ID2208 Programming Web Services

Web Services Security

Mihhail Matskin:
http://people.kth.se/~misha/ID2208/

Spring 2016

Content

Security

- What is security
- Approaches to security
- WS-Security
- WS-Trust
- WS-SecurityPolicy
- WS-SecureConversation
- WS-Federation

This lecture reference

Text-book **Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI, 2nd Edition**

Chapter 9

Why?

- Security is a biggest obstacle to enterprise Web services adoption
- Web services move transactions beyond firewalls and enable outside entities to invoke applications (which may give outsiders access to sensitive information)
- There are existing security standards for the Internet, however, Web services require additional measures of security

What about HTTP security mechanisms?

- HTTP allows Web servers to authenticate users before allowing access to resources (however, the HTTP provides no process for encrypting the body of a message)
- Secure Socket Layer (SSL) – supports point-to-point security (in combination with HTTP Basic Authentication). However, when messages travel through intermediaries, end-to-end security may be required
- Web Services introduce more security and privacy concerns than previous technologies.

What is security 

”If I take a letter, lock it in a safe, hide the safe somewhere in New York, then tell you to read the letter, that’s not security. That’s obscurity. On the other hand, if I take the letter and lock it in a safe and then give you the safe along with the design specifications of the safe and a hundred identical safes with their combinations so that you and the world’s best safecrackers can study the locking mechanisms – and you still can’t open the safe and read the letter – that’s security.”

From “Applied Cryptography”

Design Principles for Security

- Least privilege – (need-to-know) only the smallest set of privileges to complete the job; the access rights should be acquired by explicit permission only.
- Economy of mechanism- security mechanisms should be as small as possible; an integral part of the design.
- Acceptability - security mechanisms should not interfere unduly with the work of the users.
- Complete mediation – every access must be checked against the access control information.
- Open design – mechanisms can be reviewed by many experts, and users must have high confidence in them.

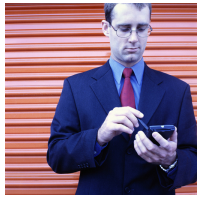
Security requirements for Web services

- Privacy/confidentiality
  - How to ensure that the transmitted information has not been captured or passed to a third party without your knowledge?
- Integrity
  - How to ensure that the sent or received information has not been compromised or altered?
- Authentication
  - How do sender and receiver verify their identities to each other?
- Authorization
  - How to manage access to protected resources on the basis of user credentials?
- Non-repudiation
  - How to prove that a message was sent or received?
- Availability
  - How to ensure that the system operates continuously

Cryptosystems

- To protect information data should be encrypted
- Cryptography transforms data using a cipher or a cryptosystem (mathematical algorithm)
- A key (string of digits that acts as password) is input to the cipher
- Unencrypted data - plaintext; encrypted data – ciphertext
- Only the intended receivers should have the key to decrypt the ciphertext into plaintext
- Symmetric and asymmetric encryptions

Symmetric encryption



sender

Buy 100 shares
of company XXX

plaintext

encrypt

Symmetric
secret key

ciphertext

HGFu76JH
%&7bvc

Same
symmetric
secret key

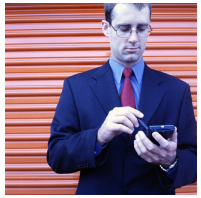
decrypt

Buy 100 shares
of company XXX

plaintext

receiver

Asymmetric encryption



sender

Buy 100 shares
of company XXX

plaintext

encrypt

Receiver's
public key

ciphertext

HGFu76JH
%&7bvc

Receiver's
private key

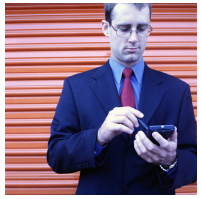
receiver

Buy 100 shares
of company XXX

plaintext

decrypt

Authentication with a public-key algorithm



sender

Buy 100 shares of company XXX

plaintext

encrypt

Receiver's public key

ciphertext

HGFu76JH %&7bvc

Sender's private key

Signed ciphertext

KHSTYJ7368 &%IUIO

receiver

Buy 100 shares of company XXX

plaintext

decrypt

Receiver's private key

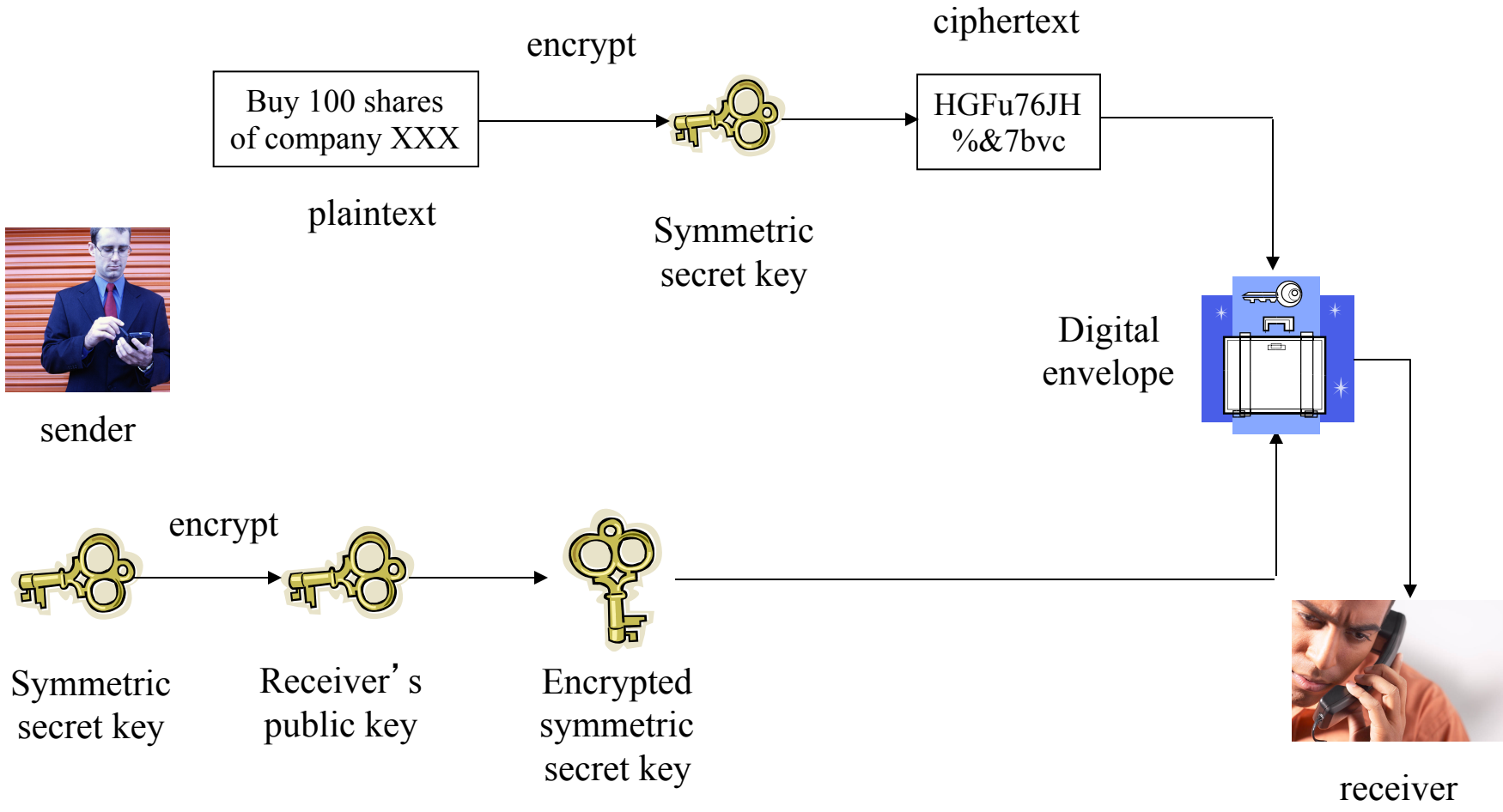
HGFu76JH %&7bvc

ciphertext

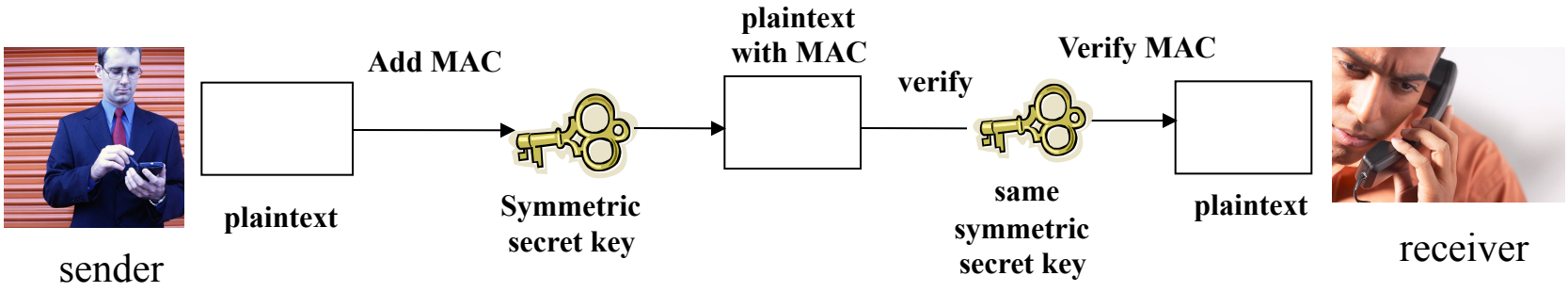
decrypt

Sender's public key (authenticates sender)

