Introduction to CORBA Security

Belinda Fairthorne, ICL
Bob Blakley, IBM
Agenda

• Introduction to features, requirements, scope
• Outline of the security model
  – Security facilities for secure ORBs
  – Security policy administration
  – Application facilities
  – Interoperability
  – Implementation and trust implications

Including why certain choices made
Key Security Features

Security

Confidentiality  Integrity  Availability  Accountability

Authentication  Access Control  Secure Communications  Security Auditing  Non-repudiation

Administration of security policies
Requirements

• Focus on distributed object systems
  – including large scale ones on heterogeneous systems

• Usability
  – application developers should not generally need to be security aware

• Range of commercial and government use
  – choice of security policies, mechanisms
  – different priorities: security v cost v performance

• International
  – so must be capable of meeting government regulations
Scope of CORBA Security Specification

• Core security facilities, interfaces & protocols for
  – applications
  – security policy administration
  – implementation

• Supporting descriptions
  – security model and architecture
  – guidelines for trustworthy systems

• Not
  – specific assurance level, response to one specific set of threats
  – particular security mechanisms, crypto (though CSI defines more)
  – interoperability between unlike domains
CORBA Security Conformance

- Several conformance levels defined
  - level 1: entry level for security unaware applications
  - level 2: adds application facilities and policy administration
  - non-repudiation separate optional facility

- Security replacability
  - allows security services used with ORB to be replaced
  - also interceptors (those these CORBA Core extension)

- Secure Interoperability
  - IOR and security enhancements to CORBA 2 GIOP/IIOP
Security Model

User

Client

Client application security controls

request

reply

ORB

Target Object

target application security controls

request

reply

Client-side security controls

Target-side security controls
User logon to object invocation

- Authentication results in credentials for the user
  - logon gives default credentials in current environment
    - contains security attributes - identity, role etc
  - logon can be outside object environment, shared with other systems
Object Invocation

Client

ORB

Target Object

request

request

client-side security
security association establishment,
access control, audit,
message protection

target-side security
security association establishment,
access control, audit,
message protection
Security at Object Invocation

• Security controls can be used at client and/or target
• Access control
  – is this client allowed to do this operation on this object?
• Security “associations” between client and target
  – establish trust between client and target
  – propagate principal’s (user’s) attributes
• Message protection (integrity, confidentiality)
• Audit what happened (if required)
Managing Security Policy

• There are different types of security policies
  – for access control, audit, message protection etc

• Objects belong to security policy domains
  – Policy objects in domain manage/enforce policies
    • administrative interfaces for managing policy
    • enforcement interfaces automatically invoked by the ORB

• Policy objects are replaceable
  – to allow choice of policy

• Standard policies defined e.g. for access control
Security Policy Domains

- Security policy management, not domain management
  - c.f. CORBA management specification
Example of AccessPolicy
- using the standard DomainAccessPolicy

• Bank branch
  – bank employees
    • bank manager
    • bank clerk
  – bank branch objects
    • accounts

User’s Privilege Attributes
- from credentials

<table>
<thead>
<tr>
<th>Principal</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>role = bank manager</td>
</tr>
<tr>
<td>Joe</td>
<td>role = bank clerk</td>
</tr>
</tbody>
</table>

• Access policy per bank branch (domain)
  – specifies who has what rights to access which objects
Access Policy Example (2)

<table>
<thead>
<tr>
<th>Principal</th>
<th>Role</th>
<th>Effective Rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>role = bank manager</td>
<td>“get”, “set”</td>
</tr>
<tr>
<td>Joe</td>
<td>role = bank clerk</td>
<td>“get”</td>
</tr>
</tbody>
</table>

The policy specifies that
- a user (principal) with the specified role (privileges)
- has the specified effective rights in this domain

Use of standard rights are recommended
- get, set, manage
### Access Policy Example (3)

#### User’s Privilege Attributes

<table>
<thead>
<tr>
<th>Principal</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>role = bank manager</td>
</tr>
<tr>
<td>Joe</td>
<td>role = bank clerk</td>
</tr>
</tbody>
</table>

#### Access Policy for domain

<table>
<thead>
<tr>
<th>Role</th>
<th>Effective Rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>bank manager</td>
<td>get, set</td>
</tr>
<tr>
<td>bank clerk</td>
<td>get</td>
</tr>
</tbody>
</table>

