Applications and Challenges in Securing Time

Fatima M. Anwar
UCLA

Mani Srivastava
UCLA

Abstract

In this paper, we establish the importance of trusted time for the safe and correct operation of various applications. There are, however, challenges in securing time against hardware timer manipulation, software attacks, and malicious network delays on current systems. To provide security of time, we explore the timing capabilities of trusted execution technologies that put their root of trust in hardware. A key concern is that these technologies do not protect time integrity and are susceptible to various timing attacks by a malicious operating system and an untrusted network. We argue that it is essential to safeguard time-based primitives across all layers of a time stack – the hardware timers, platform software, and network time packets. This paper provides a detailed examination of vulnerabilities in current time services, followed by a set of requirements to build a secure time architecture.

1 Introduction

Applications such as telecommunications, satellite navigation, banking systems, Internet of things, and many others are underpinned by accurate timing. An attacker may incentivize by compromising time on these systems. The incentive can be monetary [15], location based [20], cryptographic keys [18], or copyright theft: an attacker can 1) cause trade time discontinuities for illegal trading activities especially in high frequency trading [15], 2) move time backward for digital rights management and movie rentals, 3) disrupt temporal forensic analysis by violating causality [5], and 4) prove to be physically present at a place where it is not [20]. An adversary can also cause complete system shutdown by manipulating the time of Phasor Measurement Units (PMUs) in smart grids [19], and distributed servers in self-programming networks [8]. A malicious host may degrade system performance and user experience by increasing user perceived delays [17].

A time stack provides timekeeping and timestamping capabilities to all systems, and consists of three major components: hardware timers that count oscillations, software that maintains time, and network packets that carry time information. A compromised operating system (OS) can manipulate timer registers, a host can lie about time [2], and a network attacker can delay time packets [7]. We argue that it is imperative to protect all layers of time stack – the hardware timers, system software, and the network packets.

Critical functionalities for securing unmodified binaries in Haven [2], Panoply and Graphene-SGX rely on untrusted system time. A malicious OS may lie about the time or signal early timeouts. Traditional cryptographic techniques, trusted execution technologies, and network security mechanisms may guarantee data security but do not provide timing integrity. Some techniques such as fTPM [16] propose secure clock implementations on trusted execution environments, but it relies on OS acknowledgements for clock writes, and are susceptible to attacks in the presence of untrusted OS and network.

In this work, we examine in detail the timing capabilities of various Trusted Execution Environments (TEE) such as Intel SGX, ARM TrustZone, and Trusted Platform Module (TPM), and establish that the mechanisms provided by these TEE are insufficient to establish timing integrity. For example, Intel SGX provides access to a trusted timer in the Converged Security and Manageability Engine through secure platform services [4]. A privileged adversary like OS is not able to manipulate this trusted timer. We establish that SGX trusted time can still be attacked, and the SGX community verifies our claim. SGX platform service transfers trusted time packets over a secure session through OS inter-process communication [4]. As these packets are encrypted and integrity protected, OS cannot change the packet's content. It can however delay these packets. We mount a delay attack in the OS, and its consequence is a time value that cannot be trusted as it is not received in a timely manner. The violation of timeliness of trusted time packets gives a wrong perception of elapsed time durations to applications. The failure to measure fixed durations affect many applications and result in unstable system operation.

We also put forward other categories of timing attacks in TEE such as scheduling attacks launched by a compromised OS and time transfer attacks from a malicious network element. In the presence of these attacks, we provide a set of requirements that guide the design and implementation of a secure time architecture.

2 Applications of Trusted Time

Clocks are considered synchronized if their time is aligned with each other. It has long been established that synchronized clocks in distributed systems improve performance by reducing complexity. With the Internet of Things (IoT), health care, connected vehicles, digital assistants, and augmented/virtual reality bringing a massive amount of data, there is an ever increasing need of cloud-based services that guarantee a rapid response on recent data and scale to distributed servers. The realization of these services is not possible without understanding temporal characteristics of the data. The emerging temporal use cases of "delayed use" or "schedulable demand" in cloud services push towards the trend in data warehouse products to support real-time applications and time-indexed queries [14]. It is imperative that services performing temporal forensic analysis on data should be temporally trusted.

