Middleware and Web Services
Security Mechanisms

Secure Application Development
Module 9
Konstantin Beznosov
Conventional Computer Security

<table>
<thead>
<tr>
<th>Protection</th>
<th>Assurance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>Requirements Assurance</td>
</tr>
<tr>
<td>Accountability</td>
<td>Design Assurance</td>
</tr>
<tr>
<td>Availability</td>
<td>Development Assurance</td>
</tr>
<tr>
<td>Access Control</td>
<td>Operational Assurance</td>
</tr>
<tr>
<td>Data Protection</td>
<td></td>
</tr>
<tr>
<td>Audit</td>
<td></td>
</tr>
<tr>
<td>Non-Repudiation</td>
<td></td>
</tr>
<tr>
<td>Service Continuity</td>
<td></td>
</tr>
<tr>
<td>Disaster Recovery</td>
<td></td>
</tr>
<tr>
<td>Authentication</td>
<td></td>
</tr>
<tr>
<td>Cryptography</td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Middleware and Web services
  - What are middleware and Web services?
  - What’s special about middleware and Web services security?
- Security in middleware and Web services
  - What are common architectures for security mechanisms in most middleware and Web service technologies?
  - What are the differences among security mechanisms of COM+ and EJB?
- Conclusions
  - Summary
  - Where to go from here?
What is middleware?

It’s what’s between topware and underwear.
Distributed Application Built Using DOS
Distributed Application Built Using NOS

Distributed Application Built Using Middleware

# Software Support for Distributed Applications

<table>
<thead>
<tr>
<th>System</th>
<th>Description</th>
<th>Main Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS</td>
<td>Tightly-coupled operating system for multi-processors and homogeneous multicomputers</td>
<td>Hide and manage hardware resources</td>
</tr>
<tr>
<td>NOS</td>
<td>Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)</td>
<td>Offer local services to remote clients</td>
</tr>
<tr>
<td>Middleware</td>
<td>Additional layer atop of NOS implementing general-purpose services</td>
<td>Provide distribution transparency</td>
</tr>
</tbody>
</table>
Most Middleware Uses Remote Procedure Call (RPC)

1. Client call to procedure
2. Stub builds message
3. Message is sent across the network
4. Server OS hands message to server stub
5. Stub unpacks message
6. Stub makes local call to "add"

RPC Clients and Servers

Distributed Objects

- Distributed Computing Environment (DCE) Remote Objects
- Common Object Request Broker Architecture (CORBA)
- Microsoft’s Distributed Component Object Model (DCOM) & COM+
- Java Remote Method Invocation (RMI)
- Enterprise Java Beans (EJB)
- .NET Remoted Objects
Middleware Services

- Communication facilities
- Naming
- Persistence
- Concurrency
- Distributed transactions
- Fault tolerance
- Security
Middleware Openness

What’s Web Services?
How do middleware and Web services differ?

<table>
<thead>
<tr>
<th>Features/properties</th>
<th>middleware</th>
<th>Web services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>traditional</td>
<td>MOM</td>
</tr>
<tr>
<td>Client server</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>RPC</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>OS independent</td>
<td>mostly</td>
<td>mostly</td>
</tr>
<tr>
<td>Completeness and portability</td>
<td>yes</td>
<td>mostly</td>
</tr>
<tr>
<td>interoperability</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
Promise of Web Services

- Interoperability across lines of business and enterprises
  - Regardless of platform, programming language and operating system
- End-to-end exchange of data
  - Without custom integration
- Loosely-coupled integration across applications
  - Using Simple Object Access Protocol (SOAP)
Web Services Features

XML-based messaging interface to computing resources that is accessible via Internet standard protocols

- WS help intranet (business units) and extranet (business partners) applications to communicate
- SOAP – format for WS communications
  - Defined in XML
  - Supports RPC as well as document exchange
    - No predefined RPC semantics
  - Stateless
  - Can be sent over various carriers: HTTP, FTP, SMTP, ... postal service
<?xml version="1.0" ?>
<env:Envelope xmlns:env="http://www.w3.org/2002/06/soap-envelope">
  <env:Header>
    <n:alertcontrol xmlns:n="http://example.org/alertcontrol">
      <n:priority>1</n:priority>
      <n:expires>2001-06-22T14:00:00-05:00</n:expires>
    </n:alertcontrol>
  </env:Header>
  <env:Body>
    <m:alert xmlns:m="http://example.org/alert">
      <m:msg>Pick up Mary at school at 2pm</m:msg>
    </m:alert>
  </env:Body>
</env:Envelope>
Typical Web Service Environment

