An Introduction to DB2 and MLS

Adding another level to your DB2 security

Meir Zohar, Senior Consultant
IBM Certified DBA for DB2 V8 for Z/OS
CA-Israel

An Introduction to DB2 and MLS

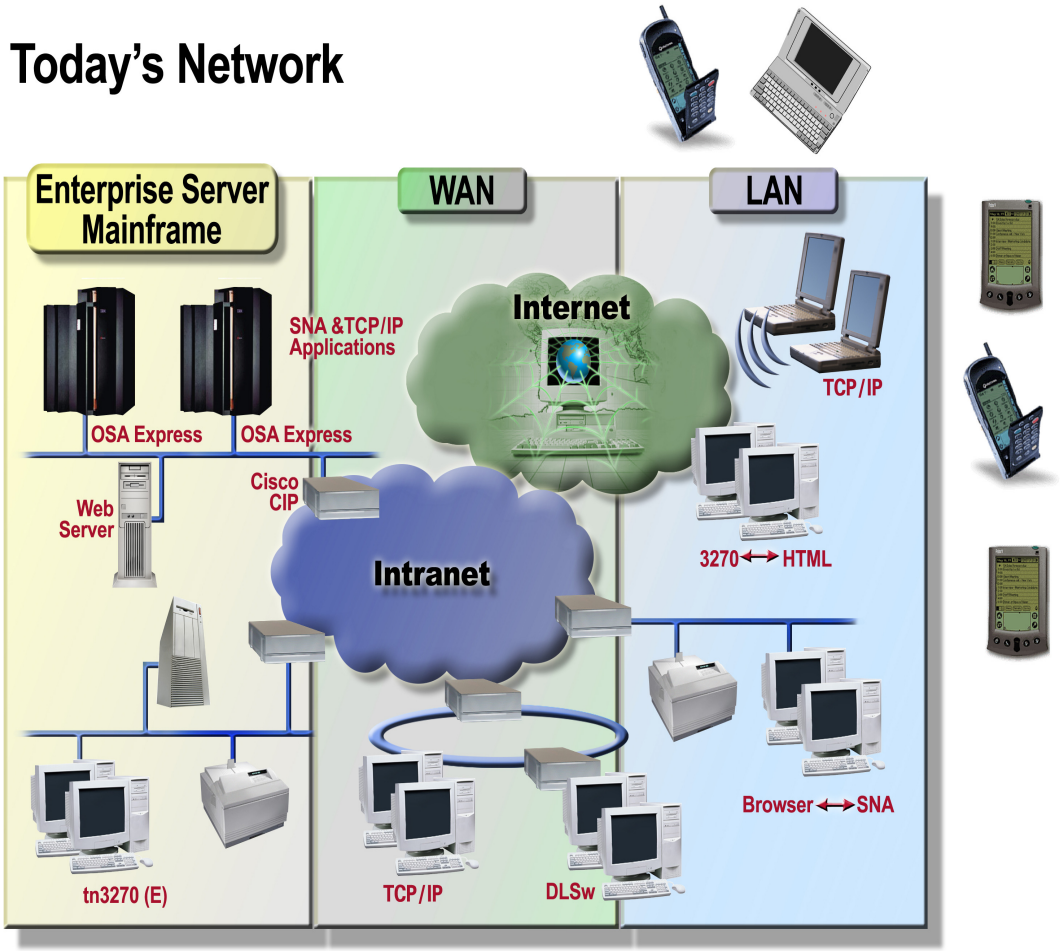
- Once upon a time ….
- MAC, DAC and other SRAs
- MLS and DB2 V8
- Getting it to work

Once upon a time

- We had a Mainframe
- With “dumb” terminals
- A few users
- Using silly passwords
- And the only way to get in, was to be inside (and read the sticker pasted to the terminal)

And then came ….