Key agreement protocols

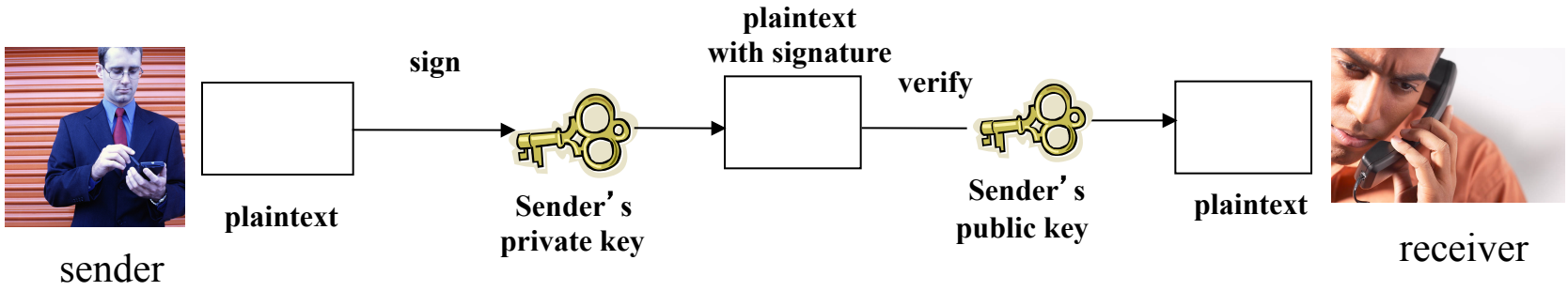


Message Authentication Code (MAC)



Digital Signatures

- The electronic equivalent of written signatures
- Uses public-key cryptography
- Asymmetric digital signature technology



Digital Signatures

sender

Buy 100 shares of company XXX

plaintext

create digest

Hash function

digest

TYAUIS &#¤6&/

Sender's private key

encrypt

Receiver's public key

HGFu76JH %&7bvc

ciphertext

Signed digest and cipher message

TYAUIS &#¤6&/ HGFu76JH %&7bvc

Sender's public key

decrypt

TYAUIS &#¤6&/

Hash function

receiver

Buy 100 shares of company XXX

plaintext

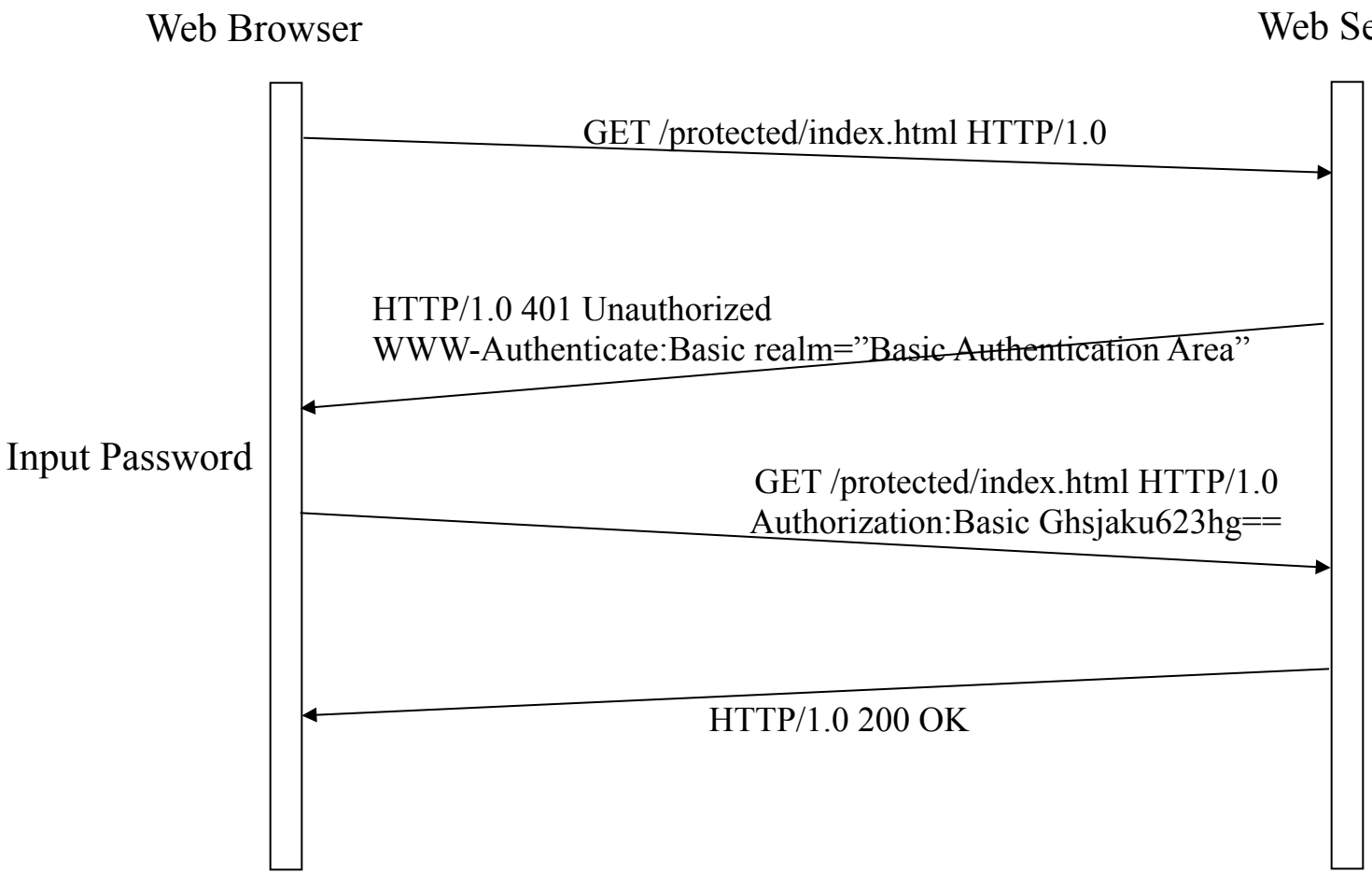
Receiver's private key

decrypt

HGFu76JH %&7bvc

ciphertext

Authentication (HTTP Basic)



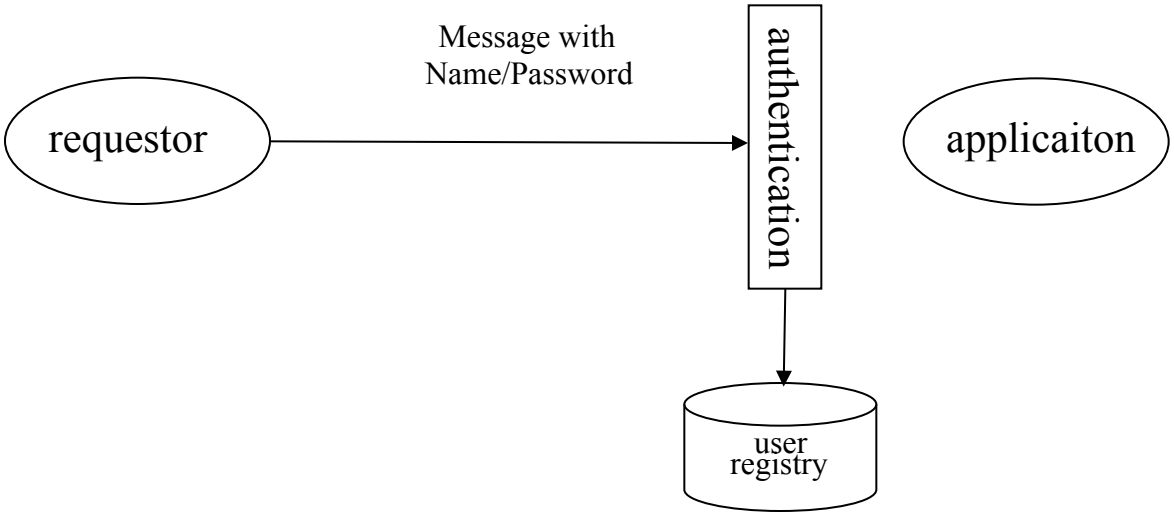
Secure Sockes Layer (SSL)

