



IBM Integration Bus

MQ SSL: Connecting MQ nodes to an SSL enabled queue manager

Featuring:

- Running MQ applications on IIB Nodes without an associated queue manager
- Configuring a queue manager to use SSL
- IIB Flow Exerciser testing
- Connecting to queues defined on an SSL enabled queue manager

June 2015

Hands-on lab built at product
Version 10.0.0.0

1. INTRODUCTION	3
1.1 THE MQ BASED APPLICATIONS	4
1.1.1 <i>The (MQ) Provider application</i>	4
1.1.2 <i>The (MQ) Client application</i>	5
2. RESET THE IIB NODES	6
3. CREATE MQ ENDPOINT POLICES	7
3.1 CHECK THE MQENDPOINT POLICIES	8
3.1.1 <i>Check QM4SSLConsumer</i>	8
3.1.2 <i>Check QM4SSLProvider</i>	9
4. IMPORT THE MQ APPLICATIONS	11
5. TEST THE APPLICATIONS (NO SSL CONFIGURATION)	12
5.1 START THE FLOW EXERCISER (MQ PROVIDER)	12
5.2 TEST THE MQ CONSUMER APPLICATION	13
5.3 VERIFY THE TWO APPLICATIONS HAVE WORKED CORRECTLY	15
6. SCENARIO: SSL CONFIGURED ONLY ON MQ SIDE	16
6.1 CONFIGURE SSL ON QM4SSL	16
6.2 CONFIGURE SSL ON CHANNEL TOQM4SSL	19
6.3 RETEST THE MQ APPLICATIONS	20
7. SCENARIO: SSL CONFIGURED ON IIB AND MQ	21
7.1 THE (MQCLIENT) KDB KEY STORE(S)	21
7.2 MQ CLIENT BROKERREGISTRY PROPERTIES	22
7.2.1 <i>Set the mqKeyRepository property</i>	22
7.3 MODIFY THE MQENDPOINT POLICIES	23
7.3.1 <i>Modify QM4SSLConsumer</i>	23
7.3.2 <i>Modify QM4SSLProvider</i>	24
7.4 RETEST THE MQ APPLICATIONS	25
8. USING SECURITY IDENTITIES	26
8.1 DEFINING SECURITY IDENTITIES TO AN IIB NODE	26
8.2 SCENARIO: AUTHENTICATION CHECK: FAILED	28
8.2.1 <i>Modify QM4SSLConsumer</i>	28
8.2.2 <i>Retest the MQ applications</i>	28
8.2.3 <i>Verify the Security Identity passed to MQ</i>	29
8.3 SCENARIO: AUTHENTICATION CHECK: PASSED	31
8.3.1 <i>Modify QM4SSLConsumer</i>	31
8.3.2 <i>Modify QM4SSLProvider</i>	32
8.3.3 <i>Retest the MQ applications</i>	32
END OF LAB GUIDE	33

1. Introduction

This lab guide covers how to configure MQ applications to connect to remote SSL enabled queue managers using MQEndpoint policies.

The MQ applications used in the MQ Topology lab guide are also used in this lab guide. The guide will cover:

- a) Configuring SSL on a queue manager
- b) SSL Cipher spec configuration on a MQ client server Connection definition
- c) Error situations when only the MQ side is configured to use SSL
- d) How to configure IIB to ensure the MQ client can successfully communicate with an SSL enabled queue manager
- e) Using Security Identities

1.1 The MQ based applications

(Note: these are the same application used in the Lab Guide “MQ flexible Topology using MQ Connection Properties”, the explanation of how these applications work is duplicated here for consistency).

Similar to the Web Services based JSON client and EmployeeService provided in other guides throughout this workshop, a Client and Provider application is supplied.

The Provider application obtains requests and provides responses using queues. The client is driven by providing JSON data to a URL controlled by an HTTP input node and provides a JSON based response over http.

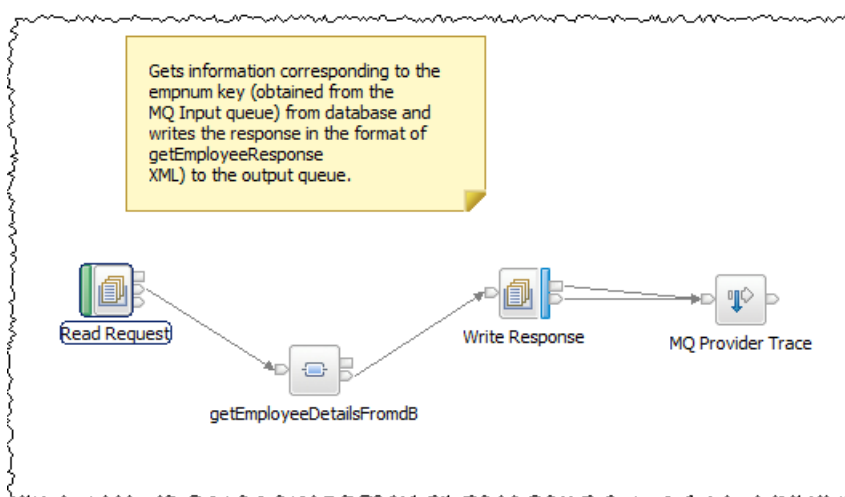
The following sections describe these applications in more detail.

NOTE: The applications are provided purely to show (within the context of this workshop).

- 1) The flexibility of MQ in IIB V10 and
- 2) The reuse of Maps stored in a Shared Library

The applications only work as expected when one user is submitting requests in a controlled way. A more complex Request/Response message correlation Pattern is available in the Patterns gallery, however is out of the scope of this lab guide.

1.1.1 The (MQ) Provider application

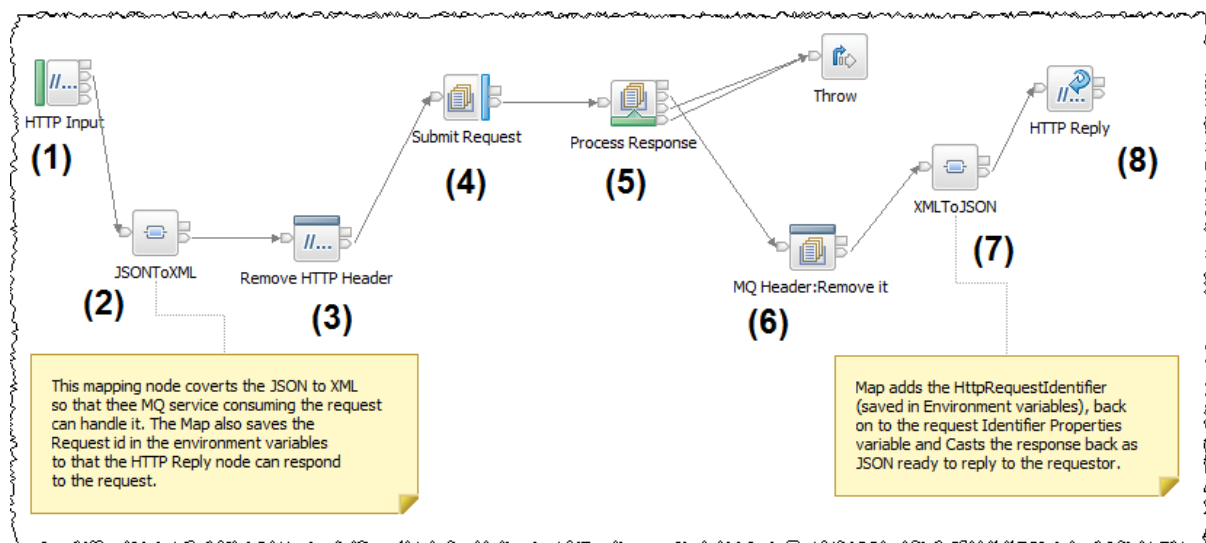


The function of the EmployeeMQProvider application is to retrieve Employee details from a DB2 database.

It will use two queues (MQREQUEST and MQRESPONSE) to handle requests and provide the responses. Request data from MQREQUEST queue is passed to a mapping node. The Mapping node uses XML data passed in the request to obtain details of the employee from the EMPLOYEE table in the database and provides XML Response data. The Response data is then written to a queue called MQRESPONSE.

The Mapping node is supplied in a Shared Library. The lab guide will demonstrate how to reuse assets previously created and stored in a Shared Library in V10.

1.1.2 The (MQ) Client application



The `EmployeeService_JSON_MQClient` contains a message flow that:

- 1) Accepts a JSON request from an HTTP Input node
- 2) Converts the JSON to the required XML format for the `EmployeeMQProvider` to process the request using a mapping node. Note this node also demonstrates a new Graphical Data mapping feature in IIB V10 where it is possible to address the IIB Environment tree in the mapping node. The map saves the HTTP Request ID in Environment variables so that the HTTP Reply node works correctly after removing the HTTP headers from the Message tree in the scenario.
- 3) Removes the HTTP Headers
- 4) Writes the XML version of the request to the `MQREQUEST` queue
- 5) Waits for XML response data to appear on the `MQRESPONSE` queue
- 6) Removes the MQ headers from the Message Tree
- 7) Uses a second mapping node:
 - a. Transforms the XML provided through the `MQRESPONSE` queue back to JSON
 - b. Reinstates the data saved HTTP Request ID from the Environment Variables into the message tree so that the HTTP reply node can work correctly.
- 8) Provides the Response data back to the requestor as JSON data.

