



IBM MQ for z/OS Security review

Carl Farkas
IBM Europe zWebSphere consultant
Internet : farkas@fr.ibm.com
(based largely upon slides from Morag Hughson and Marcel Amrein)

Agenda



- **Look at an example scenario**
 - What protectable resources do we have?
 - Who administers / uses the system?
 - Who is accessing what?
- **MQ for z/OS SAF Concepts**
 - Determine the level of protection – Switch profiles
 - How to protect various types of resources and operations

Non-agenda



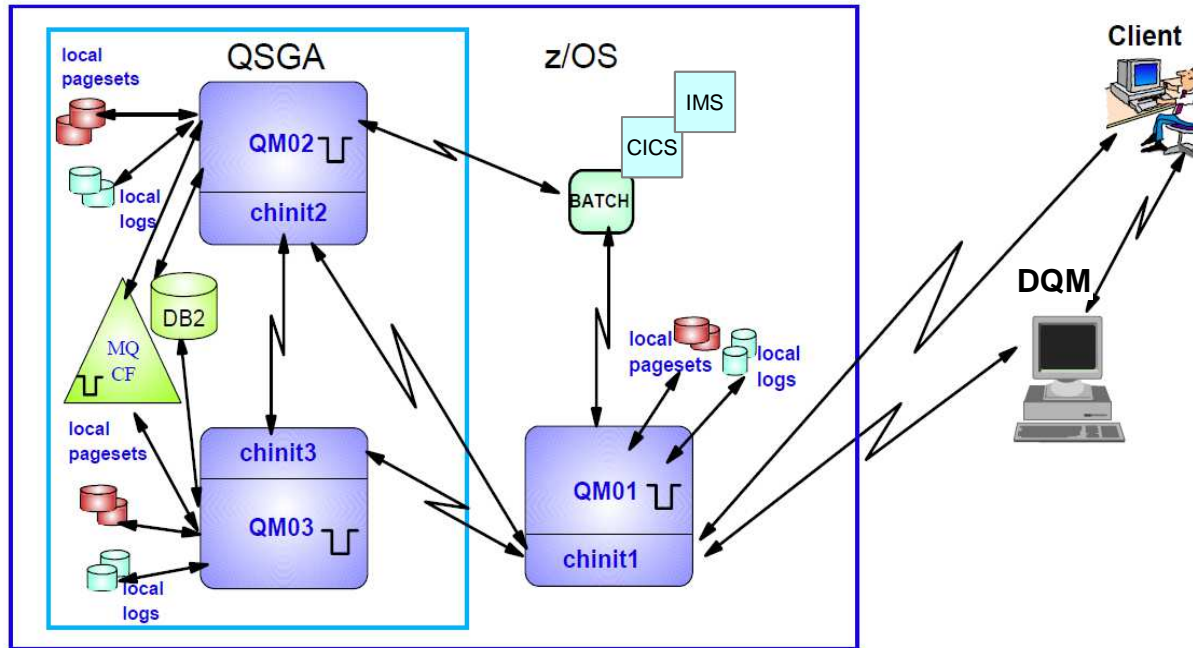
- Most of the MQ “cross-platform” (z/OS & Distributed) concepts, eg.
 - SSL
- Recent (cross-platform) enhancements, eg.
 - Channel Authentication
 - Connection Authentication
 - MQ AMS

Security “layers”

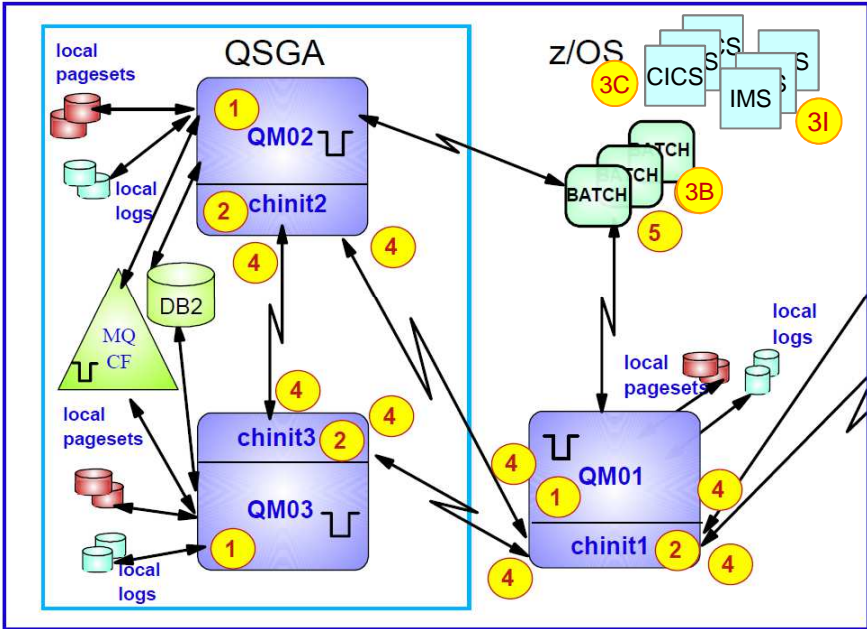


- **Identification**
 - Being able to uniquely identify a user of a system or an application
- **Authentication**
 - Prove that a user is who he says he is
- **Access Control (authorisation)**
 - Protects critical resources in a system by limiting access only to authorised users
- **Auditing**
 - Tracking who has done what to what and when
- **Confidentiality**
 - Protect your sensitive data from unauthorised disclosure
- **Data Integrity**
 - Check unauthorised changes have not been made to data

