Trusted Heterogeneous Disaggregated Architectures

Atsushi Koshiba
Technical University of Munich
atsushi.koshiba@tum.de

Felix Gust
Technical University of Munich
gustf@in.tum.de

Julian Pritzi
Technical University of Munich
julian.pritzi@tum.de

Anjo Vahldiek-Oberwagner
Intel Labs
anjovahldiek@gmail.com

Nuno Santos
INESC-ID/Instituto Superior Técnico, University of Lisbon
nuno.m.santos@tecnico.ulisboa.pt

Pramod Bhatotia
Technical University of Munich
pramod.bhatotia@tum.de

Abstract

The rising performance demands and increasing heterogeneity in cloud data centers lead to a paradigm shift in the cloud infrastructure, from monolithic servers to a *disaggregated architecture*. In a multi-tenant cloud, users should be able to leverage trusted computing to protect their applications from untrusted parties. While Trusted Execution Environments (TEEs) are a well-known technique to realize trusted computing on monolithic servers, we cannot adopt existing TEE technologies to the disaggregated architecture due to their distributed nature and heterogeneity of devices. To address these challenges, we propose *trusted heterogeneous disaggregated architectures*, which allows cloud users to construct *virtual TEEs (vTEEs)*: TEE-based, secure, isolated environments assembled with any combination of disaggregated components.

CCS Concepts

• **Security and privacy** → **Trusted computing**; • **Computer systems organization** → **Cloud computing**.

Keywords

Resource Disaggregation, Trusted Computing, Hardware-Software Co-Design

ACM Reference Format:
Atsushi Koshiba, Felix Gust, Julian Pritzi, Anjo Vahldiek-Oberwagner, Nuno Santos, and Pramod Bhatotia. 2023. Trusted Heterogeneous Disaggregated Architectures. In *14th ACM SIGOPS Asia-Pacific Workshop on Systems (APSys '23), August 24–25, 2023, Seoul, Republic of Korea.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3609510.3609812

ACM ISBN 979-8-4007-0305-8/23/08…$15.00
https://doi.org/10.1145/3609510.3609812

1 Introduction

The rising performance demands and increasing heterogeneity in cloud data centers trigger a paradigm shift in the cloud infrastructure, from monolithic servers to a *disaggregated architecture*. Hardware acceleration using heterogeneous cores (GPUs, FPGAs, ASICs) achieves significant performance improvement of modern cloud workloads, such as machine learning [12, 18, 22, 24, 40, 52, 58]. These heterogeneous accelerators make it difficult and inflexible for monolithic servers to offer machine configurations for applications' demands [48]. Whereas resource disaggregation, which is expected to overcome the limitation of monolithic servers, distributes every hardware component over the network and allows users to choose any number and combination of CPUs, accelerators, memories, and disks according to their requirements. The resource disaggregation drastically improves resource utilization, scalability, and flexibility for managing heterogeneous devices [15, 33, 36, 44, 46, 48, 50].

At the same time, many modern cloud workloads, such as AI-based intelligent services, deal with large amounts of confidential and security-sensitive private data: medical records, voice recordings, and financial information [14, 19, 34, 47]. In this case, in addition to the data being operated on, the machine learning model itself, its calculations, and queries must not be exposed to or manipulated by any unauthorized party, including software run by other tenants, OS/hypervisors, devices, and networking infrastructure. Therefore, we not only need to protect data in transit between distributed heterogeneous compute and storage components, but before any component is involved in any computation, we also need to attest to its authenticity and integrity to the user.

To ensure code integrity and provide strong isolation from untrusted parties, Trusted Execution Environments (TEEs) have been widely studied for decades [11, 31, 32, 59, 60]. TEEs offer a hardware-assisted isolated sandbox to execute security-sensitive workloads, where application code and data are protected from other tenants and applications, even from compromised privileged software. TEEs have been proposed not only for commodity CPUs (e.g., Intel® SGX [11], ARM TrustZone [9]) but also heterogeneous accelerators such as GPUs [30, 51] and FPGAs [54, 57].

Although TEEs offer essential security features in today's cloud environments, such hardware-assisted isolation has not been exploited well for the disaggregated architecture [48, 50]. Moreover, existing TEE technologies do not fit the emerging disaggregated architecture due to the customizability and heterogeneity; most existing TEE technologies are device-specific, and their protection domains are limited to the target devices. However, the disaggregated architecture distributes user code and data across an arbitrary set of various hardware components, making it difficult for such device-specific TEEs to protect all the domains used by an application. These facts motivate us to *propose a new hardware and software design that offers TEE-based confidential computing on disaggregated heterogeneous architectures.*

The research gap. Particularly, we highlight three key challenges to establishing trustworthy isolated execution environments on the disaggregated architecture.

