



IBM Software Group

# Security in SOAP nodes in WebSphere Message Broker V6.1

**Vivek Grover**

[vgrover@us.ibm.com](mailto:vgrover@us.ibm.com)

**Minsung Byun**

[mbyun@us.ibm.com](mailto:mbyun@us.ibm.com)

**WebSphere Message Broker Level 2 Support, IBM**



WebSphere® Support Technical Exchange



# Agenda

- **Introduction**
- **Transport Security Basics**
- **Transport Security Configuration**
  - ▶ Transport security
  - ▶ Configuration of SOAPRequest nodes
  - ▶ Configuration of SOAPInput nodes
- **Message security**
  - ▶ WS-security concepts
  - ▶ WS-security configuration

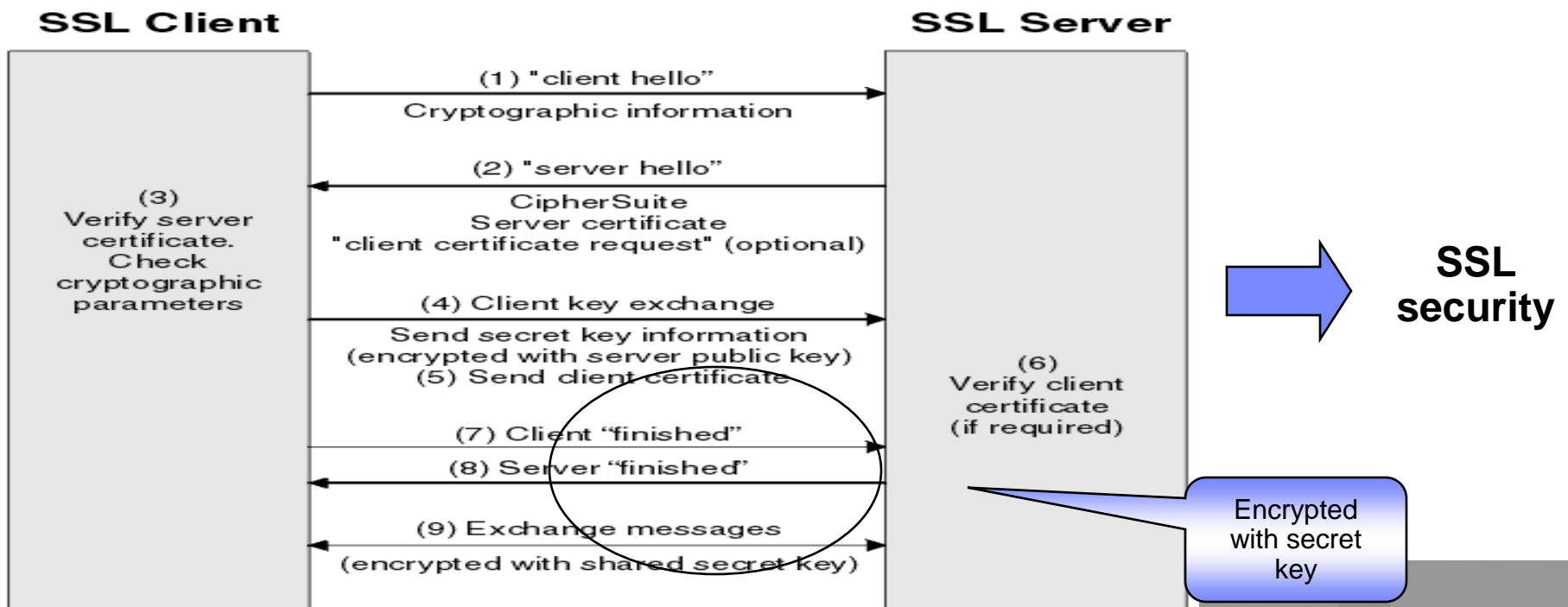
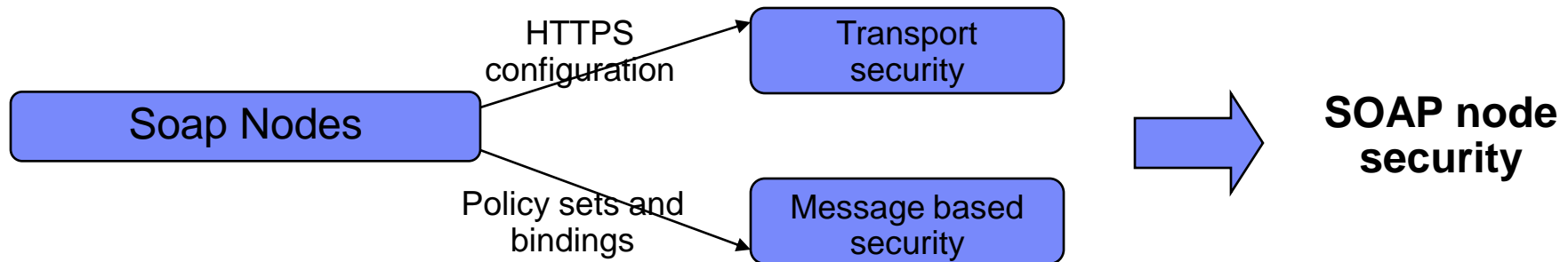


# Introduction

- **Different types of message security**
  - ▶ Transport Security
    - SSL
    - TLS
  - ▶ Network Security
    - IPSec
  - ▶ Message based security
    - WS-Security
  
- **SOAP nodes can use Transport security and/or Message based security**

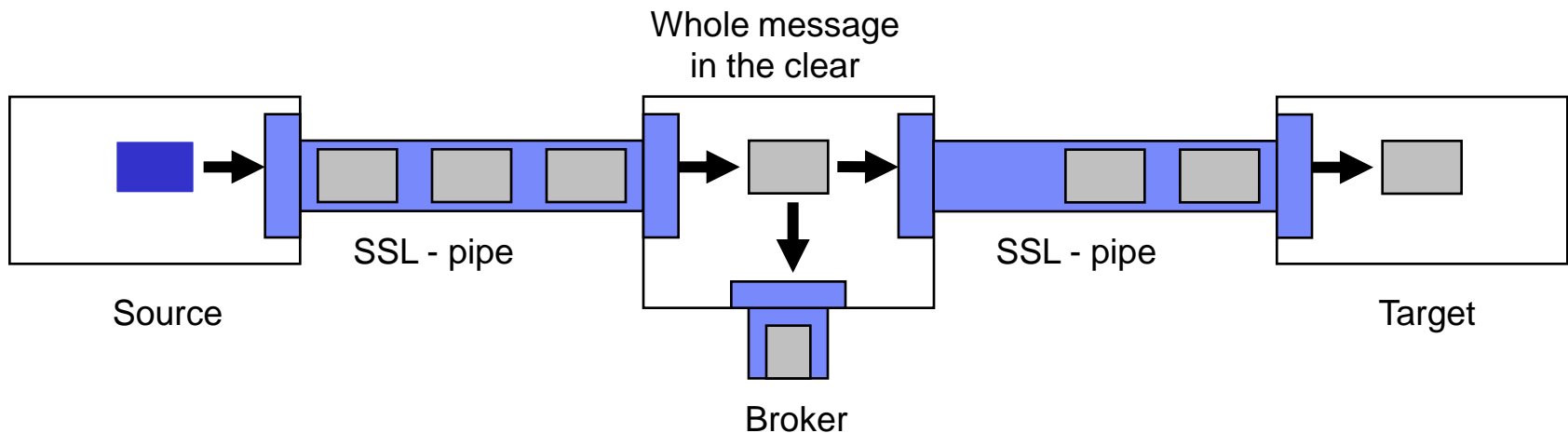


# Introduction



# Transport Security Basics

- Protects the stream of data being passed from one endpoint to another
- Typically ALL of the data is encrypted with same key
- Does not discriminate on a per message basis (everything is encrypted) with same key
- Only Point-to-point Security
  - ▶ When passing messages via another intermediary, if the intermediary needs to 'see' any part of the message, it can access all of it



