



DB2 database security

IBM Information Management Cloud Computing Center of Competence
IBM Canada Labs

Agenda

- DB2 security overview
- Authentication
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context

Supporting reading material & videos

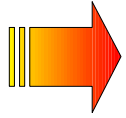
- **Reading materials**

- Getting started with DB2 Express-C eBook
 - Chapter 10: Database security

- **Videos**

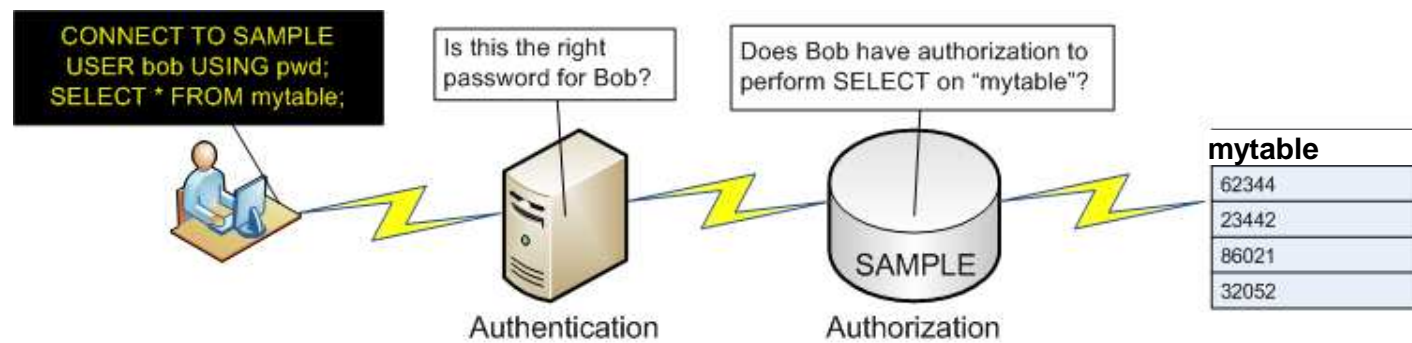
- db2university.com course AA001EN
 - Lesson 9: Database security

Agenda



- DB2 security overview
 - Authentication
 - Authorization
 - Roles
 - The PUBLIC group
 - Granular access through views
 - Label-based access control
 - Extended Security (Windows only)
 - Trusted context

DB2 security overview



DB2 security has two steps:

■ Authentication (External to DB2)

- Checks user name and password
- Done by security plug-in that invokes a facility outside of DB2 (Default is the operating system)

■ Authorization (Performed by DB2)

- Check if authenticated user may perform requested operation
- Done by DB2 using information stored in the DB2 catalog, DBM configuration file

05/24/2011

Template Documentation

5

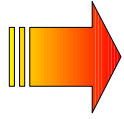
© 2011 IBM Corporation

This slide shows that security in DB2 is performed in two steps: The first step is called "**authentication**" and is normally performed by the operating system, or an external security facility outside of DB2. The second step is called "**authorization**" and that part is performed by DB2. For example, at the top, we see user Bob issuing a connect to the SAMPLE database, and he's passing "*USER bob using pwd*", where "pwd" is the password. When he issues this connect statement, the statement will first go to DB2, it will check the value of the dbm cfg parameter *authentication* at the server, and depending on the value of that parameter, it will request the operating system of the client or server machines (or external security facility) to perform authentication.

So the authentication, again, is not performed by DB2, but normally by the operating system, or an external facility, or even you can write your own code to perform the authentication. This is different from other relational database management systems, because these other systems normally can define users within their database systems, but in DB2 you really cannot define a user within DB2. The users are defined outside DB2; the same thing with groups. They will be defined outside of DB2. Normally this would be operating system users or operating system groups as well. This approach avoids having too many user ids/passwords defined at different levels.

Once the operating system verifies there is a user "bob" with password "pwd", it will tell DB2 that, yes, this person is OK to connect. After that, DB2 takes over with the "**authorization**" part (2nd step), where DB2 checks if the user can for example issue a: **select * from mytable**. When the users start executing operations in the database, DB2 will check, in this case, that user bob has "select" privilege and he can do it on the table, "mytable".

Agenda



- DB2 security overview
- **Authentication**
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context

Authentication

- Where is authentication performed (userID/psw checking)?
- DBM CFG parameter AUTHENTICATION (at the DB2 server) determines where/how authentication is performed
- Some valid values for AUTHENTICATION parameter are:

| Value | Description |
|-------------------------|--|
| SERVER (default) | Authorization takes place on the server |
| CLIENT | Authorization takes place on the client |
| SERVER_ENCRYPT | Like SERVER except user IDs and passwords are encrypted |
| DATA_ENCRYPT | Like SERVER_ENCRYPT but data is also encrypted |
| KERBEROS | Authentication takes place using a Kerberos security mechanism |
| GSSPLUGIN | Authentication uses an external GSS API-based plug-in security mechanism |

05/24/2011

Template Documentation

7

© 2011 IBM Corporation

As indicated earlier, DB2 does not perform authentication; however, it does have a dbm cfg parameter called "AUTHENTICATION" to specify where/how is authentication to be performed.

The table shows different values for this parameter.

SERVER is the default

SERVER_ENCRYPT is like SERVER but the password is encrypted through the network

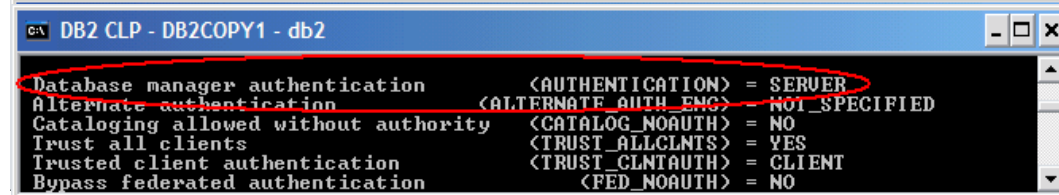
DATA_ENCRYPT is like SERVER_ENCRYPT but it also encrypts the data through the network

GSSPLUGIN is what you need to select if you want to plug-in your code so that it performs authentication.

