MQ Channel Authentication Records

Morag Hughson - hughson@uk.ibm.com

IBM Hursley - UK

Abstract

- WebSphere MQ V7.1 introduced a new feature for securing channels, known as Channel Authentication Records, or CHLAUTH for short. This new feature allows you to set rules to indicate which inbound connections are allowed to use your queue manager and which are banned. In V8, CHLAUTH was updated to tie in with a number of other new security features, including connection authentication (using CHCKCLNT on CHLAUTH); more advanced certificate checking (using SSLCERTI on CHLAUTH) and hostname support.
- This session will take you through the concepts behind this feature, how to create these rules and how to monitor and manage their use.

NOTES

Capitalware's MQ Technical Conference v2.0.1.4

Channel Authentication Records

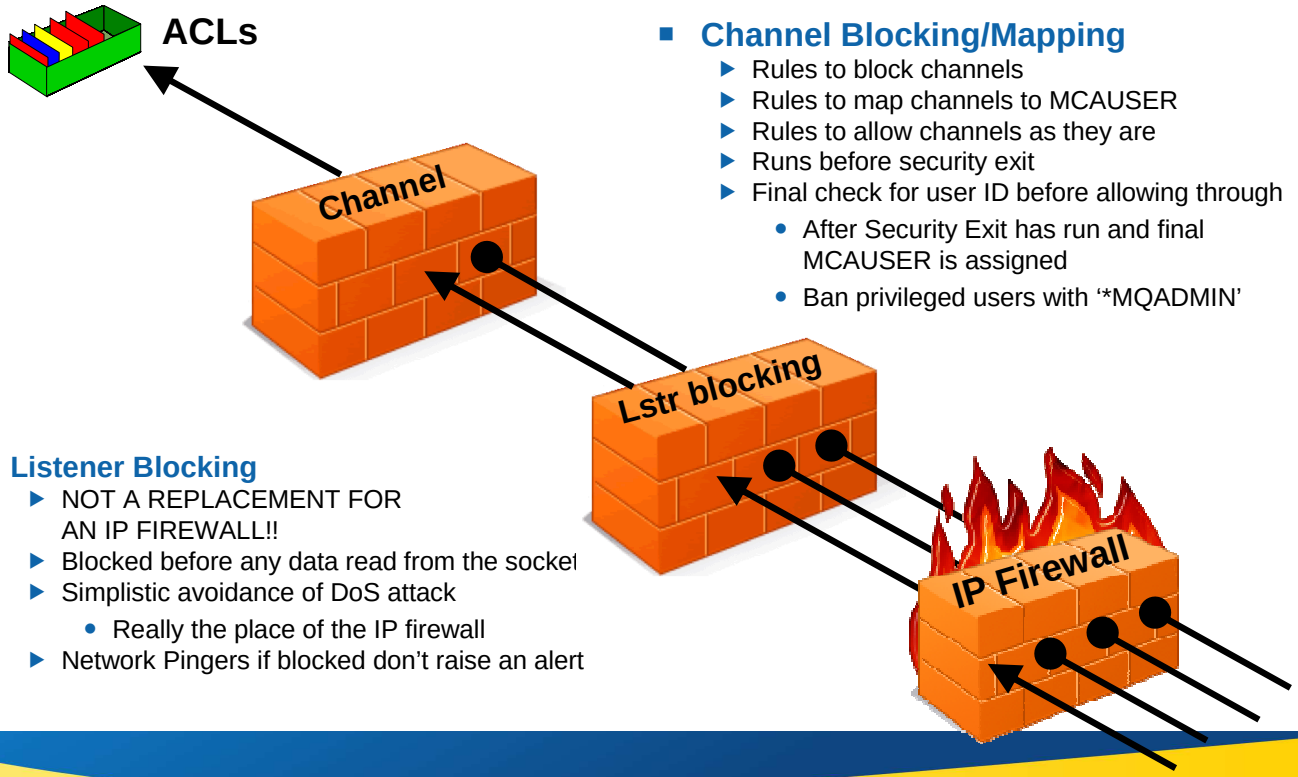
- **Set rules to control how inbound connections are treated**
  - Inbound Clients
  - Inbound QMgr to QMgr channels
  - Other rogue connections causing FDCs
- **Rules can be set to**
  - Allow a connection
  - Allow a connection and assign an MCAUSER
  - Block a connection
  - Ban privileged access
  - 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

Channel Authentication Records – Notes

NOTES

- 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.