# Transport Security Basics

- In v6.1 there is a hierarchy for defining the keystores and truststores that SOAP nodes in an execution group will use
- Keystores can be defined at the:
  - ▶ Broker registry level - applies to all users of keystores in all execution groups, eg. SOAPInput, SOAPRequest/AsyncRequest (for client Auth), AsyncResponse
  - ▶ Execution group JVM Manager level - applies to all users of keystores in specified execution groups, eg. SOAPInput, SOAPRequest/AsyncRequest (for clientAuth), AsyncResponse
  - ▶ httpsConnector level - applies to httpsConnector users in the specified execution group, eg. SOAPInput, AsyncResponse
- Settings at httpsConnector level override those at the execution group jvm manager level which in turn override those at the broker registry level



# Transport Security Basics

- Truststores can be defined at the:
  - ▶ Broker registry level - applies to all users of truststores in all execution groups, eg. SOAPInput (for client Auth), SOAPRequest/AsyncRequest nodes
  - ▶ Execution group JVM Manager level - applies to all users of truststores in the specified execution groups, eg SOAPInput (for client Auth), SOAPRequest/AsyncRequest nodes
- To enable clientAuth for SOAPInput nodes set clientAuth = true at the HTTPSCConnector level
- Settings at the Execution group JVM Manager level override those at the broker registry level
- Certificate Management Tools are available with IBM JRE
  - Command line tool – keytool (<WMB install directory>/jre/bin)
  - GUI-based tool – Ikeyman (<WMB install directory>/jre/bin)



# Transport Security Configuration - SOAPRequest

## Configuration of SOAPRequest & SOAPAsyncRequest nodes

- ▶ Create the required keystores and truststores and exchange & add certificates
- ▶ Configure the broker registry

OR

- ▶ Configure the JVM Manager for Execution Group containing the message flow with these nodes
- ▶ Configure the message flow node and bar file
- ▶ Deploy the bar file to the broker





# Transport Security Configuration - SOAPRequest

## One-way SSL

### 1. Import the certificate provided by Provider into truststore

```
keytool -import -alias mykey -file <name of certificate file> -keystore <fully qualified path to truststore file> -storepass <password>
```

### 2. Set the broker to use the truststore

```
mqschangeproperties <broker name> -o BrokerRegistry -n brokerTruststoreFile -v <fully qualified truststore file>
```

### 3. Set the truststore password entry

```
mqschangeproperties <broker name> -o BrokerRegistry -n brokerTruststorePass -v brokerTruststore::password
```

### 4. Set the truststore password in the broker registry

```
mqsisetdbparms <broker name> -n brokerTruststore::password -u temp -p <password>
```



# Transport Security Configuration - SOAPRequest

## Two-way SSL

### 1. Import the certificate provided by Provider into truststore

```
keytool -import -alias mykey -file <name of certificate file> -keystore <fully qualified path to truststore file> -storepass <password>
```

### 2. Create a keystore with a new self-signed certificate

```
keytool -genkey -storepass <password> -keystore <keystore file> -alias <self-signed certificate>
```

### 3. Extract the certificate from the keystore file for Provider

```
keytool -export -alias tomcat -file <name of certificate file> -keystore <keystore file> -storepass <password>
```

## Configure the broker registry

### 4. Point the broker to the new keystore file

```
mqsichangeproperties <broker name> -o BrokerRegistry -n brokerKeystoreFile -v <fully qualified keystore file>
```

### 5. Point the broker to truststore file

```
mqsichangeproperties <broker name> -o BrokerRegistry -n brokerTruststoreFile -v <fully qualified truststore file>
```



# Transport Security Configuration - SOAPRequest

## 6. Set the broker keystore password

```
mqschangeproperties <broker name> -o BrokerRegistry -n brokerKeystorePass -v  
brokerKeystore::password
```

## 7. Set the broker truststore password

```
mqschangeproperties <broker name> -o BrokerRegistry -n brokerTruststorePass -v  
brokerTruststore::password
```

## 8. Set the keystore password in registry

```
mqssetdbparms <broker name> -n brokerKeystore::password -u temp -p <password>
```

## 9. Set the truststore password in registry

```
mqssetdbparms <broker name> -n brokerTruststore::password -u temp -p <password>
```

## 10. Restart the broker for changes to take effect



# Transport security Configuration - SOAPRequest

## ■ SOAPRequest node configuration

The screenshot shows the 'SOAPRequest node Properties' dialog box. The 'Web service URL' field is highlighted with a red error message: 'Web service URL: A value must be set for this property.' The 'Protocol' dropdown is set to 'SSL'. The 'Request timeout (in seconds)' is set to 120. The 'HTTP(S) proxy location' field is empty. The 'Allowed SSL ciphers (if using SSL)' field is empty.

| Property                           | Value   |
|------------------------------------|---|
| Web service URL*                   | <specify full service URL><br><i>e.g. http://server/path/to/service</i> |
| Request timeout (in seconds)       | 120   |
| HTTP(S) proxy location             | <enter your proxy server (if any)>                                      |
| Protocol (if using SSL)            | SSL   |
| Allowed SSL ciphers (if using SSL) | <enter any specific SSL Ciphers you wish to use>                        |

## ■ Select Protocol

- ▶ SSL (default until V6.1.0.3)
- ▶ SSLV3
- ▶ TLS (default after V6.1.0.4)

## ■ Optional - HTTPS proxy location and allowed SSL ciphers

# Transport Security Configuration - SOAPInput

## Configuration of SOAPInput nodes

- ▶ Create the required keystores and truststores
- ▶ Exchange and add certificates
- ▶ Configure the httpsconnector at ExecutionGroup level

OR

- ▶ Configure the JVM Manager for Execution Group containing the message flow with these nodes

OR

- ▶ Configure the broker registry
- ▶ Configure the message flow SOAPInput node
- ▶ Deploy the message flow



# Transport Security Configuration - SOAPInput

## One-way SSL

### 1. Create a keystore with a new self-signed certificate

```
keytool -genkey -storepass <password> -keystore <keystore file> -alias <self-signed certificate>
```

### 2. Extract the certificate from the keystore file for Requestor

```
keytool -export -alias tomcat -file <name of certificate file> -keystore <keystore file> -storepass <password>
```

## Two-way SSL – In addition to the above

### 3. Enable ClientAuth to true

```
mqsischangeproperties <broker name> -e <eg name> -o HTTPSConnector -n clientAuth -v true
```

