

---

## Web Services Security - Basics

Michael Pühlhöfer, Senior IT-Architect, IBM Software Group  
Member of IBM Technical Expert Council

## Agenda

---

|    |  |
|----|--|
| 1. | Security Requirements for Peer-to-Peer Applications  |
| 2. | Web Services Security Details                        |
| 3. | WS-Security Extensions Overview                      |
| 4. | WS-Security: Support in WebSphere Application Server |

# Brief recap: What are Web Services ?

A Web Service is a program that can be called from other programs via a network in a way that is independent of a platform, a language and an object model.



**Simple:**

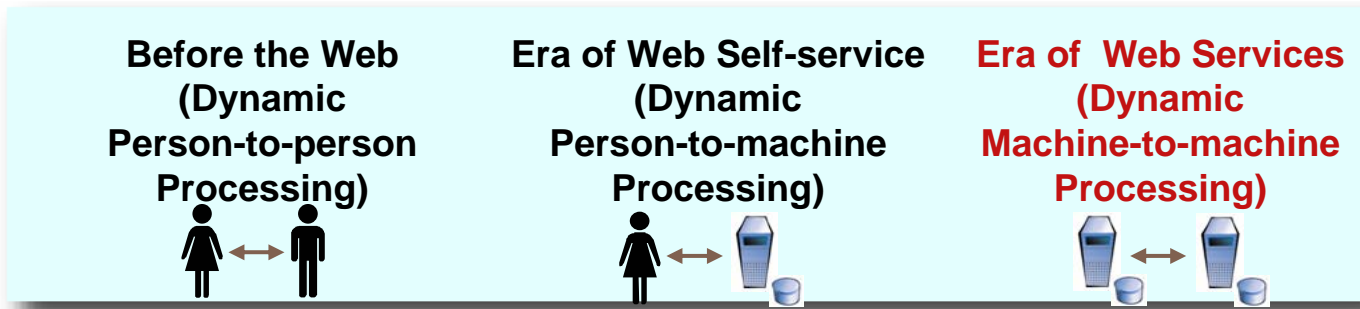
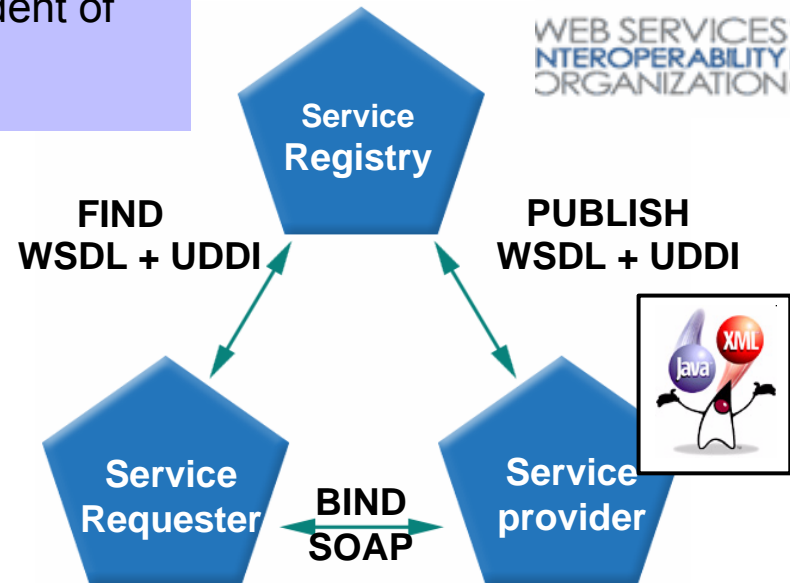
- ➔ XML, HTTP

**Dynamic:**

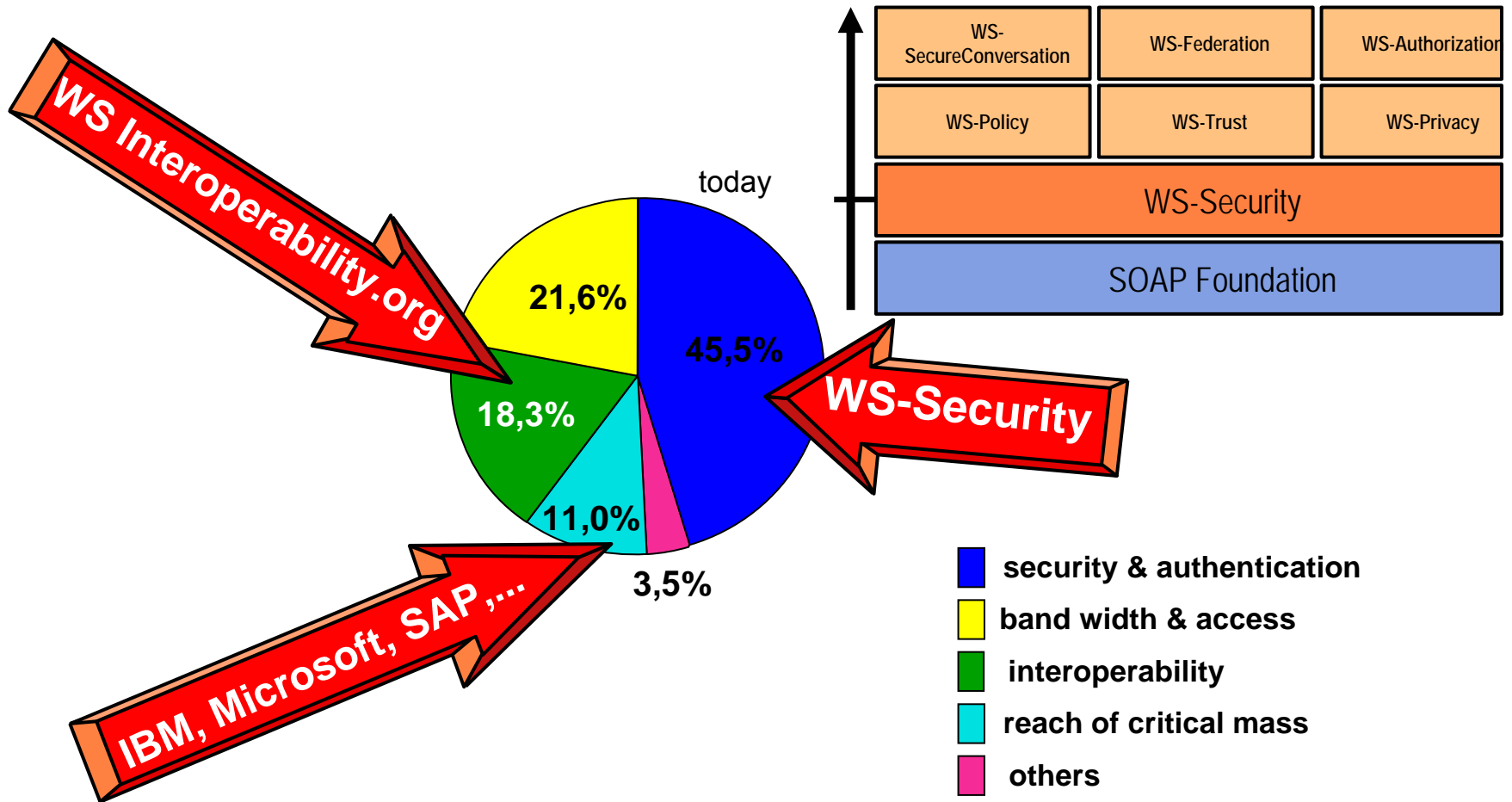
- ➔ Publish, Find, Bind

**Standards based:**

- ➔ XML, XML Schema, SOAP, WSDL, UDDI, ...



# Reasons for NOT implementing Web Services today

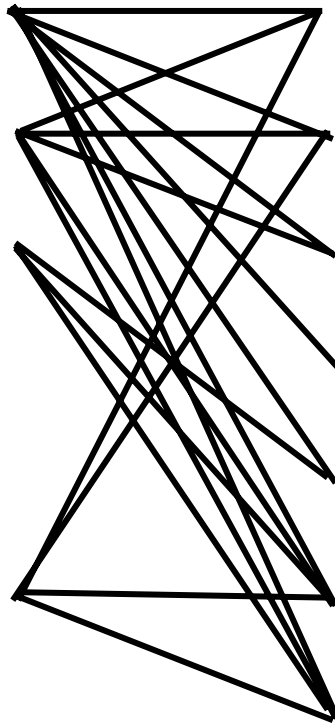


Source: Evans Data Corp. Survey of 400 "Enterprise Development Manager,, (NetworkWorld 1/21/02).

# Possible Attacks and Security Requirements for Peer-to-Peer Applications

## Attacks

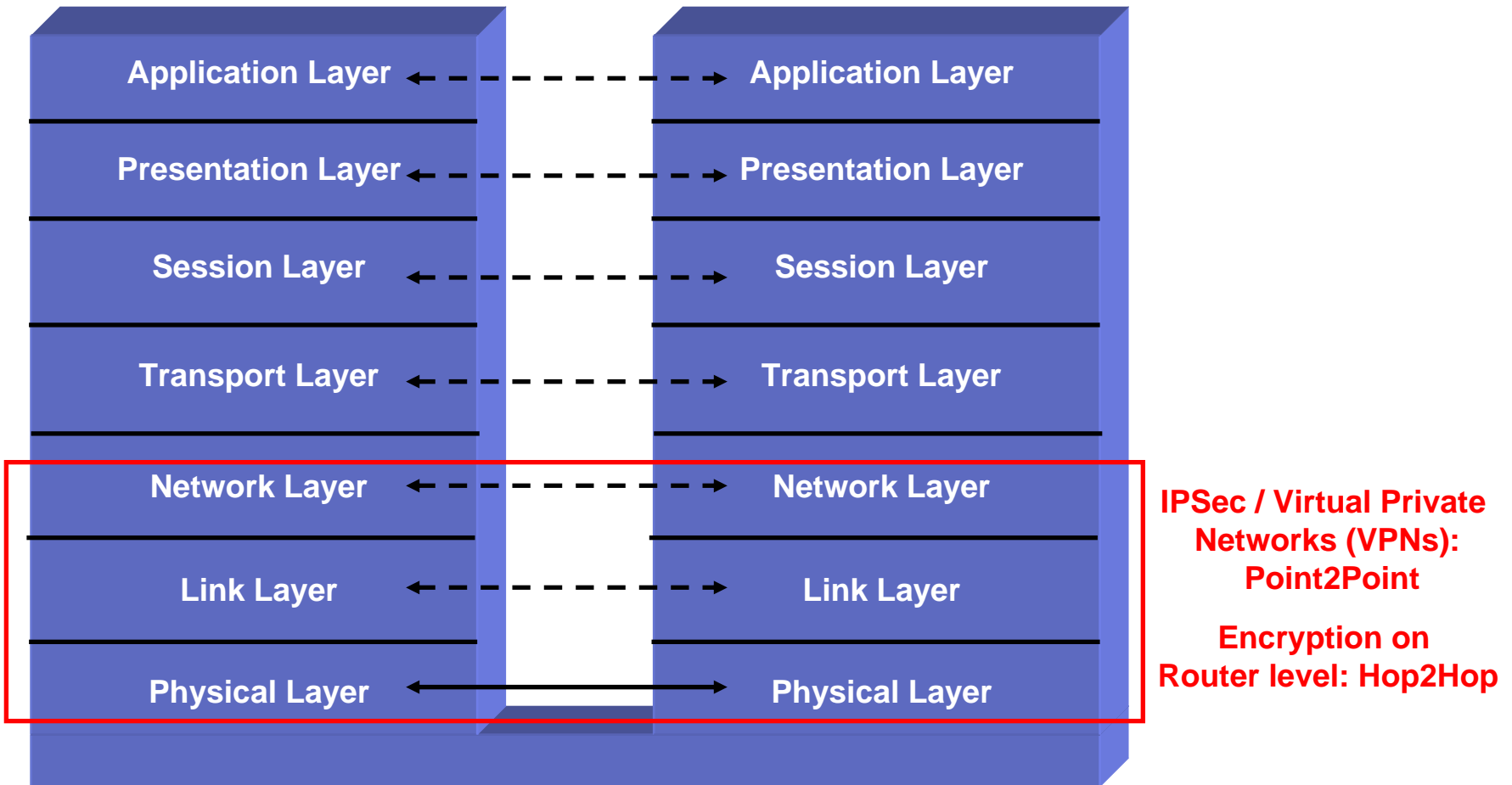
- Spoofing
- Tampering with Data
- Repudiation of Action
- Information Disclosure
- Denial of Service (DoS)
- Escalation of Privileges