- Was not designed specifically for securing transactions
- Secures WWW connections (between two computers in the Internet or Web)
- Implements public key cryptography and digital certificates to authenticate server
- Doesn’t require client authentication

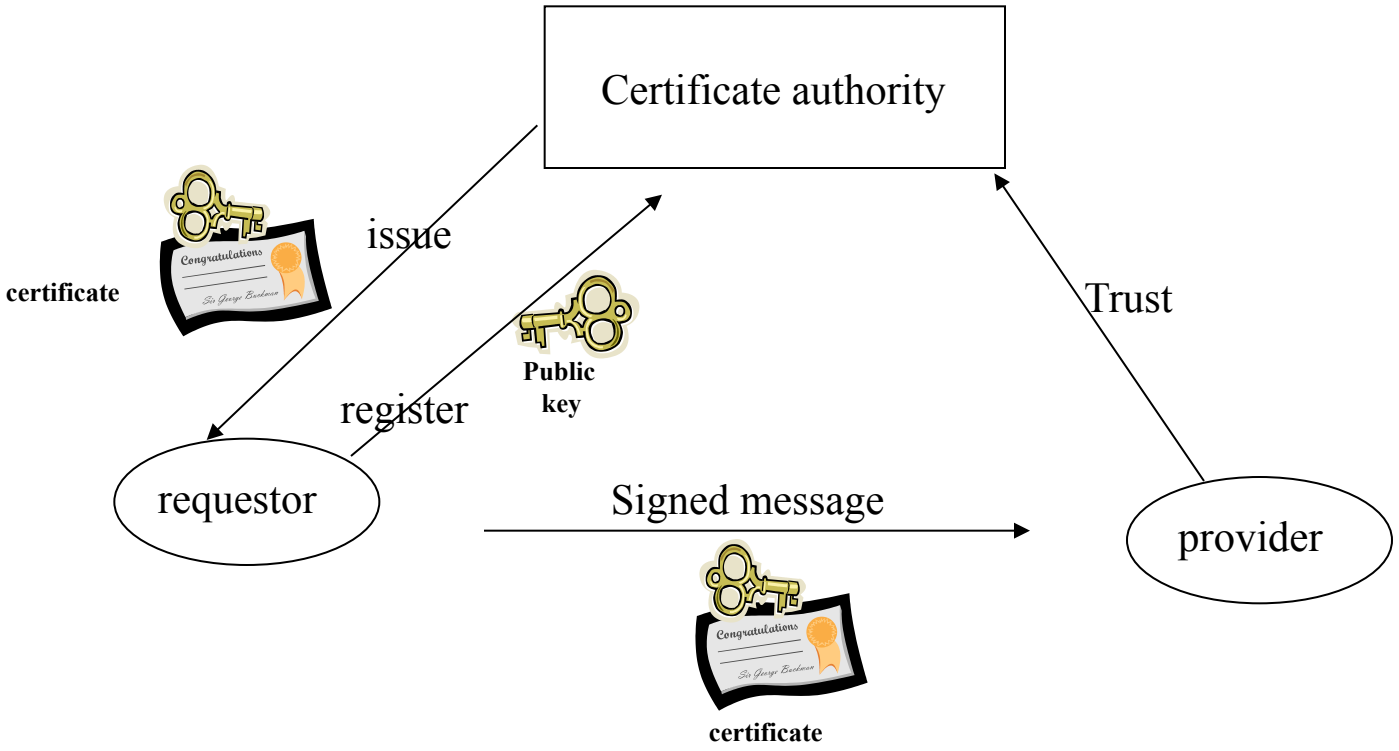
SSL (general)

- The client accesses server
- The server returns its certificate
- The client generates a random number (seed) encrypts the seed with server's public key from the certificate and send it to the server
- The server decrypts the data and extracts the seed
- The server and the client have the same secret number to generate a symmetric key

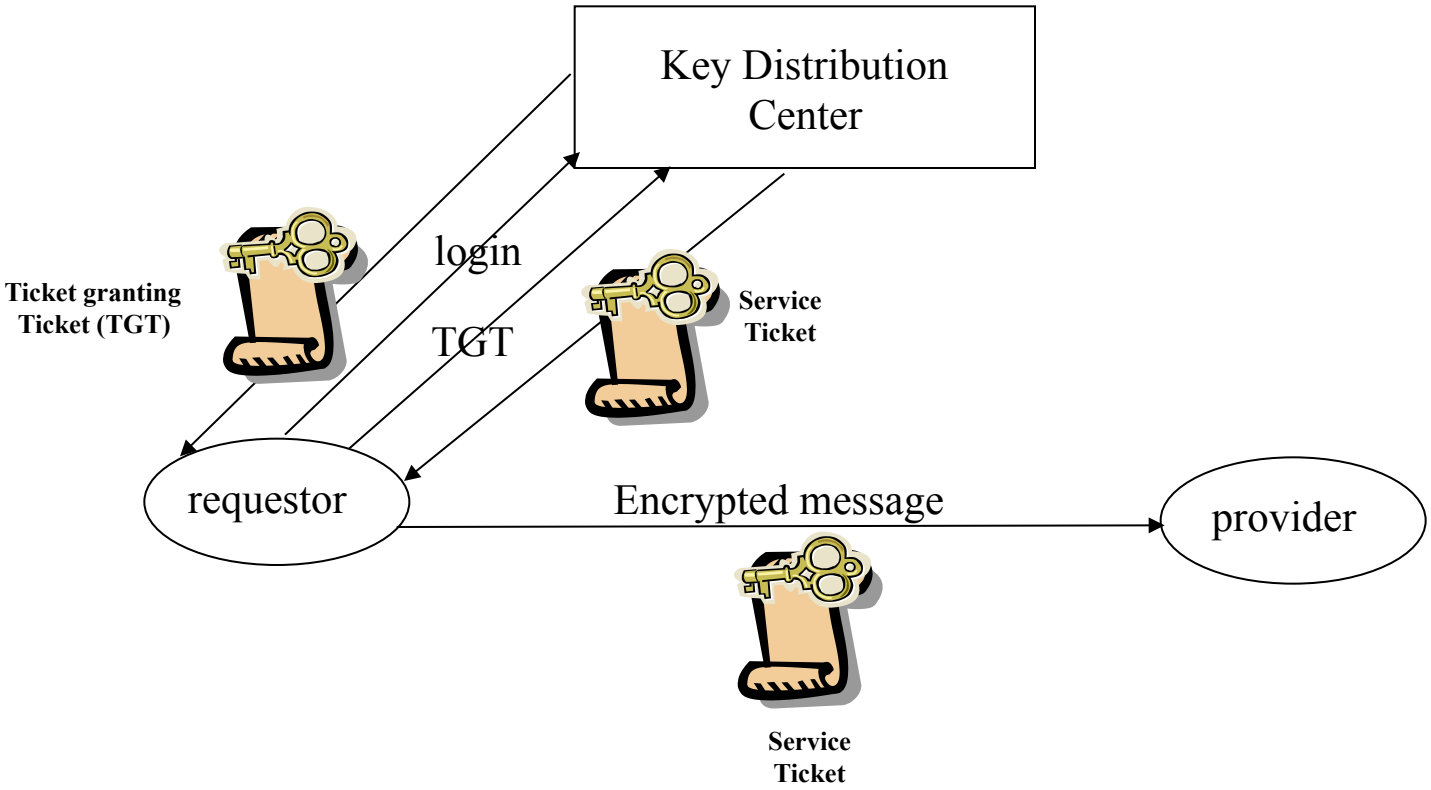
Security infrastructures (registries)



