Click to add text

MQ v8 Security: Connection Authentication Multiple Certificates

Robert Parker

parrobe@uk.ibm.com

Agenda

- New Security Features
  - Changes for Channels using SSL/TLS Certificates
  - User ID & Password Connection Authentication

Click to add text

Changes for Channels using SSL/TLS Certificates

Agenda

- Requests for Enhancement
- Changes for Channels using SSL/TLS Certificates
  - Recap
  - Single Queue Manager Certificate
  - Per Channel Certificate
  - Certificate Matching

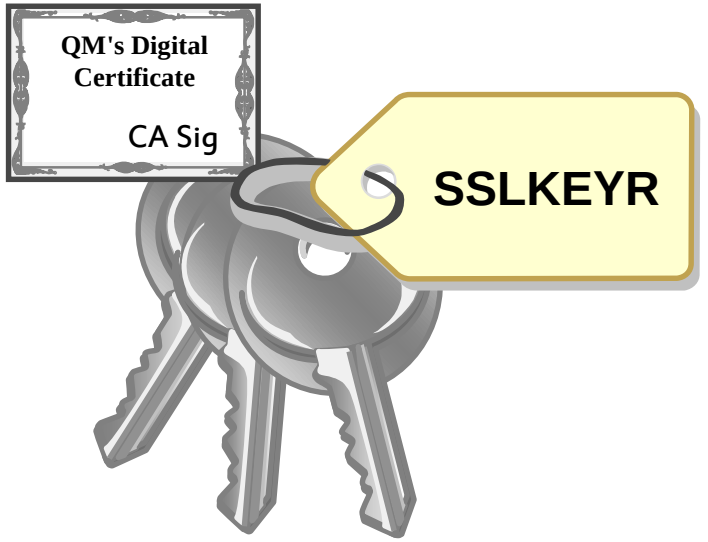
Request for Enhancement (26672)

Headline:	Requesting the enhancement to support for SSL certificate per channel or group of channels
ID:	26672

Details | Comments | Attachments | Reconsideration | Release plans

Status:	Under Consideration
Visibility:	Public
Description:	Currently mq supports only one default signed certificate per queue manager. When one firm is connecting with multiple external firms, then any of these external firms can pretend to be a different external firm, if they can guess the channel name and sslpeername and connect. Especailly if the channel names and sslpeers are following certain naming conventions. Another problme is, every time when the certificate chain changes, every party that is connecting to this qmgr needs to refresh their store with the new chain. So having a certficate per channel or group channels instead of one certificate for all channels on the queue manager is the solution here. We would like IBM to consider this as high priority.
Use case:	The description itself is covering the use case scenario.
Bookmarkable URL:	http://www.ibm.com/developerworks/rfe/execute?use_case=viewRfe&CR_ID=26672 A unique URL that you can bookmark and share with others.

Key Repository



- Contains Entity's own Digital Certificate
  - z/OS Queue Manager
    - ibmWebSphereMQ<QMgr Name> (mixed case) label
  - Distributed Queue Manager
    - ibmwebspheremq<qmgr name> (lower case) label
  - Client
    - ibmwebspheremq<logon userid> (lower case) label
  - Digital Certificates from various Certificat
- On z/OS Queue Managers
  - Keyring name
- On Unix®, Windows®, iSeries® QMgrs
  - Key database path
- Clients: mqclient.ini file
  - SSL Stanza – SSLKeyRepository
- MQCONNX (MQSCO structure)

ALTER QMGR SSLKEYR(CSQ1RING)

ALTER QMGR SSLKEYR('var/mqm/qmgrs/QM1/ssl/key')

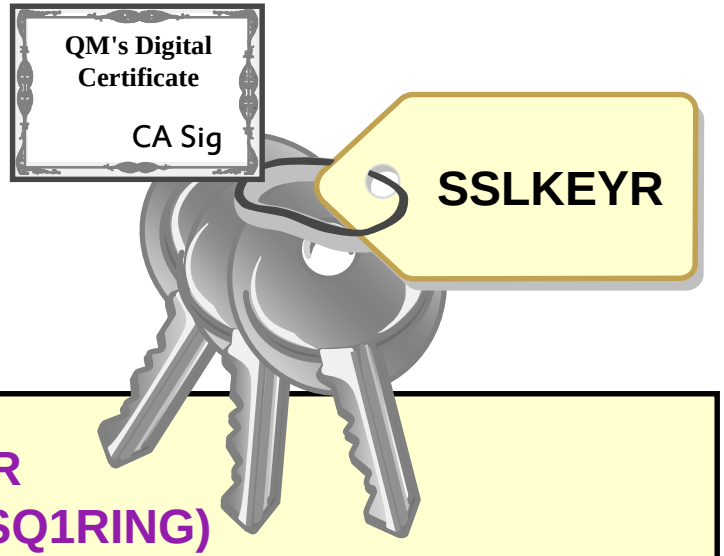
mqclient.ini
SSL:
SSLKeyRepository=C:\key

Single Queue Manager Certificate

- Name Queue Manager Certificate
  - Using CERTLABL attribute
- Name Client Certificate
  - mqclient.ini file SSL Stanza
    - CertificateLabel
  - MQCONNX (MQSCO structure)
    - CertificateLabel

```
MQCNO cno    = {MQCNO_DEFAULT};
MQSCO sco    = {MQSCO_DEFAULT};

cno.Version =  MQCNO_VERSION_4;
sco.Version =  MQSCO_VERSION_5;
memcpy(sco.KeyRepository, ... );
memcpy(sco.CertificateLabel,..);
cno.SSLConfigPtr = &sco;
MQCONNX(QMName,
        &cno,
        &hConn,
        &CompCode,
        &Reason);
```



ALTER QMGR
SSLKEYR(CSQ1RING)
CERTLABL('CSQ1Certificate')
CERTQSGL('SharedCert')

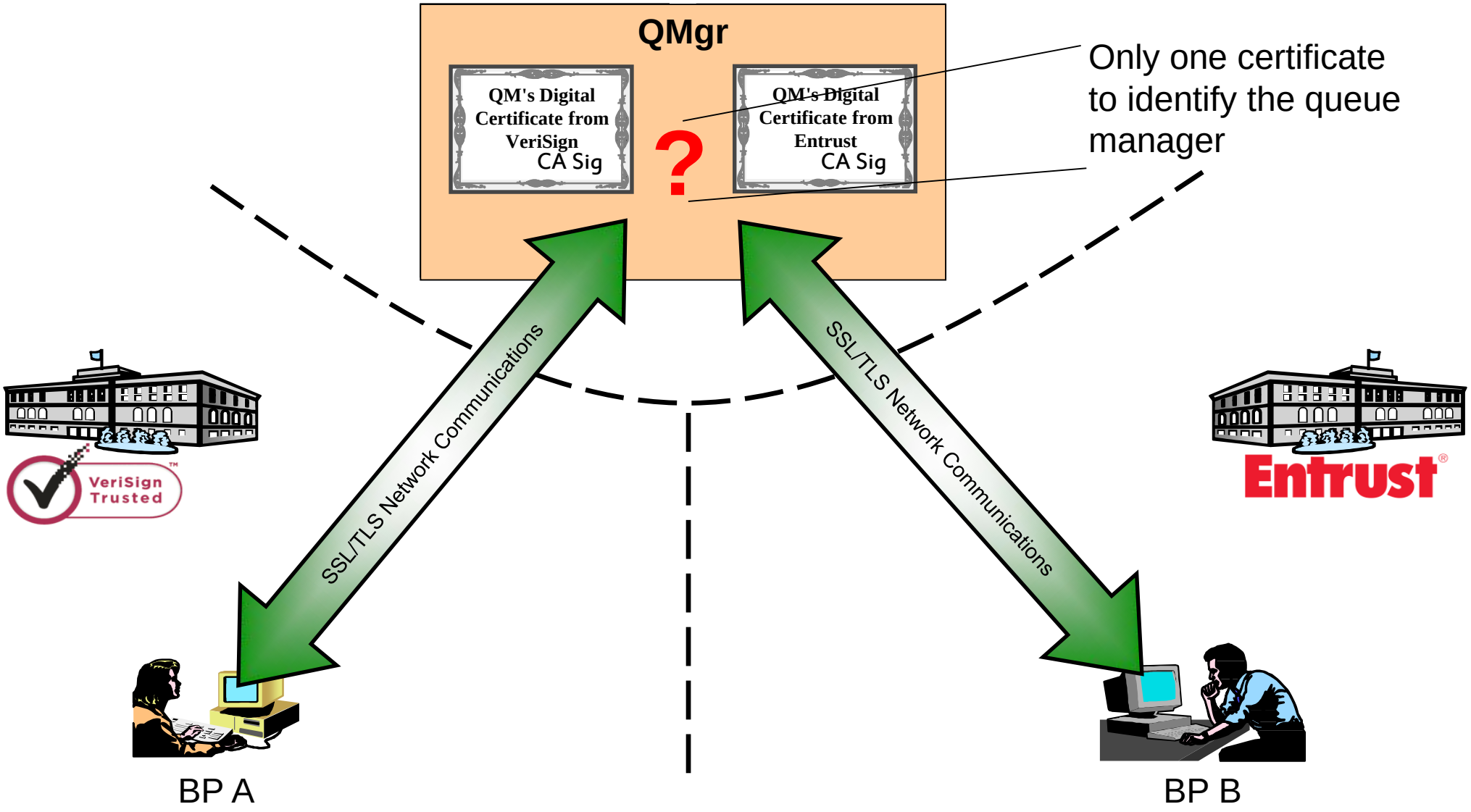
ALTER QMGR
SSLKEYR('var/mqm/qmgrs/QM1/ssl/key')
CERTLABL('QM1Certificate')