2. Reset the IIB nodes

1. Stop IB10NODE (right click on the node in the Integration Toolkit and click stop).
2. If the IB10NODE_CMQ and IB10NODE_PMQ nodes are not defined on your system (**note there is no need to do this if you have the nodes already defined**), open an Integration Bus Console and Navigate to:

```
C:\student10\MQ_SSL\commands
```

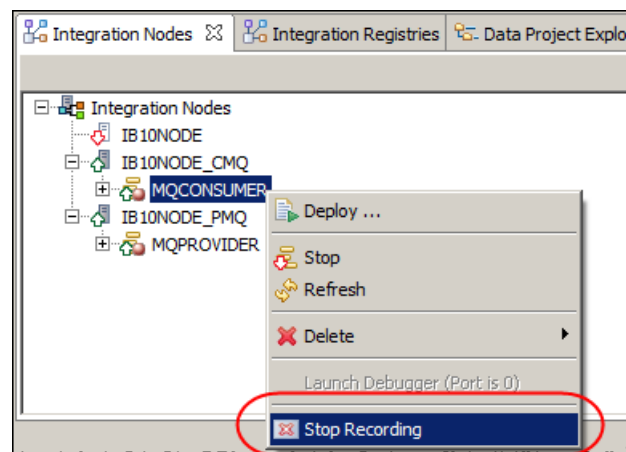
Run the command:

```
00CreateMQIIBNodes.cmd
```

(This command file will create the two IIB nodes and corresponding integration servers).

3. If the servers do exist in your environment, check that the Flow Exerciser recording is not active on the server.

If you see a red circle on the Integration Server, right click on it and click "Stop Recording":



Stop recording on both MQCONSUMER and MQPROVIDER Integration Servers.

When stopped the red circle will disappear.

3. Create MQ Endpoint Policies

The MQ applications you will use in this lab guide are configured to use MQEndpoint policies. In this section you will run a script that will create two policies:

- 1) **QM4SSLConsumer**: this policy is defined on IB10NODE_PMQ and used by the MQ provider application.
 - 2) **QM4SSLProvider**: this policy is defined on IB10NODE_CMQ and used by the MQ consumer application.
1. In an IIB Administration Console navigate to **c:\student10\MQ_SSL\Commands** and run the following command:

```
02Setup_QM4SSL_MQEP.cmd
```

Accept the defaults for the command when prompted.

The script will attempt to delete both policies before they are created, you may see error messages for attempted `mqsDeletePolicy` command however you can safely ignore the errors.

The response when creating the MQEndpoint policies should be:

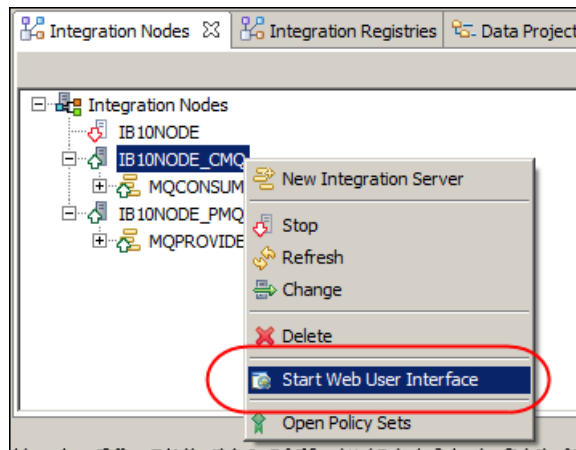
```
BIP8071I: Successful command completion.
```

3.1 Check the MQEndpoint Policies

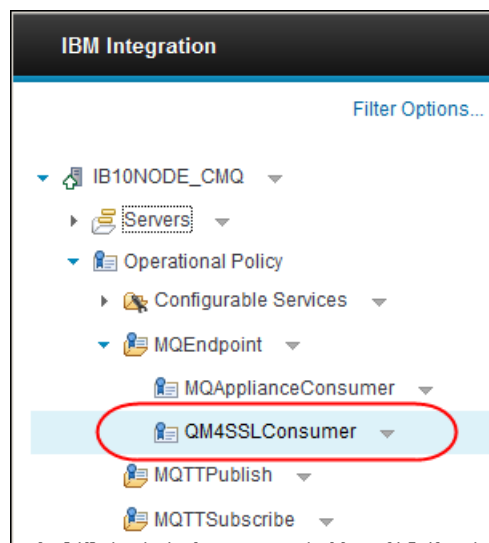
In this section you check the MQEndpoint Policies using the Web UI.

3.1.1 Check QM4SSLConsumer

1. In the “Integration Nodes” view in Toolkit, right click on the IB10NODE_CMQ node and Start the Web User Interface:

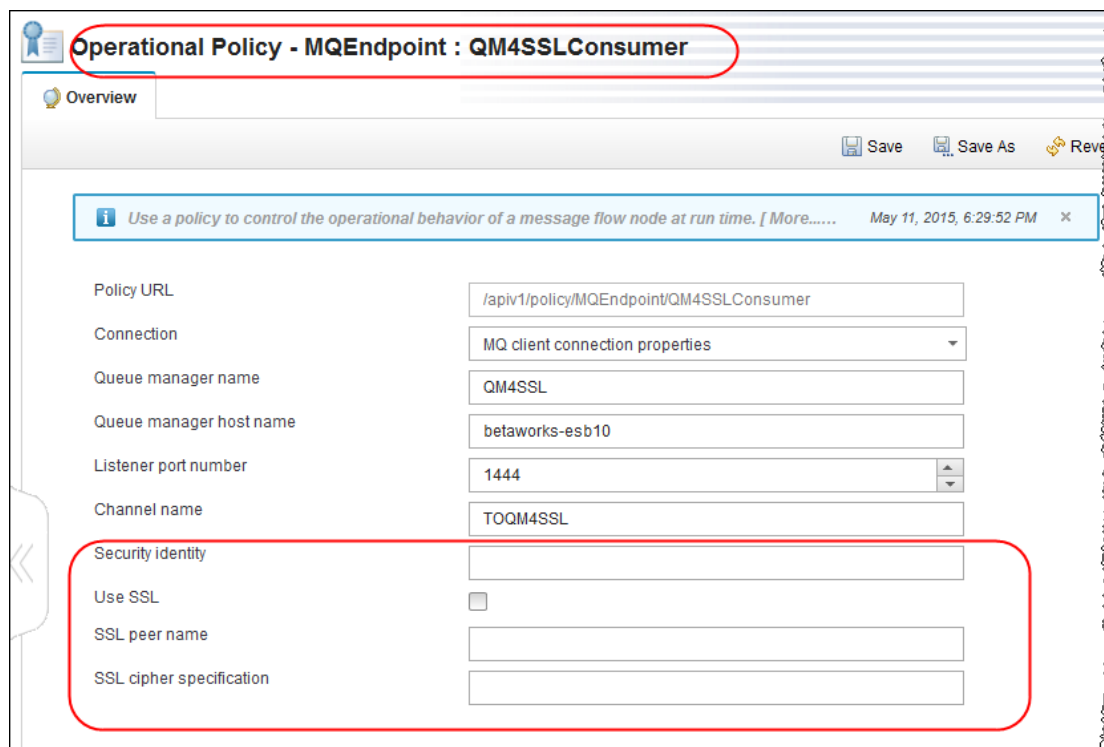


2. In the Web UI, expand **IB10NODE_CMQ > Operational Policy > MQEndpoint**



Click “**QM4SSLConsumer**” to show the MQEndpoint policy details.

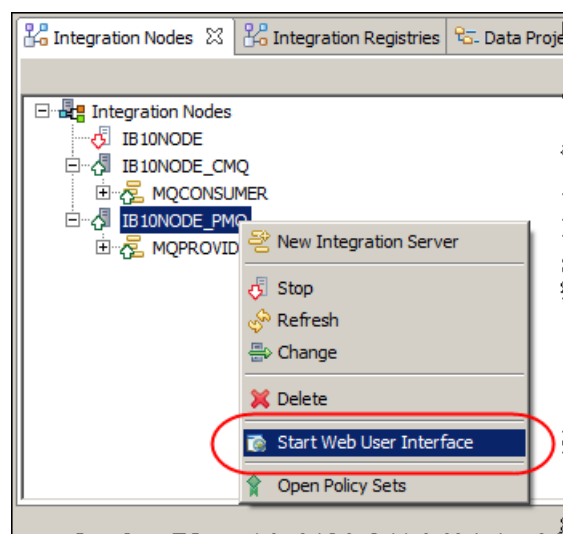
- The Policy is configured to communicate with the queue manager QM4SSL using client connection properties. It has no SSL configuration, this will be added when you have verified the application connectivity to the QM4SSL queue manager:



- Leave the browser open on this window, you will use it later in the lab guide.

3.1.2 Check QM4SSLProvider

- In the "Integration Nodes" view in Toolkit, right click on the IB10NODE_PMQ node and Start the Web User Interface:



(Note: if the Integration Nodes view does not show that the IIB nodes are running, right click on the node and click refresh).

2. In the Web UI, expand **IB10NODE_PMQ > Operational Policy > MQEndpoint**
Click **QM4SSLProvider** to show the MQEndpoint policy details.
3. The QM4SSLProvider policy should look exactly the same as the QM4SSLConsumer policy:
(do not change any of the fields)

The screenshot displays the configuration page for the 'QM4SSLProvider' policy. The title bar indicates 'Operational Policy - MQEndpoint : QM4SSLProvider'. Below the title bar, there are buttons for 'Save', 'Save As', and 'Revert'. A blue information banner at the top states: 'Use a policy to control the operational behavior of a message flow node at run time. [More...]' with a timestamp of 'May 11, 2015, 6:30:49 PM'. The configuration fields are as follows:

Policy URL	/apiv1/policy/MQEndpoint/QM4SSLProvider
Connection	MQ client connection properties
Queue manager name	QM4SSL
Queue manager host name	betaworks-esb10
Listener port number	1444
Channel name	TOQM4SSL
Security identity	
Use SSL	<input type="checkbox"/>
SSL peer name	
SSL cipher specification	

4. Leave the browser open on this window, you will use it later in the lab guide.

4. Import the MQ applications

1. In the Integration Toolkit, switch to a new workspace, call it "MQSecurity"
(File > Switch Workspace > Other)
2. Import the following into your workspace:
 - a) "**EmployeeServiceInterface.V10.zip**" in "**C:\student10\Integration_service\solution**"
 - b) "**MQEmployeeService.SSL.V10.zip**" in "**C:\student10\MQ_SSL\Resources**"
3. Open JSON_MQClient.msgflow in the message flow editor and verify that the Policy URL on the following nodes is set to "**/apiv1/policy/MQEndpoint/QM4SSLConsumer**":
 - MQOutput node: Submit Request
 - MQOutput node: Copy Request
 - MQOutput node: Copy Response
 - MQGet node: Process Response
4. Open getEmployeeDetails.msgflow in the message flow editor and verify that the Policy URL on the following nodes is set to "**/apiv1/policy/MQEndpoint/QM4SSLProvider**":
 - MQInput node: Read Request
 - MQOutput node: Write Response

