CS370 Operating Systems

Colorado State University
Yashwant K Malaiya

Fall 2025 L26
Security and Protection

Slides based on

- **Text by Silberschatz, Galvin, Gagne**
- **Various sources**

Project Updates

Option A

- **Project Slides** and videos for both section need to be posted 24 hours in advance of the viewing period.
- Sec 001: Groups 1A -16A: post slides & videos by Sun 12/08/25 2 PM
- Sec 001: 17A - 33A: post slides & videos by Tues 12/010/25 2 PM
- Sec 801: All Groups post slides & videos by Mon 12/09/25 2 PM

Option B

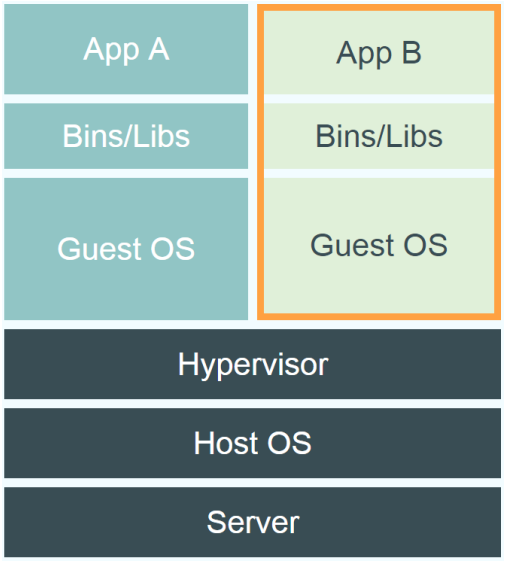
- Sec 001: Schedule your 15-min demo (Mon-Wed) with the assigned TA by Thurs 12/04/25.
- Sec 801: Post your slides and videos by Tues 12/010/25 2 PM.
- Please follow instructions for videos, including responding to specific questions and being visible in the video. No demos schedules with the TAs.

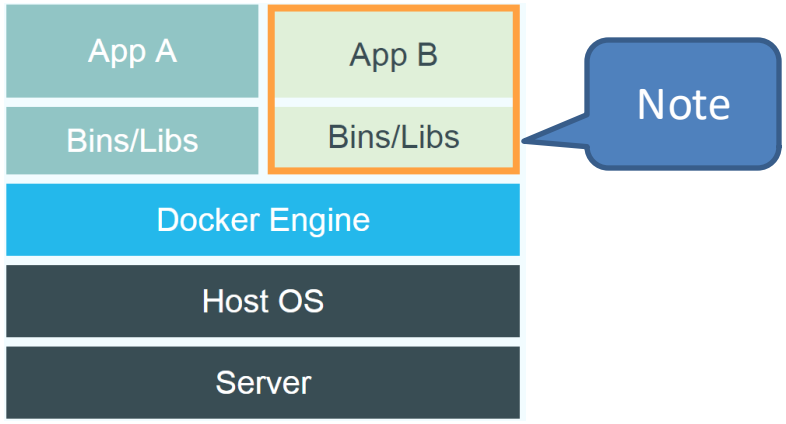
Sec 801 online Option B

- No need to schedule demos with TAs.
- Please see Option B: Project Presentation Requirements for 801 Section, including
  - video should not exceed 15
  - at the beginning of the video please do a brief introduction of yourself and your teammates
  - identify each component of the hardware by pointing to them to show that it satisfies the requirements (sensors, actuators, computer it is communicating with, etc.)
- Please address these questions:
  - What was your biggest takeaway from doing this project?
  - biggest hurdle that you encountered while developing the project?
  - Does the project have any aspects that were not initially part of the plan (i.e. you had to add something you didn't expect to make it work)?
  - Are there any aspects of your implementation that you think will stand out compared to similar projects?
  - What two specific attributes of your finished project have you evaluated? How?

Linux Containers and Docker

- Linux containers (LXC 2008) are “lightweight” VMs
- Comparison between LXC/docker (2013) and VM