### 4. Import the certificate provided by requester into truststore

```
keytool -import -alias mykey -file <name of certificate file> -keystore <fully qualified path to truststore file> -storepass <password>
```



# Transport Security Configuration - SOAPInput

## Configure the JVM Manager for Execution Group

### 1. Set the keystore file to the Execution Group

```
mqschangeproperties <broker name> -e <eg name> -o ComIbmJVMMManager -n keystoreFile -v <keystore  
file name>
```

### 2. Set the keystore password

```
mqschangeproperties <broker name> -e <eg name> -o ComIbmJVMMManager -n keystorePass -v  
brokerKeystore::password
```

### 3. Set the truststore file to the Execution Group

```
mqschangeproperties <broker name> -e <eg name> -o ComIbmJVMMManager -n truststoreFile -v <keystore  
file name>
```

### 4. Set the truststore password

```
mqschangeproperties <broker name> -e <eg name> -o ComIbmJVMMManager -n truststorePass -v  
brokerTruststore::password
```



# Transport Security Configuration - SOAPInput

## 5. Set the broker keystore password in registry

```
mqsisetdbparms <broker name> -n brokerKeystore::password -u temp -p <password>
```

## 6. Set the broker truststore password in registry

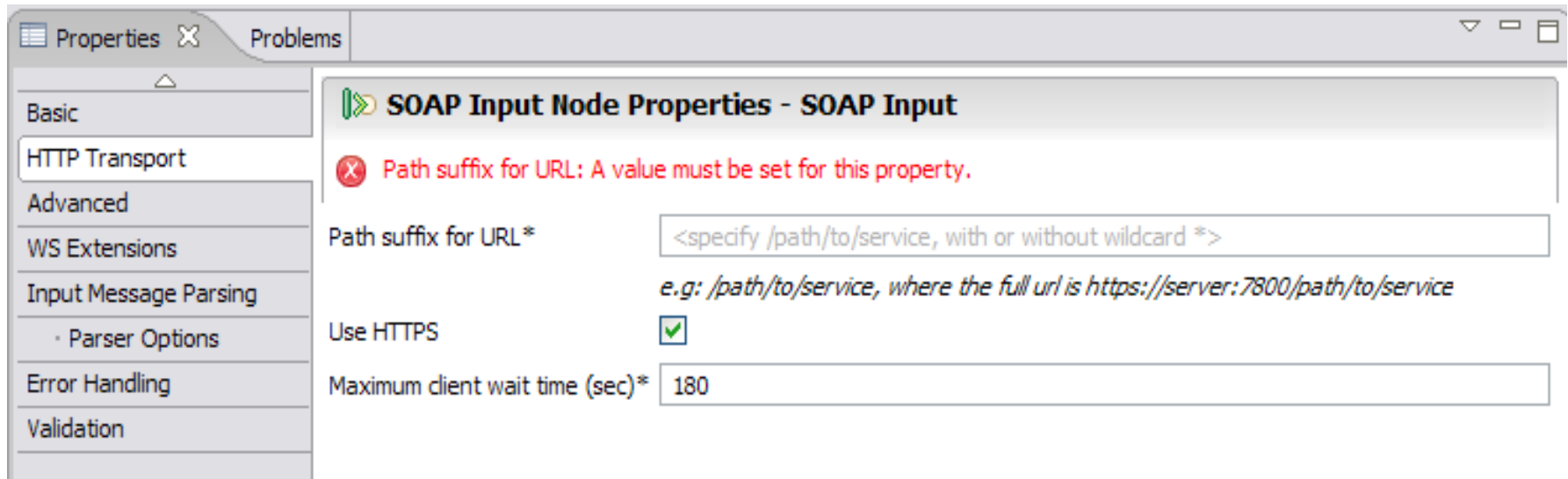
```
mqsisetdbparms <broker name> -n brokerTruststore::password -u temp -p <password>
```





# Transport security Configuration - SOAPInput

- SOAPInput node configuration



- Check Use HTTPS box in SOAPInput node properties
- If the address contains an https URL, the check box is automatically selected
- User can manually override this property value

# Known Problems

- Known problems with default protocols – fixed in Fixpack 04

SOAPRequest node defaults to SSL but SOAPInput node defaults to TLS

- Modify the SOAPRequest node to use TLS under Protocol setting in the "HTTP Transport" tab in the toolkit

Or

- Change the protocol on the execution group HTTPListener via command `mqschangeproperties <Broker name> -e <EG Name> -o HTTPSCConnector -n sslProtocol -v SSLv3`

- SOAPInput node ignores value set for keystoreFile at execution group HTTPSCConnector level – fixed in Fixpack03



# WS-Security



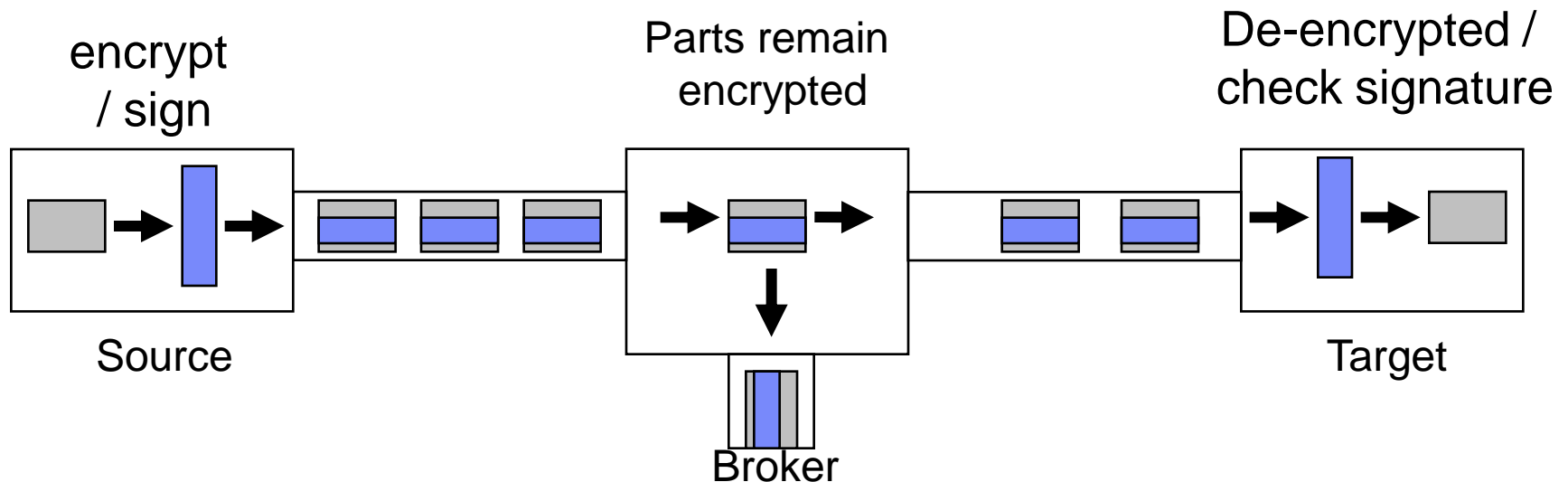
# Agenda

- **Introduction to WS-Security**
- **WS-Security Configuration**
  - ▶ The Policy Set Editor
  - ▶ Policy Set Assignment