## Security Requirements

- Identification
- Authentication
- Authorization
- Confidentiality & Data Protection
- Integrity
- Traceability
- Non-Repudiation

# ISO OSI Model and Security Mechanisms

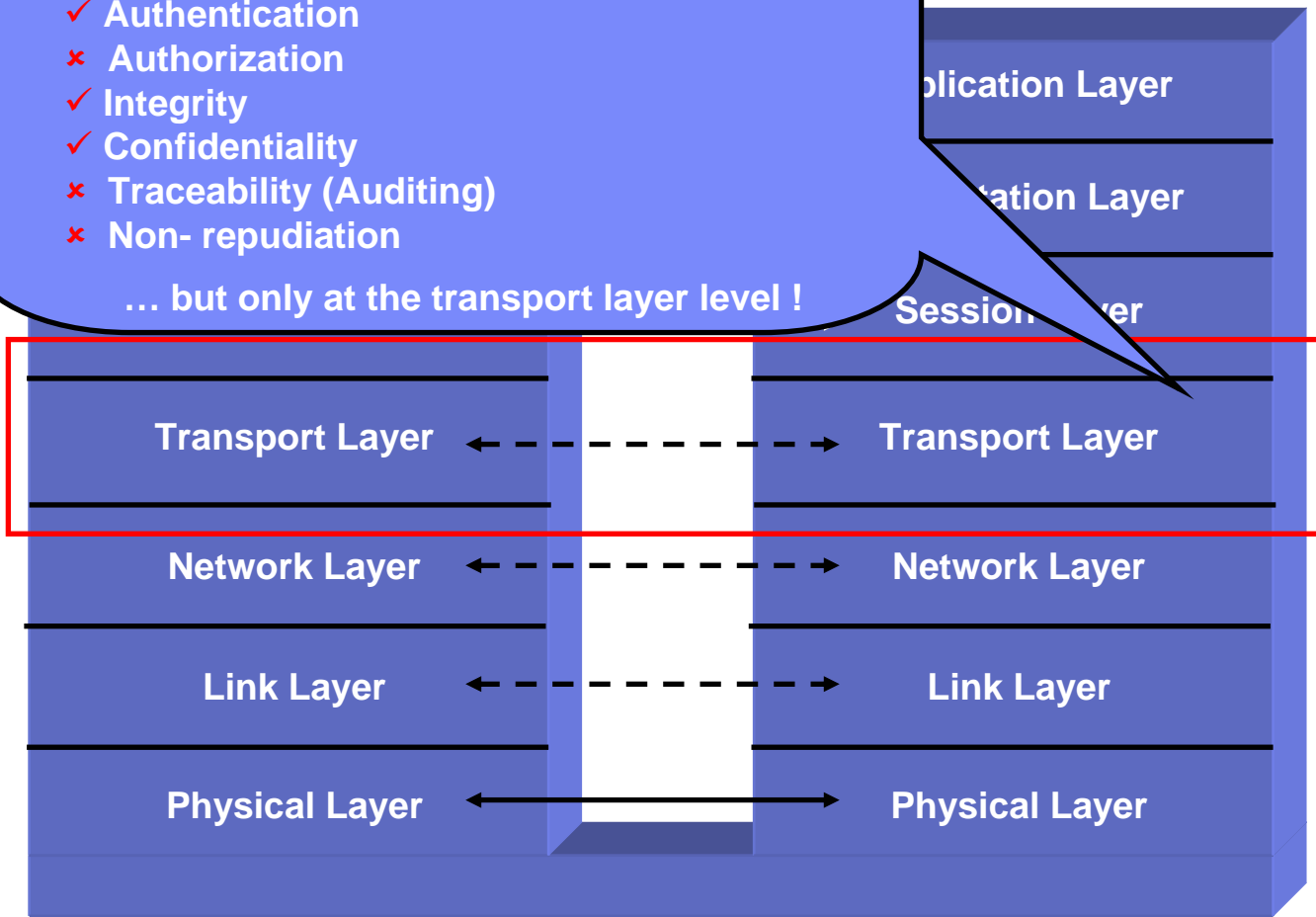


What do SSL / HTTPS and HTTP BA offer ?

- ✓ Identification
- ✓ Authentication
- ✗ Authorization
- ✓ Integrity
- ✓ Confidentiality
- ✗ Traceability (Auditing)
- ✗ Non- repudiation

... but only at the transport layer level !

