Secure Access Control

Shiu-Kai Chin
Department of Electrical Engineering and Computer Science
Syracuse University
http://www.ecs.syr.edu/faculty/chin
Rights and Permissions

• Systems have:
  – *Subjects*: actors, agents, people, processes, principals
  – *Objects*: files, record fields, printers, services
  – *Operations*: read, write, execute, hire, fire, etc.

• A *right* is a pair: *(operation, object)*

• A *permission* is a set of rights
Access Matrix

- A matrix where columns correspond to objects and rows correspond to subjects.
- Matrix entries are the operations on the object that are permitted for a given subject.
Example

• The application is a database consisting of *secret agents* and the *organizations* they are spying on.

• The *objects* in the database are pairs \((agent, organization)\):
  \[\text{Objects} = \{(006, \text{SPECTRE}), (007, \text{SPECTRE}), (\text{Sydney}, \text{SD6})\}\]

• The *operations* on the objects are \(R_{Agent}\) and \(R_{Org}\) where:
  \[R_{Agent}(agent, organization) = agent\]
  \[R_{Org}(agent, organization) = organization\]
Policy

• Policies dictate who (or what role) has what kind of rights to any object
• Ultimately, users or principals in a system have permissions assigned to them
  – Ad hoc: usually done by some system administrator – *dangerous and error prone*
  – Systematically: usually based on user job function or user role – *recommended*
## Example of User Permission Assignment

<table>
<thead>
<tr>
<th>Users</th>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(006, SPECTRE)</td>
</tr>
<tr>
<td></td>
<td>(007, SPECTRE)</td>
</tr>
<tr>
<td></td>
<td>(Sydney, SD6)</td>
</tr>
<tr>
<td>Alice</td>
<td>$R_{\text{Agent}}$</td>
</tr>
<tr>
<td>Bob</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td></td>
</tr>
<tr>
<td>Dan</td>
<td>$R_{\text{Agent}}, R_{\text{Org}}$</td>
</tr>
<tr>
<td>Ellen</td>
<td></td>
</tr>
<tr>
<td>Frank</td>
<td>$R_{\text{Org}}$ $R_{\text{Org}}$ $R_{\text{Org}}$</td>
</tr>
</tbody>
</table>
Managing Permissions

- Managing User-Permission (UP) assignments becomes unwieldy quickly (consider people change jobs or leaving)
  - Cost is proportional to $|U| \cdot |P|$ where $|U|$ is the number of users in U and $|P|$ is the number of permissions in job position $|P|$
- Easier for each job position $P$ to have a relationship specifying which users have position $P$ and have another relationship specifying what permissions $P$ has
  - User Assignment (UA): $\{(u,P) \mid u \in U\}$ where $P$ is a role
  - Permission Assignment (PA): $\{(P,p) \mid p \in P\}$ where $p$ is $(operation, object)$
- Associated cost for each role $P$ is $|U| + |P|$
# Example Job Functions

<table>
<thead>
<tr>
<th>Users</th>
<th>Job Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>006’s Handler</td>
</tr>
<tr>
<td>Bob</td>
<td>007’s Handler</td>
</tr>
<tr>
<td>Carol</td>
<td>Sydney’s Handler</td>
</tr>
<tr>
<td>Dan</td>
<td>SPECTRE Manager</td>
</tr>
<tr>
<td>Ellen</td>
<td>SD6 Manager</td>
</tr>
<tr>
<td>Frank</td>
<td>Counsel</td>
</tr>
</tbody>
</table>
Example Role Hierarchy

- Partial Ordering on Roles
  - If $R_1 \geq R_2$ then all permissions of $R_2$ are also had by $R_1$
  - If $R_1 \geq R_2$ then all users in $R_1$ are in $R_2$
- $SD6\text{ Manager} \geq Sydney\text{ Handler}$
- $SPECTRE\text{ Manager} \geq 006\text{ Handler}$
- $SPECTRE\text{ Manager} \geq 007\text{ Handler}$
- $Counsel$ unrelated to any other role
Example Assignments

User Assignments
• UA = \{ (Alice, 006 Handler), (Bob, 007 Handler), (Carol, Sydney Handler), (Dan, 006 Handler), (Dan, 007 Handler), (Dan, SPECTRE Manager), (Ellen, Sydney Handler), (Ellen, SD6 Manager), (Frank, Counsel) \}

Permission Assignments
• PA = \{ 
  (006 Handler, 
  \{(R_{agent}(006, SPECTRE)\})), 
  (SPECTRE Manager, 
  \{(R_{agent}(006, SPECTRE)), 
  (R_{Org}(006, SPECTRE)), 
  (R_{agent}(007, SPECTRE)), 
  (R_{Org}(007, SPECTRE)) \}), \ldots \text{etc.} \}
Implementing Access Control

• Identify principal and requested operation(s) on a specific object(s) or resources
• Determine if principal is authorized to access specified resource in requested fashion
  – If so, grant access and log request
  – If not, deny access and log request
• The above is known as a reference monitor
Reference Monitor Model for Access Control

Access Policy

Authentication

Authorization

Principal → Do Operation → Reference Monitor → Objects

Source Request Guard Resource

Audit Log

Objects Resource
How do we authenticate and authorize?

• Authentication
  – Requests are *digitally signed* and checked for integrity using cryptographic keys
  – Keys are bound to principals by *certificates*
  – Delegations are verified by certificates and keys

• Authorization
  – Authenticate source of request then authorized requested exercise of *right* against user assignments and permission assignments
Keys and Key Management

• Standard topics found in network security texts and courses
• System builders: pay particularly close attention to the basis for believing in key bindings
  – Example: Saddam Hussein says, “use $K_{\text{bogus}}$ to communicate with George W. Bush.”
  – Would you use the key even if the statement was integrity-checked?