z/OS LDAP Overview and Security Function Update

Jon Furminger
IBM z/OS LDAP Development
furming@us.ibm.com

Disclaimer

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as-is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used instead.

Any performance data contained in this document was determined in a controlled environment and therefore, the results which may be obtained in other operating environments may vary significantly.

Users of this document should verify the applicable data for their specific environments. It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country or not yet announced by IBM. Such references or information should not be construed to mean that IBM intends to announce such IBM products, programming, or services.

Permission is hereby granted to publish an exact copy of this paper in the Solutions proceedings. IBM retains the title to the copyright in this paper, as well as the copyright in all underlying works. IBM retains the right to make derivative works and to republish and distribute this paper to whomever it chooses in any way it chooses.

Trademarks

The following are trademarks of the IBM Corporation. An asterisk following the name denotes a registered trademark.

DB2*
IBM*
OS/390
Parallel Sysplex
RACF
Tivoli
z/OS
zSeries

The names listed below are trademarks or registered trademarks and are the properties of their respective companies.

JNDI is a trademark registered by Sun Microsystems, Inc.
Kerberos is a trademark of MIT
Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
UNIX is a registered trademark of The Open Group
Windows is a trademark of Microsoft, Inc.

All statements regarding IBM's future intent are subject to change without notice, and represent goals and objectives only.

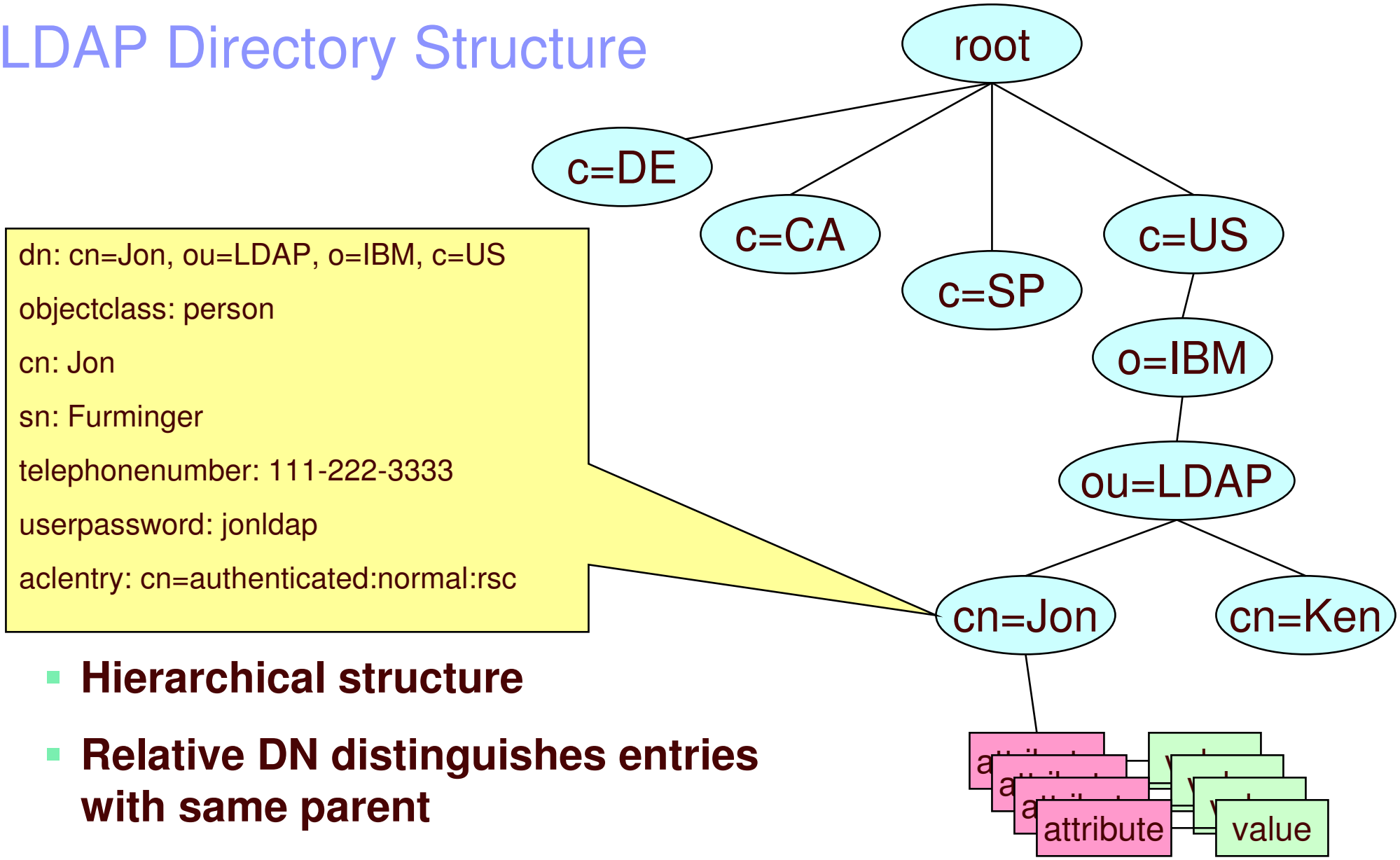
Abstract

- This presentation explains:
  - What is LDAP
  - Why a customer would want to use LDAP
  - Special features of z/OS LDAP

What is LDAP?

