Java™ Technology and Web Services Security in Action

Marc Chanliau and Vikas Jain
Security Product Management

Oracle Corporation
www.oracle.com

TS-8131

Goal

Learn about key industry and Java™ technology security standards and how they are implemented in protected service-oriented architecture (SOA) deployments

Agenda

- The Need for Security in SOA Environments
  - “Define Once, Enforce Anywhere”
- Paradigmatic Use Cases
  - SOA Environments
  - Web Applications
- Key Industry Standards
  - XML Frameworks
  - Java Technology Frameworks
- Implementation Choices
- Product Demo

Agenda

The Need for Security in SOA Environments

- **"Define Once, Enforce Anywhere"**

Paradigmatic Use Cases

- SOA Environments
- Web Applications

Key Industry Standards

- XML Frameworks
- Java Technology Frameworks

Implementation Choices

Product Demo

The Need for Security in SOA Environments

- Access to resources and services over HTTP (mainly)
  - Insecure port 80
  - Readable messages (XML)—Message-level security required
- Declarative security
  - No hard-coded security
- Define security centrally
  - Policies are in a single point of control and administration
- Enforce security locally
  - Policy enforcement points are distributed across the environment
- Heterogeneous environments
  - Industry standards for integration and interoperability
  - Flexible deployment (multiple-platform support)

Agenda

The Need for Security in SOA Environments

- "Define Once, Enforce Anywhere"

