Integrating Verifiable Delay Functions into Confidential Computing Systems

Artem Grigor

01/10/2022

Abstract

Integrating the concept of time into secure computation systems, notably Confidential Computing, has been a longstanding yet elusive goal. Traditional approaches face significant challenges within the Confidential Computing threat model, which precludes trust in the operating system and any hardware other than the CPU. This paper introduces a novel approach to embed a proof of elapsed time within a Confidential Computing environment. Our method facilitates the verification of a minimum time lapse since a designated checkpoint, bridging a critical gap in the temporal capabilities of such systems. Although it does not fully achieve synchronization with real-world clock time, this advancement represents a significant step towards that objective. The cornerstone of our solution is the utilization of Verifiable Delay Functions (VDFs), designed without reliance on complex security assumptions. We propose an innovative construction of VDFs specifically tailored for use within the Confidential Computing framework. These VDFs distinguish themselves by their simplicity, eschewing intricate mathematical dependencies, and offer enhanced accuracy, energy efficiency, and reduced verification times compared to traditional VDF implementations.

1 Introduction

1.1 Time in Computers

Time Synchronization and Maintenance are well-established in regular computational settings, with computers capable of maintaining accurate global time, known as Wall Clock time.

1.1.1 Network Time Protocol

To establish correct Wall Clock time, computers utilize the Network Time Protocol (NTP), a protocol that employs a hierarchical system of time sources, extending from atomic clocks. NTP aids in identifying a reliable true time source within a network and uses probabilistic calculations over the connection time to enable a computer to recalibrate its Wall Clock time accurately [Mil91].

1.1.2 CPU Cycles and Quartz Crystals

Several methods exist for maintaining time. A basic approach involves counting CPU cycles, estimating time based on the number of cycles and average cycle duration. However, this method fails when the system powers down, as it loses track of time, necessitating recalibration via NTP upon restart. Another method employs a quartz crystal embedded in a special chip, similar to wristwatches, allowing the clock to maintain time independently and continue counting even when the system is offline.

However, both methods heavily depend on external trust, subject to control by the computer host.

1.2 Confidential Computing

Confidential Computing refers to a paradigm where computational operations and data remain concealed, safeguarded from manipulation by any external entity, including the system administrator. This is accomplished through the use of specially designed CPUs capable of executing a broad range of logic operations, making them comparable to standard CPUs in functionality. The essence of Confidential Computing lies in its ability to engage in secure interactions with external entities by transmitting

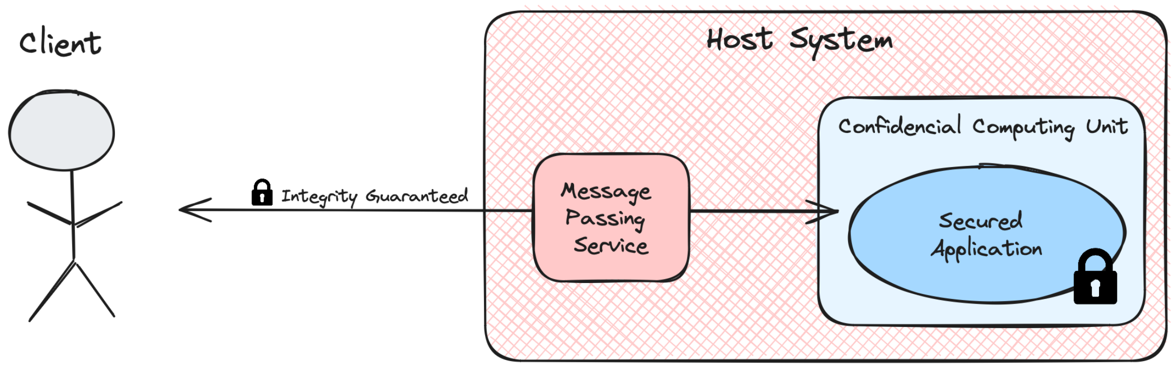

Figure 1: Client communication with a Confidential Computing unit

and receiving encrypted data. A pivotal feature of this model is Remote Attestation, ensuring the authenticity and integrity of the CPU and the applications running within the Confidential Computing environment [Con21].

In recent times, several platforms have adopted this model. Notably, Intel SGX (Software Guard Extensions) is a prominent example, offering an environment for developing applications in higher-level languages like C++ [Spr19]. However, the full potential of these systems remains unrealized due to unresolved challenges, such as state maintenance immune to host manipulation and accurate Wall Clock time acquisition in a trusted manner.