Channel Access Blocking Points



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

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

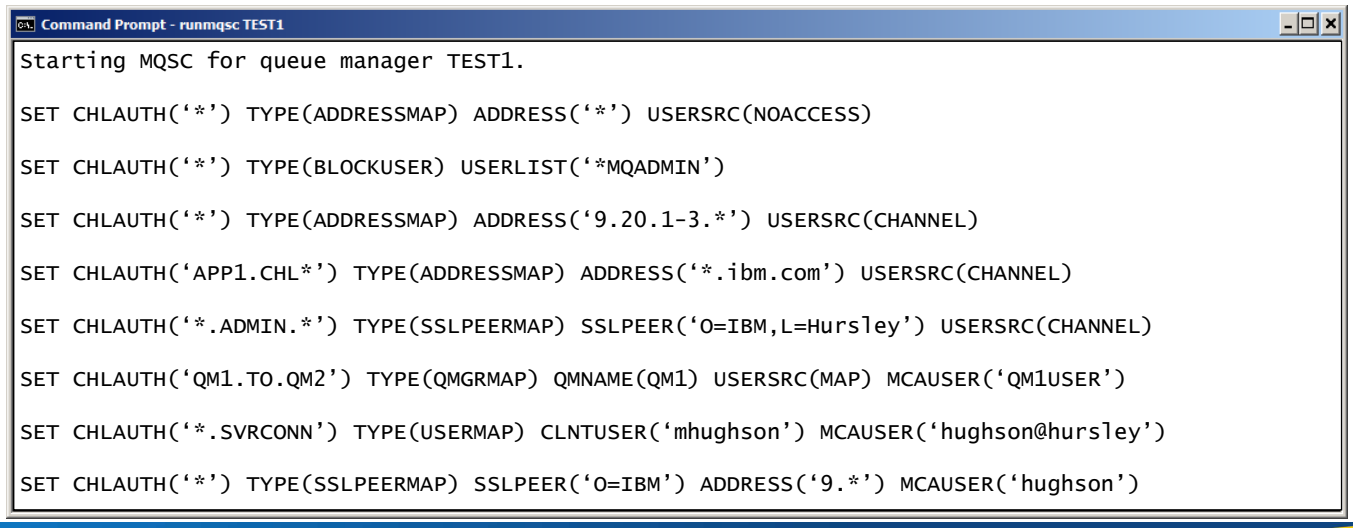
Channel Access Blocking Points – Notes

N O T E S

- In this picture we illustrate that there are a number of points that an inbound connection must get through in order to actually make use of an MQ queue.
- First, we remind you that your IP firewall is included in this set of blocking points and should not be forgotten, and is not superseded by this feature in MQ.
- One point of note, the inbound connections can be from any version of MQ. There is no requirement that the clients or remote queue managers also be on WebSphere MQ V7.1 to be blocked or mapped by these rules.

Channel Authentication Records – 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)



Channel Authentication – 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.

IP Address Pattern Matching

Single Address	9.20.4.6
Wildcard at the end	9.20.*
Wildcard in the middle	9.20.*.6
Ranges	9.20.4.1-10
IPV4 or IPV6	3ffe:1900:4545:3:200:f8ff:fe21:67cf
IPV6 wildcarded	3ffe:1900:4545:3:200:*
IPV4 will also block IPV6 and vice versa	0:0:0:0:0:ffff:192.1.56.10

IP Address Pattern Matching – Notes

N O T E S

- The IP addresses can be specified as single addresses, e.g. 9.20.4.6 or as patterns, e.g. 9.20.* which would of course also match the former. There patterns can also be generic in the middle, not just at the end, e.g. 9.20.*.6; and can provide ranges (rather akin to how you might configure a firewall) e.g. 9.20.4.1-20.
- These patterns of course will also understand IPV6 address, so as another example one might provide 3ffe:1900:4545:3:200:f8ff:fe21:67cf or 3ffe:1900:4545:3:200:* which would also match the specific address. We must also understand that 0:0:0:0:0:ffff:192.1.56.10 is the same as 192.1.56.10 so that the correct refusals are made when IPV6 and IPV4 are both in use.
- Hostnames cannot be specified in the BLOCKADDR list – only IP addresses. They can be used in ADDRESS fields rules though.

Channel Authentication Rules using Hostnames

- **Initial Listener blocking list**
  - Hostnames not allowed

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

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

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

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

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

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

```
SET CHLAUTH('*') TYPE(SSLPEERMAP)
SSLPEER('CN="Morag Hughson"')
ADDRESS('s*.ibm.*') MCAUSER(HUGHSON)
```

