Sindh Univ. Res. Jour. (Sci. Ser.) Vol.50(003)) 355-362 (2018)

http://doi.org/10.26692/sujo/2018.09.0060

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

Implementing Trust Computing Technique in Cloud Computing and Internet of Things (IoT)

M. NASEEM, S. A. KHAN, S. S. ZIA*, I. MALA**

Department of Computer Science, UBIT, Karachi University, Karachi- Pakistan

Received 08th November 2017 and Revised 24th June 2018

Abstract: Transition of all possible physical resources towards digital analysis has compliment the efficiency of resultant in more persuasive manner for its users. The data resources to become functional for its users are connected to cloud through network for further processing. Since the prevailing era is more focused in converging and forming smart environments that would enable us to live better lives and makes an efficient way out for resolving hitches faced in analytical phases. Although sometime digital expeditions could crop up with hazards as well in the form of privacy and security risks. Industry and academia are here with a list of trusted computing architectures to protect by painful behavior to our digital world, but these solutions have some limitation. This research provides the sketch of attestation and lightweight architectures from industry and academia, which are proposed to defeat the security issue. We compare various architecture with their security features and implementation/ usage which is necessary for the current hybrid system, also provide future direction of this study.

Keywords: Cloud Computing, Internet of Things (IoT), Hybrid System, Trust Computing, Attestation, Lightweight

1. INTRODUCTION

IoT is a system for dealing with the network of physical devices or sensor, while pervasive computing focuses on the system that is dealing with HCI (Human-Computer Interaction). But both the communities have shared the same goals and technical interests that is the convergence of everyday data to digital world in a manner that could made the environment more at ease (Ebling, 2016).

The world of wireless communications is changing rapidly and radically entering a new unexploited area. Mobile data traffic is growing faster than existing 4G networks. Though, any of these computing systems could be hacked by hackers, from a server hosted on cloud infrastructure to the ubiquitous or IoT devices, which can operate by micro-controller. Serious software defenselessness has found in commercial and non-commercial appliances (Proofpoint, 2014, Miller and Valasek, 2015, Falliere, *et al.*, 2011). Consequently the system will be no more upto the mark of the designer's expectation. For modern household appliance attackers send unwanted messages but researchers who have discovered the vulnerability of the vehicle have taken full control on it.

Academic and industrial research communities in IoT and pervasive computing are surfacing the real-time restraints from the hackers who could manipulate it wrong and made these technologies off putting in practice. The latest services that support user mobility, the security, privacy of multimedia and collaboration services are playing crucial part in everyday practices. Variety of trusted computing architectures is developed to provide manipulators assurances regarding the software performance on their devices and make it user trusted device(Martin and others, 2008). The devices always behave in general even if an attacker can control the system.

The differences between trusted computing and other object related with the term trust should be mention in (Gollmann, 2006). The core element of trusted computing is set of rule, also called Roots of Trust (RoTs), and the system's security should depend upon the usage of RoTs by the users, security failure occurred, if found any breaches in RoTs, Beside that, the unbreakable system security is achieved by improving the process of Security Development Lifecycle (SDL) (Lipner, 2004) and protect the OS and application from the attacker. Verities of the software-based and hardware-based trusted computing architectures are proposed by the industry and academia provides us interesting results in finite configuration.

Hardware-based architecture has a capability to protect the applications from the malicious operating system(OS), but software-based architecture cannot have this capability, because any application installed over the architecture could have been exploited by the attacker. Therefore, attacker easily performed amendment in the OS. Besides that, amendment in the OS is much difficult in hardware-based architecture.

++ Correspondence author: Muhammad Naseem, mnaseem105@gmail.com,

*Department of Computer Engineering, Sir Syed University of Engg. and Technology, Karachi- Pakistan.

**Department of Electrical Engineering, Usman Institute of Technology, Karachi- Pakistan.

Due to this positive feature lot of hardware-based implementation are proposed, but this solution has several limitations, like encryption key, Encryption algorithm, additional module etc. to overcome this problem some composite solution hasbeen proposed. These composite solutions have been implemented in term of hardware and software based architecture. The core of the trusted computing module is implemented over the hardware-based architecture and optional or additional module implemented on the software-based architecture.

