

# OPTEE 개인 텀프로젝트

2024. 11. 26

소프트웨어 및 시스템 보안  
00 분반  
TA 김기훈

# OP-TEE 개인 텀프로젝트

## 1. 목적

- OP-TEE를 통해 TEE(Trusted Execution Environment)에 대해 이해를 하고 OP-TEE 내에서 동작하는 어플리케이션(TA, Trusted Application + CA, Client Application)을 작성해본다.

## 2. 기본 사항

- OP-TEE가 빌드된 VM 이미지 활용하여 설치 진행

<https://drive.google.com/file/d/1yU3M-2ZK1v3Bq9bBcYakt7tUBBwqWpLm/view?usp=sharing>

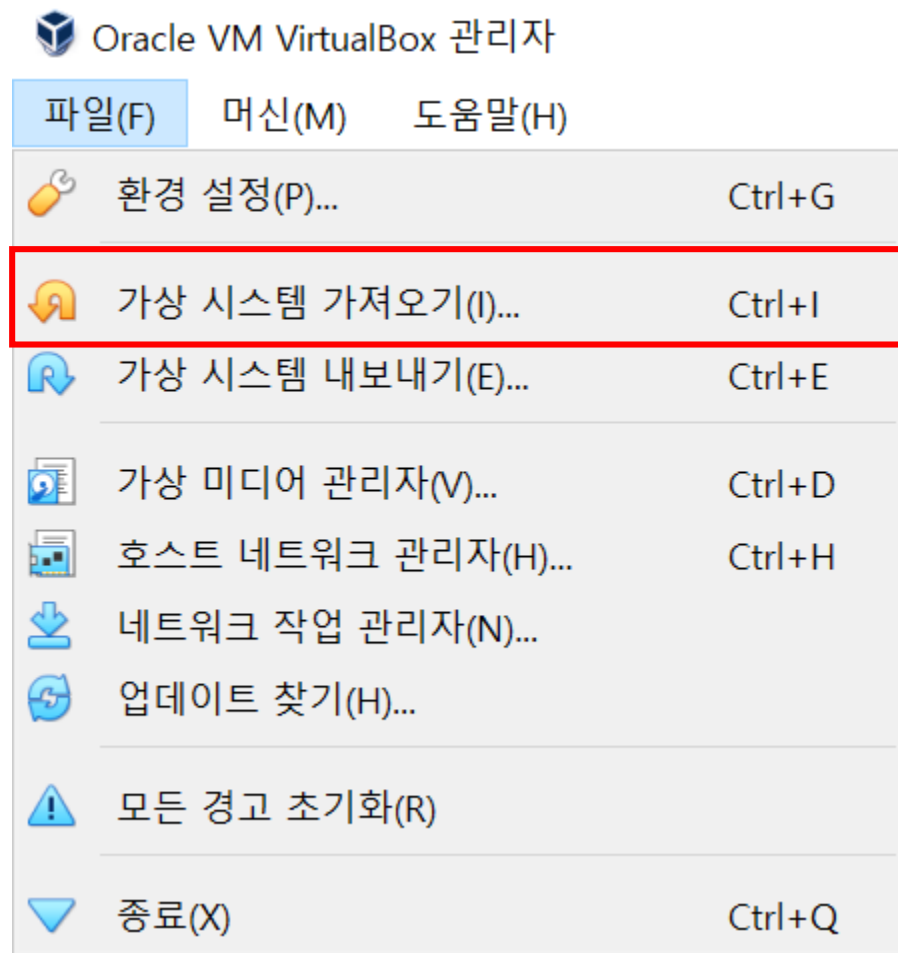
Virtualbox 6.0.4(기존 virtualbox 설치된 버전이 있다면 사용해도 무방)

[https://www.virtualbox.org/wiki/Download\\_Old\\_Builds\\_6\\_0](https://www.virtualbox.org/wiki/Download_Old_Builds_6_0)

# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 설치

- 파일 -> 가상 시스템 가져오기



# OP-TEE 개인 템프로젝트

## ■ OP-TEE 설치

- OPTEE.ova 파일 선택 -> 다음 -> 가져오기

← 가상 시스템 가져오기

### 가져올 가상 시스템

VirtualBox에서는 열린 가상화 형식(OVF)으로 저장된 가상 시스템을 가져올 수 있습니다. 계속 진행하려면 아래에서 가져올 파일을 선택하십시오.

C:\Users\시스템 보안 연구실\Desktop\2022 시스템 및 네트워크 보안 실습 자료\OPTEE.ova

전문가 모드(E)

다음(N)

취소

← 가상 시스템 가져오기

### 가상 시스템 설정

아래 목록은 가상 시스템 설명 파일에 나와 있는 가상 머신이며, 이를 VirtualBox로 가져왔을 때의 형태입니다. 보여져 있는 속성을 두 번 누르면 변경할 수도 있으며, 체크 상자를 사용해서 비활성화시킬 수도 있습니다.

#### 가상 시스템 1

이름	OPTEE
게스트 운영 체제 종류	Ubuntu (64-bit)
CPU	2
RAM	2048 MB
DVD	<input checked="" type="checkbox"/>
USB 컨트롤러	<input checked="" type="checkbox"/>
사운드 카드	<input checked="" type="checkbox"/> ICH AC97

You can modify the base folder which will host all the virtual machines. Home folders can also be individually (per virtual machine) modified.

C:\Users\시스템 보안 연구실\VirtualBox VMs

MAC Address Policy: Include only NAT network adapter MAC addresses

Additional Options: ☒ Import hard drives as VDI

가상 시스템이 서명되지 않았음

기본값 복원

가져오기

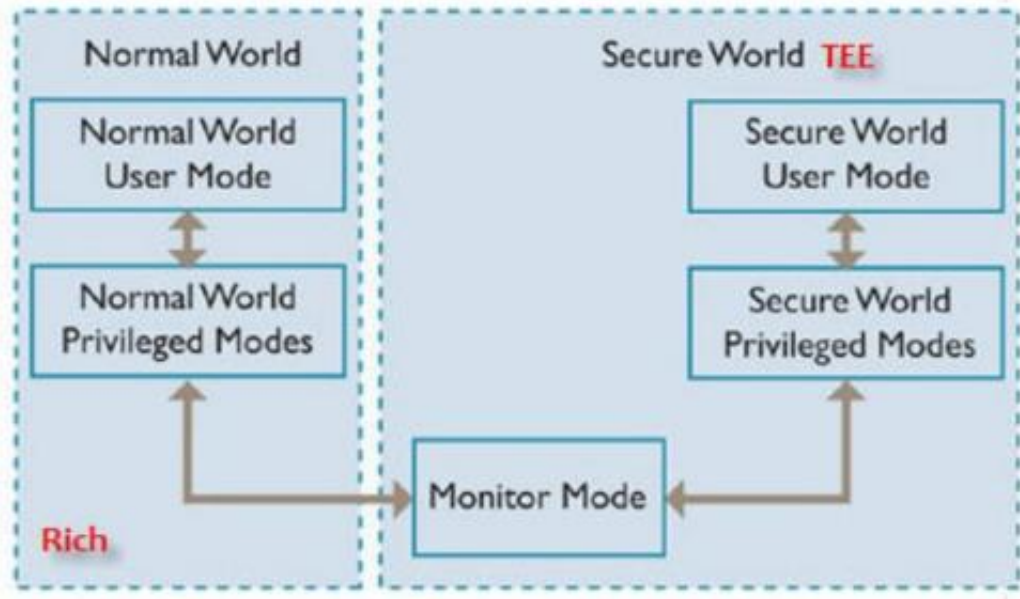
취소

# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 개념

### • OP-TEE(Open Portable Trusted Execution Environment)

- OP-TEE는 ARM 기반의 리눅스 오픈 소스를 지향하는 비영리 단체 Linaro에서 배포하는 TrustZone 기술이 적용된 TEE(Trusted Execution Environment)를 구현한 오픈소스



<ARM Trusted Model>

### • ARM TrustZone

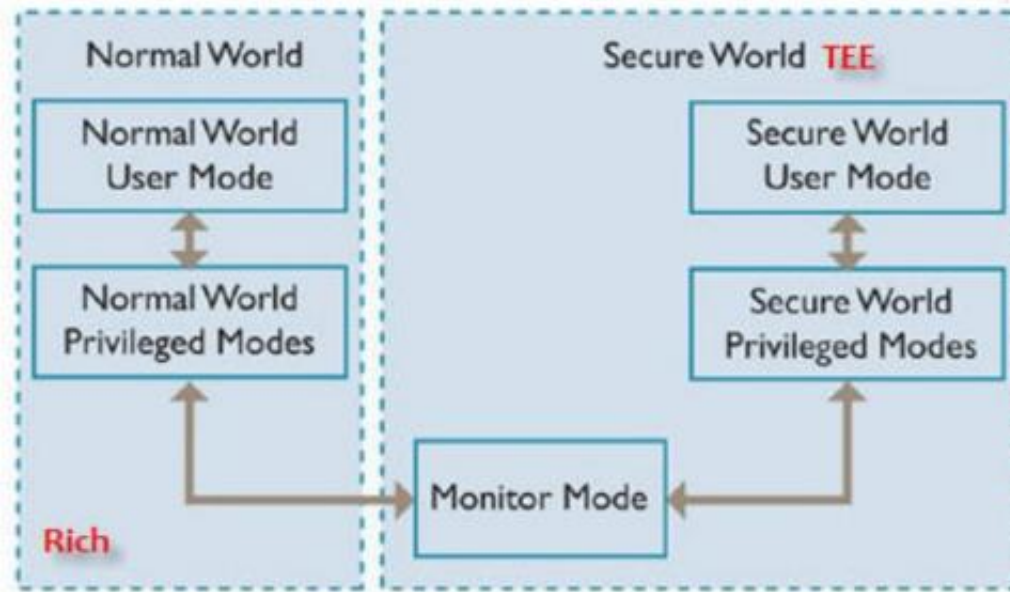
- 신뢰 실행 환경(TEE)
- Normal World & Secure World 두 개의 독립된 환경 제공
- ARM 아키텍처의 하드웨어 기반 보안 솔루션

# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 개념

### • ARM TrustZone

- 하나의 장치에 분리된 두 개의 환경을 제공하여 보안이 필요한 정보를 격리된 환경에서 안전하게 보호하는 기술
- 하드웨어적으로 분리된 REE와 TEE로 구성되어 있으며, 각 실행환경은 Normal/Secure World로 지칭
- 일반적인 실행환경을 REE라고 하며, TEE는 보안 기능을 보호하고 신뢰된 코드만을 실행하는 공간



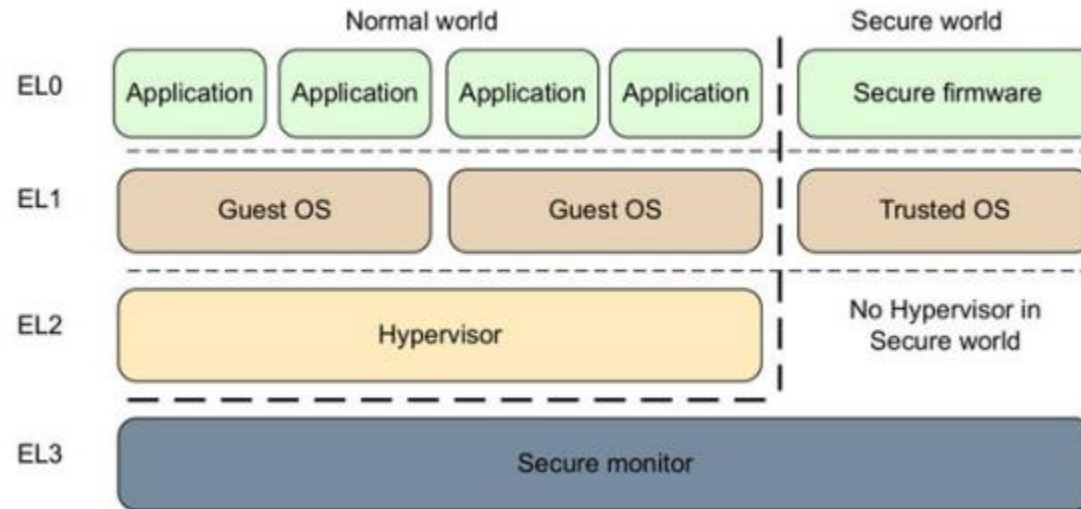
<ARM Trusted Model>

# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 개념

### • ARM TrustZone

- 각 world 간의 전환은 ARM 아키텍처에서 Application 계층(EL0) 및 OS 계층(EL1) 보다 높은 권한의 계층인 EL3에 구현되어 있는 Secure monitor를 통해서만 가능하기 때문에 Secure world에서는 보안이 필요한 중요한 로직의 안전한 실행을 보장