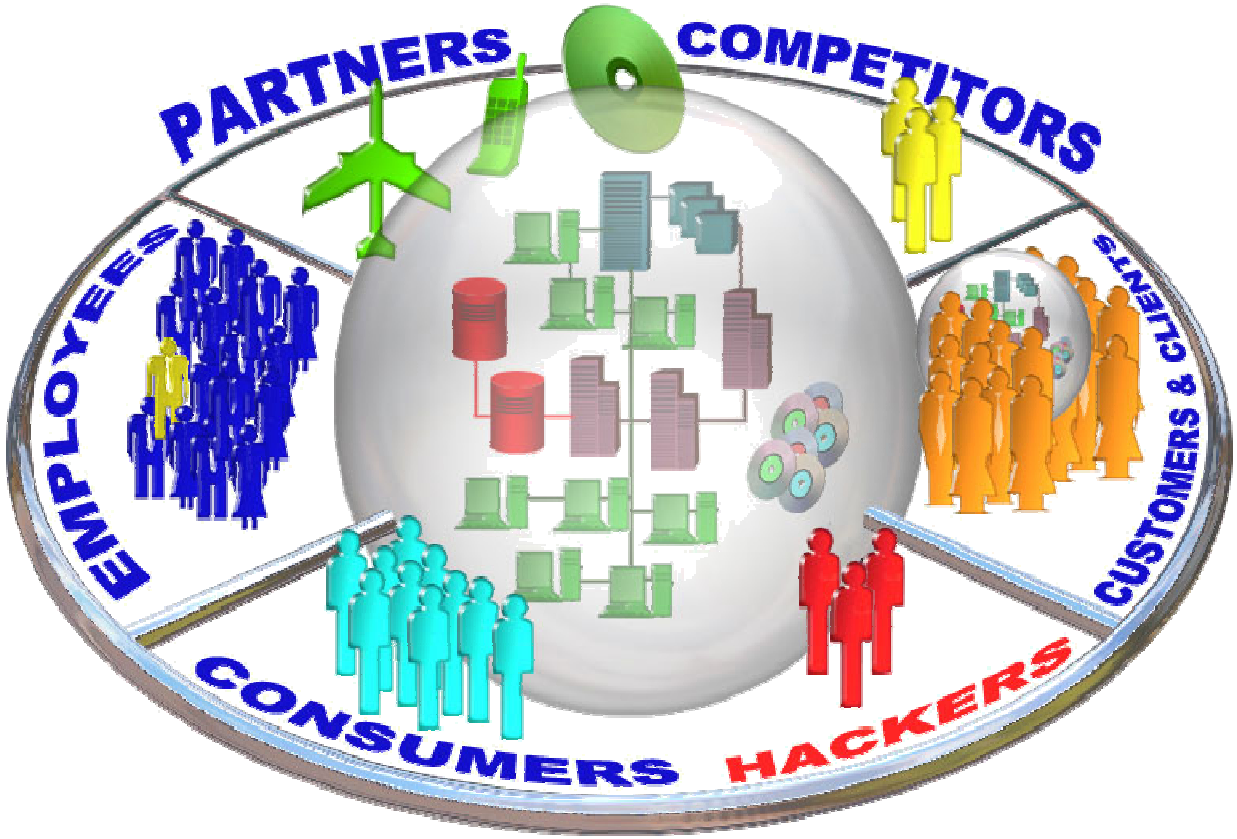
- The Internet
- Wireless communication
- 1000000s of users and customers
- Open networks and data sources
- And a demand for more security (not less)



And everyone wants your data …

- Network security
  - Designed to allow people in
- System Security
  - Opening the system to strangers
  - Wireless systems
  - Undefined users
- So, the only way to protect your data …. is to lock down your database

Opening your network …..



means protecting your database

Protecting your database …

Means that:

- GRANT to PUBLIC is no longer an option
  - PUBLIC ***is*** the public
- GRANT and REVOKE functionality is insufficient
  - Possible inconsistencies
  - GRANT with GRANT / CASCADE
- Your DAC based security may be full of holes
  - Access rights are rarely revoked

- And it may be time for a Multi Level Security project ….

MLS – More Security

- MLS adds an additional layer of data security by:
  - Purging storage objects before reuse
  - Enforcing accountability and creating comprehensive audit records
  - Hiding datasets when the user is not authorized to access them.
  - Inhibiting “write down” (i.e. declassifying data by lowering it classification).
  - Row level security

Discretionary Access Controls (*DAC*)

- Each defined User is a member of one or more Groups
- Data Owners grant access to the data to Users or to Groups
- Security administrators can add users to groups without asking the owner of the data
- Data Owner can grant access to unauthorized users ….

