Deep Dive - IBM MQ
Latest Security Features

Lyn Elkins – *elkinsc@us.ibm.com*
Mitch Johnson – *mitchj@us.ibm.com*

Agenda

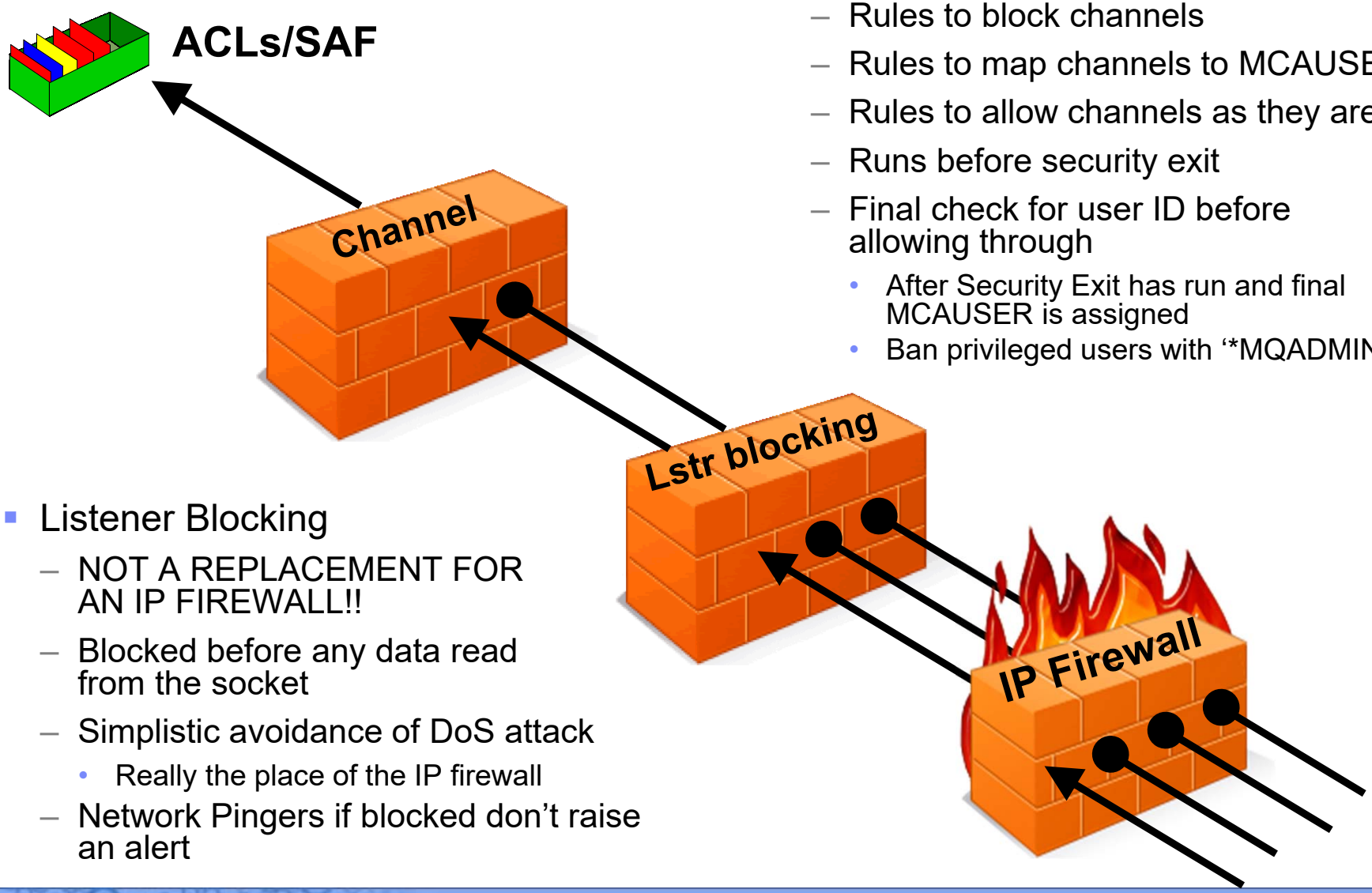
- Channel Authentication with Hostnames
- MQ Enhancements for Digital Certificates
- Connection Authentication Enhancements

Channel Authentication – V7.1 feature review

- Set rules to control how inbound connections are treated
  - Inbound Clients
  - Inbound QMgr to QMgr channels
  - Other rogue connections causing FDCs
- Rules can be set to
  - Allow a connection
  - Allow a connection and assign an MCAUSER
  - Block a connection
  - Ban privileged access
  - Provide multiple positive or negative SSL Peer Name matching
- Rules can use any of the following identifying characteristics of the inbound connection
  - IP Address
  - SSL/TLS Subject's Distinguished Name
  - Client asserted user ID
  - Remote queue manager name
  - And with MQv8….. hostname

Channel Access Blocking Points

- Channel Blocking/Mapping
  - Rules to block channels
  - Rules to map channels to MCAUSER
  - Rules to allow channels as they are
  - Runs before security exit
  - Final check for user ID before allowing through
    - After Security Exit has run and final MCAUSER is assigned
    - Ban privileged users with '*MQADMIN'



- Listener Blocking
  - NOT A REPLACEMENT FOR AN IP FIREWALL!!
  - Blocked before any data read from the socket
  - Simplistic avoidance of DoS attack
    - Really the place of the IP firewall
  - Network Pingers if blocked don't raise an alert

Channel Authentication Rules using IP Addresses

- Initial Listener blocking list
  - Should be used sparingly
  - List of IP addresses/range/pattern
  - Not replacing IP firewall

```
SET CHLAUTH('*') TYPE(BLOCKADDR)
ADDRLIST('9.20.*', '192.168.2.10')
```

- Channel based *blocking* of IP addresses
  - Single IP address/range/pattern

```
SET CHLAUTH('APPL1.*')
TYPE(ADDRESSMAP)
ADDRESS('9.20.*') USERSRC(NOACCESS)
```

- Channel *allowed in*, based on IP addresses
  - Single IP address/range/pattern

```
SET CHLAUTH('*.SVRCONN')
TYPE(ADDRESSMAP)
ADDRESS('9.20-21.*') MCAUSER(HUSER)
```

- Further qualified rule including IP address on another rule type
  - Works with SSLPEER, QMNAME and CLNTUSER

```
SET CHLAUTH('*') TYPE(SSLPEERMAP)
SSLPEER('CN="Mitch Johnson"')
ADDRESS('9.20.*') MCAUSER(MITCHJ)
```

Channel Authentication Rules using Hostnames (new in V8)

- Initial Listener blocking list
  - Hostnames not allowed

~~SET CHLAUTH('*') TYPE(BLOCKADDR) ADDRLIST( )~~

- Channel based blocking of Hostnames
  - Single IP address/range/pattern or hostname/pattern

```
SET CHLAUTH('APPL1.*')
TYPE(ADDRESSMAP)
ADDRESS('*.isis.org')
USERSRC(NOACCESS)
```

- Channel allowed in, based on Hostnames
  - Single IP address/range/pattern or hostname/pattern

```
SET CHLAUTH('*.SVRCONN')
TYPE(ADDRESSMAP)
ADDRESS('mach123.ibm.com') MCAUSER(HUSER)
```

- Further qualified rule including hostname on another rule type
  - Works with SSLPEER, QMNAME and CLNTUSER

```
SET CHLAUTH('*') TYPE(SSLPEERMAP)
SSLPEER('CN="Joe User"')
ADDRESS('s*.ibm.*') MCAUSER(JUSER)
```

Precedence Order

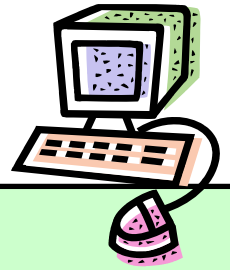
```
DISPLAY CHLAUTH(APPL1.*)
returns ===>
  CHLAUTH(APPL1.*)
  TYPE(SSLPEERMAP)
  SSLPEER('O="IBM UK"') MCAUSER(UKUSER)

  CHLAUTH(APPL1.*)
  TYPE(USERMAP)
  CLNTUSER('mhughson') MCAUSER(HUGHSON)

  CHLAUTH(APPL1.*)
  TYPE(ADDRESSMAP)
  ADDRESS('9.180.165.163') MCAUSER(MORAG)

  CHLAUTH(APPL1.*)
  TYPE(ADDRESSMAP)
  ADDRESS('*.ibm.com') MCAUSER(IBMUSER)
```

