Lecture #1

• Course overview
• Basics of security
• Access control matrix
• Primitive operations and commands
• Miscellaneous points
Course Overview:
Administration

- Web sites
  - Main site
    - http://smartsite.ucdavis.edu
  - Secondary Site:
- Being recorded for Livermore students
- If you (or I 😊) miss a class, you can view it
  - But please try to come!
Course Overview: Questions

- What can security decide, and what can it not decide?
- Policy models: what can systems and people do, and what can they not do?
- Information flow: how can information move around a system?
Functionality

• Confidentiality
  – Keeping data and resources hidden

• Integrity
  – Data integrity (integrity)
  – Origin integrity (authentication)

• Availability
  – Enabling access to data and resources
Assurance

- Specification
  - Requirements analysis
- Design
  - How system will meet specification
- Implementation
  - Program/systems that carry out design
- Operation and maintenance
  - How to update, modify, use program/system
Trust and Assumptions

• Underlie all aspects of security

• Policies: what is, is not allowed
  – Unambiguously partition system states
  – Correctly capture security requirements

• Mechanisms: what enforce policies
  – Assumed to enforce policy
  – Support mechanisms work correctly
People and Organizations

• Organizational Problems
  – Power and responsibility
  – Financial benefits

• People problems
  – Outsiders and insiders
  – Social engineering
Models

• Abstract irrelevant details of entity or process being modeled
  – Allows you to focus on aspects that are of interest
  – If done correctly, results from analyzing the model apply to entity or process

• Assumption: nothing you omit affects the application of the results
Why Access Control Matrix?

• Protection state of system
  – Describes current settings, values of system relevant to protection

• Access control matrix
  – Describes protection state precisely
  – Matrix describing rights of subjects
  – State transitions change elements of matrix
### Description

<table>
<thead>
<tr>
<th>objects (entities)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
</tr>
<tr>
<td>$o_1$</td>
</tr>
</tbody>
</table>

- Subjects $S = \{ s_1, \ldots, s_n \}$
- Objects $O = \{ o_1, \ldots, o_m \}$
- Rights $R = \{ r_1, \ldots, r_k \}$
- Entries $A[s_i, o_j] \subseteq R$
- $A[s_i, o_j] = \{ r_x, \ldots, r_y \}$ means subject $s_i$ has rights $r_x, \ldots, r_y$ over object $o_j$
### Example 1

- **Processes** $p, q$
- **Files** $f, g$
- **Rights** $r, w, x, a, o$

<table>
<thead>
<tr>
<th></th>
<th>$f$</th>
<th>$g$</th>
<th>$p$</th>
<th>$q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>rwo</td>
<td>$r$</td>
<td>rwxo</td>
<td>$w$</td>
</tr>
<tr>
<td>$q$</td>
<td>$a$</td>
<td>ro</td>
<td>$r$</td>
<td>rwxo</td>
</tr>
</tbody>
</table>
Example 2

- Procedures `inc_ctr`, `dec_ctr`, `manage`
- Variable `counter`
- Rights `+`, `−`, `call`

<table>
<thead>
<tr>
<th>counter</th>
<th>inc_ctr</th>
<th>dec_ctr</th>
<th>manage</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>inc_ctr</code></td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>dec_ctr</code></td>
<td>−</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>manage</code></td>
<td></td>
<td>call</td>
<td>call</td>
</tr>
</tbody>
</table>
Boolean Expression Evaluation

• ACM controls access to database fields
  – Subjects have attributes
  – Verbs define type of access
  – Rules associated with objects, verb pair

• Subject attempts to access object
  – Rule for object, verb evaluated
  – Result controls granting, denying access
Example

- **Subject** annie
  - Attributes role (artist), groups (creative)
- **Verb** paint
  - Default 0 (deny unless explicitly granted)
- **Object** picture
  - Rule:
    
    ```
    paint: 'artist' in subject.role and 
    'creative' in subject.groups and 
    time.hour ≥ 0 and time.hour < 5
    ```
ACM at 3AM and 10AM

At 3AM, time condition met; ACM is:

```
annie

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

... picture ...

At 10AM, time condition not met; ACM is:

```
annie

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

... picture ...
History

Database:

<table>
<thead>
<tr>
<th>name</th>
<th>position</th>
<th>age</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>teacher</td>
<td>45</td>
<td>$40,000</td>
</tr>
<tr>
<td>Bob</td>
<td>aide</td>
<td>20</td>
<td>$20,000</td>
</tr>
<tr>
<td>Cathy</td>
<td>principal</td>
<td>37</td>
<td>$60,000</td>
</tr>
<tr>
<td>Dilbert</td>
<td>teacher</td>
<td>50</td>
<td>$50,000</td>
</tr>
<tr>
<td>Eve</td>
<td>teacher</td>
<td>33</td>
<td>$50,000</td>
</tr>
</tbody>
</table>

Queries:

1. \( \text{sum}(\text{salary}, \text{"position = teacher"}) = 140,000 \)

2. \( \text{sum}(\text{salary}, \text{"age > 40 & position = teacher"}) \)

   should not be answered (deduce Eve’s salary)
ACM of Database Queries

\[ O_i = \{ \text{objects referenced in query } i \} \]

\[ f(o_i) = \{ \text{read} \} \quad \text{for } o_i \in O_i, \text{ if } \forall k, |O_k - \bigcup_{j=1}^{i} O_j| > 1 \]

\[ f(o_i) = \emptyset \quad \text{for } o_i \in O_i, \text{ otherwise} \]

1. \[ O_1 = \{ \text{Alice, Dilbert, Eve} \} \text{ and no previous query set, so:} \]

\[ A[\text{asker, Alice}] = f(\text{Alice}) = \{ \text{read} \} \]

\[ A[\text{asker, Dilbert}] = f(\text{Dilbert}) = \{ \text{read} \} \]

\[ A[\text{asker, Eve}] = f(\text{Eve}) = \{ \text{read} \} \]

and query can be answered
But Query 2

From last slide:

\[ f(o_i) = \{ \text{read} \} \quad \text{for } o_i \in O_i, \text{ if } \forall k, |O_k - \bigcup_{j=1,\ldots,i; j \neq k} O_j| > 1 \]
\[ f(o_i) = \emptyset \quad \text{for } o_i \in O_i, \text{ otherwise} \]

2. \( O_2 = \{ \text{Alice, Dilbert} \} \) but \(|O_1 - O_2| = 1\) so
\[ A[\text{asker, Alice}] = f(\text{Alice}) = \emptyset \]
\[ A[\text{asker, Dilbert}] = f(\text{Dilbert}) = \emptyset \]
and query cannot be answered