DYNAMIC ROOT OF TRUST AND CHALLENGES

Sandeep Romana, Himanshu Pareek and P R Lakshmi Eswari

Center for Development of Advanced Computing,
Hyderabad, India

ABSTRACT

Trusted Computing intends to make PC platform trustworthy so that a user can have level of trust when working with it. To build a level of trust TCG gave specification of TPM, as integral part of TCB, for providing root(s) of trust. Further TCG defined Dynamic Root of Trust Measurement in Trusted Computing systems in its specification as a technology for measured platform initialization while system is in running state. The DRTM approach is contrary to Static Root of Trust Measurement where measurements are taken during boot process. In this study, since this technology was first introduced, we list and discuss upon publically available open source solutions that either implement DRTM or are applications of these DRTM based solutions. Further, the challenges faced by the DRTM technology along with observations from authors are listed.

KEYWORDS

SRTM, DRTM, TPM, TCB

1. INTRODUCTION

When working with commodity PC platform we need a level of assurance if system is executing the code that we are instructing it to execute; and not anything else. Thus, there is a need for building a level of trust on these commodity PC platform(s). For having a level of trust, Trusted Computer Systems Evaluation Criteria (TCSEC) defined the Trusted Computing Base (TCB) of a computer system as the part of the system (hardware, firmware, software and /or other) that is critical to its security and whose failure (due to bugs or vulnerabilities or any other reason) may lead to compromise of the system [1]. TCB is now a fundamental concept in computer security. It is the portion of the system that is relied on to enforce the security policy of the platform [4]. Trust is the expectation that a device will behave in a particular manner for specific purpose [4], [5]. In order to trust something we should be able to measure it and compare against known good values. To achieve this we have Trusted Platform Module (TPM), as an integral part of TCB, by Trusted Computing Group (TCG), which is piece of hardware (for strong protection of platform-specific unique secret key) attached to PC motherboard [5], [3]. The TCG is a de facto standards body of nearly 140 international companies. TCG is mainly working for creating specifications that define PC TPMs, trusted modules for other devices, trusted infrastructure requirements, APIs and protocols necessary to operate a trusted environment [6]. TPM is intended to provide three roots of trust i.e. a) Root of Trust for Measurement (RTM), b) Root of Trust for Storage (RTS), and c) Root of Trust for Reporting (RTR) [5].

TCG defines RTM as a trusted implementation of hash algorithm which is responsible for the first measurement on a platform. This measurement can be done at a boot time or later time to put platform into trusted state. RTM where entire boot chain is measured is defined as Static RTM (SRTM) [9], [5]. As described in [4], [11], [7], [8], maintaining a chain of trust for a length of time is challenging because boot chain is very long and thus large amount of code needs to be measured for building trust and some elements may change due to updating (e.g. boot loader configuration). To address this challenge posed by SRTM, TCG provides specification for Dynamic Root of Trust for Measurement (DRTM).

The purpose of the DRTM is to reduce the complexity of the TCB so that evaluation of the platform state is easy to deal with [4]. The DRTM allows launch of the measured environment at any time without a platform reset (i.e. at runtime). In DRTM, the chain of trust starts by invocating special instruction which resets the TPM PCR (Platform Configuration Register) from 17 to 22 [4]. This eliminates the need for platform reset for trust establishment [4]. The support for this special instruction(s) is available with commodity microprocessors such as Intel - with Trusted Execution Technology (TXT) [9] and AMD with Secure Virtual Machine (SVM) [10]. In DRTM, a cryptographic hash of a code is calculated before execution. The hash is extended into a PCR on TPM chip after necessary initializations [4]. The DRTM process by making use of TPM solves the purpose of establishing a TCB that can either unseal data or attest platform state to remote entity. An interested reader can further refer to [4] for details on DRTM process.

2. DRTM BASED IMPLEMENTATIONS

This section lists publically available open source implementations of dynamic root of trust measurement on commodity PC platforms, along with the open source applications of these implementations, known to the authors.

