Approaches to Achieving SABLE Objectives on ARM*

Adam T. Wiethuechter and Stuart W. Card

***Abstract*— This paper explores the ARM environment, specifically TrustZone, to see how the Syracuse Assured Boot Loader Executive (SABLE) objectives can be achieved on ARM systems. It explores the Trustonic Trusted Execution Environment (known as Kinibi) and other alternatives in the current ecosystem to accomplish this goal of bringing SABLE-like features into the ARM world. Future technology is also explored to think about how it affects current models and approach for hardware based security.**

I. INTRODUCTION

Critical Technologies Inc. (CTI) under contract with the Defense Advanced Research Projects Agency (DARPA) is creating a framework for remote attestation of given systems that have undergone a secure and trusted boot. Under other various contracts CTI has been working towards this goal for a number of years. SBIR Topic AF121-051 Remote Attestation & Distributed Trust in Networks (RADTiN) sponsored by the Air Force Research Laboratory Information Directorate (AFRL/RI) paired CTI with Syracuse University to develop the Distributed Attestation for Mobile, Multicast and Multi-Operator Networks (DAM3ON) architecture to bring remote attestation to mobile wireless clouds.

II. KEY TECHNOLOGIES

A. Remote Attestation & Trusted Boot

Remote attestation is a method where a client system will authenticate its state (mainly in hardware and software) to a host system with which it wishes interact [1]. The end goal is generally to gain access to resources remotely on a network. A critical part of this method is that the systems have come up in a ”trusted boot” and can attest to other entities that they are not corrupted or compromised, enabling the other party to verify state without jeopardizing either parties’ security. However in ”trusted boot”, the booting of a malicious OS or application can still occur. Trusted boot implementations only focus on recording measurements of the boot code, operating system and application code, not ensuring the actual integrity of these pieces [2].

B. SABLE & Trusted Platform Module & Secure Boot

Syracuse Assured Boot Loader Executive (SABLE) uses the Trusted Platform Module (TPM) chip on x86 systems to measure and verify boot options before allowing them to launch giving the system a true secure boot scenario - avoiding the issue of trusted boot described above. The SABLE effort has the following primary attribute goals:

- Capability-based
- Secure boot
- Trusted & Trustworthy boot

We would like to point out that CTI sees a difference between ”Trusted” and ”Trustworthy”. In most contexts ”Trusted” is given to software or hardware that has been checked by an internal process or third party. The definition of ”trusted” used by the third party or process may not agree with that of the end users. CTI hopes that this framework, along with SABLE and its formal verification, will bring systems closer to being ”trustworthy” rather than just ”trusted”.

After booting, these same measurements from the TPM can be used to assure a trusted boot and perform remote attestation with other systems. Dynamic Root of Trust for Measurement (DRTM, also known as late launch) instructions on both AMD and Intel platforms give the ability to measure Trusted Computing Base (TCB) components during and even after boot.

ARM systems do not have DRTM functionality - instead ARM has what is known as TrustZone. With the goals of SABLE in mind, we explored the TrustZone ecosystem, to see in what way SABLE attributes could be included in ARM devices using current technology or possible future technology.

III. TRUSTZONE ECOSYSTEM

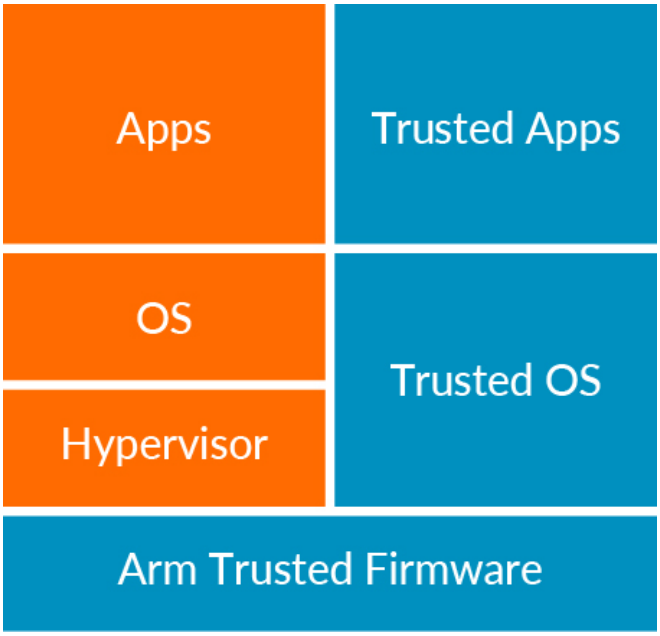

Fig. 1. TrustZone architecture defined by ARM [4]

