WebService Security in WebSphere Integration Developer

Wang Hui

IBM Level 2 Tech Support
huiwwang@cn.ibm.com

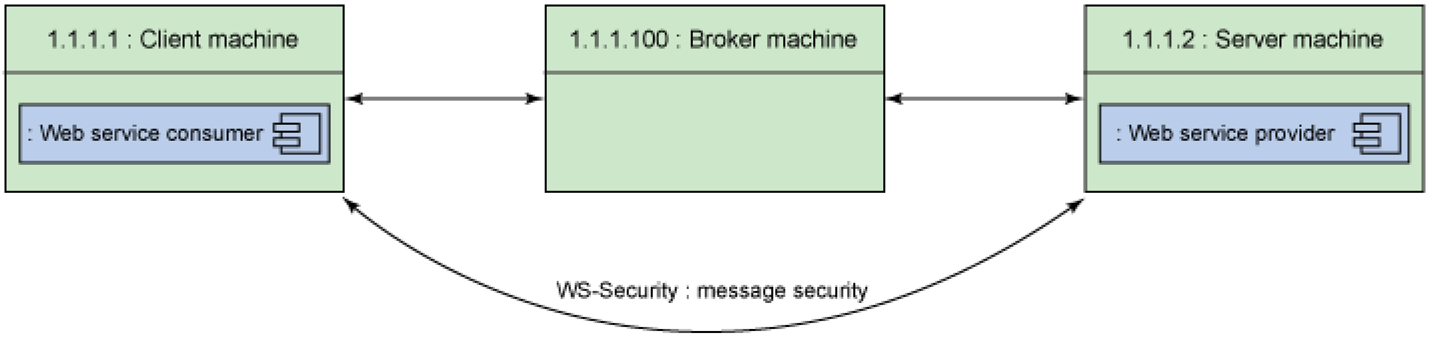
ON DEMAND BUSINESS™

Agenda

1	Message Level Security Introduction
2	Security Implementation by using WID Module Deployment Editor
3	Security Implementation by using Policy Set
4	Comparison of the 2 Implementations

Transport Level VS Message Level

Transport Level	Message Level
HTTPS	WSS
Point-to-point channel security	End-to-end message security
synchronous	asynchronous
Public Private Key Encryption	Public Private Key Encryption



Web Services Security

Web Services Security (WS-Security) is an OASIS standard to describe how to implement message-level security with Web services.

Identification: The party accessing the resource is able to identify itself to the system.

Authentication: Authentication is the process of validating the user, whether a client is valid in a particular context. A client can be either an end user, a machine, or an application.

Authorization: Authorization is the process of checking whether the authenticated user has access to the requested resource.

