

# End to End Security of My Queue Manager on z/OS

Morag Hughson - [hughson@uk.ibm.com](mailto:hughson@uk.ibm.com)

 @MoragHughson

Session # 17062



SHARE is an independent volunteer-run information technology association that provides education, professional networking and industry influence.

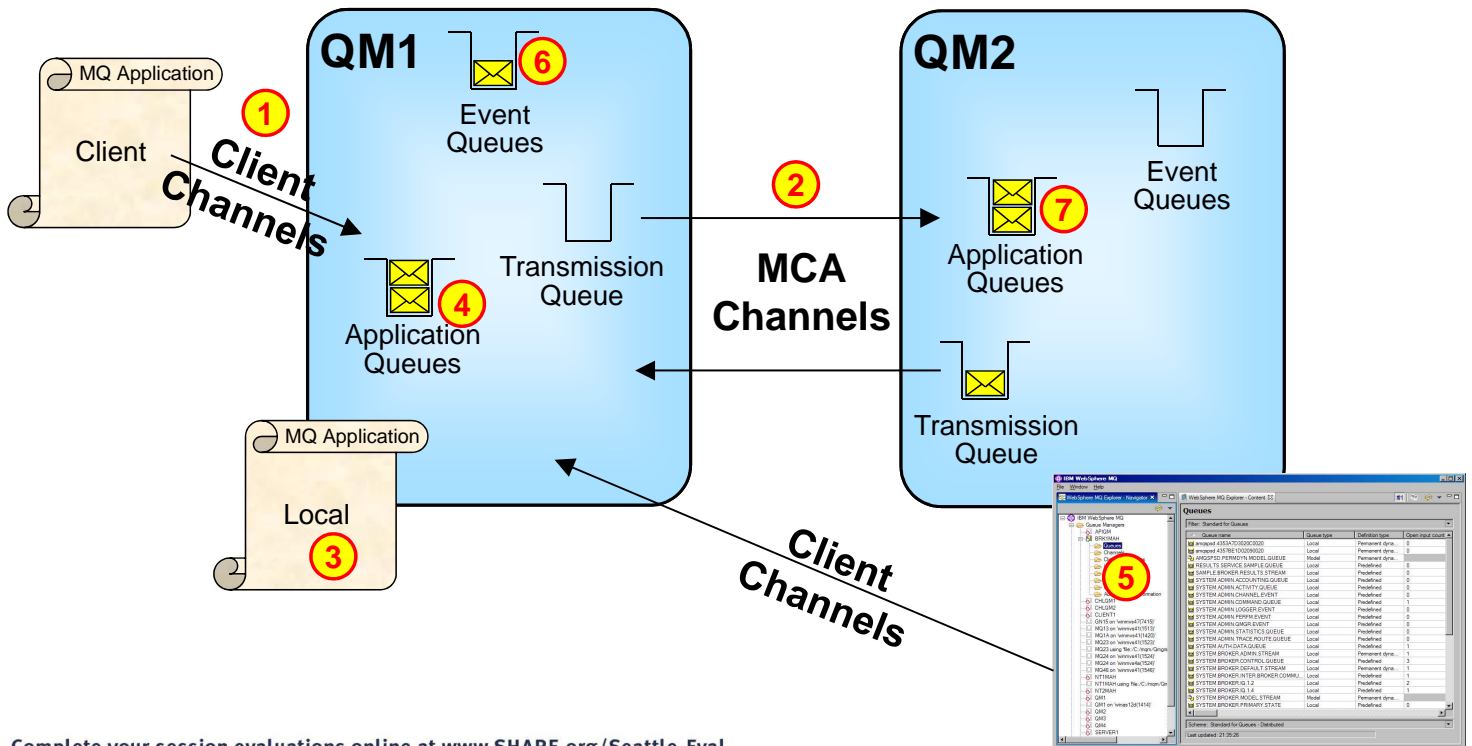


## Abstract

N  
O  
T  
E  
S

- IBM MQ has a wealth of security options, starting from an authentication and audit point for your administration and application connectivity to encryption of data both in transit and at rest. This session will take you through configuring your queue manager to keep your messages secure from the moment it is put, all through the system till it is retrieved and processed.

# Agenda



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

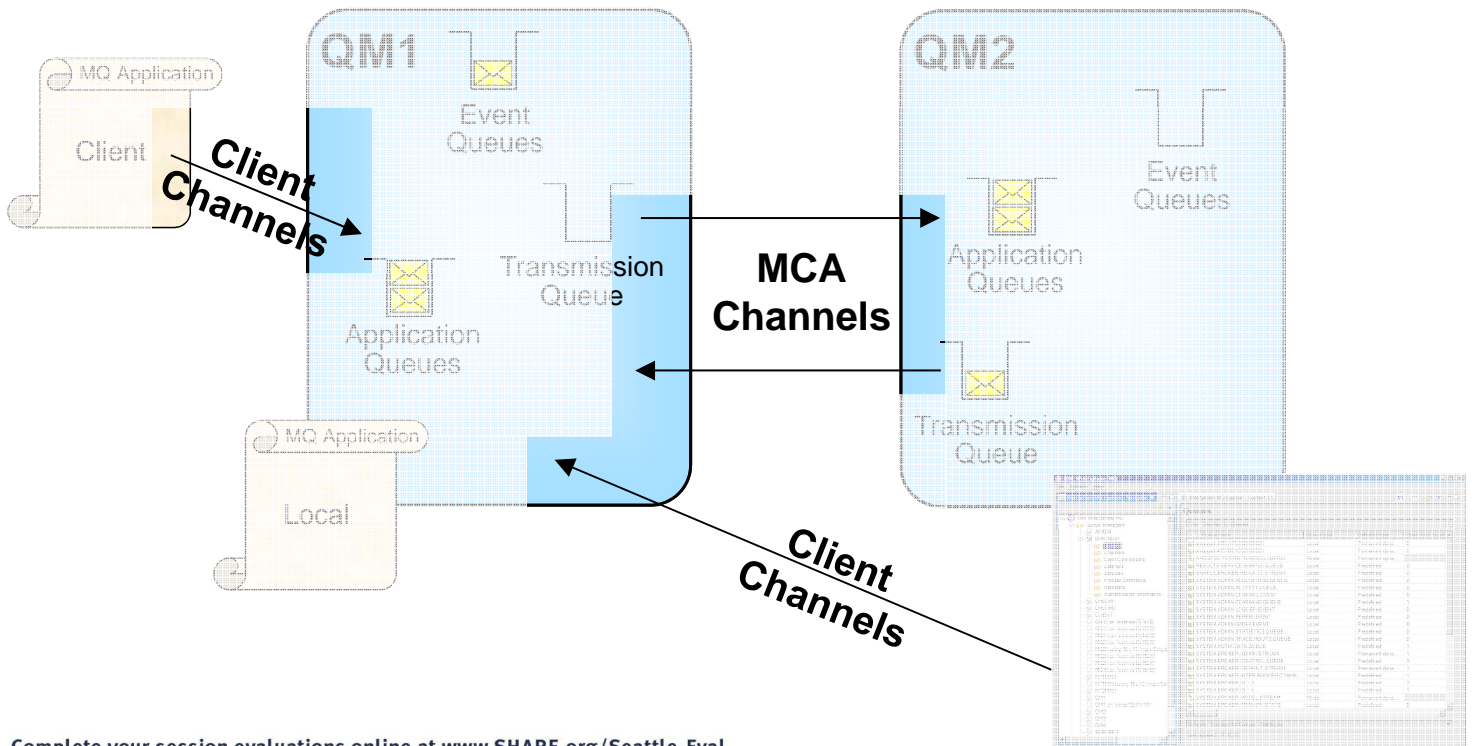
@MoragHughson

# Agenda

N  
O  
T  
E  
S

- This picture is going to act as our agenda today. We're going to look at the various different points in this diagram where security is important and introduce the facilities available for you to use.
- These include:-
  - Authentication
    - Client channels
    - MCA channels
    - Applications (local or client)
  - Authorization
    - Application tasks
    - Administration tasks
  - Auditing
  - Message Level Protection
    - Encryption
    - Tamper Proof

# Authentication - Channels



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Authentication – Channels – Notes

N

- First we shall look at authentication of remote partners, whether clients or remote queue managers, which connect into your system.
- It is very important to ensure that you have good authentication mechanisms in place for any remote partners. You must know that the connections coming into your queue manager can be trusted.

O

## Identification

- When an MQ application connects remotely to a queue manager it can assert an identity across the network connection to the queue manager. This identity could be anything and so should not be trusted without some form of queue manager side authentication.

T

## Authentication

- Authentication is the way in which a channel ensures that the other end of the channel is who they say they are. Channels can make use of SSL/TLS to authenticate a digital certificate sent by the partner. In WebSphere MQ V7.1 Channel Authentication Records can be used to do many of the jobs a security exit can do, such as allowing or blocking a channel based on IP address, Certificate DN, Remote Queue Manager Name or Client User ID.
- Once a remote partner has been authenticated, Channel Authentication Records or a security exit can also set the identity that this channel will use for all access control checks.

E

## Confidentiality

- In an ideal environment all channels would be running inside the enterprise with good physical security. However, often there will be cross enterprise channels or channels running on networks where physical security can not be guaranteed. In those cases it is worth considering adding some level of encryption to the data flow. This can either be done in channel exits or by using SSL/TLS on the channels.

S

## Authentication – Channels – Facilities

---

- **Transport Layer Security (SSL/TLS)**
  - Using Digital Certificates
- **Channel Authentication Records**
  - New in WebSphere MQ V7.1
  - Updated in IBM MQ V8
- **Security Exits**
  - Many Vendor exits available

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



## Authentication – Channels – Facilities – Notes

---

N  
O  
T  
E  
S

- Over the next few pages we are going to introduce each of the following facilities which allow you to provide some authentication for your client or MCA channels. The strength of the authentication provided varies by each facility so the choice of facility should take that into account when making a business decision as to the level of authentication required.