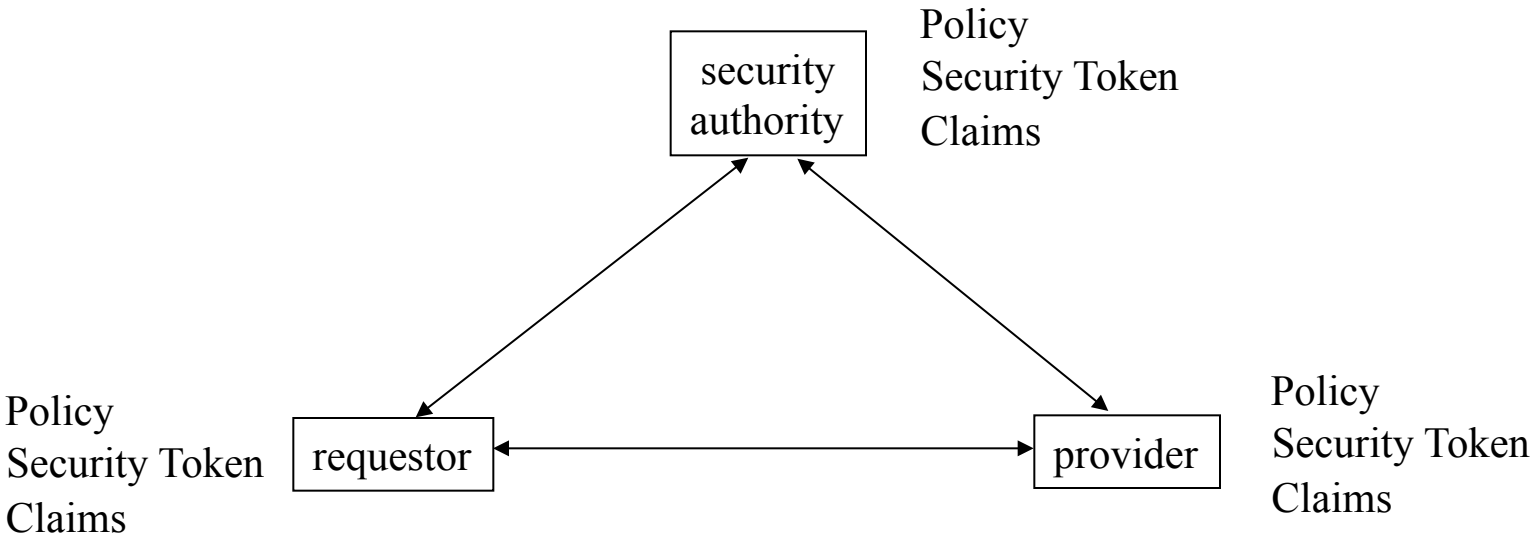
Security infrastructures (Public Key Infrastructure - PKI)



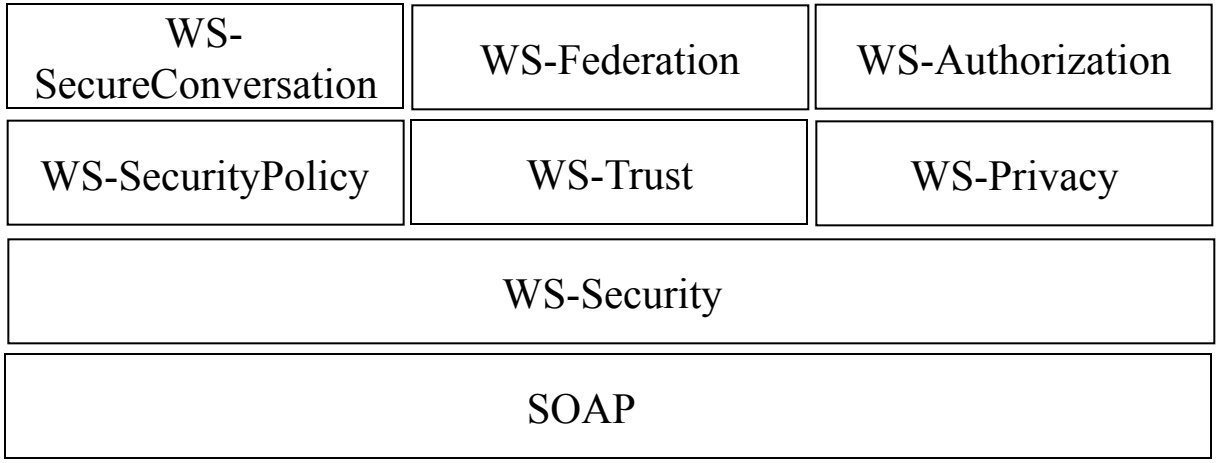
Kerberos


Key Distribution Center
login
TGT
Ticket granting Ticket (TGT)
Service Ticket
requestor
Encrypted message
provider
Service Ticket

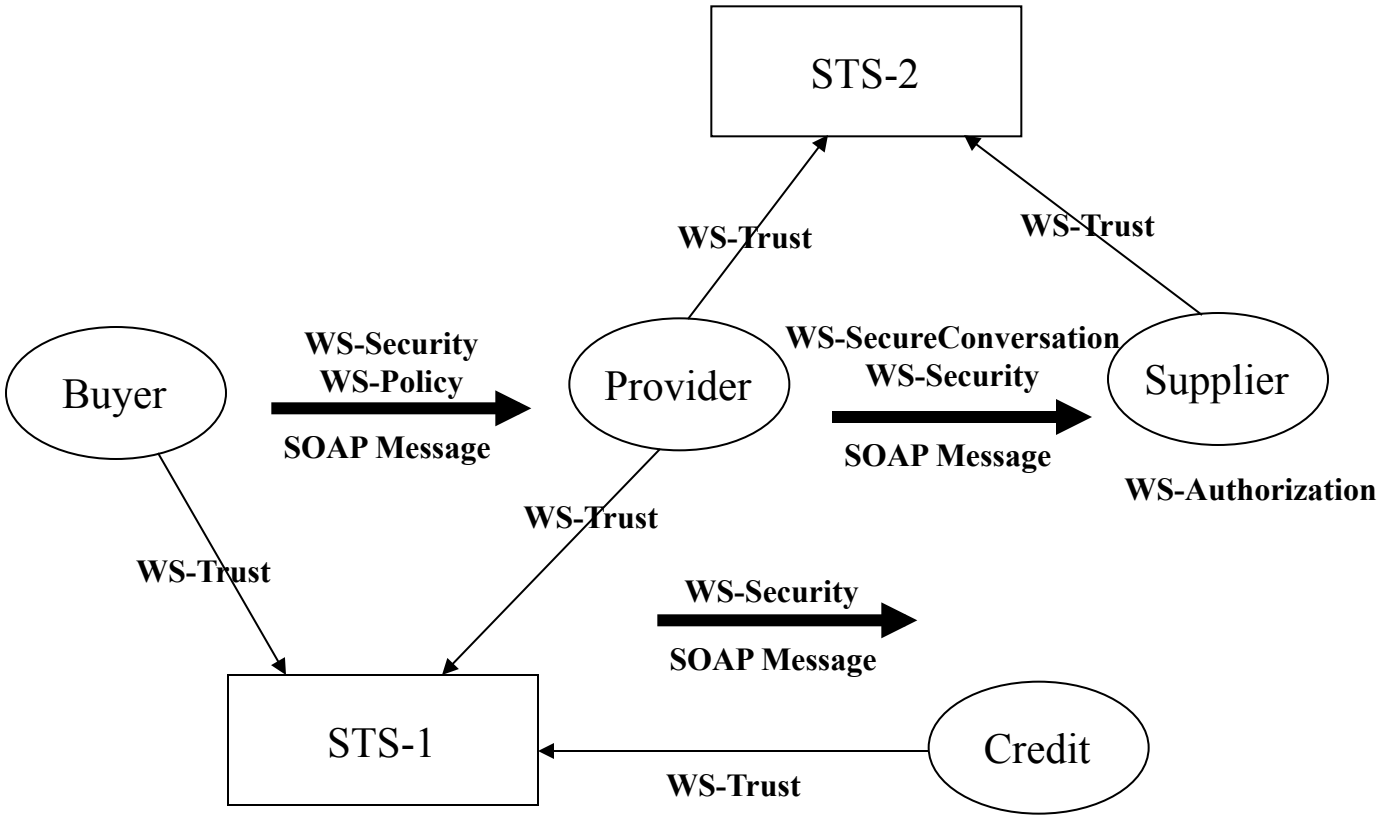
Web Services Security Model



Web Services Security Specifications



Scenario


STS-2
WS-Trust
WS-Trust
Buyer
WS-Security
WS-Policy
SOAP Message
Provider
WS-SecureConversation
WS-Security
SOAP Message
Supplier
WS-Authorization
WS-Trust
WS-Trust
WS-Security
SOAP Message
STS-1
Credit
WS-Trust

