95-702 Distributed Systems
Lecture 15
Securing Web Services
XML Web Services

Hot topic
Foundation of Service Oriented Architectures
Interoperable
Remote Method Invocation
Messaging
Supported by all the big players

Notes adapted from the required reading “Web Services Security”, Bilal Siddiqui
Existing XML Web Services

Google
EBay
Amazon
XIgnite (financial computations)
Many others in the cloud!
See www.xmethods.com
But remember, many are not public.
An SOA would have many in house web services.
XML Web Services & Cryptography

Bob and Alice want to exchange SOAP messages.
Eve and Mallory need to be taken seriously.
What’s going on?

Web Services Security (WSS) specification from OASIS.
Adds message confidentiality to SOAP.
Adds message Identification, authentication, authorization, and non-repudiation to SOAP.
Why not simply use SSL?
SSL is cool but point to point.
An end-to-end approach carries the encrypted data and signatures and permits persistence.
SSL may be used along with WSS.
The WS Cryptography Stack

XML Web Services Security
SAML (Security Assertion ML), XKMS (XML Key Management Specification),
XACML (eXtensible Access Control Markup Language)

XMLDSIG (W3C)
XMLENC (W3C)

.NET Crypto API’s
Java Security API’s
Interoperable Web Services

We need application integration within the enterprise.
We need application integration across enterprise boundaries:
  customers
  partners
  suppliers
A Service Oriented Architecture may be built on a web service foundation, using services within the enterprise and in the cloud.
A Tourism Supply Chain

Tourists ← Tour Operator → Car Rental → Hotel → Hotel

RoomRentInfoForAll()

RoomRentInfoForPartnersOnly()

Anyone may call

Restricted callers

95-702 Distributed Systems
Master of Information System Management
Service Oriented Architecture

SOAP over HTTP

SOAP Server

Hotel

RoomRentInfoForAll()

RoomRentInfoForPartnersOnly()
Listing 1 SOAP Request

POST /Vendors HTTP/1.1
Host: www.myHotel.com
Content-Type: text/xml;Charset=utf-8
Content-Length: 350
SOAPACtion:"

Listing 2 SOAP Response

HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: 1474

<?xml version="1.0">
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV='http://schemas.xmlsoap.org/soap/envelope/' >
<SOAP-ENV:Body>
    <m:GetSpecialDiscountedBookingForPartnersResponse
        xmlns:m="http://www.MyHotel.com/partnerservice/" >
    <!-- Booking confirmation details-->
    </m:GetSpecialDiscountedBookingForPartnersResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
1st Generation Web Services

SOAP Client

SOAP Server

Hotel Class

RDBMS
2ND Generation Web Services

Diagram:

- SOAP Client
- SOAP Server
- Hotel Class
- RDBMS
- Tour Planning Class

Network connections between SOAP Client and SOAP Server, Hotel Class, RDBMS, and Tour Planning Class.
3rd Generation Web Services

WS-Transaction

SOAP Server  Plane Class  RDBMS

SOAP Server  Hotel Class  RDBMS

SOAP Client

SOAP Server  Tour Planning Class

95-702 Distributed Systems
Master of Information System Management
WS Security

SOAP Server (SOAP Aware Firewall)
- inspect SOAP message
- match user roles with access lists
- XML Signature
- XML Encryption
- WSS (SOAP specific use of XMLEnc and XMLDsig)
- Security Access Markup Language (SAML) for single sign on replacing HTTP cookies
- XACML (extensible Access Control Markup Language) to express authorization and access policies
XML Signature
An IETF/W3C Recommendation
XML Digital Signatures

Quick Review

Message Digest
message + digest algorithm -> hash value
transmit (message,hash value) pair
useful for checking if errors occurred

Problem
Mallory might replace the message, hash value pair with her own message, hash value pair.
XML Digital Signatures

- Solution (1): get a symmetric key involved in the calculation of the hash.
- Solution (2): Given a message m, compute a hash of m and encrypt the hash with an asymmetric private key.
- Mallory doesn’t know the keys. So, she can’t forge the signature.
- But how do we do this in XML?
XML Signature

- XML Signatures are digital signatures used in XML transactions
- May be used to sign only a portion of an XML document. The document might have a long history with different parts holding different signatures
- The signature may apply to XML or non-XML data
Referencing What is Signed

• The XML Signature may hold a URI.
• This allows to point to a signed entity that may reside elsewhere.
• Or, the signed content may be available in the XML document holding the signature.
XMLDSig General Form

Each resource to be signed has its own `<Reference>` element, identified by the URI attribute.

