KWANGWOON UNIVERSITY

2024년 2학기 **운영체제실습** 5주차

Module Programming

System Software Laboratory
School of Computer and Information Engineering
Kwangwoon Univ.

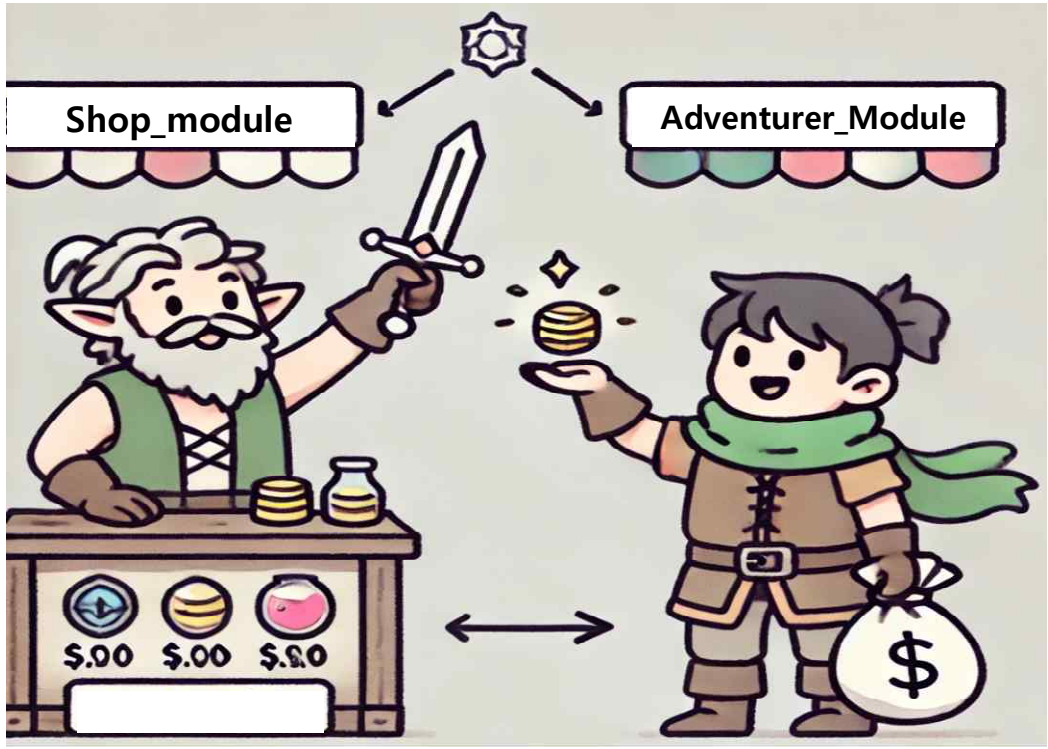
Module dependency

- **특정 커널 모듈은 때때로, 하나 이상의 다른 커널 모듈에 의존함**
  - /lib/modules/<KERNEL_VERSION>/modules.dep 파일에 모듈 종속 항목이 포함되어 있음
  - $depmod -a

```
kernel/net/ax25/ax25.ko:                                                    0개 의존
kernel/net/can/can.ko:
kernel/net/can/can-raw.ko: kernel/net/can/can.ko                            1개 의존
kernel/net/can/can-bcm.ko: kernel/net/can/can.ko
kernel/net/can/can-gw.ko: kernel/net/can/can.ko
kernel/net/bluetooth/bluetooth.ko: kernel/crypto/ecdh_generic.ko
kernel/net/bluetooth/rfcomm/rfcomm.ko: kernel/net/bluetooth/bluetooth.ko kernel/
crypto/ecdh_generic.ko                                                      2개 의존
kernel/net/bluetooth/bnep/bnep.ko: kernel/net/bluetooth/bluetooth.ko kernel/cryp
to/ecdh_generic.ko
kernel/net/bluetooth/cmtp/cmtp.ko: kernel/drivers/isdn/capi/kernelcapi.ko kernel
/net/bluetooth/bluetooth.ko kernel/crypto/ecdh_generic.ko                   3개 의존
kernel/net/bluetooth/hidp/hidp.ko: kernel/drivers/hid/hid.ko kernel/net/bluetoot
h/bluetooth.ko kernel/crypto/ecdh_generic.ko
```

예제 – Module dependency

- **Example**
  - 상점 상인과 모험가의 거래를 예시로 들어보자.
    - **Q1. 상점이 닫혀 있으면..?**
      - 상점(shop_module)이 열려 (insmod) 있어야 모험가(adventurer_module)는 아이템을 구매 가능하다.(Dependency)
    - **Q2. 모험가는 아이템의 가격을 어떻게 아는가..?**
      - 상점(shop_module)이 물건의 가격을 알려줘야 한다(EXPORT_SYMBOL 매크로 사용)



예제 – Module dependency(Cont'd)

Shop_module.c

```c
// shop_module.c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Shop Module for RPG Game");
MODULE_VERSION("1.0");

// 상점에서 판매할 아이템의 가격
int sword_price = 100; // 100골드짜리 검
int potion_price = 50; // 50골드짜리 포션

// 상점에서 아이템 구매 함수
int buy_item(const char *item) {
    if (strcmp(item, "sword") == 0) {
        printk(KERN_INFO "shop_module: 검을 구매했습니다! 가격: %d골드\n", sword_price);
        return sword_price;
    } else if (strcmp(item, "potion") == 0) {
        printk(KERN_INFO "shop_module: 포션을 구매했습니다! 가격: %d골드\n", potion_price);
        return potion_price;
    } else {
        printk(KERN_INFO "shop_module: 해당 아이템은 없습니다.\n");
        return -1; // 아이템 없음
    }
}
```