- **Lightweight Directory Access Protocol (LDAP) is a global directory model**
- **Originally developed as front-end of X.500 (DAP)**
- **The LDAP protocol runs over TCP**
- **Global directory model is based on entries**
  - Each entry identified by its DN (distinguished name)
    - Often uses cn (common name), ou (organization unit), o (organization)
    - Example: DN: cn=jon,ou=LDAP,o=IBM,c=US
- **Each entry is a collection of attributes**
  - Each attribute has a type and values
  - Attributes are grouped into object classes
    - Determine mandatory and optional attributes for an entry
  - Schema defines attributes and object classes

LDAP Directory Structure



- **Hierarchical structure**
- **Relative DN distinguishes entries with same parent**
- **Attributes are protected by Access Control Lists (ACL)**

What is it used for?

- **Storing information**
  - Information that is mostly read
  - Application configuration
    - Centrally located for easy administration, i.e. you don`t have to go to each workstation
  - Authorization checking
  - Identity checking (authentication)
  - User information
    - Phone numbers, address, etc

LDAP Parts

- **z/OS LDAP provides**
  - LDAP server: manages directory entries
  - LDAP client: C APIs to add, delete, modify, rename, compare and search entries
  - Command line client utilities: ldapadd, ldapdelete, ldapmodify, ldapmodrdn, and ldapsearch
- **Any Version 3 LDAP client can be used with z/OS LDAP server**
- **z/OS LDAP client and utilities can be used with any V3 LDAP server**

Using LDAP - Examples

- **Example : add an entry**
  - Create a file, jay.add, containing entry to be added:

```
dn: cn=jay,ou=LDAP,o=IBM,c=US
objectclass: person
cn: Jay
sn: smith
userpassword: jaypw
```

  - Invoke ldapadd utility:

```
ldapadd –h dceset3.ibm.com –p 389 –D cn=jon,ou=ldap,o=ibm,c=us
  –w jonldap –f jay.add
```

Using LDAP – Examples cont.

- **Example : modify an entry**
  - Create a file, jay.mod, containing changes:

```
dn: cn=jay,ou=LDAP,o=IBM,c=US
add: telephonenumber
telephonenumber: 222-333-4444
-
replace: sn
sn: smithson
```

  - Invoke ldapmodify utility:

```
ldapmodify –h ceset.ibm.com –D cn=jon,ou=ldap,o=ibm,c=us –w jonldap
  –f jay.mod
```

Using LDAP – Examples cont.

- **Example : search for an entry**
  - Display specific entry
    - ldapsearch –h dceset3.ibm.com –p 389 –D cn=jon,ou=ldap,o=ibm,c=us –w jonldap –s base –b cn=jay,ou=ldap,o=ibm,c=us objectclass=*

      dn: cn=jay,ou=LDAP,o=IBM,c=US

      objectclass: person

      cn: Jay

      sn: smithson

      telephonenumber: 222-333-4444
  - Display entries with telephonenumber in 222 area code and surname starting with smith:
    - ldapsearch –h dceset3.ibm.com –p 389 –D cn=jon,ou=ldap,o=ibm,c=us –w jonldap –s sub –b o=ibm,c=us "(&(telephonenumber=222*)(sn=smith*))"

Jsing LDAP – Examples cont.

- **Example : display all entries in the c=US directory tree**
  - ldapsearch –h dceset3.ibm.com –p 389 –D cn=jon,ou=ldap,o=ibm,c=us –w jonldap –b c=us objectclass=*

