

WebSphere MQ File Transfer Edition: Security and SSL

Chris Harris
WMQFTE L3 Team Lead
IBM Hursley
harriscr@uk.ibm.com



SHARE in Boston

Agenda

- Sandboxes
 - Overview
 - Gotcha's
- Granular access control
 - More about granular access control
 - Overview
 - Gotcha's
- Secure Sockets Layer
 - Overview of SSL
 - SSL and MQ
 - WMQFTE SSL
- References

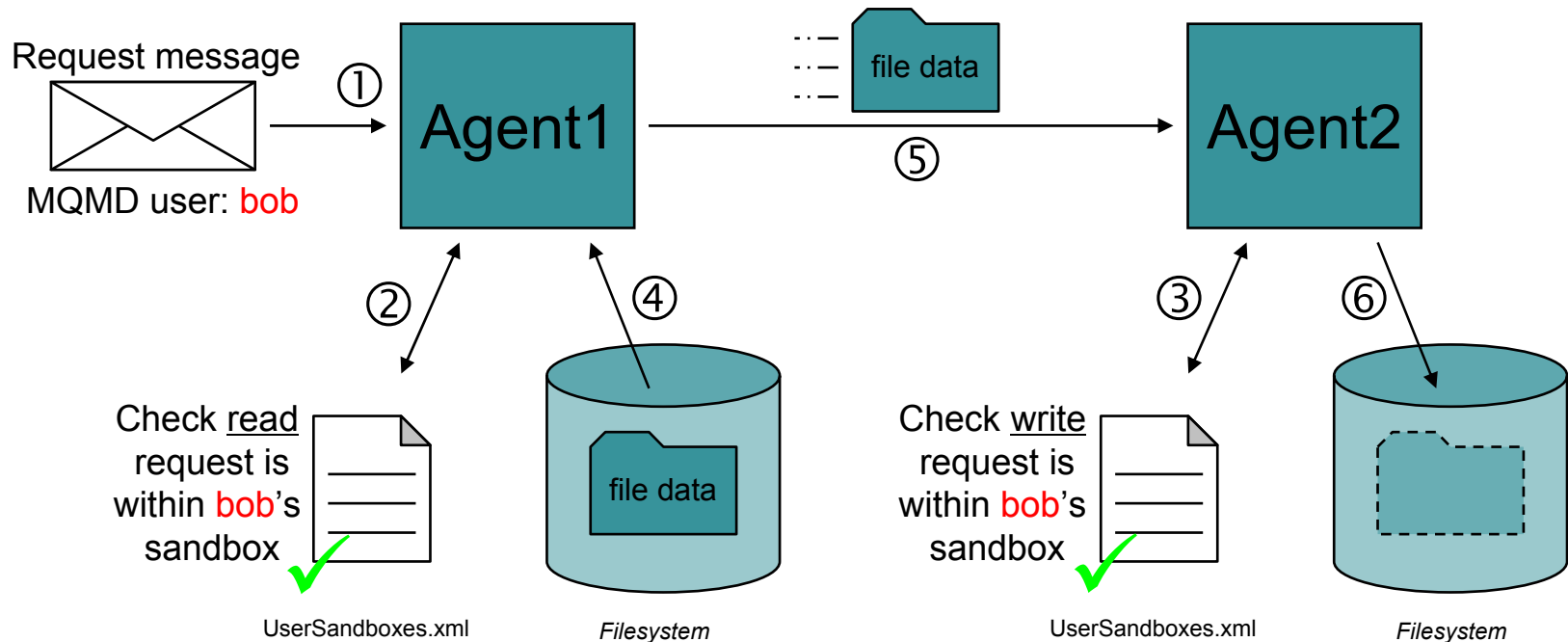


Sandboxes

Overview of sandboxes

- “Sandboxes” is the terminology used to describe the mechanism that File Transfer Edition uses to restrict what areas of the file system can be accessed
- FTE 7.0.0 / FTE 7.0.1 allow all agent file operations to be constrained to be within a ‘sandbox’ (one or more directories). However this imposes the following practical restrictions:
 - It don’t discriminate between read + write operations
 - The settings apply to the agent as a whole – there is no support for applying different settings at a per ‘user’ level.
- FTE 7.0.2 removes these restrictions
 - Read and write operations can be specified separately
 - Settings are at the per user (or collection of user) level

Overview of user sandboxes



- The checks at steps 2 and 3 must succeed otherwise the file transfer will not take place!



NOTES

- User sandboxing occurs in two places:
 - At the source agent, a check is made to ensure that the user requesting to read a file has permission to access the location on the file system
 - At the destination agent, a check is made to ensure that the user requesting to write a file has permission to access the location on the file system
- The MQMD user identifier from the transfer request message is compared against a set of rules which define which areas are readable/writeable for the user
- User sandboxes can be enabled by setting the following property in the agent.properties file:
 - `userSandboxes=true|false` (*default: false*)
- Note that user sandboxing cannot be used with the 'sandboxRoot' property
 - If you specify both 'userSandboxes=true' and the 'sandboxRoot' property - agent will complain and refuse to start!
- User sandboxes are configured via the UserSandboxes.xml document
 - UserSandboxes.xml must be located in the agent's configuration directory
 - This file is created by the 7.0.2 `fteCreateAgent` command
 - The created file contains a significant amount of documentation in the comments!

Example UserSandboxes.xml

```
<tns:userSandboxes ... >
  <tns:agent>
    <tns:sandbox user="account[0-9]" userPattern="regex">
      <tns:read>
        <tns:include name="/home/user/**"/>
        <tns:exclude name="/home/user/private/**"/>
      </tns:read>
      <tns:write>
        <tns:include name="/home/user/output/**"/>
      </tns:write>
    </tns:sandbox>
    <tns:sandbox user="guest">
      <tns:read>
        <tns:include name="/home/user/public/**"/>
      </tns:read>
    </tns:sandbox>
  </tns:agent>
</tns:userSandboxes>
```

Key elements of UserSandboxes.xml

- ‘sandbox’ element
 - Defines a sandbox and specifies a pattern that matches the MQMD user id of requests to the sandbox
 - For example:

```
<tns:sandbox user="a*">  
  <!-- sandbox definition goes here -->  
</tns:sandbox>
```
 - Matches all MQMD user ids starting with a lower case ‘a’
- UserSandboxes.xml can contain multiple ‘sandbox’ elements
 - When multiple ‘sandbox’ elements *could* match an MQMD user ID of a request – the first one that appears in the document “wins”
 - When no ‘sandbox’ elements match the MQMD user ID of a request – the default is to permit no reading or writing of files

Key elements of UserSandboxes.xml

- ‘read’ and ‘write’ elements
 - Child elements of the ‘sandbox’ element
 - Contain ‘include’ and ‘exclude’ elements which define the directories / files / datasets this sandbox can read from or write to

Key elements of UserSandboxes.xml

- ‘include’ and ‘exclude’ elements
 - Define which directories / files / datasets are can be read/written from
 - For example:

```
<tns:include name="/home/user/**"/>
```
 - Rule for what gets included / excluded:
“You can access the file if it matches at least one of the includes, and exactly zero of the excludes”
- Accept wildcards:
 - ‘*’ matches zero or more characters at the directory level
 - ‘?’ matches exactly one character at the directory level
 - ‘**’ matches zero or more directory levels
- Also note that a path name ending with a path separator (e.g. ‘/’) are treated as having an implicit ‘**’
 - ‘/home/user/’ is the same as ‘/home/user/**’