Mechanisms

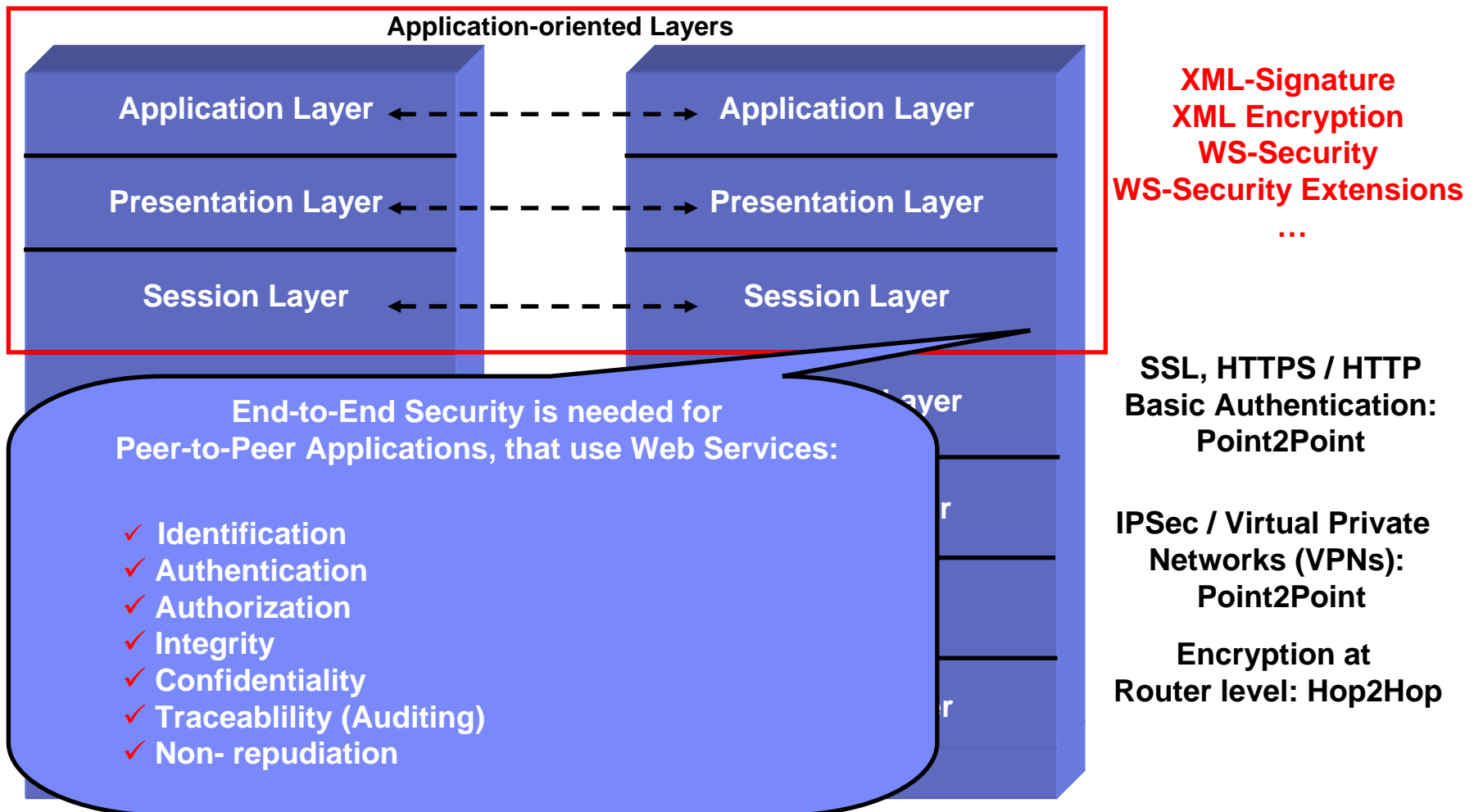


**SSL, HTTPS / HTTP  
Basic Authentication:  
Point2Point**

**IPSec / Virtual Private  
Networks (VPNs):  
Point2Point**

**Encryption at  
Router level: Hop2Hop**

# ISO OSI Model and Security Mechanisms



## Web Services Security Requirements

---

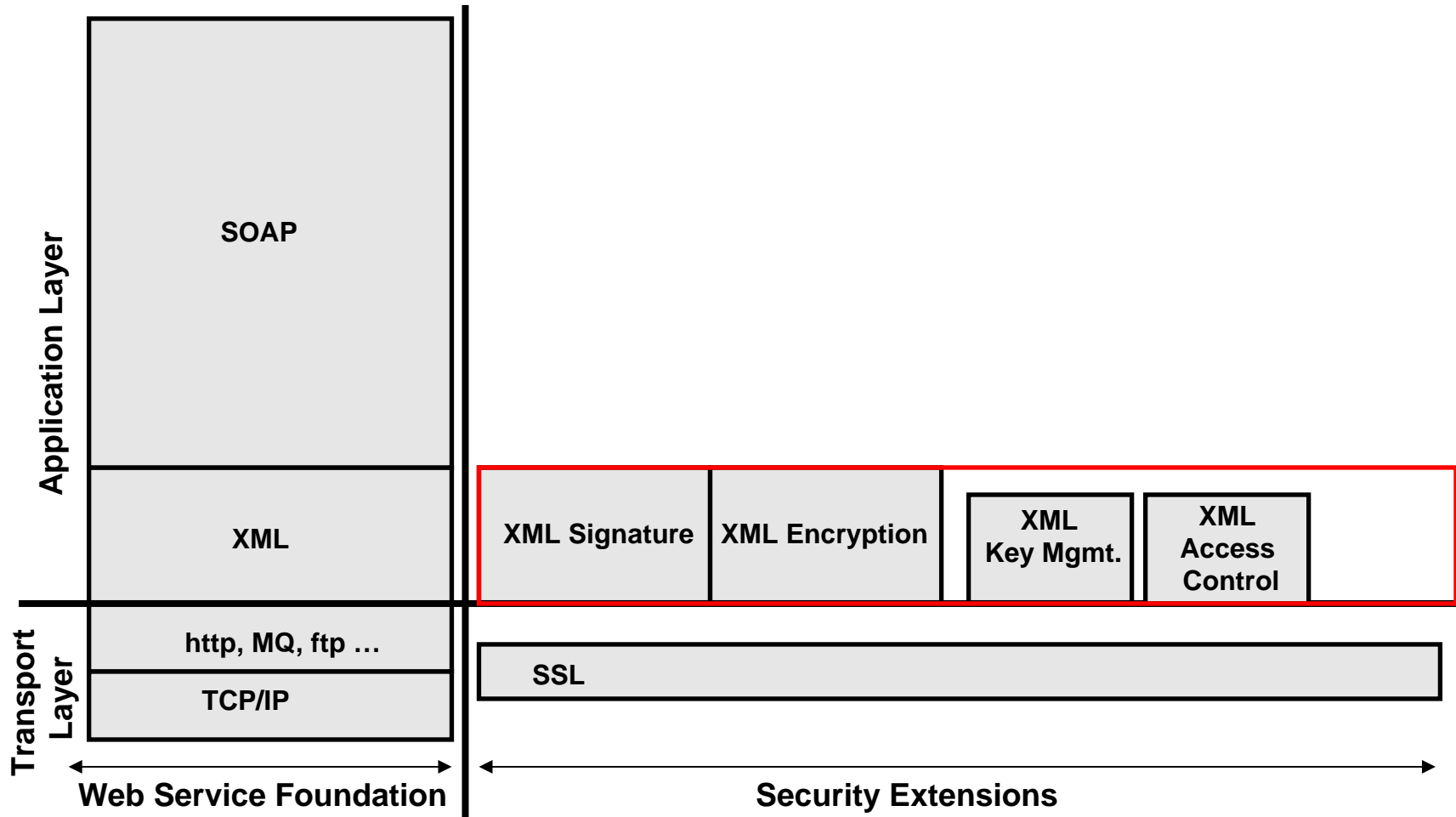
- **Interoperable**
  - ▶ People, systems, applications, and services
  - ▶ Seamlessly with reliable messaging and transactions
  - ▶ Heterogeneous environments
  - ▶ Information/processes flow across application networks
- **Autonomous Security**
  - ▶ Individual services must be autonomous
  - ▶ Operate with intermittent connectivity
- **Dynamic Security**
  - ▶ Assume change
  - ▶ Evolving set of participants
  - ▶ Mobile clients and servers
- **Decentralized Security**
  - ▶ Not owned/operated by a single entity
  - ▶ Reflect political, social, economic forces
  - ▶ Arbitrary network topology
  - ▶ Support existing business models, not force them to change
- **Internet-Ready Security**
  - ▶ Reach, scale, and capabilities to mirror the today's Internet-based world

## Agenda

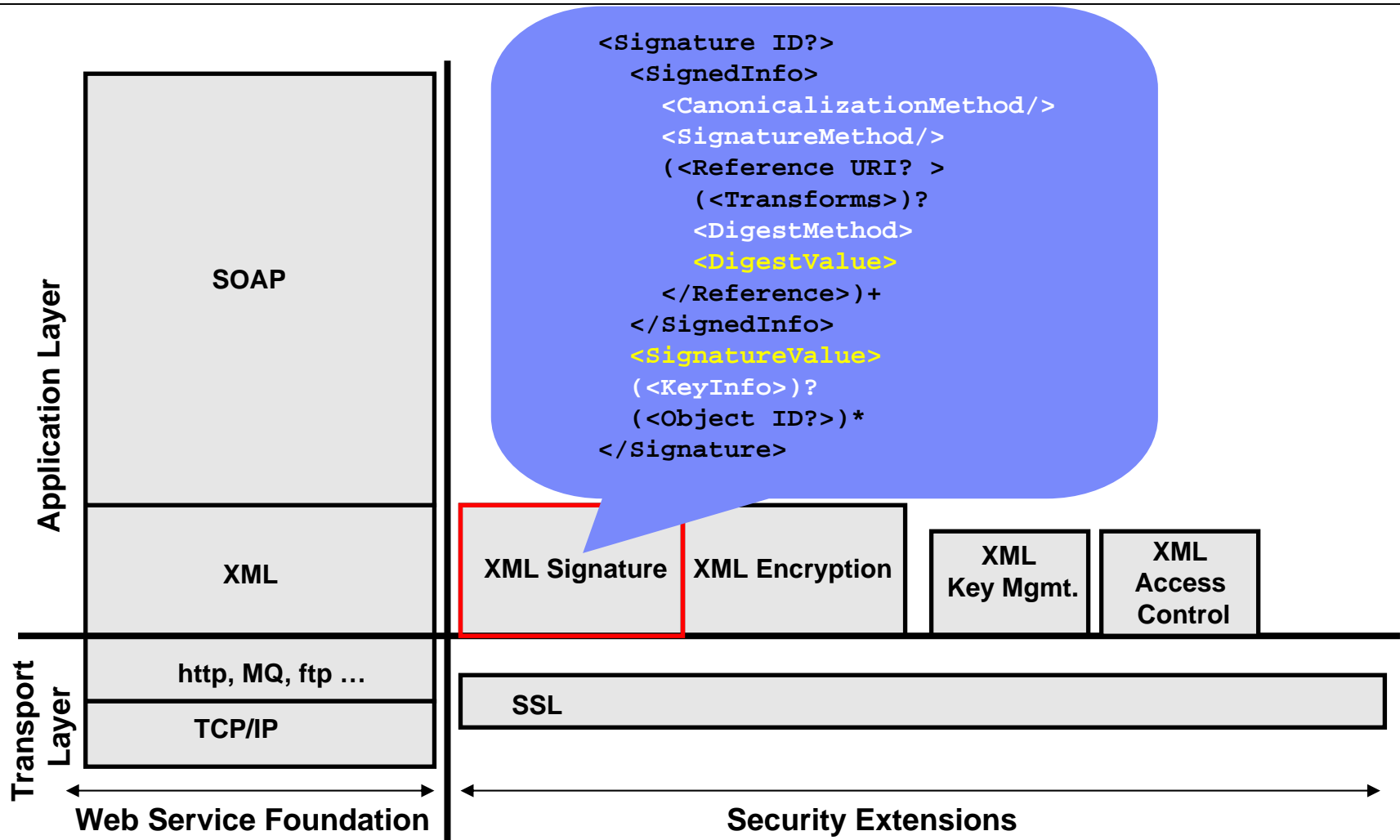
---

|    |  |
|----|--|
| 1. | Security Requirements for Peer-to-Peer Applications  |
| 2. | Web Services Security Details                        |
| 3. | WS-Security Extensions Overview                      |
| 4. | WS-Security: Support in WebSphere Application Server |

# Layers of Security – XML Security



# Layers of Security – XML Security – XML Signature



## Details about Digital Signatures

---

### Sender:

1. Creates a "digest" of the data to be signed (hash value)
2. Encryption of the "digest" using the Sender's private key
3. Result: the Digital Signature
4. The Digital Signature is transmitted along with the data

### Receiver:

1. Decryption of the Signature using the Sender's public key,  
Result: the "digest" in cleartext
2. Creating its own "digest" from the data sent
3. Comparison of the Sender's "digest" with the "digest" created by the Receiver; If both values are identical, the integrity is proven.

More details: <http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/Overview.html#sec-Algorithms>

## Example: XML Signature

```
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
```

```
<SignedInfo>
```

What is signed?

```
  <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
  <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
  <Reference URI="#wssecurity_body_id_2601212934311668096_1040651106378">
    <Transforms>
      <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
    </Transforms>
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
    <DigestValue>AWQKpmksMpzzT4PxcizO980gVHw=</DigestValue>
  </Reference>
```

```
</SignedInfo>
```

Signature Value

```
<SignatureValue>bNhT+DsNN9PR [binary data has been truncated]</SignatureValue>
```

```
<KeyInfo>
```

Public Key (optional)

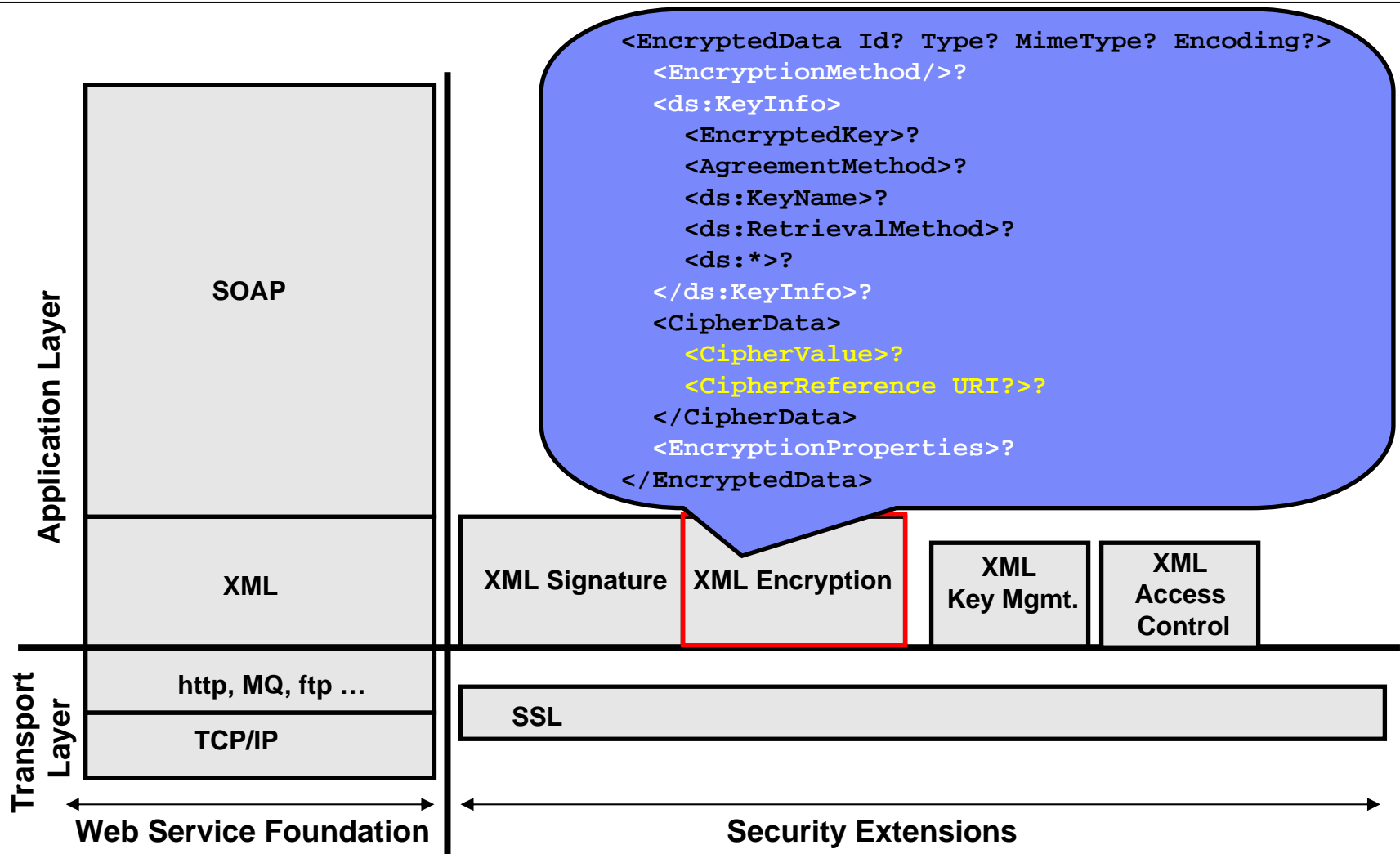
```
  <wsse:SecurityTokenReference>
    <wsse:Reference URI="#wssecurity_binary_security_token_id_1603091_4272645" />
  </wsse:SecurityTokenReference>
```

```
</KeyInfo>
```