The `<Transform>` element specifies an ordered list of processing steps that were applied to the referenced resource's content before it was digested.

The `<DigestValue>` element carries the value of the digest of the referenced resource.

The `<SignatureValue>` element carries the value of the encrypted digest of the `<SignedInfo>` element.

The `<KeyInfo>` element indicates the key to be used to validate the signature. Possible forms for identification include certificates, key names, and key agreement algorithms and information.
The `<Reference>` Element

- Each signed resource is specified with a `<Reference>` element.
- A typical `<Reference>` element will contain:
  - a pointer to what is signed
  - a digest method (for example SHA1)
  - and a digest value of the signed data in base 64 notation
The `<Reference>` Element

This is the location of the document being signed.

```
<Reference URI = "http://…/po.xml">
  <DigestMethod>….</DigestMethod>
  <DigestValue> calculated digest of po.xml</DigestValue>
</Reference>
```
We may have many references

<Reference>
    pointer, digest method, digest value
</Reference>

<Reference>
    pointer, digest method, digest value
</Reference>

<Reference>
    pointer, digest method, digest value
</Reference>
Place Within a SignedInfo Element

```xml
<SignedInfo>
  <CanonicalizationMethod> algorithm used on SignedInfo element
  <SignatureMethod> for example dsa-sha1
    <Reference>
      pointer, digest method, digest value
    </Reference>
    <Reference>
      pointer, digest method, digest value
    </Reference>
</SignedInfo>
```
Compute Digest of SignedInfo

<SignedInfo>
  <CanonicalizationMethod> algorithm used on SignedInfo element
  <SignatureMethod> for example dsa-sha1
  <Reference>
    pointer, digest method, digest value
  </Reference>
  <Reference>
    pointer, digest method, digest value
  </Reference>
</SignedInfo>
Sign the digest and place value in a SignatureValue element...

<SignedInfo>
  <CanonicalizationMethod> algorithm used on SignedInfo element
  <SignatureMethod> for example dsa-sha1
  <Reference>
    pointer, digest method, digest value
  </Reference>
  <Reference>
    pointer, digest method, digest value
  </Reference>
</SignedInfo>
<SignatureValue>Base 64 signature of the SignedInfo Element</SignatureValue>
Enclose in a Signature Element

```xml
<SignedInfo>
  <CanonicalizationMethod> algorithm used on SignedInfo element
  <SignatureMethod> for example dsa-sha1
  <Reference>
    pointer, method, digest value
  </Reference>
  <Reference>
    pointer, method, digest value
  </Reference>
</SignedInfo>
<SignatureValue>Base 64 signature of the SignedInfo Element
</SignatureValue>
```

95-702 Distributed Systems
Master of Information System Management
We may include KeyInfo

<Signature>
  <SignedInfo>
    <Canonicalization>
    <SignatureMethod>
    <Reference>…
    <Reference>…
  </SignedInfo>
  <SignatureValue>Base 64 signature of the SignedInfo Element
  </SignatureValue>
  <KeyInfo>
    <X509Data>
      <X509SubjectName>CN=Cristina McCarthy, O=CMU,…
      <X509Certificate> base 64 public key and identity signed by a CA
    </X509Certificate>
    </X509Data>
  </KeyInfo>
</Signature>
KeyInfo Element in XMLDSig

- Optional element
- Holds key information required to validate the signature or
- Points to that key information
- May have children such as
  
  `<PGPData> <SPKIData> <X509Data>`
What Can Mallory Do?

Can she modify the CA signed certificate so that someone else appears to have signed the document?

Can she modify what is being pointed by the reference element?

Can she change the canonicalization method?

Can she change the contents of the signature method tag?
Verification

1. Canonicalize the SignedInfo element.
2. Compute the digest of the SignedInfo element using the method described within it.
3. Compare the above value with that value got from applying the signer’s public key to the value in the SignatureValue element.
4. Compute digests of referenced items (after any transformations) and compare those digests found within each reference tag.
Using IBM’s XML Security Suite

Welcome to XML Security Suite!

