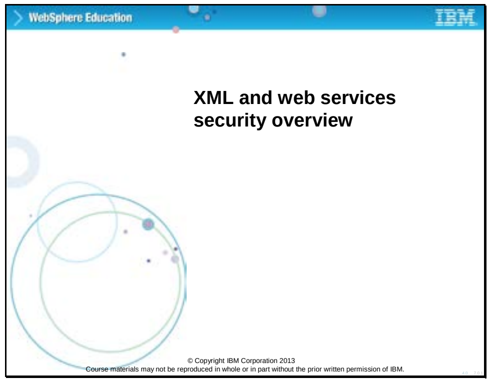
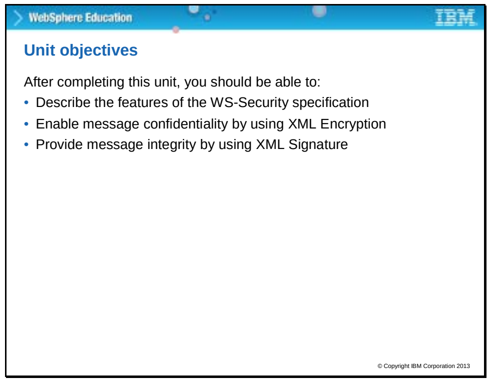
Slide 1


WebSphere Education
IBM
XML and web services
security overview
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 2

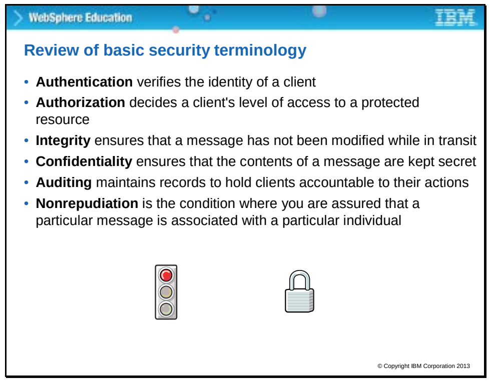


This presentation talks about how to ensure message confidentiality by using the **Encrypt** and **Decrypt** actions, and how to ensure message integrity by using the **Sign** and **Verify** actions. The objects that are required in these actions were covered in the crypto tools unit. Remember the purpose of the objects that are being referenced.

In addition, recall that unlike SSL, web services security allows you to encrypt or sign messages at a field level.

When you are using both encrypt and sign in a service policy, the good solution is to sign and then encrypt. If you encrypt, and then sign, hackers can replace their signature in the message. Encrypt the digital signature.

Slide 3



Review of basic security terminology

First, look at some of the terminology that is commonly encountered in this arena.

Authentication at it is simplest is asking, “who are you?” It is about verifying the identity of a client.

Authorization might ask, “what are you allowed to do?” It decides a clients level of access to a protected resource.

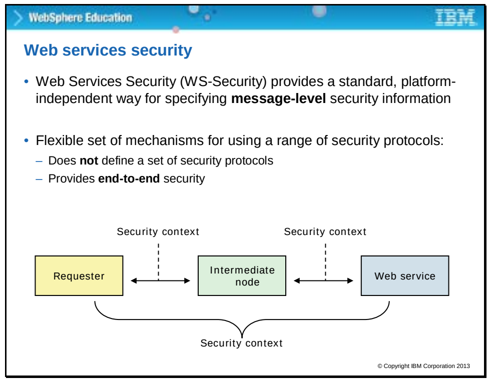
Integrity asks, “is the message intact?” In other words, ensuring that a message is not modified while in transit.

Confidentiality might be asking, “has anyone else looked at it?” It ensures that the contents of a message are kept secret.

Auditing tracks what was done. It maintains records to hold clients accountable to their actions.

Nonrepudiation is a fancy way of saying “non-denial.” It allows the client to prove that the server received a previously sent message, or vice versa.

