Cybersecurity

REVIEW **Open Access**

Confidential computing and related technologies: a critical review

Muhammad Usama Sardar* and Christof Fetzer

Abstract
This research critically reviews the definition of confidential computing (CC) and the security comparison of CC with other related technologies by the Confidential Computing Consortium (CCC). We demonstrate that the definitions by CCC are ambiguous, incomplete and even conflicting. We also demonstrate that the security comparison of CC with other technologies is neither scientific nor fair. We highlight the issues in the definitions and comparisons and provide initial recommendations for fixing the issues. These recommendations are the first step towards more precise definitions and reliable comparisons in the future.

Keywords Confidential Computing (CC), Trusted Execution Environment (TEE), Homomorphic Encryption (HE), Trusted Platform Module (TPM), Formal methods

Introduction

Regulations, such as the General Data Protection Regulation (GDPR) (European Commission: Regulation (EU) 2016/679 2016), necessitate protecting personal data. A wide variety of technologies, from purely algorithmic to hardware-based, exists as possible solutions. Purely algorithmic solutions include cryptographic methods, such as Homomorphic Encryption (HE) (Acar et al. 2018) and Secure Multi-Party Computation (MPC) (Evans et al. 2018). On the other end of the spectrum, hardware-based solutions include a secure cryptographic processor, such as Hardware Security Module (HSM) (Anderson 2020) and Trusted Platform Module (TPM) (Arthur and Challener 2015; Proudler et al. 2014). Recently, Confidential Computing (CC) using hardware-based Trusted Execution Environments (TEEs) has emerged as a promising solution (Confidential Computing consortium 2021). With such emerging technologies, it is crucial to precisely define, as well as reliably evaluate and compare technologies to choose the most suitable one for a given problem while ensuring compliance with legal standards.

As a first step towards definitions and comparison of technologies, some white papers (Confidential Computing consortium 2021a, b, c) have recently attempted to tackle this challenge. These white papers are a collaborative effort of the Confidential Computing Consortium (CCC), comprising various notable organizations, including Google, Microsoft, Huawei, Red Hat, Arm and Intel. Although CCC recommends using the definitions, we demonstrate in this work that the definitions by CCC are imprecise, incomplete and even conflicting. This is critical because of three main reasons. Firstly, it leaves room for competing vendors to claim CC to potentially give end users[1] a false sense of security. Secondly, it creates legal uncertainty in the regulatory authorities for ensuring compliance with regulations, specifically transparency obligations. Finally, for researchers and solution providers, this hampers a reliable comparison across vendor solutions in a uniform way.

*Correspondence:
Muhammad Usama Sardar
muhammad_usama.sardar@tu-dresden.de
Faculty of Computer Science, Technische Universität Dresden, Dresden, Germany

[1] Typically with less technical knowledge.

In relation to the comparison among various technologies for the protection of data, CCC white papers (Confidential Computing consortium 2021a, b, c) present security comparisons for TEE, HE and TPM. However, the threat model for such a comparison is not specified, and thus the comparison may lead to misleading conclusions. Moreover, there is a lack of scientific evidence and argumentation in the comparison.

To improve the situation, we review and highlight some of the key issues in the definitions related to CC ("Definitions" Section) as well as its comparison with related technologies ("Comparison of CC with relatedtechnologies" Section) in this work. We pointed out these issues to CCC Technical Advisory Council (TAC) in April 2021. However, the rare response from CCC TAC has generally raised more questions and concerns (Confidential Computing consortium 2020). We, therefore, pose it as an open challenge to the community. We emphasize the dire need for formal definitions for CC and believe that resolving the issues highlighted in this work would be the first step towards that. In this regard, we also provide some recommendations to achieve clarity and precision in the definitions as well as fair and scientific comparison among existing technologies. To the best of our knowledge, no other works have attempted to provide a security comparison of the technologies, so we focus on the CCC white papers and related literature. An attempt has been made to keep this paper self-contained by reproducing the original definitions, figure and table, whenever required, and to keep the paper simple and readable for broad readers from systems engineering, security and privacy, and legal communities, while still maintaining mathematical rigour.

Definitions

This section presents a review of definitions related to CC by CCC TAC. All definitions by CCC TAC are informal. Formal definitions of security are essential for a systematic design of cryptosystems, as Katz and Lindell reinforce:

“If you don’t understand what you want to achieve, how can you possibly know when (or if) you have achieved it?” [Sect. 1.4.1, p. 15 in Katz and Lindell (2020)].