XML is expected to facilitate Internet B2B messaging because of its simplicity and flexibility. One big concern that a customer may have in doing Internet B2B messaging is security. Internet is a public network, and there has been no protection against attacks such as eavesdropping and forgery. If messages are stolen or modified during transmission, B2B messaging will be almost useless. Fortunately, the recent advancement of public-key cryptography has remedied most of the security problems in communication. Using modern cryptographic protocols such as SSL, the Internet became as secure as any other networks, including VANs and WANs.

Our XML Security Suite will push the security further by introducing new security features such as digital signature, element-wise encryption, and access control that are beyond the capability of the transport-level security protocol such as SSL. Our goal is to contribute to the discussions of standard bodies by providing sample implementations, as well as to supply our advanced technologies to our partners and to hear what they think.

What’s in This Release?

XML Encryption Implementation

This is a set of experimental reference implementations of XML Encryption Syntax and Processing, which specifies a process for encrypting data and representing the result in XML, and Decryption Transform for XML Signature, which specifies an XML Signature transform that decrypts those encrypted after signing for the signature to validate.

(Requires Java 2.1.3, JCE 1.2, Xerces2 2.2, Xalan 2.3, ICU4J 2.4)
Sign a grade book

Gradebook.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<GradeBook>
    <Student>
        <Score>100</Score>
        <Score>89</Score>
    </Student>
</GradeBook>
```
We need keys...

D:\..\95-804\IBMXMLSecuritySuite\SampleSign2>

keytool -genkey -keyalg RSA -keystore test.keystore
   -dname "CN=Mike McCarthy, OU=Heinz School, O=CMU, L=Pgh, S=PA, C=US" -alias mjm
   -storepass  sesame -keypass sesame

Creates test.keystore holding keys and a self-signed certificate
Run XSS4J’s SampleSign2

D:\...\95-804\IBMXMLSecuritySuite\SampleSign2>java SampleSign2 mjm sesame sesame -embxml gradebook.xml > signature.xml

Key store: test.keystore
Sign: 851ms
Examine Signature.xml

<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"></CanonicalizationMethod>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></SignatureMethod>
  </SignedInfo>
</Signature>
We are signing resource 0

<Reference URI="#Res0">
  <Transforms>
    <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315">
    </Transform>
  </Transforms>
  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
    <DigestValue>m6f9xhOc4iEXokD/29V9EsdY3yl=
  </DigestValue>
  </DigestMethod>
</Reference>
<KeyInfo>
  <KeyValue>
    <RSAKeyValue>
      <Modulus>
        7V5eyhVaw0clED11H6PTPoKQA1VxrLAugU3QxKA0hbbUOiavFbqCdc6Z+Fe9JZFMkS
        lgdl+khwWwd+AlsRyrN4V2DWhm1f
        +xyYQf6bdZgCaVVgkST1BpQxBTgNKRCs5VbLrXf
        4MXb5TbhA+eo1Qbr2ljlV10aLbVhUk/g+ylag+k=
      </Modulus>
      <Exponent>AQAB</Exponent>
    </RSAKeyValue>
  </KeyValue>
</KeyInfo>
"<X509Data>
  <X509IssuerSerial>
    <X509IssuerName>CN=Mike McCarthy,OU=Heinz School,O=CMU,L=Pgh,ST=PA,C=US</X509IssuerName>
    <X509SerialNumber>1049138061</X509SerialNumber>
  </X509IssuerSerial>

<X509SubjectName>CN=Mike McCarthy,OU=Heinz School,O=CMU,L=Pgh,ST=PA,C=US</X509SubjectName>
<X509Certificate>
The resource 0 object

```xml
<dsig:Object xmlns="" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="Res0">
  <GradeBook>
    <Student>
      <Score>100</Score>
      <Score>89</Score>
    </Student>
  </GradeBook>
</dsig:Object>
```

The resource 0 object
Let’s change the low grade!

