

Course Guide

Web Services Support in IBM DataPower V7.5

Course code WE754 / ZE754 ERC 1.0



May 2017 edition

Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

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 and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

© Copyright International Business Machines Corporation 2017.

This document may not be reproduced in whole or in part without the prior written permission of IBM.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

| | |
|---|-------------|
| Trademarks | v |
| Course description | vi |
| Agenda | viii |
| Unit 1. XML and web services security overview | 1-1 |
| How to check online for course material updates | 1-2 |
| Unit objectives | 1-3 |
| Review of basic security terminology | 1-4 |
| Web services security | 1-6 |
| Components of WS-Security | 1-7 |
| Specifying security in SOAP messages | 1-9 |
| Scenario 1: Ensure confidentiality with XML encryption | 1-10 |
| XML encryption and WS-Security | 1-11 |
| DataPower support for XML encryption | 1-12 |
| Encrypt action | 1-13 |
| Decrypt action | 1-15 |
| Field-level encryption and decryption | 1-16 |
| XPath tool | 1-17 |
| Sample encrypted SOAP message | 1-18 |
| Scenario 2: Ensure integrity with XML signatures | 1-19 |
| DataPower support for XML signature | 1-20 |
| Sign action | 1-21 |
| Verify action | 1-23 |
| Verify action: Advanced tab | 1-24 |
| Field-level message signature and verification | 1-25 |
| Sample signed SOAP message | 1-26 |
| Summary of security and keys | 1-27 |
| Unit summary | 1-28 |
| Review questions | 1-29 |
| Review answers | 1-30 |
| Exercise: Web service encryption and digital signatures | 1-31 |
| Exercise objectives | 1-32 |
| Exercise overview | 1-33 |
| Completed exercise | 1-34 |
| Unit 2. Web service proxy service | 2-1 |
| Unit objectives | 2-2 |
| Web service proxy overview | 2-3 |
| Conceptual architecture of a web service proxy | 2-4 |
| Web service virtualization | 2-5 |
| Web service proxy benefits | 2-6 |
| Web service proxy configuration tabs (1 of 2) | 2-7 |
| Web service proxy configuration tabs (2 of 2) | 2-8 |
| Web service proxy basic configuration steps | 2-9 |
| Step 1: Obtain WSDL document | 2-10 |
| WSDL structure | 2-11 |
| Step 2: Creating a web service proxy | 2-12 |
| An alternative: Web service proxy object editor | 2-13 |

| | |
|--|------------|
| Step 3: Add WSDL document to web service proxy | 2-14 |
| Step 4: Configure WSDL endpoint | 2-16 |
| Step 5: Configure local endpoint handler | 2-18 |
| Step 6: Add the WSDL to the service | 2-19 |
| Initial WSDL completed | 2-20 |
| Other ways to get a WSDL | 2-21 |
| View WSDL services | 2-22 |
| Modifying the location in the client WSDL | 2-23 |
| Step 7: Configuring web service proxy policy | 2-24 |
| Configure web service proxy policy rule | 2-25 |
| Adding a rule | 2-26 |
| Default validation (user policies) | 2-27 |
| Create reusable rule | 2-28 |
| Advanced web service proxy configuration | 2-29 |
| WS-Policy | 2-30 |
| Conformance policy | 2-31 |
| Conformance policy object | 2-32 |
| Service priority | 2-33 |
| Proxy settings (1 of 4) | 2-34 |
| Proxy settings (2 of 4) | 2-35 |
| Proxy settings (3 of 4) | 2-37 |
| Proxy settings (4 of 4) | 2-38 |
| Web service proxy: SLM Policy tab | 2-39 |
| WSDL cache policy | 2-40 |
| Troubleshooting a web service proxy | 2-41 |
| Unit summary | 2-42 |
| Review questions | 2-43 |
| Review answers (1 of 2) | 2-44 |
| Review answers (2 of 2) | 2-45 |
| Exercise: Configuring a web service proxy | 2-46 |
| Exercise objectives | 2-47 |
| Exercise overview | 2-48 |
| Unit 3. Course summary | 3-1 |
| Unit objectives | 3-2 |
| Course objectives | 3-3 |
| Lab exercise solutions | 3-4 |
| Curriculum relationship | 3-5 |
| To learn more on the subject | 3-6 |
| Enhance your learning with IBM resources | 3-7 |
| Unit summary | 3-8 |
| Course completion | 3-9 |
| Appendix A. List of abbreviations | A-1 |

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

DataPower®

DB™

DB2®

IMS™

Notes®

Rational®

Tivoli®

WebSphere®

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

VMware and the VMware “boxes” logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the “Marks”) of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.

Course description

Web Services Support in IBM DataPower V7.5

Duration: 0.5 days

Purpose

This course teaches you the developer skills that are required to configure WS-Security features and use WSDLs to generate web service proxy services on IBM DataPower Gateway V7.5.

The initial focus of DataPower was on XML and web services workloads, and it continues that support today. As part of that support, you can use DataPower to digitally sign and encrypt a message, validate a message's signature, and decrypt an encrypted message. These functions are delivered as processing actions within the service policy. In this course, you learn about these message integrity and confidentiality capabilities, and practice them in a lab exercise.

Web service operations typically work with WSDL files. DataPower generates a web service proxy directly from a WSDL, bypassing much of the basic configuration of a service to support that WSDL. This course also covers this service type of web service proxy, and gives you an opportunity to work with one in a lab exercise.

Audience

This course is designed for integration developers who configure XML and SOAP-based services on IBM DataPower Gateways.

Prerequisites

Before taking this course, you should successfully complete course *Essentials of Service Development for IBM DataPower Gateway V7.5 (WE751G)* or *Essentials of Service Development for IBM DataPower Gateway V7.5 (ZE751G)*. You should also be familiar with basic XML, SOAP, and WS-Security concepts.

Objectives

- Use the XML Signature and XML Encryption capabilities within DataPower to support WS-Security functions in your DataPower services
- Configure a web service proxy by using a WSDL file to proxy web services-based applications
- Configure the service policy of a web service proxy that supports different behaviors at the various levels of the WSDL

Curriculum relationship

This course is a follow-on to the initial course:

- WE751: *Essentials of Service Development for IBM DataPower Gateway V7.5*

This course is a peer to the following courses:

- WE752: *Supporting REST and JOSE in IBM DataPower Gateway V7.5*
- WE753: *AAA, OAuth, and OIDC in IBM DataPower V7.5*

Agenda

**Note**

The following unit and exercise durations are estimates, and might not reflect every class experience.

Day 1

- (00:15) Course introduction
- (00:45) Unit 1. XML and web services security overview
- (01:00) Exercise 1. Web service encryption and digital signatures
- (01:00) Unit 2. Web service proxy service
- (01:00) Exercise 2. Configuring a web service proxy
- (00:15) Unit 3. Course summary

Unit 1. XML and web services security overview

Estimated time

00:45

Overview

This unit briefly describes the features of the web services security specification. This specification uses XML encryption and XML signatures to provide message-level security to ensure message confidentiality and integrity. The primary focus of the unit is to describe how the DataPower gateway supports XML encryption and XML signatures within DataPower services.

How you will check your progress

- Checkpoint
- Hands-on exercise

References

IBM Knowledge Center documentation for IBM DataPower Gateway:

http://www.ibm.com/support/knowledgecenter/SS9H2Y_7.5.0

How to check online for course material updates



Note: If your classroom does not have internet access, ask your instructor for more information.

Instructions

1. Enter this URL in your browser:
ibm.biz/CloudEduCourses
2. Find the product category for your course, and click the link to view all products and courses.
3. Find your course in the course list and then click the link.
4. The wiki page displays information for the course. If the course has a corrections document, this page is where it is found.
5. If you want to download an attachment, such as a course corrections document, click the **Attachments** tab at the bottom of the page.

| | | | |
|--------------|--------------|------------------------|-------|
| Comments (0) | Versions (1) | Attachments (1) | About |
|--------------|--------------|------------------------|-------|

6. To save the file to your computer, click the document link and follow the prompts.

Unit objectives

- Describe the features of the WS-Security specification
- Enable message confidentiality by using XML encryption
- Provide message integrity by using XML signatures

Review of basic security terminology

- **Authentication** verifies the identity of a client
- **Authorization** decides a client's level of access to a protected resource
- **Integrity** ensures that a message is not modified while in transit
- **Confidentiality** ensures that the contents of a message are kept secret
- **Auditing** maintains records to hold clients accountable to their actions
- **Nonrepudiation** is the condition where you are assured that a particular message is associated with a particular individual



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-3. Review of basic security terminology

Authentication is the act of verifying that the identity asserted by the client is valid. Normally, a security token that is attached to the message makes a claim about the client identity. Plaintext user name and password tokens, X.509 certificates, and Kerberos tickets are all examples of identity claims.

Authorization is the process of deciding whether a client has access to a protected resource. This process also determines the level of access that the server grants the client. In most cases, the authorization decision requires that the client identity is known and verified. That is, authorization occurs after authentication.

Integrity, also known as **data integrity**, makes sure that a message is not altered or tampered with while it travels between the client and the server. Digital signatures and hash codes can prove whether a message was modified in transit.

Confidentiality ensures that only authorized parties have access to protected resources. The effect of confidentiality is to keep private data or resources secret. This quality is often implemented through the encryption of data, in which only authorized parties have the means of making obscured data into legible information.

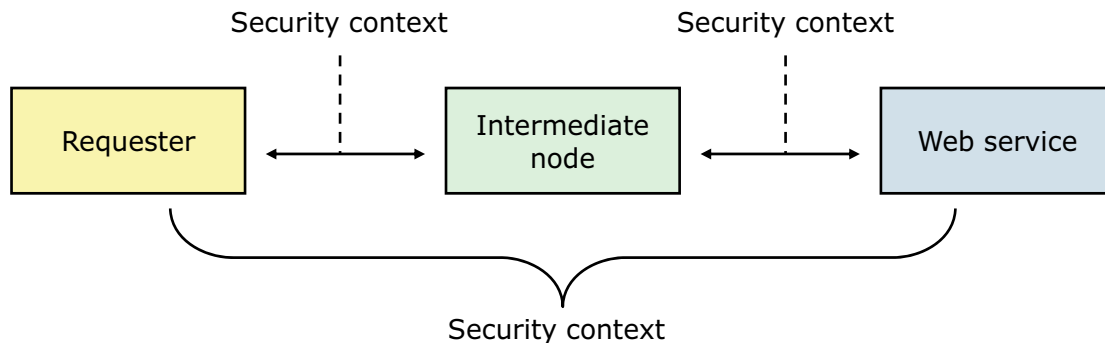
Auditing is the process of maintaining irrefutable records for holding clients accountable to their actions. Signed security logs provide one way to audit a security system. The concept of nonrepudiation is tied closely to auditing. It is the ability of one party of the communication to prove

that the other party received its message. **Nonrepudiation** is often split into two concepts: *nonrepudiation of origin* proves that one party sent a message, while *nonrepudiation of receipt* proves that one party received a message.

Verifying the digital signature and the expiration date on the message enforces nonrepudiation of origin. Nonrepudiation of receipt depends on the software environment.

Web services security

- Web Services Security (WS-Security) provides a standard, platform-independent way for specifying *message-level* security information
- Flexible set of mechanisms for using a range of security protocols:
 - Does *not* define a set of security protocols
 - Provides *end-to-end* security



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-4. Web services security

WS-Security does not describe specific security protocols. This model can use different security mechanisms, and can be configured to match the requirements of new ones as they are developed. By separating the security constraints from the actual implementation, developers can change security technologies without needing to adopt another web services security specification.

Each arrow between two boxes shows a point-to-point security context. Transport level security, such as SSL/TLS, provides a security context that persists only from one intermediate node to another.

The curved line that spans multiple boxes is an example of end-to-end security. WS-Security provides this security context.

WS-Security provides message-level security. SSL/TLS secures the entire HTTP request, and is at the transport layer. WS-Security allows security to be applied to specific message parts of the request payload.

Components of WS-Security

- Associates security tokens with a message
 - User name token profile
 - X.509 token profile
 - Kerberos token profile
 - SAML token profile: Security Assertion Markup Language
 - REL token profile: Rights Expression Language
- Confidentiality (XML encryption)
 - Process for encrypting data and representing the result in XML
- Integrity (XML signature)
 - Digitally sign the SOAP XML document, providing integrity and signer authentication
- XML canonicalization
 - Normalizes XML document
 - Ensures that two semantically equivalent XML documents contain the same octet stream

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-5. Components of WS-Security

An XML digital signature is based on the W3C recommendation specification for XML-signature syntax and processing. For more information, see: <http://www.w3.org/TR/xmlsig-core/>

XML encryption is based on the W3C recommendation for XML encryption syntax and processing. For more information, see: <http://www.w3.org/TR/xmlenc-core/>

The security token profiles that are listed are for WS-Security V1.1. For links to the list of specifications, see: <http://www.oasis-open.org/specs/index.php#wssv1.0>

The standards that a particular version of DataPower supports are listed in the “Release Notes” section in the DataPower Knowledge Center.

XML canonicalization ensures that two pieces of information with the same semantic meaning have an equivalent canonical form.

XML data is represented as an octet stream (that is, bytes of data). Two semantically equivalent XML documents can generate different digest values because they are represented as different octet streams.

For example, the following XML tags are semantically the same:

```
<customer id="13579" class="2"/>
<customer class="2" id="13579"/>
```

These two tags would represent different octet streams. One canonicalization method would maintain a similar ordering of attributes. The W3C Canonical XML standard ensures that the same octet stream is used by applying canonicalization algorithms.

Specifying security in SOAP messages

- Attach security-related information to SOAP messages in the **<wsse:Security>** header element

```
<env:Envelope
  xmlns:env="http://www.w3.org/2001/12/soap-envelope">
  <env:Header>

    <wsse:Security
      env:actor="http://www.example.com/secManager"
      env:mustUnderstand="1"
      xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      <!-- WS-Security header here -->
    </wsse:Security>

  </env:Header>
  <env:Body>
    <!-- SOAP message body here -->
  </env:Body>
</env:Envelope>
```

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-6. Specifying security in SOAP messages

A SOAP header is inserted into the message by the DataPower gateway when any WS-Security is applied to a message.

The `actor` and `mustUnderstand` are special attributes that the SOAP specification defines. The `actor` attribute contains a URL of the targeted recipient for the SOAP header. The `mustUnderstand` attribute is used to specify that the tags in the header must be understood; otherwise, a fault is thrown.

Scenario 1: Ensure confidentiality with XML encryption

- Use XML encryption to keep messages secret
 - Encrypt with the recipient certificate: Only the recipient can decrypt with associated private key
 - XML encryption specification does not describe how to create or exchange keys
- XML encryption supports:
 - Message encryption at different levels of granularity, from a single element value to a tree of XML elements
 - Secure message exchange between more than two parties: A message might pass through intermediate handlers that read only the parts of the message relevant to them

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-7. Scenario 1: Ensure confidentiality with XML encryption

By encrypting message content, the privacy of the content becomes decoupled from the transport mechanism. For example, messages sent over an SSL connection are encrypted. They are thus provided with some degree of privacy, but no further privacy is provided after the message exits the SSL connection. By encrypting the content of the message, the message can travel across transport boundaries, such as HTTP and WebSphere MQ, and remain private.

