Operating Systems

Lecture 11a

Dr Ronald Grau School of Engineering and Informatics Spring term 2018

Previously

File systems and I/O

Today

Security

- Terminology
- Cryptography
- Authentication
- Access Control
- Vulnerabilities
- Design

What is security?

Keywords that describe aspects of security

Freedom / **Protection** (from harm, damage, threat, anxiety, …)

Resilience (against attack, or unwanted change)

Control (of access to goods / resources)

What is security?

Strategies, measures and tools to ensure security in computer systems

- Confidentiality: keep data secret
- Integrity: prevent tampering with data
- Availability: keep data accessible

What is security?

Strategies, measures and tools to ensure security in computer systems

- Confidentiality: keep data secret
- Integrity: prevent tampering with data
- Availability: keep data accessible

Security threats:

- Data leak
- Manipulation of data
- Denial-of-service attack

→ security violations

What is security?

Security policy

- Assigns roles to users
- Roles have well-defined privileges

Violations:

- Internal: abusing one's role / negligence
  → trust problem in assigning roles
- External: transgressing one's role
  → technical problem in securing the system

Where is security important in an OS?

How to secure a system?

Attacks:

- Attempt to acquire privileges
  - → Assume someone else's identity
  - → Exploit a security vulnerability
- Deliberately overload or damage a system

How to secure a system?

Defenses:

- Authentication: identify users
- Accounting: log user activities
- Access control: restrict user permissions
- Isolation: detect and lock out potentially malicious users

Asymmetric Cryptography

a.k.a. Public-key cryptography

- Two keys: public key *P* and private key *R* (secret)
- Cryptographic algorithm *f*
- Encryption: *d = f (P,m)*
- Decryption: *m = f (R,d)*
- Signing: *d = f (R,m), send (m,d)*
- Signature verification: *m = f (P,d)*
- Works because it is difficult to compute *R* given *P*, *m* and *d*

Examples: RSA, elliptic curves, . . .

Applications: PGP, GPG, SSL, Bitcoin, . . .

Symmetric Cryptography

- Shared secret key *K*
- Cryptographic algorithm *f*
- Encryption: $d = f(K,m)$
- Decryption: $m = f^{-1}(K,d)$
- Works because it is difficult to compute *m* given *d* (without knowing *K*)

Example: 3DES, AES, Blowfish, . . .

Problem: key exchange
→ use asymmetric cryptography to exchange keys, e.g. SSL

Advantage: faster than asymmetric cryptography

Cryptographic Hash Functions

One-way function h with

- Input: message m
- Output: digest d
- Pre-image resistance: Given d, it is difficult to compute $m = h^{-1}(d)$
- Second-pre-image resistance: Given m_1, find an m_2 such that $h(m_1) = h(m_2)$
- Collision resistance: Find m_1 and m_2 such that $h(m_1) = h(m_2)$

Examples: MD5, SHA-1, BLAKE, . . .

Applications in verifying data integrity, source code management systems, . . .

Authentication

User

- Identity in the system (username, . . .)

Authentication by

- Something that the user is (e.g., biometric features)
- Something that the user has (e.g., token, smartphone, key card, …)
- Something that the user knows (e.g., password, pin,…)

Authentication

Example: password

- Hashed and checked against stored hash in user database

  Linux: `/etc/shadow`, e.g. SHA-512

Example 2: Two-factor authentication (TFA)

- Password + time-based one-time password (TOTP)

Access Control

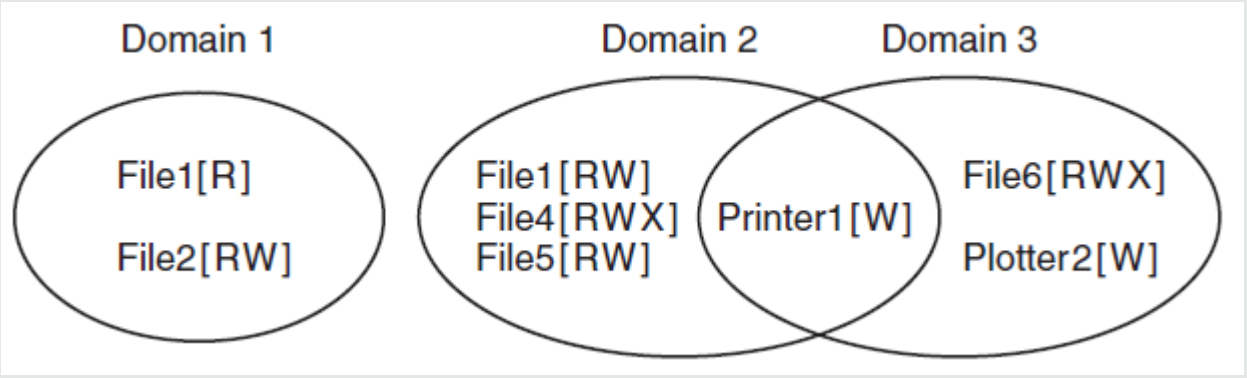
Protection domain

- Specifies the objects (resources) and access permissions
- Statically or dynamically assigned ("role")

Examples:

- User, user group, network segment, . . .
- Process, thread, procedure, . . .