dn: c=US
objectclass: country
c: US

dn: o=IBM,c=US
objectclass: organization
o: IBM

dn: ou=LDAP,o=IBM,c=US
objectclass: organizationalunit
ou: LDAP

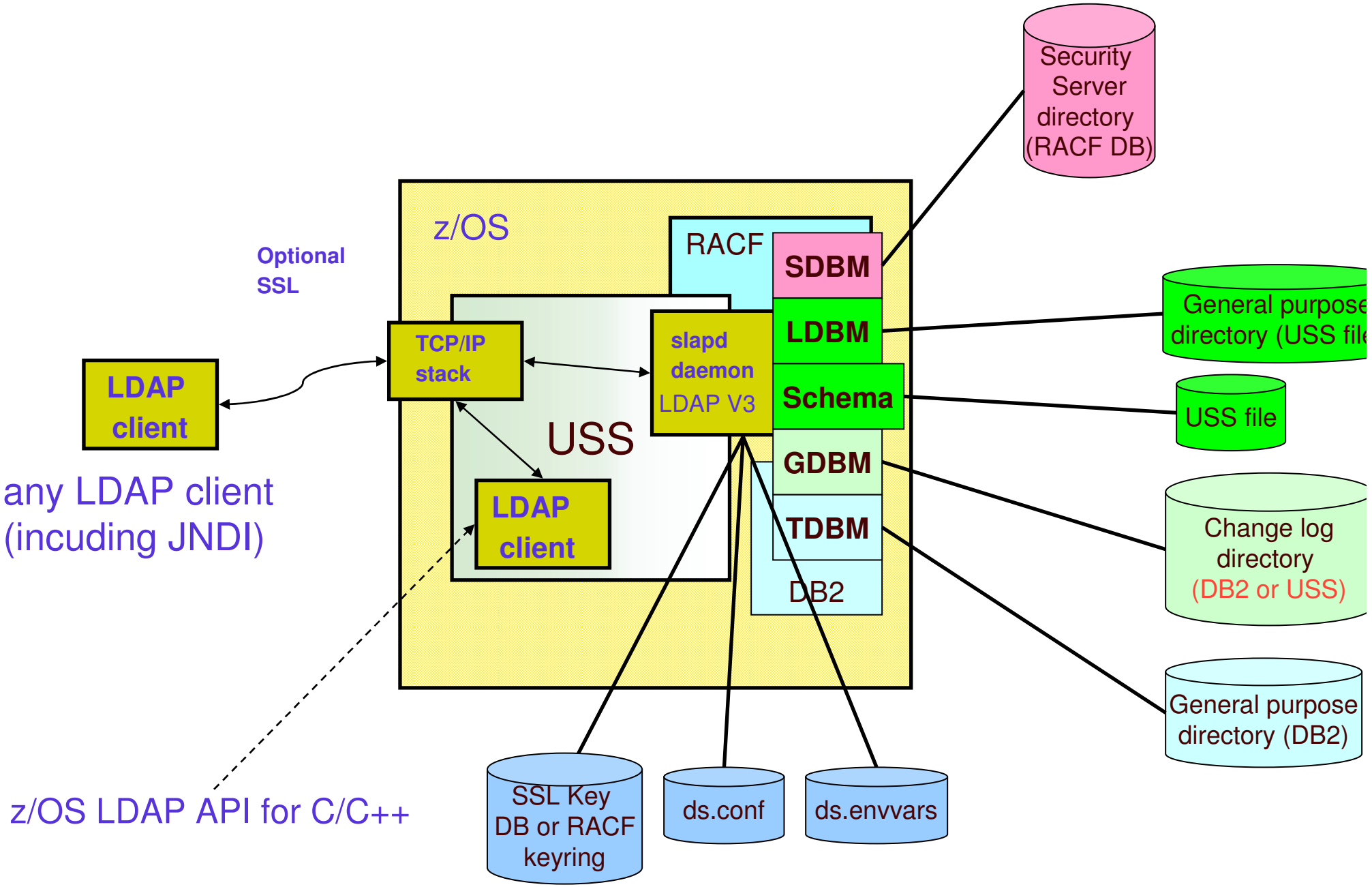
cn=ken,ou=LDAP,o=IBM,c=US
objectclass: person
cn: ken
...

Using LDAP – Examples cont.

- **Example : delete an entry**

```
ldapdelete –h dceset3.ibm.com –p 389 –D cn=jon,ou=ldap,o=ibm,c=us
  –w jonldap cn=jay,ou=ldap,o=ibm,c=us
```

