# HW-6

Graded

**Student**

Sivachandran P

**Total Points**

**50 / 50 pts**

**Question 1**

## Isolation

**10** / 10 pts

✔ **+ 10 pts Correct**

**+ 7.5 pts Mostly Correct**

**+ 5 pts Partially Correct**

**+ 2.5 pts Attempt**

**+ 0 pts Not Relevant/Plagiarism/Use of AI Tools**

**Question 2**

## Trusted vs. Untrusted Environment

**10** / 10 pts

✔ **+ 10 pts Correct**

**+ 7.5 pts Mostly Correct**

**+ 5 pts Partially Correct**

**+ 2.5 pts Attempt**

**+ 0 pts Not Relevant/Plagiarism/Use of AI Tools**

**Question 3**

## Trustzone Implementation

**10** / 10 pts

✔ **+ 10 pts Correct**

**+ 7.5 pts Mostly Correct**

**+ 5 pts Partially Correct**

**+ 2.5 pts Attempt**

**+ 0 pts Not Relevant/Plagiarism/Use of AI Tools**

Question 4

## Secure Boot

**10** / 10 pts

✔ **+ 10 pts** Correct

**+ 7.5 pts** Mostly Correct

**+ 5 pts** Partially Correct

**+ 2.5 pts** Attempt

**+ 0 pts** Not Relevant/Plagiarism/Use of AI Tools

Question 5

## Security Threats

**10** / 10 pts

✔ **+ 10 pts** Correct

**+ 7.5 pts** Mostly Correct

**+ 5 pts** Partially Correct

**+ 2.5 pts** Attempt

**+ 0 pts** Not Relevant/Plagiarism/Use of AI Tools

**Q1 Isolation**
**10 Points**

ARM Trustzone isolation between trusted vs. untrusted operating environment and Android App isolation between Apps signed by different entities, have similarity.
Please explain at least two similarities between the two mechanisms.

Sharing below the list of similarities found between two mechanisms,
ARM Trustzone isolation between trusted vs. untrusted operating environment:
---
1) Minimizing attack surface: Strategized to keep less amount of code and resources to run in the secure world
2) Memory isolation: Isolated the memory access areas between secure & non-secure worlds
3) Permission-based access control: Access is controlled via Secure Monitor Call (SMCs) between secure & non-secure worlds
4) Hardware assisted isolation: Hardware resources are segregated as secure & non-secure worlds
5) Independent authentication and validation: Secure world access via Cryptographic verification method

Android isolation between PPS signed by different entities:
---
1) Minimizing attack surface: Designed that could not access other app"s resources
2) Memory isolation: Every App got assigned with unique address space, therefore it helps to achieve non-access to other's memory space
3) Permission-based access control: Permissions are handled through app sandboxing for sensitive operations (eg. accessing Calendar, contacts, location etc.)
4) Hardware assisted isolation: Kernel level process isolation to keep apps separate
5) Independent authentication and validation: Ensures authentic verification and trusted via their signatures

**Q2 Trusted vs. Untrusted Environment**
**10 Points**

Why is it required to isolate a trusted execution environment from an un-trusted execution environment (REE) in the context of a mobile phone?

Note: Data is usually encrypted in storage and transit and is only decrypted when it's in the TEE for processing. Benefits of TEE are Data integrity & confidentiality, Data integrity & confidentiality, Code integrity, Secure collaboration, Simplified compliance.

1) Isolating (TEE) trusted execution environments makes it much harder for malicious actors to exploit vulnerabilities in the REE (Rich execution environment) to gain access to the TEE.
Un-trusted execution environment is called as REE (Rich Execution Environment).

2) Malware running in the REE won't have chance to access directly to the TEE's memory or execution, ultimately it makes significantly harder to compromise.

3) Isolation between TEE & REE ensures that the integrity and authenticity of these operations maintained even in the presence of potentially compromised applications running in the REE.

4) By ensuring that TEE is isolated from malware (trojans, spyware) and threats, the sensitive functions handled in the TEE such as secure payment transactions, authentication or provide data storage - remain safe.

5) TEE greatly enhance mobile security by isolating sensitive operations and provide secure environment for analyzing data.

**Q3 Trustzone Implementation**
**10 Points**

Today most System-on-Chips (SoCs) have multi-core CPUs. Therefore, one way to implement a trusted and untrusted OS on the same platform would have been to run the trusted OS on one CPU and untrusted one on another CPU. However, in the TrustZone architecture, the two run on the same CPU with trusted and untrusted mode context switching. Explain your own understanding of the reason for such an implementation.