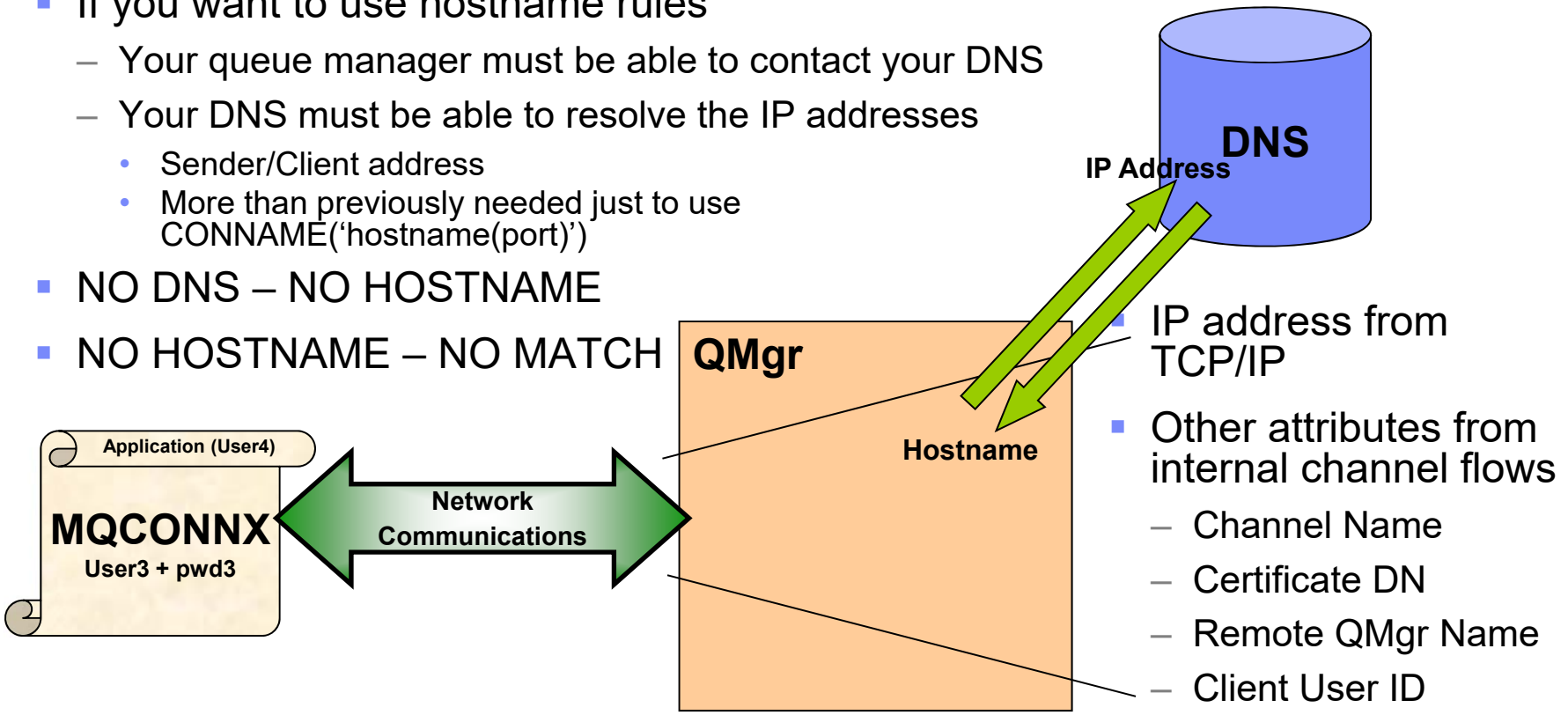
Order	Identity mechanism	Notes
0	Channel Name	
1	SSL Distinguished Name	
2=	Client asserted User ID	Clearly several different user IDs can be running on the same IP address.
2=	Queue Manager Name	Clearly several different queue managers can be running on the same IP address
4	IP address	
5	Hostname	One IP address can have multiple hostnames



Chl: APPL1.SVRCONN
DN: CN=M Hughson.O=IBM UK
UID: mhughson
IP: 9.180.165.163

Obtaining a hostname

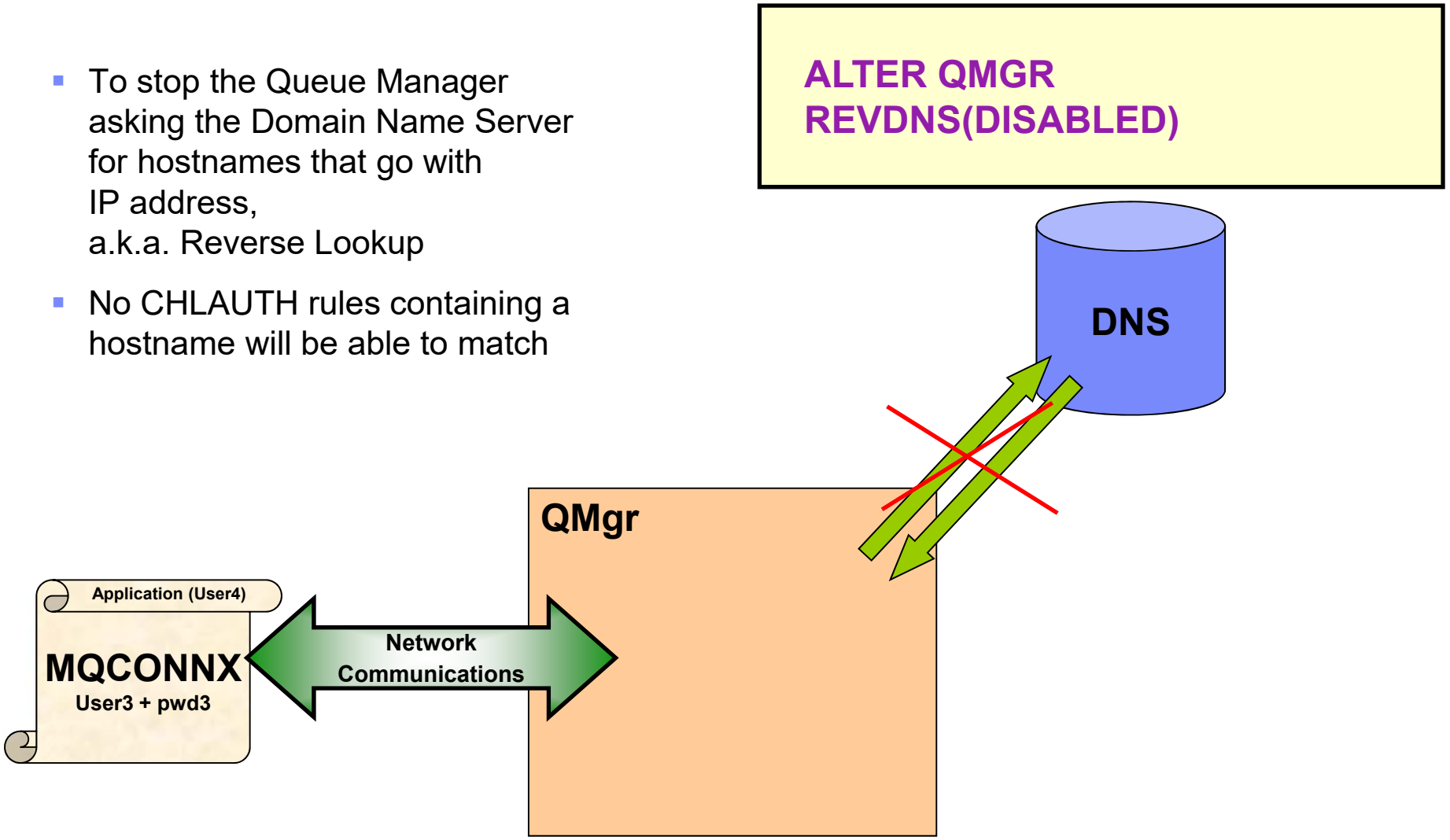
- Hostname is not 'sent' from the other end of the channel
- IP address is obtained from TCP/IP socket
- We must ask the Domain Name Server what the hostname is, a.k.a. Reverse Lookup
- If you want to use hostname rules
  - Your queue manager must be able to contact your DNS
  - Your DNS must be able to resolve the IP addresses
    - Sender/Client address
    - More than previously needed just to use CONNAME('hostname(port)')
- NO DNS – NO HOSTNAME
- NO HOSTNAME – NO MATCH



- IP address from TCP/IP
- Other attributes from internal channel flows
  - Channel Name
  - Certificate DN
  - Remote QMgr Name
  - Client User ID

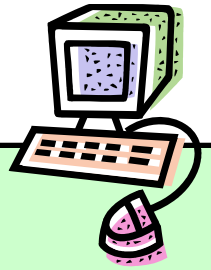
Avoiding obtaining a hostname

- To stop the Queue Manager asking the Domain Name Server for hostnames that go with IP address, a.k.a. Reverse Lookup
- No CHLAUTH rules containing a hostname will be able to match



Diagnosing hostname look-up failures

- WebSphere MQ V7.1



AMQ9777: Channel was blocked
EXPLANATION:
The inbound channel 'SYSTEM.DEF.SVRCONN' was blocked from address '9.180.165.163' because the active values of the channel matched a record configured with USERSRC(NOACCESS). The active values of the channel were 'CLNTUSER(hughson)'.