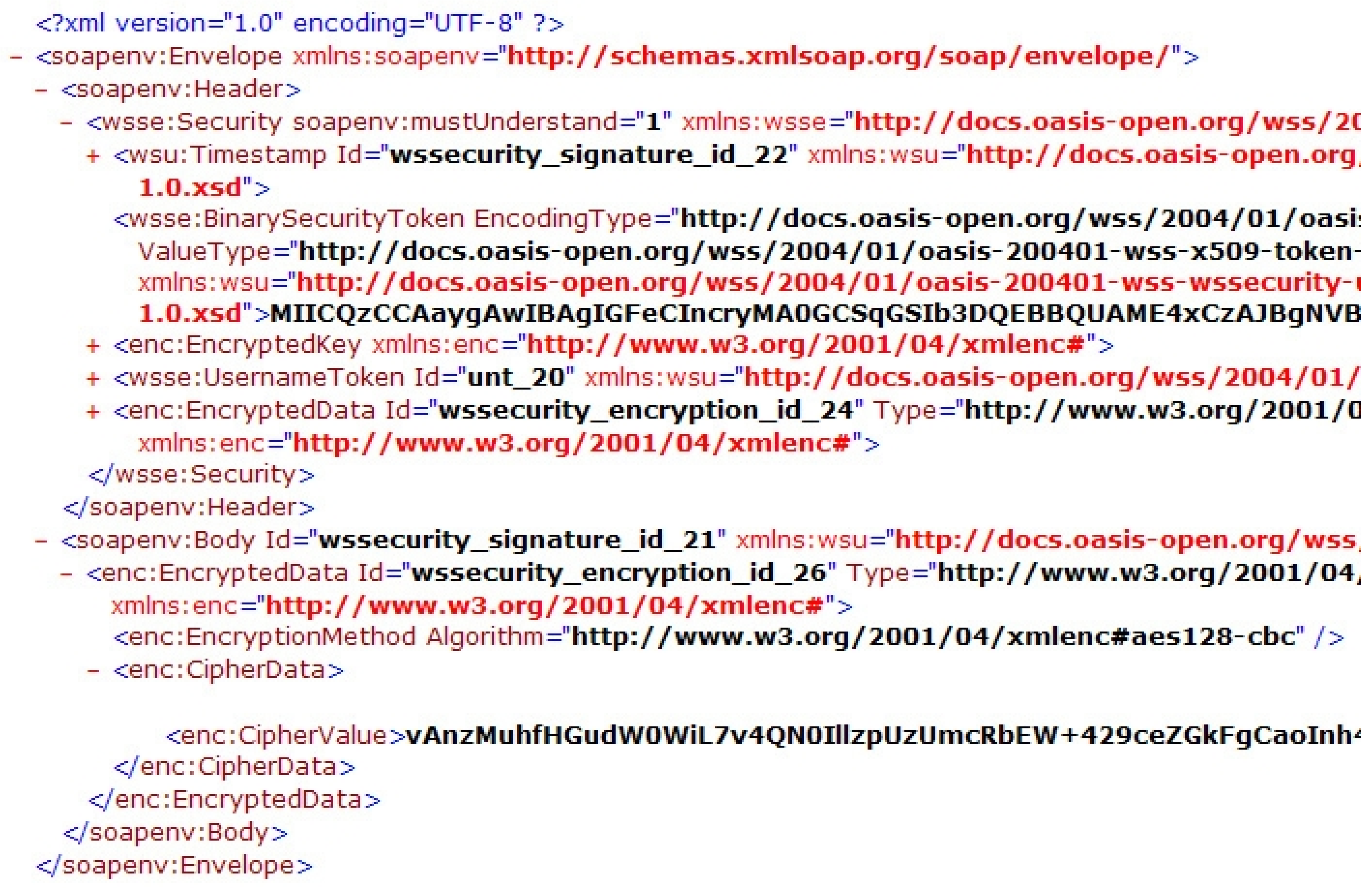
Integrity: Integrity insures that information will not be changed, altered, or lost in an unauthorized or accidental manner.

Confidentiality: No unauthorized party or process can access or disclose the information.

Auditing: All transactions are recorded so that problems can be analyzed after the fact.

Non-repudiation: Both parties are able to provide legal proof to a third party that the sender did send the information, and the receiver received the identical information. Neither involved side is "unable to deny".

Example of message-level security of a SOAP message



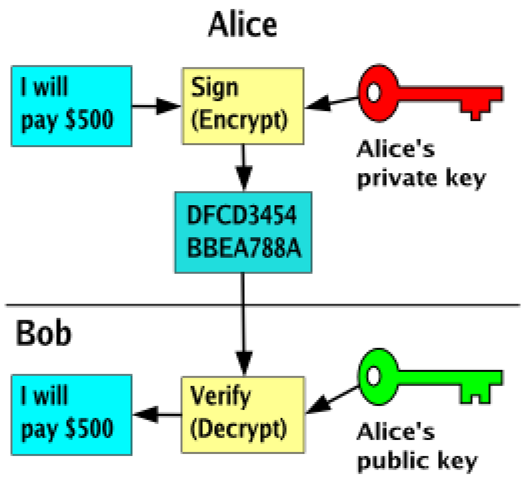
```
  <?xml version="1.0" encoding="UTF-8" ?>
- <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  - <soapenv:Header>
    - <wsse:Security soapenv:mustUnderstand="1" xmlns:wsse="http://docs.oasis-open.org/wss/20
      + <wsu:Timestamp Id="wssecurity_signature_id_22" xmlns:wsu="http://docs.oasis-open.org
          1.0.xsd">
        <wsse:BinarySecurityToken EncodingType="http://docs.oasis-open.org/wss/2004/01/oasi
          ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
          xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
          1.0.xsd">MIICQzCCAaygAwIBAgIGFeCIncryMA0GCSqGSIb3DQEBBQUAME4xCzAJBgNVB
      + <enc:EncryptedKey xmlns:enc="http://www.w3.org/2001/04/xmlenc#">
      + <wsse:UsernameToken Id="unt_20" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
      + <enc:EncryptedData Id="wssecurity_encryption_id_24" Type="http://www.w3.org/2001/0
          xmlns:enc="http://www.w3.org/2001/04/xmlenc#">
      </wsse:Security>
    </soapenv:Header>
  - <soapenv:Body Id="wssecurity_signature_id_21" xmlns:wsu="http://docs.oasis-open.org/wss
    - <enc:EncryptedData Id="wssecurity_encryption_id_26" Type="http://www.w3.org/2001/04
        xmlns:enc="http://www.w3.org/2001/04/xmlenc#">
        <enc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc" />
      - <enc:CipherData>

            <enc:CipherValue>vAnzMuhfHGudW0WiL7v4QN0IllzpUzUmcRbEW+429ceZGkFgCaoInh
        </enc:CipherData>
      </enc:EncryptedData>
    </soapenv:Body>
  </soapenv:Envelope>
```