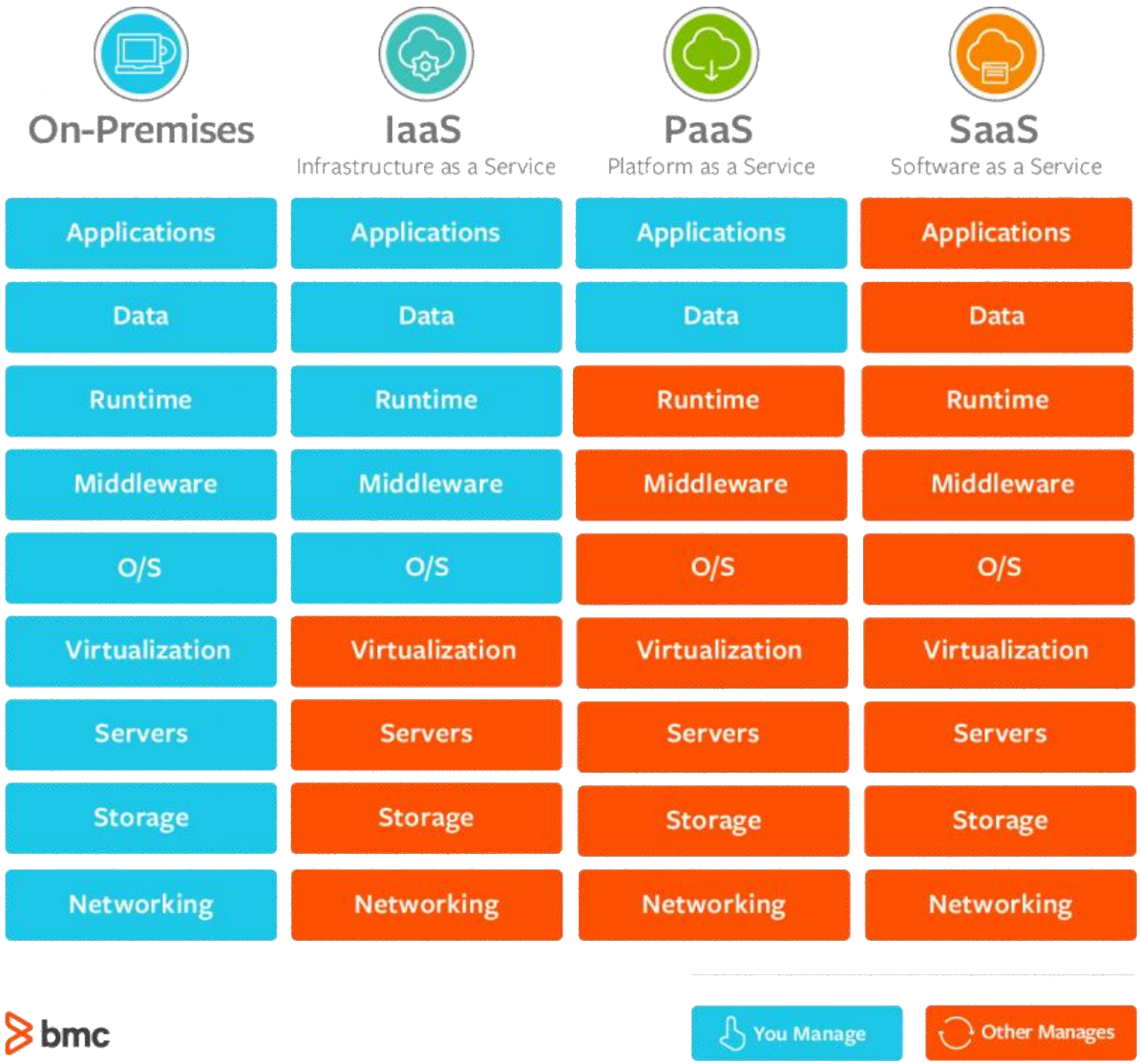
- Containers provide “OS-level Virtualization” vs “hardware level”.
- Containers can be deployed in seconds.
- Very little overhead during execution, even better than Type 1 VMM.

The cloud Service Models

Service models

- IaaS: Infrastructure as a Service
  - infrastructure components traditionally present in an on-premises data center, including servers, storage and networking hardware
  - e.g., Amazon EC2, Microsoft Azure, Google Compute Engine
- PaaS: Platform as a Service
  - supplies an environment on which users can install applications and data sets
  - e.g., Google AppEngine, Heroku, Apache Stratos
- SaaS: Software as a Service
  - a software distribution model with provider hosted applications
  - Microsoft Office365, Amazon DynamoDB, Gmail

The Service Models



https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/

bmc

Colorado State University

Cloud Management models

- **Public clouds**
  - Utility model
  - Shared hardware, no control of hardware,
  - Self-managed (e.g., AWS, Azure)
- **Private clouds**:
  - More isolated (secure?)
  - Federal compliance friendly
  - Customizable hardware and hardware sharing
- **Hybrid clouds**:
  - a mix of on-premises, private cloud and third-party, public cloud services.
  - Allows workloads to move between private and public clouds as computing needs and costs change.

