This session provides information on the new HTTPS support that was introduced in WebSphere Message Broker Version 6.

Agenda

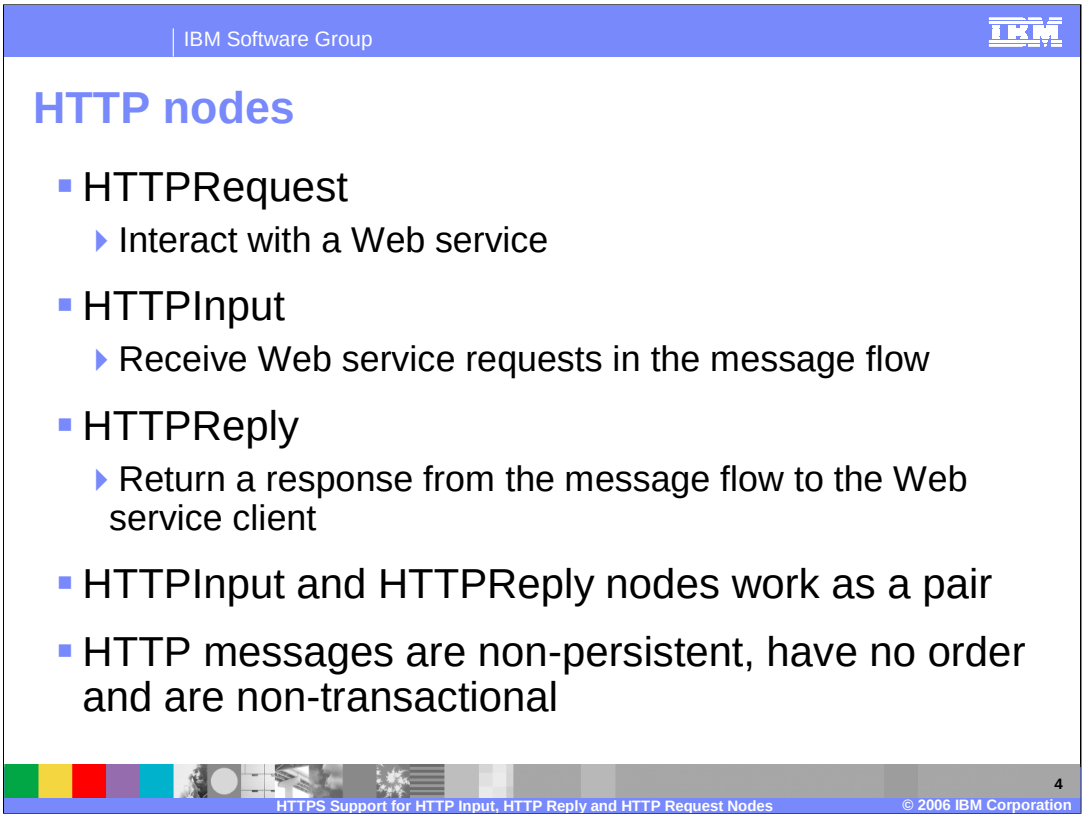
- Overview
- HTTPS enablement
- Summary and References

The agenda for this presentation is to cover HTTPS enablement in overview and in detail.

Section

Overview

First, the overview.



Use the HTTPRequest node to interact with a Web service, using all or part of the input message as the request sent to that service. You can also configure the node to create a new output message from the contents of the input message augmented by the contents of the Web service response before you propagate the message to subsequent nodes in the message flow. By default the HTTPRequest node uses the HTTP POST method when connecting to the remote Web server. To enable the request node to use the HTTP GET method instead, you must set a field in the output local environment tree.