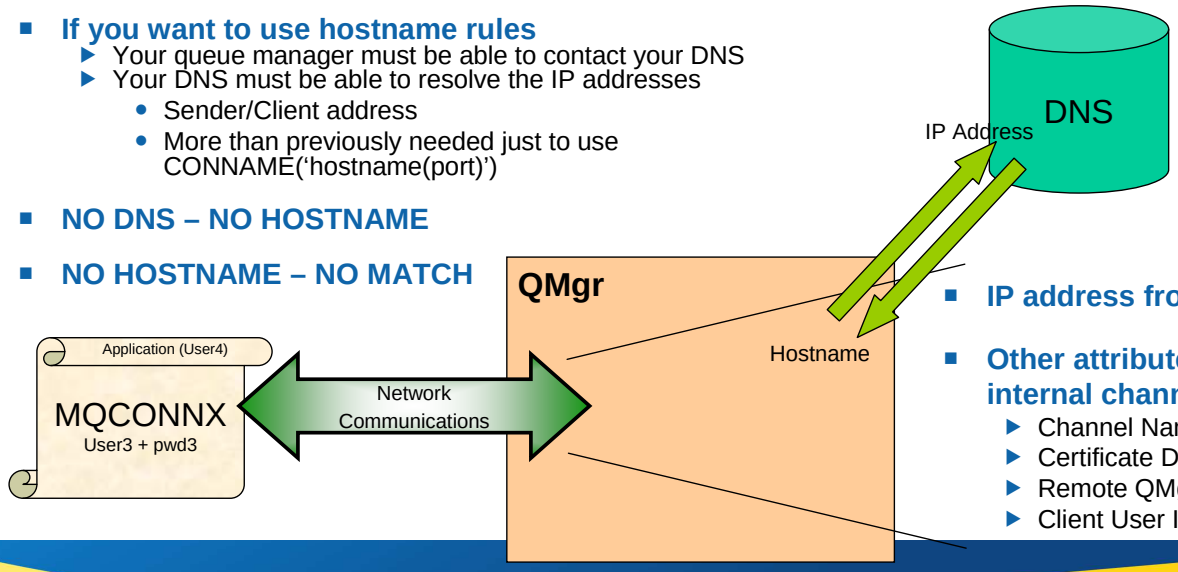

Channel Authentication Rules using Hostnames – Notes

N O T E S

- Hostnames can be used in almost all places in channel authentication records that IP address could be used. The one exception to this is the TYPE(BLOCKADDR) record. This is only going to accept IP addresses.
- If you want to block IP addresses with CHLAUTH rules permanently in MQ, rather than via your IP firewall, you should be doing it using the TYPE(ADDRESSMAP) record and specifying USERSRC(NOACCESS). This type of rules will allow hostnames as well.
- Additionally, positive mapping records allow hostnames, and address restrictors can also use hostnames.
- Channel Authentication rules utilise pattern matching to allow the most flexible control. IP Addresses have a special form of pattern matching that includes ranges and wildcards within each '.' (or ':' for IPv6) section of an IP address. Other pattern matching which is done on channel names, and queue manager names is simpler with just wild-carded string matching (in other words dots are not considered special).
- Hostnames also have pattern matching applied to them – as for channel names and queue manager names. That is it is just a wild-carded string matching and separators such as dots are not considered special.

Obtaining a hostname

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



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

Obtaining a hostname – Notes

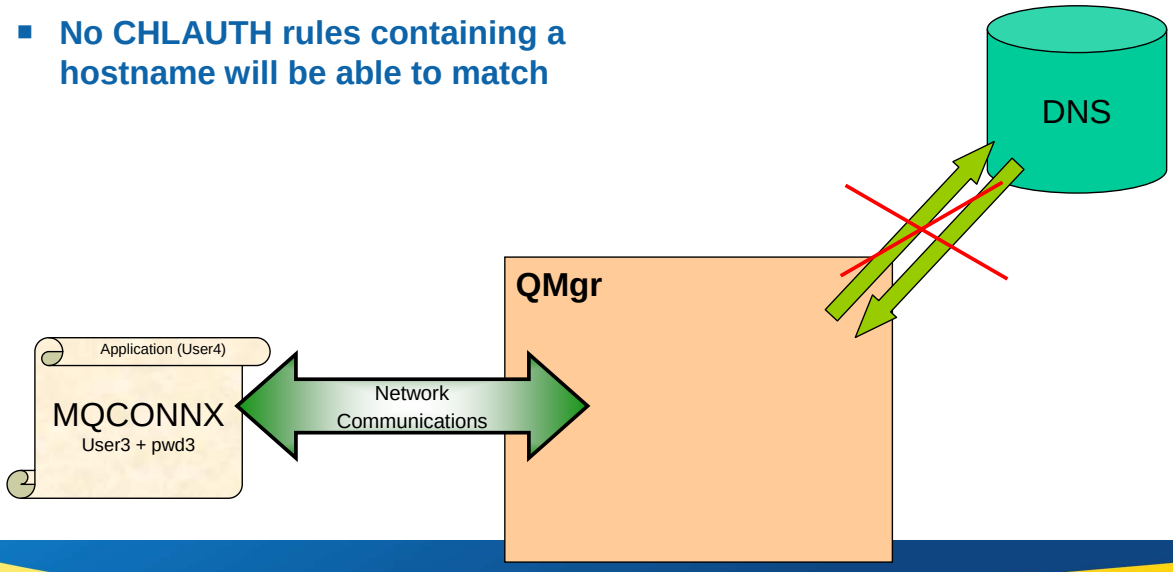
N O T E S

- In order to be able to process channel authentication records that contain rules using hostnames we need to be able to obtain the hostname that represents the IP address of the socket. The hostname is not ‘sent’ to us by the channel or by TCP/IP. We get the IP address from the socket. We get the other attributes that channel authentication records use from the various internal flows across the socket.
- To get the hostname we must ask the Domain Name System (DNS) Server what hostname goes with the IP address we are currently looking at. In order for this to be successful our queue manager must be able to use the DNS. This may already be true if you are using hostnames in CONNAME fields for example – which is certainly common-place. Also, the DNS must be able to reverse look-up the IP address and find a hostname for us. This may not be true in your current set up. Are all the sender channel or client application IP addresses currently available in your DNS? In order for hostname rules to be used, this must be the case.
- If you cannot reverse look up the hostname then CHLAUTH hostname rules will not be able to be matched.