CS370 Operating Systems

Colorado State University

Yashwant K Malaiya

Security

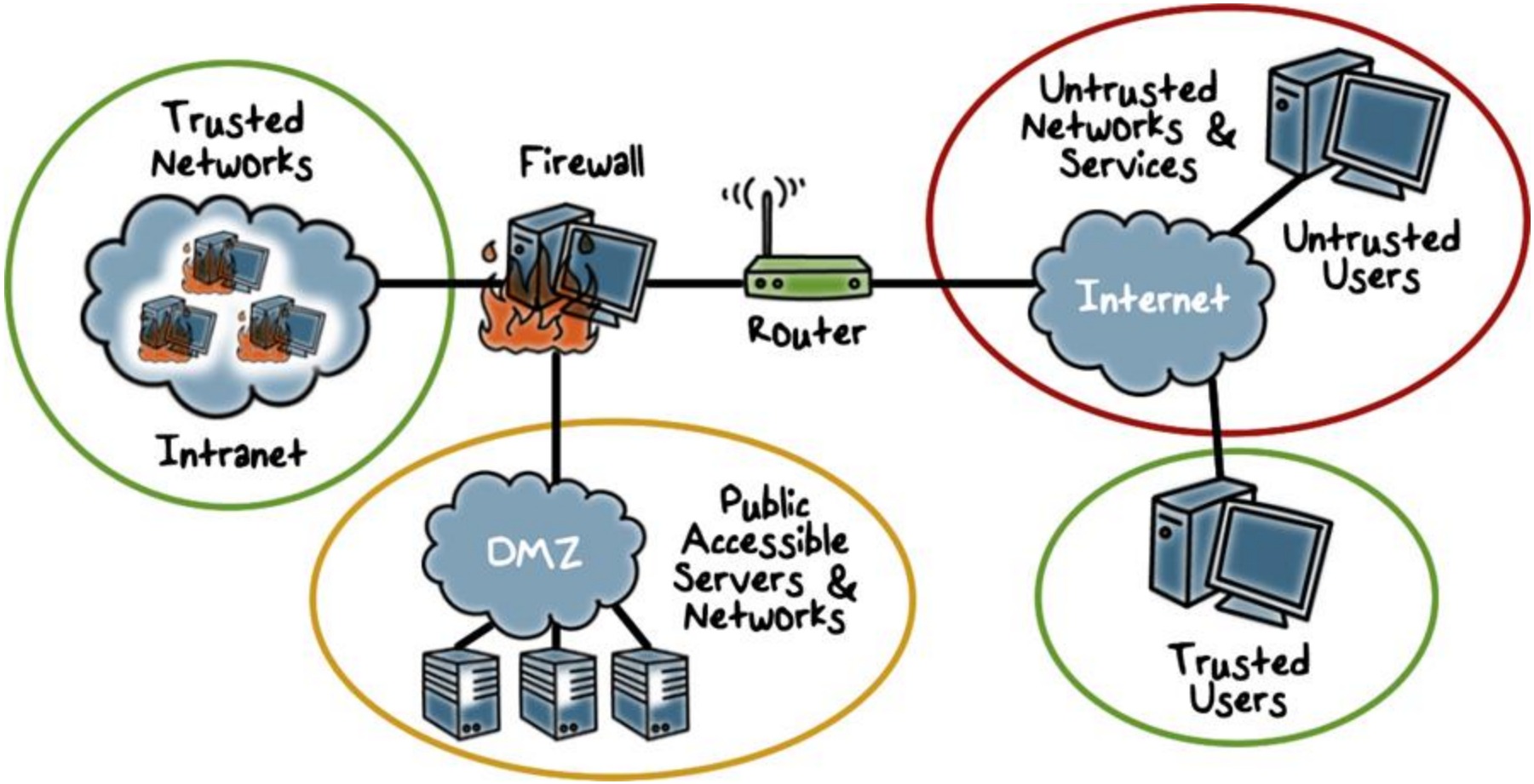
Slides based on

- Various sources

Security System Architecture

- Networked systems
  - Use of firewalls: Organization wide and system level
  - Address translation
  - Isolation of systems
- Single computing System: OS
  - Multiple levels of priviledges
  - Isolation of
    - processes,
    - cgroups,
    - virtual machines

Firewalls



DMZ: “Demilitarized zone”, distributed firewalls, From Georgia Tech
Note multiple levels of trust.

Firewalls

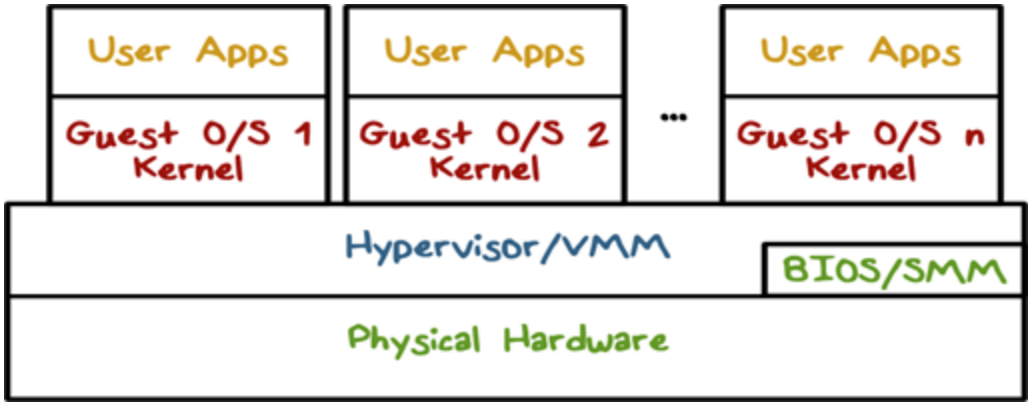
- A firewall checks traffic (packets or sessions) passing through it
- Can be programmed to check address ranges (IP addresses, ports), protocols, applications and content types.
- Can provide address translation, encryption

OS - trusted computing base

- The operating system serves as as trusted computing base (TCB) that controls access to protected resources.
  - Must establish the source of a request for a resource (authentication is how we do it)
  - Authorization or access control
  - Mechanisms that allow various policies to be supported
- How
  - Hardware support for memory protection
  - Processor execution modes (system and user modes)
  - Privileged instructions - can only be executed in system mode
  - System calls - transfer control between user and system code

Isolation in a system

- OS isolates address spaces of different processes using address translation. Also data vs code isolation.
  - Page tables governed by OS.
- In virtualization, hypervisor isolates virtual machines.
- Containers (Docker): Linux cgroups isolate process groups.



Malware

- Malware ("malicious software"):
  - a catch-all term for any type of malicious software,
  - regardless of how it works, its intent, or how it's distributed.
- Virus
  - a specific type of malware that self-replicates by inserting its code into other programs. Types:
  - The **file infector** can burrow into executable files and spread through a network. A file infector can overwrite a computer's operating system or even reformat its drive.
  - The **macro virus** takes advantage of programs that support macros. Macro viruses usually arrive as Word or Excel documents attached to a spam email, or as a zipped attachment.
  - **Polymorphic viruses** modify their own code. The virus replicates and encrypts itself, changing its code just enough to evade detection by antivirus programs.

https://www.mcafee.com/enterprise/en-in/security-awareness/ransomware/malware-vs-viruses.html

Malware: Functional types

- **Worm:** a standalone program that can self-replicate and spread over a network. Unlike a virus, a worm spreads by exploiting a vulnerability in the infected system or through email as an attachment masquerading as a legitimate file.
- **Ransomware**: demands that users pay a ransom—usually in bitcoin or other cryptocurrency—to regain access to their computer.
- **Scareware**: attempts to frighten the victim into buying unnecessary software or providing their financial data.
- **Adware and spyware**: Adware pushes unwanted advertisements at users and spyware secretly collects information about the user. Spyware may record the websites the user visits, information about the user's computer system and vulnerabilities for a future attack, or the user's keystrokes.
  - Spyware that records keystrokes is called a keylogger.
- **Fileless malware**: Unlike traditional malware, fileless malware does not download code onto a computer, so there is no malware signature for a virus scanner to detect. Instead, fileless malware operates in the computer's memory and may evade detection by hiding in a trusted utility, productivity tool, or security application.

