# IRIS Web Services Workshop II

Session 5
Friday AM, September 23

Dennis M. Sosnoski

## Outline

- Web service security
  - – Point-to-point security with SSL
  - – Granular security with WS-Security
- Advanced topics
  - – Authentication/authorization and session handling
  - – WS-Addressing
  - – Other WS-* "standards"
- A look ahead: Axis2

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## SOAP security

- Sensitive applications growing, security crucial
- WS-I BP encourages TLS/SSL for security
  - – SSL widely implemented and widely supported
  - – Services may require SSL and use HTTPS endpoint
    - Same as browser secure connection
    - Negotiation between client and server assures secrecy
  - – May also require mutual authentication
    - Separate certificates for each end
    - Assures the client is who you think it is

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## SSL SOAP

- Basic SSL provides transport confidentiality
  - – Supported by most implementations
  - – Generally requires only server configuration
  - – Operates transparently (but with added value)
- Mutual authentication SSL for SOAP
  - – Generally just a flag setting for server
  - – Not all clients are set up to support
- Reasonably fast and highly secure
  - – But only good for point-to-point

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Keys and certificates

- SSL uses public/private key pair for setup
  - – Server has private key only it knows
  - – Server sends certificate to client with public key
  - – Client normally needs to verify certificate
    - Automatic if signed by known authority (e.g. Verisign)
    - Otherwise needs special handling for client to process
  - – Public/private key pair used to establish session
- Secret key for symmetrical encryption of session determined as part of setup

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Managing certificates

- Self-signed certificates can be used
  - – Need to generate private key and export certificate
  - – Certificate needs to be installed on client
  - – Java needs to be told where certificate resides
- Mutual authentication SSL uses separate key/certificate for client
  - – Unless signed by authority, certificate must be installed on server
  - – Java again needs to be told where it resides

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Using Java tools

- Java supplies keytool program in JDK
  - – Generate key with:

```
keytool -genkey -alias tomcat -keyalg RSA
  -keypass changeit
```

  - – Export certificate with:

```
keytool -export -alias tomcat -file
  server.cert
```

  - – Import certificate with:

```
keytool -import -alias tomcat -file
  server.cert
```

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# SSL demonstration

- Default keystore is *.keystore* in user home
- Generate Tomcat key there
- Import certificate to client *.truststore*:

```
keytool -import -alias tomcat -file
  server.cert -keystore .truststore
```

- Use imported certificate from client code

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Configuring SSL (1)

- Steps to SSL with mutual authentication:
  - – Set up the keys and certificates
    - Generate keys for client and server in keystores
    - Export certificates from both
    - Import certificate from each into other's truststores
  - – Self-authorized keys can be generated directly
  - – Keys backed by an authority must be purchased

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Configuring SSL (2)

- Steps to SSL with mutual authentication (cont.):
  - – Configure Tomcat for SSL with mutual authentication
    - Uncomment SSL port in server.xml
    - Link to server keystore if not default (**keystoreFile** and **keystorePass** attributes of **<Factory>**)
    - Change **clientAuth** attribute to **"true"**
  - – Run Tomcat with truststore information
    - -Djavax.net.ssl.trustStorePassword=changeit
    - -Djavax.net.ssl.trustStore=.../server.truststore

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Configuring SSL (3)

- Steps to SSL with mutual authentication (cont.):
  - – Set required properties on client (code or JVM args):

```
// configure for SSL usage
System.setProperty("java.protocol.handler.pkgs",
    "com.sun.net.ssl.internal.www.protocol");
System.setProperty("javax.net.ssl.trustStore",
    ".../client.truststore");
System.setProperty("javax.net.ssl.trustStoreType", "JKS");
System.setProperty("javax.net.ssl.trustStorePassword",
    "changeit");
System.setProperty("javax.net.ssl.keyStore",
    ".../client.keystore");
System.setProperty("javax.net.ssl.keyStoreType", "JKS");
System.setProperty("javax.net.ssl.keyStorePassword",
    "changeit");
```

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Configuring SSL (4)

- Steps to SSL with mutual authentication (cont.):
  - – Use HTTPS request to server:

```
m_target =
    "https://localhost:8443/axis/services/seiscastor");
m_proxy = (SeismicBindingStub)
    new SeismicServiceLocator().getseiscastor();
m_proxy._setProperty
    (Stub.ENDPOINT_ADDRESS_PROPERTY, m_target);
```

