정글 8기

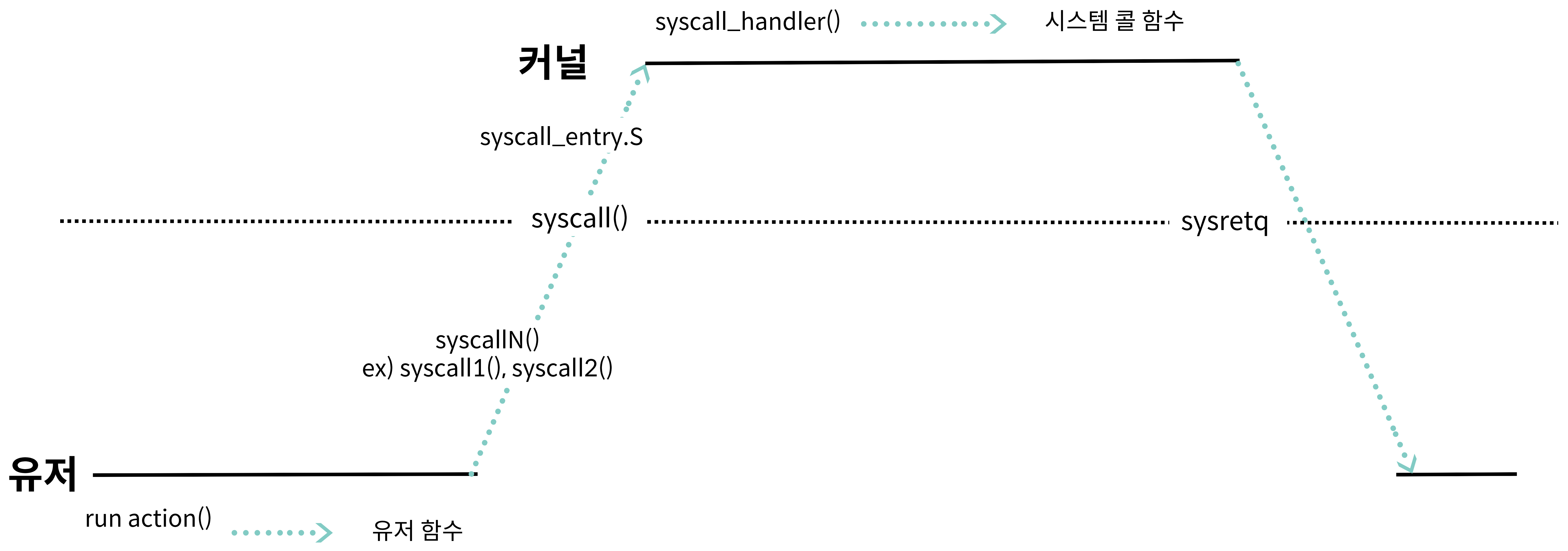
[WEEK07-11] 정글끝까지

WEEK08-09 WIL

System Call을 실행하는 과정

TEAM7 김태훈-장유선-정재혁

System Call


syscall_handler()
시스템 콜 함수
커널
syscall_entry.S
syscall()
sysretq
syscallN()
ex) syscall1(), syscall2()
유저
run action()
유저 함수

유저 함수

```c
static void __do_fork(void *aux) {
    struct intr_frame if_;
    struct thread *parent = (struct thread *)aux;
    struct thread *current = thread_current();
    bool succ = true;

    ...

error:
    sema_up(&current->fork_sema);
    exit(TID_ERROR);
}
```

시스템 콜 호출 함수

```c
void halt(void) {
    power_off();
}

void exit(int status) {
    thread_t *curr = thread_current();
    curr->exit_status = status;

    printf("%s: exit(%d)\n", curr->name, curr->exit_status);

    thread_exit();
}

pid_t fork(const char *thread_name) {
    check_address(thread_name);

    return process_fork(thread_name, NULL);
}
...
```

```c
#define syscall0(NUMBER) ( \
    syscall(((uint64_t)NUMBER), 0, 0, 0, 0, 0, 0))

#define syscall1(NUMBER, ARG0) ( \
    syscall(((uint64_t)NUMBER),  \
            ((uint64_t)ARG0), 0, 0, 0, 0, 0))
...
```

SYSCALL

```c
__attribute__((always_inline)) static __inline
int64_t syscall(uint64_t num_, uint64_t a1_, uint64_t a2_,
                uint64_t a3_, uint64_t a4_, uint64_t a5_, uint64_t a6_) {
    int64_t ret;
    register uint64_t *num asm("rax") = (uint64_t *)num_;
    register uint64_t *a1 asm("rdi") = (uint64_t *)a1_;
    register uint64_t *a2 asm("rsi") = (uint64_t *)a2_;
    register uint64_t *a3 asm("rdx") = (uint64_t *)a3_;
    register uint64_t *a4 asm("r10") = (uint64_t *)a4_;
    register uint64_t *a5 asm("r8") = (uint64_t *)a5_;
    register uint64_t *a6 asm("r9") = (uint64_t *)a6_;

    __asm __volatile(
        "mov %1, %%rax\n"
        "mov %2, %%rdi\n"
        "mov %3, %%rsi\n"
        "mov %4, %%rdx\n"
        "mov %5, %%r10\n"
        "mov %6, %%r8\n"
        "mov %7, %%r9\n"
        "syscall\n"
        : "=a"(ret)
        : "g"(num), "g"(a1), "g"(a2), "g"(a3), "g"(a4), "g"(a5), "g"(a6)
        : "cc", "memory");
    return ret;
}
```

SYSCALL ENTRY

```
syscall_entry:
    movq %rbx, temp1(%rip)
    movq %r12, temp2(%rip)     /* callee saved registers */
    movq %rsp, %rbx            /* Store userland rsp    */
    movabs $tss, %r12
    movq (%r12), %r12
    movq 4(%r12), %rsp         /* Read ring0 rsp from the tss */
    /* Now we are in the kernel stack */
    push $(SEL_UDSEG)      /* if->ss */
    push %rbx              /* if->rsp */
    push %r11              /* if->eflags */
    push $(SEL_UCSEG)      /* if->cs */

    ...

    push %r10
    pushq $0 /* skip r11 */
    movq temp2(%rip), %r12
    push %r12
    push %r13
    push %r14
    push %r15
    movq %rsp, %rdi
```

```c
tid_t process_fork(const char *name, struct intr_frame *if_ UNUSED) {
    struct intr_frame *f = (pg_round_up(rrsp()) - sizeof(struct intr_frame));

    ...

}
```

SYSCALL ENTRY

```
check_intr:
    btsq $9, %r11           /* Check whether we recover the interrupt */
    jnb no_sti
    sti                     /* restore interrupt */
no_sti:
    movabs $syscall_handler, %r12
    call *%r12
    popq %r15
    popq %r14
    popq %r13
    popq %r12
    popq %r11
    popq %r10
    popq %r9
    popq %r8
    popq %rsi
    popq %rdi
    popq %rbp
    popq %rdx
    popq %rcx
    popq %rbx
    popq %rax
    addq $32, %rsp
    popq %rcx               /* if->rip */
    addq $8, %rsp
    popq %r11               /* if->eflags */
    popq %rsp               /* if->rsp */
    sysretq
```

정글 8기

[WEEK07-11] 정글끝까지

감사합니다

TEAM7 김태훈-장유선-정재혁