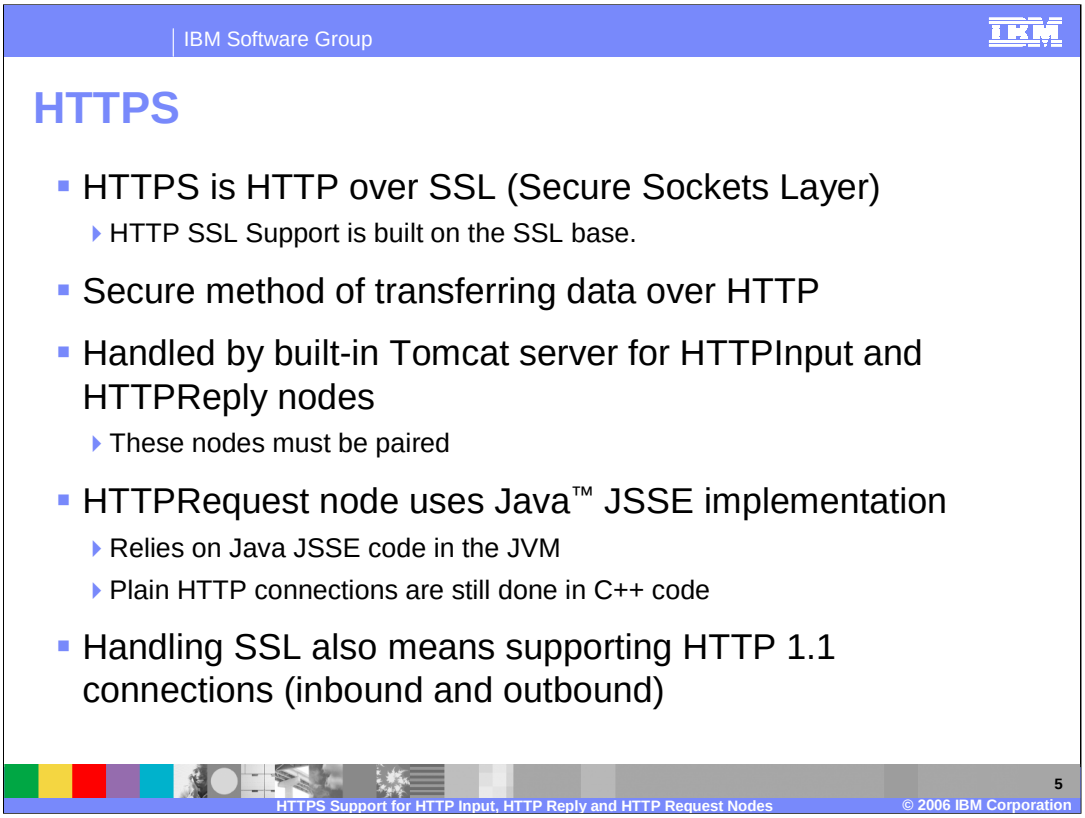
Use the HTTPInput node to receive Web service requests for processing by a message flow. Using the HTTPInput node with the HTTPReply and HTTPRequest nodes, the broker can act as an intermediary for Web services, and Web service requests can be transformed and routed in the same way as other message formats supported by WebSphere Message Broker. Web service requests can be received either in standard HTTP (1.0 or 1.1) format, and also in HTTP over SSL (HTTPS) format. You can set the *Use HTTPS* property to choose whether to handle HTTP or HTTPS requests.

Use the HTTPReply node to return a response from the message flow to the Web service client. This node generates the response to the Web service client from which the input message was received by the HTTPInput node, and waits for confirmation that it has been sent.

The HTTPInput and HTTPReply nodes are used as a pair; if you include an HTTPInput node in a message flow, you must either include an HTTPReply node in the same flow, or pass the message to another flow that includes an HTTPReply node. In the latter case, the request from, and reply to, the client are coordinated by the request identifier stored in the LocalEnvironment.

HTTP messages are always non-persistent, and have no associated order.