→ large variety of mechanisms



Access Matrix

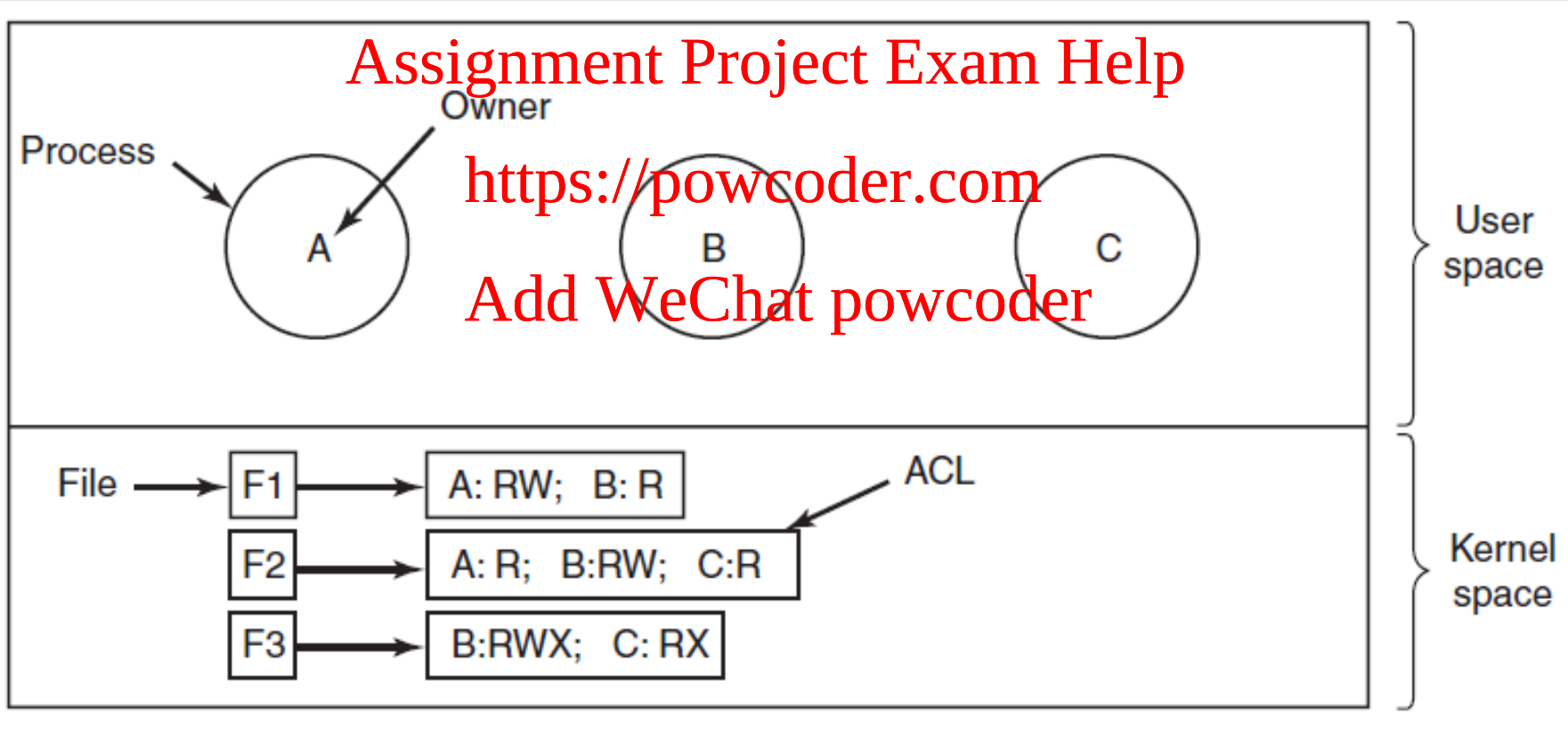
Specification of protection domains

Domain	Object: File1	File2	File3	File4	File5	File6	Printer1	Plotter2
1	Read	Read Write						
2			Read	Read Write Execute	Read Write		Write	
3						Read Write Execute	Write	Write