Configuring Authentication at the DB2 server

Example:

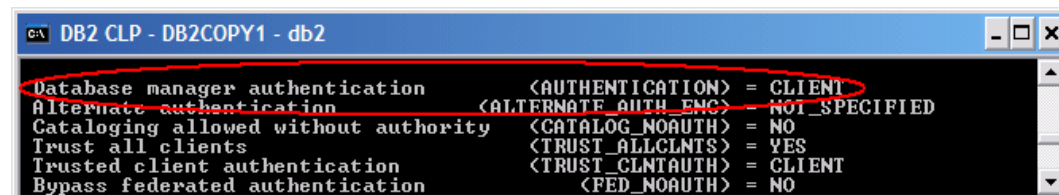
```
db2 "GET DBM CFG"
```



```
CA DB2 CLP - DB2COPY1 - db2
Database manager authentication      <AUTHENTICATION> = SERVER
Alternate authentication            <ALTERNATE_AUTH_ENG> = NOT_SPECIFIED
Cataloging allowed without authority <CATALOG_NOAUTH> = NO
Trust all clients                  <TRUST_ALLCLNTS> = YES
Trusted client authentication      <TRUST_CLNTAUTH> = CLIENT
Bypass federated authentication    <FED_NOAUTH> = NO
```

Changing to have the authentication done at the client:

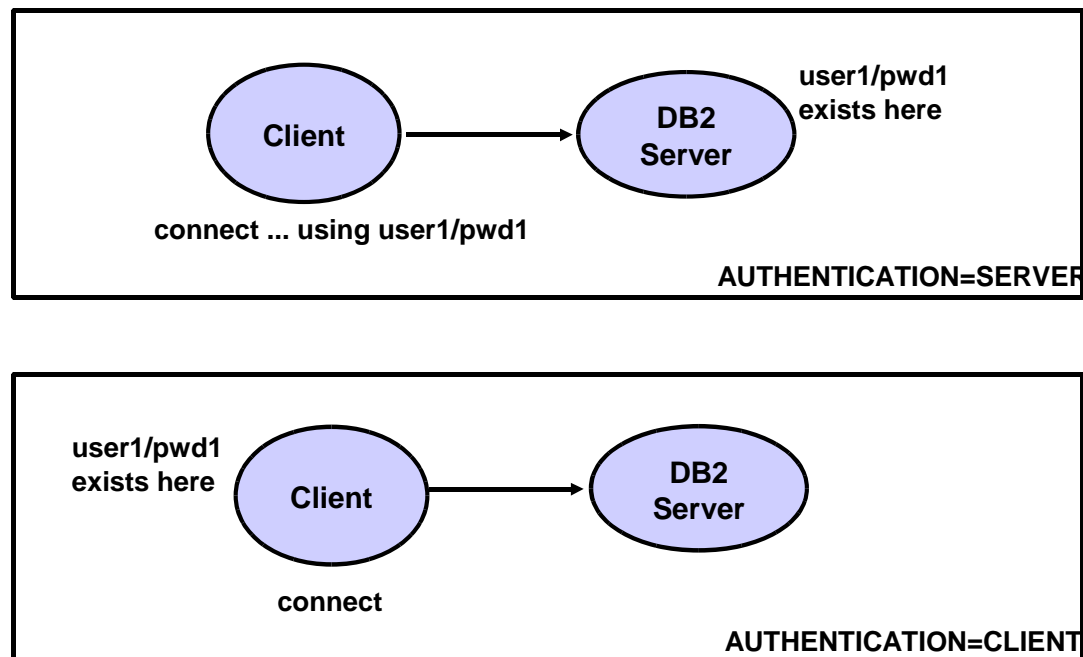
```
db2 "UPDATE DBM CFG USING AUTHENTICATION CLIENT"
```



```
CA DB2 CLP - DB2COPY1 - db2
Database manager authentication      <AUTHENTICATION> = CLIENT
Alternate authentication            <ALTERNATE_AUTH_ENG> = NOT_SPECIFIED
Cataloging allowed without authority <CATALOG_NOAUTH> = NO
Trust all clients                  <TRUST_ALLCLNTS> = YES
Trusted client authentication      <TRUST_CLNTAUTH> = CLIENT
Bypass federated authentication    <FED_NOAUTH> = NO
```

This provides an example of how to review the contents of the AUTHENTICATION parameter, and then of how to update it.

Authentication (cont'd)



05/24/2011

Template Documentation

9

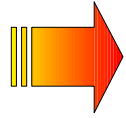
© 2011 IBM Corporation

The figure again shows where authentication will actually happen based on the value of the dbm cfg AUTHENTICATION parameter.

If AUTHENTICATION is set to server, the user “user1” with password “pwd1” need to be defined at the operating system (or external security facility) in the server

If AUTHENTICATION is set to client, the user “user1” with password “pwd1” need to be defined at the operating system (or external security facility) in the client. There are more parameters that need to be set up when setting AUTHENTICATION to Client which are not discussed in this presentation.

Agenda



- DB2 security overview
- Authentication
- **Authorization**
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context

Authorization

- **Verifies if an authorization ID has sufficient privileges to perform a desired database operation**

Eg: Does Bob have SELECT privilege on table MYTABLE?