IoT application developers should take into account the confidentiality, integrity, and credibility of data to help build trust with users and service providers. This confidence requires security and identity of the endpoint device, as well as low power, connectivity, and expandable cloud computing. The IoT solution accelerates the safety of SoC designers, equipment manufacturers, and developers by building a platform-specific security architecture that provides a powerful tool for the components needed to build the next system.

In a past decade, trusted computing play an active role in the research area, and a lot of practical and theoretical solution has been proposed for the infrastructure ranging from high-performance cloud computing system to lightweight embedded system. All of the research mainly provides the security solution specific to the technology or architecture like a cloud, IoT etc. No one provides the clear picture of security requirement or solution of hybrid technology. This research focuses on security mechanism which is suitable for hybrid technology like the interaction of IoT constrain device with the cloud.

This paper is organized as follows. Section 2 presents the requirements of hybrid system, while Section 3 presents the different architecture of trusted computing. In Section 4, we have discuss the comparison of different security architectural properties. Section 5 provides the conclusions and future directions of the researchers in this dimension.

2. REQUIREMENT OF HYBRID SYSTEM

Now the industry provides the next generation of IoT objects, and uniquely helps designers to build the right SoC units regardless of the type of devices class they build. Small devices which have limited power source, small CPU, and memory called "constrained devices" (typically used as actuators/sensors). These devices make a network for transferring information from sensor to destination, using the lossy channels with unpredictable bandwidth.

2.1. Constrained Devices

Constrained devices may be responsible for gathering information in different environments, including factories, building, ecosystems, manufacturing plants, and vehicles, and send to the cloud. They can also process information by implementing certain physical procedures, including the presentation of information. Restricted devices may operate under strict resource constraints such as battery power and limited computing, insufficient memory, insufficient wireless bandwidth and communication capabilities; these restrictions tend to intensify each other.

There are three types of constrained devices mention in (Bormann, *et al.,* 2014).

- Class 0
- Class 1 and
- Class 2

2.1.1. CLASS 0

Class 0 devices are very restricted like a sensor. It is very limited as a memory and processing power, and more likely that they do not have the capability to connect to the Internet. Class 0 devices will share Internet connections through large devices that act as agents, portals, or servers. Devices of type 0 cannot generally be protected or managed in a traditional way. They are likely to be pre-configured (and rarely remodeled, if any), which requires a very tiny data set. Besides that, it's have a feature send on/off, keepalive or acknowledge signal to the control side, for management purposes.

2.1.2. CLASS 1

Class 1 devices have very limited ROM, RAM and processing power capabilities. In addition, it'salso have a limited communication procedure to communicate with other nodes. Its use a special design lightweight communication protocol, such as CoAP over UDP, for performing meaningful communications to other nodes without using the gateway. It's also has a basic security module which fulfilled the requirements of the modern network. Finally, they need to save status RAM, ROM, and power consumption for protocols and applications usage.

2.1.3. CLASS 2

Class 2 devices are more intelligent than class 1 devices. It's have a capability to execute most of the communication protocol, which is supported by the desktop computer. On the other hand, it's also have a potential to consume minimum power and communication bandwidth. Therefore, class 2 devices are restricting the resource to improve interoperability and reduce implementation cost.

2.2. SECURITY PROPERTIES

The main security properties that are used in trusted computing are:

- Lightweight
- Attestation
- System on Chip

2.2.1. LIGHTWEIGHT

Lightweight this could explain the structure as not employing Memory Management Unit (MMU). Lightweight rooted structures have a modest memory hierarchy and hence do not need typical memory management. In addition, it works only on a certain number of applications that have shared memory space in general and do not need default memory processing.