# Introduction to WS-Security

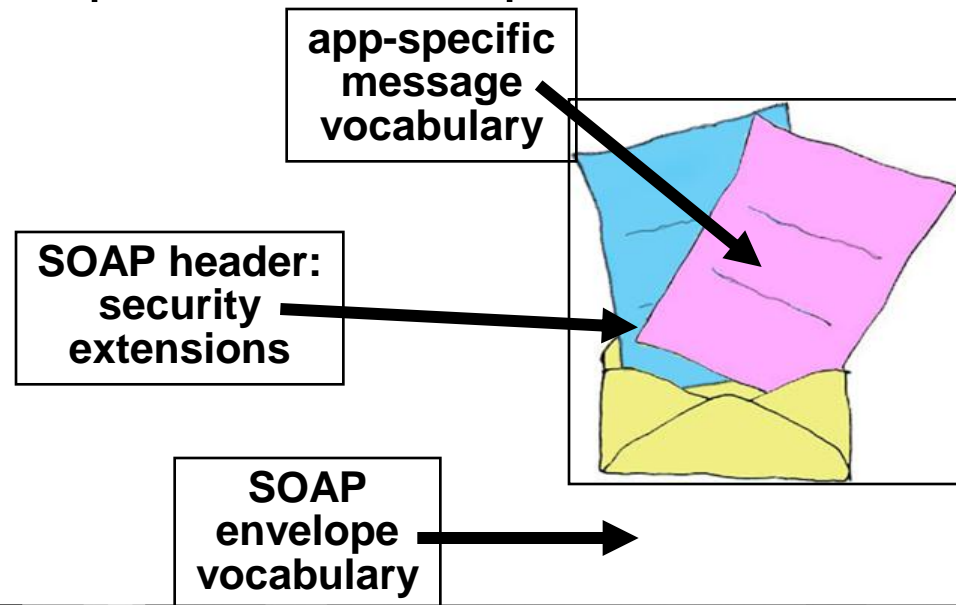
- Finer granularity
- Parts of the message may be encrypted in different ways with different keys
- Parts of a message may be (multiply) encrypted and signed
- WS-Security can be used in insecure transports



# Introduction to WS-Security

## Soap Message Structure

- The SOAP specification defines the “envelope” vocabulary
  - ▶ The "envelope" wraps the message itself
- WS-Security defines the <Security> element, which allows security extensions to be placed in <soapenv:header>
  - ▶ Username/password
  - ▶ X.509 certificate
  - ▶ Encryption details
  - ▶ XML Signature



# Introduction to WS-Security

## Mechanisms of securing web-services

- **Confidentiality**
  - ▶ Keep secrets
  - ▶ uses **message encryption** to ensure that no party or process can access the message
- **Integrity**
  - ▶ Prevent tampering
  - ▶ uses message signing to ensure that information is not changed, altered, or lost
  - ▶ XML **digital signature** is generated
  - ▶ signature is not validated if the data changes
- **Authentication**
  - ▶ verify that the identity is valid
  - ▶ Accessible and useable by an **authorized** entity
  - ▶ uses a security token to validate the user and determine whether a client is valid



# Introduction to WS-Security

The WS-Security specification defines a vocabulary that can be used inside the SOAP envelope

**<wsse:Security>** is the wrapper for security-related information

## Soap message with ws-security:

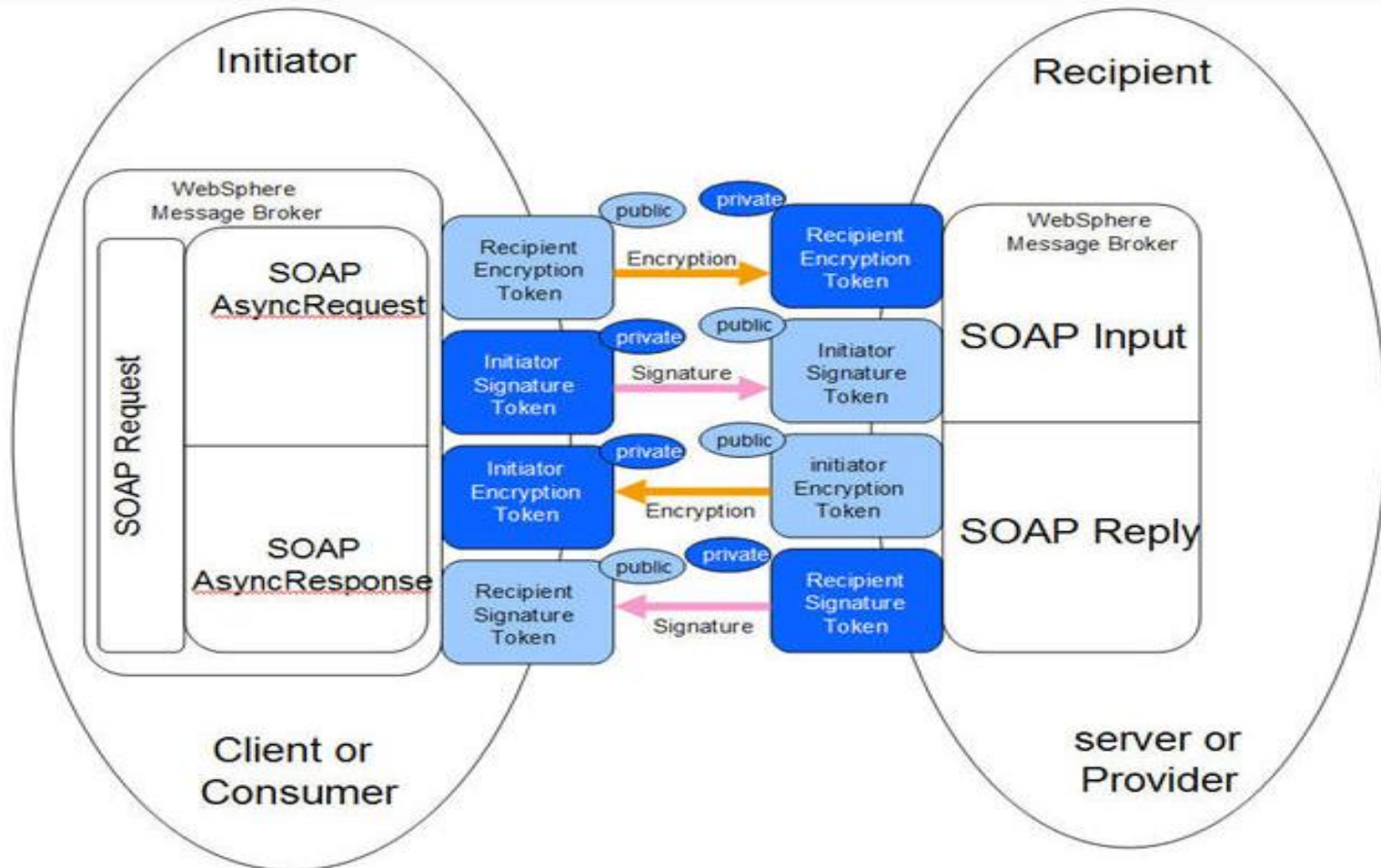
```
<Envelope
  xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Header>
    <wsse:Security
      <!-- ws-security namespace --!>
      xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/07/secext"
      <!-- Security Information for Authentication or
           XML Signature or XML Encryption is included here
      --!>
      <!-- Username token for Authentication looks like this..!>
      <wsse:UsernameToken wsu:ID="myToken">
        <wsse:Username>IBM</wsse:Username>
        <wsse:Password>p@$w0rd</wsse:Password>
        </wsse:UsernameToken>
        <!--...XML Digital Signature entries looks...!>
        <wsse:BinarySecurityToken EncodingType="wsse:Base64Binary">
          AIIgQtCC7ZxO5tlgerPcid1z ... [truncated]
        </wsse:BinarySecurityToken>
        <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
          ....signature data....
        </ds:Signature>
        <!--.....XML Encryption entries looks like...!>
        <xenc:EncryptedData xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
          <xenc:EncryptValue>akdfaknqerandfauydfrajndfh3k973...(Truncated)</xenc:EncryptValue>
        </xenc:EncryptedData>
      </wsse:Security>
    </Header>
    <Body> ..... </Body>
  </Envelope>
```



