INTEGRATE BUILD PORTAL

# Web Services Security

# Challenges

Hal Lockhart

Principal Engineering Technologist

Architecture and Standards

bea®

# Hal Lockhart

Principal Technologist, BEA Systems

Co-chair XACML TC

Editor WS Security TC Interop Specs

Vice Chair WS-I Basic Security Profile

Also Member: Security Services TC (SAML), Provisioning TC, Digital Signature Services TC, Web Services Management TC, Project Liberty, DCML Apps & Svcs TC

Represented OASIS at ITU-T re: SAML & XACML

bea

# Outline

## Overview of Web Services Security

- History
- Functionality

## Example Issues

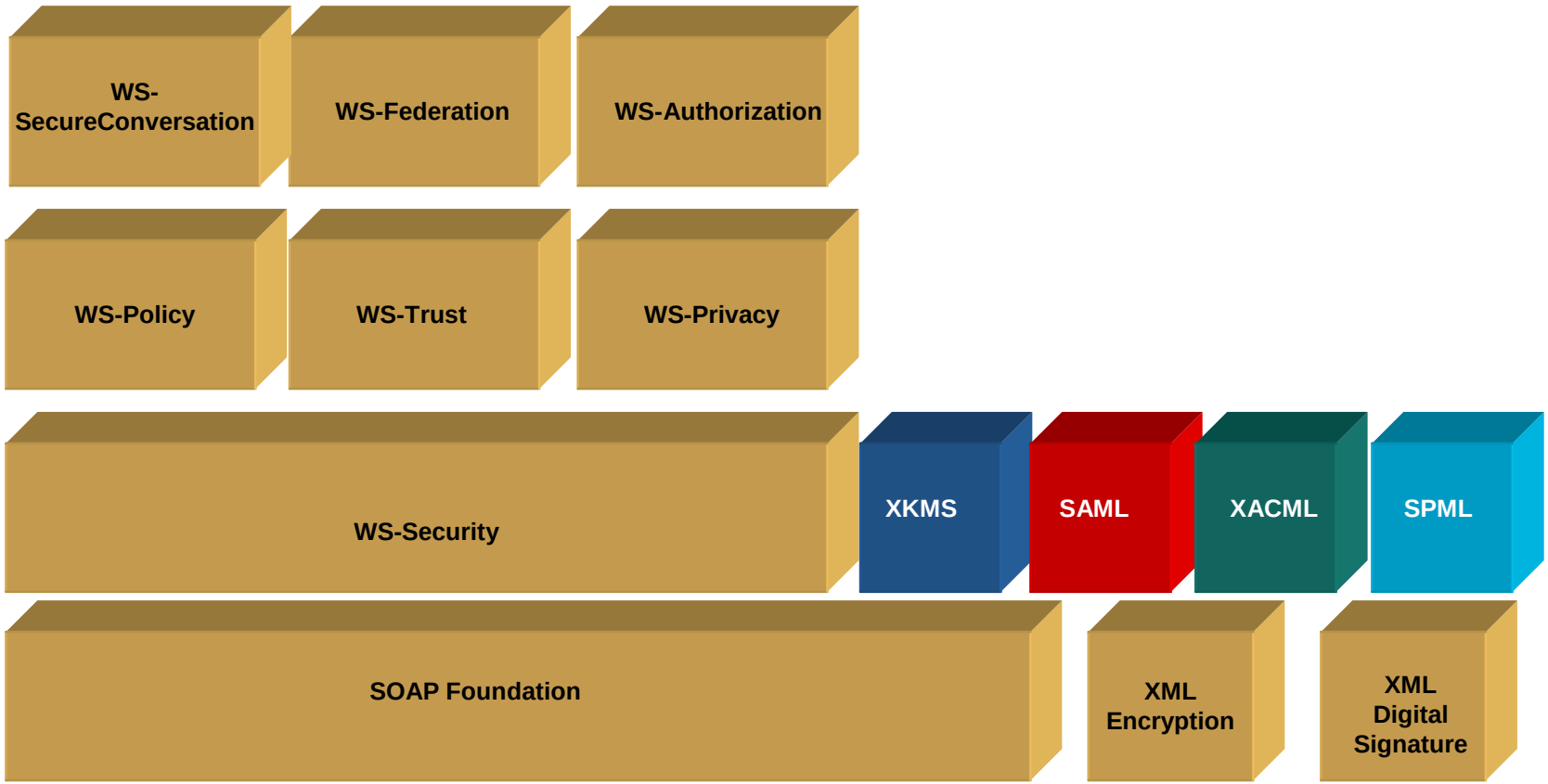
- Unexpected Properties – Encryption Key Substitution
- Scope Limitations – Canonicalization Algorithm Choice
- Composability and Layering