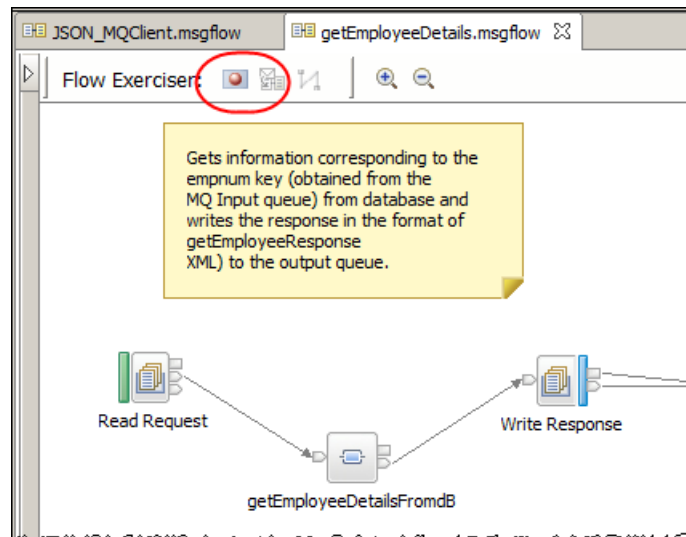
5. Test the applications (No SSL configuration)

In this section you will test that the MQ applications work with QM4SSL with no security using the IIB Flow Exerciser.

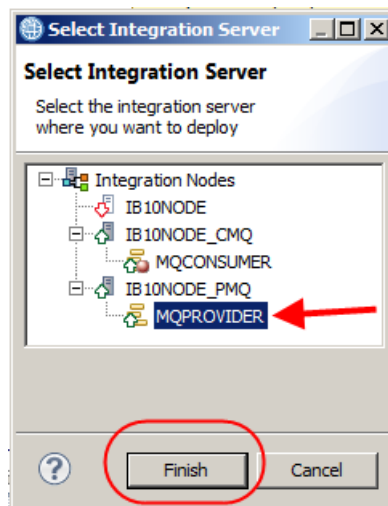
5.1 Start the Flow Exerciser (MQ Provider)

Starting the Flow Exerciser for the provider will deploy the application.

1. In the Integration Toolkit, click the record button on the `getEmployeeDetails.msgflow`:



2. When prompted to select the Integration Server, choose **IB10NODE_PMQ.MQPROVIDER** and click Finish:



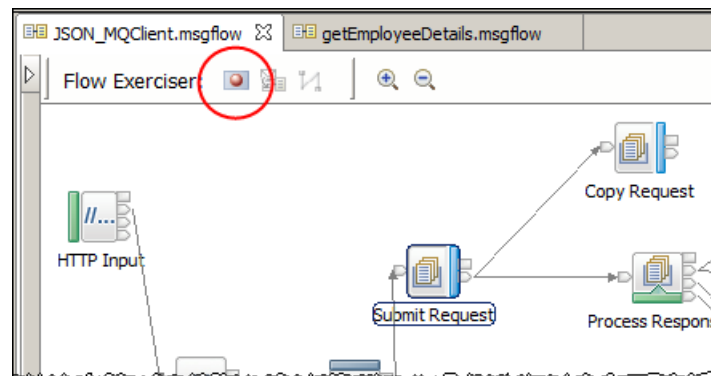
(Note if you chose a different Integration Server where the MQEndpoint Policy attached to the MQ nodes in a message flow is not defined in the IIB node's Integration Registry, the deployment of the application will fail.

If this happens you will see a message "BIP7989E: The policy of type 'MQEndpoint' with name 'MQApplianceProvider' can not be found in the Integration Registry.").

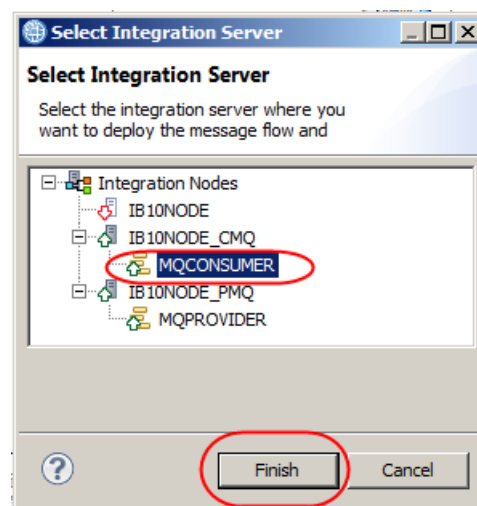
Ensure the application deploys successfully.

5.2 Test the MQ Consumer Application

1. Click the record button on the JSON_MQClient.msgflow:

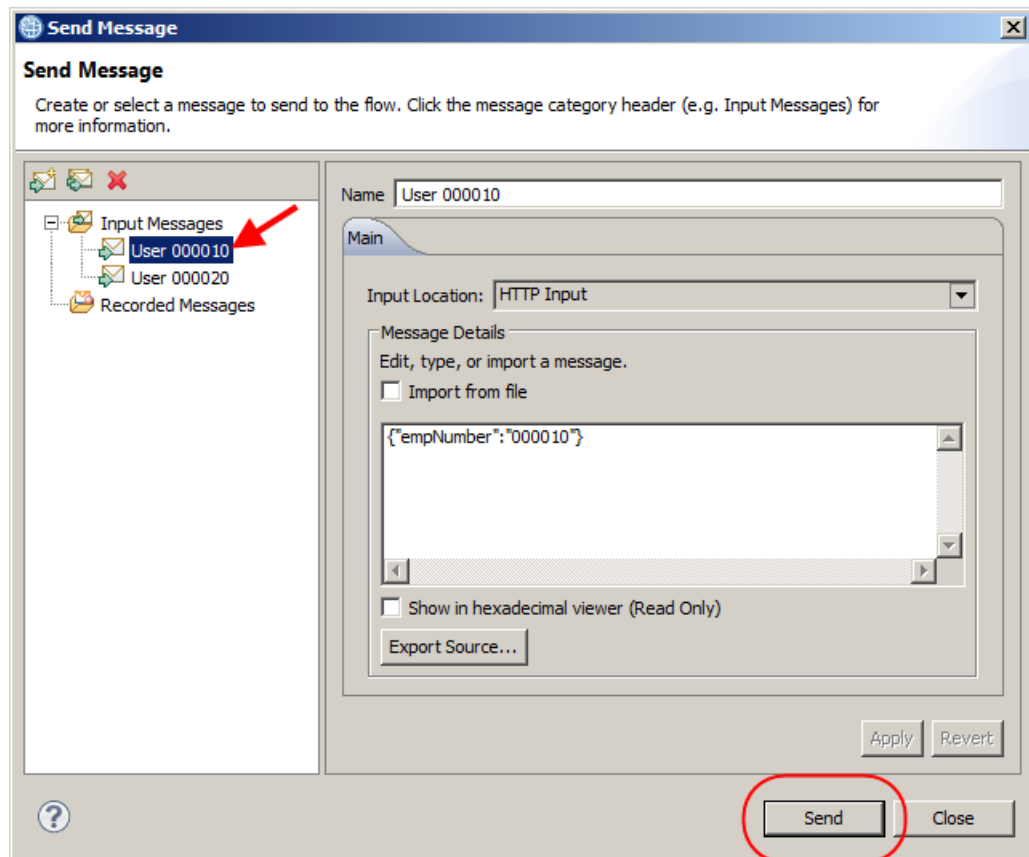


2. When prompted to select the Integration Server, choose IB10NODE_CMQ.MQCONSUMER and click Finish:



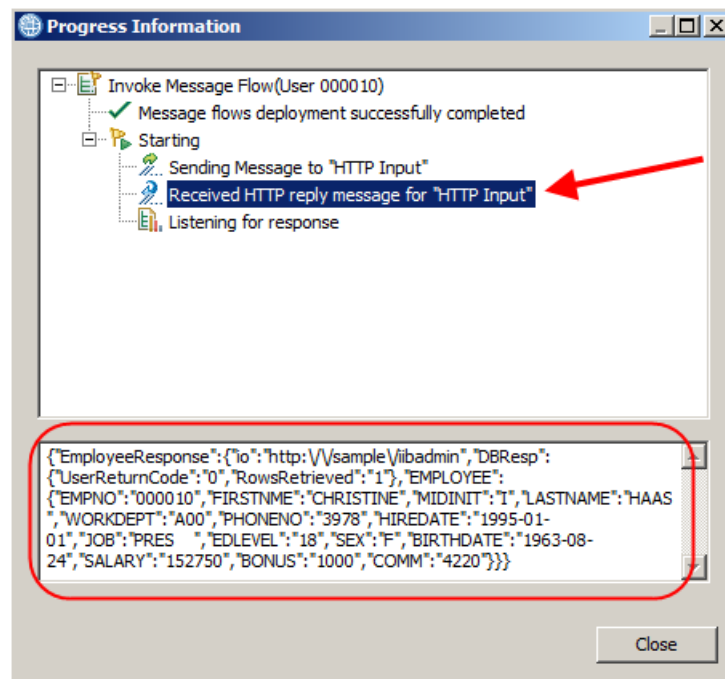
3. Dismiss the "Ready to record message" by clicking OK.
4. Click the Send Message icon to open the dialogue to send a message to the flow.

