SWE3025: Computer Security
Lecture 0x04: Access Control II

Hojoon Lee

Systems Security Lab @ SKKU

Answering Your Voice

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Answering Your Voice

이지환(2014**11)**

목요일

안녕하세요 교수님! 강의에서 unforgeable 이라는 표현을 쓰셨는데, (forge와) delegation과 비교하여 어떤 차이가 있는지 잘 모르겠습니다. 둘 사이에 어떤 명확한 차이가 있나요? (구체적인 예시를 들어서 설명해 주시면 이해가 잘될 것 같습니다!)

댓글 작성...

우병수(2018**34)**

목요일

forge는 위조를 하는 것이고, delegate는 위임을 하는 것이니까 전자는 권한이 없는데 권한이 있는 것처럼 속이는 것이고 후자는 권한을 위임을 받은 것이니 합법적인 것이 아닐까요? 아이가 과자를 사 먹고 싶을 때 위조지폐를 만들어서 과자를 사 먹느냐, 아니면 부모님께 용돈을 받아서 사 먹느냐의 차이인 것 같습니다.

댓글 작성...

Answering Your Voice

석은주(2018****50)

목요일

Confused Deputy를 설명하시면서

There has been a separation of authority from the purpose for which it is used

라는 문장이 나오는데 이 문장이 잘 이해가 안됩니다. 앨리스가 컴파일러를 시켜 빌에게 접근하는 상황에서 목적은 빌에게 접근하는 건데 실제 권한은 앨리스가 가지지 않아서 문제가 되는 상황이라는 건가요?

댓글 작성...

작성자 이름

Answering Your Voice

김현우(2016****27)

목요일

안녕하세요 교수님

Confused deputy 부분 예시에서 Alice의 요청을 compiler가 혼동한다고 하셨는데. 그 작업을 수행하는지 궁금합니다

수행한다면 보안 issue가 생긴것으로 볼 것 같은데, 이것을 막는 방법이 있는지 궁금합니다.

만약 수행하지 않는다면 capabilities는 key를 다른 process에 주면 실행 할 수 있다고 하셨는데, ACL에서 그 작업을 수행 할 수 있는 방법은 없는건가요??

글이 이상해서 이해 안 될수 도 있으실 것 같습니다..ㅠㅠ

댓글 작성...

Answering Your Voice

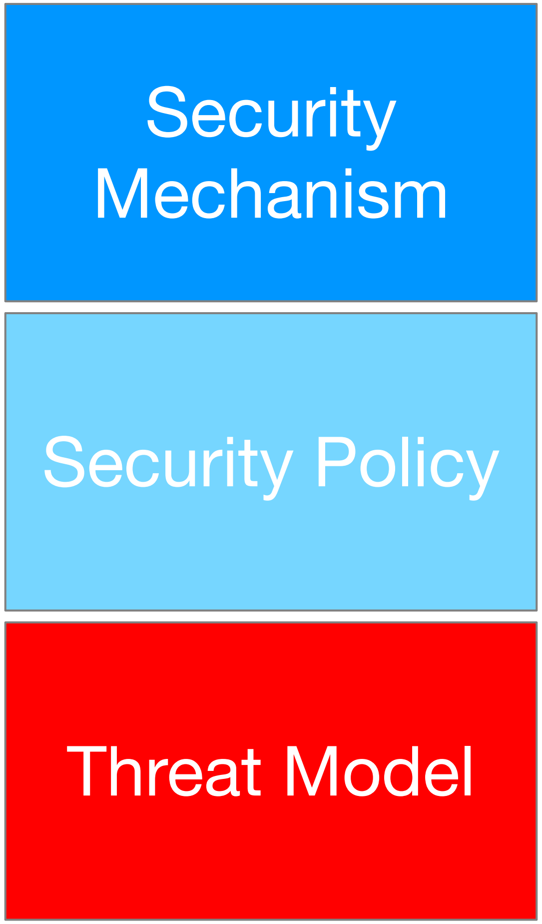
- For those of you who newly joined the course through extended add/drop period,
- Welcome to Computer Security!
- Please catch up as soon as possible because..
- There will be a assignment this week (which we will discuss at the end of this lecture)

Table of Contents

- Security Policy Model Definition
- Multilevel Security
  - Historical Models
    - Confidentiality MLS: Bell-Lapadula Properties (BLP)
    - Integrity MLS: BIBA Model
- Group-Based Access Control
- Role-Based Access Control
- Type Enforcement
- DAC vs MAC
- Case Study
  - SELinux
  - Modern Computer Architecture/OS
- Unix SETUID and Confused Deputy Problem

Table of Contents

What is a Security Policy?



Systems Security Lab

Security is Engineering

- Security is more engineering than science
- Resource for implementing security mechanisms is always limited
- Overhead induced from security must not exceed benefits from security
- A well-defined threat model allows us to build efficient security mechanisms

Threat Modeling Process

- What are the most valuable assets within the system?
  - what are the assets that cause biggest loss when leaked for stolen?
  - what would be of primary interest for our adversaries?
- What are the attack vectors?
  - which entry point would our adversaries use to enter our system?
  - which part interact with the most with the outside world?
- How powerful are our adversaries?
  - which capabilities do they have on our system?
  - what can they do and what they can not do?

Example icampus

- What are the most valuable assets on the website?
  - course grades
  - professor/TA accounts
- What are the attack vectors?
  - attack would probably start from a student account
  - database interact intensively with users
    - SQL injection attacks?
  - Discussion boards are more danagerous than other components
    - Users can upload arbitrary data
    - Complex and have huge attack surface

Example icampus (Cont'd)

- How powerful are our adversaries?
  - they are probably student themselves or have access to one of the student accounts
  - can post things on discussion boards
  - cannot view class roster (?), cannot view classmate grades
  - students with prior experience with security, hacking.
  - years of penetration testing experience? -> probably not and let's hope not

Example White House

- What are the most valuable assets within the system?
  - Classified files on UFOs and alien bodies??
  - Military strategies
- What are the attack vectors?
  - Social engineering attacks emails
  - Insider spy
  - Website or open/closed services
- How powerful are our adversaries?
  - Foreign government's elite team
  - May have been scanning the system for years
  - May be getting information from top intelligence agencies

Security Policy

- A security policy is a succinct statement of protection goals

Example icampus

- Student must not be able to edit grades
- TAs must not be able to make alter course grading
- Professors must not be able to ???

Security Policy Model

- A security policy model is a model that represents a particular policy or set of policies for access control
- General methodology that can be used as <u>templates</u> for designing (policy) and implementing (mechanism) access control

Example icampus

- Each user on icampus has his/her role
  - Student
  - TA
  - Professor
- We can adapt Role-based Access Control (which we will cover in this lecture)
- It needs to be a Mandatory Access Control

Security Mechanism

- Security mechanism is an implementation of security policies
- Can be software, hardware or both

Table of Contents

Multilevel Security

- Given Subjects (e.g., users) and Objects (e.g., information, resource) within System
- Classify subjects and objects into different clearance levels and classifications
- The questions is how do make these classifications? and on what conditions do we allow/disallow access?