#### Required rights for Bank Accounts

<table>
<thead>
<tr>
<th>Right</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>read_account_value, see_credit_status</td>
</tr>
<tr>
<td>set</td>
<td>credit_account, debit_account</td>
</tr>
</tbody>
</table>
Invocation Audit Policies

• Choice of events to audit selected by:
  – Event type e.g. object invoked, access check done
  – Success or failure of event
  – Object, or object type
  – Operation
  – Principal on whose behalf the object was invoked
  – Time

• So could audit:
  – Invocations on bank accounts in this branch by clerks using set_account_value after 5pm
Delegation

- A call on an object results in a chain of calls
- Pass on initiating or intermediate principal’s attributes?
  - delegation policy specifies which
    (when don’t trust application to pass on principal’s attributes)
**Delegation Options (1)**

- **No delegation**
  - Client → Intermediate Object → Target Object
  - *Client credentials* → *Intermediate credentials*

- **Simple delegation**
  - Client → Intermediate Object → Target Object
  - *Client credentials* → *Client credentials*

- **Composite delegation**
  - Client → Intermediate Object → Target Object
  - *Client credentials* → *Client & Intermediate’s credentials*

- Some mechanisms can restrict the extent of delegation.
## Delegation Example

- Bob delegating to Dan power of attorney to close his bank account

<table>
<thead>
<tr>
<th>CSI Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Level 0</strong> (no delegation)</td>
<td>To close his bank account, Bob has to go to bank</td>
</tr>
<tr>
<td><strong>Level 1</strong> (unrestricted delegation)</td>
<td>Bob gives Dan unlimited power of attorney to act as him. Dan can &lt;br&gt;• close Bob’s bank account &lt;br&gt;• read Bob’s medical records &lt;br&gt;• give the power of attorney to Mark</td>
</tr>
<tr>
<td><strong>Level 2</strong> (restricted delegation &amp; privileges)</td>
<td>Bob gives Dan limited power of attorney &lt;br&gt;Dan can close Bob’s bank account, but not &lt;br&gt;• read Bob’s medical records &lt;br&gt;• or give Bob’s power of attorney to Mark</td>
</tr>
</tbody>
</table>
Composite delegation options

- different implementations use different models
  - CORBASEC doesn’t dictate which (if any) to use
- distinguish option only by how many credentials at target
Security available to Applications

- Access Controls
  - for application functions and data
- Audit of application events
- Control of secure invocations
  - e.g. quality of message protection
- Authentication
  - mainly for specialist, log on applications
- Non-repudiation
Non-Repudiation
- providing irrefutable proof of actions

- Generate/verify evidence of actions e.g.
  - proof of creation, receipt, origin, delivery of data
- Evidence normally includes information to prove
  - integrity of the data, date/time, origin, action
- Use credentials to identify initiating principal
- Application transfers/stores evidence
  - depending on application type - e.g. workflow system, email
Non-repudiation Example
- sending a message which can be proved to come from you

- Message could pass through several objects
- There are NR policies c.f. access, audit policies
- Either end (or 3rd party) may store evidence for future disputes
Secure Interoperability
Secure Interoperability Model

Client

request

reply

Target Object

request

reply

ORB Security Services

Object Reference (IOR) with security info

ORB Core

secure (SECIOP) messages

using IIOP protocol

ORB Security Services
Secure Interoperability

- Interoperable Object Reference (IOR) contains
  - what security the object requires e.g. for message protection
  - what security the object supports e.g. security mechanisms
    - CORBASEC allows choice to fit existing security
- Enhanced IIOP includes SECIOP messages
  - security association set up for multiple requests
    - types of security tokens defined in CORBASEC
    - most details mechanism specific
      - but token can be GSS-API one
  - protected messages
    - may be integrity protected and/or be encrypted
Common Secure Interoperability - Requirements

• Support all CORBA security facilities
  – e.g. propagating privileges (roles, groups etc); delegation

• Low cost entry level
  – even if restricted facilities at this level

• Use of public key technology, at least as an option
  – as market is going that way
  – needed for non-repudiation, messaging etc

• Ability to use strong data protection
  – but allow other options for exportability

• Facilities independent of key technology used
CSI Specification

- CSI functionality levels
  - CSI level 2: full facilities including transmission of privileges (roles etc), separate audit_id, delegation controls
  - CSI level 1: identity only, simple (unrestricted) delegation
  - CSI level 0: identity only, no delegation

- CSI mechanism types
  - mandatory mech based on Kerberos: CSI1, secret key distribution
  - SPKM as option: CSI0, public key distribution
  - ECMA (SESAME subset) option: CSI2 with secret, public and hybrid key options