Avoiding obtaining a hostname

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

```
ALTER QMGR REVDNS(DISABLED)
```



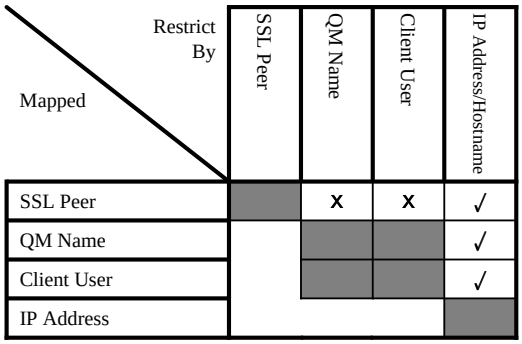
Avoiding obtaining a hostname – Notes

N O T E S

- It is possible that you wish this to always be the case. Some people are more nervous about the potential security hazards of using hostnames than others. When CHLAUTH only used IP addresses to match on, this was not something you had to worry about. Now someone might start to get lazy and use hostname rules.
- We have added a control to turn off the reverse look up of hostnames. There were previously undocumented parameters on both z/OS® and distributed to allow this, but as part of this feature we have made an official version of these.
- When REVDNS is ENABLED, the reverse look-up of the IP Address to retrieve the hostname will still only be done when it is required. If you do not use hostnames in CHLAUTH rules, then the only time a reverse look-up will be done is when writing an error message which contains that information. This is the same as the product behaviour pre-V8.