# Introduction to WS-Security

WS-Security concepts

Common terms used for all the SOAP nodes



# WS-Security Configuration

## WMB WS-Security Configuration Steps

- Set up the keystores and truststores
  - ▶ Use keytool / Ikeyman as explained in previous charts
- Create the policies
  - ▶ Use Policy Set Editor
  - ▶ Define any combination of Confidentiality, Authentication and Integrity
  - ▶ Define message parts
- Create provider and consumer policy set bindings
  - ▶ Define public and private keys for encrypting and decrypting data
- Configure and deploy the bar files



# WS-Security Configuration - Keystores

- View the contents of consumer's keystore

```
keytool -list -keystore client.keystore -storepass client -v
```

Keystore type: jks  
Keystore provider: IBMJCE

Your keystore contains 2 entries

Alias name: **servercert**  
Creation date: May 14, 2009  
Entry type: **trustedCertEntry**

Owner: **CN=server, O=Web Services Guided Tour, C=GB**

Issuer: CN=server, O=Web Services Guided Tour, C=GB

Serial number: 478b3c55

Valid from: 5/14/09 9:41 AM until: 9/31/36 9:41 AM

Certificate fingerprints:

MD5: 7E:5B:FD:31:DA:D1:81:44:74:28:56:0A:0E:0F:0F:0F

SHA1: B8:4D:85:79:9D:51:62:0F:3F:CC:9D:0A:0E:0F:0F:0F

\*\*\*\*\*

Alias name: **clientcert**

Creation date: May 14, 2009

Entry type: keyEntry

Certificate chain length: 1

Certificate[1]:

Owner: **CN=client, O=Web Services Guided Tour, C=GB**

Issuer: CN=client, O=Web Services Guided Tour, C=GB

Serial number: 478b3c49

Valid from: 5/14/09 9:41 AM until: 9/31/36 9:41 AM

Certificate fingerprints:

MD5: 63:49:B3:73:68:78:A7:44:54:94:61:25:7C:3F:7C:3C

SHA1: F9:0D:52:B4:9A:20:C9:2C:61:74:F5:CB:DE:7F:FF:3E:32:82:7F:17

\*\*\*\*\*

# WS-Security Configuration - Keystores

- View the contents of Provider's keystore

```
keytool -list -keystore server.keystore -storepass server -v
```

Keystore type: jks  
Keystore provider: IBMJCE

Your keystore contains 2 entries

Alias name: **servercert**  
Creation date: May 14, 2009  
Entry type: keyEntry  
Certificate chain length: 1  
Certificate[1]:  
Owner: **CN=server, O=Web Services Guided Tour, C=GB**  
Issuer: CN=server, O=Web Services Guided Tour, C=GB  
Serial number: 478b3c55  
Valid from: 5/14/09 9:41 AM until: 9/31/36 9:41 AM  
Certificate fingerprints:  
MD5: 7E:5B:FD:31:DA:D1:81:44:74:28:56  
SHA1: B8:4D:85:79:9D:51:62:0F:3F:CC:91  
\*\*\*\*\*

Alias name: **clientcert**  
Creation date: May 14, 2009  
Entry type: **trustedCertEntry**  
Owner: CN=client, O=Web Services Guided Tour, C=GB  
Issuer: CN=client, O=Web Services Guided Tour, C=GB  
Serial number: 478b3c49  
Valid from: 5/14/09 9:41 AM until: 9/31/36 9:41 AM  
Certificate fingerprints:  
MD5: 63:49:B3:73:68:78:A7:44:54:94:61:25:7C:3F:7C:3C  
SHA1: F9:0D:52:B4:9A:20:C9:2C:61:74:F5:CB:DE:7F:FF:3E:32:82:7F:17  
\*\*\*\*\*

# WS-Security Configuration – Policy sets

Create the Policies – To configure Encryption, Signing and Authorization



# WS-Security Configuration – Policy sets

**Policy Sets for "BK61"**

**Set up Policy Sets and Policy Set Bindings for this broker**  
Use these panels to define encryption and signing asymmetric tokens to apply to your message.

**Policy Sets**

- WSS10Default
  - WSSecurity
    - WS-Security
      - Authentication Tokens
      - Message Level Protection
- Policy Set Bindings
  - WSS10Default

**Add** **Delete**

Select message level protection to enable encryption and signing in this policy

☒ Message level protection

☐ Require signature confirmation

☐ Include timestamp in security header

Security header layout:

☒ Strict - declarations must precede use

☐ Lax - order of content can vary

☐ Lax but timestamp required first in header

☐ Lax but timestamp required last in header

**Finish** **Cancel**

2. Enable Message Part Protection

**Policy Sets for "BK61"**

**Set up Policy Sets and Policy Set Bindings for this broker**  
Use this panel to define asymmetric tokens which represents the public and private keys used for both signature and encryption.

**Policy Sets**

- WSS10Default
  - WSSecurity
    - WS-Security
      - Authentication Tokens
      - Message Level Protection
      - Tokens
      - Algorithms
      - Message Part Protection
- Policy Set Bindings
  - WSS10Default

**Add** **Delete**

**Message Integrity/ Confidentiality Policies**

| Token Name     | Token Type | WS-Security Version | X.509 Type      |
|----------------|------------|---------------------|-----------------|
| InitiatorToken | Initiator  | 1.0                 | X.509 Version 3 |
| RecipientToken | Recipient  | 1.0                 | X.509 Version 3 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |
|                |            |                     |                 |

**Add** **Delete**

**Finish** **Cancel**

3. Define Encryption and signing Tokens

# WS-Security Configuration – Policy sets

**Policy Sets for "BK61"**

**Set up Policy Sets and Policy Set Bindings for this broker**

Message security policies specify cryptographic algorithms available, allowable key lengths, as well as canonicalization algorithms for reconciling XML differences.

**Policy Sets**

- WSS10Default
  - WSSecurity
    - WS-Security
      - Authentication Tokens
        - Message Level Protection
          - Tokens
            - Algorithms**
- Policy Set Bindings
  - WSS10Default

**Algorithm Suite**

Basic128Rsa15

**Canonicalization Algorithm**

Exclusive canonicalization

☐ Use security tokens reference transformation

**Add** **Delete**

**Finish** **Cancel**

4. Require or enable Algorithms

**Policy Sets for "BK61"**

**Set up Policy Sets and Policy Set Bindings for this broker**

Use these panels to define the parts of your message to be encrypted and signed.

