T-110.5140 Network Application Frameworks and XML

XML Security Basics

30.3.2009

Sasu Tarkoma

Based on slides by Pekka Nikander

Contents

- **High-level view to WS security**
- WS Application level security
- Standardization landscape
- Basic XML security
- Summary

- Topics are continued in the next lecture

Need for XML security

- XML document can be encrypted using SSL or IPSec
  - this cannot handle the different parts of the document
  - documents may be routed hop-by-hop
  - different entities must process different parts of the document
- SSL/TLS/IPSec provide message integrity and privacy only when the message is in transit
- We also need to encrypt and authenticate the document in arbitrary sequences and to involve multiple parties

High-level view to WS security

- Security is as strong as the weakest link
- The options for an attacker are:
  - Attack the Web Service directly
    - Using ”unexpected” XML
  - Attack the Web Services platform
  - Attack a WS security tool
  - Attack the underlying operating system or network connection
- Let’s have examples from different security functions’ point of view and highlight key specifications

Authentication I

- End-users authenticate (their identity is verified) using username/password, SecurID or such, or biometrics
  - End-users do not send SOAP messages
- Authentication mechanisms
  - SSL/TSL (end-to-end)
  - IKE & IPSec (end-to-end)
  - Digital certificates and signatures in SOAP messages (between security contexts)
- Core specification: XML Signature
- WS-Security
  - SOAP with security tokens
    - A security token represents a set of claims.
    - Self-generated or issued by a trusted party
  - Relies on XML Signature & Encryption

Authentication II

- SAML (Security Assertion Markup Language)
  - A XML-based framework (schemas) for the exchange of authentication and authorization information
  - Mainly for integration, up to relying parties to decide to what authentication authority to trust
  - Assertions can convey information about authentication acts performed by subjects, attributes of subjects, and authorization decisions about whether subjects are allowed to access certain resources
  - Authentication statements merely describe acts of authentication that happened previously
- SAML & WS-Security allow a SOAP message to include information about the end-user's authentication status

Authorization

- Once the sender or end-user is authenticated, are they allowed to access the resource which they are requesting?
- XACML (XML Access Control Markup Language) defines how to represent access control rules in XML
- WS-Policy defines web service policies (algorithms, tokens, privacy requirements, encodings,..) between senders and receivers
  - Also other policies, declarative & conditional assertions
- SAML (Security Assertion Markup Language)
- Existing tools for authorization to websites
  - Distinguish resources as URLs
  - A single URL can contain many Web Services

Integrity

- Has this message been tampered with?
  - Checksums, digital signatures
  - PKCS#7 signature
    - Predates XML, ASN.1 binary format
    - How to sign only parts of a document (of a tree)?
  - XML Signature
- Has the system been tampered with?
  - Intrusion detection
  - Tamper control

Confidentiality

- Can the message be read while in transit?
  - Transport (or below) level security: HTTPS, IPSEC
  - Message-level security: XML Encryption, WS-Security
- Can the message be read while it is stored?
  - XML Database security
  - Access control
- Is the data private?
  - Gated access to private data
  - Audit trails of access

Audit

- Are transactions stored?
  - Does the storage alter the format? (e.g. splitting an XML message into elements in order to store it into a database)
- Is reporting available?
- Who can run / access the reports?

Availability

- Preventing denial-of-service attacks
  - Blocking unwanted message ”storms”
- Use of load-balancers
  - For XML communication platforms
  - For XML Gateways / Firewalls
- Design of underlying protocols

Administration

- Ease of setting up security policies
- Ability to inherit from a pre-existing policy
- Ability to ”push” security policy to multiple Web Services, and Web Services platforms
- Possibility of exporting a policy, and importing it into a different system
  - Plain text, SQL, XACML
- XKMS (XML Key Management)
  - PKI for XML-based security

Non-repudiation

- Preventing users (and services) from denying a transaction occurred
- Requires a combination of the security requirements which we have seen so far
  - Proof of sender
    - Signature
    - Logging
  - Proof of receipt
    - Signature
    - Acknowledgement & logging
- Notoriously difficult to implement