The `<Envelope>`, `<Header>`, and `<Body>` elements cannot be encrypted.

Message encryption can occur at different levels of granularity. For an online purchase, you can protect the credit card number by encrypting the credit card element value. To hide the fact that the transaction uses credit cards, you can encrypt all elements that are related to the credit card. To keep the entire transaction secret, you can encrypt the entire transaction XML tree.

XML encryption and WS-Security

- The XML encryption standard uses symmetric encryption algorithms for the actual data encryption
 - The symmetric key is passed with the message
 - The public key in the recipient's certificate is used to encrypt the symmetric key before transmission
 - Only the recipient has the private key to decrypt the symmetric key
 - Encrypted data is inside `<enc:EncryptedData>` element
- The WS-Security standard uses XML encryption
 - Places encryption metadata in SOAP header `<wsse:Security>`
 - Supports passing the symmetric key in multiple ways

DataPower support for XML encryption

- Applies XML encryption to a message by defining a processing rule that contains:
 - **Encrypt** action: Performs full or field-level message encryption
 - **Decrypt** action: Performs full or field-level message decryption
- Acts as *client/sender* to *encrypt* a message sent to a server



- Acts as a *server/recipient* to *decrypt* a message that the client sent



Encrypt action

The **Encrypt** action performs full or field-level encryption

- **Envelope Method:** Controls placement of generated security elements
- **Message and Attachment Handling:** Encrypt message, attachment, or both
- **Encryption Key Type:** How the symmetric key is protected
- **Use Dynamically Configured Recipient Certificate:** Uses passed certificate, if it exists
- **One Ephemeral Key:** Causes all encryption in this step to use the same ephemeral key
- **Recipient Certificate:** The certificate that is used to encrypt the encryption key

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-10. Encrypt action

The **Advanced** choice for Envelope Method and Message Type is not selectable.

The Standard choice indicates whether XML or JSON web security is used. If “JSON Web Security” is selected, most of the following fields change.

An ephemeral key is a key that is generated each time encryption occurs. Basically, it is the symmetric key that is used for encryption.

The DataPower device supports the following encryption schemas:

- Standard XML encryption was originally designed to handle any XML message, including those messages that are not formatted to the SOAP specification. It puts the signature and key information in the body of the message, thus adding more elements to the body of the message.
- The WS-Security standard puts the signature and key information in the WS-Security header of the SOAP message. This standard does not add elements to the body of the message. Therefore, it does not violate the underlying schema.

The DataPower gateway supports both methods of encryption. The gateway can use either standard for full message or partial encryption.

The following message types are supported:

- **SOAP message:** An encrypted SOAP document
- **Raw XML document:** An encrypted XML document (it cannot be used with WSSec encryption)
- **Selected elements (field-level):** A partially encrypted SOAP document

The following options are in the **Message and Attachment Handling** menu:

- **Attachments only:** Only the attachments of the message are encrypted.
- **Message only:** Only the message (root part) is encrypted.
- **Message and attachments:** Message (root part) and attachments are encrypted.

The encryption key type specifies how the symmetric encryption key is protected. Depending on the selection, the fields in the page might change:

- **Use Ephemeral Key Transported by Asymmetric Algorithm:** The X509 key-cert pair transports the ephemeral key with an asymmetric algorithm.
- **Use Symmetric Key Directly:** A security token protects the session key.
- **Use Ephemeral Key Wrapped by a Symmetric Key:** The ephemeral key is encrypted by a symmetric key from a security token.

If **Use Dynamically Configured Recipient Certificate** is set to **on**, the Encrypt action uses a certificate that is used in a previous Verify action. This option supports use of the certificate in a Verify action for the request message as the encrypting certificate in an Encrypt action in the response.

Decrypt action

- The **Decrypt** action performs full or field-level decryption
 - **Message Type:** Specifies how to decrypt the message
 - **Decrypt Key:** Private key object that is used to decrypt

The screenshot shows the 'Decrypt' action configuration in the 'Advanced' tab. The 'Input' field is set to 'INPUT'. Under the 'Options' section, the 'Standard' radio button is selected, with 'XML Security' chosen. The 'Message Type' section has 'Entire Message/Document' selected. The 'Asynchronous' section has 'off' selected. The 'Decrypt Key' is set to 'KeysFromDP'. The 'Output' field is set to 'dpvar_1'. There are also buttons for '+', '...', and a 'Save' button with a checkmark.

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-11. Decrypt action

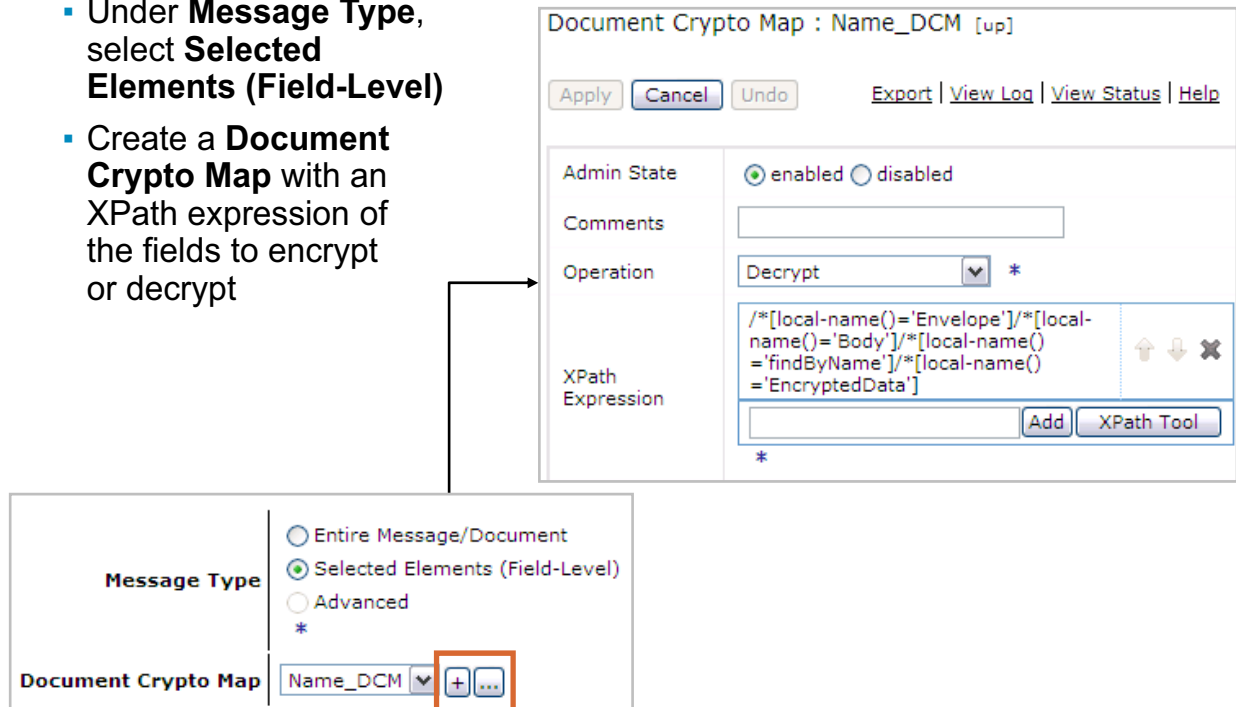
The Standard choice indicates whether XML or JSON web security is used. If “JSON Web Security” is selected, most of the following fields change.

On the **Advanced** tab, you can override the stylesheet that is used to decrypt. The default file that is used is `store:///decrypt.xsl`.

The web service proxy service type supports decryption at the proxy or service level by specifying a crypto key object instead of a **Decrypt** action. It is required when the entire message is encrypted and the web service proxy cannot determine the operation to invoke. This option is covered in the web service proxy unit and exercise.

Field-level encryption and decryption

- Performs field-level encryption and decryption on messages
 - Under **Message Type**, select **Selected Elements (Field-Level)**
 - Create a **Document Crypto Map** with an XPath expression of the fields to encrypt or decrypt



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-12. Field-level encryption and decryption

The XPath expression can be created from an XML file by selecting the elements to encrypt or decrypt. The XPath expression for field-level decryption is different from the XPath expression for encrypting the same field. Encryption occurs on an element in the original message, for example, `<name>`. When it is time to decrypt, the field is no longer known as `<name>`, but as something else, such as `<EncryptedData>`. Thus, the XPath expression to get to the apparently identical element differs depending on whether you are encrypting the original field or decrypting the encrypted field.

XPath tool

- In the document crypto map, click **XPath Tool** to create an XPath expression by using an XML file
 - **URL of Sample XML Document:** Upload or select an XML document
 - **Namespace Handling:** How the XPath statement matches namespace declarations
 - **XPath:** Generated XPath statement

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-13. XPath tool

The content of the selected XML file is omitted from the slide and is shown below the three buttons (**Refresh**, **Done**, and **Cancel**). Click the elements in the XML file to generate an XPath expression.

The three options for namespace handling are:

- **local:** This option compares only the local name (element name), ignoring the namespace.
- **prefix:** This option compares the qualified name, including the namespace prefix. It can be used when the mapping from the namespace prefix to the URI is specified on the **Namespace Mappings** tab on an object configuration page.
- **uri:** This option compares the local name and namespace URI.

Sample encrypted SOAP message

The diagram shows an XML SOAP message structure. The `<wsse:Security>` element in the header is annotated with a yellow box labeled "Key that is used to encrypt message". The `<q0:findByName>` element in the body is annotated with a yellow box labeled "Field-level XML encryption".

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  <soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1">
      <xenc:EncryptedKey>
        ...
      </xenc:EncryptedKey>
    </wsse:Security>
  </soapenv:Header>
  <soapenv:Body>
    <q0:findByName>
      <xenc:EncryptedData>
        <EncryptionMethod />
        <CipherData>
          <CipherValue>
            ...
          </CipherValue>
        </CipherData>
      </xenc:EncryptedData>
    </q0:findByName>
  </soapenv:Body>
</soapenv:Envelope>
```

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-14. Sample encrypted SOAP message

This example message does field-level encryption on the child elements of the `<q0:findByName>` element. If full-message encryption is applied, then this element would also be encrypted.

Namespace declarations were removed in this example.

When XML encryption is applied to the original SOAP message, a web services security header is inserted into the SOAP header with information about the key that was used to encrypt the message body. In this example, the child element of `<q0:findByName>` is encrypted.

Scenario 2: Ensure integrity with XML signatures

- Sign SOAP message parts to provide message integrity
 - Provide assurance that message is not changed
 - Mismatch between decrypted hash (from signature) and computed hash (from cleartext) indicates that the message is modified
 - Signatures provide a strong indication of identity
 - Only the holder of the private key can create a signature that matches the enclosed certificate (if asymmetric)
- XML signature standard provides a schema for storing digital signature information within XML messages
 - Does not describe how to digest and sign messages
 - Supports symmetric and asymmetric signing key
- Recipient validates the signature by repeating the same steps that are used to generate a digital signature
 - Compute a message digest of the received message
 - Obtain the original message digest by decrypting the signature from the received message by using the certificate that holds the public key
 - Compare the computed message digest against the original message digest

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-15. Scenario 2: Ensure integrity with XML signatures

The XML digital signature (XMLDS) is a joint effort between the World Wide Web Consortium (W3C) and Internet Engineering Task Force (IETF). For more information about signatures, see: <http://www.w3.org/signature>

An XML signature is transport-independent; it can cross multiple transport protocol boundaries.

A message digest is a hash value that is generated by applying a digest algorithm to a message part. A private key is used to generate a digital signature. Depending on the algorithm, either the same private key or a public key is used to verify the signature.

A sender can choose to sign only specific portions of the XML tree rather than the complete document.

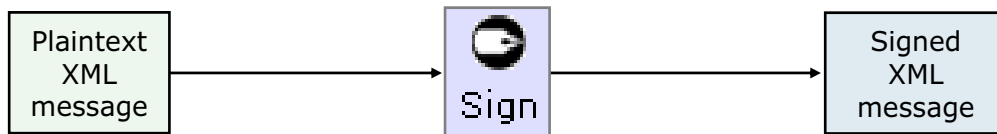
Consider the following example:

```
<transaction-info>
<user-id>jsmith</user-id>
<action>buy</action>
<symbol>IBM</symbol>
</transaction-info>
```

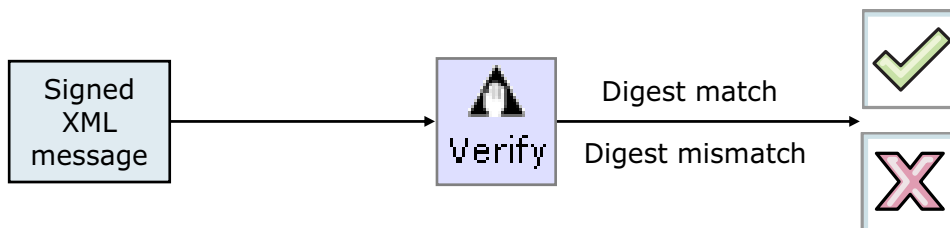
You can sign the value within the symbol element, the entire symbol element (including the value), or a group of elements within transaction-info.

DataPower support for XML signature

- Apply an XML signature to a message by defining a document processing rule that contains:
 - **Sign** action: Digitally signs a message
 - **Verify** action: Verifies the digital signature in a message
- Acts as sender/client to sign message that is sent to the server



- Acts as receiver/recipient to verify the message that the client sent



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-16. DataPower support for XML signature

A client signs a message by using its private key. The message is verified with the client public key by using the client certificate, if asymmetric. The public key is used to verify that the message is associated with the private key that was used to sign the message.

Sign action

- The **Sign** action signs specific elements or the entire message by using a crypto key object
 - **Envelope Method:** Determines placement of the signature in a message
 - **Message Type**
 - **Key:** Crypto key object that is used to sign a message
 - **Certificate:** Crypto certificate object that is associated with the crypto key object

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-17. Sign action

Digital signatures might occur anywhere in a message. The signature can be in either the header or the body of the message, depending on the style that was chosen to sign the message. For non-SOAP XML messages, the signature element might occur anywhere in the message.

The Standard choice indicates whether XML or JSON web security is used. If “JSON Web Security” is selected, most of the following fields change.

The choice of envelope method determines the placement of the XML signature (from DataPower WebGUI documentation):

- **Enveloped Method:** The signature is over the XML content that contains the signature as an element. The content provides the root XML document element (not considered a good idea).
- **Enveloping Method:** The signature is over content that is found within an object element of the signature. The object, or its content, is identified by using a reference through a URI fragment identifier or transform (not considered a good idea).
- **SOAPSec Method:** The signature is included in a SOAP header entry.
- **WSec Method:** The signature is included in a WS-Security security header.

If an envelope method of **WSec Method** and a message type of either **SOAP Message** or **SOAP With Attachments** is selected, then the page shows a **Use Asymmetric Key** option.

If **Use Asymmetric Key** option is:

- **On**, then the signing algorithm shows asymmetric choices
- **Off**, then the signing algorithm shows symmetric HMAC choices

The **Key** object is used to generate the digital signature. The **Certificate** object is included in the signed XML message, but it is not used to sign a message. The receiver of the message uses it to verify the digital signature.

Verify action

- The **Verify** action verifies a digital signature
- **Signature Verification Type**
 - Use asymmetric only, symmetric only, or either
- **Optional Signer Certificate**
 - Used instead of a passed certificate
- **Validation Credential**
 - One or more certificate objects that are used to validate the signer certificate

The screenshot shows the 'Verify' configuration window. It has a left sidebar with labels: 'Standard', 'Asynchronous', 'Signature Verification Type', 'Optional Signer Certificate', and 'Validation Credential'. The main area contains:

- Standard:** Two radio buttons, 'XML Security' (selected) and 'JSON Web Security' (marked with an asterisk).
- Asynchronous:** Two radio buttons, 'on' and 'off' (selected).
- Signature Verification Type:** A dropdown menu showing 'RSA/DSA Signatures' and a 'Save' checkbox.
- Optional Signer Certificate:** An empty text input field.
- Validation Credential:** A dropdown menu showing 'CertFromClient', a '+' button, an ellipsis button, and a 'Save' button with a checkmark.

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-18. Verify action

By default, a digital signature is verified by using the certificate (public key) that is contained in the signature. No additional configuration steps are required.

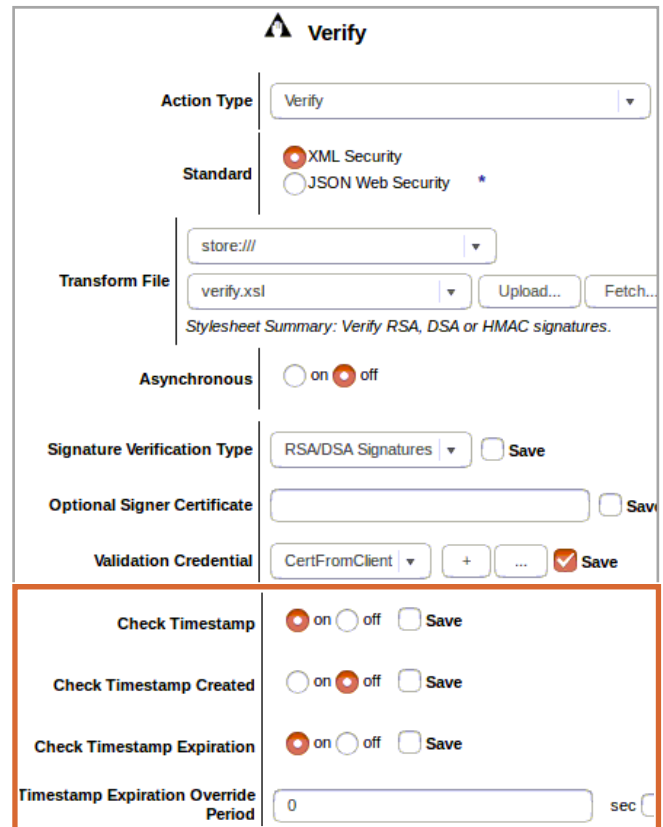
The Standard choice indicates whether XML or JSON web security is used. If “JSON Web Security” is selected, most of the following fields change.

The validation credential object validates the included certificate. If the certificate that is supplied in the signature does not validate against the validation credential object, the signature verification fails.

An alternative certificate can be specified in the **Optional Signer Certificate** field.

Verify action: Advanced tab

- The **Advanced** tab has settings for monitoring the signature timestamp
- **Check Timestamp**
 - Default is **on**, controls other checks
- **Check Timestamp Created**
 - If **on**, verifies that creation is earlier than current time
- **Check Timestamp Expiration**
 - If **on**, verifies that timestamp is not expired
- **Timestamp Expiration Override Period**
 - Seconds to extend expiration



Verify

Action Type: Verify

Standard: ☒ XML Security ☐ JSON Web Security *

Transform File: store:/// verify.xsl Upload... Fetch...

Stylesheet Summary: Verify RSA, DSA or HMAC signatures.

Asynchronous: ☐ on ☒ off

Signature Verification Type: RSA/DSA Signatures Save

Optional Signer Certificate: Save

Validation Credential: CertFromClient + ... Save

Check Timestamp: ☒ on ☐ off Save

Check Timestamp Created: ☐ on ☒ off Save

Check Timestamp Expiration: ☒ on ☐ off Save

Timestamp Expiration Override Period: 0 sec

XML and web services security overview

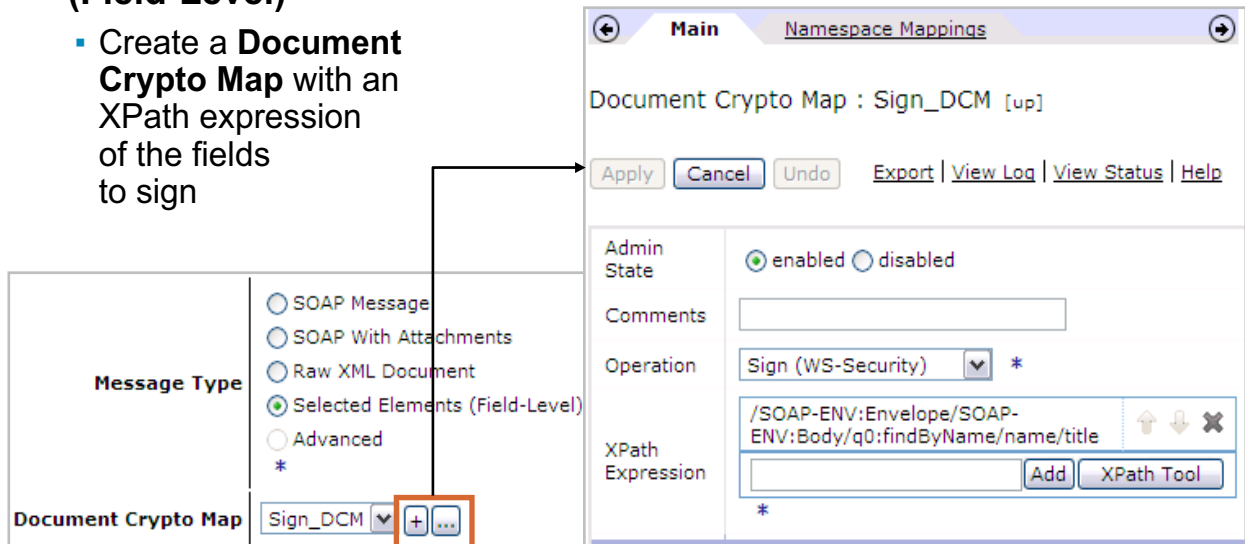
© Copyright IBM Corporation 2017

Figure 1-19. Verify action: Advanced tab

The **Verify** action uses an advanced **Check Timestamp Expiration** property, which is **on** by default. Valid signatures might expire and thus fail verification.

Field-level message signature and verification

- The **Sign** action supports signing of specific elements by using a document crypto map
 - Similar to the **Encrypt** and **Decrypt** actions
- Select **Selected Elements (Field-Level)**
 - Create a **Document Crypto Map** with an XPath expression of the fields to sign



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-20. Field-level message signature and verification

The **Verify** action does not include a **field-level** radio button. In the WSSec envelope method, an ID is inserted into the element of the message that is signed. For example, if the entire message is signed, then the child element of the SOAP body contains the ID attribute. The ID attribute can be used to determine the elements that are signed.

This ID might cause messages to fail schema validation.

The document crypto map object is the same object that the **Encrypt** and **Decrypt** actions use. The only difference is the value in the **Operation** field, which is populated based on the action that you are performing.

Sample signed SOAP message