- **Client Tier**: Application Client, Browser, Java Program
- **Presentation Tier**: Web Servers
- **Component Tier**: J2EE, CORBA, ASP.NET, COM
- **Back-Office Tier**: Mainframes, Databases

Communication Protocols:
- SOAP/HTTP
- HTML/HTTP
- RMI/DCOM
J2EE Web Service Systems

Client Tier
- Application Client
- Browser

Presentation Tier
- Web Server
- JSPs
- Servlets
- HTML

Component Tier
- EJB Container
- EJB
- EJB
- RMI-IIOP

Back-office Tier
- Mainframes
- JDBC
- JCA
- Databases

Communication Protocols:
- SOAP
- RMI-IIOP
- HTML
Outline

- Middleware and Web services
  - What are middleware and Web services?
  - What’s special about middleware and Web services security?
- Security in middleware and Web services
  - What are common architectures for security mechanisms in most middleware and Web service technologies?
  - What are the differences among security mechanisms of COM+ and EJB?
- Conclusions
  - Summary
  - Where to go from here?
requirements due to distribution

- centralized administration
- localized run-time decisions
Interface `Course` with methods
- `postMaterials` (Materials m, Module module)
- `Materials getMaterials` (Module module )
- `submitAssignment` (Assignment a)
- `Assignment getAssignment` (Student student, int number )
- `postAssignmentInstructions` (Instructions i, int number)
object paradigm & security (1/2)

- objects
  - small amounts of data $\implies$ large numbers
    - R: Scale on large numbers of objects and methods
  - diverse methods $\implies$ complex semantics
    - R: Security administrators should not have to understand semantics of methods

- collections
  - R: Similar names or locations should NOT impose membership in same collection(s).
  - R: For an object to be assigned to the same collection, name similarity and/or co-location should not be required.
Object paradigm & Security (2/2)

- Many layers of indirection and late binding
- Names
  - Multi-name, nameless, and transient objects
    - R: Transient objects should be assigned to security policies without human intervention.
  - Less rigid naming hierarchies
  - R: No assumptions that administrators know a name of each object in the system.
Outline

- Middleware and Web services
  - What are middleware and Web services?
  - What’s special about middleware and Web services security?
- Security in middleware and Web services
  - What are common architectures for security mechanisms in most middleware and Web service technologies?
  - What are the differences among security mechanisms of COM+ and EJB?
- Conclusions
  - Summary
  - Where to go from here?
Middleware Security Stack

Client

- Application
- Proxy
- ORB
- Security Service
- Security Mechanism Implementation
- OS
- Network

Server

- Application
- Skeleton
- Adapter
- ORB
- Security Service
- Security Mechanism Implementation
- OS
- Network

RPC Abstraction

Middleware Security

security context abstraction

Actual messages

Actual messages
Policy Enforcement and Decision

- Target
- Enforcement Function
- Decision Function
- Decision Request
- Middleware Security Subsystem
- Middleware
- Access Request
Distributed Authentication

- Password-based
- Symmetric key
  - e.g., Kerberos
- Asymmetric key
  - e.g., PKI
Data Protection

Client

Application
Proxy
ORB
Security Service
Security Mechanism Implementation
OS
Network

RPC Abstraction

Middleware Security
security context abstraction
Actual messages

Server
Application
Proxy
ORB
Skeleton
Adapter
ORB
Security Service
Security Mechanism Implementation
OS
Network
Data Protection in Web Services
SOAP Message with WS-Security

```xml
<? Xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2001/12/soap-envelope"
    xmlns:sig="http://www.w3.org/2000/09/xmldsig#"
    xmlns:enc="http://www.w3.org/2001/04/xmlenc#">
  <env:Header>
    <sec:Security
      sec:actor="http://www.w3.org/2001/12/soap-envelope/actor/next"
      sec:mustUnderstand="true">
      <sig:Signature>
        ...
      </sig:Signature>
      <enc:EncryptedKey>
        ...
      </enc:EncryptedKey>
      <sec:BinarySecurityToken>
        ...
      </sec:BinarySecurityToken>
    </sec:Security>
  </env:Header>
  <env:Body>
    <enc:EncryptedData>
      ...
    </enc:EncryptedData>
  </env:Body>
</env:Envelope>
```
WS-Security