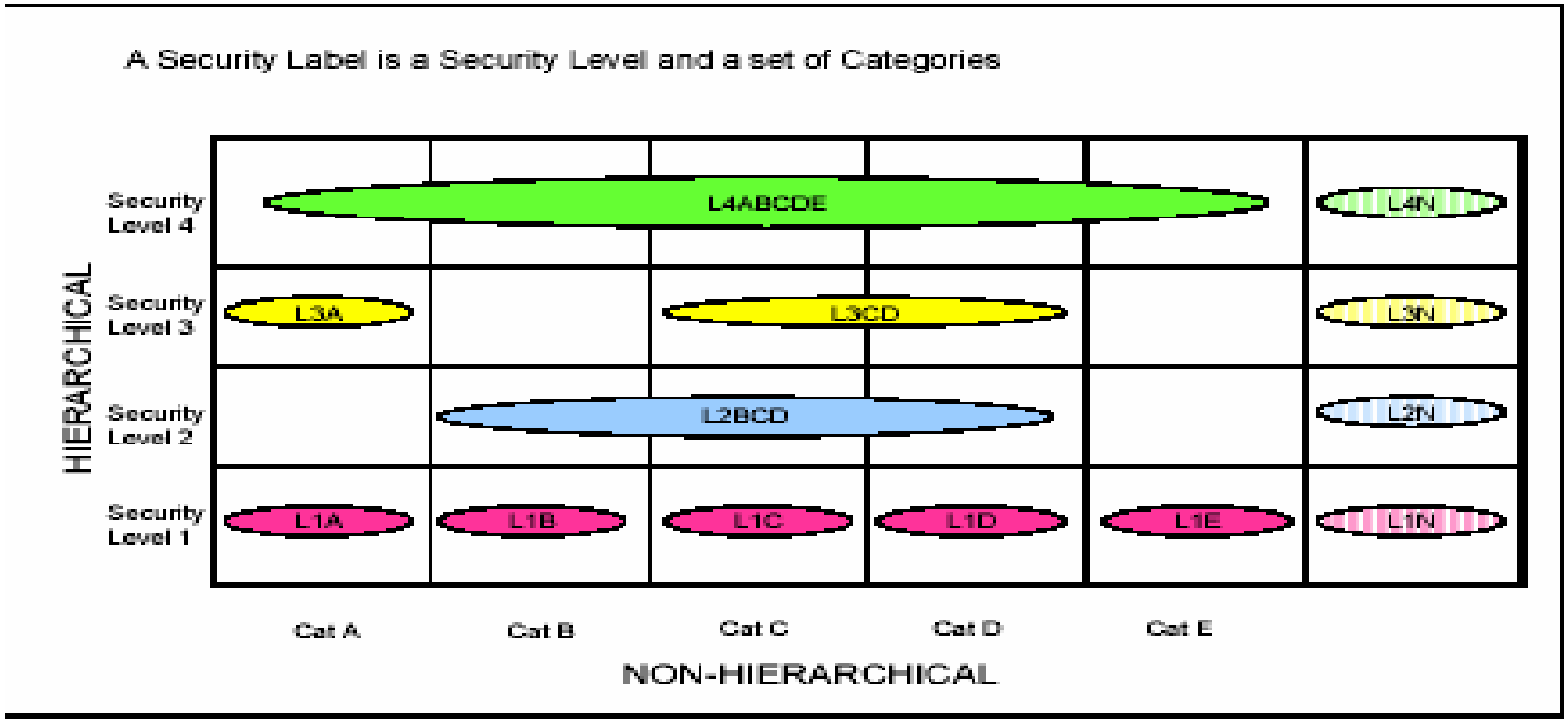
Mandatory Access Controls (*MAC*)

- Each defined User is a member of one or more Group and has a user specific classification
- The security ***administrator*** (not the data owner) creates a data specific classification
- The user can access the data only if the user's classification is at ***minimum*** the equivalent to the data's classification

- Actually not very new (TDSEC B1/Orange book 1985)