Paradigmatic Use Cases

- **SOA Environments**
- **Web Applications**

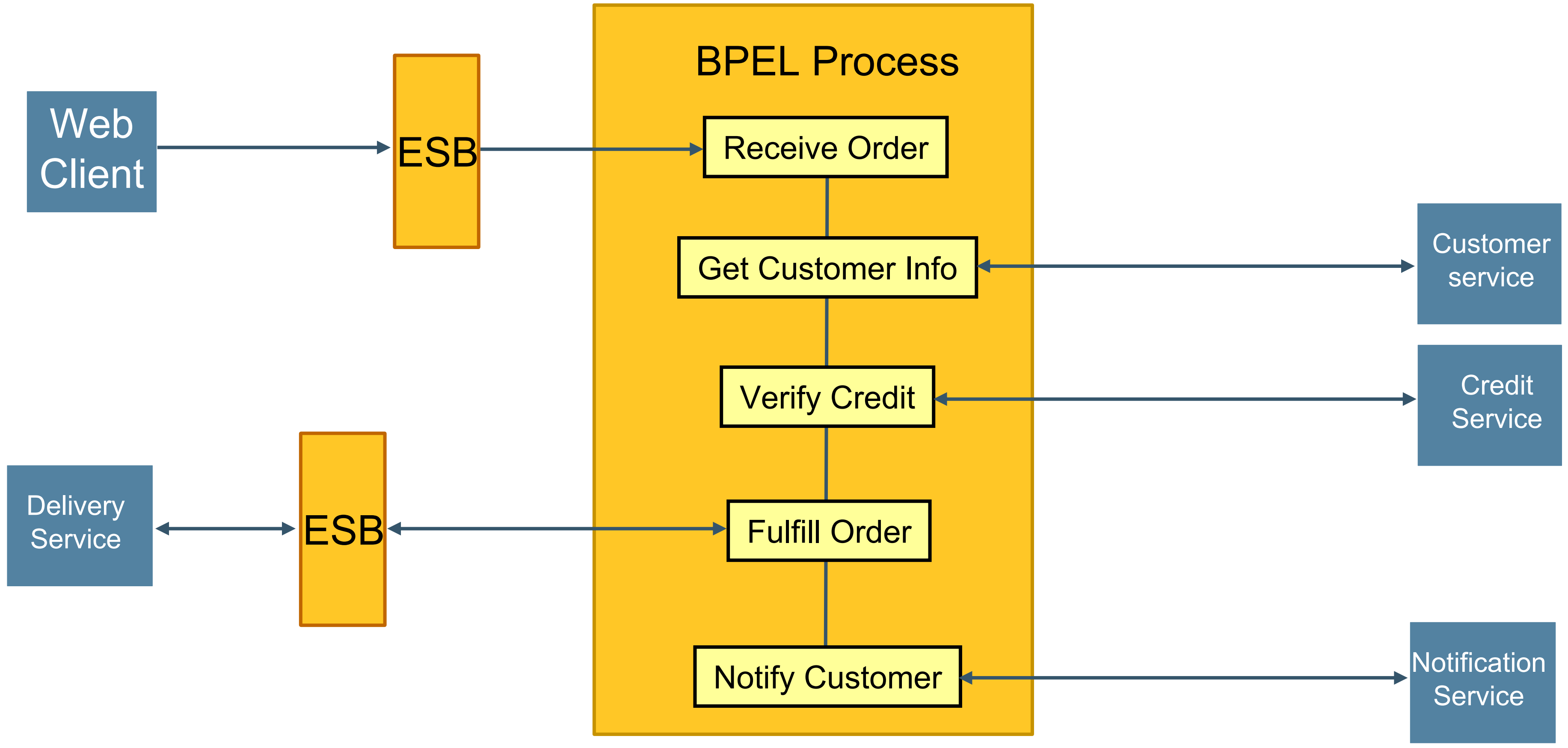
Key Industry Standards

- XML Frameworks
- Java Technology Frameworks

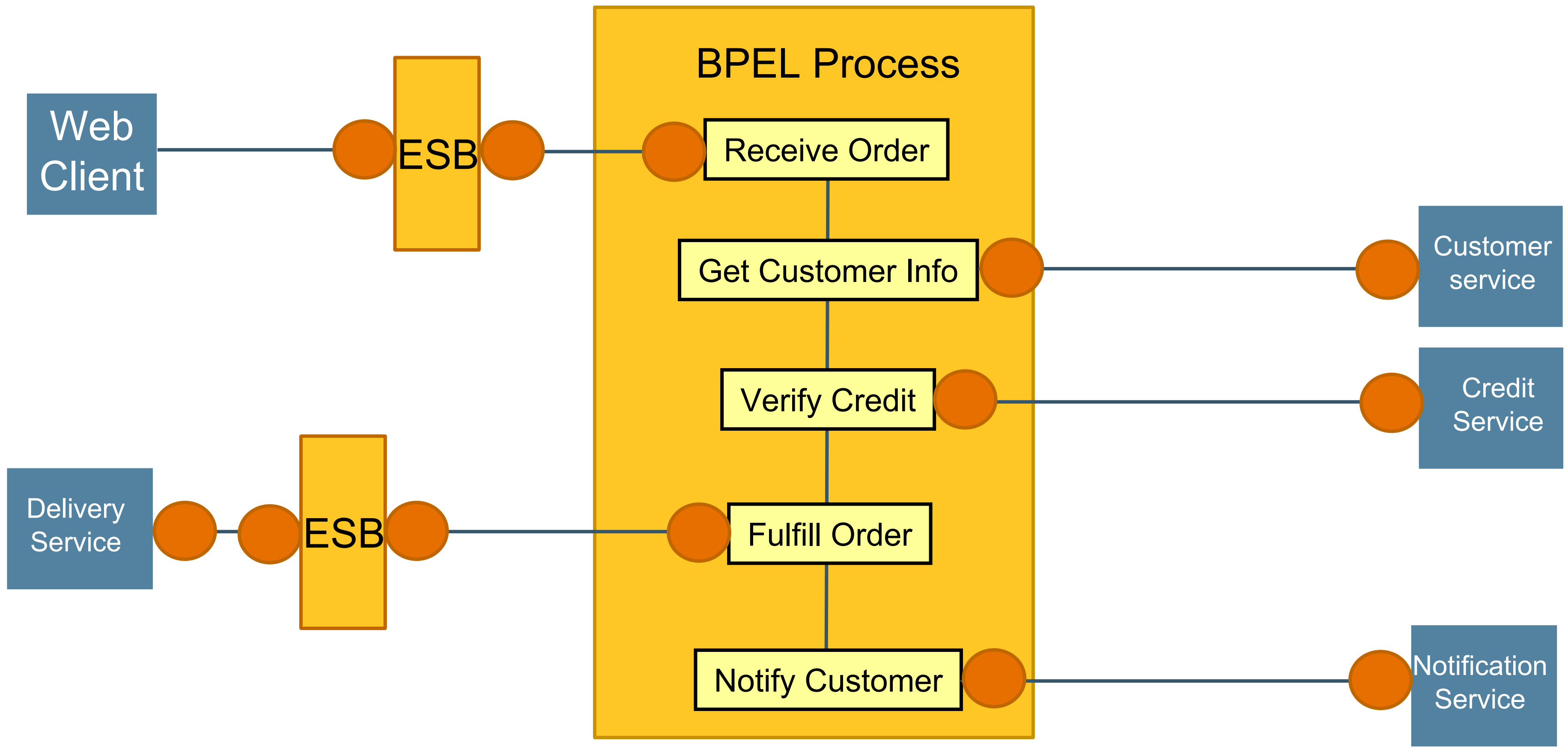
Implementation Choices

Product Demo

Use Case #1: SOA Application

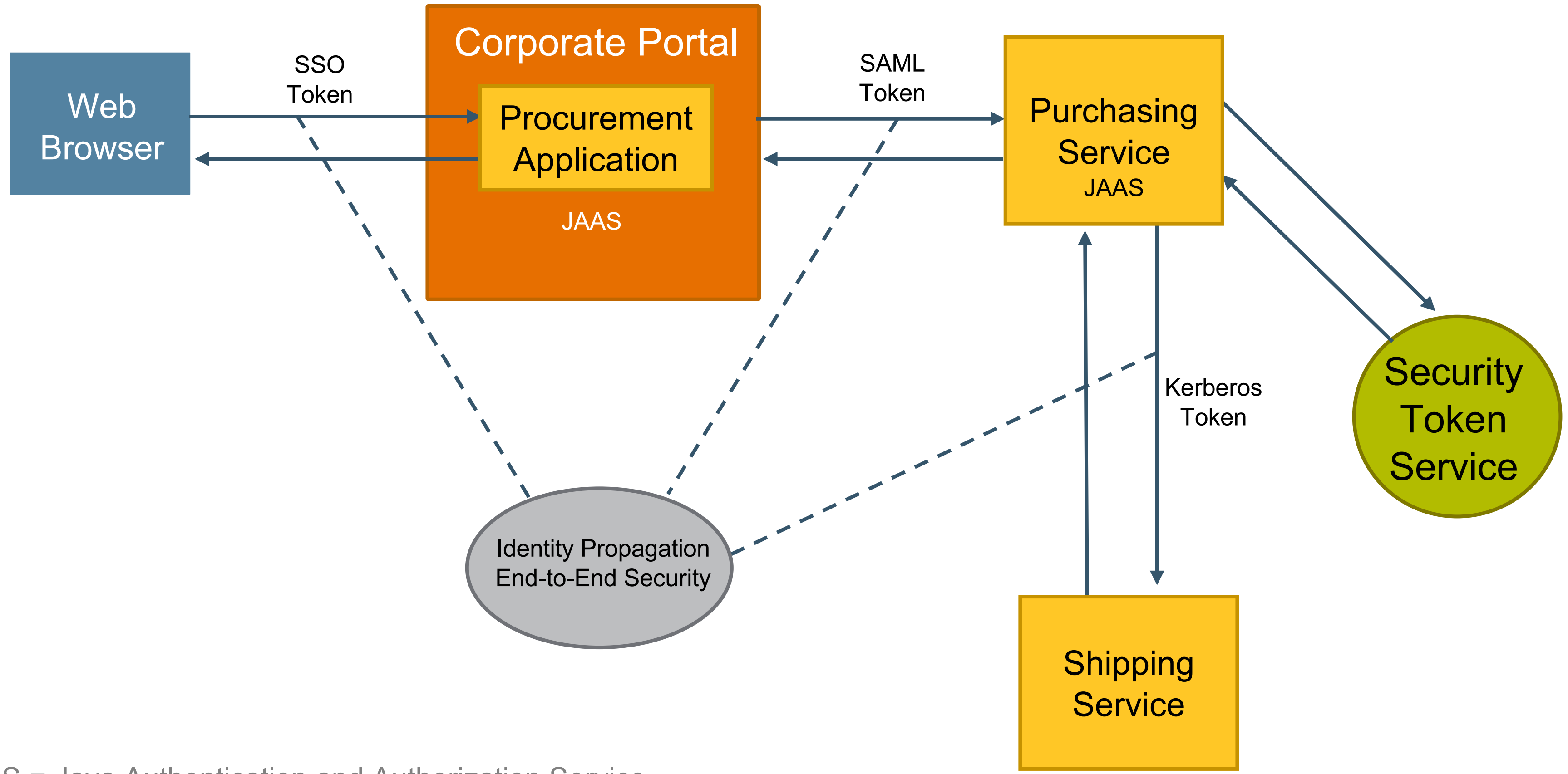


Use Case #1 Security Vulnerabilities



● : Security vulnerabilities

Use Case #2:
Web Application Invoking Web Services



JAAS = Java Authentication and Authorization Service

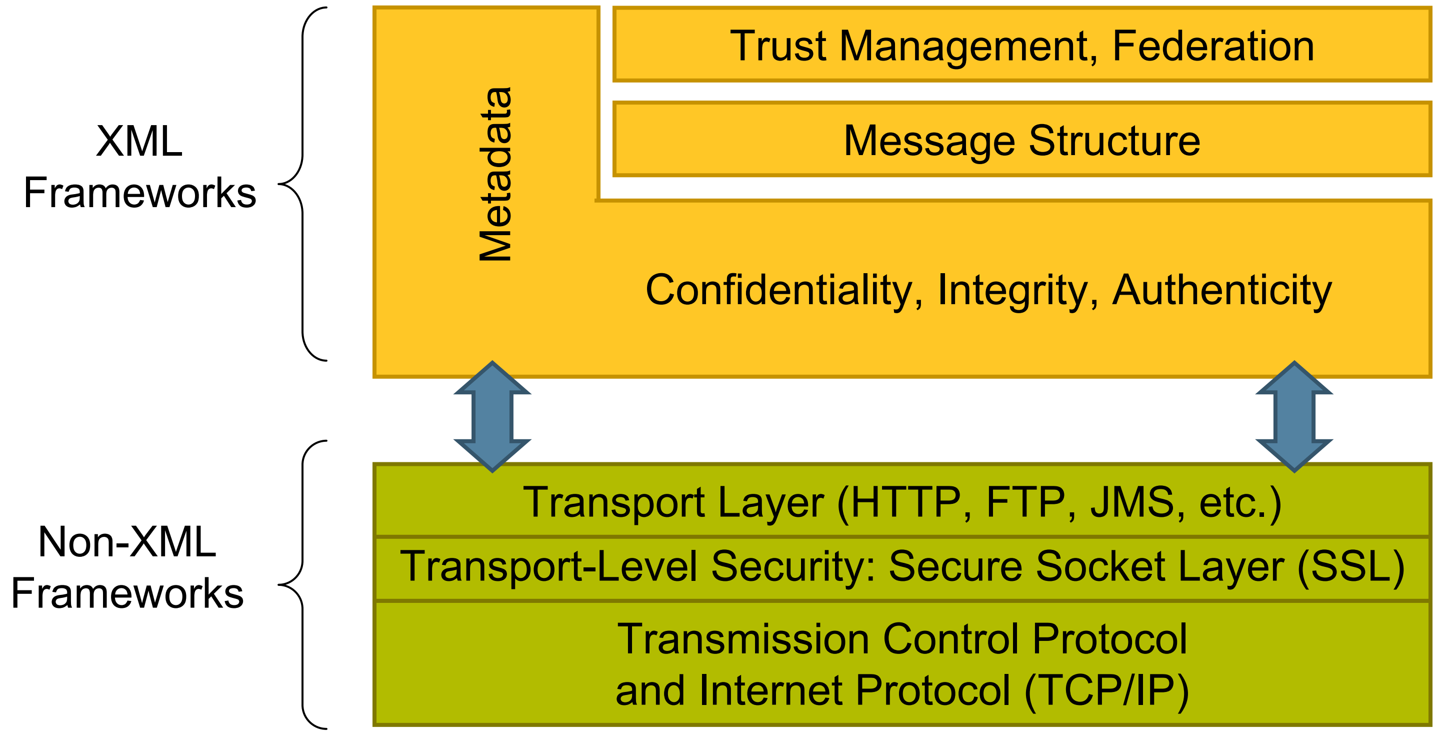
Agenda

- The Need for Security in SOA Environments
  - “Define Once, Enforce Anywhere”
- Paradigmatic Use Cases
  - SOA Environments
  - Web Applications
- **Key Industry Standards**
  - **XML Frameworks**
  - Java Technology Frameworks
- Implementation Choices
- Product Demo

Requirements

- Authentication
  - Verify that the user is who she claims to be
- Authorization—Access Control
  - Verify that the authenticated user has access rights to the service invoked