- Message integrity and message confidentiality
- Compliance with XML Signature and XML Encryption
- Encoding for binary security tokens
  - Set of related claims (assertions) about a subject
  - X.509 certificates
  - Kerberos tickets
  - Encrypted keys
XML Encryption

- Encrypt all or part of an XML message
- Separation of encryption information from encrypted data
- Super-encryption of data

```
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
  Type='http://www.w3.org/2001/04/xmlenc#Content'>
  <EncryptionMethod Algorithm='http://www.w3.org/2001/04/xmlenc#3des-cbc'/>
  <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'>
    <ds:KeyName>John Smith</ds:KeyName>
  </ds:KeyInfo>
  <CipherData>
    <CipherValue>A23B45C56</CipherValue>
  </CipherData>
</EncryptedData>
```
XML Signature

- Apply to all or part of a document
- Contains: references to signed portions, canonicalization algorithm, hashing and signing algorithm IDs, public key of the signer.
- Multiple signatures with different characteristics over the same content

```xml
<Signature Id="MySignature" xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
    <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>j6lw3rvEPO0vKtMup4NbeVu8nk=/</DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>MC0CFFrVltRlk=...</SignatureValue>
  <KeyInfo>
    <KeyValue>
      <DSAKeyValue>
        <P>...</P><Q>...</Q><G>...</G><Y>...</Y>
      </DSAKeyValue>
    </KeyValue>
  </KeyInfo>
</Signature>
```
Security Policy Decisions
Policy Enforcement and Decision

Middleware
Middleware Security
Subsystem

Access Request

Application
Target

Access Request

Enforcement Function
Decision
Function
Decision
Function
Decision

Access Request

Middleware

Decision Request

Decision

Decision Request
scaling policy decisions

- Attributes
- Permissions
- Domains, types

Clients

Methods

Subjects

Targets
Credentials Delegation

- What are credentials?
- Push and pull models
- No delegation

Client \( \rightarrow \text{client credentials} \) Intermediate Object \( \rightarrow \text{intermediate credentials} \)Target Object

- Simple delegation: impersonation or controlled

Client \( \rightarrow \text{client credentials} \) Intermediate Object \( \rightarrow \text{client credentials} \) Target Object

- Composite delegation

Client \( \rightarrow \text{client credentials} \) Intermediate Object \( \rightarrow \text{client & intermediate credentials} \) Target Object

- Also: combined privileges, traced delegation
Issues in Distributed Audit

- Monitor activity across and between objects.
- Order of the audit records is hard to determine because of the lack of global time.
- Performance
- No guarantee that an event has been logged.
Outline

- Middleware and Web services
  - What are middleware and Web services?
  - What’s special about middleware and Web services security?

- Security in middleware and Web services
  - What are common architectures for security mechanisms in most middleware and Web service technologies?
  - What are the differences among security mechanisms of COM+ and EJB?

- Conclusions
  - Summary
  - Where to go from here?
COM+ Specifics
Authentication in COM+

- **Supported mechanisms**
  - Kerberos
  - Windows NT LAN Manager (NTLM)

- **Granularity modes**
  - Never
  - At the time of establishing secure channel
  - On every call
  - With every network packet

- **Credentials delegation options**
  - No delegation
  - Unconstrained simple delegation (a.k.a., impersonation)
    - Only one hop for NTLM
Data Protection in COM+

- Supported modes
  - Origin authentication and integrity protection
  - As above + confidentiality protection
Access Control in COM+

- The three hurdles to go through
  1. Activate server process
  2. Process border checks
  3. DLL border checks

- Granularity
  - Component
  - Interface
  - Method
Administering Access Control
COM+ Access Control Architecture

- Windows Machine
  - hosts
  - COM+ Application
    - contains
      - role
        - is granted access to
          - component
            - implements
              - Interface
                - defines
                  - method
                    - is granted access to
                      - account
                        - group
Application Description