HTTP messages are non-transactional

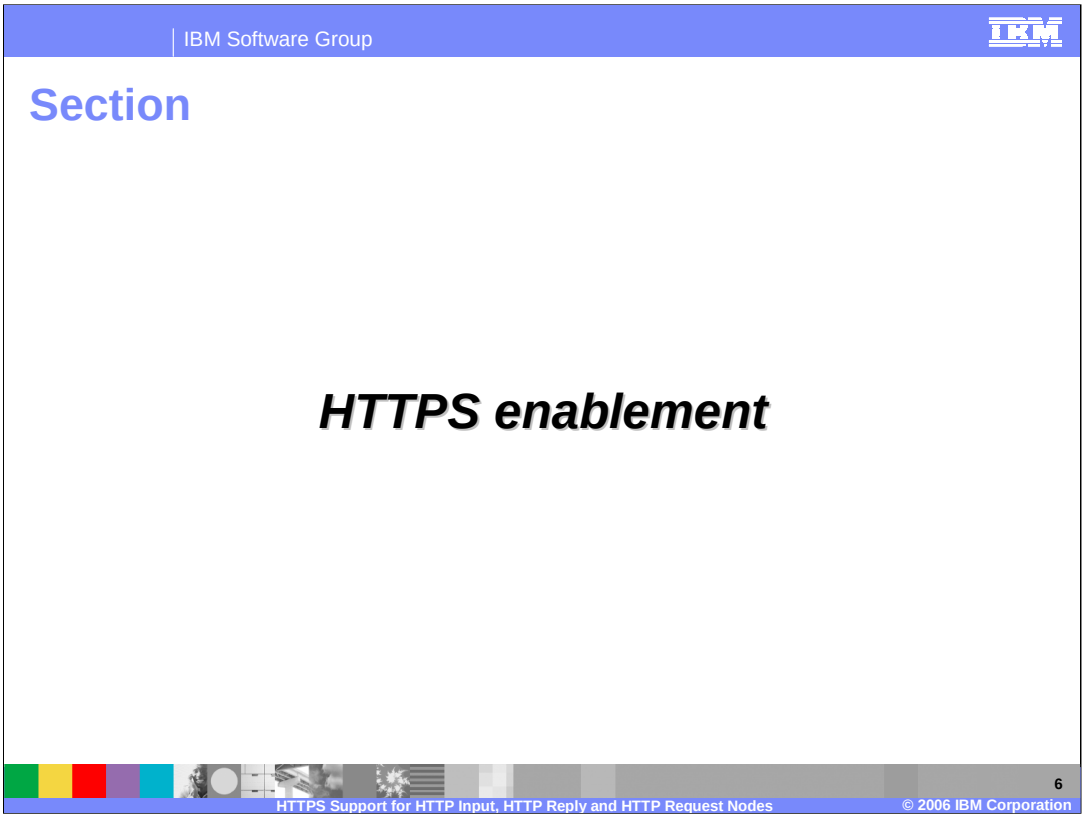


The support for HTTPS in the Version 6 broker simply means that the Secure Sockets Layer, or SSL, protocol is carried over a normal HTTP connection. It is the same support used for secure websites through a web browser.

On the HTTPInput node, the broker uses the SSL support provided by the built-in Tomcat Web server. HTTPInput nodes are paired with a corresponding HTTPReply node, which uses the same mechanism to send the HTTP response to the HTTP Client.

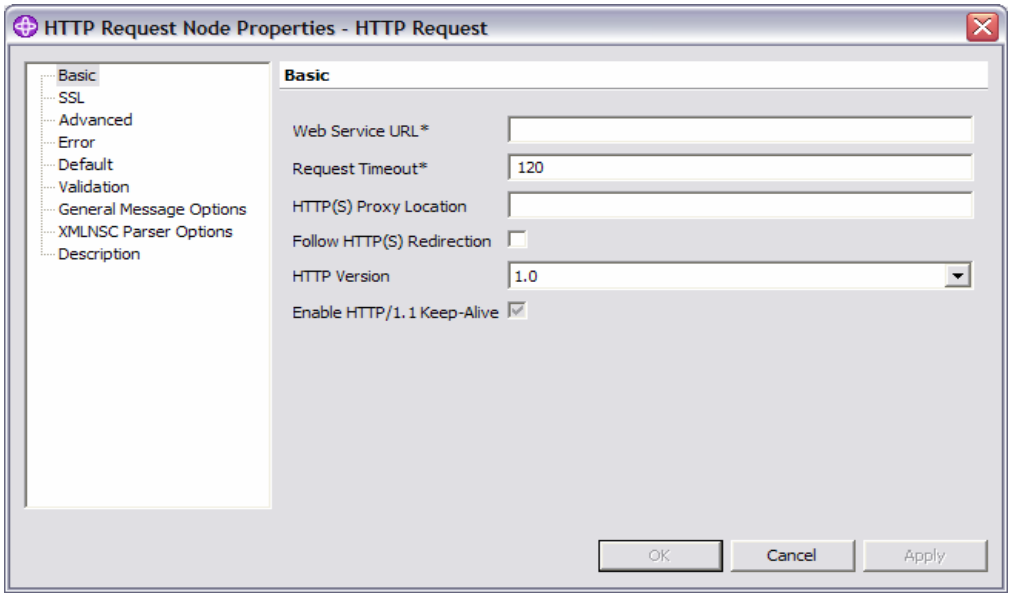
On the HTTPRequest node, the broker uses Java's JSSE implementation. Non-SSL connections are still done in custom C++ code, as they were in the Version 5 broker.

Finally, Version 6 adds support for HTTP 1.1 connections. Version 5 could only support HTTP 1.0 connections. HTTP 1.1 is required for MIME attachment support.