Unlike the traditional heuristic approach based on intuition, formal definitions also allow for rigorous proof of security. Moreover, formal definitions provide the basis for a meaningful comparison of cryptosystems, such as trade-offs between security and efficiency (Katz and Lindell 2020).

In the following subsections, we list some of the key issues related to definitions of terms related to CC by first providing CCC’s actual definitions, whenever available.

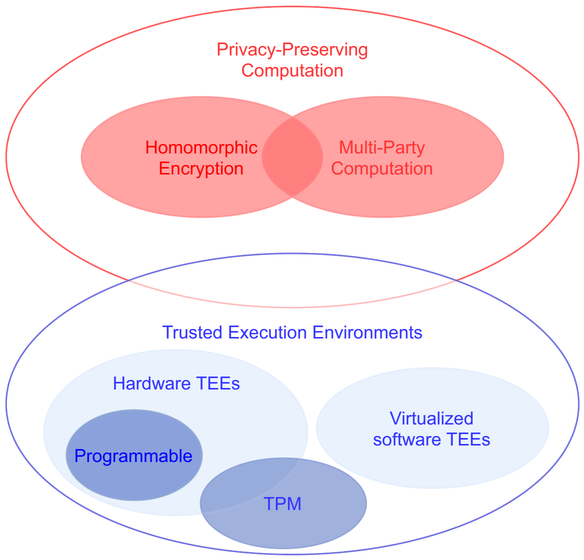
Privacy-Preserving Computation
Homomorphic Encryption
Multi-Party Computation
Trusted Execution Environments
Hardware TEEs
Programmable
TPM
Virtualized software TEEs

Fig. 1 Venn diagram, illustrating technologies for secure computation, produced by Confidential Computing Consortium (CCC) as a result of a survey (2021a)

Confidential Computing (CC) and related technologies

CCC claims that CC is uniquely defined, while several other related technologies have multiple competing definitions:

“Unlike the term ‘confidential computing’, several of the terms used in the diagram have multiple competing definitions” [Sect. 4, p. 8 in Confidential Computing consortium (2021a)].

The diagram referred to here is the Venn diagram shown in Fig. 1.

We first analyze the claim of the unique definition of CC. This claim is based on the following definition by CCC:

“The protection of data in use by performing computation in a hardware-based Trusted Execution Environment” [Sect. 2.1, p. 5 in Confidential Computing consortium (2021a)]

In contrast to the above definition based on hardware-based Trusted Execution Environment (HW TEE), in CCC scope white paper, it is considered to be based on programmable[2] HW TEE [cf. Fig. 1 in Confidential Computing consortium (2021c)]. It is evident from Fig. 1 that the set of programmable HW TEEs (`prog_HW_TEE`) is a proper subset of the set of HW TEEs (`HW_TEE`), i.e.,

$$\texttt{prog\_HW\_TEE} \subset \texttt{HW\_TEE} \tag{1}$$

[2] The definition of programmable itself in Confidential Computing consortium (2021a) is ambiguous (more details in "Programmability" Section). However, our argument still holds whatever definition of programmable is considered.

Hence, the two definitions of CC within the CCC white papers are competing. Moreover, even the members of CCC do not agree on the definition of CC. For instance, researchers at Arm (member of CCC) define CC to include everything in the Venn diagram in Fig. 1 (Mulligan et al. 2021). Therefore, the unique definition claim of CCC is false.

We now analyze the claim about multiple competing definitions of other terms. In contrast to CC, other terms used in the figure are formally defined in the literature. For instance, HE is formally defined by Fontaine and Galand (2007) as:

Let $\mathcal{M}$ (resp., $\mathcal{C}$) denote the set of plaintexts (resp., ciphertexts). An encryption scheme is said to be *homomorphic* if for any given encryption key k the encryption function E satisfies

$$\forall\, m_1, m_2 \in \mathcal{M},\; E(m_1 \odot_{\mathcal{M}} m_2) \longleftarrow E(m_1) \odot_{\mathcal{C}} E(m_2)$$

for some operators $\odot_{\mathcal{M}}$ in $\mathcal{M}$ and $\odot_{\mathcal{C}}$ in $\mathcal{C}$, where $\longleftarrow$ means "can be directly computed from," that is, without any intermediate decryption.