예제 – Module dependency(Cont'd)

Shop_module.c

```
28 // 변수를 외부에 내보냄
29 EXPORT_SYMBOL(sword_price);
30 EXPORT_SYMBOL(potion_price);
31
32 // 함수를 외부에 내보냄
33 EXPORT_SYMBOL(buy_item);
34
35 // 모듈 초기화
36 static int __init shop_module_init(void) {
37     printk(KERN_INFO "shop_module: 상점이 열렸습니다!\n");
38     return 0;
39 }
40
41 // 모듈 종료
42 static void __exit shop_module_exit(void) {
43     printk(KERN_INFO "shop_module: 상점이 닫혔습니다.\n");
44 }
45
46 module_init(shop_module_init);
47 module_exit(shop_module_exit);
```

예제 – Module dependency(Cont'd)

adventurer_module.c

```c
// adventurer_module.c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Adventurer Module for RPG Game");
MODULE_VERSION("1.0");

// 상점 모듈에서 내보낸 변수와 함수 선언
extern int sword_price;
extern int potion_price;
extern int buy_item(const char *item);

// 모듈 초기화
static int __init adventurer_module_init(void) {
    int gold = 150; // 모험가의 초기 골드
    int cost;

    printk(KERN_INFO "adventurer_module: 모험가가 상점에 방문했습니다.\n");
    printk(KERN_INFO "adventurer_module: 모험가의 소지 골드: %d골드\n", gold);

    // 검 구매 시도
    cost = buy_item("sword");
    if (cost > 0 && gold >= cost) {
        gold -= cost;
        printk(KERN_INFO "adventurer_module: 검을 구매했습니다! 남은 골드: %d골드\n", gold);
    } else {
        printk(KERN_INFO "adventurer_module: 골드가 부족합니다!\n");
    }
```

예제 – Module dependency(Cont'd)

adventurer_module.c

```
31
32      // 포션 구매 시도
33      cost = buy_item("potion");
34      if (cost > 0 && gold >= cost) {
35          gold -= cost;
36          printk(KERN_INFO "adventurer_module: 포션을 구매했습니다! 남은 골드: %d골드\n", gold);
37      } else {
38          printk(KERN_INFO "adventurer_module: 골드가 부족합니다!\n");
39      }
40
41      return 0;
42  }
43
44  // 모듈 종료
45  static void __exit adventurer_module_exit(void) {
46      printk(KERN_INFO "adventurer_module: 모험가가 상점을 떠났습니다.\n");
47  }
48
49  module_init(adventurer_module_init);
50  module_exit(adventurer_module_exit);
```

예제 – Module dependency(Cont'd)

Makefile

```
obj-m += shop_module.o
obj-m += adventurer_module.o

PWD = $(CURDIR)

all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

예제 – Module dependency(Cont'd)

- **Step 01. insert shop_module**

```
root@ubuntu:/home/os2024123456/week5# insmod shop_module.ko
root@ubuntu:/home/os2024123456/week5# lsmod | grep shop_module
shop_module            16384  0
root@ubuntu:/home/os2024123456/week5# dmesg | tail -n 1
[716305.454606] shop_module: 상점이 열렸습니다!
```

- **Step 02. insert adventurer_module**

```
root@ubuntu:/home/os2024123456/week5# insmod adventurer_module.ko
root@ubuntu:/home/os2024123456/week5# lsmod | grep adventurer_module
adventurer_module      16384  0
shop_module            16384  1 adventurer_module
root@ubuntu:/home/os2024123456/week5# dmesg | tail -n 7
[716305.454606] shop_module: 상점이 열렸습니다!
[716424.908480] adventurer_module: 모험가가 상점에 방문했습니다.
[716424.908483] adventurer_module: 모험가의 소지 골드: 150골드
[716424.908484] shop_module: 검을 구매했습니다! 가격: 100골드
[716424.908485] adventurer_module: 검을 구매했습니다! 남은 골드: 50골드
[716424.908486] shop_module: 포션을 구매했습니다! 가격: 50골드
[716424.908486] adventurer_module: 포션을 구매했습니다! 남은 골드: 0골드
root@ubuntu:/home/os2024123456/week5#
```

예제 – Module dependency(Cont'd)

- Q1. 상점을 열지 않았는데 모험가가 들어오려고 하면?

```
root@ubuntu:/home/os2024123456/week5# lsmod | grep shop_module
root@ubuntu:/home/os2024123456/week5# insmod adventurer_module.ko
insmod: ERROR: could not insert module adventurer_module.ko: Unknown symbol in module
```

- Q2. 모험가가 떠나기 전에 상점을 닫으면?