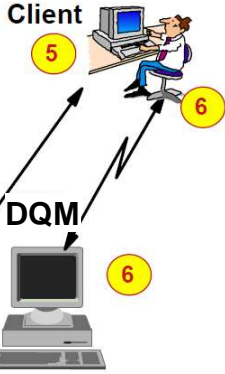
Example Scenario



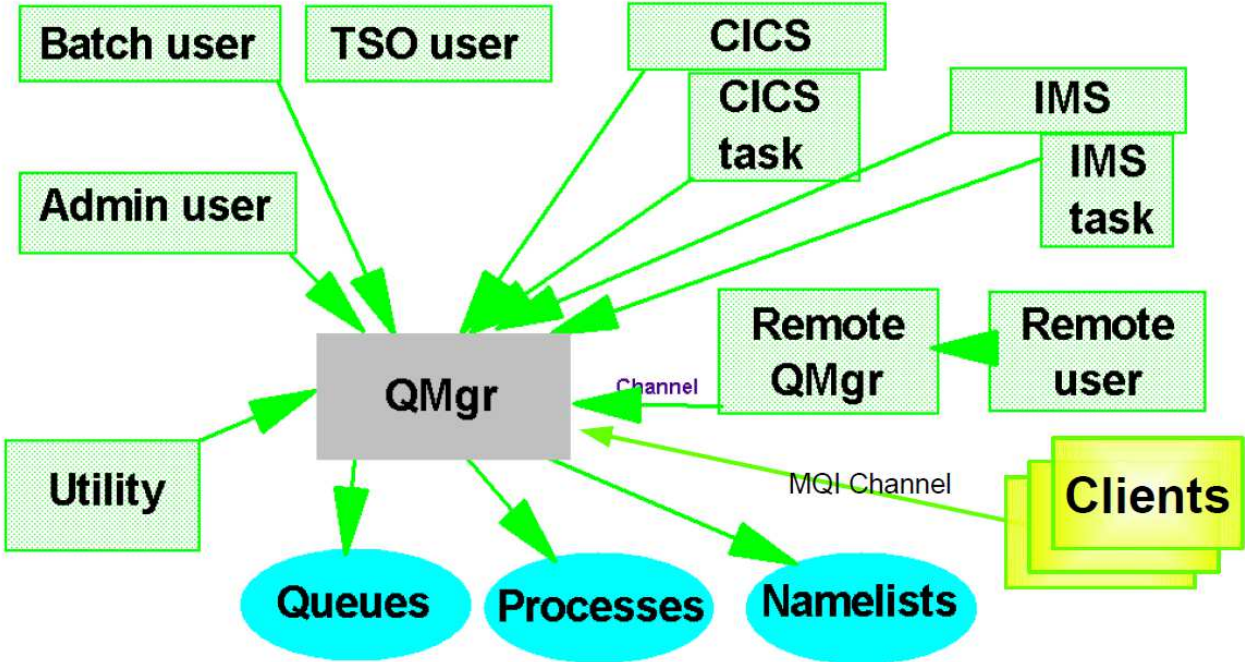
Example Scenario – User IDs involved



- 1 Queue Manager
- 2 Channel Initiator
- 3B Batch / TSO
- 3C CICS
- 3I IMS
- 4 MCA's
- 5 Administrators
- 6 Other Users



The more formal picture



Example Scenario – User IDs - Detail



<p>1</p> <p>Queue Manager id's QM01USR QM02USR QM03USR</p> <p>QM02 and QM03 in Queue Sharing Group QSGA</p> <p>2</p> <p>Channel Initiator id's MVR1USR MVR2USR MVR3USR</p> <p>3B</p> <p>Batch / TSO TSOUSR1 TSOUSR2 TSOUSR3</p> <p>3C</p> <p>CICS 3I STCCI01,STCCI02,STCCI03 CICUSR1,CICUSR2,CICSUSR3</p> <p>IMS STCIMS1,STCIMS2,STCIMS3 IMSUSR1,IMSUSR2,IMSUSR3</p>	<p>4</p> <p>Message Channel Agents (MCA's) QM01- channels MCAUSR1 MCAUSR2 MCAUSR3 MCAUSR4 QM02 - channels MCAUSR5 MCAUSR6 QM03 - channels MCAUSR7 MCAUSR8</p> <p>5</p> <p>Administrators TSOADM1 CLTADM1</p> <p>6</p> <p>Other Users CLTUSR1 DQMUSR1</p>
---	--

Agenda



- **Look at an example scenario**
 - What protectable resources do we have?
 - Who administers / uses the system?
 - Who is accessing what?
- **MQ for z/OS SAF* Concepts**
 - Determine the level of protection – Switch profiles
 - How to protect various types of resources and operations

* We have used the term “System Authorization Facility” (SAF) here, but the examples are all based upon IBM’s RACF product. An equivalent product should work in a similar way.

MQ z/OS dataset security best practices



Data Sets	MQ STC User	MQ Admin	Batch TSO	CICS IMS	App Dev
<h1q>.SCSQAUTH <h1q>.SCSQANLE	READ	READ	READ	READ	
<h1q>.SCSQMVR1	READ				
<h1q>.SCSQLOAD	READ	READ	READ	READ	READ
<h1q>.SCSQCICS	READ			READ	
<h1q>.SCSQMSGE <h1q>.SCSQPMLE <h1q>.SCSQTBLE <h1q>.SCSQSKL <h1q>.SCSQEXEC <h1q>.SCSQPNLS		READ			(READ)
<h1q>.SCSQPROC <h1q>.SCSQINST		(UPDATE)			

Fixed MQ SAF Classes



	Class	Group Class	used for ...
SCYCASE (MIXED)	MQADMIN MXADMIN	GMQADMIN GMXADMIN	<ul style="list-style-type: none"> Overall MQ admin security Component security "switches" RESLEVEL special profile Command resources
	MQCONN (upper only)	N/A	Application connection security
	MQCMDS (upper only)	N/A	Admin Command security
SCYCASE (UPPER)	MQQUEUE MXQUEUE	GMQQUEUE GMXQUEUE	Queue Resource profiles
	MQPROC MXPROC	GMQPROC GMXPROC	Process Resource profiles
	MQNLIST MXNLIST	GMQNLIST GMNLIST	Namelist Resource profiles
	MXTOPIC (mixed only)	GMXTOPIC	Topic profiles

General rules for MQ security classes



- You need to activate these classes (eg. SETROPTS CLASSACT) q011430_
- Many of the MQ SAF classes have two versions:
 - **MQADMIN** are the “classic” classes when using uppercase profiles or
 - **MXADMIN** for using mixed case profiles (introduced with MQv7)
- Use SCYCASE() QMGR attribute for UPPER or MiXeD case.
- Many of the classes have an equivalent “group class” which can be used to create single profiles that group resources with disparate names...

```
RDEFINE GMQUEUE MY.QUEUES UACC(NONE) OWNER(WMQADMIN) +  
  ADDMEM(QM01.CARLQ.PAY, +  
    QM01.SWALEH.FINANCE.QUEUE +  
    QM02.SYLVIE.MANUF.FILE +  
    QM02.PASCALE.CAR.Q)
```

[See RACF Security Administrator Guide.](#)

- All profiles are prefixed by either the the QMGR or QSG name
- Resource names in profiles can contain “*” (multiple character) or “%” (single character) wildcards
- And then there are MQ “switch” classes...