1.2.1 Alternatives to Confidential Computing

Confidential Computing is not the sole approach to secure data processing. Alternatives, relying primarily on cryptographic techniques, exist. For instance, Fully Homomorphic Encryption stands out by enabling operations on encrypted data, ensuring information remains secure throughout the computation process [Uni23]. These methods, however, share common challenges with Confidential Computing, such as the acquisition of accurate Wall Clock time.

1.3 Verifiable Delay Function (VDF)

Introduced by Boneh, Bonneau, Bünz, and Fisch in 2018, Verifiable Delay Functions (VDFs) represent a computational concept where the execution time can be precisely estimated [BBBF18a]. These functions are designed to be executed sequentially, requiring a set number of steps, and upon completion, produce a unique output that can be efficiently and publicly verified. The critical aspect of VDFs is the significant gap between the time taken for execution and the time required for verification, which is usually logarithmically less complex than the function itself.

VDFs have wide-ranging applications in decentralized systems such as public randomness beacons, leader election in consensus protocols, and proofs of replication. One of the key challenges in constructing VDFs lies in balancing the computational intensity and the verification efficiency.

In addition to their theoretical importance, VDFs have practical implications in systems like blockchain networks. For example, they are utilized in the Solana blockchain for ensuring the required time has elapsed between blocks, known as ”Proof of History” [Fou21].

2 Applications

In this section, we explore several practical applications where Verifiable Delay Functions (VDFs) within Confidential Computing units can be particularly beneficial.

2.1 Proving Timeout

One critical application is in proving timeouts within Confidential Computing environments. Often, applications depend on external inputs to progress, and any delay or unavailability of these inputs can

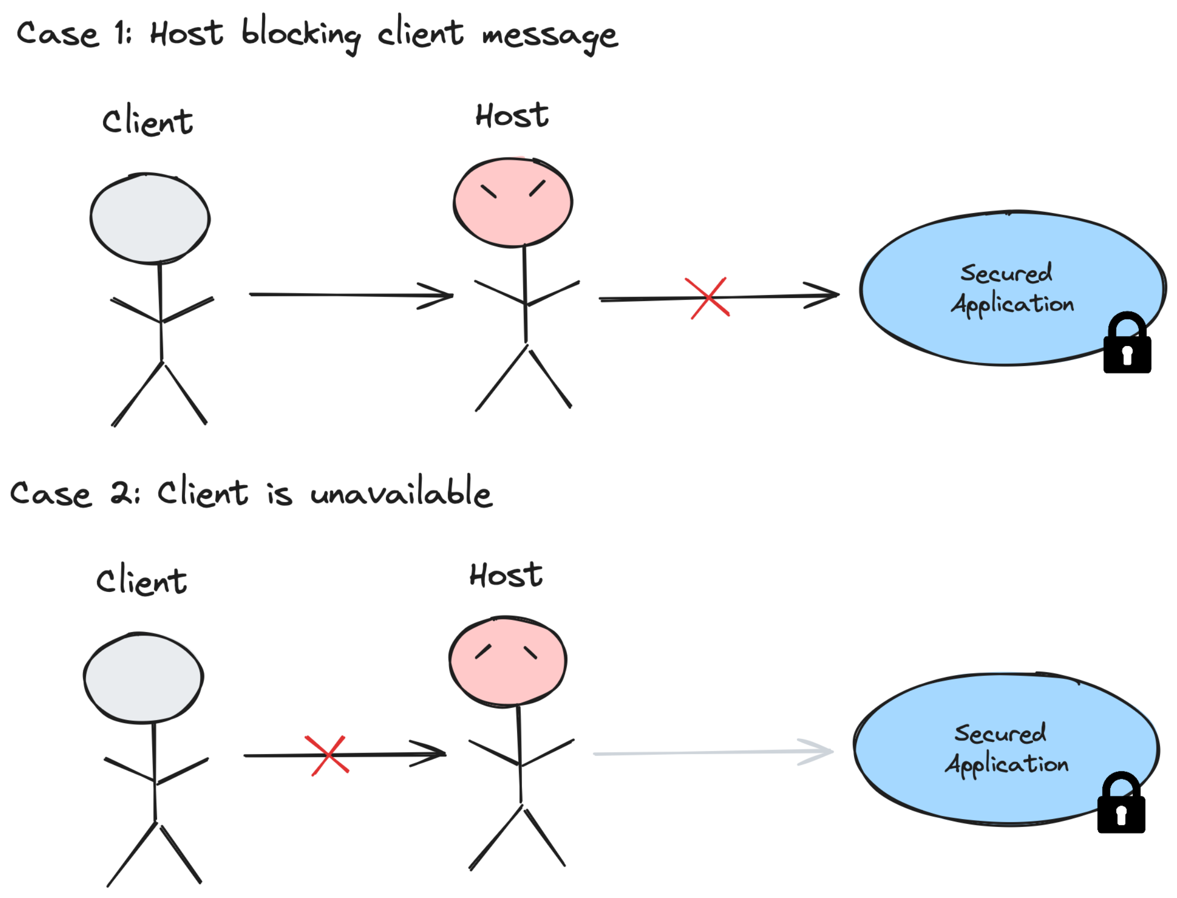
Case 1: Host blocking client message
Client
Host
Secured Application
Case 2: Client is unavailable
Client
Host
Secured Application