- **Can be assigned using:**

- **Privileges:**

- Granular

- **Authorities:**

- Built-in. Eg: SYSADM, DBADM, SECADM, etc

- **Roles:**

- Like user-defined authorities

05/24/2011

Template Documentation

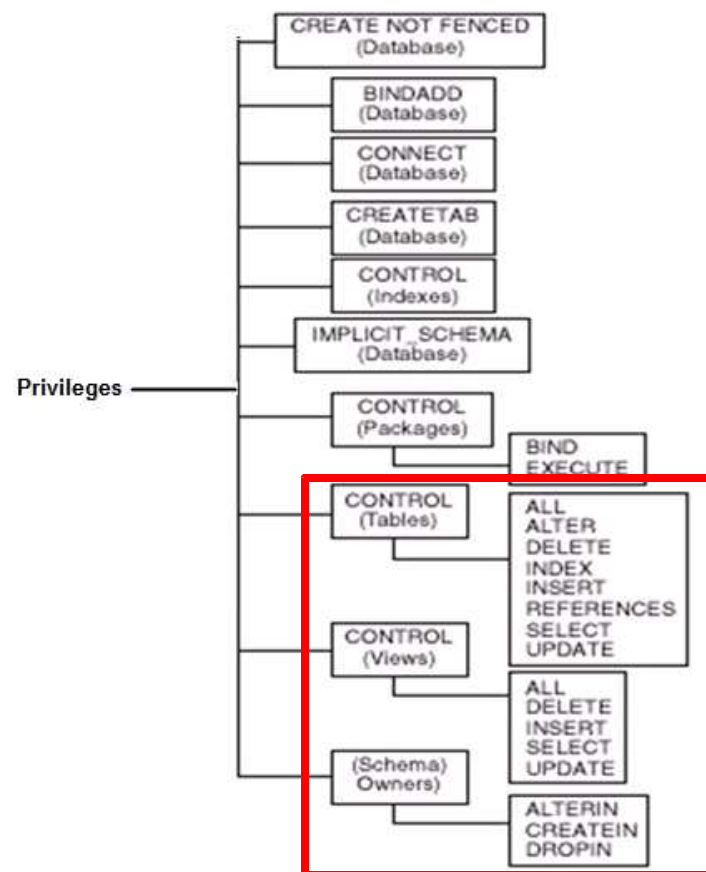
11

© 2011 IBM Corporation

This is the second part that was described in the security overview slide. In this part, DB2 performs the security checks, for example, “Does user Bob have authorization to SELECT on table mytable?”

Within the authorization topic, we will talk about privileges, authorities and roles in more detail in the next few slides.

Privileges



05/24/2011

12

Template Documentation

© 2011 IBM Corporation

Let's start with the first item within the authorization topic: Privileges. This figure shows you a list of different privileges that you can grant to a user or group. Privileges provide granular security where you can give a user just access to a given database object for a given operation. There are many privileges, so we are highlighting with a rectangle some privileges we will discuss in more detail in the next slide.

Privileges - Examples

▪ Schema privileges

- CREATEIN: can create objects within the schema
- ALTERIN: can alter objects within the schema
- DROPIN: can drop objects from within the schema

▪ Table and View privileges

- CONTROL: Full control on a table or view including drop, grant/revoke
 - DELETE: can delete rows
 - INSERT: can insert rows and run the IMPORT utility.
 - SELECT: can retrieve rows, create a view, run the EXPORT utility.
 - UPDATE: can change an entry in a column, table or view
 - ALTER: can modify a table
 - INDEX: can create an index on a table
 - REFERENCES can create and drop a foreign key

05/24/2011

Template Documentation

13

© 2011 IBM Corporation

This slide shows an example of different privileges at the SCHEMA and then TABLE and VIEW levels, and a brief description of each. Note that if you grant CONTROL privilege to a user or group, you grant not only the ability to drop, or grant this privilege to others, but also you are implicitly granting all the other privileges listed under such as DELETE, INSERT, SELECT, etc.

Privileges – Granting / Revoking

▪ Explicit

- Using explicitly the GRANT and REVOKE statements for a user or group

```
grant select on table db2inst1.employee to user mary
revoke select on table db2inst1.employee from user mary
```

▪ Implicit

- DB2 may grant privileges automatically when certain commands are issued

```
create table mytable
```

User automatically gains full access to the table

▪ Indirect

- Packages contain SQL statements in an executable format. The user only requires EXECUTE privilege to run them
- Example: package1 contains the following static SQL statements

```
select * from test
insert into test values (1,2,3)
```

- In this case a user with EXECUTE privilege on package1 is indirectly granted SELECT and INSERT privilege on table TEST

05/24/2011

Template Documentation

14

© 2011 IBM Corporation

To assign a privilege to a user, you can do it in 3 ways:

- Explicit
- Implicit
- Indirect

Something that may not be that intuitive occurs when you revoke a privilege which implicitly provided other privileges. When you revoke the privilege, the implicitly obtained ones will not be automatically revoked. You need to explicitly revoke them.

Eg:

db2 grant control on table employee to user raul

==> Grants control, and also all privileges like, ALTER, DELETE, INDEX, etc (see slide 12)

db2 revoke control on table employee from user raul

==> Only revokes CONTROL. The other privileges like ALTER, DELETE, etc, remain

After revoking CONTROL, you must issue explicitly:

db2 revoke all privileges on table employee from user raul

to ensure the implicitly granted privileges are revoked.

(Demo)

Privileges – Granting / Revoking (Examples)

```
GRANT  SELECT  ON  TABLE  T1  TO  USER  user1
GRANT  ALL      ON  TABLE  T1  TO  GROUP group1
REVOKE ALL      ON  TABLE  T1  FROM GROUP group1
GRANT  EXECUTE  ON  PROCEDURE p1  TO  USER  user1
REVOKE EXECUTE  ON  PROCEDURE p1  FROM USER  user1
REVOKE CONNECT  ON  DATABASE      FROM USER  user2
```