## Summary

bea

# Web Services Security Standards



# Web Services Security Summary

Submitted to OASIS September 2002

Interoperability testing Summer 2003

OASIS Standard in April 2004

- Core Specification + Username and X.509 Profiles

OASIS Standard December 2004

- SAML and REL Token Profiles

Attachments Profile is in public review

Kerberos Token Profile in process

WS-I Basic Security Profile

- Profiling OASIS specs as completed

bea

# Features in WSS Specification - 1

## Security Header

- Can contain mustUnderstand
- Can be addressed to Role

## Tokens

- Associated with signature or encryption or otherwise used to identify party to message exchange
- Binary Token - encapsulates binary object
  - X.509 certificate – defined by ITU/IETF
  - Kerberos ticket – defined by IETF/Microsoft
- XML Token – inserted as is
  - Username Token – defined by OASIS WSS TC
  - SAML Assertion – defined by OASIS SS TC
  - XrML License – defined by ContentGuard

bea

# Features in WSS Specification - 2

## Security Token Reference

- Points to or encapsulates a token
- Four types
  - Direct – URI or URI fragment
  - Key Identifier – specific to token type – identifies key, certificate, ticket, assertion, etc.
  - Key Name – identifies token by content, e.g. SubjectName
  - Embedded – encapsulates token, allows association of additional information with token

## Signature element

- New transform - STR Dereference Transform

## Encryption ReferenceList or EncryptedKey elements

## Timestamp element

- Only applies to security mechanisms
- Created and/or Expires

bea

# WSS TC Ongoing Work

## New in WSS 1.1

- Backward compatible with 1.0
- Encrypted SOAP Header
- Token Reference to Encrypted Key
- Signature Confirmation
- Password-based Key Derivation
- Thumbprint References
- Errata and Clarifications

## Kerberos Token Profile – Interoperability Testing

## Minimalist Profile

## Templates?

bea

# Web Services Security Issues

## Goal of specs is deliberately limited

- Many options allowed
- Impact on Interoperability and Security

## No description mechanism is defined currently

- WS-Policy to be standardized in 2005

## Use of intermediaries not well understood

- Therefore neither are security implications
- Necessary usage patterns my require private agreements

## No completely satisfactory C14N algorithm

- Essential for digital signatures
- More on this later

bea

# Outline

Overview of Web Services Security

- History
- Functionality

Example Issues

- Unexpected Properties – Encryption Key Substitution
- Scope Limitations – Canonicalization Algorithm Choice
- Composability and Layering

Summary

bea

# Understanding Public Key Operations

## A pair of related keys

- Public – In certificate
- Private – Kept secret by holder

## Encryption

- Encrypt using receiver's public key
- Receiver decrypts using private key

## Signature

- Sign using sender's private key
- Verify using public key



bea

# Example of Security “Ah Ha” From Interop Test Scenarios

## As done in Interop Test

- Simple request / response scenario
- Both messages to be signed and encrypted
- One key pair for each node
- Each signs with own key, encrypt with other party’s
- Certificates carried in SOAP Security Header
- Interoperable and secure

