Section 369
Noon Time Seminar

A Security Model for Service Oriented Architectures

Michael J. Pajevski
Distributed Systems Technologies Group (3697)
Jet Propulsion Laboratory
Agenda

• Introduction
• Changing threats and industry direction
• The move to service oriented architectures (SOAs)
• Some security considerations for SOAs
• Eliminating direct attacks in an SOA – proxies and private registries
• Access control model for SOAs – common security services
• A roadmap towards common security services
• Next steps
• Questions and answers
Introduction

• About the speaker…
  • Security architect participating in design of the security infrastructure for the global information grid being built for the entire DoD and allies.
  • Previously the design and development lead for the Common Operating Environment (COE) Kernel, which provided security and supported the interoperability of platforms such as Windows NT/2000 and Solaris.

• About this presentation…
  • Securing information systems is a complex endeavor.
  • No one aspect, such as perimeter control or virus detection, is enough.
  • This presentation takes a look at one important aspect - access control.
  • A look at SOAs and how access control is evolving to meet their needs.
What Threats?

- Threats range from inadvertent misuse to malicious attacks.
- Disgruntled or inexperienced employees can cause damage.
- Internet threats can come from anywhere in the world.
- A vulnerable partner or remote user site can compromise the system.
- Common threats are:
  - Probing of services and data
  - Unauthorized access to services and data
  - Alteration or deletion of information - in storage or in transit
  - Spoofing clients or providers, replaying captured messages.
  - Introduction of viruses, worms, …
  - Denial of service
- Why should we be concerned about these threats?
  - Improper disclosure or alteration of information could lead to the loss of revenue, schedule delays, mission loss, customer dissatisfaction, failure to meet policies and regulations, and even broken laws.
Mitigating the Security Threats

- Perimeter defenses (e.g., firewalls and filtering routers).
  - Protect against probing and other network attacks.
- Intrusion detection
  - Detect network attacks that made it past the perimeter.
- Virus protection
  - Computer software for finding and eliminating malicious programs.
- Access Control – focus of this presentation.
  - Controlling what entities in the system can do with resources.
- Confidentiality
  - Using encryption to hide sensitive information.
- Integrity
  - Making sure that data is not altered inappropriately.
- Non-repudiation
  - Ability to prove to a third party that an entity performed an action.
- Auditing
  - Keeping track of who did what.
Industry Directions for Security

- Security models are evolving due to many factors.
  - Number and sophistication of hackers.
  - Advancements in processing power.
  - Information warfare, Cyber terrorism.
  - Increasing integration for collaboration, e-commerce, etc.
  - Continuing need to reduce costs.
- “Security in-depth” (layered security) is even more important today.
  - Perimeter defense is not enough.
  - Appropriate access control measures can significantly reduce risk.
    - Bad access control practices can really get in the way.
- Multi-party exchanges require end-to-end security.
  - Current solutions are often point-to-point.
- Increased integration is driving overall system architecture.
  - Moving to service oriented architectures (SOAs) and Web Services.
Service Oriented Architecture (SOA)

- SOAs expose business processes and data through services.
  - Service oriented model supports re-use and integration.
  - Security functions can be packaged as common services.
    - New development costs are reduced through re-use.
    - Legacy systems can be converted, if necessary, gradually.
- Providers register services that consumers can find and interact with.
  - Supports new, dynamic ways of doing business.
- Standardizing technologies used in SOAs increases interoperability.
  - Increasing interoperability decreases integration costs.
  - Latest move is towards Web Services (WS).
    - Based on XML data representations and SOAP messaging.
    - Common service definitions - Web Services Description Language (WSDL).
    - Common XML schemas for security, etc.
    - Specifications still maturing.
Some SOA Security Issues

• Increased exposure brings a greater potential for compromise.
  • Each service is a potential attack point.
  • Greater damage possible due to increased exposure of data.