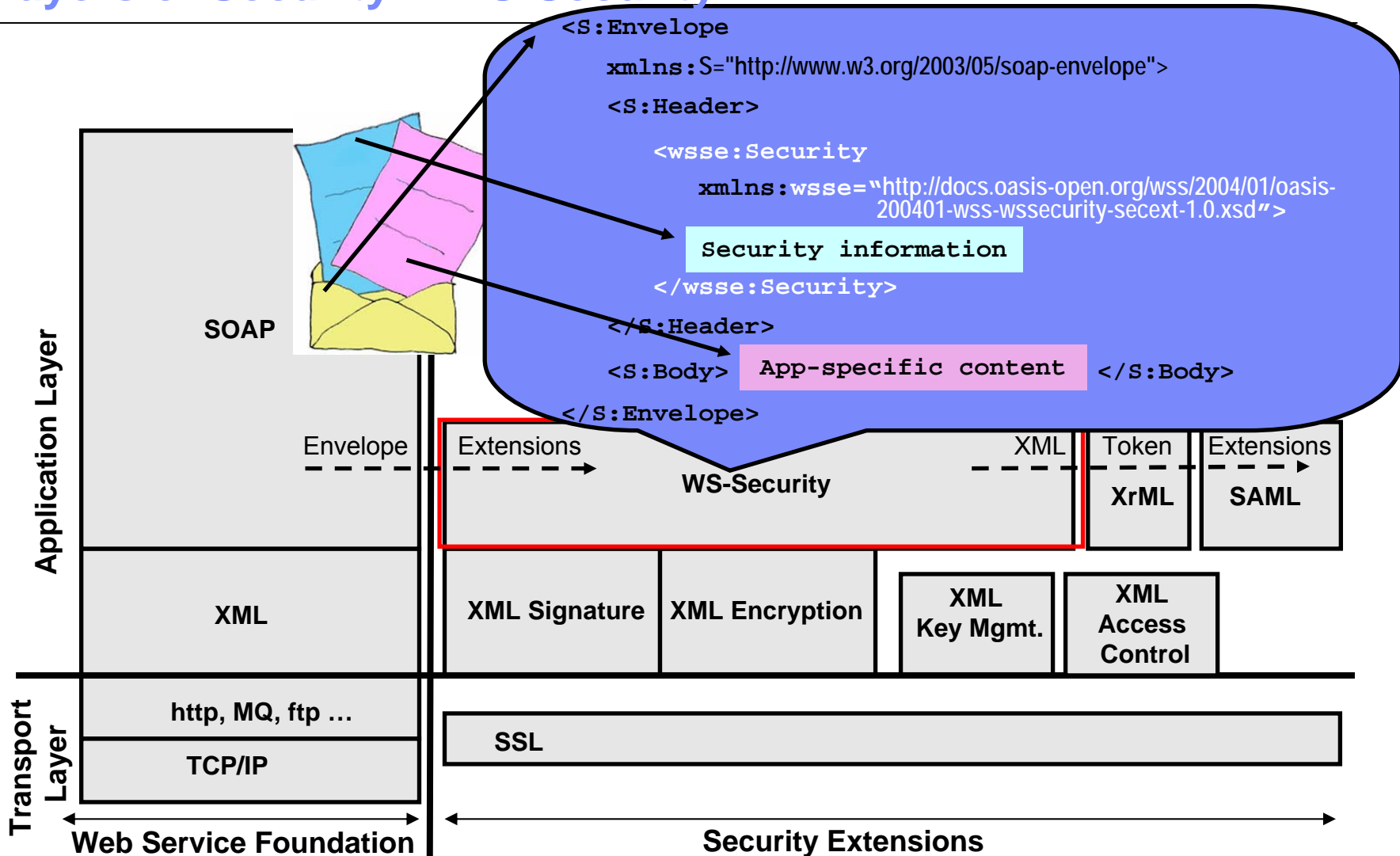
```
</Signature>
```

Signature Block

# Layers of Security – XML Security – XML Encryption

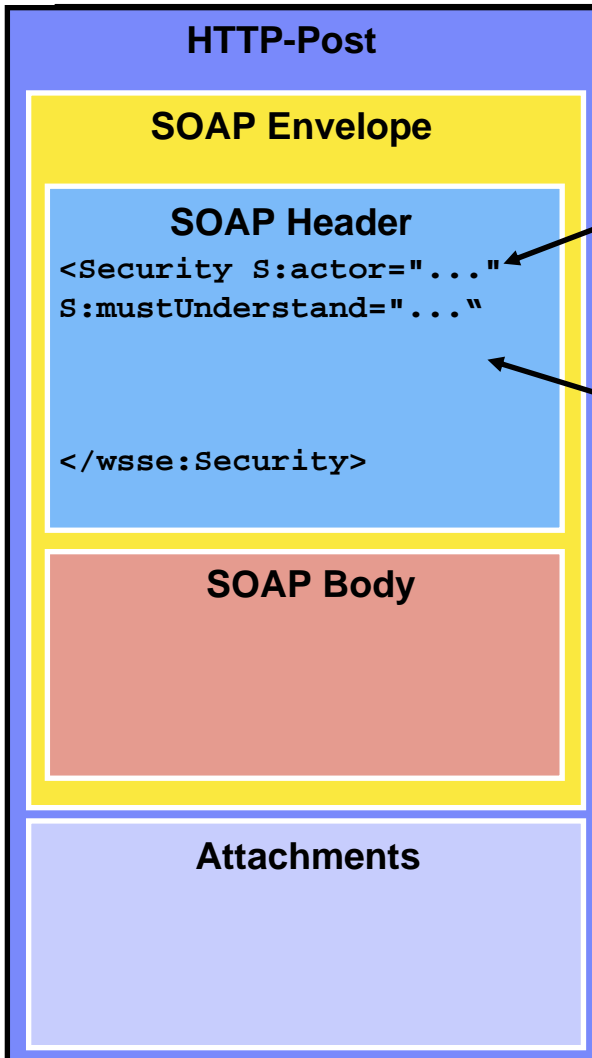


# Layers of Security – WS-Security



# WS-Security: SOAP Message Security 1.0 (WS-Security 2004)

## – OASIS Standard 200401, March 2004

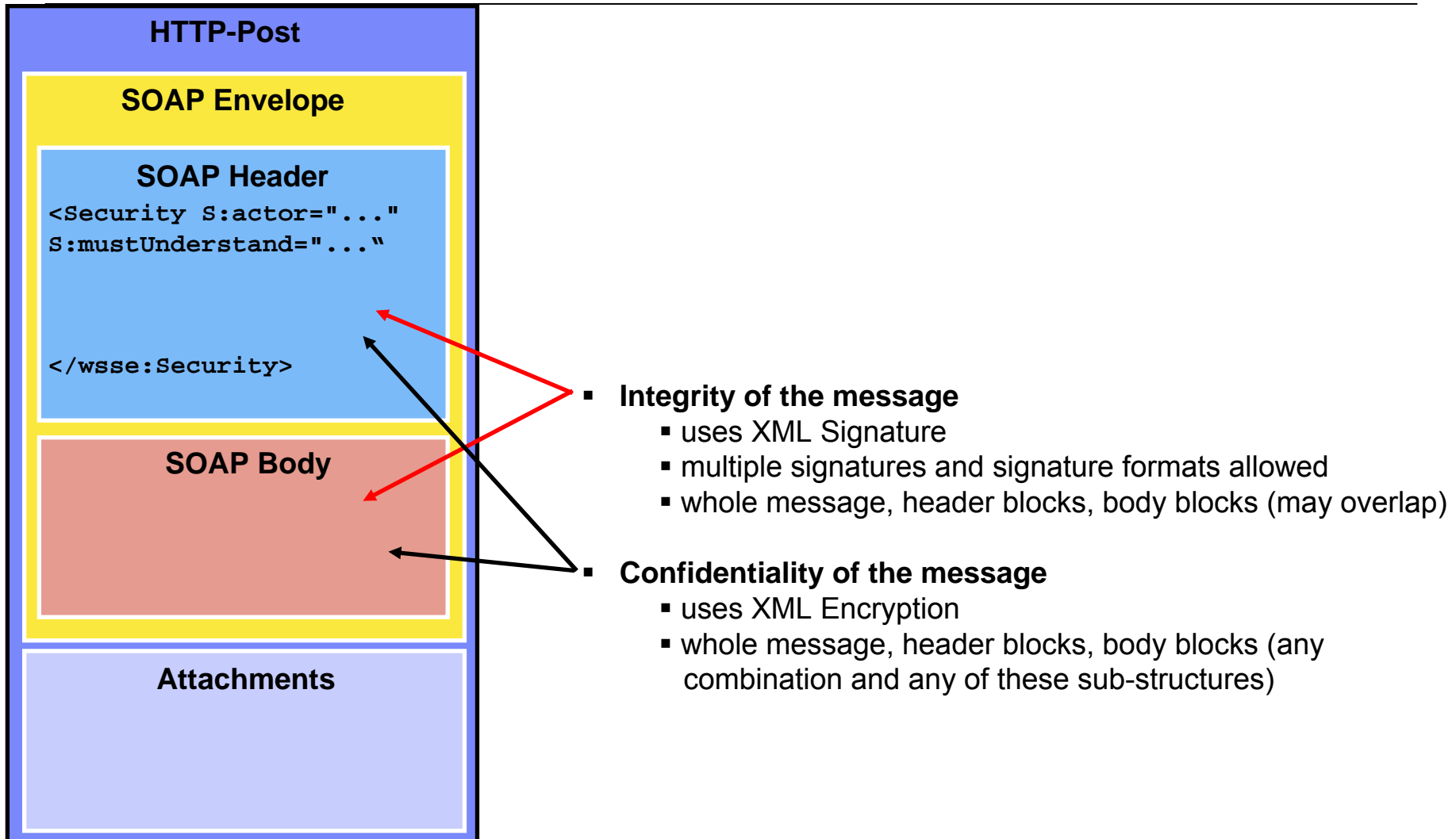


- **WS-Security**
  - Security vocabulary to be used inside SOAP Envelope
- **<Security> block to be used inside SOAP Header**
  - Security-related information can be addressed to one or more receivers (SOAP actors)
  - Can be used multiple times
- **Identification and Authentication**
  - Username Token (Unsigned Security Token), e.g.
    - Username, Password
  - Binary Security Tokens (Signed Security Tokens ), e.g.
    - X.509 certificate
    - Kerberos Ticket
    - other non-XML Tokens
  - XML Tokens
    - Identifying and referencing Security Tokens:
 

```
<wsse:SecurityTokenReference wsu:Id="...">
                ...
              </wsse:SecurityTokenReference>
```

