Outline

- Information Protection
  - Information Protection in Computer Systems
- Q&A
Information Protection Basics

- **Key Concern**
  - Multiple Use

- **System Requirement**
  - Implementing Desired Authority Structure

- **Terms**
  - **Security**
    - Controlling who may use or modify a system and information stored in it
  - **Protection**
    - Controlling access of programs to information

- **Goal**
  - Preventing All Unauthorized Use of Information
Information Protection

- Information Sharing Models
  - Multiuser System
  - Capability System
  - Access Control List System

- Essentials
  - Information Divided into Mutually Exclusive Partitions as Fundamental Objects
    - Authentication

Direct Access to Information
Multiuser System

- Use of a Descriptor Register (Base & Bound) for Each Program
  - Privileged State Bit
    - Indication of the program to load the register
    - Protection of the bit
  - Simple Authority Check on a Request to Access Memory

- Authentication
  - Password
    - With defects lying in its choice & exposure
  - Unforgeable Object
    - With weakness of having to keep the resulting bit stream secret
  - Encipherment/Decipherment

- Verifying the User at a Terminal When Associating the Terminal with a Virtual Machine
Use of a descriptor register to simulate multiple virtual machines. Program C is in control of the processor. The privileged state bit has value OFF, indicating that program C is a user program. When program S is running, the privileged state bit has value ON. In this (and later) figures, lower addresses are nearer the bottom of the figure.
Multiuser System (Cont’d)

Information Sharing

- **List-Oriented Mechanism (with Costly Associative Matching)**
  - Guard holding a list of IDs of authorized users
    - E.g., a store clerk checking list of credit customers
  - Checking at the access request time

- **Ticket-Oriented Mechanism**
  - Guard holding the description of a single ID
    - E.g., a locked door that opens with a key (ticket)
  - Checking at the information selection time

Practical Combination of a List-Oriented System at the Human Interface and a Ticket-Oriented Mechanism in the Underlying H/W

What to be Protected: Information, the Guard’s Authorization Information, Association between a User and the Label or Set of Tickets
Multiuser System (Cont’d)

- Principle of Least Privilege
  - Use of Different Principals Depending on the Purposes
- Importance of Authentication

A Principal is an Entity Accountable for the Activities of a Virtual Process

Authentication Has Allowed the Virtual Process to Enter the Domain of the Principal.

List-Oriented System

All Objects That the Principal Has Been Authorized to Use
Sharing of a Routine
Sharing Implications

- Overwriting

- Shared Area Modifications
  - Shared Routine’s Writing into Private Areas
  - Need for Generalization
    - Capability Systems (Ticket-Oriented)
    - Access Control List Systems (List-Oriented)
Separation of Addressing & Protection

- **System Address Space**
  - Consisting of All Segments (Storage Areas)
  - Each segment with a distinct name, scope, and protection

- **Processor Address Space**
  - Defined by the Protection Descriptors

These Descriptors Are First Reloaded at a Control Switch
Separation of Addressing and Protection Descriptors
Capability System

- Tagged Architecture
  - Memory Storing Protection Descriptor Values or Capabilities (with the Tag Bits On) as Well as Ordinary Data Values
  - Processor Directed to Load a Capability and then Addressing the Space
    - Supervisor Initially Starting a Processor for User Identification Using a Table (Authentication)
Simple Capability System
Capability System with Provision for Authentication
Dynamic Authorization of Sharing

Protection for Authorization Changing Mechanism (Copying of a Capability)

- **Assumption**
  - IDs previously transmitted in an external communication

- **Method: Authority Check**
  - Comparison of an inside principal ID (e.g., name) with outside authorization information

- **Issues**
  - Single mailbox segment
  - Revocation with capability-holding segments and revocable indirect objects
  - Preventing propagation with a copy bit and a depth counter