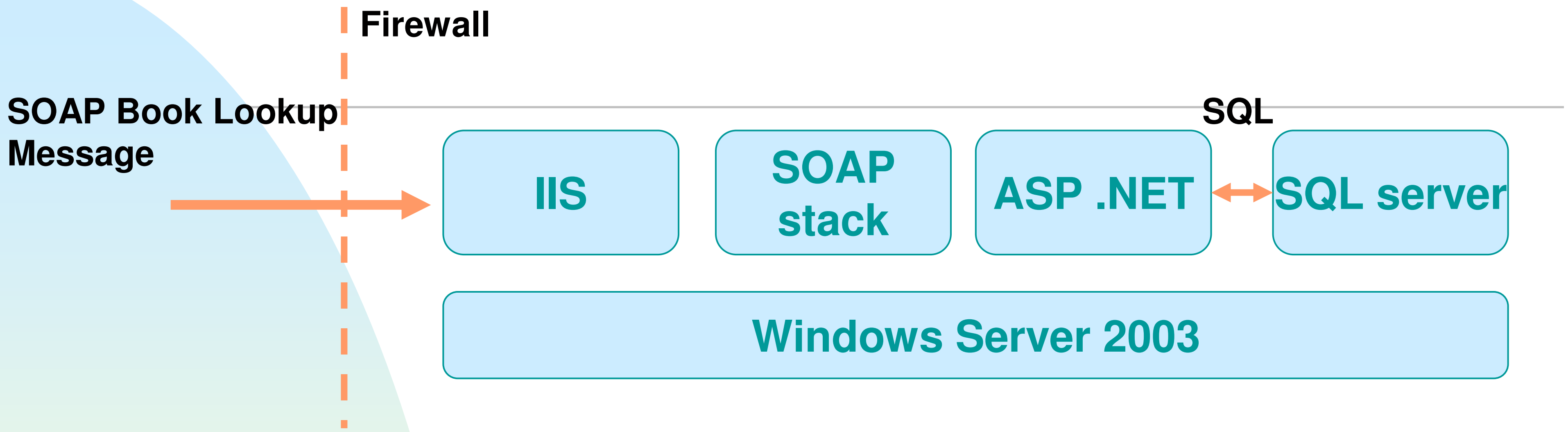
Lecture outline

- High-level view to WS Security
  - **WS Application-level security**
- Standardization landscape
- Basic XML security
- Summary

Web Application Security

- Application layer security has existed long before SOAP
- Application layer security for Web servers involves securing both the Web server itself, and Web applications which use the Web server as their platform
- Focus on attacks on Web applications rather than the platforms on which the Web applications run
  - ◆ Remember various CGI application attacks
- These attacks are specific to individual Web applications
- When bound to HTTP, SOAP itself can be seen as a Web application – albeit a more formalized one

Example – SQL Injection



Firewall

SOAP Book Lookup Message

SQL

IIS

SOAP stack

ASP .NET

SQL server

Windows Server 2003

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=”..”>
<SOAP-ENV:Header><SOAP-ENV:Header>
<SOAP-ENV:Body>
<BookLookup:searchByISBN xmlns:Booklookup=”..”>
<BookLookup:ISBN>1234567810</BookLookup:ISBN>
</BookLookup:searchByISBN>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

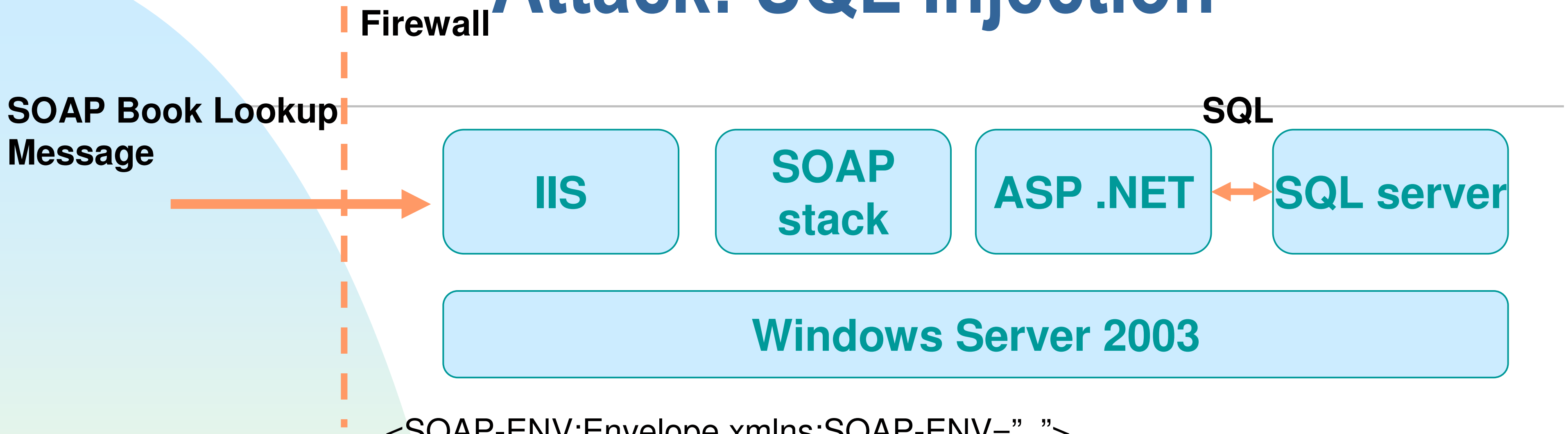
VB.NET code:

```
Set myRecordset = myConnection.execute(”SELECT * FROM myBooksTable
WHERE  ISBN=”’” & ISBN_Element_Text & ”’”)
```