WS-Security

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Header>
    <wsse:Security
              xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext">
         <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
         </Signature>
         <EncryptedKey xmlns="http://www.w3.org/2001/04/enc-enc-enc#">
         </EncryptedKey>
         <wsse:UsernameToken
              xmlns="http://schemas.xmlsoap.org/ws/2003/06/secext">
         </wsse:UsernameToken>
    </wsse:Security>
   </S:Header>
        <S:Body>
        . . .
        </S:Body>
   </S:Header>
```

WS-Security (Digital signatures)

```
<SOAP-ENV:Envelope . . .
  <SOAP-ENV:Header . . .
    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext">
      <wsse:BinarySecurityToken wsu:Id="bst_id" …>HJKJH… </wsse:BinarySecurityToken>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
         <SignedInfo>
              <CanonicalizationMethod Algorithm="www.w3.org/2001/10/xml-exc-c14n#">
              </CanonicalizationMethod>
              <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1">
              </SignatureMethod>
              <Reference URI="#body_id">
                   <Transforms>
                        <Transform Algorithm="www.w3.org/2001/10/xml-exc-c14n#"></Transform>
                   </Transforms>
                   <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1">
                   </DigestMethod>
                   <DigestValue>EWUOYOW7470KHL428LAHHLKHALAHLDH=…</DigestValue>
              </Reference>
         </SignedInfo>
         <SignatureValue>WOYU8SLLJK341089DLJB,ALHLAHDLKAHFLAH8P9+09+9+LWÖJSBLGAD
                              DASLFHLKDKDJKLADHUQYEMwresfdbytidcfhfBZXBOISFH=…
         </SignatureValue>
         <KeyInfo>
              <wsse:SecurityTokenReference>
                   <wsse:Reference URI="#bst_id"></wsse:Reference>
              </wsse:SecurityTokenReference>
         </KeyInfo>
      </Signature>
    </wsse:Security>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility"wsu:Id="body_id">
          <po . . .
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Canonicalization

- <?xml version="1.0" encoding="ascii"?>
  <example
  r="b"
  f="a"/>
- <?xml version="1.0" encoding="ascii"?>
  <example r="b" f="a"></example>

- <?xml version="1.0" encoding="ascii"?>
  <example f="a" r="b"></example>

Encryption (symmetric key)

```
<po xmlns="http://www.skatestown.com/ns/po-with-card"
       id="37465" submitted="2004-09-23" customerId="678763">
 <enc:EncryptedData
       Type="http://www.w3.org/2001/04/xmlenc#Element">
   <enc:EncryptedMethod
     Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
   <ds:KeyInfo>
       <ds:KeyName>Shared key</ds:KeyName>
   </ds:KeyInfo>
   <enc:CipherData>qewuoiyflhYUOihjkchl…</enc:CipherData>
 </enc:EncryptedData>
 <shipTo>
   . . .
</po>
```

Encryption (public key)

```
<po xmlns="http://www.skatestown.com/ns/po-with-card"
     id="37465" submitted="2004-09-23" customerId="678763">
   <enc:EncryptedData
        Type="http://www.w3.org/2001/04/xmlenc#Element">
     <enc:EncryptedMethod
       Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
     <ds:KeyInfo>
         <enc:EncryptedKey>
            <enc:EncryptionMethod
              Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
            <ds:KeyInfo>
                <ds:KeyName>Receiver's key</ds:KeyName>
            </ds:KeyInfo>
            <enc:CipherData>uyoiwahbjYTutayuthl…</enc:CipherData>
         </enc:EncryptedKey>
     </ds:KeyInfo>
     <enc:CipherData>qewuoiyflhYUOihjkchl…</enc:CipherData>
   </enc:EncryptedData>
   <shipTo>
   . . .
</po>
```

WS-Security Encryption

```
<env:Envelope . . .
  <env:Header . . .
    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext" . . .
                env:mustUnderstand="1">
      <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
        <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
        <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
             <wsse:SecurityTokenReference>
                 <wsse:KeyIdentifier>wuirop7892hhlhWTYUE=</wsse:KeyIdentifier>
             </wsse:SecurityTokenReference>
        </KeyInfo>
        <CipherData>
            <CipherValue>adhkl…</CipherValue>
        </CipherData>
        <ReferenceList>
          <DataReferenceURI="#wssecurity_Encryption_id_1269341436796964961_43267926598265">
          </DataReference>
        </ReferenceList>
      </EncryptedKey>
    </wsse:Security>
  </env:Header>
  <env:Body>
    <po xmlns="http://www.skatestown.com/ns/po-with-card" id="37465"
                  submitted="2004-09-23" customerId="678763">
      <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#"
                  Id="="#wssecurity_Encryption_id_1269341436796964961_43267926598265"
                  Type="http://www.w3.org/2001/04/xmlenc#Content">
        <EncryptedMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
        <CipherData>
                  <CipherValue>HJeyoigjkga…</CipherValue>
        </CipherData>
      </enc:EncryptedData>
      <shipTo>
    . . .
```

Security Tokens

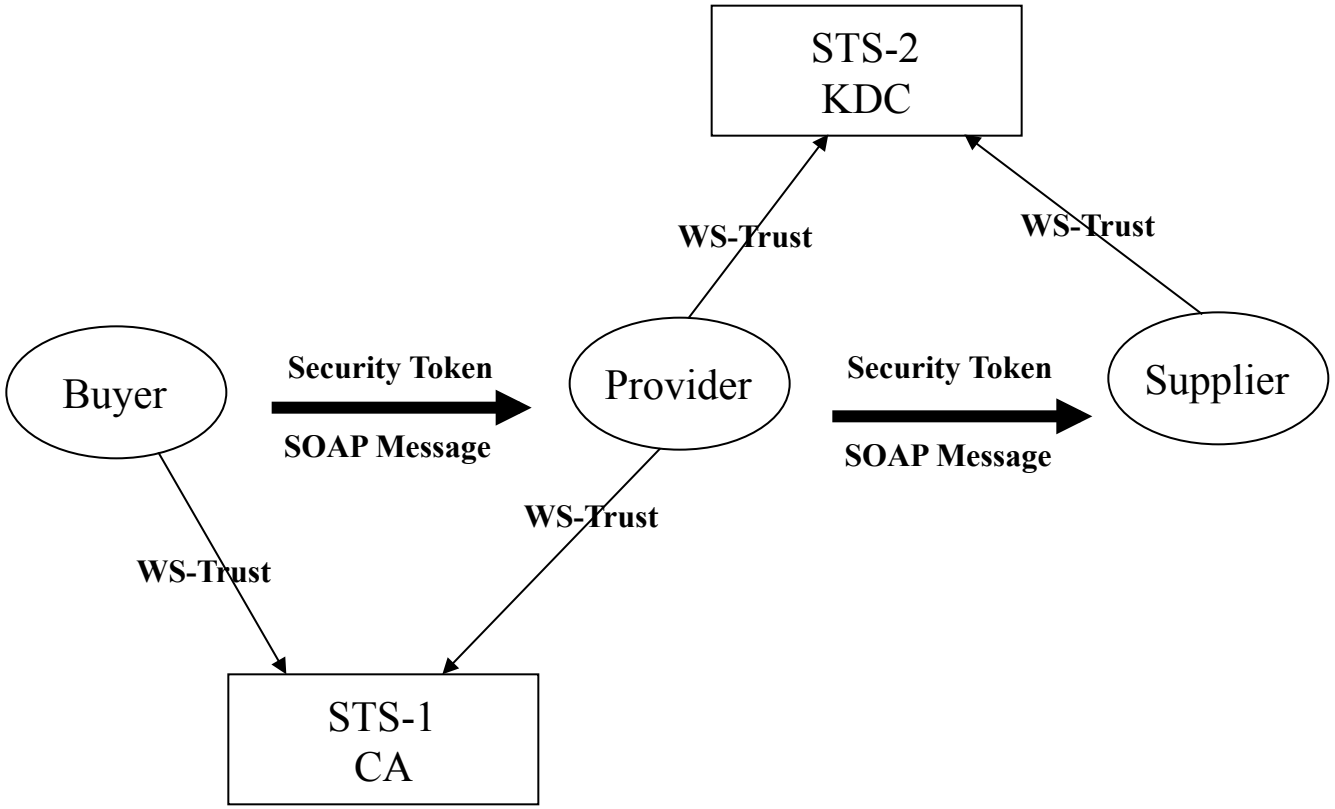
- UsernameToken

```
<wsse:UsernameToken>
   <wsse:Username>testName</wsse:Username>
   <wsse:Password>testPassword</wsse:Password>
</wsse:UsernameToken>
```