NOTES

- ‘sandbox’ element
 - Defines a sandbox and specifies a pattern that matches the MQMD user id of requests to the sandbox
 - For example:

```
<tns:sandbox user="a*">
  <!-- sandbox definition goes here -->
</tns:sandbox>
```
 - Matches all MQMD user ids starting with an ‘a’ or an ‘A’ (by default matching is not case sensitive)
- UserSandboxes.xml can contain multiple ‘sandbox’ elements
 - When multiple ‘sandbox’ elements *could* match the MQMD user ID of a request – the first one that appears in the document “wins”
 - When no ‘sandbox’ elements match the MQMD user ID of a request – the default is to permit no reading or writing of files
- ‘read’ and ‘write’ elements
 - Child elements of the ‘sandbox’ element
 - Contain ‘include’ and ‘exclude’ elements which define the directories / files / datasets this sandbox can read from or write to
- ‘include’ and ‘exclude’ elements
 - Define which directories / files / datasets are can be read/written from
 - For example:

```
<tns:include name="/home/user/**"/>
```
 - Rule for what gets included / excluded:
“You can access the file / directory if it matches at least one of the includes, and exactly zero of the excludes”
 - Accept wildcards:
 - ‘*’ matches zero or more characters at the directory level
 - ‘?’ matches exactly one character at the directory level
 - ‘**’ matches zero or more directory levels
 - Also note that a path name ending with a path separator (e.g. ‘/’) are treated as having an implicit ‘**’
 - ‘/home/user/’ is the same as ‘/home/user/**’



NOTES

Examples:

```
<tns:include name="/home/user/**"/>
```

```
<tns:exclude name="/home/user/private/**"/>
```

- Includes everything in the /home/user/ directory – or any of its child directories, except things under the /home/user/private/ directory tree

```
<tns:include name="c:\one\**\two\"/>
```

- Includes any directory tree which start c:\one\ and has a directory called 'two' somewhere in it. E.g. c:\one\two\three, c:\one\foo\two\three and c:\one\foo\bar\two\three

```
<tns:include name="/home/user/*.txt"/>
```

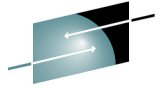
- Includes any .txt files in /home/user

```
<tns:include name="/home/user/**/*.txt"/>
```

- Includes any .txt files in any directory under /home/user (including /home/user)

(Sort-of) Gotcha's

- Reloading the UserSandboxes.xml file
 - The agent will, periodically, re-load the content of the UserSandboxes.xml document, if it changes
 - (It checks for changes to the last modification time, if it has been 30 seconds since the last time it checked)
- What if someone enters malformed data into UserSandboxes.xml?
 - The agent checks the file at startup and will not start if the file is not valid!
 - If a problem is detected at runtime (when the file is reloaded) then the agent will complain, and continue to operate using the previous settings
 - This could lead to invalid file content from stopping the agent from re-starting (*any suggestions on a better behaviour are welcome!*)
- All problems are reported in the agent's output.log file



SHARE
Technology · Connections · Results



Access Control

Access Control: Overview

- WebSphere MQ already provides access control that can be used to prevent unauthorized users from accessing MQ objects (such as queues)
- File Transfer Edition extends this to include authorities that relate to file transfer operations (e.g. should this user be able to transfer files from this system?)

Granular Access Control

Access control to agent capabilities can be broken down into a number of steps:

1. Determine a user's identity
 - (MQMD user ID of request message)
2. Work out what action is being taken
 - (Parse payload of request message)
3. Map what they are trying to do to one (or more) FTE authorities
 - (Simple 'look-up' table in the code)
4. (Optionally) determine the agent's identity
 - (MQMD user ID of messages sent by the agent)
5. Check to see if the identities have the appropriate authorities
 - (Map FTE authority to MQ authority and see if the user is authorized)
6. Permit or deny the action
 - (Either carry on as normal, or fail the request)

FTE User Authorities

Authority	Description
Administration	<ul style="list-style-type: none"> • Shut down the agent • Enable trace on the agent
Transfer source	<ul style="list-style-type: none"> • Start a transfer of files from this agent • Cancel a transfer of files from this agent started by the same user
Transfer destination	<ul style="list-style-type: none"> • Start a transfer of files to this agent • Cancel a transfer of files to this agent started by the same user
Monitor	<ul style="list-style-type: none"> • Create a resource monitor • Delete a resource monitor created by the same user
Monitor operations	<ul style="list-style-type: none"> • Delete a resource monitor created by any user
Schedule	<ul style="list-style-type: none"> • Create a schedule • Delete a schedule created by the same user
Schedule operations	<ul style="list-style-type: none"> • Delete a schedule created by any user or group
Transfer operations	<ul style="list-style-type: none"> • Cancel a transfer created by any user or group

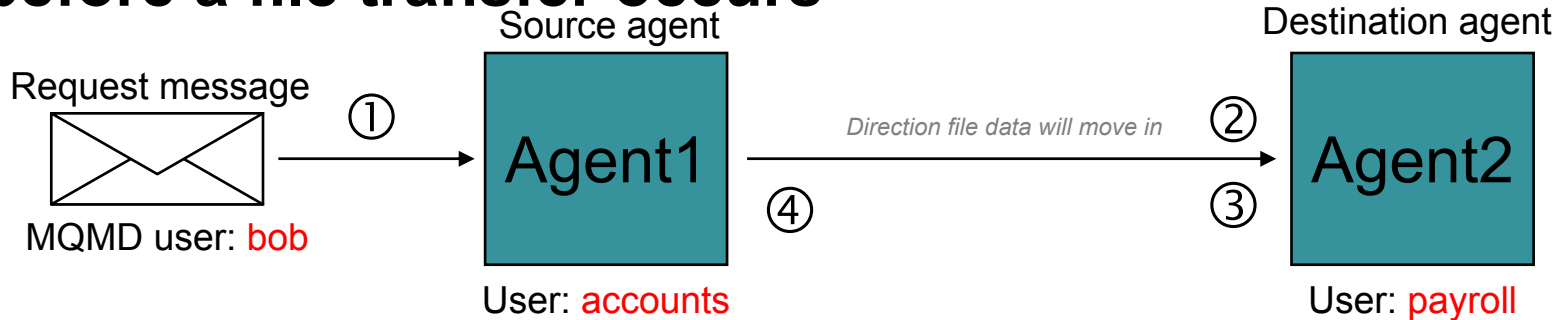
- These apply to the user ID used to submit a request message to the agent

FTE Agent Authorities

Authority	Description
Agent source	• Receive a transfer from <i><source_agent></i>
Agent destination	• Send a transfer to <i><destination_agent></i>

- These authorities apply to the user identifier associated with the agent process
 - They are used to control which agents can communicate with one another
- Confused? Hopefully the next slide should clear things up...