Defining the MAC mechanism

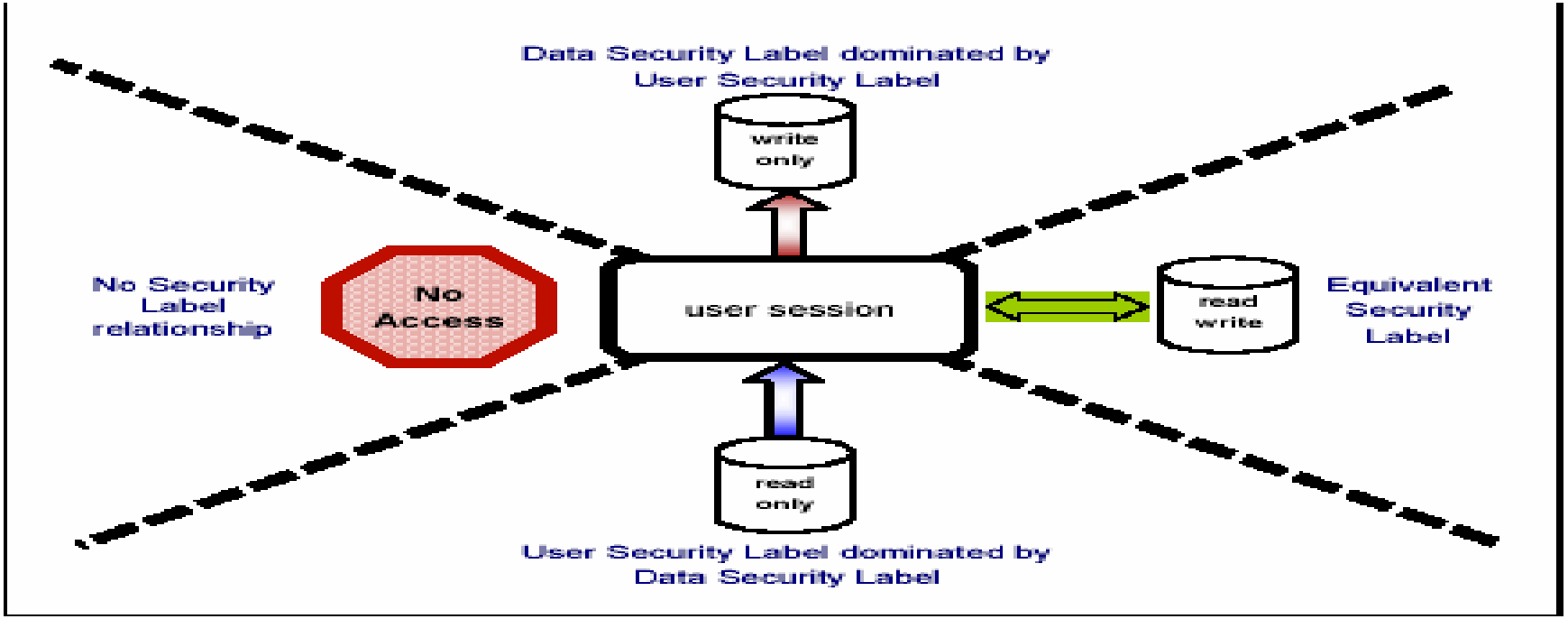
Security Labels



MAC Concepts

- Dominance
  - The dominating Label must have a Seclevel equivalent or higher then the dominated Label
  - The dominating Label must include *all* the categories in the dominated Label
- Equivalence
- Disjoint
  - If neither Label dominates the other

MAC Concepts (3)



- Write only access is not implemented in z/OS
- Explicit controlled write-down privilege can be assigned to groups/users

Security Labels

- A ***security label*** enables an installation to:
  - classify **subjects** and **objects** according to a data classification policy
  - identify **objects** to audit based on their classification
  - protect **objects** such that only appropriately classified **subjects** can access them.
- Subjects are entities requiring access to a resource
  - Users
  - STCs
  - Jobs
  - Unix processes etc.