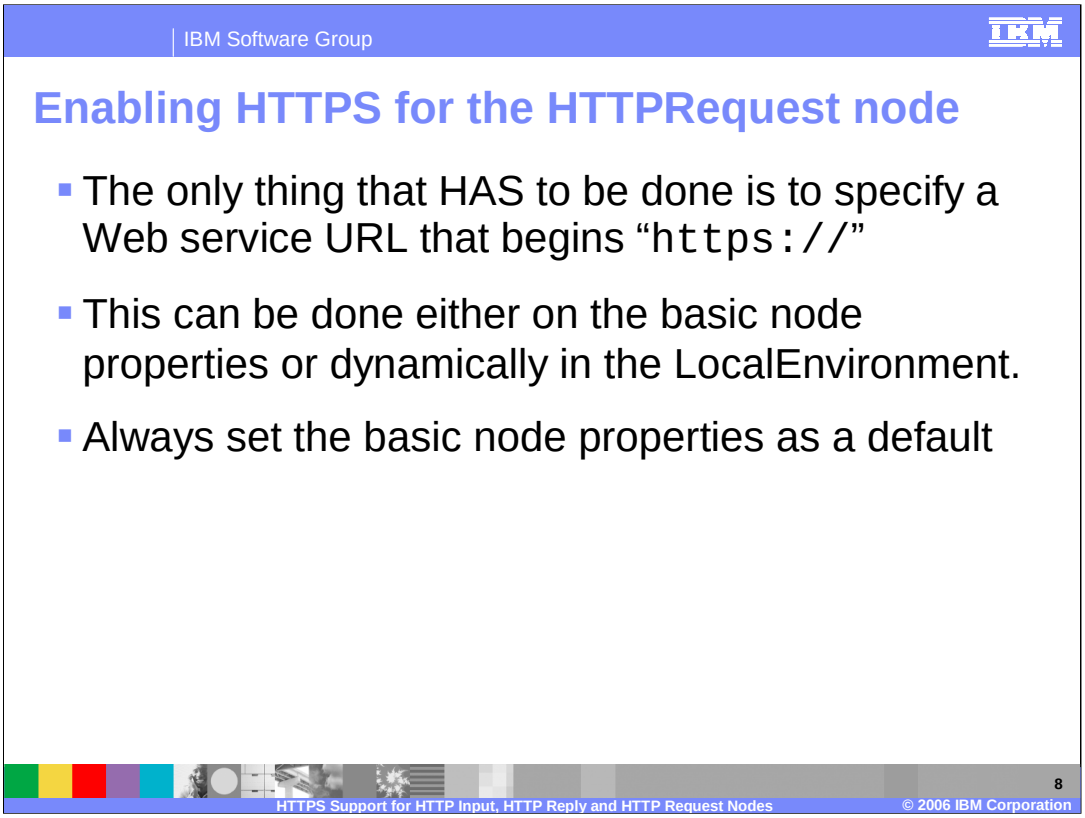


Next, HTTPS enablement.

HTTPRequest node basic properties



When using the HTTP Request node, SSL is specified simply by setting the URL to an identifier that begins ‘HTTPS’. This is exactly the same process as using a web browser to connect over HTTPS. The HTTP or HTTPS address can be specified directly on the HTTPRequest node basic properties (Web service URL).



The HTTP or HTTPS address can be specified directly on the HTTPRequest node. Alternatively, it can be dynamically changed on a per connection basis.