- Confidentiality
  - Hide whole or part of a document using encryption
- Integrity, non-repudiation
  - Have an authority digitally sign a document
- Credential mediation
  - Exchange security tokens in a trusted environment
- Service capabilities and constraints
  - Define what a service can do, under what circumstances

Key Industry-Standard Security Frameworks


XML Frameworks
Metadata
Trust Management, Federation
Message Structure
Confidentiality, Integrity, Authenticity
Non-XML Frameworks
Transport Layer (HTTP, FTP, JMS, etc.)
Transport-Level Security: Secure Socket Layer (SSL)
Transmission Control Protocol and Internet Protocol (TCP/IP)

JMS = Java Message Service

Application-Level Security

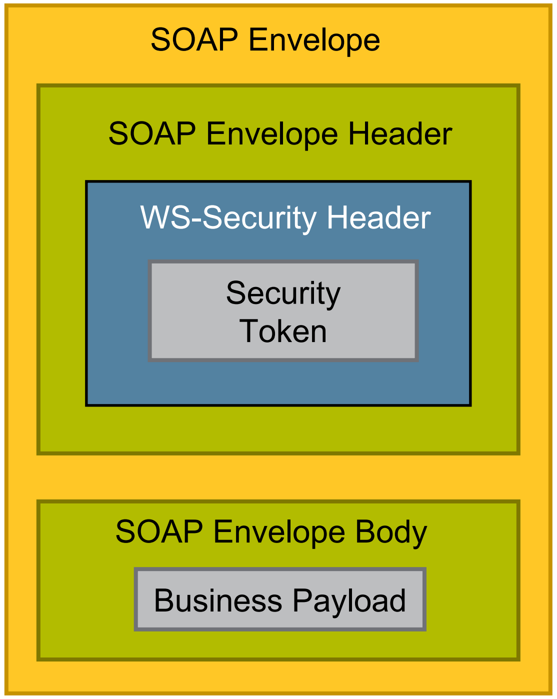
- Complements transport-level security (SSL)
- Based on XML frameworks
  - Confidentiality, Integrity, Authenticity
    - XML Encryption, XML Signature
  - Message Structure, Message Security
    - SOAP, WS-Security
  - Trust Management/Federation
    - WS-Trust
    - WS-SecureConversation
  - Metadata
    - WS-Policy, WS-PolicyAttachment
    - WS-MetadataExchange

Confidentiality, Integrity, Authenticity: XML Encryption, XML Signature

- XML Encryption (data confidentiality)
  - How digital content is encrypted and decrypted
  - How the encryption key information is passed to a recipient
  - How encrypted data is identified to facilitate decryption
- XML Signature (data integrity, authenticity)
  - Bind the sender’s identity (or “signing entity”) to an XML document
    - Signing/signature verification can be done using asymmetric or symmetric keys
  - Ensure non-repudiation of the signing entity
    - Proves that messages have not been altered since they were signed

Message Structure, Message Security: SOAP, WS-Security

- WS-Security defines how to attach XML Signature and XML Encryption headers to SOAP messages
- WS-Security provides profiles for 5 security tokens
  - Username (with opt. pwd digest)
  - X.509 cert
  - Kerberos ticket
  - SAML assertion
  - REL (rights markup) document