mqclient.ini
SSL:
SSLKeyRepository=C:\key
CertificateLabel=MyCert

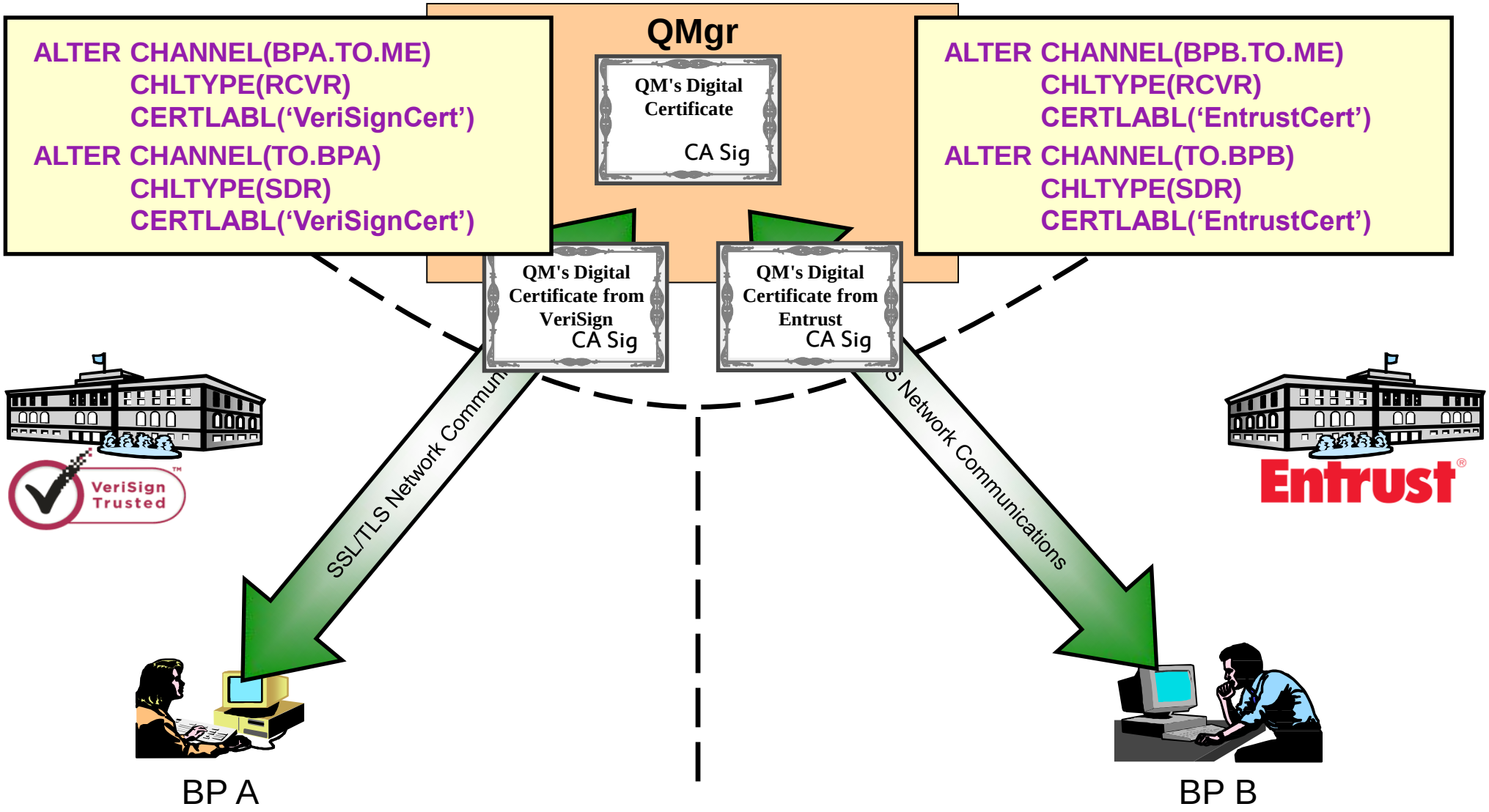
Use Cases

- Following company policy on certificate labelling
- Using the same certificate for more than one queue manager
  - Not that we would condone this!
- Migrating over to a new certificate when main certificate is ready to expire
  - Used to have to issue GSKit/RACF commands to rename certificate
    - ibmwebspheremqqm1 -> ibmwebspheremqqm1old
    - ibmwebspheremqqm1new -> ibmwebspheremqqm1
    - REFRESH SECURITY TYPE(SSL)
  - Now just MQ commands when the time comes
    - Current label is ‘QM1 Cert 2013’
    - ALTER QMGR CERTLABL(‘QM1 Cert 2014’)
    - REFRESH SECURITY TYPE(SSL)

Business Partners with different CA requirements



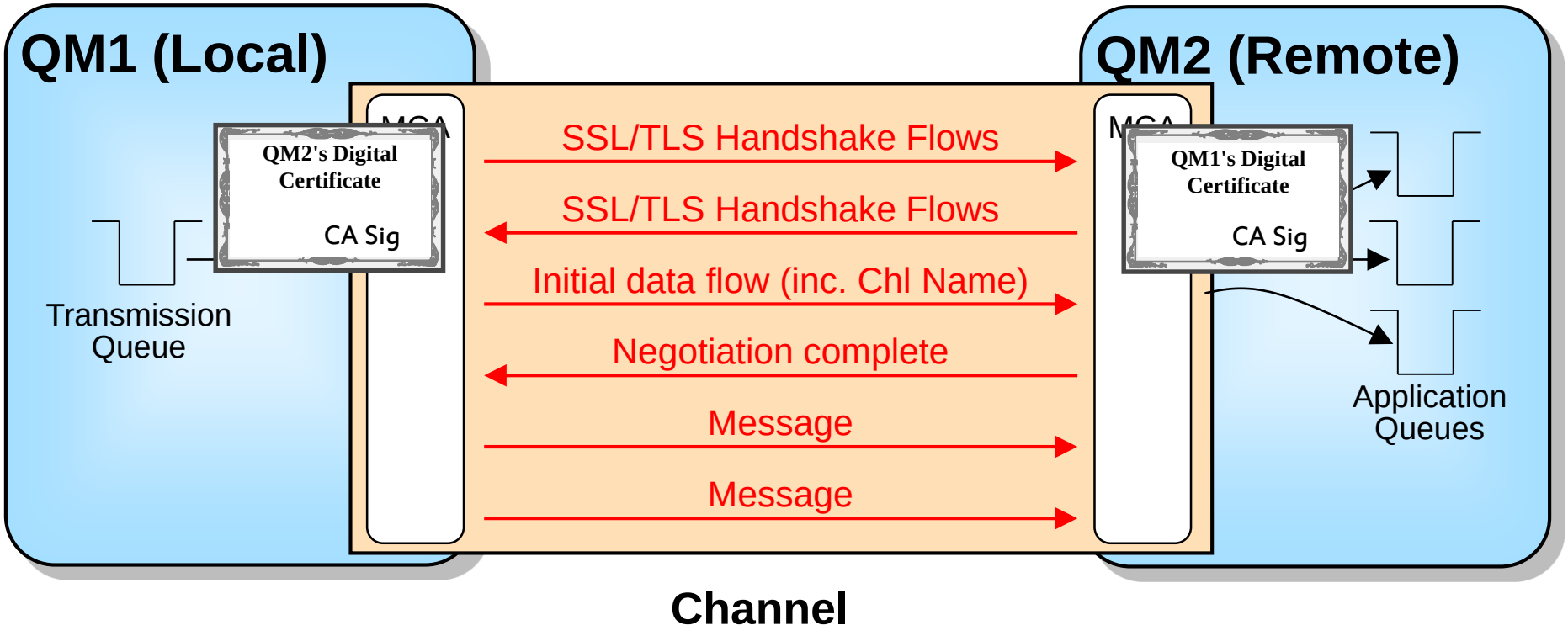
Certificate per Channel



Use Cases

- Business Partners with different CA requirements
- Connecting to machines with back level SSL support
  - Especially when you want to move to new certificate technology, e.g.
    - SHA-2 signed certificates
    - Elliptic Curve certificates

Why haven't we always done this?