- WebSphere MQ V8

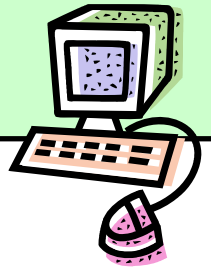
AMQ9777: Channel was blocked
EXPLANATION:
The inbound channel 'SYSTEM.DEF.SVRCONN' was blocked from address 'mhughson.ibm.com(9.180.165.163)' because the active values of the channel matched a record configured with USERSRC(NOACCESS). The active values of the channel were 'CLNTUSER(hughson) ADDRESS(mhughson.ibm.com, morag.hursley.ibm.com)'.

Using MATCH(RUNCHECK) with hostnames

```
DISPLAY   CHLAUTH(SYSTEM.ADMIN.SVRCONN) MATCH(RUNCHECK)
          SSLPEER('CN="Morag Hughson", O="IBM UK"')
          CLNTUSER('mhughson') ADDRESS('9.180.165.163')
returns ===>
                CHLAUTH(SYSTEM.ADMIN.SVRCONN)
                TYPE(ADDRESSMAP)
                ADDRESS('*.ibm.com') MCAUSER(HUGHSON)
```

- Just as before, MATCH(RUNCHECK) mandates an IP address is provided
- Then the queue manager will employ DNS to find the hostname
- MATCH(RUNCHECK) thus also tests whether your DNS is correctly set up.

Chl: SYSTEM.ADMIN.SVRCONN
DN: CN=Morag Hughson.O=IBM UK
UID: mhughson
IP: 9.180.165.163

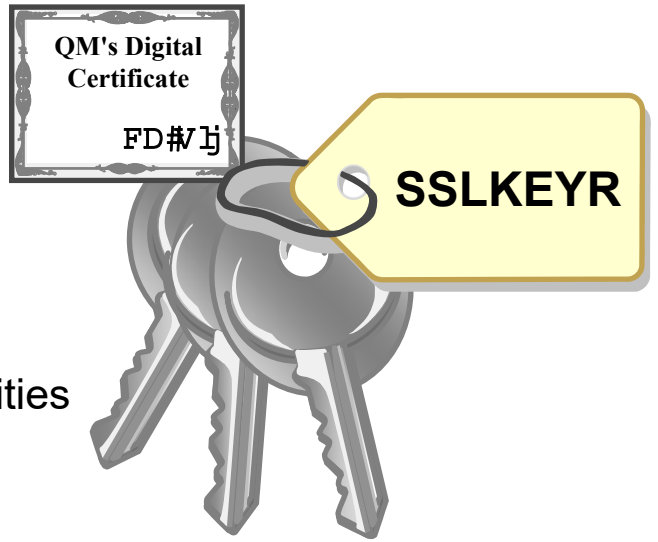


Agenda

- Channel Authentication with Hostnames
- MQ Enhancements for Digital Certificates
- Connection Authentication Enhancements

Key Repository (nothing new here!)

- 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 Certification Authorities
- 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)
  - SSLKeyRepository
- Environment variable
  - export MQSSLKEYR=/var/mqm/ssl/key



ALTER QMGR SSLKEYR(CSQ1RING) **(z/OS)**

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

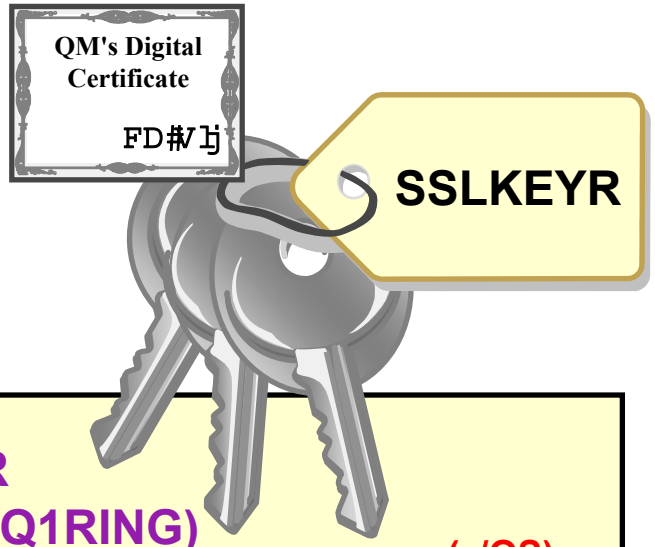
mqclient.ini **(MQ Client)**
SSL:
SSLKeyRepository=C:\key

