Security aspects of XML and Web services

Eduardo B. Fernandez
Florida Atlantic University
Boca Raton, FL
www.cse.fau.edu/~ed
Outline

• Introduction: architectures
• XML security: transmission
• XML security: documents
• Web services security
• Industry implementations
• Conclusions
Introduction

• XML is a metalanguage used for defining markup vocabularies
• SOAP is a text-based wire protocol used to transmit XML messages
• A Web Service is a type of component that is available in the web and can be incorporated in applications or used as a standalone service
Architectures

• Web services (eServices) are a part of the application layer
• Web services are built out of XML, a lower-level data layer
• A SOAP layer is used for XML message transmission
• Internet layers and web server layers provide support for these layers
Web Services Architectural Layers

- Web Services
  - WS1
  - WS2
- UDDI Layer
  - Registry (ebXML)
- SOAP
- XML
- HTTP
Security

• Protection against:
  • Illegal (unauthorized) data disclosure (confidentiality)
  • Illegal data modification (integrity)
  • Illegal data destruction
  • Denial of service (availability)
  • Repudiation of messages
Policies

• Policies are high-level institution guidelines
• There are business policies, security policies, and system policies
• From security policies we define security models for the security systems
• Protection of messages in networks and of stored data
Message security

• Message confidentiality
• Digital signatures
• Message integrity
• Key management
• Certificates
• Authentication
Security of stored data

• Access matrix: defines who can do what to a data object. Based on authorization rules with subjects, objects, and access types

• Role-Based Access Control (RBAC): users are assigned roles according to their functions and given needed rights
XML security: transmission

- Based on transport security and document encryption
- SOAP and its lower layers provide authentication, signatures, key management, and confidentiality
- XML encryption provides confidentiality
SOAP security

• No security specification
• Security delegated to lower layers: vendor-dependent
• Authentication: Kerberos, Windows NTLM,…
• Message confidentiality: SSL, XML encryption
• Authorization: web servers
XML Message

transport

xp message

header
S2ml

payload
data  S2ml  data
SOAP message security

- Headers can be used for signatures
- Authorization and authentication information in payload
- XML data can be encrypted
- Transport data can be encrypted
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns1:sayHelloTo
            xmlns:ns1="Hello"
            SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
            <name xsi:type="xsd:string">John</name>
        </ns1:sayHelloTo>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
XML encryption requirements

- XML Encryption Working Group
- Granularity of encryption to the element (including start/end tags) or element content (between the start/end tags)
- Super-encryption possible
Public Key Infrastructure

• XML Key Management Specification (XKMS)
• Registration of key pairs (X-KRSS)
• Location of keys for later use
• Validation information associated with a key (X-KISS)
• X-KRSS and X-KISS use SOAP and XML
Adding cryptographic providers

    public void addProvider(String providerClassName) {
        outln("Adding Provider: " + providerClassName);
        try {
            Class providerClass =
                Class.forName(providerClassName);
            Provider provider =
                (Provider) providerClass.newInstance();
            Security.addProvider(provider);
        } catch (ClassNotFoundException cnf) {
            throw new RuntimeException
                ("Provider class not found: "+providerClassName);
    }
XML security: Document

- One can (and should) use domain-based security according to document contents
- Languages to define authorizations on elements (access matrix)
- SAML (Security Assertion Markup Language)
- XACL (XML Access Control Language)
- Encryption of elements
- DTDs, DOMs, and links can also be used for security
Security Assertion Markup Language (SAML)

- Part of XML-based Security Services
- XML framework for exchanging authentication and authorization information
- SAML information can be added to XML messages
SAML
XACML

• Special technical committee of OASIS
• Specification of policies for information access over the Internet
• Combines work of IBM Tokyo and University of Milano, Italy.
XML Access Control Language

• XACL is being developed at IBM’s Tokyo Research Lab
• Defines access matrix authorization rules to control access to documents or portions of a document
• Rule has subject, right, object, and condition
Access matrix authorization rules

• Basic rule \((s, a, o)\), where \(s\) is a subject (active entity), \(a\) is an access type, and \(o\) is an object

• Extended rule \((s, a, o, p)\), where \(p\) is a predicate (access condition or guard)
Example

• Documents have ‘contents’ and ‘policy’
• Alice has read and write privileges on the contents element
• Bob has only read privilege on the contents element
• No other users can access this document (closed system policy)
<document>

<contents id="contents">
  <userInfo id="section1">
    <date>Oct. 8, 1999</date>
    <name>Kudo</name>
  </userInfo>
  <bidInfo id="section2">
    <price currency="USD">150</price>
    <brand name="VISA"/>
  </bidInfo>
</contents>

</document>
<policy>
  <xacl>
    <object href="id(contents)"/>
    <rule id="rule1">
      <acl>
        <subject><uid>Alice</uid></subject>
        <privilege type="read" sign="+"/>
        <privilege type="write" sign="+"/>
      </acl>
    </rule>
    <rule id="rule2">
      <acl>
        <subject><uid>Bob</uid></subject>
        <privilege type="read" sign="+"/>
      </acl>
    </rule>
  </xacl>
</policy>
<rule id="rule3">
   <acl>
      <subject/>
      <privilege type="read" sign="-"/>
      <privilege type="write" sign="-"/>
   </acl>
</rule>
</xacl>
</policy>
</document>
Other security issues

• Different representations for the same document and the same representation for different documents
• Security of links
• Trust in intermediate steps
• Security across institutions—need for abstract models
Privacy preferences