```
<soapenv:Envelope xmlns:q0="http://east.address.training.ibm.com">
  <soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1">
      <wsse:BinarySecurityToken
        wsu:Id="SecurityToken-abf72a2b-3118-4aa2-98e7-462fa3208f5a">
      </wsse:BinarySecurityToken>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"
        <SignedInfo>
          ...
        </SignedInfo>
        <SignatureValue>rFHK9ixdAm6Mq0</SignatureValue>
        <KeyInfo>
          <wsse:SecurityTokenReference xmlns="">
            ...
          </wsse:SecurityTokenReference>
        </KeyInfo>
      </Signature>
    </wsse:Security>
  </soapenv:Header>
  <soapenv:Body wsu:Id="Body-441d82cd-1613-4905-8aab-ebc7a91d8121">
    <q0:findByLocation>
      <city/>
      <state>NY</state>
    </q0:findByLocation>
  </soapenv:Body>
</soapenv:Envelope>
```

XML
signature

XML and web services security overview

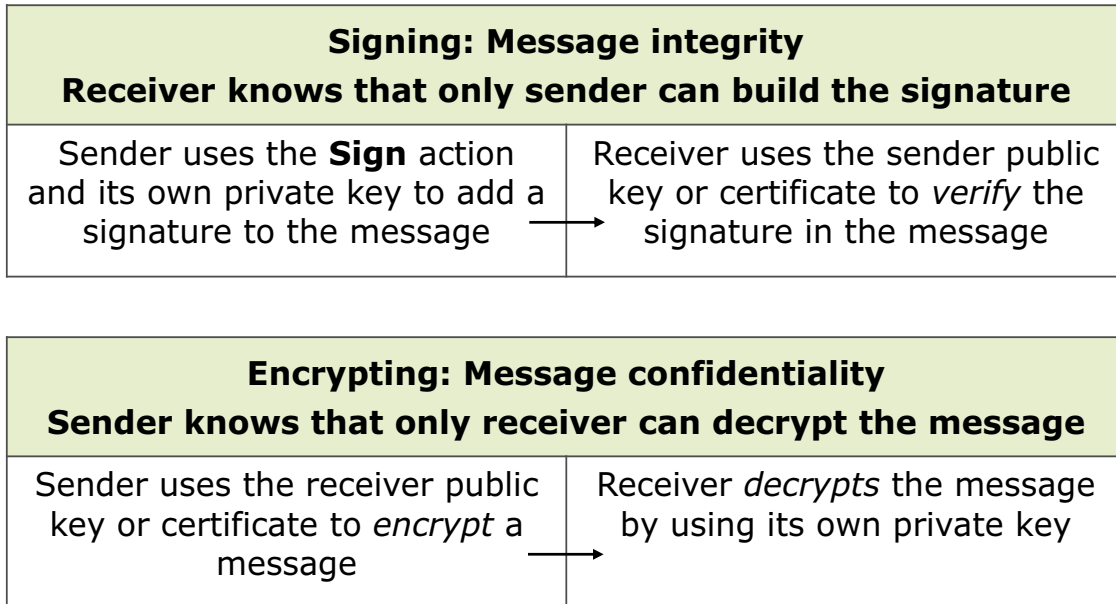
© Copyright IBM Corporation 2017

Figure 1-21. Sample signed SOAP message

This message is condensed to fit on the slide.

Signing messages might rewrite attributes in the message. Notice the `wsu:id` attribute added by the **Sign** action to the SOAP body. This action might cause the body of the message to invalidate against a schema, depending upon how the schema is written.

Summary of security and keys



Sign a message, and then encrypt, for best confidentiality and integrity

Unit summary

- Describe the features of the WS-Security specification
- Enable message confidentiality by using XML encryption
- Provide message integrity by using XML signatures

Review questions



1. True or False: A document crypto map is used to specify an XPath expression that contains the elements to encrypt, decrypt, sign, and verify.
2. True or False: Encryption and decryption can occur at both message and field levels, but sign and verify occur at the message level only.
3. True or False: The validation credential object validates the signer certificate, which is the public key that is used to generate the digital signature. This certificate is usually included in the message, but an alternative certificate can be specified in the **Optional Signer Certificate** field.

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-24. Review questions

Write your answers here:

- 1.
- 2.
- 3.

Review answers



1. True or False: A document crypto map is used to specify an XPath expression that contains the elements to encrypt, decrypt, sign, and verify.
The answer is False. A document crypto map is used to specify an XPath expression that contains the elements to encrypt, decrypt, and sign. The Verify action does not use a map since it can determine the signed elements from the headers.
2. True or False: Encryption and decryption can occur at both message and field levels, but sign and verify occur at the message level only.
The answer is False. Both scenarios are supported, even though the Verify action does not have a selected field-level radio button.
3. True or False: The validation credential object validates the signer certificate, which is the public key that is used to generate the digital signature. This certificate is usually included in the message, but an alternative certificate can be specified in the **Optional Signer Certificate** field.
The answer is True.

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-25. Review answers

Exercise: Web service encryption and digital signatures

XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-26. Exercise: Web service encryption and digital signatures

Exercise objectives

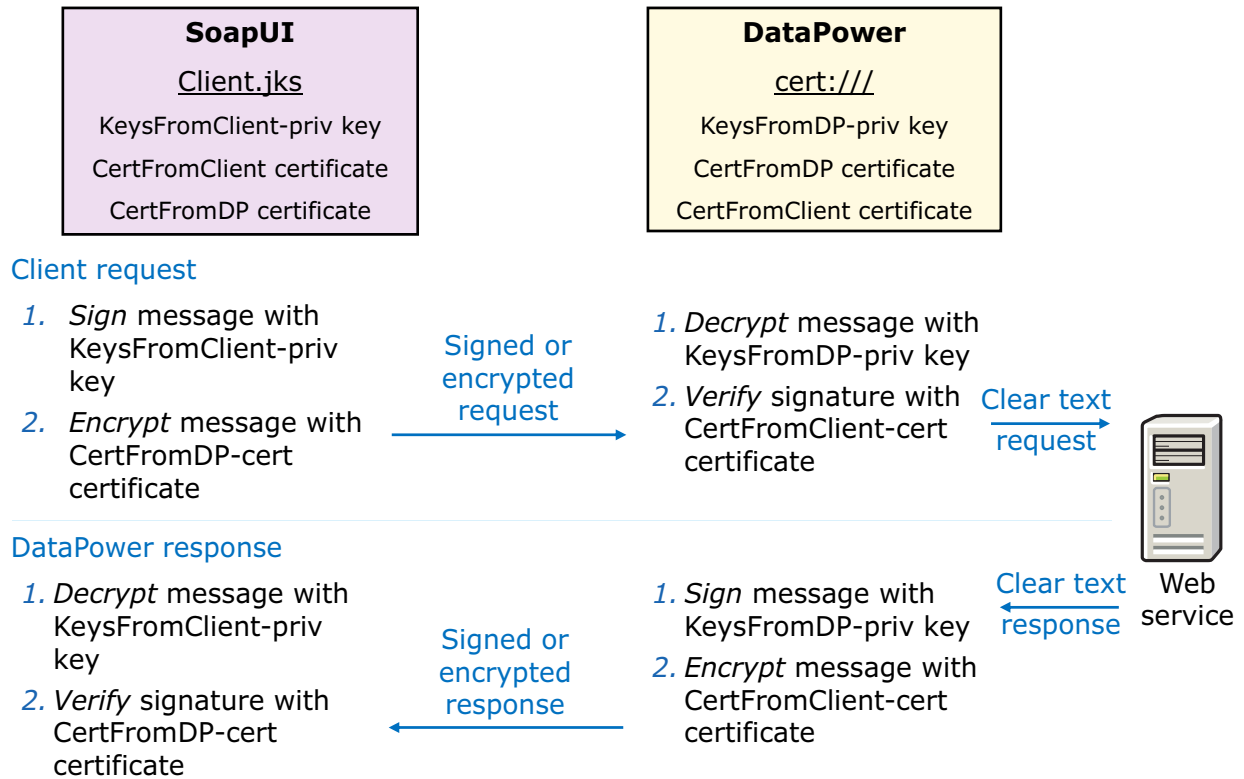
- Configure a multi-protocol gateway to decrypt and encrypt an XML message
- Configure a multi-protocol gateway to verify and sign an XML message
- Test encryption and digital signatures by using the SoapUI tool



Exercise overview

1. Create crypto DataPower objects
2. Import the BookingServiceProxy MPGW
3. Configure BookingServiceProxy to verify a signed message
4. Test the BookingServiceProxy signature verification by sending a signed message from SoapUI
5. Configure BookingServiceProxy to sign the response message
6. Test that the BookingServiceProxy returns a signed message to SoapUI
7. Configure BookingServiceProxy to decrypt a message
8. Test the BookingServiceProxy message decryption by sending an encrypted message from SoapUI
9. Configure BookingServiceProxy to encrypt the response message
10. Test that the BookingServiceProxy returns an encrypted message to SoapUI

Completed exercise



XML and web services security overview

© Copyright IBM Corporation 2017

Figure 1-29. Completed exercise

Unit 2. Web service proxy service

Estimated time

01:00

Overview

This unit describes the web service proxy service and its role in a web-services-based network. It explains the configuration steps that are required to create and manage a web services proxy. It also explains advanced web service configuration steps, such as proxy-level security, SOAPAction policy, and web service endpoint.

How you will check your progress

- Checkpoint
- Hands-on exercise

References

IBM Knowledge Center documentation for IBM DataPower Gateway Appliances Version 7.5:

www.ibm.com/support/knowledgecenter/SS9H2Y_7.5.0

Unit objectives

- Describe the web service proxy architecture
- List and explain the configuration steps that are needed to create a web service proxy
- Create and configure a web service proxy policy at various levels of the WSDL file

Web service proxy service

Figure 2-1. Unit objectives

© Copyright IBM Corporation 2017

Web service proxy overview

- A web service proxy is a middleware component that exists between the client and the web service
 - Decouples the web service client from the actual web service
 - Hides the web service endpoint address from the client
 - Flexibility to change the back-end address without affecting client code
 - Can virtualize multiple web services into a single client-facing web service
 - Performs security, validation, and transformation on a request or response to offload these tasks from the back-end web service
- You can use DataPower gateways to create a web service proxy to accelerate and mediate communication between a client and a web service
 - Rules are associated with different parts of a WSDL interface
 - Supports multiple WSDL documents
 - Fine-grained policy control
 - Built-in service level monitoring (SLM) capabilities

Web service proxy service

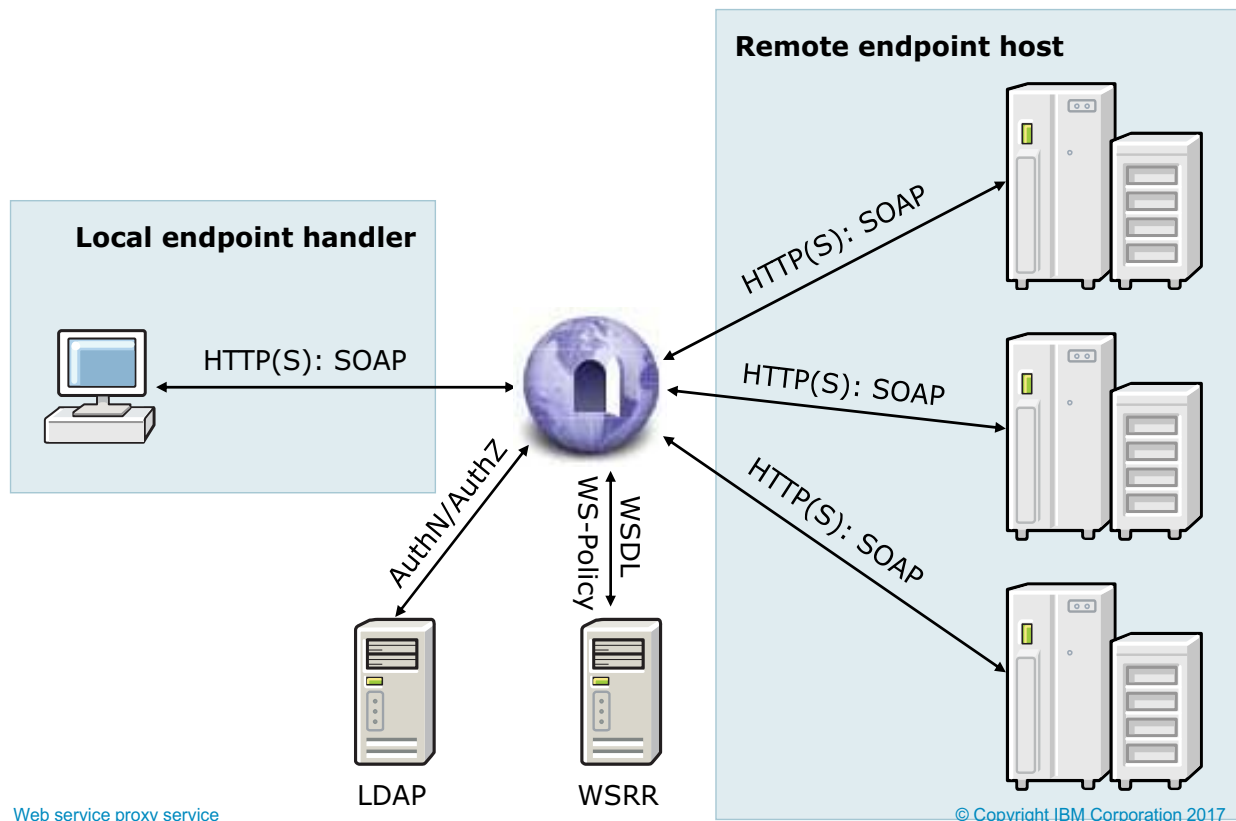
© Copyright IBM Corporation 2017

Figure 2-2. Web service proxy overview

It is not necessary for the client to know the endpoint address of the web service. It is always forwarded to the web service proxy. If the web service endpoint changes, only modifications to the web service proxy are required. The client is unaffected.

Performing security, validation, and transformation on the DataPower gateway for web service proxy requests improves application performance because it is done at a hardware level. It is offloaded from the application server, which would do these tasks in software. You can also apply a standard security policy for your web service proxy on the DataPower gateway because all requests pass through the gateway.

Conceptual architecture of a web service proxy



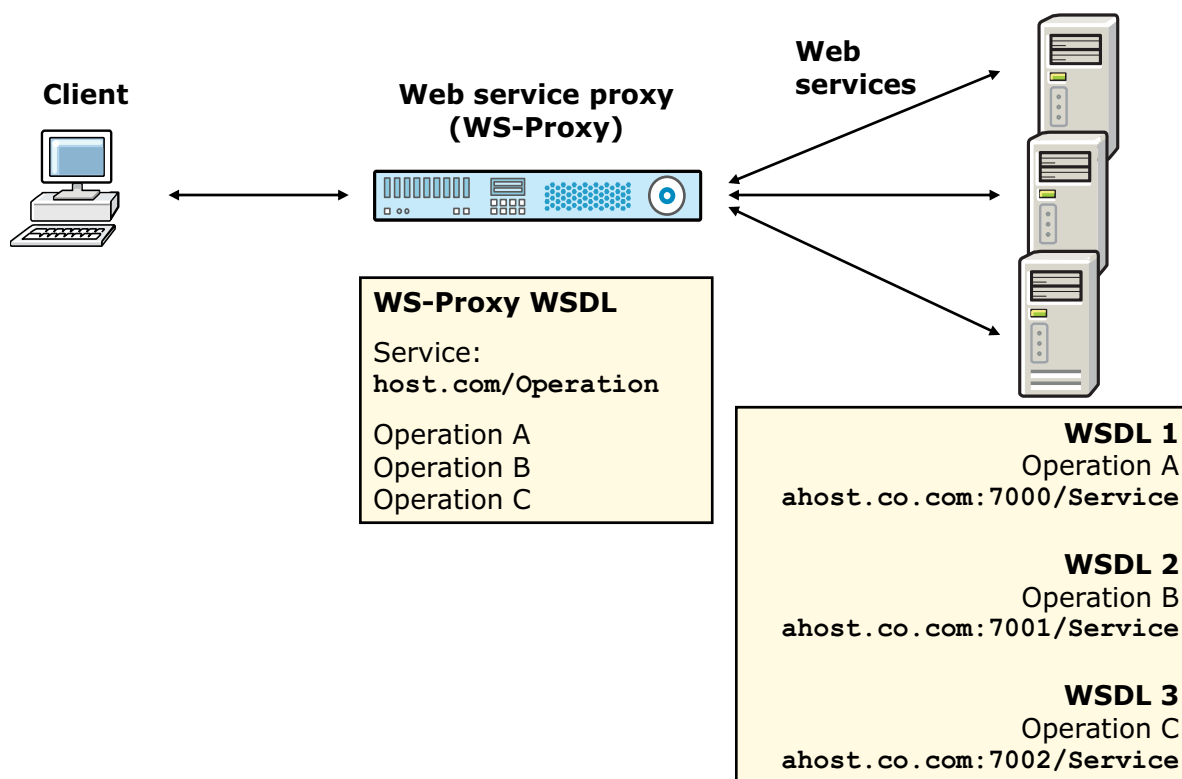
Web service proxy service

Figure 2-3. Conceptual architecture of a web service proxy

The web service proxy provides all of the same services as a Multi-Protocol Gateway service; however, it provides automatic configuration based on one or more Web Service Description Language (WSDL) files. WSDL files might be obtained through subscriptions to a Universal Description, Discovery, and Integration (UDDI) or WebSphere Service Registry and Repository (WSRR). DataPower supports enforcement of WebSphere Service Registry and Repository sophisticated service level definition and service level agreement policies. A single web service proxy object can act as a single point of entry for multiple WSDLs, automatically routing (or redirecting) the requests to the appropriate back-end service.

The web service proxy automatically applies schema validation to both inbound and outbound messages, further assuring message validity. Processing and security policies can be applied not only at the entire service level, but for individual operations within the service as well.

Web service virtualization



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-4. Web service virtualization

The web service proxy has a WSDL file that lists the operations it supports. These operations can be aggregated from multiple WSDL files that are in different locations.

The web service proxy maintains a mapping of a local endpoint and remote endpoint for each WSDL file.

Web service proxy benefits

Web service proxy quickly virtualizes your existing services

- Virtualizes a service by loading a WSDL document
 - Clients now connect directly to the web service proxy and not the back-end service
- Creates processing policy with rules and actions at a fine-grained level
 - Rules at a proxy, service, port, or operation level can process request and response messages
- Automatic schema validation of request, response, and fault messages (user policy)
 - Developer does not need to create a processing policy for this validation
- Service virtualization can occur in real time
 - Web service proxy can update the proxy WSDL automatically when the underlying WSDL is updated
 - Integrates with WebSphere Service Registry and Repository and UDDI registries
- Can enforce policy and monitor performance of services
 - Multiple gateway support for virtual services

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-5. Web service proxy benefits