Bell-LaPadula Properties (BLP)

- Developed by miltary/government and used for military/government
- All objects within system are either *object(O)* or *subject(S)*
  - S has *clearance level*
  - O has *classification level*
- US Department of Defense (DoD) uses 4 levels:
  - Top Secret
  - Secret
  - Confidential
  - Unclassified

Bell-LaPadula Properties (BLP)

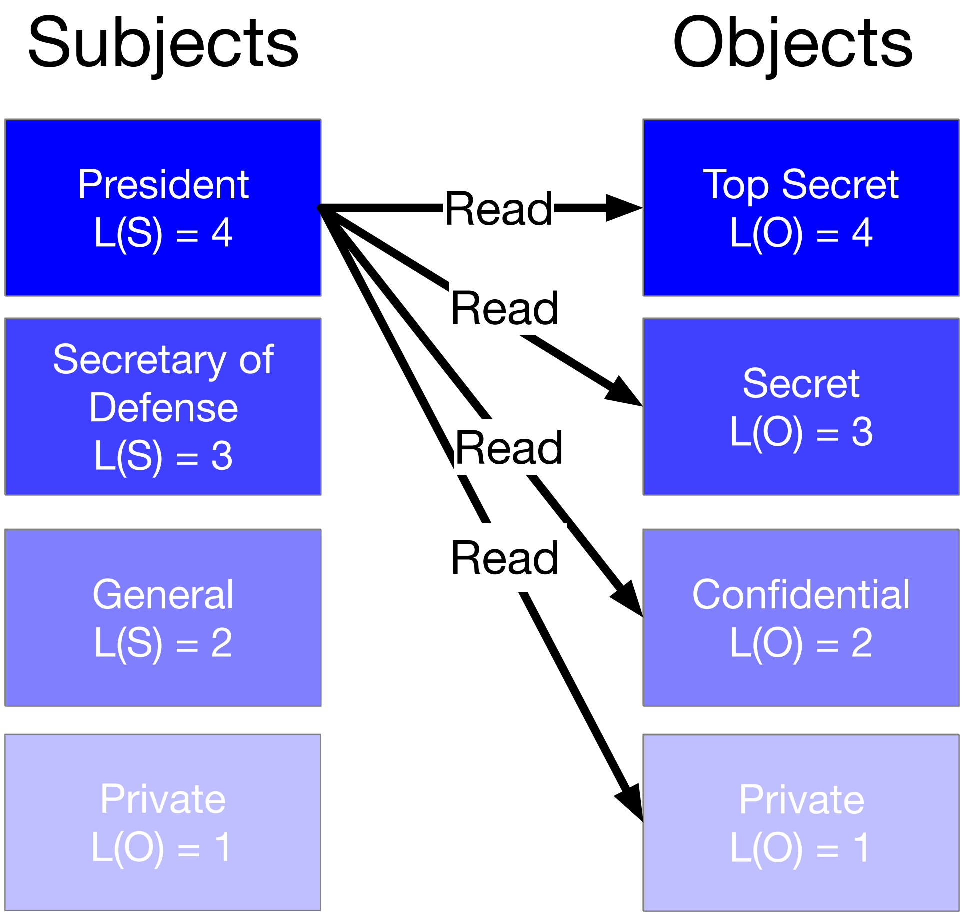
- Controls information flow for confidentiality
- Security level denoted as L(O) / L(S)
- Simple security conditions
  - No Read UP Policy:

    S can **read** **O** if and only if $L(S) \geq L(O)$

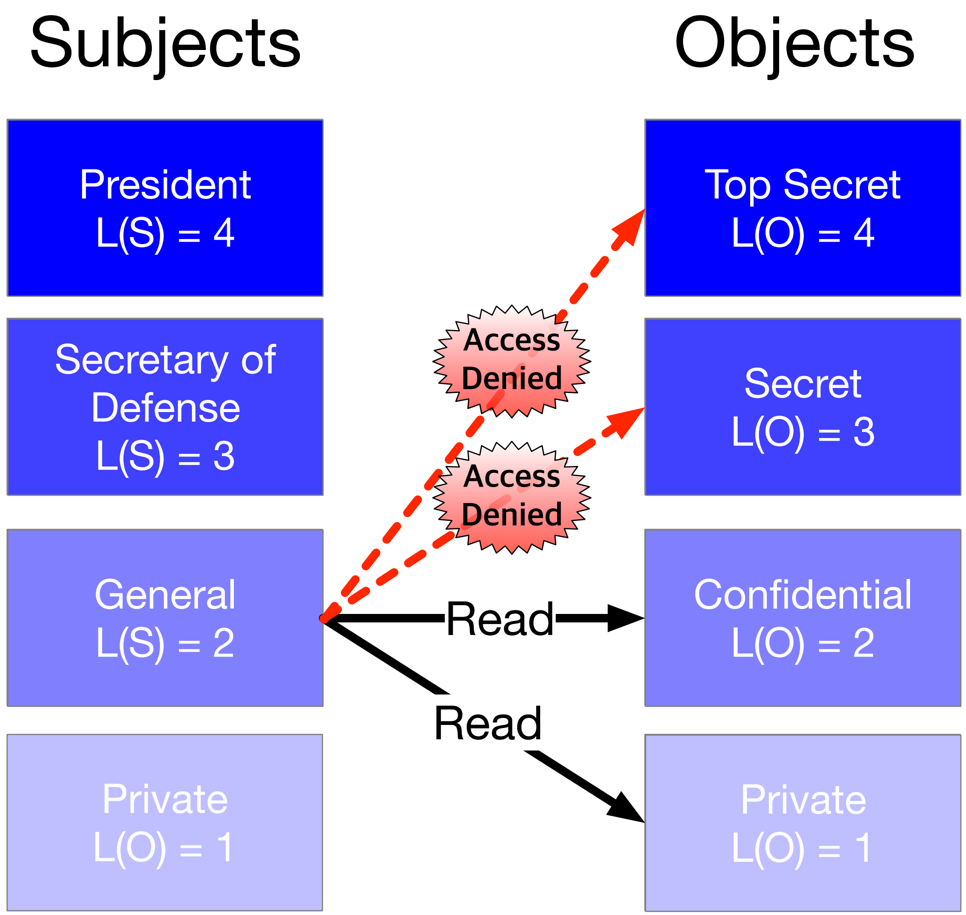
  - No Write DOWN Policy:

    S can **write** **O** if and only if $L(S) \leq L(O)$

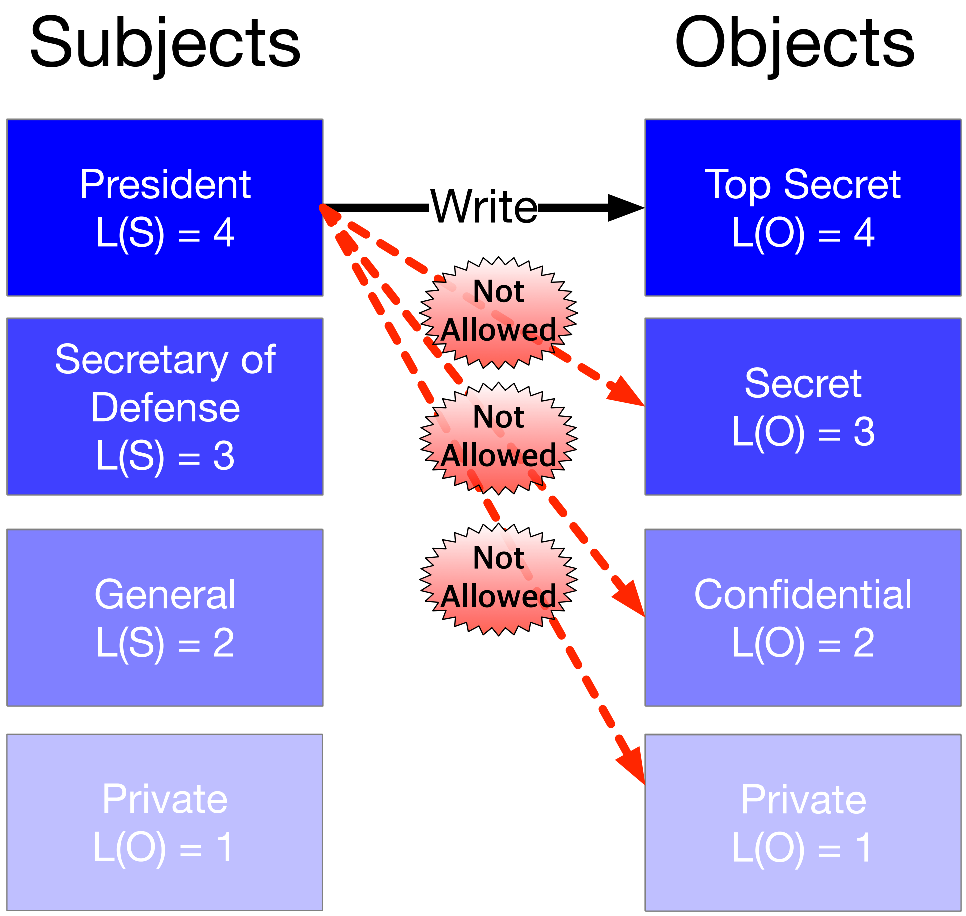
BLP: No Read Up



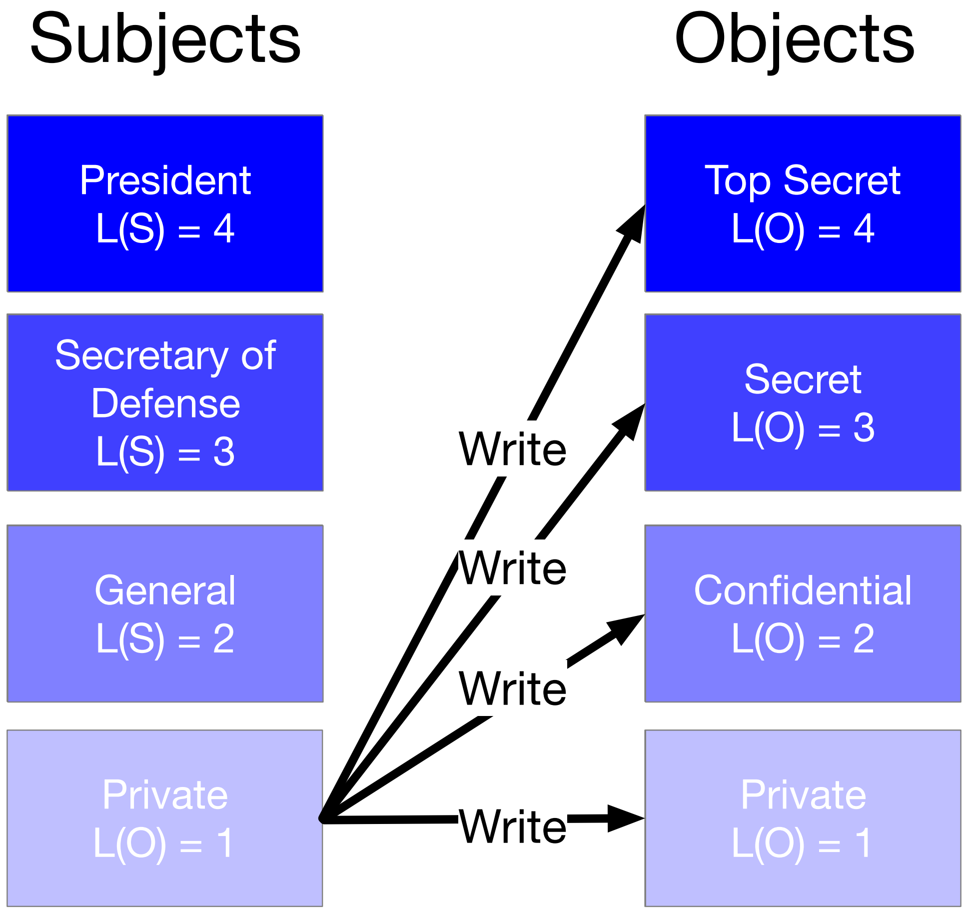
BLP: No Read Up



BLP: No Write Down



BLP: No Write Down



(Not really) Real-Life Example (From my imagination)

No Read UP

- Private Ryan (L(S)=1) obviously does not know more than he needs to about military strategy
- e.g., He does not know where the nuclear missile switch is located (L(O)=4)

(Not really) Real-Life Example (From my imagination)

No Write Down

- Private Ryan finds an alien body
- He writes report and request it to be Top Secret (L(O) = 4)
- Afterwards, only president (L(S)=4) can read Top Secret
- But president should not write press release (L(O) = 1) because he might accidently leak information (because he knows)

Biba's Model

- BLP for confidentiality, Biba for Integrity (Multi-level Integrity)
  - Prevent data modification by unauthorized subjects
  - Prevent unauthorized data modification by authorized parties
  - Maintain internal and external consistency
- Direct inverse of Bell-LaPadula Model (read down, write up)
- Integrity model
  - No Read Down Policy:
  - No Write up Policy

Example of BIBA Model

- No Write UP: If Private Ryan can modify Top Secrets such as military action plan, maybe he can start a nuclear war?
- No Read Down: The General should not depend on the lower classification documents when planning a military action
- BIBA is all about <u>Integrity</u> of information

BIBA Use Case?

- BIBA is not very common in today's access control systems
- One notable example is

Table of Contents

ACLs and Capabilities

- We covered this in our last lecture
- ACL is perhaps the most commonly used security policy (e.g., *nix OS filesystem)
- ACL may be vulnerable to confused deputy problem
- Capability to rescue for CD problem

Group-Based Access Control (GBAC)

- Allow access to Objects to Users (U_1, U_2,...U_n) who belong in Group (G)
- Unix/Linux filesystem access control implements ACL based on (User Identity + Group)

Example: SSLab Server

```
crw-rw---- 1 root kvm 10, 232 Mar 27 22:29 /dev/kvm
```

- KVM is Linux Kernel's virtualization plugin
- Access to KVM is exposed through a virtual file at `"/dev/kvm"`
- Use of Virtualization is allowed/disallowed based on the file permission of `"/dev/kvm"`

Example: SSLab Server (Cont'd)

```
crw-rw---- 1 root kvm 10, 232 Mar 27 22:29 /dev/kvm
```

Group ACL   Owner Group

- Users who belong in group "kvm" can "rw-" to `"/dev/kvm"`
- Only I and your TA can create/delete virtual machines on the server