Single Queue Manager Certificate

- Name Queue Manager Certificate
  - Using CERTLABL attribute
- Name Client Certificate
  - mqclient.ini file SSL Stanza
    - CertificateLabel
  - MQCONNX (MQSCO structure)
    - CertificateLabel
- Environment variable
  - export MQCERTLABL=MyCert

```
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') (z/OS)

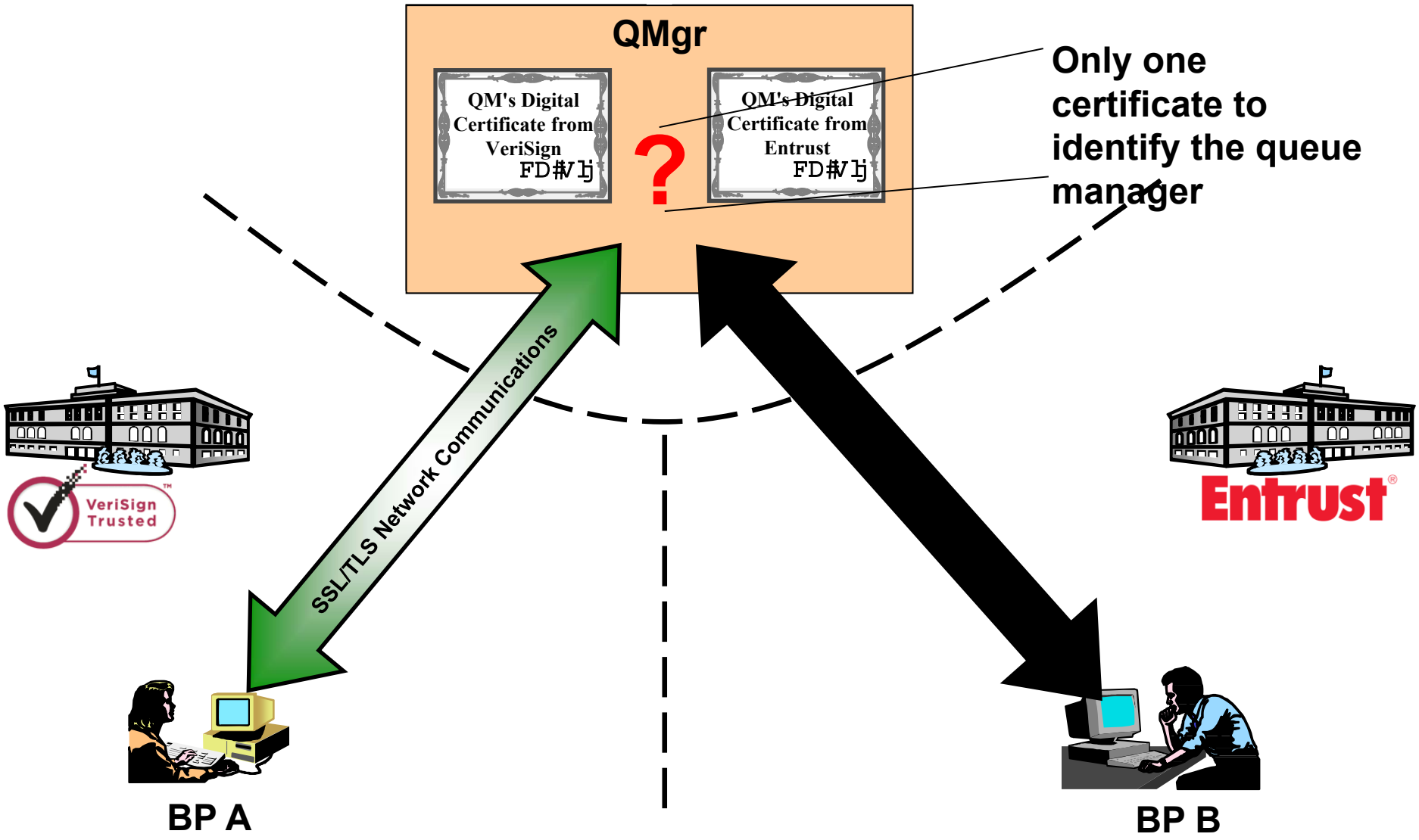
ALTER QMGR SSLKEYR('var/mqm/qmgrs/QM1/ssl/key') CERTLABL('QM1Certificate') (Distrib)

mqclient.ini (MQ Client)
SSL:
SSLKeyRepository=C:\key
CertificateLabel=MyCert

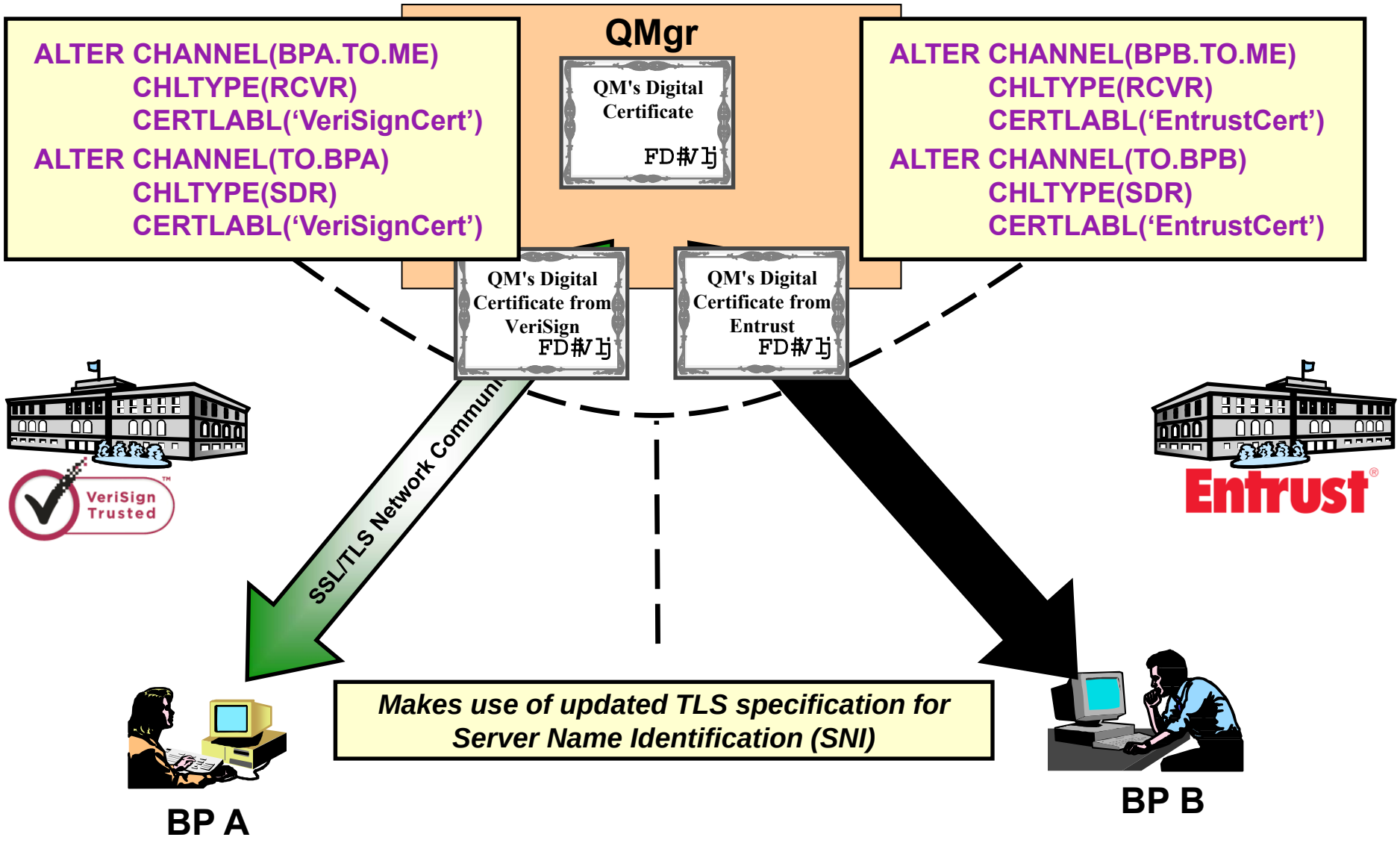
Use Cases

- Following company policy on certificate labelling
- Using the same certificate for more than one queue manager
  - Not that this would condoned!
- 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)

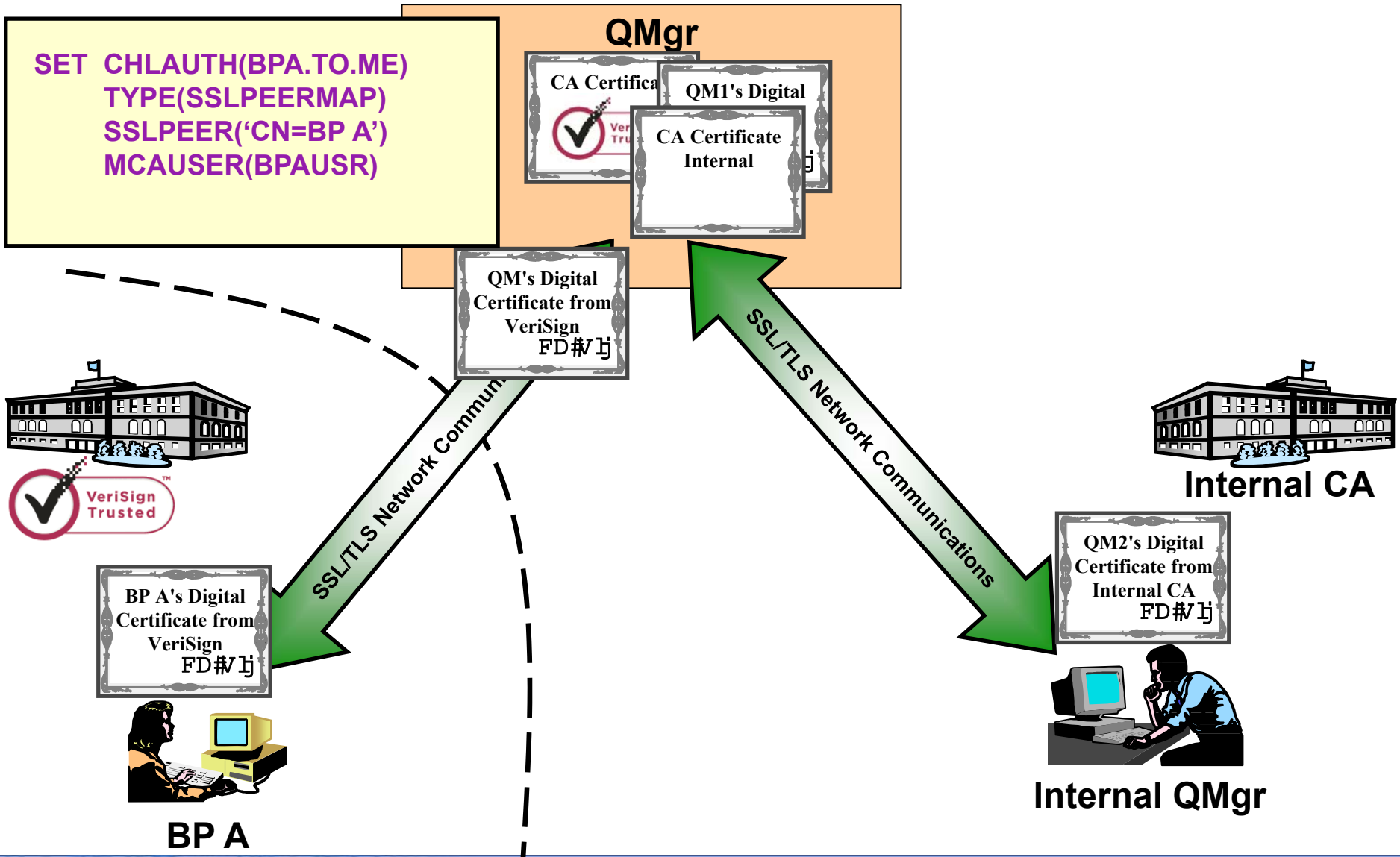
