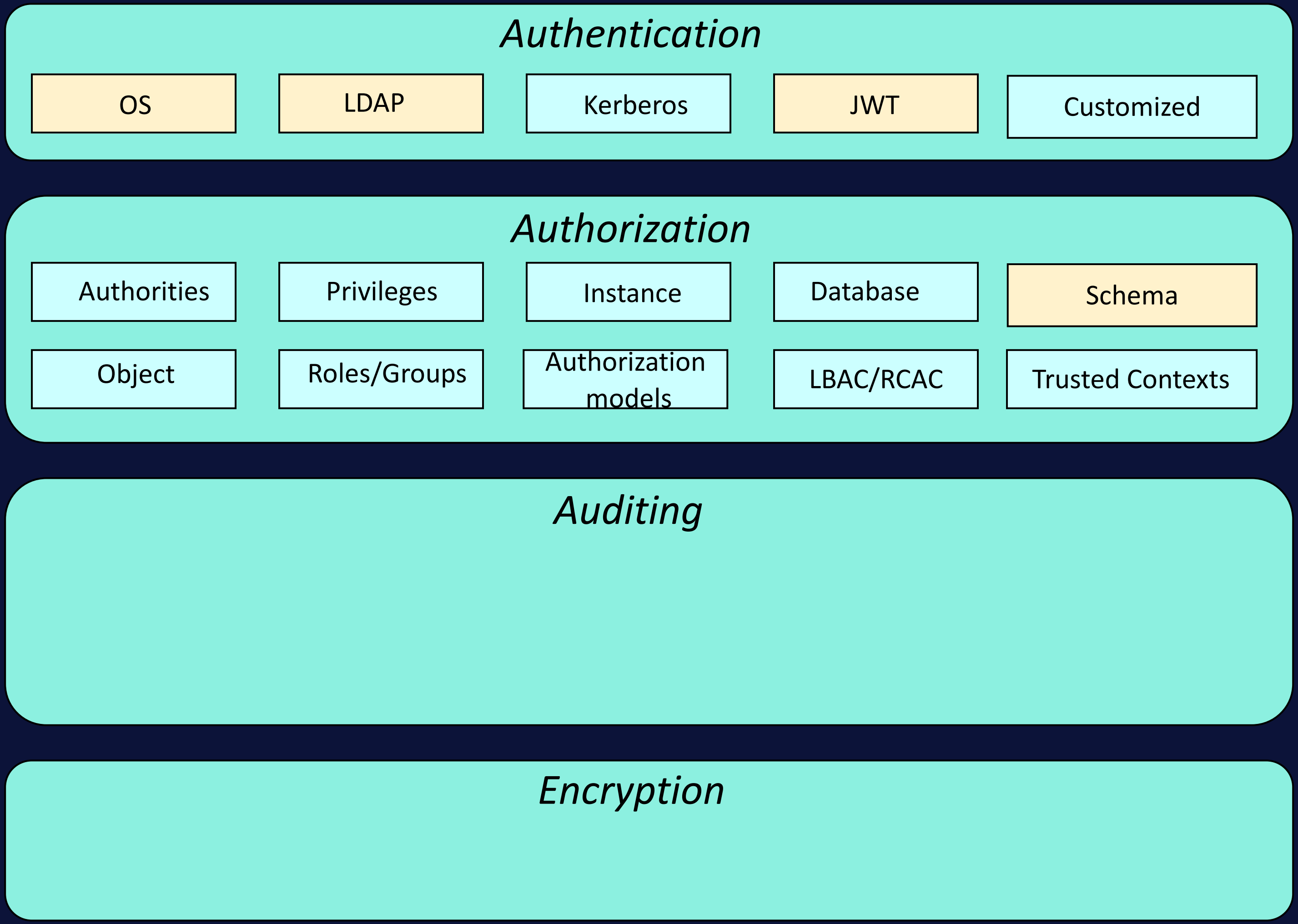
DEVELOPMENT User

EMP_ID	NAME	ADDRESS	SSN
104	BOGART, HUMPHREY	112 BEACON STREET	XXX-XX-9437
283	CAGNEY, JAMES	705 HAUSER STREET	XXX-XX-3482
112	HEPBURN, KATHARINE	221B BAKER STREET	XXX-XX-1635
347	MONROE, MARILYN	1313 MOCKINGBIRD LANE	XXX-XX-6008

SALES User

EMP_ID	NAME	ADDRESS	SSN
104	BOGART, HUMPHREY	112 BEACON STREET	-
283	CAGNEY, JAMES	705 HAUSER STREET	-
112	HEPBURN, KATHARINE	221B BAKER STREET	-
347	MONROE, MARILYN	1313 MOCKINGBIRD LANE	-

Database security landscape



Auditing

Auditing for Db2

- Two primary options offered with Db2:
  - Integration with an external product (IBM Guardium)
  - Db2 Audit facility

- Integration with IBM Guardium
  - Done using DRDA communication buffer exit on both send and receive
  - Offers programmatic ability to terminate a connection if desired

- Db2 audit facility
  - Internal audit capability for a pre-defined set of events which provide insight into who did what, when, and where

Db2 audit facility

- Audit can be configured at both instance level and within each database
    - Separate audit log for instance and each database

- Configuration can specify desire to audit one or more of the defined event categories:
    - AUDIT: Change in audit settings or audit log access
    - CHECKING: Authorization checks
    - OBJMAINT: Objects created or dropped (some but not all alterations)
    - SECMAINT: Changes to security controls
    - SYSADMIN: Use of SYSADM, SYSMAINT, or SYSCTRL authority
    - VALIDATE: Authentication or access of system security information
    - CONTEXT: Shows contextual information for a database operation
    - EXECUTE: Execution of SQL statements