1. Heterogeneity of disaggregated components. As we discussed, the heterogeneity of disaggregated computing devices makes it challenging to ensure the end-to-end security of workloads across various hardware components. A prior work [13] proposes a way to securely bridge TEE-enabled and TEE-disabled devices in distributed systems. However, heterogeneous devices from multiple vendors pose another problem; each of them is likely with its own root of trust and security posture. For instance, CPUs are dominated by several manufacturers with similar security properties (i.e., Intel, AMD, or ARM). Harmonizing that security posture is complex and leads to unforeseen issues. Hence, we need a more generic approach that offers unified interfaces and common security primitives applicable to every hardware component.

2. Data distribution through the untrusted network. Unlike monolithic servers, hardware components in disaggregated architectures are distributed and loosely connected through unsecured data paths; data confidentiality and integrity are potentially compromised because user data travel through the bus or live in shared memory. Moreover, since memory/storage devices shared among applications are exposed to the untrusted network, they are at risk of memory access-based side-channel attacks [6, 21, 49, 56]. Conventional TEE technologies [16, 59] realize memory isolation by serving dedicated memory space on its local system. However, limiting available resources loses the advantages of resource disaggregation regarding flexibility and scalability.

3. Secure domain isolation across disaggregated components. Even if all devices have the same security properties/functionalities, building an isolated domain across distributed hardware components is not easy; it is because the user's machine configuration in the disaggregated architecture elastically changes according to application requirements, making the attestation phase particularly complex. A few prior studies propose TEE-based secure computation for rack-scale distributed systems [13, 60], where a secure microcontroller is responsible for access control between devices in a rack. However, the prior studies either do not scale due to per-rack coarse-grained management [60], or rely on CPU TEEs for remote attestation [13].

In the end, we answer the following fundamental question: *How can we construct TEEs from user-defined disaggregated heterogeneous components without losing flexibility and elasticity?*

Proposal. This paper introduces an initial hardware/OS co-design for constructing virtual TEEs comprising any combination of disaggregated processing and memory/storage resources. First, we discover the minimal hardware features required to solve the aforementioned key challenges with dedicated hardware components. Second, for our components, we design a custom microkernel-based OS that contains a capability management system, local memory management, and trustworthy network-isolated communication primitives. From a user's perspective, the OS exposes required trusted computing functionalities, for example, remote attestation.

2 Overview

We propose a *trustworthy disaggregated heterogeneous architecture* that serves a secure TEE constructed from disaggregated heterogeneous resources. The proposed architecture offers a *virtual TEE (vTEE)*, a secure, isolated customizable sandbox comprising hardware devices selected by an application. Any other running applications and adversaries cannot access or tamper with confidential data inside the vTEE.

Figure 1 illustrates the trustworthy disaggregated heterogeneous architecture, and Figure 2 shows a user's perspective. We categorize two hardware domains according to device types: *worker domain* and *data domain*. Compute devices such as CPUs, GPUs, and FPGAs are classified in the worker domain and called *Worker Elements (WEs)*, whereas memory/storage devices such as DRAM, NVM, and SSDs are classified in the data domain and called *Data Elements (DEs)*. Cloud users can choose any combination of WEs and DEs to construct vTEEs based on the application's performance demands and hardware requirements. The disaggregated resource management and isolation are handled by the *Trustworthy Disaggregated Operating System (TDOS)*, a microkernel-based OS running on trusted hardware modules, *Worker Isolation Unit (WIU)* and *Data Isolation Unit (DIU)*,

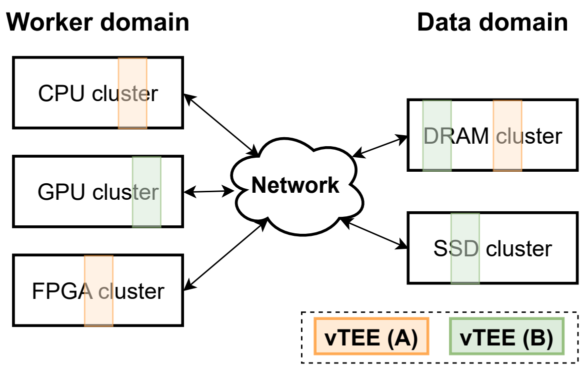

Figure 1: An overview of the trustworthy disaggregated heterogeneous architecture.

which are embedded in each WE/DE. The proposed architecture allows users to run multiple vTEEs at the same time. However, all the vTEEs are transparently isolated from each other, and they cannot read or manipulate security-sensitive data owned by the other vTEEs. To realize the vTEE management, we co-design hardware features and system software specialized for the disaggregated architecture.

