Operating Systems
INF333

TP08
Inter-Process Synchronization

Eda Bahar 16/05/2025
ebahar@gsu.edu.tr
edaabahar@gmail.com

About this TP

- In this TP, you will be learning:
  - Inter-process synchronization

Inter-process synchronization

Definition

- **Inter-process synchronization** is a coordination mechanism that ensures multiple processes can safely access shared resources, such as memory, files, or devices, without interfering with each other or causing data inconsistencies.

- Prevent race conditions between processes and ensure consistency.

Real-world examples needed synchronization

- **File Locks**: prevent simultaneous writes.
- **Databases**: ACID properties enforce synchronization.
- **Kernel**: Manages access to memory and devices.
- **Web Servers**: Thread-pools synchronize access to sockets and memory.

Inter-process synchronization with POSIX

- Semaphores
- File locking with fcntl: used for files
- Mutexes

```
struct flock lock;
lock.l_type = F_WRLCK;
fcntl(fd, F_SETLKW, &lock); // blocking write lock

write(fd, "Hello\n", 6); // write to file

lock.l_type = F_UNLCK;
fcntl(fd, F_SETLK, &lock); // unlock
```

Example with mutex

```c

#include <pthread.h>
#include <fcntl.h>
#include <sys/mman.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main() {
    int fd = open("/mymutex", O_CREAT | O_RDWR, 0666);
    ftruncate(fd, sizeof(pthread_mutex_t));
    pthread_mutex_t *mutex = mmap(NULL, sizeof(pthread_mutex_t),
                                  PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);

    pthread_mutexattr_t attr;
    pthread_mutexattr_init(&attr);
    pthread_mutexattr_setpshared(&attr, PTHREAD_PROCESS_SHARED);
    pthread_mutex_init(mutex, &attr);

    for (int i = 0; i < 5; i++) {
        pthread_mutex_lock(mutex);
        printf("Process PID %d in critical section.\n", getpid());
        sleep(1);
        pthread_mutex_unlock(mutex);
        sleep(1);
    }

    pthread_mutex_destroy(mutex);
    return 0;
}

```