https://www.mcafee.com/enterprise/en-in/security-awareness/ransomware/malware-vs-viruses.html

CS370 Operating Systems

Colorado State University
Yashwant K Malaiya
Fall 2025

Access control

Slides based on

- Various sources

Access Control

Definition according to RFC 4949:

> "a process by which use of system resources is regulated according to a security policy and is permitted only by **authorized entities** (*users, programs, processes, or other systems*) according to that policy"

RFC 4949 defines security as

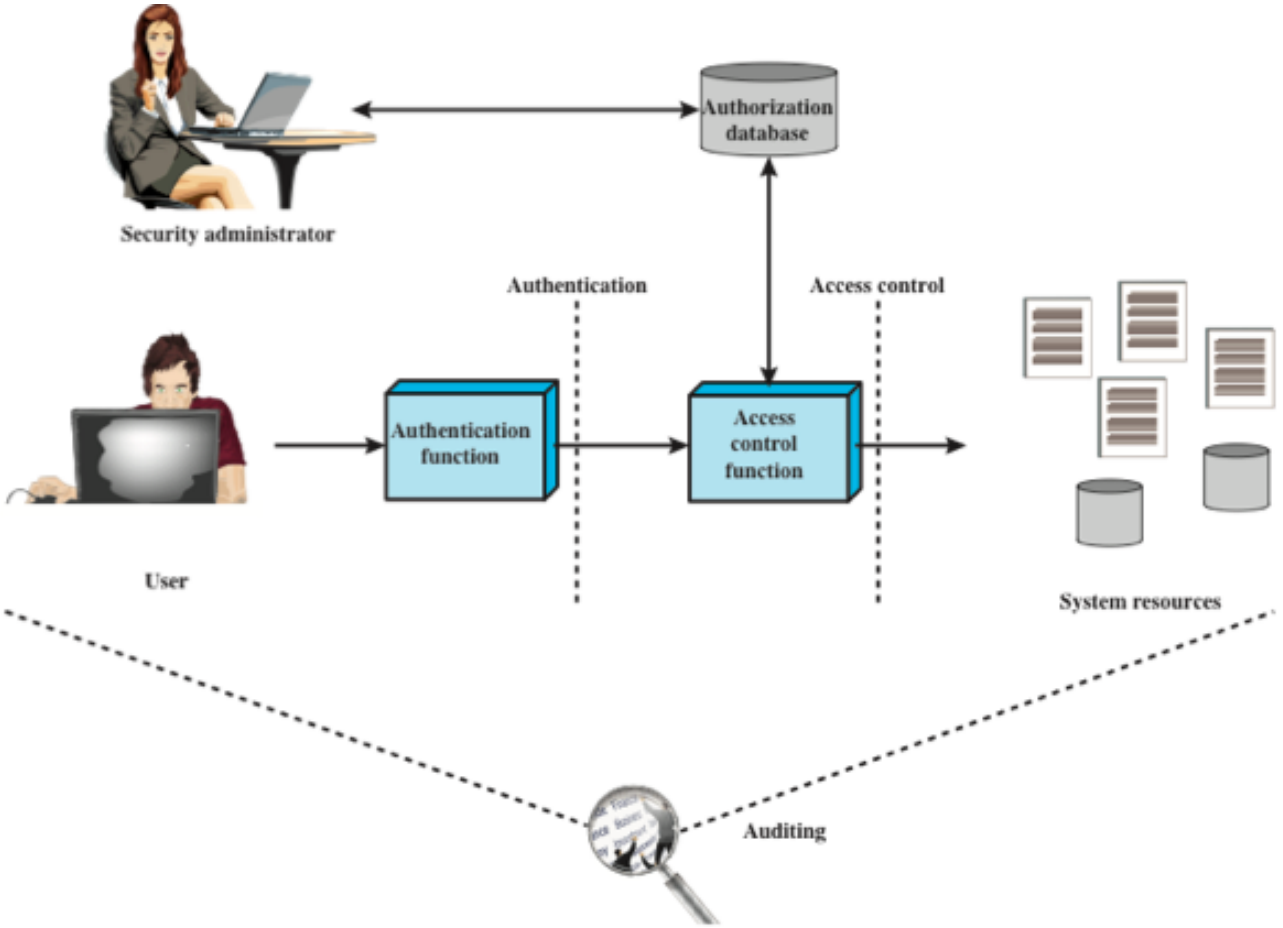
> "measures that implement and assure security services in a computer system, particularly those that assure access control service"

Thus all of computer security is concerned with access control.

Enforced by the Trusted Computing Base (OS) which performs

- authentication
- authorization

Access Control as a Security Function



Principles of Access Control

- Guiding principle – **principle of least privilege**
  - Programs, users and systems should be given just enough **privileges** to perform their tasks
- **Compartmentalization** a derivative concept regarding access to data
    - Process of protecting each individual system component through the use of specific permissions and access restrictions
- Can be
  - static (during life of system, during life of process)
  - Or dynamic (changed by process as needed) – access **domain switching**, **privilege escalation**

Principles of Access Control

- Rough-grained or Fine-grained management
  - Rough-grained privilege management easier, simpler, but least privilege now done in large chunks
    - For example, traditional Unix processes either have abilities of the associated user, or of root
  - Fine-grained management more complex, more overhead, but more protective
    - File ACL lists, RBAC
- Domain can be user, process, procedure
- **Audit trail** – recording all protection-orientated activities, important to understanding what happened, why, and catching things that shouldn't
- No single principle is a panacea for security vulnerabilities – need **defense in depth**

Subjects, Objects, and Access Rights

- A **subject** is an entity capable of accessing objects.
  - represented by a process. A user/application gains access to an object by means of a process that represents that user/application. The process takes on the attributes of the user, such as access rights.
  - Held accountable for the actions.
  - Classes: Owner (u in linux), Group (g), World (o), people with specific roles
- An **object** is a resource to which access is controlled.
  - an entity used to contain and/or receive information.
  - Examples: pages/segments, files/directories/programs, ports, devices etc.