**Policy Sets**

- WSS10Default
  - WSSecurity
    - WS-Security
      - Authentication Tokens
        - Message Level Protection
          - Tokens
            - Algorithms
  - Policy Set Bindings
    - WSS10Default

**Names with Security Type, SOAP Message and Message Body**

| Name                 | Security Type | SOAP Message | Message Body |
|----------------------|---------------|--------------|--------------|
| encryptpart_request  | Encryption    | Request      | Yes          |
| encryptpart_response | Encryption    | Response     | Yes          |
| signpart_request     | Signature     | Request      | Yes          |
| signpart_response    | Signature     | Response     | Yes          |
|                      |               |              |              |
|                      |               |              |              |
|                      |               |              |              |
|                      |               |              |              |
|                      |               |              |              |
|                      |               |              |              |

**Add** **Delete**

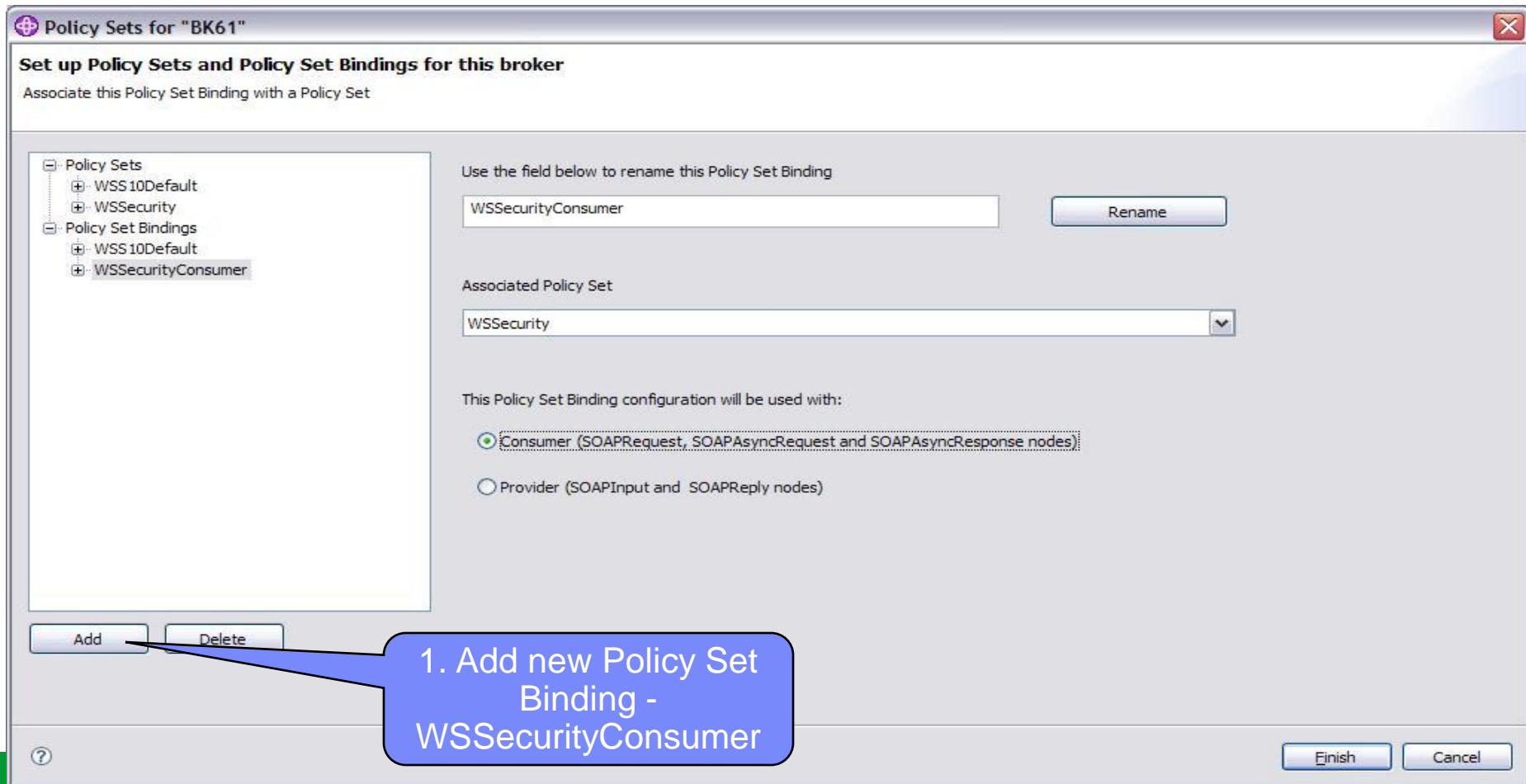
**Finish** **Cancel**

5. Define parts of message to be encrypted and signed



# WS-Security Configuration – Policy set Bindings

Create Policy Set Bindings associated with the above created Policies





# WS-Security Configuration – Consumer Bindings

Message Part Policy /  
Associate message part  
with signing and encryption  
token

Key Information /  
Define key and trust  
information

Policy Sets for "BK61"

Set up Policy Sets and Policy Set Bindings for this broker

Use this panel to associate any message part protection tokens with asymmetric encryption or signing tokens defined in the associated policy set.

Policy Sets

- WSS10Default
- WSSecurity
- Policy Set Bindings
- WSS10Default
- WSSecurityConsumer
- WSSecurity
- Authentication Tokens
- Message Part Policy**
- Key Information
- Message Expiration
- WSSecurityProvider

Add Delete

Message Part encryption policies

| Encryption Protection         | Timestamp | Nonce | Encryption | Token          | Token Type | Order |
|-------------------------------|-----------|-------|------------|----------------|------------|-------|
| request:encryptpart_request   | Yes       | Yes   | Data       | RecipientToken | KEYID      | 2     |
| response:encryptpart_response | Yes       | Yes   | Data       | InitiatorToken | N/A        | N/A   |
|                               |           |       |            |                |            |       |
|                               |           |       |            |                |            |       |
|                               |           |       |            |                |            |       |
|                               |           |       |            |                |            |       |

Message Part signature policies

| Signature Protection       | Token          | Token Type | Order |
|----------------------------|----------------|------------|-------|
| response:signpart_response | RecipientToken | N/A        | N/A   |
| request:signpart_request   | InitiatorToken | STRREF     | 1     |
|                            |                |            |       |
|                            |                |            |       |
|                            |                |            |       |

Finish Cancel

Key Information

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | N/A        |
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | TrustStore |
|                |   |            |            |
|                |   |            |            |
|                |   |            |            |
|                |   |            |            |

Add Delete

Finish Cancel

# WS-Security Configuration – Consumer Bindings

- Match Policy Set tokens to message parts
- Define the order in which signatures are applied
- Define necessary key information for signing messages

Message Part signature policies

| Signature Protection       | Token          | Token Type | Order |
|----------------------------|----------------|------------|-------|
| response:signpart_response | RecipientToken | N/A        | N/A   |
| request:signpart_request   | InitiatorToken | STRREF     | 1     |

Tokens defined in  
message set

Key Information

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | N/A        |
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | TrustStore |

