CS260 – Advanced Systems Security

Security Principles

April 9, 2025

Access Control – The Right Way

- We said that ordinary operating systems cannot control code controlled by an adversary
- Review formalisms developed for “protection”
  - and show how they are extended to enforce “security”
- Key concepts
  - Mandatory protection state
    - Adversary cannot modify access control policy
    - Only system
  - Reference monitor
    - Enforce access control comprehensively

Protection System

- Manages the authorization policy for a system
  - It describes what operations each subject (via their processes) can perform on each object
- Consists of
  - **State:** *Protection state*
  - **State Ops:** *Protection state operations*

Access Matrix

- Using the Access Matrix
- (1) Suppose J wants to protect a private key (object O_1) from being leaked to or modified by others
- (2) Suppose J wants to prevent a public key (object O_2) from being modified by others
- Design the access matrix
- Cannot protect these resources in a traditional protection system

	O_1	O_2	O_3
J	?	?	?
S	?	?	?
S	?	?	?

Protection System

- Claim: Traditional protection system is insufficient to enforce security
- Problem: in a traditional protection system we cannot determine whether an unauthorized operation will ever be allowed
  - Called the Safety Problem

Protection System

- Claim: Traditional protection system is insufficient to enforce security
- Problem: in a traditional protection system we cannot determine whether an unauthorized operation will ever be allowed
  - Called the Safety Problem
- Found to be undecideable for traditional protection systems – HRU paper
  - Can expand infinitely
- Bigger problem is that adversary can modify state

Access Matrix

- Using the Access Matrix
- (1) Suppose J wants to protect a private key (object O_1) from being leaked to or modified by others
- (2) Suppose J wants to prevent a public key (object O_2) from being modified by others
- Design the access matrix
- Can we change the protection system to protect these resources?

	O_1	O_2	O_3
J	?	?	?
S	?	?	?
S	?	?	?

Protection System Problems

- Protection system approach is inadequate for security
  - What are its fundamental limitations?