# WS-Security: SOAP Message Security 1.0 (WS-Security 2004)

## – OASIS Standard 200401, March 2004



## WS-Security – Additional Elements

---

- Originally defined as an addendum to WS-Security Specification (July 2002)
- **Now:** part of WS-Security 2004
  
- **ID References**
  - ▶ Id References: used to identify and to reference XML elements
  - ▶ Mainly to reference parts of the SOAP message where XML Encryption and / or XML Signature will be used
  - ▶ `<anyElement wsu:Id="..." > </anyElement>`
  - ▶ Defined by Web Service Utility Namespace:  
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
  
- **Security Timestamps**
  - ▶ E.g. for creation time and expiration time of a message (to be defined within Security Header)
  - ▶ Does **not** include an agreement how to synchronize time between receiver and sender
  - ▶ Example:

```
<wsu:Timestamp wsu:Id="timestamp">
    <wsu:Created>2001-09-13T08:42:00Z</wsu:Created>
    <wsu:Expires>2001-10-13T09:00:00Z</wsu:Expires>
    ...
</wsu:Timestamp>
```

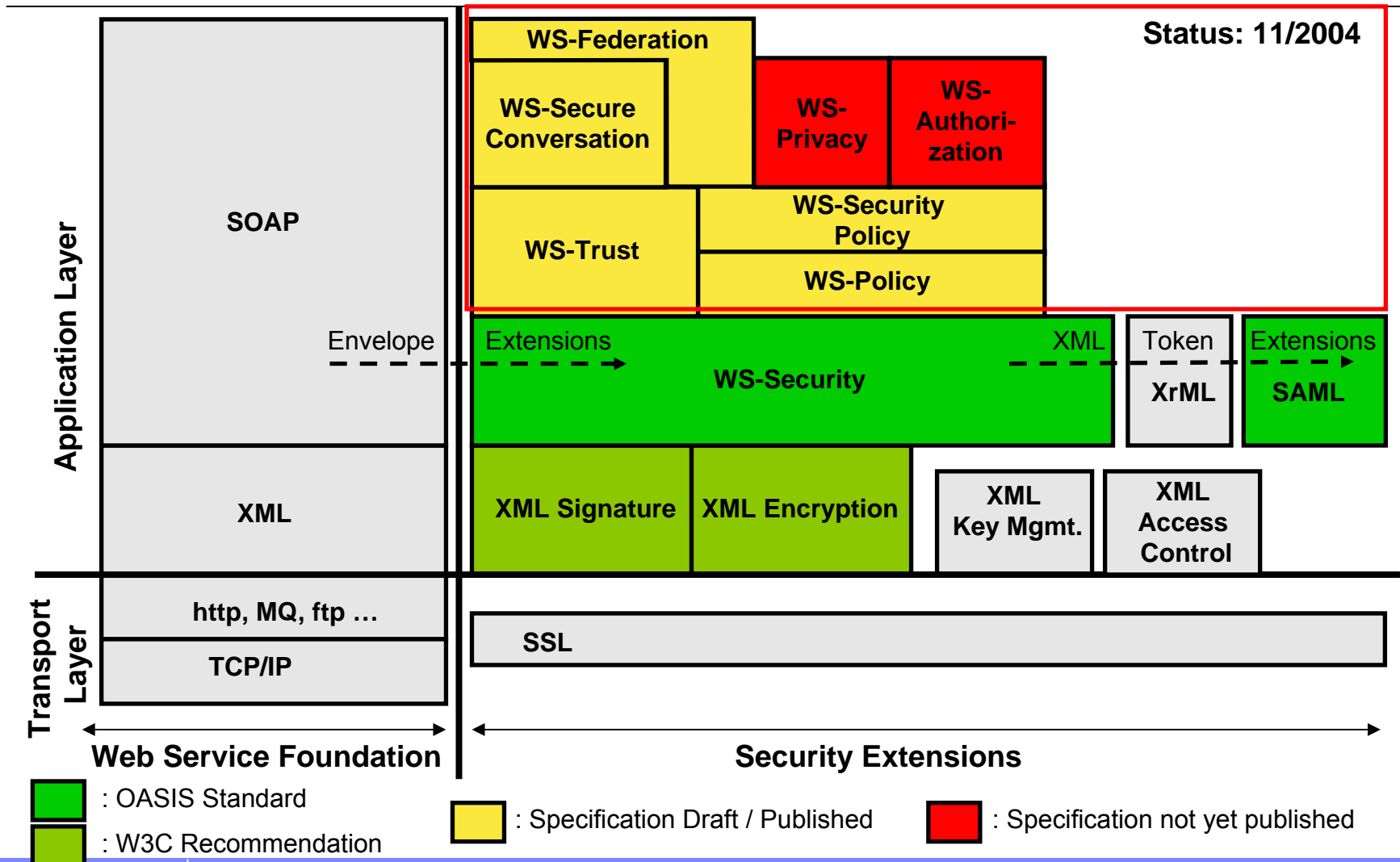


## Agenda

---

|    |  |
|----|--|
| 1. | Security Requirements for Peer-to-Peer Applications  |
| 2. | Web Services Security Details                        |
| 3. | WS-Security Extensions                               |
| 4. | WS-Security: Support in WebSphere Application Server |

# Layers of Security – WS-Security Extensions





## WS-Policy (~Attachment, ~Policy Assertion ~Security Policy)

- **Policy Expression**
  - ▶ XML InfoSet
  - ▶ Describes combinations of assertions
- **Operators**
  - ▶ Describes semantics of combinations of assertions
- **Assertions**
  - ▶ The leaf nodes in the policy expressions
  - ▶ No core assertions defined in WS-Policy
  - ▶ New assertions can be defined using namespace mechanisms

```

<Policy>
  <ExactlyOne>
    <All>
      <SecurityToken ...STA.../>
      <Algorithm ...AA.../>
    </All>
    <All>
      <SecurityToken ...STB.../>
      <Algorithm ...AB.../>
    </All>
  </ExactlyOne>
</Policy>

```

```

<wsp:Policy xmlns:wsse="..." xmlns:wsp="...">
  <wsp:ExactlyOne>
    <wsse:SecurityToken wsp:Usage="wsp:Required" wsp:Preference="100">
      <wsse:TokenType>wsse:Kerberosv5TGT</wsse:TokenType>
    </wsse:SecurityToken>
    <wsse:SecurityToken wsp:Usage="wsp:Required" wsp:Preference="1">
      <wsse:TokenType>wsse:X509v3</wsse:TokenType>
    </wsse:SecurityToken>
  </wsp:ExactlyOne>
</wsp:Policy>

```

### Example: WS-Security Policy Assertion

## WS-Trust – Web Services Trust Language

- Extensions for WS-Security
- Framework for requesting and issuing security tokens and to broker trust relationships
- Security Token Framework – Request
 

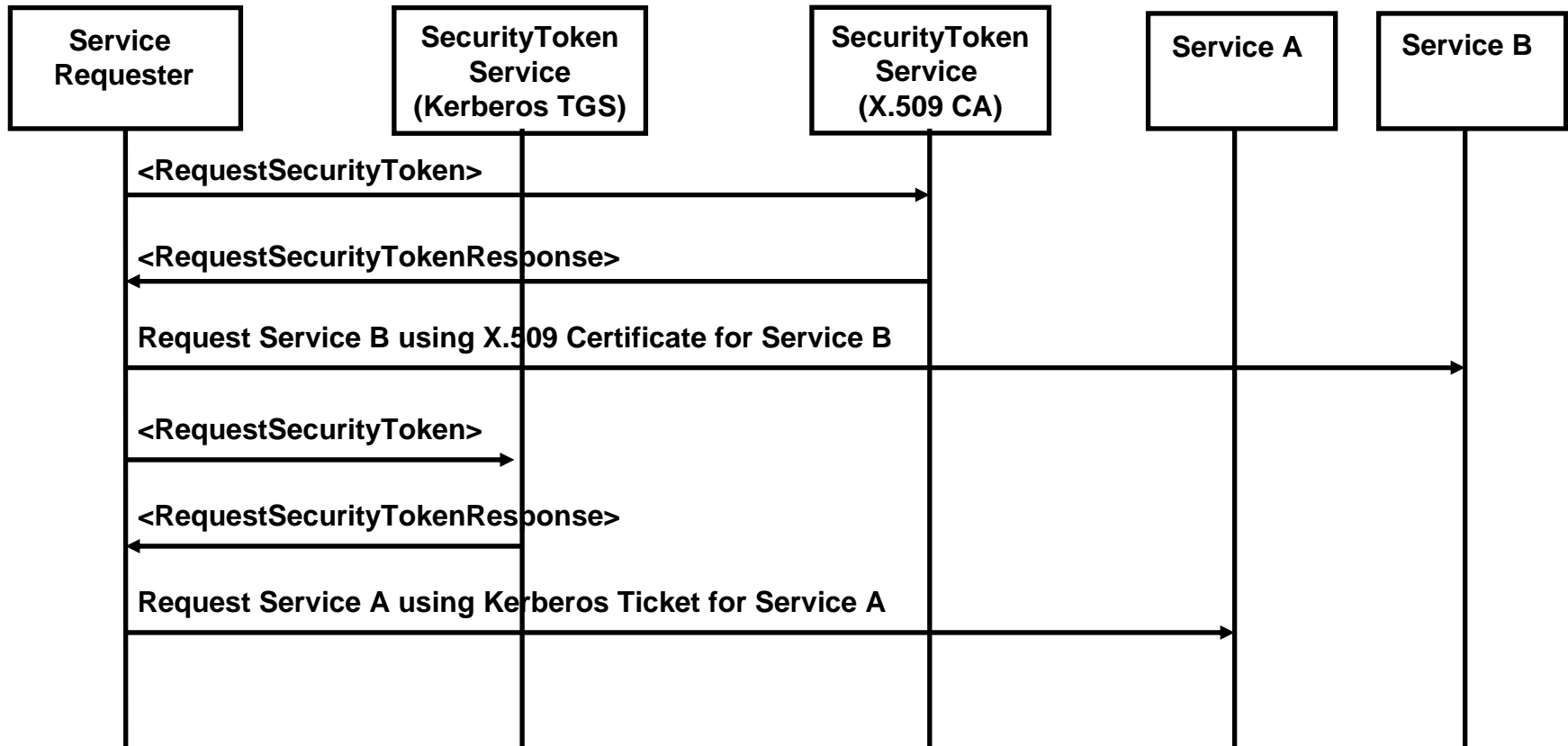
```

      <RequestSecurityToken Context="...">
        <TokenType>...</TokenType>
        <RequestType>...</RequestType>
        <Base>...</Base>
        <Supporting>...</Supporting>
        ...
      </RequestSecurityToken>
      
```
- Security Token Framework – Response
 

```

      <RequestSecurityTokenResponse Context="...">
        <TokenType>...</TokenType>
        <RequestedSecurityToken>
        </RequestedSecurityToken>
        ...
      </RequestSecurityToken>
      
```
- Specific Bindings extend the framework
  - ▶ Issuance Binding / Returning multiple Security Tokens
  - ▶ Renewal Binding
  - ▶ Validation Binding
- Negotiation and Challenge Extensions
  - ▶ Negotiation and Challenge Framework
  - ▶ Signature Challenges
  - ▶ Binary Exchanges and Negotiations
  - ▶ Key Exchange Tokens
  - ▶ Custom Exchanges
- Key and Token Parameter Extensions
- Key Exchange Token (KET) Bindings