<ARM 아키텍처의 특권 계층 – ARMv8 64bit>

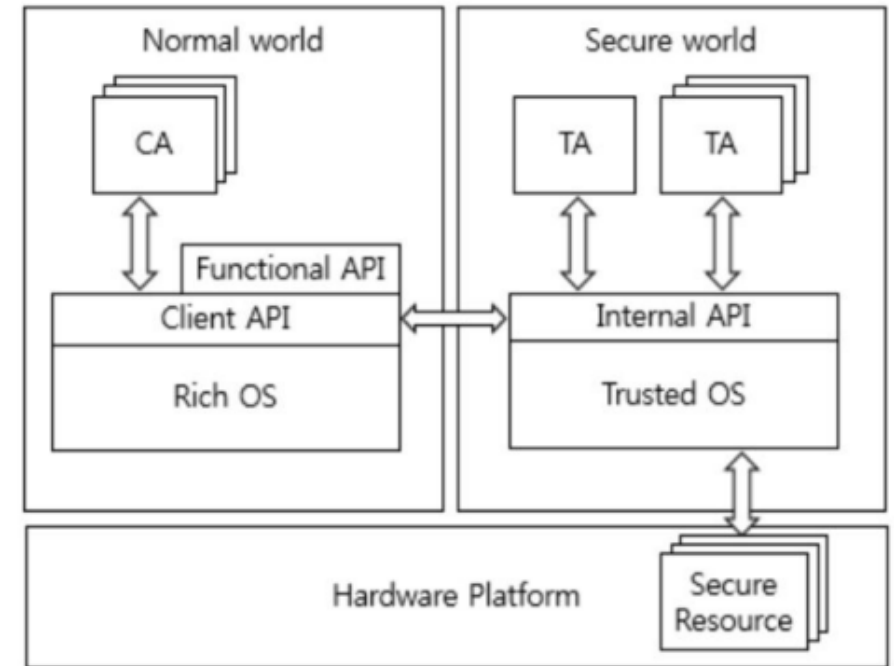
# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 개념

### • ARM TrustZone

- 어플리케이션은 CA와 TA로 구분되며 둘은 짝을 이루며 동작
  - CA(Client Application) = Normal world에서 동작하는 어플리케이션
  - TA(Trusted Application) = Secure world에서 동작하는 어플리케이션

- CA와 TA는 서로 독립된 실행환경에서 실행되므로 서로 직접적인 연결이 이루어지지 않지만, 각 TA는 ID(UUID)를 가지고 있기 때문에 이 ID에 대한 정보를 가지고 있는 CA 혹은 TA만 client/internal API를 통해 특정 TA 서비스를 요청할 수 있음



<ARM TrustZone software architecture>



# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 실행

- OP-TEE 경로로 이동

\$ cd /home/syssec/devel/optee

- optee/build 디렉토리 이동

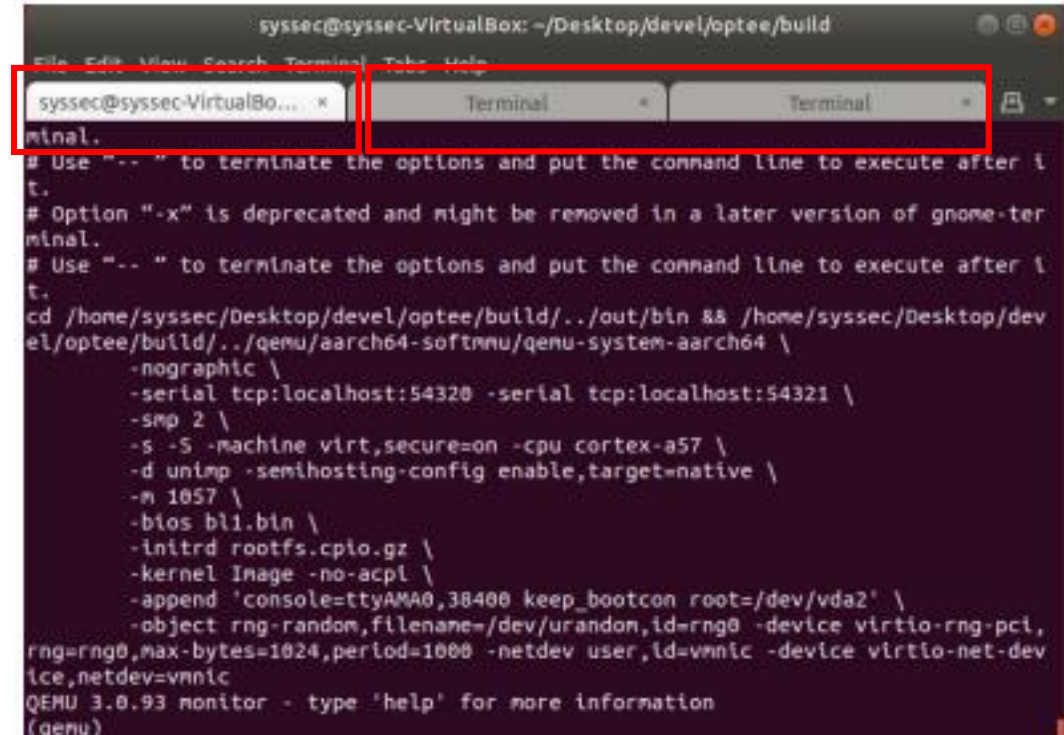
\$ cd build

- OP-TEE 실행하기

\$ make run

⇒ QEMU 콘솔과 두 개의 UART 콘솔 생성

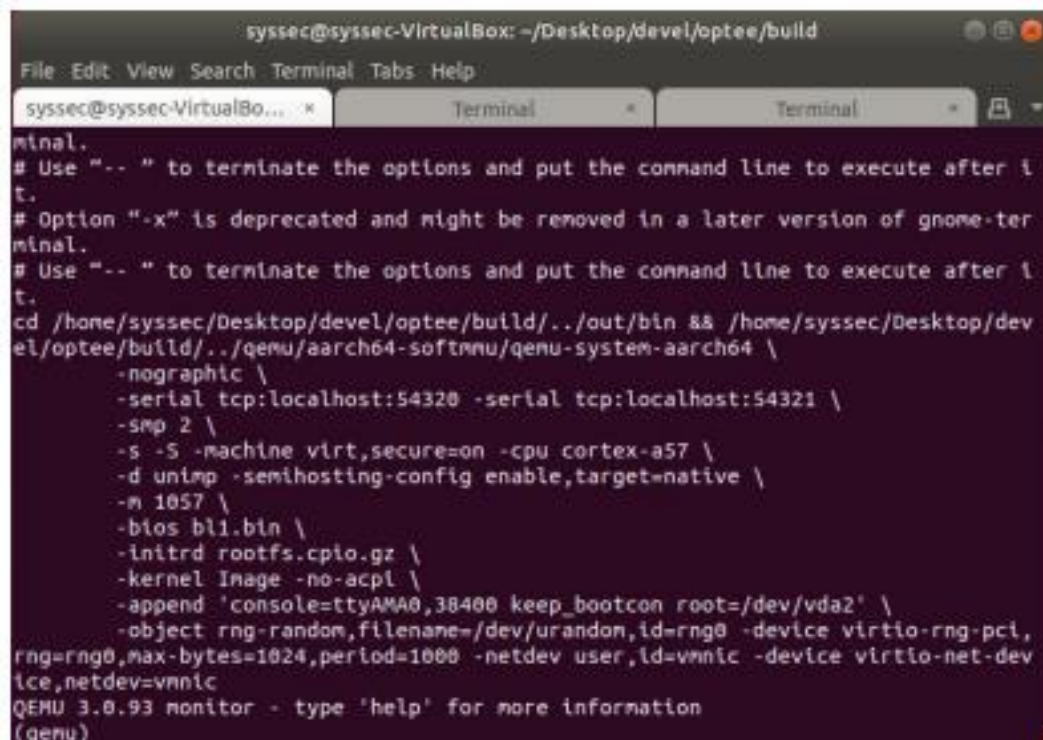
⇒ UART 콘솔 중 하나는 Normal world, 다른 하나는 Secure world



```
syssec@syssec-VirtualBox: ~/Desktop/devel/optee/build
File Edit View Search Terminal Tabs Help
syssec@syssec-VirtualBo...
minal.
# Use "--" to terminate the options and put the command line to execute after i
t.
# Option "-x" is deprecated and might be removed in a later version of gnome-ter
minal.
# Use "--" to terminate the options and put the command line to execute after i
t.
cd /hone/syssec/Desktop/devel/optee/build/../../out/bin && /home/syssec/Desktop/dev
el/optee/build/../../qemu/aarch64-softmmu/qemu-system-aarch64 \
  -nographic \
  -serial tcp:localhost:54320 -serial tcp:localhost:54321 \
  -smp 2 \
  -s -S -machine virt,secure=on -cpu cortex-a57 \
  -d unimp -semihosting-config enable,target=native \
  -m 1057 \
  -bios bl1.bin \
  -initrd rootfs.cpio.gz \
  -kernel Image -no-acpi \
  -append 'console=ttyAMA0,38400 keep_bootcon root=/dev/vda2' \
  -object rng-random,filename=/dev/urandom,id=rng0 -device virtio-rng-pci,
rng=rng0,max-bytes=1024,period=1000 -netdev user,id=vmmic -device virtio-net-dev
ice,netdev=vmmic
QEMU 3.0.93 monitor - type 'help' for more information
(qemu)
```

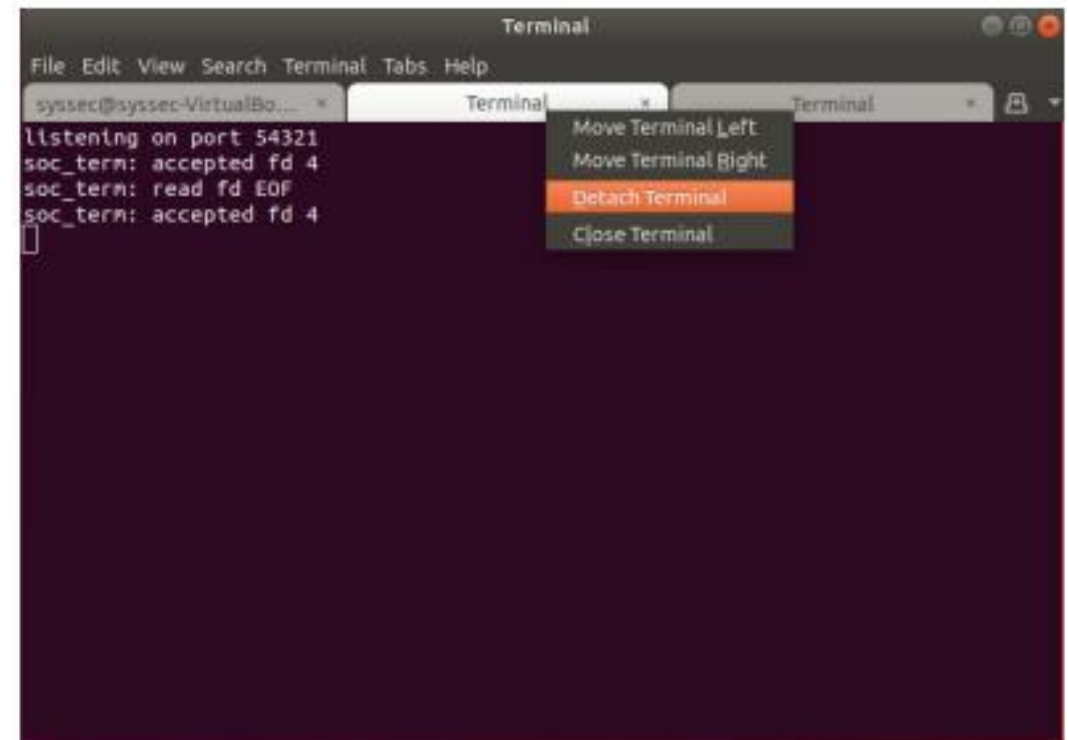
# OP-TEE 개인 텀프로젝트

- OP-TEE 실행



```
syssec@syssec-VirtualBox: ~/Desktop/dev/optee/build
File Edit View Search Terminal Tabs Help
syssec@syssec-VirtualBo... * Terminal * Terminal *
minal.
# Use "--" to terminate the options and put the command line to execute after i
t.
# Option "-x" is deprecated and might be removed in a later version of gnome-ter
minal.
# Use "--" to terminate the options and put the command line to execute after i
t.
cd /home/syssec/Desktop/dev/optee/build/./out/bin && /home/syssec/Desktop/dev
el/optee/build/./qemu/aarch64-softmmu/qemu-system-aarch64 \
  -nographic \
  -serial tcp:localhost:54320 -serial tcp:localhost:54321 \
  -smp 2 \
  -s -S -machine virt,secure=on -cpu cortex-a57 \
  -d unimp -semihosting-config enable,target=native \
  -m 1057 \
  -bios bl1.bin \
  -initrd rootfs.cpio.gz \
  -kernel Image -no-acpi \
  -append 'console=ttyAMA0,38400 keep_bootcon root=/dev/vda2' \
  -object rng-random,filename=/dev/urandom,id=rng0 -device virtio-rng-pci,
rng=rng0,max-bytes=1024,period=1000 -netdev user,id=vnnic -device virtio-net-dev
ice,netdev=vnnic
QEMU 3.0.93 monitor - type 'help' for more information
(qemu)
```