05/24/2011

Template Documentation

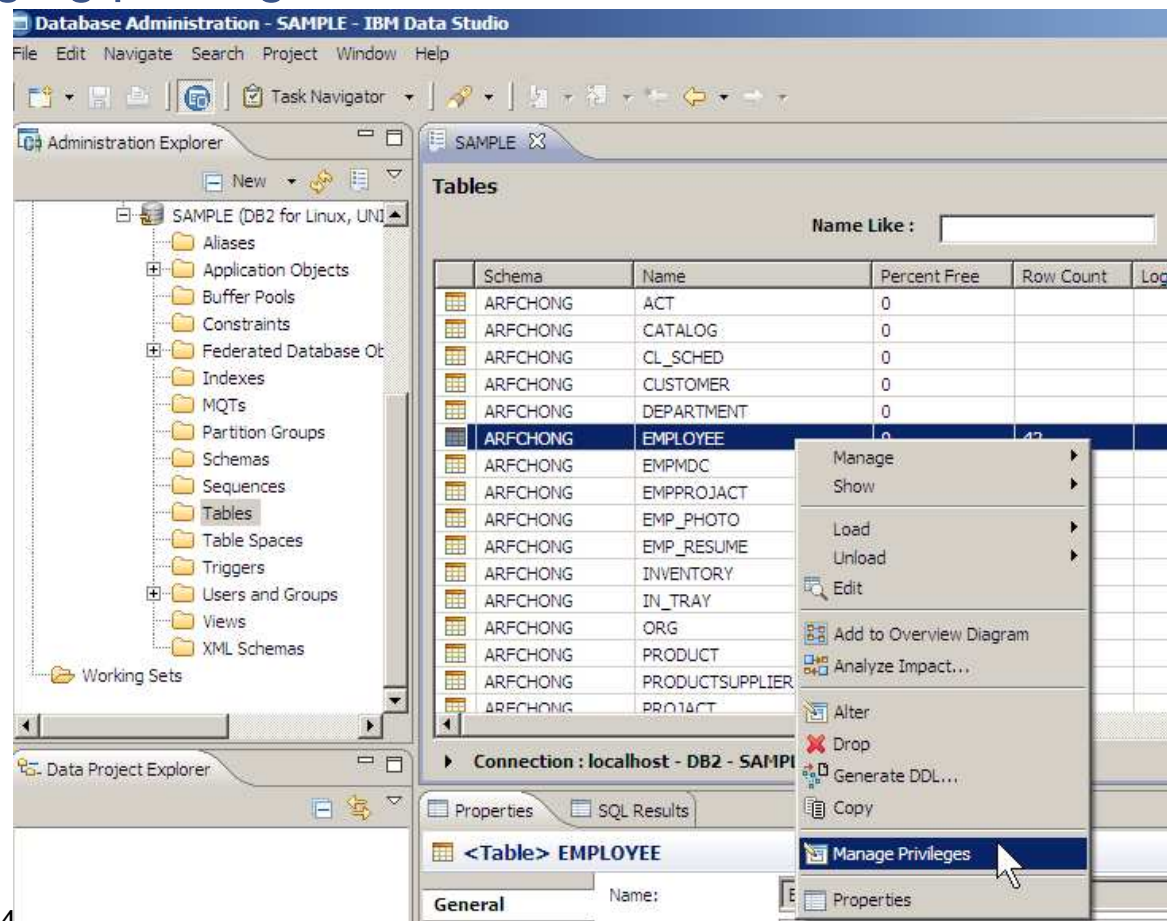
15

© 2011 IBM Corporation

These are some examples of granting explicitly some privileges to a user or a group

Note the last example does not provide the name of the database. To run the GRANT/REVOKE statements, it is assumed you are connected to a database; therefore, if you want to revoke CONNECT privilege to the database, you don't need to specify the name of the database since you are already connected to it when issuing the statement.