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




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

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

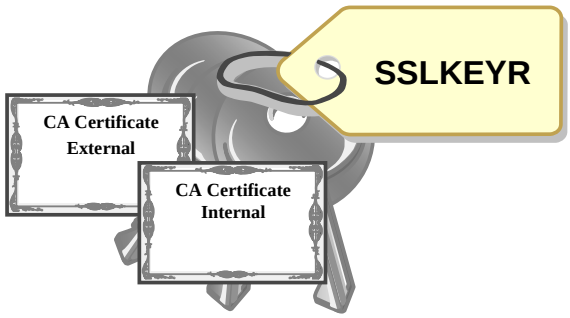

Restricting the Mappings - Notes

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')
```


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 Externals 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).

Channel Authentication Records – Configuration

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

Chl: 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

Channel Authentication – Configuration – 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)

Most Specific Match

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

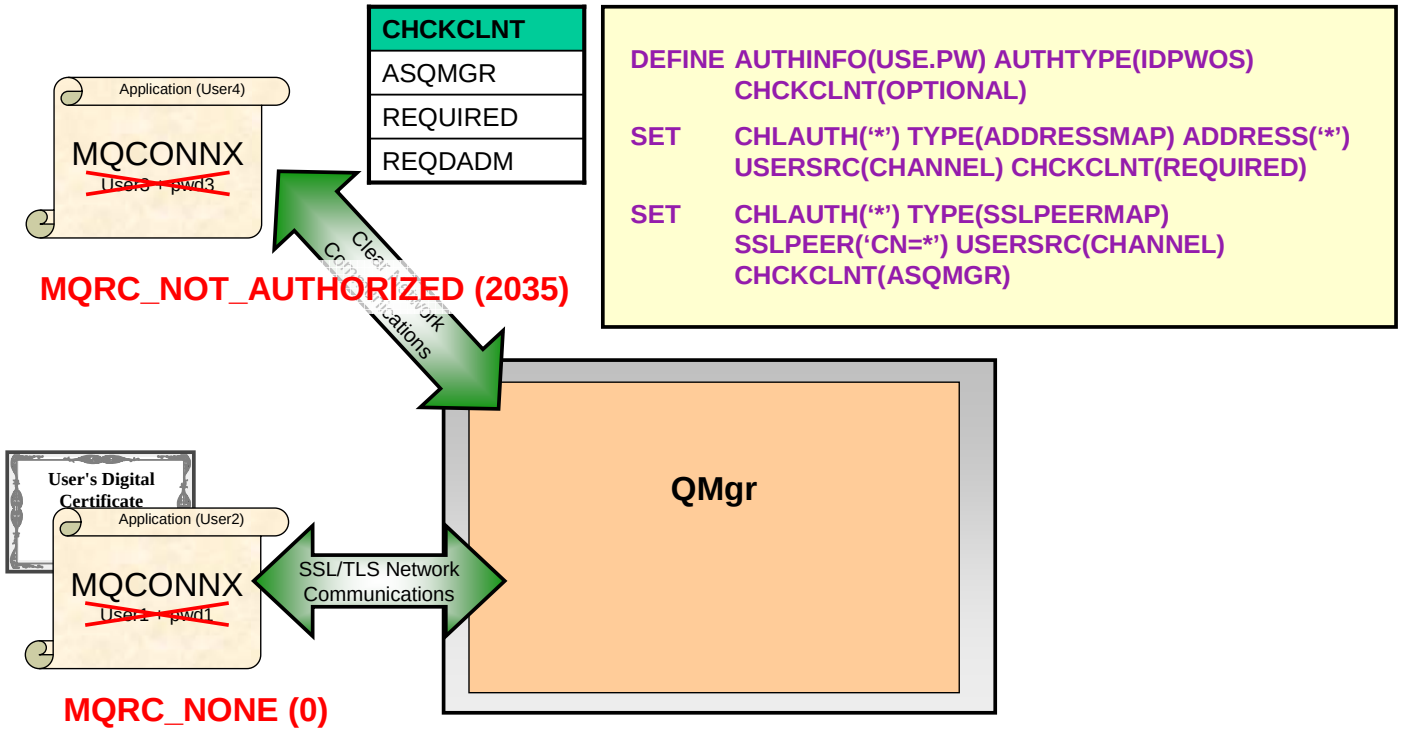
CN=Morag Hughson.OU=MQ Devt. O=IBM UK.L=Hursley.C=UK

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, search the MQ Information Centre for "Distinguished Names".

Connection Authentication – Configuration Granularity



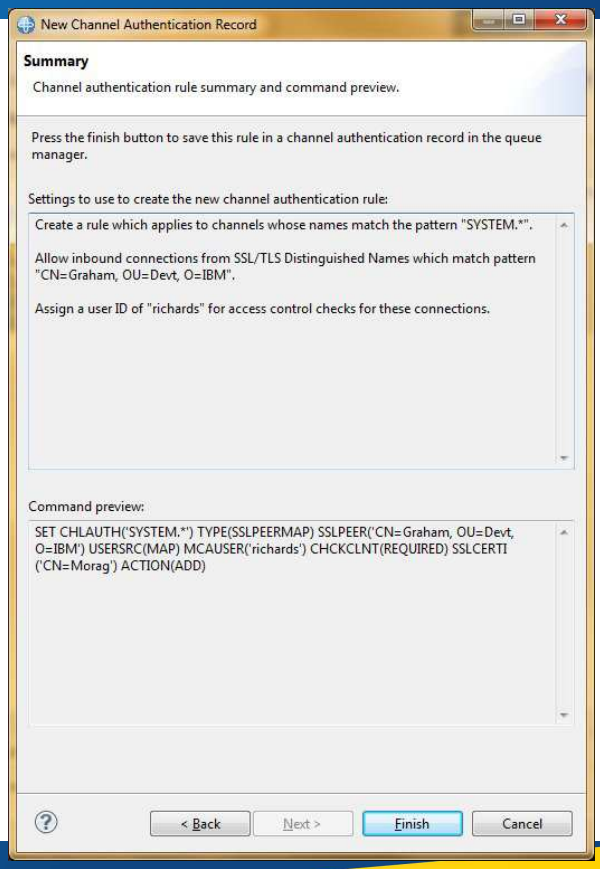
Connection Authentication – Configuration Granularity

N O T E S

- MQ V8 introduces a new feature called Connection Authentication which allows the queue manager to be configured to check user ID and password provided by applications.
- It is clear that a single switch – CHCKCLNT on the AUTHINFO object in use - to configure this for every single client application is not granular enough. So CHLAUTH is enhanced in MQ V8 to provide the ability to mandate password checking for some clients, for example, those not making use of SSL/TLS, and to indicate it is optional for others.
- You can set the overall CHCKCLNT value to OPTIONAL, and then upgrade it to be more stringent for certain channels by setting CHCKCLNT to REQUIRED or REQDADM on the CHLAUTH rule. By default, CHLAUTH rules will run with CHCKCLNT(ASQMGR) so this granularity does not have to be used.

MQ Explorer

- **New concept**
  - Wizard to walk you through the thought-process of creating a rule



Channel Authentication – Configuration – Notes

N O T E S

- Additionally, the MQ Explorer GUI provides a wizard to walk you through the steps for setting up these rules and at the end of the wizard, the MQSC command that would do the same job as you have done in the wizard, is displayed in a window that you can cut'n'paste from to put the command into a script for future use.

Capitalware's MQ Technical Conference v2.0.1.4

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.*') CHCKCLNT(REQUIRED) MCAUSER(ADMUSER)

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

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

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 hostname.
- Read more about this in a blog post here:-
  https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/websphere_mq_chlauth_the_back_stop_rule

Blocking IP address - Which type to use?