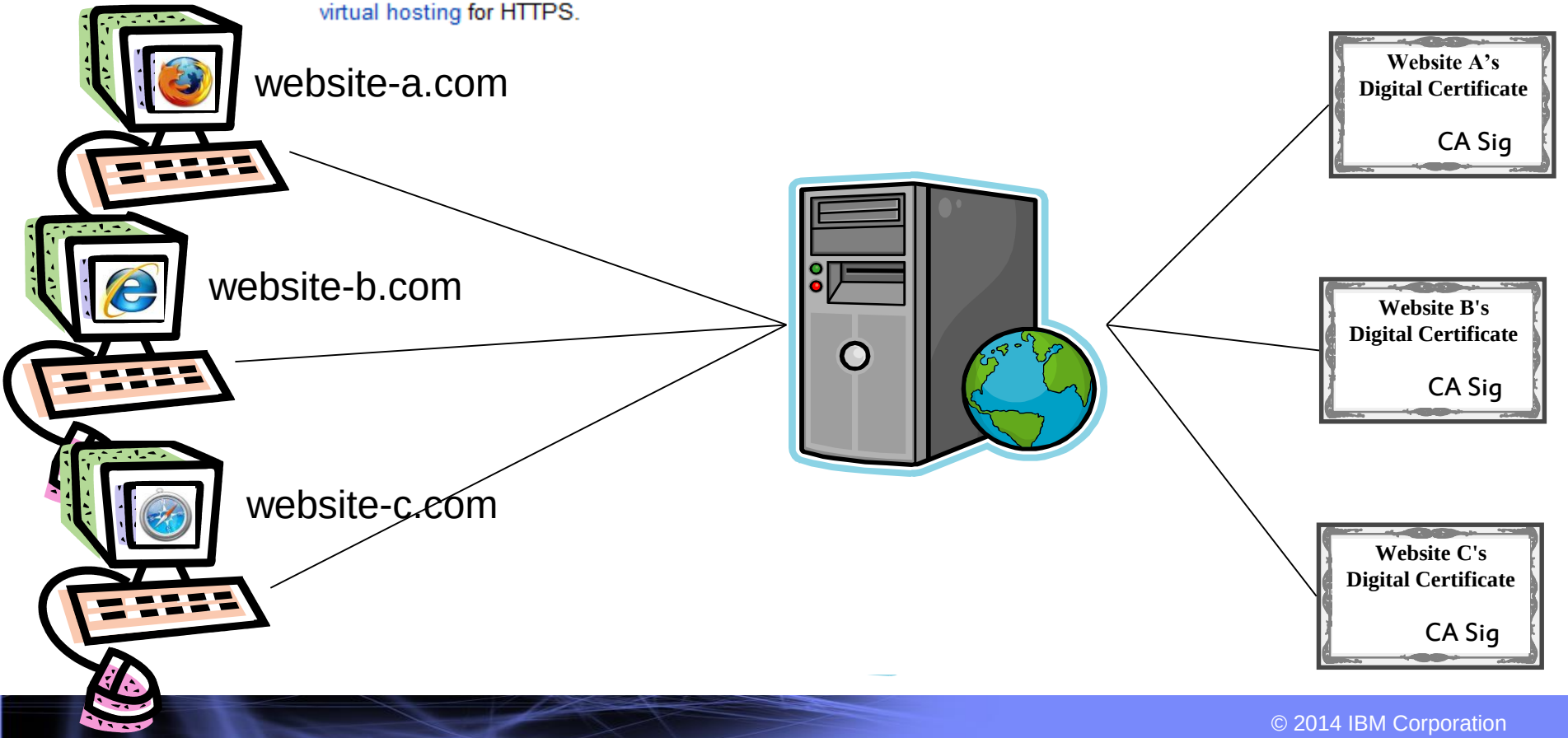


Server Name Indication

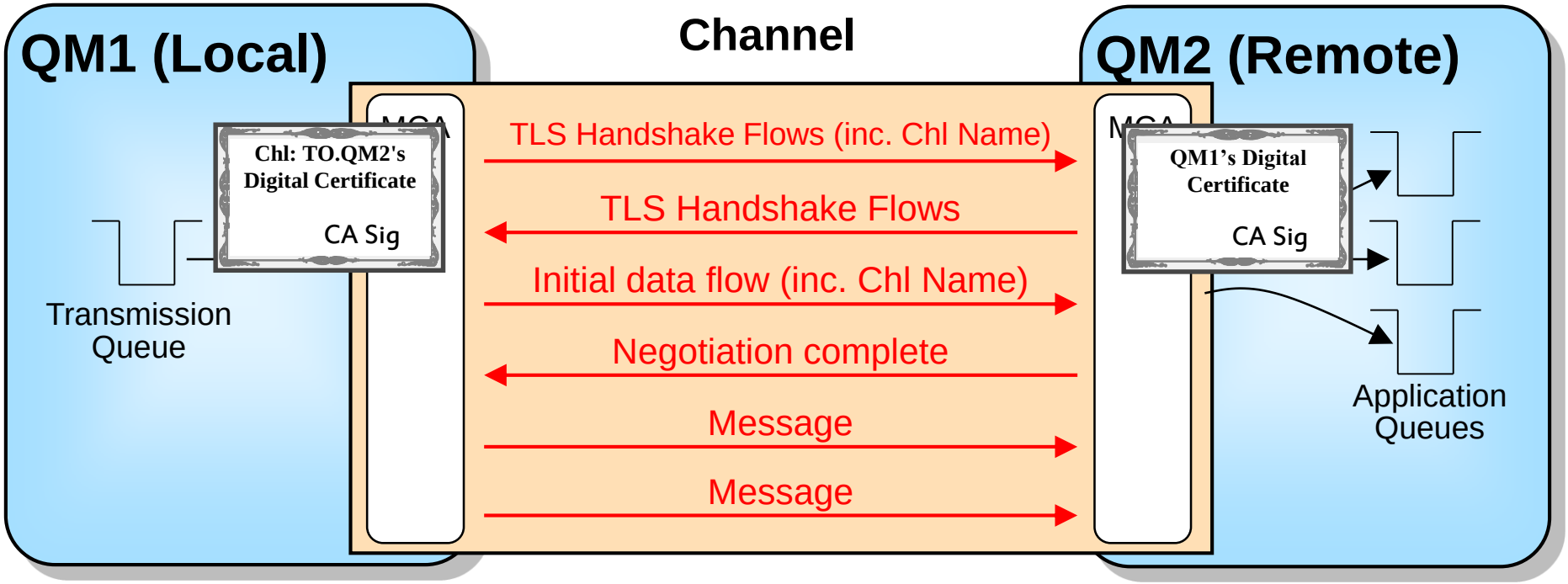
Server Name Indication

From Wikipedia, the free encyclopedia

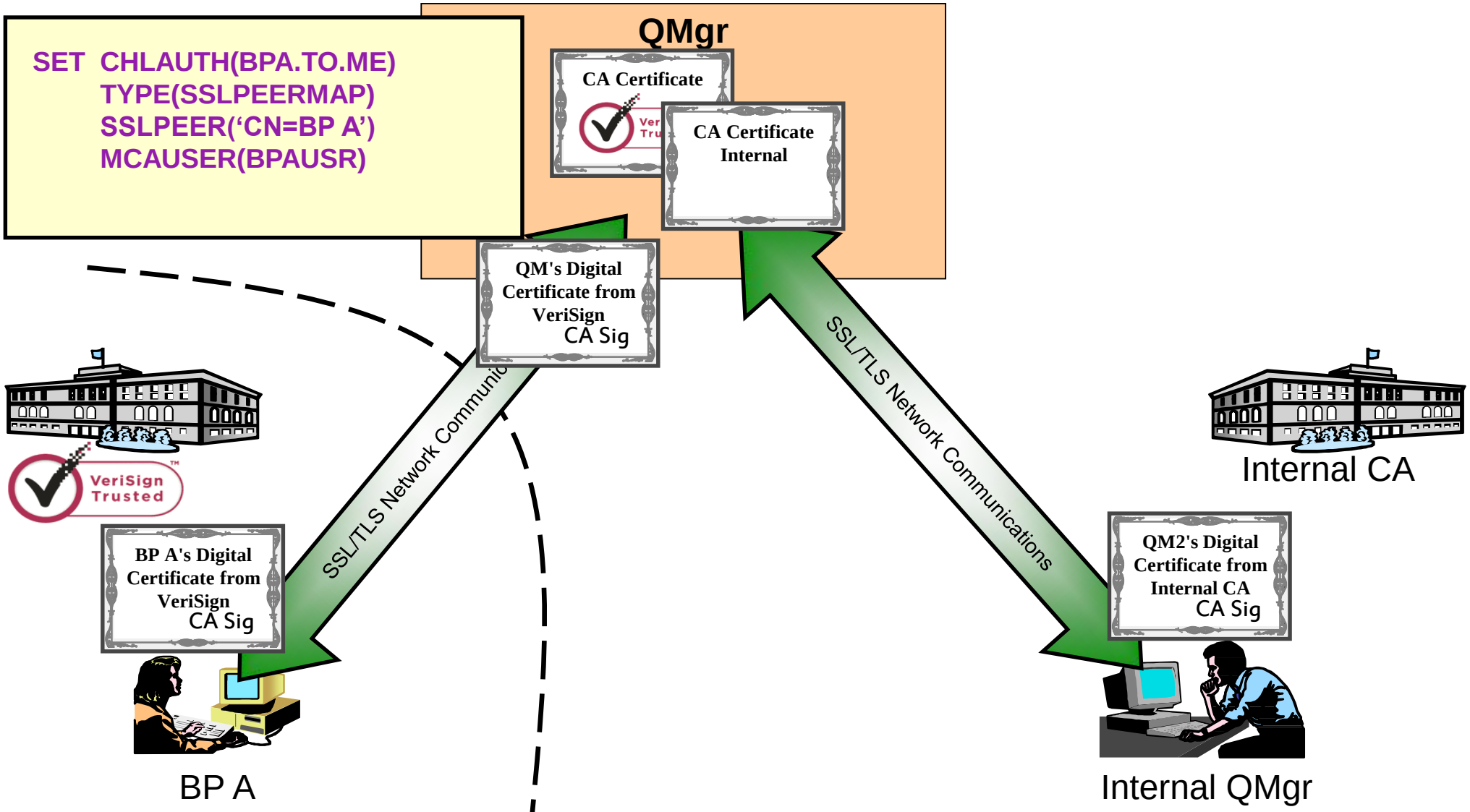
Server Name Indication (**SNI**) is an extension to the TLS protocol[1] that indicates what hostname the client is attempting to connect to at the start of the handshaking process. This allows a server to present multiple certificates on the same IP address and port number and hence allows multiple secure (HTTPS) websites (or any other Service over TLS) to be served off the same IP address without requiring all those sites to use the same certificate. It is the conceptual equivalent to HTTP/1.1 virtual hosting for HTTPS.



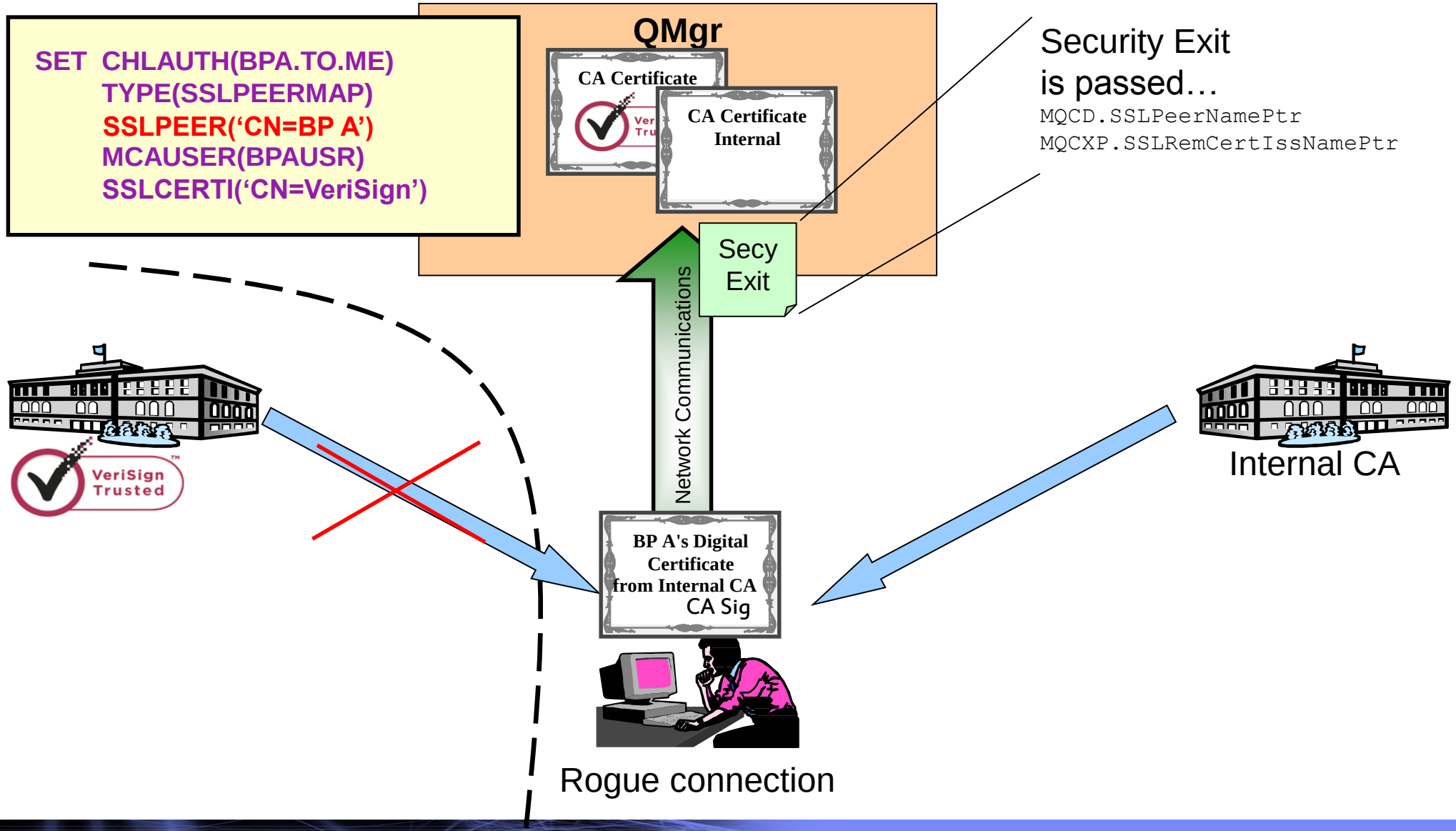
Using Server Name Indication (SNI) with a channel name



Our Business Partner Scenario again



Ensuring the Correct Certificate



Summary

- Changes for Channels using SSL/TLS Certificates
  - Single Queue Manager Certificate
    - `ALTER QMGR CERTLABL('My certificate name')`
  - Per Channel Certificate
    - `ALTER CHANNEL … CERTLABL('This channel certificate')`
  - Certificate Matching
    - ```
      SET CHLAUTH('*')
          TYPE(SSLPEERMAP)
          SSLPEER('CN=Morag Hughson')
          SSLCERTI('CN=IBM CA')
          MCAUSER('hughson')
      ```
      ```

Click to add text

Questions?