QEMU 콘솔



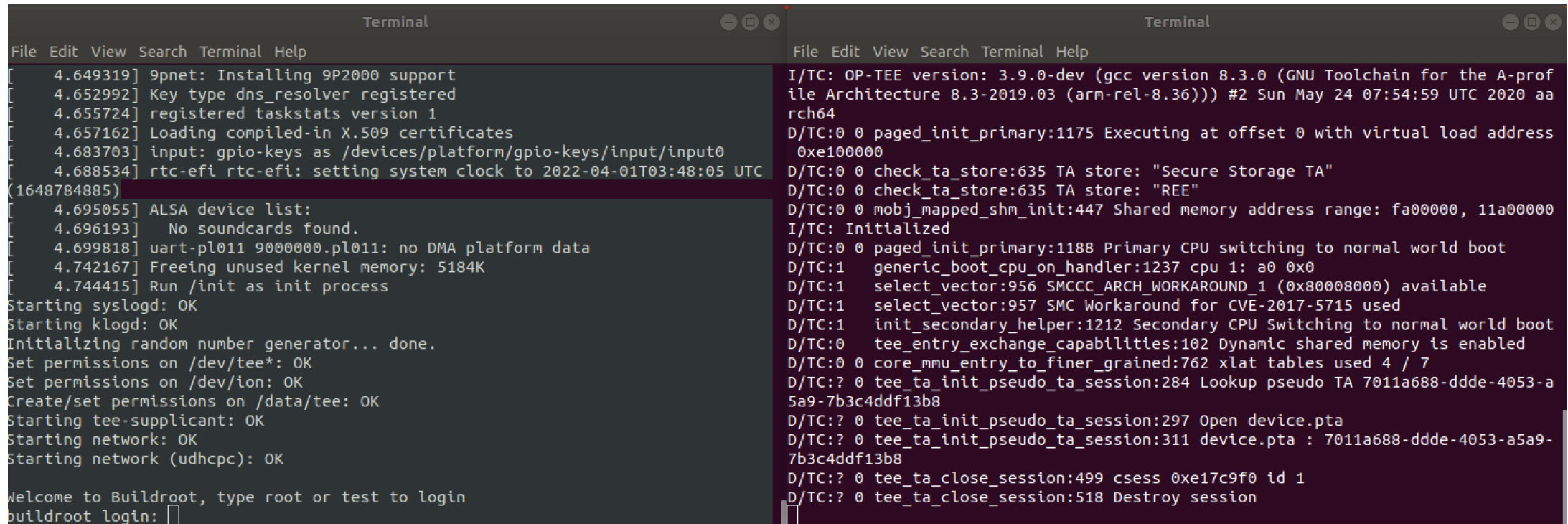
```
Terminal
File Edit View Search Terminal Tabs Help
syssec@syssec-VirtualBo... * Terminal * Terminal *
listening on port 54321
soc_term: accepted fd 4
soc_term: read fd EOF
soc_term: accepted fd 4
□
Move Terminal Left
Move Terminal Right
Detach Terminal
Close Terminal
```

편하게 보기 위해 터미널 분리

# OP-TEE 개인 텀프로젝트

## ■ OP-TEE 실행

- QEMU 콘솔 상에서 'c'를 입력하고 실행하여 두 world를 초기화  
(qemu) c
- Normal world에서 root로 로그인  
buildroot login: root



```
Terminal
File Edit View Search Terminal Help
[ 4.649319] 9pnet: Installing 9P2000 support
[ 4.652992] Key type dns_resolver registered
[ 4.655724] registered taskstats version 1
[ 4.657162] Loading compiled-in X.509 certificates
[ 4.683703] input: gpio-keys as /devices/platform/gpio-keys/input/input0
[ 4.688534] rtc-efi rtc-efi: setting system clock to 2022-04-01T03:48:05 UTC
(1648784885)
[ 4.695055] ALSA device list:
[ 4.696193]   No soundcards found.
[ 4.699818] uart-pl011 90000000.pl011: no DMA platform data
[ 4.742167] Freeing unused kernel memory: 5184K
[ 4.744415] Run /init as init process
Starting syslogd: OK
Starting klogd: OK
Initializing random number generator... done.
Set permissions on /dev/tee*: OK
Set permissions on /dev/ion: OK
Create/set permissions on /data/tee: OK
Starting tee-suplicant: OK
Starting network: OK
Starting network (udhcp): OK

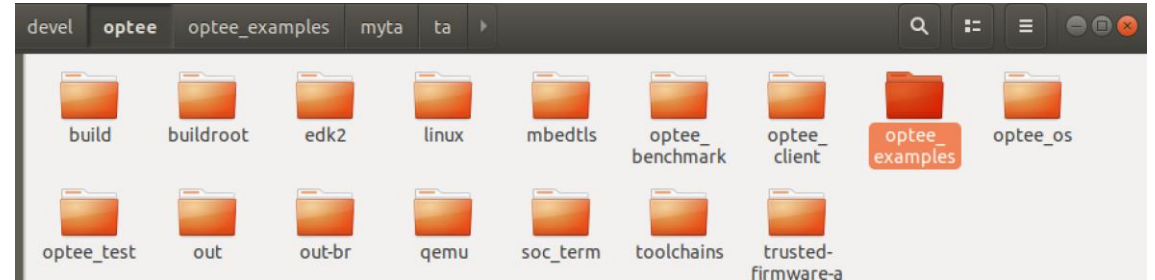
Welcome to Buildroot, type root or test to login
buildroot login:

Terminal
File Edit View Search Terminal Help
I/TC: OP-TEE version: 3.9.0-dev (gcc version 8.3.0 (GNU Toolchain for the A-profile Architecture 8.3-2019.03 (arm-rel-8.36))) #2 Sun May 24 07:54:59 UTC 2020 aa
rch64
D/TC:0 0 paged_init_primary:1175 Executing at offset 0 with virtual load address
0xe100000
D/TC:0 0 check_ta_store:635 TA store: "Secure Storage TA"
D/TC:0 0 check_ta_store:635 TA store: "REE"
D/TC:0 0 mobj_mapped_shm_init:447 Shared memory address range: fa00000, 11a00000
I/TC: Initialized
D/TC:0 0 paged_init_primary:1188 Primary CPU switching to normal world boot
D/TC:1 generic_boot_cpu_on_handler:1237 cpu 1: a0 0x0
D/TC:1 select_vector:956 SMCCC_ARCH_WORKAROUND_1 (0x80008000) available
D/TC:1 select_vector:957 SMC Workaround for CVE-2017-5715 used
D/TC:1 init_secondary_helper:1212 Secondary CPU Switching to normal world boot
D/TC:0 tee_entry_exchange_capabilities:102 Dynamic shared memory is enabled
D/TC:0 0 core_mmu_entry_to_finer_grained:762 xlat tables used 4 / 7
D/TC:? 0 tee_ta_init_pseudo_ta_session:284 Lookup pseudo TA 7011a688-ddde-4053-a
5a9-7b3c4ddf13b8
D/TC:? 0 tee_ta_init_pseudo_ta_session:297 Open device.pta
D/TC:? 0 tee_ta_init_pseudo_ta_session:311 device.pta : 7011a688-ddde-4053-a5a9-
7b3c4ddf13b8
D/TC:? 0 tee_ta_close_session:499 csess 0xe17c9f0 id 1
D/TC:? 0 tee_ta_close_session:518 Destroy session
```

# OP-TEE 개인 템프로젝트

## ■ OP-TEE 구성 요소(참고)

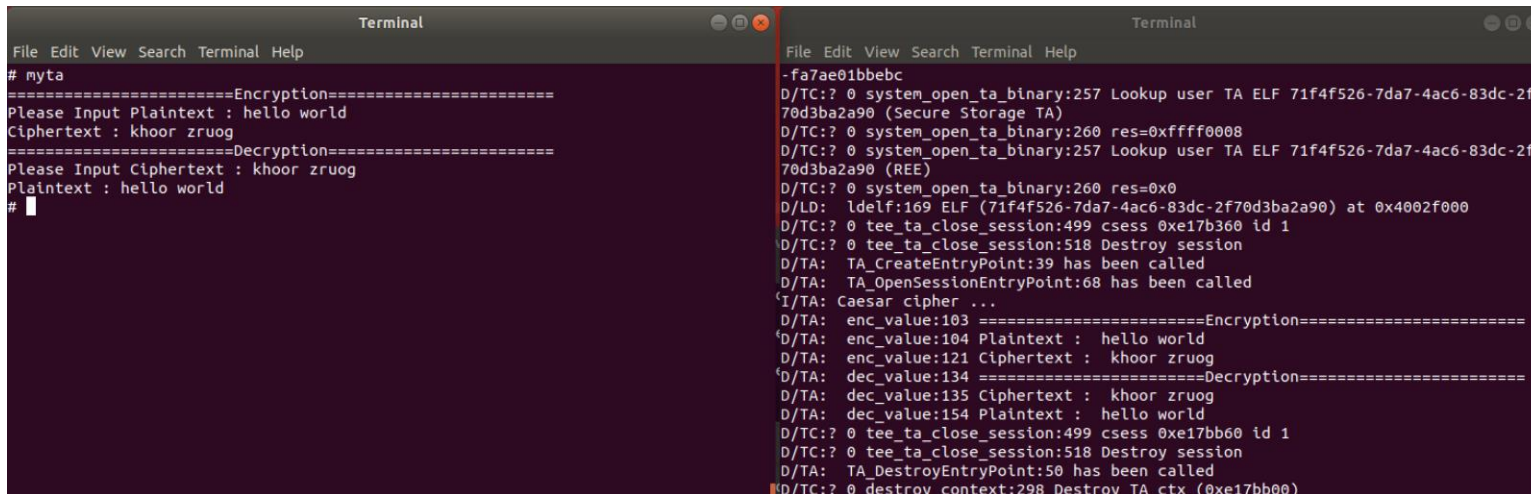
- build = OP-TEE 빌드용 스크립트
- buildroot = linux에서 사용하는 툴 빌드용 스크립트 모음
- linux = 리눅스
- optee\_client = REE와 TEE 간의 통신을 위한 라이브러리
- optee\_examples = 신뢰 실행 환경에서 사용할 수 있는 TA 예제
- optee\_os = TEE OS, 즉 TEE 내에서 동작되는 OS
- optee\_test = 다양한 기능들을 테스트하는 코드
- qemu = 에뮬레이터
- out, out-br = 코드들의 컴파일 결과
- toolchains = 컴파일하는 환경(REE)과 동작하는 환경(TEE)이 다르기 때문에 toolchains를 사용하여 크로스 컴파일
- trusted-firmware-a = secure monitor, REE와 TEE간의 전환은 가장 권한이 높은 EL3에 존재하는 secure monitor라는 소프트웨어에 의해 전환됨



# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 1. 프로젝트 주제

- 해당 실습에서는 OP-TEE 내에서 작동하는 어플리케이션을 작성해본다.
- TA를 구축하기 위해서는 Makefile, sub.mk, user\_ta\_header\_defines.h 등의 파일 또한 작성해야 하기 때문에 해당 실습에서는 제공되는 예제(hello\_world)의 소스코드를 수정하여 작성하도록 한다.
- 예제 코드 실행 확인(# optee\_examples\_hello\_world)
- 정답 예제 코드 실행 확인(# myta)
  - > 텀프로젝트 결과 구현(기능은 일부 다름) // 경로 - /home/syssec/devel/optee/optee\_examples/myta



```
File Edit View Search Terminal Help
# myta
=====Encryption=====
Please Input Plaintext : hello world
Ciphertext : khood zruog
=====Decryption=====
Please Input Ciphertext : khood zruog
Plaintext : hello world
#

-fa7ae01bbebc
D/TC:? 0 system_open_ta_binary:257 Lookup user TA ELF 71f4f526-7da7-4ac6-83dc-2f70d3ba2a90 (Secure Storage TA)
D/TC:? 0 system_open_ta_binary:260 res=0xffff0008
D/TC:? 0 system_open_ta_binary:257 Lookup user TA ELF 71f4f526-7da7-4ac6-83dc-2f70d3ba2a90 (REE)
D/TC:? 0 system_open_ta_binary:260 res=0x0
D/LD: ldelf:169 ELF (71f4f526-7da7-4ac6-83dc-2f70d3ba2a90) at 0x4002f000
D/TC:? 0 tee_ta_close_session:499 csess 0xe17b360 id 1
D/TC:? 0 tee_ta_close_session:518 Destroy session
D/TA: TA_CreateEntryPoint:39 has been called
D/TA: TA_OpenSessionEntryPoint:68 has been called
I/TA: Caesar cipher ...
D/TA: enc_value:103 =====Encryption=====
D/TA: enc_value:104 Plaintext : hello world
D/TA: enc_value:121 Ciphertext : khood zruog
D/TA: dec_value:134 =====Decryption=====
D/TA: dec_value:135 Ciphertext : khood zruog
D/TA: dec_value:154 Plaintext : hello world
D/TC:? 0 tee_ta_close_session:499 csess 0xe17bb00 id 1
D/TC:? 0 tee_ta_close_session:518 Destroy session
D/TA: TA_DestroyEntryPoint:50 has been called
D/TC:? 0 destroy_context:298 Destroy TA ctx (0xe17bb00)
```

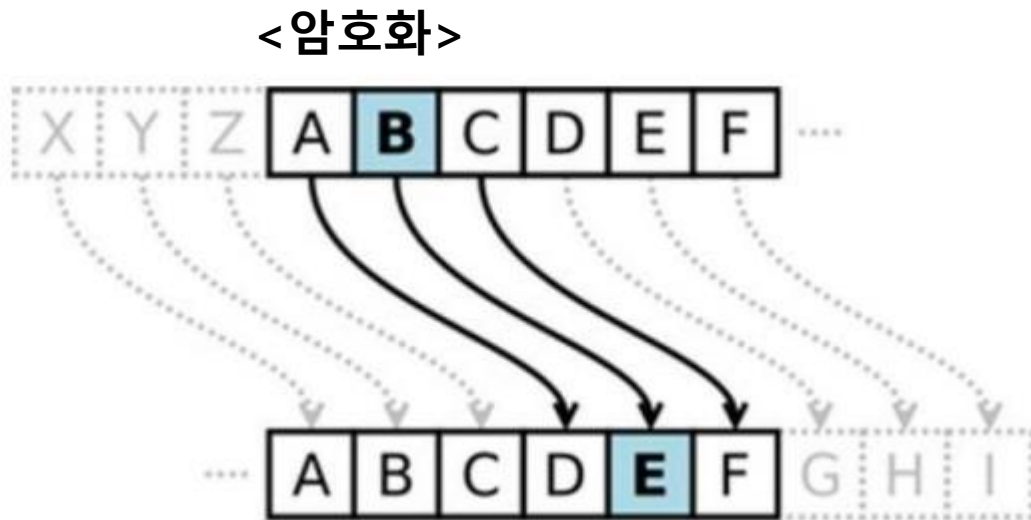
<어플리케이션 실행 화면>



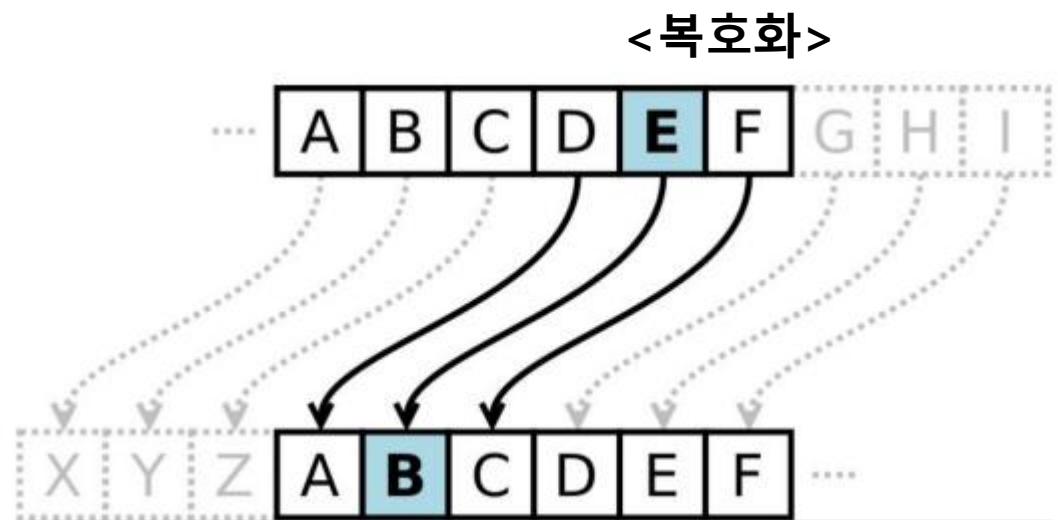
# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 1. 프로젝트 주제

- 시저 암호 (Caesar cipher)
  - 암호학에서 다루는 간단한 치환암호의 일종 (알파벳을 Key값 만큼 이동하여 치환)
  - 예를 들어 Key값이 3이라면



B를 우측으로 3칸 이동하면 E로 치환  
ex) 평문 "HELLO" -> 암호문은 "KHOOR"

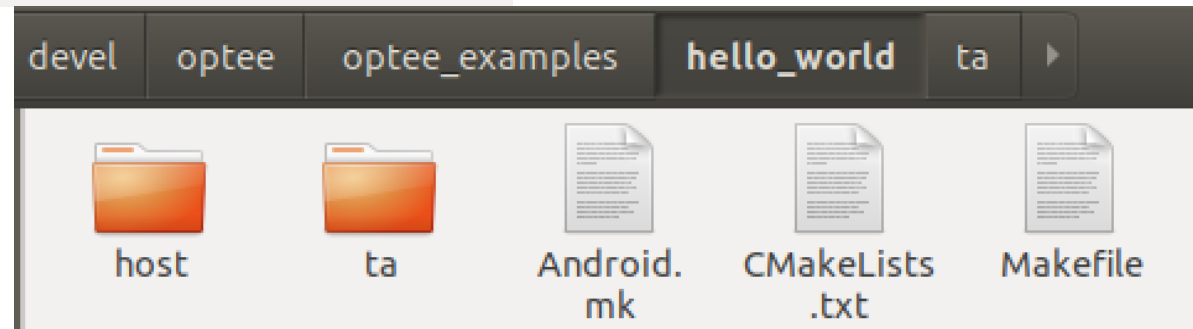
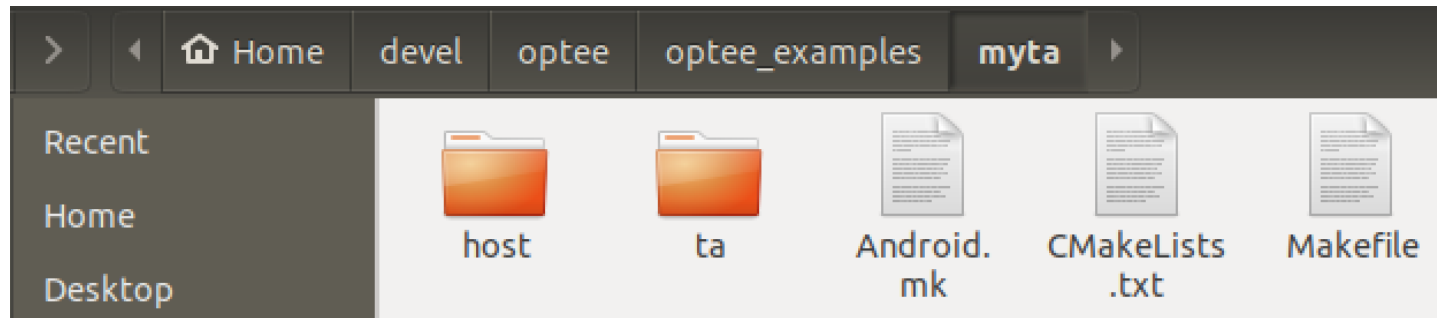


E를 좌측으로 3칸 이동하면 B로 치환  
ex) 암호문 "KHOOR" -> 평문은 "HELLO"

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 1. 프로젝트 주제

- Trusted Execution Environment(TEE)는 독립된 공간에서 보안이 필요한 로직 및 정보를 안전하게 실행 가능
- 실습 설명을 위한 소스코드는 /home/syssec/devel/optee/optee\_examples/myta에 위치
  - 실제 실습은 hello\_world 소스코드를 사용할 것
  - Client Application(CA)의 소스코드 "main.c"는 host 폴더에 위치
  - Trusted Application(TA)의 소스코드 "myta.c" 및 헤더파일 "myta.h"는 ta 폴더에 위치



# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 1. 프로젝트 주제 - 참고

- CA 및 TA에는 통신을 위해 다음과 같은 API 함수가 필수로 존재해야 하며, 각 함수는 짝을 이루며 호출  
**<CA>**

- |                           |   |
|---------------------------|---|
| 1) TEEC_InitializeContext | // TEE와 CA의 논리적인 연결인 Context 생성           |
| 2) TEEC_OpenSession       | // UUID로 특정된 TA와 CA를 연결하여 Session 생성      |
| 3) TEEC_InvokeCommand     | // Session으로 연결된 TA의 함수 또는 서비스 ID로 서비스 요청 |
| 4) TEEC_CloseSession      | // Session에 저장된 CA와 TA의 연결 해지             |
| 5) TEEC_FinalizeContext   | // Context에 저장된 논리적 연결 해지                 |

### **<TA>**

- |                               |   |
|-------------------------------|---|
| 1) TA_CreateEntryPoint        | // CA와 TA의 연결이 최초로 생성될 때 실행됨                  |
| 2) TA_OpenSessionEntryPoint   | // CA와 TA의 연결을 위해 TEEC_OpenSession과 짝을 이루어 실행 |
| 3) TA_InvokeCommandEntryPoint | // TA의 함수 또는 서비스 ID에 따라 해당 서비스를 제공하기 위함       |
| 4) TA_CloseSessionEntryPoint  | // CA와 TA의 연결 해지를 위해 실행                       |
| 5) TA_DestroyEntryPoint       | // CA와 TA의 연결이 완전히 종료될 때 실행                   |