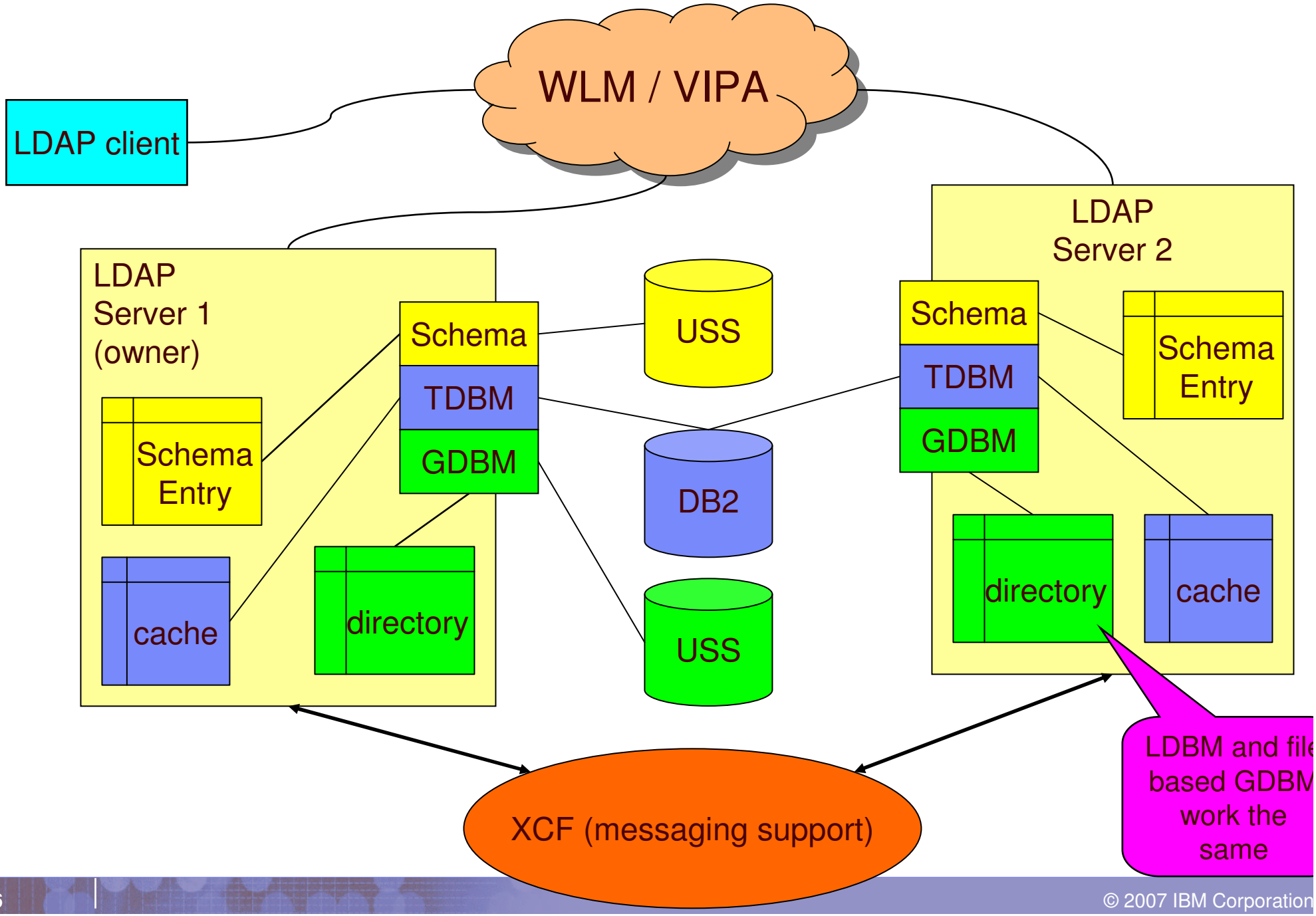
LDAP server on z/OS



LDAP Server on z/OS

- LDAP Server has multiple backends (data stores)
  - **TDBM**: General purpose directory
    - Full LDAP V3 support, including modifiable schema
    - Data stored in DB2 database
    - Full scalability
  - **LDBM:** General purpose file-based directory
    - Full LDAP V3 support, including modifiable schema
    - Uses USS file system to store directory entries
    - Useful for small to medium sized directories
  - **SDBM**: RACF users, groups, and user-group connections
    - Provides remote RACF administration and authentication
    - Fixed schema
    - Data stored in RACF database
    - Limited search capability
  - **GDBM**: Change log directory
    - Can be file-based or DB2-based
    - Similar to LDBM/TDBM but restricted operations
    - Contains records of changes to other backends and RACF
  - **Schema**
    - Single server-wide schema used by all backends, simplifies administration of server

IBM TDS Sysplex Support



Authentication with an LDAP Server

- **LDAP is a stateful protocol**
  - Session starts when client “binds” to server
  - Authentication is performed during bind
    - Check password or certificate
    - Determine groups to which user belongs (for authorization checking)
  - Session can be unauthenticated (anonymous bind)
- **LDAP supports different authentication protocols**
  - Simple bind: Distinguished Name and password
    - Session can optionally be protected with SSL
    - Passwords can be stored in LDAP directory, optionally one-way (MD5, SHA-1, crypt) or two-way (TDES) encrypted, or stored in RACF
  - Certificate bind: X.509 digital certificate over SSL
    - Distinguished name in certificate must conform with distinguished name of person authenticating – use RACF keyring or key database fileb
  - Kerberos bind: Kerberos principal sends ticket for LDAP server
    - Attribute: ibm-kn = principal@realm
  - CRAM-MD5, DIGEST-MD5 binds: DN/userid and password
    - Client hashes password using MD5 encryption