• An SOA is not practical without a common security infrastructure.
  • No common security services results in “islands of security”.
    • Multiple points of administration increase administrator burden.
    • Having several passwords increases user burden.
  • Common security services can reduce administrative and user burden.
    • Consolidation of administration.
    • Common identification services enable single sign-on (SSO).
Solving SOA Security Issues

- Need to mitigate risks caused by increased exposure.
- Eliminate direct attacks.
  - Proxy service to insulate services from consumers.
  - Split service registry into public and private areas.
- Control what “users” can do with resources.
  - Access control reduces the risk of damage through “least privilege”.
    - Limiting what “users” can do also limits the harm they can cause.
Issues with a Direct Path to Resources

- Advertising the actual location of resources gives attackers useful information.
- Direct interaction with services increases the risk to them.

A simple SOA Interaction Model

1. Register

2. Find provider

3. Use provider

Service Registry

Service Provider

Consumer Application
Securing the Path to Resources

- Eliminate direct attacks on resources.
  - Do not advertise actual provider location.
    - Provide a public service registry for consumers.
      - Public registry lists proxy service as location of all services.
    - Provide a private registry service for proxy and other trusted entities.
      - Private registry lists the actual location of each provider.
  - Use proxy service to insulate providers.
    - All requests go through the proxy service.
    - Proxy gets authorization before forwarding requests.
    - Check message format and filter malicious content.
    - Proxies can also provide load balancing, quality of service, etc.
A “security enhanced” SOA Interaction Model

Note - the system could have only one service registry, which reports different service locations depending on whether the requestor is a trusted proxy or not.
Controlling Actions on Resources

• “Least privilege” should be used to limit damage.
  • Assign privileges based on job duties.
    • e.g., don’t allow people to write data that they only need to read.
  • Some rules could come from compliance issues.
    • e.g., organizational policies, laws and regulations, etc.
• Many applications use internal mechanisms for controlling access.
  • Many “islands of security” increase cost of integration and user burden.
• The move in industry is towards common security services.
  • Reduced development costs through re-use.
  • Reduced integration costs –
    • Easier integration of services into system.
    • Easier system-to-system integration through trust models and federation.
  • Unified administration and common identification reduce user burden.
What is Needed to Control Actions?

• Consumers and providers must be identifiable.
  • Authentication (positive identification) enables access control and auditing.
  • Need a means to manage, obtain, and verify identity data.
  • Should have “authorities” to provide a basis for trust.

• Need to associate attributes with consumers and providers.
  • Rules are often based on attributes rather than names of individuals.
    • e.g., Specify access according to group, role, clearance level, citizenship, etc.
  • Attributes may be managed by different organizations.
  • The system would benefit from “authorities” for this data.

• Need to manage, obtain, and verify authorization rules.
  • Rules may be managed by various organizations.
  • Authorities for rules would also be very beneficial.

• Need to make authorization decisions.
  • This is another important function best provided by authorities.

• Must prevent circumvention of the rules.
• Should keep records of events.
Access Control Functionality as Services

• Some access control functionality can be exposed as services.
  • Managing information about identities, attributes, rules, and audit events.
    • Creating “authorities” for this data provides a basis for trust.
  • Making authorization decisions.
    • Common logic that can be re-usable.
• Some things are better done “locally”.
  • Such as cryptographic tasks.
    • Digital signatures, hash algorithms, encryption.
    • Require high confidence in code.
    • Need to be done quickly.
    • No need for centralization.
Common Security Services

• Provide access control (and more) with the following services:
  • Proxy Service
    • a.k.a., Policy Enforcement Point (PEP)
    • Prevents circumvention of the access control rules.
    • Provides some insulation between consumers and providers.
    • e.g., Blue Titan – a QoS and security solution for BEA WebLogic.
  • Authorization Service
    • a.k.a., Policy Decision Point (PDP)
    • Processes requests, identity and attribute information, and rules in order to produce an authorization decision.
    • e.g., A Kerberos Ticket Granting Service (TGS)
  • Identity Service
    • a.k.a., a [Security] Token Service
    • Maintains identification information and issues security tokens.
    • e.g., a PKI service interface or a Kerberos Key Distribution Center (KDC).
Common Security Services, continued

• Three more common security services involved in access control are:
  • Attribute Service
    • Maintains information about the properties of identifiable entities.
    • e.g., Directory service (like X.500 or LDAP) or database (like Oracle).
  • Policy Service
    • Maintains the rules used to make authorization decisions.
    • e.g., an LDAP directory or database storing XACML statements.
  • Audit Service
    • Provides centralized access to audit data.
    • e.g., Database, data mining and reporting tools.
* Auditing is used by most of the common services, and possibly the consumers and resources (via the proxy - i.e., the PEP).
Common Security Services in Action