Public Key VS Private Key

Asymmetric key algorithms: the key used to encrypt a message is not the same as the key used to decrypt it.

Public key – The key that is used by others to encrypt data for you.

Private key – The key that matches your public key and is used to decrypt data that others have encrypted with your public key. This key should not be shared with others.



Public Key VS Private Key cont.

1.Create a keystore to store Alice's private key:

```
keytool -genkey -v -alias alice -keypass senderkeys -keystore senderkeys.jks
-storepass senderkeys -storetype jks -dname "cn=alice,o=IBM,c=US" -keyalg "RSA"
```

2.Export Alice's public keys in a file called alice.cert:

```
keytool -export -v -alias alice -file alice.cert -rfc -keystore senderkeys.jks
-storepass senderkeys -storetype jks
```

3.Create reckeys.jks key store to store Bob's keys:

```
keytool -genkey -v -alias bob -keypass reckeys -keystore reckeys.jks -storepass
reckeys -storetype jks -dname "cn=bob,o=IBM,c=US" -keyalg "RSA"
```

4.Export Bob's public keys in a file called bob.cert:

```
keytool -export -v -alias bob -file bob.cert -rfc -keystore reckeys.jks
-storepass reckeys -storetype jks
```

5.Import Bob's public key into Alice's key store, viz. senderkeys.jks:

```
keytool -import -v -noprompt -alias bob -file bob.cert -keystore senderkeys.jks
-storepass senderkeys -storetype jks
```

6.Import Alice's public key into Bob's key store viz. reckeys.jks:

```
keytool -import -v -noprompt -alias alice -file alice.cert -keystore
reckeys.jks -storepass reckeys -storetype jks
```

Public Key VS Private Key cont.

Certificate authority – For others to trust that your public key really belongs to you, you normally request a CA (e.g. Verisign, GeoTrust, GoDaddy) to sign your key. Since others do the same thing, you can trust others by the CA vouching for you and them.

Digital certificate – To share your public key with others and for them to trust that you are who you say you are, you create a digital certificate which contains your public key along with your identity information (e.g. your name) and send this digital document to a CA to sign for you.

Key store – A place to store your keys. Also called a key ring.

Signer certificate – After your digital certificate has been signed by a CA, it becomes a signer certificate. Digital certificate, public key certificate, and signer certificate are often used synonymously.

Security in WID: JAX-RPC VS JAX-WS

•SOAP 1.2
JAX-RPC and JAX-WS support SOAP 1.1. JAX-WS also supports SOAP 1.2.

•XML/HTTP
The WSDL 1.1 specification defined an HTTP binding, which is a means by which you can send XML messages over HTTP without SOAP. JAX-RPC ignored the HTTP binding. JAX-WS adds support for it.

•WS-I's Basic Profiles
JAX-RPC supports WS-I's Basic Profile (BP) version 1.0. JAX-WS supports BP 1.1. (WS-I is the web services interoperability organization.)

•New Java features
JAX-RPC maps to Java 1.4. JAX-WS maps to Java 5.0. JAX-WS relies on many of the features new in Java 5.0.
Java EE 5, the successor to J2EE 1.4, adds support for JAX-WS, but it also retains support for JAX-RPC, which could be confusing to today's web services novices.