2.1 Threat model

We consider an adversary who wants to perform passive or active attacks on a genuine vTEE, which he does not control. This implies that attacks on the confidential application, which use security vulnerabilities of the application itself or originate from the same vTEE, are the responsibility of the vTEE owner.

We trust that the WIU and DIU are implemented correctly and the software running on the secure controller is correct and booted using secure boot. We also trust that the processing elements (PEs), i.e., CPU, GPU, and FPGA, of the WEs are integrated with the WIU in a secure way. We trust that this secure integration guarantees the authenticity of the PE as well as the confidentiality of the communication between WIU and PE. This could potentially be realized using a PE that is compatible with technologies such as SGX or PCIe-TDISP. We do not trust any software running inside a vTEE as an attacker may acquire genuine access to a vTEE to use as a basis for an attack. We do not trust any local memory of WEs and any memory pools of DEs, as the stored data can potentially be manipulated by an adversary and is thus required to be encrypted and integrity protected before leaving the trusted domain. We also do not trust the network as an adversary may operate the network or is otherwise capable of reading, modifying, and deleting exchanged messages. We do trust in the correctness and security properties of the cryptographic primitives and protocols we use to realize our design.

We consider both a passive adversary who wants to obtain confidential information from the vTEE as well as an active

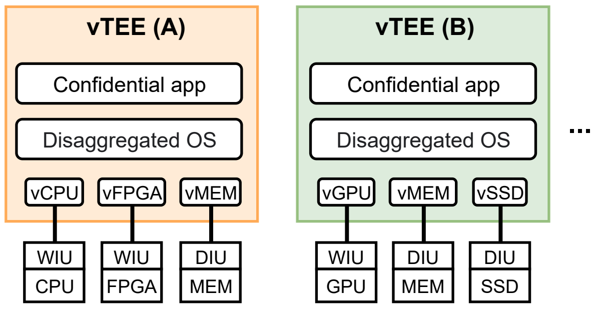

Figure 2: A virtual TEE (*vTEE*) from a user perspective.

adversary who wants to interfere with the proper execution of a genuine vTEE. The two classes of adversaries we consider are pure software and hardware adversaries:

Software adversary. An adversary without physical access but control over any number of vTEEs has a restricted set of attack vectors. We ensure that the adversary can not directly influence the execution of other vTEEs by introducing custom hardware that handles the isolation of vTEEs (§ 3). However, the adversary is still able to make use of side-channel attacks (e.g., memory-based [6, 21, 49, 56]) to obtain additional information.

Hardware adversary. An adversary with full access to the hardware and network can trivially drop all messages or power down machines to perform a Denial of Service (DoS) attack. Hence, mitigations of DoS attacks are not further considered for adversaries with physical access.

2.2 Design goals

We define three design goals of the trustworthy disaggregated heterogeneous architecture, which correspond to the aforementioned three key challenges.

1. Unified trusted hardware primitives. We propose a hardware-assisted resource abstraction mechanism to preserve the elasticity and flexibility of the disaggregated heterogeneous architecture. We design two common hardware units, *WIU and DIU*, which are designed for WEs and DEs, respectively. They provide the minimum required hardware features to construct vTEEs: unified communication interfaces, hardware root-of-trust, attestation, and memory controllers.

2. Distributed data sharing. To securely share DEs among vTEEs owned by different tenants, we propose an inter-TEE data isolation mechanism realized by the microkernel-based accessibility control. The proposed system confines a user application in assigned WEs; their binaries are only executable on the assigned WEs, which are physically isolated through the network.

3. vTEE initialization. We present a secure yet user-friendly way of initializing a vTEE, which requires security primitives specifically tailored to a distributed TEE. First, a secure

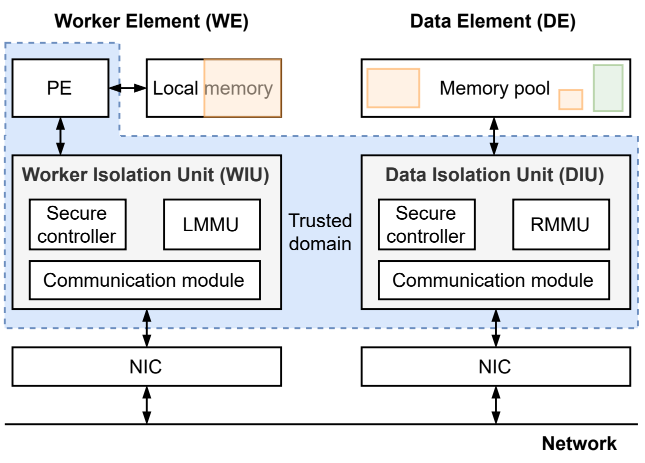

Figure 3: Worker Isolation Unit (WIU) and Data Isolation Unit (DIU).

communication interface among disaggregated devices is indispensable for constructing vTEEs through an untrusted network. To realize this, we propose a novel hardware design that offers a network-level isolation mechanism, which establishes peer-to-peer encrypted communication channels (e.g., TLS/SSL) between any combination of disaggregated devices. Second, the user must establish trust in all WEs involved in the vTEE. We achieve this by offering remote attestation of individual WEs directly to the user.