```
root@ubuntu:/home/os2024123456/week5# lsmod | grep shop_module
shop_module            16384  1 adventurer_module
root@ubuntu:/home/os2024123456/week5# lsmod | grep adventurer_module
adventurer_module      16384  0
shop_module            16384  1 adventurer_module
root@ubuntu:/home/os2024123456/week5# rmmod shop_module
rmmod: ERROR: Module shop_module is in use by: adventurer_module
```

KWANGWOON UNIVERSITY

Assignment 2

System Software Laboratory
School of Computer and Information Engineering
Kwangwoon Univ.

Assignment 2-3

- **Directory**
  - root@ubuntu:/home/os2024123456/hooking# ls
  - abc.txt ftracehooking.c ftracehooking.h iotracehooking.c Makefile test.c

- **Test case**
  - Test.c

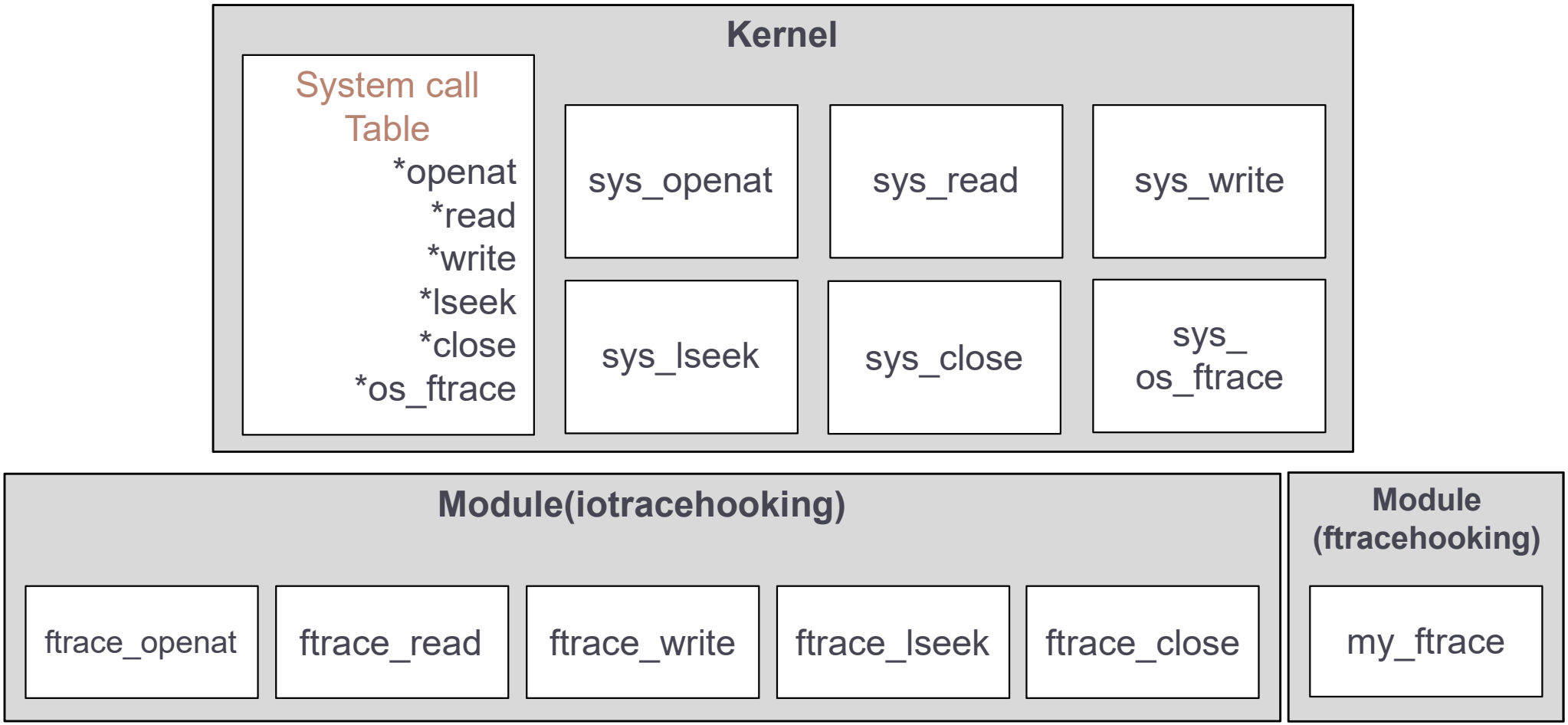
```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/syscall.h>

int main()
{
        syscall(336, getpid());
        int fd = 0;
        char buf[50];
        fd = open("abc.txt", O_RDWR);
        for (int i = 1; i <= 4; ++i)
        {
                read(fd, buf, 5);
                lseek(fd, 0, SEEK_END);
                write(fd, buf, 5);
                lseek(fd, i*5, SEEK_SET);
        }
        lseek(fd, 0, SEEK_END);
        write(fd, "HELLO", 6);
        close(fd);
        syscall(336,0);
        return 0;