Becomes

```
SELECT * FROM myBooksTable WHERE ISBN = ’1234567810’
```

Attack: SQL Injection



Firewall

SOAP Book Lookup Message

SQL

IIS

SOAP stack

ASP .NET

SQL server

Windows Server 2003

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=”..”>
<SOAP-ENV:Header><SOAP-ENV:Header>
<SOAP-ENV:Body>
<BookLookup:searchByISBN xmlns:Booklookup=”..”>
<BookLookup:ISBN>’; exec master..xp_cmdshell ’net user Joe pass /ADD’;--
</BookLookup:ISBN></BookLookup:searchByISBN>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

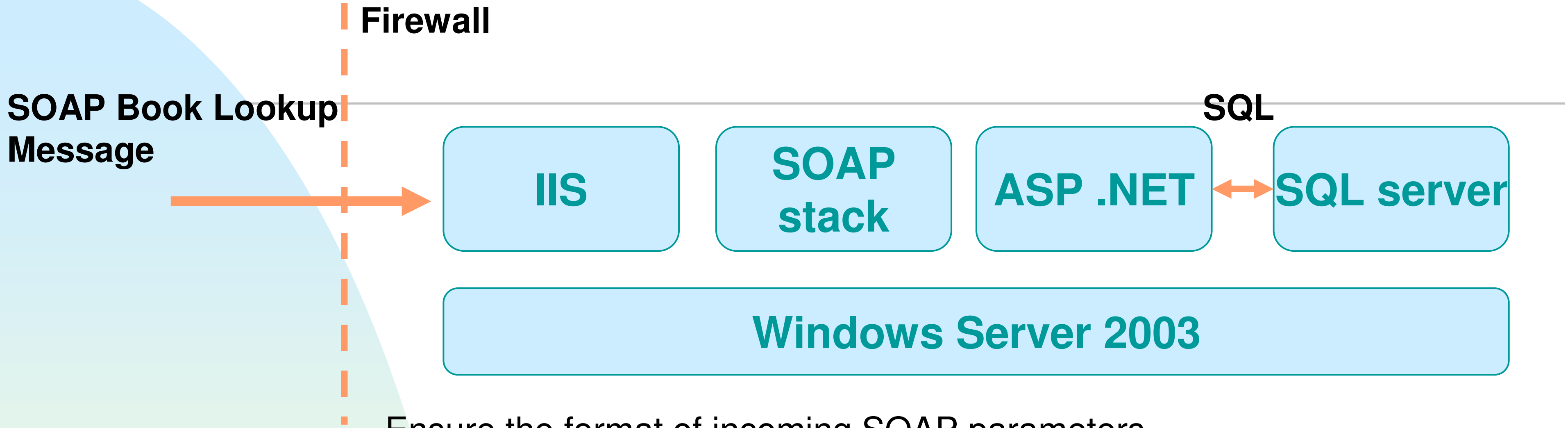
VB.NET code:

```
Set myRecordset = myConnection.execute(”SELECT * FROM myBooksTable
WHERE  ISBN=”’” & ISBN_Element_Text & ”’”)
```

Becomes

```
SELECT * FROM myBooksTable WHERE ISBN = ’’; exec master..xp_cmdshell ’ne
user Joe pass /ADD ’;—
```

Solution



Ensure the format of incoming SOAP parameters
<simpleType name=”isbn”><restrictions base=”string”><pattern value=”[0-9]{10}”/></restriction></simpleType>

Validate this Schema against the data isolated by the following XPath expression:
/Body/BookLookup:searchByISBN/BookLookup:ISBN

1234567810 passes
’exec master..xp_cmdshell ’net user Joe pass /ADD’-- fails

XML Schema Solution

```
<xsd:schema
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 targetNamespace = "https://www.books.com/Lookup"
 xmlns="https://www.books.com/Lookup"
 elementFormDefault="qualified">
  <simpleType name="isbn">
   <restriction base="string">
      <pattern value="[0-9]{10}"/>
    </restriction>
  </simpleType>
</xsd:schema>
```

Content Inspection of XML

- Integrity
  - Check integrity of data using XML Signature, WS-Security
- Schema Validation
  - Verify request structure against XML Schema
- Content Validation
  - Check content matches criteria specified in an XPath expression
- Schemas can be used to specify part of the content (for example ISBN) but they have limits
  - XPath is more expressive
  - Schema validation may always be applied to Body of SOAP msgs (rpc/literal vs. document/literal)

Application-layer Security

- Identity-based security
  - Authentication and authorization information shared across security domains
- Content-based security
  - Protecting against buffer overflow and CGI-like attacks
  - Must have knowledge about the applications to which these messages are directed
- Accountability or non-repudation
  - Need message level security
  - Maintain integrity, archived audit trails
- The standards and specifications mentioned earlier address these issues

Lecture outline

- High-level view to WS Security
  - WS Application-level security
- **Standardization landscape**
- Basic XML security
- Summary