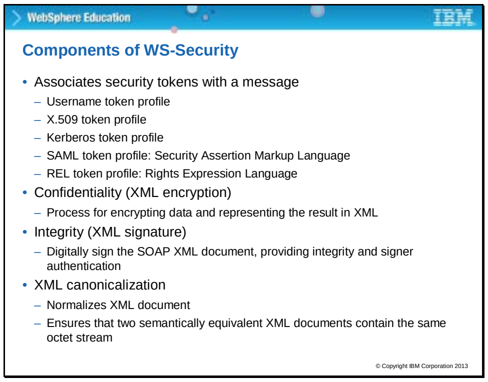
Slide 4



Web services security

The philosophy of web services security defines a common framework for propagating security between different entities. It does not matter whether the entities that are participating in a security environment are at the beginning and the end of the flow, or if they are intermediate entities in the middle of the flow. The idea is that a consistent security environment can be provided across the entire network regardless of who or what is participating, and whereabouts in the flow they participate.

Slide 5



Components of WS-Security

Components of web services security are as follows.

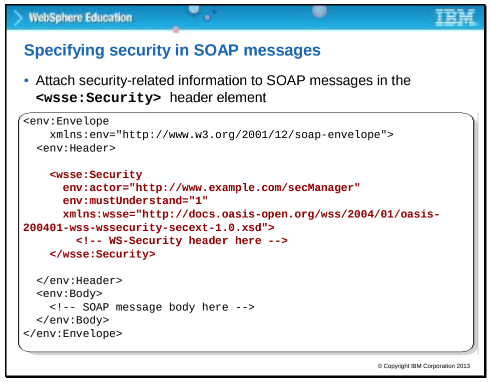
First, there is a way to associate security tokens with a message. By using a username token profile, or an X.509 token profile. Or a Kerberos token profile, or a Security Assertion Markup Language token profile. Or a Rights Expression Language token profile. Security Assertion Markup Language is commonly referred to by its acronym, S-A-M-L, usually pronounced “sammel,” and you often see Rights Expression Language that is written as “R-E-L.”

Another component of WS-Security defines a way to propagate confidentiality by “XML encryption” which is a process for encrypting data and representing the result in XML. Traditional encryption techniques usually result in a series of bytes containing arbitrary binary values, which are not compatible with the strict alphabetic contents of XML documents. So the WS-Security specification for XML Encryption defines a way to represent binary data as a series of alphabetic characters that can be encapsulated inside a well-formed XML document.

Document Integrity is usually achieved by generating a digital signature. Similar to encryption, digital signatures are typically binary bytes. Similar to the encryption standards, the WS-Security XML signature specification outlines ways to digitally sign a SOAP XML document, storing the results as properly tagged XML elements, providing integrity and signer authentication.

XML canonicalization is a fancy term that means normalizing XML text such that two semantically equivalent XML documents contain the same octet stream. For example, if one takes a typical HTML web page, and one is defining the display characteristics of an image, one might say “height=100, width=200”. On another page, one might define the exact same image as “width=200, height=100”. Any web browser does interpret those two phrases the same. It does not matter what sequence they appear in. They are semantically identical. But if one is to apply a digital signature to them both, the results would be different, since the digital signature algorithm relies on the position of each character and its value. So canonicalization is a set of rules that defines how such phrases are sequenced, and how they are represented. It might say, for instance, that height always comes before width, and that there must be no spaces between the name, the equal sign, and the value. By applying canonicalization to the second phrase in the example, it would end up the same as the first. More importantly, it would yield the same digital signature is both cases.

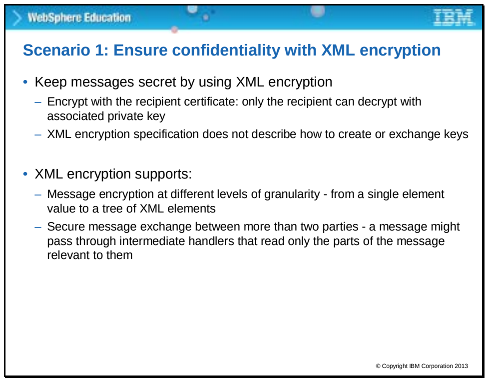
Slide 6



Specifying security in SOAP messages

To specify security in a SOAP message, you include the WS-Security header elements as shown in this example.