2.2.2. ATTESTATION

Attestation is the process of certifying the licensing authority that a particular entity is in a particular case. For the provision of strong security assurances, the evidence-supporting structure must also ensure the integrity of the case. Reliable computing can be provided as local and distant certification. The local authentication unit is in the same memory structure as the operating system unit, while the remote authentication unit is located outside the system.

2.2.3. SYSTEM ON CHIP

System on Chip (SoC) is an integrated circuit that consists of CPU, memory, I/O ports, and storage on a single substrate. Besides that, it also has some additional hardware like Analog-to-Digital converters (ADC), Digital to Analog Converters (DAC), Radio Frequency System (RFS), Digital Signal Processor (DSP) etc. SoC is playing a very important role in the mobile computing market due to its low energy consumption.

3. ARCHITECTURES

In this section, which presents five isolation and attestation design, those have been modified on their target platform. Therefore, they do not include those architectures that are fully implemented in software. The selection of the design is covered, from the lightweight design of IoT objects to a node and cloud servers. The selected architectures do not only belong to industry and also play an important role in the academic research area.

3.1. TRUSTED PLATFORM MODULE (TPM).

In 2011, Trusted Computing Group (TCG) has designated Trusted Platform Module (TPM) 1.2 (Achemlal, *et al.*, 2011). It is a shared processor on the motherboard that stores keys and performs authentication. It is a passive device which means that the program can intermingle with the TPM, but it must be done explicitly. Bootloader, operating system, and application must be monitor by the TPM and it provides guarantee for authenticity to local or remote parties

.

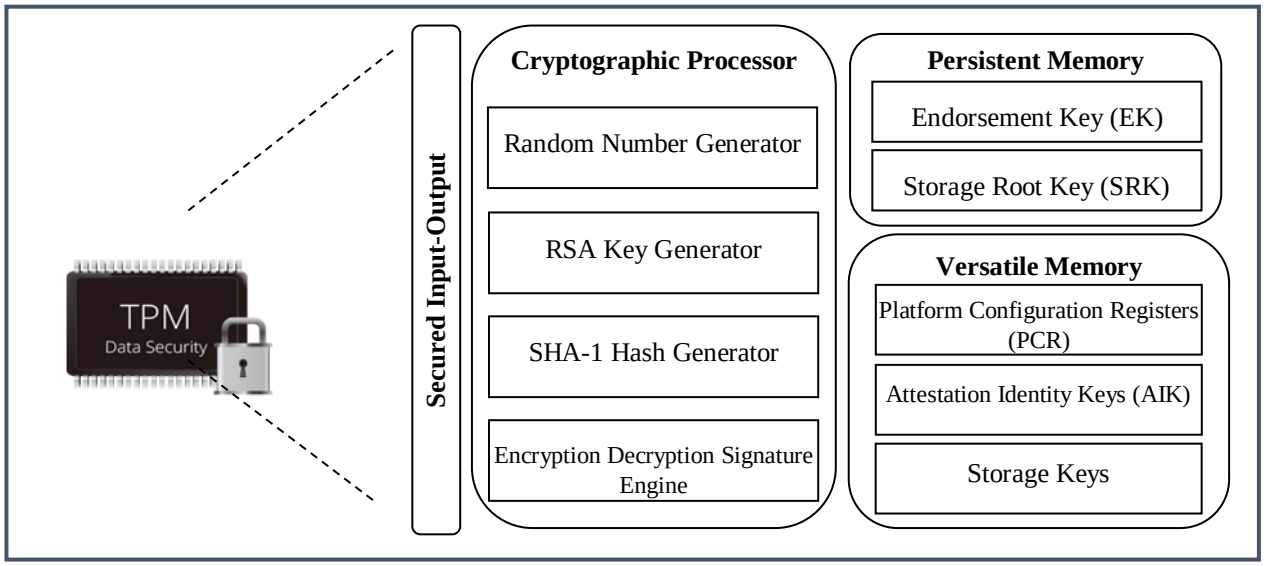

Fig. 1:Trusted Platform Module version 1.2

All module of the software is considered reliable after loading, any changes would be detected during measurement. Therefore, it cannot allow loading any new software components. This limitation is the biggest drawback of TPM, but to overcome this problem, Intel launch TXT (Trusted Execution Technology) (Grawrock, 2009). It runs the software components over the virtual environment with the TPM chip, easily measured any negative impact. Some architecture implement the functionality of TXT, like TrustVisor (McCune *et al.*, 2010), Flicker (McCune, *et al.*, 2008) and Fides (Strackx and Piessens, 2012).