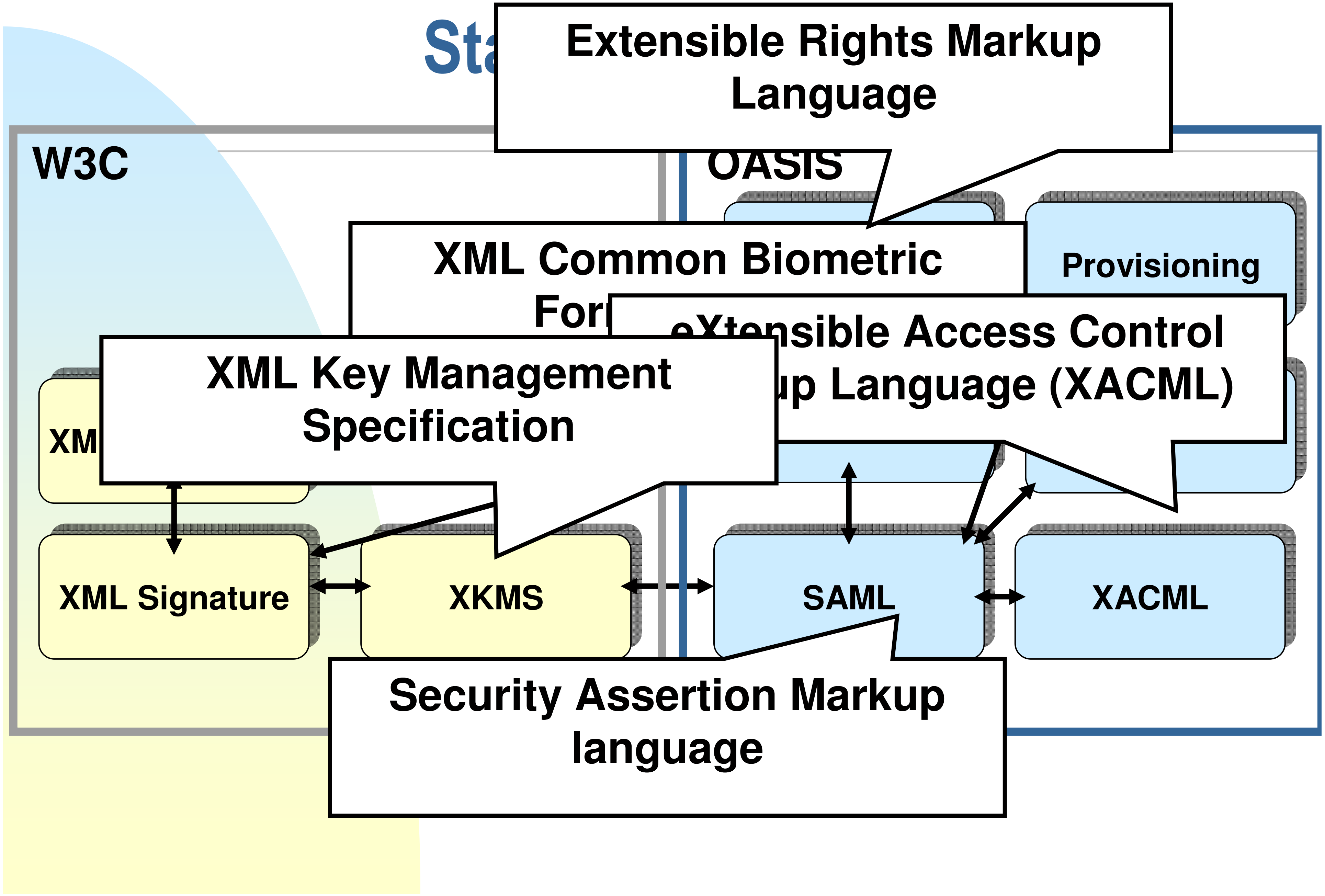
Standardization landscape

- Who are specifying the basic standards?
- Who are specifying the higher level standards?
- Who is implementing the standards?

Who are specifying the standards?

- Joint IETF/W3C
  - XML Signature (www.w3.org/Signature)
- W3C
  - XML Encryption (www.w3.org/Encryption/2001)
  - XML Key Management (XKMS) (www.w3.org/2001/XKMS)
- OASIS
  - WS-Security
    - SOAP Message Security specification etc.
  - SAML: Security Assertion Markup Language
  - XACML: Extensible Access Control Markup language
  - Electronic Business XML (ebXML) (with UN/CEFACT)
- Web Services Interoperability Organization (WS-I)
  - Basic security

Sta



Extensible Rights Markup Language

W3C

OASIS

XML Common Biometric For

Provisioning

eXtensible Access Control up Language (XACML)