ME   ME   Your TA

```
kvm:x:108:hjlee,sslab-admin,khadinh
```

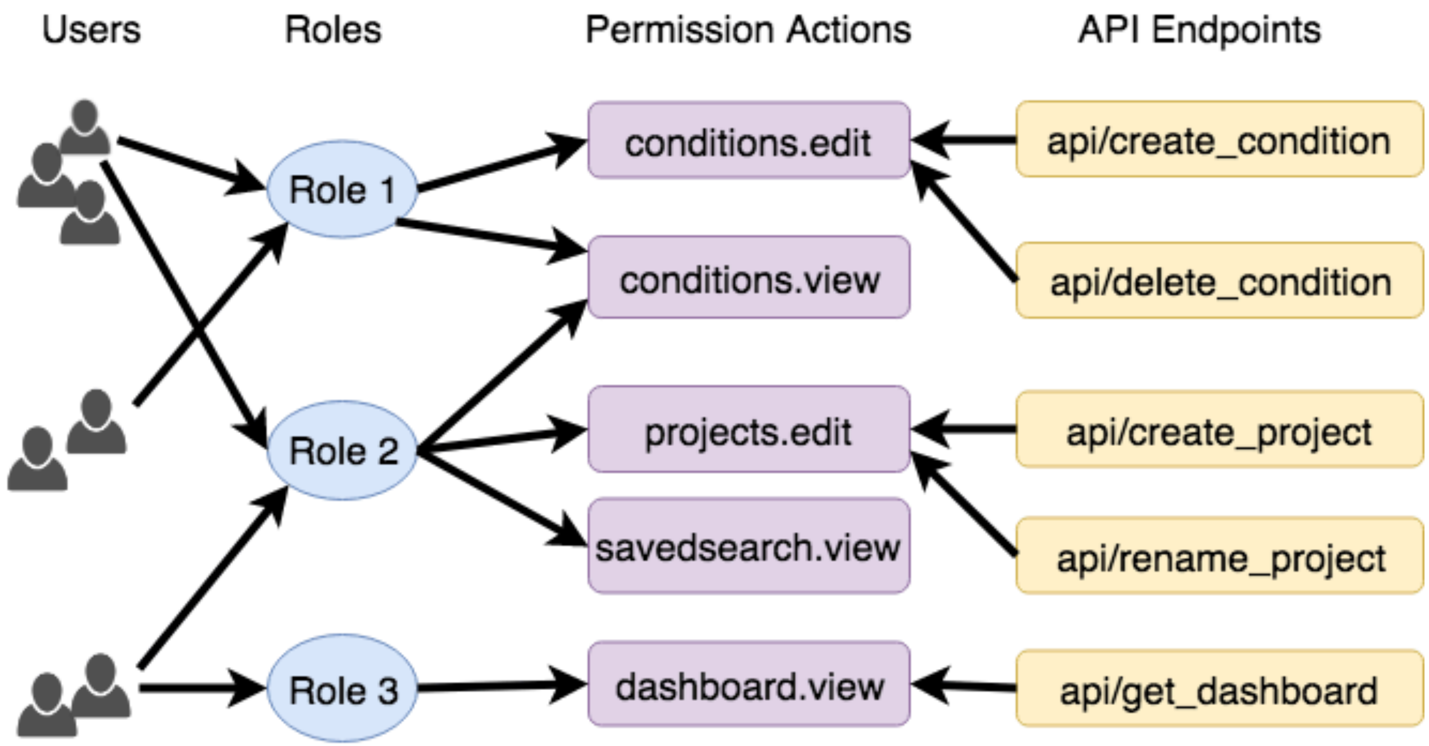
Unix/Linux filesystem

- So now we know that Unix/Linux filesystem adapts
  - ACL Access Control
  - Group-Based Access Control

Role-based Access Control (RBAC)

- Access control mechanism that evolves around the current action the user is requesting
- Very similar to GBAC but much more fine-grained and focuses more on actions than user identity
- Advantages of RBAC
  - Least privilege – allow a user to sign on with least privilege required for a specific task
  - Separation of duties – no single user should be given enough privileges
  - Object classes – objects can be grouped based on classifications

Role-based Access Control (RBAC)



RBAC vs GBAC

- Group is collection of Users
- Role is collection of Responsibilities

Type Enforcement

- Classifies Subjects and Objects into different types
- These types can be used in implementing more <u>fine-grained</u> access control rules (unlike coarse-grained ACLs)

Security Policy Models

- There many other Security Models
  - Brewer-nash model
  - Clark-Wilson Integrity Model
  - Lattice-based access control
  - etc…
- and which one is the best?
  - None
  - However, these are foundational models that influenced many access control mechanisms we use today

Table of Contents

DAC vs MAC

Security Policies can be MAC/DAC/Hybrid

- Discretionary Access Control
  - Relies on Object Owner to make access control decisions
  - Example
    - Unix/Linux file system
    - `chmod 777 my-file`
- Mandatory Access Control
  - Access control decisions are made by a central administrative entity
  - Bell-LaPadula can only be implemented in form of MAC

Security Policies are Confusing..

- It's because they are not exactly complementary to each other
- Two Security Policies may be in conflict
  - e.g., BLP vs. BIBA
- Or tries to achieve different goals
  - e.g., BLP vs. BIBA
- And has Pros and Cons depending on the system and goals
- Some of them are somewhat outdated
- Then why do we need to learn them?

Modern Implementation of Security Policies

- Modern implementation of Security Policies are heavily influenced by the security policy models
- They also often adapt security policy models for their specific needs
- They also mix different policy models

Table of Contents

SELinux

- Security Enhanced Linux
- Developed by the NSA and open sourced in 2000
- Adopts Role-Based Access Control and Type Enforcement on top of Linux ACLs
- Provides MAC to the Linux kernel
  - root can no longer do whatver she/he pleases
  - SELinux rules have higher priority
  - (but root can modify the rules or disable SELinux)

SELinux

- Allows access control rules to be written in terms of the following fields:
  - User
  - Role
  - Type
  - Level (optional field)

```
user:role:type:level(optional)
```

SELinux: Origins of Foundational Concepts

- Where did these concepts come from?
  - User (ACLs and DAC)
  - Role (RBAC)
  - Type (Type Enforcement)
  - Level (Multilevel Security – BLP, BIBA)

```
user:role:type:level(optional)
```

SELinux Examples

- Type Declaration

```
# Type Enforcement File *.te
type mytype_t; # Process Type (Domain)
type mytype_exec_t; # File Type
```

- Change file type

```
$ chcon -t mytype_t file1
```

- Policy Rule Statement

```
# under /etc/selinux/
$ {COMMAND} {SOURCETYPE} {TARGETTYPE}:{CLASS} {PERMS}
```

```
# Possible Commands
allow, dontaudit, audit2allow, neverallow

# Type Examples
etc_t,… whatever you define

# Class Example
file,dir,sock_file,tcp_socket,process …

# PERMS
read, open, write
```