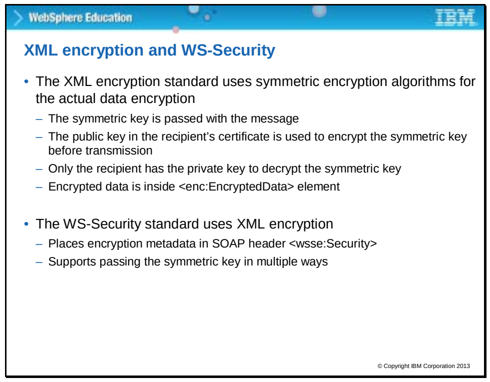
Slide 7



Scenario 1: Ensure confidentiality with XML encryption

Now look at some scenarios that might use WS-Security. The first is showing how one can ensure confidentiality by keeping messages secret by using XML encryption. One would encrypt the message with the public key in the recipient certificate, which means that only the recipient can decrypt it with their associated private key. The XML encryption specification does not describe how to create exchange keys. The XML encryption specification supports message encryption at different levels of granularity, from a single element value to a tree of XML elements, to the entire message. Secure message exchange between more than two parties can be accomplished by encrypting different parts with different keys or algorithms. In this way, a message might pass through intermediate handlers that read only the parts of the message relevant to them. For example, one might encrypt a credit card number with one key and the customer account number with another. One might then have a DataPower service that can access the account number to do some validation, but it does not see the credit card number. So only the account number key would be made available to the DataPower service, and the credit card section would be passed through untouched.

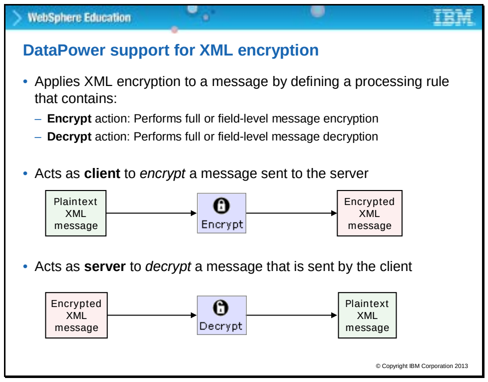
Slide 8



- ***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 recipients 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***

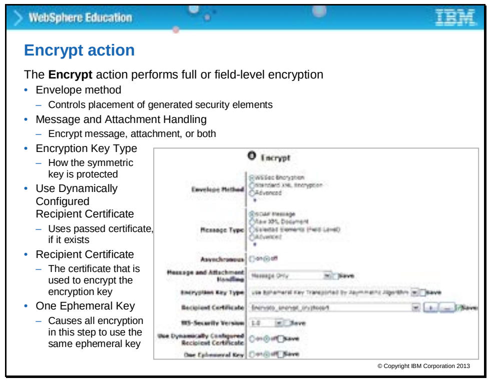
Slide 9



DataPower support for XML encryption

DataPower supports XML encryption and decryption by the use of the Encrypt and Decrypt action icons in the policy rule. They can both be configured to operate on a whole message or selected elements of a message.

Slide 10

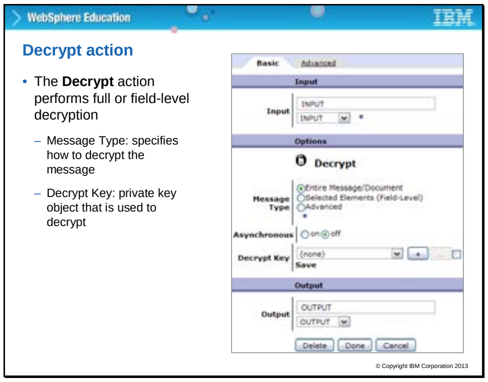


Encrypt action

The Encrypt action has a number of parameters, including:

- Envelope method, which controls placement of generated security elements.
- Message Type, which specifies the style that is used to encrypt messages.
- Message and Attachment Handling – whether to encrypt just the message, just the attachment, or both.
- Use Dynamically Configured Recipient Certificate, which defines whether to use the certificate from a previous Verify action, if it exists.
- One Ephemeral Key, which causes all encryption in this step to use the same ephemeral key.
- Recipient Certificate, which specifies which certificate is used to perform encryption.

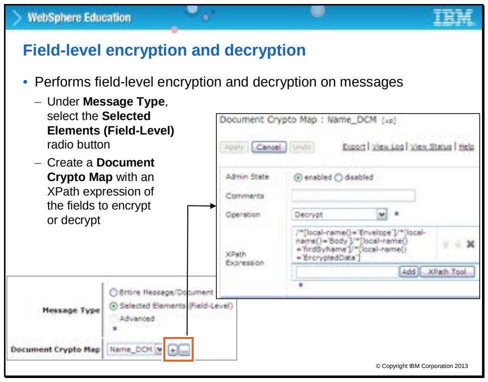
Slide 11



Decrypt action

Just like the Encrypt Action, the Decrypt Action can also act upon the entire message or just parts of it.