WS-Security With SAML Security Token

- SAML assertions and references to assertion identifiers are contained in the `<wsse:Security>` element, which in turn is included in the `<SOAP-ENV:Header>` element (described in the WS-Security SAML Token Profile)

```
<SOAP-ENV:Envelope>
  <SOAP-ENV:Header>
    <wsse:Security>
      <saml:Assertion> - - - </saml:Assertion>
    </wsse:Security>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body> - - - </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

WS-Trust

- WS-Security assumes parties already know each other and agree on the security token used
- WS-Trust addresses situations where trust must be brokered between parties that don’t use the same security tokens
  - A Security Token Service (STS) enables security token exchange, token issuance, and token validation
  - WS-Trust defines a request/response protocol
    - A client sends a **`RequestSecurityToken`** (RST) to the STS
    - The STS replies with a **`RequestSecurityTokenResponse`** (RSTR)

WS-SecureConversation

- WS-SecureConversation plays the same role in message-level security that SSL plays at the transport level
- WS-SecureConversation defines the creation and sharing of security contexts between communicating parties
  - The `<SecurityContextToken>` (SCT) element supports the requirements of security contexts
- An SCT involves a shared secret used to sign and/or encrypt messages
  - Derived keys are used for signing and encrypting messages associated with the security context
  - WS-SecureConversation defines how derived keys are computed and passed

WS-Policy

- WS-Policy enables one to specify policy information that can be used to access web services applications
- A policy is expressed as one or more policy assertions
- A policy assertion represents a capability or a requirement
  - For example, a policy assertion may stipulate that a request to a web service be encrypted, or a policy assertion can define the maximum message size that a web service can accept
- The meaning of each assertion is specific to a particular domain, for example, security, reliability, or privacy

WS-PolicyAttachment

- WS-PolicyAttachment defines how (WS-Policy) policies are attached to web services
    - Policies can be bound to WSDL or UDDI

```
<definitions>
...
  <binding name="StockQuoteWebServiceSoapBinding" ...>
    <wsp:PolicyReference xmlns:… URI="#SecureMessagePolicy"/>
  </binding>

