Smaug-sqlite

2022年4月12日 10:08

Modifying the OP-TEE

- **添加系统调用**

  用户空间代码的修改

  1. 修改optee_os/lib/libutee/arch/arm/utee_syscalls_asm.S文件，添加如下内容：

```
// optee_os/lib/libutee/arch/arm/utee_syscalls_asm.S
UTEE_SYSCALL utee_sqlite_exec, TEE_SCN_SQLITE_EXEC, 4
```

  2. 修改optee_os/lib/libutee/include/utee_syscalls.h文件，添加如下内容，申明上述函数接口，在TA的源代码中包含该头文件后就可调用该接口。

```
// optee_os/lib/libutee/include/utee_syscalls.h
TEE_Result utee_sqlite_exec(const void *sql, size_t sql_size, void *res, size_t res_size);
```

  3. 修改optee_os/lib/libutee/include/tee_syscall_numbers.h文件，添加上述系统调用接口的索引值，并修改TEE_SCN_MAX的值，需要修改和添加的内容如下：

```
// optee_os/lib/libutee/include/tee_syscall_numbers.h
#define TEE_SCN_SQLITE_EXEC             72
#define TEE_SCN_MAX             72
```

  内核空间代码的修改

  4. 修改optee_os/core/arch/arm/tee/arch_svc.c文件中系统调用数组变量tee_svc_syscall_table的内容，将上述系统调用对应的内核层接口添加到该数组中，并包含申明该接口的头文件，在该文件中添加的内容如下：

```
// optee_os/core/arch/arm/tee/arch_svc.c
#include <tee/tee_sqlite.h>
static const struct syscall_entry tee_svc_syscall_table[] = {
    ......
    SYSCALL_ENTRY(syscall_sqlite_exec),
};
```

- **添加系统服务**

  1. 在本示例中建立的系统服务的源代码为tee_sqlite.c文件，需将该文件保存到optee_os/core/tee目录中。

```
// optee_os/core/tee/tee_sqlite.c
#include <assert.h>
#include <string.h>
#include <optee_rpc_cmd.h>
#include <kernel/thread.h>
#include <kernel/msg_param.h>
#include <tee/tee_svc.h>
#include <mm/tee_mm.h>
#include <mm/mobj.h>
#include <tee/tee_sqlite.h>
TEE_Result syscall_sqlite_exec(const void *sql, size_t sql_size, void *res, size_t res_size)
{
    uint8_t *sql_ree_shm = NULL;
    uint8_t *res_ree_shm = NULL;
    struct mobj *sql_mobj = NULL;
    struct mobj *res_mobj = NULL;
    TEE_Result result;
    struct thread_param params[2];
    memset(params, 0, sizeof(params));

    params[0].attr = THREAD_PARAM_ATTR_MEMREF_IN;
    // 分配共享内存
    sql_mobj = thread_rpc_alloc_payload(sql_size);
    if (!sql_mobj)
        return TEE_ERROR_OUT_OF_MEMORY;
    if (sql_mobj->size < sql_size) {
        result = TEE_ERROR_SHORT_BUFFER;
        goto exit1;
    }
```

```
    // 获取分配的共享内存的虚拟地址被保存在ree_shm中
    sql_ree_shm = mobj_get_va(sql_mobj, 0);
    // 检查虚拟地址是否有效
    assert(sql_ree_shm);
    memcpy(sql_ree_shm, sql, sql_size);
    params[0].u.memref.mobj = sql_mobj;
    params[0].u.memref.size = sql_size;
    params[0].u.memref.offs = 0;
    params[1].attr = THREAD_PARAM_ATTR_MEMREF_OUT;
    // 分配共享内存
    res_mobj = thread_rpc_alloc_payload(res_size);
    if (!res_mobj)
        return TEE_ERROR_OUT_OF_MEMORY;
    if (res_mobj->size < res_size) {
        result = TEE_ERROR_SHORT_BUFFER;
        goto exit2;
    }
    // 获取分配的共享内存的虚拟地址被保存在ree_shm中
    res_ree_shm = mobj_get_va(res_mobj, 0);
    // 检查虚拟地址是否有效
    assert(res_ree_shm);
    params[1].u.memref.mobj = res_mobj;
    params[1].u.memref.size = res_size;
    params[1].u.memref.offs = 0;
    result = thread_rpc_cmd(OPTEE_MSG_RPC_CMD_SQLITE, 2, params);
    if (result != TEE_SUCCESS)
        goto exit2;

    //tee_shm = malloc(params[1].u.memref.size);
    //memcpy(tee_shm, res_ree_shm, params[1].u.memref.size);
    //tee_svc_copy_to_user(res, tee_shm, params[1].u.memref.size);
    //free(tee_shm);

    memcpy(res, res_ree_shm, params[1].u.memref.size);
exit2:
    thread_rpc_free_payload(res_mobj);
exit1:
    thread_rpc_free_payload(sql_mobj);
    return result;
}
```