# Authentication - Using SSL/TLS with WebSphere MQ

- **Get your certificates for Authentication**

- Digital Certificates
- Asymmetric Keys



- **Put your certificates in a place that MQ can use**

- Label them how you wish



- **Decide if you need revocation status checking (LDAP or OCSP)**



- **Decide if you need cipher spec restriction (FIPS or SUITEB)**

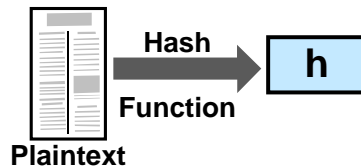
- **Configure your channels to use SSL/TLS for Confidentiality**

- Symmetric Key Cryptography



- **... and Data Integrity**

- Hash Function



@MoragHughson

- **WebSphere MQ SSL Wizard (MO04)**

SSLKEYR(QM1KEYRING)  
 CERTLABL('QM1Cert')

SSLCRLNL(REVOKE.NI)

SSLFIPS(NO)  
 SUITEB(NONE)

SSLCIPH(RC4\_MD5\_US)  
 SSLRKEYC(999 999 999)  
 SSLCAUTH(REQUIRED)  
 SSLPEER('O=IBM')  
 SSLCERTI('CN=MQ CA')



# Authentication - Using SSL/TLS with WebSphere MQ – Notes

N  
O  
T  
E  
S

- The three main issues that Transport Level Security (SSL/TLS) addresses are Confidentiality, Data Integrity and Authentication. The techniques that it uses to address these issues are
  - For Confidentiality, we have symmetric key cryptography with the capability to periodically reset the secret key;
  - For Data Integrity we have the hash function; and
  - For Authentication we have digital certificates, asymmetric keys and certificate revocation lists.
- WebSphere MQ makes use of these techniques to address these security issues. One can specify which symmetric key cryptography algorithm and which hash function to use by providing WebSphere MQ with a SSLCipherSpec (SSLCIPH on a channel). The secret key can be periodically reset by setting an appropriate number of bytes in SSLKeyResetCount (SSLRKEYC on the queue manager).
- Digital Certificates and Public Keys are found in a key repository which can be specified to WebSphere MQ (SSLKEYR on the queue manager). We can also check that we are talking to the partner we expect to be talking to (SSLPEER on a channel) and can choose to authenticate both ends of the connection or only the SSL Server end of the connection (SSLCAUTH on a channel). Also we can make choose to do certificate revocation status checking using either LDAP CRLs or OCSP (SSLCRLNL on the queue manager).
- The set of cipher specs to be used by the queue manager can be restricted to a set that are compliant to the FIPS 140-2 standard (SSLFIPS on the queue manager) available on both distributed and in WebSphere MQ V7.1 on z/OS; or to the Suite-B standard (SUITEB on the queue manager) available on the distributed platforms in WebSphere MQ V7.1

# Channel Authentication Records

- **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/TLS Distinguished Name matching
  - **Mandate user ID & password checking**
- **Rules can use any of the following identifying characteristics of the inbound connection**
  - IP Address
  - **Hostnames**
  - SSL/TLS Subject's Distinguished Name
  - **SSL/TLS Issuer's Distinguished Name**
  - Client asserted user ID
  - Remote queue manager name



 @MoragHughson

## Channel Authentication Records – Notes

N  
O  
T  
E  
S

- Channel Authentication records allow you to define rules about how inbound connections into the queue manager should be treated. Inbound connections might be client channels or queue manager to queue manager channels. These rules can specify whether connections are allowed or blocked. If the connection in question is allowed, the rules can provide a user ID that the channel should run with or indicate that the user ID provided by the channel (flowed from the client or defined on the channel definition) is to be used.
- These rules can therefore be used to
  - Set up appropriate identities for channels to use when they run against the queue manager
  - Block unwanted connections
  - Ban privileged users
- Which users are considered privileged users is slightly different depending on which platform you are running your queue manager on. There is a special value ‘\*MQADMIN’ which has been defined to mean “any user that would be privileged on this platform”. This special value can be used in the rules that check against the final user ID to be used by the channel – TYPE(USERLIST) rules – to ban any connection that is about to run as a privileged user. This catches any blank user IDs flowed from clients for example.



# CHLAUTH – Configuration

- **Create rules using**
  - MQSC: SET CHLAUTH
  - PCF
- **Pattern matching**
  - Channel Name/QMgr Name/Hostname
    - Beginning, middle, end
  - IP addresses (IPV4 or IPV6)
  - SSL Peer Name (as today)

```

Command Prompt - runmqsc TEST1

Starting MQSC for queue manager TEST1.

SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS)

SET CHLAUTH('*') TYPE(BLOCKUSER) USERLIST('*MQADMIN')

SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('9.20.1-3.*') USERSRC(CHANNEL)

SET CHLAUTH('APP1.CHL*') TYPE(ADDRESSMAP) ADDRESS('*.ibm.com') USERSRC(CHANNEL)

SET CHLAUTH('*.ADMIN.*') TYPE(SSLPEERMAP) SSLPEER('O=IBM,L=Hursley') USERSRC(CHANNEL)

SET CHLAUTH('QM1.TO.QM2') TYPE(QMGRMAP) QMNAME(QM1) USERSRC(MAP) MCAUSER('QM1USER')

SET CHLAUTH('*.SVRCONN') TYPE(USERMAP) CLNTUSER('mhughson') MCAUSER('hughson@hursley')

SET CHLAUTH('*') TYPE(SSLPEERMAP) SSLPEER('O=IBM') ADDRESS('9.*') MCAUSER('hughson')
  
```

# CHLAUTH – Configuration – Notes

N  
O  
T  
E  
S

- Here we show some example rules illustrating the commands used for creating the rules. These examples are in MQSC. There is also PCF, and this is used by the MQ Explorer GUI.
- Some of these examples illustrate the pattern matching that can be applied to channel names, IP addresses, Hostnames, SSL/TLS DNs and remote queue manager names. Also we see all three types of rules, blocking channels – USERSRC(NOACCESS); allowing channels to run with the user ID provided by the channel – USERSRC(CHANNEL); and assigning a user ID to a channel – USERSRC(MAP) MCAUSER(user-id). USERSRC(MAP) is the default so we also see in another example that it does not need to be specified on the command.

# Restricting the Mappings

- **Rules matching on**
  - SSL Peer Name
  - Remote QMgr Name
  - Client User ID
- **Can add IP address/Hostname**
- **Restrict where an SSL Certificate can be used from**
  - Specific IP address/Hostname
- **Restrict where a queue manager or client user ID can come from**
  - Specific IP address/Hostname

| Mapped      | Restrict By |         |             |                     |
|-------------|-------------|---------|-------------|---------------------|
|             | SSL Peer    | QM Name | Client User | IP Address/Hostname |
| SSL Peer    |             | X       | X           | ✓                   |
| QM Name     |             |         |             | ✓                   |
| Client User |             |         |             | ✓                   |
| IP Address  |             |         |             |                     |

```
SET CHLAUTH(*) TYPE(SSLPEERMAP)
SSLPEER('L="Hursley"') MCAUSER(HURUSER) ADDRESS('9.20.*')
```

```
SET CHLAUTH(*) TYPE(QMGRMAP)
QMNAME(CLUSQM*) MCAUSER(CLUSUSR) ADDRESS('*.ibm.com')
```

# Restricting the Mappings - Notes

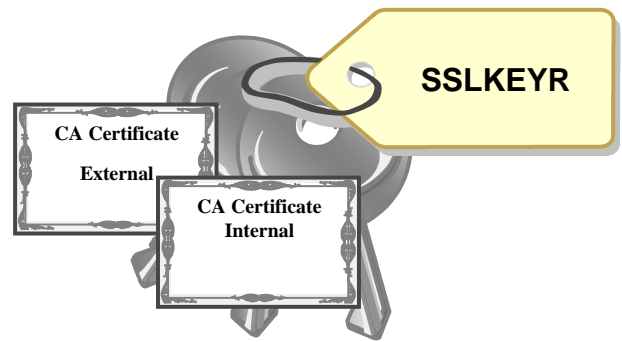
N  
O  
T  
E  
S

- When mapping from an SSL certificate DN, you may also want to ensure that certificate is being used from the correct IP address, mitigating what might happen if a certificate is stolen.
- When mapping from a queue manager name, you may also want to ensure that the queue manager is running on the correct IP address to ensure it is not a rogue queue manager with the same name as one in your cluster for example.
- We could imagine using the remote queue manager name or the client user ID as a restrictor on an SSL Peer rule, however feedback from EAP did not suggest anyone needed it so it was not implemented. For the most part, attributes within the X509 DN will contain the same information for most practical uses. For example CN=<Queue Manager Name>.



# Fully Qualifying your Peer Name rules

- **Key Repository contains**
  - All CA certs we trust
  - Multiple CAs means possible DN clashes
- **External CAs**
  - Checks and balances
  - Unlikely to have DN clashes
- **Internal CAs**
  - Less rigid
  - May give out certs exactly as requested
  - May end up with clashes
- **Could solved in a Security Exit**
  - MQCD.SSLPeerNamePtr
  - MQCXP.SSLRemCertIssNamePtr
- **CHLAUTH rules extended**
  - Check Subject's DN (SSLPEER)
  - Check Issuer's DN (SSLCERTI)



```
SET CHLAUTH(BPA.TO.ME)
TYPE(SSLPEERMAP)
SSLPEER('O=IBM')
MCAUSER(BPAUSR)
SSLCERTI('CN=External')
```



[@MoragHughson](#)

# Fully Qualifying your Peer Name rules – Notes

N  
O  
T  
E  
S

- As we just saw, you can add IP address or hostname restrictors to many of the rule types to further qualify the matching that happens.
- In the case of a Peer name map, you can fully qualify the certificate matching by providing both the Subject's DN (SSLPEER) and the Issuer's DN (SSLCERTI) on a rule. SSLCERTI is new in MQ V8.
- This is especially important if you have more than one Certificate Authority (CA) certificate in your key repository which you may be more likely to do with the introduction of multiple certificates for one queue manager which was a new feature in MQ V8.
- However, since we now accept certificates which come from two different Certificate Authorities (CAs) we can run foul of another issue.
- One of the benefits of External CAs is that they guarantee not to issue the certificates with the same DN as another certificate that they have already issued. However, an internal CA may not be so diligent. Some internal CAs may simply accept what the user requests as their DN, so our rogue could obtain a certificate with non-unique DN from such a CA.
- The only way to solve this issue in the past was to use a security exit, since security exits are presented with both the issuer's and subject's Distinguished Name. However, we are trying to get away from people having to write exits for common security issues, and this very much falls into that category.
- In MQ V8, we can solve this issue by using a new attribute on CHLAUTH rules which matches the issuer's DN – SSLCERTI. Our CHLAUTH rules can now be fully qualified to use both SSLPEER (the subject's DN) and SSLCERTI (the issuer's DN).

# CHLAUTH – Precedence


- **Precedence matching**
  - Most specific rule is matched
- **Identifying attributes are**
  - Channel Name
  - SSL Peer Name pattern
    - Precedence defined for partial patterns
  - Remote queue manager name pattern (MCA channels)
  - Client asserted user ID (MQI channels)
    - No pattern matching on this
  - IP address pattern
  - Hostname pattern (least specific)
- **Within SSL Peer Name matching**
  - Most specific substring is matched

**ChI:** MY.CHANNEL

**IP:** 9.20.1.123

**DN:** CN=Morag Hughson.O=IBM UK

**UID:** mhughson



| Order | Identity mechanism               | Notes  |
|-------|----------------------------------|--|
| 0     | Channel Name                     |  |
| 1     | SSL Subject's Distinguished Name |  |
| 2     | SSL Issuer's Distinguished Name  |  |
| 3=    | Client asserted User ID          | Clearly several different user IDs can be running on the same IP address.      |
| 3=    | Queue Manager Name               | Clearly several different queue managers can be running on the same IP address |
| 5     | IP address                       |  |
| 6     | Hostname                         | One IP address can have multiple hostnames                                     |

# CHLAUTH – Precedence – Notes

N  
O  
T  
E  
S

- When there is more than one rule that could match the inbound connection in question, then we define which rule will actually be used by defining the precedence order of what is the most specific match. The table shows that SSL Peer Names are considered a more specific match than a queue manager name or client user ID (because there is much more detailed information in an SSL Peer Name); and Hostnames are considered the least specific since clearly more than one queue manager or client can be connecting from the same IP address/Hostname. Hostnames are even less specific than IP addresses because an IP address can have multiple hostnames.

