Convert this docx to PDF before submitting it to Collab
No code submission needed

TEE Q&A (10 pts each)

Answer each question in **a few sentences**.

1. Why do we need TEE, given that the OS kernel is already providing isolation and protection to applications? State the security advantage of TEE vs. OS. What security protection that the TEE can provide but the OS cannot provide.

   The purpose of the TEE is to prevent the execution of untrusted or unauthorized code on end users devices. TEE offers data integrity and confidentiality so that no third party can access the data when unencrypted. TEE ensures code integrity. TEE provides secure conjunction for secure collaboration where organizations do not need to trust each other with sensitive data. Finally, TEE provides flexibility in managing output destinations. In addition, we cannot fully trust the OS which is why we need a TEE.

2. By design, what type of code should execute in TEE? State the design consideration: code size, programming languages, dependencies on 3rd party libraries, execution time, the level of trustworthiness, etc.

   Ideally, the type of code that should be executed in TEE should be relatively smaller because more lines can raise the chances of more bugs and security flaws. The programming language for the code that should execute in TEE should ideally be secure but varies on specific TEE implementation. Common languages include C/C++, Rust, Java and Kotlin, Go, and many other languages. The code should be verifiable, have maximum trustworthiness, and limited dependencies on 3rd party libraries since the TEE should run relatively isolated. In addition, the code should be quick because some applications may demand more emphasis on computational speed. Examples of TEE operating systems include Apple's Secure Enclave, Qualcomm QTEE, Samsung TEEgris, and others.

3. There is an argument that TEE is more trustworthy than a commodity OS kernel, e.g. Linux. Do you agree? Why? State to which attacks or threats the OS kernel is vulnerable, and to which attacks the TEE is vulnerable. Compare these attacks.

   The OS kernel is vulnerable to many things. One example is kernel exploitation which involves manipulating the kernel of an OS. Some common techniques for kernel exploitation includes buffer overflow attacks, heap spraying, and use-after-free vulnerabilities. Other attacks include the Netfilter vulnerability, Heartbleed, Spectre

and Meltdown among many others. TEE are also vulnerable to some threats or attacks. For example, if your TEE is an external coprocessor design, a malicious actor could tap into the secure coprocessor with some device or inject into the connection between the secure coprocessor and CPU. Other TEE vulnerabilities include side-channel attacks, rootkits and firmware attacks, malicious insiders, supply chain attacks and many others. It seems as if the threats and dangers to the OS kernel are generally simpler and require less expertise than that of the TEE. In addition, the threats and dangers that the OS kernel is vulnerable to are more common than the threats and dangers which make the TEE vulnerable.

4. What does trusted computing base (TCB) mean? What is the TCB of your smartphone?

   The TCB refers to all the system components that are critical to establishing and maintaining the security of a particular system. In other words, TCB refers to anything provides a secure environment for operations. In the smartphone, the TCB includes various hardware and software. One major TCB includes the security enclave which is a separate processor dedicated to handling sensitive data such as touch id and face id. Other features include App Store where each app undergoes review, hardware-based encryption, and the IOS security architecture incorporates various processes.

5. What is the role of "TEE supplicant"? What does it do?

   A TEE supplicant is a program that runs as a daemon responsible for remote services expected by the TEE OS. TEE supplicant will receive the messages and store encrypted data accordingly to the Linux file system. The TEE supplicant handles the remote services expected by the TEE OS. In other words, it embeds the OS and the TEE enabling communication between the two. In more detail, the TEE file system will encrypt the data and send REE file operation commands and encrypted data to TEE supplicant using a series of RPC messages. TEE will then store the messages into encrypted data according to the Linux file system.

Environment setup

Follow the tutorial and run the "helloworld" example

https://fxlin.github.io/p3-tee/quickstart/

1. Are you using QEMU or the Rpi3 hardware? If QEMU, are you using granger1, granger2, your local Linux, your local WSL, or others? (10 pts)

   QEMU; granger2

2. Show a screenshot of you successfully running `helloworld` . It must be generated by yourself (40 pts)

P3exp1

```
# optee_example_hello_world
Invoking TA to increment 42
TA incremented value to 43
#
```

```
# optee_example_hello_world
hello! ... Invoking TA to increment 42
TA incremented value to 43
#
```

```
# optee_example_hello_world
hello! ... Invoking TA to increment 42
TA incremented value to 44
#
```