If $(\mathcal{M}, \odot_{\mathcal{M}})$ and $(\mathcal{C}, \odot_{\mathcal{C}})$ are groups, we have a *group homomorphism*. We say a scheme is *additively homomorphic* if we consider addition operators, and *multiplicatively homomorphic* if we consider multiplication operators.

Similar to HE, MPC is defined by Evans et al. (2018). Privacy-preserving computation, like secure computation, is a general term and is interpreted differently based on the privacy goals, such as input privacy or output privacy. Hence, the claim of CCC is not true.

Trusted Execution Environment (TEE)

CCC defines TEE as:

"An environment that provides a level of assurance of the following three properties:

- Data confidentiality: Unauthorized entities cannot view data while it is in use within the TEE.
- Data integrity: Unauthorized entities cannot add, remove, or alter data while it is in use within the TEE.
- Code integrity: Unauthorized entities cannot add, remove, or alter code executing in the TEE" [Sect. 3.1, p.6 in Confidential Computing consortium (2021a)].

We observe the following issues in this definition:

- This definition of TEE is quite vague. For example, it is unclear what is meant by "a level of assurance", and it can be interpreted in various ways. Similarly, "data" is undefined and can be interpreted to mean *all* data or a well-defined *subset* of data. Moreover, no description of "unauthorized entities" is provided.
- The definition is not precise enough to characterize TEEs. For example, all three properties are satisfied by HSMs also.
- Both fundamental components of a security definition, according to Katz and Lindell (2020), are missing, i.e., the definition of TEE neither clearly defines what constitutes a successful attack, nor does it specify the threat model, i.e., (computational) power of the adversary, e.g., probabilistic polynomial time.

In summary, the definition of TEE is vague and incomplete. With such a definition, it may not be possible to classify an environment as a TEE reliably or to have a meaningful comparison among different TEE solutions from different vendors.

TEE environment

Although the term "TEE" is somewhat defined in CCC white paper, another term, "TEE environments", is used and remains undefined (Confidential Computing consortium 2021a). Expanding the abbreviation TEE, it stands for "Trusted Execution Environment environments". The two terms cannot be interpreted as synonyms because, in one place, they are used together:

"Attestation of TEEs and TEE environments" [Sect. 5.1, p. 10 in Confidential Computing consortium (2021a)]

The term "TEE environments" is also not defined in any of the references of the white paper (Confidential Computing consortium 2021a). Therefore, the difference between the two terms is not clear.

Hardware-based TEE

It is important to define HW TEE and contrast it with virtualized software TEE because one of the CC definitions is dependent on HW TEEs (Confidential Computing consortium 2021a). Given the CCC claim that these two sets are disjoint (Fig. 1), it should be possible to clearly and distinctly define the two terms. However, precise definitions of HW-based TEE and virtualized software TEE are missing. For instance, it is unclear whether hardware includes firmware.

Programmability

Since one of the definitions of CC by CCC is based on programmable HW TEEs (Confidential Computing consortium 2021c), it is also important to clearly define programmability. However, it is not the case, and there are at least two different possible interpretations of programmability:

Table 1 Summary of key issues in the definitions of terms related to CC, and recommendations for CCC

#	Terms	Issues	Recommendations
1	CC	HW TEE (Confidential Computing consortium 2021a) versus programmable HW TEE (Confidential Computing consortium 2021c)	Claim on unique definition of CC and multiple conflicting definitions of other technologies should be removed.
		Conflicting definition by researchers at Arm Mulligan et al. (2021)	
		Other technologies, e.g., HE, are even formally defined Katz and Lindell (2020).	
2	TEE	Ambiguous terms	A clear and distinguishing definition should be given.
		Definition satisfied by HSM also	
		Unclear threat model	
3	TEE Environment	Undefined	The term should be rephrased. It should be compared and contrasted with TEE for clarity.
4	HW TEE	Undefined	It should be compared and contrasted with virtualized SW TEE for clarity.
5	Programmability	Arbitrary code versus limited set of operations	It should be clarified that programmability is Turing-complete.
6	Attestation	Key components, such as Relying Party and measurement, are missing	The definition should explicitly include Relying Party and some form of trusted measurement.
7	Attestation in CC	Incomplete definition of CC	The definition of CC should include attestation as it is a key feature of CC.