SELINUX: Goals

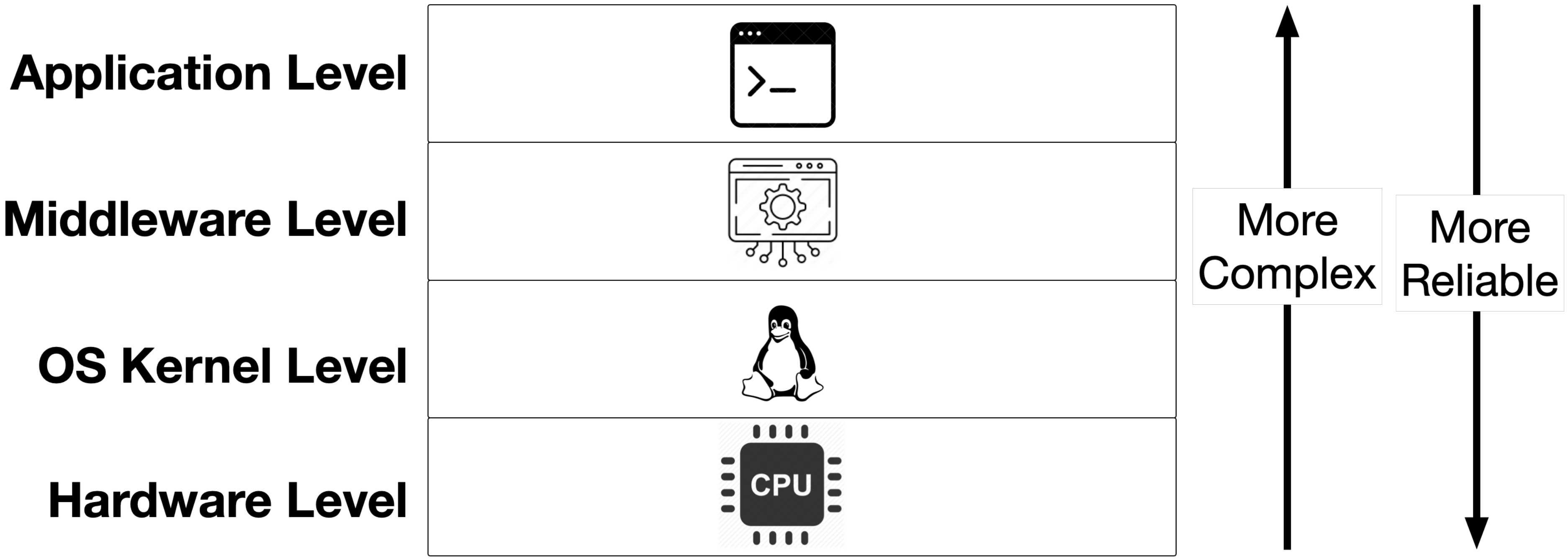
- The Principle of Least Privilege
- Expressive access control rules for diverse user applications
  - You can implement your own access control rules for your application
  - And also how different applications interact with each other

Access Control Implementation: From Bottom to Top

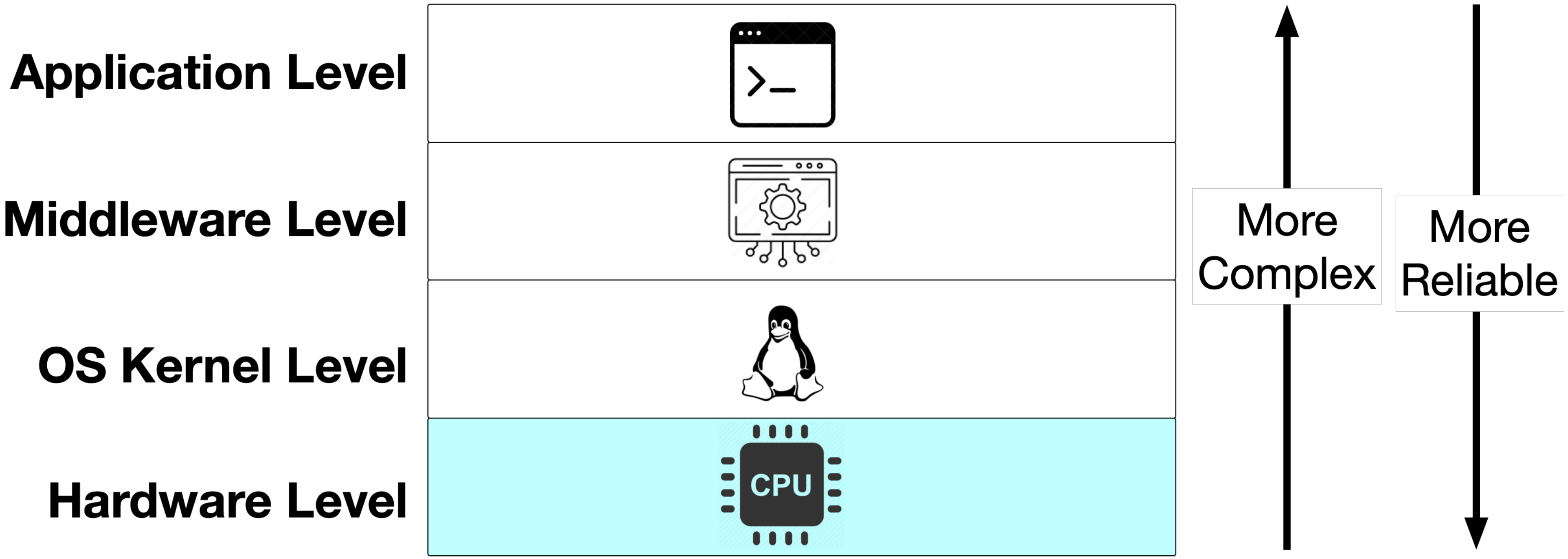
Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

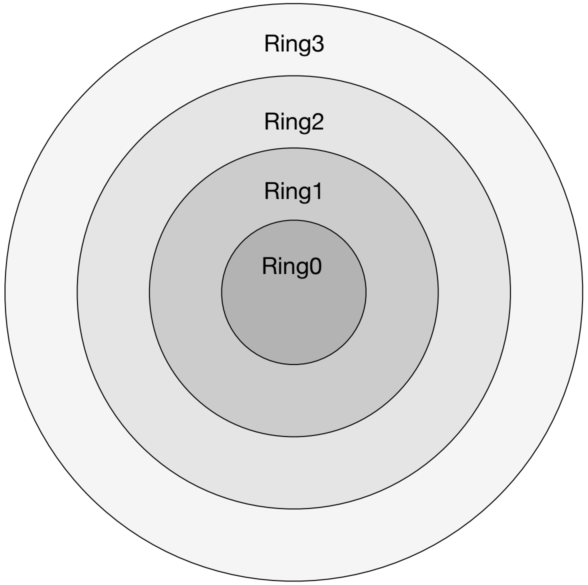
Access Control in Different Levels



Hardware Level Access Control



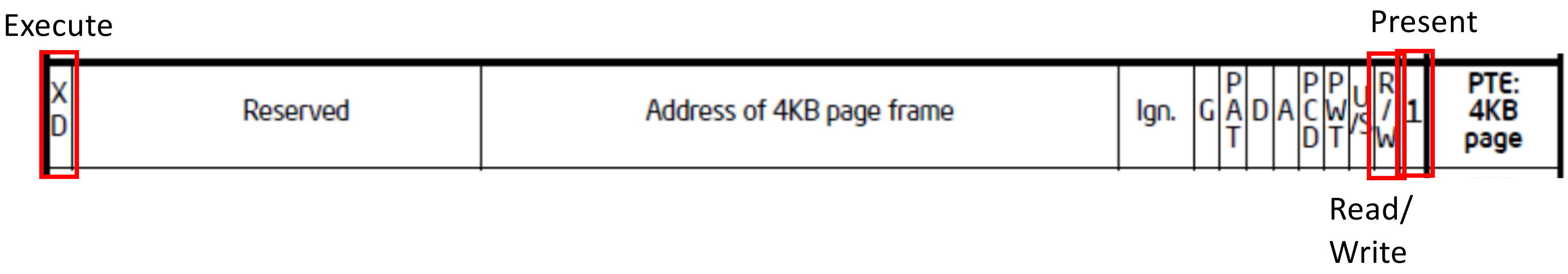
x86 CPU Execution Privilege



	Ring0(Kernel)	Ring1	Ring2	Ring3(User)
Privileged Instruction	O	X	X	X
Supervisor Page Access	O	O	O	X

