**2018063218 컴퓨터소프트웨어학부 최성진**

**운영 체제 HW#7**

**제출 일자 : 2021/04/29**

1. **과제 A**
2. **시스템 콜 추가 과정 설명**
3. **새로운 시스템 호출을 System Call Table에 등록**

**텍스트, 스크린샷, 전자기기이(가) 표시된 사진

자동 생성된 설명**

323번에 get\_task\_info라는 이름으로 entry point는 sys\_get\_task\_info로 등록을 하였습니다. 이때, Common은 Application Binary Interface로 수정하지 않은 경우(common일 경우)컴파일러 또는 링커에서 entry point가 생성됩니다.

1. **시스템 호출 함수 선언 및 구현**

**텍스트이(가) 표시된 사진

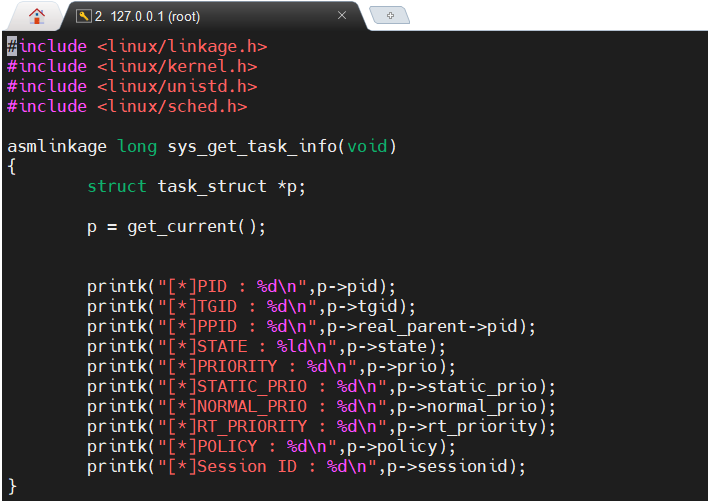
자동 생성된 설명**

Asmlinkage long sys\_get\_task\_info(void)의 형태로 선언했습니다. 이 때, Asmlinkage라는 뜻은 어셈블리 코드에서 직접 호출(링크)할 수 있다는 의미이며, 커널 소스의 <include/inux/linkage.h>에 다음과 같이 정의되어 있습니다.

#ifndef asmlinkage  
#define asmlinkage CPP\_ASMLINKAGE  
#endif

일반적으로 C는 어셈블리 코드에서 별 어려움 없이 호출할 수 있지만 함수의 인자를 넘길 때 x86 아키텍처의 경우 레지스터, 스택 혹은 별도의 메모리 영역에 함수의 인자를 저장하여 넘길 수 있도록 지원합니다. 하지만, 직접 작성한 어셈블리 코드에서 함수를 호출할 경우 문제가 발생할 수 있으므로 레지스터를 이용하지 않고 스택을 이용해서 인자를 전달하도록 선언하게 되는데 이 때 사용하는 것이 asmlinkage입니다.

1. **내부에 시스템 호출함수 구현**



<include/linux/sched.h>에 선언되어있는 task\_struct구조체의 내용을 살펴보면 state와 flag, ptrace 등 다양한 변수들이 저장되어있습니다.

Get\_current()함수를 호출하게 되면 이러한 task\_struct를 가리키는 포인터가 반환이 되는데 이를 통해 현재 프로세스(혹은 스레드)의 정보를 얻을 수 있습니다.

따라서, printk(커널영역에서의 프린트문)를 통해 state, pid, flags등을 직접 print해보겠습니다.

Pid : task id

Tgid : task group id

Ppid : parent process id

State : 태스크는 기본적으로 생성부터 소멸까지 많은 상태를 거치는데 이에 대한 관리 변수이다.

Priority: 프로세스에 할당된 정적 우선순위 값을 가지고 있습니다.

Static\_prio : user space에서 전달된 값으로 갱신된 nice값을 priority로 변환한 값입니다.

Normal\_prio : 일반적으로 priority와 같은 값을 가지게 됩니다.

Rt\_priority : realtime scheduling을 위해 1.3.81버전부터 지원하는 우선순위 설정값입니다.

Policy : 스케줄링 정책에대한 설정으로 0,1,2값을 가질 수 있습니다.

Session id : 말그대로 세션 id입니다.

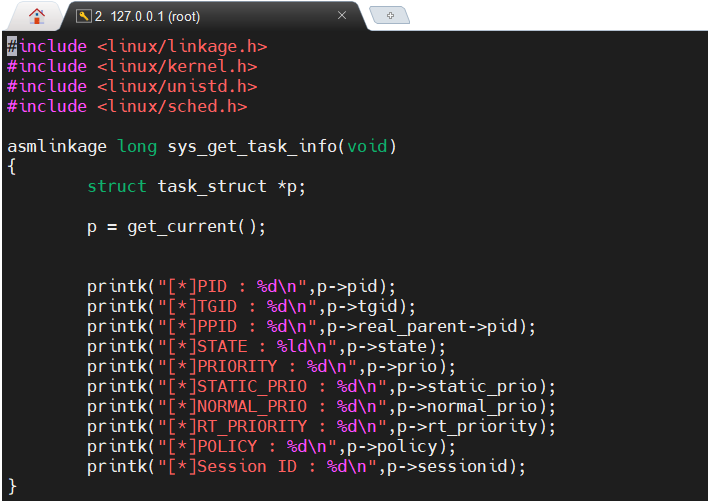
1. **테스트 프로그램 작성 및 동작 확인**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

Main문에서 syscall(323)을 호출하게 되면 실제로 시스템콜 넘버인 323번을 호출하게 되고 그에따른 출력문을 dmesg 명령어를 통해 확인해보았습니다.

1. **추가한 시스템 콜 설명**



여기서 추가한 시스템 콜은 task\_struct 포인터를 생성하여 get\_current()함수를 호출하여 현재 프로세스의 정보가 담긴 포인터를 호출하는 함수입니다.

**가상머신 링크 : https://drive.google.com/file/d/1v\_q6YEGOspyWBIETkeyxHFYk6KWKowQX/view?usp=sharing**