```xml
<dsig:Object xmlns="" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="Res0">

<GradeBook>
  <Student>
    <Score>100</Score>
    <Score>100</Score>
  </Student>
</GradeBook></dsig:Object>
```
And run verify…

D:\McCarthy\www\95-804\IBMXMLSecuritySuite\SampleSign2>java VerifyCUI < signature.xml

The signature has a KeyValue element.
The signature has one or more X509Data elements.
Checks an X509Data:
1 certificate(s).

Certificate Information:
  Version: 1
  Validity: OK
  SubjectDN: CN=Mike McCarthy, OU=Heinz School, O=CMU, L=Pgh, ST=PA, C=US
  IssuerDN: CN=Mike McCarthy, OU=Heinz School, O=CMU, L=Pgh, ST=PA, C=US
  Serial#: 0x3e88938d
Time to verify: 521 [msec]
Core Validity: NG
Signature Validity: OK
[0] "#Res0" NG: Digest value mismatch: calculated: tFvLyHns8wRB6l/HDU2dXZkzF+7Q=

Exception in thread "main" java.lang.RuntimeException: Core Validity: NG
at dsig.VerifyCUI.main(VerifyCUI.java:137)
Another Example PO.XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<PurchaseOrder xmlns="urn:purchase-order">
  <Customer>
    <Name>Robert Smith</Name>
    <CustomerId>788335</CustomerId>
  </Customer>
  <Item partNum="C763">
    <ProductId>6883-JF3</ProductId>
    <Quantity>3</Quantity>
    <ShipDate>2002-09-03</ShipDate>
    <Name>ThinkPad X20</Name>
  </Item>
</PurchaseOrder>
```
<?xml version='1.0' encoding='UTF-8'?>
<SignedPurchaseOrder>

<PurchaseOrder id="id0" xmlns="urn:purchase-order">
  <Customer>
    <Name>Robert Smith</Name>
    <CustomerId>788335</CustomerId>
  </Customer>
  <Item partNum="C763">
    <ProductId>6883-JF3</ProductId>
    <Quantity>3</Quantity>
    <ShipDate>2002-09-03</ShipDate>
    <Name>ThinkPad X20</Name>
  </Item>
</PurchaseOrder>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
    <Reference URI="#id0">
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>UfeiscUCL7QkhZtRDLWDPWLpVIA=</DigestValue>
    </Reference>
  </SignedInfo>
</Signature>
<SignatureValue>
   Ptysg8WdHI2mxwryOOt5l9r9qZm/
   2gNFNOJyH1Wak4nCUegRpe72tWnsigAKZyop
   mgUSH3TG
   aGGQF1BTSvk3JUUY/ljrw
   +5FpTpf3hgZBi7GSWF6WtXqZvMYGUKIlvR/
   421MZg7P9XRUyy37
   ZUzQHtmCYkBorEkEx1J4CYB0G2c=
</SignatureValue>
<?xml version="1.0"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<s:GetSpecialDiscountedBookingForPartners
xmlns:s="http://www.MyHotel.com/partnerservice/">
<!--Parameters passed with the method call-->
</s:GetSpecialDiscountedBookingForPartners>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

From “Web Services Security”, Bilal Siddiqui
There is no XMLDS in this example.
<?xml version="1.0"?>
<SOAP-ENV:Envelope
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <SOAP-ENV:Header>
    <ds:Signature>                <!-- wraps all other XMLDS elements -->
      <ds:SignedInfo>         <!-- note the ds prefix -->
        <ds:SignatureValue>
          <ds:KeyInfo>
            <ds:Signature>
              <ds:SignedInfo>
                <ds:SignatureValue>
                  <ds:KeyInfo>
                    </ds:KeyInfo>
                  </ds:SignatureValue>
                </ds:SignedInfo>
              </ds:SignatureValue>
            </ds:KeyInfo>
          </ds:SignatureValue>
        </ds:SignedInfo>
      </ds:Signature>
    </SOAP-ENV:Header>

  <SOAP-ENV:Body>
    <s:GetSpecialDiscountedBookingForPartners
 xmlns:s="http://www.MyHotel.com/partnerservice/"
    <!--Parameters passed with the method call-->
      </s:GetSpecialDiscountedBookingForPartners>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
<?xml version="1.0"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

<SOAP-ENV:Header>
<ds:Signature>
<ds:SignedInfo>
<ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
<ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
<ds:Reference URI="#GetSpecialDiscountedBookingForPartners">
<ds:Transforms>
<ds:Transform
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
</ds:Transforms>
</ds:SignedInfo>
</ds:Signature>
</SOAP-ENV:Header>
</SOAP-ENV:Envelope>
After Signing (2)