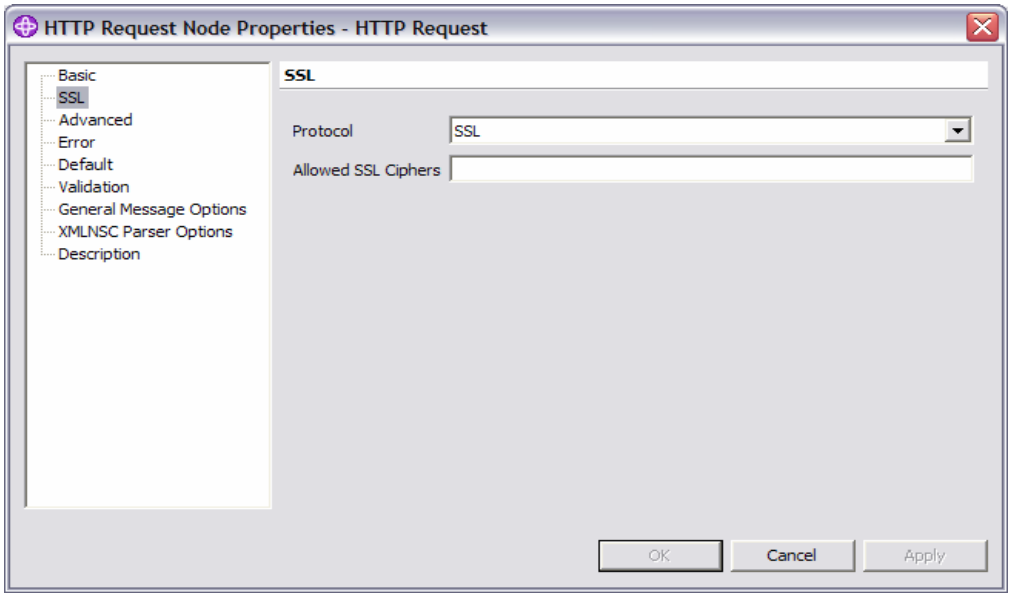
The HTTPRequest node determines the URL for the Web service to which it sends a request. You must set one of the following three options; the node checks these in the order shown (that is, the first always overrides the second, the second overrides the third):

1. X-Original-HTTP-URL in the HTTPRequest header in the input message
2. LocalEnvironment.Destination.HTTP.RequestURL in the input message
3. The *Web Service URL* property (see previous slide)

The first two options provide you with dynamic methods to set a URL for each input message as it passes through the message flow. If you want to use either of these options, you must include a Compute node in the message flow before the HTTPRequest node to create and initialize the required value.

The third option provides a value that is fixed for every message received in this node. You must set this property to contain a default setting that is used if the other fields have not been created, or contain a null value.

HTTPRequest node SSL properties



On the properties of the HTTPRequest node, there is a new tab called SSL. This tab is where all parameters related to SSL are located. There are two primary options on this tab.

HTTPRequest node property SSL options

- SSL Tab to set all SSL options
- SSL type settings are made on the “Protocol” dropdown:
  - SSL – Tries SSLv3 first, allows fallback to SSLv2
  - SSLv3 – Tries SSLv3 only
  - TLS – Tries Transport Layer Security only
- “Allowed SSL Ciphers” text entry field.
  - Default of empty means all broker JVM supported ciphers.
  - One or more explicit ciphers in the list means only these specified ciphers.

The first is an option called ‘Protocol’ which is presented as a drop-down combo box. This allows you to select which version of SSL to use. This can be SSL, SSLv3 or TLS. The default value is SSLv3.

If you specify SSL, the broker will try to connect using the SSLv3 protocol first, but allows the handshake fall back to the SSLv2 protocol where the SSLv2 protocol is supported by the underlying JSSE provider.

If you specify SSLv3, the broker will attempt to connect with the SSLv3 protocol only. Fall back to SSLv2 is not allowed.

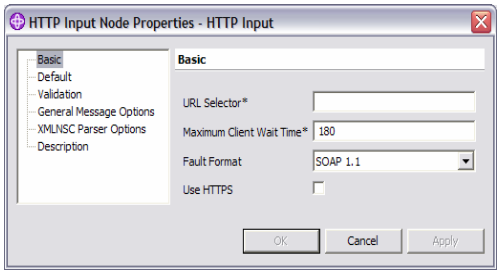
If you specify TLS, the broker will attempt just a Transport Layer Security connection.

Note that both ends of an SSL connection must agree on the protocol to use, so the chosen protocol must be one that the remote server can accept. Ensure that the HTTP provider the flow is connecting to is able to support the type of security the HTTPRequest node is using.

A second option in the SSL properties tab is the text box for ‘Allowed SSL Ciphers’. This setting allows you to specify a single cipher or a list of ciphers thus limiting the ciphers used by the connection. If nothing is provided in this box, all of the provided ciphers installed within the JVM will be used. Each JVM can have additional ciphers. Using this field, other SSL ciphers can be added, in addition to the standard ones provided with the product. Or, it is possible to limit the ciphers to ones that are acceptable in your own environment.