Application:

- 10 students: s₁, ..., s₁₀
- 3 instructors: i₁, i₂, i₃
- 5 courses: c₁, ..., c₅
  - C₁ = {i₁, {s₁, s₂, s₃}}
  - C₂ = {i₂, {s₃, s₄, s₅}}
  - C₃ = {i₃, {s₅, s₆, s₇}}
  - C₄ = {i₁, {s₇, s₈, s₉}}
  - C₅ = {{i₂, i₃}, {s₈, s₉, s₁₀}}

Policy:

1. Students can
   1. read course material and assignment instructions for the courses they are registered
   2. submit (i.e., write) their assignments for the registered courses

2. Instructors can
   1. read student submitted assignments for the courses they teach, and
   2. post (i.e., write) course material and assignment instructions for their courses

Configure COM+ online course application to implement this policy
A Possible Solution

- **Interface Course with methods**
  - `postMaterials (CourseId id, Materials m, Module module)`
  - `getMaterials (CourseId id, Module module)`
  - `submitAssignment (CourseId id, int assignmentNumber)`
  - `getAssignment (CourseId id, Student student, int number)`
  - `postAssignmentInstructions (CourseId id, Instructions i, int number)`

<table>
<thead>
<tr>
<th></th>
<th>student</th>
<th>instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>postMaterials</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>getMaterials</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>submitAssignment</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>getAssignment</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>postAssignmentInstructions</td>
<td></td>
<td>+</td>
</tr>
</tbody>
</table>
Accountability in COM+

- No out-of-the-box support
- Developers should rely on Windows event logs
EJB Specifics
EJB Run–time Security

Common Secure Interoperability (CSI) v2 defines wire protocol
Authentication in EJB

- Defines only the use of JAAS for authenticating and credentials retrieving
- Implementation-specific
- Credentials delegation options
  - No delegation
  - Unconstrained simple delegation (a.k.a., impersonation)
Data Protection in EJB

- Implementation-specific
Access Control in EJB

- Configured through deployment descriptor
- Granularity
  - Down to individual method on a class, but not bean instance
  - Can be different from JAR to JAR
- Expressiveness
  - method grouped into “method permissions”
  - Subjects grouped by plain roles
  - No role hierarchy
- Java Authorization Contract for Containers (JACC)
  - APIs for plugging authorization engines
Defining Roles in EJB

<assembly-descriptor>
  <security-role>
    <description>
      blah-blah-blah ...
    </description>
    <role-name>student</role-name>
  </security-role>

  <security-role>
    <description>
      blah-blah-blah ...
    </description>
    <role-name>instructor</role-name>
  </security-role>

...</assembly-descriptor>
Assigning Users to Roles in EJB

<security-role-mapping>
  <role-name>student</role-name>
  <principal-name>S1</principal-name>
  <principal-name>S2</principal-name>
  <group-name>students</group-name>
</security-role-mapping>

<security-role-mapping>
  <role-name>instructor</role-name>
  <principal-name>I1</principal-name>
</security-role-mapping>
Assigning Methods to Roles in EJB

```
<method-permission>
  <role-name>student</role-name>
  <method>
    <ejb-name>Course</ejb-name>
    <method-name>getMaterials</method-name>
    <method-name>submitAssignment</method-name>
  </method>
</method-permission>

<method-permission>
  <role-name>instructor</role-name>
  <method>
    <ejb-name>Course</ejb-name>
    <method-name>postMaterials</method-name>
    <method-name>getAssignment</method-name>
  </method>
</method-permission>
```
roles and permissions in EJB
Fine-grain authorization in EJB

isCallerInRole(role)

security-role

role-link

role-name

<<implicit>>

internal role

Container

Application

entity

hosts

security-role-ref

0..1

0..*

0..*

0..*

0..*

0..1

0..*

0..*

0..*

0..*

0..*

0..1

0..*

0..*

0..*

0..*
Accountability in EJB

- Implementation-specific
Summary

- Middleware & Web services
  - Software layer between OS and application to provide transparencies
  - Security-related issues: scaling, granularity, naming
- Security in Middleware & Web services
  - Common features/elements
  - Technology/product specific
Where To Go From Here?

- “Security Engineering ...” by Ross Anderson