Subjects, Objects, and Access Rights

An access right describes the way in which a subject may access an object.

- **Read**: User may view information in a system resource (e.g., a file, selected records in a file, selected fields within a record, or some combination). Read access includes the ability to copy or print.
  - Directory: ability to list the directory.
- **Write**: User may add, modify, or delete data in system resource (e.g., files, records, programs). Write access includes read access.
  - Directory: create new files
- **Execute:** User may execute specified programs.
  - Directory: enter it to access the files within it.

- Delete: User may delete certain system resources, such as files or records.
- Create: User may create new files, records, or fields.
- Search: User may list the files in a directory or otherwise search the directory.

Access Control Schemes

Discretionary Access Control (DAC): Scheme in which an entity may be granted access rights that permit the **owner** entity, by its own violation, to enable another entity to access some resources.

- Provided using an access matrix

Mandatory Access Control: **Centralized authority** sets security policy for all resources

- Example: SELinux

Role based access control (RBAC): Access depnds on the current role of the uset

Unix Access Control

- Subjects (Who?)
  - Users
- Objects (What?)
  - Files, directories
  - Files: sockets, pipes, hardware devices, kernel objects, process data
- Access Operations
  - Read, Write, Execute
  - Set by root or owner of the object
- Linux is an example of *discretionary access control*.
  - Resource owners can set the security policy for objects they own
- Superuser (root) allowed to do anything.
  - System administrators assume superuser role to perform privileged actions – Good practice to assume superuser role only when necessary

Example: Access Control Matrix

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

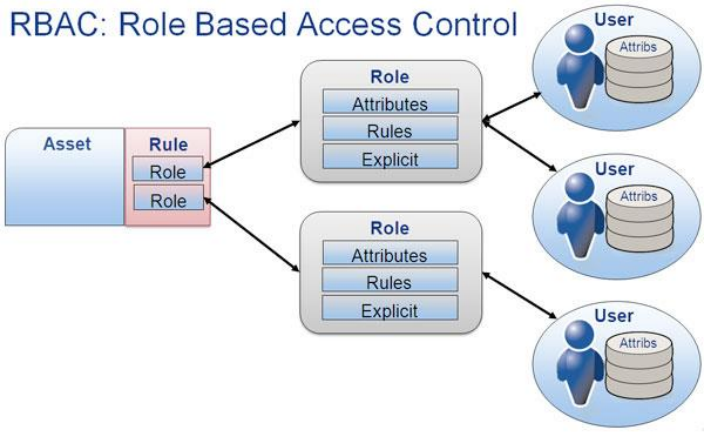
(a) Access matrix

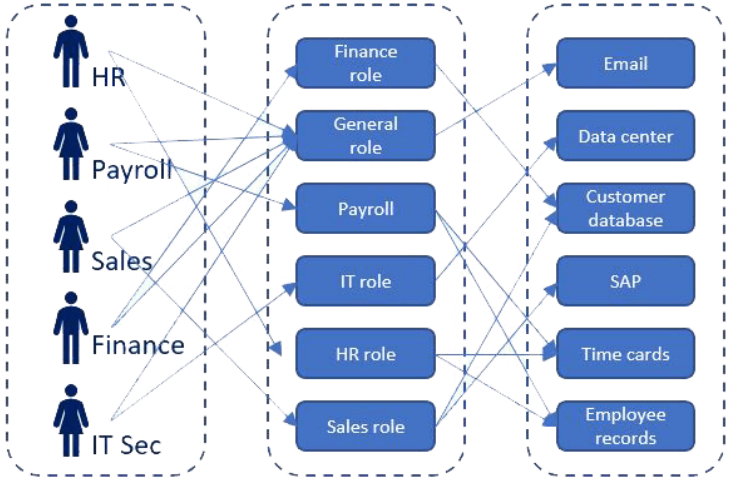
Access Control List (ACL): Every object has an ACL that identifies what operations subjects can perform. Each access to object is checked against object's ACL.

May be kept in a relational database. Access recorded in file metadata (inode).

Role Based Access Control (RBAC)