  - – And you're done!
- Same principle applies for all service types

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## Beyond point-to-point

- Consider complex distributed application
  - – Order from customer to store on credit card
  - – Store needs to see order information, not credit card
  - – Can pass encrypted credit card info on to bank
  - – Perhaps use digital signature to authorize payment
- Point-to-point security not enough

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## WS-Security

- Now an OASIS standard
- Supports wide range of security features
  - – Two basic aspects:
    - Assuring message confidentiality (encryption)
    - Assuring message integrity and authenticity (signing)
  - – Uses header fields (“tokens”) for security information
    - Targeted to a particular recipient (intermediate or end)
    - Extensible to support all types of security tokens
  - – Can build on HTTPS/SSL for transport security

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## WS-Security details

- Message signing support
  - – Based on XML Signature
  - – Multiple signatures across all or parts of document
- Message confidentiality support
  - – Based on XML Encryption
  - – Multiple encryptions across all or parts of document
- Associated standards for details (x509, user name, etc.)

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## XML Signature

- XML Signature can be applied to any data
  - – Portion of XML document
  - – Entire XML document
  - – Data external to document (but accessible)
- Provides everything necessary for verification
  - – Canonicalization method, signature method
  - – Source, digest method, and digest for each resource
  - – Signed digest of all of the above
  - – Certificate (verifiable) with public key

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# XML Canonicalization

- Digital signature guarantees data is unchanged
  - – Easy for fixed data, but can be problematic for XML
    - Parsing and serializing document may change actual text
      - – Attribute order is arbitrary
      - – Whitespace and line endings can be different
      - – Character entities, CDATA sections, etc.
  - – Canonicalization gives unique text
    - Chooses how to do serialization (such as attribute order)
    - Not intended as a general “preferred” serialization
- Standard signatures can apply to canonical text

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# XML Encryption

- XML Encryption can be applied to any data
  - – Data can be embedded within encryption element
  - – Encryption element can reference the raw data
- Encryption can be nested
  - – Embedded encryption useful for controlled access
  - – Direct recipient may not need all particulars of data
- Key “handle” information can be included
  - – Not actual key, since symmetrical encryption used

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Security in Java

- JSR-105 XML Digital Signatures
  - – Released Summer of 2005 (JavaOne)
  - – Distributed as part of JWSDP 1.6
- JSR-106 Digital Encryption
  - – Not yet available
- XWS-Security for security web services
  - – Based on JSR-105 and Apache XML-Encryption
  - – Early Access in JWSDP 1.6 (not for production!)

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Security with Axis

- WSS4J one way to do this
  - – Uses Axis handlers to intercept messages
  - – Uses Apache XML Security for processing
    - Requires recent Xalan XSL/T engine (for XPath?)
    - Build from source, fetches JCE implementation
- Axis includes security sample
  - – Apparently also based on Apache XML Security
- Expect changes with each release

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## WS-Security importance

- Small near-term, big for future
  - Use for specialized needs (intermediaries, etc.)
  - Possibly use for custom tokens (encrypted user name and password) as Header fields
  - Use basic SSL – where you can – until WS-Security is accepted and widely supported
- Probably 1-2 years before in general use
  - Limited support in current toolkits
  - Interoperability possible, but far from automatic

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## Hardening services

- Hardening a separate issue from securing
  - Securing allows only intended uses
  - Hardening blocks interference
- Some techniques apply to both
  - Access control (via IP address list, certificates, etc.)
  - Authentication (to make sure user is valid)
- Some techniques can be at cross-purposes

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## Authorization and authentication

- Authorization and authentication needs work
  - Most frameworks support HTTP Basic Auth
  - Can also pass user name / password using headers
  - These techniques insecure unless done over SSL
- Direct login operation best at present
  - Login can be done over SSL, or use WS-Security
  - Returns token used to identify user and session
  - Pass token as a parameter to all other operations
  - Can tie to particular IP address for added security

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

## Sessions

- Many different approaches used
  - Most frameworks support HTTP cookies
  - Some (.NET) prefer header fields
- Need to associate session with particular object instance on server
- Token again probably best approach for now
  - Initial operation generates returned session token
  - Other operations look up session from token