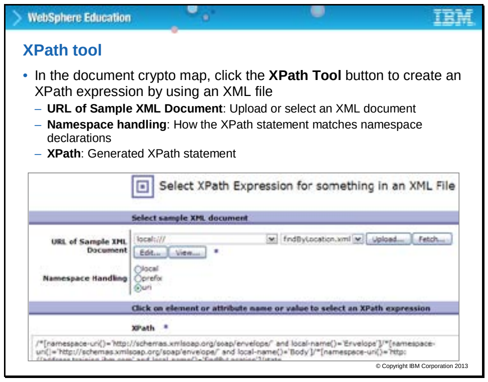
Slide 12



Field-level encryption and decryption

When it comes to doing field-level encryption and decryption, you create a “document crypto map” that specifies which parts to process. The Document Crypto Map, or “DCM” as commonly abbreviate it, consists of a series of XPath expressions that define which elements are used.

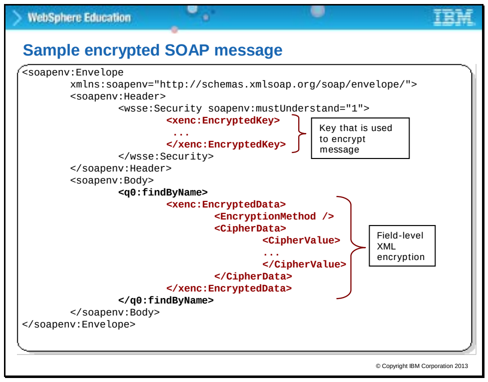
Slide 13



XPath tool

To help the creation of the XPath expressions, an XPath tool is if allows you to provide the URL of a sample XML document that is uploaded. Or you can select an existing XML document that is already in the DataPower file system. Next, you specify the namespace handling, for example, how the XPath statement matches namespace declarations. When you select the element in the sample document, the generated XPath statement appears.

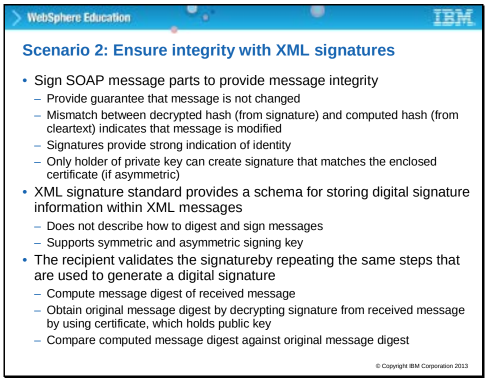
Slide 14



Sample encrypted SOAP message

Here is an example of how an encrypted SOAP message might appear. The “xenc” prefix denotes elements that form part of the encrypted data. The key that is used for encryption appears in the SOAP header, and the actual encrypted data is in the SOAP body, with an element name of “EncryptedData”.

Slide 15



Scenario 2: Ensure integrity with XML signatures

And now for the second scenario, you see how to ensure integrity with XML signatures.

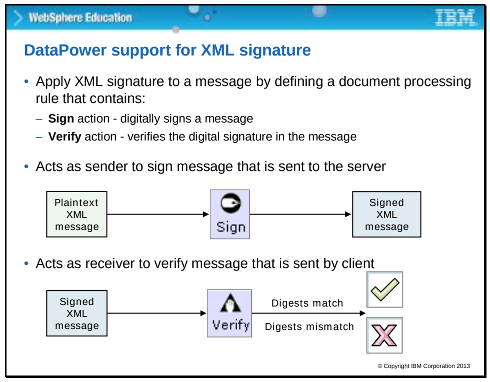
You can sign SOAP message parts to provide message integrity; in other words, you are providing an assurance that the message has not changed. Any mismatch between the decrypted hash from the signature and the computed hash from the original clear text would indicate that the message was modified.

Digital signatures also provide a strong indication of identity, since only the holder of the private key can create signature that matches the enclosed certificate.

The XML signature standard provides a schema for storing digital signature information within XML messages, but does not describe how to digest and sign messages. That is up to you.

