Ensuring Web Service Security With SecureXML™
A Look Under The Hood
Agenda

• Web Service Components
• What Are The Various Threats
• How to Architect Systems To Minimize Exposure to Threats
• A Real Life Example
• Questions
Caution

• Technology is not enough to ensure security
• Business processes must be designed with security in mind
• Security policies must be revised in a timely fashion as additional threats are discovered and business requirements change
• Policies must be enforced
Web Services Building Blocks

- Discovery: UDDI
- Description: WSDL, XML Schema
- Message Format/Encoding: SOAP/XML
- Transport: HTTP, SMTP etc.
Broad Issues With Web Services

Security

• Transport level security
• Application level security
Transport Level Security

- SSL/VPN does the job
- Configuring SSL for Web Services
  - For Apache-Axis
    - Please visit http://www.pankaj-k.net/WSOverSSL/WSOverSSL-HOWTO.html
  - For IIS
    - Configure Your Web Server for SSL
    - Install Certificate Authority's Certificate on Client
    - Modify WSDL from HTTP to HTTPS
    - Verify That It Works
    - Enforce SSL-Only Access
Application Level Security

- Authenticating data source
- Ensuring data integrity, Non-repudiation
- Protection from misbehaving clients
- Data confidentiality
STRIDE Threat Model*

- Spoofing Identity
- Tampering with Data
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege

* Idea borrowed from Scott Short
Spoofing Identity

- Authenticate principals using technologies such as X.509 certificates and 2-factor authentication
- Add XML Signature to data to ensure that the data indeed came from the right source. This could be a signature added by a known server (not necessarily an individual)
Tampering With Data

- Add Hash to data
- Add HMAC based Digital Signature
- Add X.509 certificate based digital signature (also takes care of the Spoofing Identity problem)
Repudiation

- It wasn’t me!
- Add XML Signature to the SOAP messages.
- Verify signature before accepting any message.
Information Disclosure

- Restricted functionality of the web service by using multiple WSDL files for the same web service
- Limit access to the WSDL to trusted IP addresses.
- Don’t write buggy software!
Denial of Service

• Use methods available for web server denial of service attack prevention such as proper firewall configuration.
Elevation of Privilege

- Internal application and system setup must be conformant with privilege policy.
- Use PKI/2-factor authentication along with XML Signature
Real Life Example of a Complex Web Application Which Uses Web Services and Other XML Data Exchange Mechanisms
Related W3C Standards

• XML Digital Signature
• XML Encryption
• XML Key Management Services (XKMS)
Related Oasis Standards

• SAML
Creating Digital Signature

Original Message

Hash Algorithm

Fingerprint

Encryption Algorithm

Digital Signature
XML Digital Signature

Enveloped Signature
An enveloped signature is a signature of a document, where the XML signature will itself be embedded within the signed document.
XML Digital Signature

Enveloping Signature
An enveloping signature is a signature where the signed data is actually embedded within the XML signature element.
XML Digital Signature

Detached Signature
A detached signature is a signature where the signed entities and the XML signature are separate.
What is SecureXML

- Infomosaic’s implementation of W3C XML Digital Signature Standard
- High performance implementation
- Easy to use
- Full integration with CSP layer of Windows allowing use of hardware accelerators, smart cards and USB tokens.
Using SecureXML Is Easy

- Programming Languages Supported
  - C/C++
  - Java, VB, C# etc. .NET Family of Languages

- Packaging
  - C-runtime Library
  - ActiveX Component
Using MS SOAP Client

```vbscript
<%@ LANGUAGE = JScript %>
<%

var WSDL_URL = "http://www.securexml.net/SecureXML/SecureXML.wsdl"

var soapclient
if (!Application("SecureXMLClient")) {
    soapclient = Server.CreateObject("MSSOAP.SoapClient")
    soapclient.ClientProperty("ServerHTTPRequest") = true
    soapclient.mssoapinit(WSDL_URL)
    Application.Lock
    if (!Application("SecureXMLClient")) {
        Application("SecureXMLClient") = soapclient
    }
    Application.UnLock
} else {
    soapclient = Application("SecureXMLClient")
}

var inputXML, res
inputXML = Request("inputData")
if (inputXML == "") {
    res = "No input Provided"
    Response.Write(res)
} else {
    res = soapclient.SecureXMLVerify(inputXML)
    Response.ContentType="text/xml"
    Response.Write(res)
}
inputXML = ""
%>```
Using Java (Apache-Axis) Client

```
serviceLocation = "http://www.securexml.net/SecureXML/SecureXML.wsdl"
dataFile = "signedXML.xml"

SecureXMLLocator service = new SecureXMLLocator();
SignatureSoapPort port = service.getSignatureSoapPort(new URL(serviceLocation));
File inpFile = new File(dataFile);
int fileSize = (int)inpFile.length();
FileReader fr = new FileReader(dataFile);
char[] cbuf = new char[fileSize];
int n = fr.read(cbuf, 0, cbuf.length);

System.out.println("Read " + n + " characters from file " + dataFile);

String result = port.secureXMLVerify(new String(cbuf, 0, n));

System.out.println("Result: ");
System.out.println(result);
```
import infomosaic.securexml.*;

dateFile = "signedXML.xml"
ISignature service = (ISignature) new Signature();
File inpFile = new File(dataFile);
int fileSize = (int)inpFile.length();
FileReader fr = new FileReader(dataFile);
char[] cbuf = new char[fileSize];
int n = fr.read(cbuf, 0, cbuf.length);
System.out.println("Read " + n + " characters from file " + dataFile);
String result = " ";
try {
    result = service.SecureXMLVerify(new String(cbuf, 0, n));
} catch (Exception e) {}
System.out.println("Result:");
System.out.println(service.SecureXMLVerify(new String(cbuf, 0, n)));

Demonstration of a simple purchase order web service using SecureXML to ensure data security
Questions And Follow Up

• My contact information:
  Manoj Srivastava
  manoj@infomosaic.com
  Phone: (408) 351-3337

• Download this presentation: Visit
  http://www.infomosaic.net and follow the link to WWW2002