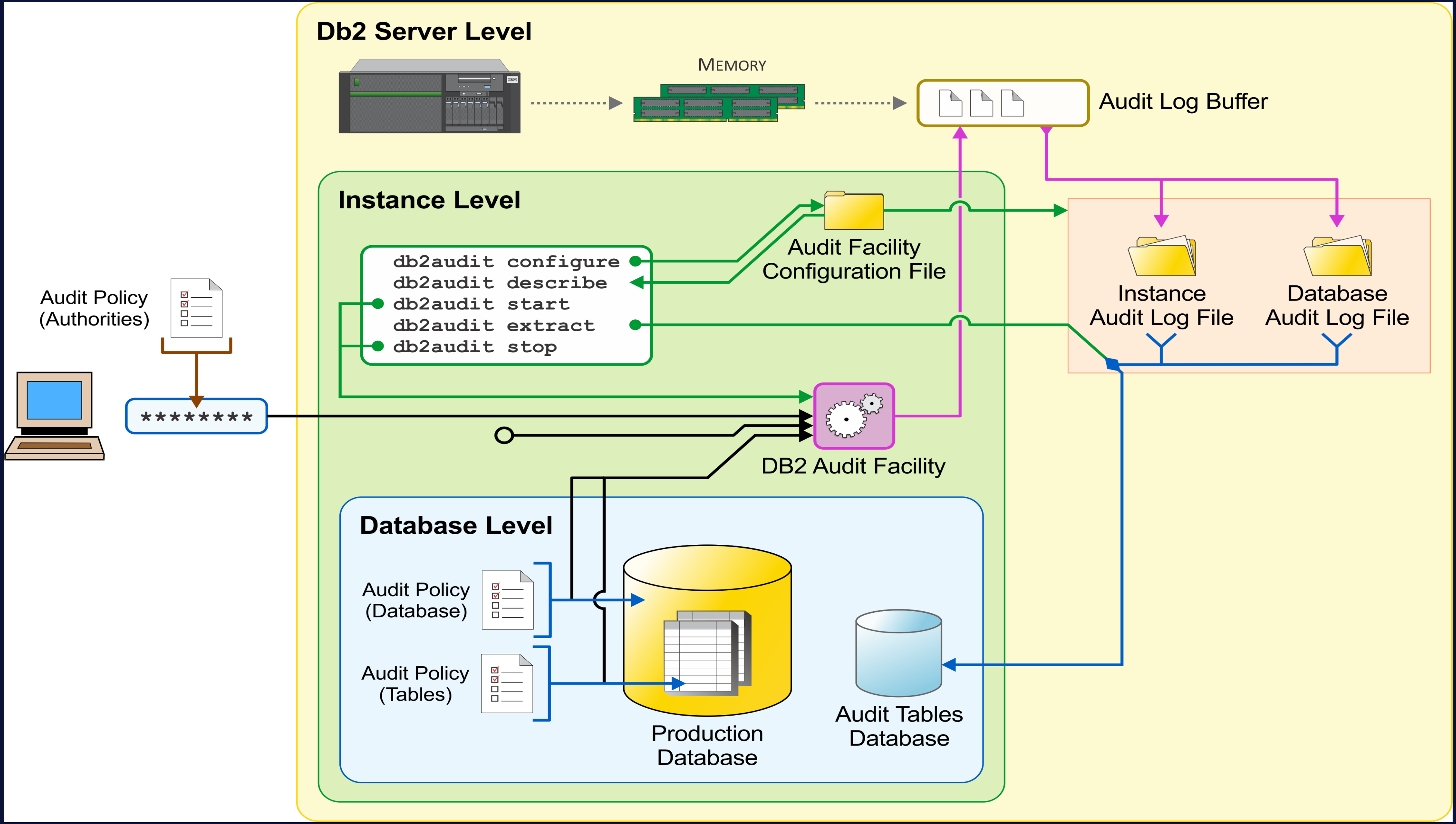
Granularity of database auditing

- Database audit is defined using audit policies which are then associated with specific objects using the AUDIT statement