3 Design and Implementation

3.1 Unified secure hardware modules

We propose minimal hardware components to establish vTEEs on the disaggregated architecture. Figure 3 represents the hardware design. The proposed architecture offers two common hardware units: *Worker Isolation Unit (WIU)* for WEs and *Data Isolation Unit (DIU)* for DEs. WIU is responsible for isolating each PE from unauthorized applications. On the other hand, DIU is responsible for securely sharing memory/storage devices among vTEEs. Both WIUs and DIUs consist of three hardware components. The *secure controller* acts as the hardware root-of-trust and provides trusted computing features such as data integrity/freshness measurement, remote attestation, and data sealing. It is also responsible for executing the TDOS. The *communication module* is a data transfer module used to create trustworthy communication channels between a pair of WIUs/DIUs. Any packets addressed to or coming from unauthorized WEs/DEs are dropped. Lastly, the *Local and Remote MMUs (LMMU, RMMU)* cooperate to mediate both local memories on WIUs and remote memory/storage on DIUs while preventing unauthorized accesses.

We plan to implement the hardware components upon FPGA-based SmartNICs such as Xilinx Alveo SN1000 [55] and Intel® IPU [29]. We will implement a prototype of the

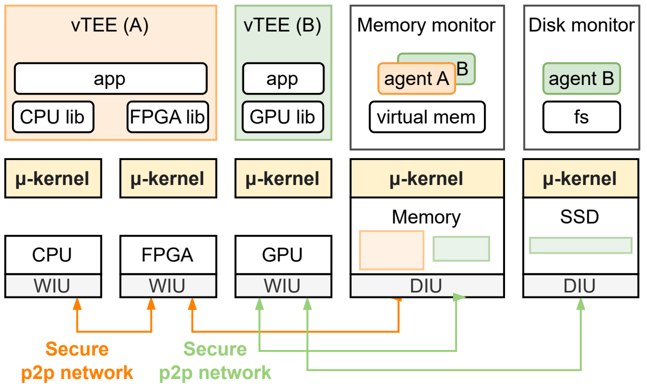

Figure 4: Trustworthy Dissagregated OS (TDOS).

secure controller based on OpenTitan [45], an open-source RISC-V-based hardware root-of-trust. For the communication module, we plan to build a packet filtering mechanism based on Corundum [17], an open-source FPGA packet controller. Finally, we plan to implement the local and remote MMUs based on TLBs/MMUs offered by Coyote [39], an open-source OS abstraction for on-FPGA accelerators.

3.2 Trustworthy disaggregated OS

On the software side, we design a *Trustworthy Dissagregated OS (TDOS)*, a microkernel-based OS that manages disaggregated resources using *capabilities* [26, 27, 53]. A capability represents an unforgeable token uniquely identifying a resource and its access rights. The TDOS manages the accessibility of vTEEs by delegating or revoking the capabilities of disaggregated resources. It also abstracts the concrete WEs/DEs to present a unified programming model among various heterogeneous devices.

Figure 4 illustrates the system design of the TDOS. We adopt distributed OS approaches [3, 26, 27, 50] to the disaggregated architecture, where multiple kernels are distributed and executed on individual devices. The TDOS kernels (μ-kernel) run on the secure controllers embedded in WIUs/DIUs. The kernels provide capability control, secure channel establishment, and trusted computing functions (e.g., remote attestation) with hardware features of WIUs/DIUs. The secure communication channels are only established among WEs/DEs whose capabilities are delegated to each other. Therefore, any unauthorized requests are rejected.

The TDOS initially targets the RISC-V architecture to support the OpenTitan core. We plan to build the TDOS kernel based on the formally verified seL4 microkernel [35], which offers strong address space isolation and a capability-based access control mechanism. We plan to extend seL4's capabilities to represent access rights to disaggregated WEs/DEs. The microkernel architecture allows us to build the TDOS around a minimal kernel with strong security guarantees.