Example of FTE authority checks that take place before a file transfer occurs



Checks that occur before the transfer starts:

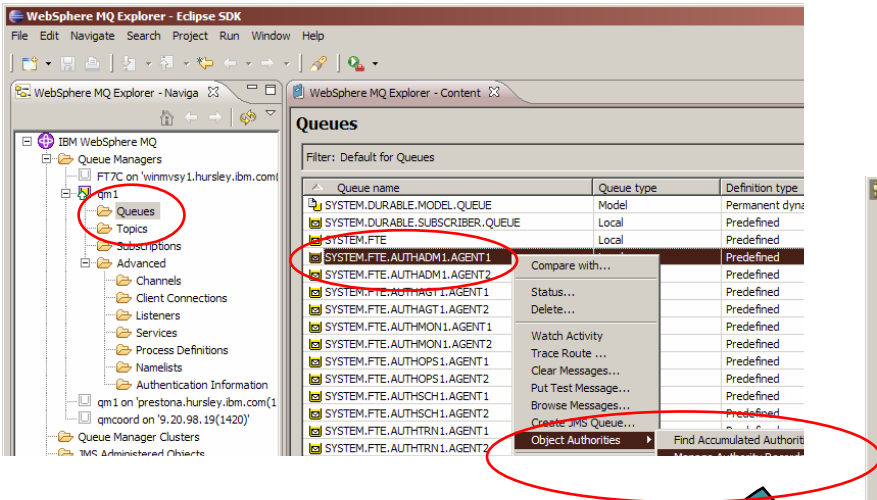
1. Does **'bob'** have 'transfer source' authority?
 - (i.e. can bob move files *off* agent1?)
2. Does **'accounts'** have 'agent source' authority?
 - (i.e. is 'agent2' going to allow 'agent1' to transfer files to it?)
3. Does **'bob'** have 'transfer destination' authority?
 - (i.e. can bob move files *onto* agent2?)
4. Does **'payroll'** have 'agent destination' authority?
 - (i.e. is 'agent1' going to allow 'agent2' to receive files from it?)

Note checks 1+4 happen at the source agent, and 2+3 at the destination agent

Mapping FTE Authorities to MQ Authorities

- So far we have talked about FTE authorities (*like ‘transfer source’ or ‘schedule’*)
 - **but how does an administrator configure these?**
- FTE authorities are mapped to MQ authorities on specific MQ objects
 - E.g. the FTE ‘administration’ authority maps to the MQ ‘browse’ authority on queue ‘SYSTEM.FTE.AUTHADM1.*agentname*’.
 - See the “notes” slide for more details

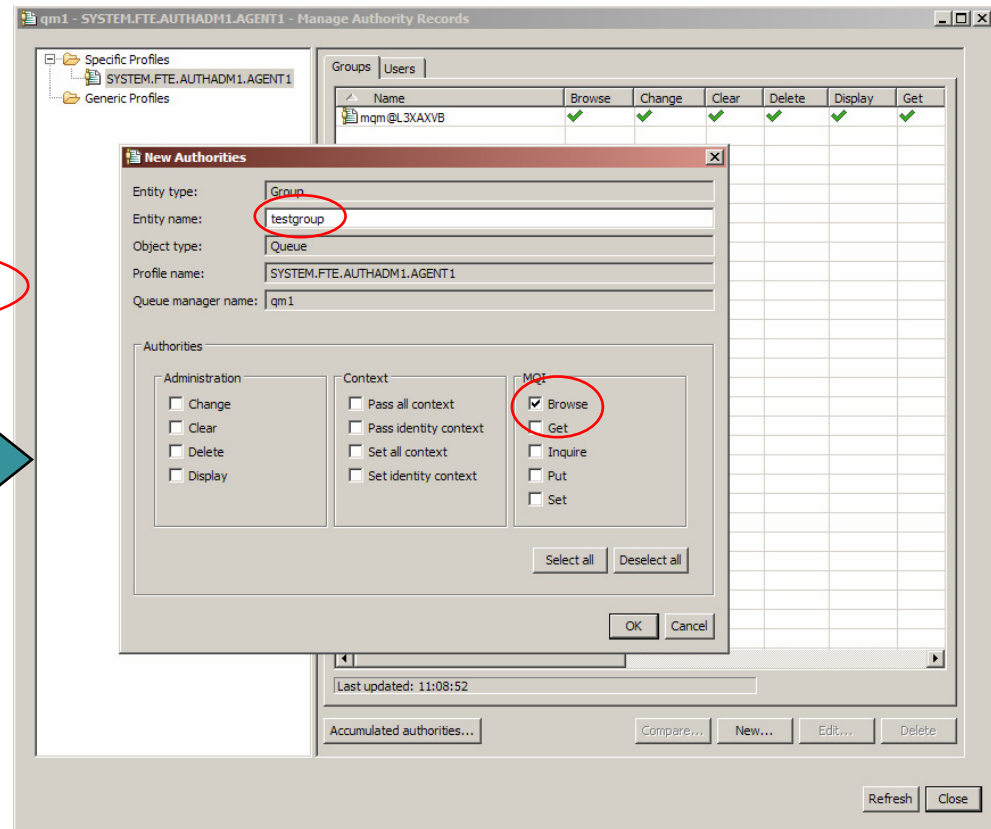
Example: setting 'administration' authority for group 'testgroup' on agent 'agent1' (using MQ Explorer)



WebSphere MQ Explorer - Eclipse SDK

Queues

Queue name	Queue type	Definition type
SYSTEM.DURABLE.MODEL.QUEUE	Model	Permanent dyn
SYSTEM.DURABLE.SUBSCRIBER.QUEUE	Local	Predefined
SYSTEM.FTE	Local	Predefined
SYSTEM.FTE.AUTHADM1.AGENT1		Predefined
SYSTEM.FTE.AUTHADM1.AGENT2		Predefined
SYSTEM.FTE.AUTHAGT1.AGENT1		Predefined
SYSTEM.FTE.AUTHAGT1.AGENT2		Predefined
SYSTEM.FTE.AUTHMON1.AGENT1		Predefined
SYSTEM.FTE.AUTHMON1.AGENT2		Predefined
SYSTEM.FTE.AUTHOPS1.AGENT1		Predefined
SYSTEM.FTE.AUTHOPS1.AGENT2		Predefined
SYSTEM.FTE.AUTHSCH1.AGENT1		Predefined
SYSTEM.FTE.AUTHSCH1.AGENT2		Predefined
SYSTEM.FTE.AUTHTRN1.AGENT1		Predefined
SYSTEM.FTE.AUTHTRN1.AGENT2		Predefined



qm1 - SYSTEM.FTE.AUTHADM1.AGENT1 - Manage Authority Records

Groups | Users

Name	Browse	Change	Clear	Delete	Display	Get
mqm@L3XAXVB	✓	✓	✓	✓	✓	✓

New Authorities

Entity type: Group
Entity name: testgroup
Object type: Queue
Profile name: SYSTEM.FTE.AUTHADM1.AGENT1
Queue manager name: qm1

Authorities

Administration	Context	MQ
<input type="checkbox"/> Change	<input type="checkbox"/> Pass all context	<input checked="" type="checkbox"/> Browse
<input type="checkbox"/> Clear	<input type="checkbox"/> Pass identity context	<input type="checkbox"/> Get
<input type="checkbox"/> Delete	<input type="checkbox"/> Set all context	<input type="checkbox"/> Inquire
<input type="checkbox"/> Display	<input type="checkbox"/> Set identity context	<input type="checkbox"/> Put
		<input type="checkbox"/> Set