3.2. TRUSTZONE

GlobalPlatform written Trusted Execution Environment (TEE) as standards used in industry to

pursue facilities of these security architecture (GlobalPlatform Device Technology TEE Client API Specification, 2010), (GlobalPlatform Device Technology TEE Internal API Specification, 2011). TEE has a secure zone for the processor and it's provide confidentiality, integrity and independent execution of trusted application resources.

The operating system provides an Untrusted Execution Platform (UEP) to access the resources. TEE is easily accessible the resources from the UEP, on the other hand, UEP does not access the resource in TEE, unless special permission. Therefore, only TEE resource accessed by another TEE resource. This is achieved through two hardwired modules. First, the AXI bus ensures that it is impossible to access secure global resources from the world's normal resources. Second, the kernel of possible processors from TrustZone uses time slots to execute secure or normal code in the world. This standard of TrustZone is an implemented in the ARM. TrustZone is an ARM application for this standard.

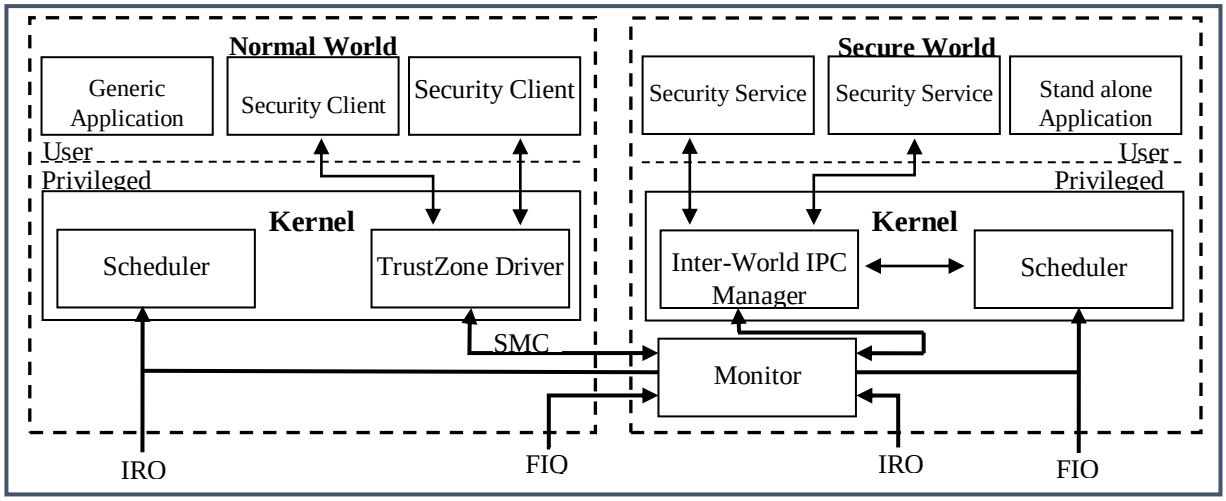

Fig. 2: TrustZone architecture proposed in(ARM, 2009)

Currently,a large number of Smartphone used a hardware-based security architecture, which is TrustZone (ARM, 2009). TEE is used to provide protection for software and hardware resources. In the first stage, TrustZone processor is initialized by trusted boot loader located at ROM and then load the second stage of the trusted bootloader, which is stored in flash memory. The second stage of trusted bootloader has the responsibility to initializes memory controller, peripherals and integrity check with the first bootloader. For more security, some trusted OS will perform integrity check with the trusted applications before initialized them. Its uses RSA-based signature schemes, vendor signs the code using own key, this signature verifies by firmware. In addition, different vendors are implemented TrustZone on a chip, using limited privileges.