1. 修改optee_os/core/tee目录下的sub.mk文件，将tee_sqlite.c文件添加编译系统中。

```
// optee_os/core/tee/sub.mk
srcs-y += tee_sqlite.c
```

2. 同时将tee_sqlite.h文件保存到optee_os/core/include/tee目录中。

```
// optee_os/core/include/tee/tee_sqlite.h
#ifndef TEE_SQLITE_H
#define TEE_SQLITE_H
#include <tee_api_types.h>
TEE_Result syscall_sqlite_exec(const void *sql, size_t sql_size, void *res, size_t res_size);
#endif
```

3. 修改optee_os/core/include/optee_rpc_cmd.h文件增加OPTEE_MSG_RPC_CMD_SQLITE宏：

```
// optee_os/core/include/optee_msg_supplicant.h
/*
 * Sqlite3
 */
#define OPTEE_MSG_RPC_CMD_SQLITE    20
```

- **Updating the OP-TEE**

```
•Building the OP-TEE
 <OP-TEE installation directory>/README.HOW_TO.txt helper file tells the instructions.
 $> export FIP_DEPLOYDIR_ROOT=$PWD/../../FIP_artifacts
 $> source <STM32MP1 SDK PATH>/environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
 $> make -f $PWD/../Makefile.sdk CFG_EMBED_DTB_SOURCE_FILE=stm32mp157c-dk2 all
 The generated FIP images are available in $FIP_DEPLOYDIR_ROOT/fip
•Updating the SDK
```

```
$> cp -r $PWD/../build/stm32mp157c-dk2/export-ta_arm32/* <STM32MP1 SDK PATH>/sysroots/cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi/usr/include/optee/export-user_ta
```

- **Deploying the OP-TEE**

```
Replace the fip-stm32mp157c-dk2-optee.bin and recreate the image.
$> cp $FIP_DEPLOYDIR_ROOT/fip/fip-stm32mp157c-dk2-optee.bin <STM32MP1 IMAGE PATH>/stm32mp1/fip
```

- **Create the Image**

```
$> cd stm32mp1-openstlinux-5.10-dunfell-mp1-21-11-17/images/stm32mp1/scripts/
$> ./create_sdcard_from_flashlayout.sh ../flashlayout_st-image-weston/optee/FlashLayout_sdcard_stm32mp157c-dk2-optee.tsv
```

- **Image flashing**

```
$> sudo dd if=../flashlayout_st-image-weston/extensible/../../FlashLayout_sdcard_stm32mp157c-dk2-optee.raw of=/dev/sdb bs=8M conv=fdatasync status=progress
```

Modifying the OPTEE-CLIENT

- **添加RPC调用**

1. 修改optee_client/tee-supplicant/src/optee_msg_supplicant.h文件增加OPTEE_MSG_RPC_CMD_SQLITE宏：

```
// optee_client/tee-supplicant/src/optee_msg_supplicant.h
/*
 * Sqlite3
 */
#define OPTEE_MSG_RPC_CMD_SQLITE        20
```