```
SET CHLAUTH('*')  TYPE(ADDRESSMAP)
                  ADDRESS('1.2.3.4')
                  USERSRC(NOACCESS)
```

- **Best practice block everything then ALLOW in specific connections**
- **Use ADDRESSMAP to ensure you get good diagnostics**

```
SET CHLAUTH('*')  TYPE(BLOCKADDR)
                  ADDRLIST('1.2.3.4')
```

- **Use BLOCKADDR as a TEMPORARY place to put rogue IP addresses**
  - that really ought to be in your firewall rules.
  - where its something that will soon be fixed
  - This list should be tending to zero!
  - Have a plan for the contents - don't let it grow and grow

Blocking IP address - Which type to use? - Notes

N O T E S

- This pages illustrates two ways to block IP addresses using CHLAUTH. Both of these examples achieve the same end goal, however, each mechanism has a specific purpose.

Using TYPE(ADDRESSMAP)

- This is the main way you should be setting up IP address rules with CHLAUTH. These rules are applied once data has been flowed so the channel name is available, although as the example above shows, you can still make rules to apply to all channels. This is the type you should use for the majority of your IP address blocking rules. When an inbound connection is blocked as a result of one of these rules, the error message that is written to your error log, and the event message that is written to the SYSTEM.ADMIN.CHANNEL.EVENT queue (if you have channel events enabled), will contain full details about the inbound connection that has been blocked. When thinking about creating CHLAUTH rules though, it is still better to think from the perspective of blocking everything and then allowing in specific addresses, as detailed on the previous page.

Using TYPE(BLOCKADDR)

- This type of CHLAUTH rule is applied to inbound connections before they send any data at all. Therefore, we do not know the channel name at this time. This is therefore used to block IP addresses that an banned across the board and are not allowed to connect in at all, over any channel. These blacklisted IP addresses are such that they should be caught by your IP firewall and never even make it as far as the MQ listener. However, it is common for updates to an IP firewall to take time to be put in place, and so using a TYPE(BLOCKADDR) CHLAUTH rule can bridge the gap from the time a rogue IP address is detected, to the time when the IP firewall is updated, at which point the TYPE(BLOCKADDR) rule can be removed again. So in short, this type of CHLAUTH rule is a temporary place that is under the control of the MQ administrator, to blacklist IP addresses until they are more permanently caught by the IP firewall. Keep an eye on your TYPE(BLOCKADDR) rule and ensure that there is always a plan for the IP addresses mentioned there - so that this list is not ever increasing in your MQ configuration.

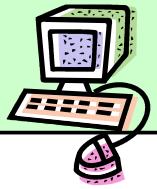
- See blog post:-
  https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/blocking_ip_addresses_with_chlauth_which_type_to_use

What happens if…?

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

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

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



What happens if…? - Notes

N O T E S

- Here is an example of the special matching version of the DISPLAY command to show exactly what would happen should a channel matching these identifying attributes, connect into the system. This should serve as a useful testing tool, service aid, and validation tool, although we would of course recommend not creating such complicated rules that you need it in the first place!
- As we noted earlier, the hostname is not one of those pieces of information, the queue manager has to go and find that information out from the Domain Name System (DNS) Server.
- So when providing information into the MATCH(RUNCHECK) command, you provide the IP address. The queue manager will then make the call to DNS as it would if the real inbound connection appeared and find out what the hostname is, then run the matching against the rules. If it was able to find out a hostname then it will match against a hostname rules, but if it was not, then it won't.
- If you have your queue manager configured to use REVDNS(DISABLED) and you also have some CHLAUTH rules that use hostnames, then a message will appear along with the output of the MATCH(RUNCHECK) display in rather the same way that it warns you that CHLAUTH is DISABLED.
- Thus DISPLAY CHLAUTH MATCH(RUNCHECK) can help you to determine whether your reverse look-up for particular IP addresses is likely to work.

Out of the Box

- **We supply these rules out-of-the-box.**
  - For all channels, ban the assertion of privileged users by inbound channels.
  - For all SYSTEM channels except SYSTEM.ADMIN.SVRCONN (the MQ Explorer GUI channel), ban anyone from using them.

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

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

SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*')
USERSRC(CHANNEL)
```

  - Difficult to supply any default rules regarding IP addresses and SSL Peer Names since they are very installation specific.

- **Enabling Switch ALTER QMGR CHLAUTH(ENABLED|DISABLED) different for Migrated or New Queue Manager**

Out of the Box - Notes

N

O

T

E

S

- Out of the box we supply some rules.
- The first is a rule which bans privileged users and blank users from being asserted by connecting inbound channels. This rule may break some channels, but it will secure many more channels than it breaks so we believe it to be a worthwhile out-of-the-box position.
- The second rules secures the use of SYSTEM channels by disallowing any address from connecting. This stops hackers from connecting in to the SYSTEM.DEF.RECEIVER for example. It also locks down the SYSTEM.DEF.SVRCONN which will hit lots of people initially!
- The third rule allows the SYSTEM.ADMIN.SVRCONN but it will still be affected by the first rule if you try to use a privileged user ID, so some work must be done to provide a user ID that has access to do what is needed.
- There is a queue manager switch which determines whether CHLAUTH rules are acted upon (it does not stop the commands from be used though). This switch is ENABLED for new queue managers, and DISABLED for migrated queue managers.