MQADMIN switch profile functions



Profile	Turns OFF security for ...
ssid.NO.SUBSYS.SECURITY qsg.NO.SUBSYS.SECURITY	... all security checking ("main switch") → qmgr running unsecured
ssid.NO.CONNECT.CHECKS qsg.NO.CONNECT.CHECKS	... connections → no checks on MQCONN
ssid.NO.QUEUE.CHECKS qsg.NO.QUEUE.CHECKS	... queue resources → no checks on q name on MQOPEN
ssid.NO.CMD.CHECKS qsg.NO.CMD.CHECKS	... commands → no checks on command processing
ssid.NO.CMD.RESC.CHECKS qsg.NO.CMD.RESC.CHECKS	... command resource checks → no checks resource names for cmds
ssid.NO.CONTEXT.CHECKS qsg.NO.CONTEXT.CHECKS	... MQMD context manipulation → no checks approp. MQOPEN option
ssid.NO.ALTERNATE.USER.CHECKS qsg.NO.ALTERNATE.USER.CHECKS	... Alternate user usage → no checks approp. MQOPEN option
ssid.NO.PROCESS.CHECKS qsg.NO.PROCESS.CHECKS	... process resources → no checks process inquiry
ssid.NO.NLIST.CHECKS qsg.NO.NLIST.CHECKS	... namelist resources → no checks namelist inquire
ssid.NO.TOPIC.CHECKS qsg.NO.TOPIC.CHECKS	... Pub/Sub resources → no checks on topic on MQOPEN

ssid | qsg.NO | YES.QMGR | QSG.CHECKS

Queue Manager start-up console messages



```
CSQH001I +MQ44 CSQHINIT Security using uppercase classes
CSQH024I +MQ44 CSQHINIT SUBSYSTEM security switch set
ON, profile 'MQ44.NO.SUBSYS.SECURITY' not found
CSQH024I +MQ44 CSQHINIT CONNECTION security switch
set ON, profile 'MQ44.NO.CONNECT.CHECKS' not found
CSQH024I +MQ44 CSQHINIT COMMAND security switch set
ON, profile 'MQ44.NO.CMD.CHECKS' not found
CSQH021I +MQ44 CSQHINIT CONTEXT security switch set
OFF, profile 'MQ44.NO.CONTEXT.CHECKS' found
CSQH021I +MQ44 CSQHINIT ALTERNATE USER security
switch set OFF, profile 'MQ44.NO.ALTERNATE.USER.CHECKS' found
CSQH021I +MQ44 CSQHINIT COMMAND RESOURCES security
switch set OFF, profile 'MQ44.NO.CMD.RESC.CHECKS' found
CSQH021I +MQ44 CSQHINIT PROCESS security switch set
OFF, profile 'MQ44.NO.PROCESS.CHECKS' found
CSQH021I +MQ44 CSQHINIT NAMELIST security switch set
OFF, profile 'MQ44.NO.NLIST.CHECKS' found
CSQH024I +MQ44 CSQHINIT QUEUE security switch set ON,
profile 'MQ44.NO.QUEUE.CHECKS' not found
CSQH024I +MQ44 CSQHINIT TOPIC security switch set ON,
profile 'MQ44.NO.TOPIC.CHECKS' not found
```

- Best practice: NO switches defined – check everything
(Note: in our example, we'll assume that security was NOT being checked... but we'll add it)

DISPLAY SECURITY Command Result



- May be issued at any time for the active Queue Manager
- Provides the same result- a bit more clearly arranged

```
+MQ44 DISPLAY SECURITY
CSQH015I +MQ44 Security timeout = 54 minutes
CSQH016I +MQ44 Security interval = 12 minutes
CSQH037I +MQ44 Security using uppercase classes
CSQH030I +MQ44 Security switches ...
CSQH034I +MQ44 SUBSYSTEM: ON, 'MQ44.NO.SUBSYS.SECURITY' not found
CSQH034I +MQ44 CONNECTION: ON, 'MQ44.NO.CONNECT.CHECKS' not found
CSQH034I +MQ44 COMMAND: ON, 'MQ44.NO.CMD.CHECKS' not found
CSQH031I +MQ44 CONTEXT: OFF, 'MQ44.NO.CONTEXT.CHECKS' found
CSQH031I +MQ44 ALTERNATE USER: OFF, 'MQ44.NO.ALTERNATE.USER.CHECKS'
found
CSQH031I +MQ44 PROCESS: OFF, 'MQ44.NO.PROCESS.CHECKS' found
CSQH031I +MQ44 NAMLIST: OFF, 'MQ44.NO.NLIST.CHECKS' found
CSQH034I +MQ44 QUEUE: ON, 'MQ44.NO.QUEUE.CHECKS' not found
CSQH034I +MQ44 TOPIC: ON, 'MQ44.NO.TOPIC.CHECKS' not found
CSQH031I +MQ44 COMMAND RESOURCES: OFF, 'MQ44.NO.CMD.RESC.CHECKS' found
CSQ9022I +MQ44 CSQHPDTC ' DISPLAY SECURITY' NORMAL COMPLETION
```

MQ Explorer Display

The screenshot shows the MQ Explorer interface. On the left, a tree view shows the hierarchy: MQ300 on 172.16.30.243(14030) > MQ44 on 172.16.36.244(44014) > Security. The 'Security' folder is expanded, showing 'Configuration' and 'Properties...'. The 'Configuration' folder is selected, displaying a table of Security Switches.

Security switch	Security switch setting	Security switch profile
Topic	Switch on; profile not found	MQ44.NO.TOPIC.CHECKS
SubSystem	Switch on; profile not found	MQ44.NO.SUBSYS.SECURITY
Queue	Switch on; profile not found	MQ44.NO.QUEUE.CHECKS
Connection	Switch on; profile not found	MQ44.NO.CONNECT.CHECKS
Command	Switch on; profile not found	MQ44.NO.CMD.CHECKS
Resources	Switch off; profile found	MQ44.NO.CMD.RESC.CHECKS
Process	Switch off; profile found	MQ44.NO.PROCESS.CHECKS

Activating Changes to MQ SAF Profiles



- After adding/deleting profiles or changing the access list to existing profiles ...

1) Issue the following RACF commands to pick up changes in RACF

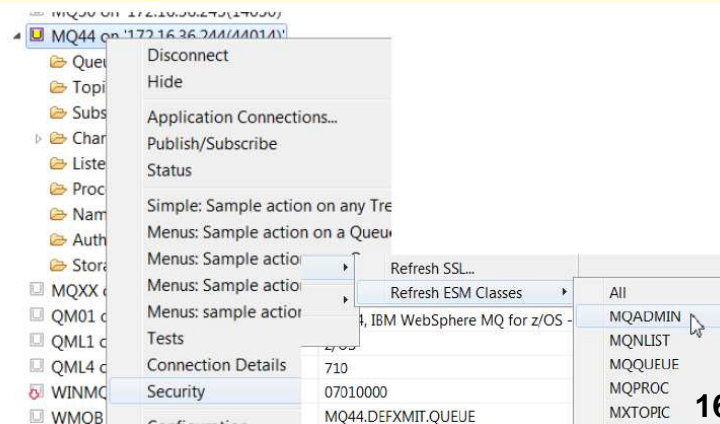
```
SETROPTS RACLIST (MQ...) REFRESH
```

```
SETROPTS GENERIC (MQ...) REFRESH
```

2) Issue the following MQ commands to refresh security settings on queue managers

```
+MQ44 REFRESH SECURITY (MQ... | *)
```