Managing privileges in Data Studio



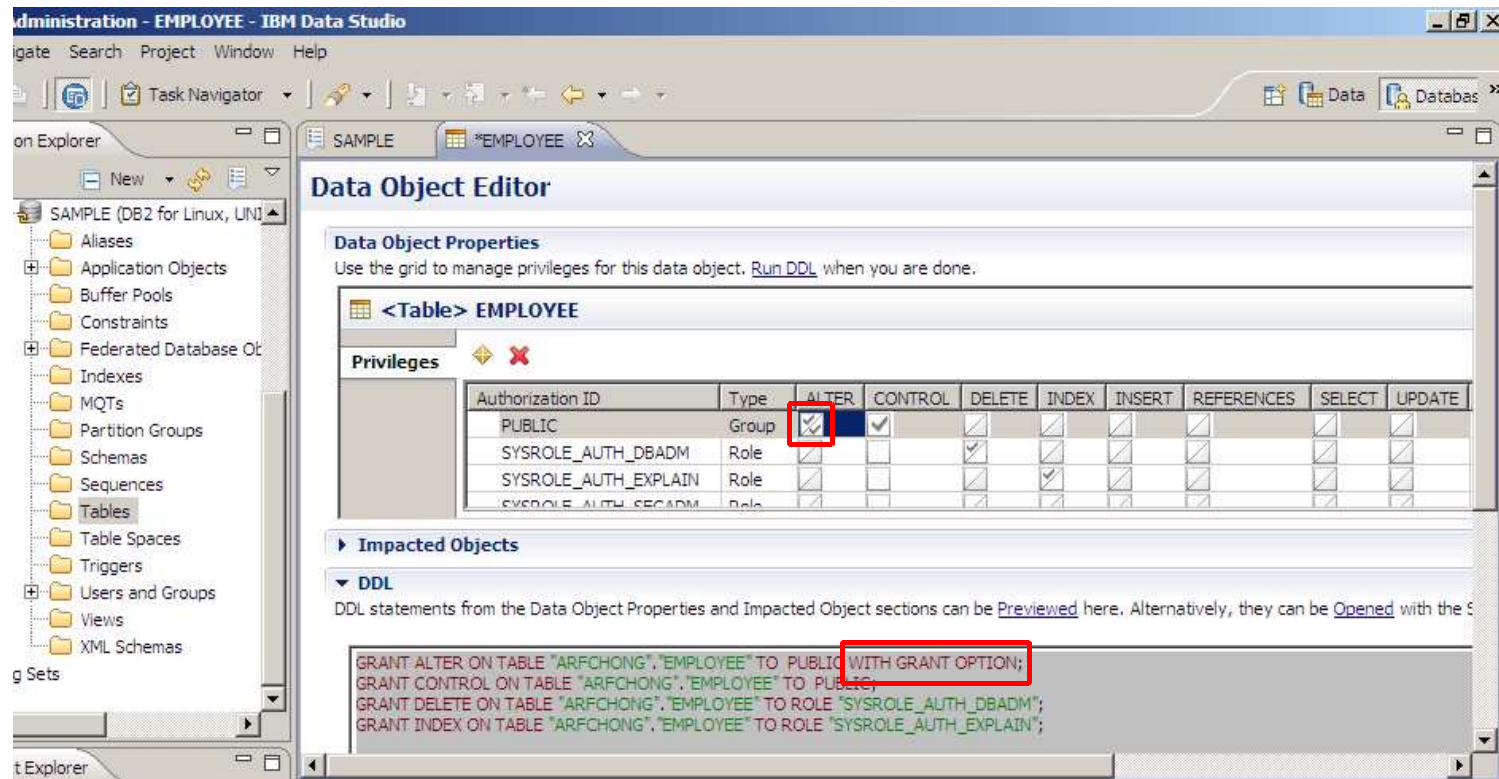
05/24

16

© 2011 IBM Corporation

In Data Studio, right-click on the table and choose “Manage Privileges”

Managing privileges in Data Studio



05/24/2011

Template Documentation

17

© 2011 IBM Corporation

In Data Studio, when you see two arrows as highlighted, it means that the user who is getting this privilege will also be able to grant it to other users.

Behind the scenes, Data Studio is generating GRANT/REVOKE statements, and two arrows would mean that the "WITH GRANT OPTION" will be included in the syntax.

(Demo)

Authorities

- **Instance-level Authorities**

- SYSADM, SYSCTRL, SYSMANT, SYSMON

- **Database-level Authorities**

- DBADM, SECADM, SQLADM, WLMADM, EXPLAIN, ACCESSCTRL, DATAACCESS, etc

05/24/2011

18

Template Documentation

© 2011 IBM Corporation

We looked at privileges, now let's take a look at the second item in the authorization topic: authorities.

Different authorities are “built-in” (they come with DB2) and can be instance-level, or database-level authorities.

You can think of an authority as a way to group privileges together and more.

So, for Instance-level Authorities we have SYSADM, SYSCTRL, SYSMANT, SYSMON

For Database-level Authorities we have DBADM, SECADM, SQLADM, WLMADM, EXPLAIN, ACCESSCTRL, DATAACCESS, etc

Instance-level authorities

| Authority | Description |
|-----------|---|
| SYSADM | Manages the instance as a whole |
| SYSCTRL | Administers a database manager instance |
| SYSMAINT | Maintains databases within an instance |
| SYSMON | Monitors the instance and its databases |

05/24/2011

Template Documentation

19

© 2011 IBM Corporation

This table provides a quick description of what each authority does. As we will see later, having any of these authorities does not imply they will have access to the data stored in databases. For example, SYSADM may not be able to see the data of your databases. In previous version of DB2, SYSADM had full power and could do anything, but starting with DB2 9.7, this is no longer the case.

Instance-level authorities

- SYSADM, SYSCTRL, SYSMANT & SYSMON are defined by operating system groups in DBM CFG:

```
update dbm cfg using SYSCTRL_GROUP <group_name>
update dbm cfg using SYSMANT_GROUP <group_name>
update dbm cfg using SYSADM_GROUP <group_name>
update dbm cfg using SYSMON_GROUP <group_name>
```

- Each instance has its own authority group definitions
- On Windows, if these values are blank, the local Windows Administrators group, LocalSystem account, and DBADMNS group (if Extended Security is enabled) would be SYSADM
- On Linux/UNIX, the group of the instance owner would be SYSADM

05/24/2011

Template Documentation

20

© 2011 IBM Corporation

To assign SYSADM or any other instance-level authority to a group, you do it in the dbm cfg as shown in this slide. The group would be defined at the operating system (or external security facility) level.

If the parameters shown in this slide are left blank, which is the default, then the LOCAL ADMINISTRATOR group on Windows, or the group that the DB2 instance owner (eg: db2inst1) belongs to on Linux would be the SYSADM.

Instance-level authorities - SYSADM

- **Highest level of administrative authority at the instance level**
- **Only a user with SYSADM authority can:**
 - Upgrade and restore a database
 - Change the database manager configuration file including specifying the groups having SYSADM, SYSCTRL, SYSMANT, or SYSMON authority
- **Does not implicit get DBADM authority, and does not automatically have access to data**
- **Specified by the `sysadm_group` parameter in the DBM CFG**
- **Example: Granting SYSADM authority to the group 'mygrp':**

```
UPDATE DBM CFG USING SYSADM_GROUP mygrp
```

05/24/2011

21

Template Documentation

© 2011 IBM Corporation

This slide provides more details about SYSADM.

Getting SYSADM authority doesn't mean the user will also become DBADM. SYSADM also doesn't necessarily have access to the data as mentioned earlier. SECADM is required to grant this access as we will see.

To grant SYSADM, you need to update the DBM CFG where an operating system group must be specified. Every user in that group would become SYSADM.

Instance-level authorities

SYSADM

- Update and restore a database manager configuration parameters (DBM CFG) including specifying groups that have SYSADM, SYSCTRL, SYSMMAINT AND SYSMON
- Grant and revoke table space privileges
- Upgrade and restore a database

SYSCTRL

- Update a database, node, or distributed connection services (DCS) directory
- Restore to a new or existing database
- Force users off the system
- Create or drop a database (NOTE: automatically gets DBADM authority)
- Create, drop, or alter a table space
- Restore to a new or existing database
- Use any table space

SYSMMAINT

- Back up a database or table space
- Restore to an existing database
- Roll forward recovery
- Start or stop an instance
- Restore or quiesce a table space, and query it's state
- Run tracing
- Database system monitor snapshots
- Reorganize tables
- Use RUNSTATS and update log history files

SYSMON

- GET DATABASE MANAGER MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET SNAPSHOT
- LIST commands: ACTIVE DATABASES, APPLICATIONS, DATABASE PARTITION GROUPS, DCS APPLICATIONS, PACKAGES, TABLES, TABLESPACE CONTAINERS, TABLESPACES, UTILITIES--
- RESET MONITOR
- UPDATE MONITOR SWITCHES
- APIs: db2GetSnapshot and db2GetSnapshotSize, db2MonitorSwitches, db2mtrk, db2ResetMonitor
- All snapshot table functions, without running SNAP_WRITE_FILE
- Can connect to a database



05/24/2011

22

© 2011 IBM Corporation

This chart, taken directly from the DB2 manuals provide a quick view of the hierarchy of the DB2 instance-level authorities, and some description of what they can do.