Click to add text

User ID & Password Connection Authentication

Agenda

- Requests for Enhancement
- Connection Authentication
  - Configuration
  - Application Changes (or not)
  - Protecting your password across a network
  - User Repositories

Request for Enhancement (22568)

Headline:	Password validation
ID:	22568

Details | Comments | Attachments | Reconsideration | Release plans

Status:	Uncommitted Candidate
Visibility:	Public
Description:	Password validation of Client connections to be delivered for all platforms. CSQ4BCX3 is supplied for z/OS. We need the similar functionality for various platforms (Windows, Linux, AIX, Solaris, HP-NSK). This would help us to prove to audit that we know who is connecting.
Use case:	Ease a secure integration with MO71 and MQ Explorer, so we can please law and audit teams. This will remove the need for using SSL to assure the identity of MQ administrators.
Bookmarkable URL:	http://www.ibm.com/developerworks/rfe/execute?use_case=viewRfe&CR_ID=22568 A unique URL that you can bookmark and share with others.

Request for Enhancement (30709)

Headline: WMQ Authentication via LDAP

ID: 30709

Details | Comments | Attachments | Reconsideration | Release plans

Status: Uncommitted Candidate

Visibility: Public

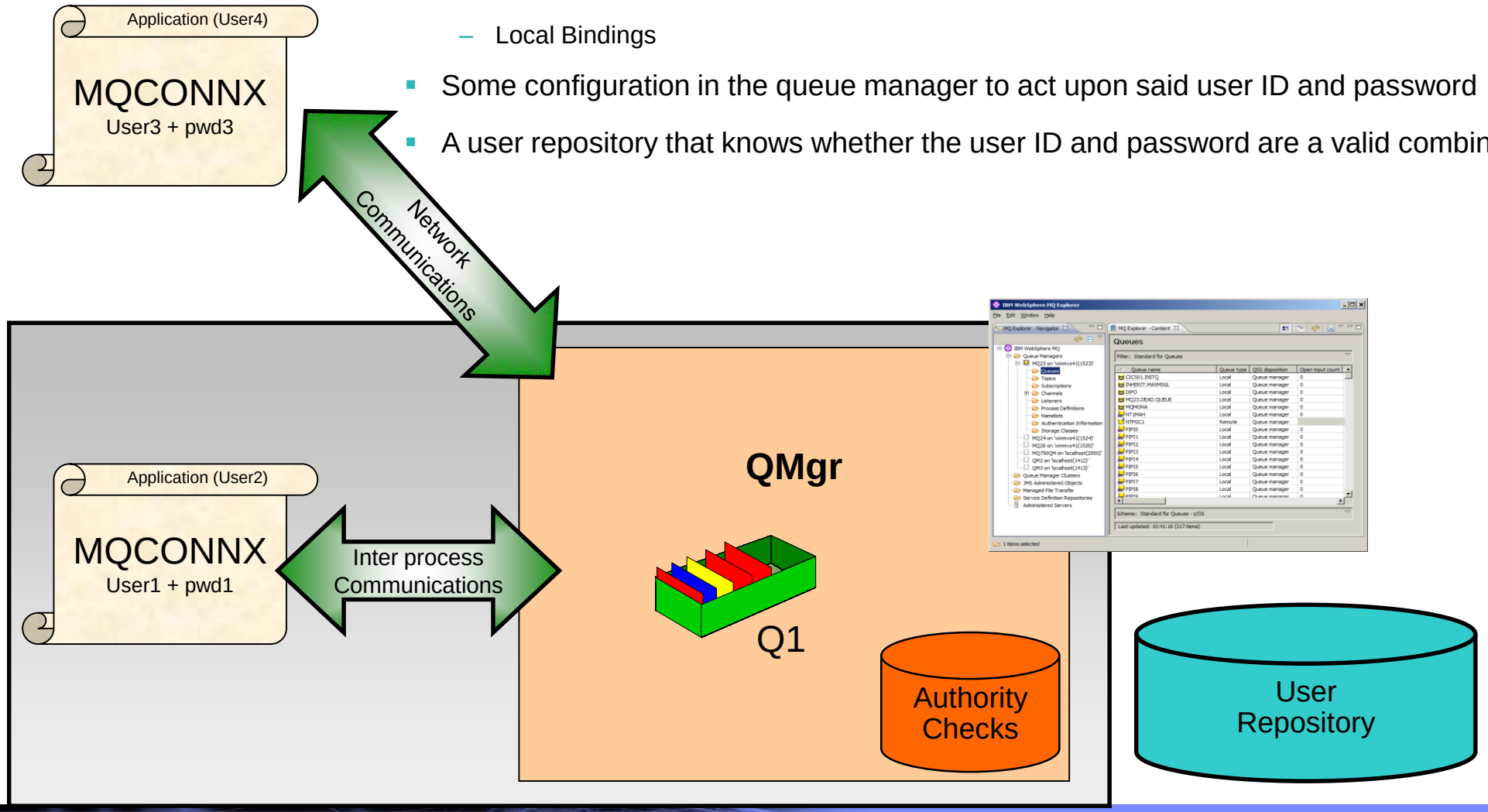
Description: Authenticate client connections with a central LDAP server. Instead of using the O/S for authentication we would like to be able to hand off a user/password combination to an LDAP server for authentication.

Use case: Clients would supply a user/password for authentication that would be validated by a central LDAP server, authorisation could be handled in the existing manner. The LDAP authentication could occur over SSL or plain TCP.

Bookmarkable URL: http://www.ibm.com/developerworks/rfe/execute?use_case=viewRfe&CR_ID=30709
A unique URL that you can bookmark and share with others.

Connection Authentication – What is it?

- The ability for an application to provide a user ID and password
  - Client
  - Local Bindings
- Some configuration in the queue manager to act upon said user ID and password
- A user repository that knows whether the user ID and password are a valid combination



Connection Authentication – Configuration

CHCK…
NONE
OPTIONAL
REQUIRED
REQDADM

```
ALTER   QMGR CONNAUTH(USE.PW)

DEFINE  AUTHINFO(USE.PW) AUTHTYPE(xxxxxx)
        FAILDLAY(1) CHCKLOCL(OPTIONAL)
        CHCKCLNT(REQUIRED)

REFRESH SECURITY TYPE(CONNAUTH)
```

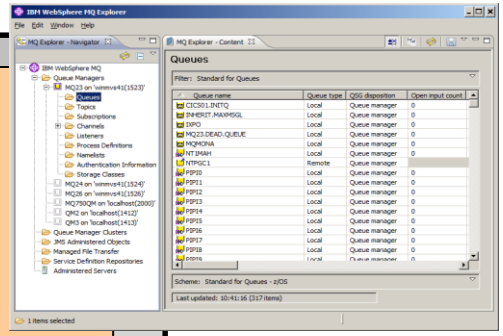
Application (User4)

MQCONNX

~~User3 + pwd3~~

MQRC_NOT_AUTHORIZED (2035)

Network Communications



QMgr

Application (User2)

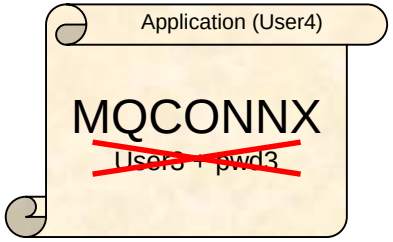
MQCONNX

~~User1 + pwd1~~

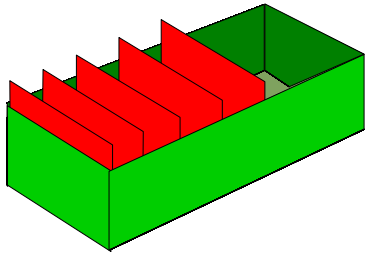
Inter process Communications

MQRC_NONE (0)

Connection Authentication – Error notification



MQRC_NOT_AUTHORIZED (2035)



SYSTEM.ADMIN.QMGR.EVENT

ALTER QMGR AUTHOREV(ENABLED)

- Application
  - MQRC_NOT_AUTHORIZED (2035)
- Administrator
  - Error message
- Monitoring Tool
  - Not Authorized Event message (Type 1 – Connect)
  - MQRQ_CONN_NOT_AUTHORIZED (existing)
    - Connection not authorized.
  - MQRQ_CSP_NOT_AUTHORIZED (new)
    - User ID and password not authorized.
  - Additional field to existing connect event
    - MQCACF_CSP_USER_IDENTIFIER

Connection Authentication – Configuration Granularity

CHCKCLNT
ASQMGR
REQUIRED
REQDADM

```
DEFINE  AUTHINFO(USE.PW) AUTHTYPE(xxxxxx)
        CHCKCLNT(OPTIONAL)

SET     CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*')
        USERSRC(CHANNEL) CHCKCLNT(REQUIRED)

SET     CHLAUTH('*') TYPE(SSLPEERMAP)
        SSLPEER('CN=*') USERSRC(CHANNEL)
        CHCKCLNT(ASQMGR)
```

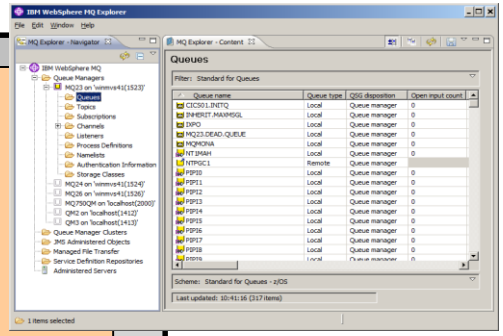
Application (User4)

MQCONNX

~~User3 + pwd3~~

MQRC_NOT_AUTHORIZED (2035)

Clear Network Communications



User's Digital Certificate

Application (User2)