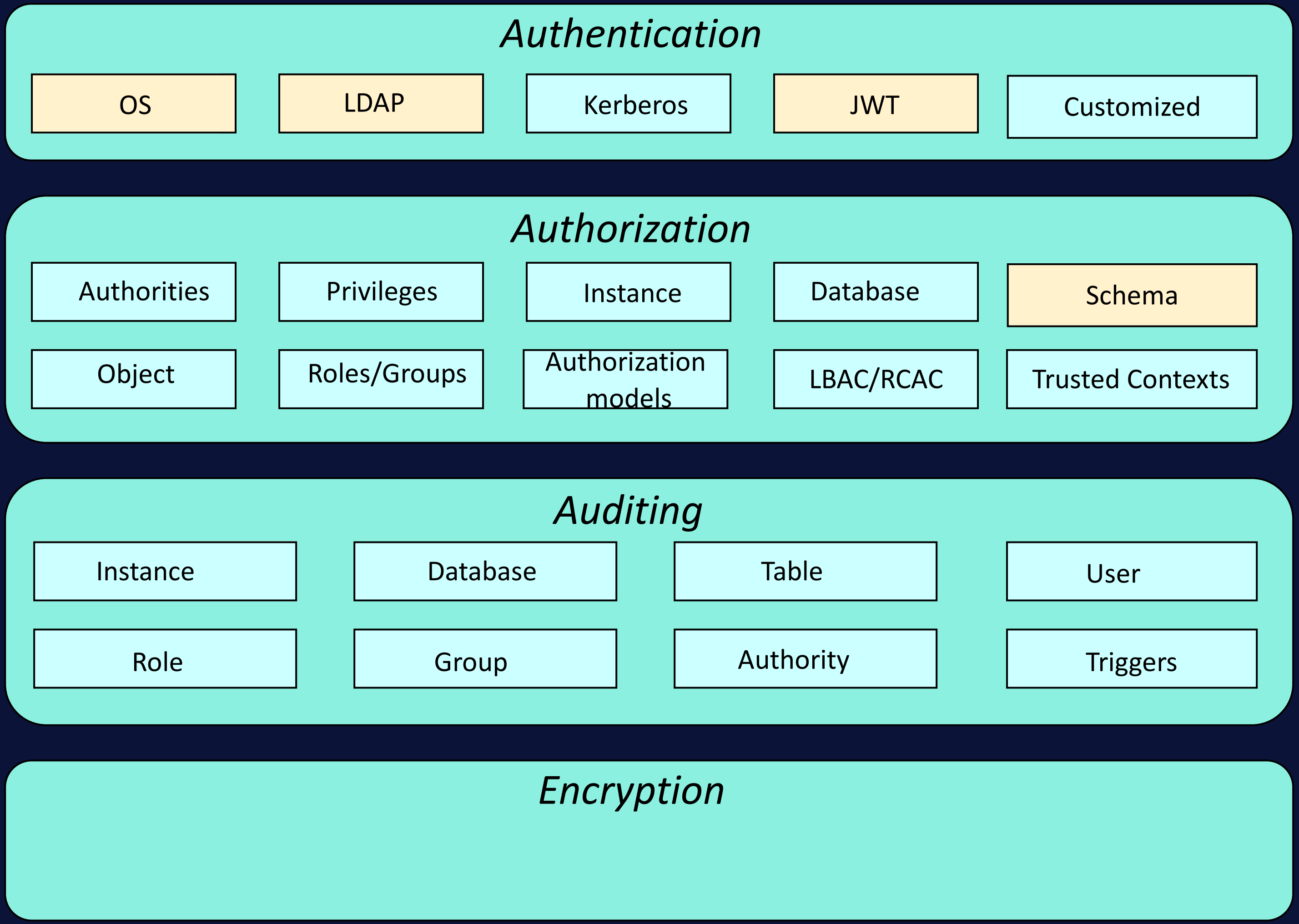
- Audit policies can be associated with different database objects to control what is audited
  - The database itself
  - Tables
  - Authorities such as SYSADM, DBADM, and SECADM
  - Users and groups
  - Roles
  - Trusted Connections

- This granularity allows a narrowed focus to be applied on for audit
  - ➢Can result in significant reductions in the amount of audit data

The Audit Facility – Illustrated



Database security landscape



Encryption

Two focus areas

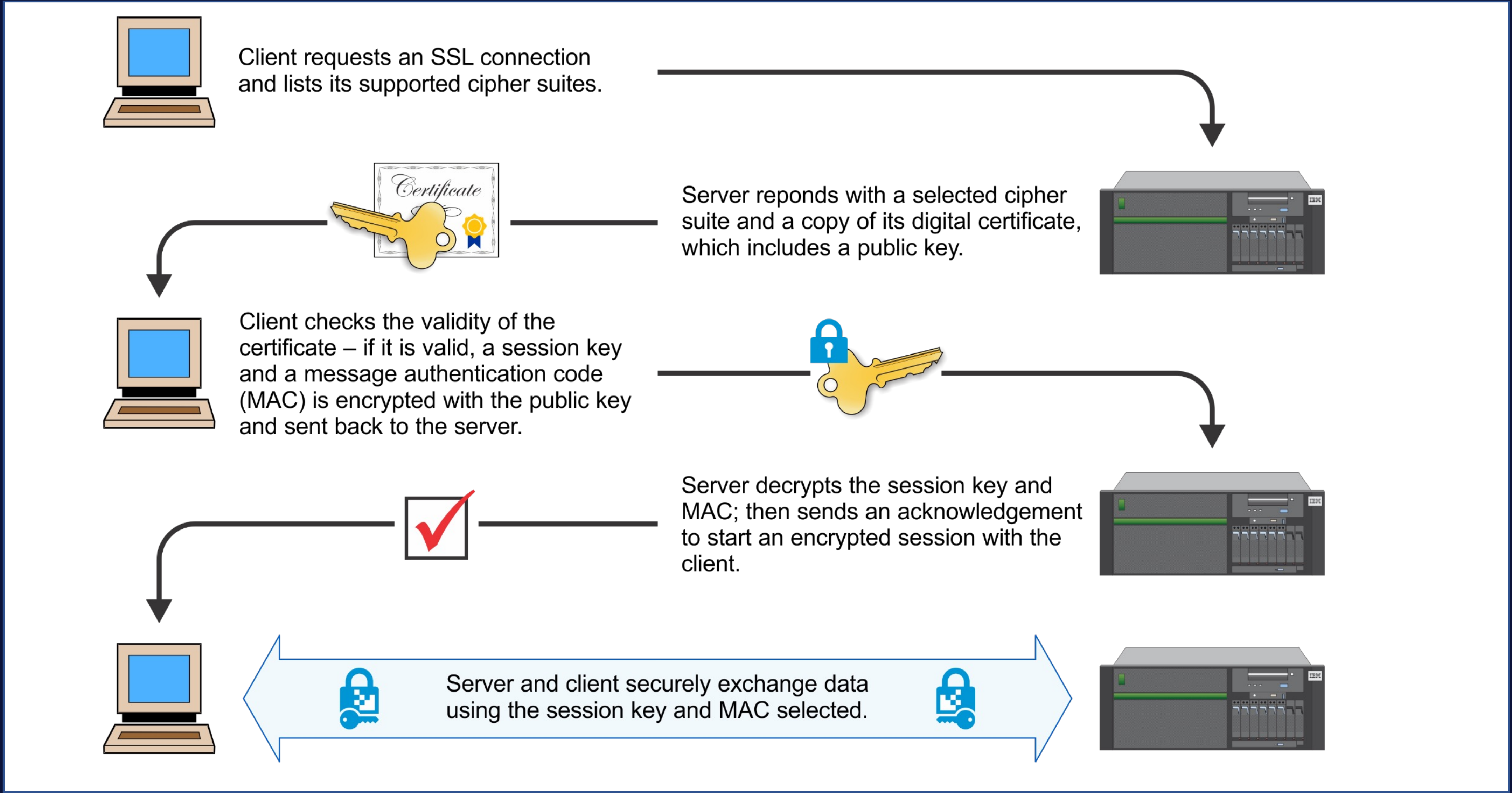
- Guarding communications → Data in transit
- Guarding database storage → Data at rest