*This work was supported by the Defense Advanced Research Projects Agency (DARPA)

ARM TrustZone is the counterpart in ARM systems to the TPM chips found on x86 based systems. TrustZone's main feature is the creation of two worlds, the Secure World and non-secure or Normal World within the CPU [3]. These two worlds are kept separated in hardware by explicit context switching in software. When a switch occurs from the Normal World to the Secure World, the process sees an entirely different setup in hardware defined by TrustZone to keep certain system functions and data safe. This change affects not only the CPU, it affects all aspects of the hardware a process could access, such as all the peripherals and memory controllers. Figure 1 shows the TrustZone architecture diagram that ARM uses to explain the concept of TrustZone.

A. Trusted Execution Environment

A Trusted Execution Environment (TEE) seems to have many slightly different definitions. However the most prevalent is that a TEE consists of the following [3]:

- Hardware with TrustZone
- A trusted boot process
- A trusted Operating System (OS)

Item one is provided by manufacturers of the hardware and generally will consist of an ARM System on Chip (SoC) such as one of the processors in the Cortex-A lineup. Item two is enabled by ARM firmware that uses TrustZone to perform boot operations in the Secure World and give access to the switching between Secure and Normal World operations. Finally, item three is rather ambiguous. There is no clear definition of what exactly a "trusted OS" is and furthermore all TEEs that exist today use proprietary "trusted" OSs in their stack.

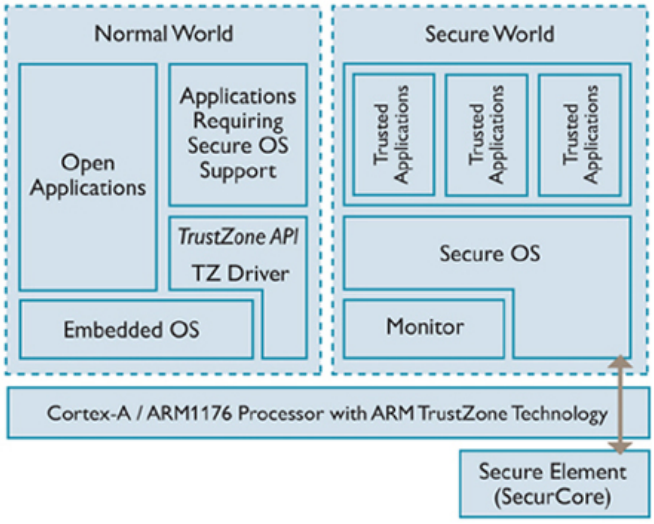

Fig. 2. A detailed view of TEE components in TrustZone [8]

However all trusted OSs have small feature sets for a small attack surface and only provide what is required to run a given security application on top of it. In comparison Android is referred to as a "rich OS" that lives in the Normal World and it, with the help of drivers and libraries in the Normal World, queries the trusted OS when security functions are required. Such functions would include the use of the fingerprint sensor, cryptographic key check and the like.

The above definition of a TEE comes from ARM itself [4]. However, GlobalPlatforms[1] has been working to create a standard for TEE's (which most TEE developers now adhere to and register compliance for). GlobalPlatforms has created many collaborative relationships in the industry such as with ANSSI and W3C to deliver strong standards such as the ones used for TEEs [5].

Current vendors that have certification through GlobalPlatforms include such big names as Samsung, Verizon and even ARM itself, giving their standard credibility in the field [6]. This is their definition of a TEE:

> The TEE is a secure area of the main processor in a smartphone (or any connected device) and ensures that sensitive data is stored, processed and protected in an isolated, trusted environment.[7]

Figure 2 shows a detail view of a typical TEE environment implemented in TrustZone based hardware. The connection between the secure and non-secure world is through the component called "TrustZone API TrustZone Driver" that allows Normal World applications support from Trusted Applications running within Secure World.