}
```

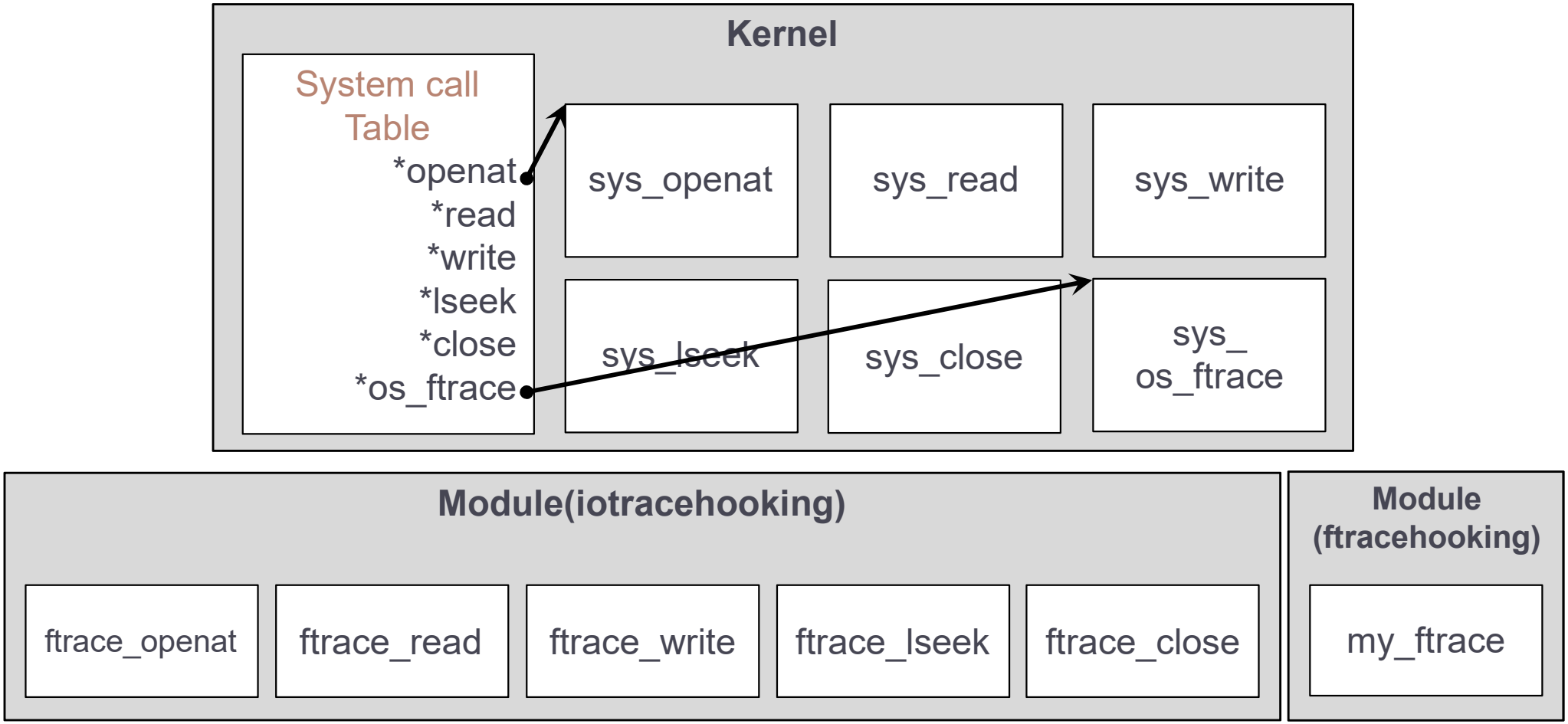
Assignment 2-3

- **Behaviors(following openat system call)**
  - **Before** inserting modules



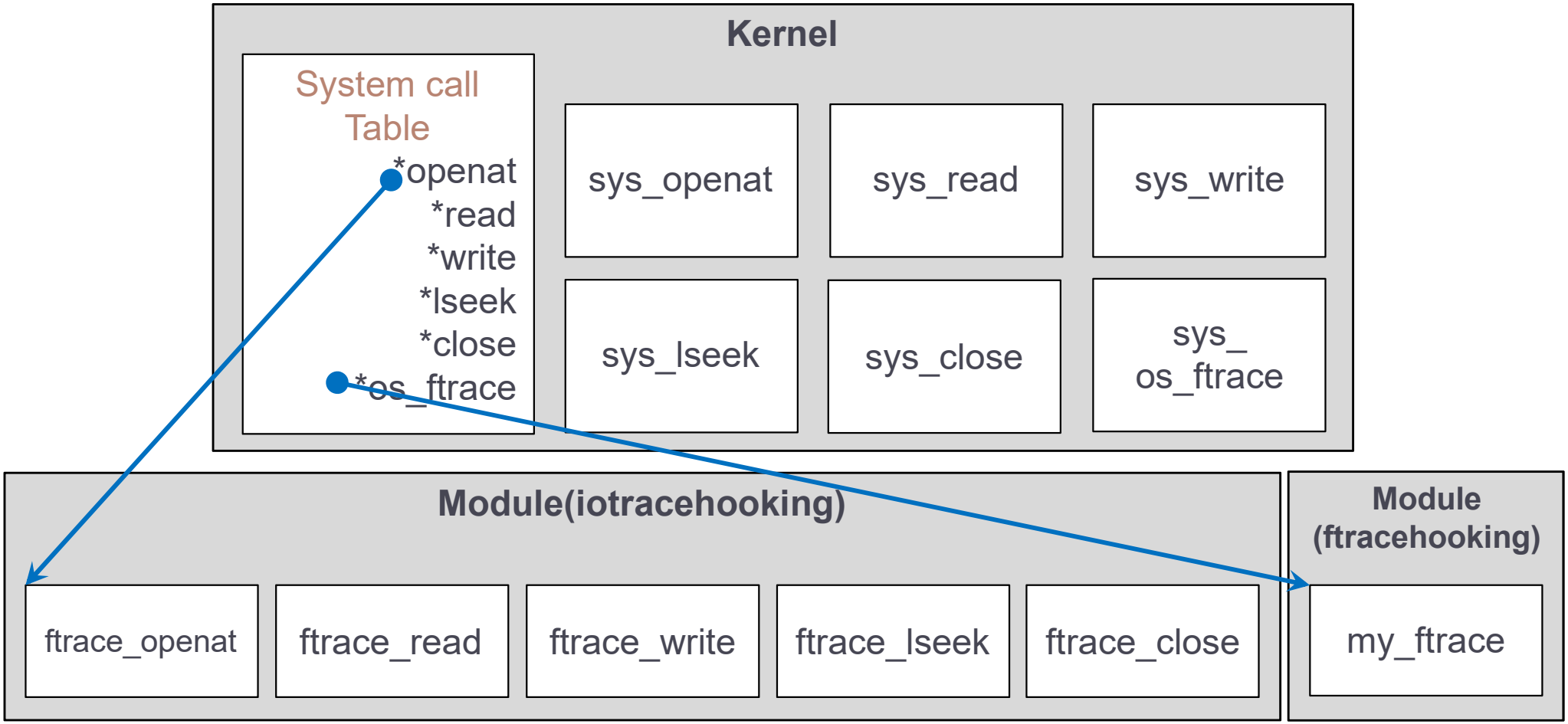
Assignment 2-3

- **Behaviors(following openat system call)**
  - **Before** inserting modules
  - (0)Systemcall table은 sys_openat, sys_os_ftrace의 주소를 가지고 있음



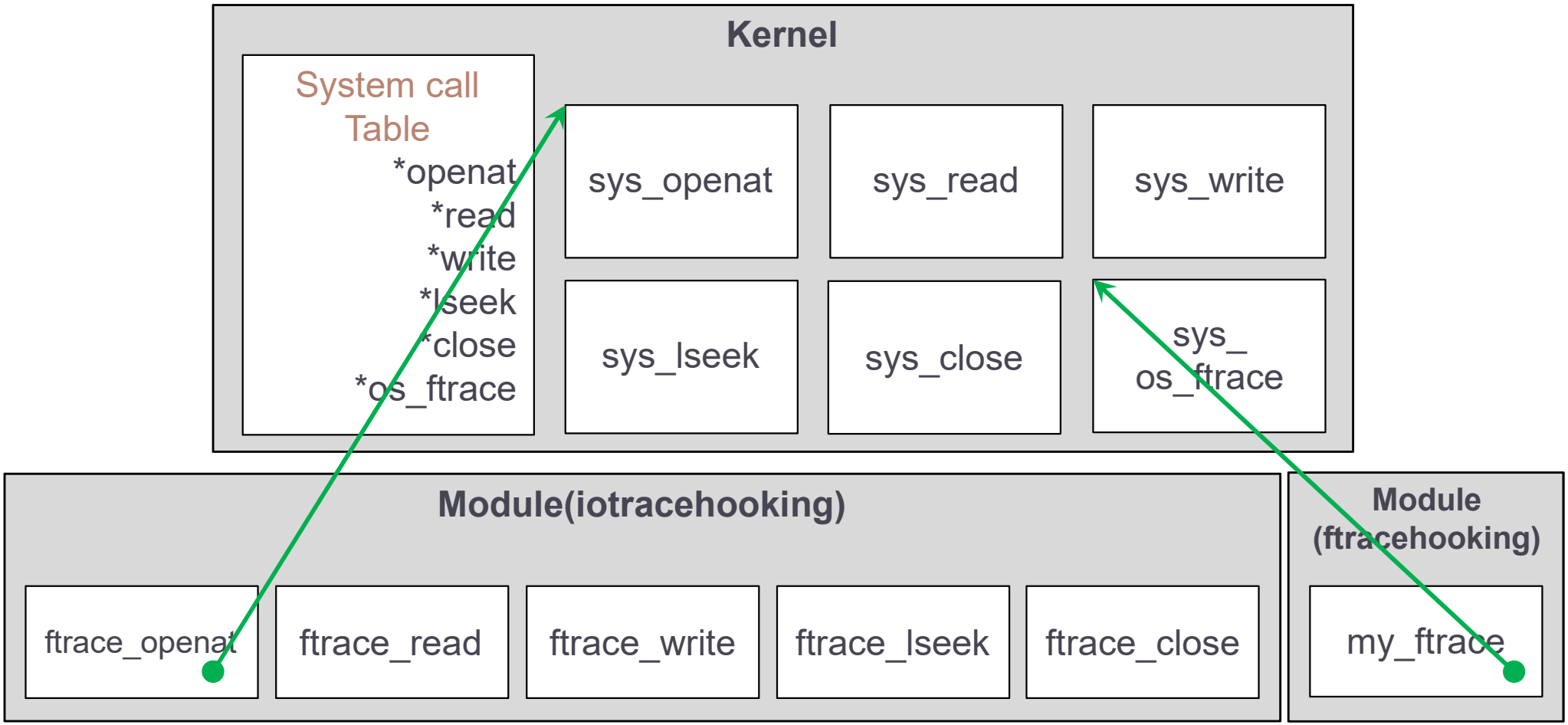
Assignment 2-3

- **Behaviors(following openat system call)**
  - **After** inserting modules