Domain	Object: File1	File2	File3	File4	File5	File6	Printer1	Plotter2	Domain1	Domain2	Domain3
1	Read	Read Write								Enter	
2			Read	Read Write Execute	Read Write		Write				
3						Read Write Execute	Write	Write			

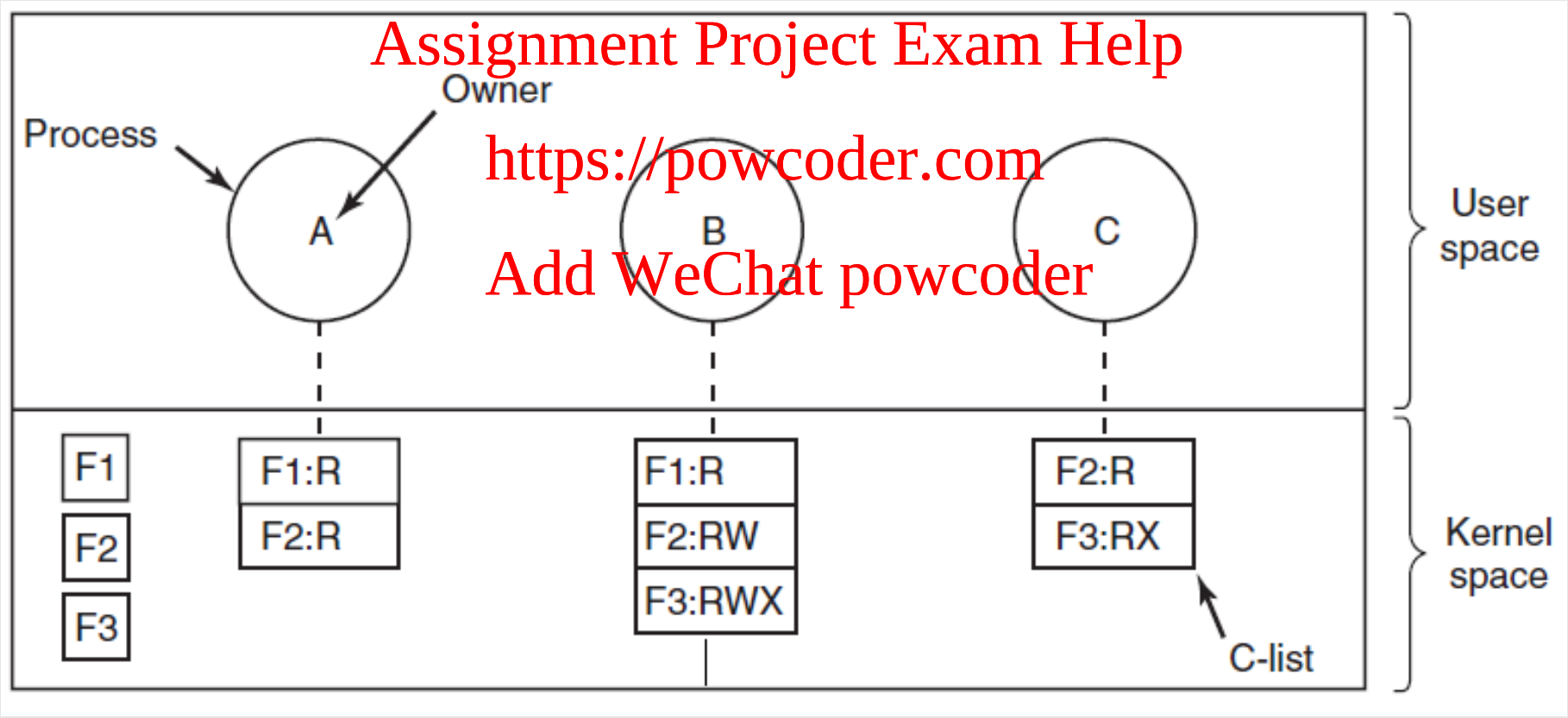
Implementation: Access Control List (ACL)

Store each user's permissions for every object



Implementation: Capability List

Store each object's user permission for every user



Mechanism vs. Policy

Mechanism

- Operating system provides way to specify rules for protection domains
- Operating system ensures that rules are enforced

Policy

- Users define policy:

  Who is allowed to access which object?