# SSL DN Precedence Mapping Example

SET CHLAUTH(\*) TYPE(SSLPEERMAP) SSLPEER('OU="MQ Devt"')  
MCAUSER(MQUSER)

SET CHLAUTH(\*) TYPE(SSLPEERMAP) SSLPEER('L="Hursley"')  
MCAUSER(HURUSER)

| Order | DN Substring | Name                   |
|-------|--------------|------------------------|
| 1     | CN=          | Common name            |
| 2     | T=           | Title                  |
| 3     | OU=          | Organizational unit    |
| 4     | O=           | Organization           |
| 5     | L=           | Locality               |
| 6     | ST=, SP=, S= | State or province name |
| 7     | C=           | Country                |

Most Specific Match

CN=Morag Hughson.OU=MQ Devt.  
O=IBM UK.L=Hursley.C=UK



## SSL DN Precedence Mapping Example – Notes

N  
O  
T  
E  
S

- Not only do we define the order of precedence between the various different identifying characteristics of an inbound connection, we also must do a similar job for SSL Peer Name.
- Here is an example to illustrate what happens when two partial patterns could both match an inbound Distinguished Name (DN) from a client.
- We want the most specific match to be used, so we have defined a precedent order of what we mean by the most specific.
- The table shown here that defines the precedence order is a subset of the contents of an SSL Peer Name in WebSphere MQ V7.1. It suffices to describe this example. For the full table of SSL Peer Name attributes, visit Knowledge Centre here:- [http://www.ibm.com/support/knowledgecenter/SSFKSJ\\_8.0.0/com.ibm.mq.sec.doc/q009860\\_.htm](http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.sec.doc/q009860_.htm)

## Channel Authentication – How should I use this?

```

SET CHLAUTH(*) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS)
SET CHLAUTH(BPCHL.*) TYPE(SSLPEERMAP) SSLPEER('O=Bank of Shetland') MCAUSER(BANK123)
SET CHLAUTH(BPCHL.*) TYPE(SSLPEERMAP) SSLPEER('O=Bank of Orkney') MCAUSER(BANK456)
SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP)
ADDRESS('9.20.1-30.*') MCAUSER(ADMUSER) CHCKCLNT(REQUIRED)
SET CHLAUTH(TO.CLUS.*) TYPE(QMGRMAP)
QMNAME(CLUSQM*) MCAUSER(CLUSUSR) ADDRESS('9.30.*')
    
```



“Our internal cluster doesn’t use SSL, but we must ensure only the correct queue managers can connect into the cluster”

 @MoragHughson

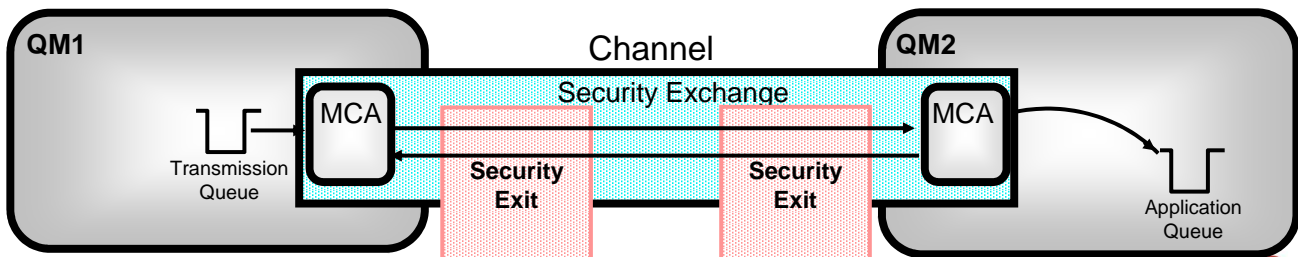
## CHLAUTH – How should I use this? - Notes

N  
O  
T  
E  
S

- Here is an example of how we expect this to be used.
- Our business requires that “We must make sure our system is completely locked down”. So we start off with a rule that blocks everyone. Therefore anyone that doesn’t match a more specific rule will not be allowed in.
- Our business requires that “Our Business Partners must all connect using SSL, so we will map their access from the certificate DNs”. So we have some rules that map specific DNs of our Business Partners to specific user IDs. Previously you might have done this by having separate channel definitions for each BP, now if you wish they can come into the same receiver definition.
- Our business requires that “Our Administrators connect in using MQ Explorer, but don’t use SSL. We will map their access by IP Address”. So we have a rule that gives them all a single administrative access user ID based on a range of IP addresses.
- Our business requires that “Our internal cluster doesn’t use SSL, but we must ensure only the correct queue managers can connect into the cluster”. So we have a rule that gives access to the correctly named queue managers but only if they come from a recognised IP address.

# Channel Authentication – Security Exits

- **Channel 'Gate Keeper'**
  - Indefinite exchange of data between exits
  - No defined format
  - No communications knowledge required
  - Can end channel
  - Can set MCAUSER



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Channel Authentication – Security Exits – Notes

N

O

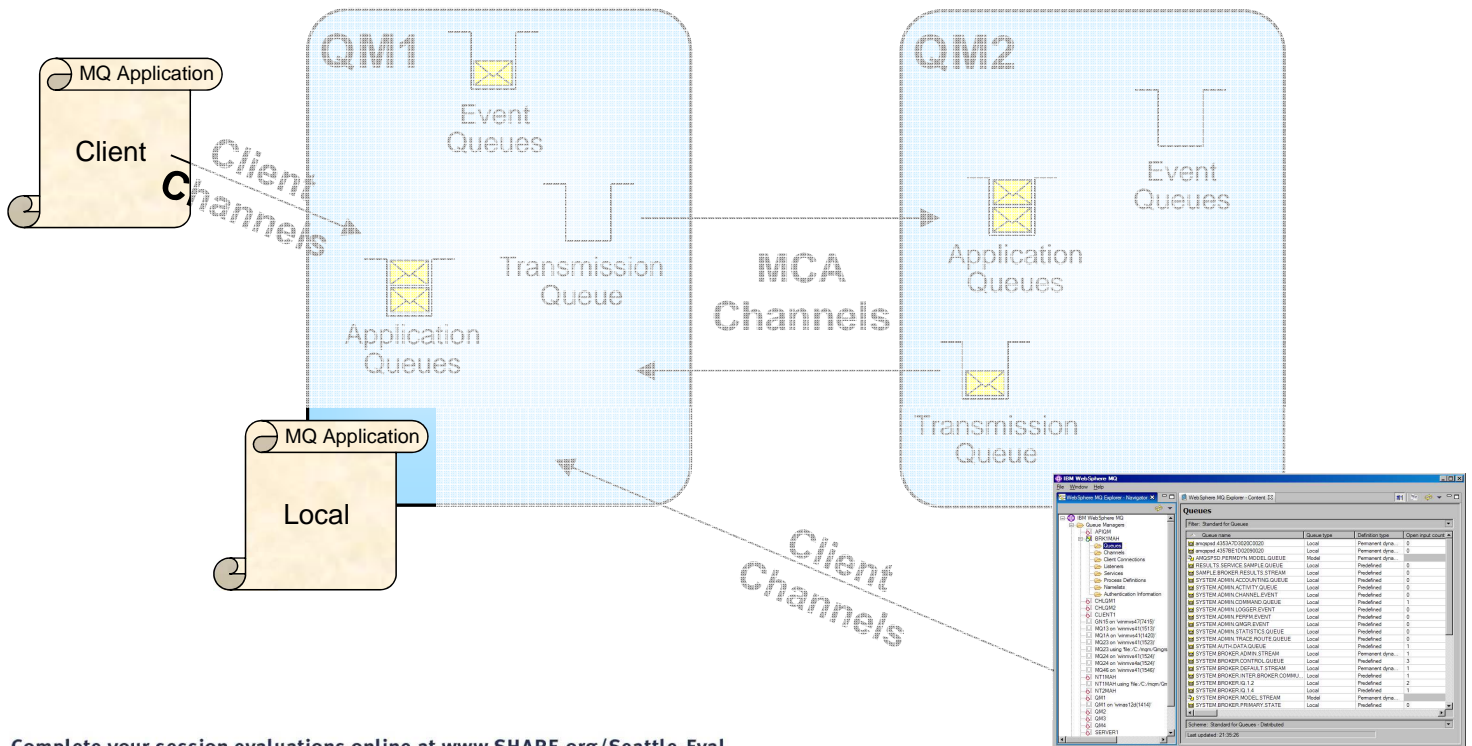
T

E

S

- One of the problems with authentication is that the industry could not decide how it should be done. Different environments suit different strategies and require different levels of security. The most common approaches seem to be third party authenticators such as Kerberos, SSL and Public/Private key encryption. WebSphere MQ decided that the most flexible approach was to make authentication a plug in service. That way each channel could have exactly the level of authentication it needed.
- Authentication can now be done without the use of a security exit, by using SSL and digital certificates and/or by using Channel Authentication Records.
- Security exits are the first channel exits to gain control of the conversation. They can exchange free format data with their remote partner, exchanging passwords, public keys etc to authenticate the remote partners request.
- No knowledge of communications is required. The exit merely passes a buffer of data back to the MCA who then transmits it to the partner machine. The data is received by the other MCA and then passed to the other security exit.
- If the security exit agrees with the authentication then it can change the default userid used for access control, known as the MCAuserid.
- A number of security exits are shipped as samples with the product. There are also some available for download from the supportpac web site. A number of third party products are also available.

# Authentication – Applications



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

@MoragHughson

# Authentication – Applications – Notes

N  
O  
T  
E  
S

- Now we shall look at authentication of applications, whether client connected or locally bound, which connect into your system.
- It is very important to ensure that you have good authentication mechanisms in place for any application. You must know that the connections made to your queue manager can be trusted.
- These applications may be business applications putting and getting messages from application queues, or administrative applications, issuing commands to the queue manager.

## Authentication – Applications – Facilities

- **O/S Logon**
  - Useful for locally bound applications
  - Not to be relied upon for client applications!
    - Use client channel authentication
- **MQCONN**
  - Connection Authentication
    - User ID and password

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



## Authentication – Applications – Facilities – Notes

N

- Over the next few pages we are going to introduce each of the following facilities which allow you to provide some authentication for your client or locally bound application.

O

### Identification

- When an MQ application connects to the queue manager the O/S is interrogated to discover the user ID that it is running under. This is used as the identity. We can see this user ID in the context information of a message.

T

### Authentication

- A locally bound MQ application is running against MQ under an user ID that the O/S has provided and which has been logged onto prior to running the application. This may be enough authentication for a locally bound application for your business purposes, or you may wish more.