HTTPInput and HTTPReply nodes

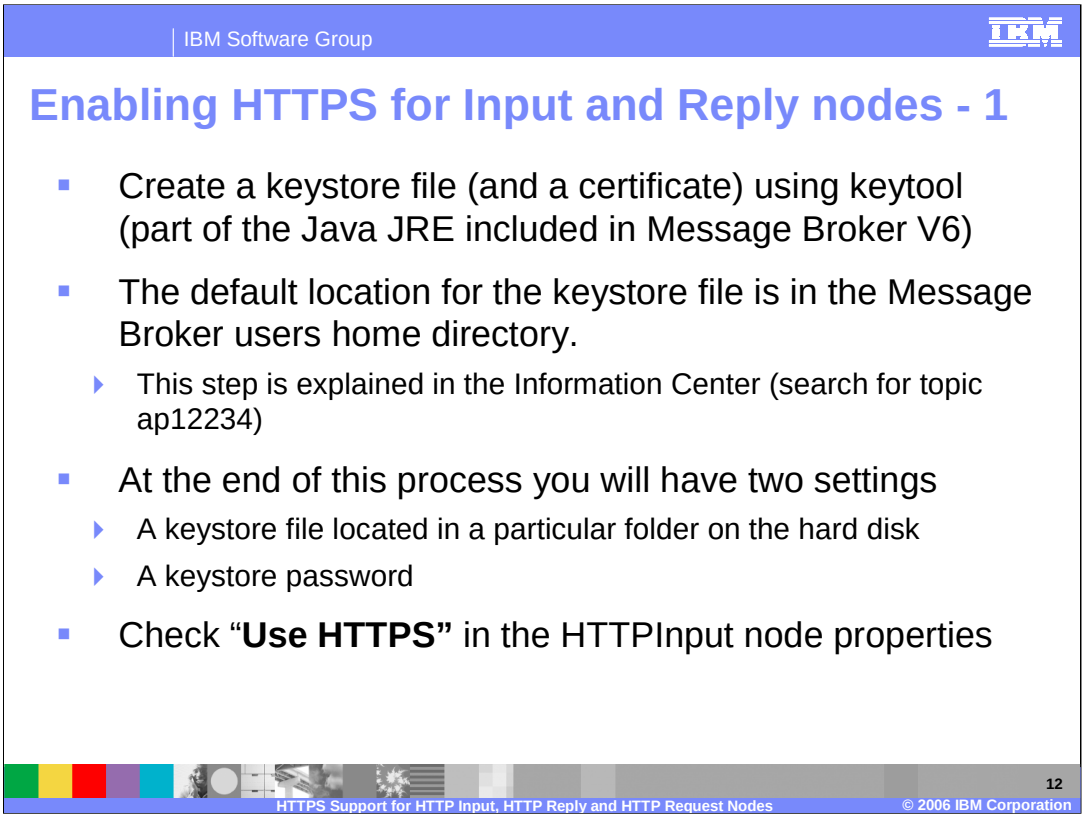
- HTTPInput node has a checkbox under basic properties to indicate: “Use HTTPS”



- HTTPInput / HTTPReply are controlled by mqsichangeproperties

If the HTTPInput node is to accept secure HTTP, select the *Use HTTPS* check box.

Further properties can be set by mqsichangeproperties command; this is covered in the following slides. These properties are modified by broker systems administrators.