XML Key Management Specification

XM

XML Signature

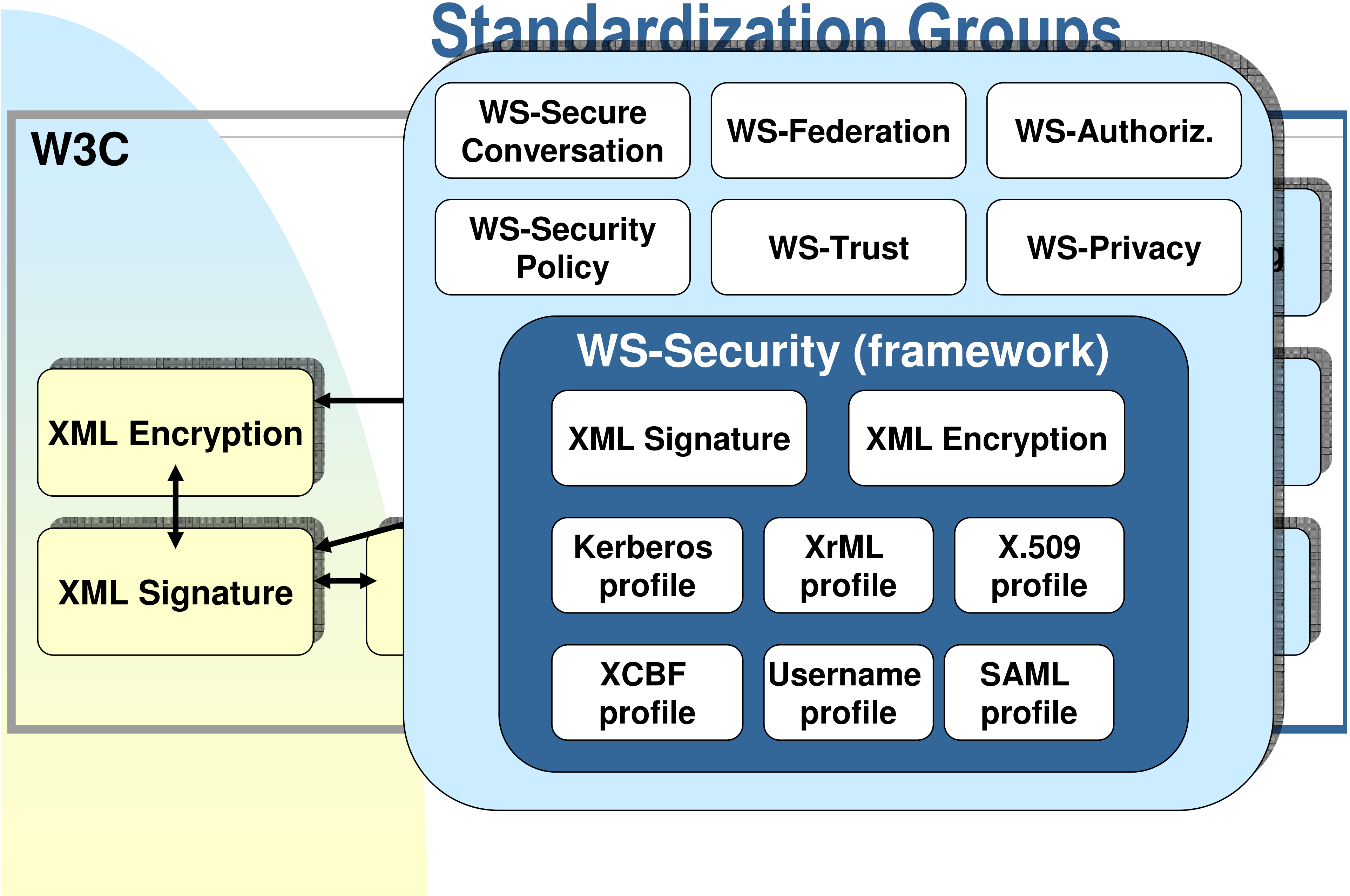
XKMS

SAML

XACML

Security Assertion Markup language

Standardization Groups



Who are specifying the higher level standards?

- Liberty Alliance (OMA)
  - Identity-based specifications (single sign-on, identity federation)
  - Specifications build on SAML, SOAP, WAP, and XML.
- Microsoft (Passport,..)
- Object Management Group (OMG)
- European Telecommunications Standards Institute (www.etsi.org)
- Organization for the Advancement of Structured Information Standards (OASIS) (www.oasis-open.org)

Who are implementing the standards?

- A lot of companies / initiatives
- Microsoft, Sun, NEC, Fuijtsu, RSA, IBM, Entrust, HP, DSTC, IAIK, Baltimore, Apache