You can schema validate request, response, and fault messages by using a **user policy**. It is automatically created when you create a web service proxy.

The web service proxy is built on top of the XML firewall. Therefore, it provides all of the functions of an XML firewall, such as encryption, validation, AAA, and more.

UDDI is a service repository that is used to search for WSDL files of a service.

Creating a WSDL cache policy enables the proxy WSDL file to be updated automatically when the underlying WSDL changes.

You can create an SLM peer group to share SLM data and enforce SLM policy between multiple DataPower gateways.

Web service proxy configuration tabs (1 of 2)



- **WSDL files**
 - Uploads or associates a WSDL document with a web service proxy
 - Configures the proxy and remote URI (address, port) of services that are contained in WSDL document
- **SLM Policy**
 - Monitors and shapes traffic that enters the web service proxy
- **Services**
 - Lists services that are defined in each WSDL document
 - Can publish services to a UDDI registry
- **Policy**
 - Configures a web service proxy policy
- **SLA Policy Details**
 - View WSDL attachments that relate to a service level agreement
- **Proxy Settings**
 - Specifies a method of forwarding to a service, security, XML manager, and HTTP settings

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-6. Web service proxy configuration tabs (1 of 2)

Each tab in the web service proxy GUI has configuration options.

The WSDL files, services, policy, and proxy settings tabs are covered in this unit.

SLM policy is covered in the prerequisite course.

Various policies (WS-Policy and WS-MediationPolicy, for example) can be specified in individual files, but point to an attachment point in a WSDL file. The SLA Policy Details tab is not covered in this course.

More tabs exist to the right that are not visible in this screen capture: WS-Addressing, WS-ReliableMessaging, Monitors, and XML Threat Protection. They are discussed on the next slide.

Web service proxy configuration tabs (2 of 2)



- **Advanced Proxy Settings**
 - Configures advanced connection settings
- **Headers/Params**
 - Add or remove HTTP headers and passes stylesheet parameters
- **WS-Addressing**
 - Specifies the WS-Addressing mode for this service
- **WS-ReliableMessaging**
 - Toggle to enable WS-ReliableMessaging (deprecated)
- **Monitors**
 - Identifies any message monitors associated with this service
- **XML Threat Protection**
 - Provides protection against XML threats

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-7. Web service proxy configuration tabs (2 of 2)

Web service proxy basic configuration steps

1. Create or obtain a WSDL document that describes your web service
2. Use the DataPower WebGUI to create a web service proxy
 - **Web Service Proxy** icon in the DataPower WebGUI Control Panel
or
 - Using the vertical navigation bar, click **Services > Web Service Proxy > New Web Service Proxy**
3. Upload the WSDL document and add it to the web service proxy
4. Configure the endpoint of services that are defined in a WSDL document
 - Define both the proxy URI and the endpoint URI for each service in the WSDL document
5. Specify a service policy that consists of rules for the web service (optional)

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-8. Web service proxy basic configuration steps

The URI consists of an address and port.

Step 5 is optional because the gateway generates a default service policy. The default service policy applies at the proxy level for each service. You can override the default service policy with a more specific policy at a fine-grained level for each service, port, or operation. Only one policy is executed per request or response.

You can also do these additional configuration steps:

- Configure how the proxy forwards requests to the back-end web service. By default, the URI defined in the WSDL document is used to determine the back-end web service.
- Select the SOAP action policy to specify how to consume messages with a SOAPAction header.
- Configure security settings, such as proxy-wide AAA settings, the decryption key, and the SSL proxy profile, to a back-end service.

Step 1: Obtain WSDL document

- A WSDL document that describes your web service is required before creating a web service proxy
- A WSDL document describes a web service interface by using XML
 - Uses the W3C XML schema type system for type information
 - Contains operations and messages that are bound to a network protocol and message format
 - Includes binding and location information for published web services
- DataPower creates a web service proxy that is based on the structure of a WSDL document
 - WSDL-based configuration consists of a service, ports, and operations
 - SLM and policy configuration can be defined at various levels of the WSDL document

Web service proxy service

© Copyright IBM Corporation 2017

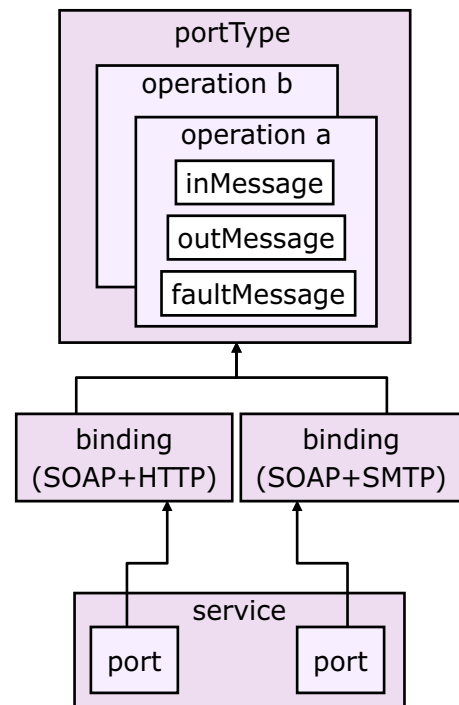
Figure 2-9. Step 1: Obtain WSDL document

A WSDL document describes the service operations that can be invoked together with their messaging protocol, transport, and endpoint address.

Each operation contains an input and output message, whose types are defined by using the XML schema type system.

WSDL structure

- portType
 - Abstract definition of a service
 - Same idea as a Java interface
- binding
 - How to access the portType
 - Multiple bindings per portType
 - HTTP, JMS, SMTP, and more
- port
 - Represents an individual endpoint
- service
 - Something that can be invoked
 - Represents a collection of ports



All levels of the WSDL are visible and available within the web service proxy

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-10. WSDL structure

This diagram shows the general structure of a WSDL and the relationships of the elements to each other.

It is important to understand the structure of the WSDL document because policy and SLM configuration can be done at different levels of the WSDL file.


For more information about WSDL, see: <http://www.w3.org/TR/wsd>

Step 2: Creating a web service proxy

- You can create a web service proxy by:
 - Clicking the **Web Service Proxy** icon in the DataPower WebGUI Control Panel and clicking **Add** on the “Configure Web Service Proxy” listing page



Web Service Proxy


Configure Web Service Proxy

| Web Service Proxy Name | Op-State | Logs | Type | Req-Type | Back Side URL | Resp-Type |
|------------------------|----------|------|------------------|----------|---------------|-----------|
| BookingServiceWSProxy | up | | Static from WSDL | SOAP | NA | SOAP |

Add

- Using the vertical navigation bar, click **Services > Web Service Proxy > New Web Service Proxy**
- You are first prompted for the name of the web service proxy

Web Service Proxy Name

Create Web Service Proxy

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-11. Step 2: Creating a web service proxy

You can use either approach in creating a web service proxy. The web pages are identical.

From the web service proxy catalog list, you click **Add** to create a service. You are first prompted for the new service name.

An alternative: Web service proxy object editor

- A web service proxy can be created by using a non-graphical, non-wizard approach (not common)
 - From the vertical navigation bar, click **Objects > Service Configuration > Web Service Proxy**
 - All configuration options are available

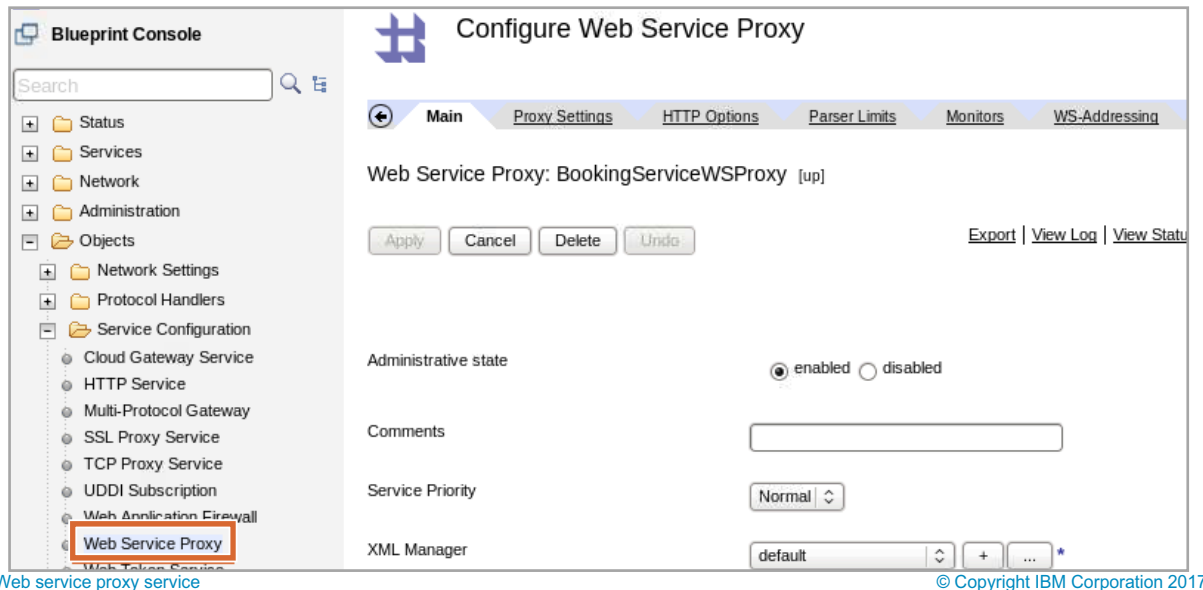


Figure 2-12. An alternative: Web service proxy object editor

The WSDL cache policy and some attachment processing specifications are example configurations that are possible only by using this editor.

Step 3: Add WSDL document to web service proxy

Configure Web Service Proxy

WSDL files | SLM Policy | Services | Policy | SLA Policy Details | Pro

Web Service Proxy Name [up]

CustomerServiceProxy *

Apply Cancel Delete

WSDLs

Edit WSDL or Subscription | Add WSDL | Add UDDI Subscription | Add WSRR Subscription

WSDL File URL

local:///

(none) Upload... Fetch... Edit... View... E

Use WS-Policy References

☒ on ☐ off

WS-Policy Parameter Set

(none) + ...

WS-Policy Enforcement Mode

Enforce

SLA Enforcement Mode

Allow

1. The **Web Service Proxy Name** is transferred from the earlier prompt
2. The **Add WSDL** option is already active for a new service
3. Add the WSDL file by using *one* of the following approaches:
 - Enter **WSDL File URL** (remote or local URL)
 - Upload WSDL file to **local:** directory
 - Select previously uploaded WSDL document
 - Retrieve from registry
4. Click **Next**

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-13. Step 3: Add WSDL document to web service proxy

The Configure Web Service Proxy page is displayed.

The **Edit WSDL or Subscription**, **Add WSDL**, **Add UDDI Subscription**, **Add WSRR Subscription**, and **Add WSRR Saved Search Subscription** options act like a button when they are selected.

Click **Upload** to upload a WSDL file to the DataPower gateway. The WSDL file can be uploaded to the `local:` directory (which is accessible in the current domain) or to the `store:` directory (which is accessible in all domains). The preference is for the “local” files to be in the `local:` directory.

When you upload a WSDL file, the WSDL file URL is automatically populated.

You can upload and add multiple WSDL files, but they are added and configured one at a time.

You can also enter an HTTP URL into the WSDL file URL, and the web page populates the fields with information from the WSDL file.

Policy References: Enable policies that are attached to WSDL by using PolicyURI attributes and PolicyReference elements. These attachments are sometimes called XML element attachments. If “off”, all PolicyURI attributes and PolicyReference elements are ignored and only external policies are enforced.