Figure 2: Illustration of the Proving Timeout and how it prevents Replay Attack in case 1

result in operational blocks. VDFs offer a solution by allowing the host to provide proof of elapsed time, ensuring that the application can identify and respond to timeouts effectively. This capability is especially vital in systems requiring constant communication, such as security monitoring systems. The use of VDFs in such scenarios enhances reliability and operational integrity [Sem18].

2.2 Replay Protection

Another vital application of VDFs is in providing enhanced protection against replay and denial attacks in Confidential Computing. By incorporating time-based mechanisms, VDFs significantly increase the time required for executing such attacks, making them more detectable and less feasible. This aspect is particularly relevant in scenarios like auctions or user polls, where the integrity of the process is paramount. Timely detection and response to potential replay attacks are crucial for maintaining the confidentiality and integrity of the data and operations within these systems [Eur21].

3 Previous Work

This section reviews alternative approaches considered or implemented to integrate time within Confidential Computing systems. We focus on the works by Fatima M. Anwar and Mani Srivastava [AS19] and Fatima M. Anwar, Luis Garcia, and Xi Han and Mani Srivastava [AGHS19].

3.1 Global Time Service

A Global Time Service involves a trusted global time authority capable of creating signed timestamps. The Confidential Computing unit can verify these timestamps to ensure the passage of a specific point in time. This concept is akin to the Newspaper Method [aut], where newspaper articles are used to prove an event occurred after a certain date. However, this method has several drawbacks:

- It requires a continuously operational, trusted global time authority, which broadcasts to all Confidential Computing units. The dependency on a constant online presence and connectivity to the time authority limits the system's operational flexibility.
- The method lacks the capability to establish current time or measure elapsed time between timestamp creation and its reading by the Confidential Computing unit, posing risks of time-duration attacks.

To our knowledge, this approach is not widely adopted in Confidential Computing units due to these limitations.

3.2 Secure Hardware Clock

The concept of a Secure Hardware Clock involves a Time Chip, typically based on a quartz clock, enhanced with security features to prevent physical tampering [AGHS19]. The Time Chip retrieves the amount of time elapsed since its last call. Key features include:

- A separate battery to ensure continuous operation, preventing the host from resetting the chip [Wik23].
- Checkpointing interactions with the clock to detect manipulations.

Despite these features, the Time Chip is vulnerable to attacks like host delay or message replay between the Time Chip and the processor. Additionally, it can only prove the elapsed time while the CPU is active, thus unable to account for periods when the system is offline or before the CPU's initial start [AGHS19]. The hardware-based nature of this solution also exposes it to potential side-channel attacks.

4 VDFs and Confidential Computing

Integrating time into Confidential Computing units poses significant challenges. We propose a novel approach using Verifiable Delay Functions (VDFs) to prove elapsed time within Confidential Computing environments, expanding the system's temporal capabilities beyond previous solutions. This approach is secure, relying on the inherent security of the chosen VDFs.

4.1 Proof Verification

Considering a potentially malicious host, the Confidential Computing unit employs a verification function to ascertain elapsed time. This function, receiving the elapsed time, a nouns, and proof of elapsed time, outputs a boolean indicating the accuracy of the proof.