Business Partners with different CA requirements (pre V8)



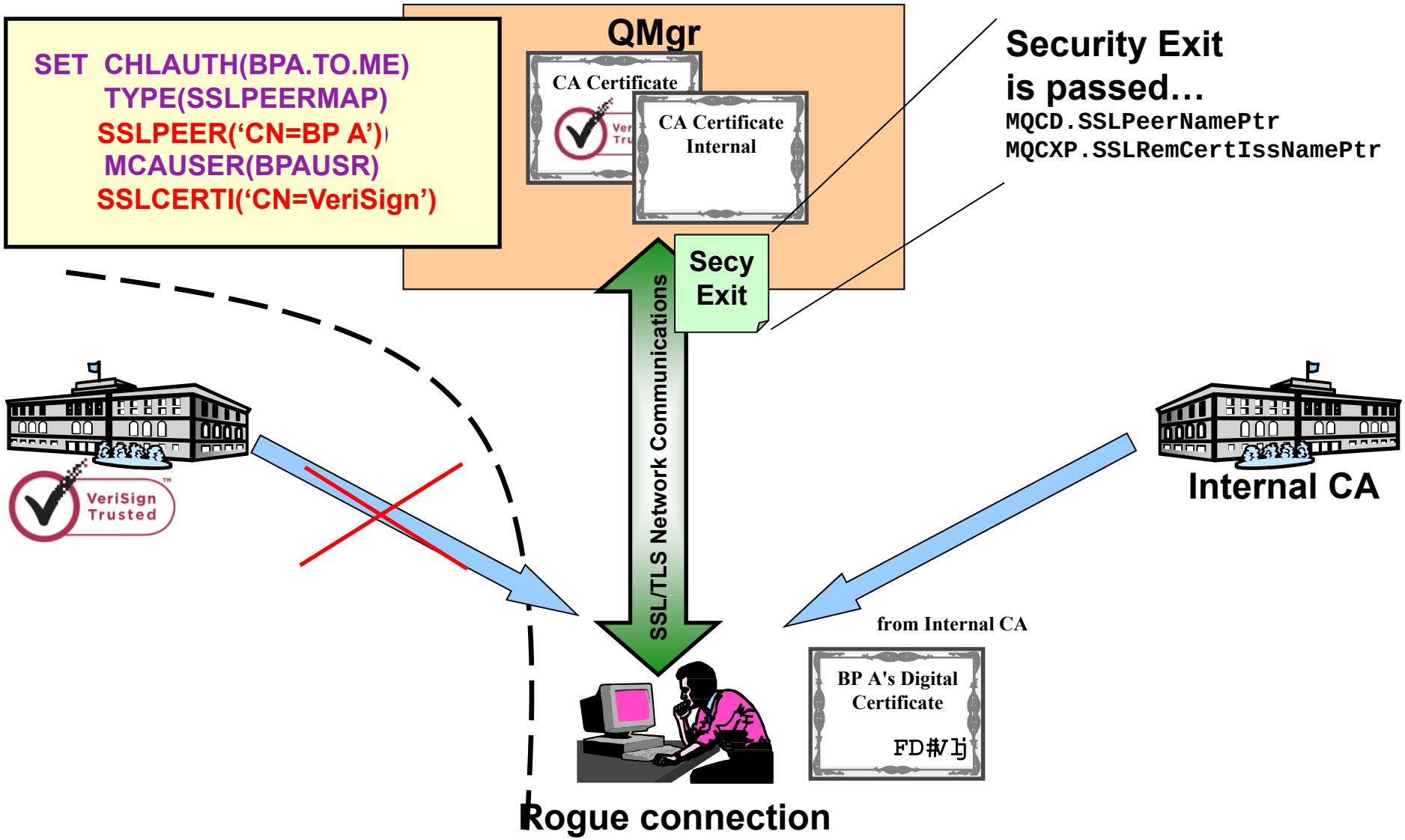
Certificate per Channel (new with V8)



Our Business Partner Scenario again



Ensuring the Correct Certificate



Summary

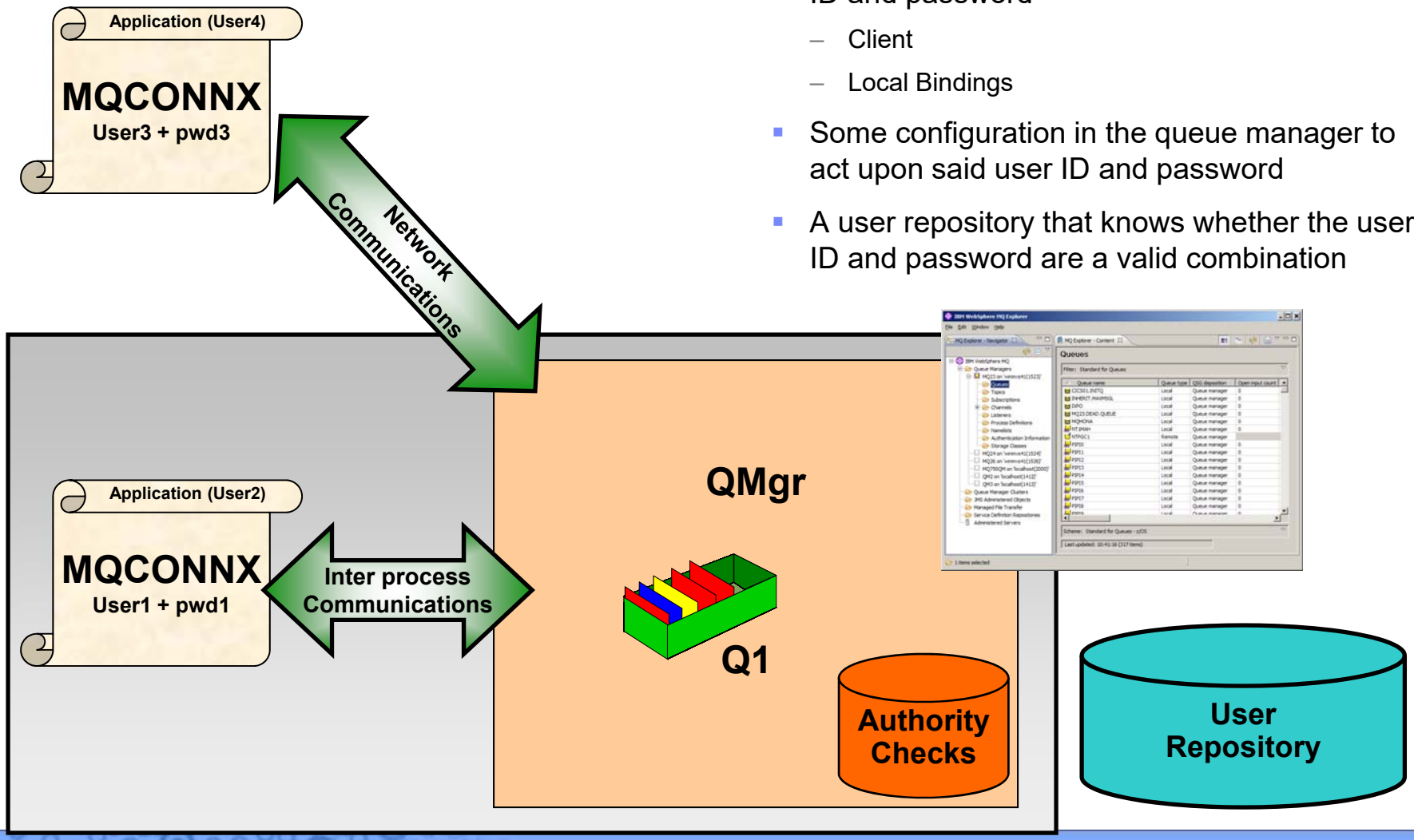
- Changes for Channels using SSL/TLS Certificates
  - Single Queue Manager Certificate
    - `ALTER QMGR CERTLABL('My certificate name')`
  - Per Channel Certificate
    - `ALTER CHANNEL … CERTLABL('This channel certificate')`
  - Certificate Matching
    - ```
      SET CHLAUTH('*')
          TYPE(SSLPEERMAP)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

Agenda

- Channel Authentication with Hostnames
- MQ Enhancements for Digital Certificates
- Connection Authentication Enhancements

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)
```