System intrusion

Exploit user's weakness

- Social engineering (phishing, . . .)
- Make user run a malicious program
- Password cracking

Exploit technical weakness (vulnerability)

- Software bugs
- Misconfigured systems
- Attack weak cryptography

Ultimate goal: get control over system

Malware

Software with malicious functionality

- Steal data (e.g. key logger)
- Manipulate data
- Unwanted encryption (ransomware)
- Launch a denial-of-service attack

Malware

Types of malware:

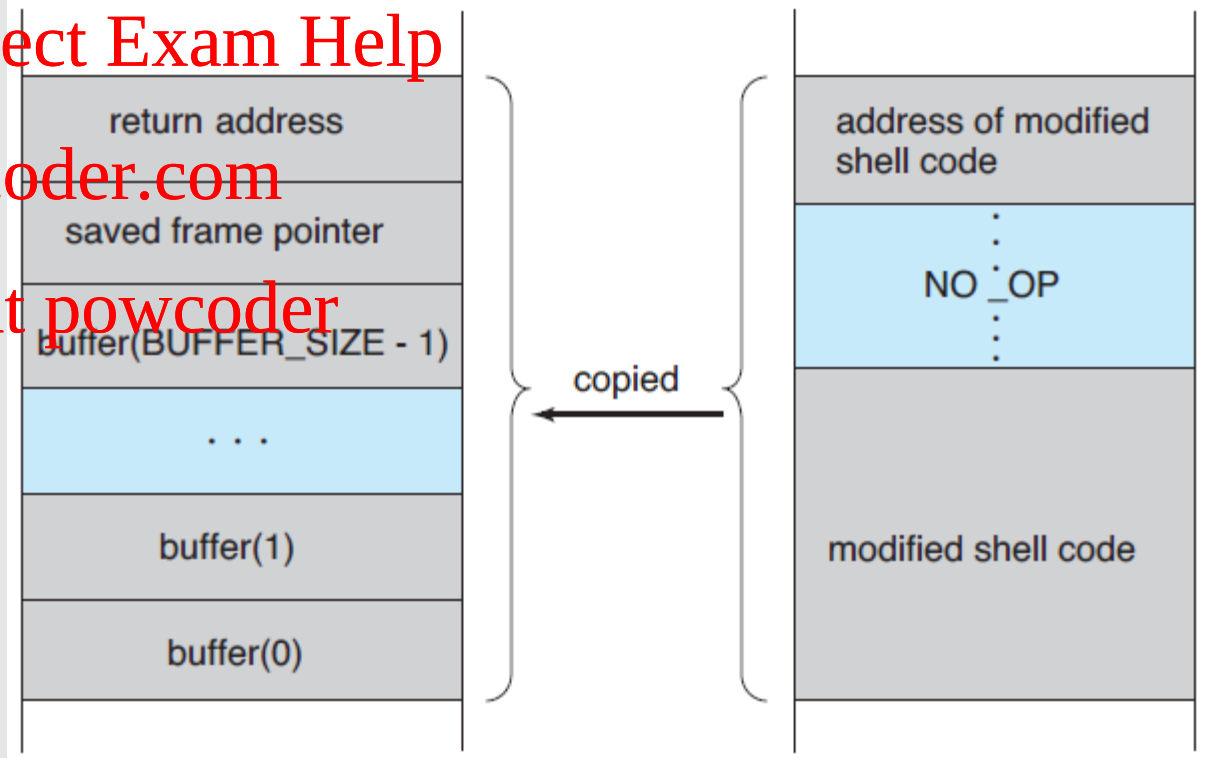
- Virus: malicious code hidden in program, copies itself into other programs
- Worm: malicious program that replicates itself over the network
- Trojan Horse: malicious code hidden in a program
- Logic Bomb: malicious program that activates itself on certain conditions
- Backdoor: hidden way to get control of the system bypassing authentication

Vulnerabilities

Example: Buffer overflow, e.g. `strcpy(buffer, argv[1])` in C

Defenses:

- Stack protection
  (e.g., canaries, NX bits, randomisation)
- Safe programming languages
  (e.g. Java)



Other vulnerabilities:

- SQL injection, cross-site scripting, etc.

https://cve.mitre.org/

Design for Security

- Open design (not: “security by obscurity"):
  Open source code of security mechanisms increases chance to find and patch vulnerabilities

- Principle of least privilege:
  e.g. default setting: no permissions

- Economy of mechanisms:
  Simplicity reduces number of possible bugs

- Acceptability:
  e.g. must not impact availability

Summary

Security goals ("CIA")

- Confidentiality
- Integrity
- Availability

Defenses

- Authentication
- Accounting
- Access control
- Isolation

Threat, attack, vulnerability, exploit, violation

Read

- Tanenbaum & Bos., Modern Operating Systems
  - Chapter 5
- Silberschatz et al., Operating System Concepts
  - Chapter 14 & 15

Next Lecture

- Introduction
- Operating System Architectures
- Processes
- Threads - Programming
- Process Scheduling - Evaluation
- Process Synchronisation
- Deadlocks
- Memory Management
- File Systems
- Input / Output
- Security
- **Virtualisation**