AUTHORIZATION, TRUST AND RISK

- Information security is fundamentally about engineering
  - authorization and
  - trust
  so as to
  - manage risk
ENGINEERING AUTHORITY & TRUST
4 LAYERS

What?

Policy
Model
Architecture
Mechanism

How?

ENGINEERING AUTHORITY & TRUST
4 LAYERS

What? Multilevel Security

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

How?
ENGINEERING AUTHORITY & TRUST
4 LAYERS

What? Role-Based Access Control (RBAC)

Policy neutral
RBAC96 model
user-pull, server-pull, etc.
certificates, tickets, PACs, etc.

How?

RBAC

◆ Policy neutral yet Policy oriented
  ● least privilege
  ● separation of duties
  ● abstract permissions
  ● separation of administration and access
  ● roles are a semantic unit around which to build policy
CLASS I SYSTEMS
ENFORCEMENT ARCHITECTURE

CLASS I SYSTEMS
ADMINISTRATION ARCHITECTURE
CLASS II SYSTEMS
SERVER-PULL

Client

Authorization Server

Server

Authentication Server

CLASS II SYSTEMS
USER-PULL

Client

Authorization Server

Server

Authentication Server

© Ravi Sandhu 1999
CLASS II SYSTEMS
THREE-TIER

Client ↔ Authorization Server ↔ Server

Authentication Server

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**
Lecture 7: Secure Attribute Services

User-Pull Model

◆ 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 Model

*Authentication Information can be either user-based or host-based.
Server-Pull Model

- 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

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

<table>
<thead>
<tr>
<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>Alice</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<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>FALSE</td>
<td>12/31/99</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

<table>
<thead>
<tr>
<th>Name_Cookie</th>
<th>Domain</th>
<th>Flag</th>
<th>Pub</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_Cookie</td>
<td>Alice*</td>
<td>FALSE</td>
<td>12/31/99</td>
<td></td>
</tr>
<tr>
<td>Role_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Role_Cookie</td>
<td>manager*</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Life_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Life_Cookie</td>
<td>12/31/99</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Pwd_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Pwd_Cookie</td>
<td>hashed_password</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Key_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Key_Cookie</td>
<td>encrypted_key*</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Seal_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Seal_Cookie</td>
<td>Seal of cookies</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
</tbody>
</table>

* Sensitive fields can be encrypted in the cookies.
** Seal of Cookies can be either MAC or signed message digest of cookies.
Note: Pwd_Cookie can be replaced with one of the other authentication cookies in Figure 4.

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>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>IP_Cookie</td>
<td>129.174.100.88</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Pwd_Cookie</td>
<td>hashed_password</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Kerberos_Ticket</td>
<td>{Alice, K_c}K_s</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Sign_Cookie</td>
<td>Signature_of_Alice</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
</tbody>
</table>

Secure Cookies for Electronic Transactions

* Sensitive fields can be encrypted in the cookies.
** Seal of Cookies can be either MAC or signed message digest of cookies.
Note: Pwd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1

© Ravi Sandhu 1999
Kerberos-Based Authentication by Secure Cookies

1. Request TGT

2. $TGT_{Cookie} = (TGT, S_A)K_{sec}$

3. $TGT_{Cookie} = TGT$

4. $KT_{Cookie} = T_C$  
   $KC_{Cookie} = (K_{C,B}, Bob)S_A$

5. $KT_{Cookie} = T_C$  
   $TSS_{Cookie}$

6. $TSS^{\prime}_{Cookie}$

$TGT = (S_A, Alice)K_{sec}$  
$TSK_{Cookie} = (timestamp)S_A, Alice, Bob$  
$T_C = (Alice, K_{C,B})K_S$ (ticket to Bob)  
$TSS_{Cookie} = (timestamp)K_{C,B}$  
$TSS^{\prime}_{Cookie} = (timestamp+1)K_{C,B}$

$TSK_{Cookie}$: Timestamp for the KDC  
$KT_{Cookie}$: Kerberos Ticket Cookie  
$KC_{Cookie}$: Kerberos Client Cookie  
$TSS_{Cookie}$: Timestamp for the Server

Secure Cookies for Pay-Per-Access

*Sensitive fields can be encrypted in the cookies.
**Seal of Cookies can be either MAC or signed message digest of cookie.
(Note: Pwd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1)
Secure Cookies for RBAC

<table>
<thead>
<tr>
<th>Cookie</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>Name_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Name</td>
<td>Alice</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Role_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Role</td>
<td>Manager</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Life_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Life_Cookie</td>
<td>12/31/99</td>
<td>FALSE</td>
<td></td>
</tr>
<tr>
<td>Pswd_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Pswd_Cookie</td>
<td>Encrypted_Password*</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>IP_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>IP_Cookie</td>
<td>129.174.142.88</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
<tr>
<td>Seal_Cookie</td>
<td>acme.com</td>
<td>TRUE</td>
<td>/</td>
<td>Seal_Cookie</td>
<td>Digital_Signing</td>
<td>FALSE</td>
<td>12/31/99</td>
</tr>
</tbody>
</table>

* Hash of the passwords is an alternative as the content of the Pswd_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

Smart Certificates

◆ 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 Certificate

- Basic Certificate
  - serial number
  - issuer
  - subject
  - validity period
  - public key info
  - optional fields (e2)

- Extensions
  - att_1_CA's Digital Signature
  - att_n_CA's Digital Signature

Signed by Basic CA

Signed by att_1_CA's Digital Signature

Signed by att_n_CA's Digital Signature

* attribute info: attributes, attribute issuer, validity period of attributes, etc.

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.

© Ravi Sandhu 1999
A Smart Certificate

Certificate Content:

Certificate:

Version: v3 [OSS]
Serial Number: 24 [OSS]
Signature Algorithm: CMS v1 HEG With RSA Encryption
Issuer: CN=data.list.gen.edu, OU=CS, C=CA, C=US
Valid From: 12-May-99 01:09:57
Valid To: 12-May-99 01:09:57
Issuer Public Key:

Public Key:

Certificate Contents:

Certificate: Certificate Type

Certified: yes
certification usage:

Identifier: role

Identifier: authority

Identifier: subject

Signature Algorithm: CMS v1 HEG With RSA Encryption

Signature:

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