5. Highlight User 000010 and click Send:



5.3 Verify the two applications have worked correctly

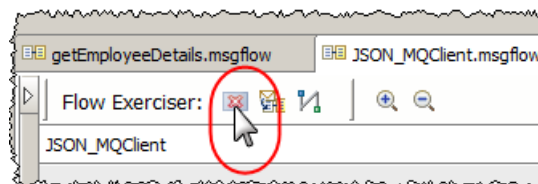
1. In the progress Information window you will see data from an HTTP reply node containing a data from the EMPLOYEE table formatted as a JSON message:



2. Close the Progress Information window to display the path to the message took through the message flow.

The remainder of this lab guide will guide you through SSL configuration necessary so that the MQ applications can access queues defined on the queue manager after it is configured to use SSL for communication.

Click the “Return Flow to edit mode” icon:



6. Scenario: SSL Configured only on MQ side

In this scenario you will cover how to configure SSL on a WebSphere MQ queue manager and identify what will happen if IIB is not configured to use SSL when attempting to communicate with the SSL enabled queue manager.

6.1 Configure SSL on QM4SSL

1. In an Integration Console, navigate to `C:\student10\MQ_SSL\commands` and enter the command

```
03Setup_QM4SSL_PKI
```

2. The script copies the supplied (kdb) keystore into the queue manager configuration. Three files are copied. The script will show the file system path and copied files:

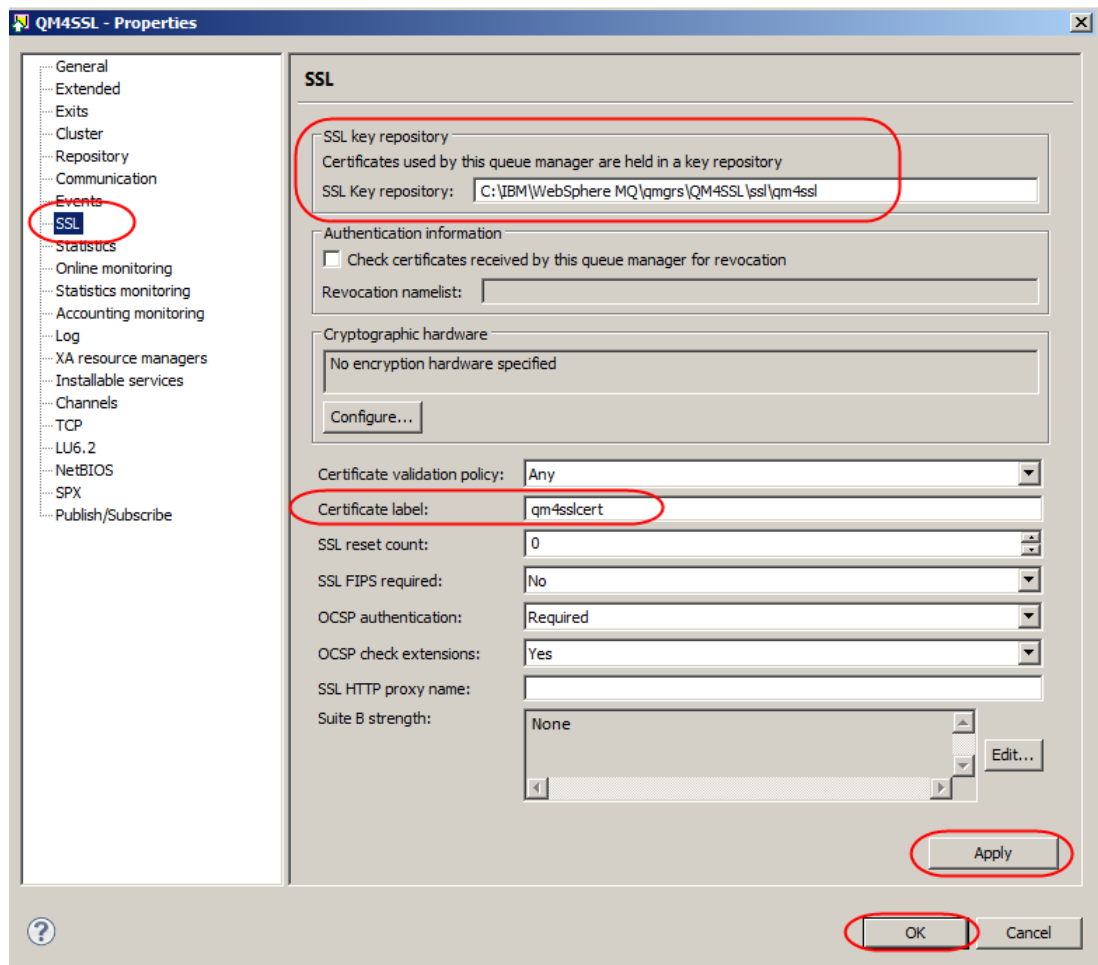
```
Directory of C:\IBM\WebSphere MQ\qmgrs\QM4SSL\ssl
10/05/2015  00:35    <DIR>          .
10/05/2015  00:35    <DIR>          ..
10/05/2015  00:40                10,080 QM4SSL.kdb
10/05/2015  00:40                 80 QM4SSL.rdb
10/05/2015  00:35                129 QM4SSL.sth
```

3. Open WebSphere MQ Explorer, expand Queue Managers and open the properties window for QM4SSL (right click on the name and select Properties)

Click SSL to show the queue manager SSL properties.

4. Configure the following:
 - a) "SSL Key repository" to the full path of the key store (kdb) file. The value must not contain the kdb file extension i.e. set the value to:
`C:\IBM\WebSphere MQ\qmgrs\QM4SSL\ssl\qm4ssl`

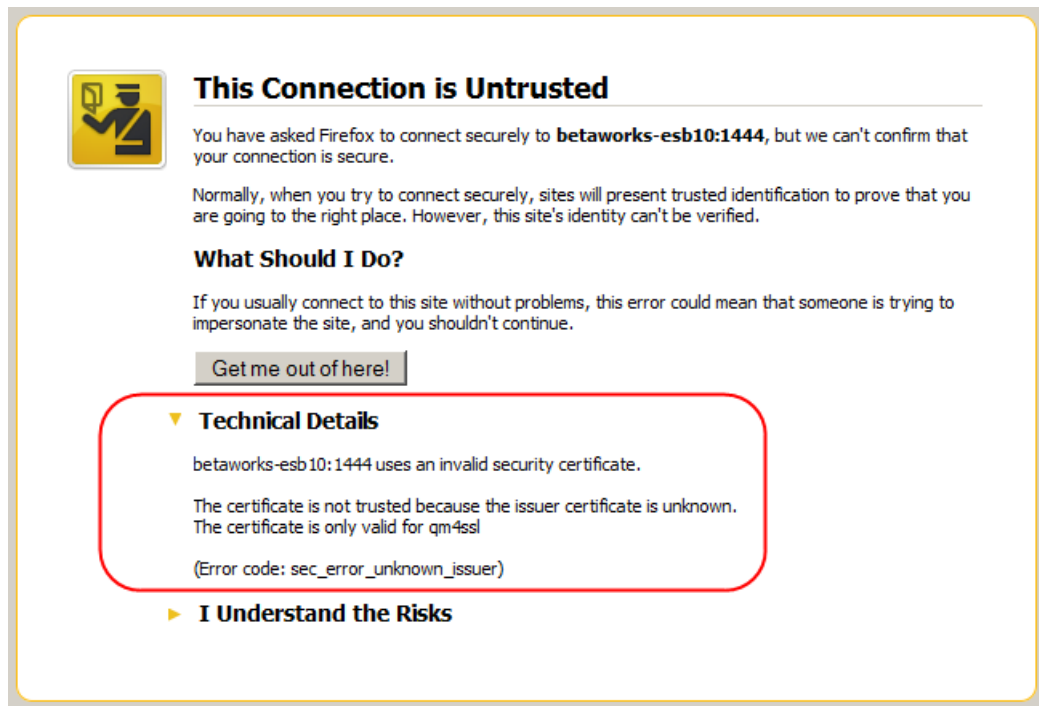
(note there is a space in "WebSphere MQ")
 - b) Set the Certificate label to "qm4sslcert"
 - c) Click Apply then click OK
 - d) When prompted, if you want to change the location of the key repository, click yes



5. Open a browser tab in Firefox and enter the following URL:

`https://betaworks-esb10:1444/`

6. The response from the browser (despite not being able to verify the connection since we have used a non-secure predefined Root CA and certificate) verified that the queue manager is capable of handling SSL requests:



This Connection is Untrusted

You have asked Firefox to connect securely to **betaworks-esb10:1444**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

[Get me out of here!](#)

▼ **Technical Details**

betaworks-esb10:1444 uses an invalid security certificate.

The certificate is not trusted because the issuer certificate is unknown.
The certificate is only valid for qm4ssl

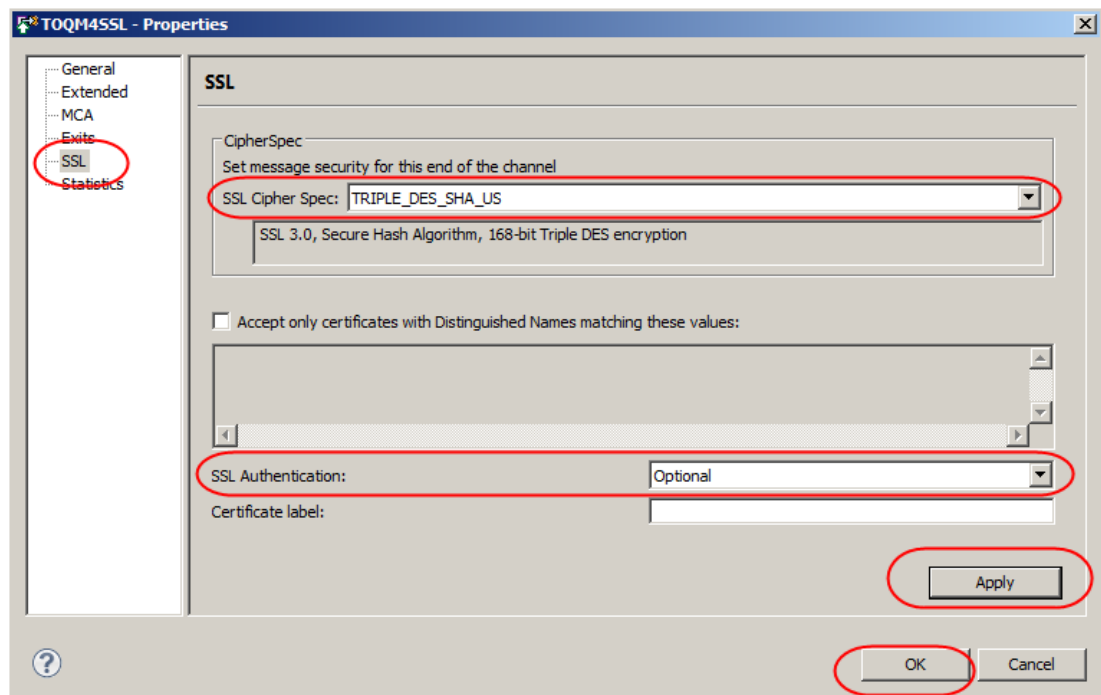
(Error code: sec_error_unknown_issuer)