Key information defined in  
keystore / truststore  
Refer to page 27  
(client.keystore)

# WS-Security Configuration – Consumer Bindings

- Match Policy Set tokens to message parts
- Define ordering and key locations
- May be combined with Integrity settings if defined in the Policy Set

Message Part encryption policies

| Encryption Protection         | Timestamp | Nonce | Encryption | Token          | Token Type | Order |
|-------------------------------|-----------|-------|------------|----------------|------------|-------|
| request:encryptpart_request   | Yes       | Yes   | Data       | RecipientToken | KEYID      | 2     |
| response:encryptpart_response | Yes       | Yes   | Data       | InitiatorToken | N/A        | N/A   |

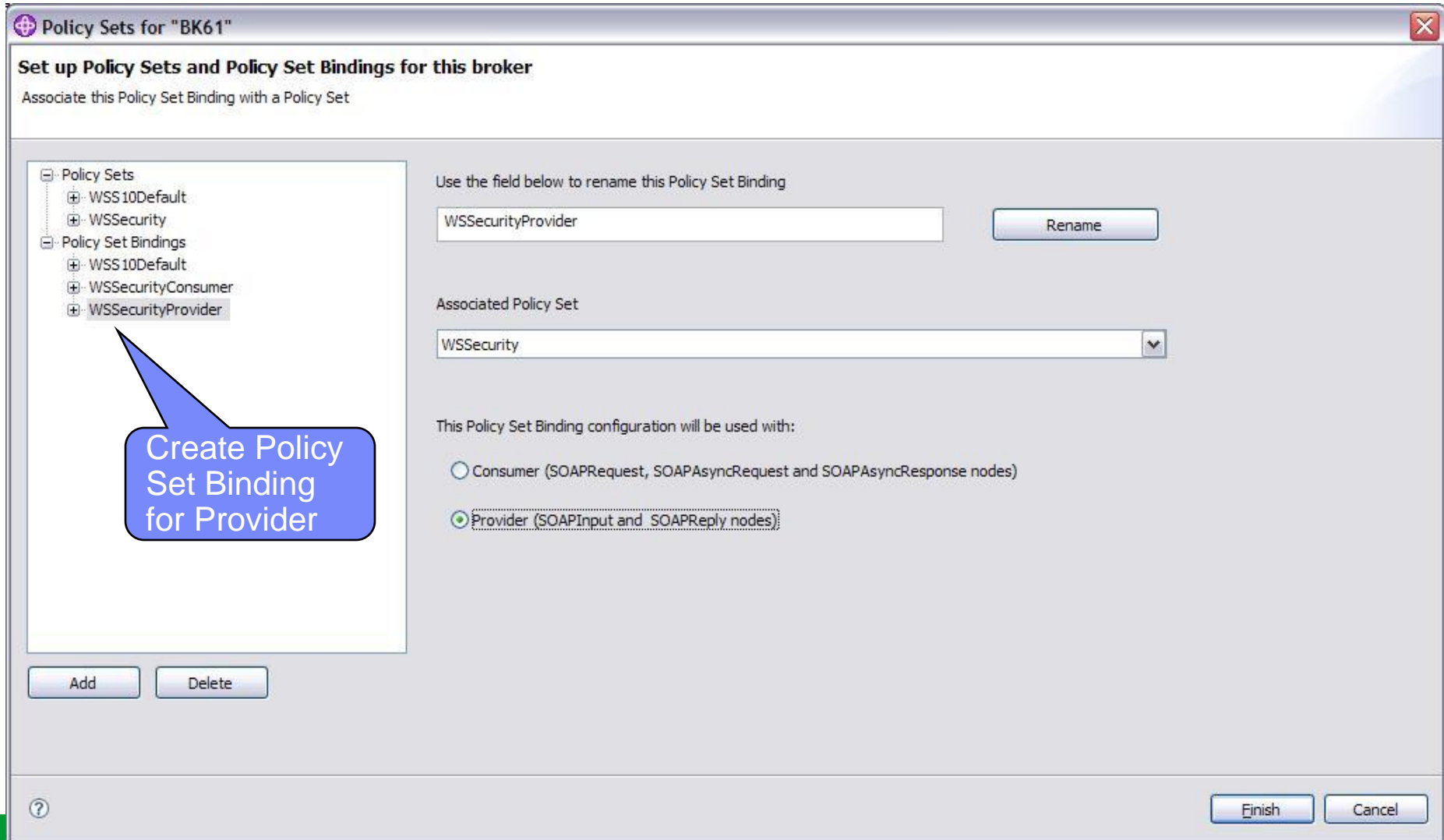
Tokens defined in message set

Key Information

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | N/A        |
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | TrustStore |

Key information defined in keystore / truststore  
Refer to page 27  
(client.keystore)

# WS-Security Configuration – Policy set Bindings



# WS-Security Configuration – Provider Bindings

Message Part Policy /  
Associate message part  
with signing and encryption  
token

Key Information /  
Define key and  
trust information

**Policy Sets for "BK61"**

**Set up Policy Sets and Policy Set Bindings for this broker**

Use this panel to associate any message part protection tokens with asymmetric encryption or signing tokens defined in the associated policy set.

**Policy Sets**

- WSS10Default
- WSSecurity
- Policy Set Bindings
  - WSS10Default
  - WSSecurityConsumer
  - WSSecurityProvider
    - WS-Security
      - Authentication Tokens
      - Message Part Policy**
      - Key Information
      - Message Expiration

**Message Part encryption policies**

| Encryption Protection         | Timestamp | Nonce | Encryption | Token          | Token Type | Order |
|-------------------------------|-----------|-------|------------|----------------|------------|-------|
| request:encryptpart_request   | Yes       | Yes   | Data       | RecipientToken | N/A        | N/A   |
| response:encryptpart_response | Yes       | Yes   | Data       | InitiatorToken | KEYID      | 2     |
|                               |           |       |            |                |            |       |
|                               |           |       |            |                |            |       |
|                               |           |       |            |                |            |       |

**Message Part signature policies**

| Signature Protection       | Token          | Token Type | Order |
|----------------------------|----------------|------------|-------|
| response:signpart_response | RecipientToken | STRREF     | 1     |
| request:signpart_request   | InitiatorToken | N/A        | N/A   |
|                            |                |            |       |
|                            |                |            |       |
|                            |                |            |       |

**Key Information**

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | N/A        |
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | TrustStore |
|                |   |            |            |
|                |   |            |            |
|                |   |            |            |

# WS-Security Configuration – Provider Bindings

- Match Policy Set tokens to message parts
- Define the order in which signatures are applied
- Define necessary key information for signing messages

Message Part signature policies

| Signature Protection       | Token          | Token Type | Order |
|----------------------------|----------------|------------|-------|
| response:signpart_response | RecipientToken | STRREF     | 1     |
| request:signpart_request   | InitiatorToken | N/A        | N/A   |

Tokens defined in  
message set

Key Information

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | N/A        |
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | TrustStore |

Key information defined in  
keystore / truststore  
Refer to page 28  
(server.keystore)



# WS-Security Configuration – Provider Bindings

- Match Policy Set tokens to message parts
- Define ordering and key locations
- May be combined with Integrity settings if defined in Policy Set