• User control over personal information
• P3P (Platform for Privacy Preferences), developed by the W3C
• A standardized set of multiple-choice questions about privacy policies
Security enforcement

- XML and Web services security is platform-independent but must be enforced by specific platforms
- Web Server and Web Application Integrator define execution environment
- Effect of JSP, ASP, J2EE, .NET components, DBMS,…
- Effect of OS and hardware
Java-based architecture security
Microsoft architecture  *

Web Client

Request to service

Web Server (IIS)

Authorization

ASP Objects

COM objects

Result of request

Remote server

remote call

COM

COM
Web services security

• Transmission security is the same as SOAP security
• UDDI registries must be secure
• WSDL should have security statements
• Registries can also be protected according to ebXML security
UDDI

• The Universal Description, Discovery, and Integration specs define a way to publish and discover information about Web services.
• The UDDI business registration is an XML file that describes a business entity and its Web services.
• Entities are discovered via marketplaces and portals.
UDDI security

• Not specified in detail, only general policies
• Only authorized individuals can publish or change information in the registry
• Changes or deletions can only be made by the originator of the information
• Each instance of a registry can define its own user authentication mechanism
Security in ebXML

- Proposal for registry security (May 2001)
- Requirements for authentication, integrity, and confidentiality
- Each request must be authenticated
- Policy: any known entity can publish and anyone can view
- UML model for registry security
ebXML Registry Security model

- **AccessControlPolicy**
  - Permission
  - **Privilege**
    - **PrivilegeAttribute**
      - **SecurityClearance**
        - **Group**
        - **Role**
          - **Identity**
            - Principal
              - identity: Identity
              - groups: collection
              - roles: collection
              - securityClearances: collection

- **RegistryObject**
  - getGUID() : String
  - setGUID(guid : String) : void
  - getURL() : URL
  - setURL(url : URL) : void
  - getName() : String
  - setName(name : String) : void
  - depricate() : void
  - delete() : void
Security at each layer

Web Services

WS1 → UDDI Security → ebXML Sec

WS2 → UDDI Layer → Registry (ebXML)

- PAYLOAD
- HEADER

SOAP

- Dig. Signatures
- Authentication
- Key Management

XML

- Encryption
- SAML
- XACL

HTTP
Some industry products

- Microsoft’s HailStorm
- IBM Web services
- Sun ONE
- Oblix
- Netegrity
- Securant
- Distributed systems
- Glue
HailStorm

- A set of web services from Microsoft that provide a centralized way to store and access user data
- Services include calendar, wallet, notification, and others.
- Users must log in through MS Passport authentication service
- Services and data in MS servers
HailStorm security

• Passport uses Kerberos for authentication
• Doesn’t use SOAP’s security
• Users are owners of their data and can see who has had access to their data
• Microsoft web servers (IIS) have rather poor security
• .NET has RBAC security
IBM Web services

• New version of WebSphere Application Server
• WS Business Integrator will allow MQSeries to deliver SOAP messages
• DB2 Version 7.2 has a new XML Extender, where Web Services can access DBMS and can store SOAP and UDDI data
• SOAP security extensions
WebSphere Security

- WebSphere has several levels of security and provides a good environment for security
- Uses RBAC authorization
- Developed by Tivoli
SUN ONE  *

- A web service can use a policy engine to dynamically adapt processing and/or results according to rules based on user identity, authorization levels, and other contextual information
- User and policy information from LDAP
- PKI and Kerberos for authentication and message protection
- SAML for exchanging security information
Sun’s iPlanet *

- Role-based authorization
- Role hierarchies
- Administrative privileges
- Domains for segmentation of roles
- One administrator per domain
- Superuser administrator over all domains
- Authentication options
Oblix

- Security product: includes facilities for user profiles (Identity service), authorization (Access), and administration (Presentation)
- New product NetPoint 5.0 includes AccessXML, IdentityXML, and PresentationXML
- AccessXML uses SAML
Netegrity *

- TransactionMinder product for management and security of web services
- Uses SAML and XKMS
- Supports Sun ONE, MS .NET, Oracle 9i, BEA
- Had already a product for security of web sites: SiteMinder
Netegrity features *

• The facilities in Delegated Management Services (DMS) of Netegrity follow closely the proposals we made in 1979 [Woo79].

• Can assign users to roles; create, modify, and delete users; create, modify, and delete organizations and their administrators [net].
Securant *

- Access control
- Users, groups, and realms (domains)
- Can apply security constraints dynamically
- Transaction authorization
- Delegated administration
- Single Sign-on (SSO)
- Policy evaluation
- Auditing and reporting
Distributed systems

- CORBA services may be used as web services [Hou99]
- Simplifies their use in applications and browsers
- Can apply CORBA security
- Glue: Java/XML mapping for Web services, uses SOAP with HTTPS
Web services brokers

• Example: Wsbang
• A proxy server to manage Web services consumed by a given company
• Performs activities such as monitoring behavior, metering, caching,…
• Can be used for authentication: storing passwords, certificates, authorization
Conclusions I

• Rather confusing state: not clear how everything fits together and much change
• A good security model is basic to produce a consistent and complete security specification
• Access matrix and Role-Based Access Control appear as obvious choices for authorization models
Conclusions II

• There is already a lot of work on cryptography, only hooks and protocols are needed

• UML models and patterns are very useful to get the complete picture and add precision

• Institution policies are important

• Security is an all-levels problem