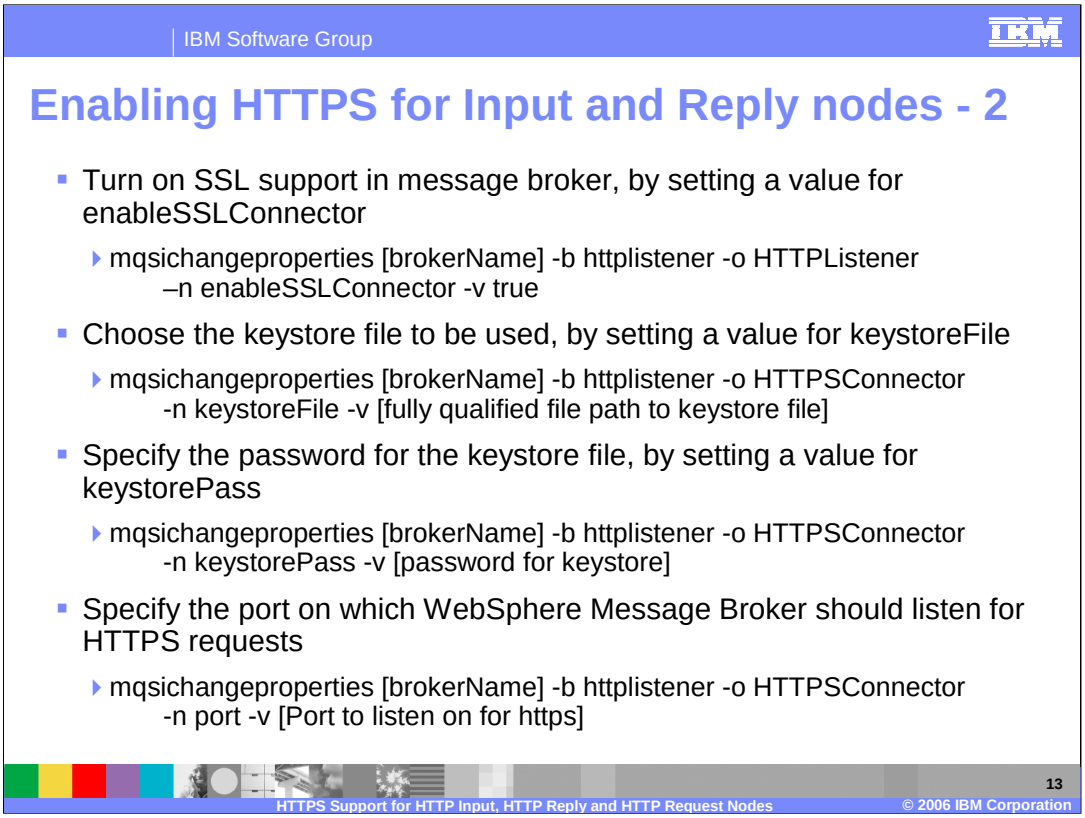


The next 4 slides describe the configuration that is required to enable SSL support for your broker environment.

When configuring HTTPS for Input and Reply nodes, the first thing to do is to create an SSL keystore file and if necessary, an SSL certificate. WebSphere Message Broker includes a Java Runtime Environment (or JRE) that supplies a keystore manipulation program, which is called keytool. Other tools are available to manage certificate stores, but these must work with the Java certificate store format. The keystore is a file that contains SSL keys and is located in a particular directory on the machine. By default, that directory is home directory of the user that is running the broker.

This process is described in the Message Broker Information Center. Search for topic "ap12234". This will describe how to set up a certificate and use it to load the keystore file. It also discusses how to set the password for the keystore file.

Once this has been done, each message flow that wants to use this should then check the 'Use HTTPS' box on the input node.



This slide describes the SSL options that you can change to enable the broker properties.

These options are set by using the “mqsichangeproperties” command. The first option, which enables SSL for the HTTP listener, is mandatory. This is set by using a new value on “-b” parameter, which is “httplistener”. This should be followed by the object parameter “-o” and its value HTTPListener. Note the uppercase letters.

Finally, the property name parameter, “-n”, is followed by the name of the property being set, and “-v” is the new value of this property.

In this case, -n is set to “enableSSLConnector”, and “-v” is set to true.

Note that all of these values are simply transmitted directly to the underlying Tomcat HTTP server.

Further options can also be set with ‘mqsichangeproperties’. The key ones are listed here.

First, you can change the location of the keystore file. The default location is the users home directory. You can do this by using the –n keystoreFile property.

Next, you can specify the password for the keystore file, which was created with the keytool in the previous slide. The default value for this is “changeit”, which is a standard value provided with the Tomcat HTTP server.