Database-level authorities

| Authority | Description |
|--|--|
| SECADM | Manages security within a database |
| DBADM | Administers a database |
|  ACCESSCTRL | Grants and revokes authorities and privileges (other than SECADM, DBADM, ACCESSCTRL, and DATAACCESS authority. Note that SECADM authority is required to grant and revoke these authorities) |
|  DATAACCESS | Provides the ability to access data in a database. |
| SQLADM | Monitors and tunes SQL queries |
| WLMADM | Manages workloads |
| EXPLAIN | Users who need to explain query plans (EXPLAIN authority does not give access to the data itself) |

05/24/2011

Template Documentation

23

© 2011 IBM Corporation

Let's now move on to “Database-level” authorities.

The table, taken from the “Getting started with DB2 Express-C” book lists the different database-level authorities and provides a description.

SECADM Authority

- SECADM is the Security Administrator for a given database
- Grants/revokes all security at the database level. No authority at instance level
- By default, the user creating the database is SECADM
- Cannot access data stored in user tables
- Can only be granted by a user with SECADM authority
- Can grant SYSADM the SECADM authority

05/24/2011

Template Documentation

24

© 2011 IBM Corporation

The SECADM (Security administrator) is the authority that administers security for a database. He cannot do anything at the instance level. By default, the SECADM is the one who creates the database. SECADM can grant SECADM authority and data access to others.

DBADM Authority

- Super user for a given database
 - No authority at instance level
 - May not have access to the data
 - May not be able to grant/revoke authorities/privileges
- SECADM authority needed to grant/revoke DBADM to a user, group or role. For example:

```
GRANT DBADM on database to user <userID>
```

- DBADM will also get these other authorities by default:
 - DATAACCESS
 - ACCESSCTRL

05/24/2011

Template Documentation

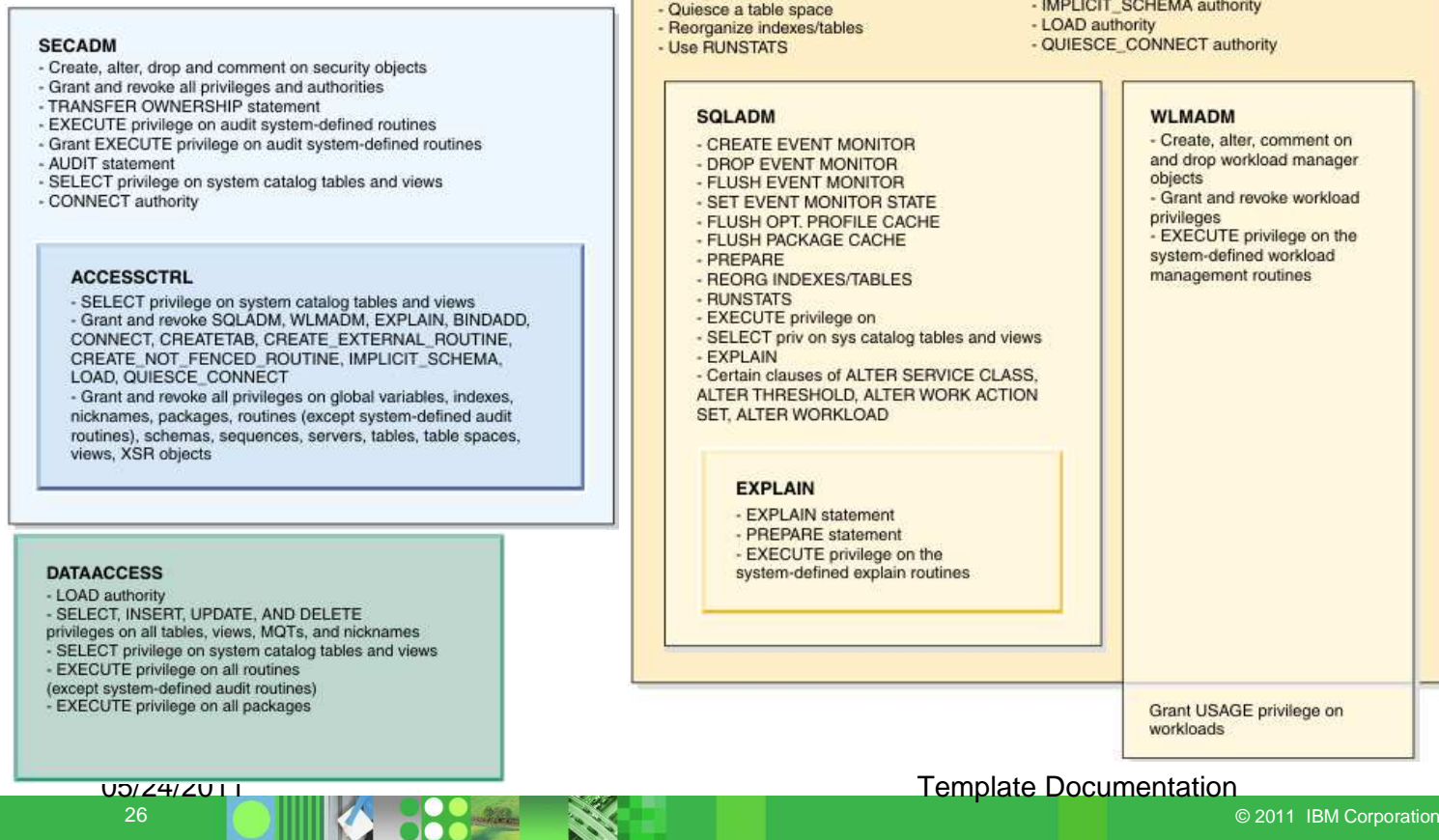
25

© 2011 IBM Corporation

DBADM is the Database Administrator. He will only have privileges at the database level, not the instance level. The DBADM does not necessarily have to have access to the data; however, by default he gets DATAACCESS and ACCESSCTRL authorities; therefore, he can access data and grant to others.