  <wsp:Policy wsu:Id="SecureMessagePolicy"... >
    <sp:SignedParts>
      <sp:Body/>
    </sp:SignedParts>
    <sp:EncryptedParts>
      <sp:Body/>
    </sp:EncryptedParts>
  </wsp:Policy>
  ...
</definitions>
```

WS-MetadataExchange

- Defines how a client can request the metadata it needs to access and communicate with a web service endpoint
  - Metadata can be WSDL, WS-Policy, schema
- Uses WS-Addressing to identify endpoints
- WS-MetadaExchange works as follows:
  - A requester sends a GetMetadata request message to an endpoint
  - The endpoint replies with a GetMetadata response message including a reference to the metadata section requested

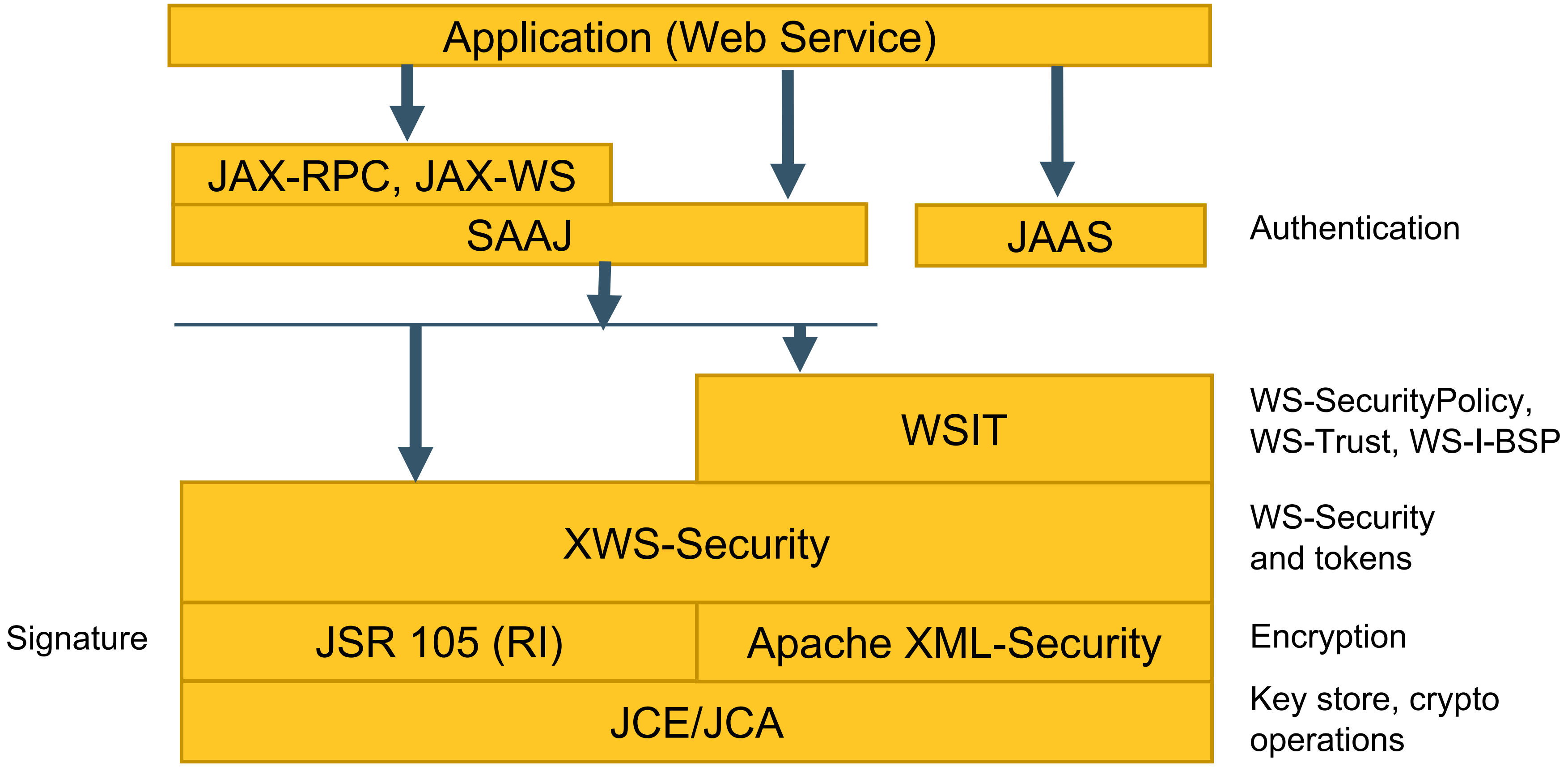
Agenda

- The Need for Security in SOA Environments
  - “Define Once, Enforce Anywhere”
- Paradigmatic Use Cases
  - SOA Environments
  - Web Applications
- **Key Industry Standards**
  - XML Frameworks
  - **Java Technology Frameworks**
- Implementation Choices
- Product Demo

Java Technology Frameworks

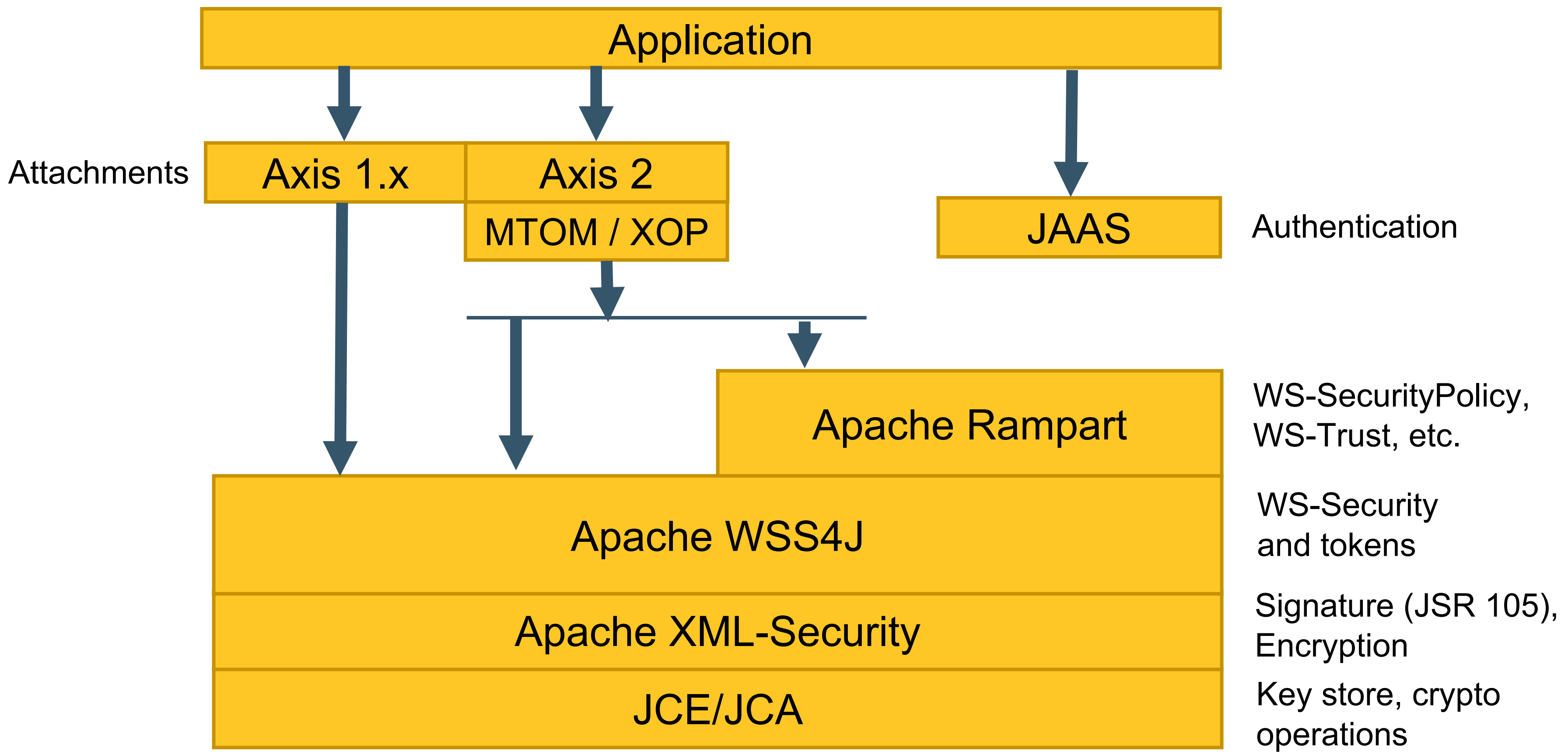
- Java Platform, Enterprise Edition (Java EE platform) 5
- Apache/WSO2
- Vendor specific (Oracle, Sun, BEA, etc.)