- The MQ commands can also be issued from the MQ Explorer

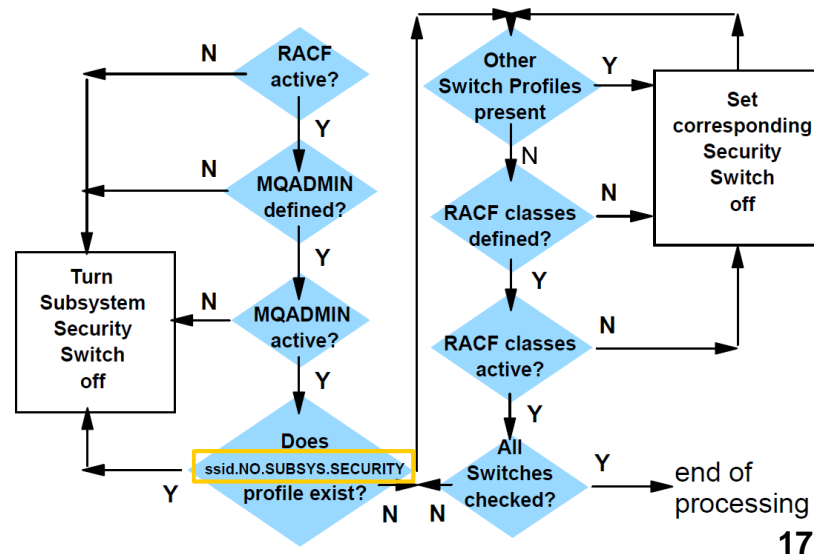


Switch profiles: the “main switch”

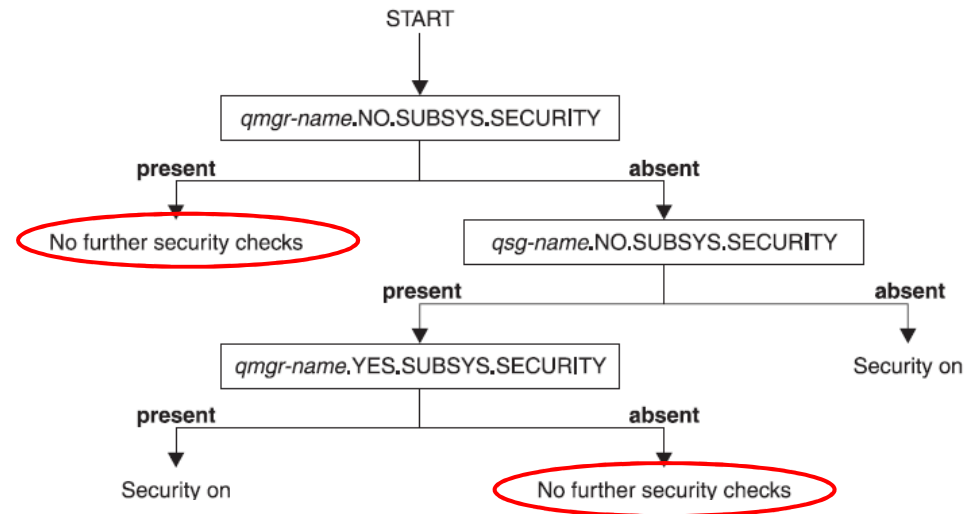
q011450_



- The existence or absence of particular profiles determines the security checkings performed by a queue manager or QSG. By default, all switches are ON for security.
- Ideally, to be fully secure, it's best to have NO switches enabled, although in practice most sites have some switches in place.



SUBSYS switch profile precedence



- Turning off security is clearly not a “best practice”, and would generally only be seen in a test environment

Queue Manager (ssid) or QSG profiles ?



q011490_

- MQ always uses prefixed SAF profiles
 - ✓ For a stand-alone queue manager, the prefix is the queue manager name, the z/OS subsystem ID
 - ✓ In a queue-sharing group environment, the QSG name may be used as the prefix.
- Additional “switch profiles” in CL(MQADMIN / MXADMIN) are used for finer control
- The use (or non-use) of these additional switch profiles can be controlled:

qmgr-name.NO.QMGR.CHECKS	no qmgr level checks for this queue manager
qsg-name.NO.QMGR.CHECKS	no qmgr level checks for this queue-sharing group
qmgr-name.YES.QMGR.CHECKS	queue manager level checks override for this queue manager
qmgr-name.NO.QSG.CHECKS	no QSG level checks for this queue manager
qsg-name.NO.QSG.CHECKS	no QSG level checks for this queue-sharing group
qmgr-name.YES.QSG.CHECKS	QSG level checks override for this queue manager

- Note that if you have SUBSYS security on, MQ will prevent you from turning off all checks for BOTH the QSG and the given QMGR.
- Use a simple approach requiring very few and clear switch profiles

“Main Switch” Settings for our scenario



QSGA

QM02

QM03

QM01

- For the QSGA queue-sharing group (queue managers QM02 and QM03), we want pure QSG-level checking so we will permanently have this profile defined:

```
RDEF CL(MQADMIN)
      QSGA.NO.QMGR.CHECKS UACC(NONE)
```
- Note that any universal access (UACC) level may be defined to the switch profiles- there is no access list!
- For the single queue manager QM01, we don't have to define anything to let security be active (be sure NOT to define `QM01.NO.SUBSYS.SECURITY`) or define `QM01.YES.SUBSYS.SECURITY`)

If setting security on for your queue manager, you might want to consider using RACF WARNING mode for the resource profiles to start so you can identify those without access before locking them out!

Start with “sub-area checks” switched OFF



- A practical starting point for a bit-by-bit implementing of MQ security is, to have subsystem security (formally) activated, but inhibit any checks by switching OFF all sub-areas.
- Profiles to be defined to class MQADMIN are:

QM01

QM01.NO.CONNECT.CHECKS

QM01.NO.CMD.CHECKS

QM01.NO.CMD.RESC.CHECKS

QM01.NO.QUEUE.CHECKS

QM01.NO.TOPIC.CHECKS

QM01.NO.PROCESS.CHECKS

QM01.NO.NLIST.CHECKS

QM01.NO.CONTEXT.CHECKS

QM01.NO.ALTERNATE.USER.CHECKS

QSGA (QM02 & QM03)

QSGA.NO.CONNECT.CHECKS

QSGA.NO.CMD.CHECKS

QSGA.NO.CMD.RESC.CHECKS

QSGA.NO.QUEUE.CHECKS

QSGA.NO.TOPIC.CHECKS

QSGA.NO.PROCESS.CHECKS

QSGA.NO.NLIST.CHECKS

QSGA.NO.CONTEXT.CHECKS

QSGA.NO.ALTERNATE.USER.CHECKS

Example RACF command to be used:

```
RDEFINE MQADMIN QM01.NO.CONNECT.CHECKS UACC(NONE)
```

CONNECT: Setting up connection security



q011540_

- Connection security is used if **no** ssid.NO.CONNECT.CHECKS switch profile is in place
- MQ connection security checks a user's permission to **MQCONNECT** to a system *from/within a particular environment*.
- Checks are performed for **READ access** against profiles in class MQCONN for these different “environments”:

What user ?