- Consumer application sends a service request to the proxy (i.e., PEP).
  - Proxy protects the services from direct interaction with consumers.
- Proxy constructs authorization decision request, sent to Authorization Service.
- Authorization Service processes the authorization decision request.
  - Get authentication data or check revocation list maintained by Identity Service
  - Get applicable rules from the Policy Service
  - Get applicable attributes from the Attribute Service
  - Make decision and return result to PEP.
  - Decision should be signed for assurance purposes.
- PEP forwards approved requests to the service provider.
- Service provider processes the request.
  - Provider is configured to trust decisions from PDP (or requests from PEP).
    - e.g., provider has public key of PDP or PEP and can verify digital signatures.
    - Provider returns response to the PEP.
- PEP forwards response from provider to consumer.
- Some steps above may generate audit data - sent to Audit Service.
A Security Roadmap

- A monolithic set of requirements is not the most practical approach.
  - Legacy systems will have difficulties complying with all rules.
  - Not all systems require the same level of security.
  - Simple pass/fail approach gives no way to measure partial compliance.
- Defining requirements through compliance levels is more useful.
  - Tiered requirements provide stepping stones.
  - Provides a way to measure progress.
  - Recognizes “one size fits all” approach is neither necessary nor practical.
- Implementation guidance should be given.
  - To assist convergence towards common tools.
- A simple example of compliance levels for access control is:
  - Level 0 – Does not support access control.
  - Level 1 – Supports access control via proprietary internal mechanism.
  - Level 2 – Supports access control via some tightly-coupled mechanism.
  - Level 3 – Provider partially uses the common security services.
  - Level 4 – Provider relies entirely on the common security services.
• Access control code and databases are built into provider application.
• Results in multiple points of administration.
• Results in different usernames and passwords for different applications.
  • Synchronizing usernames and passwords is possible, but not ideal.
• e.g., custom application with built-in access control code and databases.
Compliance Level 2 –
Tightly Coupled Access Control

- A service provider may use some internal code and tightly-coupled external databases.
- Can reduce administrative and user burden (assuming multiple applications use the same external databases).
- Tight coupling prevents changing the products that are used.
  - Creates reliance on specific vendors.
- e.g., a custom application using Oracle for user information and policy rules.
• Service provider may use some of the common security services.
• e.g., use common policy service, but use internal logic and a tightly coupled user database.
  • Such as a converted legacy application.
Compliance Level 4 –
Full Use of Common Security Services

- Service provider full relies upon the common security services.
- Access control code does not need to be coded into new applications.
- The Authorization Services uses the other common security services.
- The common security services use common support services, such as a Storage Service.
- e.g., goal for DISA’s Net-Centric Enterprise Services (NCES).
Next Steps

- Explore savings and other benefits of common security services.
- Consider ramifications of standing up relied upon services.
  - Requires a serious, long-term commitment.
- Examine alternatives for service implementations.
  - e.g., PKI vs. Kerberos vs. LDAP; Web Services; etc.
- Look at how rules would be defined and managed.
  - How are users and resources classified to support access control?
  - Does Role Based Access Control (RBAC) make sense?
- Develop requirements for common services and their consumers.
  - Use tiered assurance levels rather than one monolithic set.
- Identify areas of interest for further discussion – for example:
  - How would security services be implemented via Web Services?
  - Comparison of identification systems.
    - e.g., Kerberos vs. PKI, etc.
  - What is RBAC and is it a good fit?
  - Optimizing access control
  - Confidentiality, integrity, etc.
Conclusions

• Security models are evolving due to many factors.
• There is a move towards providing security through common services.
• A useful set of services for controlling access is:
  • Proxy Services
  • Identity (or Token) Services
  • Attribute Services
  • Policy Services
  • Authorization Services
  • Audit Services
• Defining requirements in terms of assurance levels provides a practical approach to making and measuring progress.
• Security services must be reliable – requiring a serious commitment.
• There is much to explore before making any commitments.
Q&A

• Any questions?