## WS-Trust – Example: SecurityToken Request



## WS-SecureConversation

### Requirements:

- Need ability to create a secured context for multi-message exchanges
- Should look and behave as any other security token (i.e., integrate with WS-Security)
- Needs to be uniquely identifiable, but with a specified lifetime
- Optionally carry encrypted keys or reference keys
- Allow multi-party (> 2) conversations
- Must be extensible

### Example: SecurityContextToken

- SecurityContext Header Element identifies the security context using a URI
  - indicates the creation time of the security context
  - Indicates the expiration time of the security context
  - Holds the shared secrets of the security context
  - References the shared secret of the security context
- ```

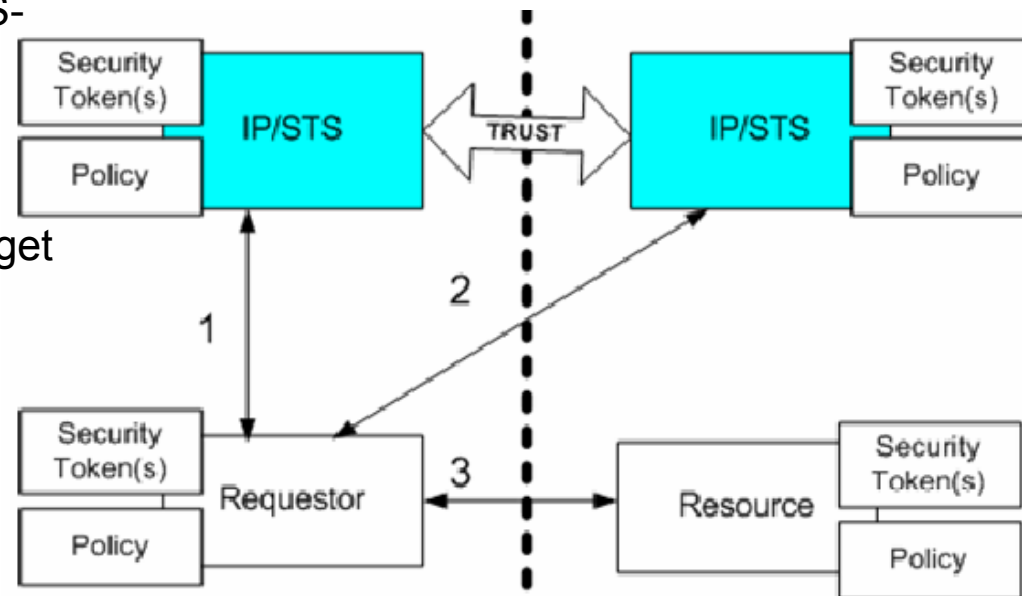
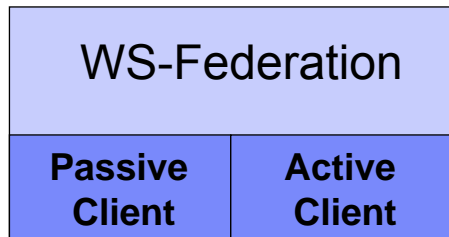
<wsse:SecurityContextToken wsu:Id="...">
  <wsu:Identifier>...</wsu:Identifier>
  <wsu:Created>...</wsu:Created>
  <wsu:Expires>...</wsu:Expires>
  <wsse:Keys>
    <xenc:EncryptedKey Id="...">...</xenc:EncryptedKey>
  <wsse:SecurityTokenReference>...</wsse:SecurityTokenReference>
  ...
</wsse:Keys>
</wsse:SecurityContextToken>

```

# WS-Federation

**The primary goal of this specification is to enable federation of identity, attribute, authentication, and authorization information.**

- Basis for federation: WS-Security, WS-Trust, WS-Policy
- Brokering of trust and security token exchange
- Local identities are not required at target services
- Optional hiding of identity information and other attributes
- Extends WS-Trust model to allow attributes and pseudonyms
- WS-Federation is made up of three specifications:



**Example: Basic Identity Provider / Security Token Service Scenario**

## Agenda

---

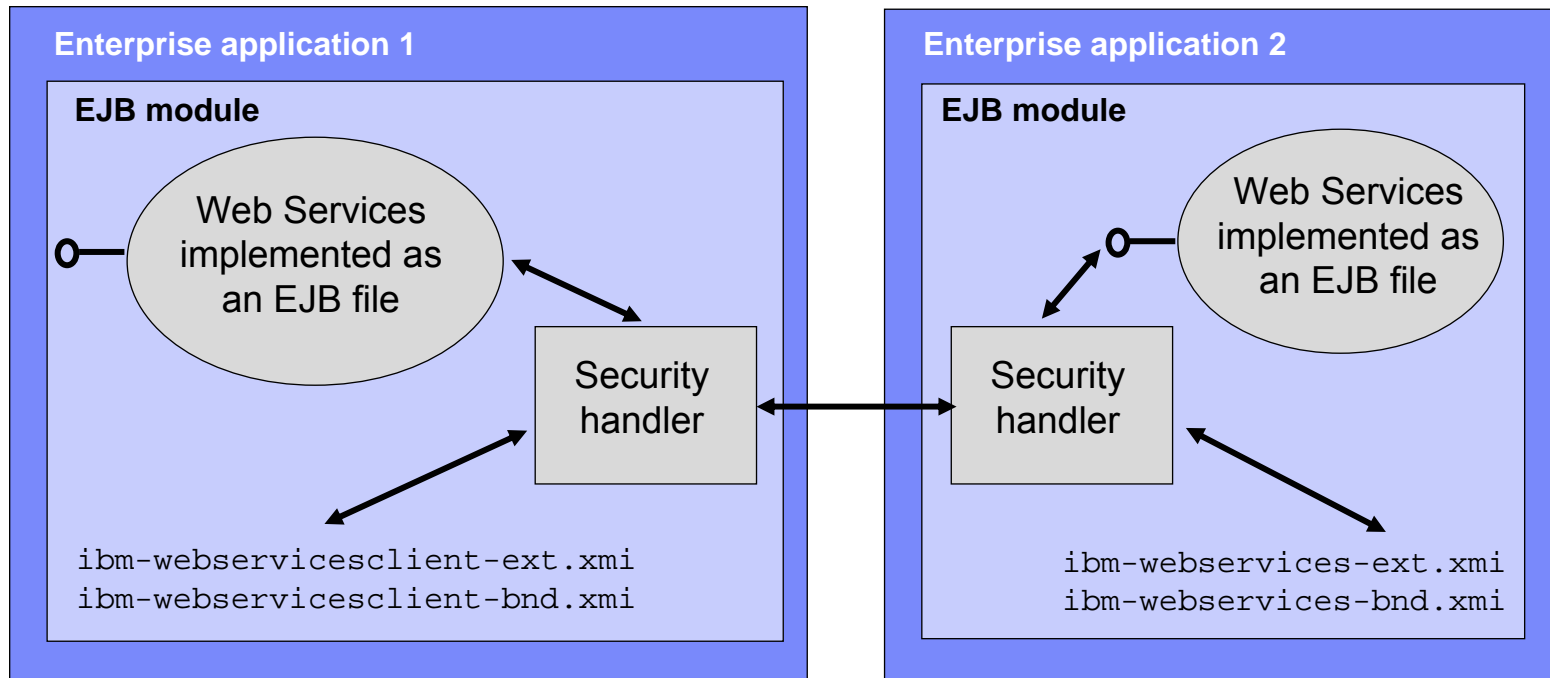
|    |                                                      |
|----|------------------------------------------------------|
| 1. | Security Requirements for Peer-to-Peer Applications  |
| 2. | Web Services Security Details                        |
| 3. | WS-Security Extensions                               |
| 4. | WS-Security: Support in WebSphere Application Server |

## WebSphere Web Services Security Support

---

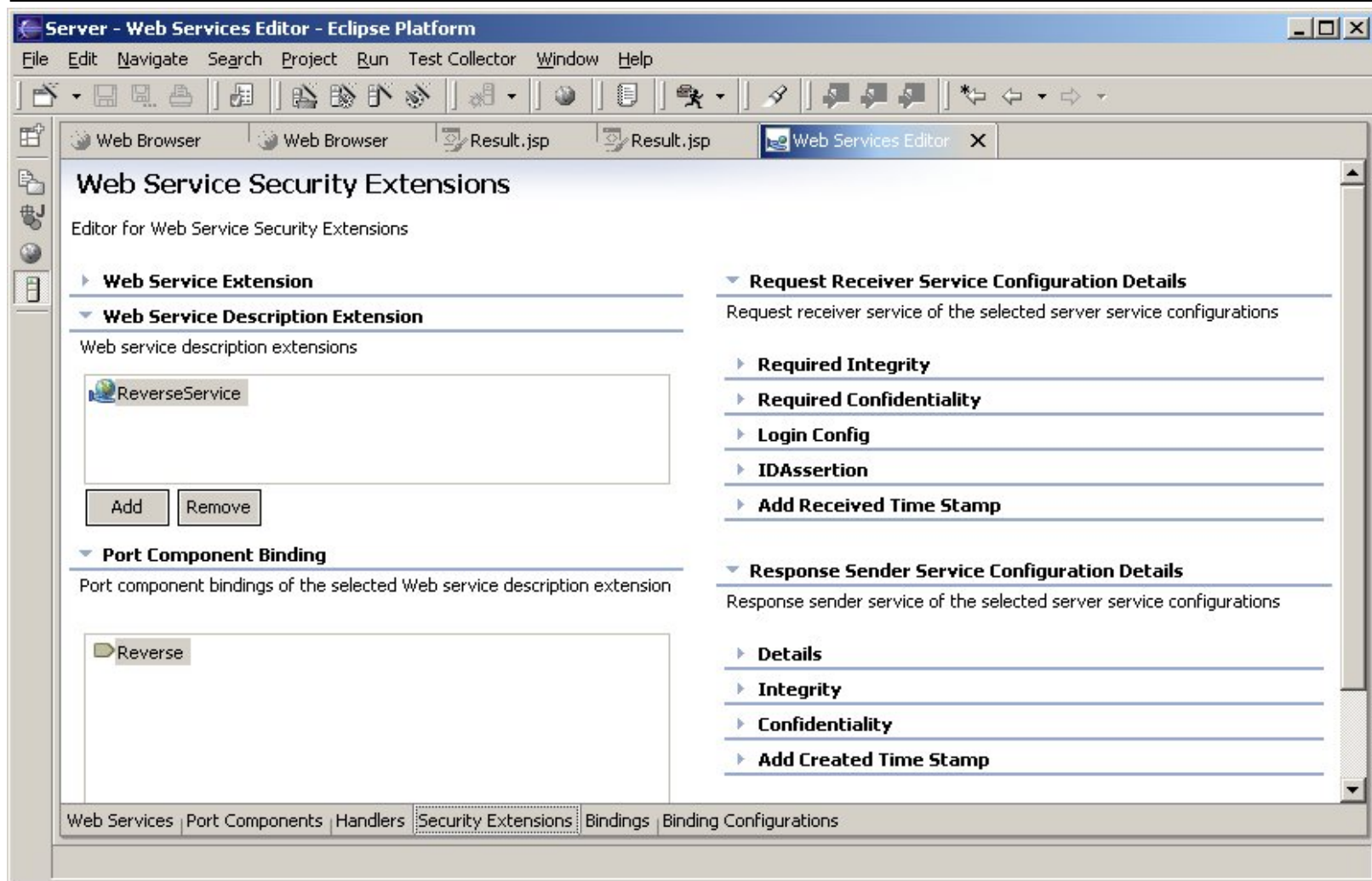
- Authentication
  - ▶ J2EE Authentication (http(s)/basic authentication - Transport level Security)
  - ▶ WS-Security (Message level authentication) – Web Services Security Extensions (WSSE)
    - Basic Authentication method
    - Digital Signature Authentication method
    - Identity propagation (IDAssertion Authentication Method)
    - LTPA
  