Web Services Policy (WS-Policy) is a specification that defines metadata to enable interoperability between web services consumers and web services providers. With WS-Policy, you can automate your service governance models to create a concrete instance of web services governance. The following options control the configuration of WS policies to this WSDL:

- **WS-Policy Parameter Set:** Configuration to persist the values of ws-policy parameters.
- **WS-Policy Enforcement Mode:** Enforcement Mode defines how the service uses WS-Policy to ensure that messages meet security requirements. Enforce is the default behavior.
- **SLA Enforcement Mode:** SLA Enforcement Mode controls the application of SLA policies to transactions. Transactions are either allowed or rejected based on whether an SLA rule is applied to the transaction.

Step 4: Configure WSDL endpoint

After clicking **Next**, specify the local and remote URI of the WSDL service

- Local (what the client sees):
 - Local endpoint handler
 - Specify URI sent by client
- Remote (where the web service really is):
 - Web service endpoint (protocol, host name, port, and URI)

WSDLs

BookingService - BookingServiceSOAP

Local

| Local Endpoint Handler | URI | Binding (Suffix) | Edit/Remove |
|--------------------------|------------------|---|-------------|
| BookingServiceWSProxyFSH | /BookingService/ | SOAP 1.1() | Edit Remove |
| (none) | + ... | <input checked="" type="checkbox"/> SOAP 1.1 <input type="checkbox"/> SOAP 1.2 <input type="checkbox"/> HTTP GET | Add |

Remote

| Protocol | Remote Endpoint Host | Port | Remote URI |
|----------|----------------------|------|------------------|
| HTTP | 172.16.78.23 | 9080 | /BookingService/ |

Published ☒ Use Local

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-14. Step 4: Configure WSDL endpoint

The **Local** section contains necessary information for the client to call a service on the web service proxy. You create a local endpoint handler to specify a port number that listens for requests of a particular service and forwards to the remote destination. The endpoint handler is another name for a front side protocol handler. These handlers can be managed individually from **Objects > Protocol Handlers**.

Under **Local**, the URI field is what the client uses prefaced with the host name of the DataPower gateway and the port that is specified in the **Local Endpoint Handler** object.

The **Remote** section contains information about the web service endpoint address that the web service proxy calls. Make sure that you change the default host name of `localhost` to the correct host name.

The **Remote** section **Protocol** choice lists the various protocols available on the back side of the service. Depending on the particular protocol that is selected, the other fields adjust:

- DPMQ
- DPTIBEMS
- DPWASJMS
- HTTP

- HTTPS
- MQ
- TIBEMS

The protocols and URLs are defined as part of the documentation of the url-open extension element. For more information, see the DataPower documentation in the IBM Knowledge Center.

Step 5: Configure local endpoint handler

- A Local Endpoint Handler (a front side handler) is used to determine the IP address, port, and protocol
- Click the plus sign (+) to create a new local endpoint handler object

The screenshot shows the configuration interface for a local endpoint handler. The 'Local' tab is active, displaying a table with columns 'Local Endpoint Handler' and 'URI'. The first entry is 'BookingServiceWSPProxyFSH' with URI '/BookingService/'. Below the table, there is a dropdown menu showing '(none)' and a plus sign (+) button. The plus sign button is highlighted with a red box. To the right of the plus sign button, a 'Create a New:' dropdown menu is open, showing a list of handler options. The 'HTTP Front Side Handler' option is highlighted with a red box. Below the 'Create a New:' dropdown, there is a 'Remote' section with a 'Protocol' dropdown set to 'HTTP' and a 'Remote Endpoint' text box containing '172.16.78.23'. Below the 'Remote' section, there is a 'Published' section with a checked 'Use Local' checkbox. Below the 'Published' section, there is a 'Local IP address' text box containing 'dp_public_ip' and a 'Port' text box containing '12315'. Below the 'Port' text box, there is an 'HTTP version to client' dropdown set to 'HTTP 1.1'. The 'Administrative state' section shows 'enabled' selected. The 'Comments' section is empty.

- Specify the gateway local IP address and port number to listen for requests
- You can also restrict access based on HTTP attributes

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-15. Step 5: Configure local endpoint handler

The handler choices are the same as for the front side handler in an MPGW.

The **Local IP address** of 0.0.0.0 means that the endpoint handler listens for requests on all of the gateway interfaces.

Make sure that the port number you specify here is unique.

Endpoint handlers and front side handlers are synonymous terms, and are configured in the same way.

Step 6: Add the WSDL to the service

Local

| Local Endpoint Handler | URI | Binding (Suffix) | Edit/Remove |
|---------------------------|------------------|--|-------------|
| BookingServiceWSPProxyFSH | /BookingService/ | <input checked="" type="checkbox"/> SOAP 1.1 <input type="checkbox"/> SOAP 1.2 HTTP GET | Add + |

BookingService - BookingServiceSOAP

Local

| Local Endpoint Handler | URI | Binding (Suffix) | Edit/Remove |
|---------------------------|------------------|--|-------------|
| BookingServiceWSPProxyFSH | /BookingService/ | SOAP 1.1() | Edit Remove |
| (none) | | <input checked="" type="checkbox"/> SOAP 1.1 <input type="checkbox"/> SOAP 1.2 HTTP GET | Add + |

Remote

| Protocol | Remote Endpoint Host | Port | Remote URI |
|----------|----------------------|------|------------------|
| HTTP | booking.FLY.com | 9070 | /BookingService/ |

Published ☒ Use Local

Next Cancel

Click **Add** to add the customized WSDL information to the service; then, click **Next**

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-16. Step 6: Add the WSDL to the service

Clicking **Add** adds the selected WSDL to the service.

As soon as a WSDL is added, it can be edited or removed by selecting the appropriate icon to the right of the WSDL.

Clicking **Next** commits the WSDL to the service.

Initial WSDL completed

- Completed WSDL is listed
 - WSDL status is listed
 - **Edit WSDL or Subscription** option is highlighted
- Other options, such as adding another WSDL, are now available

The screenshot shows the 'Configure Web Service Proxy' interface. The 'WSDL files' tab is selected. The 'Web Service Proxy Name' is 'BookingServiceWSPProxy'. The 'WSDLs' section contains a table with the following data:

| WSDL Source Location | Endpoint Handler Summary | WSDL Status | WS-I BP Status |
|------------------------------|--------------------------|-------------|----------------|
| local:///BookingService.wsdl | 1 up / 1 configured | Okay | Okay |

The 'Edit WSDL or Subscription' button is highlighted in the 'WSDLs' section. An arrow points from the 'Delete' button to the 'Edit WSDL or Subscription' button.

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-17. Initial WSDL completed

The WSDL is now part of the service.

More WSDLs or subscriptions can be added to the service.

Other ways to get a WSDL

- Subscribe to a UDDI registry
- Subscribe to WebSphere Service Registry and Repository
- Subscribe to a WSRR Saved Search
 - Can automatically “push” changes to the web service proxy

Configure Web Service Proxy

WSDL files | SLM Policy | Services | Policy | SLA Policy Details | Proxy Settings | Advanced Proxy Settings | Headers/Params

Web Service Proxy Name [up]
 *

Apply Cancel Delete

[Export](#) | [View Log](#) | [View Status](#) | [View Operations](#) | [Show Probe](#) | [Conformance](#)

WSDLs

[Edit WSDL or Subscription](#) [Add WSDL](#) [Add UDDI Subscription](#) [Add WSRR Subscription](#) [Add WSRR Saved Search Subscription](#)

| WSDL Source Location | Endpoint Handler Summary | WSDL Status | WS-I BP Status |
|--|--------------------------|-------------|----------------|
| <input checked="" type="checkbox"/> local:///BookingService.wsdl | 1 up / 1 configured | Okay | Okay |

[Web service proxy service](#)

© Copyright IBM Corporation 2017

Figure 2-18. Other ways to get a WSDL

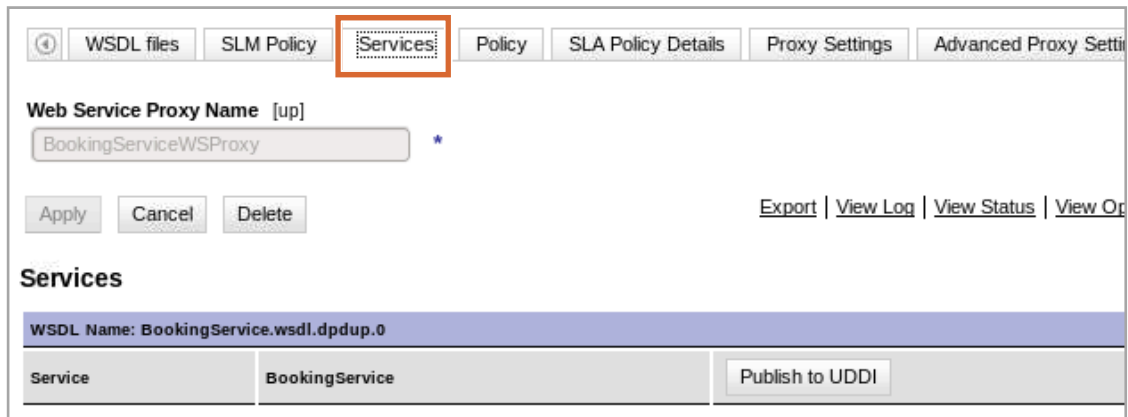
A subscription to a registry might also retrieve a WSDL file.

Generally, the registries are polled for the WSDL file on a timed basis, and can also be explicitly polled.

A WSRR Saved Search can be configured to send a WSDL file update from WebSphere Service Registry and Repository to the service.

View WSDL services

- Click the **Services** tab to view the services that are extracted from the WSDL document
 - Click **Publish to UDDI** to configure a connection to a UDDI registry



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-19. View WSDL services

The gateway automatically generates the services in this tab when you add a WSDL file to the web service proxy.

Universal Description, Discovery, and Integration (UDDI) is an XML-based registry that is used to search for WSDL documents.

UDDI is implemented as a web service that you can publish and search for web services.

The DataPower gateway does not provide a UDDI registry, only a connection. The gateway supports connection to only the UDDI V2 repository.

The **View Operations** opens another window that lists the operations that are defined in the WSDLs exposed by this service.

Modifying the location in the client WSDL

- The WSDL retrieved from the web service proxy by using `?wsdl` by default places the IP address and port of the gateway in the “location”

