The Journal of Open Source Software

mTower: Trusted Execution Environment for MCU-based devices

Taras A. Drozdovskyi[1] **and Oleksandr S. Moliavko**[1]

1 Samsung R&D Institute Ukraine, Kiev, Ukraine

DOI: 10.21105/joss.01494

Software

- Review
- Repository
- Archive

Submitted: 07 May 2019
Published: 27 August 2019

Background

Embedded computing systems have already become ubiquitous in daily life. In many cases these systems deal with confidential data and control access to valuable resources. Significant efforts are put into securing these systems against malicious access. However, the software and data are often protected by outdated mechanisms that contain well-known vulnerabilities, thus do not pose any real obstacles for technically savvy adversary. Consequences of deliberate attack against embedded systems can be severe, e.g.:

- poor data protection in network-connected medical device can pose life risk (Basu, 2013)
- hardcoded cryptographic keys and application vulnerable to reverse engineering in automotive platform (Valasek & Miller, 2014)

Yet developing safe and efficient data protection mechanism from scratch is time-consuming and costly endeavor. Without an independent assessment there is always a risk of some undiscovered vulnerability persist. However, current generations of ARM-based microcontrollers offer a solid hardware foundation for building security mechanisms: ARM TrustZone technology. Initially developed for ARM family of CPUs, TrustZone technology was later adapted for microcontroller (MCU) implementations of ARM architecture. Software libraries that implement security-related operations based on ARM TrustZone are readily available for Linux family of OSes including the one used in Android-based smartphones. However, these libraries are too cumbersome and resource-consuming to be directly adopted to embedded systems and MCU-based devices where clock speeds are orders of magnitude lower and RAM available for use is severely limited. There are several attempts to develop a TrustZone-based security solution for MCU-based systems: Kinibi-M (Trustonic, 2017), ProvenCore-M (Prove & Run, 2018), CoreLockr-TZ (Sequitur Labs Inc., 2018). However, these solutions are either proprietary and not available as a source code for independent review, or have technical limitations that make them not fully standard compliant. mTower is an experimental industrial standard-compliant implementation of GlobalPlatform Trusted Execution Environment (GP TEE) APIs based on ARM TrustZone. From the very beginning mTower was designed to have minimal RAM footprint and to avoid the use of time-consuming operations. As of May 2019 mTower contains a partial implementation of GP TEE API, with remaining part of API to be implemented soon. Source code and compatibility table are available at https://github.com/Samsung/mTower.

Implementation overview

Secure applications that utilize TrustZone protection live in two interacting environments: Normal World (NW) and Secure World (SW). Normal World part is typically a standard Linux executable that uses TEE library containing API functions to interact with Secure World part.

The corresponding Secure World part is essentially a set of event handlers that are executed in hardware-protected area of RAM under control of a specially designed operating system. Secure World part processes calls received from Normal World and deals with sensitive data that must be protected from unauthorized access such as cryptographic keys, passwords, user identification data, etc. Common operations performed by Secure World part of the application include data encryption/decryption, user authentication, key generation, digital signing and signature verification.

Secure application architecture should comply with TEE System Architecture v1.0 document (GlobalPlatform, 2018a). mTower follows the layout described in this specification. Also it provides the implementation of APIs described in GP TEE specification documents. The two most important GP specifications are:

- TEE Client API Specification v1.0 (GlobalPlatform, 2010) describes the communication between NW applications and Trusted Applications residing in the SW
- TEE Internal Core API Specification v1.0 (GlobalPlatform, 2018b) describes the internal operations of Trusted Applications

Most up-to-date versions of these specifications are available at Global Platform web site (GlobalPlatform, 2018c) after registration.

mTower's modular architecture allows for build-time configuration of the required features to optimize memory footprint and performance. Originally resource management for mTower is based on FreeRTOS real-time operating system. It can be replaced by other operating systems if needed. As of May 2019, mTower runs on Nuvoton M2351 board based on ARM Cortex-M23 MCU. There are plans to include support for other platforms based on ARM Cortex-M23/33/33p family MCUs.

Potential future development

After completing the implementation of relevant APIs, we plan to provide support for dynamic loading, remote updating of secure applications including the essential security mechanisms of OS. Also we plan the extension of Resource Manager that provides secure access to H/W. Another planned component is the set of instrumentation hooks in mTower code that will simplify specification compliance evaluation, performance measurements, assessment and debugging of secure applications.

Expected audience for mTower project

mTower project is expected to be used for proof-of-concepts and evaluation models for ARM TrustZone-based secure applications in MCU-based systems. It can be of interest for:

- Internet-of-Things and Smart Home appliance developers
- embedded system developers in general
- computer security specialists

Research applications

mTower is being developed as a part of search for efficient hardware/software security solution for low-cost IoT devices. It is expected to demonstrate the feasibility of porting GP TEE-compliant Trusted Applications from full-feature CPU-based to MCU-based systems. Another

mTower targeted application is using it as a platform for developing and evaluating secure applications for EDGE devices. Also only a limited information is available about resource consumption penalty incurred by adding security mechanisms into IoT applications. The prevailing opinion seems to be that security through obscurity is sufficient for embedded systems, and any attempt to introduce sophisticated security features incurs an unacceptable overhead. Thus another objective of mTower development is to provide a reference platform that allows the researchers to actually measure this overhead by comparing system performance with mTower security mechanisms and without them. We hope that mTower will contribute to TrustZone-based security adoption for low-cost IoT.

References

Basu, E. (2013). Hacking insulin pumps and other medical devices from Black Hat. *Forbes*. Retrieved from https://www.forbes.com/sites/ericbasu/2013/08/03/hacking-insulin-pumps-and-other-medical-devices-reality-not-fiction

GlobalPlatform. (2010). TEE Client API Specification v1.0. Retrieved from https://globalplatform.org/specs-library/tee-client-api-specification/

GlobalPlatform. (2018a). TEE System Architecture v1.2. Retrieved from https://globalplatform.org/specs-library/tee-system-architecture-v1-2/

GlobalPlatform. (2018b). TEE Internal Core API Specification v1.2.1. Retrieved from https://globalplatform.org/specs-library/tee-internal-core-api-specification-v1-2/

GlobalPlatform. (2018c). GlobalPlatform Specification Library. Retrieved from https://globalplatform.org/specs-library/

Prove & Run. (2018). ProvenCore-M. Retrieved from http://www.provenrun.com/products/provencore-m/

Sequitur Labs Inc. (2018). CoreLockr-TZ. Retrieved from https://www.sequiturlabs.com/corelockrtz/

Trustonic. (2017, October). Not just droning on! The rise of Kinibi-M. Retrieved from https://www.trustonic.com/news/blog/not-just-droning-rise-kinibi-m/

Valasek, C., & Miller, C. (2014). *Adventures in automotive networks and control units*. IOActive, Inc. Retrieved from http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive_Networks_and_Control_Units.pdf