The recommendation "Formal, or at least precise, clear and consistent, definitions should be given." applies to all rows and therefore, it is not mentioned in the table

- TEE can be programmed with arbitrary code.
- TEE supports only a limited set of operations and is thus limited to specialized problems that can be framed in terms of supported operations.

It should be clarified that the interpretation of programmability is Turing-complete, i.e., arbitrary code can be run in a programmable TEE. Moreover, for completeness in Fig. 1, the programmable set should also be depicted for virtualized software TEEs since it is shown only for HW TEEs.

Attestation

CCC defines attestation as:

"Attestation is the process by which one party, called a 'Verifier', assesses the trustworthiness of a potentially untrusted peer, i.e., the 'Attester'" [Sect. 6, p. 14 in Confidential Computing consortium (2021a)].

Here, the definition is missing a vital party namely 'Relying Party', as defined in Birkholz et al. (2023). Moreover, the notion of measurement, which is a key component of attestation (Szefer 2018), is not explicit in the definition. A fundamental property of attestation is that the attested entity must not be able to lie about the state (i.e., code, configuration and data) that it is in. Hence, the definition of attestation should explicitly include some form of trusted measurements of both code and configuration of a TEE's platform as well as the code that executes inside the TEE, its configuration and some TEE-held data.

Attestation in CC

Compared to purely cryptographic technologies (such as Fully Homomorphic Encryption (FHE) and Secure Multi-Party Computation), CC using HW TEEs on commercial architectures has much more trust in hardware and software components (Szefer 2018). Hence, in HW TEEs, it is vital to attest that the hardware is up to date with the latest security features. Without attestation, TEEs do not provide better security guarantees than conventional computing under the adversary models considered in CC. This is a consequence of the fact that without attestation, a remote user cannot distinguish between a malicious platform and a genuine one. This holds even with alternatives of attestation, such as authentication. Therefore, the definition of CC is incomplete without the fundamental process of attestation.

CCC acknowledges the importance of attestation by devoting a complete section (Sect. 6) in the white paper (Confidential Computing consortium 2021a) to attestation and by stating:

"Any attack that could compromise the attestation of a TEE instance could lead to a workload or data being compromised in turn" [Sect. 5.2.1, p. 11 in Confidential Computing consortium (2021a)].

However, it is not reflected in the definition of CC.

In summary, the definitions by CCC are imprecise, incomplete and even conflicting. Table 1 provides a summary of issues and some recommendations.

Table 2 Comparison of security properties of hardware-based trusted execution environment (HW TEE), Homomorphic encryption (HE) and trusted platform module (TPM) by CCC (cf. Table 1 in Confidential Computing consortium (2021a)

	HW TEE	Homomorphic encryption	Secure element e.g., TPM
Data integrity	Yes	Yes (subject to code integrity)	Keys only
Data confidentiality	Yes	Yes	Keys only
Code integrity	Yes	No	Yes
Code confidentiality	Yes (may require work)	No	Yes
Authenticated Launch	Varies	No	No
Programmability	Yes	Partial ("circuits")	No
Attestability	Yes	No	Yes
Recoverability	Yes	No	Yes

Comparison of CC with related technologies

This section presents a review of the survey conducted by CCC TAC regarding the comparison of CC with other related technologies. One of the most important criteria—threat model—for the security comparison of technologies is not specified by CCC. Moreover, only a few references are provided by CCC on the survey. The references provided are too broad (such as Wikipedia pages) and unreviewed (such as blog posts). Additionally, wherever provided, argumentation and reasoning are generally poor. Hence, there is insufficient scientific evidence, in general, for a reader to trust the survey results, e.g., figures and tables.