- BinarySecurityToken

```
<wsse:BinarySecurityToken xmlns:wsu="http://
   schemas.xmlsoap.org/ws/2003/06/utility"
   EncodingType="wsse:Base64Binary" ValueType="wsse:X509v3"
   wsu:Id="wssecurity_binary_security_token_id_159786294759126
   59_3245710743103740">
         RWYOLSHL472Q8097ALH7FX89hljkhryiiuljklh. . .
</wsse:BinarySecurityToken>
```

WS-Trust


STS-2
KDC
WS-Trust
WS-Trust
Buyer
Security Token
SOAP Message
Provider
Security Token
SOAP Message
Supplier
WS-Trust
WS-Trust
STS-1
CA

WS-Trust (request)

```
<env:Envelope . . .
  <env:Header wsu:Id="req">
    <wsse:Security>
        <wsse:UsernameToken wsu:Id="Me">
            <wsse:Username>Tom</wsse:Username>
            <wsse:Password>Buyer</wsse:Password>
        </wsse:UsernameToken>
    </wsse:Security>
  </env:Header>
  <env:Body>
    <wst:RequestSecurityToken>
        <wst:TokenType>wsse:X509v3</wst:TokenType>
        <wst:RequestType>wsse:ReqIssue</wst:RequestType>
        <wst:Base>
             <wsse:Reference URI="#Me ValueType="wsse:UsernameToken"/>
        </wst:Base>
        <wsp:AppliesTo
              xmlns:wsp="http://schemas.xmlsoap.org/ws/2002/12/policy">
           <wsa:EndpointReference>
               <wsa:Address>http://credit.com/service</wsa:Address>
           </wsa:EndpointReference>
        </wsp:AppliesTo>
    <wst:RequestSecurityToken>
  </env:Body>
</env:Envelope>
```

WS-Trust (response)

```
<env:Envelope . . .
  . . .
  <env:Body wsu:Id="req">
    <RequestSecurityTokenResponse>
      <RequestedSecurityToken>
        <BinarySecurityToken ValueType="wsse:X509v3"
                             EncodingType="wsse:Base64Binary">
                               TEIKJJGKJDTYVN623862gjsggaGJ…
        </BinarySecurityToken>
      </RequestedSecurityToken>
    </RequestSecurityTokenResponse>
  </env:Body>
</env:Envelope>
```

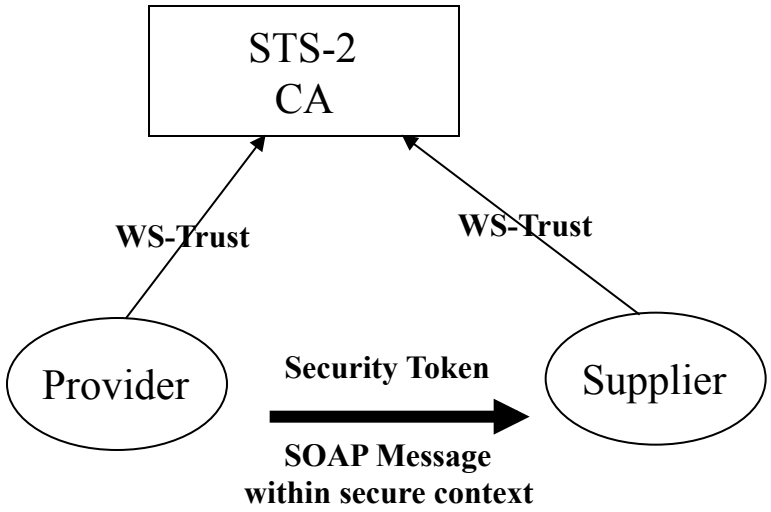
WS-Trust (request kerberos)

```
<SOAP-ENV:Envelope . . .
  <SOAP-ENV:Header wsu:Id="req">
    <wsse:Security>
       <wsse:UsernameToken wsu:Id="Me">
         <wsse:Username>Tom</wsse:Username>
         <wsse:Password>Buyer</wsse:Password>
       </wsse:UsernameToken>
    </wsse:Security>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <RequestSecurityToken>
       <TokenType>wsse:Kerberos5TGT</TokenType>
       <RequestType>wsse:ReqIssue</wst:RequestType>
       <Base>
          <Reference URI="#Me" ValueType="wsse:UsernameToken"/>
       </wst:Base>
    </RequestSecurityToken>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

WS-SecurityPolicy

```
<wsp:Policy xmlns:wsp=… xmlns:wsse=…>
  <wsp:All wsp:Preference="100">
     <wsse:Integrity wsp:Usage="wsp:Required">
        <wsse:Algorithm Type="wsse:AlgCanonicalization"
              URI="http//www.w3.org/Signature/Drafts/xml-exc-c14n"/>
        <wsse:Algorithm Type="wsse:AlgSignature
              URI="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
        <MessageParts
           Dialect="http://schemas.xmlsoap.org/2002/12/wsse#soap">
              S:Body
        </MessageParts>
     </wsse:Integrity>
     <wsse:SecurityToken>
         <wsse:TokenType>wsse:X509v3</wsse:TokenType>
     </wsse:SecurityToken>
   </wsp:All>
</wsp:Policy>
```

WS-SecureConversation



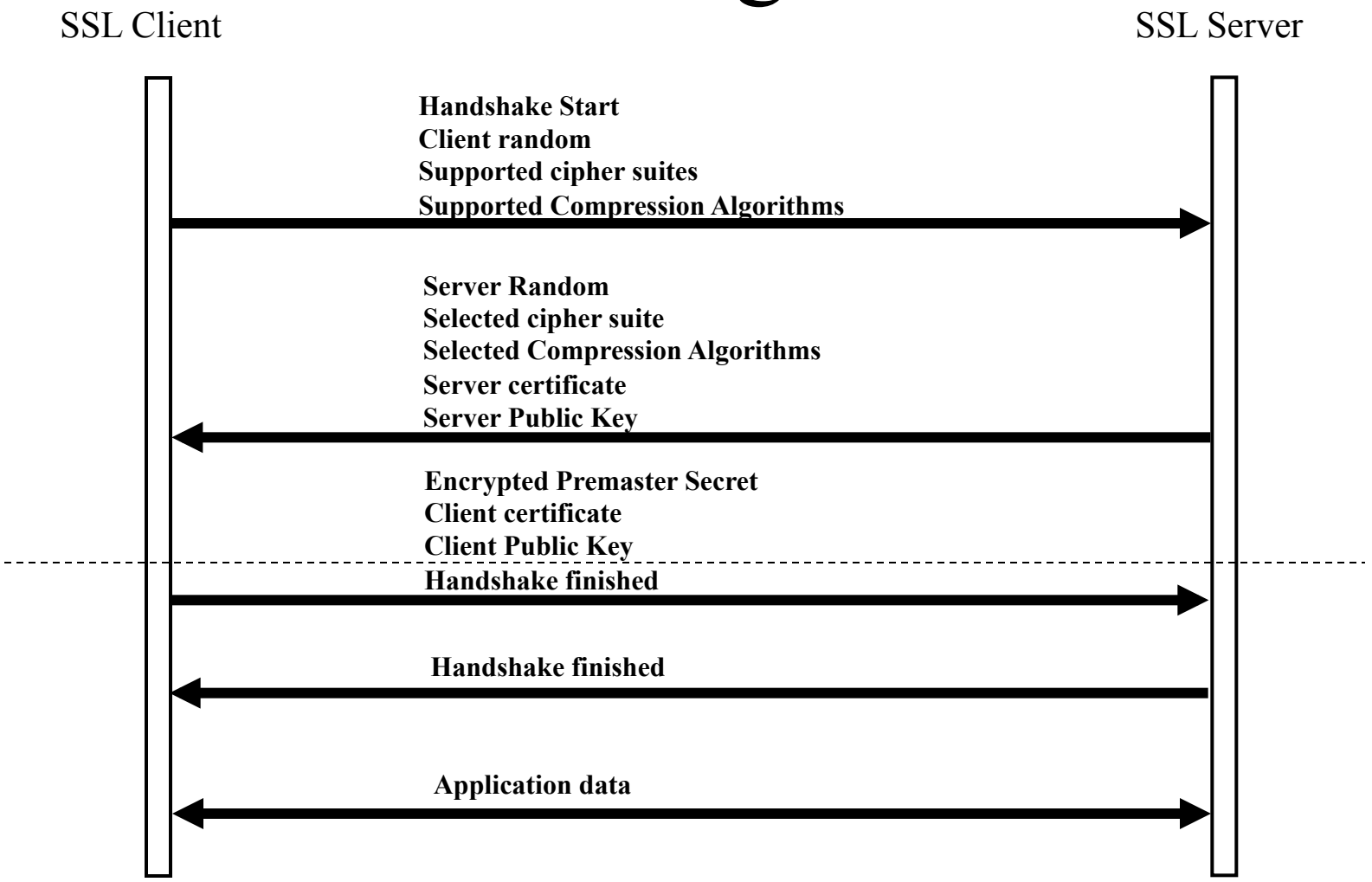
WS-SecureConversation

