Topics in Systems and Program Security

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

August 29, 2008
Systems Enable Interaction

- If it was solely about isolating processes, security would be easy
- However, process interaction is fundamental to operating systems
  - How can processes interact?
  - For what purposes?

- Challenge: ensure security goals are met given all means of interaction
Secure Operating System

• Provides security mechanisms that ensure that the system’s security goals are enforce despite threats from attackers
  ‣ Security mechanisms?
  ‣ Security goals?
  ‣ Threats?
  ‣ Attackers?

• Can we build a truly secure operating system?
Security Goals

- Lots of unsatisfying definitions
  - Users can perform only authorized operations (safety)
  - Processes perform only their necessary operations (least privilege)
  - Operations can only permit information to be written to more secret levels (MLS)

- We’ll discuss these
  - Defining practical and achievable security goals is a difficult task
Trust Model

• For operating system
  ‣ Trust model == TCB

• What’s in a TCB?

• What are we trusting?
Threat Model

• Threats are means that an attacker can use to violate security goals
  ‣ Where do threats come from?
  ‣ What mechanisms enable threats?
  ‣ What do threats threaten?

• Secure OS must protect TCB against threats
  ‣ Why is this sufficient?
Security Model

- Composed from Trust Model and Threat Model
- Can we state a security model for an idealized system?
  - Two processes
  - One root process
  - OS provides information flow (interaction) mechanisms
  - OS depends on the root process to identify the subjects for the processes
Protection System

- Manages the access control policy for a system
  - Security goal
- It presents
  - 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*|OI) 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>
Protection State

• Using an access matrix representation
  ‣ Current state of matrix

• Can modify the protection state
  ‣ Via protection state operations
  ‣ E.g., can create subjects and objects
  ‣ E.g., owner can add a subject, operation mapping for their objects
Protection Domain

• Specifies the objects that a subject can access and the operations the subject can perform upon those objects
  ▸ What is this in the access matrix?

• Capabilities and Access Control Lists
  ▸ How do these define domains?
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
Example

• 2 subjects

• Mandatory protection state
  ‣ Subject secret has a secret file
  ‣ Subject public has a public file

• What happens when subject secret creates a new file?
  ‣ What happens to the access matrix?
  ‣ What if the subject public creates a file?

• What happens when subject public executes a new process?
  ‣ Suppose the process is trusted to access secret files
  ‣ How does it obtain its label?
Mandatory Protection System

```
<table>
<thead>
<tr>
<th></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
File: newfile
File: acct
Transition State
Protection State

Process: newproc
Process: other
Reference Monitor

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

- Purpose: Ensure enforcement of security goals
  - Mandatory protection state defines goals
  - Guarantees ensure enforcement
Secure Operating System

• Possible?
• Ideally, satisfies the reference monitor guarantees
  ‣ Is that so hard?
• Mediation
  ‣ Challenges: what’s an operation?
• Tamperproof
  ‣ Challenges: Trust is rampant
• Verifiable:
  ‣ Challenges: Code verification? What’s the goal?
Evaluation

- **Mediation**: Does interface mediate correctly?
- **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?
Take Away

• Identify core security approach
  ‣ Goals, trust model, threat model, security model

• Secure OS analogues
  ‣ Goals == protection system
  ‣ Trust model == TCB
  ‣ Threat model -- Mediated by Reference Monitor
  ‣ Security model -- how the reference monitor of the TCB enforces the mandatory protection system