Data in transit

Guarding communications

- Where is it relevant?
  - Between client and server
  - Between HADR primary and standby(s)
  - Between Db2 and external products or services
    - e.g., security products, remote storage repositories
- How is it done?
  - Encrypting the data being transmitted using a mechanism generically referred to as SSL (especially in Db2 documentation) or TLS
    - Current industry standard is TLS 1.2
    - SSL = Secure Sockets Layer
    - TLS = Transport Layer Security

“SSL handshake”



Client requests an SSL connection and lists its supported cipher suites.

Server reponds with a selected cipher suite and a copy of its digital certificate, which includes a public key.

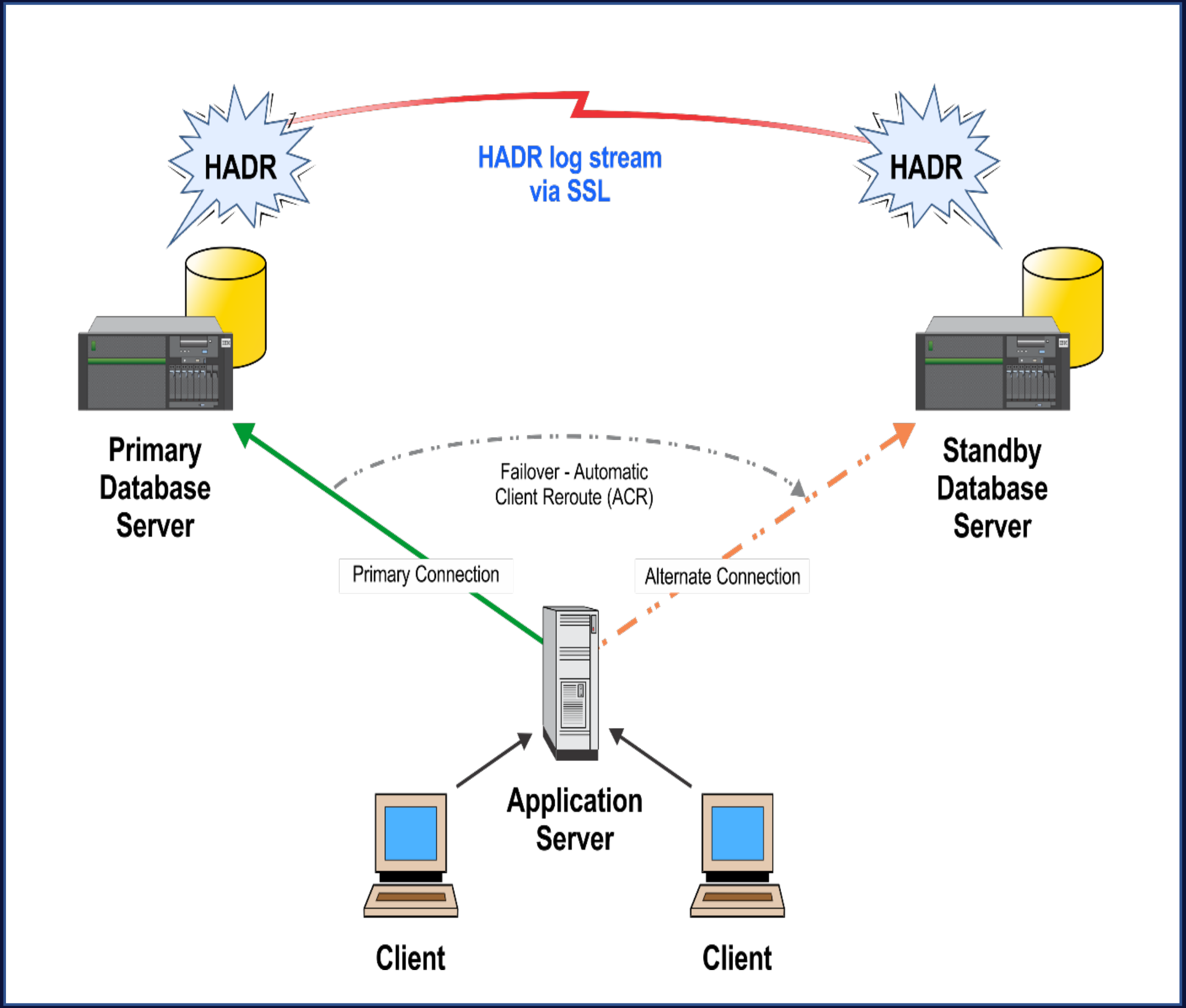
Client checks the validity of the certificate – if it is valid, a session key and a message authentication code (MAC) is encrypted with the public key and sent back to the server.

Server decrypts the session key and MAC; then sends an acknowledgement to start an encrypted session with the client.

Server and client securely exchange data using the session key and MAC selected.

SSL between HADR Primary and Standby servers

- Provides integrated protection of sensitive data in the log stream
- Enabled via the HADR_SSL_LABEL database configuration parameter
- Supports all HADR synchronization modes
- Supports multiple standbys



Remember SSL certificates expire!

