Information security is fundamentally about managing
- authorization and
- trust
so as to manage risk

THE OM-AM WAY

What?  How?
- Objectives
- Model
- Architecture
- Mechanism

LAYERS AND LAYERS

- Multics rings
- Layered abstractions
- Waterfall model
- Network protocol stacks
- Napolean layers
- RoFi layers
- OM-AM
- etcetera

OM-AM AND MANDATORY ACCESS CONTROL (MAC)

What?  How?
- No information leakage
- Lattices (Bell-LaPadula)
- Security kernel
- Security labels

OM-AM AND DISCRETIONARY ACCESS CONTROL (DAC)

What?  How?
- Owner-based discretion
- numerous
- ACLs, Capabilities, etc
OM-AM AND ROLE-BASED ACCESS CONTROL (RBAC)

What?

Objective neutral
RBAC96, ARBAC97, etc.
user-pull, server-pull, etc.
certificates, tickets, PACs, etc.

How?

DISTRIBUTED RBAC (DRBAC) CASE STUDY

Approximately a dozen physical sites
Approximately 2-3 simulation models/site
Fewer than 100 roles structured in a very shallow hierarchy
  A subset of roles is used in any single simulation model
Fewer than 100 users
A user uses only one role at a time
  Convenient but not critical
Moderate rate of change

DISTRIBUTED RBAC (DRBAC) CASE STUDY

Permission-role assignment
  Locally determined at each simulation model
User-role assignment
  A user can be assigned to a role if and only if all simulation models using that role agree
  A user is revoked from a role if and only if any simulation model using that role revokes the user

DISTRIBUTED RBAC (DRBAC) CASE STUDY

Each simulation model has a security administrator role authorized to carry out these administrative tasks
A simulation model can assign permissions to a role X at any time
  even if X is previously unused in that simulation model
Consequently any simulation model can revoke any user from any role!

MODEL CUSTOMIZATION

Each session has a single role
SM = \{sm1, ..., smk\}, simulation models
OP = \{op1, ..., op1\}, operations
P = SM \times OP, permissions
SMA = \{sma1, ..., smk\}, administrative roles
R \cap SMA = \emptyset
Admin: SM \leftrightarrow SMA
MODEL CUSTOMIZATION

- Can formalize the administrative rules given earlier
- For each simulation model designate a unique user to be the chief security administrator who is authorized to assign and revoke users from the security administrator role for that model

DRBAC ARCHITECTURES

- Permission-role
  - Enforced locally at each simulation model
- Permission-role administration
  - Enforced locally at each simulation model
  - May need to communicate to other simulation models
- User-role
  - See following slides
- User-role administration
  - Centralized or decentralized
THE OM-AM WAY

<table>
<thead>
<tr>
<th>What?</th>
<th>Assurance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objectives</td>
<td>Model</td>
</tr>
<tr>
<td>Architecture</td>
<td>Mechanism</td>
</tr>
</tbody>
</table>

Secure Attribute Services on the Web

- **WWW (World Wide Web)**
  - widely used for electronic commerce and business
  - supports synthesis of technologies
  - mostly, Web servers use identity-based access control
  - scalability problem

Background

- **An attribute**
  - a particular property of an entity
    - e.g., role, identity, SSN, clearance, etc.
- If attributes are provided securely,
  - Web servers can use those attributes
    - e.g., authentication, authorization, access control, electronic commerce, etc.
- **A successful marriage of the Web and secure attribute services is required**

User-Pull Architecture

- Each user
  - pulls appropriate attributes from the Attribute Server
  - presents attributes and authentication information to Web servers
- Each Web server
  - requires both identification and attributes from users
- **High performance**
  - No new connections for attributes

Server-Pull Architecture
Related Technologies

- **Cookies**
  - in widespread current use for maintaining state of HTTP
  - becoming standard
  - not secure
- **Public-Key Certificates (X.509)**
  - support security on the Web based on PKI
  - standard
  - simply, bind users to keys
  - have the ability to be extended