<code><prefix>.CHIN</code>	channel (MCA) users	The Channel Initiator address space user ID
<code><prefix>.IMS</code>	Connections from IMS	IMS region address space user ID(s)
<code><prefix>.CICS</code>	Connections from CICS	CICS region address space user ID(s)
<code><prefix>.BATCH</code>	All other connections: batch, TSO	Batch, TSO user, and also includes USS, DB2 stored procs, etc.

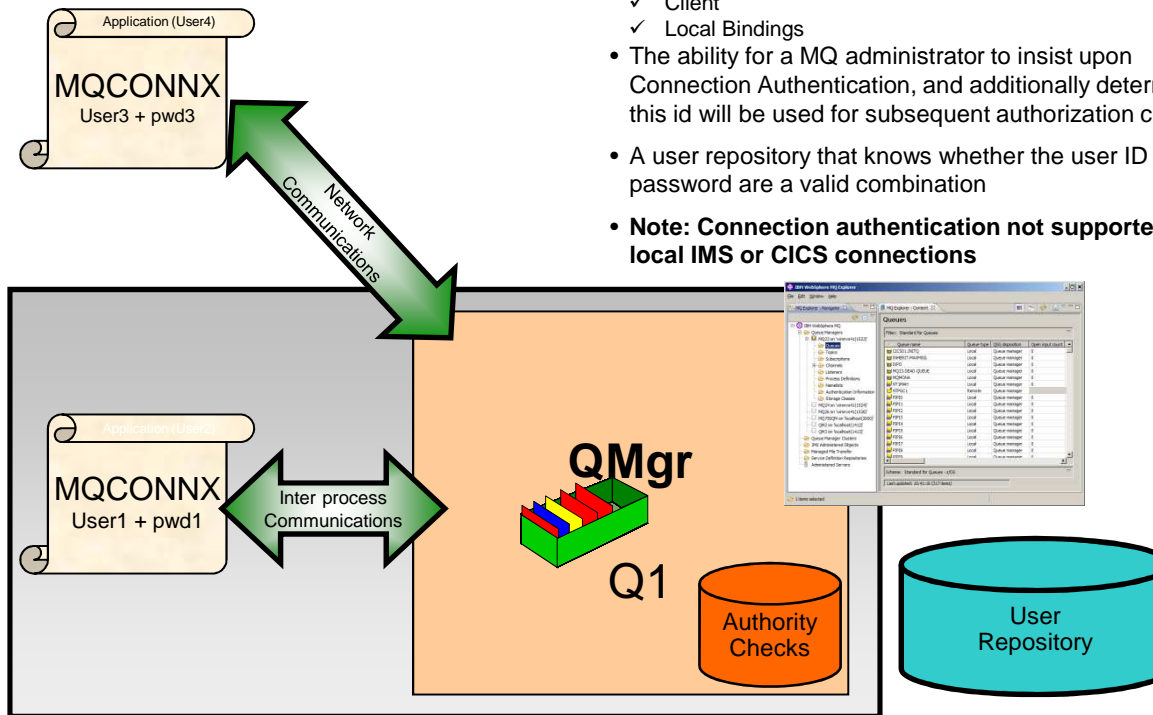
MQ v8 Connection Authentication

q0119860_

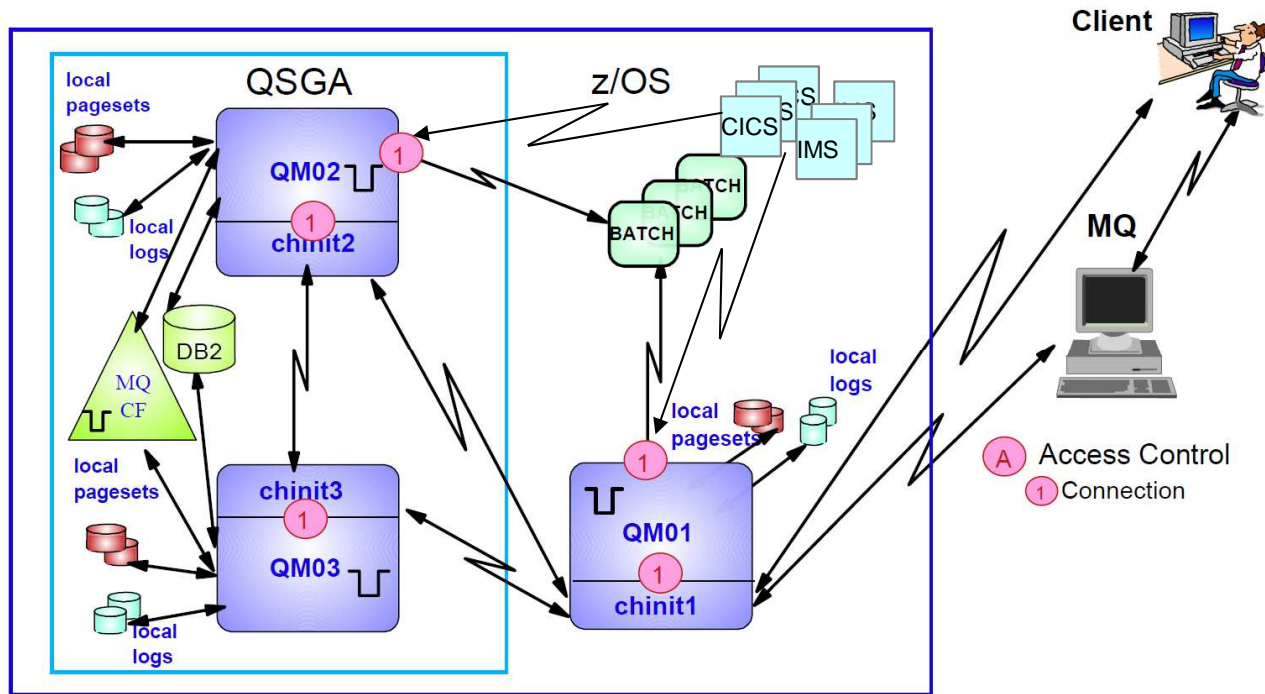


MQv8 introduced a new optional Connection Authentication feature:

- The ability for an application to provide a user ID and password
 - ✓ Client
 - ✓ Local Bindings
- The ability for a MQ administrator to insist upon Connection Authentication, and additionally determine if this id will be used for subsequent authorization checking
- A user repository that knows whether the user ID and password are a valid combination
- **Note: Connection authentication not supported for local IMS or CICS connections**



Sample scenario - connections



Sample scenario MQCONN profiles



With reference to our sample scenario, we have to define the following profiles in class MQCONN:

QM01.BATCH	connections coming into QM01 from Batch,TSO
QM01.CHIN	connections coming into QM01 from CHIN1
QM01.CICS	connections coming into QM01 from CICS
QM01.IMS	connections coming into QM01 from IMS
QSGA.BATCH	connections coming into QM02,QM03 from Batch,TSO
QSGA.CHIN	connections coming into QM02,QM03 from CHIN2 and CHIN3, respectively
QSGA.CICS	connections coming into QM02,QM03 from CICS
QSGA.IMS	connections coming into QM02,QM03 from IMS

... using RACF commands as follows:

```
RDEFINE MQCONN QM01.BATCH UACC(NONE)
```

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

...