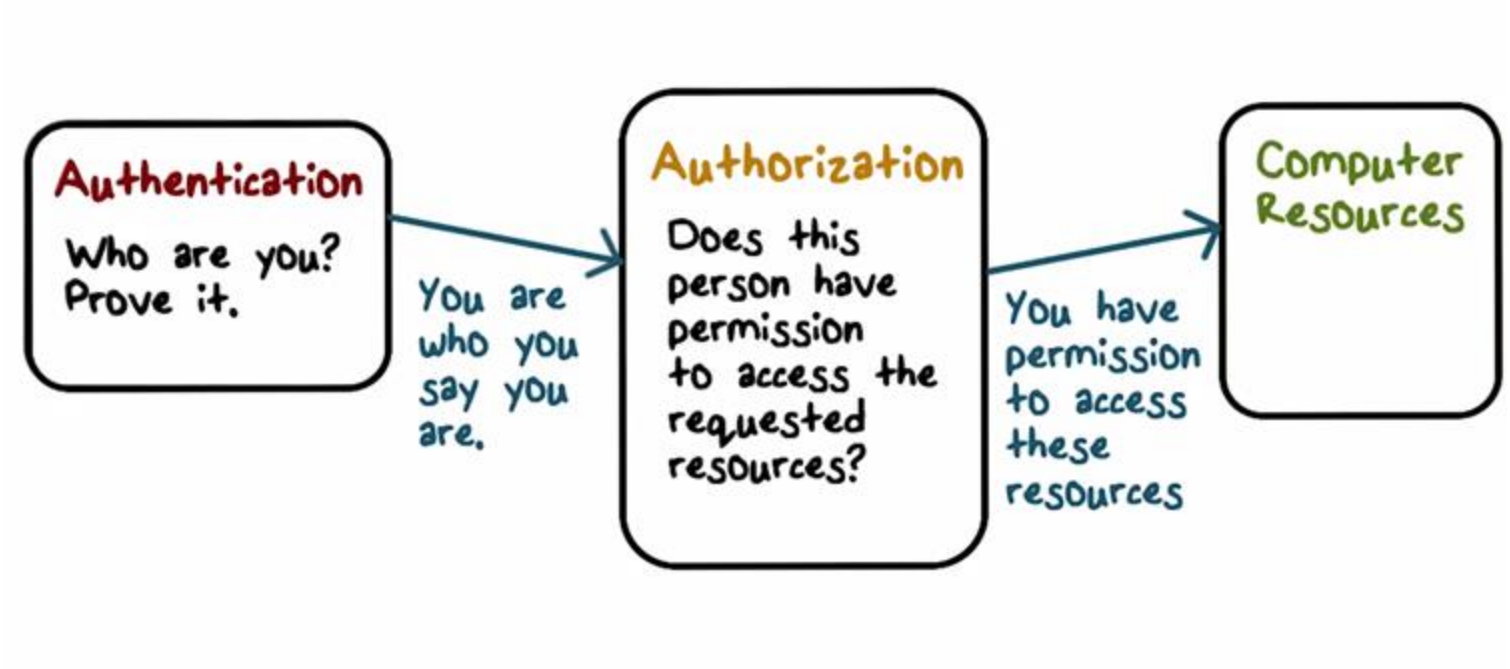
- **Role-based access control (RBAC):** based on the roles that users have within the system and on rules specifying the accesses are allowed to users in given roles.
- Widely used commercially in larger organizations.





Colorado State University

Authentication



Georgia Tech

Colorado State University

Authentication

- OS (Trusted Computing Base) needs to know who makes a request for a protected resource
- A process that makes the request does it on behalf of a certain user
- Authentication handles the question: on whose behalf the requesting process runs?
- Involves
  - claims about an identity and
  - verification of the claimed identity
- Goals
  - No false negatives
  - No false positives (major consideration)

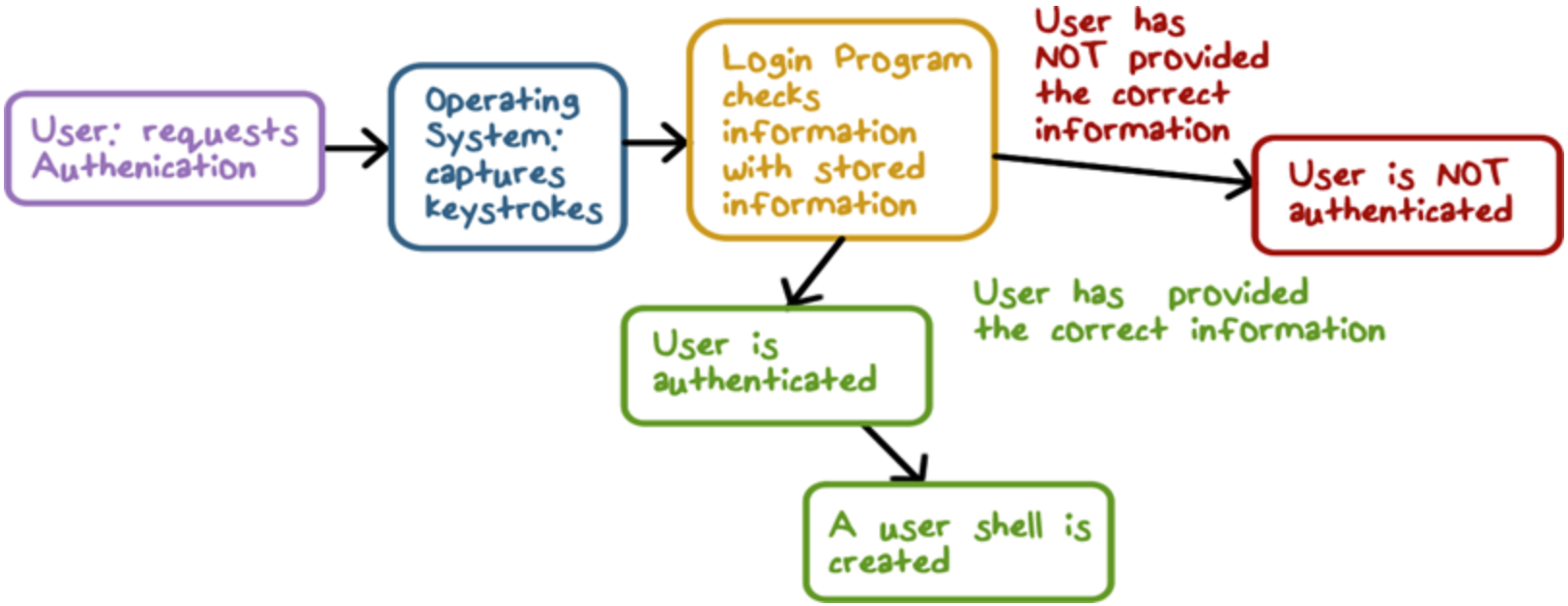
Authentication Methods

Three existing and two new.

- Something a user knows
  - Password, answers to questions
- Something a user has
  - Ex. Id card, Phone
- Something a user is
  - Biometric (face, iris, fingerprint)
- Somewhere you are geographically
- Something you do based on recognizable patterns of behavior

- Can be multifactor to reduce false positives
- After-access confirmation

Implementation: Password based Authentication



The system must provide a trusted path from keyboard to the OS.