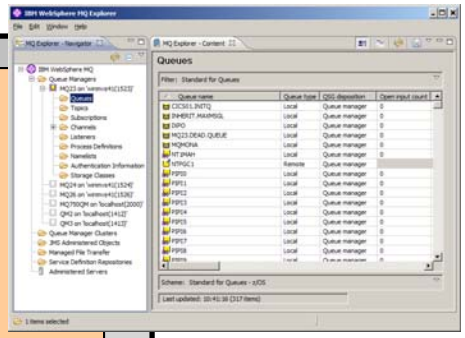
Application (User4)

MQCONNX

~~User3 + pwd3~~

Network Communications

MQRC_NOT_AUTHORIZED (2035)



QMgr

Application (User2)

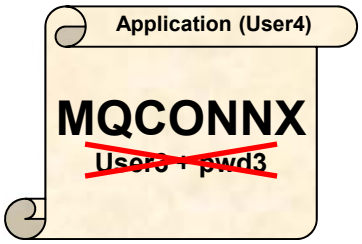
MQCONNX

~~User1 + pwd1~~

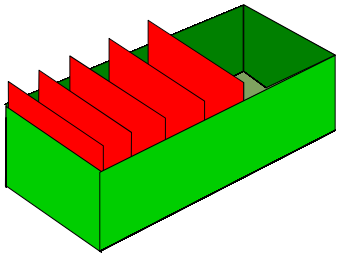
Inter process Communications

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

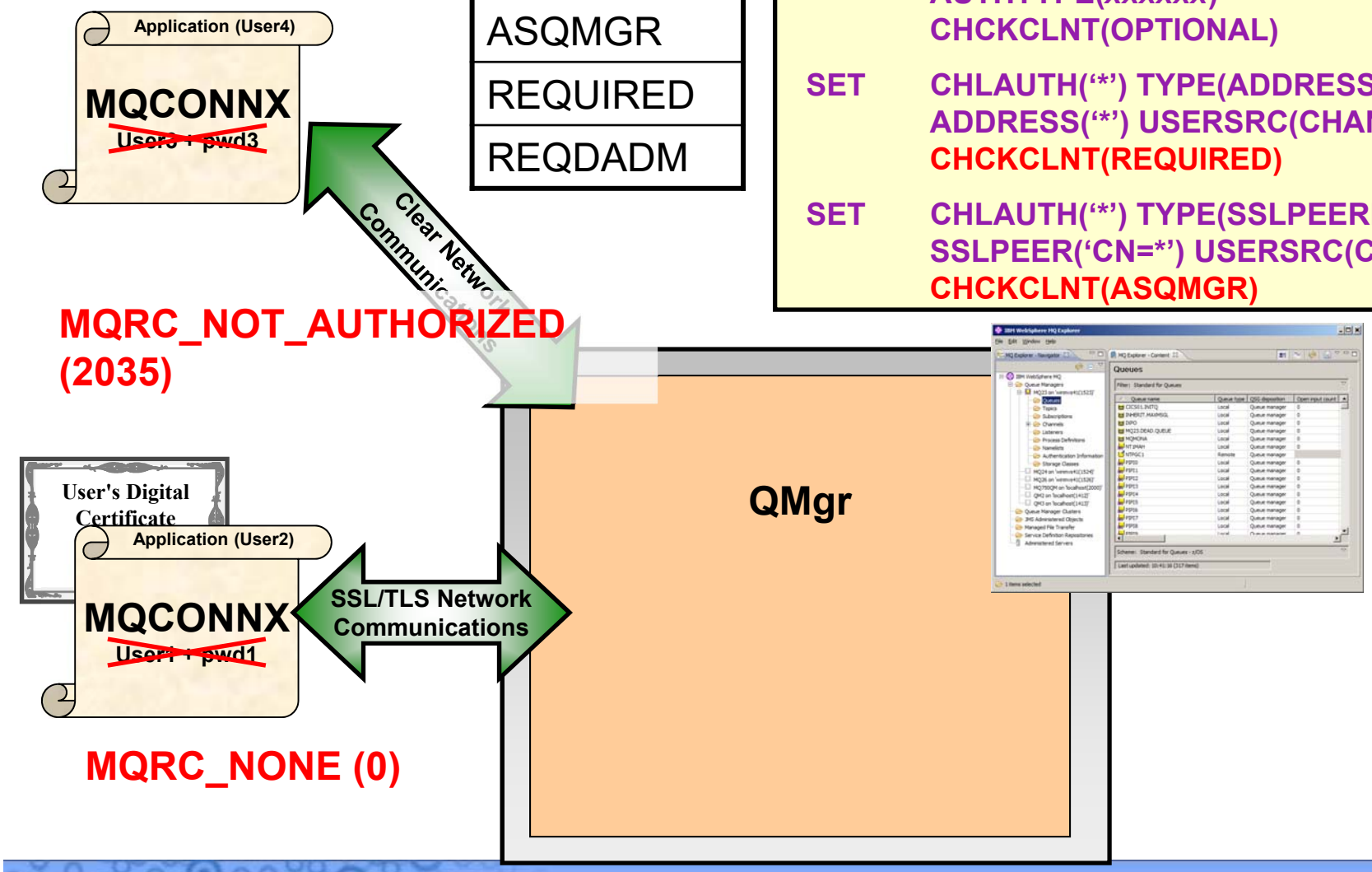
CHCKCLNT
ASQMGR
REQUIRED
REQDADM

```
ALTER   QMGR CONNAUTH(USE.PW)

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

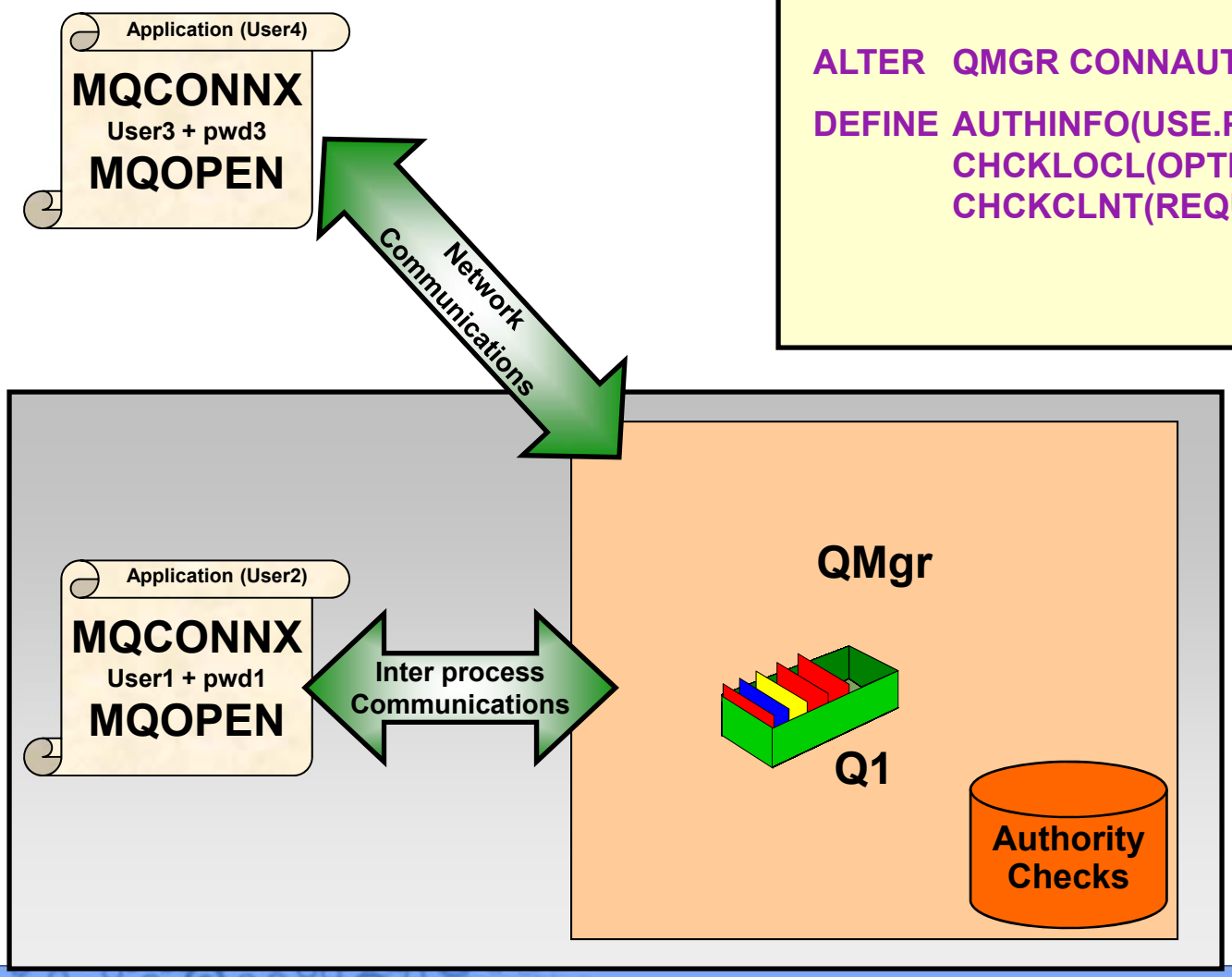
SET     CHLAUTH('*') TYPE(ADDRESSMAP)
        ADDRESS('*') USERSRC(CHANNEL)
        CHCKCLNT(REQUIRED)

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