MQCONNX

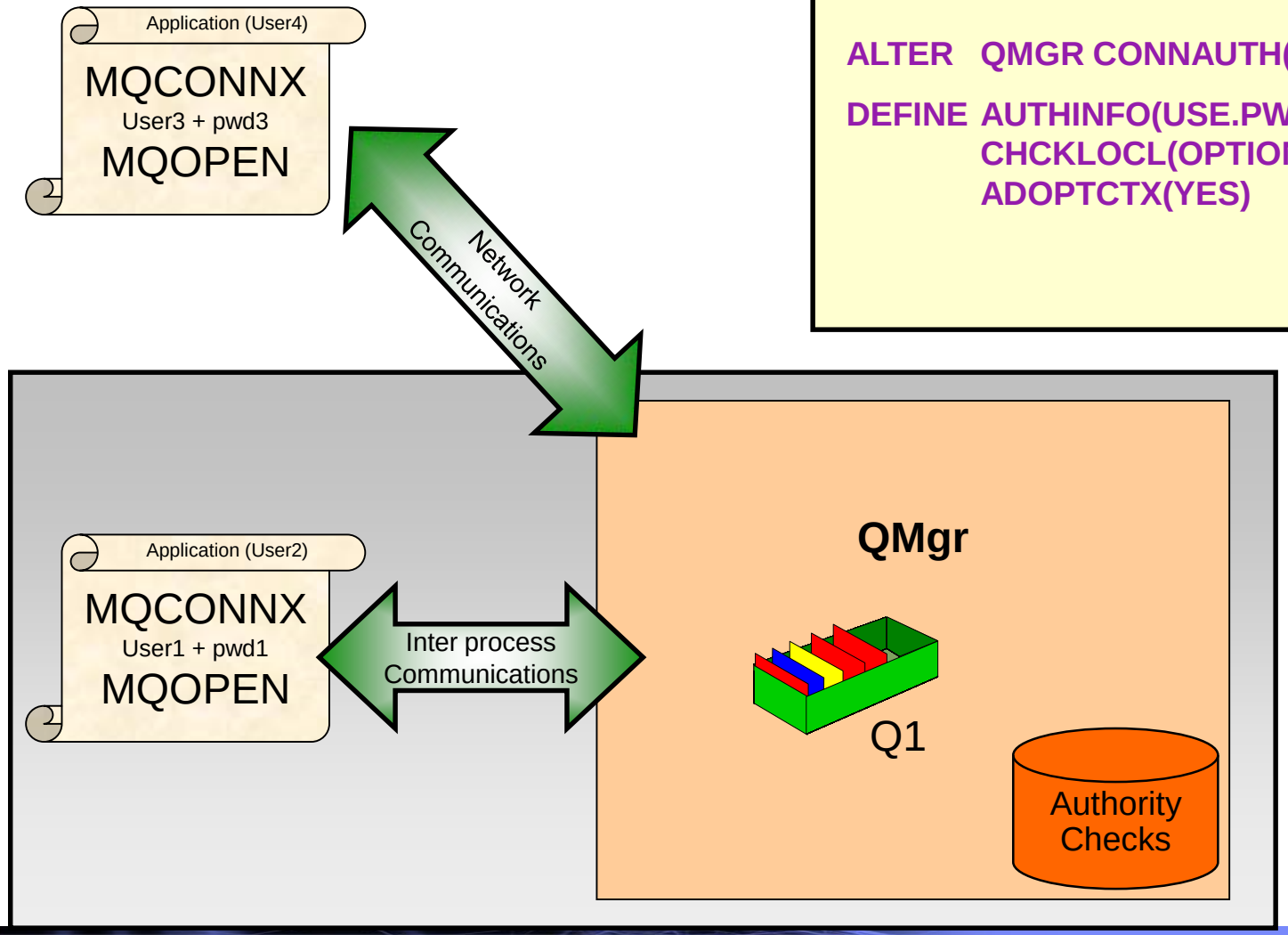
~~User1 + pwd1~~

MQRC_NONE (0)

SSL/TLS Network Communications

QMgr

Connection Authentication – Relationship to Authorization



ALTER QMGR CONNAUTH(USE.PWD)

DEFINE AUTHINFO(USE.PWD) AUTHTYPE(*xxxxxx*)
CHCKLOCL(OPTIONAL) CHCKCLNT(REQUIRED)
ADOPTCTX(YES)

Authority Records
Q1: User1 +put
Q1: User2 +none
Q1: User3 +get
Q1: User4 +none

Connection Authentication – Application changes

- Code changes
  - Procedural – MQCSP on MQCONNX
  - OO classes – MQEnvironment
  - JMS/XMS – createConnection
  - XAOpen string
- Alternatively Exits can provide MQCSP
  - Client side security exit
    - Provided
  - Client side Pre-conn exit

Procedural MQI changes

- MQCSP structure
  - Connection Security Parameters
  - User ID and password
- MQCNO structure
  - Connection Options
- WebSphere MQ V6
  - Passed to OAM (Dist only)
  - Also passed to Security Exit
    - Both z/OS and Distributed
    - MQXR_SEC_PARMS
- WebSphere MQ V8
  - Acted upon by the queue manager (all platforms)

```
MQCNO cno = {MQCNO_DEFAULT};

cno.Version = MQCNO_VERSION_5;

cno.SecurityParmsPtr = &csp;

MQCONNX(QMName,
        &cno,
        &hConn,
        &CompCode,
        &Reason);
```

```
MQCSP csp = {MQCSP_DEFAULT};

csp.AuthenticationType = MQCSP_AUTH_USER_ID_AND_PWD;
csp.CSPUserIdPtr       = "hughson";
csp.CSPUserIdLength    = 7;          /* Max: MQ_CLIENT_USER_ID_LENGTH */
csp.CSPPasswordPtr     = "passw0rd";
csp.CSPPasswordLength  = 8;          /* Max: MQ_CSP_PASSWORD_LENGTH   */
```

Object Oriented MQ classes changes

```
MQEnvironment.properties = new Hashtable();
MQEnvironment.userID = "hughson";
MQEnvironment.password ="passw0rd";

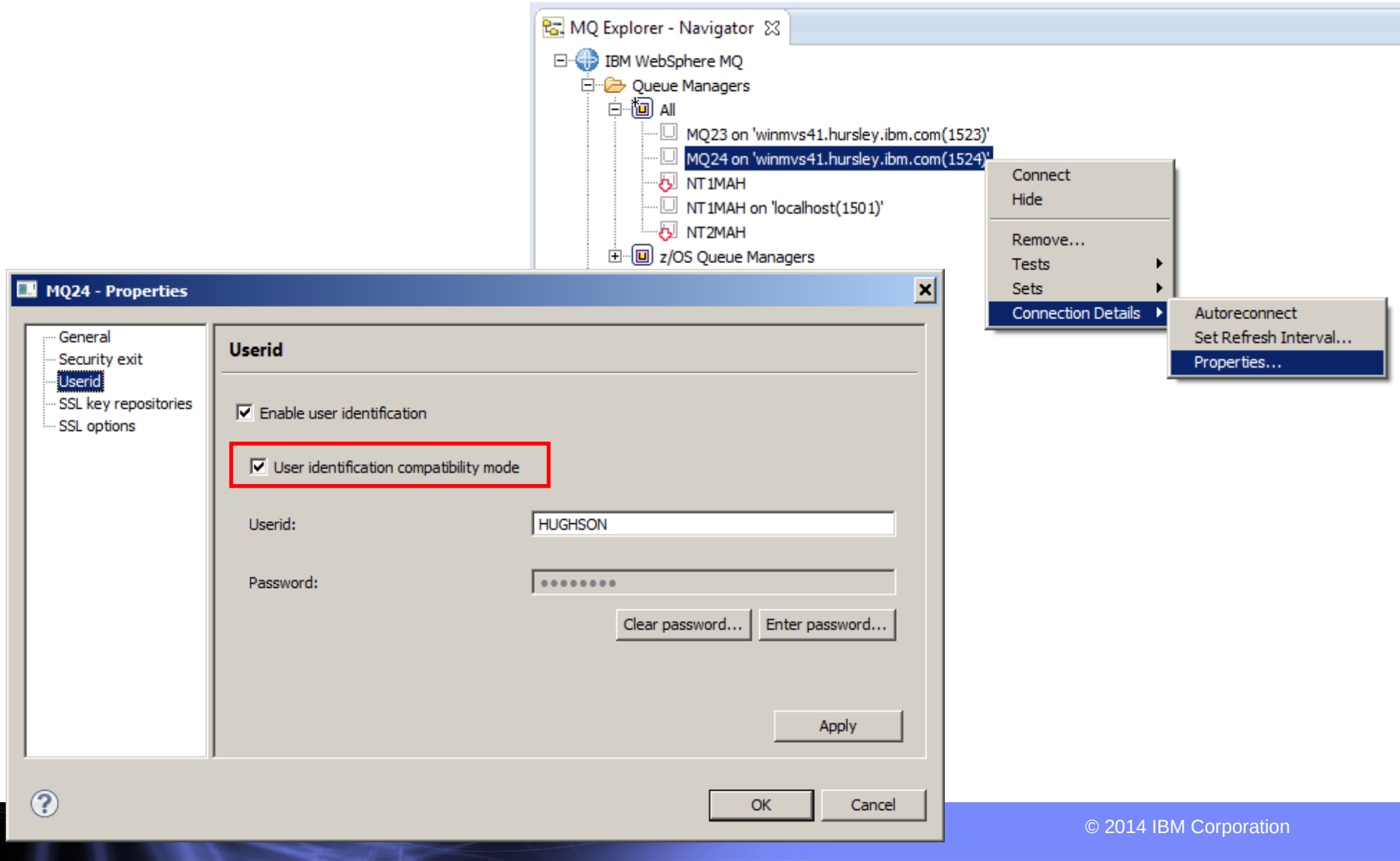
System.out.println("Connecting to queue manager");
MQQueueManager qMgr = new MQQueueManager(QMName);
```

JMS/XMS classes changes

```
cf = getCF();

System.out.println("Creating the Connection with UID and Password");
Connection conn = cf.createConnection("hughson", "passw0rd");
```

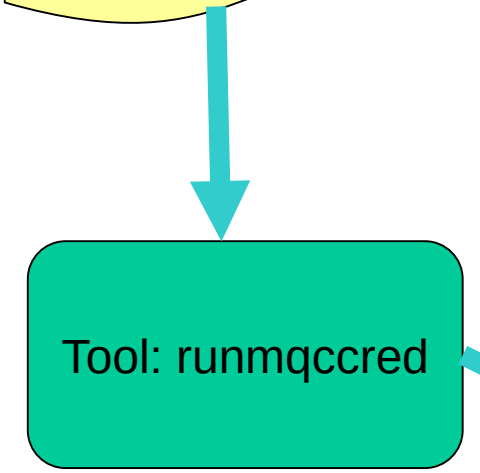
Using it from the MQ Explorer GUI



Client side Security Exit

mqccred.ini

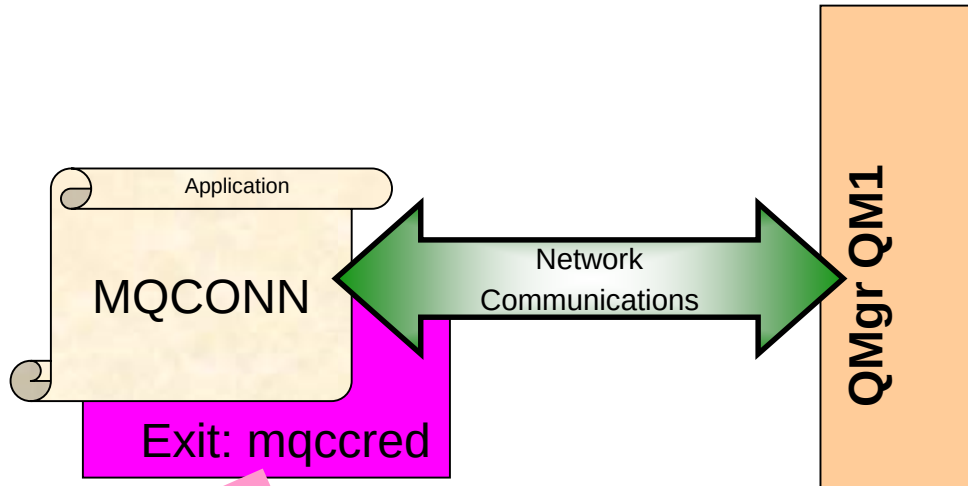
```
AllQueueManagers:
   User=abc
   password=newpw
QueueManager:
   Name=QMA
   User=user1
   password=passw0rd
```



mqccred.ini

```
AllQueueManagers:
   User=abc
   OPW=%^&aervrgtsr
QueueManager:
   Name=QM1
   User=user1
   OPW=H&^dbgfh
```