Fine-grained network measurements using packet timestamps enable networks to monitor themselves and requires the distributed servers in the network to be securely synchronized in time. ShieldBox and Slick [22] provide secure middlebox framework for deploying network functions over untrusted commodity servers using SGX enclaves. For line rate processing in middleboxes, fine-grained cycle level measurements are made inside enclave via Network Interface Card's clock as it can be read reasonably fast. However, as noted by the authors, this clock source is not secure against OS attacks, and providing precise trusted time remains an open problem [22]. Other systems such as SCONE, Haven [2], and Panoply also rely on untrusted OS time.

There is a need of trusted timestamping for certificate revocation, Digital Rights Management (DRM), and to preserve the creation time of patents, digital documents, electronic commerce or even virtual patents [3]. Unfortunately, clocks are subject to rollback attacks [12] where time goes backward, replay attacks where time doesn't move forward, and delay attacks where time durations are dilated or compressed. Mere encryption cannot prevent these attacks.

Interactive mobile applications that rely on cloud services also benefit from the trusted notion of time. These mobile applications expect a timely response to their requests. The application developers have to ensure that the end-to-end delays do not exceed a particular value. Timecard [17] calculates 'elapsed time' since the initiation of a request and predicts 'remaining time' that servers tune its 'work time' to meet the required deadline. An untrusted time can impact application performance.

There are specific applications such as grid monitoring, precision docking, and high frequency trading, where time attacks result in catastrophic consequences. Astronomical applications such as Quantum key distribution from satellite to ground [10], and distributed telescopes' black hole imaging [1] require high precision time synchronization – in the order of picoseconds – hence the use of exclusive and secure atomic clocks is recommended. In short, the correctness and safety of numerous applications rely on a trusted notion of time.

3 Challenges in Providing Secure Time

Timekeeping and timestamping capabilities are essential for all systems. Therefore, every system maintains a time stack with three discrete components: hardware timers, timekeeping software, and time transfer network packets. We start by defining our threat model followed by the attacks on time.

3.1 Threat Model

We do not trust the OS and hypervisors as they can be easily corrupted. Hence we do not trust the correctness of the received timing information from the underlying system. Note that the OS is a privileged adversary and can manipulate all timers on a device. Network entities are also not trustworthy. GPS time packets can be spoofed and NTP packets can be attacked. A local or a remote attacker is capable of delaying, dropping, replaying and forging time packets, thus it can hold time, or move it backward or forward. We do trust the hardware execution environments from ARM and Intel.

3.2 Attacks on Time Stack

A compromised OS or a network element manipulates time by attacking components at all layers of the stack.

Hardware timers are not considered secure if a malicious software is able to write to their registers. In Intel architecture, RDTSC/RDTSCP are instructions that are used to read the TimeStamp Counter (TSC), but these instructions are not immune to influences by privileged software , e.g. TSC can be written to by the OS. Similarly, High Precision Event Timer (HPET), Programmable Interval Timer (PIT), and Advanced Programmable Interrupt Controller (APIC) timer can be controlled by the OS. The timers on ARM and other architectures face the same issue. Hence a privileged software is capable of adding discontinuities in time. To protect hardware timers against a malicious software, it is essential to restrict timer writes only to trusted entities.

Software maintained by the OS is responsible for timekeeping i.e. convert hardware timer ticks to a human understandable time. A malicious OS may lie about time, signal timeouts early or late, or gradually change the notion of time for applications to deceive them. It can also schedule out the timekeeping threads for an indefinite period of time. As a result, we no longer trust the timekeeping, scheduling, and timeout services provided by an OS.