Security Labels

- Objects are any system resources being accessed
  - Data Sets
  - Rows in a table
  - Commands
  - Terminals
  - Printers
  - DASD Vols

Security Labels

- Combination of a hierarchical Security Level
  - RACF SECDATA(SECLEVEL)
  - TSS {Add | Remove | List} (MLS) SECLEVEL(level) LVLNAME(seclevel-name)
  - up to 254 levels
- And zero or more Security Categories
  - non hierarchical (administrative) label to partition the levels by groups.
- Assigned specifically to subjects (not groups)
- Can be separated by system

System Created Security Labels

- Default definitions –
  - *Syshigh – Dominates any other seclabel – use for sysprogs, consoles etc.*
  - *Syslow – Lowest security, no categories should be used for Datasets that are not updated but everyone needs to read*
  - *Sysnone – Equal to anything compared to should be used to access data that everyone needs to update and write down is not implemented.*
  - *Sysmulti – equivalent to any seclabel for servers and daemons that spawn subtasks with different seclabels.*

MAC and DB2

- Each DB2 object can be assigned a SECLABEL that determines access rights to it

  - Tables
  - Views
  - Rows (enabling Row Level Security)

Implementing MLS on z/OS

- Plan the implementation in detail
- Implement carefully
  - Any changes made in the way MLS is setup will affect your entire LPAR or PLEX.
- Designing MLS should be from bottom up (plan application protection, then system protection).

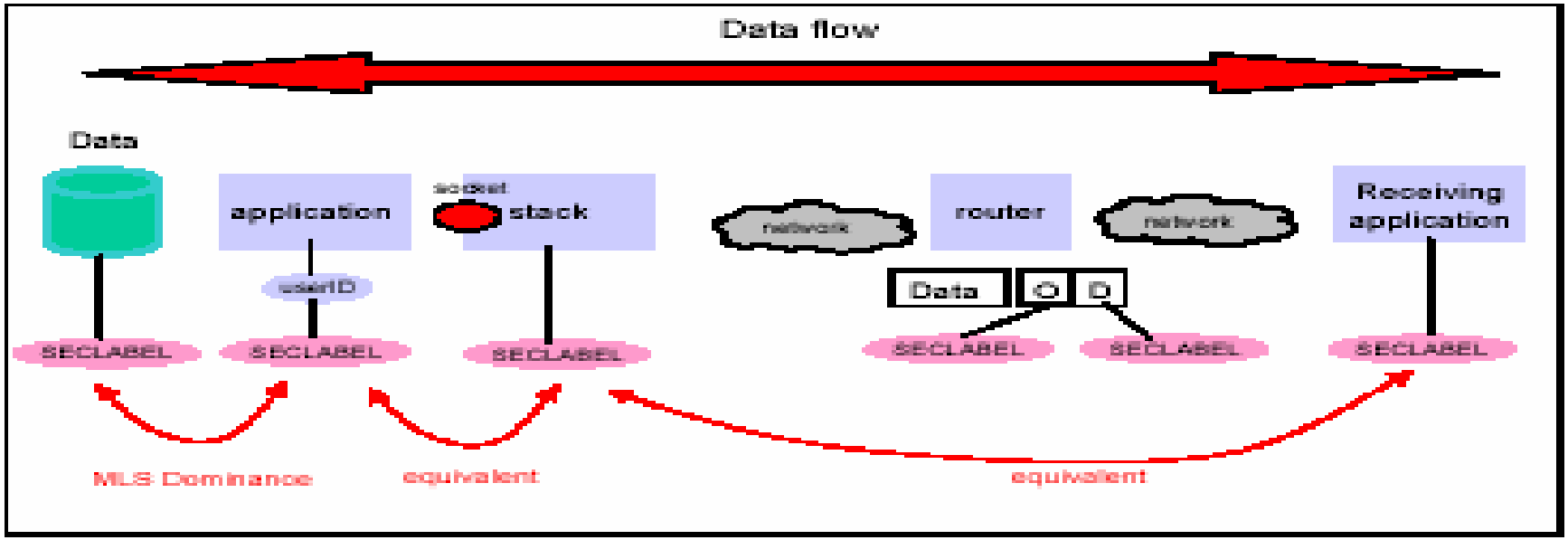
Implementing MLS on z/OS

- Analyze your system in detail to determine how many Seclevels and how many Security Categories are required.
- Create Seclabels based on combinations of Seclevels and Categories
- Seclabels must be linked to all resource class profiles
- All Resources must be connected to profiles that include seclabels
  - *failure to do this may cause the system to fail when attempting to access resources if MLS is initiated before the mapping is complete*

Implementing MLS on z/OS

- HFS does not fully support Seclabels
  - Implementing MLS will require migration to zFS first
  - Once a SECLABEL has been added it cannot be removed
- Planning will include mapping of common shared datasets by groups
  - datasets accessed by all users for read-only (i.e. SYS1.LINKLIB)
  - datasets that should be accessed by almost no one (SYS1.PARMLIB)
- Datasets that *may* have sensitive data should be SYSHIGH (this will let all users write but only authorized users to read).
  - *Log Data sets*
  - *Dump DS*
  - *SMF*
  - *Page*
  - *Etc.*

MLS and TCP/IP Network Security…..