Lecture outline

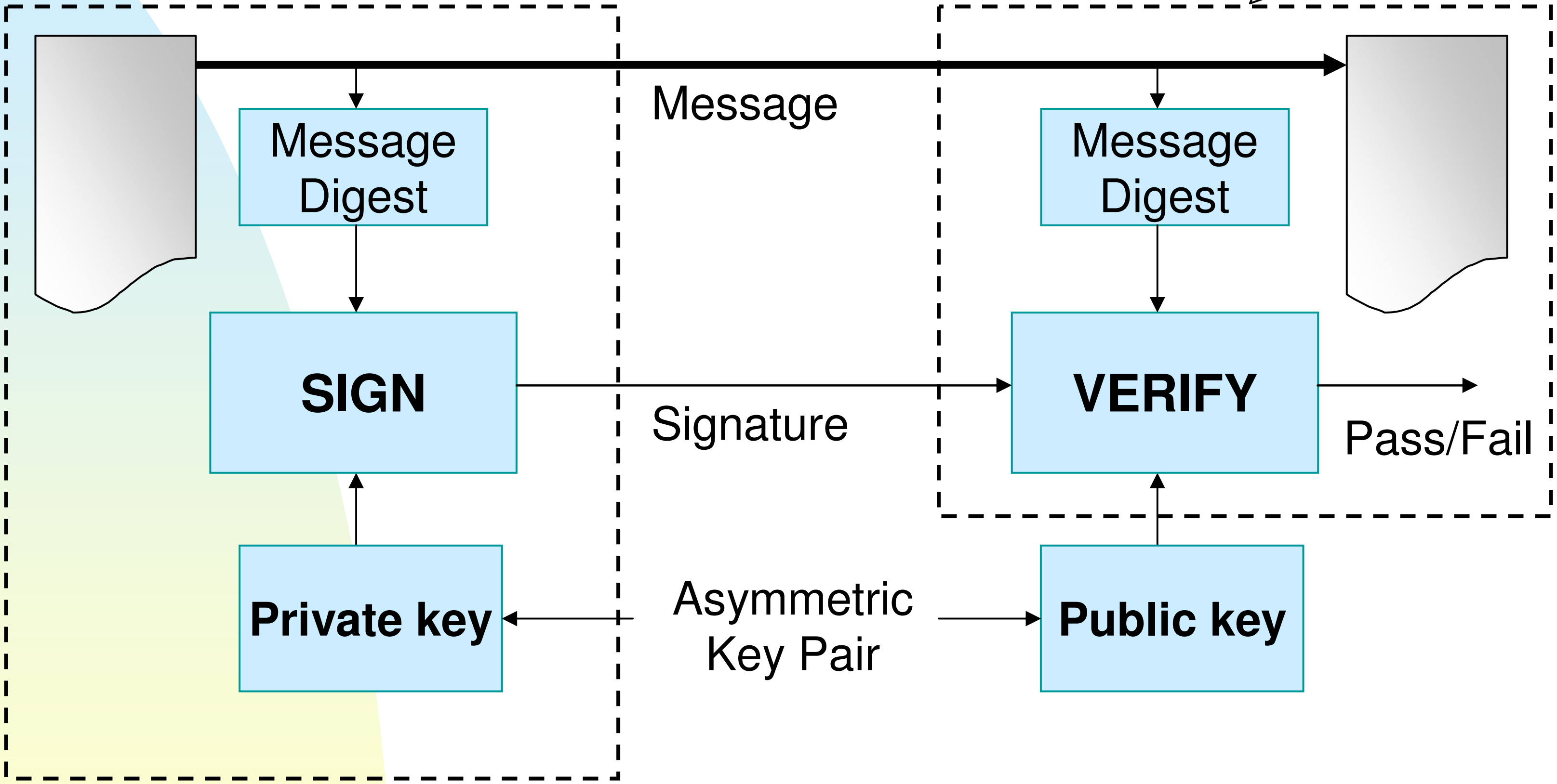
- High-level view to WS Security
  - WS Application-level security
- Standardization landscape
- **Basic XML security**
- Summary

Basic XML Security

- XML Digital Signatures (XMLDSIG)
- XML Encryption
- XML Canonicalization

Digital Signatures

Need to know the message, digest, and algorithm (f.e. SHA1)



XML Digital Signatures

- Digests calculated and a <Reference> created
- <Reference (URI=)? (Id=)? (Type=)?>(Transforms)?(DigestMethod)(DigestValue)</Reference>
- Then a <Signature> element created from <Reference>, keying information, signature algorithm, and value
    - The signature is actually calculated over the SignedInfo subset of this information
- NOTE: This means that the actual signature algorithm is ALWAYS applied to XML

XML Digital Signatures (cont.)

```
<Signature ID?>
  <SignedInfo>
    <CanonicalizationMethod/>
    <SignatureMethod/>
    (<Reference URI?>
           (<Transforms>)?
          <DigestMethod></DigestMethod>
           <DigestValue></DigestValue>
     </Reference>)+
   </SignedInfo>
   <Signaturevalue></Signaturevalue>
   (<KeyInfo>)?
   (<Object ID?>)*
</Signature>
```

detached signature of the content of the HTML 4 in XML

```
[s01] <                                                          ig#">
[s02] <
[s03] <                                    w.w3.org/TR/2001/REC-xml-c14n-200
[s04] <Signat         Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
[s05] <                                    EC-xhtml1-20000126/">
[s06] <
[s07] <
[s08] <
[s09] <DigestM
[s10] <DigestVa
[s11] </Referenc
[s12] </SignedInfo>
[s13] <SignatureV
[s14] <KeyInfo>
[s15a] <KeyValue>
[s15b] <DSAKeyValue>
[s15c] <P>...</P><Q>...</Q><G>...</G><Y>...</Y>
[s15d] </DSAKeyValue>
[s15e] </KeyValue>
[s16] </KeyInfo>
```