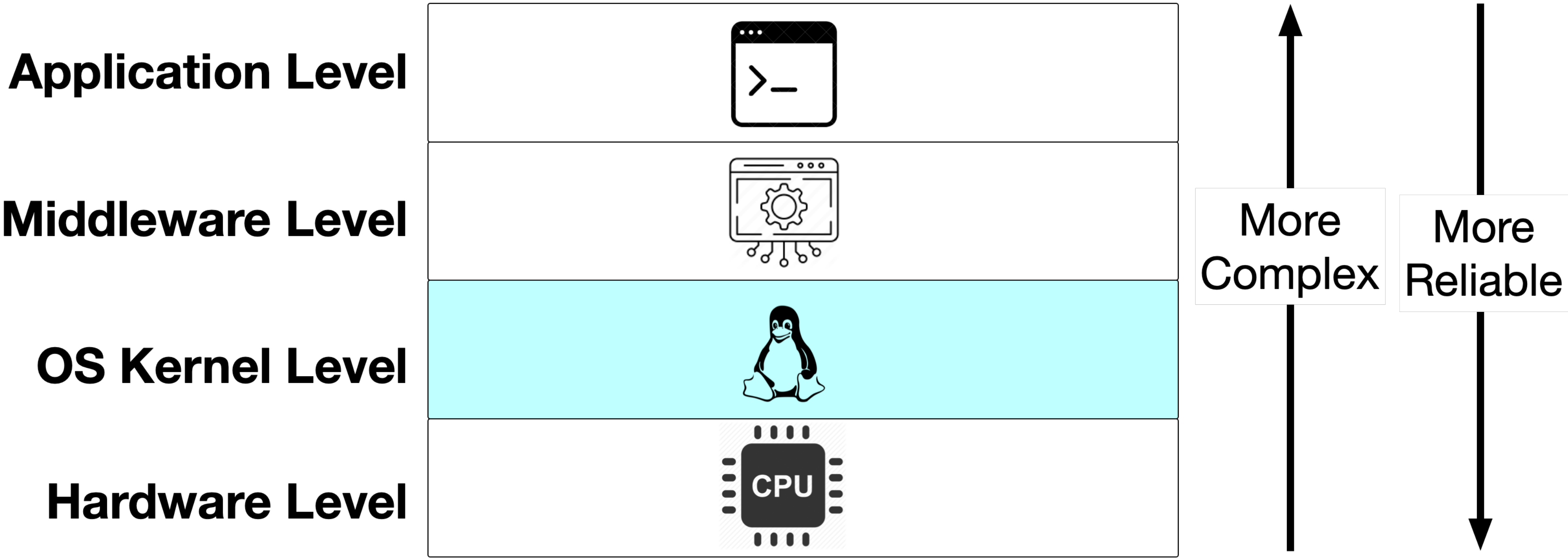
- **Privileged instructions:** change hardware configuration
  - e.g., disable/enable memory protection, load new page tables, etc..
  - only Ring0 can execute privileged instructions
- **Supervisor Pages:** Memory pages can be Supervisor/User
  - only Ring0~Ring2 can access Supervisor pages

x86 CPU Execution Privilege

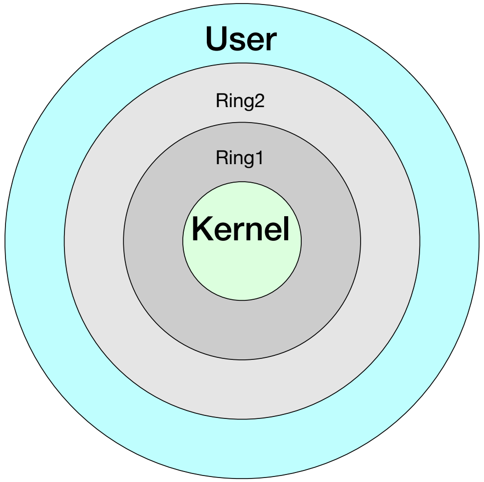


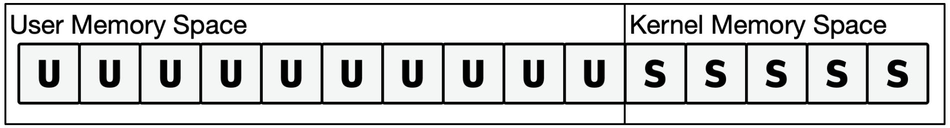
- Paging system creates *virtual memory* on top of *physical memory* and apply access control
- *Page Table Entry* has flags that represent permission associated with page
  - P bit: if set, page can be accessed
  - R/W bit: if set, page can be modified
  - XD bit: if set, page can be executed as code

OS Kernel Level Access Control



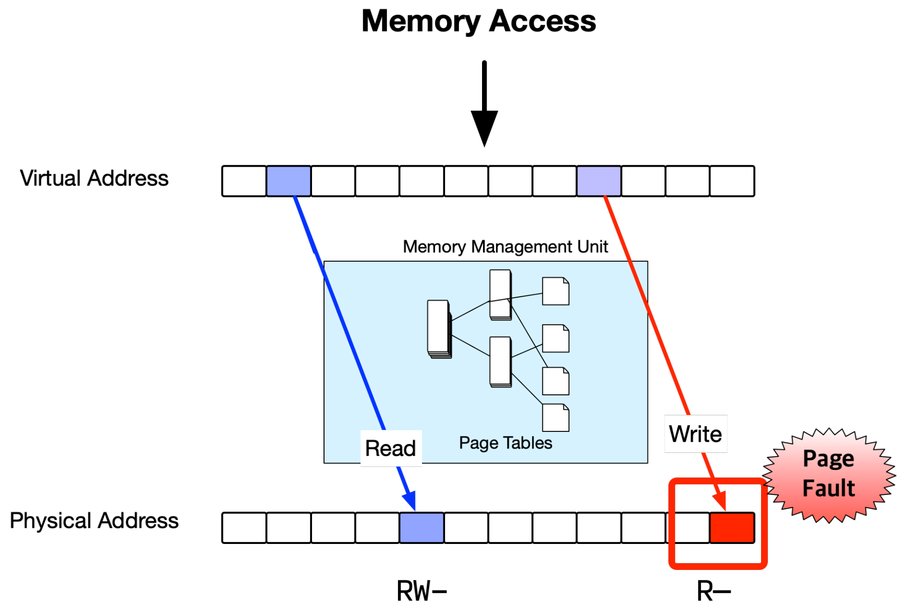
Memory Access Control





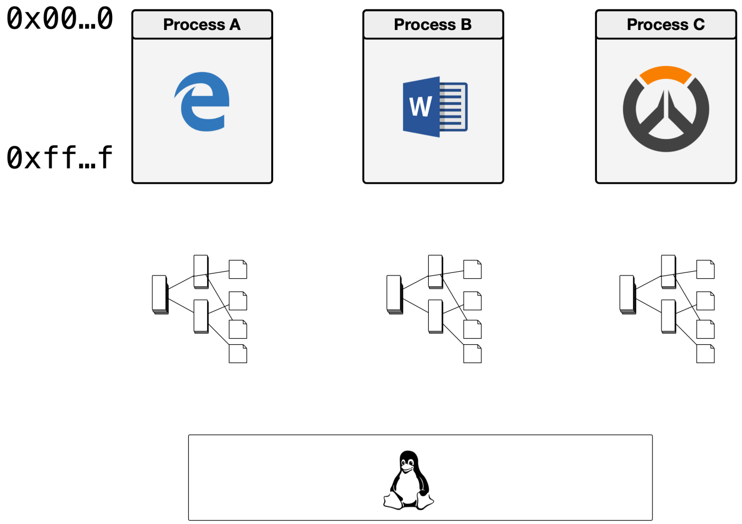
- Kernel(runs in Ring0 mode) maps itself as Supervisor pages to protect itself from user processes
- This configuration cannot be arbitrarily changed since user(Ring3) can not execute <u>privileged instructions</u>