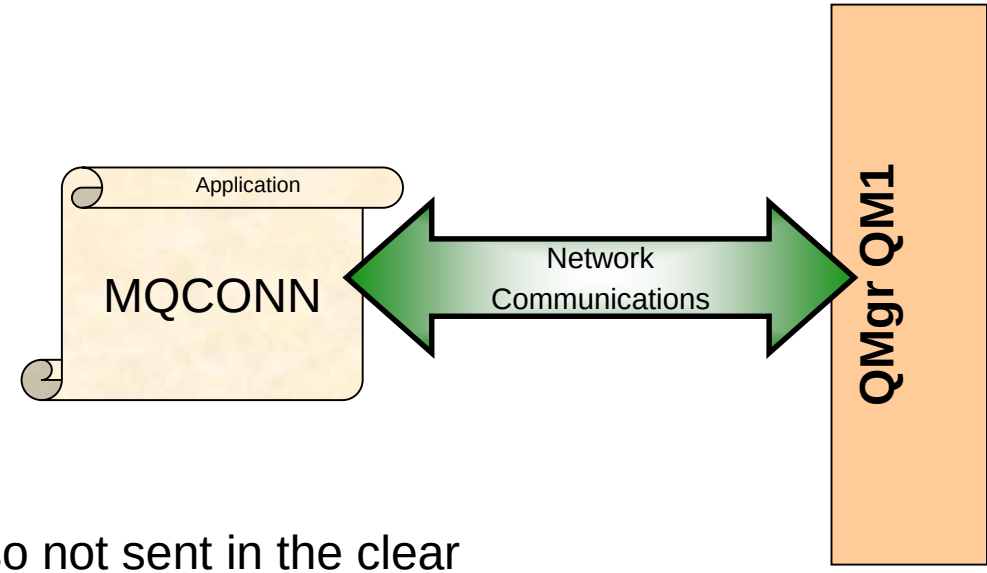
File permissions



Exit can be used by clients from V7.0.1 and later (by copying from a V8 installation)

Protecting your password across a network

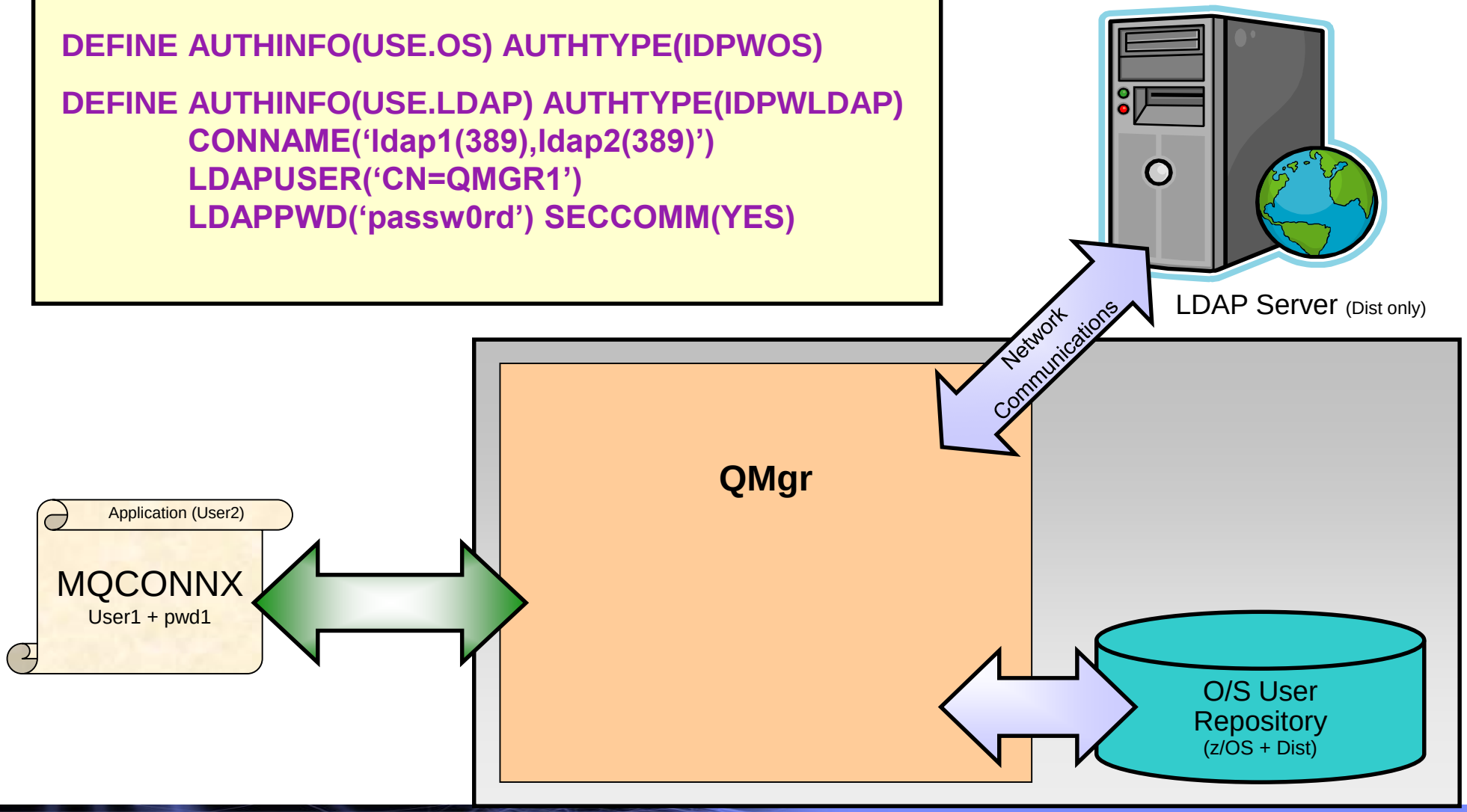
- Use SSL/TLS
  - Perhaps with anonymous clients
- If no SSL/TLS
  - If both ends are V8
  - MQ Code will protect the password – so not sent in the clear
- If client is < V8
  - No MQ password protection
  - Consider SSL/TLS



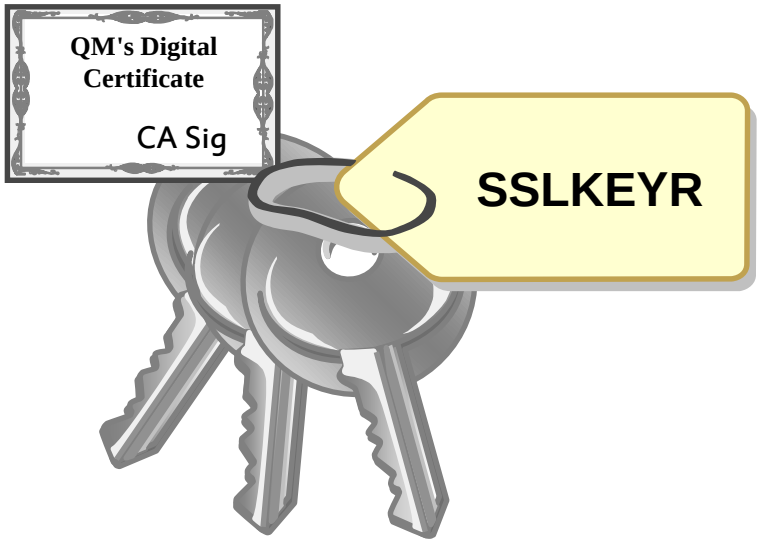
Connection Authentication – User Repositories

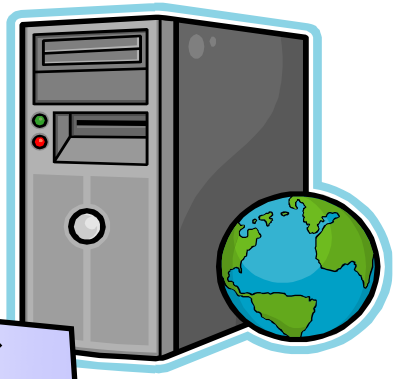
```
DEFINE AUTHINFO(USE.OS) AUTHTYPE(IDPWOS)

DEFINE AUTHINFO(USE.LDAP) AUTHTYPE(IDPWLDAP)
       CONNAME('ldap1(389),ldap2(389)')
       LDAPUSER('CN=QMGR1')
       LDAPPWD('passw0rd') SECCOMM(YES)
```



Secure connection to an LDAP Server





LDAP Server

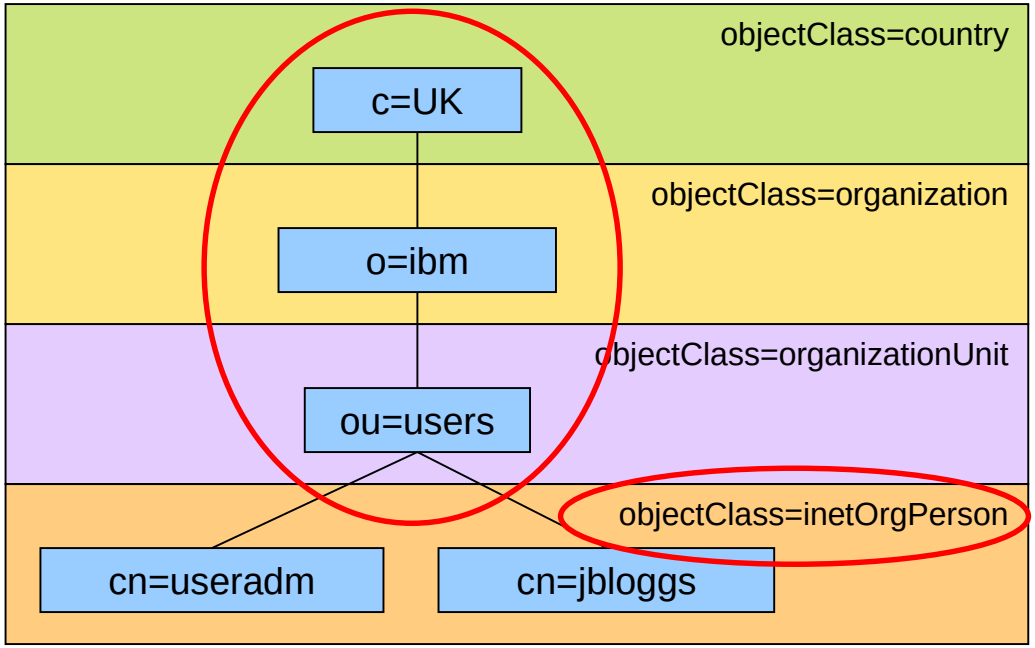
Network Communications

```
ALTER   QMGR CONNAUTH(USE.LDAP)
        SSLFIPS(NO) SUITEB(NONE)
        CERTLABL('ibmwebspheremqqm1')
        SSLKEYR('var/mqm/qmgrs/QM1/ssl/key')

DEFINE  AUTHINFO(USE.LDAP)
        AUTHTYPE(IDPWLDAP)
        SECCOMM(YES)
        CONNAME('ldapserver(389)')
```