•The data mapping model
JAX-RPC has its own data mapping model, which covers about 90 percent of all schema types. Those that it does not cover are mapped to javax.xml.soap.SOAPElement.
JAX-WS's data mapping model is JAXB. JAXB promises mappings for all XML schemas.

Security in WID: JAX-RPC VS JAX-WS cont.

•The interface mapping model

JAX-WS's basic interface mapping model is not extensively different from JAX-RPC's; however:

JAX-WS's model makes use of new Java 5.0 features.

JAX-WS's model introduces asynchronous functionality.

•The dynamic programming model

JAX-WS's dynamic client model is quite different from JAX-RPC's. Many of the changes acknowledge industry needs:

It introduces message-oriented functionality.

It introduces dynamic asynchronous functionality.

JAX-WS also adds a dynamic server model, which JAX-RPC does not have.

•MTOM (Message Transmission Optimization Mechanism)

JAX-WS, via JAXB, adds support for MTOM, the new attachment specification. Microsoft never bought into the SOAP with Attachments specification; but it appears that everyone supports MTOM, so attachment interoperability should become a reality.

•The handler model

The handler model has changed quite a bit from JAX-RPC to JAX-WS.

JAX-RPC handlers rely on SAAJ 1.2. JAX-WS handlers rely on the new SAAJ 1.3 specification.

Security in WID: Extension VS Binding

The Web Service Client Security Extension:

- Add, edit, or remove a service reference
- Add, edit or remove a port qualified name (Qname) binding
- Add, edit or remove the default mappings for a selected service
- Modify the Request Generator configuration for the Selected Port Qname Binding, including integrity, confidentiality and security settings.
- Modify timestamp and other information

The Web Service Client Binding Configuration:

- add or remove a web service to a client binding
- view service reference details, including the deployed WSDL file
- add or remove a port qualified name binding
- specify setting for the selected port qualified name binding
- add or remove default mappings
- add or remove parameters for the selected service reference
- modify the security configuration for generating request messages
- modify the security configuration for consuming response messages
- add or remove parameters of the selected port qualified name binding

Security in WID: Policy Set

Policy sets provide a declarative way to define qualities of service (QoS) for Web services. This simplifies the management of multiple Web services as policy sets can be reused across them. Let's discuss the differences in policy set terminology:

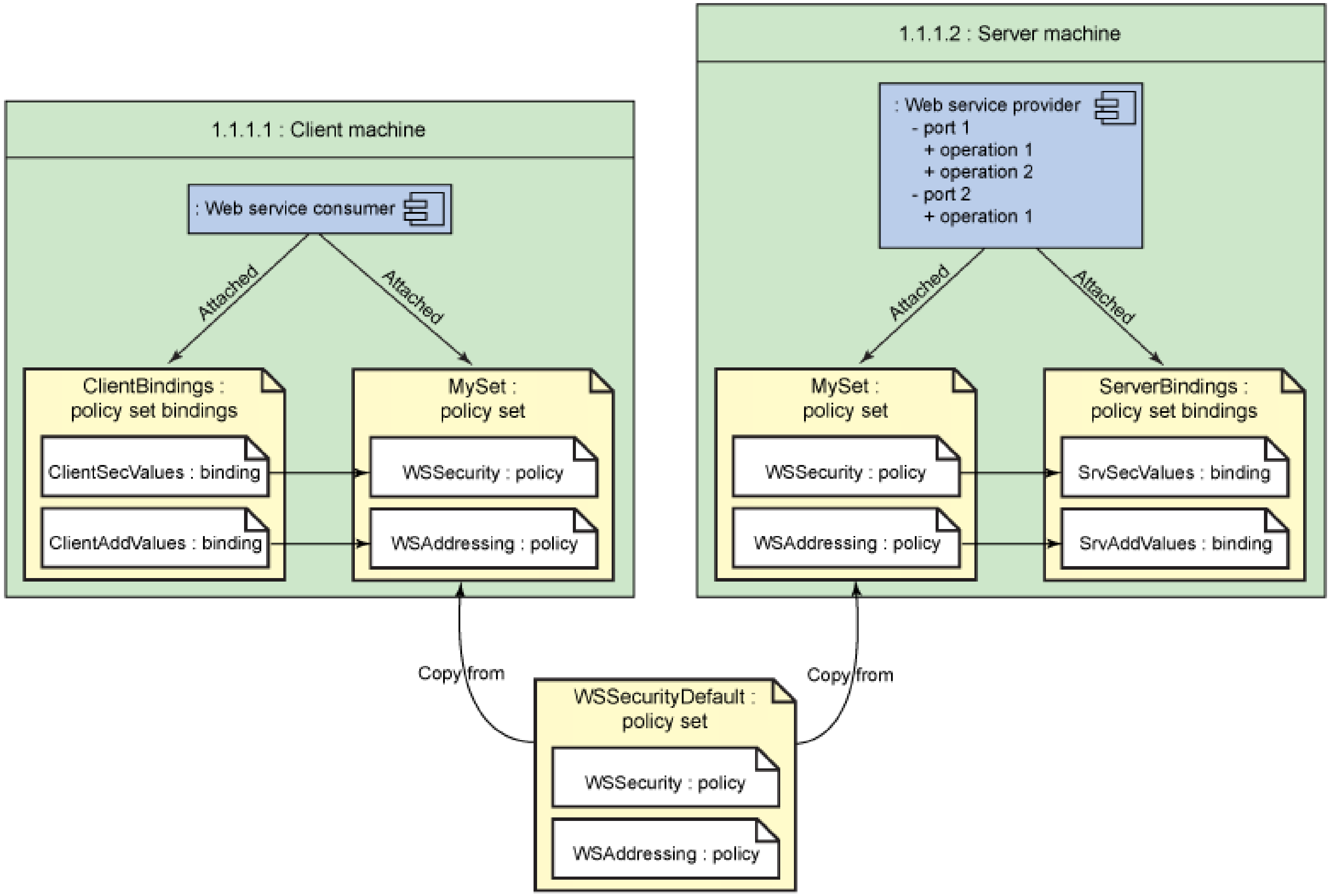
•Policy – A policy describes a configuration that defines qualities of service (QoS) for Web services. Examples include WS-Security and WS-Addressing.

•Policy sets – A policy set is a collection of policies.

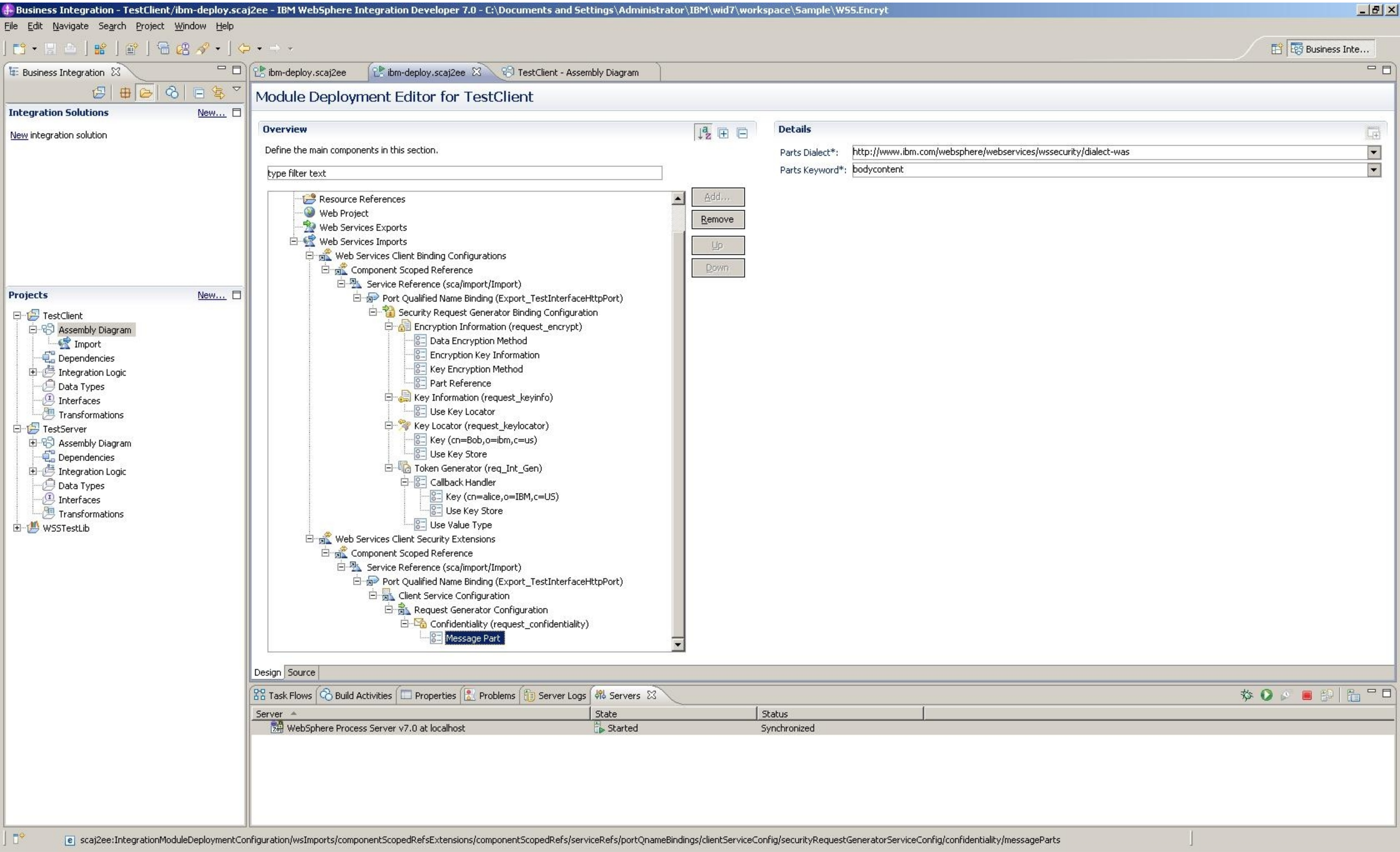
•Policy set attachment – In order to apply policy sets to Web services, they need to be attached.