Message Part encryption policies

| Encryption Protection         | Timestamp | Nonce | Encryption | Token          | Token Type | Order |
|-------------------------------|-----------|-------|------------|----------------|------------|-------|
| request:encryptpart_request   | Yes       | Yes   | Data       | RecipientToken | N/A        | N/A   |
| response:encryptpart_response | Yes       | Yes   | Data       | InitiatorToken | KEYID      | 2     |

Tokens defined in message set

Key Information

| Token          | Key Name                                    | Key Alias  | Trust      |
|----------------|---|------------|------------|
| RecipientToken | CN=server, O=Web Services Guided Tour, C=GB | servercert | N/A        |
| InitiatorToken | CN=client, O=Web Services Guided Tour, C=GB | clientcert | TrustStore |

Key information defined in keystore / truststore  
Refer to page 28 (server.keystore)

# WS-Security Configuration - Runtime

## ■ Runtime Configuration

Use the following commands to set up the provider keystore and truststore:

- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystoreFile -v [Location of server keystore]`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystoreType -v JKS`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystorePass -v Provider::password`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststoreFile -v [Location of server keystore]`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststoreType -v JKS`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststorePass -v Provider::password`
- `mqsisetdbparms <broker> -n Provider::password -u temp -p server`

Use the following commands to set up the consumer keystore and truststore:

- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystoreFile -v [Location of server keystore]`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystoreType -v JKS`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n keystorePass -v Consumer::password`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststoreFile -v [Location of server keystore]`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststoreType -v JKS`
- `mqsichangeproperties <broker> -e <EG> -o ComIbmJVMMManager -n truststorePass -v Consumer::password`
- `mqsisetdbparms <broker> -n Consumer::password -u temp -p client`







# WS-Security Configuration - Deploy

- Selection of Policy Set and Binding made in the .bar file

 Configure properties of selected built resource:

|                              |  |
|------------------------------|--|
| Additional Instances         | <input type="text" value="0"/>   |
| Commit Count                 | <input type="text" value="1"/>   |
| Commit Interval              | <input type="text" value="0"/>   |
| Consumer Policy Set          | <input type="text" value="WSecurity"/> <a href="#">Edit...</a>   |
| Consumer Policy Set Bindings | <input type="text" value="WSecurityConsumer"/> <a href="#">Edit...</a>                                   |
| Coordinated Transaction      | <input type="checkbox"/>   |
| Monitoring Profile Name      | <input type="text"/>   |
| Provider Policy Set          | <input type="text"/> <a href="#">Edit...</a>   |
| Provider Policy Set Bindings | <input type="text"/> <a href="#">Edit...</a>   |
| Security Profile Name        | <input type="text"/>  |

 Configure properties of selected built resource:

|                              |  |
|------------------------------|--|
| Additional Instances         | <input type="text" value="0"/>   |
| Commit Count                 | <input type="text" value="1"/>   |
| Commit Interval              | <input type="text" value="0"/>   |
| Consumer Policy Set          | <input type="text"/> <a href="#">Edit...</a>   |
| Consumer Policy Set Bindings | <input type="text"/> <a href="#">Edit...</a>   |
| Coordinated Transaction      | <input type="checkbox"/>   |
| Monitoring Profile Name      | <input type="text"/>   |
| Provider Policy Set          | <input type="text" value="WSecurity"/> <a href="#">Edit...</a>   |
| Provider Policy Set Bindings | <input type="text" value="WSecurityProvider"/> <a href="#">Edit...</a>                                     |
| Security Profile Name        | <input type="text"/>  |

# WS-Security Configuration

- Validate the configuration
  - ▶ Which WS-Security capabilities are used
    - Integrity Inbound/Outbound
    - Confidentiality Inbound/Outbound
  - ▶ Which certificates are used and where must they be kept
    - The same key may be used for multiple scenarios and specified in different places in the Policy Set Binding
    - Encrypt a message on output of the request node
    - Encrypt a message on response to a Soap Input node
- Deploy the bar file



# Additional Resources

- WebSphere Message Broker V6.1 Information Center  
<http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp>
- Implementation of new security features in WMB V6.1  
<http://www-01.ibm.com/support/docview.wss?uid=swg27015336&aid=1>
- Implementating SSL with HTTP nodes in WMB V6.x  
<http://www-01.ibm.com/support/docview.wss?uid=swg27012172&aid=1>
- Using new features in WebSphere Message Broker V6.1  
<http://www.redbooks.ibm.com/redpapers/abstracts/redp4458.html?Open>
- Session Q31 at WSTC 2008 by Stephen Cox & Peter Crocker
- Implementing WS-Security  
<http://www.ibm.com/developerworks/webservices/library/ws-security.html>
- Signing flows for WS-Security  
<http://www.ibm.com/developerworks/webservices/library/ws-security/index.html>



# Additional WebSphere Product Resources

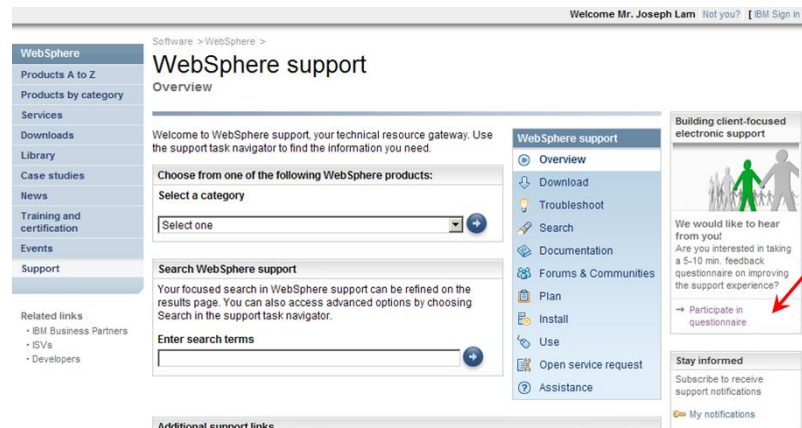
- Discover the latest trends in WebSphere Technology and implementation, participate in technically-focused briefings, webcasts and podcasts at:  
<http://www.ibm.com/developerworks/websphere/community/>
- Learn about other upcoming webcasts, conferences and events:  
[http://www.ibm.com/software/websphere/events\\_1.html](http://www.ibm.com/software/websphere/events_1.html)
- Join the Global WebSphere User Group Community: <http://www.websphere.org>
- Access key product show-me demos and tutorials by visiting IBM® Education Assistant:  
<http://www.ibm.com/software/info/education/assistant>
- View a Flash replay with step-by-step instructions for using the Electronic Service Request (ESR) tool for submitting problems electronically:  
<http://www.ibm.com/software/websphere/support/d2w.html>
- Sign up to receive weekly technical My Notifications emails:  
<http://www.ibm.com/software/support/einfo.html>



# IBM Support Wants to Hear From You!

## Tell us about your support needs and wants

1. Visit any product support pages on IBM.com.
2. Click on “Participate in Questionnaire” on top right of page.
3. Takes 5-10 minutes to complete.



Or go to [https://www.ibm.com/survey/oid/wsb.dll/s/ag21f?wsb34=swg\\_user](https://www.ibm.com/survey/oid/wsb.dll/s/ag21f?wsb34=swg_user)

# Questions and Answers

