Lightweight Security Service for CORBA

orbLOCK™

Bob Burt
Why Build A Lightweight Security Service?

- Why developing implementation of the Resource Access Decision Facility is driving development of a lightweight security service.
- Overview, objectives, features, ...
Resource Access Decision Facility

- Specification developed by the OMG to address the problem of business logic access control.
  - Standard format for naming protected resources.
  - Removes security policy from business logic.
  - Standard administrative interfaces for assigning policy to a resource.
  - Supports multiple policy engines and does not dictate policy mechanisms.
  - Security policies support multiple operations (e.g. “read”, “write”, “obliterate”, …)
  - Dynamic attribute service can modify security attributes that are used to make decisions.
Fully compliant implementation of RAD with Patterns.

Default Policy Evaluator supports:

- Multiple rules per operation.
- Each rule can be “AND ACL”, “OR ACL”, “DENY ACL”, “Anybody”, or “Nobody”.
- Supports time constraints by date, time, day of week.

- Collocation, Policy caching/notification, ...
- CORBA interfaces and CORBA convenience classes
- Java iLock Interface (JII) supports local policy evaluators and dynamic attribute service. CORBA transparent.
- Multiple platforms, multiple ORB’s, location services, ...
How do I get the attributes?

Heart of the RAD specification is the AccessDecision interface.

- Main operation is:
  
  boolean access_allowed ( ResourceName name, string operation, SecAttributeList attributes);

- The SecAttributeList comes from:
  
  SecurityLevel1.Current.get_attributes();

- **Problem: Where does SecurityLevel1.Current come from?**
- **Solution: Build our own.**
Lightweight Security Service

 Goals

- Meet requirements of 80%, expend 20% effort.
- 100% transparent for application code.
- 100% portable. Do what we can with Portable Interceptors, leave work requiring message interception alone.
- Flexible support for different security mechanisms.
- Support for authentication, delegation, access decision, and auditing.
- Use iLock for access decision engine.
- Simple administration
Lightweight Security Service - cont.

- Goals - cont
  - Supplement existing transport layer security services.
  - Provide minimal application control.
    - Object security domains
    - Unsecured objects
    - Control delegation

- Non - goals
  - Provide transport layer protocols (e.g. SSL)
  - Support message cryptography
  - Support for C++ ORB’s (this could change)
  - Compliance with any specifications
Common Secure Interoperability V2

Features

- Exchange protocol elements via service contexts
- Layers above transport layer security (SSL/TLS or SECIOP)
- Authentication layer for client authentication
- Attribute layer to push security attributes

- Pushed SecAttributes
- Supplemental client authentication
- Message Protection, Authentication

SAS Service Context Protocol
SSI/TLS or SECIOP
Problems

- Conformance (0, 1, and 2) requires support for SSL/TLS connections.
  - Violates our goal to not implement transport layer security.

- Only one `TAG_CSI_SEC_MECH_LIST` tagged component.
  - Violates our goal to interoperate with existing transport layer protocols.

  - Violates our goal to interoperate with existing solutions
Lightweight Security Service

Features

- Supports authentication with X.509 digital certificates.
- Replaceable authentication to support other mechanisms including proprietary and standard transport layer security services.
- Attribute management with LDAP.
- Replaceable attribute management.
- Applications can secure objects/operations with no code changes.
- Supports delegation (SecNoDelegation, SecSimpleDelegation, …)
- Support auditing.
Lightweight Security Service

Features continued

- Programming interfaces allow:
  - Get current security attributes
  - Set POA policy to turn off security.
  - Client control of delegated attributes.
- Use iLock for AccessDecision engine.
- Simple administration
  - IDL parser to create operation resources
  - Simple policies with “invoke” operation.
  - Create security domain resources.
  - Auditing, …
Lightweight Security Service

Using iLock for AccessDecision

- IDL Operations map to RAD Resource Name Mapping
  - IDL: omg.org/DfResourceAccessDecision/AccessDecision:1.0
    - Op = access_allowed
  - iLock Security Policies define “invoke” operation.
Lightweight Security Service

Replaceable Components

- Authenticator
  - byte [ ] getToken()
  - SecAttribute [] authenticateToken(byte [ ] token)
    - throws AuthenticatorError

- Attribute Manager
  - SecAttribute [] lookup(String name)
  - SecAttribute [] lookup(byte [ ] token)
Lightweight Security Service

- Authenticator
- Attribute Mgmt (LDAP)
- iLock
- Portable Interceptors
- IIOP
- [TLS]
- CLIENT
- INTERMEDIATE
- FINAL

O1
O2
Lightweight Security Service - Deployment

```
IDL
```
Bob Burt

bburt@2ab.com

www.2ab.com