Canonicalization method: etc. Applied to edInfo

Signature algorithm: DSA

Reference to HTML 4 XML spec (detached)

Digest value calculated over the identified data after transformations

is the output of canonic. + digest + encrypt. For

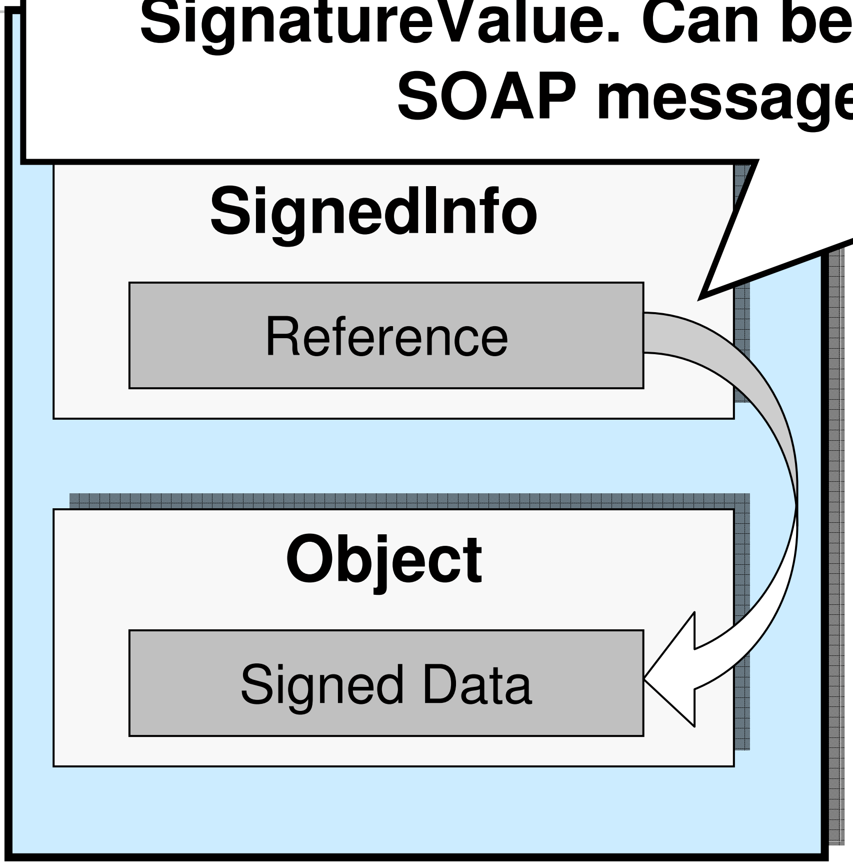
gned! validation of the and validation gest within

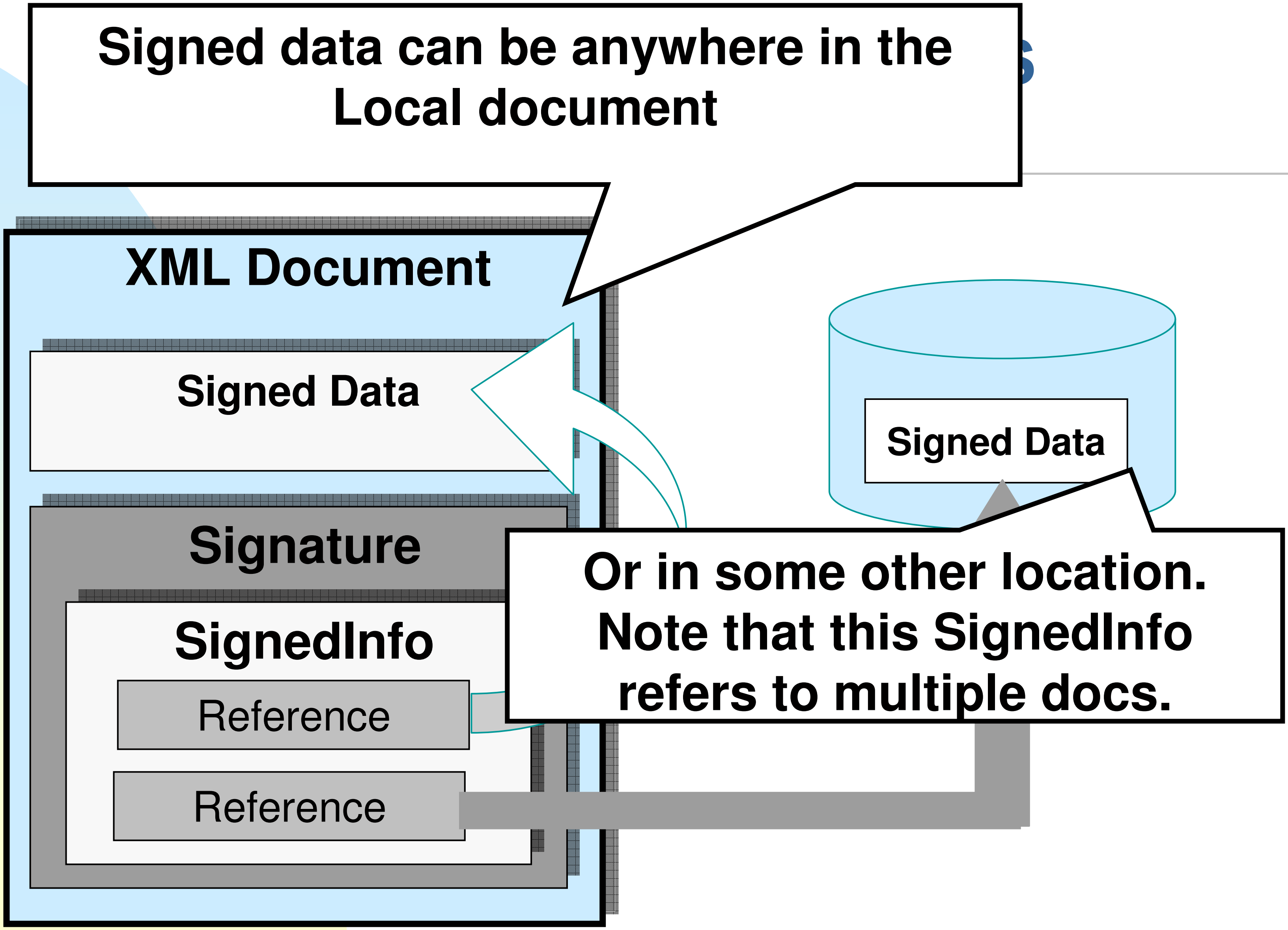
KeyInfo indicates the key to be used to validate the signature