Reason for TrustZone architecture:
---
The Rich Operating System (ROS) have a fare largest footprint than their predecessors, and, give that the number of potential security bugs increases with the number of lines of code, the threat surface is also increased.

With the widespread deployment of app stores and third-party app support the threat surface is extended significantly, as such malware is now a real threat to mobile devices.

To combat these new threats, a robust platform architecture is needed.
The architecture must be able to provide a trusted environment that is isolated and protected from the ROS.
TrustZone technology supports this requirement by providing a binary partition that devides the system into two isolated worlds: trusted & Non-trusted

Context switches between security states can only be made using dedicated instructions and code that ensures that strict isolation is maintained.
The context switch mechanism enforces fixed code entry points and ensures that code running in the Non-secure state cannot access registers that belong to the Secure state.

Conceptually, the secure and non-secure states can be regarded as two virtual processor cores.

Trusted Base Security Architecture (TBSA) is founded on ARM TrustZone technology - to ensure confidentiality, authenticity and Integrity.

Sharing below few benefits of such implementation to provide a cost-effective security, efficient way to run both trusted and untrusted software while maintaining a high level of security and performance.
1) Flexible and efficient use of hardware resources
2) Isolation via secure and non-secure modes

3) Minimal overhead and low latency
4) Enhanced security through hardware isolation, Software separation
5) Cost effective, scalable,
6) Enabling secure execution in a Multi-tenant environment

**Q4 Secure Boot**
**10 Points**

Explain in your own words the need for secure boot? Explain the sequence of activities involved in a secure boot process?

Need for Secure boot:
---
i) It is essential to prevent unauthorized executable files to run
ii) It offers sufficient protection against malware, rootkits, bootkits, that could compromise the system from the moment it starts
iii) It guarantees integrity by validating and allowing genuine software to load, and helps authorities regulate relevant industry players to comply with data security standards
iv) Critical for enhancing security for sensitive environments (financial institutions, government agencies etc.)
v) It is a foundational security mechanism to maintain the integrity of the device and its operations from the very beginning

Sequence of activities involved in secure boot system:
---
i) Power-ON or RESET: Initialization of Hardware (Software not loaded yet)

ii) Execution of (Boot) ROM: Mostly located at SoC is the first to execute. In general this ROM is small and secured program hard-coded into the SoC area or stored in a non-volatile memory

iii) Verification of Firmware (Bootloader): Ensures integrity of system's firmware or bootloader

iv) Loading and verification of OS boot loader: In this step system process to load the boot loader, and responsible for loading the operating system and additional software

v) OS kernel verification: Loading of OS kernel

vi) Initialization of Secure execution environment: Load and initialization of trusted execution environment

vii) System initialization and user-level boot: Initialization of user-level process, drivers and applications

viii) Run-time monitoring and integrity checks: Monitoring and enforce

security policies in the runtime to ensure the integrity of environment, this can include "Runtime Integrity checks & Firmware and Software update"

**Q5 Security Threats**
**10 Points**

In the lecture, we have talked about 9 major threats that TrustZone architecture intends to mitigate. Write one sentence explanation for each of the 9 threats explaining why loss of confidentiality, or availability or integrity may result from each threat. (Please do not reproduce the content from the slides but explain why for example cloning of trusted service may endanger a security property or why NVS.READ will lead to endangering some security property etc.)

9 major threats that TrustZone architecture intends to mitigate:
---

1) T.FUNC_ABUSE – Functional abuse
Misuse of services in unintended ways, service functions, enabling unauthorized actions or exposing sensitive data.

2) T.CLONE – Trusted Service cloning
Exploitation of unauthorized duplicated secure service in the multiple instances.

3) T.DEBUG_ABUSE – Debug feature abuse
Exploitation of debug functions which are meant for development purpose.

4) T.NVS.READ – Reading of non-volatile storage (Flash/HDD)
Expose of sensitive data stored in Flash memory or Hard drives by unauthorized entities.

5) T.NVS.WRITE – Writing to non-volatile storage (Flash/HDD)
Alter or corrupt the sensitive data stored in Flash memory or hard drives without proper authorization.

6) T.RAM.READ – Reading of Trusted Service RAM
Expose of critical information (encryption keys, user data, intermediate computations etc.) held in memory by unauthorized entities.

7) T.RAM.WRITE – Writing to Trusted Service RAM
Injection of malicious code, Data corruption, manipulation of trusted operations by unauthorized entities.

8) T.COVERT_PATH – Covert leakage paths
Leakage allows transmit to non-standard channels, enabling unauthorized access to sensitive information.

9) T.ROLLBACK – Rollback to previous versions of code/data

Reverting to potentially vulnerable version of code or data or re-enable previously patched vulnerabilities.