- Certificates have an expiration date associated with them
  - ➢ Once that date (and time) has passed, SSL negotiation will fail

- You will need to update the certificates being used by Db2 in the affected area(s)
  - Client/server
  - HADR
  - Keystore

- In some of these, a restart of DB2 will be required
  - https://www.ibm.com/support/pages/do-we-need-restartrecyle-db2-after-revisingrenewing-ssl-certificate
  - ➢ As of Db2 11.5.2.0, the SSL_SVR_LABEL database manager parameter can now be updated dynamically (client/server SSL)

Recent changes to TLS features

- 11.5.8:
  - TLS 1.3 support
  - TLS data exposed in MON_GET_CONNECTION
    - TLS version negotiated
    - TLS ciphers negotiated
- 11.5.6
  - Hostname validation
    - Ability to configure client to validate the hostname in server certificate matches the hostname client is connecting to (prevents person-in-the-middle attack)
  - Client TLSVersion configuration supported
  - Simplified SSL setup for CLP and embedded
  - SNI support from C based client (always there for Java)

Data at Rest

Guarding (on-disk) database storage

- Where is it relevant?
  - Files containing user data such as database files, backup images, and transaction logs

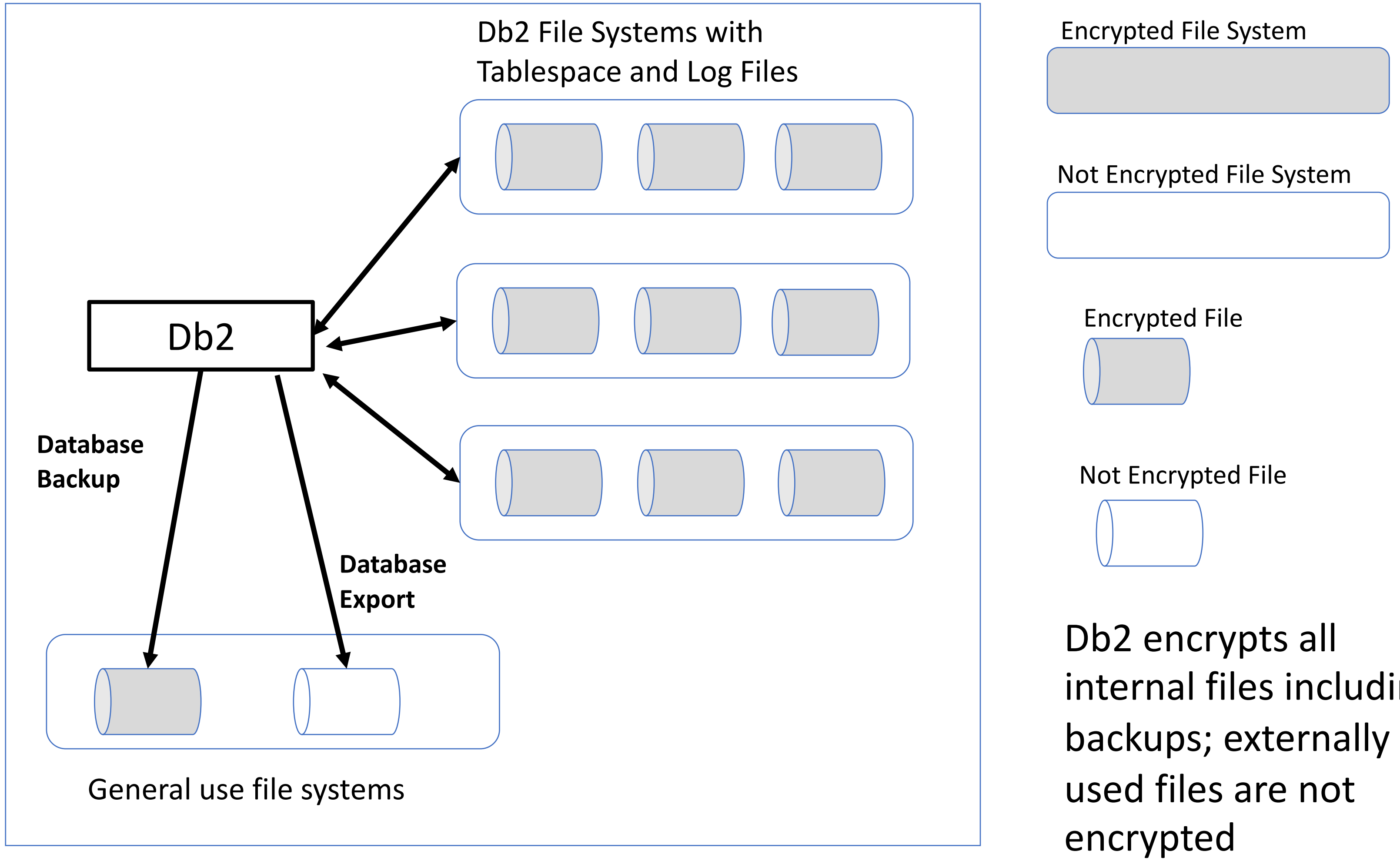
- How is it done?
  - File system encryption such as with Guardium Data Encryption (GDE)
  - Db2 native encryption

Db2 Encryption

- Db2 Native Encryption (naturally) encrypts Db2 databases only
  - Encrypts your data as it is written to disk.
  - The encryption is implemented within Db2 itself.