B. Trustonic TEE

The Trustonic TEE is known as Kinibi and is part of what they call their Trusted Secure Platforms (TSP) solution. Trustonic sent us their SDK for evaluation in this effort. The SDK includes the required documentation for the various APIs along with samples of various small Trusted Applications (TAs) and Client Applications (CAs).

The TEE documents specify that applications can be built to target the Linux world. Building applications to target Android specifically is a different option. In the setup process it is stated clearly that targeting Linux should only be used to build connectors. The main architectures of ARM, x86 and x86_64 are supported. Applications can be written in either Java or C. It is recommended that C be used for portability to different system types.

Kinibi follows the model of two worlds from TrustZone - one secure and one normal. Kinibi's core and Trusted Applications live in the Secure World. The various APIs allow Client Applications (in the Normal World) to access security functions (in the secure world) in a server/client like fashion.

Applications in the Normal World typically will have a Trusted Application Connector component, known as the TLC, that handles the interfacing to Kinibi and the requested Trusted Application through various different APIs (depending on legacy factors). This is viewed more as a library that CAs can access from the rich OS. A TLC can give access to one or more TAs in Kinibi and handles all the necessary commands to initialize the communication between Worlds.

[1] https://globalplatform.org/

C. Trustonic TEE Conclusion

Kinibi, the Trustonic TEE, has been built with the mobile Android world in mind, not the general PC consumer market. Trustonic has built a version of Kinibi for embedded systems as well (known as Kinibi-M), however its practicality for this effort was not explored.

After our exploration of Kinibi, we conclude that we can not use it to meet our goals. This is due to the following factors:

- Kinibi is closed-source and proprietary
- Kinibi must be embedded on the SoC in the factory and can not be altered in any way
- The SDKs, which were provided to us for review, limit interactions with the TrustZone environment to Kinibi methods

If smartphones are the target system and if we wish to incur the high up front cost (and subsequent limitations) of proprietary software, then Kinibi would be an excellent choice. However, CTI is a firm believer in the open-source model (which Kinibi violates) and our target hardware is not smartphones but instead general purpose systems that have ARM processors.

IV. Future Plans & Technologies

Because Trustonic TEE does not meet our needs, we began to explore other technologies to give us our desired results. We also looked to the future of the ecosystem and attempted to make a judgment of our best path forward as the technology changes.

A. Future Options for SABLE Functionality in ARM

With our focus on SABLE and its functions, one route forward that was identified was the use of the ARM firmware. The firmware is available as open source from GitHub [9] allowing anyone to create their own TEE by pairing the firmware with an OS they trust.

Our goal is to have SABLE attributes inserted somewhere in the stack for ARM - thus giving true secure and trusted boot. ARM firmware could be altered to insert SABLE into the boot process - however the question of how to perform measurements and attest to them is a big one. We would most likely have to be implement a virtual TPM at the firmware level for this to occur, as most ARM based boards do not have TPM chips.

Another option is to trust that ARM firmware will properly boot the secure world. Genode (possibly with seL4) will run in this Secure World. On top of Genode acting as TAs will be two functionalities. The first is a virtual TPM that the other TA (which is SABLE) would use to perform measurements on the normal world system. Virtual TPM chips have begun to appear and show promising results [10], giving us an indication that this is the best way to move forward with the current technology.

B. Research & Opinion of Others

Genode Labs has done some experiments with TrustZone as reported in various articles that go in depth into the technology and its use cases using the Genode software. Genode Labs has come to the conclusion TrustZone is not useful in visualization, however it has merit as a tool comparable to TPM in the ARM world [3]. They have also done work with the Genode software creating a USB Armory example, driving them to adjust the hw-kernel to support TrustZone [11].

The seL4 developers at Data61 have concluded that TrustZone is pointless and provides nothing that seL4 doesnt already support. These developers bring two scenarios to light to demonstrate this [12]:

- First is that seL4 is used in the Secure World to isolate different TEEs and use the TrustZone monitor to perform world switching
- Second is that the TrustZone monitor from Secure World only does what is needed to boot the main system in Normal World and nothing ever runs in the Secure World again

The seL4 developers argue the TrustZone monitor (if not formally verified) is a huge attack surface out of the developer's control. The second scenario uses TrustZone for just the boot up, giving the user secure boot. However a response from Norman Feske from Genode Labs clears a misconception that TrustZone itself provides secure-boot; it does not, the SoC vendor provides secure boot in some form like High Assurance Boot (HAB) from FreeScale [13].

