Advanced Systems Security: Principles

Trent Jaeger
Systems and Internet Infrastructure Security (SIIS) Lab
Computer Science and Engineering Department
Pennsylvania State University
Authorization and Authentication

• Authentication
  ‣ Def: Verifying someone or something’s identity (subject)
  ‣ E.g., XSS content and media player code

• Authorization (also access control)
  ‣ Def: Deciding whether a subject can perform a requested operation on an object
  ‣ Deciding whether the media player can read content

• Combination
  ‣ Authentication is performed for authorization
Protection System

• Manages the authorization policy for a system
  ‣ Security goal

• It represents
  ‣ Protection state
  ‣ Protection state operations

• It describes what operations each subject (via their processes) can perform on each object
## The Access Matrix

- An access matrix is one way to represent policy.
  - Frequently used mechanism for describing policy.

- Columns are objects, subjects are rows.
- To determine if $S_i$ has right to access object $O_j$, find the appropriate entry.

- Succinct descriptor for $O$ (ISI*IOI) entries.
- Matrix for each right.

<table>
<thead>
<tr>
<th></th>
<th>$O_1$</th>
<th>$O_2$</th>
<th>$O_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>$S_2$</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>$S_3$</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>
Access Matrix Protection System

- Protection State
  - Current state of matrix

- Can modify the protection state
  - Via protection state operations
  - E.g., can create objects
  - E.g., owner can add a subject, operation mapping for their objects

- Lampson’s “Protection” paper
  - Can even delegate authority to perform protection state ops
XSS Problems

- Protection system approach is inadequate
- Processes can change their own permissions
  - Processes are untrusted, but can impact security goals
- Processes, files, etc. are created and modified
  - How can we manage their relationship with security goals
- What do we need to achieve necessary controls?
Define and Enforce Goals

• Claim: *If we can define and enforce a security policy that ensures security goals, then we can prevent such attacks*

• How do we know the policy expresses effective goals?
  ‣ Will look into this in depth later

• How do we know the enforcement mechanism will enforce policy as expected?
  ‣ Look into this today
Mandatory Protection System

- Is a *protection system* that can be modified only by *trusted administration* that consists of
  - A *mandatory protection state* where the protection state is defined in terms of a set of *labels* associated with subjects and objects
  - Label set is defined by trusted administration
  - A *labeling state* that assigns system subjects and objects to those labels in the mandatory protection state
  - A *transition state* that determines the legal ways that subjects and objects may be relabeled
- MPS is *immutable*
## Mandatory Protection System

![Diagram showing the Mandatory Protection System with states and permissions]

<table>
<thead>
<tr>
<th>Protection State</th>
<th>secret</th>
<th>unclassified</th>
<th>trusted</th>
<th>untrusted</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td>read</td>
<td>write</td>
<td>read</td>
<td></td>
</tr>
<tr>
<td>unclassified</td>
<td>read</td>
<td>write</td>
<td>read</td>
<td></td>
</tr>
<tr>
<td>trusted</td>
<td>write</td>
<td>read</td>
<td>write</td>
<td></td>
</tr>
<tr>
<td>untrusted</td>
<td>read</td>
<td>write</td>
<td>read</td>
<td>write</td>
</tr>
</tbody>
</table>

**Labeling State**

- Newfile

**File State**

- File: newfile
- File: acct

**Transition State**
Mandatory Protection State

- Immutable table of
  - Subject labels
  - Object labels
  - Operations authorized for former upon latter

- MPS for OS
  - Isolate media player from root processes
  - No network access

- MPS restricts media player to operations based on its label (trusted vs untrusted input)
Labeling State

- Immutable rules mapping
  - Subjects to labels (in rows)
  - Objects to labels (in columns)

- Labeling State of OS
  - Browser, Media Player have own subject labels
  - Label inputs from network (network connection)
  - Root and TCB program files have labels based on their trust

- Defines requirements for determining labels of all objects
Transition State

- Immutable rules mapping
  - Subject labels to conditions that change their subject labels
  - Object labels to conditions that change their object labels

- Transition State of OS
  - Change label of processes that receive untrusted input
  - Change label of files that receive secret input

- Transition State of Programs
  - Server, Browser, Media Player may be trusted to change their labels (down only?)
Managing MPS

• Challenge
  ‣ Determining how to set and manage an MPS in a complex system involving several parties

• Parties
  ‣ What does programmer know about deploying their program securely?
  ‣ What does an OS distributor know about running a program in the context of their system?
  ‣ What does an administrator know about programs and OS?
  ‣ Users?
Reference Monitor

• Purpose: Ensure enforcement of security goals
  ‣ Mandatory protection state defines goals
  ‣ Reference monitor ensures enforcement

• Every component that you depend upon to enforce your security goals must be a reference monitor
Reference Monitor

- Components
  - Reference monitor interface (e.g., LSM)
  - Authorization module (e.g., SELinux)
  - Policy store (e.g., policy binary)
Reference Monitor Guarantees

- **Complete Mediation**
  - The reference validation mechanism must always be invoked

- **Tamperproof**
  - The reference validation mechanism must be tamperproof

- **Verifiable**
  - The reference validation mechanism must be subject to analysis and tests, the completeness of which must be assured
Complete Mediation

• Every security-sensitive operation must be mediated
  ‣ What’s a “security-sensitive operation”?
  ‣ E.g., operation that enables a subject of one label to access an object that may be a different label

• How do we validate complete mediation?
  ‣ Every such operation must be identified
  ‣ E.g., we can check for dominance of mediation

• **Mediation**: Does interface mediate?
• **Mediation**: On all resources?
• **Mediation**: Verifably?
Tamperproof

- Prevent modification by untrusted entities
  - Interface, mechanism, policy of reference monitor
  - Code and policy that can affect reference monitor mods

- How to detect tamperproofing?
  - Transitive closure of operations
  - Challenge: Often some untrusted operations are present

- Tamperproof: Is reference monitor protected?
- Tamperproof: Is system TCB protected?
Verification

- Test and analyze reference validation mechanism
  - Does code do its job correctly?
  - Does policy do its job correctly?
- Determine correctness of code and policy
  - What defines correct code?
  - What defines a correct policy?
- **Verifiable**: Is TCB code base correct?
- **Verifiable**: Does the protection system enforce the system’s security goals?
Evaluation

• **Mediation**: Does interface mediate?
• **Mediation**: On all resources?
• **Mediation**: Verifably?
• **Tamperproof**: Is reference monitor protected?
• **Tamperproof**: Is system TCB protected?
• **Verifiable**: Is TCB code base correct?
• **Verifiable**: Does the protection system enforce the system’s security goals?
Multiple Reference Monitors

- The reference monitor concept approach was designed with a centralized reference validation mechanism in mind
  - What about the case where there are several such mechanisms grouped together?
Take Away

• Mandatory Protection System
  ‣ Means to define security goals that applications cannot impact

• Reference Monitor Concept
  ‣ Requirements for a reference validation mechanism that can correctly enforce an MPS
  ‣ NOTE: This will be a major focus of this course

• Until we come up with coherent approach to defining MPS and validating reference monitor guarantees, we will continue to have insecure systems
  ‣ That is the challenge of systems security research