Again, if setting security on for your queue manager, consider using RACF WARNING mode for the resource profiles to start

Authorizing users for connections



2 Channel Initiator id's

QM01CHIN - MVR1USR
QM02CHIN - MVR2USR
QM03CHIN - MVR3USR

... need access to their respective QMs

```
PERMIT QM01.CHIN CLASS(MQCONN)
ACC(READ) ID(MVR1USR)
PERMIT QSGA.CHIN CLASS(MQCONN)
ACC(READ) ID(MVR2USR,MVR3USR)
```

3B Batch / TSO

TSOUSR1
TSOUSR2

... need access to all queue managers

```
PERMIT QM01.BATCH CLASS(MQCONN) ACC(READ)
ID(TSOUSR1,TSOUSR2)
PERMIT QSGA.BATCH CLASS(MQCONN) ACC(READ)
ID(TSOUSR1,TSOUSR2)
```

3C CICS

STCCI01,STCCI02
CICUSR1, CICUSR2

... the CICS address space (STC) user IDs need access to all queue managers

```
PERMIT QM01.CICS CLASS(MQCONN) ACC(READ)
ID(STCCI01,STCCI02)
PERMIT QSGA.CICS CLASS(MQCONN) ACC(READ)
ID(STCCI01,STCCI02)
```

3I IMS

STCIMS1,STCIMS2
IMSUSR1, IMSUSR2

... so do the IMS address space (STC) user IDs

```
PERMIT QM01.IMS CLASS(MQCONN) ACC(READ)
ID(STCCI01,STCCI02)
PERMIT QSGA.IMS CLASS(MQCONN) ACC(READ)
ID(STCIMS1,STCIMS2)
```

Activating MQ connection security



When the MQCONN profiles access lists have been created, connection security can be switched ON for all queue managers

1) Remove the profiles that switch OFF connection security

```
RDELETE MQADMIN QM01.NO.CONNECT.CHECKS  
RDELETE MQADMIN QSGA.NO.CONNECT.CHECKS
```

2) Issue the following RACF commands to pick up changes in RACF

```
SETROPTS RACLIST(MQADMIN) REFRESH  
SETROPTS GENERIC(MQADMIN) REFRESH
```

3) Issue the following MQ commands to refresh security settings on queue managers

```
+QM01 REFRESH SECURITY(* )  
+QM02 REFRESH SECURITY(* )  
+QM03 REFRESH SECURITY(* )
```

Note: the **REFRESH SECURITY** MQ command has to be issued against every single queue manager, whether stand-alone or member of a QSG.

The RESLEVEL profile for z/OS

q011760_



- There is just one RESLEVEL profile for each system – `ssid/qsg.RESLEVEL`
- The RESLEVEL profile is *used to determine how many userids are to be checked* for resource (i.e. mainly **queue**) checks carried out on a given connection for the life of that connection.
- Note that setting RESLEVEL to CONTROL/ALTER also prevents AUDIT records for that id by default!
- At MQCONN time, the connecting user's authority is checked, and resource security is performed thereafter, as per the following rules:

- If no RESLEVEL exists, default is NONE. Best practice is to define this.
- For CHIN, the ids to be checked determined by channel type, and by PUTAUT option.

Userids checked?	NONE	READ	UPDATE	CONTROL/ALTER
TSO/job userid (or alternate)	Yes	Yes	Yes	No checking
CICS address space (or alternate)	Yes	Yes	Yes	No checking
CICS task userid (or alternate)	Yes	No	as RESSEC	No checking
IMS address space	Yes	Yes	Yes	No checking
IMS second userid (or alternate)	Yes	No	No	No checking
CHIN ID	2	1	1	

q011920_

- A common (but not necessarily best!) practice is setting UACC(READ) with no further access list:

```
RDEFINE MQADMIN Q%%.RESLEVEL UACC(READ)
```

Channel resulting Security checks

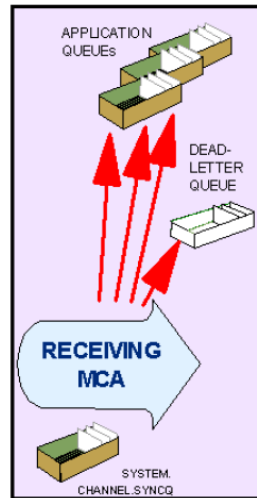
q011920_



RESLEVEL sets: 1 check

2 checks

QM2



(z/OS only)

PUTAUT specification	Channel User	MCA User	MQMD Context User
DEFAULT	X		
CTX	X		
ONLYMCA		X	
ALTMCA		X	

PUTAUT specification	Channel User	MCA User	MQMD Context User
DEFAULT	X	X	
CTX	X		X
ONLYMCA		X	
ALTMCA		X	X

- „1 check“ or „2 checks“ depends by the Chinit STC user's access permission to the RESLEVEL profile

Setting up Command Security



- Commands against MQ for z/OS may be issued in different ways, by different tools and methods
- Depending on the way you enter an MQ command, security checking differs:

Origin of command	User ID checked
Qmgr / Chin startup-JCL CSQINP*,CSQINPX data sets	No security check
Commands put to <code>SYSTEM.COMMAND.INPUT.QUEUE</code> (eg. Via CSQREXX menus, MQExplorer...or for channel connections perhaps modified by MCAUSER on channel, MQ channel authentication or connection authentication)	MQMD.UserID
z/OS console	Signed-on user CSQOPR (ZPARM)
SDSF / TSO	TSO or job user ID
CSQUTIL command option	Job user ID

Command Security Profiles

q011730_



- Note that MQ command security just protects the command itself and, in case of addressing objects, the **type** of the object affected by the command (the “primary keyword” pkw), which may be “QUEUE”, “QREMOTE”, “CHANNEL”, etc., *but not the object’s particular name*.
- You can restrict to particular queues, channels, etc... via Command Resource profiles (we’ll see later).
- As follows, some examples of MQCMDSD profiles – refer to the MQ Knowledge Center for the full list and associated access levels

Command	MQCMDSD profile	Access need
ALTER <pkw>	<prefix>.ALTER.pkw	ALTER
DEFINE <pkw>	<prefix>.DEFINE.pkw	ALTER
DELETE <pkw>	<prefix>.DEFINE.pkw	ALTER
DISPLAY <pkw>	<prefix>.DISPLAY.pkw	READ
MOVE QLOCAL	<prefix>.DEFINE.QLOCAL	ALTER
ARCHIVE LOG	<prefix>.ARCHIVE.LOG	CONTROL
START CHANNEL	<prefix>.START.CHANNEL	CONTROL