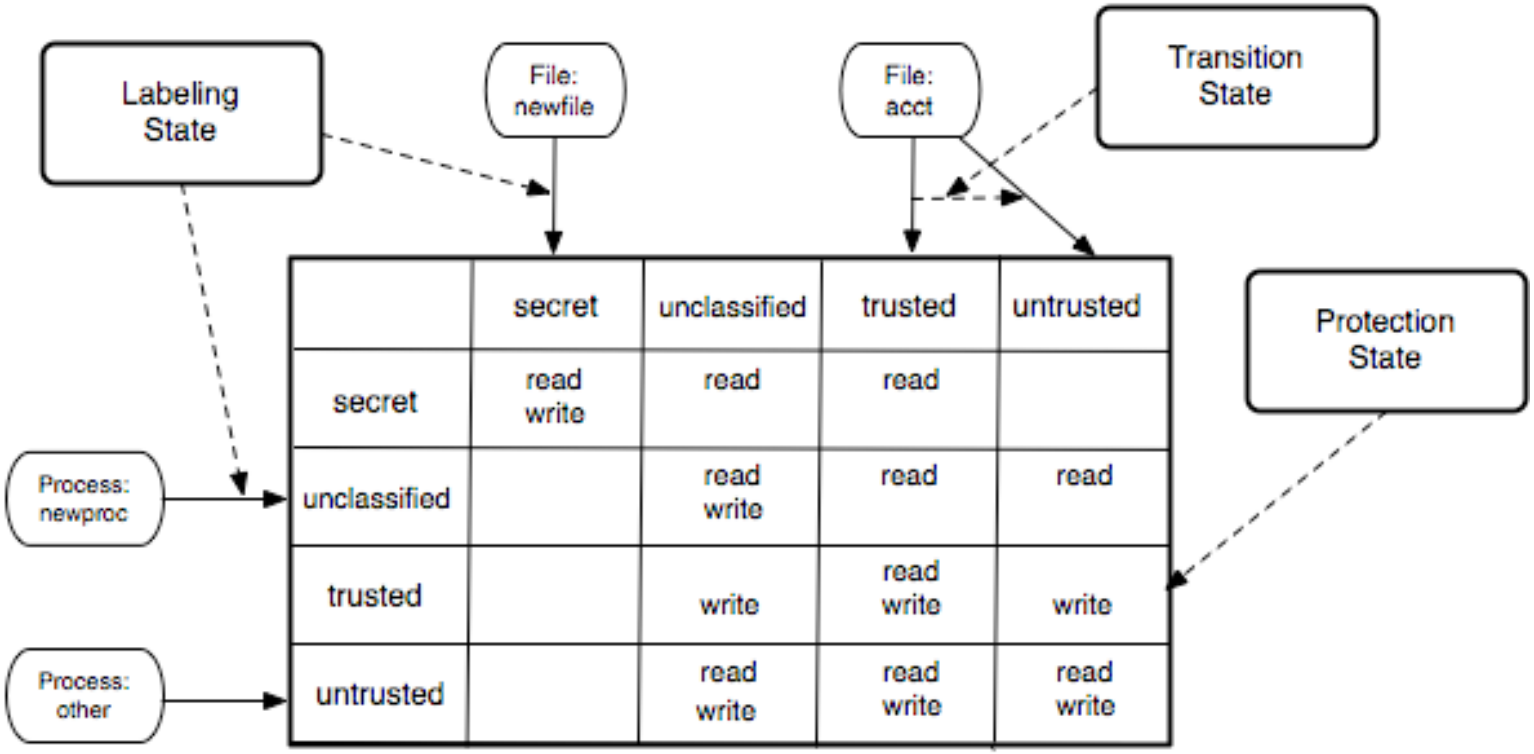
Protection System Problems

- Protection system approach is inadequate for security
  - What are its fundamental limitations?
- Processes can change their own permissions
  - Processes may become untrusted, but can modify policy
- Processes, files, etc. are created and modified
  - Cannot predict in advance
  - Forever (safety problem)
- What do we need to achieve necessary controls?

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 an immutable set of *labels* and the *operations that subject labels can perform on object labels*
  - 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* to (untrusted) user-space processes

Mandatory Protection System



Mandatory Protection State

- Immutable table of
  - Subject labels
  - Object labels
  - Operations authorized for former upon latter
- How can you use an MPS to control use of bad code?

MPS Access Matrix

- Using the MPS Access Matrix
- (1) Suppose J wants to protect a private key (object O_1) from being leaked to or modified by others
- (2) Suppose J wants to prevent a public key (object O_2) from being modified by others
- How do you protect these results with an MPS?

	O_1	O_2	O_3
J	?	?	?
S	?	?	?
S	?	?	?

Mandatory Protection State

- Can we leverage user identities as subject labels?
- Can we use files as object labels?

MPS Access Matrix

- Using the MPS Access Matrix
- Only program that creates keys is given write access to key files (K1)
  - Must be trusted – should be vetted
- Only signature program is given read access to private keys (K2)
  - Must be trusted – should be vetted
- Any other programs (P) have read access to the public key, if needed
- Limit to user?

	SK	PK	OL
K1	?	?	?
K2	?	?	?
P	?	?	?

Labeling?

- Immutable table of
  - Subject labels
  - Object labels
  - Operations authorized for former upon latter
- How do subjects (processes) get their labels?

Labeling State

- Labeling state consists of a set of immutable rules mapping…
  - Subjects to labels (in rows)
  - Objects to labels (in columns)

Labeling State

- Using the MPS Access Matrix
- (1) Suppose J wants to protect a private key (Object O_1) from being leaked to or modified by others
- (2) Suppose J wants to prevent a public key (Object O_2) from being modified by others
- What is the labeling state for this case?

	SK	PK	OL
K1	?	?	?
K2	?	?	?
P	?	?	?