2.1. OSLO

OSLO stands for Open Secure Loader which is a bootloader and is the first implementation of DRTM using SKINIT instruction on AMD platform. Kauer presents a) the security analysis of bootloader bugs in trusted bootloader that was part of the Bear project from Dartmouth College and GRUB v0.97 from IBM Japan, b) TPM reset problem with v1.1 TPM which resets the chip without resetting the whole platform and c) BIOS attacks with HP nx6325 business notebook with a TPM v1.2. OSLO bootloader implementation has 1500 lines of code approximately and shortens the trust chain by eliminating BIOS and bootloader from TCB and establishes DRT for trusted boot, thus overcoming the listed bugs with SRT based Trusted Computing Systems (TCS) [11]. OLSO opened a new door of opportunity by giving the implementation of DRTM based solution and laid the foundation for further research.

2.2. Flicker

Taking OSLO as the starting point, Flicker architecture provides with 'meaningful attestation' and 'minimal TCB' in addition to 'isolation' and 'provable protection' provided by AMD SVM and Intel TXT. For this, flicker leverages DRTM to execute security sensitive part of application in an isolated environment with extremely small TCB of 250 lines approximately. This system allows Piece of Application Logic (PAL) to execute in isolated environment and don’t require Virtual Machine Monitor (VMM) or OS [12]. There are two applications of flicker available in open source and known to the authors of this text, i.e. Softer Smartcards (or soft cards) [13] and Bottle Cap [14]. The next two subsections list each of the applications built over flicker framework.

2.3. Soft Cards

Soft cards are cryptographic tokens which use ‘secure execution technology’ for providing protection against software attacks and basic protection from hardware attacks. They are virtual PKCS 11 (public-key cryptography standards) tokens running in software which makes use of 'trusted execution' for assurance of their secure and isolated execution. Soft cards approach is a compromise between tokens with dedicated card reader and software-based solutions. The prototype presented is usable with applications such as enterprise level single sign-on solutions, authentication in virtual private networks (VPNs), e-Mail and WiFi clients and password managers. Softer Smartcards leverage the use of Flicker system along with Open Cryptoki for this purpose [13].

2.4. Bottle Cap

Bottle Cap binds the capabilities (for e.g. password capability) to the machine to which it is issued while holding secrets in sealed storage. Users can use the rights represented by these capabilities but can only uncover the secrets of these capabilities under the supervision of Bottle Cap. This helps to contain the issue of uncontrolled rights propagation. This system extends the functionality of flicker for achieving this. Bottle Cap attempts to solve two problems, a) 'infinite delegation' or capability duplication arising from accidental or malicious leakage of password capability, and b) 'space-exhaustion denial of service attacks' on server-side systems arising from storing large number of capabilities on a system [14].

2.5. Trust Visor

To improve upon the performance bottleneck and other issues with flicker, McCune at el introduced Trust Visor, which is a special-purpose hypervisor providing code integrity as well as data integrity and secrecy for user space Pieces of Application Logic (PALs). Trust Visor implements software based micro-TPM which executes at high speed on CPU. This improves the performance significantly from flicker system where TPM falls in the critical system path. The size of TCB with Trust Visor is approximately 6351 lines of code, which is greater than flicker but still significantly smaller than the commodity VMMs [15].

2.6. Tboot

Rather than research project as with the OSLO, Flicker and its applications (i.e. Soft Cards and Bottle Cap) and Trust Visor; Trusted Boot (tboot) is a pre-kernel/VMM module maintained in open source and actively developed and/or supported by developers from Intel. It uses Intel TXT to perform a measured and verified launch of an OS kernel/VMM. Tboot can function as a bootloader and launch whole operating system in protected or measured environment. It includes tools for Intel TXT Launch Control Policy (LCP) creation and provisioning [16], [9]. Tboot supports Xen hypervisor and Linux based distributions such as Ubuntu, Fedora etc. [16], [17].

3. CHALLENGES

This section list the author's observations and other challenges faced by above listed DRTM implementations. Along with using this it lists the challenge from attacks that are available on DRTM enabling technologies (mainly with Intel TXT).

3.1. Attacks on Intel TXT

Rutkowska et al demonstrated that Intel TXT is vulnerable to System Management Mode (SMM) attacks on Intel microprocessors [19]. SMM is known to be the most privileged mode on these microprocessors. Rutkowska et al presented another approach to misconfigure the chipset so that GETSEC [SENTER] instruction is unable to properly setup VT-d protections [20], which however is now patched. Rutkowska et al further demonstrated attack on Intel TXT via SINIT modules [21]. This concludes to that, the implementations of technologies that enable DRTM can contain bugs (as with other software's) and can be exploited [19].