Authorizing Users for Commands



- What controls are required?
 - Probably lots of profiles to provide suitable protection to MQ commands
 - Use generic profiles where appropriate; implement granular control with specific profiles
- An extract of profiles and permissions to be defined
 - Permit the systems administrator to define channels and queues for queue-sharing group QSGA

```
RDEFINE MQCMDS QSGA.DEFINE.CHANNEL UACC(NONE)
PERMIT MQCMDS QSGA.DEFINE.CHANNEL ACC(ALTER) ID(TSOADM1)
RDEFINE MQCMDS QSGA.ALTER.QLOCAL UACC(NONE)
PERMIT MQCMDS QSGA.ALTER.QLOCAL ACC(ALTER) ID(TSOADM1)
```

- For QM01, let anyone perform DISPLAY commands, but grant full command control only to administrators

```
RDEFINE MQCMDS QM01.** UACC(READ)
PERMIT MQCMDS QM01.** ACC(ALTER) ID(TSOADM1,CLTAMD1)
```


Command Resource Security Profiles



- If you have a need to differentiate command permissions based on the *names of resources addressed by the command*, you have to activate Command resource security and provide appropriate profiles in class **MQADMIN**. q011740_

Command	MQADMIN cmd resource profile	Access needed
ALTER-DEFINE-DELETE <any type of queue>	<prefix>.QUEUE.<qname>	ALTER
ALTER-DEFINE-DELETE PROCESS	<prefix>.PROCESS.<procname>	ALTER
ALTER-DEFINE-DELETE NAMELIST	<prefix>.NAMELIST.<nlistname>	ALTER
DISPLAY <any type of resource>	NO CHECK !	N/A

- Applies to other resources, such as CHANNELs, AUTHINFOs, etc.

Authorizing Users for Command Resources



- **What controls are required?**
 - A practical use case for resource command security could be, to allow developers to maintain (DEFINE, ALTER, etc.) the objects of their own application/project, but to that of other ones.
 - In that case particularly, you will appreciate the value of a good naming convention
- **Example profiles and permissions for command resource protection**

```
RDEFINE MQADMIN QM01.QUEUE.ABC*.* UACC(NONE)
PERMIT MQADMIN QM01.QUEUE.ABC*.* ACC(ALTER) ID(GRADMIN,GRABC)
RDEFINE MQADMIN QM01.QUEUE.XYZ*.* UACC(NONE)
PERMIT MQADMIN QM01.QUEUE.XYZ*.* ACC(ALTER) ID(GRADMIN,GRXYZ)
```

- **However, if command resource security is active, it applies also to processes and namelists – so you may consider to define ...**

```
RDEFINE MQADMIN QM01.*.ABC*.* UACC(READ)
PERMIT MQADMIN QM01.*.ABC*.* ACC(ALTER) ID(GRADMIN,GRABC)
```

Securing Access to Queues

q011590_



- You are very likely to have a need to protect your queues
- Queue security checks are always performed at **MQOPEN** time based on the OPEN options (MQOO) specified by the application
- For dynamic queues, another check is performed at MQCLOSE
- **The profile is always `<prefix>.<qname>` in class **MQQUEUE****

MQOPEN option	Access needed
MQOO_INQUIRE MQOO_BROWSE	READ
MQOO_INPUT_*	UPDATE
MQOO_OUTPUT	UPDATE
MQOO_SET	ALTER

MQCLOSE option	Access needed
MQCO_DELETE MQOO_DELETE_PURGE	ALTER

- The same rules apply to namelist and process objects for which protection is controlled via a separate switch profile, with profiles
 - `<prefix>.<nlistname>` in class **MQNLIST**
 - `<prefix>.<processname>` in class **MQPROC**

Queue Profile Samples



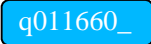
- **What controls are required?**
 - You probably start with protecting the SYSTEM queues
 - Transmission queues and dead-letter queues tend to require special permissions
 - A good naming convention for application queues allows you to use (few) generic profiles
- **Example profiles and permissions for queue protection for QSGA (roughly designed profiles)**

```
RDEFINE MQQUEUE QSGA.SYSTEM.* UACC(NONE)
RDEFINE MQQUEUE QSGA.ALIAS.*.DEADQ UACC(NONE)
RDEFINE MQQUEUE QSGA.ABC*.* UACC(NONE)

PERMIT MQQUEUE QSGA.SYSTEM.* ACC(ALTER) ID(GRADMIN,GRSTC)
PERMIT MQQUEUE QSGA.ALIAS.*.DEADQ ACC(ALTER) ID(GRADMIN,CLTADM1)
PERMIT MQQUEUE QSGA.ABC*.* ACC(UPDATE) ID(GRABC,GRADMIN)
```

- Best practice: create a “backstop” profile MQQUEUE ssid.** UACC(NONE)

Queue Access Requirements for CHIN User ID

- **Just to mention that the CHIN started task ID has very special security needs for SYSTEM.* queues** 
- In case you have to implement a granular queue protection

SYSTEM.ADMIN.CHANNEL.EVENT	UPDATE access required
SYSTEM.CHANNEL.INITQ	UPDATE access required
SYSTEM.CHANNEL.SYNCQ	UPDATE access required
SYSTEM.CLUSTER.COMMAND.QUEUE	ALTER access required
SYSTEM.CLUSTER.REPOSITORY.QUEUE	UPDATE access required
SYSTEM.CLUSTER.TRANSMIT.QUEUE	ALTER access required
SYSTEM.COMMAND.QUEUE	UPDATE access required
SYSTEM.COMMAND.REPLY.MODEL	UPDATE access required
SYSTEM.CSQXCMD.*	UPDATE access required
SYSTEM.QSG.CHANNEL.SYNCQ	UPDATE access required
SYSTEM.QSG.TRANSMIT.QUEUE	UPDATE access required

ALIAS Queue Considerations

q011750_

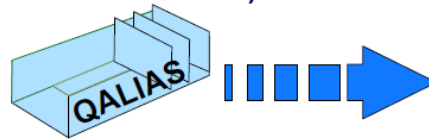


Access to Alias

OPTIONS = MQOO_OUTPUT
MQOD_OBJECTNAME = 'REQ1A'
MQOPEN

DEFINE QALIAS(REQ1A)

TARGQ(APPL1.REQUEST)



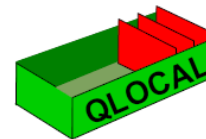
RACF profile checked
ssid.REQ1A in class MQQUEUE

No check against
ssid.APPL1.REQUEST

Access to base object

OPTIONS = MQOO_OUTPUT
MQOD_OBJECTNAME = 'APPL1.REQUEST'
MQOPEN

DEFINE QLOCAL(APPL1.REQUEST)