## Likely real-world variant

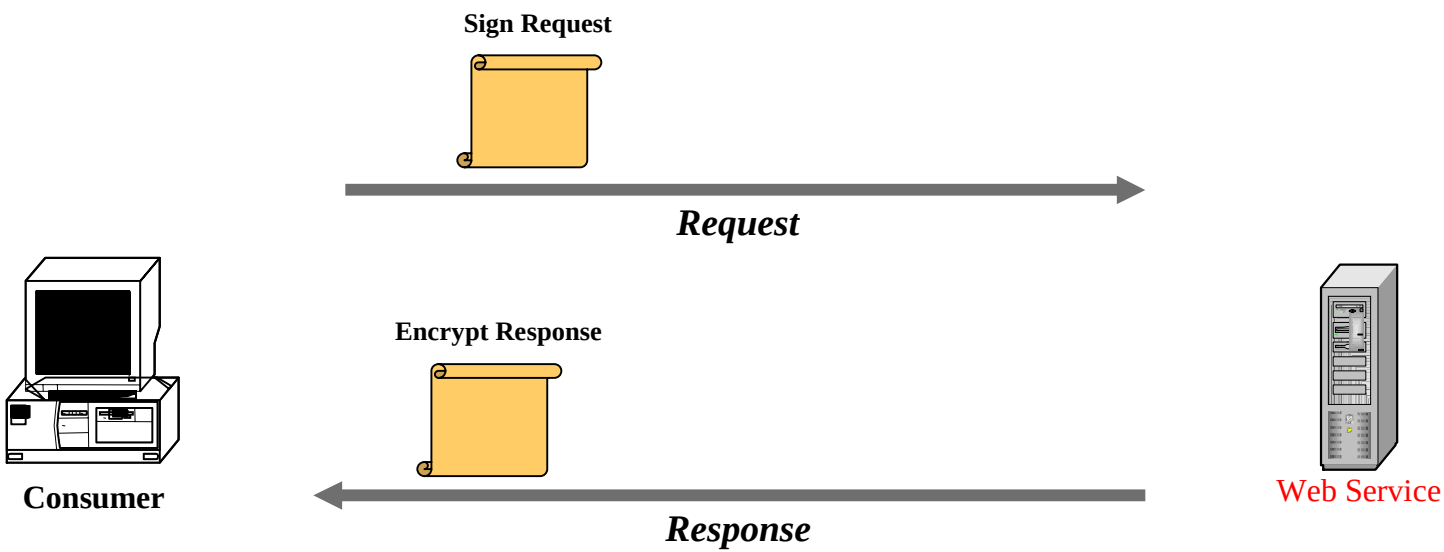
- Each node has two key pairs
- One for encryption, one for signature
- Best practice recommended by many authorities