E

S



# Connection Authentication - MQCONNX

- **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};

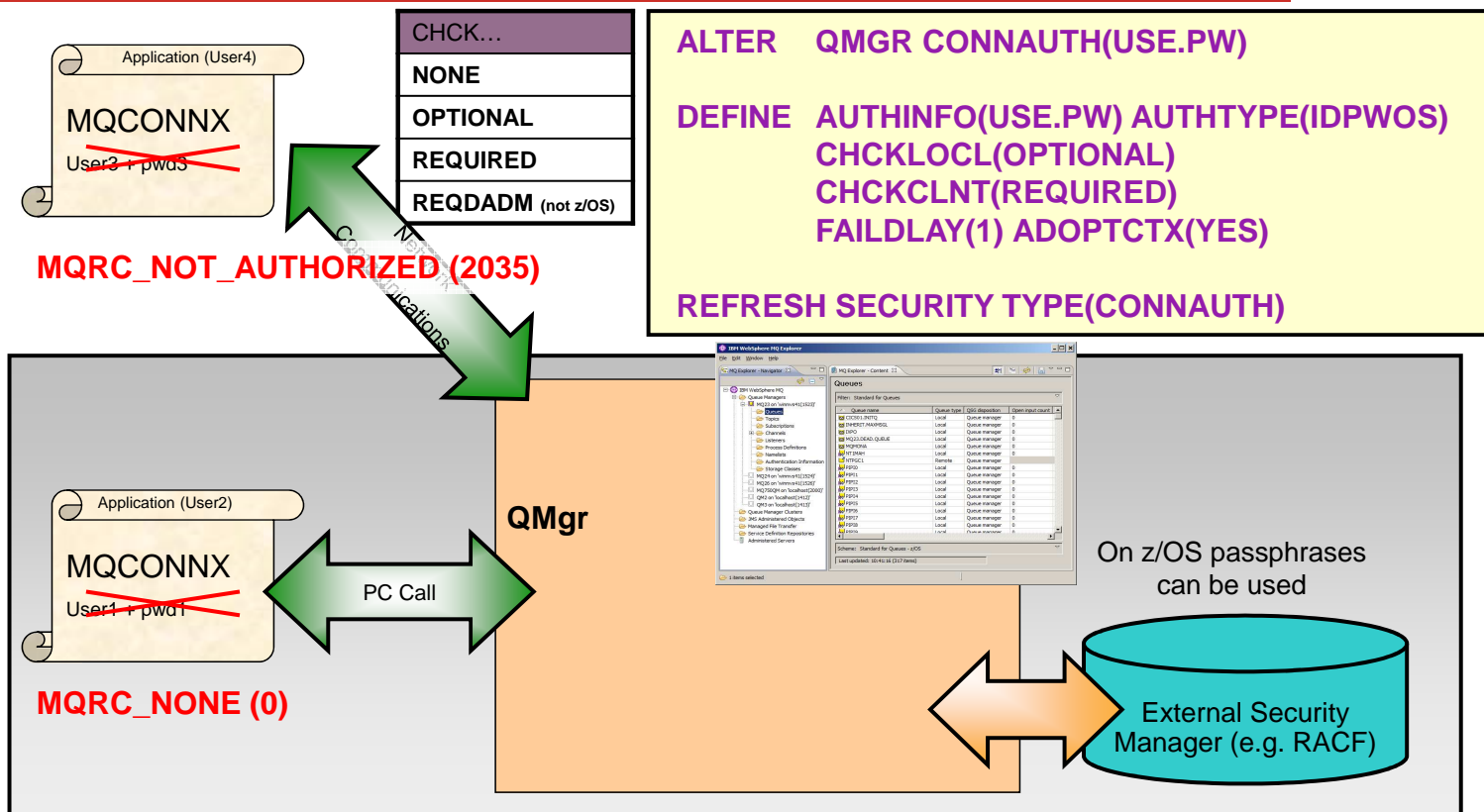
csp.AuthenticationType = MQCSP_AUTH_USER_ID_AND_PWD;
csp.CSPUserIdPtr       = "hughson";
csp.CSPUserIdLength    = 7;           /* Max: MQ_CLIENT_USER_ID_LENGTH */
csp.CSPPasswordPtr     = "passw0rd";
csp.CSPPasswordLength  = 8;         /* Max: MQ_CSP_PASSWORD_LENGTH */
```

# Authentication - MQCONNX - Notes

N  
O  
T  
E  
S

- On MQCONNX an application can provide a user ID and password (in the Connection Security Parameters (MQCSP) structure in the MQCNO), which are passed to the queue manager at V8 to be checked (depending on whether the queue manager is configured to do this).
- The MQCSP structure was available since WebSphere MQ V6, but a security exit or (custom OAM on distributed) needed to be written to check the password. Now this is built into the queue manager in V8.
- Pre V8, if the application is running client bound, this user ID and password are also passed to the client side and server side security exits. A server side security exit can be used to call RACF (for example) to check the user ID and password. A sample security exit is provided (CSQ4BCX3) to illustrate how to do this. This exit can be used for setting the MCAUser attribute of a channel instance.

# Connection Authentication - Configuration



## Configuration - Notes (1)

N  
O  
T  
E  
S

- We'll start with the basic configuration side of things. How do I turn on this connection authentication feature on the queue manager.
- On the queue manager object there is a new attribute called CONNAUTH (short for connection authentication) which points to an object name. The object name it refers to is an authentication information object of type IDPWOS – one of two new types. The other new type – IDPWLDAP – does not apply to z/OS. There are two existing types of authentication information objects from earlier releases of WebSphere MQ, these original two types cannot be used in the CONNAUTH field.
- We show here a new authentication information object which has two fields to turn on user ID and password checking, CHCKLOCL (Check Local connections) and CHCKCLNT (Check Client connections). Changes to the configuration of this must be refreshed for the queue manager to pick them up.
- Both of these fields have the same set of attributes, allowing for a strictness of checking. You can switch it off entirely with NONE; set it to OPTIONAL to ensure that if a user ID and password are provided by an application then they must be a valid pair, but that it is not mandatory to provide them – a useful migration setting perhaps; set it to REQUIRED to mandate that all applications provide a user ID and password; and, only on Distributed, REQDADM which says that privileged users must supply a valid user ID and password, but non-privileged users are treated as per the OPTIONAL setting.

## Configuration - Notes (2)

N  
O  
T  
E  
S

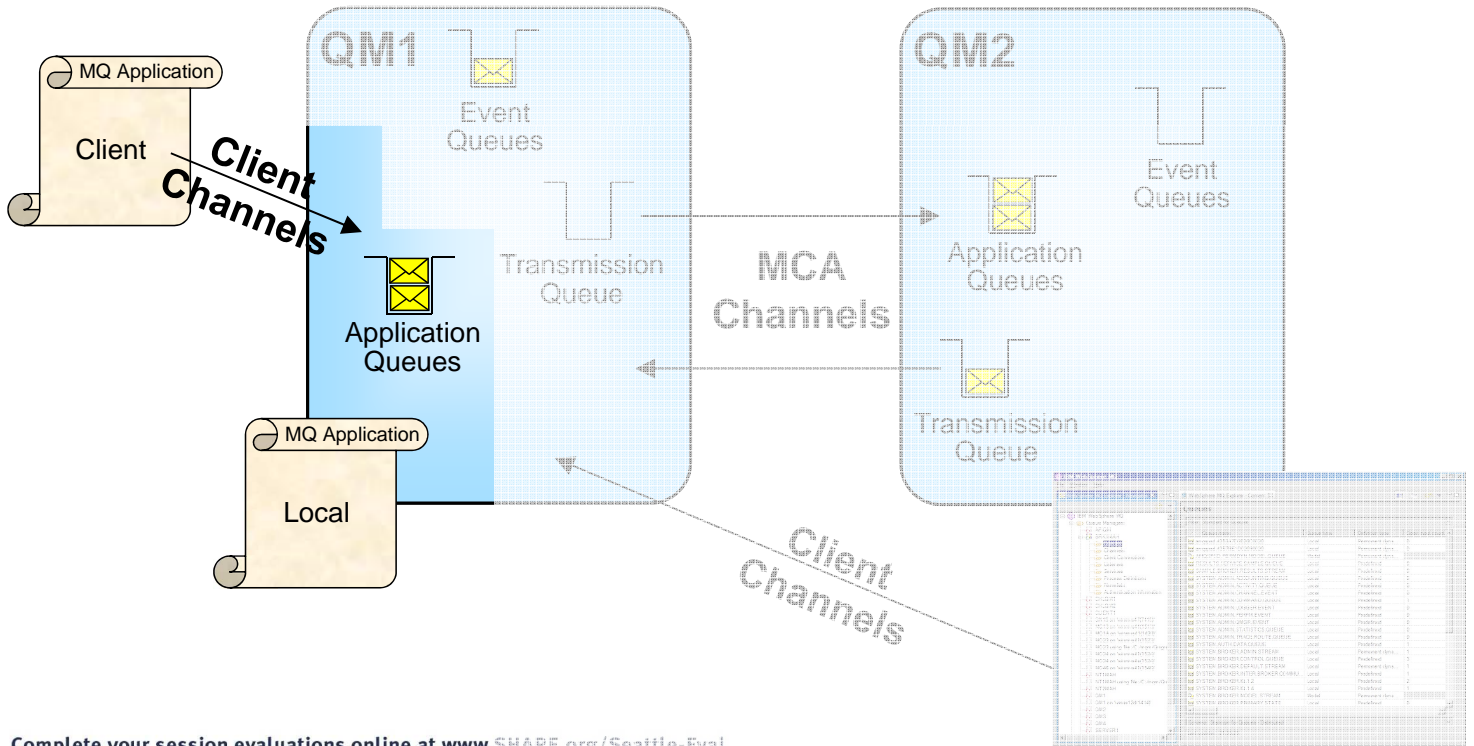
- Any application that does not supply a user ID and password when required to, or supplies an incorrect combination even when it is optional will be told 2035 (MQRC\_NOT\_AUTHORIZED). N.B. When password checking is turned off using NONE – then invalid passwords will not be detected.
- Any failed authentications will be held for the number of seconds in the FAILDLAY attribute before the error is returned to the application – just some protection against a busy loop from an application repeatedly connecting.

## Configuration - Notes (3)

N  
O  
T  
E  
S

- So we have seen that we can configure our queue manager to mandate user IDs and passwords are provided by certain applications. We know that the user ID that the application is running under may not be the same user ID that was presented by the application along with a password. So what is the relationship of these user IDs to the ones used for the authorization checks when the application, for example, opens a queue for output.
- There are two choices, in fact, controlled by an attribute on the authentication information object – ADOPTCTX.
- You can choose to have applications provide a user ID and password for the purposes of authenticating them at connection time, but then have them continue to use the user ID that they are running under for authorization checks. This may be a useful stepping stone when migrating, or even a desirable mode to run in, perhaps with client connections, because authorization checks are being done using an assigned MCAUSER based on IP address or SSL/TLS certificate information.
- Alternatively, you can choose the applications to have all subsequent authorization checks made under the user ID that you authenticated by password by selecting to adopt the context as the applications context for the rest of the life of the connection.
- If the user ID presented for authentication by password is the same user ID that the application is also running under, then of course this setting has no effect.

# Authorization – Application tasks



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Authorization – Application tasks – Notes

N  
O  
T  
E  
S

- Authorization is the process by which you allow particular users to access resources, such as queues or topics. Until you have some authentication in place however, you cannot trust the user ID. To be blunt, authorization without any authentication is simply not secure.
- We will introduce the authorization facilities available for allowing applications to access resources. In the next section we will look at administrative tasks, and some of these facilities will also be used for securing administrative tasks as they are often applications that put and get messages too, albeit to specific queues related to issuing commands.

## Authorization – Application tasks – Facilities

- **Connecting to the Queue Manager**
  - MQCONN or MQCONNX
- **Using WebSphere MQ resources**
  - MQOPEN
    - Queues, topics, namelists, processes
  - MQSUB
    - Topics
- **Setting message context**
  - Out of scope of this presentation

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



## Authorization – Application tasks – Facilities – Notes

N

O

T

E

S

- MQ provides access control facilities to control which users may run applications which issue MQCONN API calls. This will control which users may access the running Queue Manager.
- Once a program is connected to the queue manager, it is very likely that MQ resources will be used. The queue manager will control which users have access to which resources and in which way. Note that all (well, most) access control checks are made when a resource is opened. There are no resource checks made at MQGET and MQPUT time.

## Authorization – Security Switches on z/OS

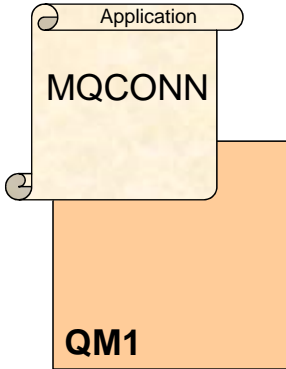
- **Subsystem security**
  - hlq.NO.SUBSYS.SECURITY
- **Qmgr or QSG Security**
  - hlq.NO.QMGR.CHECKS
  - hlq.NO.QSG.CHECKS
- **Connection Security**
  - hlq.NO.CONNECT.CHECKS
- **MQ Command Security**
  - hlq.NO.CMD.CHECKS
  - hlq.NO.CMD.RESC.CHECKS
- **MQ API Security**
  - hlq.NO.QUEUE.CHECKS
  - hlq.NO.NLIST.CHECKS
  - hlq.NO.PROCESS.CHECKS
  - hlq.NO.CONTEXT.CHECKS
  - hlq.NO.ALTERNATE.USER.CHECKS
  - hlq.NO.TOPIC.CHECKS

## Authorization – Security Switches on z/OS – Notes

N  
O  
T  
E  
S

- Switch profiles are RACF profiles which control the level of security checking carried out by WebSphere MQ. These profiles are not used in the same way that profiles are normally used - for access control. They are used simply as switches to activate/deactivate access control checking for various components of Queue Manager processing. Thus, userids are not permitted various access rights to these profiles.
- The default action for WebSphere MQ is to have access control checks activated. The presence of a switch profile will deactivate security checking for the appropriate component. As these switch profiles are not present by default, it means that explicit action is required to deactivate security processing within an WebSphere MQ environment, once RACF is active and the WebSphere MQ classes are defined (as no checking is possible otherwise !).
- If the hlq.NO.SUBSYS.SECURITY profile is present then no further checks are made for other switch profiles.
- Activating security support in this way means that no security system management is done from within the Queue Manager, which is deemed to be a good thing. However, the use of profiles in this way is quite unusual.

# Authorization – MQCONN(X) calls