Memory Access Control: Kernel vs. User Separation



- Kernel maintains page tables for kernel itself and user processes to control memory permissions
- e.g., Code must not be modified, read-only data should not be modified

Memory Access Control: Per-process Address Space

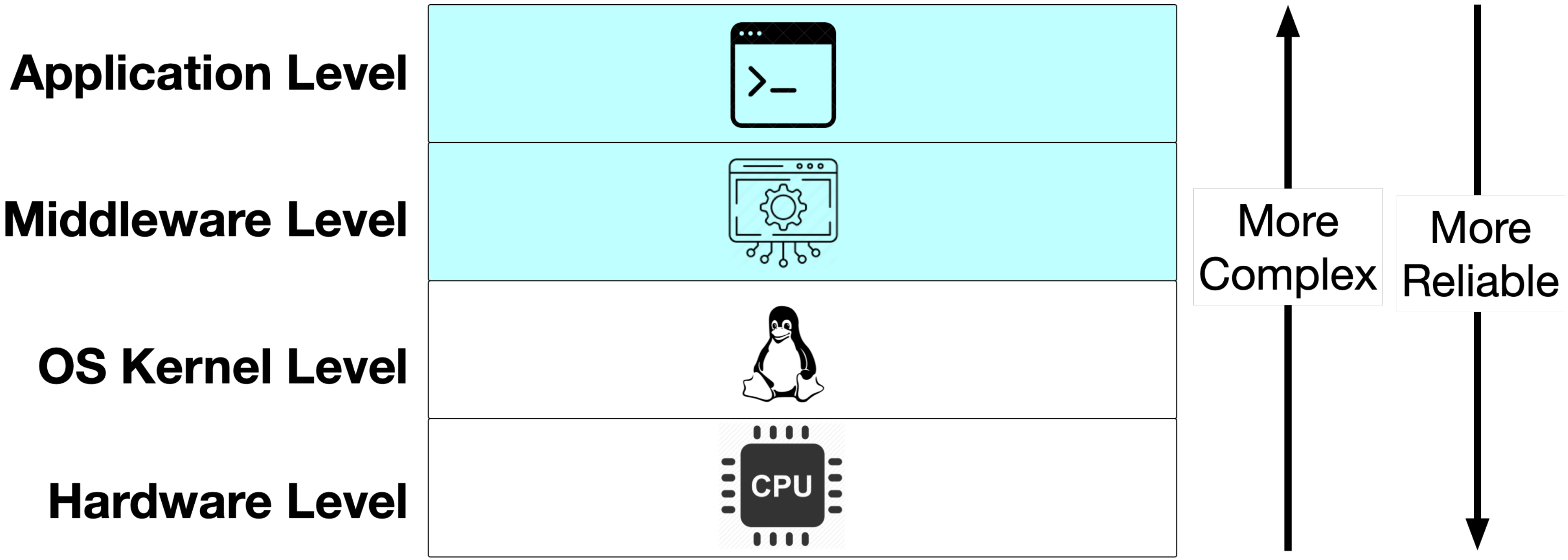


- Kernel creates *private* address space for each process
- One process cannot arbitrarily access memory space of other processes

File System Access Control

- Kernel maintains ACLs for each directory and files
- User makes system calls to ask permission
- e.g., fopen(), fprintf() all makes system calls internally to kernel

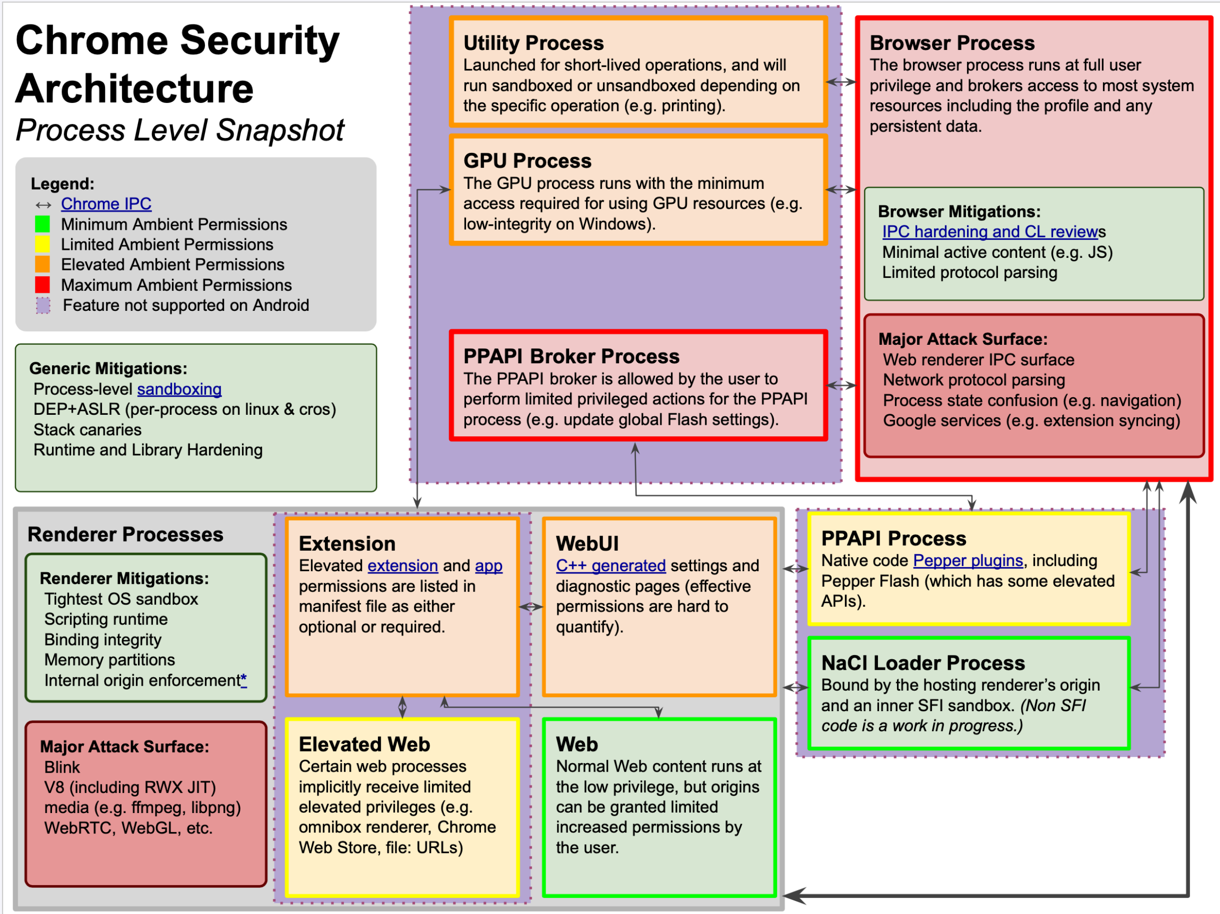
Application Level Access Control



Application Level Access Control

- Access control is application-specific and must be defined by programmer
  - What must be protected?
  - Who should be able to access?
  - What conditions make an access illegal?
  - What kind of access control scheme should we adapt?
  - Are there any loopholes in the access control scheme?