## A threat appears from nowhere!

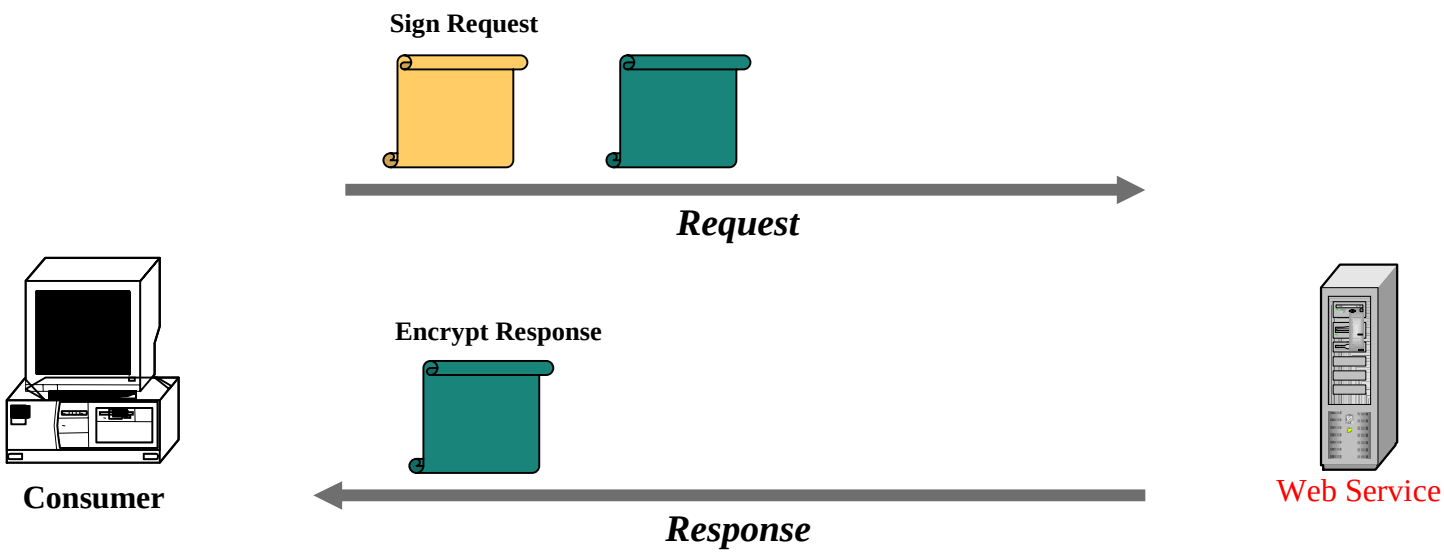
 

bea

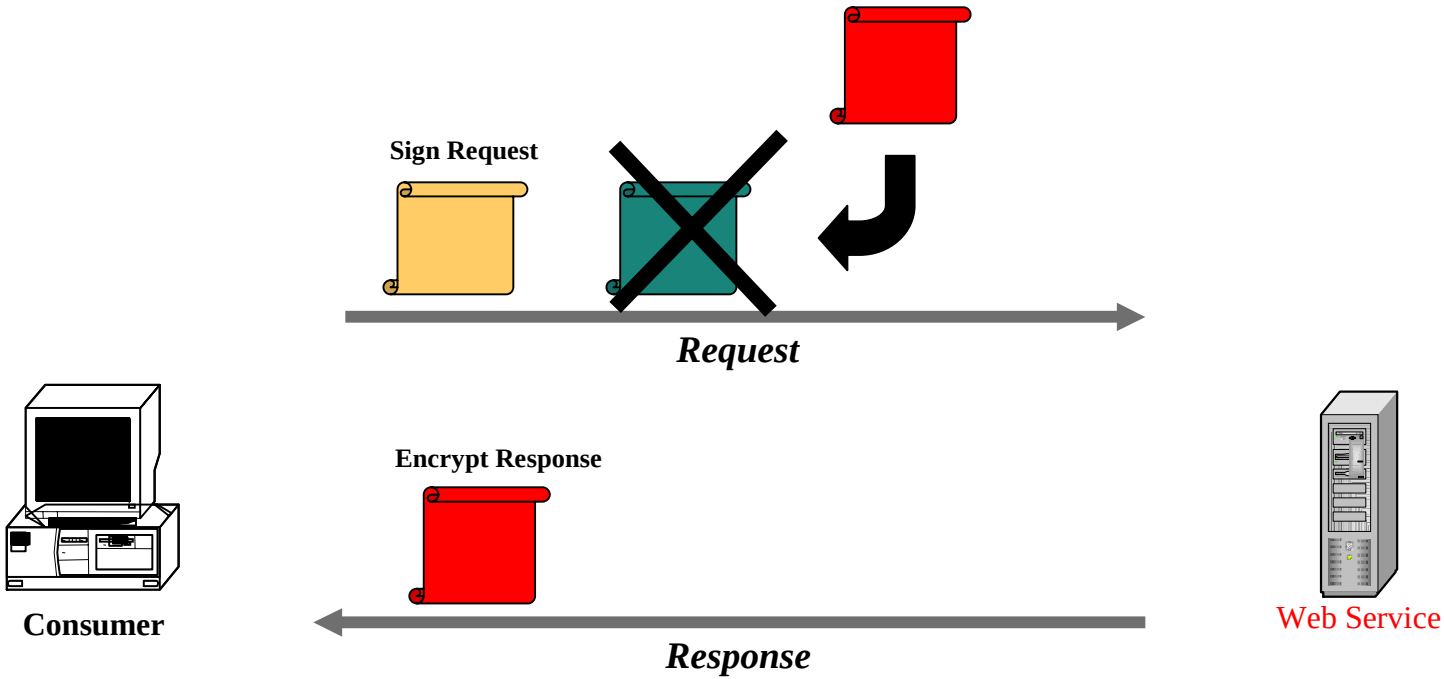
# Sign and Encrypt with One Key



# Sign and Encrypt with Two Keys



# Sign and Encrypt Attack



# WSS Templates

Document in detail most common usage patterns

Use of templates is optional

Organized by

- MEP, token type, services required

Based on Interoperability test document format

To be closely scrutinized manually and with tools

Proposed by me as work item in WSS TC

Initial version published soon

bea

# Outline

Overview of Web Services Security

- History
- Functionality

Example Issues

- Unexpected Properties – Encryption Key Substitution
- Scope Limitations – Canonicalization Algorithm Choice
- Composability and Layering

Summary

bea

# Canonicalization Choice Issue

## Problem: XML on the wire can change in ways that have no significance

## XML DSIG defines C14N Algorithms

- Compute C14N form, digest and sign, send orginal
- Receiver does the same
- Two algorithms – Inclusive & Exclusive
- Both deal with whitespace, redundant declarations, element ordering, etc.
- Inclusive includes ALL enclosing namespaces
- Exclusive includes only those which are “visibly used” plus an optional list provided by signer

bea

# Canonicalization Choice Issue

## Inclusive is “safe” - always includes all NS in use

## Very fragile

- Chunk of XML inserted in other document
- Even if not moved, auto generated declarations in ancestor (e.g. xsi:type)

## Exclusive can miss NS declarations

- Can change semantic of data (standard:paymentTerms)
- Ok, if you know NS being used, unrealistic in WS enviroment
- Can still fail in other cases
- Insertion may not work for other reasons, e.g. Id name conflicts