Network based time synchronization mechanisms are responsible to align clocks at different devices connected over the network. These time synchronization mechanisms are also not safe from attacks. Network services such as GPS and Network Time Synchronization Protocol (NTP) are used to synchronize local client time to global time. GPS provides accurate Position Navigation and Time (PNT) information to almost all the civilian infrastructure. However GPS spoofers can manipulate GPS derived UTC time and cause damage to various applications in smart grids, financial market, and autonomous agents. Martyn Thomas urges for a backup timing system for GPS because of heavy reliance of critical infrastructure on GPS, from precision docking of oil containers to high tech farming [21]. eLoran is considered as a PNT replacement for GPS but it still suffers from low accuracy [9].

As time is a critical service needed at all devices at all time, most digital devices use NTP [13] to synchronize their local clocks with the global time. NTP clients exchange messages with Time servers in the network achieving 100's of milliseconds time synchronization accuracy. As discussed in [6] and [11], an attacker can establish Man in the Middle (MitM) capabilities for the NTP packets in the network. It can delay, drop, replay, and forge the NTP packets. Recent cryptographic techniques incorporated to NTP have eliminated most of these attacks, however delay attacks on NTP packets are considered too strong to protect against [11]. There exist solutions that offer to mitigate delay attacks on NTP packets under the assumption that half of the time servers and network links should not be compromised, but an adversary sitting on a gateway router can easily manipulate and delay all NTP packets for a particular client.

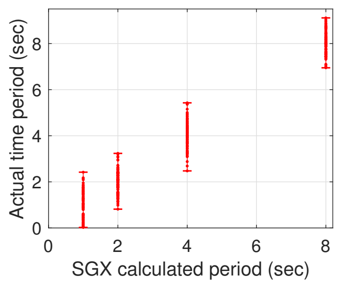

Figure 1: Delay attacks by an OS distorts SGX time periods as compared to actual time periods

3.3 Timing capabilities of Trusted Execution Environments

The security features of ARM TrustZone and Intel SGX are limited to the boundary of a CPU with limited support of secure peripherals such as clocks. There exists Trusted Platform Module (TPM); a trusted microcontroller / crypto coprocessor that provides secure peripherals such as secure storage, clock, and entropy.

ARM TrustZone has its own implementation challenges when it comes to secure peripherals. Memory and interrupts of different I/O peripherals in ARM TrustZone can be mapped to the 'secure world'. Unfortunately, the controller of the peripheral can be programmed by the 'normal world' OS making the peripherals open to different kinds of attacks [16].

Intel SGX trusted time service provides time value in seconds with reference to a random epoch. It is good for calculating time periods to enforce certain policies. However, it has coarse resolution, random epochs, and no sense of absolute time [4]. Hence, at any point in time, all enclaves have a different sense of time, which is not desirable for time-triggered applications. For example, data sampled from multiple trusted sensors placed on distributed enclaves yield timestamps that cannot be correlated. SGX trusted time is accessed via encrypted inter-process communication [4] managed by a compromised OS. We know that the OS cannot attack the encrypted and integrity protected packet, we show however that the OS can delay this packet. Delaying violates the timeliness of SGX time value hence distorting the sense of time for application enclaves. We delay all SGX time packets with a random value sampled from a uniform distribution of 0 to 1 seconds, and are able to accumulate an error in the order of multiple seconds as shown in Figure 1. We provide the source code to reproduce this attack at SGX Delay Attack. The use cases of sampling multiple trusted sensors at the same time, or sampling sensors in their allotted time slots are also affected by these attacks, thus compromising all applications in need of secure time.

Trusted Platform Module (TPM) mandates a millisecond resolution clock since the TPM boot, and provides a reliable indication when it has been reset. The TPM time is written to a non volatile memory every 2^{22} milliseconds that could possibly help in protecting against rollback attacks [12], where an adversary moves the clock back in time. The effective resolution of the TPM clock is reduced when it is accessed from an application with an access latency of almost 32 milliseconds that is three times more than SGX time latency of few milliseconds. Because of the limited TPM resources, multiple applications cannot access it frequently, further decreasing its effective resolution. The TPM clock is mostly used to timestamp internally stored keys and data, and not designed for timestamping network packets or sensors data etc. Thus the TPM clock does not satisfy the needs of broader applications.