LDAP LDBM/TDBM Authentication

Enter userid : jon

Enter password : ********

dn="cn=jon,ou=LDAP,o=IBM,c=US"

pw=jonldap

ldap_bind_s(ld,host,port,dn,pw)

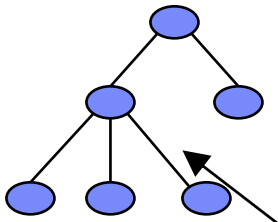
Bind request

LDAP Client (API)

z/OS LDAP Server LDBM or TDBM

Successful bind

dn: cn=jon,ou=LDAP,o=IBM,c=US



In LDBM or TDBM directory

objectclass: person

cn: jon

sn: furminger

userpassword: jonldap

LDAP Database

LDAP Authentication with SDBM (RACF)

Enter userid : u12345

Enter password : ********

dn="racfid=u12345,profiletype=user,cn=myRACF"

pw=racfpw

ldap_bind_s(ld,host,port,dn,pw)

Bind request

LDAP Client (API)

z/OS LDAP Server LDBM or TDBM

Successful bind

Verify user and passwor

__passwd(U12345,racfpw

dn: racfid=u12345,profiletype=user,cn=myRACF

objectclass: racfUser

objectclass: racfBaseCommon

racfid: U12345

racfprogrammername: Jon Furminger

racfdefaultgroup: racfid=group1,profiletype=group,cn=myRACF

racfconnectgroupname: racfid=group1,profiletype=GROUP,cn=myRACF

racfconnectgroupname: racfid=group2,profiletype=GROUP,cn=myRACF

RACF Database

z/OS LDAP Server Native Authentication

- **Disadvantages of authentication in LDBM/TDBM:**
  - Password in LDBM/TDBM directory entry
  - Another password repository to manage
- **Disadvantages of authentication in RACF:**
  - SDBM backend required with its funny DN (racfid,profiletype)
  - Fixed schema: only RACF info, cannot add attributes
- **Native Authentication – LDBM/TDBM with RACF authentication**
  - Standard Distinguished Name (e.g. cn, ou, o)
  - Any schema supported by LDAP V3 for an entry can be used
    - Any information supported by the schema can be retrieved
    - Use LDBM/TDBM groups and group membership in ACLs
  - Authentication (password verification) performed by RACF
    - Password for entry is in Security Server, not in LDAP
    - No need for administration or synchronization of multiple password registries
    - RACF authentication triggered by attribute ibm-nativeId in LDBM/TDBM entry
  - Can configure which entries use which sort of authentication

LDAP Native Authentication

Enter userid : jon

Enter password : ********

dn="cn=jon,ou=LDAP,o=IBM,c=US"

pw=jonldap

ldap_bind_s(ld,host,port,dn,pw)

LDAP Client (API)