In the following, we present some of the key issues concerning the comparison of CC with related technologies:

1. One of the most important criteria for any meaningful comparison of security properties is the specification of the threat model. Unlike TEEs, neither the hardware nor the processor manufacturer needs to be trusted[3] in FHE (Gentry 2009). Among the software privilege levels, at least one of the application, operating system or hypervisor needs to be trusted in the commercial TEE solutions available, whereas none of these needs to be trusted in FHE. Therefore, there is not a TCB anymore in FHE in some sense. On the contrary, any bug or vulnerability in the TCB in TEEs can be exploited to nullify the security protections, e.g., System Management Mode (SMM)-based rootkits (Embleton et al. 2013) and platform management engine-based attacks. Additionally, TEE users have to trust the code running as part of the TCB, which is often proprietary, notably for the security engine, with no runtime monitoring or reporting capabilities, leading to security through obscurity. Typically, side channels are not a part of the threat model of commercial TEEs, whereas in FHE plaintext information is not available anywhere on the system that could leak out (Szefer 2018). Therefore, with an unspecified threat model in Sect. 4.1, a security comparison by CCC (2021a) shown in Table 2 has limited value.
2. CCC claims that TPM is a TEE (Confidential Computing consortium 2021a). This claim is made by showing TPM completely within TEEs in the Venn diagram shown in Fig. 1, where according to CCC, the elements of sets are existing definitions of various concepts (Confidential Computing Consortium 2020). Although there is a disclaimer about multiple competing definitions of the terms, no supporting evidence, reference or argument is provided that justifies or even implies that all TPMs are TEEs. Assuming this claim to be true, TPM must satisfy the CCC definition of TEEs, i.e., satisfy the three properties of data integrity, data confidentiality and code integrity. However, this is contradicted by Table 2 in CCC white paper itself (Confidential Computing consortium 2021a), where it is shown that TPM can provide integrity and confidentiality for keys *only*, as opposed to arbitrary data. The clear contradiction to the claim is further confirmed by the statement of Dave Thaler, the chair of CCC TAC, in one of the CCC webinars (2020d): "If something does not have a Y, Y, Y (each Y corresponding to 'Yes' for the properties data confidentiality, data integrity, and code integrity), then we don't call it a TEE, we call it something else." More formally, according to the definition of TEE by CCC, each element of the *TEE* set should satisfy the three properties. We represent this as follows:

$$TEE = \{x : (x \in DataConf) \wedge (x \in DataInt) \wedge (x \in CodeInt)\} \quad (2)$$

[3] The discussion here is not relevant for availability because none of the commercial TEEs currently address availability issues.

where *DataConf*, *DataInt* and *CodeInt* represent the sets of technologies satisfying data confidentiality, data integrity and code integrity, respectively. Now, from Fig. 1 by CCC, *TPM* is a proper subset of *TEE*, i.e.,

$$TPM \subset TEE \tag{3}$$

Since *TPM* is a subset of *TEE*, each element of *TPM* should also satisfy the three properties of *TEE*. This contradicts Table 2 by CCC, where TPM provides confidentiality and integrity for keys *only*, as opposed to TEE which provides confidentiality and integrity for *arbitrary* data. Hence, CCC literature itself contradicts the claim of TPM as a TEE.

3. The only place in the CCC white papers where some argumentation is provided is related to Table 2 in Sect. 4.1 in Confidential Computing Consortium (2021a). Here we list a couple of issues related to Table 2:

    - Although some reasoning is provided in Sect. 4.1 for TPM, it does not trivially lead to the results concluded for all Secure Elements in Table 2. Whatever is reasoned about TPM cannot be generalized to all Secure Elements without further reasoning. A logical way of reasoning would be to justify it for Secure Element, and then it can be applied to TPM as a special case.
    - We argue that the comparison criteria in Table 2 are sometimes misleading. A "No" does not necessarily mean that technology offers lower security than a "Yes". It may be the case that the property is not relevant. For example, attestability may not be required for HE for semi-honest adversaries.

4. Trusted Computing is a related technology that is often confused with CC. For example, some authors, such as Costan et al. (2016, 2017a, 2017b) use Trusted Computing as a synonym for CC (Ahmad-Reza Sadeghi 2021). Similarly, the Wikipedia page for Trusted Computing states, "Trusted Computing, also often referred to as Confidential Computing (2021)." To avoid confusion, it is important to clearly distinguish CC from Trusted Computing, whereas CCC does not mention Trusted Computing in white papers.
5. It is not clear why related technologies, such as HSMs (Anderson 2020), MPC (Evans et al. 2018) and Zero-Knowledge Proofs are not considered for a fair comparison in Sect. 4 in Confidential Computing consortium (2021a).
6. CCC claims that ambiguity in definitions of HE and MPC leads to an intersection between the two sets in Fig. 1 (Confidential Computing consortium 2021a). However, the ambiguity is not specified. Similarly, the reasoning for the overlap between TPM and HW TEEs in Fig. 1 is not presented. Such ambiguities should be justified with precise definitions, references and arguments.