XML Digital Signatures (cont.)

- The data being signed can be inside the <Signature>, within an <Object> element (enveloping), or
- external to the <Signature> in the same document or elsewhere (detached), or
- surrounding the <Signature> (enveloped), or
- any combination of these.

SignedInfo refers to object (sig is parent), object digested & thus in SignatureValue. Can be useful for SOAP messages

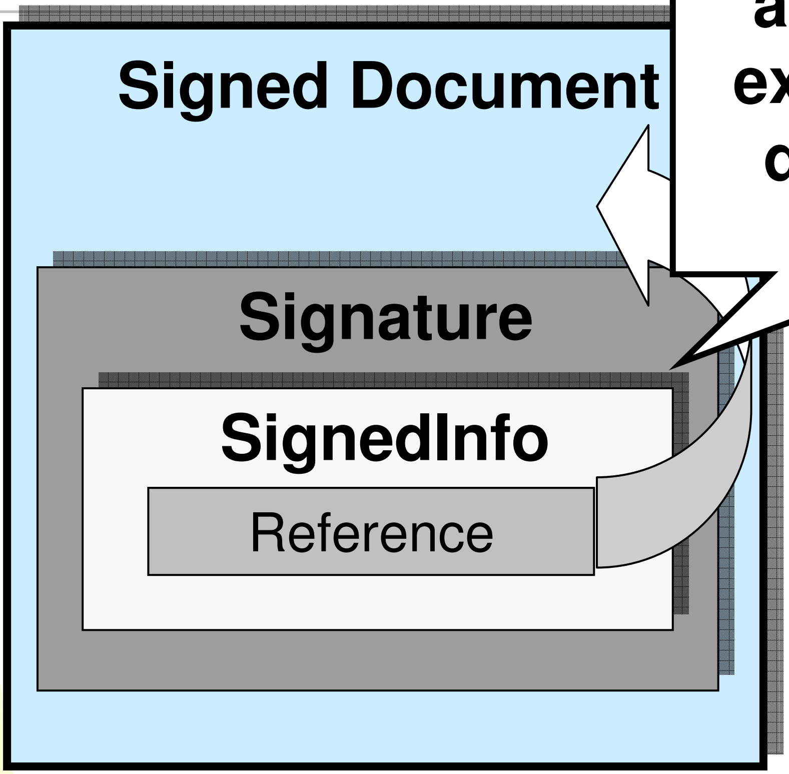




Signed data can be anywhere in the Local document

Or in some other location. Note that this SignedInfo refers to multiple docs.

Enveloped Sign



The sig is in the signed document as a child. For example: insert data to SOAP msgs

XML Signatures (cont.)

- To verify an XML digital signature
  - Verify the digests in each Reference, and
  - Verify the signature value over the SignedInfo with the appropriate key and given signature algorithm
- Note that transformations are symmetric for creation / verification! (different from transformations for encryption)

What about <Transforms>?

- A way to specify a sequence of algorithmic processing steps to apply
  - to the results retrieved from a URI to
  - Produce the data to be signed, verified, or decrypted.
  - Can include compression, encoding, subset extraction, etc. For example using XPath
  - Not needed in simple cases, but essential in complex cases

Next week

- Continue on service security
- Conclusions