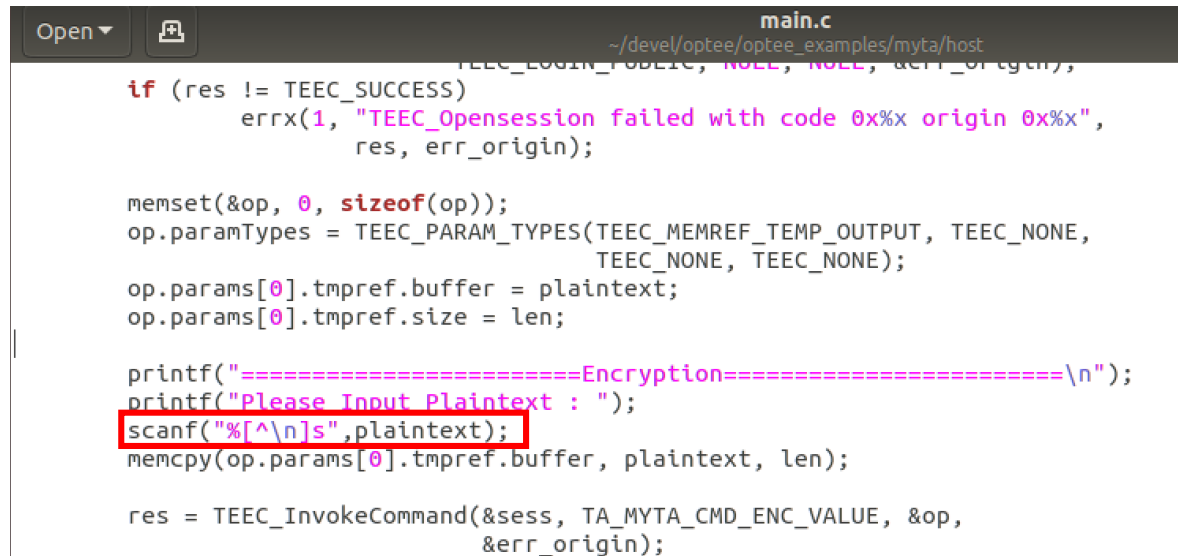
# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

- 암호화

- (1) CA에서 평문 텍스트 파일 읽기, TA 호출

- 실습을 위해 제공되는 myta 소스코드는 사용자의 입력을 받고 단순히 출력해주는 기능만 있음
    - 현재 구현되어 있는 것처럼 scanf()로 사용자의 입력을 받는 것이 아니라 파일을 입력으로 받고 결과를 파일로 돌려주는 방식을 구현해야 함



```
main.c
~/devel/optee/optee_examples/myta/host

TEEC_InvokeCommand(&sess, TA_MYTA_CMD_ENC_VALUE, &op, &err_origin);

if (res != TEEC_SUCCESS)
    errx(1, "TEEC Opensession failed with code 0x%x origin 0x%x",
        res, err_origin);

memset(&op, 0, sizeof(op));
op.paramTypes = TEEC_PARAM_TYPES(TEEC_MEMREF_TEMP_OUTPUT, TEEC_NONE,
                                TEEC_NONE, TEEC_NONE);
op.params[0].tmpref.buffer = plaintext;
op.params[0].tmpref.size = len;

printf("=====Encryption=====\\n");
printf("Please Input Plaintext : ");
scanf("%[^\\n]s", plaintext);
memcpy(op.params[0].tmpref.buffer, plaintext, len);

res = TEEC_InvokeCommand(&sess, TA_MYTA_CMD_ENC_VALUE, &op,
                        &err_origin);
```

.txt의 어떤 텍스트 파일을 읽어서 시저 암호로 TA쪽에서 암호화하고 CA가 TA쪽에서 받은 암호문을 REE 쪽에 저장하는 것이 목표

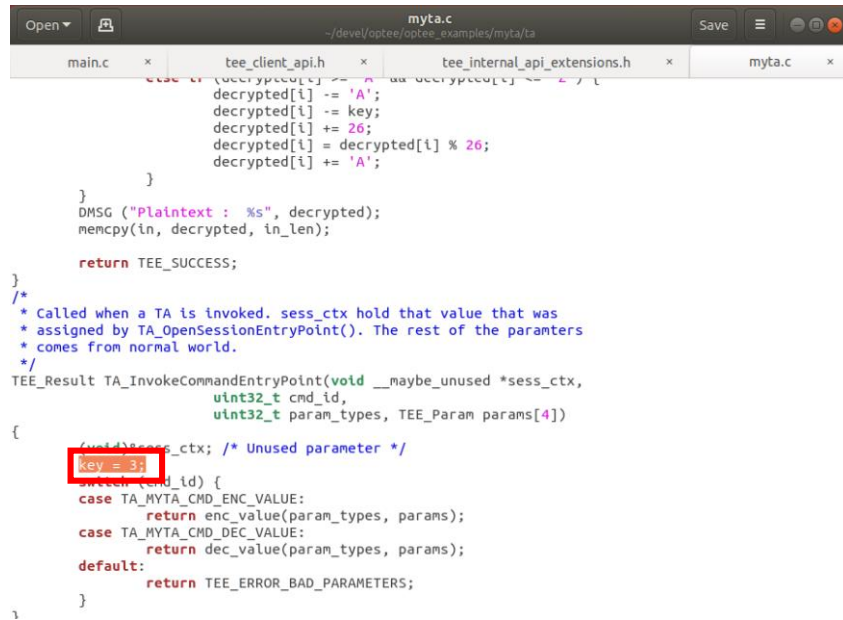
# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

### • 암호화

(2) TA에서 랜덤키 생성 – 평문을 암호화할 수 있는 Key

- 해당 프로그램에서 사용하는 모든 암호화 알고리즘은 시저암호
- 현재 코드에는 key = 3으로 키 값이 고정되어 있음
- 텀프로젝트에서는 랜덤으로 키를 생성하는 기능을 구현해야 함



```
Open ▾ myta.c
~/dev/optee/optee_examples/myta/ta
main.c x tee_client_api.h x tee_internal_api_extensions.h x myta.c x

    decrypted[i] -= 'A';
    decrypted[i] -= key;
    decrypted[i] += 26;
    decrypted[i] = decrypted[i] % 26;
    decrypted[i] += 'A';
}
DMSG ("Plaintext : %s", decrypted);
memcpy(in, decrypted, in_len);

return TEE_SUCCESS;
}
/*
 * Called when a TA is invoked. sess_ctx hold that value that was
 * assigned by TA_OpenSessionEntryPoint(). The rest of the params
 * comes from normal world.
 */
TEE_Result TA_InvokeCommandEntryPoint(void __maybe_unused *sess_ctx,
                                     uint32_t cmd_id,
                                     uint32_t param_types, TEE_Param params[4])
{
    (void)sess_ctx; /* Unused parameter */
    key = 3;
    switch (cmd_id) {
        case TA_MYTA_CMD_ENC_VALUE:
            return enc_value(param_types, params);
        case TA_MYTA_CMD_DEC_VALUE:
            return dec_value(param_types, params);
        default:
            return TEE_ERROR_BAD_PARAMETERS;
    }
}
```

- rand() 함수와 같은 일반적인 난수 생성 함수는 사용 할 수 없음
- TEE OS 내에 구현되어 있는 API 사용  
`void TEE_GenerateRandom(void *randomBuffer, uint32_t randomBufferLen);`
- 랜덤키를 통해 파일마다 다른 키를 사용하려는 이유는 하나의 파일 키가 노출되어도 다른 파일 키는 안전하게 하기 위함
- 유의할 점은 키의 길이가 알파벳의 개수인 26을 넘어가면 안됨

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

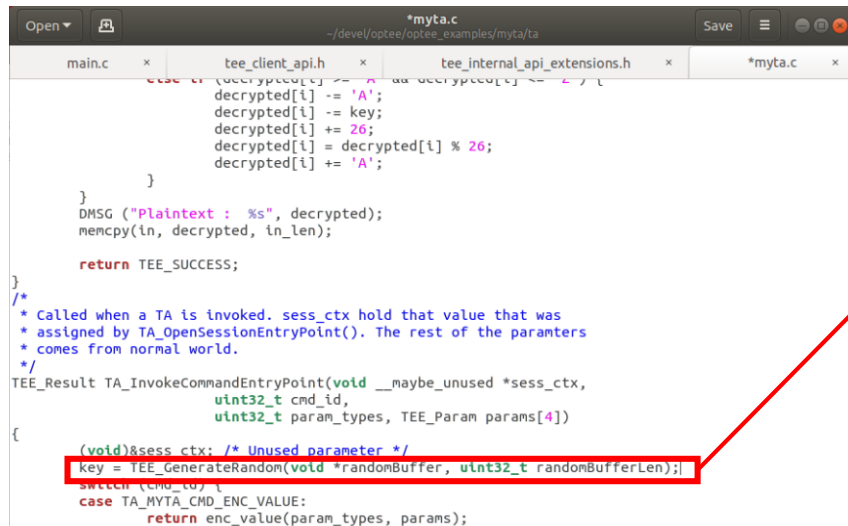
### • 암호화

(2) TA에서 랜덤키 생성 – 평문을 암호화할 수 있는 Key

- rand() 함수와 같은 일반적인 난수 생성 함수는 사용 할 수 없음
- TEE OS 내에 구현되어 있는 API 사용

```
void TEE_GenerateRandom(void *randomBuffer, uint32_t randomBufferLen);
```

- 랜덤키를 통해 파일마다 다른 키를 사용하는 이유는 하나의 파일키가 노출되어도 다른 파일키는 안전하게 하기 위함
- 유의할 점은 키의 길이가 알파벳의 개수인 26을 넘어가면 안됨



```
Open ▾ [icon] myta.c [icon] Save [icon] [icon] [icon]
~/dev/optee/optee_examples/myta/ta
main.c x tee_client_api.h x tee_internal_api_extensions.h x myta.c x

    case TA_MYTA_CMD_DEC_VALUE:
        decrypted[i] -= 'A';
        decrypted[i] -= key;
        decrypted[i] += 26;
        decrypted[i] = decrypted[i] % 26;
        decrypted[i] += 'A';
    }
}
MSG ("Plaintext : %s", decrypted);
memcpy(ln, decrypted, ln_len);

return TEE_SUCCESS;
}
/*
 * Called when a TA is invoked. sess_ctx hold that value that was
 * assigned by TA_OpenSessionEntryPoint(). The rest of the paramters
 * comes from normal world.
 */
TEE_Result TA_InvokeCommandEntryPoint(void __maybe_unused *sess_ctx,
                                     uint32_t cmd_id,
                                     uint32_t param_types, TEE_Param params[4])
{
    (void)sess_ctx; /* Unused parameter */
    key = TEE_GenerateRandom(void *randomBuffer, uint32_t randomBufferLen);
    switch (cmd_id) {
        case TA_MYTA_CMD_ENC_VALUE:
            return enc_value(param_types, params);
    }
}
```

해당 코드는 단순히 이해를 돕기 위한 예시

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

- 암호화

(3) 랜덤키로 평문 암호화, 랜덤키는 TA의 root키로 암호화

- 생성된 랜덤키는 TA의 root 키로 암호화

- TA의 root키는 TA 내에 정의되어 있는 고유의 키로, 1~25의 숫자 중 하나로 root키를 정의해두면 됨

(기존 코드의 key 정의 방식 참고 – 18pg)

⇒ Root키로 랜덤키를 암호화하면 결과값으로 암호화된 키(ex. enc\_key)를 얻을 수 있을 것

⇒ 이 키는 TA 고유의 루트 키로 암호화되어 있기 때문에 키(enc\_key)가 CA에 노출되어도 공격자는 그 키를 복호화하여 랜덤키를 얻어낼 수는 없음 (∵ TA 고유의 키이므로 TA에서만 복호화 가능)

(4) TA에서 CA로 암호문 + 암호화된 TA키(=랜덤키) 전달

(5) CA에서는 받은 암호문과 암호화된 키를 텍스트 파일 형태로 REE에 저장

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

- 복호화

- ⇒ 현재 구현된 코드는 main.c에서 scanf()로 암호문을 입력 받고 TA 호출, TA가 받아서 해당 암호문을 다시 복호화
- ⇒ 암호화와 마찬가지로 사용자 입력을 텍스트 파일로 받고 복호화된 결과물을 텍스트 파일로 저장해야 함

(1) CA에서 암호문과 암호화키 텍스트 파일을 읽고 TA로 복호화 요청

(2) TA에서 암호화된 키를 root키로 복호화 (복호화된 기존 랜덤키를 얻을 수 있음)

(3) 랜덤키로 암호문을 복호화 (기존 평문을 얻을 수 있음)