- Cryptographic profiles
  - to give choice of algorithms to meet regulations
  - mandatory entry level - data integrity only
Other Possible Security Mechanisms

• DCE Security part of DCE-CIOP, not IIOP, so not part of CSI

• SSL can be implemented but
  – fits below IIOP, rather than using SECIOP
    • confidentiality/integrity specified for connection
      – can’t specify finer grained protection
    – provides only CSI level 0 (if SSL authentication used)
      • identity only based authorisation (no privileges)
      • no delegation
Requirements

• Clean interface to security services
  – so ORB is independent of particular services used
  – choice of mechanisms for authentication, secure communications
    • allowing use of existing mechanisms
  – choice of access control (and other policies)
    • as different systems require different ones

• Allow construction of high security ORBs
  – but also lower assurance ones meeting other requirements
  – implementations to meet different threat models
Implementation
Security Replaceability at Invocation

- Request
- Reply

Client
Access Control
ORB Security Services
Secure Invocation

Client Access Decision
Vault

Target Access Decision
Security Context
Vault

Access Control
Secure Invocation

ORB Core

CORBA Security  Belinda Fairthorne, ICL
Access Decisions

- Effective rights are the rights for this principal in this domain
- Required rights are those required to access this type of object
- Access Decision and Policy objects are replaced for new policy
Vault and Security Context Objects

- Responsible for security of client-object communication
  - and cryptographic keys for this

- Vault creates security context objects at start of association
  - several objects may share identity, security context

- Interfaces based on Generic Security Services API (GSS-API)
  - so implementations can use GSS-API internally to access security mechanisms
Which objects are trusted for what?

- **Client**
- **Target Object**

**ORB core and interceptors**

**Security objects**
- Vault, security context, credentials etc

**Application objects**
- May be security unaware
- (may enforce application security policy)

**ORB**
- Must function correctly e.g. invoke required security objects in right order;
  - Using lower layer comms
  - And operating systems

**Core Security Objects**
- Must enforce security;
  - Using security services
Trusted components within domain

(Distributed) Trusted Computing Base

ORB core and interceptors

lower layer comms

Security objects
Vault, security context, credentials etc

External security services
e.g. key mang’t, CA, crypto

operating system, hardware

Client

Target Object
**Protection Boundary Options**

- **Client**
- **Target Object**
- **ORB core and interceptors**
- **Security objects** (Vault, security context, credentials etc)
- operating system, hardware

- Protect (sets of) application objects from each other
- Protect ORB from applications
- Protect security objects (and their keys etc) from applications and ORBs
- Protect security objects for one principal from other principals
Protection Boundaries

- IDL interface defined where protection boundaries may be needed
- May protect sets of (rather than individual) application objects from each other if:
  - set of objects is related, all handled by same code
  - confidentiality not needed between objects in set
- May not need strong run-time boundary to protect ORB from applications if:
  - applications are generated using trusted tools
  - threat to ORB is mainly accidental corruption and is low
- Generally separate out core security objects
  - reduce size of security enforcing code
  - protect critical security data such as cryptographic keys
So different CORBA Security Implementations for different needs

• Main options
  – way of constructing the system
    • protecting ORB, security services, applications from each other
    • ease of bypassing security enforcing code
    • number of application objects sharing identity
  – choice of security services, operating systems etc
    • what protection do these provide within node and across network?
    • what strength of protection, depending on cryptography used?

• Balance against
  – threats from applications & external sources (eavesdropping etc)
  – performance and other requirements
Model of Policy Domains

- Domain can span several nodes with shared policies
  - nodes can be of different types (UNIX, Windows etc)
  - if same distributed security technology
Policy between Domains

- CORBASEC has client and target policies
  - client and target may be in different domains
  - policy may use e.g. identity which includes domain
- CORBA Security model shows federated domains
  - but doesn’t specify “gateway” policy objects and interfaces e.g.
    - for separate interdomain access and audit policies
    - for privilege mapping at interface
Security Technology Domains in Model

- Specification covers multiple ORB technologies with same security technology
  - not security technology gateways
Summary

• CORBA Security specifies
  – security facilities for applications including those unaware of security
  – facilities for large distributed systems including role based access controls, delegation
  – administration of security policies using domains
  – secure interoperability within IIOP

• It allows a choice of security policies and security mechanisms via replacable interfaces

• It enables a choice of implementation architecture, to meet different needs, threats