The signature is validated by the recipient by repeating the same steps that are used to generate a digital signature. First, they compute the message digest of the received message. Then, they obtain the original message digest by decrypting the signature from the received message by using the certificate, which holds the public key. Finally, they compare the computed message digest against the original message digest. If they match, they know that the message is intact.

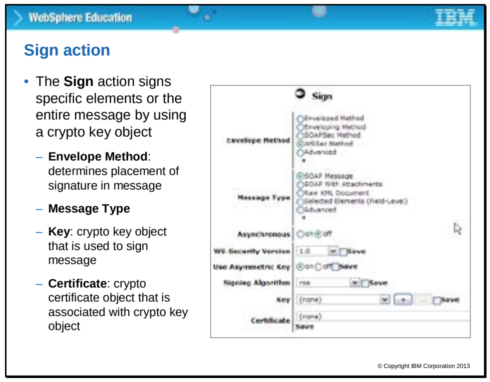
Slide 16



DataPower support for XML signature

In a similar fashion to XML encryption support, DataPower also supports digital signature creation and verification. The “Sign” action can be added to a policy rule to generate a signature, and the “Verify” action can be used to check an existing signature. If the Verify action fails, it throws an error.

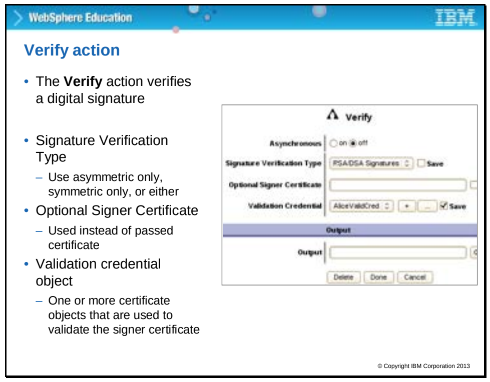
Slide 17



Sign action

The Sign action signs specific elements or the entire message by using a crypto key object. Various parameters can be set to control its actions. These parameters include as Envelope Method that determines the placement of the signature in a message. The Message Type parameter indicates the style that is used to sign the message. Key identifies the crypto key object that is used to sign the message. Certificate specifies the crypto certificate object that is associated with the crypto key object.

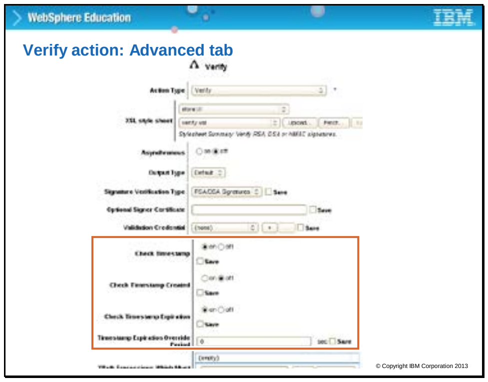
Slide 18



Verify action

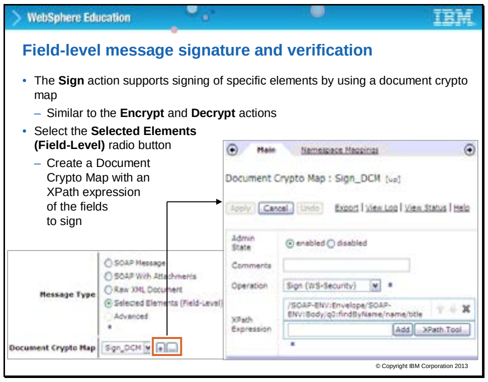
The Verify Action has two tabs – the Basic one identifies the validation credential to use.

Slide 19



Verify action: Advanced tab
The Verify Action “Advanced” tab includes more parameters that specify how the signature is verified. The parameters include certificate details, validation credentials, timestamp criteria, signature reuse, and more.

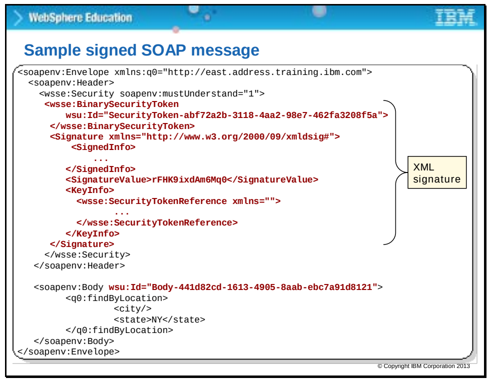
Slide 20



Field-level message signature and verification

Just like the Encrypt Action, the Sign Action can do field-level signature generation, and it also uses a “document crypto map” that specifies which parts are processed. As you no doubt remember, the “DCM”, consists of a series of XPath expressions that define which elements are used, and the XPath tool can generate the XPath expression list for you from a sample document.