•Bindings – Policy sets are meant to be reused across Web services and thus do not contain environment specific settings such as key stores or passwords. Instead, a binding contains these environment specific values.

Security in WID: Policy Set cont.



Security Implementation by using WID Module Deployment Editor


Business Integration - TestClient/ibm-deploy.scaj2ee - IBM WebSphere Integration Developer 7.0 - C:\Documents and Settings\Administrator\IBM\wid7\workspace\Sample\WSS.Encryt

Module Deployment Editor for TestClient

Overview

Define the main components in this section.

```
Resource References
Web Project
Web Services Exports
Web Services Imports
  Web Services Client Binding Configurations
    Component Scoped Reference
      Service Reference (sca/import/Import)
        Port Qualified Name Binding (Export_TestInterfaceHttpPort)
          Security Request Generator Binding Configuration
            Encryption Information (request_encrypt)
              Data Encryption Method
              Encryption Key Information
              Key Encryption Method
              Part Reference
            Key Information (request_keyinfo)
              Use Key Locator
            Key Locator (request_keylocator)
              Key (cn=Bob,o=ibm,c=us)
              Use Key Store
            Token Generator (req_Int_Gen)
              Callback Handler
                Key (cn=alice,o=IBM,c=US)
                Use Key Store
              Use Value Type
  Web Services Client Security Extensions
    Component Scoped Reference
      Service Reference (sca/import/Import)
        Port Qualified Name Binding (Export_TestInterfaceHttpPort)
          Client Service Configuration
            Request Generator Configuration
              Confidentiality (request_confidentiality)
                Message Part
```