In the same discussion a starting point was proposed to use OP-TEE as a base and porting its functions as seL4 services [14]. However, these discussions are from 2016 and no record of the path can be found.

C. Future Technology

During this investigation, we also explored the landscape of TrustZone and TPM based systems. We found there may soon be a paradigm shift in the technologies that may require us to refocus our efforts towards different solutions. This shift could solve many of the problems we have faced in attempting to get TPM-like functionality from TrustZone by actually giving us a cleaner foundation to work on.

Intel has introduced Software Guard Extensions (SGX), a set of instructions that break away from the normal convention of privilege levels in computing. It creates what are known as enclaves, which are protected areas of user-level code that can not be accessed by users of higher privilege layers [15]. This is done by cryptographically hashing all values in its dedicated memory space and performing decryption and encryption of the data for every read and write.

AMD has followed suit, although in a different manner. AMD's x86 processors also contain one or more ARM co-processors used for security. The ARM co-processors are on the same piece of silicon as the x86 CPU's however these co-processors have independent segments vital components -

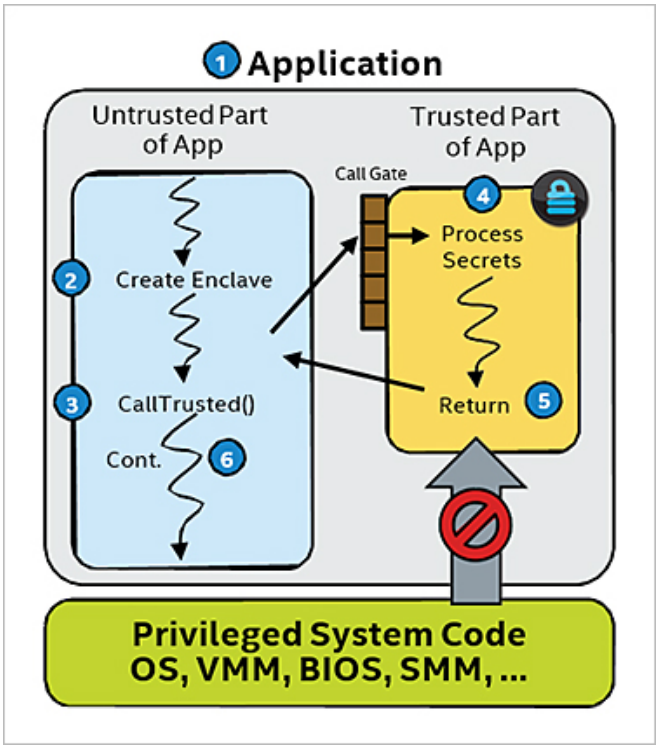

Fig. 3. Example of Intel SGX during a runtime exception [15]

creating a true hardware based isolation. These co-processors are used to encrypt data of various processes directly in memory to prevent leaking of data. [16]

Past events have shown that Intel, AMD and ARM tend to stay in step with each other in feature sets by offering similar functions. Changes in how Intel and AMD handle security could change how TrustZone operates in the near future as ARM moves to stay up to date with its competitors. For us this means that instead of struggling to implement SABLE's objectives in an already defined ecosystem we can perhaps incorporates the objectives more easily.

Another trend that we explored that could extend into the future is the use of field programmable gate array (FPGA) units to be used as the base of systems. These FPGAs can be dynamically flashed by software to emulate hardware units [17]. In a very basic form, FPGAs can be viewed as malleable pieces of hardware that are highly flexible making them desirable.

A lot of FPGAs are flash once systems. Other FPGAs can be flashed as many times as desired by the user. Either case poses an issue for hardware based security because an FPGA can be flashed by software to masquerade as a specific piece of hardware. The only way to confirm the FPGA is behaving as it should is by actually examining it under a microscope and comparing it to valid hardware.