2. 修改optee_client/tee-supplicant/src/tee_supplicant.c文件增加tee_sqlite_process函数：

```
// optee_client/tee-supplicant/src/tee_supplicant.c
#include <sqlite3.h>
#include <smaug_guorui.h>
struct callbackres {
    char *res;
    int res_size;
};
static int callback(void *cres, int argc, char **argv, char **azColName)
{
    int n = ((struct callbackres *)cres)->res_size;
    char *res = ((struct callbackres *)cres)->res;
    for (int i=0; i<argc; i++) {
        n += sprintf(res+n, "%s : %s\n", azColName[i], argv[i]?argv[i]:"NULL");
    }
    ((struct callbackres *)cres)->res_size = n;
    return 0;
}
static uint32_t tee_sqlite_process(size_t num_params, struct tee_ioctl_param *params)
{
    static int flag = 0;
    char *sql = NULL;
    sqlite3 *db = NULL;
    int rc = 0;
    char *zErrMsg = NULL;
    struct timeval startTime, endTime;
    struct callbackres cres;
    memset(&cres, 0, sizeof(cres));

    sql = tee_supp_param_to_va(params + 0);  //
    cres.res = tee_supp_param_to_va(params + 1);  //
    if (!sql || !cres.res)
        return TEEC_ERROR_BAD_PARAMETERS;
    // smag_guorui start
    if (flag == 0) {
        gettimeofday(&startTime,NULL);
        printf("time smaug_guorui_initial_start: %d %f\n", startTime.tv_sec,
startTime.tv_usec/1000.0);
        initFile();
        flag = 1;
        gettimeofday(&endTime,NULL);
        printf("time smaug_guorui_initial_end: %d %f\n", endTime.tv_sec, endTime.tv_usec/1000.0);
        printf("time smaug_guorui_initial: %f\n\n", 1000*(endTime.tv_sec-
startTime.tv_sec)+(endTime.tv_usec-startTime.tv_usec)/1000.0);
    }
    if( 0 == strncmp(sql, "select", 6))
    {
        gettimeofday(&startTime,NULL);
        printf("time smaug_guorui_select_start: %d %f\n", startTime.tv_sec, startTime.tv_usec/1000.0);
        selectHash(sql);
```

```
        gettimeofday(&endTime,NULL);
        printf("time smaug_guorui_select_end: %d %f\n", endTime.tv_sec, endTime.tv_usec/1000.0);
        printf("time smaug_guorui_select: %f\n\n", 1000*(endTime.tv_sec-
startTime.tv_sec)+(endTime.tv_usec-startTime.tv_usec)/1000.0);
    }
    else if(0 == strncmp(sql, "insert", 6) || 0 == strncmp(sql, "update", 6))
    {
        gettimeofday(&startTime,NULL);
        printf("time smaug_guorui_insert/update_start: %d %f\n", startTime.tv_sec,
startTime.tv_usec/1000.0);
        updateHash(sql);
        gettimeofday(&endTime,NULL);
        printf("time smaug_guorui_insert/update_end: %d %f\n", endTime.tv_sec,
endTime.tv_usec/1000.0);
        printf("time smaug_guorui_insert/update: %f\n\n", 1000*(endTime.tv_sec-
startTime.tv_sec)+(endTime.tv_usec-startTime.tv_usec)/1000.0);
    }
    // smag_guorui end
    if( 0 == strncmp(sql, "insert", 6) || 0 == strncmp(sql, "update", 6))
    {
        return TEEC_SUCCESS;
    }
    gettimeofday(&startTime,NULL);
    printf("time querystart: %d %f\n", startTime.tv_sec, startTime.tv_usec/1000.0);
    rc = sqlite3_open("/data/tee-database.db", &db);
    if (rc) {
        printf("Can't open database: %s", sqlite3_errmsg(db));
        sqlite3_close(db);
        return TEEC_ERROR_STORAGE_NOT_AVAILABLE;
    }
    rc = sqlite3_exec(db, sql, callback, (void *)&cres, &zErrMsg);
    if (rc != SQLITE_OK) {
        printf("SQL error: %s", zErrMsg);
        sqlite3_free(zErrMsg);
    }

    MEMREF_SIZE(params + 1) = cres.res_size + 1;
    sqlite3_close(db);
    gettimeofday(&endTime,NULL);
    printf("time queryend: %d %f\n", endTime.tv_sec, endTime.tv_usec/1000.0);
    printf("timeSQL: %f\n\n", 1000*(endTime.tv_sec-startTime.tv_sec)+(endTime.tv_usec-
startTime.tv_usec)/1000.0);

    return TEEC_SUCCESS;
}


static bool process_one_request(struct thread_arg *arg)
{
    ......
    switch (func) {
    case OPTEE_MSG_RPC_CMD_SQLITE:
        ret = tee_sqlite_process(num_params, params);
        break;
    ......
    }
}
```

3. 添加optee_client/tee-supplicant/src/smaug_guorui.c、sqlite3.c、defs.c、dbqueue.c、mhtdefs.c、mhtfile.c，这6个文件为郭蕊修改过的sqlite3.c以及进行merkleTree完整性验证的源代码文件。
   其中，smaug_guorui.c中创建TA会话进行哈希验证过程，所以需要将9aaaf200-2450-11e4-abe2-0002a5d5c51b.ta文件（在郭蕊提供的my_test/ta/目录下进行make编译）存入STM32MP157C-DK2开发板的/lib/optee_armtz/目录。

4. 同时添加optee_client/tee-supplicant/src/smaug_guorui.h、sqlite3.h、defs.h、dbqueue.h、mhtdefs.h、mhtfile.h头文件。

5. 修改optee_client/tee-supplicant/Makefile文件增加需要编译的源文件：

```
TEES_SRCS   := tee_supplicant.c \
           teec_ta_load.c \
           tee_supp_fs.c \
           rpmb.c \
           handle.c \
```

```
            tee_tpm.c \
            smaug_guorui.c \
            sqlite3.c \
            defs.c \
            dbqueue.c \
            mhtdefs.c \
            mhtfile.c

TEES_LFLAGS += -lpthread
TEES_LFLAGS += -lteec
TEES_LFLAGS += -lm
TEES_LFLAGS += -ldl
TEES_LFLAGS += -lcrypto
# Needed to get clock_gettime() for for glibc versions before 2.17
TEES_LFLAGS += -lrt
```

- **Updating the OPTEE-CLIENT**

```
• Building the OPTEE-CLIENT
  $> source <STM32MP1 SDK PATH>/environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
  $> make
• Deploying the OPTEE-CLIENT
  Replace the tee-supplicant in board.
  $> scp out/tee-supplicant/tee-supplicant root@<ip of board>:/usr/bin
```