- Db2 encrypts all internal files including backups but externally used files are not encrypted

Db2 Encryption Only



Db2 encrypts all internal files including backups; externally used files are not encrypted

Db2 native encryption

- Db2 Native Encryption is built into Db2 to protect data when it is at rest
- Available in all Db2 offerings free of charge
- Automatically detects and uses CPU hardware acceleration when available
    - Intel AES-NI hardware acceleration
    - Power8 in-core support for the Advanced Encryption Standard (AES)

Highlights of Db2 Native Encryption

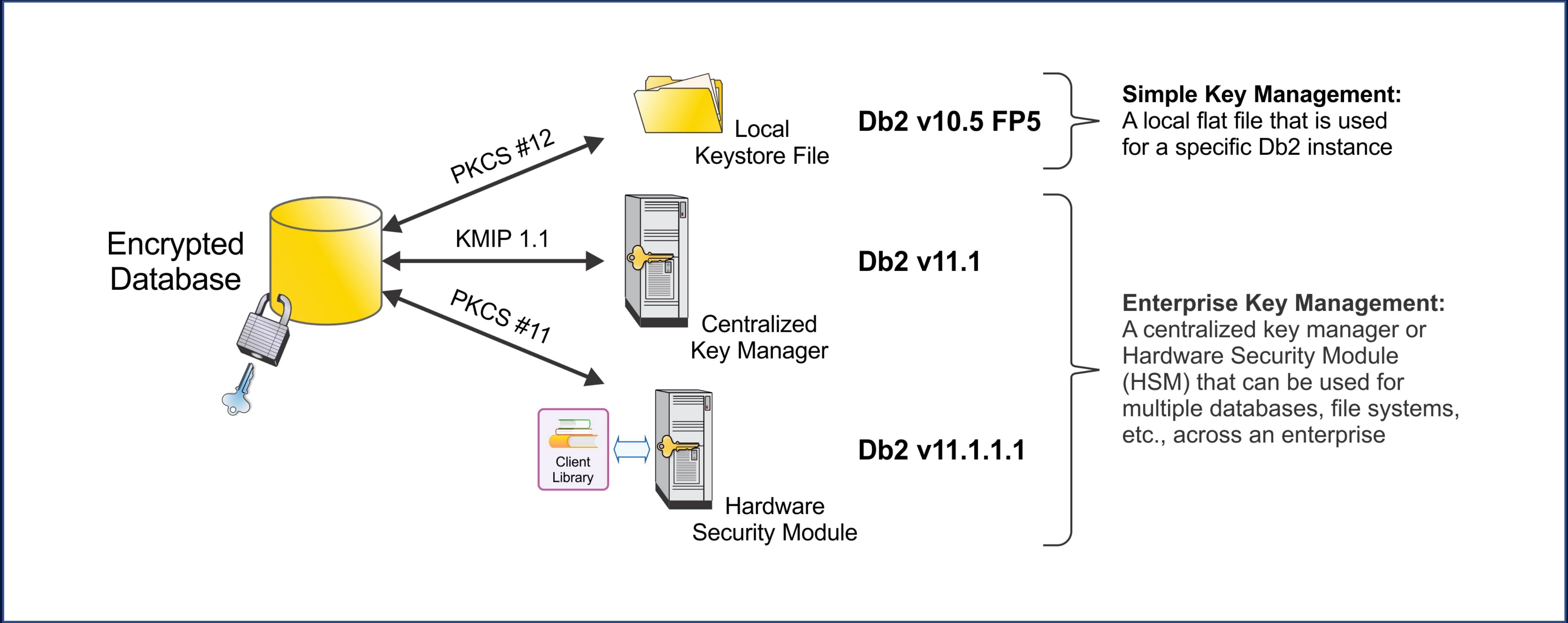
- Easy to deploy and works on all Db2 platforms
  - Transparent to applications!
  - No system administrator needed!
- Industrial strength
  - FIPS 140-2 certified encryption libraries
  - NIST compliant use of cryptography (e.g., NIST SP 800-131)
- Secure and transparent key management
- Encrypts all critical data

Basics of key management

- Db2 Native Encryption uses an industry standard 2-tier model
  - Actual data is encrypted with a Data Encryption Key (DEK)
  - DEK is encrypted with a Master Key (MK)

- DEK is managed within the database while the MK is stored externally in a keystore