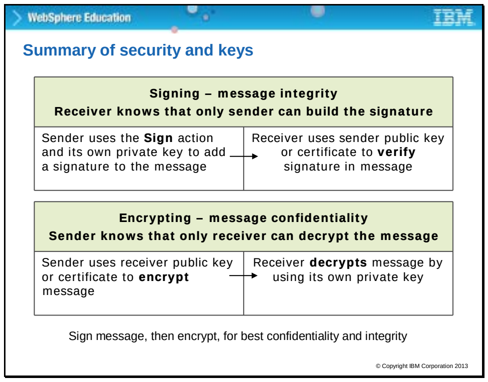
Slide 21



Sample signed SOAP message

Here is an example of a signed SOAP document, showing how the XML Signature information is embedded in the SOAP header and references a “wsu” prefix that appears in the document body with the signature ID.

Slide 22



Summary of security and keys

Here is a summary of everything that is covered regarding signing and encrypting messages.

Remember that these two things are opposite functions. You sign by encrypting with a private key because then your message can be opened with the equivalent public key, thus proving that the sender is who they say they are. You encrypt with a public key because then your message can be decrypted only with the equivalent private key. The receiver is the only one who can read the message.

Slide 23

WebSphere Education

IBM

Unit summary

Having completed this unit, you should be able to:

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

Checkpoint questions

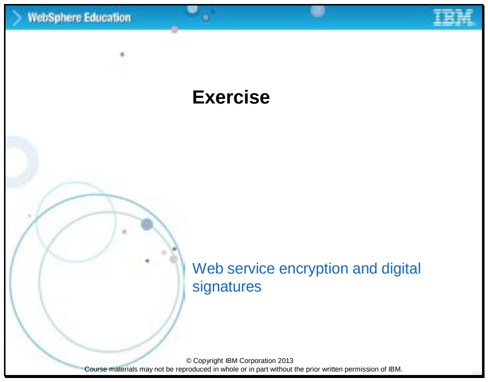
1. True or False: A document crypto map is used to specify an XPath expression that contains the elements to encrypt, decrypt, sign, and verify.
2. True or False: Encryption and decryption can occur at both the message and field levels, but sign and verify occur at the message level only.
3. True or 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 **Signer Certificate** field.

Slide 25

Checkpoint answers

1. 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. False. Both scenarios are supported, even though the Verify action does not have a selected field-level radio button.
3. True. 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 **Signer Certificate** field.

Slide 26


WebSphere Education
IBM
Exercise
Web service encryption and digital signatures
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 27

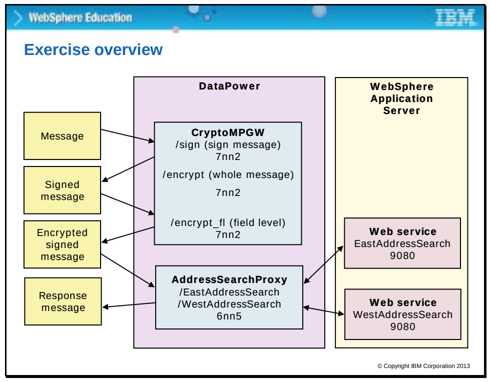
WebSphere Education

Exercise objectives

After completing this exercise, you should be able to:

- Create a multi-protocol gateway to generate a message with XML encryption
- Create a multi-protocol gateway to generate a message with an XML digital signature
- Perform field-level encryption and decryption on XML messages
- Create a rule to decrypt messages and verify digital signatures that are contained in a message within a web service proxy policy

Slide 28


WebSphere Education
IBM
Exercise overview
DataPower
WebSphere Application Server
Message
Signed message
Encrypted signed message
Response message
CryptoMPGW
/sign (sign message)
7nn2
/encrypt (whole message)
7nn2
/encrypt_fl (field level)
7nn2
AddressSearchProxy
/EastAddressSearch
/WestAddressSearch
6nn5
Web service
EastAddressSearch
9080
Web service
WestAddressSearch
9080
© Copyright IBM Corporation 2013