Details

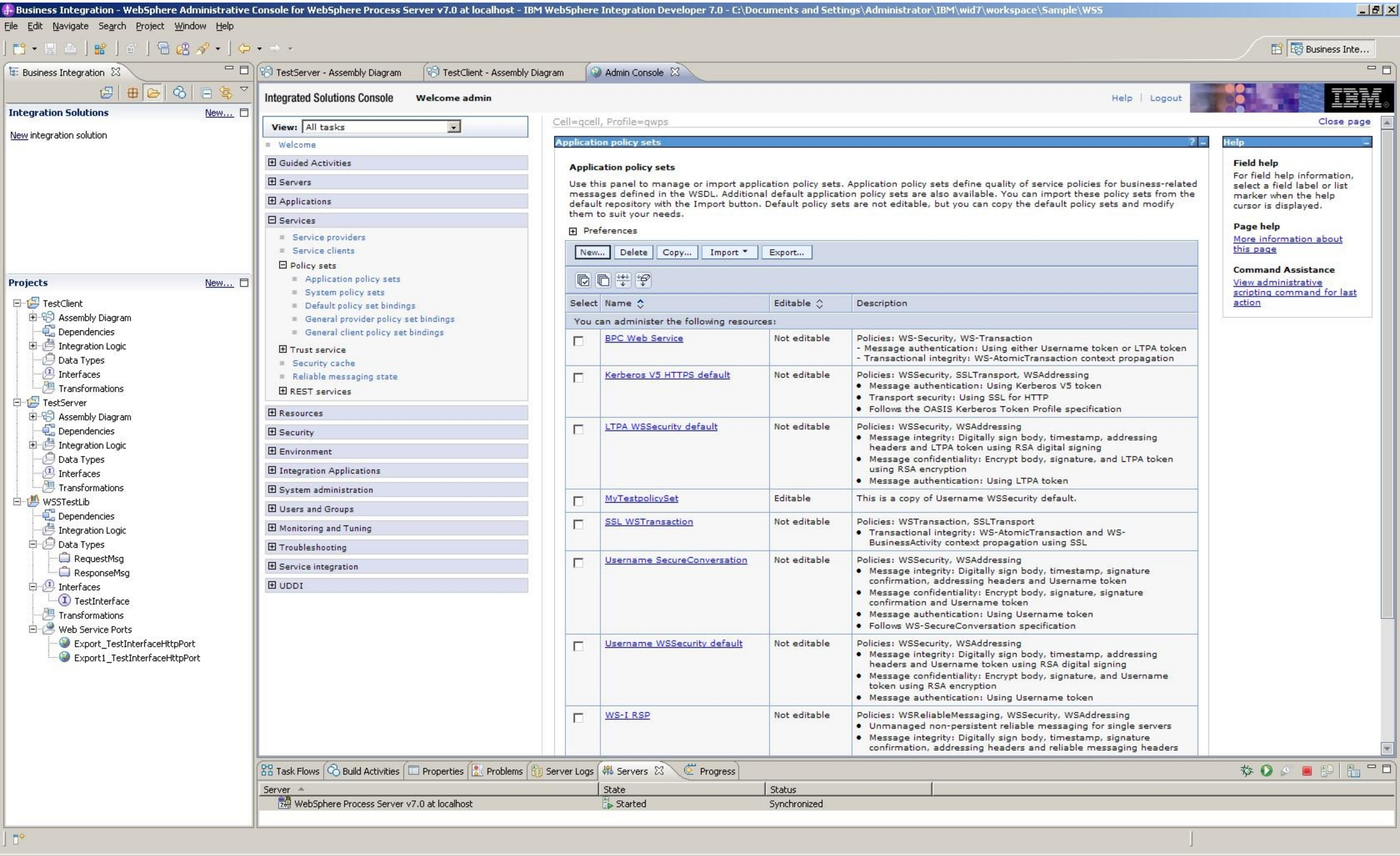
Parts Dialect*: http://www.ibm.com/websphere/webservices/wssecurity/dialect-was

Parts Keyword*: bodycontent

Server	State	Status
WebSphere Process Server v7.0 at localhost	Started	Synchronized

scaj2ee:IntegrationModuleDeploymentConfiguration/wsImports/componentScopedRefsExtensions/componentScopedRefs/serviceRefs/portQnameBindings/clientServiceConfig/securityRequestGeneratorServiceConfig/confidentiality/messageParts

Security Implementation by using Policy Set



Business Integration - WebSphere Administrative Console for WebSphere Process Server v7.0 at localhost - IBM WebSphere Integration Developer 7.0 - C:\Documents and Settings\Administrator\IBM\wid7\workspace\Sample\WSS

File Edit Navigate Search Project Window Help

Business Integration

Integration Solutions New...

New integration solution

Projects New...