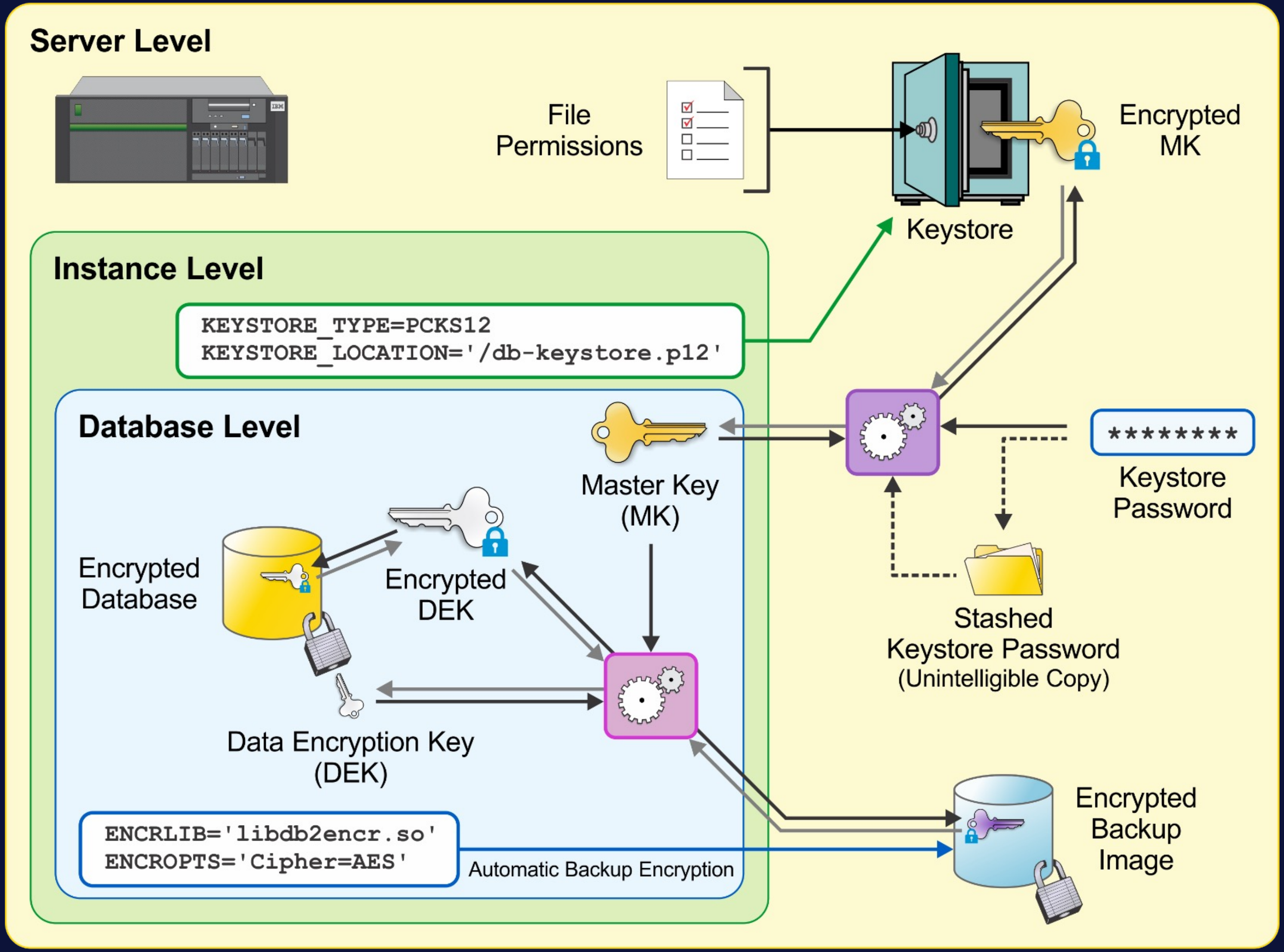
Keystore options



Simple Key Management:
A local flat file that is used for a specific Db2 instance

Enterprise Key Management:
A centralized key manager or Hardware Security Module (HSM) that can be used for multiple databases, file systems, etc., across an enterprise

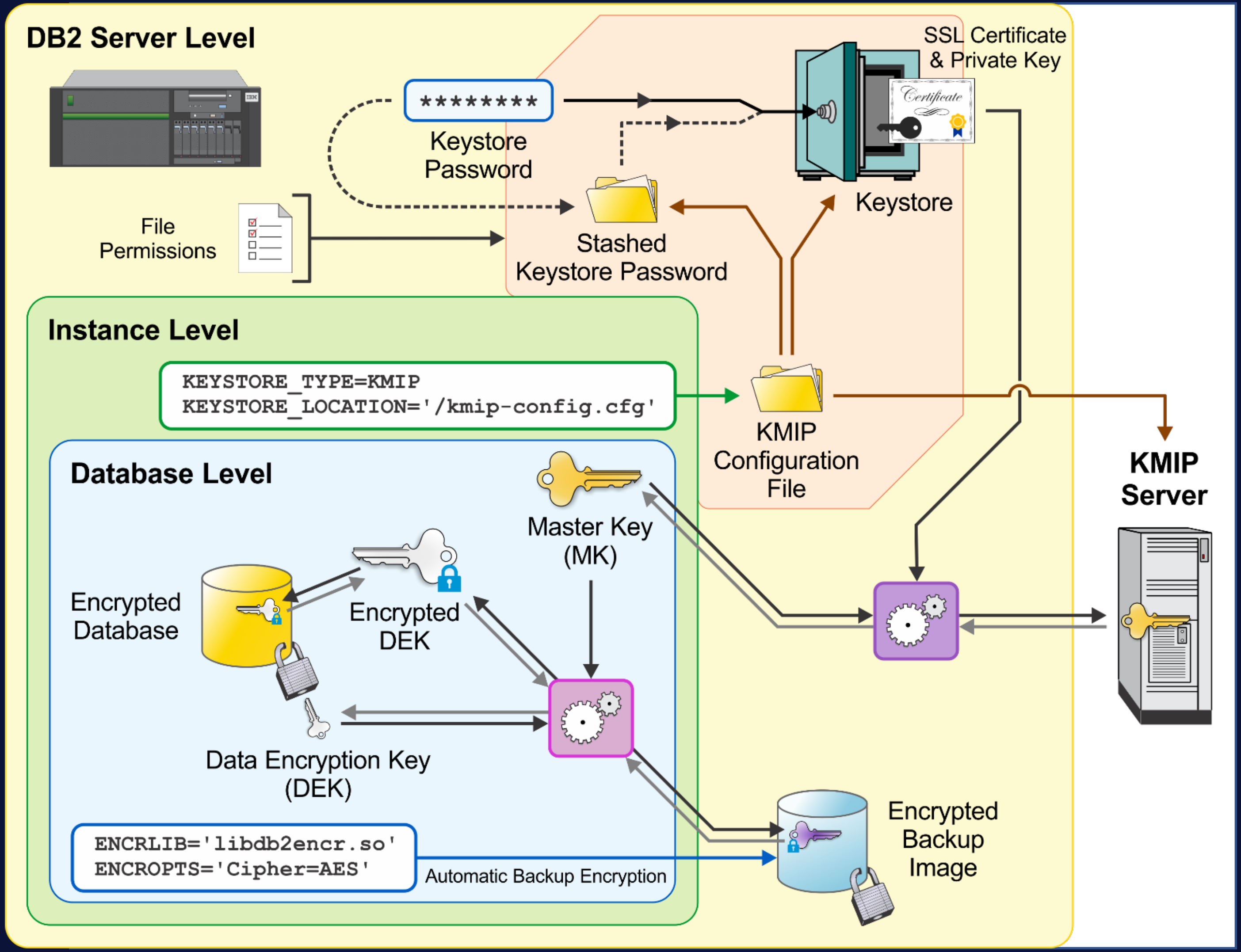
Local PKCS#12 keystore



```
KEYSTORE_TYPE=PCKS12
KEYSTORE_LOCATION='/db-keystore.p12'
```

```
ENCRLIB='libdb2encr.so'
ENCROPTS='Cipher=AES'
```