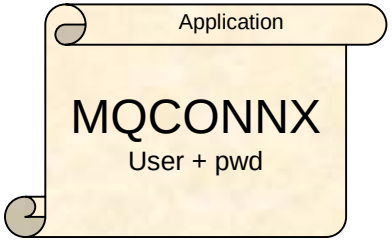
```
DISPLAY  QMSTATUS
         LDAPCONN
```

LDAP User Repository



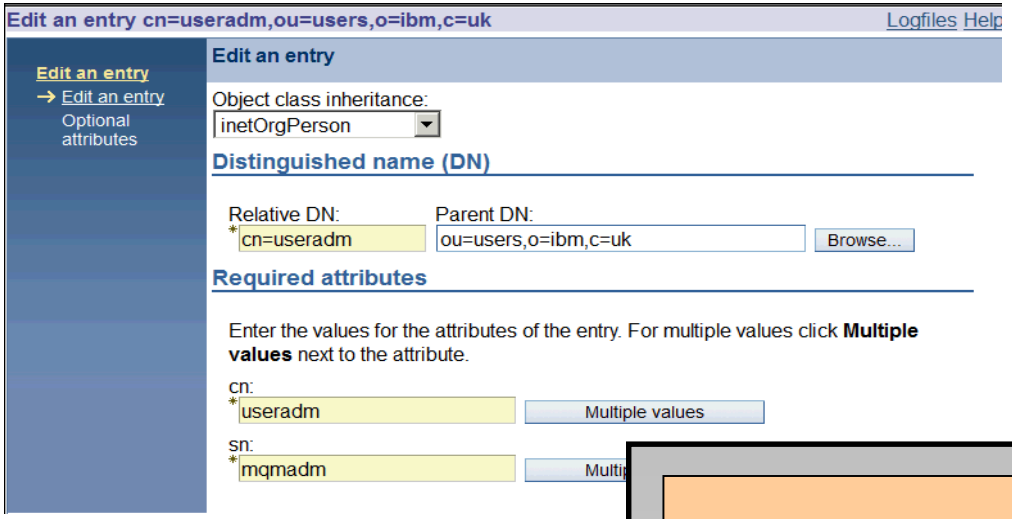
LDAP Server

```
DEFINE AUTHINFO(USE.LDAP)
       AUTHTYPE(IDPWLDAP)
       CONNAME('ldapserver(389)')
       CLASSUSR('inetOrgPerson')
       BASEDNU('ou=users,o=ibm,c=uk')
       USRFIELD('cn')
```

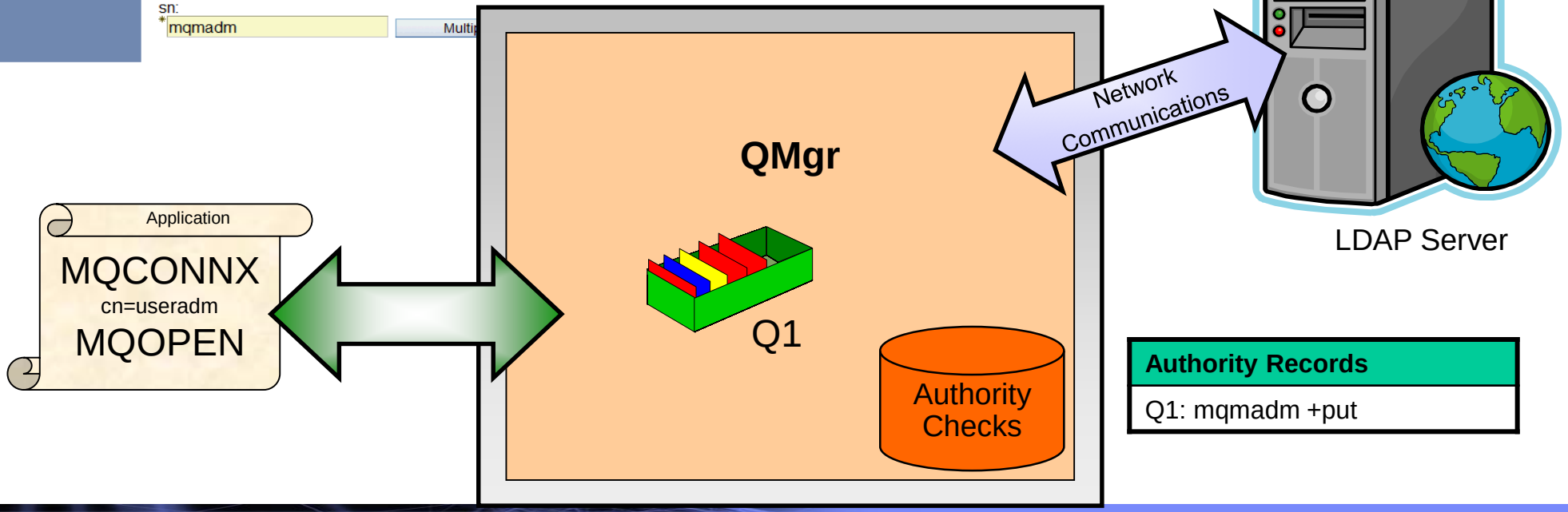


Application provides	USRFIELD	BASEDNU
cn=useradm,ou=users,o=ibm,c=uk		
cn=useradm		Adds ou=users,o=ibm,c=uk
useradm	Adds cn=	Adds ou=users,o=ibm,c=uk

Connection Authentication – Relationship to Authorization – LDAP




```
DEFINE AUTHINFO(USE.LDAP)
       AUTHTYPE(IDPWLDAP)
       CONNAME('ldap(389)')
       ADOPTCTX(YES)
       SHORTUSR('sn')
```



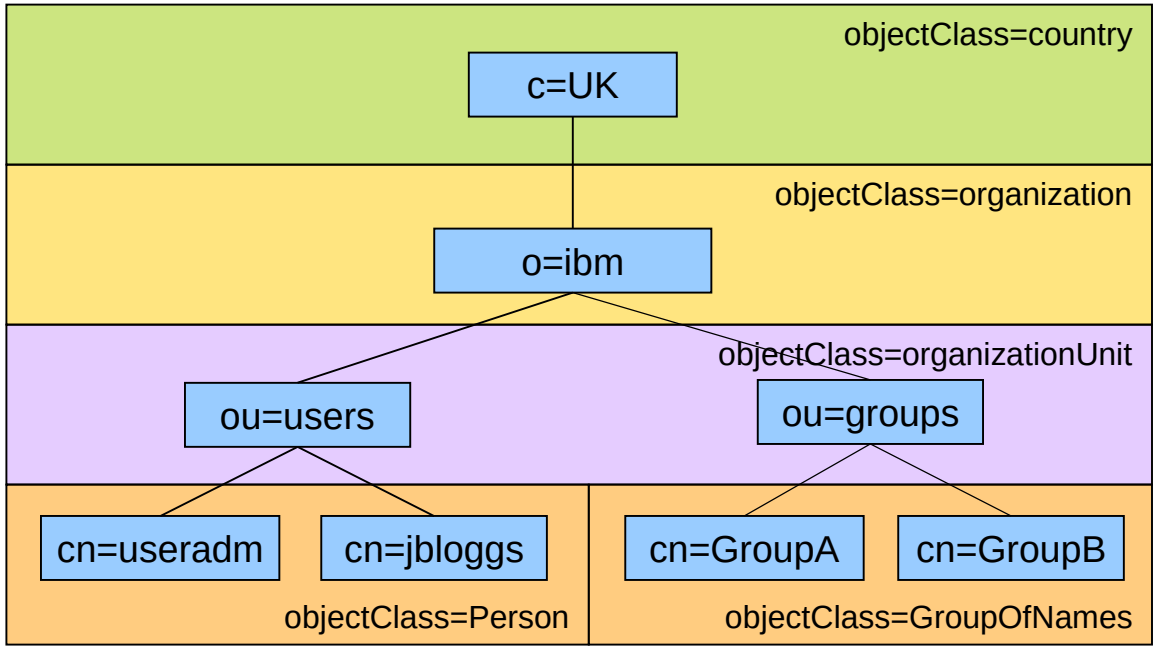
Authority Records
Q1: mqmadm +put

LDAP Authorization

NOTE: The following is only valid for MQ 8.0.0.2 on UNIX platforms only.

- From Connection Authentication we know:
  - We can adopt an authenticated user to be used for authorisation checks
  - MQ can be configured to use LDAP for authentication
- As such LDAP groups can users can be used for authorization
  - In addition to the previously mentioned LDAP ADOPTCTX(YES) option.
- However, if you use OS for authentication you can only use OS for authorization
  - Unlike where if you use LDAP for authentication you can use OS for authorization.

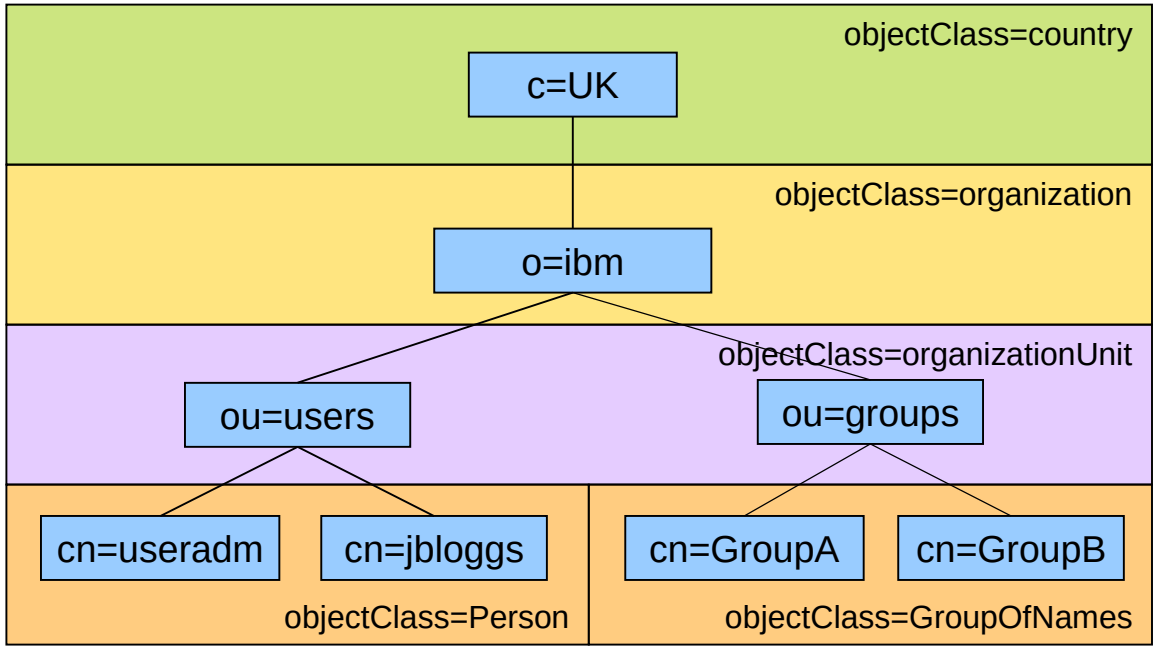
LDAP Authorization




```
DEFINE AUTHINFO('USE.LDAP')
AUTHTYPE(IDPWLDAP)
CONNAME(‘ldap.machine.host(398)’)
ADOPTCTX(YES)
USRFIELD('cn')
BASEDNU('ou=users,o=ibm,c=UK')
CLASSUSR(‘Person')
SHORTUSR('sn')

AUTHORMD(XXXXXX)
FINDGRP(XXXXXX)
GRPFIELD('cn')
BASEDNG('ou=groups,o=mqst')
CLASSGRP(‘GroupOfNames')
```