1. **Checkpoint-Based nouns Generation:** The unit generates a random nouns at any desired point, such as upon receiving a message or at an event's occurrence. This method is illustrated in the context of polls, where the checkpoint is set at the poll's commencement, and the host computes the VDF until the required time elapses before sending the proof back for verification [BBBF18b].
2. **Timestamp-Based nouns Generation:** Utilizing a trusted timestamp, the Confidential Computing unit uses the hash of the timestamp as the nouns. This approach ensures that, along with the proof of elapsed time, the specific date and time are also verified, enhancing the trustworthiness of the process [rfc01].

4.2 Proof Generation

We delve into the methodology for generating proof of elapsed time in Confidential Computing systems using Verifiable Delay Functions (VDFs). The procedure leverages a nouns defined by the Confidential Computing unit to validate the time lapse from a designated checkpoint.

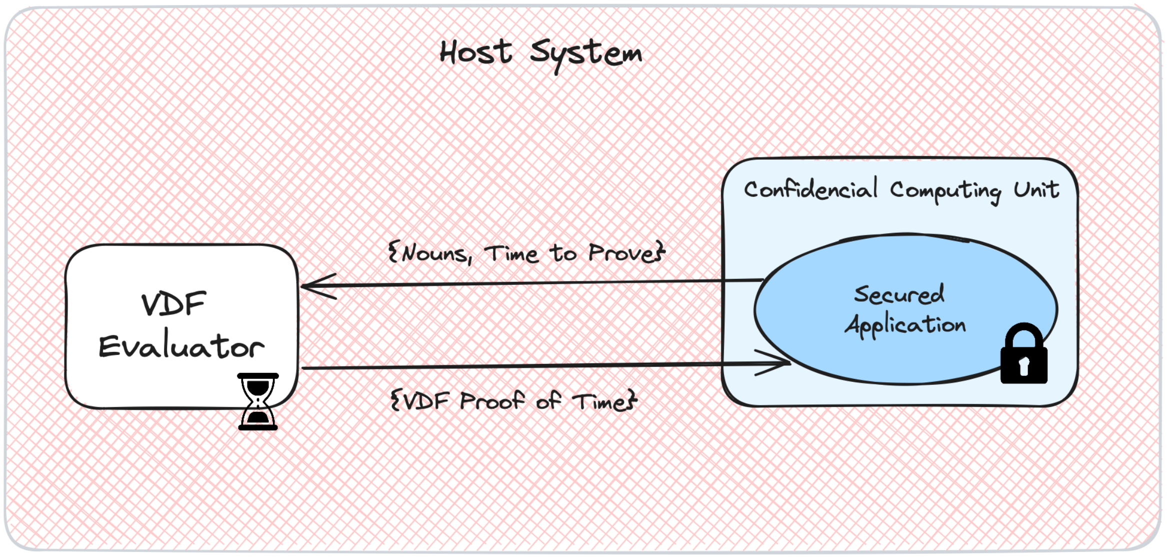

Figure 3: Illustration of Host-Based VDF Operation

4.2.1 Host Based

The initial consideration is the delegation of VDF proof calculation to the host, a process devoid of trust requirements. The host, operating on a distinct machine, executes the VDF computation in the background. Upon completion, the proof is submitted to the Confidential Computing unit, affirming the elapsed time.

However, this method exhibits a limitation: the host may utilize diverse processors with varying capabilities to generate the VDF proof. This variability necessitates an "overproof" strategy, potentially requiring more than tenfold the necessary time to ensure robust validation. Such a precaution counters potential exploitation by a malicious host employing high-frequency CPUs to accelerate computation and subvert the system [Wes20].

4.2.2 Confidential Computing Based

An effective approach leverages the unique capabilities of Confidential Computing units. One key feature is Remote Attestation, which can confirm the specific processor used in the unit. This capability allows for more accurate estimation of elapsed time by analyzing the average CPU cycle duration of the identified processor, significantly reducing the overhead from 1000% to approximately 20% in time estimation accuracy [Hat23b].

Enhancing this solution, Confidential Computing can address potential host attacks aimed at optimizing VDF calculations. Since VDFs inherently involve sequential computations that are challenging to compute but straightforward to verify, ensuring adherence to a preapproved algorithm via Remote Attestation mitigates the risk of computational shortcuts. This approach ensures that the VDF is computed as intended, maintaining the integrity of the proof generation process [Hat23a].