V. CONCLUSIONS

Our research, documented in this paper, gives a clear view of the path forward. Trustonic TEE is a great technology, however for our purposes does not help us in any meaningful way. This is partially due to being proprietary technology (which CTI strongly opposes) and being focused on the smart-phone market rather than general computing.

CTI believes SABLE attributes can be added into the ARM ecosystem by adding virtual TPM chip TAs into the secure-world system or directly into the ARM firmware/secure-world operating system to allow various TAs to use the technology. However, the ever changing landscape may make the best option to wait for ARM to bring itself in step with Intel and AMD and start fresh in a newer ecosystem free of the restrictions and expectations of the past.

Consideration must also be given towards the fundamental shift of using dynamic systems, such as FPGAs, and the way they break our current security models (specifically hardware based models). CTI believes, looking towards this new possible future, including such cases in the security models would benefit by mitigating future security nightmares.

REFERENCES

[1] L. Jain and J. Vyas "Security Analysis of Remote Attestation." Internet: https://seclab.stanford.edu/pcl/cs259/projects/cs259_final_lavina_jayesh/CS259_report_lavina_jayesh.pdf, [Aug. 6, 2018]

[2] S. Edwards. "Trusted boot: a key strategy for ensuring the trustworthiness of an embedded computing system." Internet: https://www.militaryaerospace.com/articles/2018/03/trusted-boot-embedded-computing.html, Mar. 28, 2018 [Aug. 6, 2018]

[3] "An Exploration of ARM TrustZone Technology." Internet: https://genode.org/documentation/articles/trustzone, [Jul. 31, 2018]

[4] "TrustZone - ARM." Internet: https://www.arm.com/products/security-on-arm/trustzone, [Aug. 6, 2018]

[5] "Industry Partnerships." Internet: https://globalplatform.org/about-us/industry-partners/, [Jul. 31, 2018]

[6] "Current Members." Internet: https://globalplatform.org/current-members/, [Jul. 31, 2018]

[7] "Introduction to Trusted Execution Environments." Internet: https://globalplatform.org/resource-publication/introduction-to-trusted-execution-environments/, May. 15, 2018 [Jul. 31, 2018]

[8] A. Williams. "What is... a Trusted Execution Environment (TEE)." Internet: https://www.electronicsweekly.com/blogs/eyes-on-android/what-is/what-is-a-trusted-execution-environment-tee-2015-07/, Jul. 2, 2015 [Aug. 6, 2018]

[9] Internet: https://github.com/ARM-software/arm-trusted-firmware, [Jul. 31, 2018]

[10] H. Raj, S. Saroiu, A. Wolman, R. Aigner, J. Cox, P. England, C. Fenner, K. Kinshumann, J. Loeser, D. Mattoon, M. Nystrom, D. Robinson, R. Spiger, S. Thom, D. Wooten. "fTPM: A Software-Only Implementation of a TPM Chip." Internet: https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_raj.pdf, Aug. 10-12, 2016 [Jul. 31, 2018]

[11] ""An in-depth look into the ARM virtualization extensions." Internet: https://genode.org/documentation/articles/arm_virtualization, [Jul. 31, 2018]

[12] G. Heiser. "Re: [seL4] sel4 in TrustZone." Internet: https://www.mail-archive.com/devel@sel4.systems/msg00742.html, Mar. 23, 2016 [Jul. 31, 2018]

[13] N. Feske. "Re: [seL4] sel4 in TrustZone." Internet: https://www.mail-archive.com/devel@sel4.systems/msg00750.html, Mar. 27, 2016 [Jul. 31, 2018]

[14] S. Wallentowitz. "[seL4] TEE and seL4, was: sel4 in TrustZone." Internet: https://www.mail-archive.com/devel@sel4.systems/msg00743.html, Mar. 24, 2016 [Jul. 31, 2018]

[15] "Intel SGX Details." Internet: https://software.intel.com/en-us/sgx/details, [Aug. 6, 2018]

[16] "AMD Secure Encrypted Virtualization (SEV)." Internet: https://developer.amd.com/amd-secure-memory-encryption-sme-amd-secure-encrypted-virtualization-sev/, [Aug. 6, 2018]

[17] J. Rajewski. "What is an FPGA?." Internet: https://alchitry.com/blogs/tutorials/what-is-an-fpga, Jan. 17, 2018 [Aug. 6, 2018]