3.3. BASTION

Bastion is a firmware architecture, which is a Trust-based management program. It ensures confidentiality and integrity check between the software and hardware module. Its only provides memory protection other than physical attacks.Unfortunately, the multi-core processor has not supported this architecture.

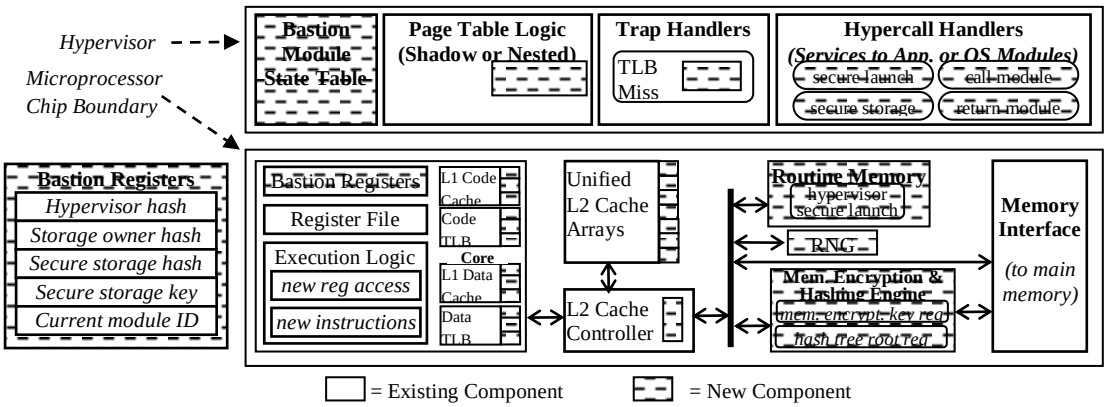

Fig. 3: Bastion Architecture proposed in (Champagne and Lee, 2010)

Bastion protects the structure of the hypervisor first, after that provides protection to software modules.For this purpose, secure_launch procedure calls by hypervisor, it's calculated hash value using data and code of hypervisor, generate new key for cryptographic functions and permanently stored in crypto engine's registerAfter loading the trusted hypervisor, again secure_launch procedure is initialized by software module to calculate hash of runtime memory including virtual memory and permanently stored on the disk. In order to invoke the security function, two more special hypercall module,call_module, and return_module are added to calculate the hash of access point of the target module and the similarly, on returning of restoring all state information.

3.4. SMART

Secure and Minimal Architecture for Root of Trust (SMART) (Eldefrawy *et al.,* 2012) specially design for minimal hardware and its provide Dynamic Root of Trust (DRoTs)in the remote firmware devices. It is the oldest designs to use a software signature for firmware to build a lightweight trust system. (Francillon, *et al.,* 2014) have enhanced its performance after a minor set of changes. Demonstrate the feasibility of the prototype based on open source versions of ATmega103 and openMSP430. It is very difficult for attacker to tempering this trusted system, because of the changes perform in the firmware. When you implement SMART, you must also disable any terminal device that can access memory directly.