Select all Deselect all

OK Cancel

Last updated: 11:08:52

Accumulated authorities... Compare... New... Edit... Delete

Refresh Close

Example: setting 'administration' authority for group 'testgroup' on agent 'agent1' (using the command line)

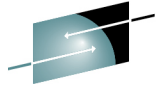


```
setmqaut -m QM1 -t queue -n SYSTEM.FTE.AUTHADM1.agentname -g testgroup +browse
```

- **See:** [http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/topic/com.ibm.mq.amqzmq.doc/fa15980 .htm](http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/topic/com.ibm.mq.amqzmq.doc/fa15980.htm)



NOTES



FTE Authority	MQ Authority Queue	MQ authority <small>SHARE Technology · Collaborations · Results</small>	
		distributed	z/OS
Administration	• SYSTEM.FTE.AUTHADM1.<agent_name>	Browse	Read
Transfer source	• SYSTEM.FTE.AUTHTRN1.<source_agent_name>	Browse	Read
Transfer destination	• SYSTEM.FTE.AUTHTRN1.<destination_agent_name>	Put	Update
Monitor	• SYSTEM.FTE.AUTHMON1.<monitor_agent_name>	Browse	Read
Monitor operations	• SYSTEM.FTE.AUTHOPS1.<agent_name>	Set	Alter
Schedule	• SYSTEM.FTE.AUTHSCH1.<source_agent_name>	Browse	Read
Schedule operations	• SYSTEM.FTE.AUTHOPS1.<agent_name>	Put	Update
Transfer operations	• SYSTEM.FTE.AUTHOPS1.<source_agent_name> • SYSTEM.FTE.AUTHOPS1.<destination_agent_name>	Browse	Read
Receive a transfer from <source_agent>	• SYSTEM.FTE.AUTHAGT1.<source_agent_name>	Browse	Read
Send a transfer to <destination_agent>	• SYSTEM.FTE.AUTHAGT1.<destination_agent_name>	Put	Update



NOTES

- Security controlled via 'agent.properties':
 - authorityChecking=true|false (*default: false*)
 - Enables granular access control function
 - userIdForClientConnect=none|java (*default: none*)
 - Sets the user ID that gets flowed on MQCONN calls to the user name associated with the process (as returned by the JVM). The default is not to flow a user ID
- Agent process must run under a user with the 'ALT_USER' MQ Authority!
 - A new requirement, that applies when 'authorityChecking=true' is set
- Command line '-ac' switch on *fteCreateAgent*
 - Automatically sets:
 - 'authorityChecking=true'
 - 'userIdForClientConnect=java'
- To help problem determination, the agent can be configured to log when authority checks are performed
 - Controlled by 'logAuthorityChecks' property in 'agent.properties':
 - logAuthorityChecks=none|failures|all (default: none)
 - *None = don't log anything*
 - *Failure = only log authority checks that fail*
 - *All = log all authority checks.*
 - Output goes to agent's output.log file.
- Example output:
 - [14/10/2009 14:32:48:437 BST] 0000000c WMQAuthorityC I BFGAG0104I: The authority check for user 'preston' and authority 'SCHEDULE' has mapped to checking the 'BROWSE' authority on queue 'SYSTEM.FTE.AUTHSCH1.AGENT1'.
 - [14/10/2009 14:32:48:437 BST] 0000000c WMQAuthorityC I BFGAG0105I: The authority check for user 'preston' and authority 'SCHEDULE' has passed.

Enabling security

- Security controlled via ‘agent.properties’:
 - `authorityChecking=true|false` (*default: false*)
 - Enables granular access control function
 - `userIdForClientConnect=none|java` (*default: none*)
 - Sets the user ID that gets flowed on MQCONN calls to the user name associated with the process (as returned by the JVM). The default is not to flow a user ID
- Agent process must run under a user with the ‘ALT_USER’ MQ Authority!
 - A new requirement, that applies when ‘`authorityChecking=true`’ is set
- Command line ‘-ac’ switch on `fteCreateAgent`
 - Automatically sets:
 - ‘`authorityChecking=true`’
 - ‘`userIdForClientConnect=java`’

... but it's not working the way I expected!

- To help problem determination, the agent can be configured to log when authority checks are performed
 - Controlled by 'logAuthorityChecks' property in 'agent.properties':
 - logAuthorityChecks=none|failures|all (*default: none*)
 - *None = don't log anything*
 - *Failure = only log authority checks that fail*
 - *All = log all authority checks.*
 - Output goes to agent's output.log file.
- Example output:
 - [14/10/2009 14:32:48:437 BST] 0000000c WMQAuthorityC I BFGAG0104I: The authority check for user 'preston' and authority 'SCHEDULE' has mapped to checking the 'BROWSE' authority on queue 'SYSTEM.FTE.AUTHSCH1.AGENT1'.
 - [14/10/2009 14:32:48:437 BST] 0000000c WMQAuthorityC I BFGAG0105I: The authority check for user 'preston' and authority 'SCHEDULE' has passed.

Gotcha's

- Q: I've upgraded to 7.0.2 and enabled 'authorityChecking=true' – now my agent reports message BFGAG0109A, and doesn't start.
- A: You need to define the MQ objects used for authority checking (see: '*Managing user authorities on actions*' Information Center page). These are automatically created for an agent when using the 7.0.2 `fteCreateAgent` command. However, these do not get created for an existing agent when FTE is upgraded.

Gotcha's

- Q: I've enabled security for agent 'X' but now it won't transfer files to agent 'Y'. My transfer fail with message 'BFGCH0085'.
- A: Agents will only communicate if they have consistent authority checking settings (either both enabled, or both disabled). Ensure that the 'authorityChecking' property of both agents is set appropriately.

Gotcha's

- Q: Every single one of my FTE authority checks fails and tells me I'm not authorized. I thought I'd set all the correct MQ permissions for the users/groups that use FTE.
- A: Check that the user ID that the agent connects using has the MQ 'ALT_USER' authority. If it doesn't all authority checks will fail

Gotcha's

- Q: The process submitting transfer requests is connecting as a client, and all the MQ authority checks are being done against the MQ administrative user and not the user that the client process is running under.
- A: Check that the client process has the 'userIdForClientConnect=java' set (this can be in either the agent.properties, command.properties or coordination.properties file). Otherwise, the process will connect without specifying a user ID. The MQ default in this case is to assume the ID that the queue manager process is running under.

Gotcha's

- Q: I've given user 'X' the authority to schedule transfers, or create monitors. However, when the schedule or monitor 'fires' the associated transfer fails reporting that the user is not authorized.
- A: Have you also given the user the appropriate authorities to perform transfers? To successfully perform scheduled transfers or monitored transfers, a user needs both the schedule/monitor authorities as well as the authorities required to perform the transfer which will "fire" as a result.

Gotcha's

- Q: I've enabled security but some users can still do everything – irrespective of the authorities they have been assigned.
- A: Is the user ID of the requestor a member of group mqm? Are they a Windows Administrator? Are they a Windows Domain Administrator? By default, MQ implicitly authorizes these users to everything.