```
<env:Envelope . . .
 <env:Header . . .
  <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext"
   env:mustUnderstand="1">
   <wsse:SecurityContextToken wsu:Id="SecContext">
      <wsu:Identifier>uuid: . . . </wsu:Identifier>
   </wsse:SecurityContextToken>
   <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
     <SignedInfo>
       <CanonicalizationMethod Algorithm="www.w3.org/2001/10/xml-exc-c14n#"/>
       <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-shal"/>
       <Reference
           URI="#wssecurity_binary_security_token_id_15978629475912659_3245710743103740">
        <Transforms>
          <Transform Algorithm="www.w3.org/2001/10/xml-exc-c14n#"></Transform>
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#shal"></DigestMethod>
        <DigestValue>asJHuoyuLHLyuIOLUYIO=</DigestValue>
     </Reference>
    </SignedInfo>
    <SignatureValue>TYUITUYuipuoiYUIOYOIYIO…=</SignatureValue>
    <KeyInfo>
     <wsse:SecurityTokenReference>
       <wsse:Reference URI="#SecContext"/>
       </wsse:SecurityTokenReference>
    </KeyInfo>
   </Signature>
  </wsse:Security>
 </env:Header>
 <env:Body xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility"
   wsu:Id="wssecurity_binary_security_token_id_15978629475912659_3245710743103740">
    <orderSupplies . . .
 </env:Body>
</env:Envelope>
```

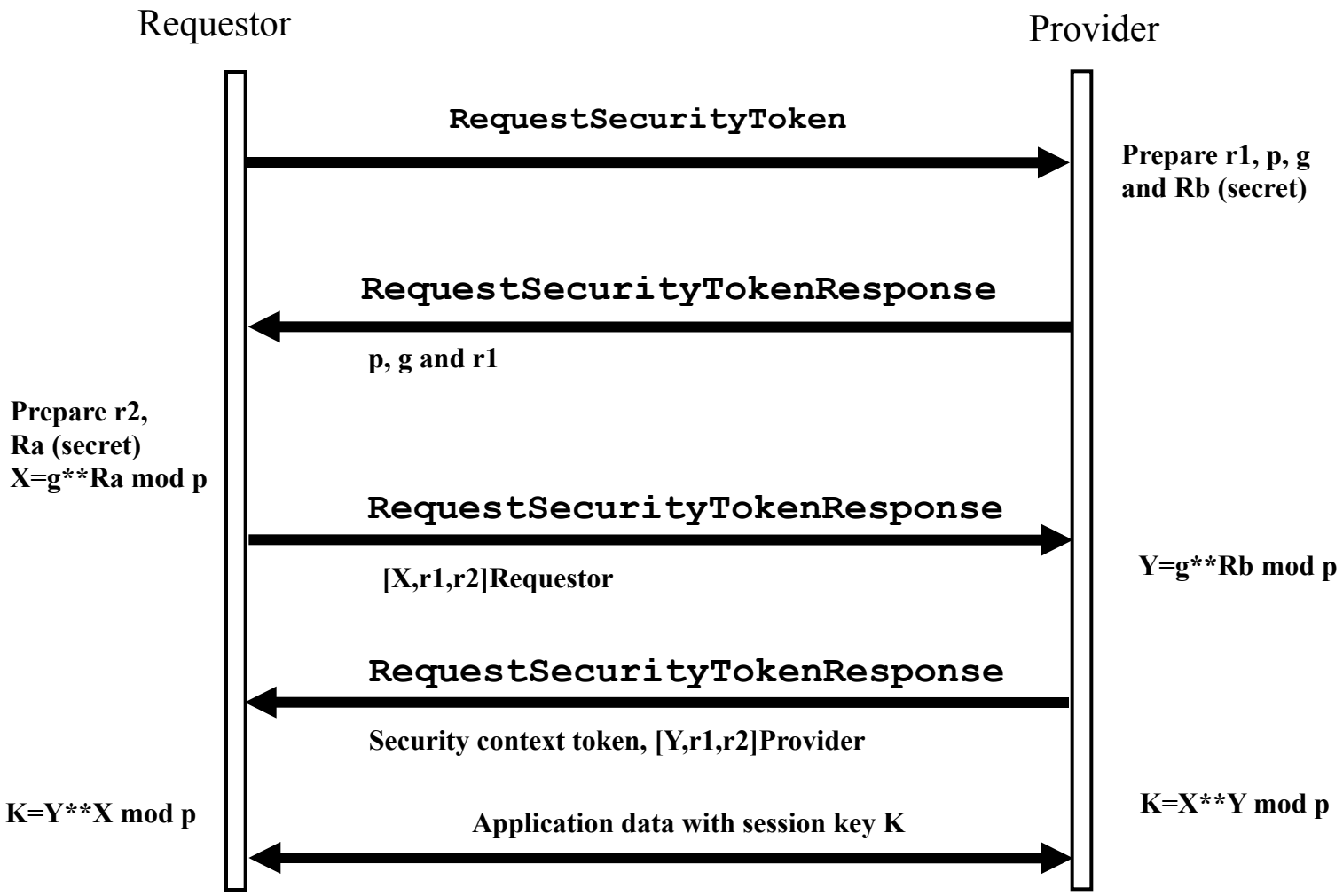
WS-SecureConversation