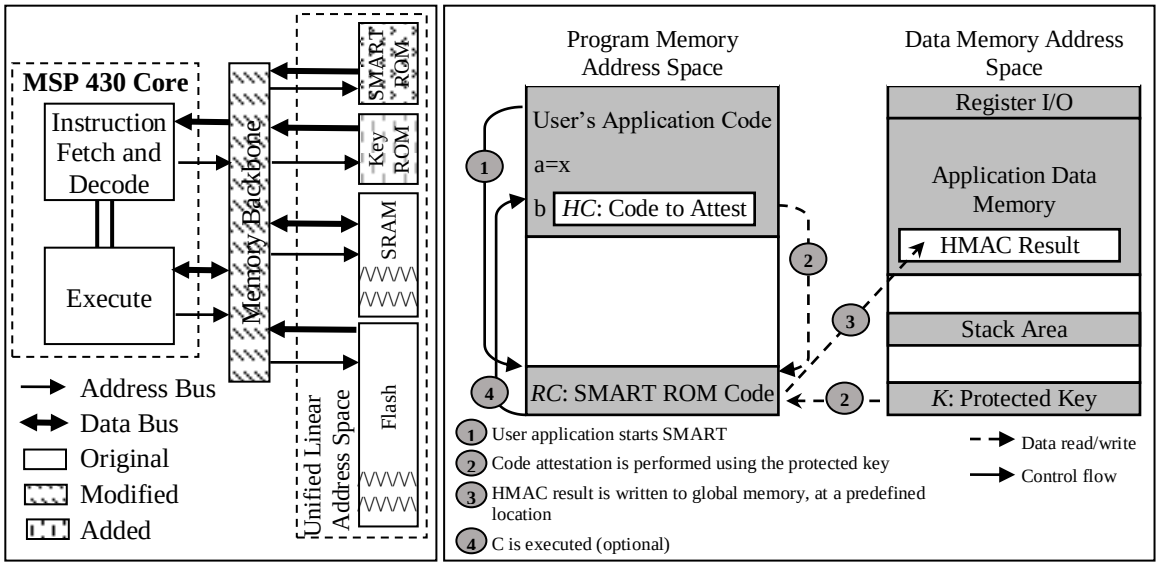

Fig. 4: SMART Computing Model proposed in (Eldefrawy *et al.,* 2012)

SMART typically provides a memory scope for remote authentication define by the checker. It's have four section, ROM for SMART, ROM for Key, SRAM for MCU and Flash for memory erase and reset.It'scalculated the hash of the specified memory location using SHA-256 and stores it in the ROM for verification. When the verification request called for remote authentication then it verify by the stored hash code. This process dynamically determines Root of Trust.

3.5. AEGIS

AEGIS is the oldest reliable computing architecture is designed by (Suh, *et al.,* 2003) in 2003. It has an ability to provide Tamper-Evident Environment (TE) with programs, which is very helpful to identify the physical and software tempering in the system memory. The privacy and reliable tamper resistant (PTR) environment provides stronger safeguards, besides that it also provides the confidentiality of code and data. Placement of external peripherals and memory outside the TCB, which protect CPU from hardware and software attack, the CPU itself should be trusted.

The attestation can be formed by calculating the hash of program with data, and it signs by the private key of the CPU. The operating system may be harmful, but when Secure Kernel (SK) to implement the hardwired AEGIS function, then operating systems must be reliable or trusted. Due to this role, this architecture is not completely implemented on-chip.

(Szefer and Lee, 2012) present a concept of HyperWall, which allows a hypervisor to freely manage the memory, processor, and other resources. It is created with the help of AEGIS.