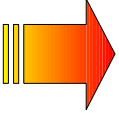
Many people get confused and think that DBADM is assigned also using the dbm cfg; but as you can see, it's assigned using a GRANT statement.

Database-level authorities



These figures taken directly from the DB2 manuals show the different database-level authorities and their hierarchy, including some descriptions as to what each can do.

Agenda

- DB2 security overview
- Authentication
- Authorization
-  • **Roles**
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context

Now let's move on to the 3rd item within the Authorization topic: Roles

Roles

- Roles are like “user-defined” database-level authorities
- You cannot assign instance-level authorities (e.g: SYSADM) to a role
- Example:

```
CREATE ROLE TESTER
GRANT SELECT ON TABLE STAFF TO ROLE TESTER
GRANT SELECT ON TABLE DEPT TO ROLE TESTER
GRANT ROLE TESTER TO USER RAUL, USER JIN
REVOKE ROLE TESTER FROM USER JIN
```

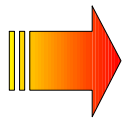
You can think of roles as “user-defined” database-level authorities.

You first create a role and grant different privileges to it.

Then you can grant/revoke a role to a user or group, in the same way you grant/revoke a database-level authority to a user/group.

Agenda

- DB2 security overview
- Authentication
- Authorization
- Roles
- **The PUBLIC group**
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context



The PUBLIC group

- Special group defined in DB2 automatically
 - ANY user id identifiable by the operating system/network authentication service belongs to the PUBLIC group
- The following (and more) are granted to PUBLIC by default:
 - CONNECT
 - CREATE TAB
 - IMPLICIT_SCHEMA
 - BINDADD
- To "lock down" your system, you can:
 - CREATE DATABASE with RESTRICTIVE option or
 - Revoke these privileges from PUBLIC

05/24/2011

Template Documentation

30

© 2011 IBM Corporation

The PUBLIC group is a special group in DB2 where every member the group is a user from the operating system.

So let's say, if I'm right now working on Windows or Linux, and I create a new operating system user, immediately that new operating system user will be part of the PUBLIC group.

The PUBLIC group, by default, has several privileges:

CONNECT, CREATE TABLE, IMPLICIT_SCHEMA, "BINDADD", etc.

So what does this mean? If I right now go to Windows or Linux, and I create a user, that user can connect to a database.

The PUBLIC group

- Special group defined in DB2 automatically
 - ANY user id identifiable by the operating system/network authentication service belongs to the PUBLIC group
- The following (and more) are granted to PUBLIC by default:
 - CONNECT
 - CREATE TAB
 - IMPLICIT_SCHEMA
 - BINDADD
- To "lock down" your system, you can:
 - CREATE DATABASE with RESTRICTIVE option or
 - Revoke these privileges from PUBLIC

```
REVOKE CONNECT          ON DATABASE FROM PUBLIC
REVOKE CREATETAB        ON DATABASE FROM PUBLIC
REVOKE IMPLICIT_SCHEMA  ON DATABASE FROM PUBLIC
REVOKE BINDADD          ON DATABASE FROM PUBLIC
```

05/24/2011

Template Documentation

31

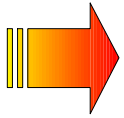
© 2011 IBM Corporation

If you don't like to allow any of these privileges to PUBLIC at all, you can revoke these privileges as shown, using these revoke statements, or create the database with the RESTRICTIVE option.

(Demo)

Agenda

- DB2 security overview
- Authentication
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)
- Trusted context



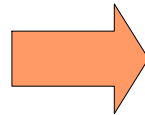
Let's now talk about views, and how they can help with security

Granular access through views

- Allows multiple users to see different presentations of the same data
- Nice for simple security policy, but complicated to manage in large settings
- Procedure:
 - Create a view (subset of the data from the base table)
 - Authorize the user to access the view
 - Revoke access from the user to the base table

EMPLOYEE

| ID | NAME | AGE | PHONE | SALARY |
|----|-------|-----|-------|--------|
| 12 | Peter | 30 | 99999 | 10000 |
| 3 | Mary | 23 | 88888 | 15000 |



EMP_VIEW

| ID | NAME | AGE | PHONE |
|----|-------|-----|-------|
| 12 | Peter | 30 | 99999 |
| 3 | Mary | 23 | 88888 |

```
CREATE VIEW EMP_VIEW AS (
  SELECT ID, NAME, AGE, PHONE
  FROM EMPLOYEE);
```

05/24/2011

33

Template Documentation

© 2011 IBM Corporation

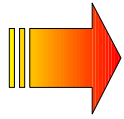
Using views you can provide granular access to your data; and it's very easy to set up as shown in this slide.

So in the example, you have a table with five rows, the last one containing SALARY information. Since this is sensitive information, you don't want all users to see it, therefore, you can create a view EMP_VIEW that omits this column.

Next you grant access to the user to this view, and revoke access from the user to the base table.

Agenda

- DB2 security overview
- Authentication
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- **Label-based access control**
- Extended Security (Windows only)
- Trusted context



Label-based access control (LBAC)

| | No LBAC | SEC=254 | SEC=100 | SEC=50 | ID | SALARY |
|---|---------|---------|---------|--------|-----|--------|
| <pre>SELECT * FROM EMP WHERE SALARY >= 50000</pre> | | | | | 255 | 60000 |
| | | | | | 100 | 50000 |
| | | | | | 50 | 70000 |
| | | | | | 50 | 45000 |
| | | | | | 60 | 30000 |
| | | | | | 250 | 56000 |
| | | | | | 102 | 82000 |
| | | | | | 100 | 54000 |
| | | | | | 75 | 33000 |
| | | | | | 253 | 46000 |
| | | | | | 90 | 83000 |
| | | | | | 200 | 78000 |

05/24/2011

Template Documentation

35

© 2011 IBM Corporation

Label-based access control provides granular security at the row and column level. It uses a label that is associated with both user sessions and data rows or columns to grant access to data in your table. The figure illustrates how LBAC works.