Bind request

z/OS LDAP Server LDBM or TDBM

RACF Database

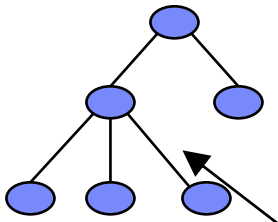
Successful bind

Find LDBM/TDBM entry

Verify native id and password in RACF

__passwd(U12345,racfpw)

Get LDBM/TDBM groups



In LDBM or TDBM directory

dn: cn=jon,ou=LDAP,o=IBM,c=US

objectclass: person

cn: jon

sn: furminger

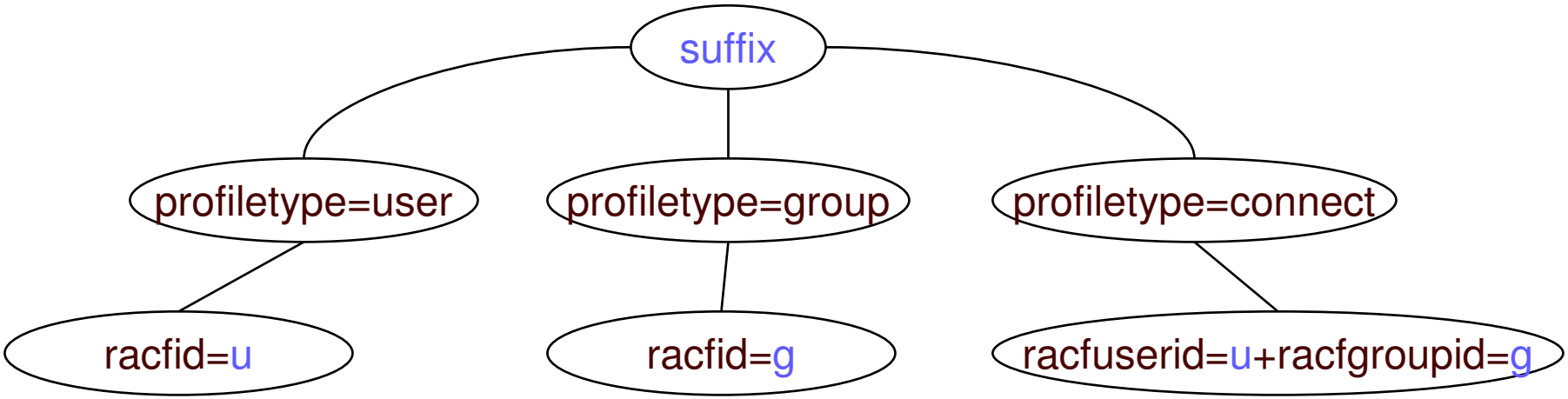
userpassword: jonldap

Ibm-nativeId: U12345

LDBM TDBM database

SDBM Support for RACF

- **Use LDAP to add, modify, delete, display RACF users, groups, and user-group connections – remote admin**
  - Equivalent to RACF commands: ADDUSER, ALTUSER, DELUSER, LISTUSER, CONNECT, REMOVE
  - SDBM directory structure



  - Limited search capabilities – predefined by SDBM
  - All data accessed via RACF
    - No RACF data in LDAP
    - Authorization controlled by RACF, based on bound userid

SDBM Support of RACF - cont

- **Hard coded schema definitions**
  - Each RACF user/group/connect profile segment mapped to an LDAP object class
  - Example:

    User OMVS segment ← → racfUserOmvsSegment object class
    - Object class contains all the attributes in that segment
  - Each RACF add/alt/listuser, add/alt/listgrp, connect keyword mapped to an LDAP attribute
    - Example: OMVS UID keyword ←→ racfOmvsUid attribute

Using SDBM - Examples

- **Example: add a RACF user entry**
  - Create a file, u1234.add, containing entry to be added:

```
dn: racfid=U1234,profiletype=user,cn=myRACF
objectclass: racfUser
objectclass: racfUserOmvsSegment
racfid: u1234
racfdefaultgroup: dce1
racfowner: radmin
racfattributes: special
racfomvsuid: 321
racfomvshome: /home/u1234
```

  - Invoke ldapadd utility:

```
ldapadd –h dceset3.ibm.com –p 389 –D
  racfid=radmin,profiletype=user,cn=myRACF –w radminpw –f u1234.add
```

  - SDBM executes:

```
ADDUSER u1234 OWNER(radmin) DFLTGRP(dce1) special OMVS(UID(321)
  HOME(/home/u1234)
```

Using SDBM – Examples cont.

- **Example: display a RACF user-group connection**
  - Invoke ldapsearch utility

    ldapsearch –h dceset3.ibm.com –p389
    –D racfid=radmin,profiletype=user,cn=myRacf -w radminpw
    –b racfuserid=u1234+racfgroupid=dce1,profiletype=connect,cn=myracf
    objectclass=*

  - SDBM executes LISTUSER u1234 and returns connection info for group dce1

    dn: racfuserid=u1234+racfgroupid=dce1,profiletype=connect,cn=myracf
    objectclass: racfConnect
    racfuserid: u1234
    racfgroupid: dce1
    racfconnectowner: racfid=RADMIN,profiletype=user,cn=myRacf
    racfconnectgroupauthority=USE
    racfconnectauthdate=04.279
    ...

Changing the RACF Password

- **ldapmodify can be used to change RACF password**
  - Via SDBM:

    dn: racfid=u1234,profiletype=user,cn=myRACF
    replace: racfpassword
    racfpassword: mynewpw
    racfattributes: noexpired

  - Via TDBM with native authentication

    dn: cn=jon,ou=LDAP,o=IBM,c=US
    delete: userpassword
    userpassword: jonldap
    -
    add: userpassword
    userpassword: mynewpw
    -

    - Note: replace: userpassword cannot be used by SDBM - not supported

  - LDAP SDBM or native authentication bind can be used to change a password (even if it is expired)
    - Specify old_password/new_password when binding

LDAP-RACF Change Logging

- **Provides way to propagate RACF user changes (including password changes) to other systems**
- **RACF part:**
  - Notifies LDAP when a change to a user occurs
  - Creates PKCS7 envelope containing clear password
- **LDAP part:**
  - Creates an entry containing the RACF info in the changelog directory (GDBM backend)
    - Can access entry using normal LDAP operations from any LDAP client
  - Retrieves RACF password envelop via LDAP SDBM search
- **Used by IBM Tivoli Directory Integrator to synchronize passwords:**
  - Periodically does LDAP search of change log for new entries
    - Persistent search can also be used
  - If password changed, performs LDAP search of RACF user to retrieve enveloped password
  - Decrypts envelop and sets password on other systems