3.2. OS Support

While flicker demonstrates the capability to interlude the executing specific versions of Linux kernel (as well as Microsoft Windows) and resuming the kernel after executing security sensitive code in complete isolation, the methodology is not portable to next generation kernels without in-depth knowledge of changes in the kernel and platform specific implementation details [18]. Thus support from the operating system (OS) developers is required to enable such facility.

3.3. Performance

As with flicker, during execution, the OS appears to be in hung state. This is due to the slow TPM chip being added to systems critical path. Hence, this kind of set up is not usable in production environments where performance expectations are very high [12], [15].

3.4. Stability

Interesting observation authors made is that the operating system(s) especially the Microsoft Windows after a flicker session (i.e. when the Windows kernel resumes) is highly unstable [18] and the causes of this un-stability are unknown till the date of writing. This may partially be due to lack of proper support and documentation from the vendor for such implementations and may be partially due to lack of inability to gain in-depth knowledge to implement such code because of closed source nature of Windows code.

3.5. Unwelcome security setup

The Trust Visor's hypervisor approach tried to overcome the shortcomings with flicker, but it requires at least two operating systems to be present on the same system (one as a host and/or hypervisor and others as guest(s)). This kind of setup is desirable and useful in cloud based environments, whereas in security sensitive environment's it is highly undesirable to have such as setup as the two (or more than one) operating systems can be completely incompatible and unknown about each other's security enforcement mechanisms, thus lowering the overall effective security of each operating system.

4. Related Work

OSLO, Flicker [12], and Trust Visor [15] are the solutions that implement DRTM. All these, starting from OSLO and then flicker till Trust Visor, improved upon one another in terms of performance, size of TCB and/or feature set. Further Rutkowska et al in [19], [20] and [21] presented the attacks on Intel's implementation of DRTM i.e. Intel TXT. Apart from the solutions listed above we came across a single publication published in 2007 by Nie which analyses the implementation problems with SRT based solutions and the size of TCB and, in parallel mentions the two DRTM based solutions (OSLO and SEA/Flicker), that were present at that time, to the mentioned problems [7]. As the DRTM technology has evolved over a period of time, more solutions have been implemented to make use of it. Our work attempts to provide a more nearly complete picture of it and its applications along with the challenges faced by it in the current scenario.

5. CONCLUSION

In this text we listed the solutions that implemented Dynamic Root of Trust Measurement and applications over these solutions that extend the functionality of the listed solutions. The evolution of solutions that implement DRTM have been; firstly, on the basis of efforts to reduce the size of Trusted Computing Base and secondly, on removing the slow Trusted Platform Module hardware from systems critical path. Finally, Trust Visor provides hypervisor approach to reducing size of TCB and removing TPM from systems critical path with software based micro-TPM but at the expense of having multiple operating systems on a single machine.

To conclude this text, we would like to mention that with the hypervisor approach to DRTM; it can be a useful technology in cloud environments, but in desktop based environments it has its own security implications. With the list of available attacks on Intel TXT it is clear that it is a challenge to have bug free implementations of such technologies. DRTM based technology, especially for desktop based environments, have challenge of support from commodity operating system providers and slow TPM hardware that remain unsolved till date.

ACKNOWLEDGEMENTS

The authors would like to thank everyone, just everyone!