(4) TA에서 복호화된 평문과 랜덤키를 CA로 전달

(5) 전달받은 결과를 CA에서 텍스트 파일로 저장

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 기본 기능(90점)

- 실행 커맨드

(암호화)

- TEEencrypt -e [평문파일(.txt)]

- > -e (encryption)

- > 커맨드 실행 결과물 = ciphertext.txt + encryptedkey.txt

(텍스트 파일 이름은 예시 – 해당 파일이 어떤 파일인지 알기 쉽도록 작성하기만 하면 됨)

(이 두개의 결과물을 각각의 파일로 저장할지 하나의 파일에 같이 나오도록 할지는 자유)

(복호화)

- TEEencrypt -d [암호문 + 암호화키 파일]

(동일한 파일에 저장한 경우)

- TEEencrypt -d [암호문 파일][암호화키 파일]

(각각 다른 파일로 저장한 경우)

- > -d (decryption)

소스코드를 수정할 경우 OP-TEE 종료(QEMU 콘솔에서 Ctrl+C) 후 build 디렉토리에서 make run을 입력하여 OP-TEE를 재구동 할 것  
make run을 입력하면 수정사항이 자동으로 빌드 됨

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## (암호화) 프로그램 동작 순서 정리

- CA가 텍스트 파일(.txt)을 읽어와서 암호화 요청을 위해 TA 호출
- TA 쪽에서는 랜덤키를 하나 생성
  - 이 랜덤키는 평문을 암호화할 수 있는 키
- 랜덤키로 평문 암호화 + 랜덤키는 TA의 root키로 암호화
- TA에서 텍스트 파일의 평문을 암호화하고 암호화된 암호문과 TA키를 CA로 전달
- CA에서는 TA에게 받은 평문 암호문과 암호화된 TA의 랜덤키를 파일로 저장

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## (복호화) 프로그램 동작 순서 정리

- CA에서 암호문과 암호화키 텍스트 파일을 읽고 TA로 복호화 요청
- TA에서 암호화된 랜덤키를 root 키로 복호화
- 평문은 복호화된 랜덤키로 복호화
- TA에서 복호화된 결과를 CA로 전달
- 전달받은 결과를 CA에서 텍스트 파일로 저장



# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 추가 기능(10점)

- RSA 알고리즘 추가

- 구현된 예시 코드 사용

OP-TEE RSA example(github): [https://github.com/cezane/optee\\_rsa\\_example](https://github.com/cezane/optee_rsa_example)

~/devel/optee/optee\_examples\$ `git clone https://github.com/cezane/optee_rsa_example`

- 시저 암호와 마찬가지로 텍스트 파일 읽어오기 및 저장 기능 포함해야 함
  - RSA(비대칭키 알고리즘)의 경우 랜덤키 사용할 필요 없음

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 2. 기능 명세 – 추가 기능(10점)

- 실행 커맨드 – Caesar + RSA 둘 다 구현했을 경우 해당 실행 커맨드로 구현

(암호화)

- TEEencrypt -e [평문파일(.txt)][알고리즘]

-> 알고리즘 = Caesar or RSA

(즉 Caesar를 입력하면 시저 암호, RSA를 입력하면 RSA 암호가 사용되도록)

(복호화)

- 복호화는 구현하지 않습니다.

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 3. 평가 기준

- 최종 보고서: 40% (텀프로젝트 평가 기준)

<필수 내용>

- 주요 코드에 대한 설명: 50점

\* 해당 과제에 대해 모르는 사람에게 알려준다고 생각하고(=자세하게) 작성

- 실행 결과에 대한 스크린샷: 25점

- 프로그램 사용 방법 설명: 25점

- 추가 기능 구현 여부(O 또는 X로 표기)

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 3. 평가 기준

- 프로그램 실행 결과: 60% (텀프로젝트 평가 기준)

<평가 기준>

- 기본 기능: 90점

- 추가 기능: 10점

\* 기능이 정상 수행되지만 기능 명세와 다를 경우 부분 감점

\* 기능이 정상 수행되지만 비정상적인 오류 발생 시 부분 감점

\* 기능이 정상 수행되지 않을 경우 평가 기준 점수의 절반 부여

-> 코드가 정답이어도 프로그램 사용 방법에 대한 설명 부족 혹은 기타 실행 오류로 인해 조교의 PC에서 실행할 수 없을 경우

-> 따라서 코드 실행 여부 여러 번 확인하고 제출하기

# OP-TEE 개인 텀프로젝트 – 과제 공지 추가 설명

## 4. 제출물 및 기한


- 보고서 1부(사이버 캠퍼스)
  - 파일 형식: [SSS\_Term]학번\_이름.pdf
- 소스코드
  - TEEencrypt 디렉토리 안의 내용을 그대로 다 올려야 함(RSA example github 링크 참고)
  - [SSS\_Term\_Code]학번\_이름.zip
- 제출 기한
  - 2024.11.26(화) ~ 2024.12.17(화) 23:59 (약 3주간)
- 문의: timelessp@o.cnu.ac.kr
  - 메일 형식: [SSS][학번\_이름]제목

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

### • 예제 소스코드 hello\_world 사용

- hello\_world 디렉토리 복붙하고 디렉토리 및 파일 이름 변경(TEEencrypt)
- "main.c" "TEEencrypt\_ta.c" "TEEencrypt\_ta.h" 로 수정하여 진행할 것

```
Open ▾  *hello_world_ta.h
~/dev/tee/optee/examples/hello_world/ta/include

*
* 1. Redistributions of source code must retain the above copyright notice,
* this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright notice,
* this list of conditions and the following disclaimer in the documentation
* and/or other materials provided with the distribution.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
* AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
* LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
* SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*/
#ifndef TA_HELLO_WORLD_H
#define TA_HELLO_WORLD_H

/*
 * This UUID is generated with uuidgen
 * the ITU-T UUID generator at http://www.itu.int/ITU-T/asn1/uuid.html
 */
#define TA_HELLO_WORLD_UUID \
    { 0x8aaaf200, 0x2450, 0x11e4, \
      { 0xab, 0xe2, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x1b } }

/* The function IDs implemented in this TA */
#define TA_TEEencrypt_CMD_ENC_VALUE 0
#define TA_HELLO_WORLD_CMD_DEC_VALUE 1
#define TA_TEEencrypt_CMD_RANDOMKEY_GET 2
```

수정

추가

<헤더파일(hello\_world\_ta.h) 수정>

- TA\_HELLO\_WORLD를 TA\_TEEencrypt로 수정
  - > TA\_HELLO\_WORLD 전부 수정
- 빨간 박스의 #define은 함수 이름으로 취급 가능(ENC\_VALUE 등)
  - > 암호화, 복호화 등 구현할 기능으로 수정하거나 새로 추가
  - > 이름은 자유 (어떤 기능인지 알 수 있을만한)

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

### • UUID 추가

- 각 TA는 ID(UUID)를 가지고 있기 때문에 이 ID에 대한 정보를 가지고 있는 CA 혹은 TA만 특정 TA 서비스를 요청할 수 있음 (8pg 참고)
- 우리가 사용할 TA(TEEencrypt)에 대한 uuid 생성 필요

#### Note

Generating a fresh UUID with suitable formatting for the header file can be done using:

```
python -c "import uuid; u=uuid.uuid4(); print(u); \n = [' ', 0x'] * 11; \n[::2] = ['{:12x}'.format(u.node)[i:i + 2] for i in range(0, 12, 2)]; \nprint('\\n' + '#define TA_UUID\\n\\t{ ' + \n'0x{:08x}'.format(u.time_low) + ', ' + \n'0x{:04x}'.format(u.time_mid) + ', ' + \n'0x{:04x}'.format(u.time_hi_version) + ', \\ \\n\\n\\t{ ' + \n'0x{:02x}'.format(u.clock_seq_hi_variant) + ', ' + \n'0x{:02x}'.format(u.clock_seq_low) + ', ' + \n'0x' + ''.join(n) + ' } }'"
```

[https://optee.readthedocs.io/en/latest/building/trusted\\_applications.html](https://optee.readthedocs.io/en/latest/building/trusted_applications.html)

- OP-TEE Document의 Build and run -> Trusted Applications 참고  
-> 해당 문서의 uuid를 생성하는 파이썬 스크립트 사용

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

### • UUID 추가

- 각 TA는 ID(UUID)를 가지고 있기 때문에 이 ID에 대한 정보를 가지고 있는 CA 혹은 TA만 특정 TA 서비스를 요청할 수 있음 (8pg 참고)
- 우리가 사용할 TA(TEEencrypt)에 대한 uuid 생성 필요

```
syssec@syssec-VirtualBox: ~  
File Edit View Search Terminal Help  
syssec@syssec-VirtualBox:~$ python -c "import uuid; u=uuid.uuid4(); print(u); \  
> n = [' ', 0x'] * 11; \  
> n[::2] = ['{:12x}'.format(u.node)[i:i + 2] for i in range(0, 12, 2)]; \  
> print('\n' + '#define TA_UUID\n\t{ ' + \  
> '0x{:08x}'.format(u.time_low) + ', ' + \  
> '0x{:04x}'.format(u.time_mid) + ', ' + \  
> '0x{:04x}'.format(u.time_hi_version) + ', \\ \n\t\t{ ' + \  
> '0x{:02x}'.format(u.clock_seq_hi_variant) + ', ' + \  
> '0x{:02x}'.format(u.clock_seq_low) + ', ' + \  
> '0x' + ''.join(n) + ' } }')"  
741dfcaf-12d8-4c12-8dea-c0bc62ab3777  
  
#define TA_UUID  
{ 0x741dfcaf, 0x12d8, 0x4c12, \  
  
{ 0x8d, 0xea, 0xc0, 0xbc, 0x62, 0xab, 0x37, 0x77 } }  
syssec@syssec-VirtualBox:~$
```

- OP-TEE Document의 Build and run -> Trusted Applications 참고  
-> 해당 문서의 uuid를 생성하는 파이썬 스크립트 사용  
-> 복사 후 터미널 창에 입력하고 새로운 uuid 생성

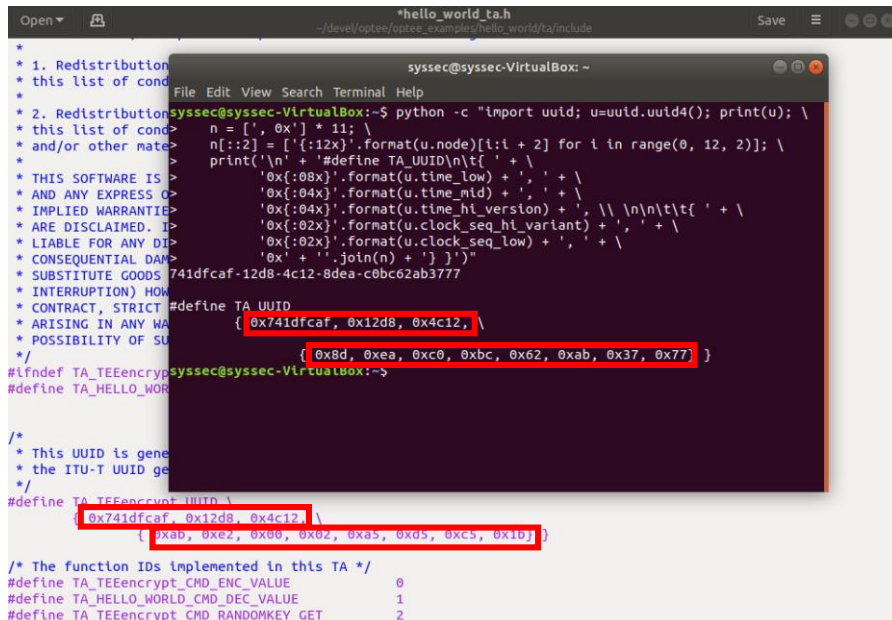


# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

### • UUID 추가

- 각 TA는 ID(UUID)를 가지고 있기 때문에 이 ID에 대한 정보를 가지고 있는 CA 혹은 TA만 특정 TA 서비스를 요청할 수 있음 (8pg 참고)
- 우리가 사용할 TA(TEEencrypt)에 대한 uuid 생성 필요



```
/*
 * 1. Redistribution
 * this list of conditions and/or other materials
 * 2. Redistribution
 * this list of conditions and/or other materials
 * AND ANY EXPRESS OR IMPLIED WARRANTIES,
 * ARE DISCLAIMED. IN NO EVENT SHALL THE
 * LIABILITY FOR ANY DIRECT, INDIRECT,
 * CONSEQUENTIAL DAMAGES OR THE
 * SUBSTITUTE GOODS AND SERVICES
 * INTERRUPTION) HOWEVER CAUSED AND
 * CONTRACT, STRICT LIABILITY, OR
 * ARISING IN ANY WAY OUT OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 */
#define TA_UUID { 0x741dfcaf, 0x12d8, 0x4c12, \
                 { 0x8d, 0xea, 0xc0, 0xbc, 0x62, 0xab, 0x37, 0x77 } }

/*
 * This UUID is generated by the ITU-T UUID
 */
#define TA_TEEencrypt_UUID { 0x741dfcaf, 0x12d8, 0x4c12, \
                             { 0xab, 0xe2, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x10 } }

/* The function IDs implemented in this TA */
#define TA_TEEencrypt_CMD_ENC_VALUE 0
#define TA_HELLO_WORLD_CMD_DEC_VALUE 1
#define TA_TEEencrypt_CMD_RANDOMKEY_GET 2
```