Change logging continued

- **Search the change log**
  - ldapsearch ….. –b cn=changelog changenumber>=1023

```
dn: CHANGENUMBER=1023,CN=CHANGELOG
objectclass: CHANGELOGENTRY
objectclass: IBM-CHANGELOG
changenumber: 1023
targetdn: racfid=U1234,profiletype=user,CN=MYRACF
changetime: 20030611161820.374472Z
changetype: MODIFY
changes: replace: racfpassword
racfpassword: *ComeAndGetIt*
-

ibm-changeinitiatorsname: racfid=radmin,profiletype=user,cn=myRACF
```

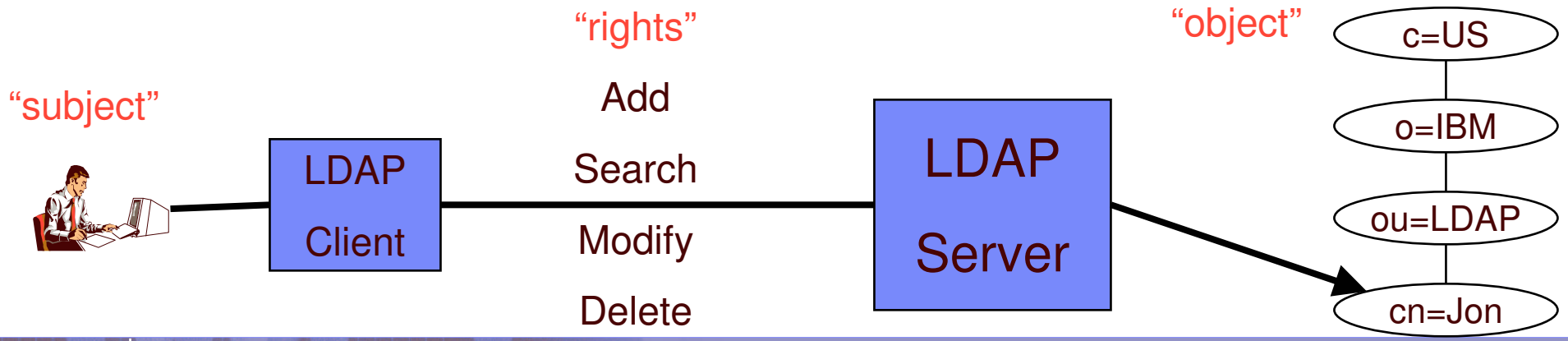
  - Retrieving RACF envelope containing new password

```
ldapsearch –D racfid=radmin,profiletype=user,cn=myRacf –w radminpw –L
–b racfid=U1234,profiletype=user,cn=myRacf objectclass=* racfpasswordenvelope

racfid=U1234,profiletype=USER,cn=myRacf
racfpasswordenvelope:: <base64 encoded password envelope>
```

Access Control Checking

- **Does subject have the right to perform the requested operation on an object?**
  - "subject" – the "bound" LDAP client identity: DN of requestor + DNs of groups to which requestor belongs
  - "object" – the entries or the attributes of the entries involved in the operation
  - "rights" – the access required to perform the requested operation (add/delete, read/write/search/compare attribute)



