1. Initial Setup and Understanding of the Problem

- **Objective:**
    - The goal was to integrate file reading functionality from the **Secure World** (TA) to the **Normal World** (CA) in OP-TEE, ensuring that reading from secure storage works correctly.
- **Problem:**
    - The TA was successfully loading and handling basic operations (e.g., incrementing a value) but crashed (panic: `0xffff0001`) when trying to read a file from secure storage. The crash was occurring after calling `TEE_OpenPersistentObject()` in the `READ_FILE` command.
- **Key Findings:**
    - The **normal execution (Hello World)** works as expected, meaning the integration between Normal World and Secure World is functional for basic commands.
    - The issue arose during the **secure storage access** when trying to read a file object (`my_secure_file`), resulting in a **TA panic**.

---

2. Troubleshooting Steps Taken

Step 1: Analyzing the Logs

- **OP-TEE Debug Logs:**
    - The logs provided insight into the **TA loading successfully**, but the **panic occurred after attempting to read the file** from secure storage:

```
I/TA: READ_FILE: id='my_secure_file' out=0x40016000 cap=512
E/TC: TA panicked with code 0xffff0001
```

-
    - This indicated that the **parameters** (`object id` and `output buffer size`) passed to the TA were correct, but the issue was related to **secure storage**

access (likely path or permissions).

Step 2: `tee-supplicant` Verification

- **`tee-supplicant` Process:**
    - We checked if the **`tee-supplicant`** process was running, which handles secure storage operations between the Normal and Secure World.

The `ps` command confirmed it was running as expected:

```
ps -ef | grep tee-supplicant
```

    -
    - It was using the device **`/dev/teepriv0`**.

Step 3: Ensuring Correct Permissions for REE Storage

- We needed to verify the permissions of the **secure storage directory**, which is typically used by `tee-supplicant`.
- We explored the logs from `tee-supplicant -d` and ensured it was using directories like:
    - `/data/tee`
    - `/var/lib/optee`

We verified that these directories had proper ownership and permissions, allowing the `tee` user to read/write. The necessary commands were:

```
 sudo mkdir -p /data/tee
sudo chown tee:tee /data/tee
sudo chmod 700 /data/tee
```

- If the directory wasn't accessible, it would result in a **TA panic** when the TA tried to read from secure storage.

---

3. Defensive Programming and Improved Logging

Step 4: Adding Debugging Logs

- To better diagnose where exactly the crash happened, we added detailed debug logs before and after each critical operation (like opening the file, reading from secure storage, etc.) inside the `TA` code.

- This helped us identify whether the crash was occurring at **file opening** or **during reading**.

Example of added debug prints:

```
IMSG("about to OPEN '%s'", id_buf);
r = TEE_OpenPersistentObject(TEE_STORAGE_PRIVATE, id_buf,
strlen(id_buf), TEE_DATA_FLAG_ACCESS_READ, &oh);
IMSG("after OPEN r=0x%x", r);
```

-

Step 5: Using `tee-supplicant` in Debug Mode

- We ran `tee-supplicant -d` in **debug mode** to monitor its logs while running the application. This gave us insights into the REE storage path and permissions, and if there were any file access failures.

    - If any file access fails (due to permission issues), `tee-supplicant` will log it clearly.

Step 6: Ensuring Proper File Handling and Buffer Allocation

- We modified the `READ_FILE` function to handle file reading in a **more defensive manner**, including:

    - Checking if the **output buffer** and **its capacity** are valid before proceeding.

- **Small probe reads** were done first (in a local buffer) to detect potential buffer overflow or memory reference issues before reading into the actual allocated buffer.

- This ensured that the TA wouldn't crash unexpectedly due to incorrect memory management.

---

4. Additional Fixes & Suggestions

Step 7: Ensuring `tee-supplicant` is Running Properly

- We manually ensured that **`tee-supplicant`** was running and listening on the right path for file storage.

    - The process should be kept running in the background to handle any interactions between the Normal World and Secure World.

    - We also made sure that the **correct directory for secure storage** was writable by the `tee` user.

---

5. Summary of Issues and Solutions

Problem

- The **TA crashed** when attempting to read a file from secure storage, indicated by a **panic code `0xffff0001`**.

- The issue was likely related to **incorrect permissions** or **missing directories** for secure storage managed by `tee-supplicant`.

Solution

1. **Ensured `tee-supplicant` is running**: We verified its operation and debugged using `tee-supplicant -d` to monitor logs.

2. **Fixed REE FS directory permissions**: The directories used by `tee-supplicant` (e.g., `/data/tee` or `/var/lib/optee`) were checked and permissions were

corrected.

3. **Improved TA error handling and logging**: We added detailed prints in the TA to debug each step of the file reading process.

4. **Validated buffer allocation**: A **probe read** was performed before trying to read the full file, ensuring no buffer overflow or memory issues occurred.

Next Steps

- If the issue persists:

    1. Review the **`tee-supplicant` debug output** for errors like “permission denied” or “failed to open file”.

    2. Ensure that **`TEE_OpenPersistentObject()`** is correctly accessing the file, and permissions are set correctly for **`tee:tee`** ownership.

    3. **Post error details** if available, especially any new logs from `tee-supplicant`.

Conclusion

We systematically walked through the issue, improved the code's robustness with more logging and defensive checks, and also verified the necessary **storage directory permissions** for `tee-supplicant`. These actions should have helped identify and resolve the panic issue you encountered.

If you still face problems, sharing the **final logs from `tee-supplicant`** or the **TA's debug output** will help pinpoint the root cause even more precisely.

Let me know if you need further assistance!