4 Secure Time Architecture

After putting forward the timing capabilities of various TEE, we are now in a position to formalize three broad categories of timing attacks. In our threat model, we trust the TEE by different vendors such as ARM Trustzone, and Intel SGX. We do not trust the OS and hypervisors as they can be corrupted. Though a TEE provides access to a secure hardware timer, there are still three possible attacks on this trusted timer: As shown in Figure 2, (1) *Scheduling attack* is launched by a compromised OS, where it maliciously schedules out the timekeeping threads for indefinite period of time, thus manipulating the rate at which time advances. (2) *Delay attack* intercepts all transmitted and received time packets and maliciously add delays in packet delivery thus violating timeliness of packet arrival. The attacker has the choice to delay a packet by any arbitrary value thus distorting the sense of time for an application.

The final category of attack in Figure 2 is the (3) *network attack*. We successfully launch attacks on time synchronization packets that contain global time information from the time servers in the network. These packets are either delayed by a compromised router, or the global time information inside the packets is corrupted due to a compromised time server. We consider a network attacker that can compromise all time servers and the links to those servers.

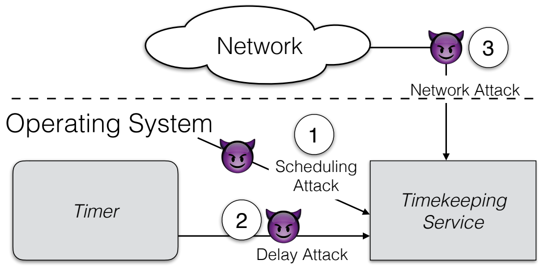

Figure 2: A secure clock should mitigate three categories of attacks: (1) OS scheduling attack on timekeeping service, (2) OS delay attack on time access, and (3) Network attack on global time synchronization packets.
Note: We assume that the 'Timer' and 'Timekeeping Service' are within hardware protection

A secure time architecture detects and mitigates these three categories of attacks. Key requirements that guide the design of this architecture are, 1) a high resolution trusted timer that no privileged adversary can manipulate, 2) a scheduling policy that retains the rate at which timekeeping service advances time, 3) a domain specific solution to detect and compensate for malicious OS delays, and 3) a network attack-tolerant time synchronization mechanism for maintaining secure global time.

5 Conclusion

A time value is not considered secure unless all elements in a time stack are secured against attacks. In an untrusted OS and network, securing time is a challenge. Though trusted execution environments provide timer protection, they pose timing limitations and vulnerabilities in reading, maintaining and synchronizing time. This paper provides the first detailed examination of timing limitation in TEE along with requirements to design a secure time architecture.

Acknowledgment

This research is funded in part by the National Science Foundation under awards CNS-1329755 and CNS-1705135. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copy- right notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, or the U.S. Government.

References

[1] Taking the first picture of a black hole. *European Southern Observatory* (2017).

[2] BAUMANN, A., PEINADO, M., AND HUNT, G. Shielding applications from an untrusted cloud with haven. *ACM Transactions on Computer Systems (TOCS) 33*, 3 (2015), 8.

[3] BREITINGER, C., AND GIPP, B. Virtualpatent-enabling the traceability of ideas shared online using decentralized trusted timestamping. In *Proceedings of the 15th Int. Symposium of Information Science (2017)* (2017).

[4] CEN, S., AND ZHANG, B. Trusted time and monotonic counters with intel sgx platform services. *Intel Resource Library* (2017).

[5] CHEN, G. J., WIENER, J. L., IYER, S., JAISWAL, A., LEI, R., SIMHA, N., WANG, W., WILFONG, K., WILLIAMSON, T., AND YILMAZ, S. Realtime data processing at facebook. In *Proceedings of the 2016 International Conference on Management of Data* (2016), ACM, pp. 1087–1098.

[6] DEUTSCH, O., SCHIFF, N. R., DOLEV, D., AND SCHAPIRA, M. Preventing (network) time travel with chronos.