Cookies

- **Cookies**
  - Domain
  - Flag
  - Path
  - Cookie_Name
  - Cookie_Value

<table>
<thead>
<tr>
<th>Cookie 1</th>
<th>Domain</th>
<th>Flag</th>
<th>Path</th>
<th>Cookie_Name</th>
<th>Cookie_Value</th>
<th>Secure</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Name</td>
<td></td>
<td></td>
<td>FALSE</td>
<td>2/25/06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cookie 2</th>
<th>Domain</th>
<th>Flag</th>
<th>Path</th>
<th>Cookie_Name</th>
<th>Cookie_Value</th>
<th>Secure</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Role</td>
<td>manager</td>
<td></td>
<td>FALSE</td>
<td>2/25/06</td>
</tr>
</tbody>
</table>

Security Threats to Cookies

- Cookies are not secure
  - No authentication
  - No integrity
  - No confidentiality
- can be easily attacked by
  - Network Security Threats
  - End-System Threats
  - Cookie Harvesting Threats

Secure Cookies on the Web

Secure Cookies on the Web

A Set of Secure Cookies

How to Use Secure Cookies
Applications of Secure Cookies

- User Authentication
- Electronic Transaction
- Eliminating Single-Point Failure
- Pay-per-Access
- Attribute-based Access Control

Authentication Cookies

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Status</th>
<th>Expires</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP_Cookie</td>
<td>TRUE</td>
<td>10/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port_Cookie</td>
<td>TRUE</td>
<td>10/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KT_Cookie</td>
<td>TRUE</td>
<td>10/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SFA_Cookie</td>
<td>TRUE</td>
<td>10/31/09</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Server-Pull Architecture

- Each user presents only authentication information to Web servers
- Each Web server pulls users' attributes from the Attribute Server
- Authentication information and attribute do not go together
- More convenient for users
- Less convenient for Web servers

Secure Cookies for Electronic Transactions

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Status</th>
<th>Expires</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>User_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KT_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SFA_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Kerberos-Based Authentication by Secure Cookies

- Request TGT
- TGT_Cookie = TGT
- TOT, Cookie = TOT
- KT_Cookie = KT
- KT_Cookie = KT
- SFA_Cookie

Secure Cookies for Pay-Per-Access

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Status</th>
<th>Expires</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>User_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KT_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SFA_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key_Cookie</td>
<td>TRUE</td>
<td>12/31/09</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Secure Cookies for RBAC

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>John</td>
<td>User</td>
<td>Alice</td>
<td>User</td>
</tr>
<tr>
<td>Role</td>
<td>Manager</td>
<td>Role</td>
<td>NULL</td>
<td>Role</td>
</tr>
<tr>
<td>Cookie</td>
<td>Secure</td>
<td>Cookie</td>
<td>NULL</td>
<td>Cookie</td>
</tr>
<tr>
<td>Cookie</td>
<td>NULL</td>
<td>Cookie</td>
<td>NULL</td>
<td>Cookie</td>
</tr>
<tr>
<td>Cookie</td>
<td>NULL</td>
<td>Cookie</td>
<td>NULL</td>
<td>Cookie</td>
</tr>
</tbody>
</table>

*Stash the passwords in an alternative to the one of **Cookie**.

RBAC on the Web

by Secure Cookies

X.509 Certificate

- Digitally signed by a certificate authority
  - To confirm the information in the certificate belongs to the holder of the corresponding private key
- Contents
  - Version, serial number, subject, validity period, issuer, optional fields (v2)
  - Subject’s public key and algorithm info.
  - Extension fields (v3)
  - Digital signature of CA
- Binding users to keys
- Certificate Revocation List (CRL)

Smart Certificates

- Short-Lived Lifetime
  - More secure
    - Typical validity period for X.509 is months (years)
    - Users may leave copies of the corresponding keys behind
    - The longer-lived certificates have a higher probability of being attacked
  - No Certificate Revocation List (CRL)
    - Simple and less expensive PKI

- Containing Attributes Securely
  - Web servers can use secure attributes for their purposes
  - Each authority has independent control on the corresponding information
    - Basic certificate (containing identity information)
    - Each attribute can be added, changed, revoked, or re-issued by the appropriate authority
      - E.g., role, credit card number, clearance, etc.
  - Short-lived certificate can remove CRLs
Separate CAs in a Certificate

Smart Certificates

- **Postdated Certificates**
  - The certificate becomes valid at some time in the future
  - possible to make a smart certificate valid for a set of duration
  - supports convenience

- **Confidentiality**
  - Sensitive information can be
    - encrypted in smart certificates
      - e.g. passwords, credit card numbers, etc.