```
<env:Envelope . . .
 <env:Header . . .
  <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext"
                              env:mustUnderstand="1">
   <wsse:SecurityContextToken wsu:Id="SecContext">
      <wsu:Identifier>uuid: . . . </wsu:Identifier>
   </wsse:SecurityContextToken>
   <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
     <SignedInfo>...</SignedInfo>
     <SignatureValue>TYUITUYuipuoiYUIOYOIYIO…=</SignatureValue>
     <KeyInfo>
        <wsse:DerivedKeyToken wsse:Algorithm="wsse:PSHA1">
           <wsse:SecurityTokenReference>
              <wsse:Reference URI="#SecContext"></wsse:Reference>
           </wsse:SecurityTokenReference>
           <Properties>
              <Label>NewLabel</Label>
              <Nonce>FEFH…</Nonce>
           </Properties>
           <wsse:Generation>2</wsse:Generation>
       </wsse:DerivedKeyToken>
     </KeyInfo>
   </Signature>
  </wsse:Security>
 </env:Header>
 <env:Body xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility"
   wsu:Id="wssecurity_binary_security_token_id_1597862947591265 9_3245710743103740">
    <orderSupplies . . .
 </env:Body>
</env:Envelope>
```

SSL again


SSL Client
SSL Server
Handshake Start
Client random
Supported cipher suites
Supported Compression Algorithms
Server Random
Selected cipher suite
Selected Compression Algorithms
Server certificate
Server Public Key
Encrypted Premaster Secret
Client certificate
Client Public Key
Handshake finished
Handshake finished
Application data

Security handshake protocol (example)



Security handshake protocol messages(example)

1.
```
<env:Envelope xmlns:S="http://schenas.xmlsoap.org/soap/envelope/">
  <env:Body xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility">
      <wst:RequestSecurityToken xmlns:wst=
                    "http//schemas.xmlsoap.org/ws/2002/12/secext">
          <wst:TokenType>wsse:SecurityContextToken</wst:TokenType>
          <wst:RequestType>wsse:ReqIssue</wst:RequestType>
      </wst:RequestSecurityToken>
  </env:Body>
</env:Envelope>
```

2.
```
<env:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
      <wst:RequestSecurityTokenResponse xmlns:wst=
                    "http//schemas.xmlsoap.org/ws/2002/12/secext">
          <wst:SignChallenge>
              <wst:Challenge>
                <val:ExValue xmlns:val="http://dumml.org/exvalue">
                   <val:P>toiudgduoP…</val:P>
                   <val:G>hoiusrewrwr…</val:GP>
                   <val:R1>aoisgswhOuoP…</val:R1>
                </val:ExValue>
           </wst:Challenge>
          </wst:SignChallenge>
      </wst:RequestSecurityTokenRespons>
  </env:Body>
</env:Envelope>
```

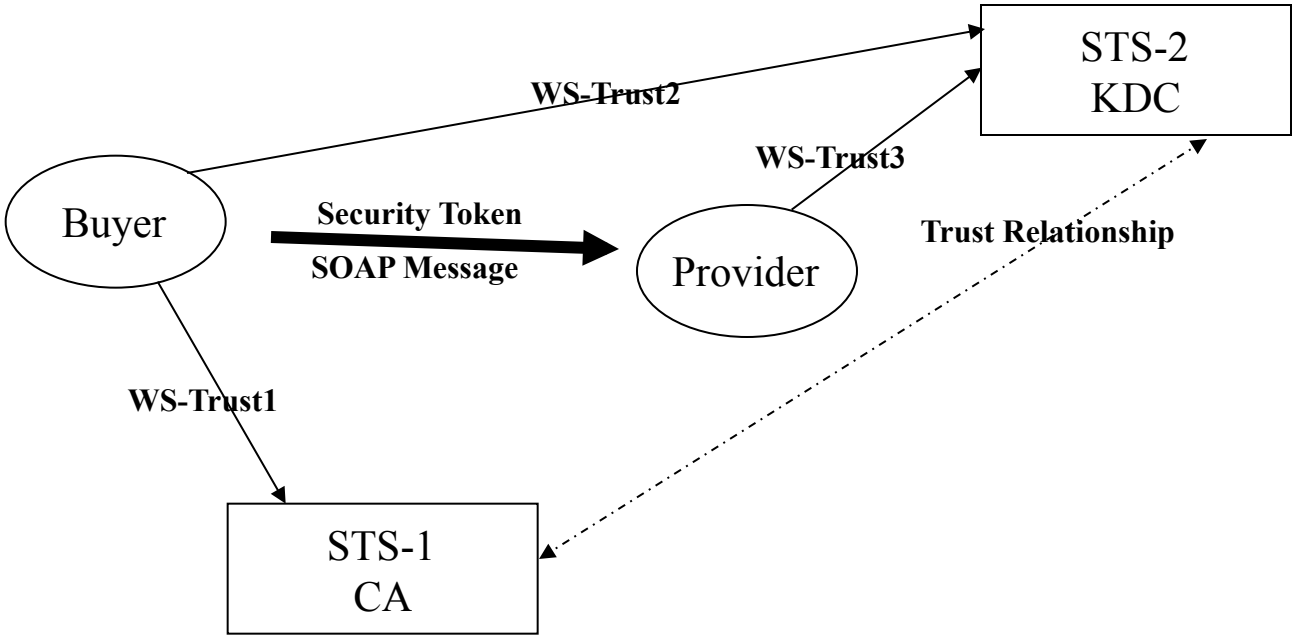
Security handshake protocol (example)

```
3.<env:Envelope . . .
    <env:Header . . .
    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext"
                                env:mustUnderstand="1">
      <wsse:BinarySecurityToken ws:Id="wssecurity_binary_security_token_id_15978629475912659_3245710743103740">
               saghkjHKHGKuyyiu. . .
      </wsse:BinarySecurityToken>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <sig:SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
          <sig:CanonicalizationMethod Algorithm="www.w3.org/2001/10/xml-exc-c14n#"/>
          <sig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-shal"/>
          <sig:Reference URI="#BODY">
            <sig:Transforms>
              <Transform Algorithm="www.w3.org/2001/10/xml-exc-c14n#"/>
            </sig:Transforms>
            <sig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#shal"/>
            <sig:DigestValue>asJHuoyuLHLyuIOLUYIO=</sig:DigestValue>
          </sig:Reference>
        </sig:SignedInfo>
        <SignatureValue>TYUITUYuipuoiYUIOYOIYIO…=</SignatureValue>
          <KeyInfo>
            <wsse:SecurityTokenReference>
              <wsse:Reference URI="#wssecurity_binary_security_token_id_15978629475912659_3245710743103740">
              </wsse:Reference>
            </wsse:SecurityTokenReference>
          </KeyInfo>
        </Signature>
    </wsse:Security>
    </env:Header>
    <env:Body wsu:Id="BODY">
      <wst:RequestSecurityTokenResponse xmlns:wst="http//schemas.xmlsoap.org/ws/2002/12/secext">
          <wst:SignChallengeResponse>
               <val:ExValue xmlns:val="http://dumml.org/exvalue">
                    <val:X>UoiushOuoP…</val:P>
                    <val:R2>tyiutiutiu…</val:R2>
                    <val:R1>UoiushOuoP…</val:R1>
               </val:ExValue>
          </wst:SignChallengeResponse>
     </wst:RequestSecurityTokenRespons>
  </env:Body>
</env:Envelope>
```

Security handshake protocol (example)

```
4.<env:Envelope . . .
   <env:Header . . .
    <wsse:Security env:mustUnderstand="1">
          xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext">
      <wsse:BinarySecurityToken ws:Id=. . . >
             . . .
             <sig:Reference URI="#BODY">
             . . .
   <env:Body wsu:Id="BODY" . . .>
      <wst:RequestedSecurityTokenResponse>
         <wsse:SecurityContextToken xlns:wsse=
                    "http://schemas.xmlsoap.org/ws/2003/06/secext">
              <wsu:Identifier>uuid:…</wsu:Identifier>
          </wsse:SecurityContextToken>
          <val:ExValue xmlns:val="http://dumml.org/exvalue">
             <val:Y>iyjzTYUijh…</val:Y>
             <val:R2>tyiutiutiu…</val:R2>
             <val:R1>UoiushOuoP…</val:R1>
          </val:ExValue>
       </wst:RequestedSecurityToken>
      </wst:RequestSecurityTokenResponse>
    </env:Body>
  </env:Envelope>
```

WS-Federation



WS-Federation

```
<env:Envelope . . .
 <env:Header . . .
  <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2003/06/secext" SOAP-ENV:mustUnderstand="1">
    <wsse:BinarySecurityToken xmlns:wsse="schemas.xmlsoap.org/ws/2003/06/utility"
           ValueType="wsse:X509v3" EncodingType="wsse:Base64Binary" ws:Id="myToken">
               saghkjHKHGKuyyiu. . .
    </wsse:BinarySecurityToken>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
          <CanonicalizationMethod Algorithm="www.w3.org/2001/10/xml-exc-c14n#"/>
          <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"/>
          <Reference URI="#body_id">
            <Transforms>
              <Transform Algorithm="www.w3.org/2001/10/xml-exc-c14n#"></Transform>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
            <DigestValue>asJHuoyuLHLyuIOLUYIO=</DigestValue>
          </Reference>
        </SignedInfo>
        <SignatureValue>TYUITUYuipuoiYUIOYOIYIO…=</SignatureValue>
          <KeyInfo>
            <wsse:SecurityTokenReference>
              <wsse:Reference URI="#myToken"></wsse:Reference>
            </wsse:SecurityTokenReference>
          </KeyInfo>
      </Signature>
    </wsse:Security>
  </env:Header>
  <env:Body wsu:Id="body_id">
    <RequestSequrityToken>
      <TokenType>wsse:Kerberos5ST</TokenType>
      <RequestType>wsse:ReqIssue</RequestType>
      <Base>
          <Reference URI="#myToken"/>
      </Base>
      <wsp:AppliesTo xmlns:wsp="http://schemas.xmlsoap.org/ws/2002/12/policy">
          <wsa:EndpointReference>
              <wsa:Address>http://skatestown.com/service</wsa:Address>
          </wsa:EndpointReference>
    . . .
```

WS-Federation

```
<wsp:Policy>
    <wsse:RelatedService wsse:ServiceType="wsse:ServiceSTS">
        <wsa:EndpointReference>
            <wsa:Address>
                http://www.sts_2.com/tempIdentity
            </wsa:Address>
        </wsa:EndpointReference>
    </wsse:RelatedService>
</wsp:Policy>
```

Next lecture

- Service Coordination

Text-book **Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI, 2nd Edition**

Chapter 12