Java EE Platform 5



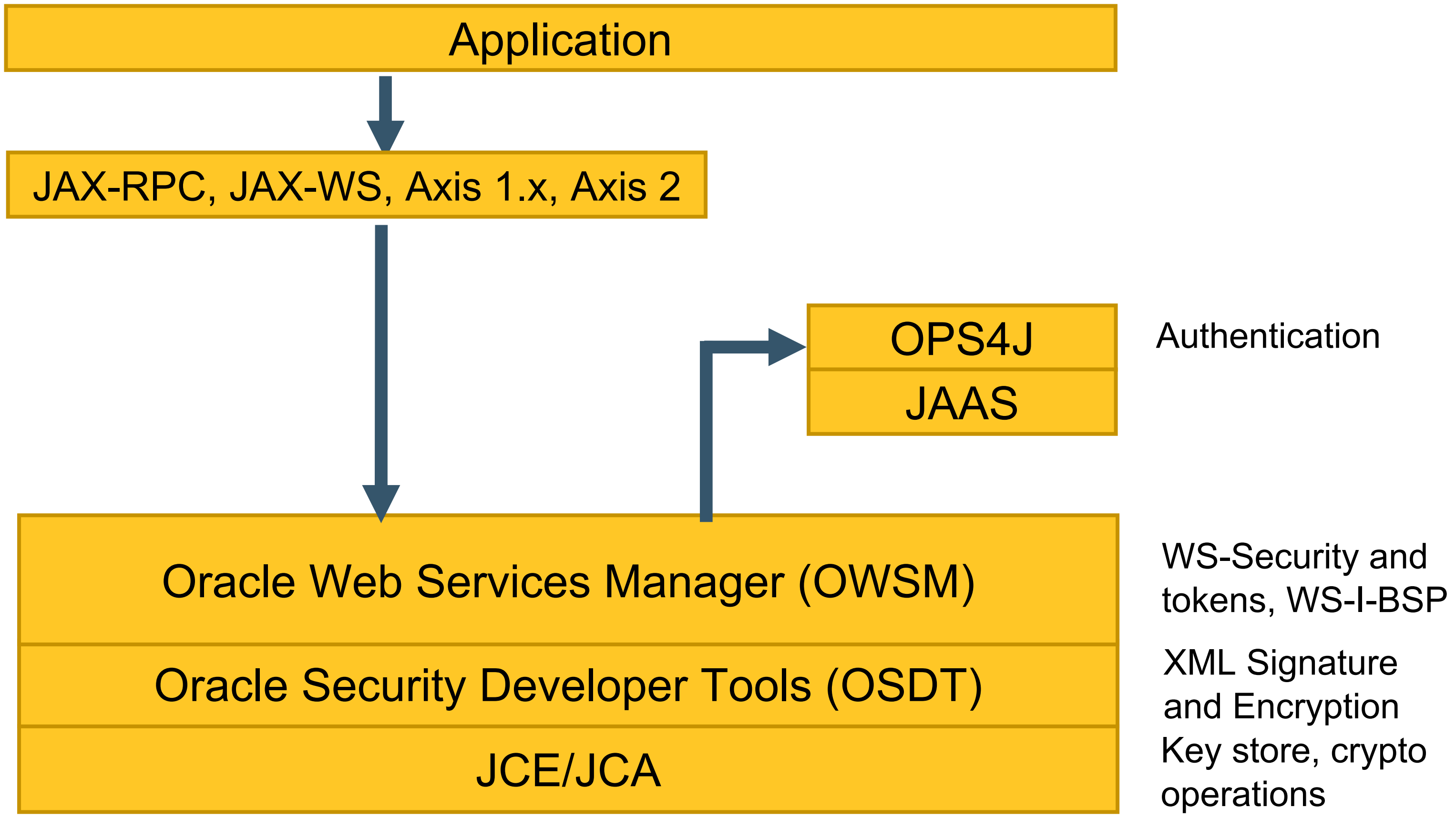
JAX-RPC = Java API for XML-based RPC | JAX-WS = Java APIs for XML Web Services
JSR = Java Specification Request | SAAJ = The SOAP with Attachments API for Java
JCE = Java Cryptography Extension | JCA = Java Cryptography Architecture

Apache/WSO2



Application
Attachments
Axis 1.x
Axis 2
MTOM / XOP
JAAS
Authentication
Apache Rampart
WS-SecurityPolicy, WS-Trust, etc.
Apache WSS4J
WS-Security and tokens
Apache XML-Security
Signature (JSR 105), Encryption
JCE/JCA
Key store, crypto operations

Vendor Specific (Oracle 11g)


Application
JAX-RPC, JAX-WS, Axis 1.x, Axis 2
OPS4J
JAAS
Authentication
Oracle Web Services Manager (OWSM)
Oracle Security Developer Tools (OSDT)
JCE/JCA
WS-Security and tokens, WS-I-BSP
XML Signature and Encryption
Key store, crypto operations

Agenda

The Need for Security in SOA Environments

- “Define Once, Enforce Anywhere”

Paradigmatic Use Cases

- SOA Environments
- Web Applications

Key Industry Standards

- XML Frameworks
- Java Technology Frameworks

Implementation Choices

Product Demo

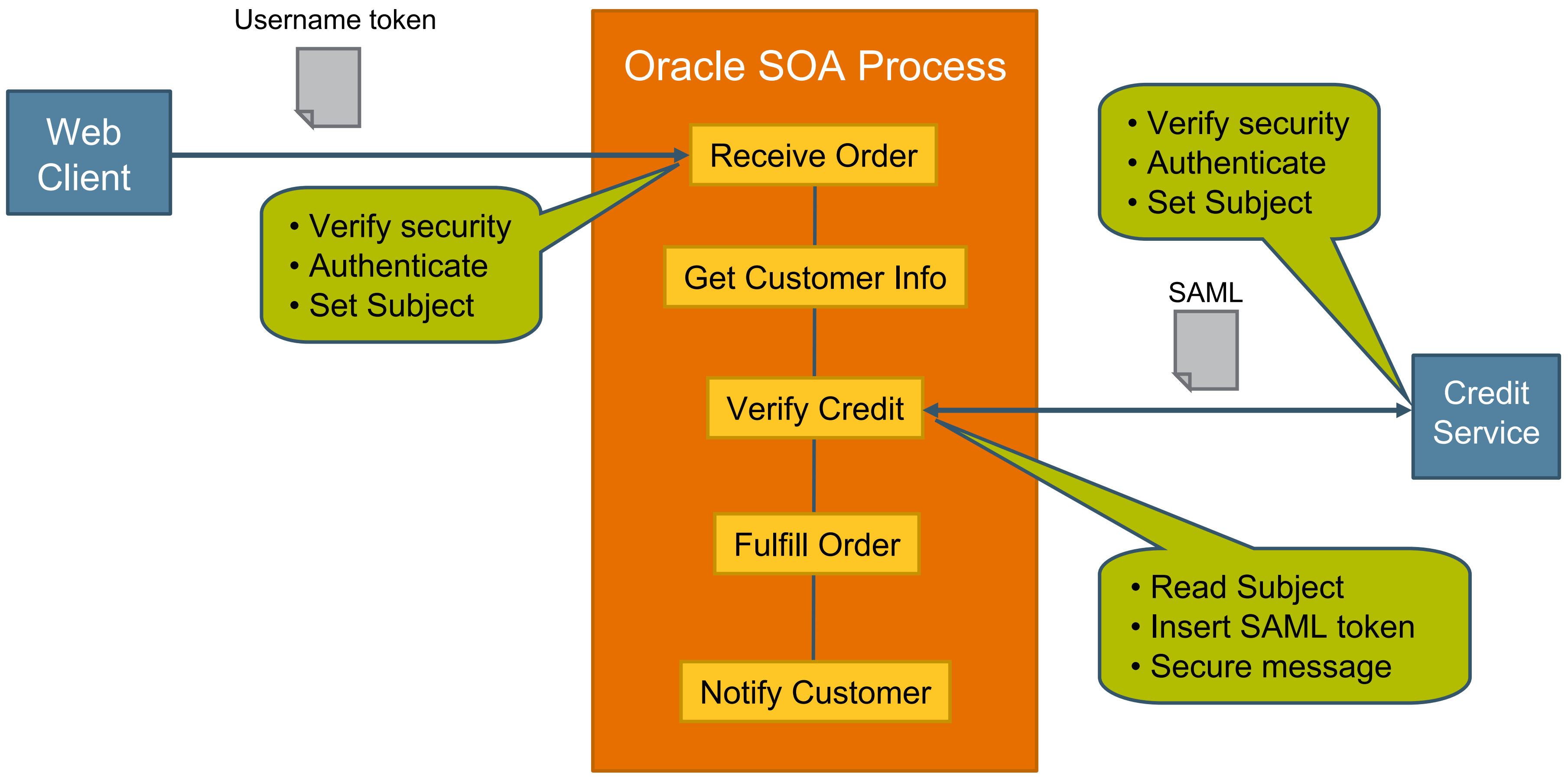
Application Server vs. External

- Security can be implemented in the application server or external to the application server
- Application Server security
  - Focused on a specific platform (Oracle, BEA, Sun, etc.)
- External security (Oracle WSM, XML Appliances, etc.)
  - Defined in a single policy manager
  - Enforced across heterogeneous platforms
  - Deployment flexibility
  - Monitoring capabilities

DEMO

Web Services Security

Demo Scenario



Summary

- SOA security is based on XML frameworks and Java technology standards
- Security includes authentication, authorization, integrity, confidentiality, trust
- SOA security should be externalized for flexible deployment and easier administration

For More Information

- OASIS Web Services Security (WSS) TC
  - http://www.oasis-open.org
- GlassFish XWSS
  - http://xwss.dev.java.net
- Oracle Technology Network
  - http://www.oracle.com/technology/products/middleware
- Blog
  - http://ws-security.blogspot.com

Q&A

Marc Chanliau and Vikas Jain

Java™ Technology and Web Services Security in Action

Marc Chanliau and Vikas Jain
Security Product Management

Oracle Corporation
www.oracle.com

TS-8131