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Addressing

- Normal web services use HTTP features
  - Connecting to a particular endpoint and service
  - Invoking an operation on that service
  - Receiving back a response
- Other transports (e.g., SMTP) have different features
- WS-Addressing designed to provide common handling

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Addressing example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <soap:Header>
    <wsa:MessageID>
      uuid:6B29FC40-CA47-1067-B31D-00DD010662DA
    </wsa:MessageID>
    <wsa:ReplyTo>
      <wsa:Address>http://business456.example/client1</wsa:Address>
    </wsa:ReplyTo>
    <wsa:To>http://fabrikam123.example/Purchasing</wsa:To>
    <wsa:Action>http://fabrikam123.example/SubmitPO</wsa:Action>
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope>
```

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Addressing importance

- Brings other transports to par with HTTP
- Allows more flexible processing
  - Efficient queuing systems for decoupled services
  - Multistage message exchange patterns
    - Request-Response-Response, for example
- Candidate recommendation from W3C
- Likely to be widely supported in next year
  - Integrated support in Axis2

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# WS-Reliability/WS-RM

- Goal is to support reliable message exchange
  - Guaranteed delivery and ordering features
  - Requires some form of message identifiers
  - Acknowledgements of messages received
- WS-Reliability OASIS standard (v1.1)
- WS-ReliableMessaging (WS-RM) is BEA/IBM/Microsoft/TIBCO alternative
- Not much use until standard is settled

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# More WS-*

- Notification standards (OASIS)
  - WS-BaseNotification, WS-Topics, WS-BrokeredNotification
  - Designed for publish/subscribe services
- Resource framework standards (OASIS)
  - WS-Resource, WS-ResourceProperties, WS-ResourceLivetime, WS-ServiceGroup, etc.
  - Modeling and accessing stateful resources
- Many others proposed or in progress

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Apache Axis2

- Axis2 a full redesign of web services support
  - Core is flexible structure for SOAP infrastructure
  - Support pluggable data binding frameworks for XML processing
  - Will support JAX-WS 2.0 with a wrapper around the core (not baked-in, as with current Axis)
- Available now as 0.91 release
  - Partially usable, though problems in many areas

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# AXIOM

- AXIOM (AXIs Object Model)
  - Essentially a just-in-time DOM
    - Works off pull parser
    - Constructs components as requested
    - Provides access to pull parser for data not yet read
  - Very nice model for working with SOAP
    - Envelope and headers can be processed through OM
    - Body can be processed by data binding
  - See http://ws.apache.org/axis2/OMTutorial.html

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Axis2 handlers

- Handler architecture based on Axis
  - Can configure handlers inserted into message flow
  - Handlers can access message, make changes, etc.
  - Separate in and out message flows
- Adds more phases of processing
  - Things need to be done in a particular order (such as encryption/decryption)
  - Handlers must be able to specify the processing stage they need

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Axis2 deployment

- Web service deployment (HTTP) uses *.aar*
  - – Basically just a jar with one added file
  - – *META-INF/service.xml* gives service configuration
  - – Being extended for multiple services per *.aar*
- Add service *.aar* to deployed Axis2 server
  - – Just drop into expanded *axis2/WEB-INF/services* directory

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Asynchronous support

- Axis2 supports asynchronous operations
  - – By default, generates both synchronous and asynchronous APIs on client
  - – Client can use either form to access service
- Usable with both simplex and duplex transports
  - – Duplex transport (e.g. HTTP) uses single connection for request and response
  - – Simplex (e.g., SMTP) needs separate connections

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# Axis2 data binding

- Currently ships with XMLBeans support
  - – Generate all XMLBeans artifacts from WSDL
  - – No support for wrapped, only basic doc/lit
- Axis2 demonstration
- Looks good so far, but progress slow
  - – Originally supposed to be 1.0 in August
  - – Now looks like Q1 2006, but no stated goal
- Likely to take over from Axis when done

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.

# When to SOAP?

- SOAP for highly-interoperable applications
  - – Published service interfaces
  - – Clients using different platforms or languages
- Works best with granular interface
  - – Try to avoid repeated calls for single use case
- Works best with moderate data volume
  - – For larger data volumes, use attachments
- Build toward composable functions for long term use

Copyright © 2002-2005, Sosnoski Software Solutions, Inc. All rights reserved.