MQ Gotcha's

- WMQFTE also inherits some general MQ security Gotcha's:
 - Forgetting to refresh the authorization service
 - z/OS and IBM i use upper-case user IDs. Distributed platforms preserve case
 - User IDs are truncated to fit into the 12 character MQMD user field (commonly 'Administrator' -> 'Administrato')
 - MQ expects user IDs to be known to the system the queue manager is running on
 - MQMD user IDs are subject to change by MCA user ID channel settings
 - Not being authorized to use transmission queues
- See the 'references' slide for pointers to general MQ security information



Command Path

The Agent Command Path

- fteAnt gives the ability to execute a program on a remote machine
 - As part of a <call>
 - As part of <presrc>, <postsrc>, <predst> and <postdst>
- The Agent commandPath property restricts the location that files can be executed from
 - By default no commands can be executed
 - List of directories where commands can be executed from



Agent commandPath

- This is a property on an agent that is set in the agent.properties file. It must be set for each agent in your WMQFTE network
- Specify the commandPath agent property as follows:
 - **commandPath=<command_directory_name><separator>...<command_directory_name>**
- Or for z/OS only:
 - **commandPath=<command_directory_name_or_data_set_name_prefix><separator>...<command_directory_name_or_data_set_name_prefix>**
- Unix example You want to run commands that reside in the directories /home/user/cmds1 and /home/user/cmds2
 - **commandPath=/home/user/cmds1:/home/user/cmds2**
- Windows example: You want to run commands that reside in the directories C:\File Transfer\commands and C:\File Transfer\agent commands, set the commandPath agent property as follows:
 - **commandPath=C:\File Transfer\commands;C:\File Transfer\agent commands**
- z/OS: You want to run commands that are in data sets that start with //'USER.CMD1', //CMD2 and are members of a fully qualified PDS named //'USER.CMDS' set the commandPath agent property as follows:
 - **commandPath=/home/user/cmds1:/home/user/cmds2:'//USER.CMD1'://CMD2:'//USER.CMDS'**



Secure Sockets Layer (SSL)

Secure Sockets Layer

- Protocol to allow transmission of secure data over an insecure network
- Combines these techniques
 - Symmetric / Secret Key encryption
 - Asymmetric / Public Key encryption
 - Digital Signature
 - Digital Certificates
- to combat security problems
 - Eavesdropping
 - Encryption techniques
 - Tampering
 - Digital Signature
 - Impersonation
 - Digital Certificates



NOTES

Secure Sockets Layer - Notes



- Secure Sockets Layer (SSL) is an industry-standard protocol that provides a data security layer between application protocols and the communications layer, usually TCP/IP. The SSL protocol was designed by the Netscape Development Corporation, and is widely deployed in both Internet applications and intranet applications. SSL defines methods for data encryption, server authentication, message integrity, and client authentication for a TCP/IP connection. SSL uses public key and symmetric techniques to provide the following security services:
 - **Message privacy**
 - SSL uses a combination of public-key and symmetric key encryption to ensure message privacy. Before exchanging messages, an SSL server and SSL client perform an electronic handshake during which they agree to use a session key and an encryption algorithm. All messages between the client and the server are then encrypted. Encryption ensures that the message remains private even if eavesdroppers intercept it.
 - **Message integrity**
 - SSL uses the combination of a shared secret key and message hash functions. This ensures that nothing changes the content of a message as it travels between client and server.
 - **Mutual authentication**
 - During the initial SSL handshake, the server uses a public-key certificate to convince the client of the server's identity. Optionally, the client may also exchange a public-key certificate with the server to ensure the authenticity of the client.

SSL Terms

Encryption

+

=

CipherSpec

Hash

Function

CipherSpec

+

=

CipherSuite

Authentication/Key

Exchange



NOTES

SSL Terms



- When we set up an SSL session, we can specify what encryption algorithm we wish to use. We can also specify what hash function to use to generate the message digest, or MAC (message authentication code). This combination is called the CipherSpec.
- An SSL session also needs to know what algorithm to use for authentication and key exchange. In the current implementation of SSL that we will be using, the only option for this is RSA.
- The combination of a authentication/key exchange algorithm and the CipherSpec is called a CipherSuite.



CipherSpecs

NOTES

- RC2, from RSA Data Security Inc. This is a block cipher algorithm (that is, it encrypts the data by blocks, 8-byte long) with a key length of 40 bits, 64 bits and 128 bits.
- RC4, from RSA Data Security Inc. This is a stream cipher algorithm (that is, this algorithm operates on each byte of data, as opposed to a block of bytes) with a key length of 40 bits, 64 bits and 128 bits.
- DES, the old US Data Encryption Standard. This is a block cipher algorithm, with 8-byte long blocks and a key length of 56 bits.
- Triple DES is a variation of DES, with a key length of 168 bits
- In order fastest to slowest they are: RC4, DES, RC2, TripleDES.
- AES, Advanced Encryption Standard, the new US standard. This is a block cipher algorithm, with 16-byte long blocks. Key lengths include 128 and 256 bits.
- There are two choices of hash function that can be used in the provided selection of CipherSpecs. These are SHA and MD5. SHA stands for Secure Hash Algorithm and MD5 stands for Message Digest (version) 5. The choice between MD5 and SHA-1 is a trade off between security and performance. The SHA algorithm produces a 160-bit output and the MD5 algorithm only a 128-bit output. This is compared with the fact that the MD5 algorithm is much faster in calculating the message digest.

Combining these techniques

- SSL Handshake
 - Negotiate level of SSL being used
 - Exchange random numbers that are used to build one-time keys
 - Negotiate cryptographic algorithms
 - Authenticate parties



SSL Handshake

NOTES

- The SSL Handshake combines these techniques. It will also negotiate the level of SSL being used, exchange random numbers that are used to build one-time keys, negotiate cryptographic algorithms, and authenticate parties. The steps (at a high-level) are as follows.
 1. The 'Client Hello'
 - Alice sends Bob some random text, she also sends what CipherSpecs and compression methods she can use. Alice is considered the client since she started the handshake.
 2. The 'Server Hello'
 - Bob sends Alice some random text and chooses the CipherSpec to be used, from Alice's list. He will also send over his certificate - the Server Certificate and may optionally request that Alice sends him her certificate - the Client Certificate Request.
 3. Verify Server Certificate
 - Alice will verify Bob's certificate is valid by checking this digital signature made by the CA (see next foil for more details).
 4. Client Key Exchange
 - Alice builds the Secret Key and sends it to Bob to use in a message encrypted with Bob's Public Key. This means that only Alice who invented the Secret Key and Bob who can decrypt the message containing it, know the Secret Key. Alice also sends her certificate (since Bob requested it).
 5. Verify Client Certificate
 - Bob will verify Alice's certificate is valid by checking this digital signature made by the CA
 6. This is now a 'secure line'
 - Bob and Alice can now send Information using agreed Secret Key which is a randomly generated 1-time key just used for this session.

