운영체제 프로젝트 중간보고서

2. 프로젝트 진행 상황

a. system call의 구조

시스템 콜은 사용자 공간에서 커널 공간으로의 전환을 통해 수행됨

1) User mode

- User mode에서 실행 중인 응용프로그램에서 시스템 콜을 통해 커널에 특정 작업을 요청함. 이때, system call은 라이브러리 함수를 통해 호출됨.

2) System call Interface

- User mode에서 호출된 라이브러리 함수는 실제 system call을 수행하기 위해 system call 인터페이스를 사용. system call 인터페이스에서 사용자 공간에서 커널 공간으로의 전환을 처리함.
- 이 전환 과정은 먼저 system call 번호와 인수들을 레지스터에 설정하고 소프트웨어 인터럽트 또는 시스템 트랩을 발생시켜 사용자 공간에서 커널 모드로 전환함.

3) Kernel mode

- Kernel mode로 전환되면 커널의 system call handler가 호출됨. handler는 system call 번호를 확인하고 적절한 kernel 함수를 호출함.

4) User mode로 반환

- system call이 완료되면 결과가 일반적으로 레지스터를 통해 반환됨. 그 후 user mode에서 해당 결과를 처리하고 응용 프로그램의 나머지를 수행.

b. 새로운 system call 생성

1) System call 함수 정의

- 새로운 system call 함수를 kernel 소스에 정의.

```
// kernel/sys_hello.c

#include <linux/kernel.h>
#include <linux/syscalls.h>

// 시스템 콜 함수 정의
asmlinkage long sys_hello(void) {
    printk(KERN_INFO "Hello, world!\n");
    return 0;
}
```

2) System call 테이블에 등록

- 새로운 system call을 system call table에 등록 해야함. system call table 은 아키텍처 별로 다를 수 있기 때문에 적절한 파일을 수정해야 함.

```
// arch/x86/entry/syscalls/syscall_64.tbl

...
333     common  sys_hello                sys_hello
```

3) System call 선언

- system call을 kernel에 선언.

```
// include/linux/syscalls.h

asmlinkage long sys_hello(void);
```

4) Kernel 구성 및 컴파일

- 위 수정 사항을 적용하고, kernel을 다시 컴파일하고 설치함.