5 Confidential Computing VDFs

With the advancements in Confidential Computing, the necessity for complex Verifiable Delay Functions (VDFs) is significantly reduced. The key lies in the assurance that the Confidential Computing unit is executing a preapproved, non-manipulable algorithm. When such an algorithm indicates sufficient computational tasks—like hashing or other efficient operations—have been performed since a checkpoint, this output is deemed reliable. The authenticity of this operation is verified through the attestation of the Confidential Computing unit, ensuring the system's integrity and the algorithm's compliance with predefined properties [**?**].

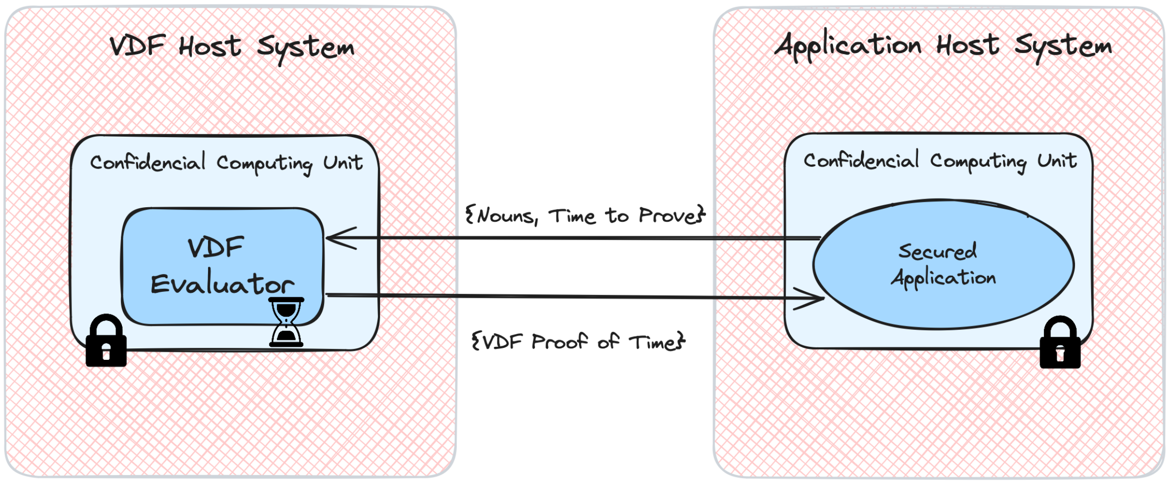

Figure 4: Operation of a Confidential-Computing-Based VDF

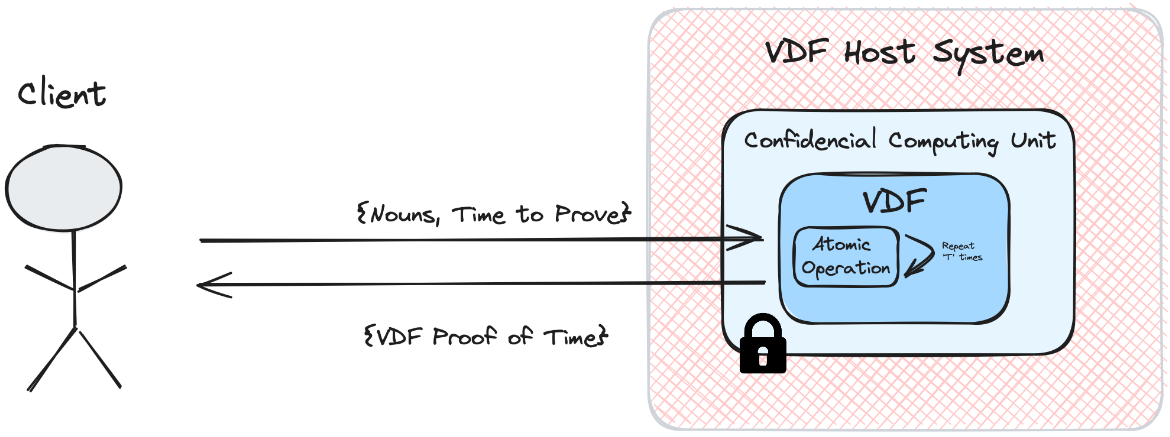

Figure 5: Implementation of Confidential Computing VDF

This innovation leads to the development of a new genre of hardware-based VDFs characterized by a constant verification time, irrespective of the actual time span being validated. A significant advantage of this approach is the inherent trust in Confidential Computing units. If these units are considered secure, the VDF output generated within them is equally trustworthy, as it does not introduce any new trust vectors.