Secret Keys

- Use to encrypt main data pay load
- Subject to brute force attack
- Periodic renegotiation reduces amount of exposed data



NOTES

Secret Key

- We have seen already that the SSL protocol uses both Secret (Symmetric) Keys and Public/Private Key Pairs. The Secret Keys are negotiated as part of the SSL Handshake and are used to encrypt the bulk data that is sent between parties.
- Keys are subject to brute force attacks. With enough time, or enough data, these keys could be cracked by trying every possible combination. As computers get faster, smaller keys get easier to crack and so larger keys are more favourable, and as simpler algorithms are cracked, more complex algorithms are developed - hence the ever extending list of CipherSpecs.
- Renegotiating the secret key used to encrypt the main payload of data periodically means that there is less data that can be exposed if a brute force attack is successful.

Benefits of SSL

- Provides a protocol for the function we need
 - Encryption
 - Message Integrity Checking
 - Authentication
- Supports a range of cryptographic algorithms
- Uses Public/Private Keys
 - No key distribution problem
- Widely accepted in the Internet community
- Subjected to significant testing by the hacker community

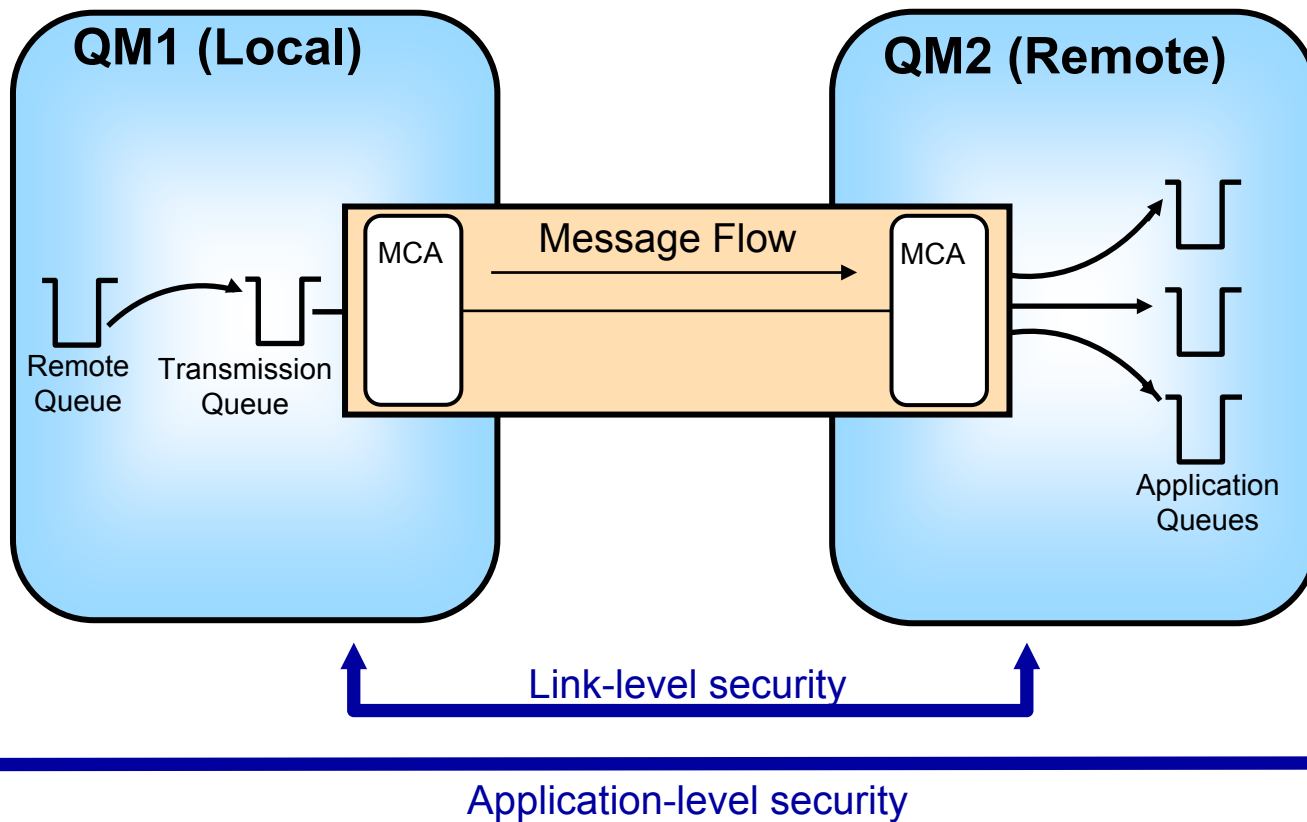


NOTES

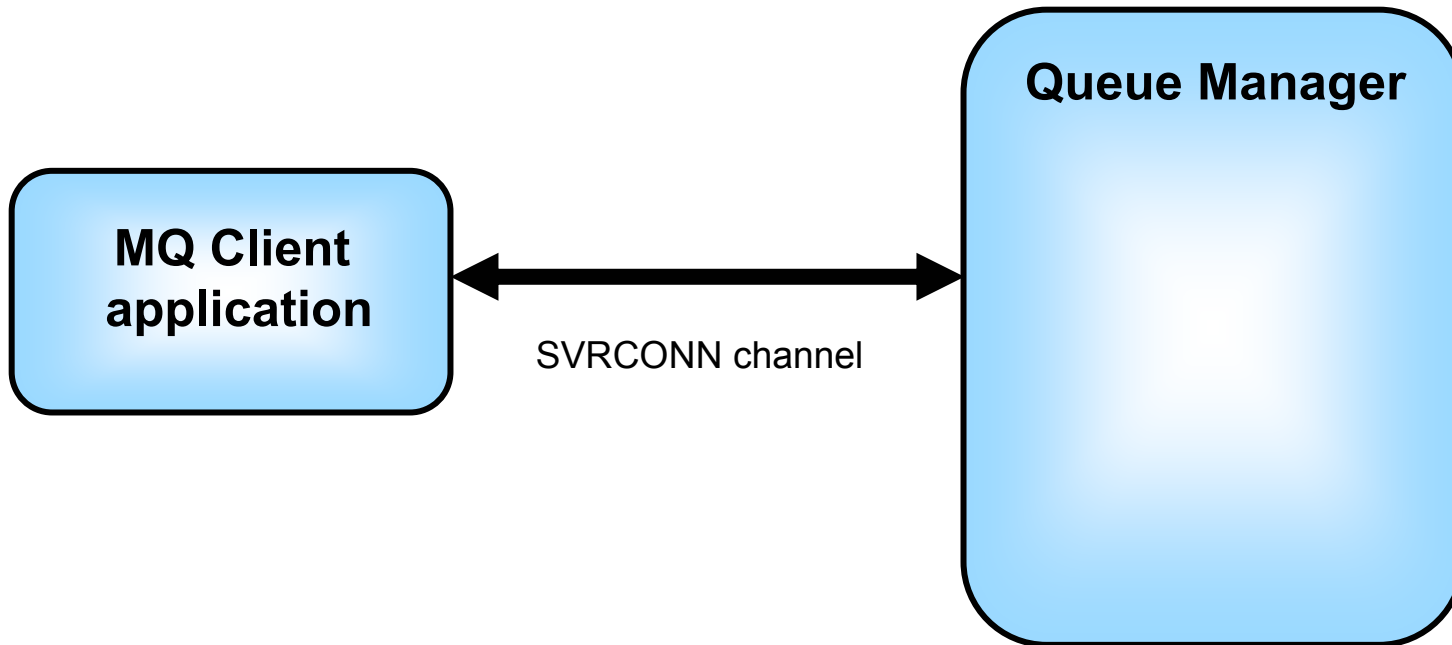
Benefits of SSL

- The Secure Sockets Layer (SSL) provides authentication, message integrity checking and data encryption for messages sent across the Internet. It has become the de facto standard for Internet security and is widely available on different operating systems.
- As we have already seen, it supports a wide range of cryptographic algorithms and makes use of public/private keys for authentication which removes the need for an online key distribution center.