- Message Security
  - ▶ Integrity ensured using Digital Signature
  - ▶ Confidentiality using Encryption
  
- Authorization is based on existing J2EE Authorization mechanisms
  - ▶ Session bean – EJB security
  - ▶ Java bean – URL protection of SOAP entry point

## Web Services Security Model in WAS 5.x



- A „**Declarative Security Model**“ is supported; no programatic interactions with Web Services Security
- Service Provider APIs (SPIs) allow to enhance some security behaviour
- Security Constraints have to be defined in Deployment Descriptor Extensions
  - ▶ Client: `ibm-webservicesclient-ext.xmi` und `ibm-webservicesclient-bnd.xmi`
  - ▶ Server: `ibm-webservices-ext.xmi` und `ibm-webservices-bnd.xmi`

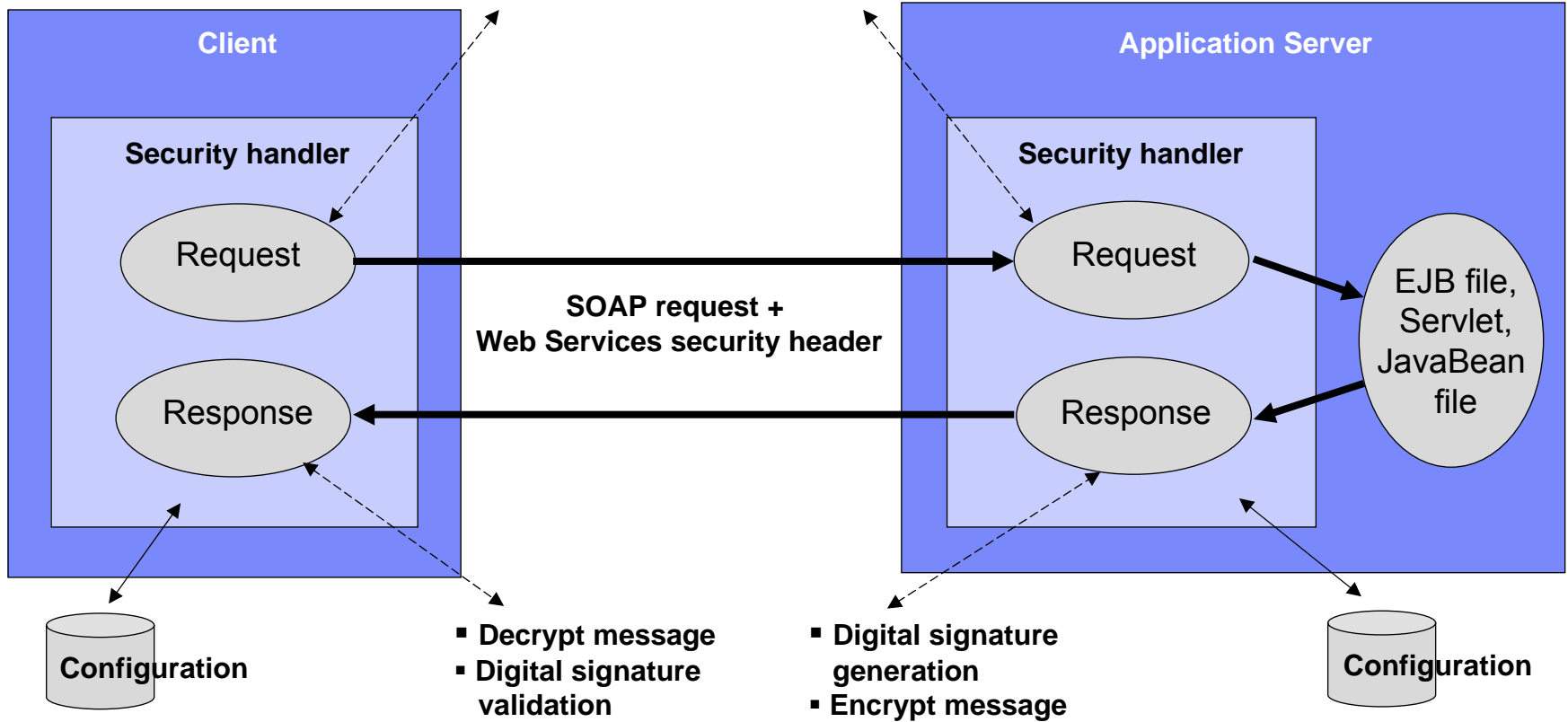
## Tool Support: Web Services Editor - Security tab



# Web Services Security Message Interpretation

- Security token generation
- Digital signature generation
- Encrypt message

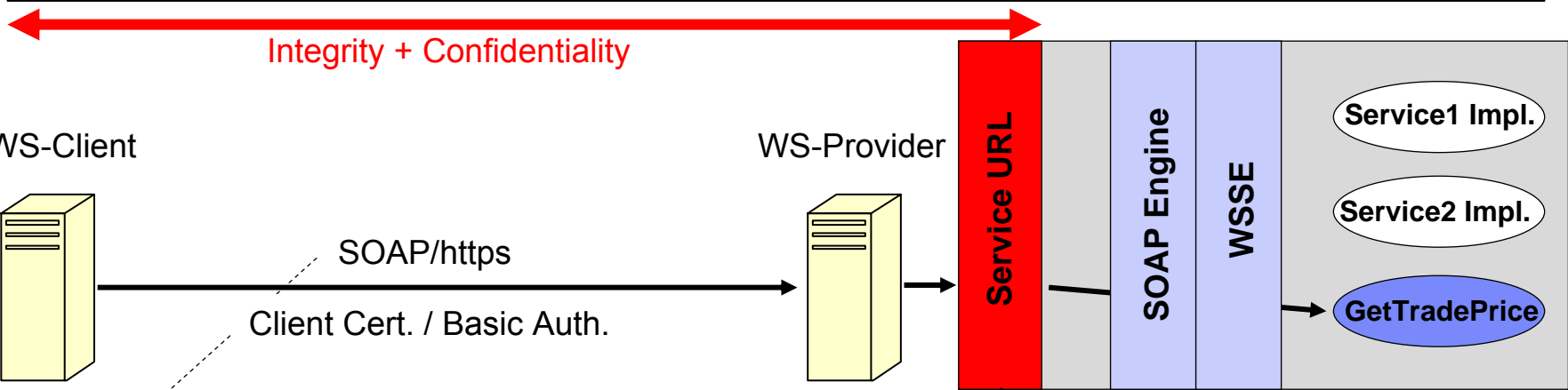
- Decrypt message
- Digital signature validation
- Security token validation and setup security context



ibm-webservicesclient-ext.xmi  
 ibm-webservicesclient-bnd.xmi

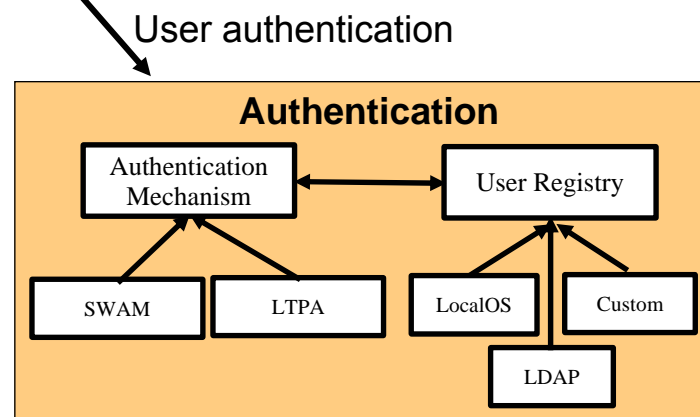
ibm-webservices-ext.xmi  
 ibm-webservices-bnd.xmi

# Scenario1: Transport Level Security only

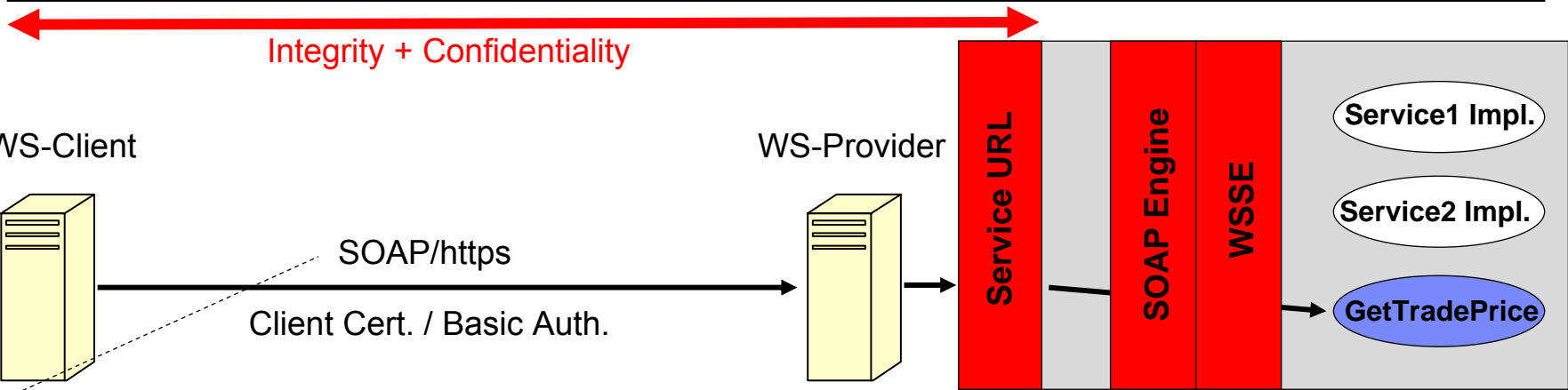


```

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header> ...
</SOAP-ENV:Header>
  <SOAP-ENV:Body id="Body">
    <m:GetTradePrice xmlns:m="some-URI">
      <m:symbol>IBM</m:symbol>
    </m:GetTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
    
```