Georgia Tech

Password authentication

Possible approaches

1. Store a list of passwords, one for each user in the system file, readable only by the root/admin account
   - Why the admin need to know the passwords?
   - If security is breached, the passwords are available to an attacker. No longer used.
2. Do not store passwords, but store something that is derived from them
   - Use a hash function and store the result
     - Dictionaries of hashed passwords exist. “Salt” is added to make cracking harder.

Security Challenges

- Password guessing
  - List of bad passwords. 123456, password, ..
- Brute force guessing
  - more later
- Good passwords are the ones harder to remember
- May be stolen using
  - keyloggers,
  - compromised websites where same password was used
  - eavesdropping *(Alice, Bob and Eve?)*
- Solution: multi-factor authentication

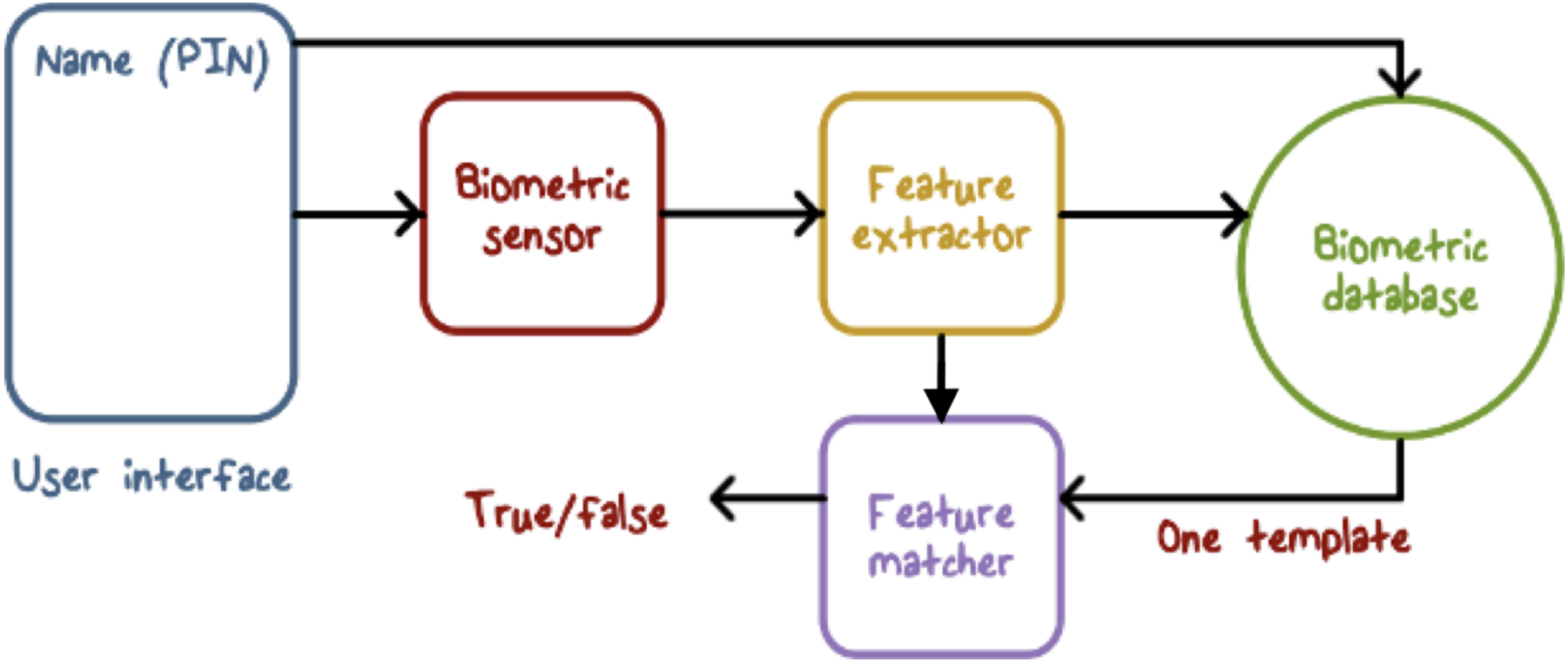
Biometric Authentication

- Fingerprints (finger swipes)
- Keystroke dynamics
- Voice
- Retina scans

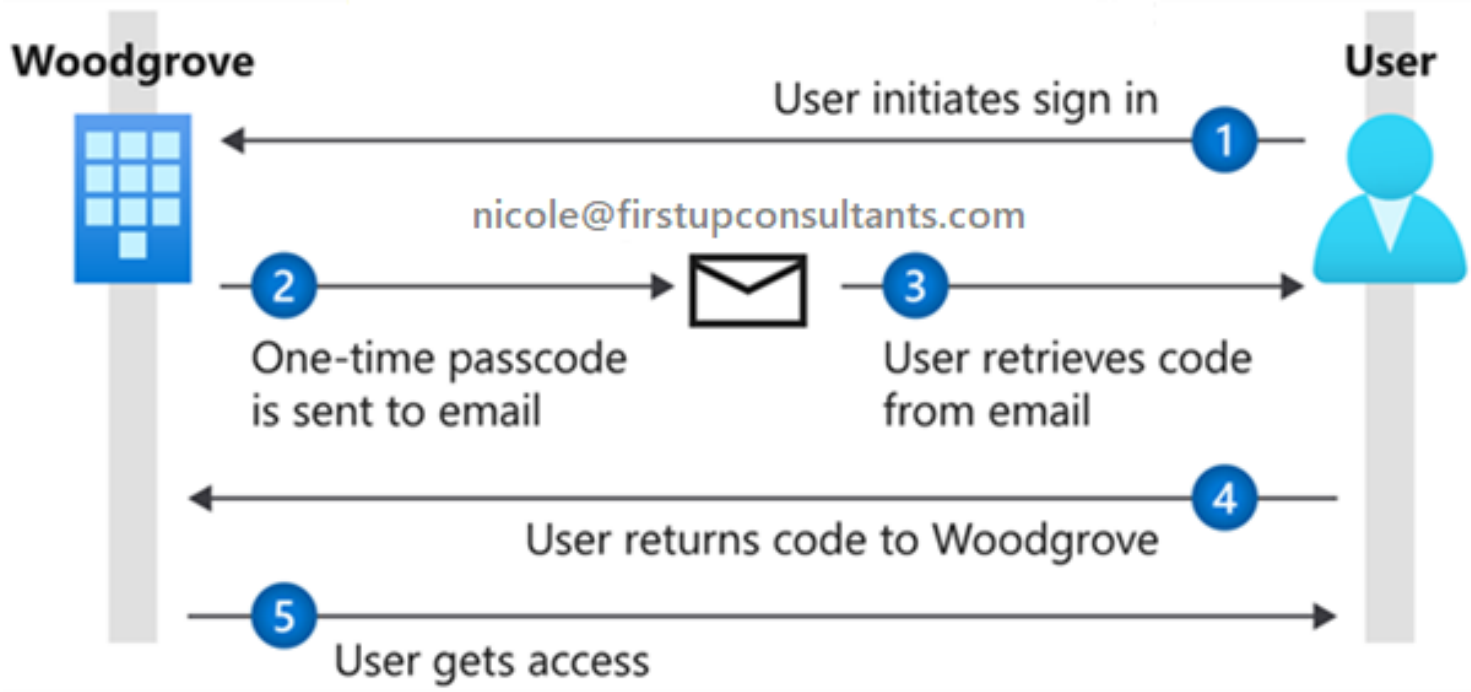
Issues

- Feature value distribution or a range
- False positives and negatives

Implementing Biometric Authentication



Smartphone OTP Authentication



OTP (one-time passcode authentication) texted or email to unique phone number/email address.

Summary

Access control

- Who can access what
- Authentication

Quantitative Security

Colorado State University

Yashwant K Malaiya

CS 370

Risk and its components

CSU Cybersecurity Center
Computer Science Dept

Risk

Topics

- Risk and its components
  - Visualization using Risk matrix
  - FAIR
- Insurance

Risk: Formal definition

Definition: The Risk due to an adverse event e_i is

$$Risk_i = Likelihood_i \times Impact_i$$

- $Likelihood_i$: Probability of the adverse event i occurring within a specific time-frame.
  - The time-frame is often chosen to be a year. Note that the probability of an adverse event happening depends on the duration of the time-frame.
  - Probability is a number between 0 and 1.
- $Impact_i$: The impact of the adverse event, measured in monetary terms.
  - Note that impact me be direct or indirect.
  - Common units are dollars (US$)#.

US$ is a common and convenient scale. Non-monetary losses, including human life, can be converted into US$, if you are a business or insurance company .

Risk: Possible Actions

How to handle risk?

Example: Credit card fraud

- Risk acceptance
  - Ex: fraud cost paid through fees charged to merchants
- Risk mitigation
  - Ex: install anti-fraud technology, adds to costs
- Risk avoidance