WebSphere MQ and SSL



WebSphere MQ Clients and SSL



- MQ client applications generally connect to their queue manager over a SVRCONN channel
- SSL can be applied to this channel to secure the connection between the client application and the queue manager

SSL functions and WebSphere MQ



- Supported
 - SSL V3.0
 - TLS 1.0 (at WebSphere MQ V6.0)
 - Choice to authenticate client
 - OCSP (at WebSphere MQ V7.0.1 – Distributed platforms)
 - Secret Key renegotiation on a running channel
- Not Supported
 - List of CipherSpecs, only one must be provided
 - SSL session reuse



NOTES

SSL functions and WebSphere MQ



- On many CipherSpecs WebSphere MQ uses the SSL V3.0 protocol. SSL V3.0 was introduced in 1996; lower levels of SSL are no longer widely used. At V6.0, WebSphere MQ also supports some CipherSpecs which use the TLS 1.0 protocol.
- On the SSL handshake an optional feature is to authenticate the client certificate as well as the server certificate. WebSphere MQ offers this choice.
- SSL can allow lots of CipherSpecs to be passed by the client on the SSL handshake and the server will choose one that it supports. WebSphere MQ will impose the restriction that only one CipherSpec can be supplied on the channel definition which must match at both ends.
- SSL can allow its sessions to be reused. This could be useful for short lived web queries. However, since WebSphere MQ channels are likely to be longer running, we will not be using this feature of SSL.

Securing WMQFTE with SSL

- For the purposes of queue manager SSL setup WMQFTE is a standard Java client application
- The agent connects to the coordination, command and agent queue managers in either bindings or client mode.
- When connecting in client mode the connection between the agent and the agent queue manager can be secured using SSL.
- The connection to the coordination and command queue managers can also be secured with SSL.
 - As can the WMQFTE GUI.
- In conjunction with SSL in MQ this can be used to secure the end-to-end transfer of a file through an WMQFTE network



Securing WMQFTE with SSL

NOTES

- The WMQFTE agent can connect to 3 different WMQFTE queue manager roles:
 - Agent queue manager
 - Command queue manager
 - Coordination queue manager
- When it connects to these in client (rather than bindings) mode it does this over an MQ channel. SSL can be enabled on this channel to secure the data sent between the agent and the queue manager.
- To fully secure a WMQFTE network the connections between the MQ queue managers would also have to be secured using SSL.
- For the tasks associated with using SSL with WMQFTE see the information centre

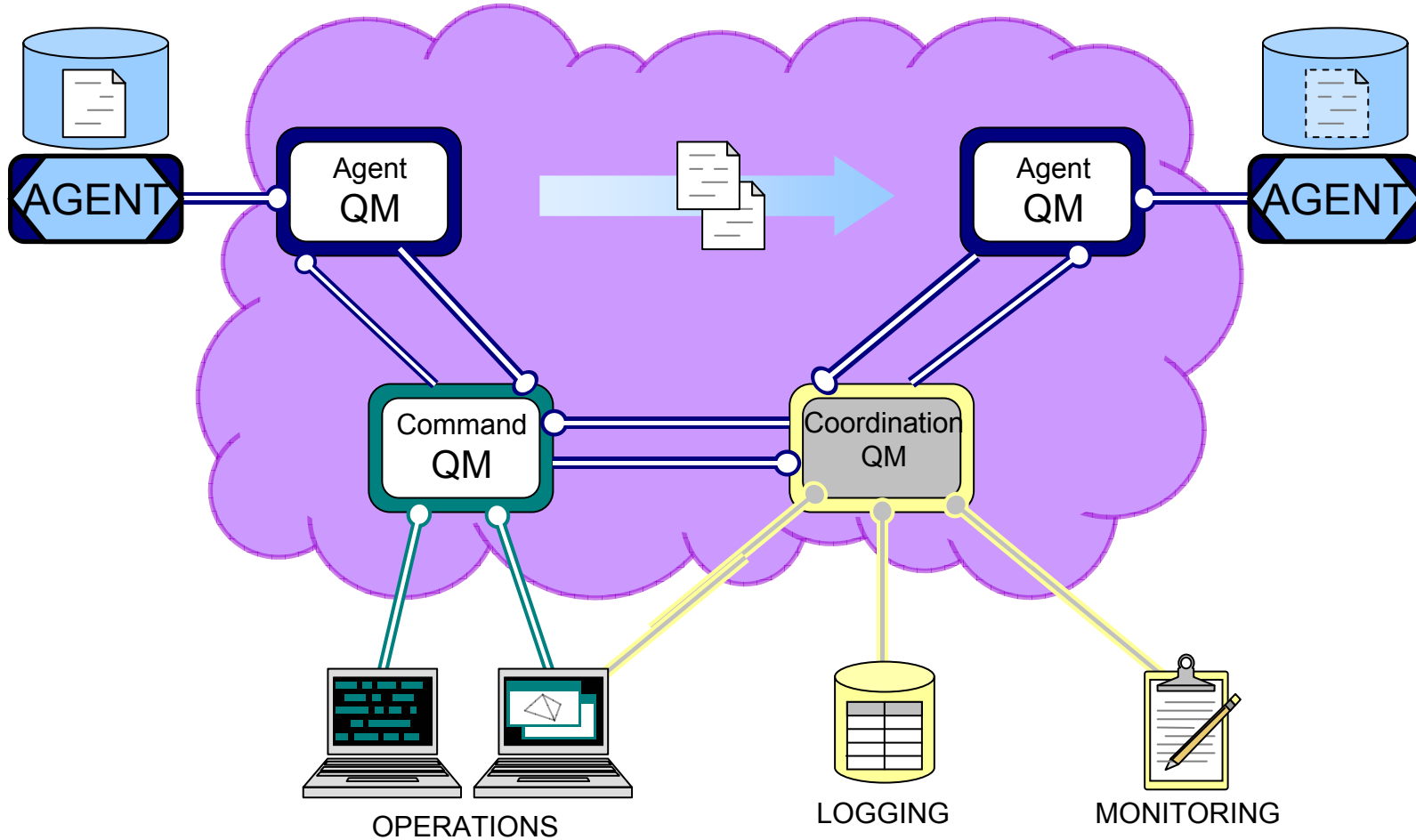
:

<http://publib.boulder.ibm.com/infocenter/wmqfte/v7r0/topic/com.ibm.wmqfte.admin.doc/ssl.htm>

- And the DeveloperWorks article:

http://www.ibm.com/developerworks/websphere/library/techarticles/1001_bonney/1001_bonney.html