Application-Specific Access Control Schemes



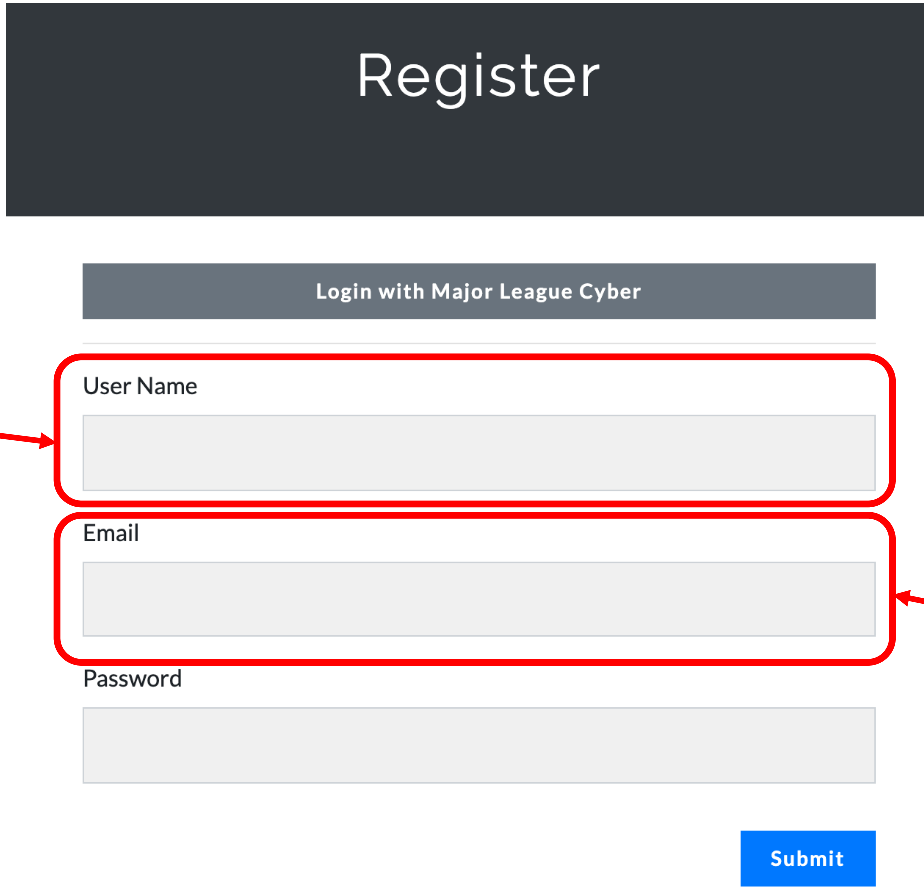
Lab1. CTF Challenge: Confused Deputy

Lab1. CTF Challenge: Confused Deputy

ctf.skku.edu

Lab1. CTF Challenge: Confused Deputy

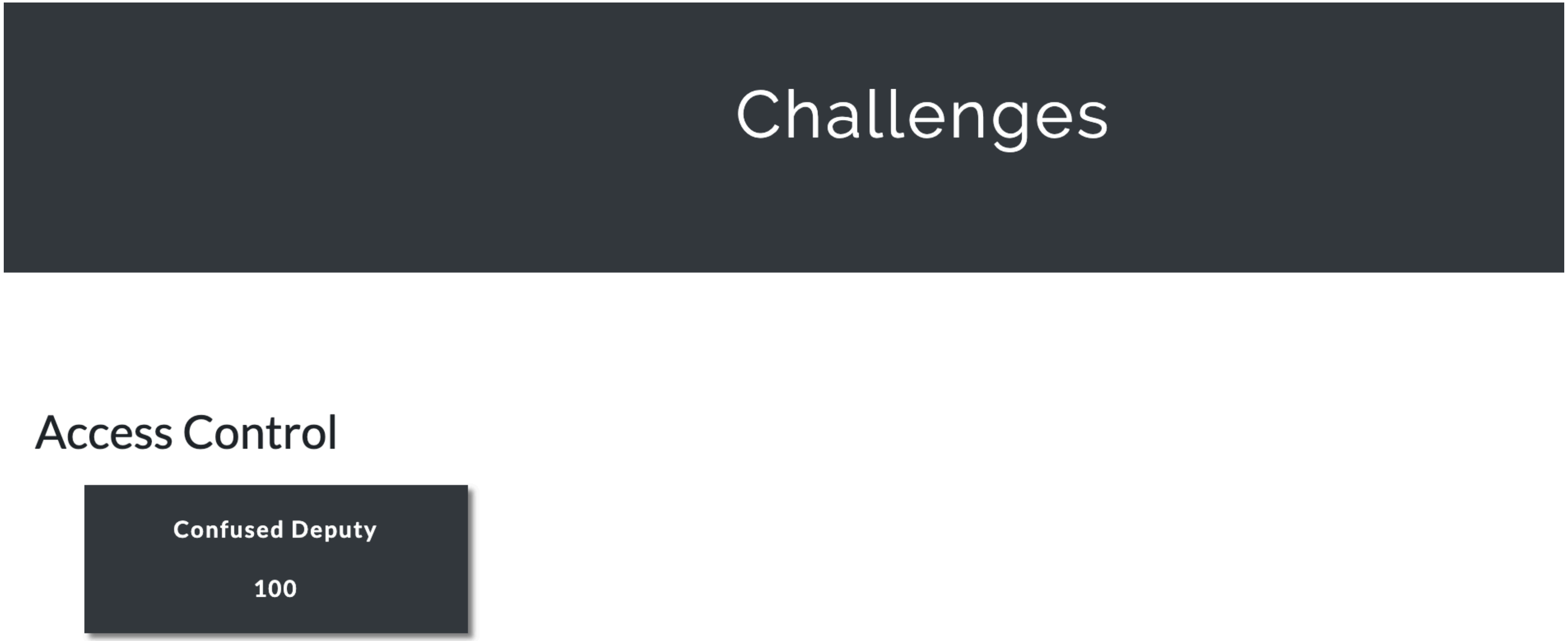
- Anything You'd like
- So that people won't be able to recognize you in the scoreboard.
- You can also reveal your identity if you want to show off



- **Must match E-mail address registered on *icampus!!!***

Systems Security Lab

Lab1. CTF Challenge: Confused Deputy



Lab1. CTF Challenge: Confused Deputy

- Your Goal:
  - Read this file (Hint: You're not flagkeeper) using Confused Deputy attack

```
-r--------  1 flagkeeper flagkeeper   24 Mar 28 12:43 flag
```

  - Enter them on `ctf.skku.edu` using your account

- Required knowledge
  - How to use Linux (basic commands, etc)
  - ACLs and Unix/Linux file system permission
  - What *setuids* are: research on the internet
  - How *Confused Deputy* attack works

Lab1. CTF Challenge: Confused Deputy

- Warning: Flags are designed to be unique for everyone
- Entering your friend's flag is a very efficient way to get a "0" on the Lab
  - If you happen to made a mistake and your mistake coincidentally matches your friend's flag (1/zillion chance), you need to prove this to me
- Your activities inside Docker Container will be recorded, so give up your privacy while being on our server
- In case there is any suspicion, we can compare your logs on the server against your lab report
- If you find a vulnerability of the server itself, report and get +10 on your lab grade
  - But don't get 0 on your lab and try to find 10 vulnerabilities

Lab1. CTF Challenge: Confused Deputy

- You will be notified by email when Lab1 is ready
- The reason for the delay
  - The challenge is being adjusted (someone said it is too easy ☺)
  - We're testing simultaneously running Docker container for 70+ people
  - Access Control
    - We need to ensure that you cannot escape the container to access the server itself
    - We need to ensure that you cannot somehow access other student's container

Thank you for attention!
and as always please post
questions/feedback on icampus!

and also feel free to discuss among
yourselves!