Privileged Users

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

- **Banned by default**
  - Applies to all channels

```
SET CHLAUTH(MY.SVRCONN)
    TYPE(BLOCKUSER)
    USERLIST('rubbish')

SET CHLAUTH(MY.SVRCONN)
    TYPE(SSLPEERMAP)
    SSLPEER('CN=Admin')
    ADDRESS('1.2.3.4')
    USERSRC(CHANNEL)
    DESCR('SSL auth for admins')
```

- **What if you want to allow some?**
  - Over-ride on one specific channel
  - With authentication
    - SSL/TLS
    - User & Password validation
    - IP address checking (not really authentication!)

Privileged Users - Notes

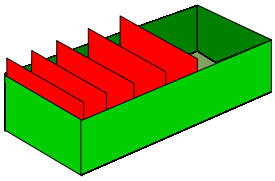
N O T E S

- Users of MQ in the few releases may have noticed a crack-down on remotely connecting privileged users! By a privileged user I mean one that has full access rights, sometimes called an 'mqm' user due to the Unix user of the same name and the group membership which conveys this full access privilege.
- Privileged users are part of the distributed platform MQ world, however although use for running queue managers and making administrative changes locally on the box with the queue manager, we consider them to be a risk when connecting remotely, i.e. with a client connection.
- So in MQ V7.1, remote privileged users were banned by default out of the box with the special *MQADMIN value making it easy to use one different platforms. As an aside, in MQ V8 the Connection Authentication feature also mandates a user ID and password for privileged users as a default stance too. We're serious about this folks!
- You may wish to continue to allow some privileged users to connect is, but only over a certain channel, and only when they have, for example, authenticated in some way, e.g. by a SSL/TLS digital certificate, or at the very least with a password (MQ V8).
- Rather than removing the default '*MQADMIN' rule, you can over-ride it on one specific channel by adding an additional rule to provide an alternate BLOCKUSER list for that channel alone. Unfortunately having an empty BLOCKUSER list does not cause an over-ride, so you need to put something in it. The suggestion is to use a user ID like 'rubbish'.
- See also blog post:- https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/chlauth_allow_some_privileged_admins

Events

- **Command events (as normal)**
- **Configuration events (as normal)**
- **Channel event**
- **Controlled by existing switch**
  - Considered to be an EXCEPTION
- **Written to existing queue**
- **One event for each type of connection refusal**
- **MQRC_CHANNEL_BLOCKED**
  MQRC_CHANNEL_BLOCKED_WARN
  - MQRQ_CHANNEL_BLOCKED_ADDRESS
  - MQRQ_CHANNEL_BLOCKED_USERID
  - MQRQ_CHANNEL_BLOCKED_NOACCESS

ALTER QMGR CHLEV(ENABLED|EXCEPTION)



SYSTEM.ADMIN.CHANNEL.EVENT

Events - Notes

N

O

T

E

S

- These commands will generate command events and configuration events (assuming that these events are enabled by the existing CMDEV and CONFIGEV switches).
- There are some new events to record whenever an inbound connection attempt is blocked. Controlled by the current CHLEV switch (and considered to be an EXCEPTION) this new event message will be issued to the SYSTEM.ADMIN.CHANNEL.EVENT queue when a channel or listener blocks an attempt to connect.
- The reason qualifier of the event message can be
  - MQRQ_CHANNEL_BLOCKED_ADDRESS
    Channel was blocked due to its IP address being in the list to be refused.
  - MQRQ_CHANNEL_BLOCKED_USERID
    Channel was blocked due to its asserted (or mapped) user ID being in the list to be refused.
  - MQRQ_CHANNEL_BLOCKED_NOACCESS
    Channel was blocked due to its identity (e.g. IP address or SSL Peer name) being mapped to a rule that says it is to be blocked.

Updating CHLAUTH rules

Key Fields

Channel Name	Type	Value(s) for type	Address (restrictor)	Source of ID	Mapped ID	User ID & Password
SET CHLAUTH(APP1.*) TYPE(SSLPEERMAP) SSLPEER('CN=Morag') ADDRESS('1.2.3.4')				USERSRC(MAP) MCAUSER('hughson')		
SET CHLAUTH(ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*.ibm.com')				USERSRC(CHANNEL) CHCKCLNT(REQUIRED)		

```
SET    CHLAUTH('APP1.*') TYPE(SSLPEERMAP) SSLPEER('CN=Morag') ADDRESS('1.2.3.4')
       ACTION(REMOVE)

SET    CHLAUTH(ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*.ibm.com')
       USERSRC(MAP) MCAUSER('IBMUSER') ACTION(REPLACE)

SET    CHLAUTH('APP1.*') TYPE(SSLPEERMAP)
       ACTION(REMOVEALL)
```