RACF profile checked
ssid.APPL1.REQUEST in class MQQUEUE

No check against
ssid.REQ1A

- A base queue and all its aliases all need specific protection

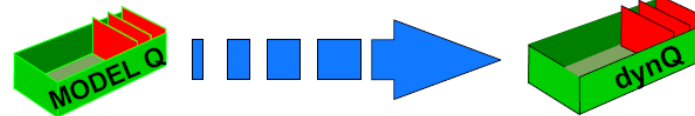
Model Queue and Dynamic Queues

q011620_



Requester side

Replier side



```
DEFINE QMODEL(REPLMODA) +  
  DEFTYPE(PERMDYN)
```

```
MQOD_OBJECTNAME = 'REPLMODA'  
MQOD_DYNAMICQNAME =  
  'APPL1.REPLY*'
```

MQOPEN

Profiles checked

```
ssid.REPLMODA  
ssid.APPL1.REPLY*
```

Dynamic 'APPL1.REPLY123456. ...'

```
MQOD_OBJECTNAME =  
  'APPL1.REPLY123456. ...'
```

MQOPEN, MQCLOSE with MQCO

Profile checked

```
ssid.APPL1.REPLY*
```

- To use a model queue, you must have access to both the model queue and the resulting dynamic queue

Controlling Access to Remote Queues

q011640_



Using a QREMOTE

```
DEFINE  
QREMOTE(REQ1A)  
RNAME(APPL1.REQUEST)  
RQMNAME(QM2)
```

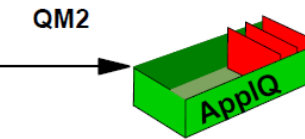
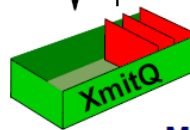
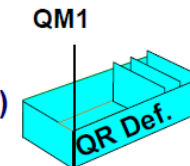
```
DEFINE QL(QM2)  
USAGE(XMITQ)
```

```
MQOD_OBJECTNAME =  
  'REQ1A'
```

MQOPEN

Profile checked
ssid.REQ1A
ssid.QM2

Using explicit remote QMgr & Q



```
DEFINE QL(B)  
DEFINE QL(APPL1.REQUEST)
```

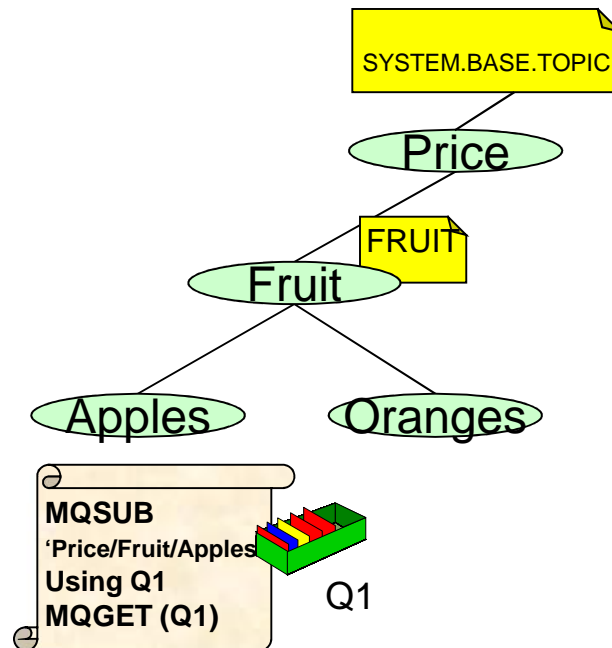
```
MQOD_OBJECTNAME =  
  'APPL1.REQUEST'  
MQOD_OBJECTQMGRNAME = 'QM2'  
MQOPEN
```

Profile checked
ssid.APPL1.REQUEST

- For remote queue access, access authorization required for the remote queue as well as the resolved QM name.

Topic (Pub/Sub) Security

q011680_



- ALTER needed on TOPIC when an application Subscribes or Publishes to that TOPIC using
 - ✓ MQSUB
 - ✓ MQOPEN / MQPUT1
- ALTER needed when an application removes a subscription using
 - ✓ MQCLOSE - with option MQCO_REMOVE_SUB
- Authority checks on topic objects
 - Only administered nodes (have an MQ object) can have security
 - “Walk up the tree” until the root looking for authorization
 - May be more than one check
- Authority check on destination queue, eg. Q1, for a Publish
 - When not using MQSO_MANAGED, check is for PUT to that queue

How the TOPIC Profiles Look Like



- SAF class **MXTOPIC**
- **Publish** protected by profiles `ssid.PUBLISH.<topicname>`
- **Subscribe** protected by profiles `ssid.SUBSCRIBE.<topicname>`

- A practical approach:

```
RDEFINE MXTOPIC WMQ7.SUBSCRIBE.SYSTEM.BASE.TOPIC  
ACC(NONE)
```

```
RDEFINE MXTOPIC WMQ7.SUBSCRIBE.PRICE ACC(NONE)
```

so by default, no one can access PRICE or its children

- Authorize the user group that needs access

```
PERMIT WMQ7.SUBSCRIBE.PRICE CLASS(MXTOPIC)  
ID(GROUP) ACC(ALTER)
```

- Of course, generic profiles are possible- but handle with care:

```
PERMIT WMQ7.SUBSCRIBE.PRICE.** CLASS(MXTOPIC)  
ID(GROUP) ACC(ALTER)
```

Backup



Bibliography



- MQ Knowledge Center, for the Security topic, search “[q009710](#)”
- WebSphere MQ for z/OS System Setup Guide (SC34-6927)
- WebSphere MQ Security (SC34-6832)
- IBM DeveloperWorks “WebSphere MQ for z/OS security” article at http://www.ibm.com/developerworks/websphere/library/techarticles/0906_schneider/0906_schneider.html
- IBM DeveloperWorks “Getting started with WebSphere MQ for z/OS security” at http://ibm.com/developerworks/websphere/library/techarticles/1103_schneider/1103_schneider.html
- IBM DeveloperWorks “What you didn’t know about WebSphere MQ security” article at http://www.ibm.com/developerworks/websphere/techjournal/0701_col_wyatt/0701_col_wyatt.html
- WebSphere MQ Security in an Enterprise Environment (SG24-6814)
- IBM MQ V8 Features and Enhancements (SG24-8218)
- Secure Messaging Scenarios with WebSphere MQ (SG24-8069)
- IBM WebSphere MQ V7.1 and V7.5 Features and Enhancements (SG24-8087)

What is MQ AMS?



WebSphere MQ Advanced Message Security

- Announced and available since 2010
- Provides security for MQ messages, **end-to-end** with no application changes
- It is a simple “add-on” product that enhances IBM MQ v7, v8
- Security policies are used to define the security level **per queue** which leverage X.509 certificates