- TestClient
  - Assembly Diagram
  - Dependencies
  - Integration Logic
  - Data Types
  - Interfaces
  - Transformations
- TestServer
  - Assembly Diagram
  - Dependencies
  - Integration Logic
  - Data Types
  - Interfaces
  - Transformations
- WSSTestLib
  - Dependencies
  - Integration Logic
  - Data Types
    - RequestMsg
    - ResponseMsg
  - Interfaces
    - TestInterface
  - Transformations
  - Web Service Ports
    - Export_TestInterfaceHttpPort
    - Export1_TestInterfaceHttpPort

TestServer - Assembly Diagram | TestClient - Assembly Diagram | Admin Console

Integrated Solutions Console   Welcome admin   Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
- Services
  - Service providers
  - Service clients
  - Policy sets
    - Application policy sets
    - System policy sets
    - Default policy set bindings
    - General provider policy set bindings
    - General client policy set bindings
  - Trust service
  - Security cache
  - Reliable messaging state
  - REST services
- Resources
- Security
- Environment
- Integration Applications
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=qcell, Profile=qwps   Close page

Application policy sets

Application policy sets

Use this panel to manage or import application policy sets. Application policy sets define quality of service policies for business-related messages defined in the WSDL. Additional default application policy sets are also available. You can import these policy sets from the default repository with the Import button. Default policy sets are not editable, but you can copy the default policy sets and modify them to suit your needs.

Preferences

New... | Delete | Copy... | Import | Export...

Select	Name	Editable	Description
	You can administer the following resources:		
☐	BPC Web Service	Not editable	Policies: WS-Security, WS-Transaction - Message authentication: Using either Username token or LTPA token - Transactional integrity: WS-AtomicTransaction context propagation
☐	Kerberos V5 HTTPS default	Not editable	Policies: WSSecurity, SSLTransport, WSAddressing ● Message authentication: Using Kerberos V5 token ● Transport security: Using SSL for HTTP ● Follows the OASIS Kerberos Token Profile specification
☐	LTPA WSSecurity default	Not editable	Policies: WSSecurity, WSAddressing ● Message integrity: Digitally sign body, timestamp, addressing headers and LTPA token using RSA digital signing ● Message confidentiality: Encrypt body, signature, and LTPA token using RSA encryption ● Message authentication: Using LTPA token
☐	MyTestpolicySet	Editable	This is a copy of Username WSSecurity default.
☐	SSL WSTransaction	Not editable	Policies: WSTransaction, SSLTransport ● Transactional integrity: WS-AtomicTransaction and WS-BusinessActivity context propagation using SSL
☐	Username SecureConversation	Not editable	Policies: WSSecurity, WSAddressing ● Message integrity: Digitally sign body, timestamp, signature confirmation, addressing headers and Username token ● Message confidentiality: Encrypt body, signature, signature confirmation and Username token ● Message authentication: Using Username token ● Follows WS-SecureConversation specification
☐	Username WSSecurity default	Not editable	Policies: WSSecurity, WSAddressing ● Message integrity: Digitally sign body, timestamp, addressing headers and Username token using RSA digital signing ● Message confidentiality: Encrypt body, signature, and Username token using RSA encryption ● Message authentication: Using Username token
☐	WS-I RSP	Not editable	Policies: WSReliableMessaging, WSSecurity, WSAddressing ● Unmanaged non-persistent reliable messaging for single servers ● Message integrity: Digitally sign body, timestamp, signature confirmation, addressing headers and reliable messaging headers

Help

Field help
For field help information, select a field label or list marker when the help cursor is displayed.

Page help
More information about this page

Command Assistance
View administrative scripting command for last action

Task Flows | Build Activities | Properties | Problems | Server Logs | Servers | Progress

Server	State	Status
WebSphere Process Server v7.0 at localhost	Started	Synchronized

Security Implementation

Module Deployment Editor VS Policy Set

Module Deployment Editor	Policy Set
More straight	Reuse
Do not need WAS runtime	Do not need to redeploy module

Trugarez
Breton

Gracias
Spanish

Merci
French

شكرا
Arabic

감사합니다
Korean

谢谢
Simplified Chinese

धन्यवाद
Hindi

謝謝
Traditional Chinese

Tack så mycket
Swedish

Obrigado
Brazilian Portuguese

Thank You
English

Tak
Danish

Danke
German

Grazie
Italian

Dankon
Esperanto

תודה רבה
Hebrew

ありがとうございました
Japanese

Dank u
Dutch

ขอบคุณ
Thai

go raibh maith agat
Gaelic

நன்றி
Tamil

Dekujeme Vam
Czech