LDAP Authorization

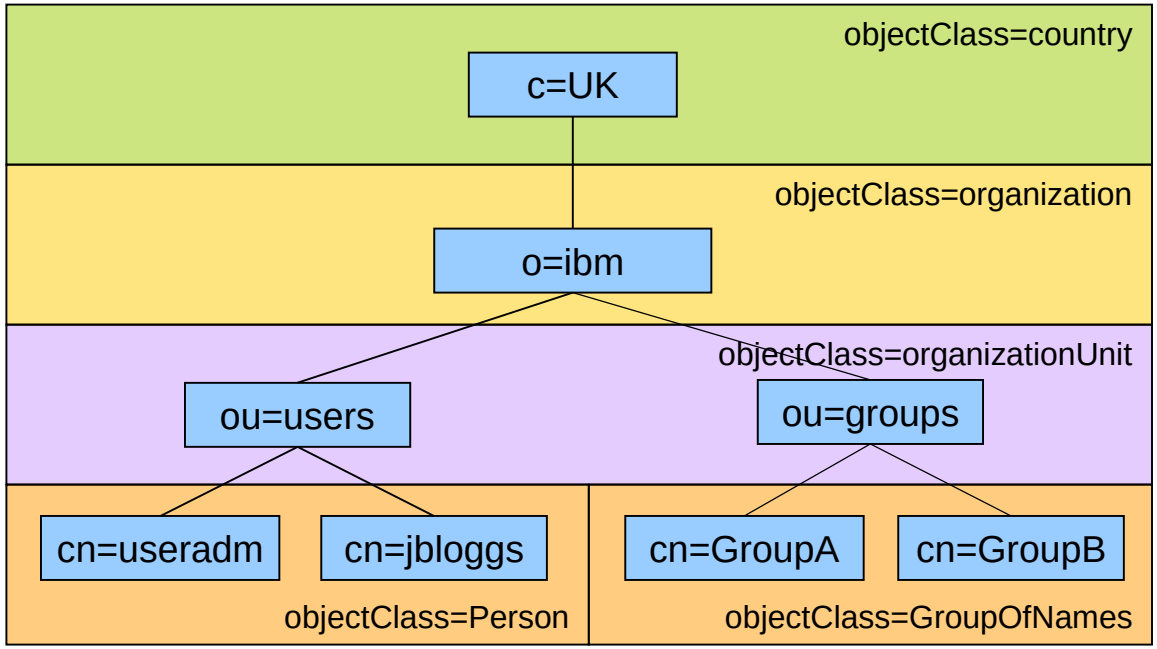


dn: cn=useradm,ou=users,o=ibm,c=UK
objectClass: Person
cn: useradm
sn: admuser
userPassword: xxxxxx
memberof:
cn=GroupA,ou=groups,o=ibm,c=UK

OR

dn: cn=GroupA,ou=groups,o=ibm,c=UK
objectClass: GroupOfNames
cn: GroupA
member:
cn=useradm,ou=users,o=ibm,c=UK
member:
cn=jbloggs,ou=users,o=ibm,c=UK

LDAP Authorization

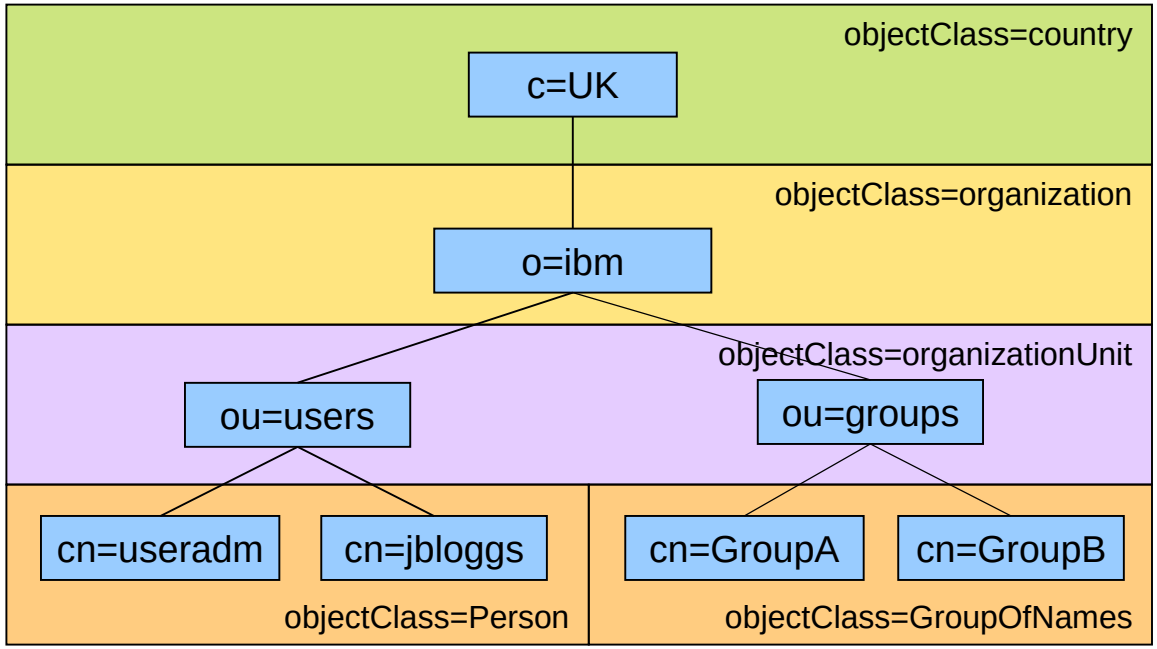


dn: cn=useradm,ou=users,o=ibm,c=UK
objectClass: Person
cn: useradm
sn: admuser
userPassword: xxxxxx
memberof:
cn=GroupA,ou=groups,o=ibm,c=UK

AUTHORMD(SEARCHUSR)

FINDGRP('memberof')

LDAP Authorization



AUTHORMD(SEARCHGRP)

FINDGRP('member')

dn: cn=GroupA,ou=groups,o=ibm,c=UK
objectClass: GroupOfNames
cn: GroupA

member:
cn=useradm,ou=users,o=ibm,c=UK
member:
cn=jbloggs,ou=users,o=ibm,c=UK

LDAP Authorization

- Creating authority records works in a similar fashion
  - The tools used for OS authorization are used for LDAP authorization
- The only difference is that:
  - Principal names will be in LDAP form
  - Group names will be in LDAP form

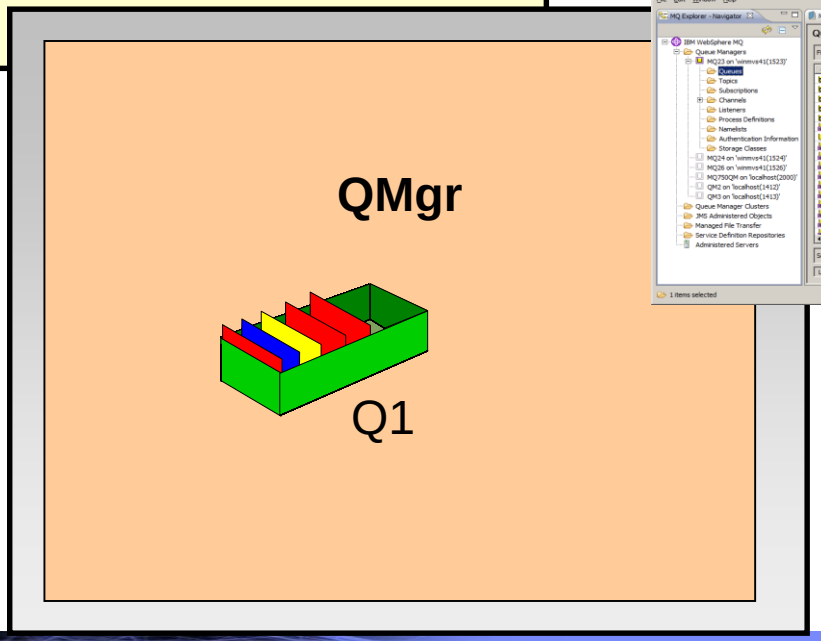
- As in Connection authentication if you have set up BASEDNU/G then you can shorten Principal/Group names

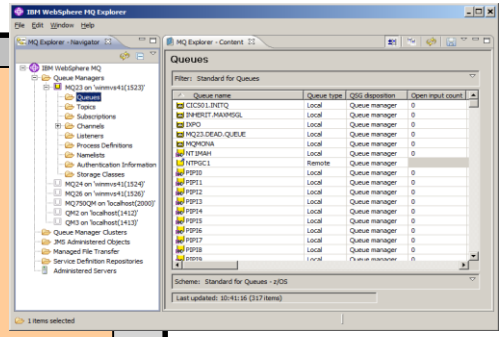
```
setmqaut –m QM –t queue –n Q1 –p cn=useradm,ou=users,o=ibm,c=UK +put
setmqaut –m QM –t queue –n Q1 –p cn=useradm +put
setmqaut –m QM –t queue –n Q1 –p useradm +put
```

Migration / Defaults

```
AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPWOS)
        AUTHTYPE(IDPWOS)
        CHCKLOCL(OPTIONAL)
        CHCKCLNT(REQDADM)
        FAILDLAY(1)
        DESCR( )
        ALTDATE(2013-12-25)
        ALTTIME(12.00.00)
```

- Defaults
  - Migrated queue manager
    - CONNAUTH(‘ ’)
  - New queue manager
    - CONNAUTH( ) ⬅





Summary

- Connection Authentication
  - Application provides User ID and password in MQCSP
    - Or uses mqccred exit supplied
  - Queue Manager checks password against OS or LDAP
    - `ALTER QMGR CONNAUTH('CHECK.PWD')`
    - `DEFINE AUTHINFO('CHECK.PWD')`
      `AUTHTYPE(IDPWOS|IDPWLDAP)`
      `CHCKLOCL(NONE|OPTIONAL|REQUIRED|REQDADM)`
      `CHCKCLNT(NONE|OPTIONAL|REQUIRED|REQDADM)`
      `ADOPTCTX(YES)`
      + various LDAP attributes
    - `REFRESH SECURITY TYPE(CONNAUTH)`
  - Password protection is provided when SSL/TLS not in use
    - Both ends of client channel are V8 or above

Click to add text

Questions?