- OP-TEE Document의 Build and run -> Trusted Applications 참고
  - > 해당 문서의 uuid를 생성하는 파이썬 스크립트 사용
  - > 복사 후 터미널 창에 입력하고 새로운 uuid 생성
  - > UUID 부분만 복사하여 원래 코드 수정
  - > 헤더 파일 수정 완료

# OP-TEE 개인 텀프로젝트 - TA 작성법

## ■ TA 작성법

- ta 디렉토리의 TEEencrypt\_ta.c 수정 및 함수 추가
  - 아래 예시 참고 (이외에도 필요한 부분 수정)
  - 함수는 현재 hello\_world 예제 함수로 구현되어 있으므로 필요한 기능에 맞게 수정해야 함

```

* this list of conditions and the following
* and/or other materials provided with
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
* AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING THE IMPLIED WARRANTIES OF
* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
* EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT,
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
* SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
* THE POSSIBILITY OF SUCH DAMAGE.
*/

#ifndef TA_HELLO_WORLD_H
#define TA_HELLO_WORLD_H

/*
 * This UUID is generated with uuidgen
 * the ITU-T UUID generator at http://www.itu-t.org/asn1-x/uuid/
 */
#define TA_HELLO_WORLD_UUID \
    { 0x8aaaf200, 0x2450, 0x11e4, \
      { 0xab, 0xe2, 0x00, 0x02 } }

/* The function IDs implemented in this module */
#define TA_TEEencrypt_CMD_ENC_VALUE
#define TA_TEEencrypt_CMD_DEC_VALUE
#define TA_TEEencrypt_CMD_RANDOMKEY_GET
#define TA_TEEencrypt_CMD_RANDOMKEY_SET

TEEC_Result TA_InvokeCommandEntryPoint(void *session_ctx,
    uint32_t cmd_id,
    uint32_t param_types, TEE_Param params[4])
{
    (void)session_ctx; /* Unused parameter */

    switch (cmd_id) {
        case TA_TEEencrypt_CMD_ENC_VALUE:
            return inc_value(param_types, params);
        case TA_HELLO_WORLD_CMD_DEC_VALUE:
            return dec_value(param_types, params);
        case TA_TEEencrypt_CMD_RANDOMKEY_GET:
            return dec_value(param_types, params);
        default:
            return TEE_ERROR_BAD_PARAMETERS;
    }
}

```

```

* assigned by TA_OpenSessionEntryPoint().
*/
void TA_CloseSessionEntryPoint(void __maybe_unused *sess_ctx)
{
    (void)&sess_ctx; /* Unused parameter */
    IMSG("Goodbye!\n");
}

static TEE_Result inc_value(uint32_t param_types,
    TEE_Param params[4])
{
    uint32_t exp_param_types = TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_INOUT,
        TEE_PARAM_TYPE_NONE,
        TEE_PARAM_TYPE_NONE,
        TEE_PARAM_TYPE_NONE);

static TEE_Result enc_value(uint32_t param_types,
    TEE_Param params[4])
{
    uint32_t exp_param_types = TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_INOUT,
        TEE_PARAM_TYPE_NONE,
        TEE_PARAM_TYPE_NONE,
        TEE_PARAM_TYPE_NONE);
}

```

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- host 디렉토리의 main.c 수정
  - main 함수 내의 모든 주석 및 아래 사진의 블록 부분 삭제

```
        &err_origin);
if (res != TEEC_SUCCESS)
    errx(1, "TEEC_InvokeCommand failed with code 0x%x origin 0x%x",
        res, err_origin);
printf("TA incremented value to %d\n", op.params[0].value.a);

/*
 * We're done with the TA, close the session and
 * destroy the context.
 *
 * The TA will print "Goodbye!" in the log when the
 * session is closed.
 */

TEEC_CloseSession(&sess);

TEEC_FinalizeContext(&ctx);

return 0;
}
```

```
/* Initialize a context connecting us to the TEE */
```

```
res = TEEC_InitializeContext(NULL, &ctx);
if (res != TEEC_SUCCESS)
    errx(1, "TEEC_InitializeContext failed with code 0x%x", res);
```

```
/*
 * Open a session to the "hello world" TA, the TA will print "hello
 * world!" in the log when the session is created.
 */
```

```
res = TEEC_OpenSession(&ctx, &sess, &uuid,
    TEEC_LOGIN_PUBLIC, NULL, NULL, &err_origin);
```

```
/*
 * Open a session to the "hello world" TA, the TA will print "hello
 * world!" in the log when the session is created.
 */
res = TEEC_OpenSession(&ctx, &sess, &uuid,
    TEEC_LOGIN_PUBLIC, NULL, NULL, &err_origin);
if (res != TEEC_SUCCESS)
    errx(1, "TEEC_OpenSession failed with code 0x%x origin 0x%x",
        res, err_origin);

/*
 * Execute a function in the TA by invoking it, in this case
 * we're incrementing a number.
 *
 * The value of command ID part and how the parameters are
 * interpreted is part of the interface provided by the TA.
 */

/* Clear the TEEC Operation struct */
memset(&op, 0, sizeof(op));

/*
 * Prepare the argument. Pass a value in the first parameter,
 * the remaining three parameters are unused.
 */
op.paramTypes = TEEC_PARAM_TYPES(TEEC_VALUE_INOUT, TEEC_NONE,
    TEEC_NONE, TEEC_NONE);
op.params[0].value.a = 42;

/*
 * TA_HELLO_WORLD_CMD_INC_VALUE is the actual function in the TA to be
 * called.
 */
```

이 부분은 42pg 확인 후 삭제

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- host 디렉토리의 main.c 수정
  - main 함수에 호출할 서비스가 인자 값으로 들어가면 TA 쪽에서 그에 맞는 함수 호출
  - 아래 참고 (TEEC\_InvokeCommand())

```
int main(void)
{
    TEEC_Result res;
    TEEC_Context ctx;
    TEEC_Session sess;
    TEEC_Operation op;
    TEEC_UUID uuid = TA_TEEencrypt_UUID;
    uint32_t err_origin;

    res = TEEC_InitializeContext(NULL, &ctx);

    res = TEEC_OpenSession(&ctx, &sess, &uuid,
                          TEEC_LOGIN_PUBLIC, NULL, NULL, &err_origin);

    memset(&op, 0, sizeof(op));

    op.paramTypes = TEEC_PARAM_TYPES(TEEC_VALUE_INOUT, TEEC_NONE,
                                     TEEC_NONE, TEEC_NONE);
    op.params[0].value.a = 42;


    res = TEEC_InvokeCommand(&sess, TA_TEEencrypt_CMD_ENC_VALUE, &op,
                             &err_origin);
    res = TEEC_InvokeCommand(&sess, TA_TEEencrypt_CMD_DEC_VALUE, &op,
                             &err_origin);
    res = TEEC_InvokeCommand(&sess, TA_TEEencrypt_CMD_RANDOMKEY_GET, &op,
                             &err_origin);
    res = TEEC_InvokeCommand(&sess, TA_TEEencrypt_CMD_RANDOMKEY_ENC, &op,
                             &err_origin);

    TEEC_CloseSession(&sess);
}
```

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- host 디렉토리의 Makefile 수정
  - 실행 커맨드가 TEEencrypt -e [파일] 형식이므로 makefile의 BINARY = TEEencrypt로 수정



The image shows a text editor window titled '\*Makefile' with the path '~/dev/optee/optee\_examples/TEEencrypt/host'. The file contains a Makefile for building a Trusted Application (TA). The 'BINARY' variable is highlighted with a red box and set to 'TEEencrypt'. The Makefile includes standard compiler variables (CC, LD, AR, NM, OBJCOPY, OBJDUMP, READELF), object files (OBJS = main.o), compiler flags (CFLAGS += -Wall -I../ta/include -I\$(TEEC\_EXPORT)/include -I./include), linker flags (LDADD += -lteec -L\$(TEEC\_EXPORT)/lib), and the final build rules for the 'all' target.

```
Open [icon] *Makefile ~/dev/optee/optee_examples/TEEencrypt/host Save

CC      ?= $(CROSS_COMPILE)gcc
LD       ?= $(CROSS_COMPILE)ld
AR       ?= $(CROSS_COMPILE)ar
NM       ?= $(CROSS_COMPILE)nm
OBJCOPY  ?= $(CROSS_COMPILE)objcopy
OBJDUMP  ?= $(CROSS_COMPILE)objdump
READELF  ?= $(CROSS_COMPILE)readelf

OBJS = main.o

CFLAGS += -Wall -I../ta/include -I$(TEEC_EXPORT)/include -I./include
#Add/link other required libraries here
LDADD += -lteec -L$(TEEC_EXPORT)/lib

BINARY = TEEencrypt

.PHONY: all
all: $(BINARY)

$(BINARY): $(OBJS)
        $(CC) -o $@ $< $(LDADD)
```

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- ta 디렉토리의 Android.mk, Makefile 수정
  - 새로 생성한 UUID로 수정

```
Android.mk
~/devel/optee/optee_examples/TEEencrypt/ta

LOCAL_PATH := $(call my-dir)

local_module := 8aaaf200-2450-11e4-abe2-0002a5d5c51b.ta
include $(BUILD_OPTEE_MK)

Makefile
~/devel/optee/optee_examples/TEEencrypt/ta

CFG_TEE_TA_LOG_LEVEL ?= 4
CPPFLAGS += -DCFG_TEE_TA_LOG_LEVEL=$(CFG_TEE_TA_LOG_LEVEL)

# The UUID for the Trusted Application
BINARY_8aaaf200-2450-11e4-abe2-0002a5d5c51b

-include $(TA_DEV_KIT_DIR)/mk/ta_dev_kit.mk

ifeq ($(wildcard $(TA_DEV_KIT_DIR)/mk/ta_dev_kit.mk), )
clean:
    @echo 'Note: $(TA_DEV_KIT_DIR)/mk/ta_dev_kit.mk not found, cannot clean TA'
    @echo 'Note: TA_DEV_KIT_DIR=$(TA_DEV_KIT_DIR)'
endif
```