    - (1)Systemcall table 의 주소가 ftrace_openat, my_ftrace 로 변경(wrapping)



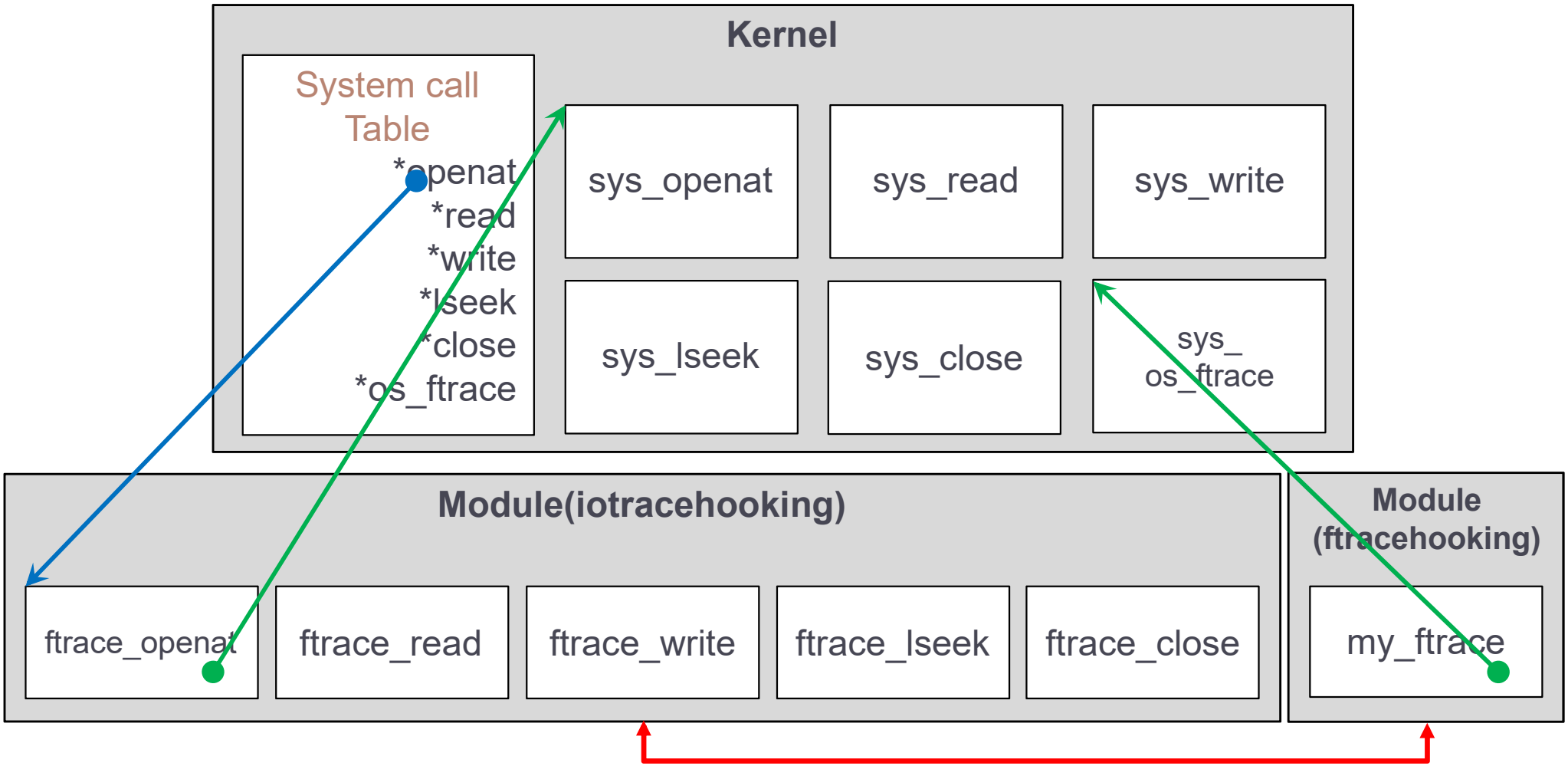
Assignment 2-3

- **Behaviors(following openat system call)**
  - **After** inserting modules
    - (2) ftrace_openat 함수는 sys_openat을 리턴 / my_ftrace 는 sys_os_ftrace 를 리턴



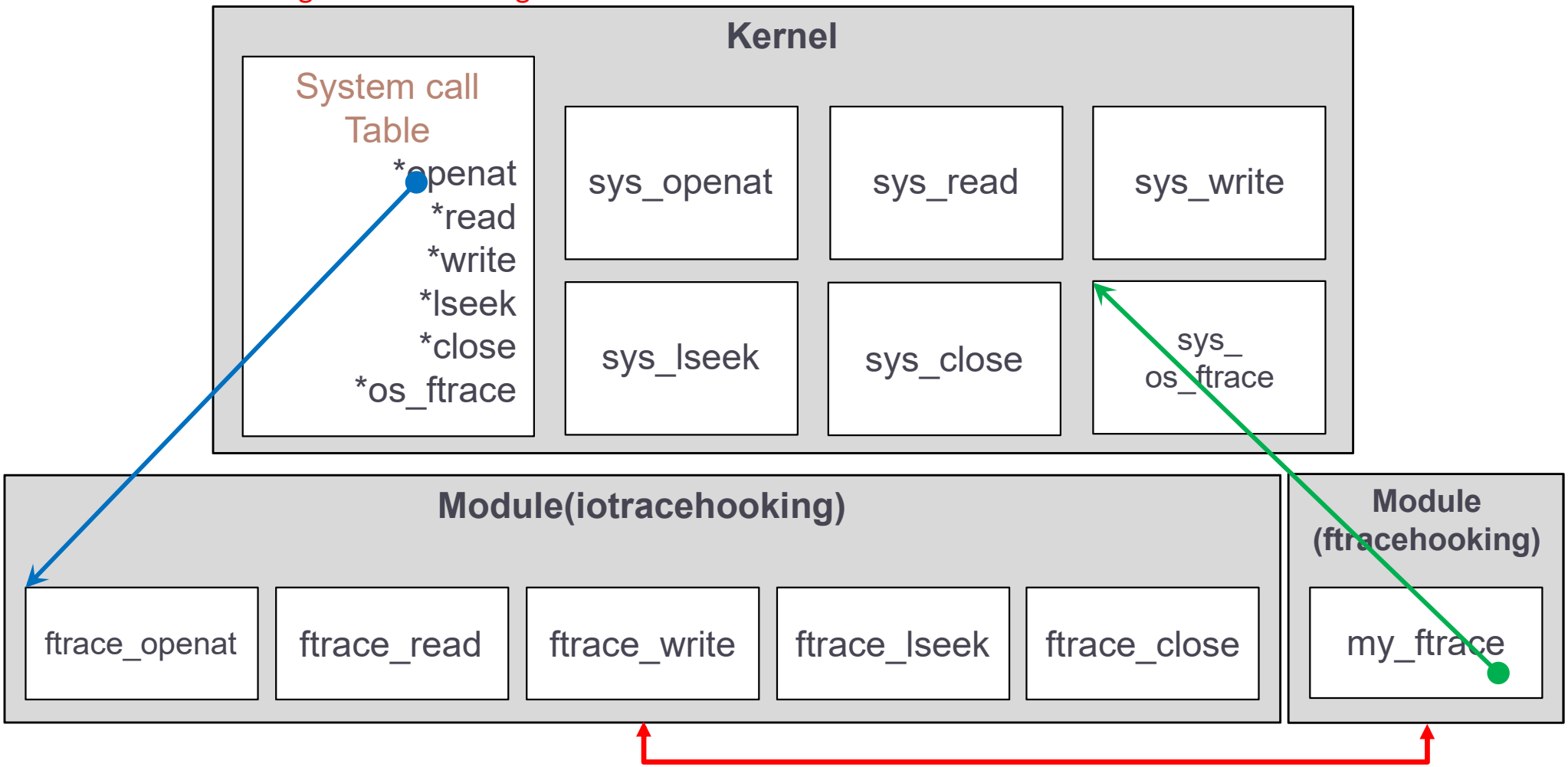
Assignment 2-3

- **Behaviors(following openat system call)**
  - **After** inserting modules
  - (3) ftrace_openat 함수는 sys_openat을 리턴 / my_ftrace 는 sys_os_ftrace 를 리턴



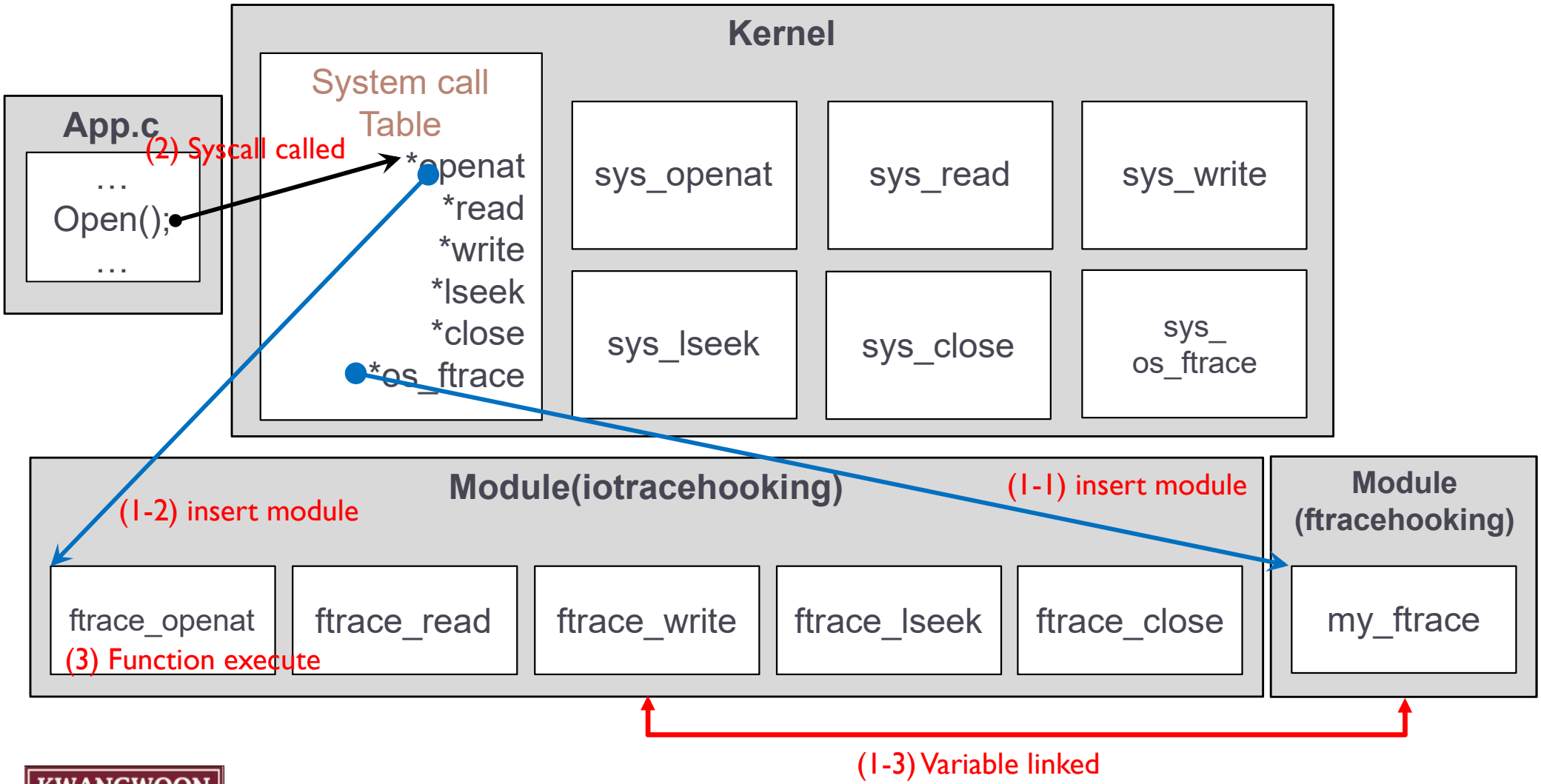
Assignment 2-3

- **Behaviors(following openat system call)**
  - **After** inserting modules
  - (4) ftrace_openat 함수에서 측정한 값을 EXTERNEL SYMBOL 매크로를 사용하여 iotracehooking -> ftracehooking 으로 전달



Assignment 2-3

- **Behaviors(following openat system call)**
  - Process Flow : **After** inserting modules



KWANGWOON UNIVERSITY

Appendix

System Software Laboratory
School of Computer and Information Engineering
Kwangwoon Univ.

Strace

- 시스템 호출(system call) 및 신호(signal) 추적하는 디버깅 툴
  - $strace ./test.out