```
RDEFINE MQCONN hlq.BATCH UACC(NONE)

PERMIT hlq.BATCH CLASS(MQCONN)
ID(usrname) ACCESS(READ)
```

| Connection Type   | MQCONN profile |
|-------------------|----------------|
| Batch             | hlq.BATCH      |
| CICS              | hlq.CICS       |
| IMS               | hlq.IMS        |
| Channel Initiator | hlq.CHIN       |

- **Connection Security**
  - hlq.NO.CONNECT.CHECKS

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



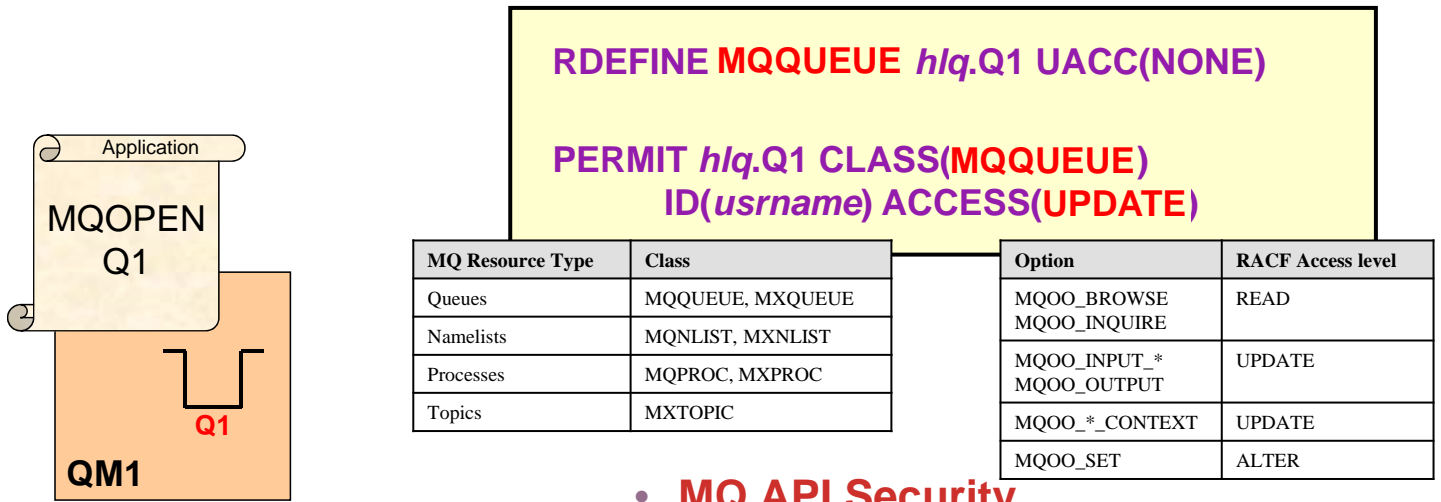
# Authorization – MQCONN(X) calls – Notes

N  
O  
T  
E  
S

- Before an application can work with MQ resources, it must first connect to the queue manager, using the MQCONN or MQCONNX call. This is the first point at which an application has a security check made against it.
- On z/OS, the check is made to see whether the user ID of the application has READ access to the appropriate profile in the MQCONN class. The profile is one of the four listed depending on the environment the application is running under.



# Authorization – MQOPEN calls



## • MQ API Security

- hlq.NO.QUEUE.CHECKS
- hlq.NO.NLIST.CHECKS
- hlq.NO.PROCESS.CHECKS
- hlq.NO.TOPIC.CHECKS

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

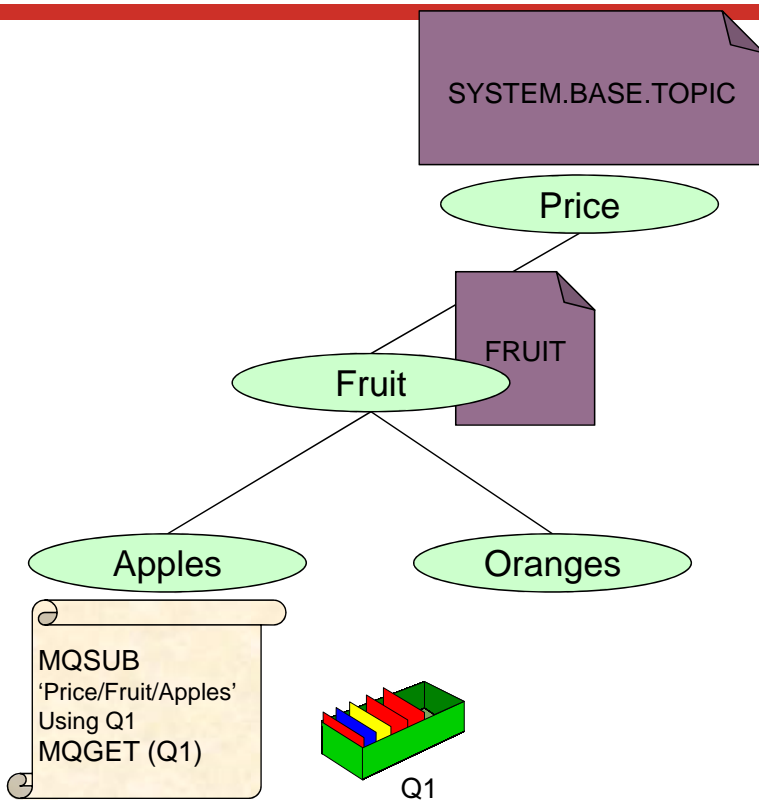


# Authorization – MQOPEN calls – Notes

N  
O  
T  
E  
S

- In order to work with an MQ resource, such as a queue or topic, the application must first open that resource with the MQOPEN call. This verb requires the application to state its intention for the resource, for example, to get from a queue, or to publish to a topic. This intention allows the correct security check to be made. Most authorization checks are made an open time instead of at get or put time to ensure the cost of MQGET and MQPUT are kept low.
- On z/OS, the check is made to see whether the user ID of the application has the relevant access to the appropriate profile in the MQQUEUE class (or other classes for other resource types).

# Topic Security



- **When an application Subscribes or Publishes to a Topic using**
  - MQSUB
  - MQOPEN / MQPUT1
- **When an application removes a subscription using**
  - MQCLOSE - with option MQCO\_REMOVE\_SUB
- **Authority check on topic objects**
  - “Walk up the tree”
  - May be more than one check
- **Authority check on destination queue**
  - When not using MQSO\_MANAGED
  - Check is for PUT to that queue

 @MoragHughson

# API Security - Topics - Notes

N

## **MQSUB**

- A security is performed during MQSUB processing to see whether the application making the request has the required access to that topic.
- An additional authorisation check is done for an MQSUB call when the application wishes to use a specific destination queue (i.e. is not using the MQSO\_MANAGED option). In this case we also check that this user ID has authority to PUT to that destination queue.

O

## **MQOPEN, MQPUT1**

- if an application is making a publication to a topic using an MQOPEN or MQPUT1 request a security check is performed to see whether that application has the required access to that topic.
- If an application is making a publication to a topic via an alias queue that resolves to a topic then two checks will take place, one to ensure that the application has access to the alias queue and then one to ensure that the application has the required access to the topic. This is additional processing to that when the alias queue points to another queue and is done to ensure that no matter how the topic tree is accessed, the same security is applied to it.

T

## **MQCLOSE**

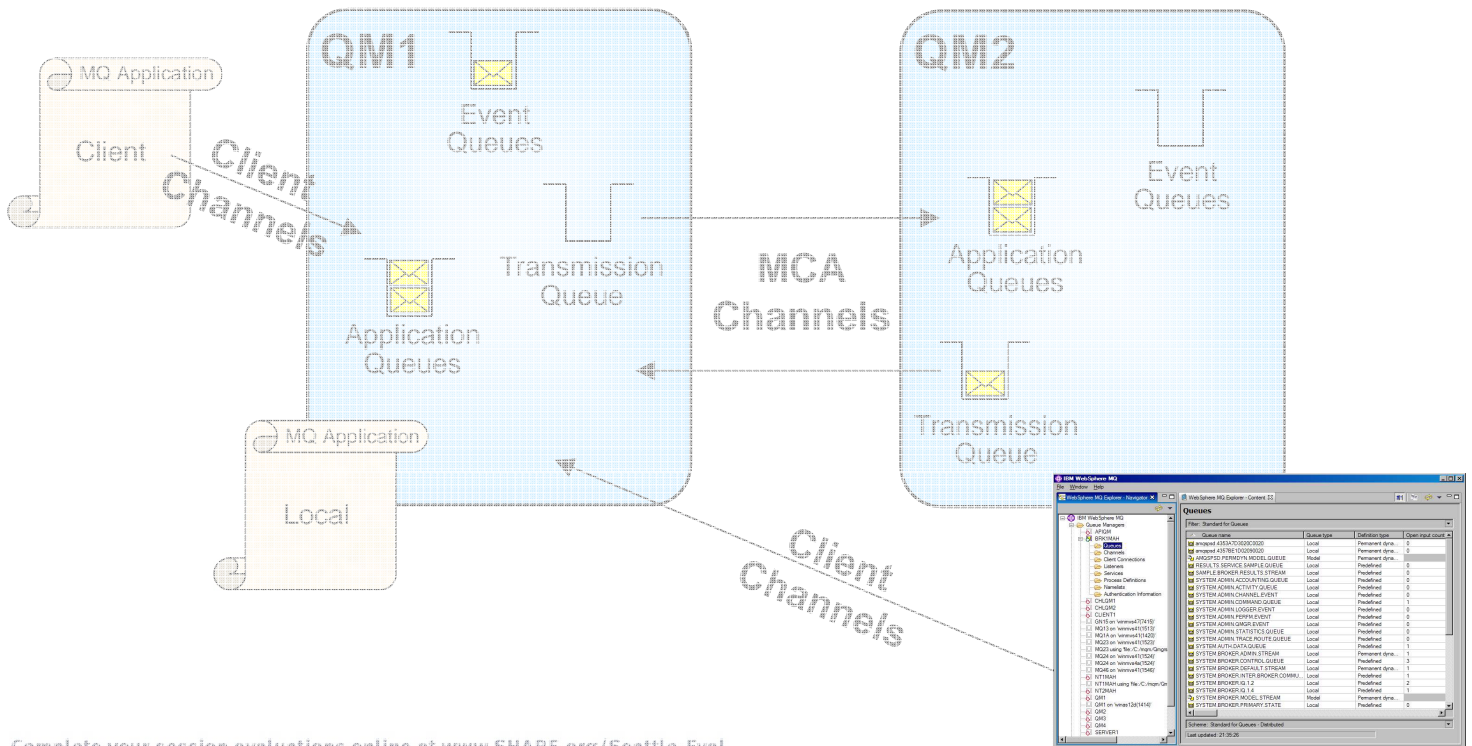
- A check can also be performed when an MQCLOSE is performed for a subscription using the remove sub option.

E

- In our example we have called MQSUB at the point in the topic tree, “Price/Fruit/Apples”. There is no topic object at this point in the topic tree, so to find the profile we need to check authorities against we walk up the topic tree to find a node which does have a topic object. The next point is “Price/Fruit”. This does have a topic object, FRUIT, so we will check that this user ID has subscribe authority on the profile for the FRUIT topic. If that user ID does have authority, our search stops there. If it does not, we carry on searching up the topic tree and will check the SYSTEM.BASE.TOPIC to see if this user ID has subscribe authority there. This means that the structure of your topic tree and the administration of it requires careful thought.

S

# Authorization – Administrative tasks



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

@MoragHughson

# Authorization – Administrative tasks – Notes

N  
O  
T  
E  
S

- Authorization is the process by which you allow particular users to issue commands. Until you have some authentication in place however, you cannot trust the user ID. To be blunt, authorization without any authentication is simply not secure.
- We will introduce the authorization facilities available for allowing administrative tasks to be performed. Remember some of the facilities we introduced in the previous section also apply here as administrative tasks are often done using applications that put and get messages too, albeit to specific queues related to issuing commands.

## Authorization – Administrative tasks – Facilities

---

- **MQSC/PCF commands**
  - DISPLAY QLOCAL
    - Read-only commands
  - DELETE QLOCAL
  - STOP CHANNEL
    - More destructive commands!
- Command & Command Resource security

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



## Authorization – Administrative tasks – Facilities – Notes

---

N

- Over the next few pages we are going to introduce each of the following facilities which allow you to provide authorize users to do administrative tasks on your queue manager.

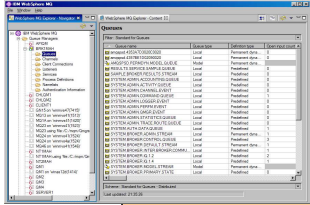
O

T

E

S

# Authorization – Read-only commands



QM1