Conclusion

Despite the undeniable importance of formal definitions, CC and its related terms still have ambiguous, incomplete, and conflicting definitions. We demonstrated this with the help of concrete examples of various terms, including TEEs, hardware-based TEEs and attestation. We also demonstrated that the security comparison among the various technologies presented by CCC without a clear threat model is neither fair nor scientific. We also highlighted other issues, such as lack of scientific evidence and argumentation in the comparison. We provided some initial recommendations for fixing definitions as well as comparison. However, formal definitions of the terminology related to CC and a detailed scientific survey of the technologies for a fair comparison remains a crucial area to be explored in the future. We are actively engaging with the CCC TAC to address the identified issues.

Abbreviations

CC	Confidential Computing
CCC	Confidential Computing Consortium
GDPR	General Data Protection Regulation
FHE	Fully Homomorphic Encryption
HE	Homomorphic Encryption
HSM	Hardware Security Module
HW TEE	Hardware-based Trusted Execution Environment
MPC	Secure Multi-Party Computation
TAC	Technical Advisory Council
TEE	Trusted Execution Environment
TPM	Trusted Platform Module

Acknowledgements
We would like to thank Andrey Brito, Mark F. Novak and SatyadharmaTirtarasa for their feedback on the initial drafts of the paper.

Author contributions
MUS created the initial draft of the paper, based on extensive discussions on each point with CF. CF provided key insights. Both authors read and approved the final manuscript

Authors' information
Muhammad Usama Sardar Since October 2021, he is a Research Associate at TU Dresden. From 2017 until 2021, he was recipient of the prestigious DAAD research grant for his Ph.D. at TU Dresden. His current research interests include the formal specification and verification of the remote attestation process in Trusted Execution Environments, with a special focus on Intel SGX

and TDX. He is also a tutor for the master's courses: Systems Engineering 1, Systems Engineering 2, Principles of Dependable Systems and Software Fault Tolerance. He is also serving as a volunteer at CAVlinks. He has received best poster award in Postgraduate category in the International Conference on Digital Futures and Transformative Technologies (ICoDT2) in 2021, South Asia Triple Helix Association (SATHA) Innovation Award in 2018, best researcher of the year award in System Analysis and Verification lab in Pakistan in 2017, and best speaker awards at Workshop on Applications in ASIC Design (December 2016) and Workshop on Recent Trends in Theorem Proving (April 2016). His research work has resulted in publications at top international forums, such as the Journal of Parallel and Distributed Computing, the NASA Formal Methods Symposium, and Journal of Automated Reasoning.
Christof Fetzer has received his diploma in Computer Science from the University of Kaiserslautern, Germany (Dec. 1992) and his PhD from UC San Diego (March 1997). He joined AT&T Labs-Research in August 1999 and had been a principal member of technical staff until March 2004. Since April 2004 he heads the endowed chair (Heinz-Nixdorf endowment) in Systems Engineering in the Computer Science Department at TU Dresden. He is the chair of the Distributed Systems Engineering International Masters Program at the Computer Science Department. He has published over 150 research papers in the field of dependable distributed systems. Prof. Dr. Fetzer has been member of more than 50 program committees, has won three best paper awards (DEBS2013, LISA2013, SRDS2014), his PhD students have won two best student paper awards (IEEE Cloud 2014, DSN2015), and the EuroSys Roger Needham Award 2014. As a student, he received a two-year scholarship from the DAAD and won two best student paper awards (SRDS and DSN). He was a finalist of the 1998 Council of Graduate Schools/UMI distinguished dissertation award and received an IEE mather premium in 1999.

Funding
Funded by DFG Grants 389792660 as part of TRR 248—CPEC and390696704 as part of CeTI.

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable.