```
root@ubuntu:/home/os2024123456/hooking# strace ./test.out
execve("./test.out", ["./test.out"], 0x7ffd801aa070 /* 22 vars */) = 0
brk(NULL)                               = 0x5583bb6a9000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffe169f67f0) = -1 EINVAL (Invalid argument)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=61814, ...}) = 0
mmap(NULL, 61814, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fcc1a5a4000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\300A\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\7\2C\n\357_\243\335\2449\206V>\237\374\304"..., 68, 880) = 68
fstat(3, {st_mode=S_IFREG|0755, st_size=2029592, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fcc1a5a2000
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0", 32, 848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\7\2C\n\357_\243\335\2449\206V>\237\374\304"..., 68, 880) = 68
mmap(NULL, 2037344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fcc1a3b0000
mmap(0x7fcc1a3d2000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x22000) = 0x7fcc1a3d2000
mmap(0x7fcc1a54a000, 319488, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x19a000) = 0x7fcc1a54a000
mmap(0x7fcc1a598000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7fcc1a598000
mmap(0x7fcc1a59e000, 13920, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fcc1a59e000
close(3)                                = 0
```

Strace

- **Options**
  - **-e : 조회 대상의 systemcall을 한정하여 출력**
    - $strace -e [조회하려는 시스템콜 이름] [명령어 or 실행파일] <- 단일 명령어 조회
    - $strace -e trace=[조회하려는 시스템콜들..] [명령어 or 실행파일] <- 다중 명령어 조회