Connection Authentication – Relationship to Authorization

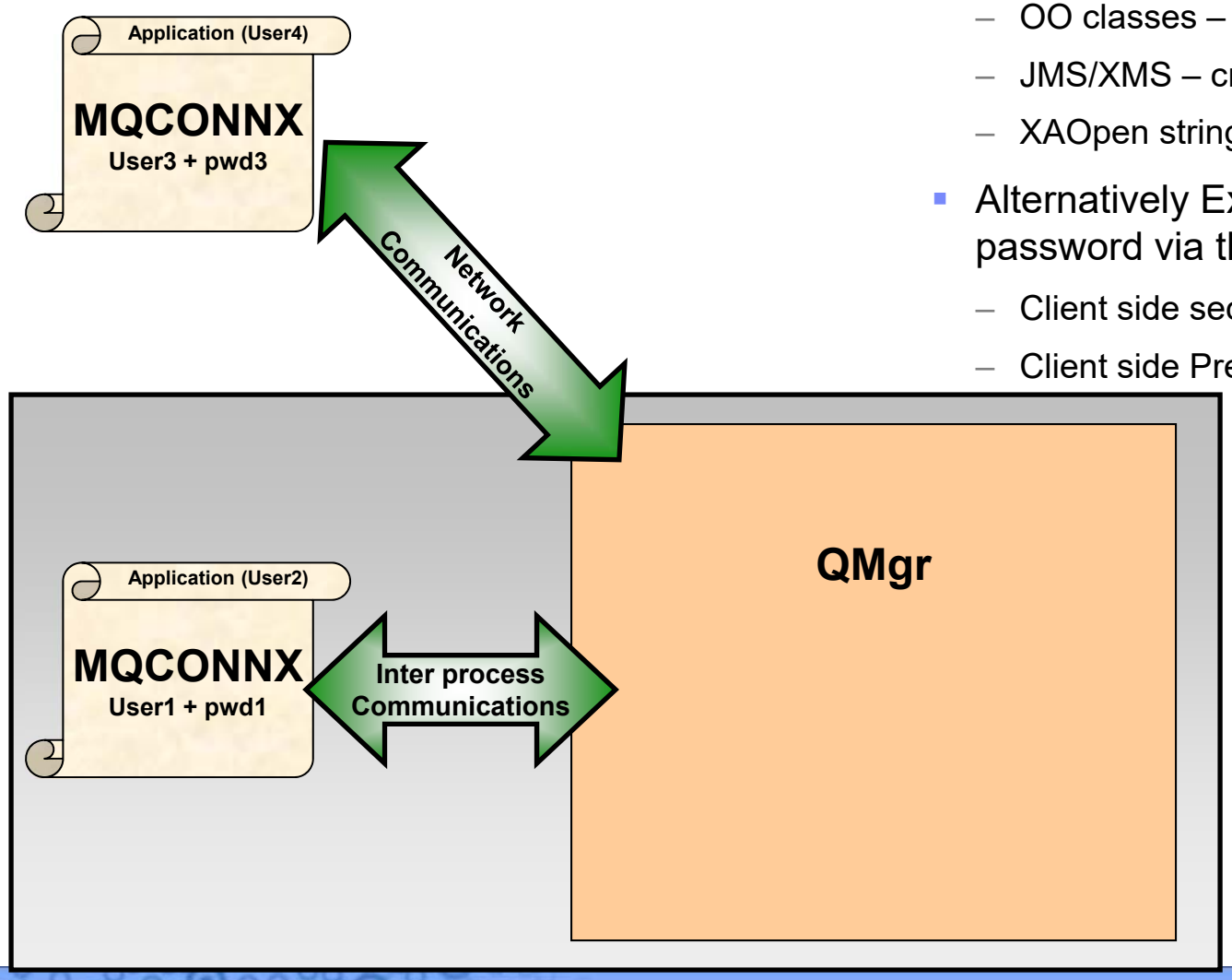
```
ALTER   QMGR CONNAUTH(USE.PWD)
DEFINE  AUTHINFO(USE.PWD) AUTHTYPE(xxxxxx)
        CHCKLOCL(OPTIONAL)
        CHCKCLNT(REQUIRED) ADOPTCTX(YES)
```



User's Access to SAF MQQUEUE resource Q1	
User1	READ
User2	NONE
User3	UPDATE
User4	NONE

Connection Authentication – Application changes

- Code changes
  - Procedural – MQCSP on MQCONNX
  - OO classes – MQEnvironment
  - JMS/XMS – createConnection
  - XAOpen string
- Alternatively Exits can provide userid and password via the 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       = “johnson";
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

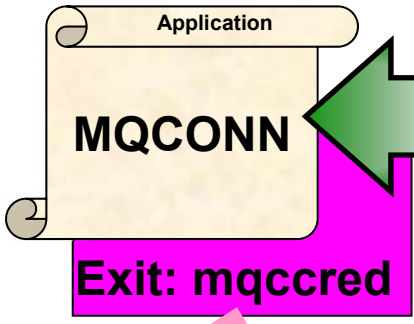
```
cf = getCF();