- Each component requires that relevant Seclabels be defined for it
- Network devices do not change the origin and destination of the data and therefore can be considered “transparent” to the application and security.
- The packet security identity can include both its origin (network address) and the userid (or POE) associated with it. This enables limiting specific users to specific network locations when accessing an application.

MLS & DB2 V8

DB2 Based Security today

- Users are defined to the system security (TSS,RACF etc.) and assigned to groups
- Users or tasks connect or sign on to DB2 and receive authorizations based on the USER (primary authid) or the users group ids (secondary authids)
- The Current SQLID can have explicit authorizations and/or inherited authorizations from the group

MLS & DB2 v8

DB2 Security Challenges

- Lack of comprehensive Control
  - GRANT with GRANT OPTION
  - CASCADE processing
- Time sequences may result in unintended results when CASCADE processing is used
- **DROP**ping an object removes all security definitions
  - which is why **PUBLIC** is still being used

MLS & DB2 V8

DB2 External Security (TSS or RACF)

- Determines whether a user can access DB2
- Assigns an identity or identity group to the accessing user
- Can protect the underlying physical datasets

MLS & DB2 V8

External Security (TSS/ACF-2 or RACF) benefits

- Control multiple DB2s from a single point
- Security definition and maintenance even when the object doesn't exist
- Protect multiple objects with single rules
- Preserve authorities when objects are DROPped
- Eliminate the CASCADING revoke
- Let DBA's be DBA's
  - and security admins do their jobs

MLS & DB2 v8

MLS modifies the rules

- A User (subject) is now a:
  - User
  - Job
  - Stored Proc
- An object is anything that needs to be protected
  - A Data Set
  - A Table
  - A row in a table
  - A command

MLS & DB2 v8

Row Level Security

- Limit access to specific segments of data
  - w/o VIEWs
  - sub-tables
  - separate data structures
- Propagation of security data in the referential structure
- Maintaining security data when row data is updated

MLS & DB2 v8

Why not use VIEWs ?

- Complex security requirements may require a large number of VIEWs
- Read only VIEWs will require complex workarounds to support required updates
  - and views including JOINS to authorization tables will inevitably be read-only.
- Complex data structures may require complex data manipulation just to control access
- VIEWs are only relevant to SQL and will not protect the data outside a SQL based application

MLS & DB2 v8

Other options

- Security exits
  - Must be coded in assembler
  - Expensive (processing overhead)
  - Auditor skills
- Triggers …

MLS & DB2 V8

Seclabel Columns

- SECLABELs can be added to row definitions by adding a COLUMN with the descriptor AS SECURITY LABEL
- The SECLABEL columns is updated automatically by the security system whenever a row is inserted, updated or loaded
- Once a SECLABEL is added to an object, it will have to be DROPPED to be eliminated

MLS & DB2 v8

Seclabels and Data Access

- Select – The User SECLABEL must dominate the row SECLABEL
- INSERT – Row SECLABEL column is set to the User SECLABEL unless the user is authorized to *writedown*
- UPDATE – User SECLABEL must be equivalent to row SECLABEL unless the user is authorized to *writedown*
- DELETE – User SECLABEL must be equivalent to row SECLABEL unless the user is authorized to *writedown*.