► **I Understand the Risks**

6.2 Configure SSL on Channel TOQM4SSL

1. In the MQ Explorer click QM4SSL > Channels.
In the list of channels, right click on TOQM4SSL and click Properties.
2. In the Properties window, select SSL and configure the following:
 - a) SSL Cipher Spec: **TRIPLE_DES_SHA_US**
 - b) SSL Authentication: **Optional**

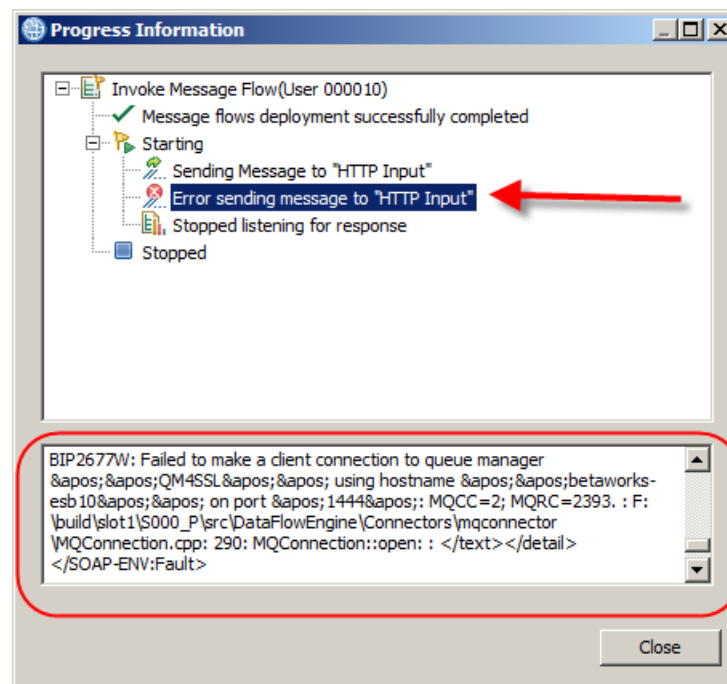
Click Apply and then OK.



6.3 Retest the MQ applications

1. In the Integration Toolkit, Start the flow exerciser (press the record button) and send a message to the MQ JSON_MQClient message flow.
2. With the TOQM4SSL connection now expecting to communicate via SSL, the attempt to communicate to with the queue manager, fails.

Scroll down to the bottom of the messages to see the MQ return code **MQRC = 2393** (**MQRC_SSL_INITIALIZATION_ERROR**).



The error occurs as there is currently no SSL set up on the IIB side of the communication attempt.

7. Scenario: SSL configured on IIB and MQ

When communicating with a remote MQ queue manager, IIB uses an MQ Client. When the IIB queue manager is enabled for SSL the IIB node requires the following configuration:

- 1) A KDB key store with a key to identify the MQ Client and the public certificate used by the SSL enabled queue manager.
- 2) A BrokerRegistry definition to identify the KDB key store.
- 3) *(Optionally) a Client Connection Definition Table (CCDT) with MQ Client Connection definition that matches the MQ Server Connection definition on the queue manager (in the SSL case this will need to contain the Cipher spec defined in the SERVER CONN definition on the SSL queue manager)*
- 4) *(Optionally) a BrokerRegistry definition to identify the CCDT to the IIB node.*

7.1 The (MQClient) KDB key store(s)

In this lab guide two KDB key stores (one for each IIB node) will be used to identify the two nodes IB10NODE_CMQ (MQ Consumer application) and IB10NODE_PMQ (MQ Provider application).

The KDB key stores are supplied in `C:\student10\MQ_SSL\Keystores\SSLConfig`. *(There is no need to change these files)*. The files will be used when the command to define the IIB MQ Client PKI is run later in this lab guide.

7.2 MQ Client BrokerRegistry properties

Two new BrokerRegistry properties now exist to define MQ Client properties:

- a) `mqCCDT`: Used to define the full path of the client connection definition table. IIB will parse this property and set `MQCHLLIB` and `MQCHLTAB` environment variables. NOTE: if these variables are already set (for example by the MQ Client system settings) IIB will **<not>** override these properties and will use the environment variables that are already set. A message "BIP2282W" will be written to warn that the environment variables are already set and the `mqCCDT` property will not be used. *This property will not be used in this section of the lab guide.*
- b) `mqKeyRepository`: used to define the key store used by the MQ client.

7.2.1 Set the `mqKeyRepository` property

1. In an Integration Console window, navigate to :

```
C:\student10\MQ_SSL\commands
```

and enter the command:

```
04Setup_MQCPKI.cmd
```

Accept the defaults when prompted:

```
This command file must be run within an Integration Bus Command Console.
BetaWorks MQ SSL Lab guide set BrokerRegistry properties:
Enter PROVIDER IIB Node name (default is IB10NODE_PMQ):
Enter CONSUMER IIB Node name (default is IB10NODE_CMQ):
Set mqCCDT property (n):
Thankyou, using values "IB10NODE_PMQ", "IB10NODE_CMQ", mq CCDT (n) Ok to
proceed
? Use Ctrl-C to terminate.
Press any key to continue . . .
-
```

The command sets the `mqKeyRepository` property value in the BrokerRegistry for `IB10NODE_CMQ` and `IB10NODE_PMQ`.

2. The response from the command will be:

```
Press any key to continue . . .
-
Configuring MQ Client CCDT and KDB keystore for node "IB10NODE_PMQ"
-
BIP8071I: Successful command completion.
BIP8071I: Successful command completion.
-
Configuring MQ Client CCDT and KDB keystore for node "IB10NODE_CMQ"
-
BIP8071I: Successful command completion.
BIP8071I: Successful command completion.
-
```

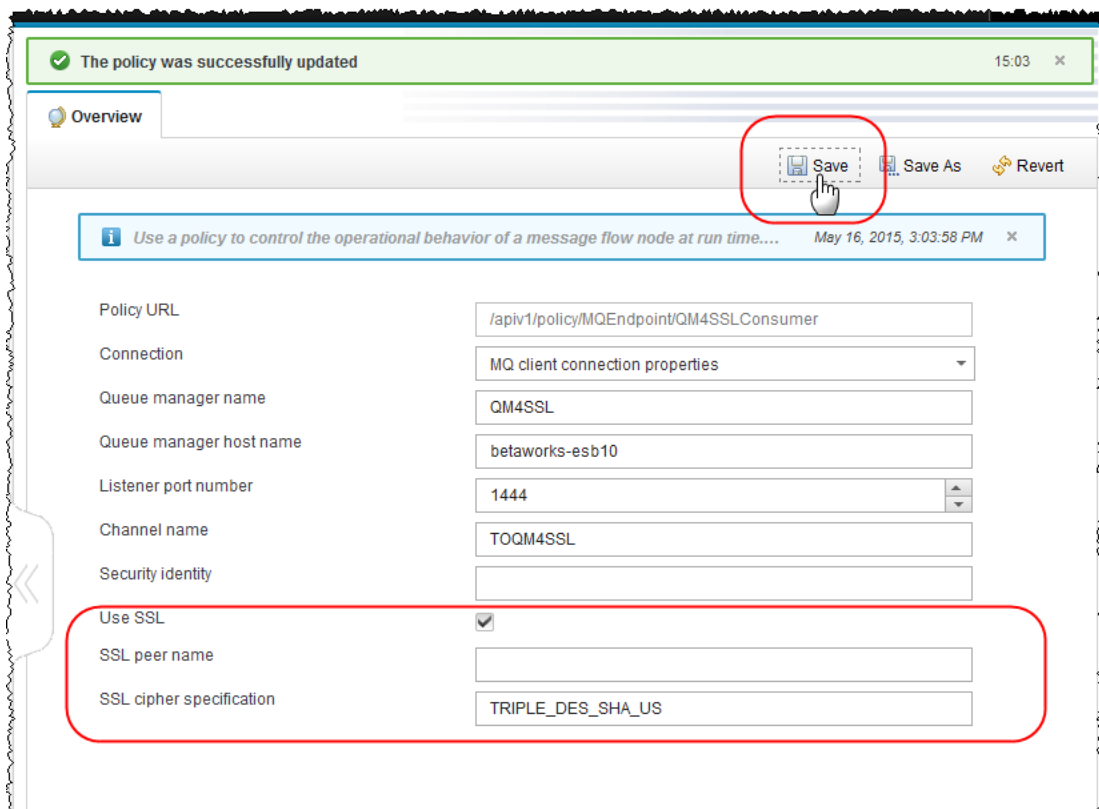
The Script will then stop and restart both IIB nodes `IB10NODE_PMQ` and `IB10NODE_CMQ`. (ensure they restart correctly, response from script not shown)

7.3 Modify the MQEndpoint Policies

The IIB nodes are now configured to handle client connection attempts to SSL enabled queue managers. The MQ nodes in the MQ based Consumer and Provider application are configured to use MQ Endpoint policies. These policies will now be configured to use SSL.