All the key fields

Updating CHLAUTH rules - Notes

N O T E S

- You can make changes to, or delete, pre-existing CHLAUTH rules. One way to do this is of course via a GUI which allows you to do this easily - click on the thing you want and edit away.
- You can also do this with MQSC commands in a script, and its here that you probably need to be more aware of the key fields in a CHLAUTH rule.
- Clearly there is a channel name pattern that is a key field in the CHLAUTH rule, but as you have no doubt noticed, you can have several rules all using the same channel name. Clearly this field on its own is not enough to identify a specific rule that you wish to edit or delete.
- The type is another, perhaps obvious key field, so this is needed to identify the specific rule that you wish to edit or delete as well. But that is still not enough. You could for example have multiple TYPE(SSLPEERMAP) rules on one specific channel - perhaps assigning different MCAUSERs for each certificate presented.
- You also need some other key fields, and in fact you need everything on the left hand side of the page to uniquely identify the rule you wish to edit or delete. This is also the same set of fields that you cannot duplicate. You cannot, for example, have two rules for the same certifiate DN, mapping to different MCAUSERs. MQ wouldn't know which one to apply, so we disallow you from defining two the same.
- If you want to remove all the rules you have put in place for a particular channel name pattern and TYPE, you can omit the remainder of the key fields when using ACTION(REMOVEALL).

Updating CHLAUTH List rules

ACTION value	Result
ADD	Adds the specified member to the list
REMOVE	Removes the specified member to the list
REMOVE(last member)	Clear out the list - and thus remove the record
REPLACE	You provide the whole list you want
REPLACE(empty list)	Clear out the list - and thus remove the record
REMOVEALL	Clear out the list - and thus remove the record

- For TYPE(BLOCKUSER) and TYPE(BLOCKADDR) rules, i.e. those with lists
  - USERLIST
  - ADDRLIST

```
SET   CHLAUTH('*') TYPE(BLOCKUSER)
      USERLIST('*MQADMIN','Morag')
      ACTION(REPLACE)

SET   CHLAUTH('*') TYPE(BLOCKUSER)
      USERLIST('hughson') ACTION(ADD)

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

SET   CHLAUTH('*') TYPE(BLOCKUSER)
      ACTION(REMOVEALL)
```

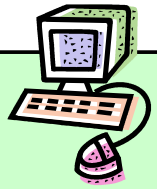

Updating CHLAUTH List rules

N O T E S

- On the previous page we discussed the key fields in the main set of CHLAUTH rules, the MAP types of rules.
- There are also two other CHLAUTH rules types which I'll call the LIST types. BLOCKUSER and BLOCKADDR. This page looks at updating those rules via MQSC scripts.
- We were conscious when creating CHLAUTH list rules that we didn't want to go down the same route as the NAMELIST object in MQ, where to add a name to the list of names, you must provide the whole list. Instead the ACTION field allows you to ADD a single member to the list, REMOVE a single member from the list, REPLACE the list completely (aka the NAMELIST model) and REMOVEALL to clear out the list completely.

Capitalware's MQ Technical Conference v2.0.1.4

Troubleshooting



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

```
DISPLAY    CHLAUTH( 'SYSTEM.DEF.SVRCONN' ) MATCH(RUNCHECK)
           ADDRESS(  '9.180.165.163') CLNTUSER('hughson') ALL

AMQ8878: Display channel authentication record details.
  CHLAUTH(SYSTEM.*)                TYPE(ADDRESSMAP)
  DESCR(Default rule to disable all SYSTEM channels)
  CUSTOM( )                        ADDRESS(*)
  USERSRC(NOACCESS)                WARN(NO)
  ALTDATE(2013-09-03)              ALTTIME(12.20.25)
```

Descriptions are Good!!

Troubleshooting – Notes

N O T E S

- Most people's first experience of Channel Authentication Records is being blocked by them, so very quickly people learn how to issue the command:-
  - ALTER QMGR CHLAUTH(DISABLED)
- However, working out why you have been blocked is not really that difficult – all the information you need is provided, so instead why not add in the rule that allows you in, instead of turning it all off?
- When an inbound connection is blocked, an error is written to the AMQERR01.LOG (or CHINIT joblog on z/OS) indicating that is was blocked and providing additional information describing exactly the inbound connection. As we've just seen, this information is also written to the event queue. You can use this information to work out exactly why it was blocked.
- In WebSphere MQ V8, this message will also now contain the hostname (possibly several) that go with the IP address, assuming that we have been able to find one. The description of the message will indicate that if a hostname is not shown this implies that either REVDNS is DISABLED or that reverse DNS lookup was unable to obtain a hostname for this IP address.
- We saw an example earlier of the DISPLAY CHLAUTH command running in the MATCH(RUNCHECK) mode. We can use this command with the information from the error message (or event message) to determine exactly which rule caused the connection to be blocked.
- This also shows why it is so useful to make use of the description field when putting your rules in place.
- Read more about this in a blog post:-
  https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/blocked_by_chlauth_why

Channel Authentication Records - Recap

- **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

Recap

N O T E S

- We saw this page at the beginning, but we will use it again as a summary. We have learned today how to use this feature which was introduced in WebSphere MQ V7.1 to control how our inbound connections will behave.
- We have seen a number of new features added to CHLAUTH in MQ V8 (these are highlighted in red on this page).