Access Control Implementation

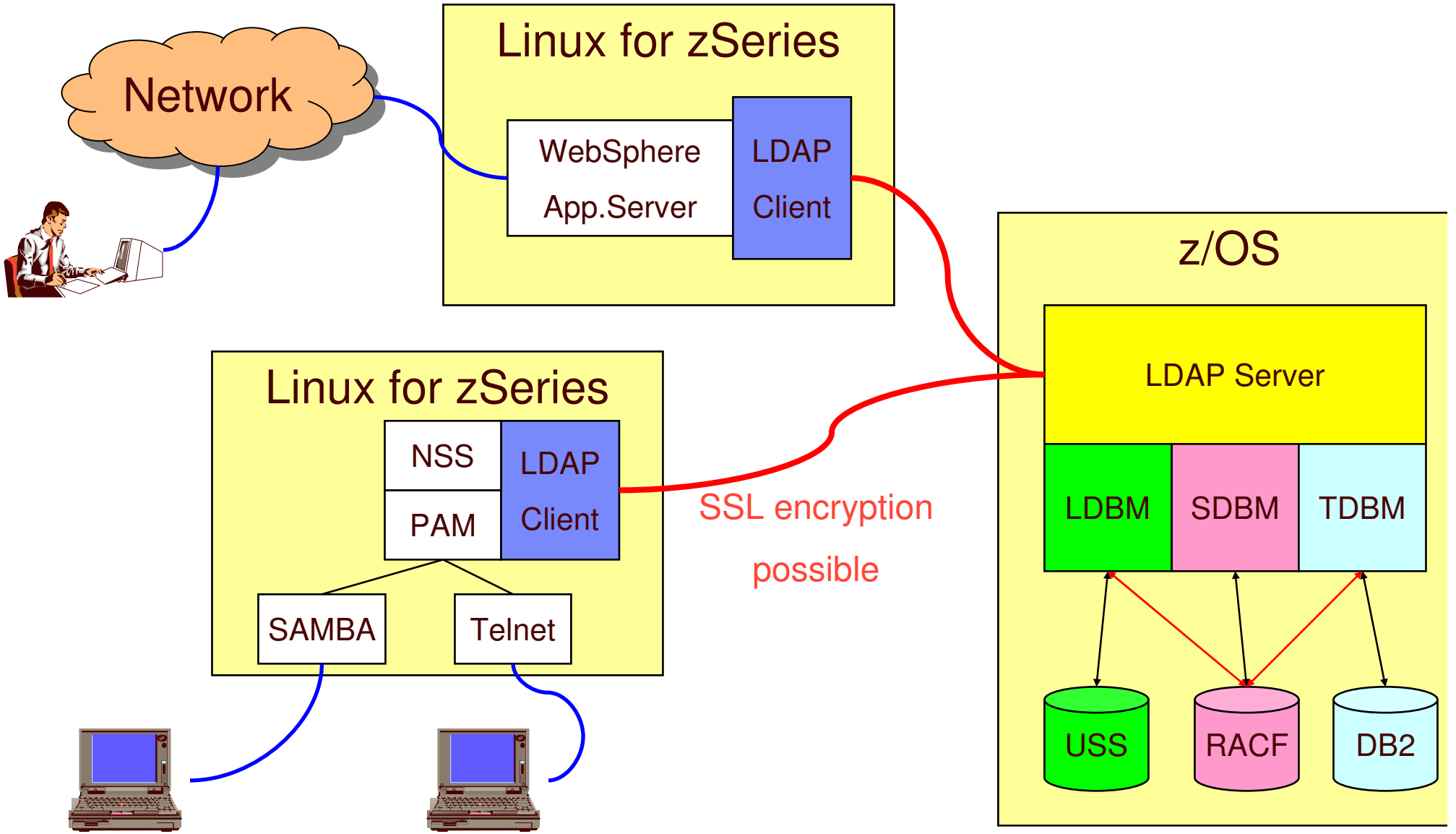
- **LDBM/TDBM uses an Access Control List (ACL) to control access to an entry**
- **Can specify LDBM, TDBM and SDBM (RACF) users and groups in ACL**
- **Can control access to individual attributes or to classes of attributes (normal, sensitive, critical, restricted and system)**
  - Attribute's access class defined in the schema
- **Use LDAP modify operation to set ACL and search operation to display ACL info**
  - Examples:

    aclentry: cn=jay,ou=LDAP,o=IBM,c=US:normal:rwsc:sensitive:rsc
    aclentry: racfid=u1234,profiletype=user,cn=myRacf:object:ad
    aclentry: group:cn=mgrs,o=IBM,c=US:at:userpassword:rwsc
    aclentry: grpu:racfid=g1,profiletype=group,cn=myRacf:normal:rwsc
- **Can propagate an entry's ACL to the subtree below it**

Special aclEntry “pseudo-DNs”

- **cn=anybody**
  - Applies when no other specific ACL value applies
  - Server can be configured to prevent anonymous binds
- **cn=authenticated**
  - Applies when the requestor has authenticated to the directory but no other specific ACL value applies
  - Meant to allow more access than cn=anybody ACL value
- **cn=this**
  - Applies when the requestor has authenticated with the same DN as the entry being accessed
  - Used to grant individuals access to their own entry
- **Example:**
  - aclentry: cn=anybody:normalize:rsc
  - aclentry: cn=authenticated:normal:rsc:sensitive:rs
  - aclentry: cn=this:normal:rscw:sensitive:rscw:critical:rsc

User Information and Authentication in LDAP



BM TDS RAS

- **Can be configured to:**
  - Monitor resources
    - DB2
      - TDBM
    - File system
      - LDBM and GDBM use HFS or zFS
    - Network
    - Client connections
      - Warning message when number of connections reach a level
      - Usage
        - SMF 83 audit records
        - Activity log
  - Automatically restart in the case of failure

References

- IBM TDS
  - SC23-5191 IBM Tivoli Directory Server Server Administration and Use for z/OS
- IBM TDS LDAP client
  - SA23-2214 IBM Tivoli Directory Server Client Programming for z/OS
- Redpaper: Linux on IBM zSeries and S/390: Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)
  - http://www.redbooks.ibm.com/redpapers/abstracts/redp0221.html
- PAM Documentation
  - http://www.kernel.org/pub/linux/libs/pam/
- NIS Schema for z/OS LDAP Server
  - ftp://www.redbooks.ibm.com/redbooks/REDP0221
- Contacting me
  - E-mail: furming@us.ibm.com