7.3.1 Modify QM4SSLConsumer

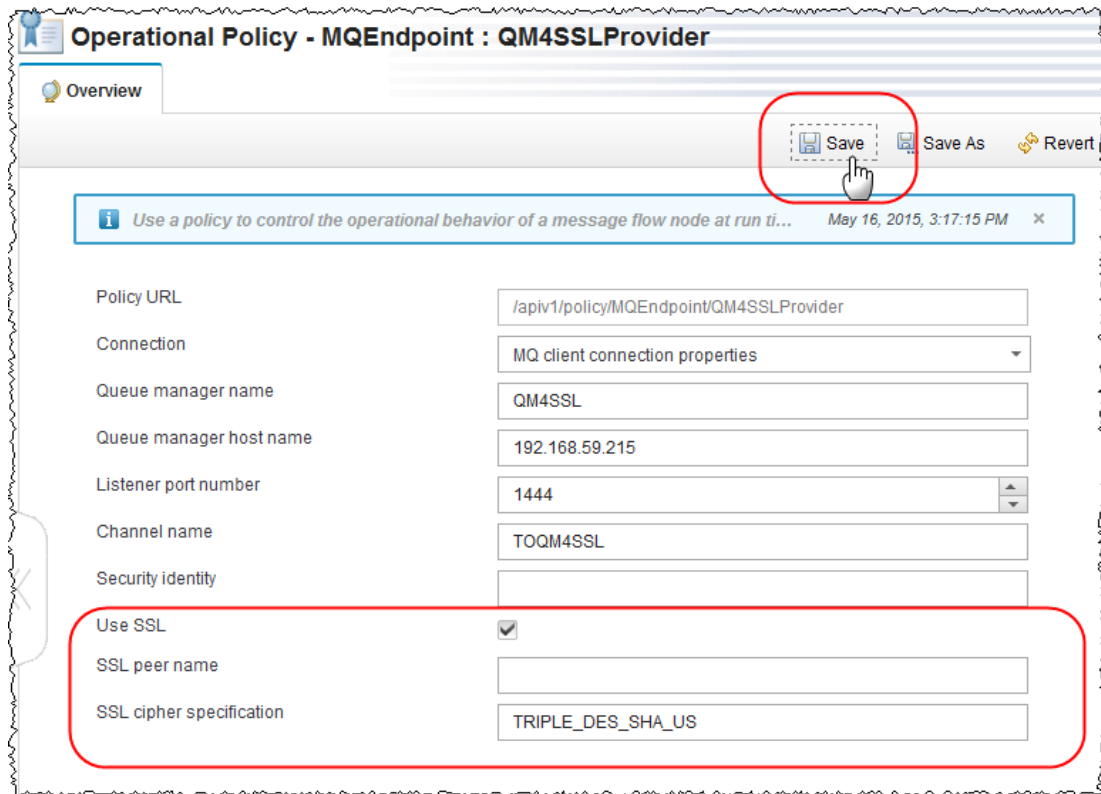
1. In the browser window displaying the MQEndpoint Policy QM4SSLConsumer, set the following SSL properties in the policy and then click save:
 - Use SSL (check the tick box)
 - SSL cipher specification: `TRIPLE_DES_SHA_US`



2. Leave the browser window open, you will reconfigure the policy later in this guide

7.3.2 Modify QM4SSLProvider

1. In the IIB web user interface (browser) displaying the MQEndpoint Policy QM4SSLProvider, set the same properties as above then click save:
 - Use SSL (**check the tick box**)
 - SSL cipher specification: **TRIPLE_DES_SHA_US**



Operational Policy - MQEndpoint : QM4SSLProvider

Overview

Save Save As Revert

Use a policy to control the operational behavior of a message flow node at run ti... May 16, 2015, 3:17:15 PM x

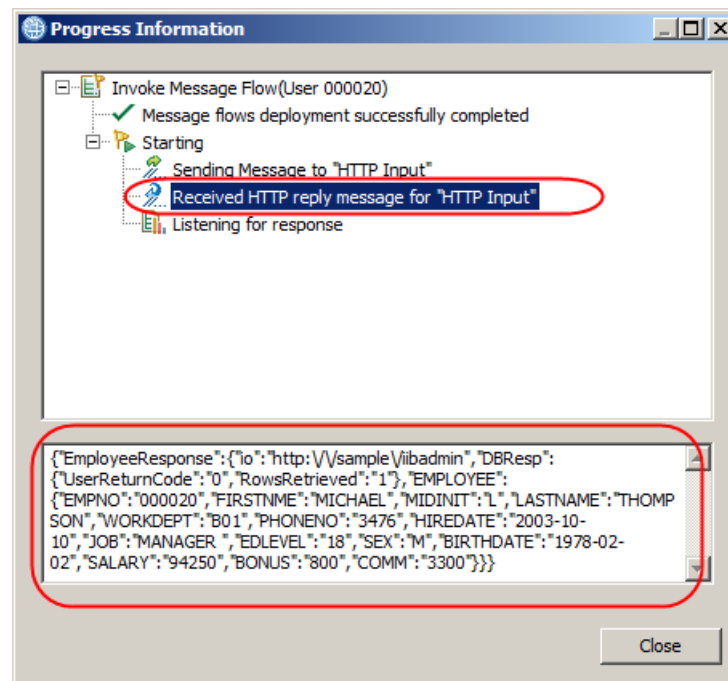
Policy URL	/apiv1/policy/MQEndpoint/QM4SSLProvider
Connection	MQ client connection properties
Queue manager name	QM4SSL
Queue manager host name	192.168.59.215
Listener port number	1444
Channel name	TOQM4SSL
Security identity	
Use SSL	<input checked="" type="checkbox"/>
SSL peer name	
SSL cipher specification	TRIPLE_DES_SHA_US

2. Leave the web user interface window open, you will reconfigure the policy later in this guide

7.4 Retest the MQ applications

1. Note you may need to refresh the Integration Nodes view to see the up to date status of the two IIB nodes. Make sure the IIB Toolkit is aware that the two nodes are started, otherwise the Flow Exerciser will stop recording messages through the message flow.
2. In the Integration Toolkit, send a message to the JSON_MQClient message flow.
3. With the complete SSL set up on:
 - a. the queue manager
 - b. the IIB MQ Client

sending the message now works correctly:



4. Close the progress window to show the route that the message took through the message flow.

8. Using Security Identities

When you configure an MQ connection from an MQ node to a WebSphere MQ queue manager, you can optionally configure the connection to use a security identity for authentication, SSL for confidentiality, or both. The security identity, which passes user name and password security credentials to the queue manager, can be used on connections to local or remote queue managers.

8.1 Defining security identities to an IIB Node

A security identity is defined on an IIB node using the `mqsisetdbparms` command. In this section you will define two MQ Security Ids. One with no MQ Access and a separate security identity with full MQ access.

1. In an Integration Console window, enter the following command:

```
mqsisetdbparms IB10NODE_CMQ  
-n mq: :NOMQAUTH  
-u NOMQAUTH  
-p passw0rd
```

This command defines a security identity called "NOMQAUTH" and associates a userid of "NOMQAUTH" with a password of "passw0rd".

Ensure the command response is successful ("BIP8071I: Successful command completion")

NB: For the purposes of this lab guide the security identity will only be defined on IB10NODE_CMQ as the attempt to write a request to the MQREQUEST queue will fail when this identity is used (so there is no reason to define it on the IB10NODE_PMQ).

2. Enter the following command:

```
mqsisetdbparms IB10NODE_CMQ  
-n mq: :FULLMQAUTH  
-u iibadmin  
-p passw0rd
```

This command defines a security identity called "FULLMQAUTH" and associates a userid of "iibadmin" with a password of "passw0rd".

Ensure the command response is successful ("BIP8071I: Successful command completion")

3. We expect the attempt to write a message to the MQREQUEST queue on QM4SSL to be successful. The FULLMQAUTH security identity also needs to be defined on the IIBNODE_PMQ.

Enter the following command:

```
mqsisetdbparms IB10NODE_PMQ  
-n mq: :FULLMQAUTH  
-u iibadmin  
-p passw0rd
```

This command defines a security identity called "FULLMQAUTH" on the IIB node IB10NODE_PMQ, and associates a userid of "iibadmin" with a password of "passw0rd"

Ensure the command response is successful ("BIP8071I: Successful command completion").

4. To check the security identity properties on IB10NODE_CMQ, type the follow command:

```
mqsireportdbparms IB10NODE_CMQ
-n mq::*
```

The command response should be:

```
C:\student10\MQ_SSL\commands>mqsireportdbparms IB10NODE_CMQ -n mq::*
BIP8180I: The Resource name 'mq::FULLMQAUTH' has userID 'iibadmin'
BIP8180I: The Resource name 'mq::NOMQAUTH' has userID 'NOMQAUTH'

BIP8071I: Successful command completion.
```

5. To check the security identity properties on IB10NODE_PMQ, type the follow command:

```
mqsireportdbparms IB10NODE_PMQ
-n mq::*
```