  - downgrade high-risk cardholders to debit or require additional verification: lost time/business
- Risk transfer
  - buy cyber-insurance to cover excess losses

Risk as a composite measure

Formal definition:

- **Risk** due to an adverse event e_i

  $Risk_i = Likelihood_i \times Impact_i$

- A specific time-frame, perhaps a year, is presumed for the likelihood.
- $Likelyhood_i$ may be replaced by $frequency_i$, when it may happen multiple times a year.
- This yields the expected value. Sometimes a worst-case evaluation is needed.

In classical risk literature, the internal component of Likelihood is termed “Vulnerability” and external “Threat”. Both are probabilities. There the term “vulnerability” does not mean a security bug, as in computer security.

Risk as a composite measure

- Likelihood can be split in two factors

  $\text{Likelihood}_i = P\{\text{A security hole}_i \text{ is exploited}\}.$

  $= P\{\text{hole}_i \text{ present}\}.$

  $P\{\text{exploitation} \mid \text{hole}_i \text{ present}\}$

- $P\{\text{hole}_i \text{ present}\}$: an internal attribute of the system.
- $P\{\text{exploitation} \mid \text{hole}_i \text{ present}\}$: depends on circumstances outside the system, including the adversary capabilities and motivation.
- In the literature, the terminology can be inconsistent.

Caution: In classical risk literature, the internal component of Likelihood is termed "Vulnerability" and external "Threat". Both are probabilities. There the term "vulnerability" does not mean a security bug, as in computer security.

Annual Loss Expectancy (ALE)

Note the terminology is from the Risk literature.

- Single loss expectancy (SLE)

  SLE = AV x EF,

  – AV is the value of the asset. EF is exposure factor which describes the loss that will happen to the asset as a result of the threat, expressed as fractional (or %) value.

- Annual loss expectancy (ALE)

  ALE = SLE x ARO

  – Where ARO is Annualized rate of occurrence.

- Example: Asset value is $100,000, exposure factor is 30%, and ARO is 0.5/year (once every two years). Thus
  – ALE = ($100,000 x 0.30) x 0.5 = $15,000.
- Note that ALE is essentially what we term as “risk”, with an annual time frame.

Annual value of the countermeasure

Cost/benefit analysis of countermeasures

- A countermeasure reduces the ALE by reducing one of its factors.

COUNTERMEASURE_VALUE

= (ALE_PREVIOUS – ALE_NOW) – COUNTERMEASURE_COST

Where ALE_PREVIOUS: ALE before implementing the countermeasure.

ALE_NOW: ALE after implementing the countermeasure

COUTERMEASURE_COST: *annualized* cost of countermeasure

- The COUNTERMEASURE_VALUE should be positive.