[7] GANERIWAL, S., PÖPPER, C., ČAPKUN, S., AND SRIVASTAVA, M. B. Secure time synchronization in sensor networks. *ACM Transactions on Information and System Security (TISSEC) 11*, 4 (2008), 23.

[8] GENG, Y., LIU, S., YIN, Z., NAIK, A., PRABHAKAR, B., ROSENBLUM, M., AND VAHDAT, A. Exploiting a natural network effect for scalable, fine-grained clock synchronization. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)* (Renton, WA, 2018), USENIX Association.

[9] JOHNSON, G. W., SWASZEK, P. F., HARTNETT, R. J., SHALAEV, R., AND WIGGINS, M. An evaluation of eloran as a backup to gps. In *Technologies for Homeland Security, 2007 IEEE Conference on* (2007), IEEE, pp. 95–100.

[10] LIAO, S.-K., CAI, W.-Q., LIU, W.-Y., ZHANG, L., LI, Y., REN, J.-G., YIN, J., SHEN, Q., CAO, Y., LI, Z.-P., ET AL. Satellite-to-ground quantum key distribution. *Nature 549*, 7670 (2017), 43.

[11] MALHOTRA, A., VAN GUNDY, M., VARIA, M., KENNEDY, H., GARDNER, J., AND GOLDBERG, S. The security of ntp's datagram protocol. In *International Conference on Financial Cryptography and Data Security* (2017), Springer, pp. 405–423.

[12] MATETIC, S., AHMED, M., KOSTIAINEN, K., DHAR, A., SOMMER, D., GERVAIS, A., JUELS, A., AND CAPKUN, S. {ROTE}: Rollback protection for trusted execution. In *26th {USENIX} Security Symposium ({USENIX} Security 17)* (2017), pp. 1289–1306.

[13] MILLS, D. L. Internet time synchronization: the network time protocol. *Communications, IEEE Transactions on 39*, 10 (1991).

[14] NORTON, D. Instrumenting a data center with influxdb. USENIX Association.

[15] PSIAKI, M. L., AND HUMPHREYS, T. E. Gnss spoofing and detection. *Proceedings of the IEEE 104*, 6 (2016), 1258–1270.

[16] RAJ, H., SAROIU, S., WOLMAN, A., AIGNER, R., COX, J., ENGLAND, P., FENNER, C., KINSHUMANN, K., LOESER, J., MATTOON, D., ET AL. ftpm: A software-only implementation of a tpm chip. In *USENIX Security Symposium* (2016), pp. 841–856.

[17] RAVINDRANATH, L., PADHYE, J., MAHAJAN, R., AND BALAKRISHNAN, H. Timecard: Controlling user-perceived delays in server-based mobile applications. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles* (2013), ACM, pp. 85–100.

[18] SARMENTA, L. F., VAN DIJK, M., O'DONNELL, C. W., RHODES, J., AND DEVADAS, S. Virtual monotonic counters and count-limited objects using a tpm without a trusted os (extended version).

[19] SHEPARD, D. P., BHATTI, J. A., HUMPHREYS, T. E., AND FANSLER, A. A. Evaluation of smart grid and civilian uav vulnerability to gps spoofing attacks. In *Proceedings of the ION GNSS Meeting* (2012), vol. 3, pp. 3591–3605.

[20] SINGELEE, D., AND PRENEEL, B. Location verification using secure distance bounding protocols. In *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on* (2005), IEEE, pp. 7–pp.

[21] THOMAS, M., NORTON, J., JONES, A., HOPPER, A., WARD, N., CANNON, P., ACKROYD, N., CRUDDACE, P., AND UNWIN, M. Global navigation space systems: reliance and vulnerabilities. *The Royal Academy of Engineering, London* (2011).

[22] TRACH, B., KROHMER, A., GREGOR, F., ARNAUTOV, S., BHATOTIA, P., AND FETZER, C. Shieldbox: Secure middleboxes using shielded execution. In *Proceedings of the Symposium on SDN Research* (2018), ACM, p. 2.