Possible SSL connections in WMQFTE



SSL Properties for a WMQFTE agent

- SSL properties for connecting to the agent, command and coordination queue managers are set up in the relevant properties file
 - agent.properties for the agent queue manager
 - coordination.properties for the coordination queue manager
 - command.properties for the command queue manager
- SSL can be enabled for a particular queue manager role
- The settings for the command queue manager and the coordination queue manager are also used by the WMQFTE GUI to connect.
- WMQFTE uses the standard Java keytool that comes as standard with the IBM JVM
- This differs from MQ which can use both GSKit and the Java keytool



NOTES

SSL Properties for a WMQFTE agent



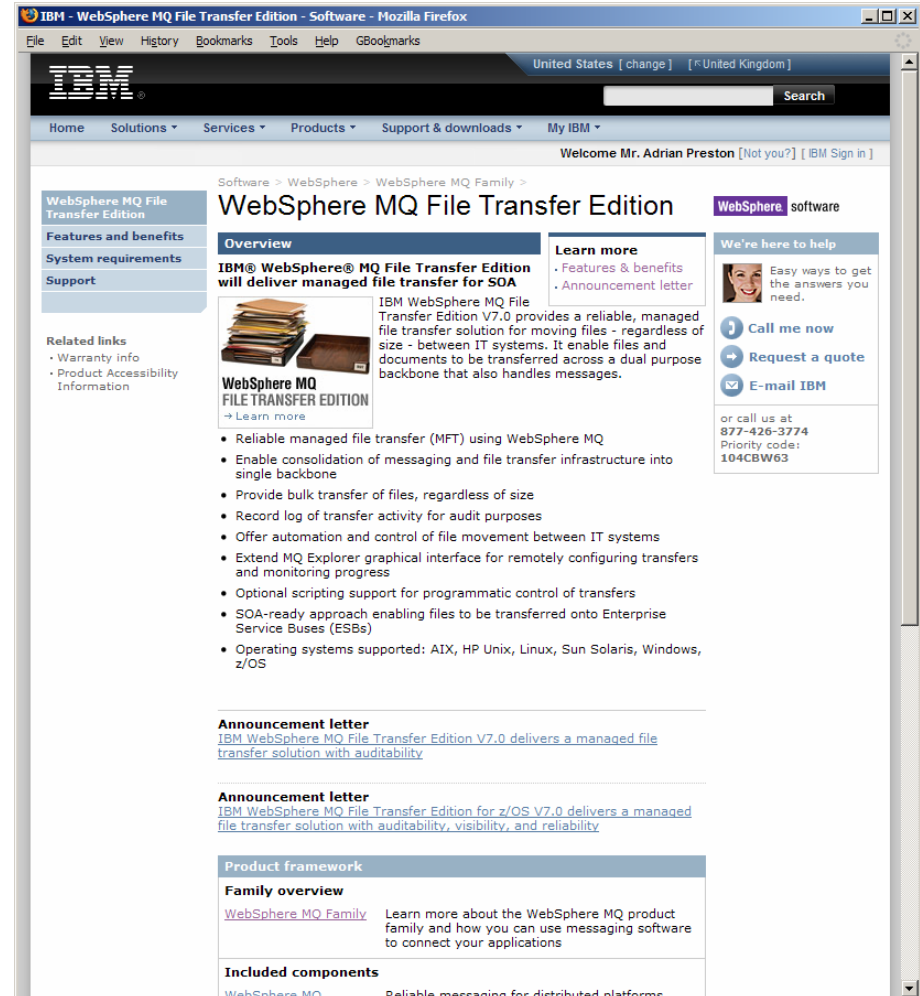
- Each of the 3 queue manager roles within an WMQFTE network has its own property file. It is in these files that the SSL properties for the client connection can be set.
- SSL can be enabled on any combination of queue manager roles
- The passwords in these files are stored in a recoverable format, so restricting who has access to these files a file system level is recommended.
- See the following page for which commands connect to which queue managers
http://publib.boulder.ibm.com/infocenter/wmqfte/v7r0/topic/com.ibm.wmqfte.admin.doc/command_summary.htm

m

Agent.properties file	Coordination.properties file	Command.properties file
agentSslCipherSpec	coordinationSslCipherSpec	connectionSslCipherSpec
agentSslCipherSuite	coordinationSslCipherSuite	connectionSslCipherSuite
agentSslPeerName	coordinationSslPeerName	connectionSslPeerName
agentSslTrustStore	coordinationSslTrustStore	connectionSslTrustStore
agentSslTrustStorePassword	coordinationSslTrustStorePassword	connectionSslTrustStorePassword
agentSslKeyStore	coordinationSslKeyStore	connectionSslKeyStore
agentSslKeyStorePassword	coordinationSslKeyStorePassword	connectionSslKeyStorePassword

Apply to Join the Early Access Program

- **What is it?**
 - Opportunity to get early access to pre-GA code
 - Influence product directions and features
 - Build product knowledge and skills
- **When does it start?**
 - You can apply to join anytime during 2010
- **What is required to join?**
 - A specific Non-Disclosure Agreement (NDA) must be signed
 - Can be signed electronically via the Betaworks Web site
- **What could I receive?**
 - Download regular code drops of the latest pre-GA iterations
 - Opportunity to provide product feedback direct to labs
 - Product roadmap details and materials
- **Who can I contact with more questions?**
 - Contact your local IBM representative



The screenshot shows the IBM WebSphere MQ File Transfer Edition software page. The browser window title is "IBM - WebSphere MQ File Transfer Edition - Software - Mozilla Firefox". The page features the IBM logo, navigation menus (Home, Solutions, Services, Products, Support & downloads, My IBM), and a search bar. The main content area is titled "WebSphere MQ File Transfer Edition" and includes an "Overview" section with the heading "IBM® WebSphere® MQ File Transfer Edition will deliver managed file transfer for SOA". Below this, there is a list of features and benefits, such as "Reliable managed file transfer (MFT) using WebSphere MQ" and "Enable consolidation of messaging and file transfer infrastructure into single backbone". There are also sections for "Announcement letter" and "Product framework".

Resources

- Information Center:
 - <http://publib.boulder.ibm.com/infocenter/wmqfte/v7r0/index.jsp>
- MQ Information Centre (see 'Security' section)
 - <http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp>
- WebSphere MQ Security Presentation Series
 - <http://ausgsa.ibm.com/~trwyatt/public/wmqsecurityseries/>
- Redbooks / Redguides / Redpapers:
 - Getting Started with WebSphere MQ File Transfer Edition V7
 - <http://www.redbooks.ibm.com/abstracts/sg247760.html>
 - IBM WebSphere MQ File Transfer Edition Solution Overview
 - <http://www.redbooks.ibm.com/abstracts/redp4532.html>
 - Managed File Transfer for SOA using IBM WebSphere MQ File Transfer Edition
 - <http://www.redbooks.ibm.com/abstracts/redp4533.html>
 - B2B Enabled Managed File Transfer using WebSphere DataPower B2B Appliance XB60 and WebSphere MQ File Transfer Edition
 - <http://www.redbooks.ibm.com/abstracts/redp4603.html>
- Trial Download:
 - <http://www.ibm.com/software/integration/wmq/filetransfer/>