Moreover, Confidential Computing VDFs are not limited to specialized systems but are also applicable to regular computing systems. They offer a consistent verification overhead, ease of development and validation, and enhanced reliability compared to traditional cryptographic VDFs. Furthermore, these VDFs are expected to be more computationally efficient, as they do not necessitate maximum CPU utilization, but rather leverage low-cost operations.

6 Limitations

While the proposed system offers significant advancements, it is important to recognize its limitations:

- The reliance on estimations rather than concrete measurements means there is no absolute guarantee of the elapsed time since the checkpoint. However, the Confidential Computing VDF minimizes these error margins, closely approximating an exact measure.
- The methodology may lead to considerable power consumption, especially over extended periods, due to computational demands. This aspect is a common limitation across all VDFs, though Confidential Computing VDFs potentially offer greater efficiency.

7 Future Work

Future research will focus on:

- Refining the confidence bounds of the Confidential Computing VDF's time proof and optimizing power consumption.
- Comparing the performance of the proposed Confidential Computing VDFs with traditional cryptographic VDFs, with preliminary projections indicating a potential tenfold improvement in efficiency and effectiveness.

8 Conclusion

This paper has explored the synergy between Verifiable Delay Functions (VDFs) and Confidential Computing. We introduced a new system for implementing VDFs within the Confidential Computing framework and proposed an innovative VDF design based on Confidential Computing technology. The expected outcomes include significantly reduced error bounds, constant verification times, and lower power consumption during computation. These advancements are anticipated to propel the study of VDFs forward and encourage their widespread adoption.

References

[AGHS19] Fatima Anwar, Luis Garcia, Xi Han, and Mani Srivastava. Securing time in untrusted operating systems with timeseal. In *2019 IEEE Real-Time Systems Symposium (RTSS)*, 2019.

[AS19] Fatima M Anwar and Mani Srivastava. Applications and challenges in securing time. *USENIX Conference on Cyber Security Experimentation and Test*, 2019.

[aut] Proof of ealieast time using a news paper. https://tvtropes.org/pmwiki/pmwiki.php/Main/AuthenticationByNewspaper. Accessed: 2022-09-30.

[BBBF18a] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. https://eprint.iacr.org/2018/601, 2018.

[BBBF18b] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. *Cryptology ePrint Archive, Paper 2018/601*, 2018. https://eprint.iacr.org/2018/601.

[Con21] Confidential Computing Consortium. Confidential computing and related technologies: a critical review. *Cybersecurity*, 2021.

[Eur21] EurekAlert. Resilience against replay attacks in computer systems. *EurekAlert*, 2021.

[Fou21] Solana Foundation. Proof of history in solana blockchain. *Solana Whitepaper*, 2021.

[Hat23a] Red Hat. Attestation flow in confidential computing. https://www.redhat.com/en/topics/security/what-is-attestation, 2023.

[Hat23b] Red Hat. Attestation in confidential computing. https://www.redhat.com/en/topics/security/what-is-attestation, 2023.

[Mil91] David L Mills. Network time protocol (version 3) specification, implementation. *RFC 1305, March*, 1992, 1991.

[rfc01] Rfc 3161 - internet x.509 public key infrastructure time-stamp protocol (tsp). https://tools.ietf.org/html/rfc3161, 2001.

[Sem18] Semanticscholar. Verifiable delay function and its blockchain-related application. https://www.semanticscholar.org/paper/Verifiable-Delay-Function-and-Its-Application%3A-A-Wu-Xi/3b3d21297c3ed5d2e1504074d38cacee92e266aa, 2018.

[Spr19] SpringerLink. A survey of intel sgx and its applications. *Frontiers of Computer Science*, 2019.

[Uni23] Harvard University. Fully homomorphic encryption. https://dash.harvard.edu/bitstream/handle/1/37376404/Thesis.pdf?sequence=1, Accessed 2023.

[Wes20] Benjamin Wesolowski. Efficient verifiable delay functions. *Journal of Cryptology*, 2020. https://dl.acm.org/doi/10.1007/s00145-020-09364-x.

[Wik23] Wikipedia. Real-time clock. *Wikipedia, the free encyclopedia*, 2023. https://en.wikipedia.org/wiki/Real-time_clock.