```
RDEFINE MQCMDSDS hlq.DISPLAY.* UACC(NONE)
```

```
PERMIT hlq.DISPLAY.* CLASS(MQCMDSDS)  
ID(usrname) ACCESS(READ)
```

*hlq.verb.pkw*

- **MQ Command Security**
  - hlq.NO.CMD.CHECKS

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



# Authorization – Read-only commands – Notes

N  
O  
T  
E  
S

- To allow a user to issue read-only commands, that is the MQSC DISPLAY commands or PCF Inquire commands, you can grant them only the authority to those commands.
- On z/OS you can grant access to all the DISPLAY commands at once using a generic profile hlq.DISPLAY.\* If you want more granularity you can use profiles such as hlq.DISPLAY.CHANNEL or hlq.DISPLAY.NAMELIST, i.e. profiles of the form hlq.verb.primary-key-word.

## Authorization – MQ Command Security – Two Types

- **MQCMDS class - profiles look like**
  - hlq.verb.pkw
- **for example,**
  - hlq.DEFINE.QLOCAL
  - hlq.DEFINE.CHANNEL
- **Access required to profile is dependent upon the verb**
- **Controls who is allowed to issue each individual command**
- **Profiles always uppercase**
- **MQADMIN or MXADMIN class - command resource profiles look like**
  - hlq.type.resourcename
- **for example,**
  - hlq.QUEUE.queueName
  - hlq.CHANNEL.channelName
- **Access required to profile is dependent upon the verb and is usually ALTER or CONTROL**
- **Controls which resources a user can issue given commands against.**
- **'resourcename' can be mixed in MXADMIN**

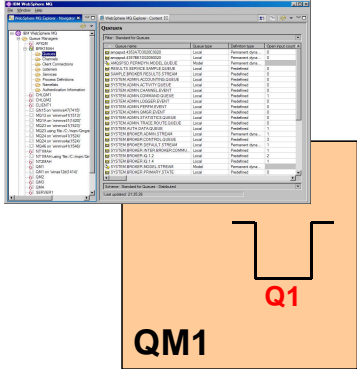
Together they allow very granular control over MQ commands

## Authorization – MQ Command Security - Notes

N  
O  
T  
E  
S

- These two profiles allow very granular control of the WebSphere MQ commands. There is a separate profile for each WebSphere MQ command (the verb) and target (the primary keyword), allowing each command to be controlled individually. Thus a particular userid may be able to define Local Queues but not define Remote Queues or may be able to display queues but not define queues. It is also possible to control access to the resources accessed by these commands. Thus, a user may be authorised to use the ALTER QLOCAL command but not alter a specified queue.
- Clearly, there is a price to pay with respect to this control; if this type of granular control is required then many profiles may need to be defined to facilitate this access control.
- Refer to [http://www.ibm.com/support/knowledgecenter/SSFKSJ\\_8.0.0/com.ibm.mq.sec.doc/q011730\\_.htm](http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.sec.doc/q011730_.htm) for a table showing the profiles and access required for each profile.

# Authorization – Other commands



```
RDEFINE MQCMDS hlq.ALTER.QLOCAL UACC(NONE)
RDEFINE MQADMIN hlq.QUEUE.Q1 UACC(NONE)