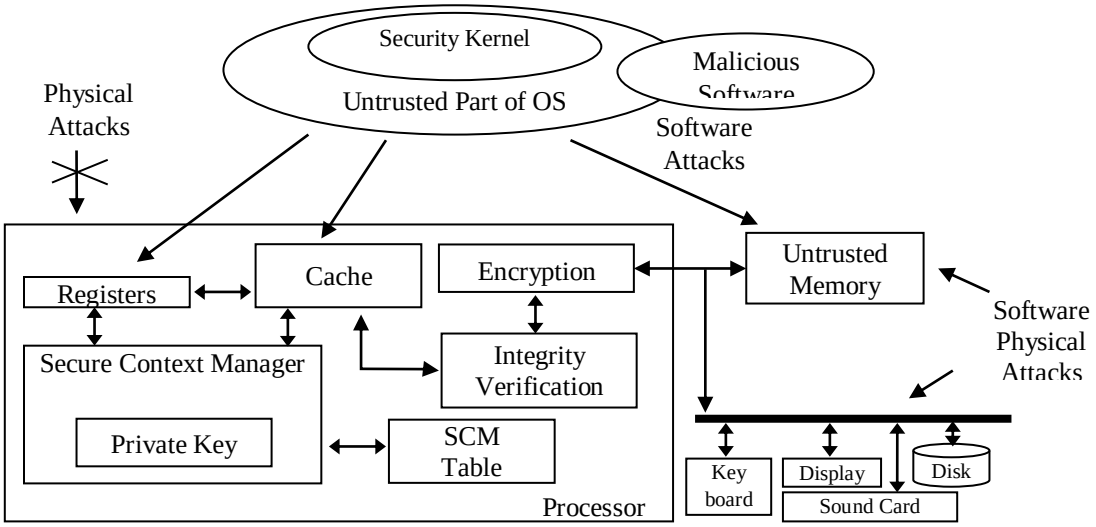

Fig. 5: AEGIS Computing Model proposed in (Suh *et al.*, 2003)

4. COMPARISON

Comparison of different security architectural properties is deal in this research. Table 1, demonstrate the complete photography of the security architecture mention in this research.

Table 1: Summarized detail of trusted computing architectures.

Architecture	Attestation	Lightweight	SoC	Cloud Support	Types of Devices (Bormann *et al.*, 2014)			Hardware	Firmware
					Class 2	Class 1	Class 0		
AEGIS (Suh *et al.*, 2003) (Szefer and Lee, 2012)	⊗	O	⊗	⊗	⊗	⊗	O	⊗	-
TPM (Achemlal *et al.*, 2011)	⊗	O	O	⊗	⊗	∅	O	⊗	-
TXT (Grawrock, 2009)	⊗	O	O	⊗	⊗	∅	O	⊗	-
TrustZone (ARM, 2009)	O	O	⊗	O	⊗	O	O	⊗	-
Bastion (Champagne and Lee, 2010)	O	O	O	⊗	∅	O	O	-	⊗
SMART (Eldefrawy *et al.*, 2012)	⊗	⊗	O	O	∅	O	O	-	⊗

⊗=Yes, ∅=Partial, O=No,

The designing of all mechanisms are a focus on the security property of attestation, except for TPM and SMART. All security mechanisms are easily implemented in the software but some of them have the capability to implement in firmware like Bastion and SMART.

For achieving the better performance, some of the security mechanism is specially designed for hardware implementation like AEGIS, TPM, TXT, and TrustZone,SMART is an example of a lightweight architecture, such designs have very simple memory hierarchies, therefore a limited number of applications can be executed. At the industry level, System-on-Chip (SoC) is hardware-based security architecture. TrustZone and AEGIS belong to the SoC.

Attestation protocol is implemented in software firmware and hardware based on symmetric or asymmetric key algorithms. Symmetric key attestation

is simple to implement in hardware but on the other hand implementation of attestation using asymmetric key `is complex. Another way to provide a sensible attestation procedure, it is partially implemented in software and partially in hardware another word it is called firmware e.g. SMART.

Table 1 shows that most of the architectures have a capability to implement over the class 2 devices and some of them support to the cloud. We can conclude that AEGIS is covered most of the features, therefore it is suitable for the hybrid system. Beside that TPM and TXT is the best architecture, but due to the limitation of memory and processing power, it partially implemented on class 1 devices.

5. CONCLUSION

New research challenges emerging from the convergence of IoT and cloud computing environment. No one can fulfill the complete requirement of the current hybrid system, because in the IoT domain Class 2 devices have a support to secure hybrid system by using trust architectures. But class 1 and class 0 have no support for it. There is opening two main areas for future research. The first line of research focuses on devices registration policy to decide how to secure class 0 and class 1 devices (based on their nature of usage, current environment, desires, etc), and how these policy are imposed on the devices.

The second area focuses on assemblage and their communications. These groupings are influenced by networks conditions the infrastructure capabilities. Therefore, we need to develop the supporting framework for the functions and acting as an interface between the cloud and IoT devices of class 0 and class 1.

REFERENCES:

Achemlal, M., S., Gharout, and C. Gaber, (2011). Trusted platform module as an enabler for security in cloud computing. In *Network and Information Systems Security (SAR-SSI), 2011 Conference on* 1–6.