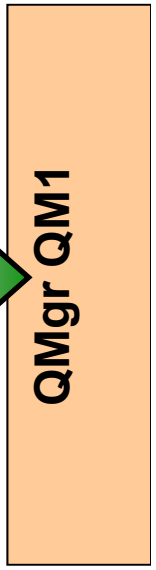
System.out.println("Creating the Connection with UID and Password");
Connection conn = cf.createConnection("hughson", "passw0rd");
```

Client side Security Exit

mqccred.ini

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





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

Tool: runmqccred

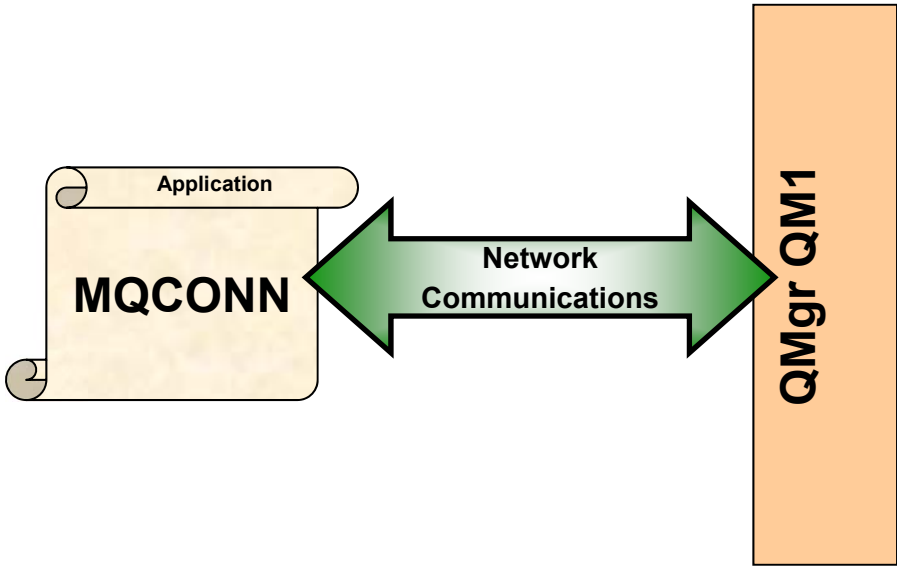
mqccred.ini

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

File permissions

Protecting your password across a network

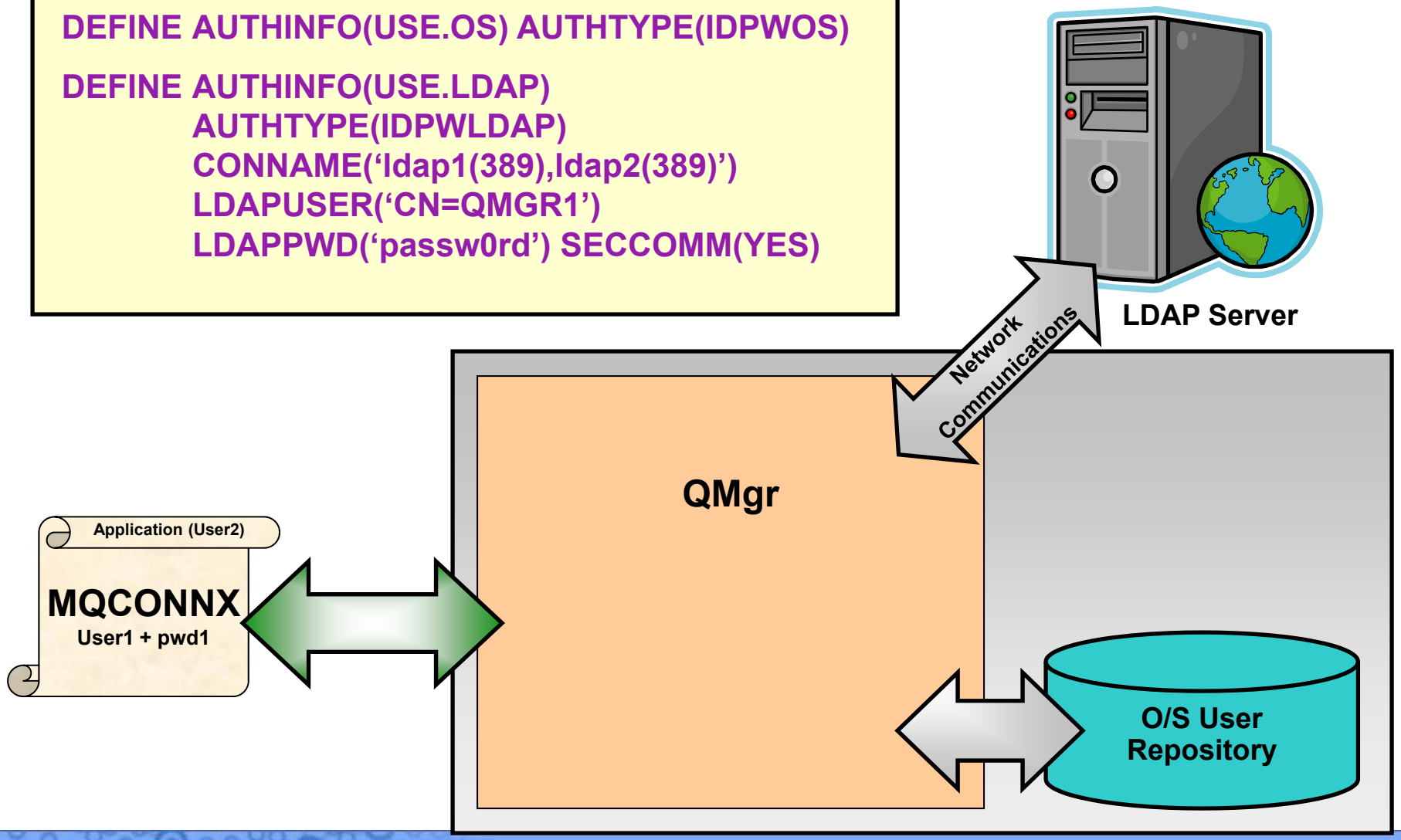
- Use SSL/TLS
  - Perhaps with anonymous clients
- If no SSL/TLS
  - If both ends are V8
  - MQ Code will protect the password – so not sent in the clear
- If client is < V8
  - No MQ password protection
  - Consider SSL/TLS



Connection Authentication – User Repositories

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

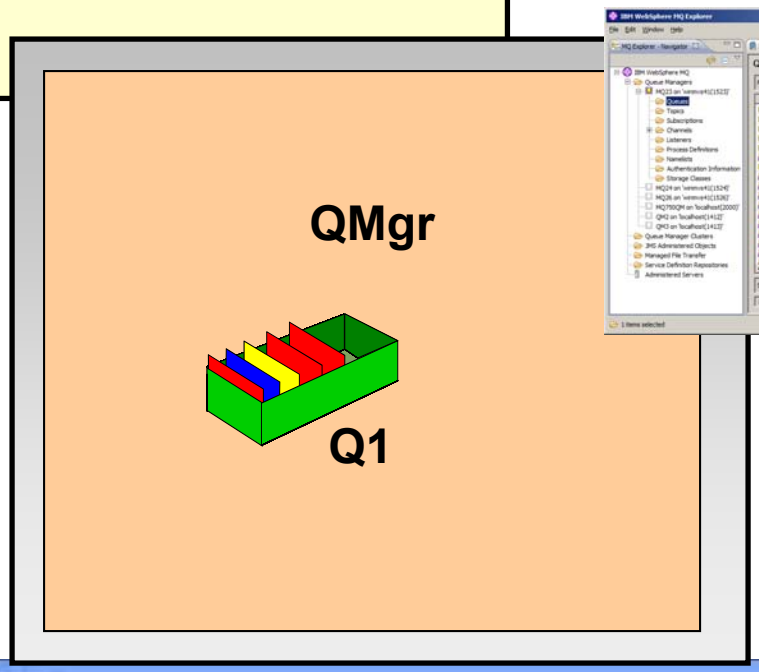
DEFINE AUTHINFO(USE.LDAP)
       AUTHTYPE(IDPWLDAP)
       CONNAME('ldap1(389),ldap2(389)')
       LDAPUSER('CN=QMGR1')
       LDAPPWD('passw0rd') SECCOMM(YES)
```

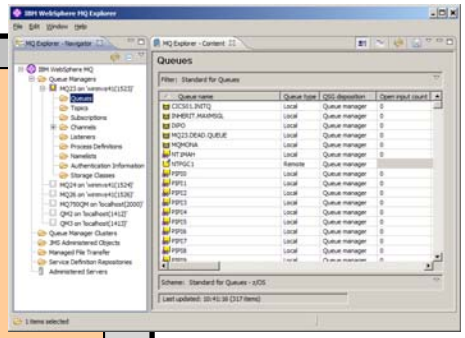


Migration / Defaults

```
AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPW
         OS) 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(SYSTEM.DEFAULT.AUTHINFO.IDPWOS)





Summary

- Channel Authorization (CHAUTH) Rules now support hostnames and domains
  - **SET CHLAUTH('APPL1.*') TYPE(ADDRESSMAP) ADDRESS('*.matahari.com') USERSRC(NOACCESS)**
- Queue Managers now support digital certificates labels at the queue manager and channel levels
- Connection Authentication enhanced to support userid and password checking with the local OS
  - User exits (e.g. CSQ4BCX3) may no longer be required

Questions?