PERMIT hlq.ALTER.QLOCAL CLASS(MQCMDS)
ID(usname) ACCESS(ALTER)
PERMIT hlq.QUEUE.Q1 CLASS(MQADMIN)
ID(usname) ACCESS(ALTER)
```

| Command  | MQCMDS profile   | Access  | MQADMIN profile              | Access  |
|--|--|---------|------------------------------|---------|
| ALTER resource<br>DEFINE resource<br>DELETE resource | hlq.ALTER.resourcetype<br>hlq.DEFINE.resourcetype<br>hlq.DELETE.resourcetype | ALTER   | hlq.resourcetype.resourcenam | ALTER   |
| DISPLAY resource                                     | hlq.DISPLAY.resourcetype   | READ    | None                         |         |
| command CHANNEL                                      | hlq.command.CHANNEL  | CONTROL | hlq.CHANNEL.channelname      | CONTROL |
| Others e.g.<br>BACKUP, PING, RESET                   | hlq.command.pkw  | CONTROL | None                         |         |

- **MQ Command Security**

- hlq.NO.CMD.CHECKS
- hlq.NO.CMD.RESC.CHECKS

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

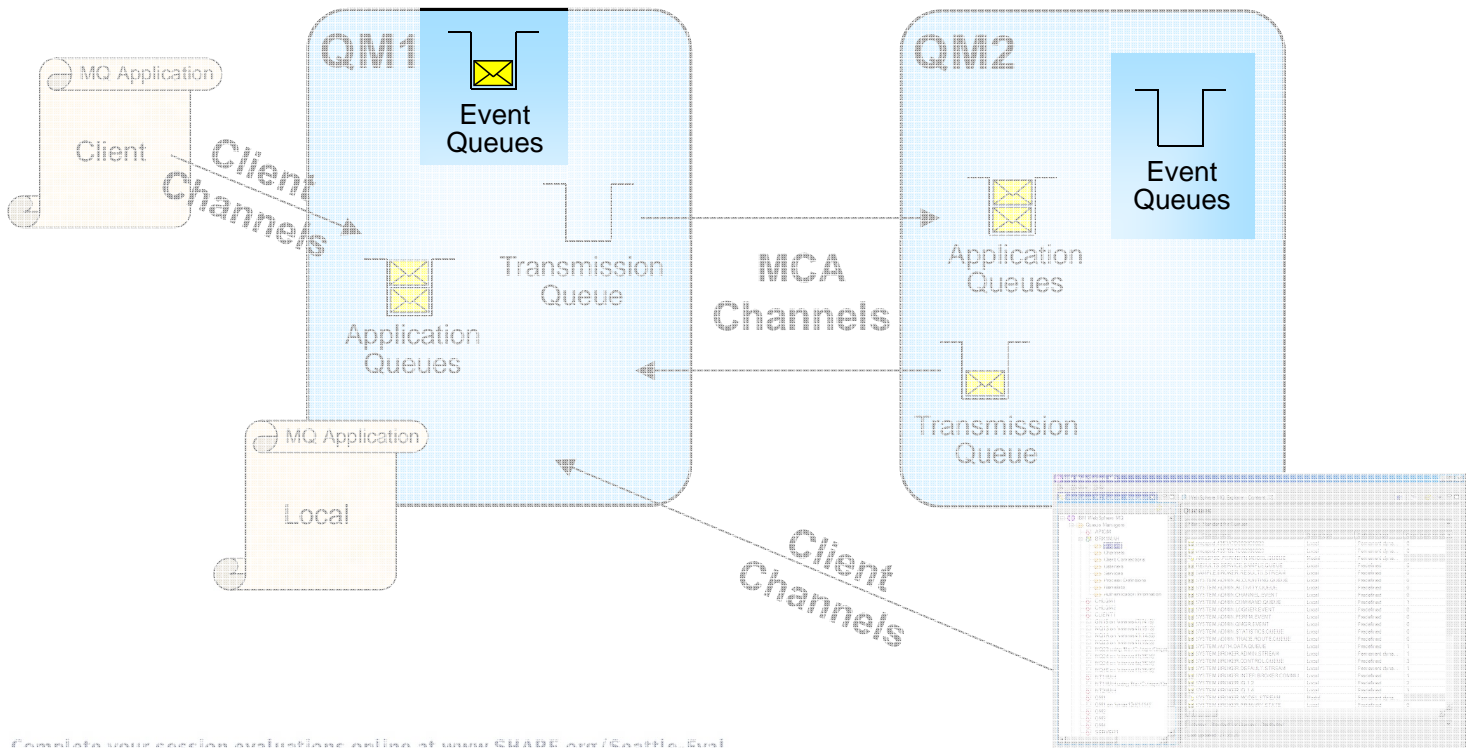
# Authorization – Other commands – Notes

NOTES

- To allow a user to issue other, perhaps more destructive, commands, you can grant them only the authority to those commands.
- On z/OS you can grant access to commands using profiles of the form hlq.verb.primary-key-word. The access required to these profiles varies a little by command, as shown in the table. A more complete table (this is a summary that holds true for the majority) can be found in the Information Center. In addition to granting access to the command profile, where a command works on a resource, e.g. DEFINE CHANNEL or ALTER QLOCAL you also grant access to the command resource profile which takes the form, hlq.object-type.object-name. We will cover this more on the next page.



# Auditing



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Auditing – Notes

N  
O  
T  
E  
S

- When WebSphere MQ needs to emit information it often does so in the form of event messages. Several of the auditing features in WebSphere MQ make use of this mechanism as well as many monitoring features.
- In addition on z/OS, we get auditing via the features of the External Security Manager (ESM) in use. This might be RACF, ACF2 or TopSecret for example.

# Auditing Facilities

---

- **Security Failures (z/OS)**
  - ESM facilities
- **Commands Issued**
  - Command events
- **Configuration Changes**
  - Configuration events

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



# Auditing Facilities – Notes

---

N

- Over the next few pages we are going to introduce each of the following facilities which allow you to audit your queue manager.

O

T

E

S

## Auditing – Security Failures (z/OS)

- **Standard External Security Manager (ESM) facilities, to record**
  - changes to security profiles and access to them
  - failed accesses to resources controlled by those profiles
  - successful accesses to resources controlled by those profiles
- **Reslevel audit records**
  - RACROUTE REQUEST=AUDIT
- **Controlled via**
  - ZPARM: RESAUDIT(YES|NO)
- **IMS Bridge audit records**
  - RACROUTE REQUEST=AUDIT

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

## Auditing – Security Failures (z/OS) – Notes

N

O

T

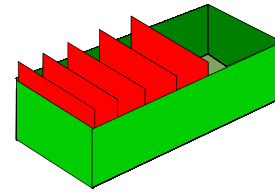
E

S

- When using the WebSphere MQ for z/OS queue manager, you can use the standard External Security Manager (ESM) facilities to create an audit trail for any changes made to your security set up.
- This can be set up to do any / all of the list shown depending on the ESM.
- In addition to the standard ESM facilities, there are two other types of audit records written. Due to the different way the enquiry is made to RACF, normal RACF audit records are not written so MQ requests a general audit record is written for these two types.
- Whether these RACF audit records are written for RESLEVEL security checks is controlled by ZPARM RESAUDIT(YES|NO).
- These RACF audit records for the IMS bridge cannot be turned off.

# Auditing – Commands Issued

- **Audit Trail of MQSC/PCF commands issued on your queue manager**
- **Queue Manager Attribute CMDEV**
  - NODISPLAY
- **Command Failed => No event**
- **Event contains**
  - Full command issued
  - User ID who issued it
  - Origin of command
    - Console
    - CSQINP data sets
    - Message on CommandQ



SYSTEM.ADMIN.COMMAND.EVENT

ALTER QMGR CMDEV(ENABLED)

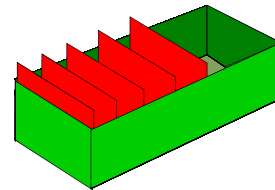
# Auditing – Commands Issued - Notes

N  
O  
T  
E  
S

- An audit trail of commands issued is kept by means of event messages which are written to the SYSTEM.ADMIN.COMMAND.EVENT queue. You can enable these events to be written by means of the CMDEV switch on ALTER QMGR.
- You can choose to record all commands that are issued, or perhaps more usefully, all commands except DISPLAY commands (PCF Inquire commands), so that you only capture a record of those potentially destructive or interesting commands. This is done using CMDEV(NODISPLAY).
- If the command issued failed, for example a syntax error, then no command event is generated.
- Command events are available on z/OS from V6 and Distributed platforms from V7.0.1.
- The contents of the command event message varies depending on how the command was issued. If the command was a PCF message then the content of the input PCF message is part of the command event. Alternatively, if the command was an MQSC message then this text string will be found in the event message instead of the PCF input message.
- If the command was issued by putting a message on the command server queue (MQEVO\_MSG) then there will be more application identifying information than in other cases because the Message Descriptor (MQMD) of the command message written by the application contains lots of extra data.
- In all cases you will get the user ID issuing the command, the queue manager where the command was entered, and one of the two aforementioned command data variants.

# Configuration Changes

- **Audit trail of changes to the configuration of the queue manager.**
  - Commands acting on objects
  - MQSET calls
- **Queue Manager Attribute CONFIGEV**
- **Create a base-line view with REFRESH QMGR**
- **Different Possible Events**
  - MQRC\_CONFIG\_CHANGE\_OBJECT
    - 2 event messages
    - Attributes before change
    - Attributes after change
  - MQRC\_CONFIG\_CREATE\_OBJECT
    - 1 event message
    - Attributes after create
  - MQRC\_CONFIG\_DELETE\_OBJECT
    - 1 event message
    - Attributes before deletion
  - MQRC\_CONFIG\_REFRESH\_OBJECT
    - 1 event message
    - Current attributes of object



SYSTEM.ADMIN.CONFIG.EVENT

**ALTER QMGR CONFIGEV(ENABLED)**

**REFRESH QMGR TYPE(CONFIGEV)**  
**OBJECT(ALL) NAME(\*)**



 @MoragHughson

# Configuration Changes - Notes

N  
O  
T  
E  
S

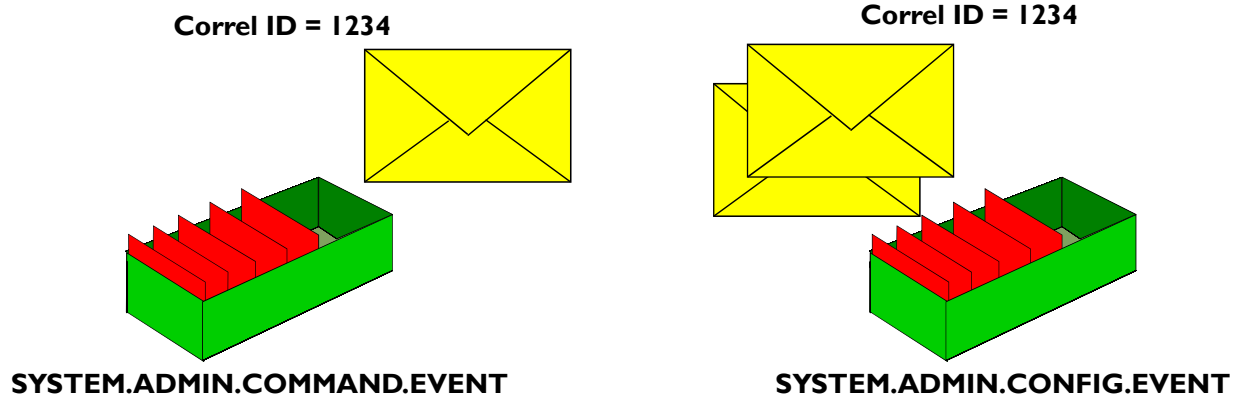
- An audit trail of changes to the queue manager configuration is kept by means of event messages which are written to the SYSTEM.ADMIN.CONFIG.EVENT queue. You can enable these events to be written by means of the CONFIGEV switch on ALTER QMGR.
- These events will be generated when a DEFINE, ALTER or DELETE command acts upon an object, or an MQSET command is used.
- A base-line picture of the current queue manager configuration can be created by using the REFRESH QMGR TYPE(CONFIGEV) command which will create an event message for every object in the queue manager. Since this could be a heavyweight operation if you have a lot of objects, you can break it down into smaller sets of objects using the NAME and OBJECT qualifiers on the command.
- The event message will record one of four possible Reasons, MQRC\_CONFIG\_CHANGE\_OBJECT, MQRC\_CONFIG\_CREATE\_OBJECT or MQRC\_CONFIG\_DELETE\_OBJECT for the respective MQSC or PCF commands that you might issue upon an object or MQRC\_CONFIG\_REFRESH\_OBJECT for those event messages written when creating the base-line picture.
- Config events are available on z/OS from V5.3 and Distributed platforms from V7.0.1.
- The contents of the command event message varies depending on how the command was issued just as with command events. If the command was issued by putting a message on the command server queue (MQEVO\_MSG) then there will be more application identifying information than in other cases because the Message Descriptor (MQMD) of the command message written by the application contains lots of extra data.
- In all cases you will get the user ID issuing the command, the queue manager where the command was entered.
- In the specific case of the MQRC\_CONFIG\_CHANGE\_OBJECT, you will get two messages, one containing the object attributes before the change and one containing those after the change.

# Combining Command and Config Events

- ALTER Q(FRED) MAXDEPTH(1)**

- Command Event
- Before Change Config Event
- After Change Config Event

ALTER QMGR  
CMDEV(NODISPLAY)  
CONFIGEV(ENABLED)



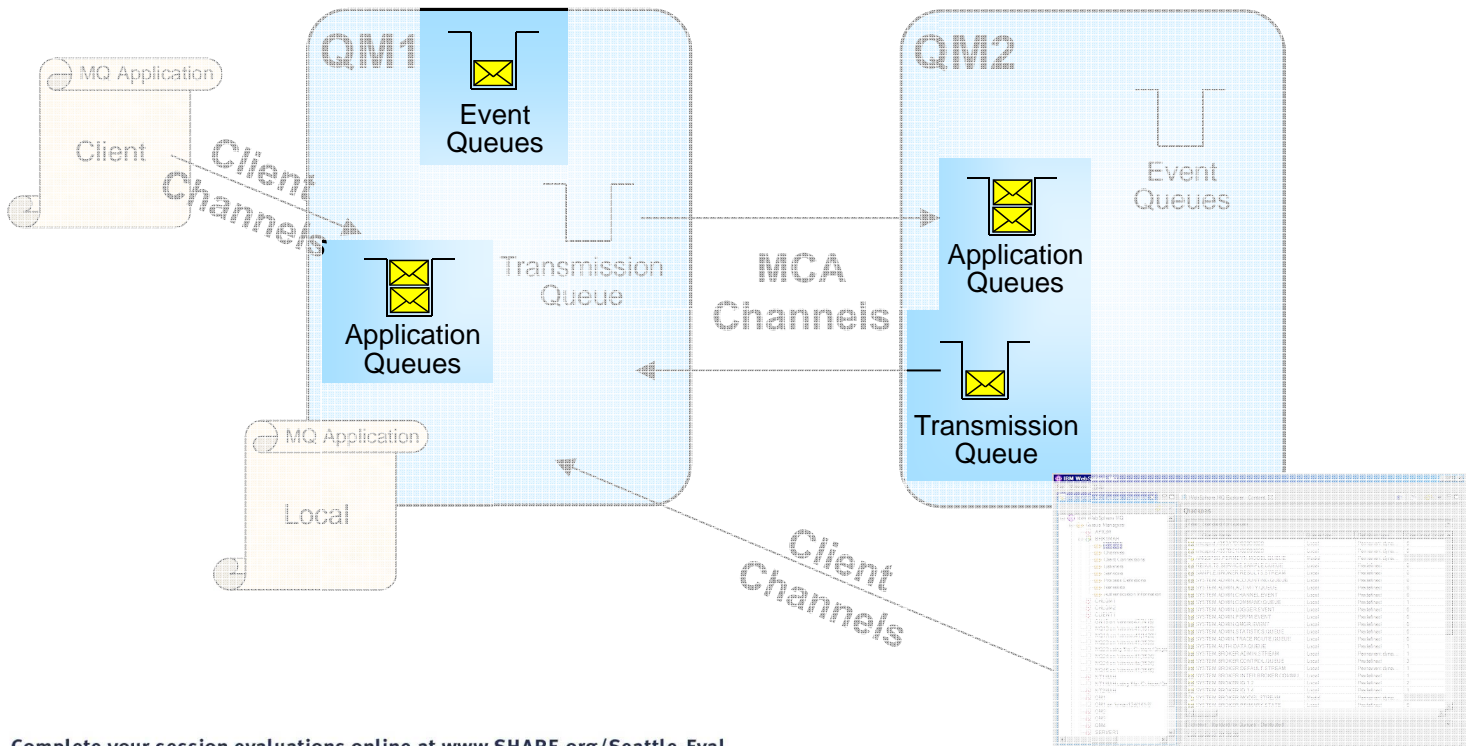
 @MoragHughson

## Combining Command and Config Events - Notes

N  
O  
T  
E  
S

- If you have both Command events and Configuration events enabled, then when an object is changed, the event messages will share the same correlation ID in their MQMDs.

# Message Level Protection



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

## Message Level Protection – Notes

N  
O  
T  
E  
S

- Advanced Message Security is a feature of WebSphere MQ that provides Application Level Security, also known as Message Level Protection.
- Message Level Protection provides assurance that messages have not been altered in transit. For example, when issuing payment information messages, ensure the payment amount does not change before reaching the receiver.
- Message Level Protection provides assurance that messages originated from the expected source . For example, when processing control messages, validate the sender.
- Message Level Protection provides assurance that messages can only be viewed by intended recipient(s). For example, when sending confidential information.



## Message Level Protection – Facilities

---

- **Message Integrity**
  - Digital signature allow detection of tampering
- **Message Privacy**
  - Encryption ensures only authorized recipients can read the message content

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson



## Message Level Protection – Facilities – Notes

---

N  
O  
T  
E  
S

- There are two types of message protection policies, message integrity policies where a digital signature is applied to the message, but the contents of the message remain in the clear; and message privacy policies where the contents of the message are also encrypted. Message privacy policies also include message integrity.

## Advanced Message Security - Message protection

- **Two types of policies:**

- Message Integrity policy
- Message Privacy policy

- **Created or updated or removed by command 'setmqspl' run using CSQ0UTIL**

- Defining message integrity policies
- Defining message privacy policies

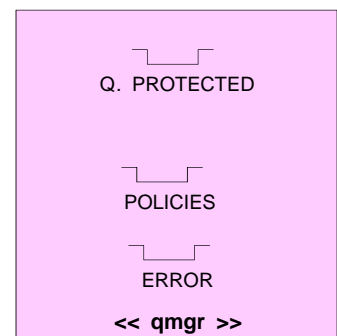
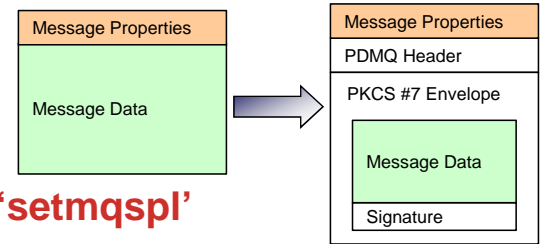
- **Policies are stored in queue 'SYSTEM.PROTECTION.POLICY.QUEUE'**

- **Display policies with command 'dspmqspl'**

- **Each protected queue can have only one policy**

- For distributed queuing, protect the queue locally (source QM) as well as the remote (target QM)

- **"Compromised messages" in queue 'SYSTEM.PROTECTION.ERROR.QUEUE'**

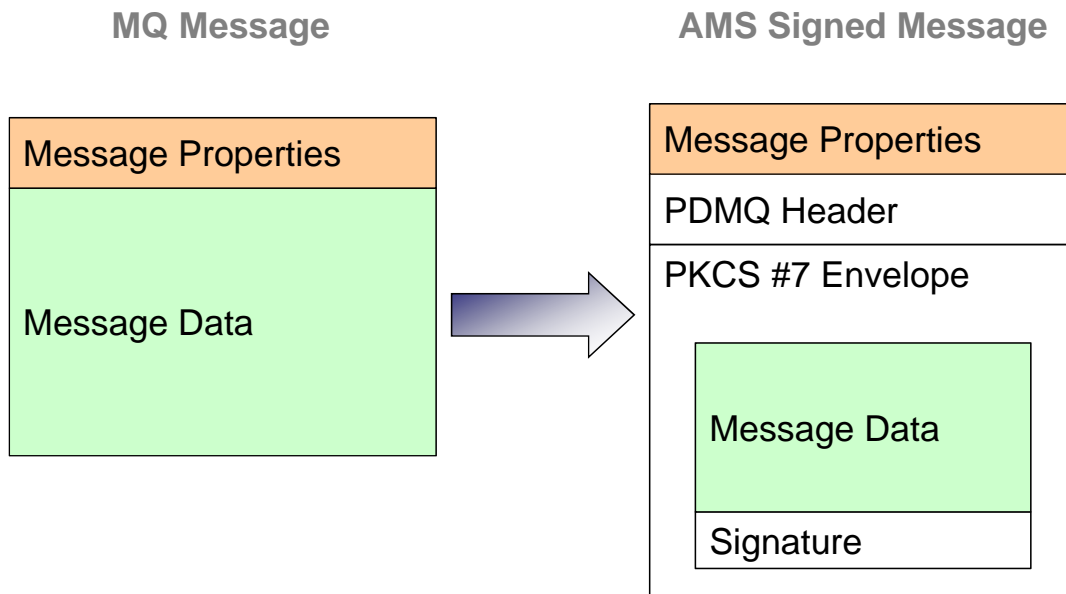


## Advanced Message Security – Notes

N  
O  
T  
E  
S

- Advanced Message Security (AMS) provides message protection policies to allow message content to be signed and encrypted. The application is unaware of the service and so the application programmer need not worry about coding it into his application, however, before the message is even placed on the queue it can be encrypted, thus ensuring that it's contents are never exposed. The message is encrypted while it resides on the queue, while it is transported across the network - the channels are unaware that the content is encrypted since they are content agnostic anyway - and is still encrypted when it is placed on the target queue. At the point where the receiving application gets the message off the queue the application level security service decrypts the data and presents it to the application.
- Configuration of these policies is done using the setmqspl (set MQ security policy) option on the CSQ0UTIL command. Once defined these policies are stored in a special queue called the SYSTEM.PROTECTION.POLICY.QUEUE. The policies can also be displayed, using the dspmqspl option of the CSQ0UTIL command.

# Integrity message format



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Message integrity policy definition

- **Signature algorithms:**
  - MD5, SHA1, SHA256\*, SHA384\* or SHA512\*
- **The list of authorized signers is optional**
  - If no authorized signers are specified then any application can sign messages.
  - If authorized signers are specified then only messages signed by these applications can be retrieved.
  - Messages from other signers are sent to the error queue
- **On z/OS, these parms are used as SYSIN DD for PGM=DRQUTIL**

## Syntax:

```
setmqspl
-m <queue_manager>
-p <protected_queue_name>
-s <SHA1 | MD5>
-a <Authorized signer DN1>
-a <Authorized signer DN2>
:
```

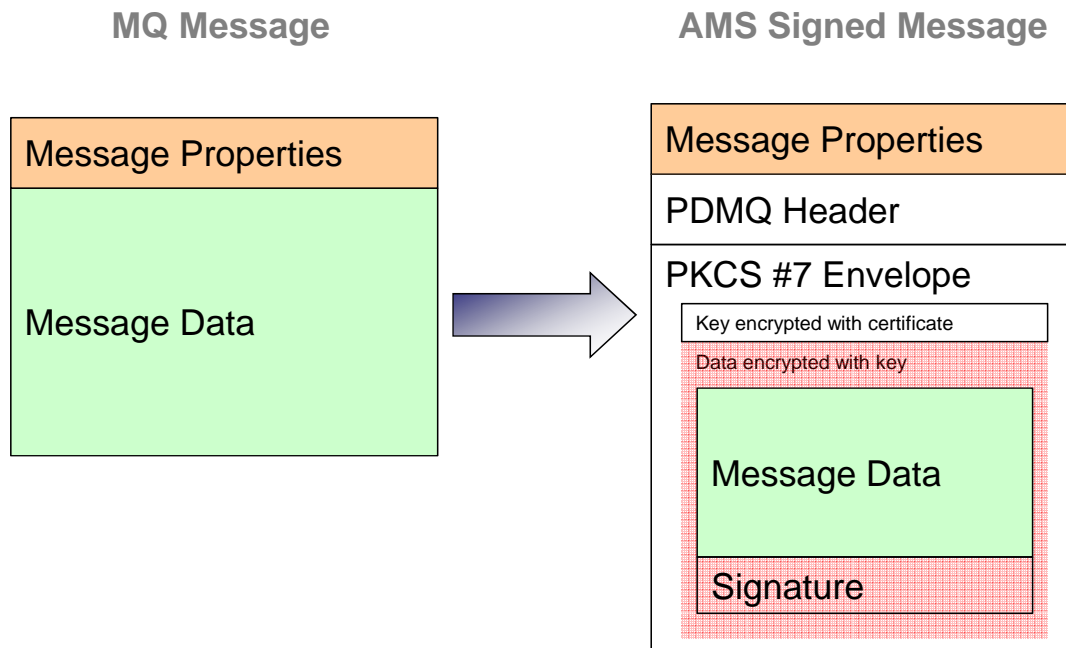
## Example:

```
setmqspl -m MYQM
-p MY.Q.INTEGRITY
-s SHA1
-e NONE
-a 'CN=carl,O=ibm,C=US'
```

Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Privacy message format



Complete your session evaluations online at [www.SHARE.org/Seattle-Eval](http://www.SHARE.org/Seattle-Eval)

 @MoragHughson

# Message privacy policy definition

- **Encryption algorithms:**
  - RC2, DES, 3DES, AES128 and AES256
  - Encrypted messages are always signed
- **The list of authorized signers is optional**
- **It is mandatory to specify at least one message recipient**
- **Retrieved messages which do not meet AMS policy sent to the SYSTEM.PROTECTION.ERROR.QUEUE**
  - Eg: Policy contains authorized signer list and sender is not on it

## Syntax:

```
setmqspl
-m <queue_manager>
-p <protected_queue_name>
-s <SHA1 | MD5>
-e <encryption algorithm>
-a <Authorized signer DN1>
-a <Authorized signer DN2>
-r < Message recipient DN1>
-r < Message recipient DN2>
```

## Example:

```
setmqspl -m MYQM
-p MY.Q.PRIVACY
-s SHA1
-e AES128
-a 'CN=carl,O=ibm,C=US'
-r 'CN=ginger,O=catunion,C=JP'
-r 'CN=saadb,OU=WBI,O=IBM,C=FR'
```

# Summary of WebSphere MQ Security Facilities

---

- **Authentication – Channels**
    - Transport Layer Security (SSL/TLS)
    - Channel Authentication Records
    - Security Exits
  - **Authentication – Applications**
    - O/S Logon
    - Connection Authentication
  - **Auditing**
    - Security Failures
    - Commands Issued
    - Configuration Changes
  - **Authorization – Applications**
    - Connecting to the Queue Manager
    - Using WebSphere MQ resources
  - **Authorization – Admins**
    - Control commands
    - MQSC/PCF commands
    - Command & Command Resource security
  - **Message Level Protection**
    - Message Integrity
    - Message Privacy
-