ARM. (2009). ARM Security Technology. Building a Secure System using TrustZone Technology ARM. *ARM White Paper*, 108. Retrieved from http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf

Bormann, C., M. Ersue, and A. Keranen, (2014). *Terminology for Constrained-Node Networks*. *Internet Engineering Task Force (IETF)*. https://doi.org/10.17487/rfc7228

Champagne, D., and R. B. Lee, (2010). Scalable architectural support for trusted software. In *HPCA - 16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture* 1–12. https://doi.org/10.1109/HPCA.2010.5416657

Ebling, M. R. (2016). Pervasive Computing and the Internet of Things. *IEEE Pervasive Computing*, *15*(1), 2–4. https://doi.org/10.1109/MPRV.2016.7

Eldefrawy, K., A. A. Francillon, D. Perito, G. Tsudik, K. Defrawy, A. A El, Francillon, G. Sudik, (2012). SMART: Secure and Minimal Architecture for (Establishing a Dynamic Root of Trust. *Ndss*, *12*, 1–15. Retrieved from https://pdfs.semanticscholar.org/c265/ea208212d0f49ba93ce32c38b282b6982e5c.pdf

Falliere, N., L. O. Murchu, and E. Chien, (2011). W32. stuxnet dossier. *White Paper, Symantec Corp., Security Reponse*, *5*(6), 29Pp.

Francillon, A., Q. Nguyen, K. B. Rasmussen, and G. Tsudik, (2014). A minimalist approach to Remote Attestation. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014* 1–6. https://doi.org/10.7873/DATE2014.257

GlobalPlatform Device Technology TEE Client API Specification. (2010). Retrieved from http://www.globalplatform.org/

Gollmann, D. (2006). Why trust is bad for security. *Electronic Notes in Theoretical Computer Science*, *157*(3), 3–9.

Grawrock, D. (2009). *Dynamics of a Trusted Platform: A building block approach*. *Portal.Acm.Org*. Intel Press. Retrieved from http://portal.acm.org/citation.cfm?id=1610416%5Cnpapers2://publication/uuid/FA1AEF9F-3724-4055-94B7-643B33F20A1F%5Cnhttp://dl.acm.org/citation.cfm?id=1610416

GlobalPlatform Device Technology TEE Internal API Specification. (2011). Retrieved from http://www.globalplatform.org/

Lipner, S. (2004). The trustworthy computing security development lifecycle. In *Computer Security Applications Conference, 2004. 20th Annual* (pp. 2–13).

Martin, A., (2008). The ten page introduction to trusted computing. *Computing Laboratory, Oxford University Oxford*, *49*.

McCune, J. M., Y. Li, N. Qu, Z. Zhou, A. Datta, V. Gligor, and A. Perrig, (2010). TrustVisor: Efficient TCB reduction and attestation. In *Security and Privacy (SP), 2010 IEEE Symposium on* 143–158.

McCune, J. M., B. J. Parno, A. Perrig, M. K. Reiter, and H. Isozaki, (2008). Flicker: An execution infrastructure for TCB minimization. In *ACM SIGOPS Operating Systems Review* Vol. 42, 315–328.

Miller, C., and C. Valasek, (2015). Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 91.

Proofpoint, I. (2014). Proofpoint uncovers internet of things (iot) cyberattack.

Strackx, R., and F. Piessens, (2012). Fides: Selectively hardening software application components against kernel-level or process-level malware. In *Proceedings of the 2012 ACM conference on* 2–13. https://doi.org/10.1145/2382196.2382200

Suh, G. E., D. Clarke, B. Gassend, , M. van Dijkand S. Devadas, (2003). AEGIS: Architecture for Tamper-evident and Tamper-resistant Processing. In *Proceedings of the 17th ACM Annual International Conference on Supercomputing (ICS)* 357–368.

Szefer, J., and R. B. Lee, (2012). Architectural support for hypervisor-secure virtualization. In *ACM SIGPLAN Notices* Vol. 47, 437–450.