Competing interests
All authors read and approved the final manuscript

Received: 8 March 2022 Accepted: 9 January 2023
Published online: 02 May 2023

References

Acar A, Aksu H, Uluagac AS, Conti M (2018) A survey on homomorphic encryption schemes. ACM Comput Surv 51(4):1–35. https://doi.org/10.1145/3214303
Ahmad-Reza Sadeghi: Trusting The Trust Anchor: the struggle and challenges of trusted computing (2021). https://www.esat.kuleuven.be/cosic/events/secsi2020/2021/10/12/ahmad-reza-sadeghi-tu-darmstadt-trusting-the-trust-anchor-the-struggle-and-challenges-of-trusted-computing/ Accessed 23 Nov, 2021
Anderson R (2020) Security engineering: a guide to building dependable distributed systems, 3rd edn. Wiley, Hoboken, pp 1–1232
Arthur W, Challener D (2015) A practical guide to TPM 2.0: using the new trusted platform module in the new age of security. Apress, Berkeley, CA
Birkholz H, Thaler D, Richardson M, Smith N, Pan W (2023) Remote ATtestation procedureS (RATS) Architecture. RFC Editor. https://doi.org/10.17487/RFC9334. https://www.rfc-editor.org/info/rfc9334
Confidential Computing Consortium: Whitepaper feedback from Muhammad Usama Sardar, Issue #77 (2020). https://github.com/confidential-computing/governance/issues/77 Accessed 13 Sept, 2021
Confidential Computing consortium: a technical analysis of confidential computing, v1.2 (2021a). https://confidentialcomputing.io/wp-content/uploads/sites/85/2022/11/CCC-A-Technical-Analysis-of-Confidential-Computing-v1.2_updated_2022-11-02.pdf
Confidential Computing Consortium: confidential computing: hardware-based trusted execution for applications and data, v1.2 (2021b). https://confidentialcomputing.io/wp-content/uploads/sites/85/2021/03/confidentialcomputing_outreach_whitepaper-8-5x11-1.pdf
Confidential Computing Consortium: confidential computing consortium scope (2021c). https://confidentialcomputing.io/scope/. Accessed 19 Sept, 2021
Confidential Computing Consortium: Webinar: protecting applications and data in use: confidential computing Consortium (2020d). https://confidentialcomputing.io/webinar-outreach/. Accessed 05 Oct, 2021
Costan V, Devadas S (2016) Intel SGX explained. In: IACR Cryptology ePrint Archive. https://eprint.iacr.org/2016/086.pdf
Costan V, Lebedev I, Devadas S (2017a) Secure processors part I: background, taxonomy for secure enclaves and Intel SGX Architecture. Now Publishers Inc. https://doi.org/10.1561/1000000051. https://ieeexplore.ieee.org/document/8186867
Costan V, Lebedev I, Devadas S (2017b) Secure processors part II: Intel SGX security analysis and MIT sanctum architecture. Found Trends Electr Design Autom 11(3):249–361. https://doi.org/10.1561/1000000052
Embleton S, Sparks S, Zou CC (2013) SMM rootkit: a new breed of OS independent malware. Secur Commun Netw 6(12):1590–1605. https://doi.org/10.1002/sec.166
European Commission: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA Relevance). https://eur-lex.europa.eu/eli/reg/2016/679/oj
Evans D, Kolesnikov V, Rosulek M (2018) A pragmatic introduction to secure multi-party computation. foundations and trends® in privacy and security. 2(2-3):70–246. https://doi.org/10.1561/3300000019
Fontaine C, Galand F (2007) A survey of homomorphic encryption for non-specialists. EURASIP J Inf Secur 2007:1–10. https://doi.org/10.1155/2007/13801
Gentry C (2009) A Fully Homomorphic encryption scheme. PhD thesis, Stanford University. https://crypto.stanford.edu/craig/craig-thesis.pdf
Katz J, Lindell Y (2020) Introduction to modern cryptography, 3rd edn. Taylor & Francis Group
Mulligan, D.P., Petri, G., Spinale, N., Stockwell, G., Vincent, H.J.M.: Confidential Computing: a brave new world. In: International symposium on secure and private execution environment design (SEED), pp. 132–138. IEEE, (2021). https://doi.org/10.1109/SEED51797.2021.00025
Proudler G, Chen L, Dalton C (2014) Trusted computing platforms: TPM2.0 in context. Springer, Cham
Szefer J (2018) Principles of secure processor architecture design. 13:1–173. https://doi.org/10.2200/S00864ED1V01Y201807CAC045
Wikipedia contributors: Trusted Computing: Wikipedia, The Free Encyclopedia (2021). https://en.wikipedia.org/wiki/Trusted_Computing. Accessed 23 Nov, 2021

Publisher's Note