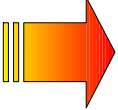
In the figure, the table **EMP** has one column, **SALARY**, and an internal column **ID** containing the label for a given row. The other columns in the figure are used only for illustration purposes. If the query shown in the figure is executed, depending on the label the user has, he will be able to see different rows. The column with the title 'No LBAC' represents the rows that would be selected if LBAC was not implemented. As you can see, all the rows with a salary greater or equal to 50,000 are selected.

Now let's say the user issuing the query has a security label of 100. You can see the rows selected in this case on the third column counting from the left. In this case, DB2 will find the rows where salary is greater or equal to 50,000, and then it will review the security label for the row. For example, the first row has a salary of 60000 and a label ID of 255. Since this user has a label ID of 100 which is smaller than 255, he cannot see this row, and therefore the output from the query will not return it.

LBAC security needs to be implemented by a security administrator that has the SECADM authority.

LBAC is not a feature available with DB2 Express-C

Agenda

- DB2 security overview
- Authentication
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
-  • Extended Security (Windows only)
- Trusted context

Extended security (Windows only)

- Controls access (through the operating system) to DB2 system files
- **DB2ADMNS** Windows group
 - ▶ This group and local administrators will have complete access to all DB2 objects through the operating system.
- **DB2USERS** Windows group
 - ▶ This group will have read and execute access to all DB2 objects through the operating system.

05/24/2011

Template Documentation

37

© 2011 IBM Corporation

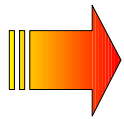
Extended security which applies only to Windows prevents access to DB2 files for users who are not part of the DB2USERS or DB2ADMNS windows groups. This way, not all Windows users will have access to DB2 system files from the operating system.

You put in the DB2ADMNS group all the administrators of DB2 who will be able to do pretty much everything on DB2, and they can even access the DB2 system files through the operating system.

In the DB2USERS group you put users who can work with DB2 objects through the operating system.

Agenda

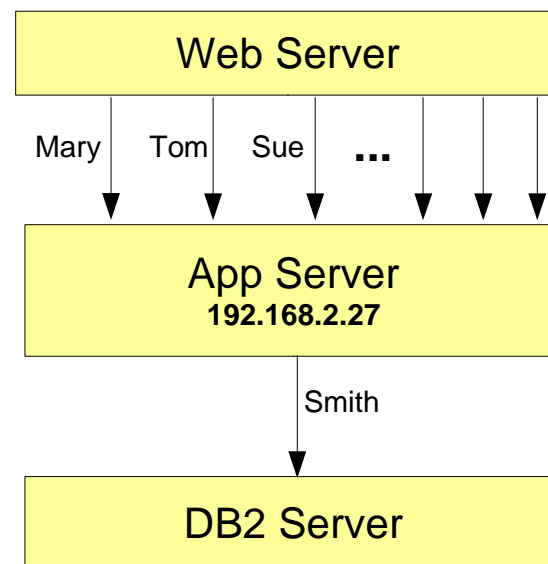
- DB2 security overview
- Authentication
- Authorization
- Roles
- The PUBLIC group
- Granular access through views
- Label-based access control
- Extended Security (Windows only)



- **Trusted context**

Trusted context

```
CREATE TRUSTED CONTEXT ctxt
BASED UPON CONNECTION USING SYSTEM AUTHID smith
ATTRIBUTES (ADDRESS '192.168.2.27')
DEFAULT ROLE managerRole ENABLE
```



05/24/2011

Template Documentation

39

© 2011 IBM Corporation

Trusted context applies to 3-tiered environments where you have a Web server, an application server, and a Database server.

It tries to solve the problem of identifying who is doing what at the database server. What happens is that, though there can be many users coming from the Web Server to the application server with their own user IDs (say Mary, Tom, Sue, etc in the example), the connection from the application server to the database server is normally done by one or few connections with fixed user IDs (Smith in the example). This is done for performance reasons since connecting/disconnecting often from the database can be costly. However, this raises a security problem of not being able to map 1 to 1 which Web user performed what action on the database server. For example, was it user "Mary" who deleted a given record? At the database server, we cannot determine this, since all auditing will point to user "Smith".

Using trusted contexts, you will not break the application server to database server connection to be per user; you keep using the same fixed connections, however, you can identify through these contexts (eg: Based on IP address, domain name, etc) the user performing the operation. You can also assign roles to the trusted contexts.

In the example, the trusted context is created between the application server located at IP address 192.168.2.27 using authorization "Smith" to connect from the application server to the DB2 server. So when there is a connection coming from user Smith from this IP address, the trusted context will take place, and this connection will have the role of "managerrole". Now that this context is "trusted", at the application server, you can code (depending on the language) a way to switch users at the app server level. So say from the Web server you access the app server as "mary", then at the app server you put some logic that if it's "mary" (who happens to be a manager, switch "smith" to this user ID for the trusted context ctxt; so from the db server, "mary" is now the user performing operations.

Trusted context

- **Useful for 3-tier environments**
- **“Trusted Context”:**
 - A trusted relationship between the DB and the application
 - Switch current user ID
 - Acquire additional privileges via role inheritance
 - Relationship identified by connection attributes used
 - IP Address,
 - Domain Name,
 - Authorization ID,
 - Data Encryption

05/24/2011

40

Template Documentation

© 2011 IBM Corporation

So a trusted context applies to 3-tiered environments

The context is established between the App server and the DB server. The Web server is not involved.

And there are several attributes that can be used for identifying the trusted context such as the IP Address, domain name, authorization ID, and Data encryption.

If the same authorization ID for example, connects from a different machine with a different IP address than the one used for the trusted context, he may not be able to access the same data.



Thank you!

Use the forum in the db2university.com course AA001EN if you have technical questions about the materials covered in this course. Fellow students, faculty and IBMers can help you!