```
root@ubuntu:/home/os2024123456/hooking# strace -e openat ./test.out
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
syscall_0x150(0x144f1, 0x7ffe87942ca8, 0x7fe69b1b40cb, 0, 0x7fe69b2e5d60, 0xc2) = 0
openat(AT_FDCWD, "abc.txt", O_RDWR)     = 3
syscall_0x150(0, 0x6, 0x7fe69b1dea37, 0x7fe69b2e5d60, 0xc2, 0xc2) = 0
+++ exited with 0 +++
```

  - **-c : 조회하는 대상의 system call 정보를 출력**
    - $strace –c [명령어 or 실행파일]

```
root@ubuntu:/home/os2024123456/hooking# strace -c ./test.out
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
  0.00    0.000000           0         5           read
  0.00    0.000000           0         5           write
  0.00    0.000000           0         3           close
  0.00    0.000000           0         2           fstat
  0.00    0.000000           0         9           lseek
  0.00    0.000000           0         7           mmap
  0.00    0.000000           0         3           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         1           brk
  0.00    0.000000           0         6           pread64
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           getpid
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2         1 arch_prctl
  0.00    0.000000           0         3           openat
  0.00    0.000000           0         2           (null)
------ ----------- ----------- --------- --------- ----------------
100.00    0.000000                    52         2 total
```