REFERENCES

[1] Latham, Donald C. "Department of Defense trusted computer system evaluation criteria." Department of Defense (1986).
[2] David Challener, Kent Yoder, Ryan Catherman, David Safford, and Leendert Van Doorn. 2007. A Practical Guide to Trusted Computing (First ed.). IBM Press.
[3] A. Martin, "The ten page introduction to trusted computing," Technical Report RR-08-11, OUCL, December 2008, http://web.comlab.ox.ac.uk/files/1873/RR-08-11.PDF
[4] TCG D-RTM Architecture, Version 1.0.0, June 2013, http://www.trustedcomputinggroup.org/files/static_page_files/5DB17390-1A4B-B294-D029166C91F3512B/TCG_D-RTM_Architecture_v1\%200_Published_06172013.pdf
[5] Trusted Computing Group: TCG Specification Architecture Overview, revision 1.4 (August 2, 2007), http://www.trustedcomputinggroup.org/
[6] Trusted Computing Group, Trusted Platform Module (TPM) Summary, http://www.trustedcomputinggroup.org/resources/trusted platform module tpm summary
[7] Nie, Cong. "Dynamic root of trust in trusted computing." TKK T1105290 Seminar on Network Security. 2007.
[8] Butterworth, John, et al. "Problems with the Static Root of Trust for Measurement."
[9] Intel® Trusted Execution Technology (Intel® TXT), Measured Launched Environment Developer's Guide, May 2014.
[10] Advanced Micro Devices, AMD64 Architecture Programmer's Manual Volume 2: System Programming, Publication no. 24593, Revision 3.23, May 2013.
[11] Bernhard Kauer. 2007. OSLO: improving the security of trusted computing. In Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium (SS'07), Niels Provos (Ed.). USENIX Association, Berkeley, CA, USA, Article 16, 9 pages.
[12] Jonathan M. McCune, Bryan J. Parno, Adrian Perrig, Michael K. Reiter, and Hiroshi Isozaki. 2008. Flicker: an execution infrastructure for tcb minimization. In Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems 2008 (Eurosys '08). ACM, New York, NY, USA, 315-328. DOI=10.1145/1352592.1352625 http://doi.acm.org/10.1145/1352592.1352625
[13] Brasser, Franz Ferdinand, et al. "Softer Smartcards." Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2012. 329-343.
[14] Justin King-Lacroix and Andrew Martin. 2012. BottleCap: a credential manager for capability systems. In Proceedings of the seventh ACM workshop on Scalable trusted computing (STC '12).

ACM, New York, NY, USA, 45-54. DOI=10.1145/2382536.2382545 http://doi.acm.org/10.1145/2382536.2382545

[15] Jonathan M. McCune, Yanlin Li, Ning Qu, Zongwei Zhou, Anupam Datta, Virgil Gligor, and Adrian Perrig. 2010. TrustVisor: Efficient TCB Reduction and Attestation. In Proceedings of the 2010 IEEE Symposium on Security and Privacy (SP '10). IEEE Computer Society, Washington, DC, USA, 143-158. DOI=10.1109/SP.2010.17 http://dx.doi.org/10.1109/SP.2010.17

[16] Trusted Boot (TBOOT), http://sourceforge.net/projects/tboot/

[17] Joseph Cihula, Trusted Boot: Verifying the Xen Launch, http://www-archive.xenproject.org/files/xensummit_fall07/23_JosephCihula.pdf

[18] Flicker: Minimal TCB Code Execution, Mailing Lists, http://sourceforge.net/p/flickertcb/mailman/

[19] Wojtczuk, Rafal, and Joanna Rutkowska. "Attacking Intel trusted execution technology." Black Hat DC (2009).

[20] Wojtczuk, Rafal, Joanna Rutkowska, and Alexander Tereshkin. "Another way to circumvent Intel trusted execution technology." Invisible Things Lab (2009).

[21] Wojtczuk, Rafal, and Joanna Rutkowska. "Attacking Intel TXT via SINIT code execution hijacking." ITL: http://www.invisiblethingslab.com/resources/2011/Attacking_Intel_TXT_via_SINIT_hijacking.pdf (2011).

AUTHORS

Sandeep Romana obtained his B.Tech from Punjab Technical University and MS in Software Systems from Bits Pilani and is a CISSP. He has more than 8 years of experience in the field of research and development of security software for desktop's and small networks. His areas of research include malware detection and analysis.

Himanshu Pareek received his B.E. degree in 2005 from University of Rajasthan and M.S. By Research degree in 2013 from JNTU, Hyderabad. He has around nine years of experience in developing and design of end point and gateway based security solutions. His main research interests include malware detection and data science.

Mrs P.R.Lakshmi Eswari, is presently working as Joint Director, e-Security R&D, C-DAC Hyderabad. She is currently involved in the Research & Development of end system security solutions focusing on anti-Malware & device control solutions. As an outcome of R&D efforts solutions like USB Pratirodh, AppSamvid, Browser JS Guard, Malware Resist etc. are developed by their team.