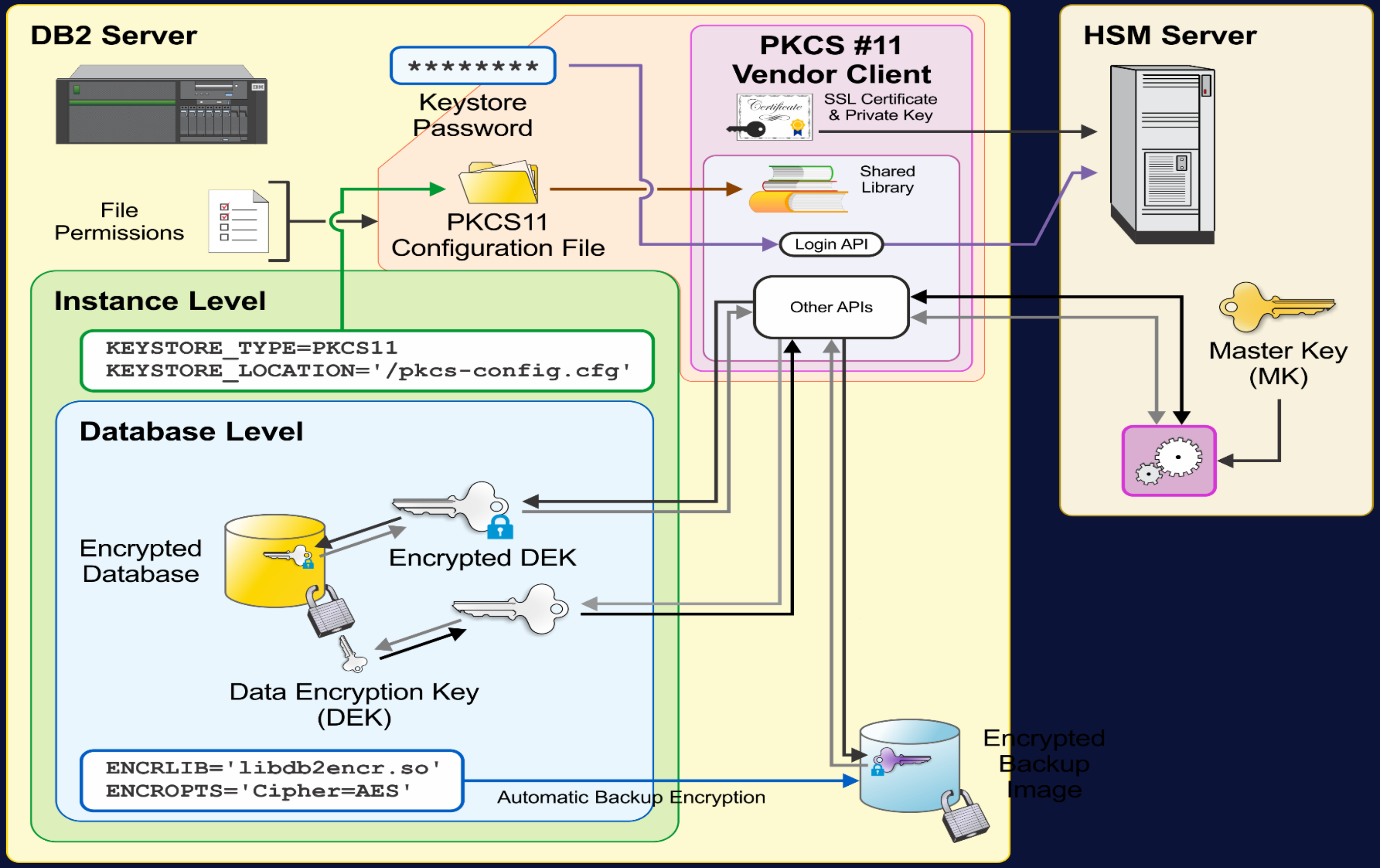
Centralized key manager



```
KEYSTORE_TYPE=KMIP
KEYSTORE_LOCATION='/kmip-config.cfg'
```

```
ENCRLIB='libdb2encr.so'
ENCROPTS='Cipher=AES'
```

Hardware Security Module (HSM)



```
KEYSTORE_TYPE=PKCS11
KEYSTORE_LOCATION='/pkcs-config.cfg'
```

```
ENCRLIB='libdb2encr.so'
ENCROPTS='Cipher=AES'
```

Considering encryption?

- Encryption is not something to rush into as it has implications for availability, operations, and performance!

- Availability:
  - https://www.ibm.com/docs/en/db2/11.5?topic=considerations-keystore-availability-recoverability
  - ➢ Keystore availability issues now become data availability issues

- Operations:
  - https://www.ibm.com/docs/en/db2/11.5?topic=considerations-impact-encryption-database-operations

- Performance:
  - https://www.ibm.com/docs/en/db2/11.5?topic=considerations-impact-encryption-performance
  - ➢ CPU hardware acceleration is critical
  - ➢ You should plan on completely re-tuning a newly encrypted system

Questions?