MLS & DB2 v8

Implementation

- Identify which users and groups need access to which rows
- Design the relevant SECLABEL for user and objects
- Define SECLABELs in your security system (TSS, RACF)
- Add the SECLABEL column to the required tables
- Update the labels with the required values
- Be careful when enforcing write-down

MLS & DB2 v8

Utility Access

- Utilities running on objects protected by SECLABELs **must** have the required privileges
- If the utilities do not have the required privileges or writedown is not enabled, UNLOAD, LOAD and REORG may corrupt your data
  - only data containing matching SECLABELs will be deleted, updated etc.
- DSN1* utilities will ignore DB2 SECLABEL definitions
  - (secure the physical datasets using the system security mechanisms)

MLS & DB2 v8

Seclabel Restrictions

- Sysplex query parallelism is not supported for queries accessing Tables with a SECLABEL column
- Global temporary tables cannot contain a SECLABEL
  - if defined with LIKE, the column will be inherited but must be maintained by the application)
- MQTs can inherit only one SECLABEL
  - checked when used and not at creation or REFRESH
- Constraints
  - Unique Constraint – can be enforced
  - no Referential or Check constraints on a Security Column

MLS & DB2 v8 Seclabel Usage

Seclabel Usage

- What is the current userids SECLABEL ?
  - GETVARIABLE('SYSIBM.SECLABEL')

- Create a VIEW that lets a user access only data with their SECLABEL

  - CREATE VIEW *view* AS
    SELECT * FROM *table*
    WHERE SECURITY = GETVARIABLE('SYSIBM.SECLABEL');

MLS & DB2 V8

Object Level Implementation

- A SECLABEL is defined for each object
- The higher the object in the hierarchy, the higher the SECLABEL (*SSID,DB,TS (table,col,row),View, SG,BP,Plan,Collection (package), Schema (sproc,udf,jar,distinct type,sequence*)
- In RACF - define a resource class for each object type with the relevant Seclabels
- In TSS - just add the Seclabel to the object

  TSS ADD(MLS) DB2(TEST.QEWRQER.*.ASDF) SECLABEL(LABELA)
- Uses the DB2 external security exit and therefore the Install SYSADM does not fall under its control

MLS & DB2 v8

Things to Remember

- Add the Seclabel to all Indexes
  - if omitted, all access will go to the base table
- Detailed planning
  - Names, Naming Conventions etc.
- If possible, plan RLS from design
  - adding RLS after an object is being used can cause interesting results
  - Column functions will behave differently
- Internal politics (who runs your DB2 security ?)

Bottom Line

- Z platforms running z/OS and CA Top Secret/ACF-2 or RACF provide a robust and reliable security environment.
- MLS helps enhance this environment by adding layers to the security infrastructure, minimizing the possibility of “access by default”
- MLS can be partially implemented, but even a partial implementation requires careful and comprehensive planning

An Introduction to DB2 and MLS

- Once upon a time ….
- MAC, DAC and other TLAs
- MLS and DB2 V8
- Getting it to work

MLS & DB2 v8 – Implementation Stages

- Adding Seclabels to existing objects …

  - Sizing (8 bytes more to each row and index key)
  - SECLabel values (defaults ?)
  - Shut down apps
  - STOP the TS and IS
  - For every table
    - Drop the primary key, referential constraints and unique constraints (CHKP)
    - Opportunity to go to table controlled partitioning – drop the partitioning index
    - Alter the table to ADD the SECURITY LABEL column
    - Alter the TS to resize
    - Alter the remaining indexes adding the SECLABEL column (COMMIT here – AREO – otherwise you will get a RBDP)

MLS & DB2 v8 – Implementation Stages

- Recreate the partitioning index using DEFER
- Recreate the Primary Key
- Recreate the constraints adding the seclabel column where relevant

- Restart the TS and IS
- Rebuild the Indexes that you DEFERed
- Run a Check TS for all CHKPs
- Run the SEClabel update process you designed
- REORG
- Rebind
- Restart the apps and hope for the best ☺

MLS & DB2 v8 - Seclabel update process

- Default value when adding the Seclabel is the process Seclabel (DBADM or SYSADM will probably be SYSHIGH or SYSLOW)
- Propagate the Seclabels in a logical sequence down the hierarchy
- Beware of write down issues
  - set MLS(Failures) only after updating the labels or ascertain users have write down privileges
- Get it right the first time and the next time will be easy …

Some interesting reading material

- Multilevel Security and DB2 Row-Level Security Revealed - SG24-6480-00
- eTrust™ CA-Top Secret. Security for z/OS Multilevel Security Planning Guide r8 - H00422-1E
- D*B2 UDB for z/OS Version 8 Performance Topics*, SG24-6465-00.

Meir Zohar

Computer Associates

Meir.zohar@ca.com