Xilinx Zynq FPGA, TI DSP, MCU기반의 프로그래밍 및 회로 설계 전문가 과정

강사 - Innov (이상훈)
gcccompil3r@gmail.com
학생 - 이유성
dbtjd1102@naver.com

*커널 내부 소스코드 드라이빙

monitor_hack.c

syscall_hooking_init() 함수 분석

```
int syscall_hooking_init(void)
{
    unsigned long cr0;

    if((sys_call_table = locate_sys_call_table()) == NULL)
    {
        printk("<0>Can't find sys_call_table\n");
        return -1;
    }

    printk("<0>sys_call_table is at[%p]\n", sys_call_table);

    // CR0 레지스터를 읽어옴
    cr0 = read_cr0();
    // Page 쓰기를 허용함
    write_cr0(cr0 & ~0x00010000);

    /* set_memory_rw 라는 심볼을 찾아와서 fixed_set_memory_rw 에 설정함 */
    fixed_set_memory_rw = (void *)kallsyms_lookup_name("set_memory_rw");
    if(!fixed_set_memory_rw)
    {
        printk("<0>Unable to find set_memory_rw symbol\n");
        return 0;
    }

    /* 시스템 콜 테이블이 위치한 물리 메모리에 읽고 쓰기 권한 주기 */
    fixed_set_memory_rw(PAGE_ALIGN((unsigned long)sys_call_table) - PAGE_SIZE, 3);

    orig_call = (void *)sys_call_table[__NR_open];
    sys_call_table[__NR_open] = (void *)sys_our_open;
    write_cr0(cr0);
    printk("<0>Hooking Success!\n");
    return 0;
}
```

read_cr0() 함수

```
static inline unsigned long read_cr0(void)
{
    return native_read_cr0();
}
```

native_read_cr0() 함수

```
static inline unsigned long native_read_cr0(void)
{
    unsigned long val;
    asm volatile("mov %%cr0,%0\n\t" : "=r" (val), "=m" (__force_order));
    return val;
}
```

%0은 함수 내 맨 처음 정의된 변수 r은 레지스터 , m은 메모리

write_cr0 함수

```
static inline void write_cr0(unsigned long x)
{
    PVOP_VCALL1(pv_cpu_ops.write_cr0, x);
}
```

PVOP_VCALL1

```
#define PVOP_VCALL1(op, arg1)                                   \
    __PVOP_VCALL(op, "", "", PVOP_CALL_ARG1(arg1))
```

PVOP_VCALL

```
#define ____PVOP_VCALL(op, clbr, call_clbr, extra_clbr, pre, post, ...) \
    ({                                  \
        PVOP_VCALL_ARGS;                \
        PVOP_TEST_NULL(op);             \
        asm volatile(pre                \
                paravirt_alt(PARAVIRT_CALL)     \
                post                    \
                : call_clbr             \
                : paravirt_type(op),        \
                  paravirt_clobber(clbr),       \
                  ##__VA_ARGS__             \
                : "memory", "cc" extra_clbr);   \
    })

#define __PVOP_VCALL(op, pre, post, ...)                \
    ____PVOP_VCALL(op, CLBR_ANY, PVOP_VCALL_CLOBBERS,       \
                VEXTRA_CLOBBERS,                \
                pre, post, ##__VA_ARGS__)
```

kallsyms_lookup_name() 함수

```
unsigned long kallsyms_lookup_name(const char *name)
{
    char namebuf[KSYM_NAME_LEN];  //128
    unsigned long i;
    unsigned int off;

    for (i = 0, off = 0; i < kallsyms_num_syms; i++) {
        off = kallsyms_expand_symbol(off, namebuf, ARRAY_SIZE(namebuf));

        if (strcmp(namebuf, name) == 0)
            return kallsyms_addresses[i];
    }
    return module_kallsyms_lookup_name(name);
}
```

kallsyms_expand_symbol()함수

```
static unsigned int kallsyms_expand_symbol(unsigned int off,
                       char *result, size_t maxlen)
{
    int len, skipped_first = 0;
    const u8 *tptr, *data;

    /* Get the compressed symbol length from the first symbol byte. */
    data = &kallsyms_names[off];
    len = *data;
    data++;

    /*
     * Update the offset to return the offset for the next symbol on
     * the compressed stream.
     */
    off += len + 1;

    /*
     * For every byte on the compressed symbol data, copy the table
     * entry for that byte.
     */
    while (len) {
        tptr = &kallsyms_token_table[kallsyms_token_index[*data]];
        data++;
        len--;

        while (*tptr) {
            if (skipped_first) {
                if (maxlen <= 1)
                    goto tail;
                *result = *tptr;
                result++;
                maxlen--;
            } else
                skipped_first = 1;
            tptr++;
        }
    }

tail:
    if (maxlen)
        *result = '\0';

    /* Return to offset to the next symbol. */
    return off;
}
```

set_memory_rw() 함수

```
int set_memory_rw(unsigned long addr, int numpages)
{
    return change_page_attr_set(&addr, numpages, __pgprot(_PAGE_RW), 0);
}
```

change_page_attr_set() 함수

```
static inline int change_page_attr_set(unsigned long *addr, int numpages,
                       pgprot_t mask, int array)
{
    return change_page_attr_set_clr(addr, numpages, mask, __pgprot(0), 0,
        (array ? CPA_ARRAY : 0), NULL);
}
```

page_rw 정의

```
#define _PAGE_RW    (_AT(pteval_t, 1) << _PAGE_BIT_RW)
```

AT 함수 정의

```
#ifdef __ASSEMBLY__
#define _AC(X,Y)    X
#define _AT(T,X)    X
#else
#define __AC(X,Y)   (X##Y)
#define _AC(X,Y)    __AC(X,Y)
#define _AT(T,X)    ((T)(X))
#endif
```

pgrot 함수 정의

```
#define __pgprot(x) ((pgprot_t) { (x) } )
```