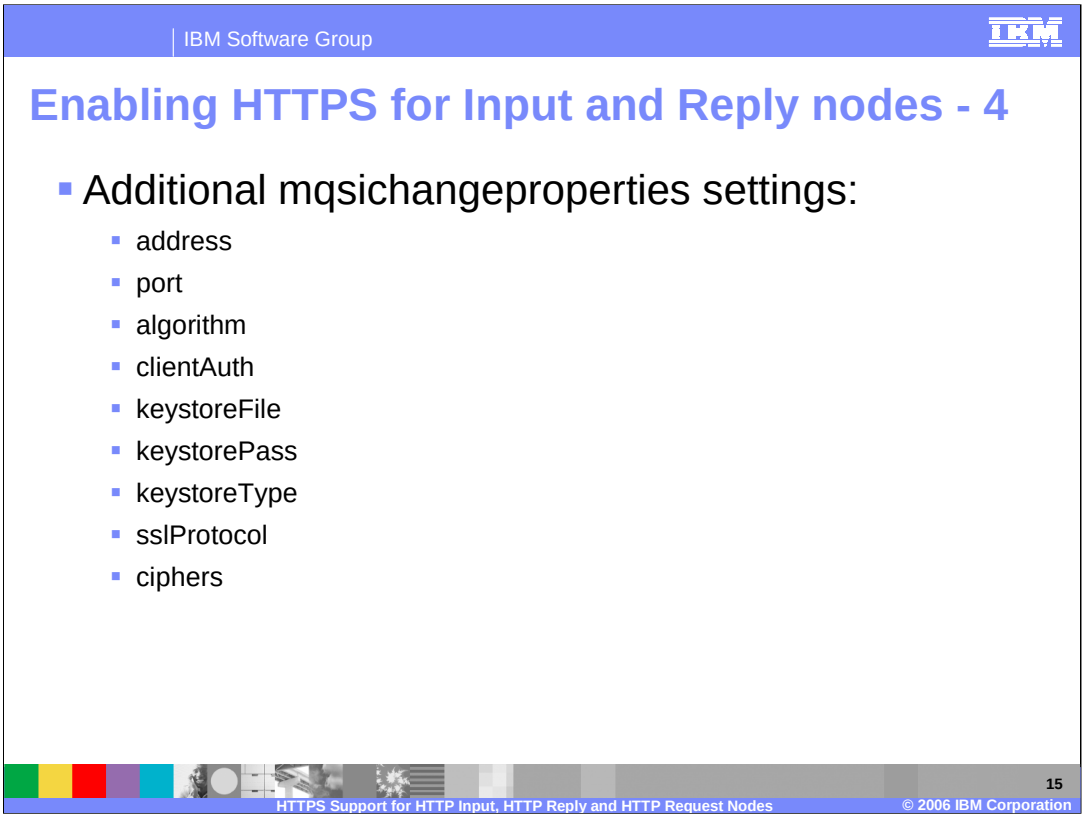
And finally, you can specify the port on which you will be listening for HTTPS settings. In

Enabling HTTPS for Input and Reply nodes - 3

- If the following are true…
  - At least one HTTPInput node has checked Use HTTPS
  - The enableSSLConnector property has been set to true
- … the HTTPListener process will start a connector listening on the specified port for inbound HTTPS connections (default 7083) and a BIP3132 message will be seen in Event/Sys log to confirm this.
  - But the keystore file will be accessed with the keystore password at this time (so must be available and accessible).

The HTTP Listener will start an SSL connector if the ‘Use HTTPS’ property on the HTTPInput node has been indicated and the ‘enableSSLConnector’ property has been set to true. The default port number for HTTPS connections is 7083, and when this is started a BIP3132 message will be seen in the event log or the system log which will say “Connector started for HTTPS”.

Message Broker Version 6 has been enhanced to display the type of connection it is listening for, and to determine if it is an SSL connection. The broker will attempt to access keystore file with the keystore password. A success message is written in the event log if the keystore file has been accessed correctly. An exception indicates a wrong password or a corrupt keystore file. Messages are output to the event log to help diagnose the cause of the error.



There are a number of other properties that can be set using mqsichangeproperties. Note that all of these values are simply passed straight through to the embedded Tomcat HTTP server. The Message Broker does not actually change these values in any way.

All of these properties are described in the “mqsichangeproperties” section of the Information Center. Use “mqsichangeproperties” or “an09140” to search the documentation.

Section

Summary and references

In summary,

Summary

- HTTP SSL support available in broker V6
  - HTTPInput node
  - HTTPReply node
  - HTTPRequest node
- Property settings needed to enable SSL
- SSL configuration for broker
  - mqsichangeproperties command

This session discussed the changes made to the HTTP nodes in Message Broker Version 6. These focused on the SSL support for HTTPRequest, HTTPInput and HTTPReply nodes. There are property settings needed to enable SSL on these nodes; and configuration using mqsichangeproperties command is required to enable the broker to use SSL.

Information Center topics

- “Configuring HTTPInput and HTTPReply nodes to use SSL (HTTPS)” topic: ap12234_
- “Configuring an HTTPRequest node to use SSL (HTTPS)” topic: ap12235_
- “mqsichangeproperties command” topic: an09140_
- “mqsireportproperties command” topic: an09150_
- The topics for the three nodes themselves.

The Information Center provides a number of useful topics covering the SSL area. You can either search for the topics directly, or use the topic identifiers listed on this slide to go straight to the particular topic.

References

- WebSphere Message Broker library:

http://www-306.ibm.com/software/integration/wbimessagebroker/library/

- WebSphere Message Broker Information Center:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp

- Working with HTTP flows

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac20450_.htm

- Non-web-service interface to new Web service

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac34560_.htm

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM WebSphere

Java, JRE, JVM, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.