The command response should be:

```
C:\student10\MQ_SSL\commands>mqsireportdbparms IB10NODE_PMQ -n mq::*
BIP8180I: The Resource name 'mq::FULLMQAUTH' has userID 'iibadmin'

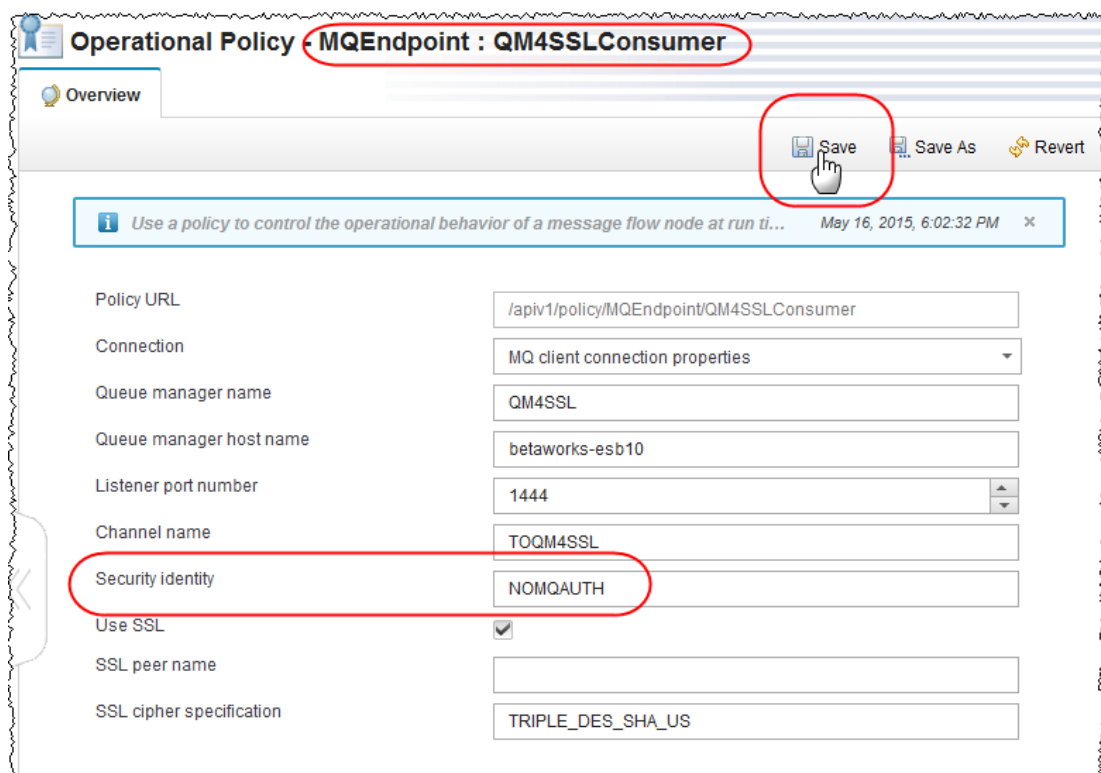
BIP8071I: Successful command completion.
```

8.2 Scenario: Authentication check: Failed

In this scenario you specify that the MQ applications will use the Security Identity that will fail MQ Authentication checks.

8.2.1 Modify QM4SSLConsumer

1. In the browser window displaying the MQEndpoint Policy **QM4SSLConsumer**, set the following SSL properties in the policy and then click save:
 - Security Identity: **NOMQAUTH**



The screenshot shows the configuration page for the Operational Policy **MQEndpoint : QM4SSLConsumer**. The 'Security identity' field is set to **NOMQAUTH**. The 'Save' button is highlighted with a red circle. The 'Use SSL' checkbox is checked. The 'SSL cipher specification' is set to **TRIPLE_DES_SHA_US**.

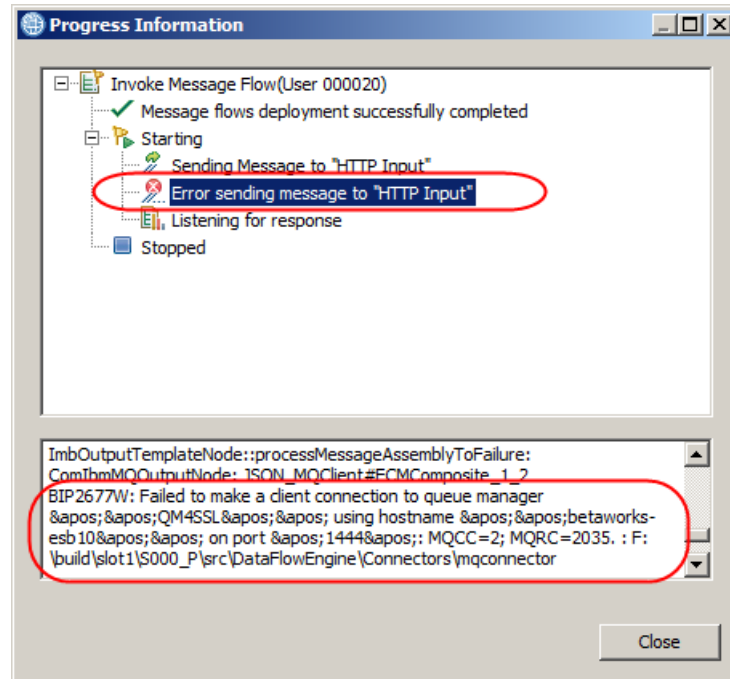
Policy URL	/apiv1/policy/MQEndpoint/QM4SSLConsumer
Connection	MQ client connection properties
Queue manager name	QM4SSL
Queue manager host name	betaworks-esb10
Listener port number	1444
Channel name	TOQM4SSL
Security identity	NOMQAUTH
Use SSL	<input checked="" type="checkbox"/>
SSL peer name	
SSL cipher specification	TRIPLE_DES_SHA_US

2. Leave the browser window open, you will reconfigure the policy later in this guide.

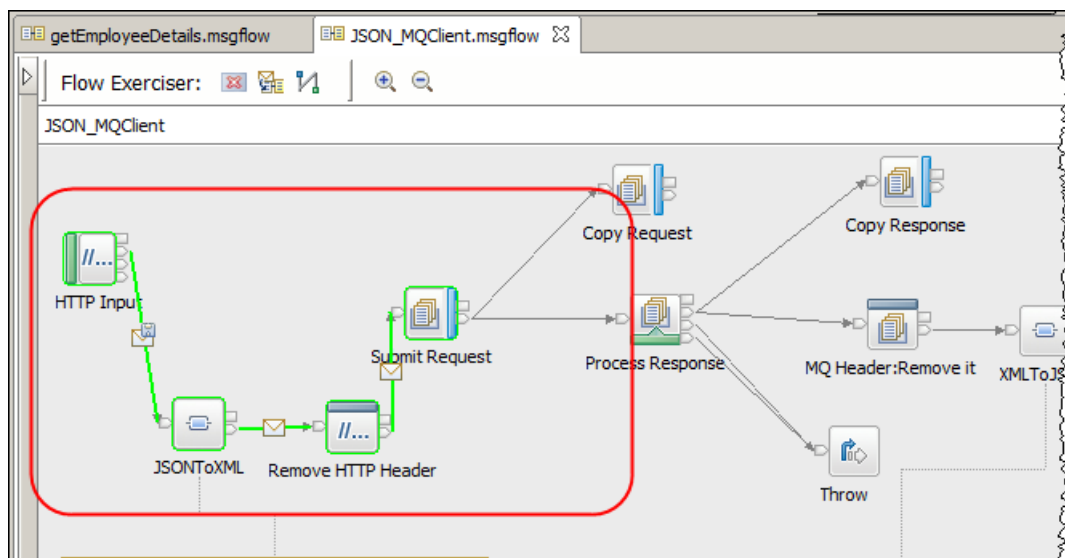
8.2.2 Retest the MQ applications

1. Open the IIB Event Log Monitor (from the Windows Start menu) so that you can see messages being written to the system logs by the IIB nodes.
2. In the Integration Toolkit, send a message to the JSON_MQClient message flow.

- The Request will fail. Click the Error message and scroll down the bottom window to see that the attempted write to the MQREQUEST queue failed, with MQRC = 2035 (Not Authorised)



- Close the progress window to show the route that the message took through the message flow. This time the route taken finishes at the "Submit Request" MQOutput node:



8.2.3 Verify the Security Identity passed to MQ

- Using Windows Explorer, navigate to:

C:\IBM\WebSphere MQ\qmgrs\QM4SSL\errors

Three logs with a (prefixed name of "AMQERR" exist in this folder.

Note the current time and edit the file with the closest "Date modified" to the current time.

2. Search the log for the text **NOMQAUTH**.

You will see a message explaining that the user ID supplied by IIB to WebSphere MQ failed authentication checks:

```
----- cmqxrsrv.c : 2199 -----  
5/16/2015 18:07:30 - Process(3920.4383) User(MUSR_MQADMIN) Program(amqzlaa0.exe)  
Host(BETAWORKS-ESB10) Installation(Installation1)  
VRMF(8.0.0.0) QMgr(QM4SSL)
```

```
AMQ5534: User ID 'NOMQAUTH' authentication failed
```

EXPLANATION:

The user ID and password supplied by 'erver\bin\DataFlowEngine.exe' could not be authenticated.

ACTION:

Ensure that the correct user ID and password are provided by the application.
Ensure that the authentication repository is correctly configured. Look at previous error messages for any additional information.

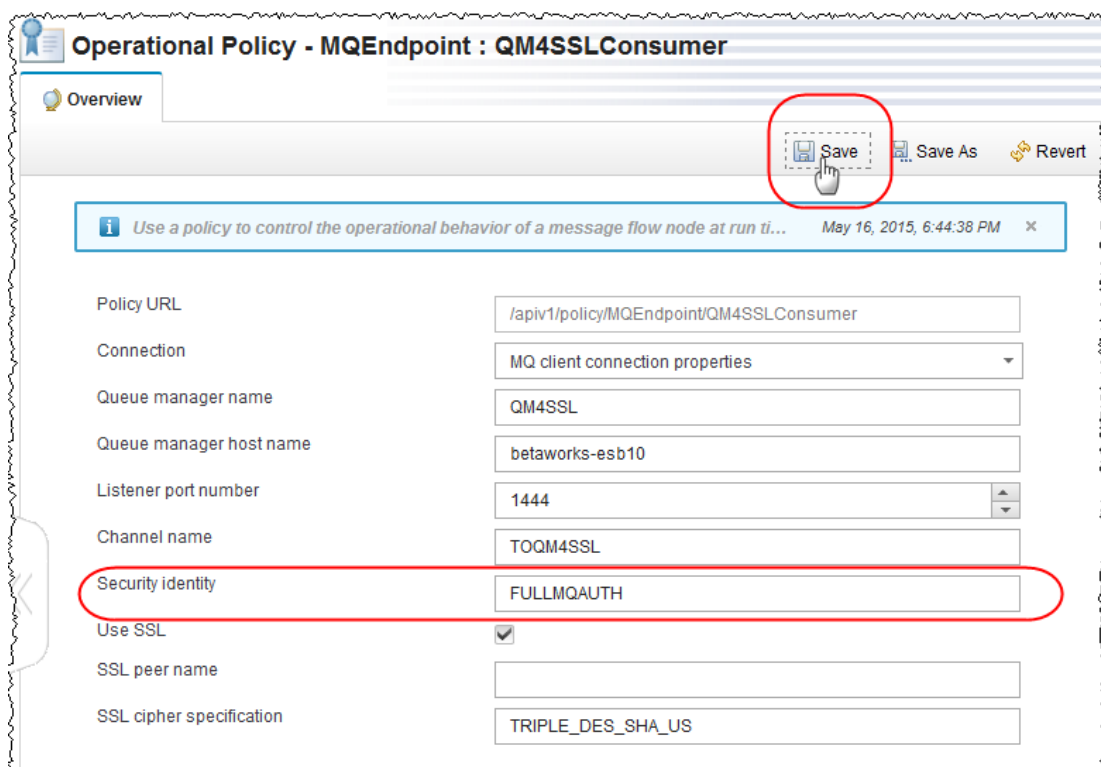
```
----- amqzfuca.c : 4242 -----
```

8.3 Scenario: Authentication check: Passed

In this scenario you specify that the MQ applications will use the Security Identity that will pass the WebSphere MQ authentication check.