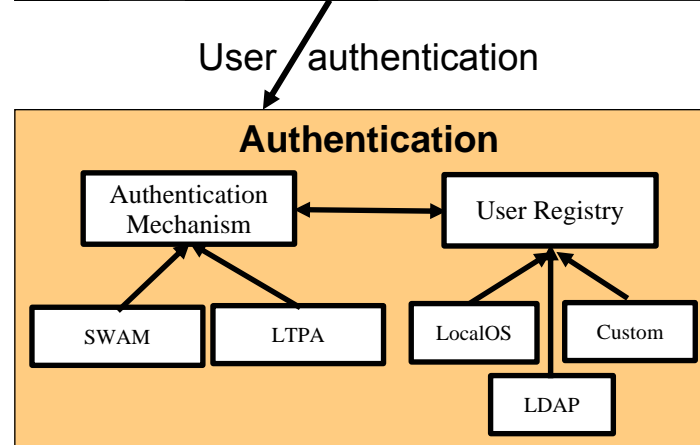


## Scenario2: WSSE for Authentication; SSL for the rest

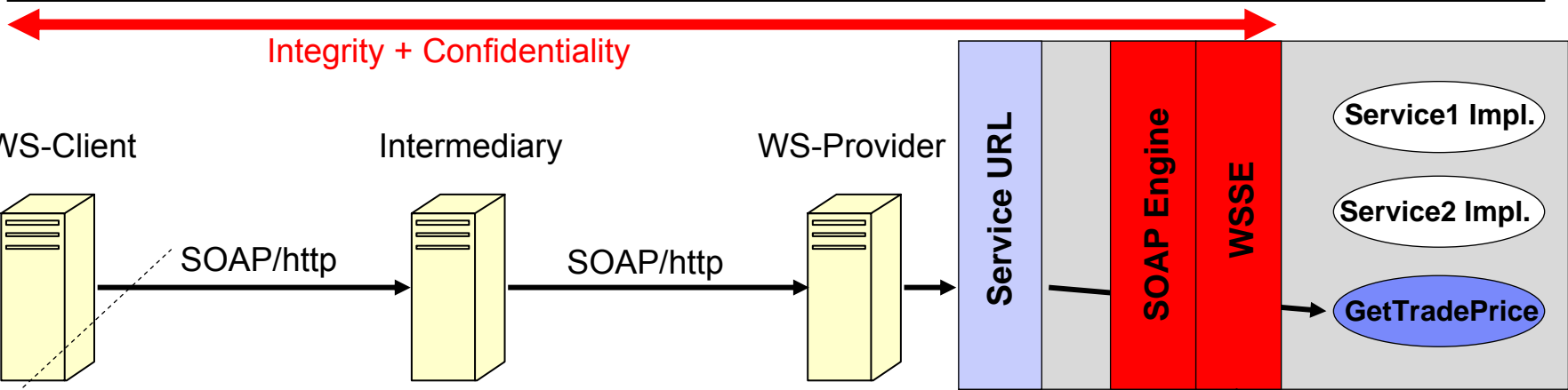


```

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header>
<wsse:Security
xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12">
<wsse:UsernameToken>
<wsse:Username>Bob</wsse:Username>
<wsse:Password>IloveDogs</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
</SOAP-ENV:Header>
<SOAP-ENV:Body id="Body">
<m:GetTradePrice xmlns:m="some-URI">
<m:symbol>IBM</m:symbol>
</m:GetTradePrice>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
    
```

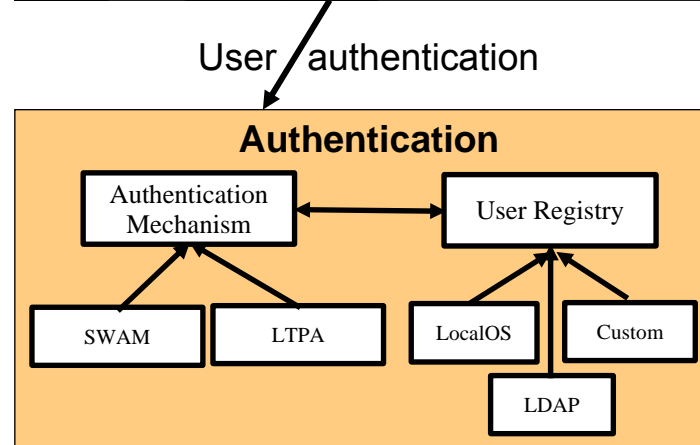


# Scenario3: WSSE for everything



```

<SOAP-ENV:Envelope (*)
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header>
    <wsse:Security
      xmlns:SOAP-SEC="http://schemas.xmlsoap.org/soap/security/2000-12">
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:Reference URI="#Body"../>
      </ds:Signature> ...
    </wsse:Security>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body id="Body">
    <m:GetTradePrice xmlns:m="some-URI">
      <m:symbol>IBM</m:symbol>
    </m:GetTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
    
```



(\*) only Integrity using XML-Signature shown in the simplified example

## Security Options – Pros & Cons

---

- Transport Level Security
  - ⊕ Recommended for high transaction volumes, good performance, scales well
  - ⊕ When interoperability has highest priority in heterogeneous environments (SSL has proven interoperability and high maturity)
  - ⊕ If attachments need to be encrypted (SSL encrypts entire packets of the transport layer)
  - ⊕ If clients are standalone Java applications
  - Intermediary processing (e.g. filtering or content based routing) ist not possible because of SSL
- WSSE for Authentication SSL for the rest
  - ⊕ Propagation of user identity and using WSSE capabilities in WAS to propagate the Security Context via LTPA
  - ⊕ Propagation of user identity and using WSSE capabilities to propagate the Security Context using a custom Token Format (WSSE custom security callbacks)
  - ⊕ Security Codes is decoupled from Application Code, since WSSE Stack is completely controlled by Deployment Descriptors (Declarative Security Modell)
- WSSE for all
  - ⊕ If Intermediary processing is necessary
  - ⊕ If an encrypted or signed message must be stored persistently (XML Sign. and XML Encr. are effective beyond transport layer)
  - ▶ Still performance degradations at the moment

## Conclusions

---

- Web Services Security on a message level is a major requirement for productive use of Web Services across enterprise boundaries.
- **WS-Security mechanisms are supported in WebSphere Application Server and WSAD(-IE) V5.1.**
- Security options should be selected carefully according to the non-functional requirements (performance, level of security needed, etc.) of the solution.
- **Dynamic Web Services across the Internet / across enterprise boundaries will need security protocols in addition to WS-Security (WS-Security Extensions).**
- Current WS-Security Extensions promise to support numerous application scenarios (... which could be inherently complex !)

## References (small selection)



- **„Perspectives on Web Services“, Springer Verlag**  
see: Chapter 5.4 *Securing a Web Services Implementation*
- **WS-Security Specifications**  
see: <http://www.oasis-open.org/home/index.php>
- **„Security in a Web Services World – A Proposed Architecture and Roadmap“;**  
Joint Whitepaper from IBM and Microsoft, April 2002
- **IBM Tutorial: „Configuring WS-Security Basic Authentication and confidentiality using WebSphere Studio Application Developer 5.1“ – Tom Hyland**
- **Etc.**

# Backup

---

## Example: WS-Security Signature

```
[...]
[07]     ...
[08]     xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
[09]     <s:Header>
[10]     <wsse:Security>
[11]     <wsse:BinarySecurityToken>
[12]     Value="wsse:X509v3"
[13]     EncodingType="wsse:Base64Binary"
[14]     wsu:Id="X509Certificate"
[15]     HgfZTle...
[16]     </wsse:BinarySecurityToken>
[17]     <ds:Signature>
[18]     <ds:SignedInfo>
[19]     <ds:CanonicalizationMethod Algorithm="
[20]     "http://www.w3.org/2001/10/xml-exc-c14n#">
[21]     <ds:SignatureMethod Algorithm="
[22]     "http://www.w3.org/2000/09/xmldsig#rsa-sha1/">
[23]     <ds:Reference URI="#MsgBody">
[24]     <ds:DigestMethod Algorithm="
[25]     "http://www.w3.org/2000/09/xmldsig#sha1/">
[26]     <ds:DigestValue>aBJchm4LuPW...</ds:DigestValue>
[27]     </ds:Reference>
[28]     </ds:SignedInfo>
[29]     <ds:SignatureValue>Hgjchz6gD...</ds:SignatureValue>
[30]     <ds:KeyInfo>
[31]     <wsse:SecurityTokenReference>
[32]     <wsse:Reference URI="#X509Certificate"/>
[33]     </wsse:SecurityTokenReference>
[34]     </ds:KeyInfo>
[35]     </ds:Signature>
[36]     </wsse:Security>
[37]     </s:Header>
[38]     <s:Body wsu:Id="MsgBody">
[39]     ...
[40]     </s:Body>
[41]     ...
```

## Web Services Security Elements in WAS 5.1 (1/2)

---

- **Username Token:**
  - + Username + password / BasicAuth
  - + Username / Identity assertion Auth. Method
  - Password Digest
  - + Nonce
  - Created Attributes
  
- **Binary Security Tokens:**
  - + X.509 Zertifikate
  - + LTPA
  - Kerberos Tickets

(**but:** Binary Token Generation / Verification are „pluggable“ based on JAAS APIs)
  
- **Signatures:**
  - + X.509 Certificate-based / referenced
  - Signature-based using shared keys

### Remarks:

- Based on WS-Security OASIS Working Draft13 from May 01, 2003.
- Based on WS-Security Username Token Profile Draft 2
- Not yet WS-I (Security Profile) conform, since WS-I Working Group Security Profile was released not until Mai 12, 2004

## Web Services Security Elements in WAS 5.1 (2/2)

- **Encryption:**
  - + EncryptedKey XML Tag
  - + ReferenceList XML Tag
  - + KeyIdentifier -> public keys
  - + KeyName -> secret keys
  - + Map authenticated identity to key for encryption
  - + Use signer certificate to encrypt the response msg
  
- **Timestamp:**
  - + Created attribute
  - + Expires attribute
  - Receive attribute (Addendum)
  - + TimestampTrace attribute (OASIS)
  
- **XML based Tokens:**
  - + Insert + validate arbitrary format of XMLTokens (basiert auf JAAS APIs)

### Remarks:

- Based on WS-Security OASIS Working Draft13 from May 01, 2003.
- Based on WS-Security Username Token Profile Draft 2
- Not yet WS-I (Security Profile) conform, since WS-I Working Group Security Profile was released not until Mai 12, 2004