bea

# Outline

Overview of Web Services Security

- History
- Functionality

Example Issues

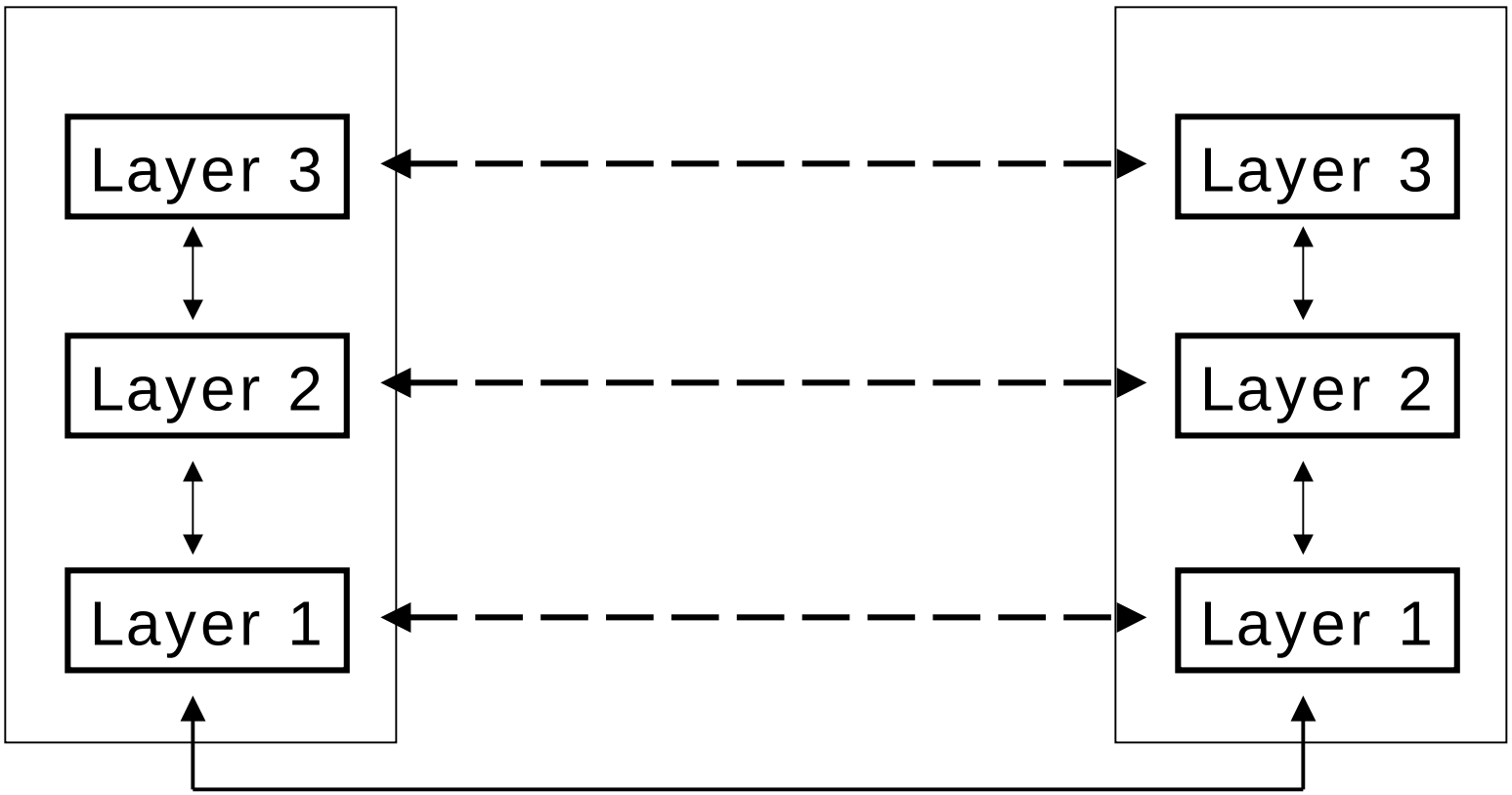
- Unexpected Properties – Encryption Key Substitution
- Scope Limitations – Canonicalization Algorithm Choice
- Composability and Layering

Summary

bea

# Layering



# Layering Principles

A form of encapsulation

Corresponding layers communicate via peer protocol

Messages pass through all active layers

Guidelines

- Layers must operate sequentially
- Layers must operate on distinct data
- Layers should not duplicate each other