8.3.1 Modify QM4SSLConsumer

1. In the browser window displaying the MQEndpoint Policy **QM4SSLConsumer**, set the following SSL properties in the policy and then click save:
 - Security Identity: **FULLMQAUTH**

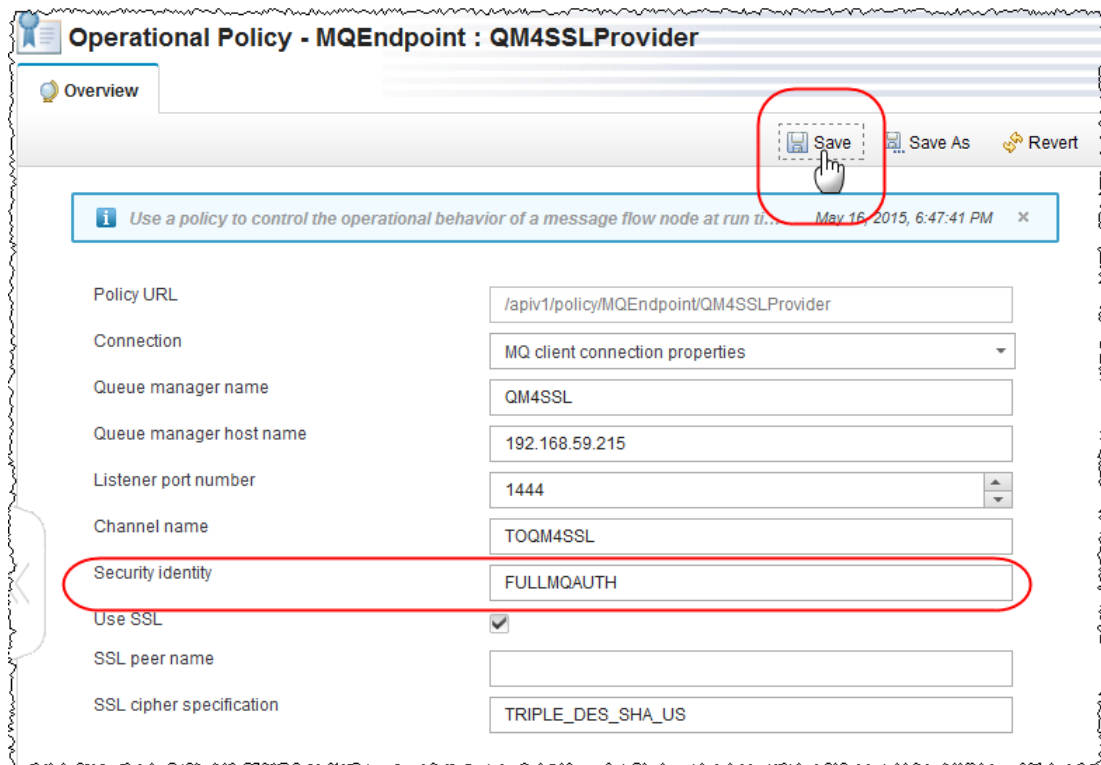


The screenshot displays the configuration page for the Operational Policy - MQEndpoint : QM4SSLConsumer. The 'Security identity' field is highlighted with a red oval and contains the value 'FULLMQAUTH'. The 'Save' button is also highlighted with a red circle. The configuration includes the following fields:

Field	Value
Policy URL	/apiv1/policy/MQEndpoint/QM4SSLConsumer
Connection	MQ client connection properties
Queue manager name	QM4SSL
Queue manager host name	betaworks-esb10
Listener port number	1444
Channel name	TOQM4SSL
Security identity	FULLMQAUTH
Use SSL	<input checked="" type="checkbox"/>
SSL peer name	
SSL cipher specification	TRIPLE_DES_SHA_US

8.3.2 Modify QM4SSLProvider

1. In the browser window displaying the MQEndpoint Policy **QM4SSLProvider**, set the following SSL properties in the policy and then click save:
 - Security Identity: **FULLMQAUTH**



The screenshot shows the configuration page for the Operational Policy - MQEndpoint : QM4SSLProvider. The 'Security identity' field is set to 'FULLMQAUTH'. The 'Save' button is circled in red, and the 'Security identity' field is also circled in red. The 'Use SSL' checkbox is checked. The 'SSL cipher specification' is set to 'TRIPLE_DES_SHA_US'.

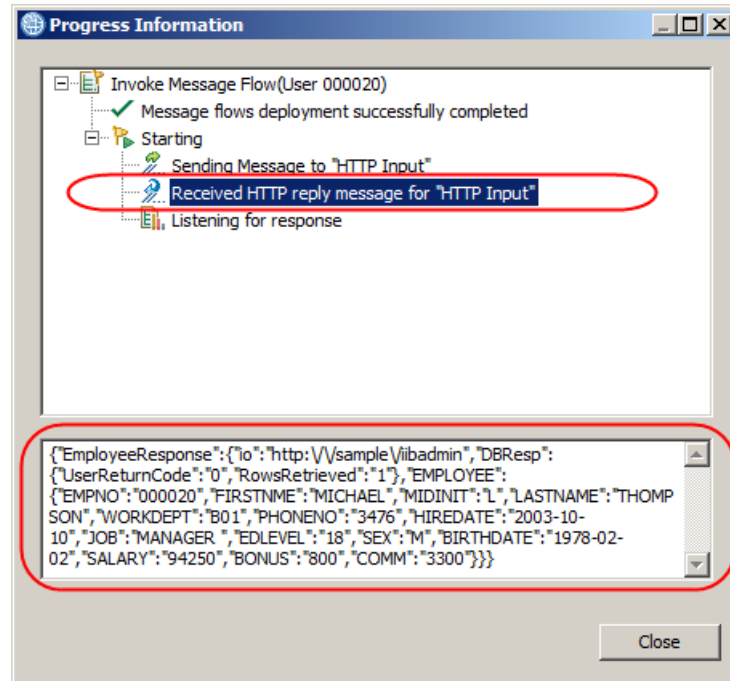
Policy URL	/apiv1/policy/MQEndpoint/QM4SSLProvider
Connection	MQ client connection properties
Queue manager name	QM4SSL
Queue manager host name	192.168.59.215
Listener port number	1444
Channel name	TOQM4SSL
Security identity	FULLMQAUTH
Use SSL	<input checked="" type="checkbox"/>
SSL peer name	
SSL cipher specification	TRIPLE_DES_SHA_US

2. Leave the browser window open, you will reconfigure the policy later in this guide

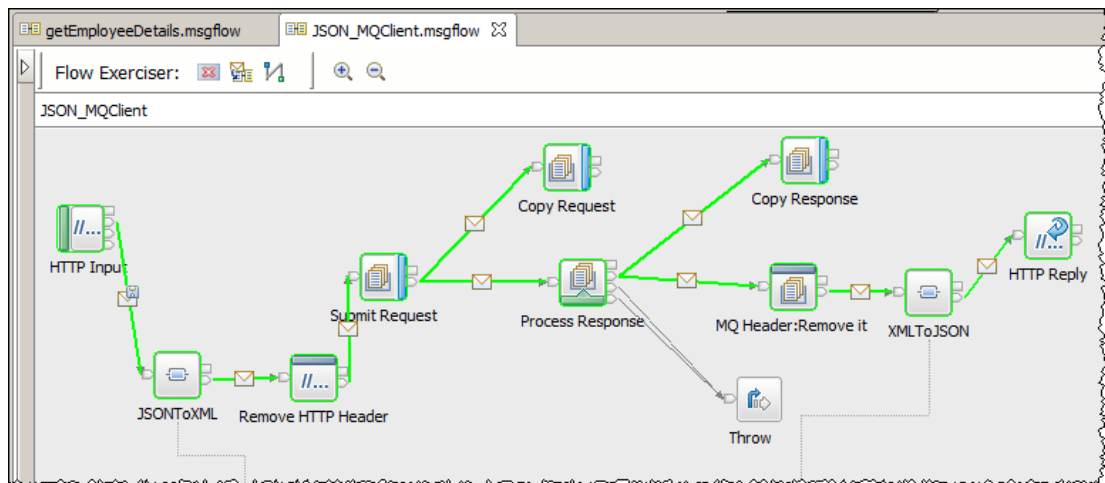
8.3.3 Retest the MQ applications

1. Open the IIB Event Log Monitor so that you can see messages being written to the system logs by the IIB nodes.
2. In the Integration Toolkit, send a message to the MQJSON_MQClient message flow.

- The Request will now work as expected. Click the reply message message to see the data sent by the applications:



- Close the progress window to show the route that the message took through the message flow. This will be the full path through the message flow:



END OF LAB GUIDE