```
<wsdl:port binding="..." name="BookingService">
  <address location="http://192.168.10.41:12315/BookingService" />
</wsdl:port>
```

- You can specify a different host name or port to place in the WSDL
 - Clear **Use Local** to enter your own values
 - Now retrieved by `?wsdl`

```
<wsdl:port binding="..." name="BookingService">
  <address location="http://MyDP.proxy.com:12315/BookingService" />
</wsdl:port>
```

| Remote | | | |
|----------|----------------------|------|------------------|
| Protocol | Remote Endpoint Host | Port | Remote URI |
| HTTP | booking.FLY.com | 9070 | /BookingService/ |

| Published | | | |
|-----------|-----------------------|------|------------------|
| Protocol | Hostname (IP Address) | Port | Publish URI |
| HTTP | MyDP.proxy.com | 9070 | /BookingService/ |

Web service proxy service

© Copyright IBM Corporation 2017

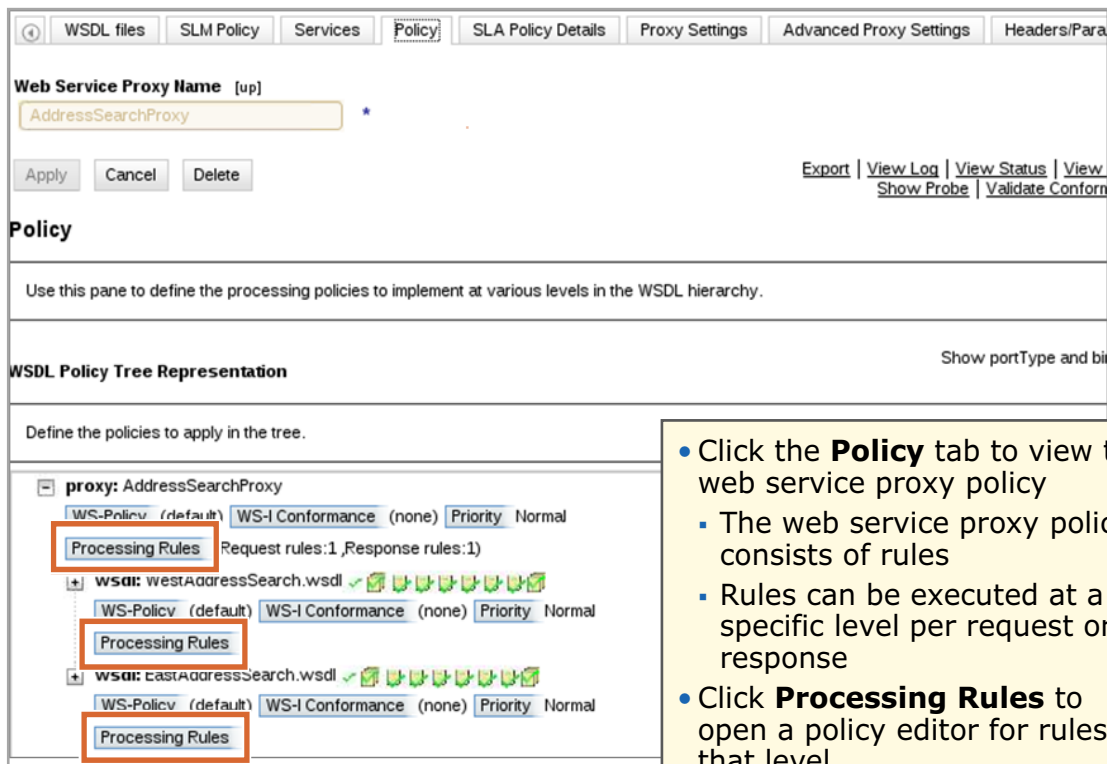
Figure 2-20. Modifying the location in the client WSDL

The WSDL retrieved by `?wsdl` contains the IP address and port of the gateway and web service proxy service.

By clearing the **Use Local** check box, you can explicitly specify the host name, port, and URI that are included in the retrieved WSDL.

This feature becomes especially useful if you have a load balancer that fronts the gateway. By using the explicit approach, you can specify the load balancer details in the retrieved WSDL so the clients send their requests to the correct host name, port, or URI on the load balancer. The load balancer must be configured to forward requests to the gateway host name, port, or URI.

Step 7: Configuring web service proxy policy



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-21. Step 7: Configuring web service proxy policy

The **Policy** tab shows the rules that are defined at the various levels within any WSDLs defined for this service.

Default proxy-level rules are provided.

Clicking **Processing Rules** opens the policy editor in a lower section of the page.

A toggle is available to show the portType and binding nodes in the WSDL levels. The default is to not display them in the policy tree.

The **more** links display more help text on the page.

Configure web service proxy policy rule

Policy Configuration

Define the processing rules and the actions to perform against requests and responses and the processing for error conditions.

Rule:

Rule Name: Rule Direction:

Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action icon.

Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results SLM Action

CLIENT

Configured Rules

| Order | Rule Name | Direction | Actions |
|-------|--|------------------|--------------|
| 1 | addressSearchProxy_default_request-rule | Client to Server | SLM, Results |
| 2 | addressSearchProxy_default_response-rule | Server to Client | Results |

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-22. Configure web service proxy policy rule

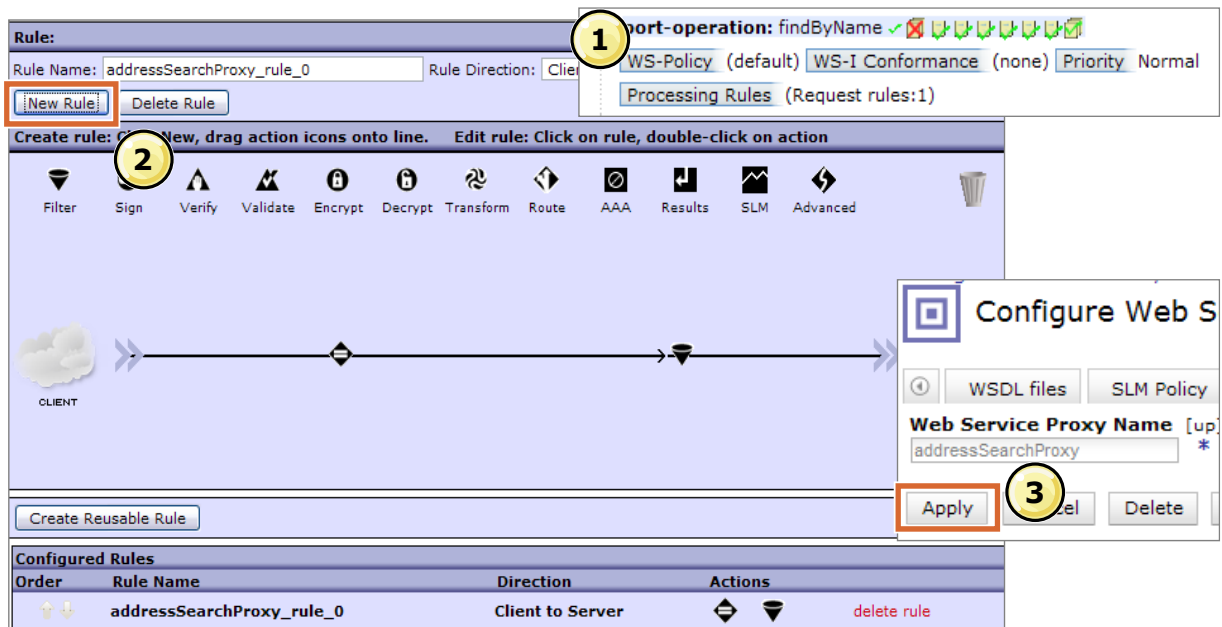
You can add, view, or modify rules by selecting **Processing Rules** at the intended level.

To show or hide the levels of the WSDLs, click the plus sign (+) or the minus sign (-).

The default proxy-level rule contains two actions, an **SLM** action and a **Results** action. The **SLM** action is a checkpoint event that calls the web service proxy SLM policy. You can verify the **SLM** action by double-clicking it and noting the SLM policy name. Click the **SLM Policy** tab to verify that the proxy name that is listed in the page is the same as the **SLM** action.

Adding a rule

1. Add a rule to the **findByName** operation by clicking **Processing Rules**
2. Click **New Rule** in policy editor (default name can be typed over)
3. When finished, click **Apply** at the Web Service Proxy (page) level



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-23. Adding a rule

The number and type of rules that are defined at that level in the WSDL are displayed next to the **Processing Rules** button.

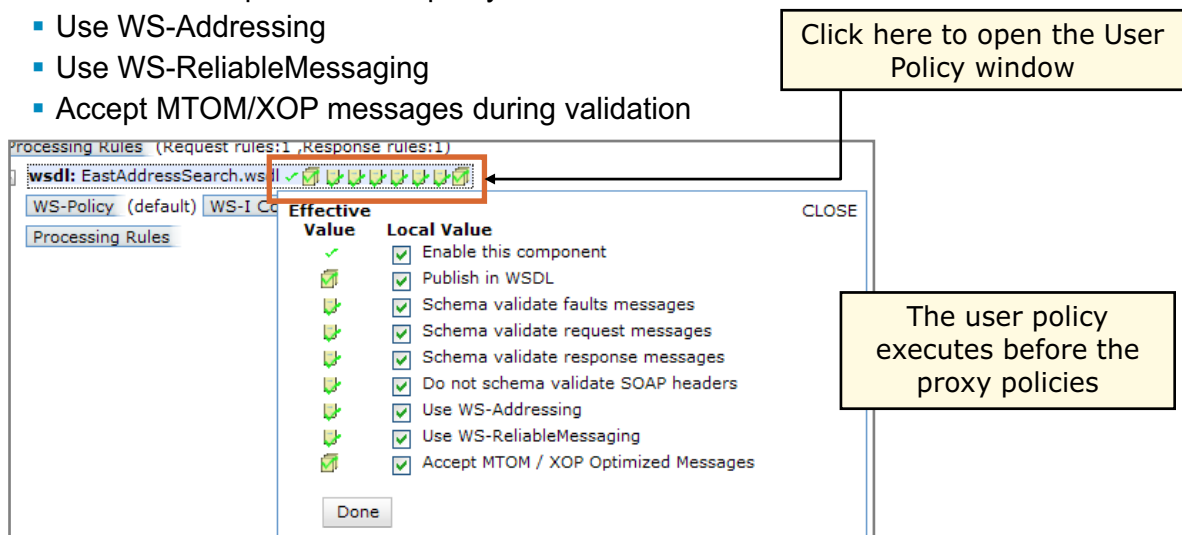
You create and configure the actions in the rule as you normally would.

All rules that are configured at this level are in the **Configured Rules** section of the policy editor.

Clicking **Apply** at the page (service) level commits this rule to the policy.

Default validation (user policies)

- By default, each level of the proxy (proxy, WSDL, service, port, operation) defines a user policy to:
 - Schema validate messages (against schema in WSDL): request, response, and fault
 - Schema validate SOAP headers (against schema in WSDL)
 - Enable or disable a component
 - Publish a component in the proxy WSDL file
 - Use WS-Addressing
 - Use WS-ReliableMessaging
 - Accept MTOM/XOP messages during validation



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-24. Default validation (user policies)

Click any of the icons at each level to view the user policy dialog box.

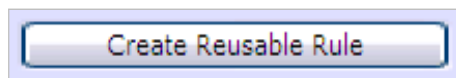
The first check mark enables the component. Each option that is shown in the dialog box maps to an icon with a green check mark or red X.

Each policy level contains a user policy that can be enabled or disabled.

The web service proxy policy and user policy are separate from each other. The user policy is executed before the web service proxy policy.

Create reusable rule

- Use the rule configuration area to select a set of actions to be invoked as a reusable rule
 1. Click **Create Reusable Rule**.
 2. Draw a box around the actions to include in the reusable rule.
 3. Click **Apply**. A gray rectangle appears around the reusable rule in the rule configuration area and generates a new rule name for the new reusable rule.
 4. Use the **Advanced – Call Processing Rule** action to reuse the rule.



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-25. Create reusable rule

Reusable rules are useful for applying a common set of actions at many levels of the web service proxy. More actions can be added before or after the reusable rules. With reusable rules, you can more easily manage a set of actions that repeat across many levels of the web service proxy.

Reusable rules can be defined in the other service type processing policies, such as an MPGW or XML firewall policy. The new reusable rule is given a name by the policy editor. You cannot designate your own.

When you create a reusable rule, the original included actions are replaced with a Call Processing Rule action.

Advanced web service proxy configuration

More options and tabs are available for advanced web services proxy configuration:

- Proxy settings
 - Web service proxy type
 - Security settings (AAA, cryptographic key)
 - SOAP Action Policy
 - XML Manager
- Advanced proxy settings
 - HTTP connection settings
- Headers/Parameters
 - Adds or removes HTTP headers and passes stylesheet parameters
- WS-Addressing
 - Indicates support for WS-Addressing for front-end or back-end server
- WS-ReliableMessaging
 - Indicates whether to use WS-ReliableMessaging
- XML threat protection
 - Provides protection against XML threats

Web service proxy service

© Copyright IBM Corporation 2017

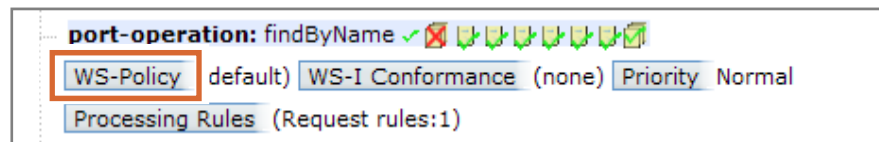
Figure 2-26. Advanced web service proxy configuration

In this presentation, only the proxy settings are examined, in later slides.

For more information, see the DataPower documentation in the IBM Knowledge Center. Look for information about the settings that are contained in the **Advanced Proxy Settings**, **Headers/Parameters**, **WS-Addressing**, **WS-ReliableMessaging**, and **XML Threat Protection** tabs.

WS-Policy

- WS-Policy is a specification that defines metadata to enable interoperability between web service consumers and web service providers
- The WS-Policy specifications enable organizations to automate their service governance models by creating a concrete instance of web service governance
- Behaviors:
 - Parse WSDL with policy elements already included in the WSDL and recognize standardized policy “domains” (WS-Security Policy, WS-ReliableMessaging Policy)
 - DataPower supports retrieving WSDL by using WebSphere Service Registry and Repository queries
 - DataPower supports retrieving WSDL by using a UDDI interface



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-27. WS-Policy

WS-Policy is used to assert policies on security, quality of service (QoS), required security tokens, privacy, and other items. A web service can stipulate what it can provide, and a consumer can stipulate its requirements.

Conformance policy

- Defines which profiles to use to validate whether received messages are in conformance to the selected interoperability profiles
- When a client sends nonconforming requests for a conforming back-end server:
 - The conformance policy can be used to fix nonconforming requests during message processing
- For signed and encrypted nonconforming data:
 - The cryptographic protection must be removed before and after conformance correction
- It can be added to a WS-Proxy in the Policy editor



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-28. Conformance policy

Supported profiles are as follows:

- WS-I Basic Profile Version 1.0
- WS-I Basic Profile Version 1.1
- WS-I Attachments Profile Version 1.0
- WS-I Basic Security Profile Version 1.0

Any conformance correction must be coded in a stylesheet; the firmware does not automatically provide it.

Conformance policy object

1. **Profiles:** Profiles against which conformance is checked
2. **Ignored Requirements:** Conformance requirements to ignore
3. **Corrective Stylesheets:** XSL sheets are invoked after conformance analysis
4. **Record Report:** Degree of nonconformance that causes a report to be recorded
5. **Reject non-conforming messages:** Degree of nonconformance that causes a rejected message
6. **Other Options:** Deliver conformance analysis report as an action result

The image displays two overlapping windows of the 'Operation Conformance Policy' configuration interface. The left window is in the 'Basic' tab, showing a list of profiles (WS-I BP 1.0, WS-I BP 1.1, WS-I AP 1.0, WS-I BSP 1.0) and radio buttons for rejecting non-conforming messages (Never, Failure, Warning). The right window is in the 'Advanced' tab, showing radio buttons for the record report (Never, Failure, Warning, Always), a text field for ignored requirements, a text field for corrective stylesheets, and a checkbox for 'Use analysis as result'. Numbered callouts (1-6) are placed around the windows to highlight specific features: 1 points to the profile list, 2 to the ignored requirements field, 3 to the corrective stylesheets field, 4 to the record report options, 5 to the reject non-conforming messages options, and 6 to the other options section.

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-29. Conformance policy object

Ignored requirements are entered as a text string. For example, `BSP1.0:R4221` would ignore requirement R4221 in the Basic Security Profile V1.0.

Record report options include:

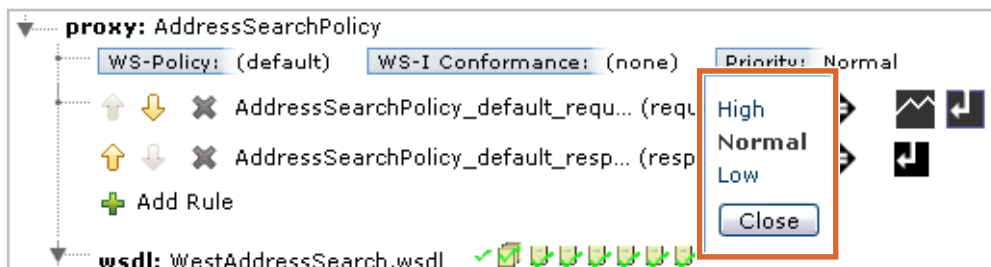
- **Never:** Never record reports
- **Failure:** Record reports with conformance failures
- **Warning:** Record reports with conformance warnings
- **Always:** Record reports for all outcomes

Reject nonconforming messages include:

- **Never:** Never reject messages
- **Failure:** Reject messages with conformance failures
- **Warning:** Reject messages with conformance warnings or failures

Service priority

- The policy editor has a **Priority** field
 - Sets priority for resource allocation and scheduling



- The different levels of priority are:
 - **High:** Receives higher than normal priority
 - **Normal:** (default) Receives normal priority
 - **Low:** Receives lower than normal priority

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-30. Service priority

The priority has no effect until the gateway encounters a resource constraint.

Proxy settings (1 of 4)

- Click the **Proxy Settings** tab to view the proxy settings
 - Many options have default values
- **Type**
 - **Dynamic Backend**: Web service proxy determines the back-end server during service policy processing
 - **Static Backend**: Web service proxy forwards to a single back-end server
 - **Static from WSDL** (default): Service section in the WSDL file determines the back-end server

The screenshot shows the 'Proxy Settings' tab of a configuration window. At the top, there are tabs for 'WSDL files', 'SLM Policy', 'Services', 'Policy', 'SLA Policy Details', 'Proxy Settings', and 'Advanced Proxy Settings'. The 'Web Service Proxy Name' field is set to 'AddressSearchProxy'. Below this are 'Apply', 'Cancel', 'Delete', and 'Refresh' buttons. The 'General Configuration' section has a 'Comments' text area. The 'Type' section, highlighted with a red box, contains three radio buttons: 'Dynamic Backend', 'Static Backend', and 'Static from WSDL' (which is selected). To the right, the 'XML Manager' shows 'default' and the 'Authorization AAA Policy' shows '(none)'. There are also 'Export', 'View', and 'Show' links in the top right.

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-31. Proxy settings (1 of 4)

When the web service proxy receives requests from a client, it forwards them to a back-end server for a service request.

The **Type** section specifies how that back-end server is determined. A back-end server is identified with a URL and port. The default option is **Static from WSDL**, which uses the WSDL file to determine the back-end server. The **Dynamic Backend** option determines the back-end server during document processing, and the **Static Backend** option always forwards to a single back-end server.

If the **Static Backend** type is selected, the page reloads, and you are supplied fields in which you can enter the back-end information. Several URL Helper buttons (WebSphere MQ, TibcoEMS, WebSphereJMS, and IMS Connect) are presented to help build the back-end URL.

Proxy settings (2 of 4)

- **Decrypt Key**
 - Selects a cryptographic key object to decrypt the message payload
- **EncryptedKeySHA1 Cache Lifetime**
 - Cache Lifetime for the decrypted generated key
- **Preserve EncryptedKey Chain**
 - Whether to output the element chain that is used to decrypt
- **Decrypt with Key from EncryptedData**
 - Enable decrypt action to attempt decryption with the key that is inside the EncryptedData element
- **Client Principal**
 - The client principal name when decrypt is required
 - Used when the encryption uses a Kerberos session key or uses a key that is derived from the session key
- **Server Principal**
 - The server principal name when decrypt is required
 - Used when the encryption uses a Kerberos session key or uses a key that is derived from the session key

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-32. Proxy settings (2 of 4)

The message payload refers to the message body.

Encrypting a message introduces new elements into the SOAP message that would cause automatic message validation to fail because a typical schema validation does not check for these elements.

An example SOAP message with encrypted payload might look like:

```
<SOAP:Body>
<EncryptedData ...>
```

Using a cryptographic key ensures that a message can pass automatic validation by decrypting the message payload before validation. When decrypting the message payload, either the root node of the message must be `<EncryptedData ...>` or the first child of the SOAP body element must be `<EncryptedData ...>`.

This method does not support field-level decryption. To do field-level decryption on the message, you must turn off automatic validation to create a rule with a **Decrypt** action.

EncryptedKeySHA1 Cache Lifetime is the cache lifetime for the decrypted generated key. Setting the value to 0 means that the decrypted generated key is not cached.

Preserve EncryptedKey Chain, if it is on, outputs the chain of elements that the decrypted EncryptedData uses, such as `xenc:EncryptedKey` or `wsc:DerivedKeyToken`. Otherwise, all `xenc:EncryptedKey` elements are removed after decryption, and even some of the encrypted data might not be decrypted successfully.

Decrypt with Key from EncryptedData: In scenarios in which the key is inside an EncryptedData element (such as “encrypted SAML Assertion”), the decrypt action cannot locate the key to decrypt the corresponding EncryptedData elements. Select **on** to enable the decrypt action to attempt decryption with the key that is inside the EncryptedData element.

The **Client Principal** field contains the full name of the client principal when the web service proxy must automatically decrypt encrypted requests. Use this property when the encryption uses a Kerberos session key, or a key that was derived from the session key.

Client and server principals are needed only when using Kerberos.

Proxy settings (3 of 4)

- **Kerberos Keytab**

- Select the Kerberos keytab file that contains the principals

- **SOAP Action Policy:**

Validates messages that contain a SOAPAction HTTP header

- **Lax:** Validates messages with empty SOAPAction HTTP header or empty string within SOAPAction HTTP header
- **Off:** SOAPAction HTTP header is ignored
- **Strict:** Message must contain exact match of SOAPAction header that is specified in WSDL file

- **Monitor via Web Services Management Agent:** Allows for autonomous monitoring of this service by a WS-Management agent

The screenshot shows a configuration window titled 'Kerberos Keytab'. It has a dropdown menu currently showing '(none)' with a '+' button and a '...' button to its right. Below this is the 'SOAP Action Policy' section with three radio buttons: 'Lax' (which is selected), 'Off', and 'Strict'. At the bottom is the 'Monitor via Web Services Management Agent' section with two radio buttons: 'on' (which is selected) and 'off'.

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-33. Proxy settings (3 of 4)

Select the Kerberos Keytab object that contains the principals for the **Kerberos Keytab** list. The web service proxy uses these principals to automatically decrypt encrypted requests and responses.

The WSDL file for a service defines the value that a SOAPAction header must contain for a SOAP request. The SOAPAction header is defined in the HTTP header, not the SOAP header.

The **SOAP Action Policy** setting specifies how to validate messages with a SOAPAction HTTP header.

A WS-Management agent can monitor the web service proxy without any monitors that are defined on the Monitors tab.

Proxy settings (4 of 4)

- **XML Manager:** Assigns an XML manager to the web service proxy
- **Authorization AAA Policy:** Selects or creates a AAA policy to apply to all service endpoints configured for this web service proxy
 - AAA policy can also be applied at a fine-grained level in the **Policy** tab

Web service proxy service

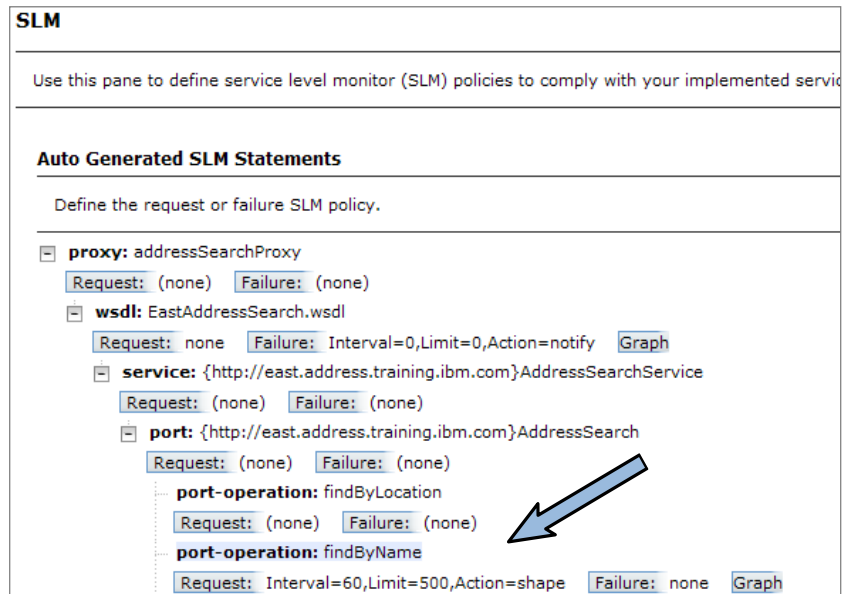
© Copyright IBM Corporation 2017

Figure 2-34. Proxy settings (4 of 4)

The Authorization AAA policy specifies how incoming messages are authenticated and authorized. The proxy AAA policy is applied for all service endpoints within the proxy.

Web service proxy: SLM Policy tab

- Click the **SLM Policy** tab to monitor requests that enter the web service proxy
 - Provides monitoring at a fine-grained level
 - Controls traffic that enters the web service proxy by using the **Throttle** and **Shape** action
 - Can view graph to see results of the traffic



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-35. Web service proxy: SLM Policy tab

Under **Request**, you can count the number of transactions that occur with a specific **interval** (in seconds). If the transaction **limit** is exceeded, you can specify an **action** to:

- **Notify**: Generate a log message when the transaction limit is exceeded.
- **Throttle**: Any transactions above the limit are rejected, and log messages are generated.
- **Shape**: The first 2500 transactions in excess of the maximum transaction rate are queued for later transmission, and subsequent transactions in excess of the 2500 limit are rejected. Log messages are generated.

Under **Failure**, you can specify the same information as **Request**, except that these settings apply to error messages.

WSDL cache policy

- Create a **WSDL cache policy** to update the WSDL proxy with changes from underlying WSDL file
 - Scheduled poll of underlying WSDL
 - If changes are detected, then the proxy WSDL is automatically updated
- Option is available only from **Objects** view:
 - Click **Objects > Service Configuration > Web Service Proxy**
 - Click an existing web service proxy
 - Use the arrows at the top to select the **WSDL Cache Policy** tab

Configure Web Service Proxy

Web Service Proxy: AddressSearchProxy [up]

Apply Cancel Delete Undo

Export | View Log | View Status | Show Probe | Validate Conformance | Help Quiesce | Unquiesce

WSDL Cache Policy

| URL Match expression | TTL | |
|---|-----|--|
| http://myWSDLserver.com/AddressSearch/* | 900 | |

Add

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-36. WSDL cache policy

Troubleshooting a web service proxy

Check active web service operations by using **Status > Web Service > Web Services Operations**

| WSProxy | Index | Interface | Port | URL | SOAP Action | SOAP Body | Status | |
|--------------------|-------|--------------|------|--------------------|-------------|--|------------|------|
| AddressSearchProxy | 0 | 172.16.78.44 | 6975 | /EastAddressSearch | | {http://east.address.training.ibm.com}retrieveAll | Registered | http |
| AddressSearchProxy | 1 | 172.16.78.44 | 6975 | /EastAddressSearch | | {http://east.address.training.ibm.com}findByLocation | Registered | http |
| AddressSearchProxy | 2 | 172.16.78.44 | 6975 | /EastAddressSearch | | {http://east.address.training.ibm.com}findByName | Registered | http |
| AddressSearchProxy | 3 | 172.16.78.44 | 6975 | /WestAddressSearch | | {http://west.address.training.ibm.com}retrieveAll | Registered | http |
| AddressSearchProxy | 4 | 172.16.78.44 | 6975 | /WestAddressSearch | | {http://west.address.training.ibm.com}findByLocation | Registered | http |

List of validation checks for web service proxy

- Request
 - Web service proxy active and listening on port
 - Verify that client submitted correct URI
 - Web service proxy received request (system log, probe)
 - SOAPAction header must agree with operation name in SOAP body
 - Passed automatic schema validation (user policy)
 - Back-end service active and available (system log)
 - Request is transmitted to correct back-end URL (system log)
- Response
 - Response is received from back-end service (system log)
 - Response passed automatic schema validation (user policy)
 - Response is transmitted completely to client (system log, probe)

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-37. Troubleshooting a web service proxy

Unit summary

- Describe the web service proxy architecture
- List and explain the configuration steps that are needed to create a web service proxy
- Create and configure a web service proxy policy at various levels of the WSDL file

Web service proxy service

Figure 2-38. Unit summary

© Copyright IBM Corporation 2017

Review questions



1. True or False: A web service proxy and an SLM policy can be defined at a fine-grained level.
2. Which of the following levels can be configured with a web service proxy policy?
 - A. Proxy
 - B. Message
 - C. Service
 - D. Port
3. True or False: A WSDL must be uploaded onto the gateway when creating a web service proxy.
4. List the three options under the SOAPAction policy:
 - A. **lax**: This option validates messages with an empty SOAPAction HTTP header or an empty string within the SOAPAction HTTP header.
 - B. **strict**: The message must contain an exact match of the SOAPAction header that is provided in the WSDL file.
 - C. **off**: The SOAPAction HTTP header is ignored.
 - D. **lazy**: The SOAPAction allows all messages through.

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-39. Review questions

Write your answers here:

- 1.
- 2.
- 3.
- 4.

Review answers (1 of 2)



1. True or False: A web service proxy and an SLM policy can be defined at a fine-grained level.

The answer is True.

2. Which of the following levels can be configured with a web service proxy policy?

- A. Proxy
- B. Message
- C. Service
- D. Port

The answer is A, C, and D.

Review answers (2 of 2)



3. True or False: A WSDL must be uploaded onto the gateway when creating a web service proxy.

The answer is False. A WSDL can be uploaded onto the gateway when creating a web service proxy, but it can also be retrieved from a subscription.

4. List the three options under the SOAPAction policy:

- A. lax: This option validates messages with an empty SOAPAction HTTP header or an empty string within the SOAPAction HTTP header.
- B. strict: The message must contain an exact match of the SOAPAction header that is provided in the WSDL file.
- C. off: The SOAPAction HTTP header is ignored.
- D. lazy: The SOAPAction allows all messages through.

The answer is A, B, and C.

Exercise: Configuring a web service proxy

Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-42. Exercise: Configuring a web service proxy

Exercise objectives

- Configure a web service proxy to virtualize an existing web service
- Configure the service policy within the web service proxy

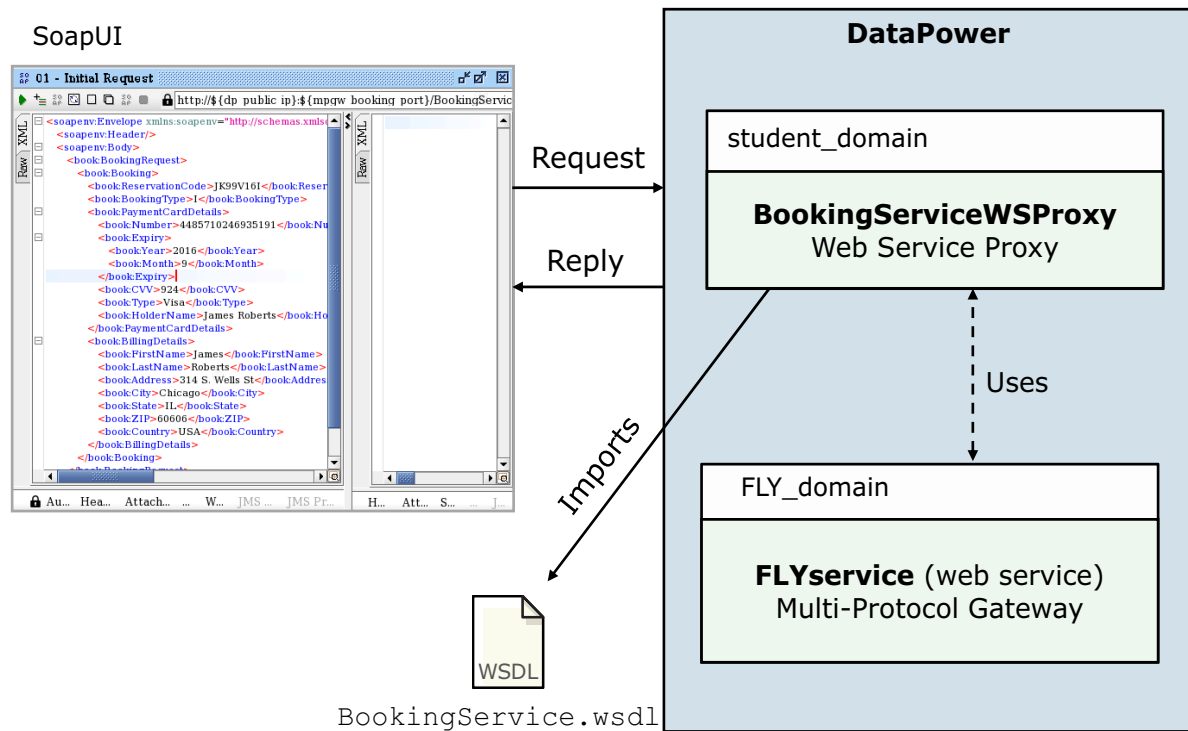


Web service proxy service

Figure 2-43. Exercise objectives

© Copyright IBM Corporation 2017

Exercise overview



Web service proxy service

© Copyright IBM Corporation 2017

Figure 2-44. Exercise overview

Unit 3. Course summary

Estimated time

00:15

Overview

This unit summarizes the course and provides information for further study.

Unit objectives

- Explain how the course met its learning objectives
- Access the IBM Training website
- Identify other IBM Training courses that are related to this topic
- Locate appropriate resources for further study

Course summary

© Copyright IBM Corporation 2017

Figure 3-1. Unit objectives

Course objectives

- Use the XML Signature and XML Encryption capabilities within DataPower to support WS-Security functions in your DataPower services
- Configure a web service proxy by using a WSDL file to proxy web services-based applications
- Configure the service policy of a web service proxy that supports different behaviors at the various levels of the WSDL

[Course summary](#)

© Copyright IBM Corporation 2017

Figure 3-2. Course objectives

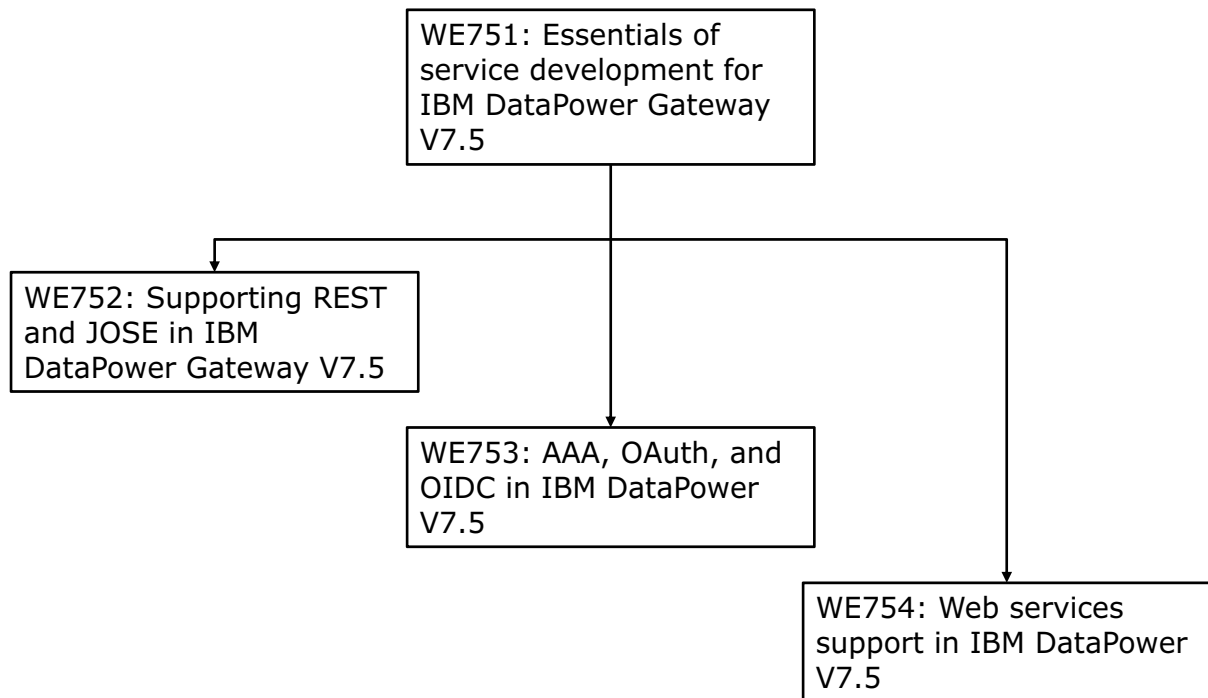
Lab exercise solutions

- Solutions are available in the **Solution** subdirectory:

`<lab_files>/Solutions`

- Remember to change
 - Port numbers
 - Back-end server (**Network > Interface > DNS Settings > Static Hosts**)
 - Front IP addresses (**Network > Interface > Host Alias**)

Curriculum relationship



[Course summary](#)

© Copyright IBM Corporation 2017

Figure 3-4. Curriculum relationship

To learn more on the subject

- IBM Training website:
<http://www.ibm.com/training>
- Webcast: How to define developer resources in DataPower Virtual Edition for Developers v7
<http://youtu.be/EaQyQWwQVIY>
- Webcast: VW750, Technical Introduction to IBM WebSphere DataPower Gateway Appliance V7.5.0
<https://youtu.be/yYk5Bzuie4g>
https://mediacenter.ibm.com/media/t/1_fb2tsml1
- DataPower V7.5 documentation in the IBM Knowledge Center
http://www.ibm.com/support/knowledgecenter/SS9H2Y_7.5.0
- IBM Redbooks:
<http://www.redbooks.ibm.com>
Search on “DataPower”
- developerWorks articles:
<http://www.ibm.com/developerworks/>
Search on “DataPower”

Course summary

© Copyright IBM Corporation 2017

Figure 3-5. To learn more on the subject

Enhance your learning with IBM resources

Keep your IBM Cloud skills up-to-date

- IBM offers resources for:
 - Product information
 - Training and certification
 - Documentation
 - Support
 - Technical information



- To learn more, see the IBM Cloud Education Resource Guide:
 - www.ibm.biz/CloudEduResources

Course summary

© Copyright IBM Corporation 2017

Figure 3-6. Enhance your learning with IBM resources

Unit summary

- Explain how the course met its learning objectives
- Access the IBM Training website
- Identify other IBM Training courses that are related to this topic
- Locate appropriate resources for further study

Course summary

Figure 3-7. Unit summary

© Copyright IBM Corporation 2017

Course completion



You have completed this course:

Web Services Support in IBM DataPower V7.5

Do you have any questions?

[Course summary](#)

© Copyright IBM Corporation 2017

Figure 3-8. Course completion

Appendix A. List of abbreviations

A

AAA authentication, authorization, and auditing

D

DNS Dynamic Name Server

F

FSH front side handler

G

GUI graphical user interface

H

HMAC hash message authentication code

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

HTTPS HTTP over SSL

I

IETF Internet Engineering Task Force

IP Internet Protocol

J

JMS Java Message Service

JOSE JSON Object Signing and Encryption

JSON JavaScript Object Notation

L

LDAP Lightweight Directory Access Protocol

M

MPGW Multi-Protocol Gateway

O

OAuth Open standard for Authorization

| | |
|-------------|--|
| OIDC | OpenID Connect |
| P | |
| PDF | Portable Document Format |
| R | |
| REL | Rights Expression Language |
| REST | Representational State Transfer |
| RSA | Rational Software Architect |
| S | |
| SAML | Security Assertion Markup Language |
| SHA | secure hash algorithm |
| SLM | service level management |
| SLM | service level monitoring |
| SOAP | Usage note: SOAP is not an acronym; it is a word in itself (formerly an acronym for Simple Object Access Protocol) |
| SPVC | self-paced virtual classroom |
| SSL | Secure Sockets Layer |
| T | |
| TLS | Transport Layer Security |
| U | |
| UDDI | Universal Description, Discovery, and Integration |
| URI | Uniform Resource Identifier |
| URL | Uniform Resource Locator |
| W | |
| W3C | World Wide Web Consortium |
| WS | web services |
| WSDL | Web Services Description Language |
| WSP | web service proxy |
| WWW | World Wide Web |
| X | |
| XML | Extensible Markup Language |

| | |
|--------------|---|
| XMLDS | XML digital signature |
| XPath | XML Path Language |
| XSL | Extensible Stylesheet Language |
| XSLT | Extensible Stylesheet Language Transformation |



IBM Training