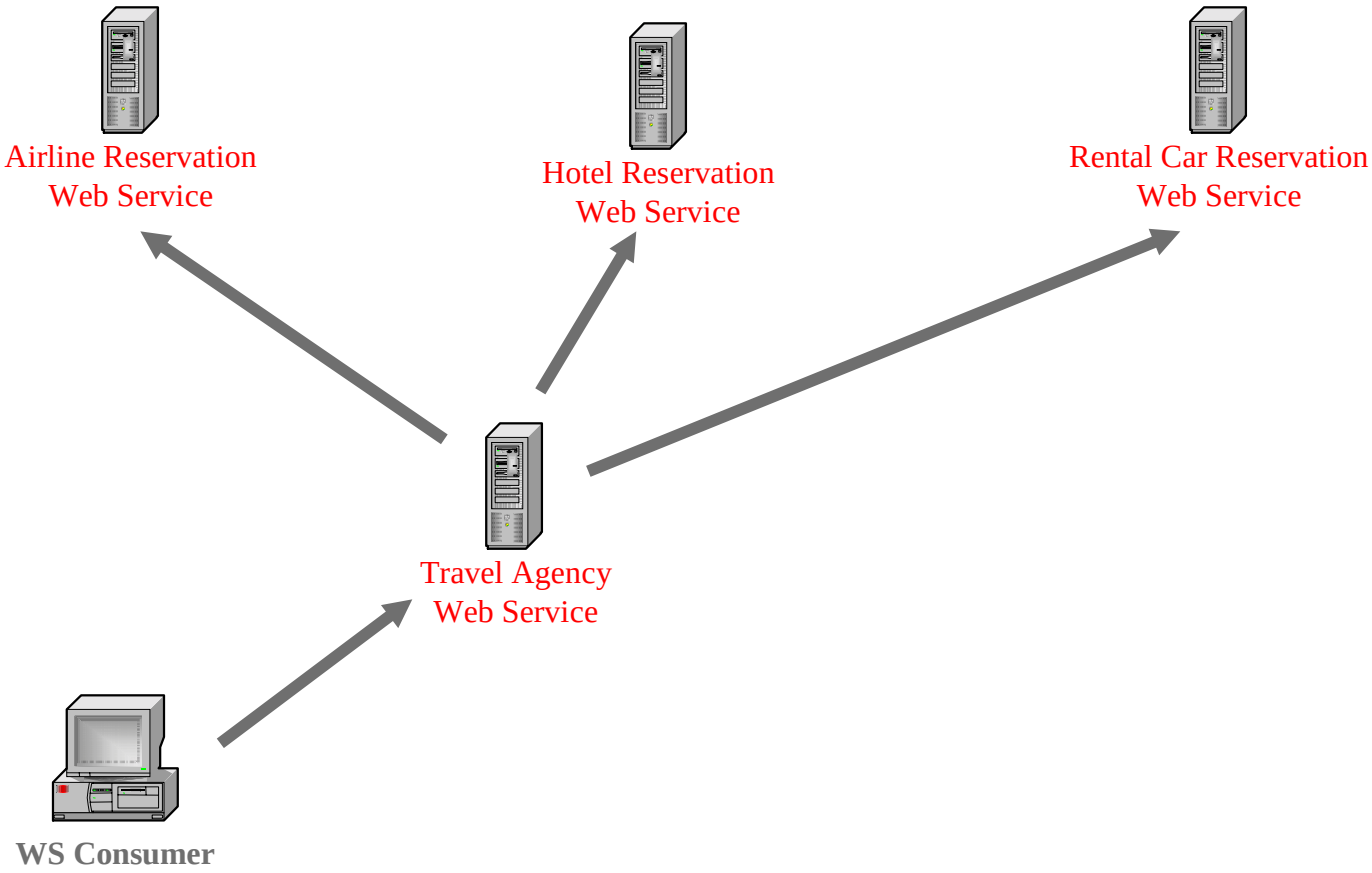
Composability

- Stronger condition
- Layers may be omitted

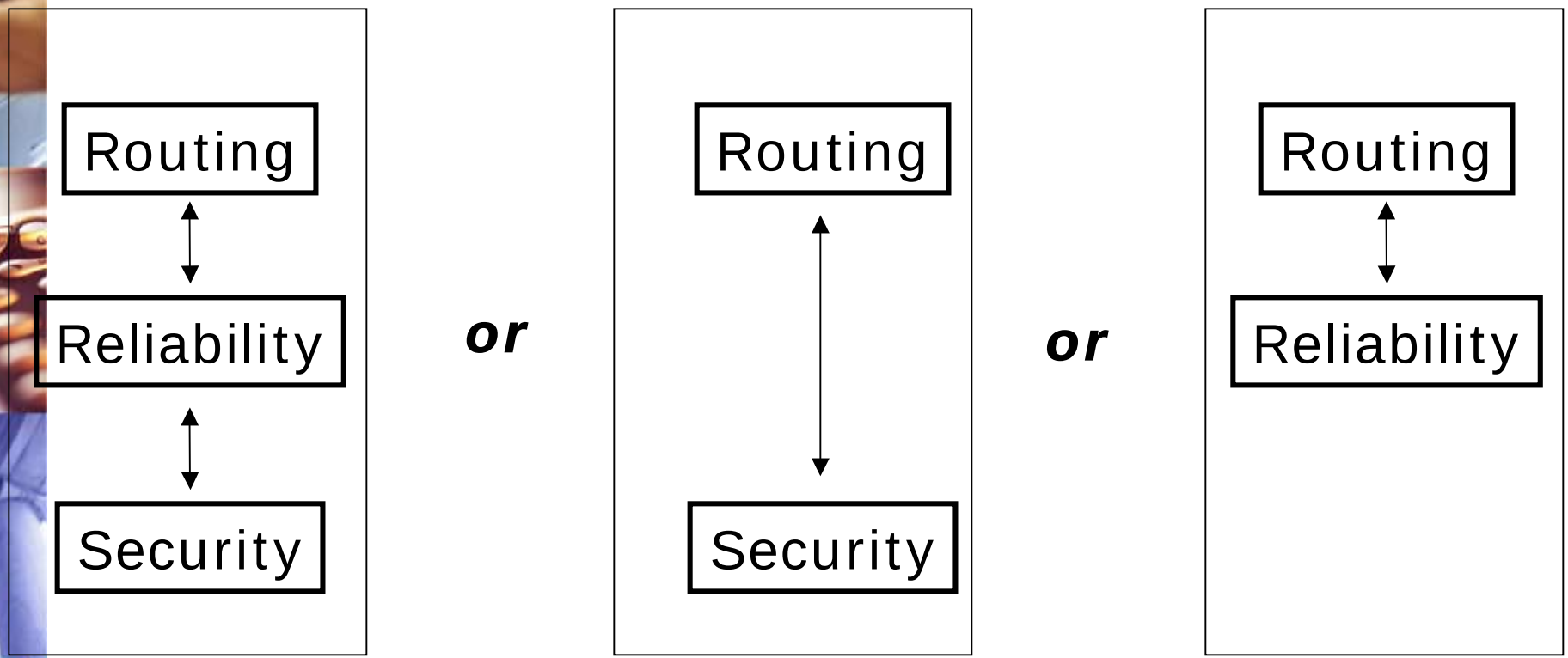
bea

# Composable Services



# Composable Layers



# Combining Security & Reliability

## Security should be below Reliability

- Discard invalid messages
- Reliability (if present) will retransmit
- SSL/TLS are “broken” in this regard

## Security should be below other layers

- Validate signatures
- Decrypt data
- Issue: must pass along metadata with message (e.g. what was signed, who was authenticated)

## Security & Reliability could be intertwined

- Loss of composibility

bea

# WS-ReliableMessaging

## Proposed by Microsoft, IBM, et al.

- (Not to be confused with WS-Reliability)

## <CreateSequence> may contain Security Token Ref.

- Indicates Token used for signature with reliability semantic
- Intended for use with WS-SecureConversation

## Violates layering

- Reliability layer must know what token will be used
- Security must sign <CreateSequence> using token

## Possible solution

- Use somewhat generic references
- Difficult to provide full generality with WS-SecureConversation

bea

# Summary

WSS is nearly complete

WSS provides useful basic functionality

WSS is a toolkit not a protocol

- Can have unexpected properties
- Templates may help

WSS leaves some problems out of scope

- Users need to be aware of implications of C14N
- Don’t use Qnames in content

Composability and clean layering are important

- Not a slam dunk
- Pressures to violate

bea

INTEGRATE BUILD PORTAL

www.bea.com

bea®