Labeling State

- Using the Labeling State
- Programs are assigned labels
  - Key generation (K1)
  - Private key signatures (K2)
  - Others (P)
- Public and private keys are given different labels from fixed set
  - Public key (PK)
  - Private key (SK)
  - Others (OL)

	SK	PK	OL
K1	?	?	?
K2	?	?	?
P	?	?	?

Transition State

- Immutable rules mapping
  - Subject labels to conditions that change their subject labels
  - Object labels to conditions that change their object labels
- How can you use the transition state to control bad code?

Transition State

- How do we launch a program to generate a signature?
  - E.g., from a shell (P) to execute a program as (K2)
- Fork/exec generates a new process based on the shell
  - Same label as the parent by default (P)

	SK	PK	OL
K1	?	?	?
K2	?	?	?
P	?	?	?

Transition State

- How do we launch a program to generate a signature?
  - E.g., from a shell (P) to execute a program as (K2)
- Fork/exec generates a new process based on the shell
  - Same label as the parent by default (P)
- Transition rule for a process:
  - Relates the current label (P) and the label of the file being exec'd (K2_Exec) to the result label of the process (K2)

	SK	PK	OL
K1	?	?	?
K2	?	?	?
P	?	?	?

Managing MPS

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

Managing MPS

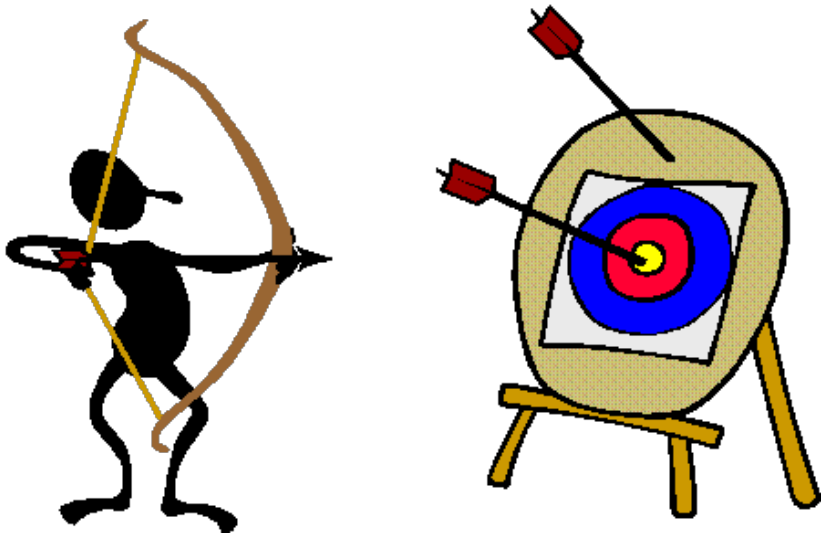
- Challenge
  - Determining how to set and manage an MPS in a complex system involving several parties
- Parties
  - What does programmer know about deploying their program securely?
  - What does an OS distributor know about running a program in the context of their system?
  - What does an administrator know about programs and OS?
  - Users?

Managing MPS

- Current methods use dynamic analysis to setup MAC policies – run the program and collect the permissions used
  - Really a functional policy