<ds:DigestMethod
    Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>
    BIUddkJKko2...
</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
    halHJghyf765....
</ds:SignatureValue>
<ds:KeyInfo> <!-- the key name for signature verification →
    <ds:KeyName>MyKeyIdentifier</ds:KeyName>
</ds:KeyInfo> <!-- application dependent, perhaps a symmetric key ID →
</ds:Signature>
</SOAP-ENV:Header>
After Signing (3)

```xml
<SOAP-ENV:Body>
  <s:GetSpecialDiscountedBookingForPartners
    xmlns:s="http://www.MyHotel.com/partnerservice/"
    ID="GetSpecialDiscountedBookingForPartners">
    <!--Parameters passed with the method call-->
  </s:GetSpecialDiscountedBookingForPartners>
</SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```
XML Encryption

- W3C Recommendation 10 December 2002
- Notes from
  by Bilal Siddiqui

  And “Secure XML” by Eastlake and Niles Addison Wesley
General Form 1

<EncryptedData>
  <CipherData>
    <CipherValue>
      cipher text in Base 64
    </CipherValue>
  </CipherData>
</EncryptedData>
General Form 2

<EncryptedData>
  <CipherData>
    <CipherReference>
      pointer (URL) to cipher text
    </CipherReference>
  </CipherData>
</EncryptedData>
EncryptedData is the core element

Replaces the encrypted element or
Serves as the new document root
May contain a KeyInfo element that describes the key needed for decryption (borrowed from XML Digital Signature) or signature verification
<MedInfo>
  <ID>
    <Name>
    <Address>
  </ID>
  <Medical>…</Medical>
  <Financial>…</Financial>
</MedInfo>
<MedInfo>
  <ID>....</ID>
  <EncryptedData>
    <KeyInfo>
      <KeyName>Medical</KeyName>
    </KeyInfo>
    <CipherData>
      <CipherValue> cipher text</CipherValue>
    </CipherData>
  </EncryptedData>
</MedInfo>
<Financial>
  <EncryptedData>
    <KeyInfo>
      <KeyName>Pay</KeyName>
    </KeyInfo>
    <CipherData>
      <CipherValue> cipher text</CipherValue>
    </CipherData>
  </EncryptedData>
</Financial>

</MedInfo>
<purchaseOrder>
  <Order>
    <Item>book</Item>
    <Id>123-958-74598</Id>
    <Quantity>12</Quantity>
  </Order>
  <Payment>
    <CardId>123654-8988889-9996874</CardId>
    <CardName>visa</CardName>
    <ValidDate>12-10-2004</ValidDate>
  </Payment>
</purchaseOrder>
Encrypting the Entire File (Listing 2)

```xml
<?xml version='1.0' ?>
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#
    Type='http://www.isi.edu/in-notes/iana/assignments/media-types/text/xml'>
    <CipherData>
        <CipherValue>A23B45C56…</CipherValue>
    </CipherData>
</EncryptedData>
```

IANA = Internet Assigned Numbers Authority a function of
The Internet Corporation
for Assigned Names and Numbers
Encrypting The Payment (Listing 3)

```xml
<?xml version='1.0' ?>
<PurchaseOrder>
  <Order>
    <Item>book</Item>
    <Id>123-958-74598</Id>
    <Quantity>12</Quantity>
  </Order>
  <EncryptedData
    Type='http://www.w3.org/2001/04/xmlenc#Element'
    xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CipherData>
      <CipherValue>A23B45C564587…</CipherValue>
    </CipherData>
  </EncryptedData>
</PurchaseOrder>
```
<?xml version='1.0' ?>
<PurchaseOrder>
  <Order>
    <Item>book</Item>
    <Id>123-958-74598</Id>
    <Quantity>12</Quantity>
  </Order>
  <Payment>
    <CardId>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
                      Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C564587</CipherValue>
        </CipherData>
      </EncryptedData>
      <CardName>visa</CardName>
      <ValidDate>12-10-2004</ValidDate>
    </CardId>
  </Payment>
</PurchaseOrder>
Encrypting Non-XML Data (Listing 5)

```xml
<?xml version='1.0' ?>
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlen#'
    Type='http://www.isi.edu/in-notes/iana/assignments/media-types/jpeg' >

    <CipherData>
        <CipherValue>A23B45C56…</CipherValue>
    </CipherData>

</EncryptedData>
```