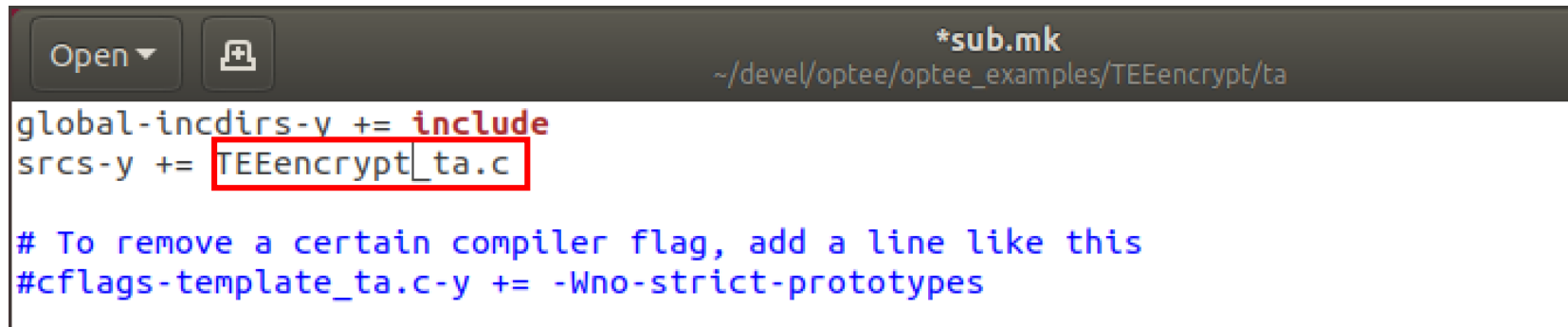
```
syssec@syssec-VirtualBox: ~
File Edit View Search Terminal Help
syssec@syssec-VirtualBox:~$ python -c "import uuid; u=uuid.uuid4(); print(u); \
> n = [' ', 0x'] * 11; \
> n[::2] = ['{:12x}'.format(u.node)[l:l + 2] for l in range(0, 12, 2)]; \
> print('\n' + '#define TA_UUID\n\t{ ' + \
> '0x{:08x}'.format(u.time_low) + ', ' + \
> '0x{:04x}'.format(u.time_mid) + ', ' + \
> '0x{:04x}'.format(u.time_hi_version) + ', || \n\n\t\t{ ' + \
> '0x{:02x}'.format(u.clock_seq_hi_variant) + ', ' + \
> '0x{:02x}'.format(u.clock_seq_low) + ', ' + \
> '0x' + ' ' * 10 + '}'"
741dfcaf-12d8-4c12-8dea-c0bc62ab3777

#define TA_UUID
{ 0x741dfcaf, 0x12d8, 0x4c12, \
    { 0x8d, 0xea, 0xc0, 0xbc, 0x62, 0xab, 0x37, 0x77} }
```

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ▪ TA 작성법

- ta 디렉토리의 sub.mk 수정
  - TA가 어떤 스크립트를 make할 것인지에 대한 정보가 담겨있으므로 TEEencrypt\_ta.c로 수정



```
*sub.mk
~/devel/optee/optee_examples/TEEencrypt/ta

global-incdirs-y += include
srcs-y += TEEencrypt_ta.c

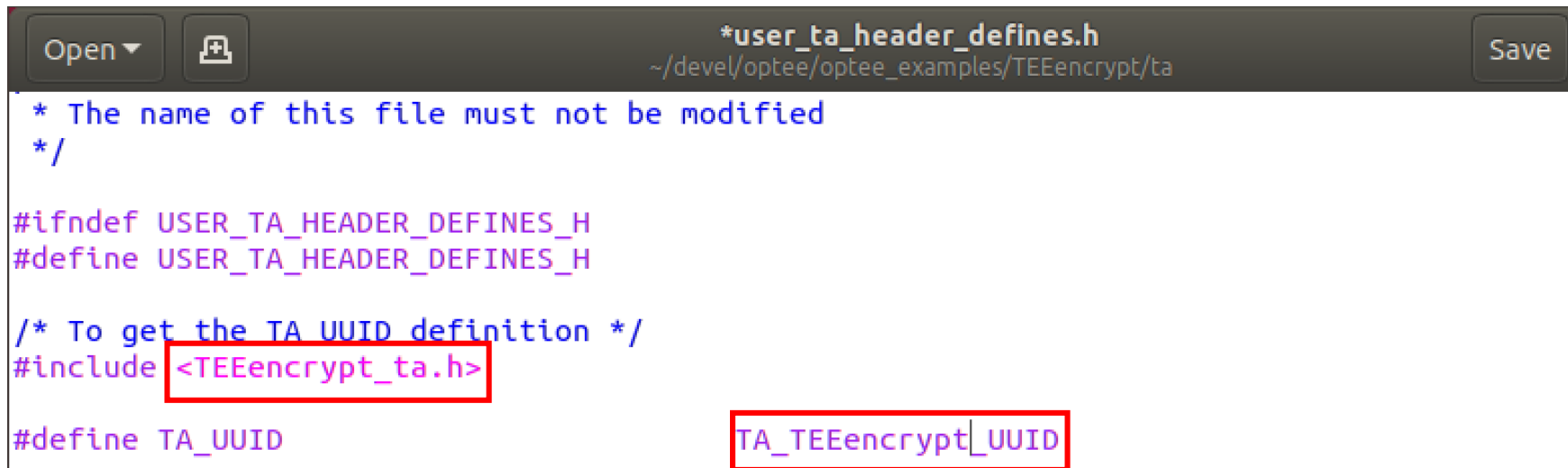
# To remove a certain compiler flag, add a line like this
#cflags-template_ta.c-y += -Wno-strict-prototypes
```



# OP-TEE 개인 텀프로젝트 – TA 작성법

- TA 작성법

- ta 디렉토리의 user\_ta\_header\_defines.h 수정
  - 아래 참고



```
*user_ta_header_defines.h
~/devel/optee/optee_examples/TEEencrypt/ta

* The name of this file must not be modified
*/

#ifndef USER_TA_HEADER_DEFINES_H
#define USER_TA_HEADER_DEFINES_H

/* To get the TA UUID definition */
#include <TEEencrypt_ta.h>

#define TA_UUID TA_TEEencrypt_UUID
```



# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- TEEencrypt 디렉토리의 Android.mk, CMakeLists.txt 수정
  - 아래 참고

```
##### optee-hello-world #####
LOCAL_PATH := $(call my-dir)

OPTEE_CLIENT_EXPORT = $(LOCAL_PATH)/../../optee_client/out/export

include $(CLEAR_VARS)
LOCAL_CFLAGS += -DANDROID_BUILD
LOCAL_CFLAGS += -Wall

LOCAL_SRC_FILES += host/main.c

LOCAL_C_INCLUDES := $(LOCAL_PATH)/ta/include \
    $(OPTEE_CLIENT_EXPORT)/include \

LOCAL_SHARED_LIBRARIES := libteec
LOCAL_MODULE := TEEencrypt
LOCAL_VENDOR_MODULE := true
LOCAL_MODULE_TAGS := optional
include $(BUILD_EXECUTABLE)

include $(LOCAL_PATH)/ta/Android.mk
```

```
*CMakeLists.txt
project (TEEencrypt C)

set (SRC host/main.c)

add_executable (${PROJECT_NAME} ${SRC})

target_include_directories(${PROJECT_NAME}
    PRIVATE ta/include
    PRIVATE include)

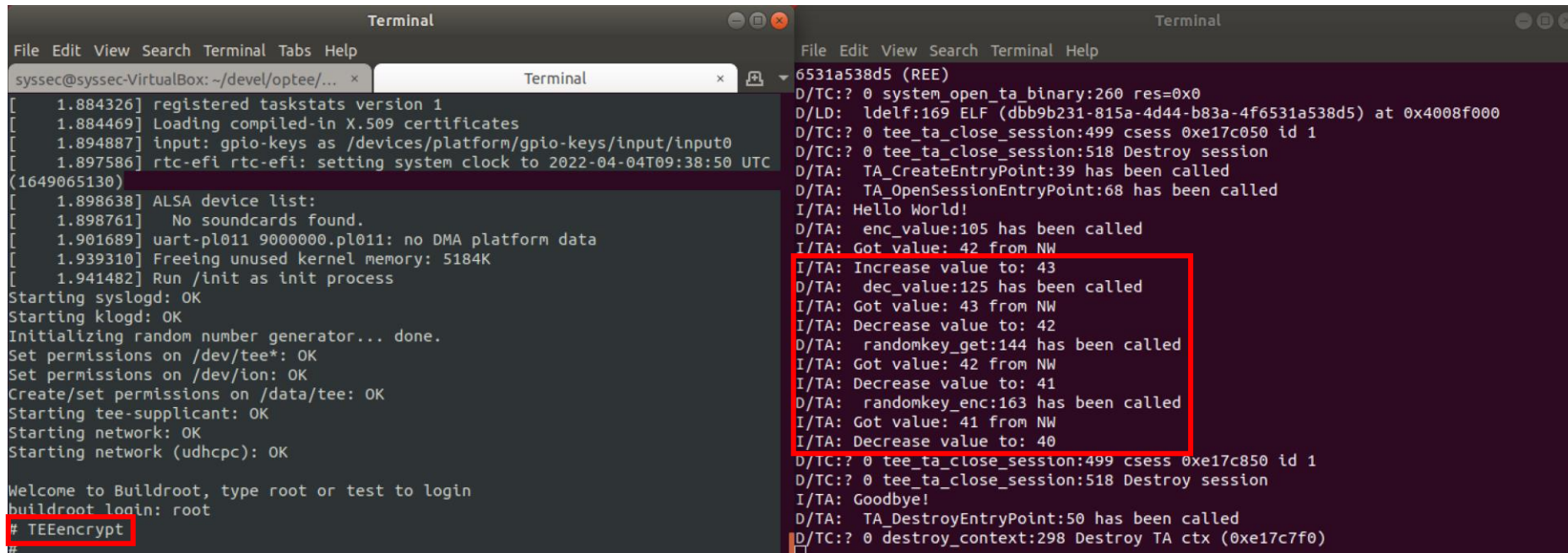
target_link_libraries (${PROJECT_NAME} PRIVATE teec)

install (TARGETS ${PROJECT_NAME} DESTINATION ${CMAKE_INSTALL_BINDIR})
```

# OP-TEE 개인 텀프로젝트 – TA 작성법

## ■ TA 작성법

- optee/build 디렉토리에서 make run 입력하여 수정 사항 빌드
- (qemu) c => Normal world에서 buildroot login: root
- # TEEencrypt 입력하여 실행해보기
  - > 아직 함수 내용을 수정하지 않았기 때문에 기존의 함수가 4번 호출(호출 횟수는 구현한대로 작동)



```
syssec@syssec-VirtualBox: ~/devel/optee/... x Terminal
[ 1.884326] registered taskstats version 1
[ 1.884469] Loading compiled-in X.509 certificates
[ 1.894887] input: gpio-keys as /devices/platform/gpio-keys/input/input0
[ 1.897586] rtc-efl rtc-efl: setting system clock to 2022-04-04T09:38:50 UTC
(1649065130)
[ 1.898638] ALSA device list:
[ 1.898761] No soundcards found.
[ 1.901689] uart-pl011 9000000.pl011: no DMA platform data
[ 1.939310] Freeing unused kernel memory: 5184K
[ 1.941482] Run /init as init process
Starting syslogd: OK
Starting klogd: OK
Initializing random number generator... done.
Set permissions on /dev/tee*: OK
Set permissions on /dev/ion: OK
Create/set permissions on /data/tee: OK
Starting tee-supplciant: OK
Starting network: OK
Starting network (udhcpc): OK

Welcome to Buildroot, type root or test to login
buildroot login: root
# TEEencrypt
#
```

```
6531a538d5 (REE)
D/TC:? 0 system_open_ta_binary:260 res=0x0
D/LD: ldelf:169 ELF (dbb9b231-815a-4d44-b83a-4f6531a538d5) at 0x4008f000
D/TC:? 0 tee_ta_close_session:499 csess 0xe17c050 id 1
D/TC:? 0 tee_ta_close_session:518 Destroy session
D/TA: TA_CreateEntryPoint:39 has been called
D/TA: TA_OpenSessionEntryPoint:68 has been called
I/TA: Hello World!
D/TA: enc_value:105 has been called
I/TA: Got value: 42 from NW
I/TA: Increase value to: 43
D/TA: dec_value:125 has been called
I/TA: Got value: 43 from NW
I/TA: Decrease value to: 42
D/TA: randomkey_get:144 has been called
I/TA: Got value: 42 from NW
I/TA: Decrease value to: 41
D/TA: randomkey_enc:163 has been called
I/TA: Got value: 41 from NW
I/TA: Decrease value to: 40
D/TC:? 0 tee_ta_close_session:499 csess 0xe17c850 id 1
D/TC:? 0 tee_ta_close_session:518 Destroy session
I/TA: Goodbye!
D/TA: TA_DestroyEntryPoint:50 has been called
D/TC:? 0 destroy_context:298 Destroy TA ctx (0xe17c7f0)
```