Reference Monitor

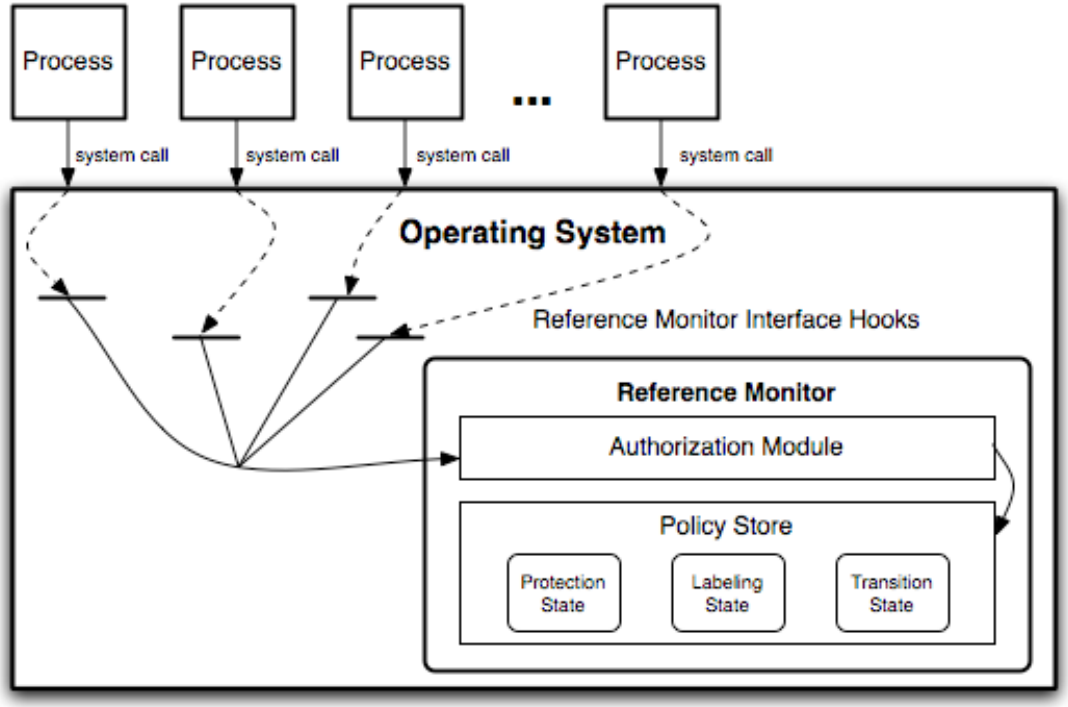
- Purpose: Ensure enforcement of security goals
  - Define goals in the mandatory protection system
  - Reference monitor ensures enforcement



- *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)
  - Reference validation mechanism (e.g., SELinux)
  - Policy store (e.g., policy database)



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 is a “security-sensitive operation”?

Complete Mediation

- Every security-sensitive operation must be mediated
  - What is a “security-sensitive operation”?
  - E.g., operation that may not be authorized for every subject
- How do we validate complete mediation?

Complete Mediation

- Every security-sensitive operation must be mediated
  - What is a “security-sensitive operation”?
  - E.g., operation that may not be authorized for every subject
- How do we validate complete mediation?
  - Every security-sensitive operation must be identified
  - E.g., ensure every execution of that operation is checked
- **Mediation**: Does interface mediate?
- **Mediation**: On all resources?
- **Mediation**: Verifably to enforce security goals?

Tamperproof

- Prevent modification by untrusted entities
  - Prevent modification of what?

Tamperproof

- Prevent modification by untrusted entities
  - Prevent modification of what?
  - Code and data that can affect reference monitor
- How to detect tampering?

Tamperproof

- Prevent modification by untrusted entities
  - Prevent modification of what?
  - Code and data that can affect reference monitor
- How to detect tampering?
  - Challenge: Often some untrusted operations are present
- **Tamperproof**: Is reference monitor protected?
- **Tamperproof**: Is system TCB protected?
- **Tamperproof**: Is the MPS protected?

Verification

- Determine correctness of code and policy
  - What defines correct code?
  - What defines a correct policy?
- Test and analyze reference validation mechanism
  - Does code/policy do its job correctly?
  - For all executions (completeness must be assured)
- **Verifiable**: Is TCB code base correct?
- **Verifiable**: Does the MPS enforce the system's security goals?

Define and Enforce Goals

- Claim: *If we can define and enforce security policy that ensures security goals for all executions, then we can prevent attacks*
- How do we know what policy meets security goals?
  - How do we write a policy for all executions?
- How do we know the enforcement mechanism will enforce policy as expected?
  - Look into this today
- How do we know the policy expresses effective goals?
  - Will look into this in depth later

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 validating that an MPS meets security goals and validating reference monitor guarantees, we will continue to have insecure systems*
  - That is the challenge of systems security research

Questions