A Smart Certificate

Applications of Smart Certificates

- **On-Duty Control**
- **Compatible with X.509**
- **User Authentication**
- **Electronic Transaction**
- **Eliminating Single-Point Failure**
- **Pay-per-Access**
- **Attribute-based Access Control**

Injecting RBAC to Secure a Web-based Workflow System

Gail-Joon Ahn and Ravi Sandhu
George Mason University

Myong Kang and Joon Park
Naval Research Laboratory

**WORKFLOW MANAGEMENT SYSTEMS**

- Control and coordinate processes that may be processed by different processing entities
- Received much attention
- Marriage with Web technology
- Minimal security services
OBJECTIVE

- Inject role-based access control (RBAC) into an existing web-based workflow system

WHY RBAC?

- A mechanism which allows and promotes an organization-specific access control policy based on roles
- Has become widely accepted as the proven technology

SIMPLIFIED RBAC MODEL

![Simplified RBAC Model Diagram]

ROLE-BASED SECURE WORKFLOW SYSTEM

- Workflow Design Tool
- Workflow (WF) System
- Role Server

BASIC COMPONENTS

![Basic Components Diagram]

ARCHITECTURES

- USER-PULL STYLE
- SERVER-PULL STYLE
**USER-PULL STYLE**

- WF design tool
- WF system
- Role Server
- Certificate server
- User-role DB

**SERVER-PULL STYLE**

- WF design tool
- WF system
- Role Server
- Certificate server
- User-role DB

**NRL (Naval Research Lab.) DESIGN TOOL**

- Design workflow model
- Create role and role hierarchies
- Assign role to task
- Exporting role hierarchies to role server

**NRL DESIGN TOOL (Cont’d)**

- Platform: Windows NT, JDK 1.2

**WORKFLOW SYSTEM**

- Each task server is web server
- User should present client authentication certificate
- User’s privilege is authorized by content of certificate (specially client’s role information)

**ROLE AUTHORIZATION ON WORKFLOW SYSTEM**

- Task Server (Web Server)
- User presents client certificate
- Retrieve role information
- Check authorization status
- Display resource

- Resources

- Client
**ROLE SERVER**

- User Role Assignment
- Certificate Server

**USER ROLE ASSIGNMENT**

- maintain role hierarchies and user database
- assign users to roles
- generate user-role database

**USER ROLE ASSIGNMENT** (Cont’d)

- maintain role hierarchies and user database
- assign users to roles
- generate user-role database

**CERTIFICATE SERVER**

- authenticate client
- retrieve client’s role information from user-role database
- issue certificate with client’s role information

**X.509 CERTIFICATE**

- Serial number: 0000/0000
- Validity: 01/01/99 - 01/01/00
- Common Name = Gail J. Ahn
- Organization Unit = Staff
- Public key:
  - 1e3542767fatew76585098327djkfh9974-72ks78610092wef3
- Private Key
- Certificate Authority

**CERTIFICATE ISSUE**

1. Client Certificate Request
2-3. Challenge-Response based on Password
4-5. Retrieving Role Information of a User
6-7. Creating Certificate Enrollment Form and Public Key Embedded
8-9. Issuing Client Certificate
10. Downloading Client Certificate
11. Logging Certificate Information
CERTIFICATE AUTHORIZATION OVER SSL

Client

Server certificate

Role authorization

SSL connection

Task Server

REVERSE PROXYING (MINIMAL CHANGES IN SERVER SIDE)

Client

Proxy Server

Task Server

SSL connection

Request resource

Send modified request

Forward resource

Send resource

task.html

http://b.com/task.html

http://a.com/task.html

FINAL SCENARIO

Client

Proxy Server

SSL

IP checking

Step 1

Step 2

Step 3

Certificate Server

Step 4

Step 5

Role Server

Task Server

http://b.com/task.html

http://a.com/task.html