3.3 TEE life cycle

The life cycle of an individual vTEE involves the following.

vTEE creation request. Cloud users request the processing and memory/storage resources they require for the vTEE from a gateway server offered by the cloud service provider. Afterward, the gateway server informs the WIUs about the vTEE request. A vTEE's WEs are statically allocated during vTEE creation because they have to be attested by the user before executing trusted user code. In contrast, the user does not directly interact with DEs managed by DIUs. Thus, DEs can be dynamically assigned to an attested vTEE at runtime.

Remote attestation. Before the vTEE starts executing, trust needs to be established in all WEs via remote attestation. Traditionally, a Trusted Third Party (TTP) has been responsible for this process. Involving a TTP, however, would increase our TCB beyond our trusted hardware components [2, 41]. Instead, to keep the TCB minimal and improve scalability, we want WEs to attest to each other directly via their WIUs. We will develop a remote attestation mechanism based on MAGE [10] that allows a group of enclaves – in our case, a group of WEs belonging to the same vTEE – to attest each other in a peer-to-peer fashion without the need for a TTP. To achieve this, every WIU stores the expected measurements of all WEs of the same vTEE in its local storage, separate from the vTEE code and data. While storing all measurements at vTEE creation time requires a static arrangement of WEs, it eliminates the need for a TTP, thereby reducing the TCB to just the WIUs themselves.

For the initial attestation of all WEs, the gateway server sends an attestation request for the whole vTEE to any WIU, which we call the leader in the following. The user is informed which WIU is the leader, after which the user and leader perform mutual authentication. From this point onward, the gateway server is no longer involved in the attestation process. Afterwards, the leader initiates mutual remote attestation with all other WIUs. The domain of collective remote attestation, which involves attesting a vast number of devices efficiently, has been explored in prior research. [1, 2, 7, 8, 28, 37, 38]. We will evaluate various collective remote attestation schemes in terms of scalability and security. Finally, the leader sends the attestation result to the gateway server which informs the user whether the attestation of all WEs succeeded.

Memory and storage allocation. DEs are dynamically allocated to a vTEE at runtime. Accessibility and ownership of memory/storage resources are managed by capability delegation and revocation between the TDOS kernels. After a DIU has confirmed a WIU's access rights, the two components establish a trusted peer-to-peer connection. Afterward, the WIU can request or release memory/storage from the DIU via a unified API provided by TDOS. The TDOS instance running on the DIU creates a proxy (agent) that handles memory/storage allocation and access on the underlying device.

vTEE execution. During the execution of the application inside the vTEE, WIUs and DIUs allow for decentralized peer-to-peer communication between components. For example, inside one vTEE, an FPGA may read data from disaggregated memory that was previously populated by a GPU. Alternatively, the GPU may send the data directly to the FPGA. After a vTEE has finished executing the user application, its associated resources are freed and available for a new vTEE.

4 Related Work

OSes for disaggregated hardware. There are several prior studies of operating systems for disaggregated computing, including LegoOS [48] and FractOS [50]. However, their discussion is not sufficient in terms of trusted computing on disaggregated architectures. LegoOS allows a user application to run on multiple disaggregated processor, memory, and storage components. The researchers do not mention security properties. FractOS is a distributed OS for disaggregated heterogeneous architectures, treating heterogeneous devices as first-class citizens. Its trust model poses an optimistic assumption that tenants trust other tenant services or third-party tools running in the same cluster. Unlike these previous approaches, security and trusted computing are at the core of our approach.

Trusted microkernel-based OSes. OSes for various trusted computing functions relying on the microkernel architecture have been developed, including MicroTEE [31], UniTEE [23], SANCTUARY [5], and a software-only TPM for RISC-V processors [4]. Building upon a minimal microkernel, like the formally verified seL4 kernel [35], such OSes can minimize their TCB. While these systems do not target disaggregated or heterogeneous architectures and are not fit for our solution, they demonstrate the feasibility of leveraging a microkernel-based OS for trusted computing.

TEEs. Security properties for disaggregated computing have not been well studied yet, despite an increasing number of research [25, 43, 48, 50]. In contrast to this, TEEs have been widely analyzed for their security properties [11, 31, 32, 59, 60]. However, existing TEEs have been developed mainly for CPU-centric architectures: They often rely on specific hardware features [11] or follow a software-based approach [42, 59] that is not applicable to heterogeneous architectures. Our hardware/OS co-design will be custom-built for such architectures.

Heterogeneous TEEs. Custom TEEs for heterogeneous architectures exist but are very limited; they only support a particular accelerator such as GPUs [30, 51], FPGAs [54, 57], or, in the case of multiple accelerators connected over a bus, do not allow for spatial sharing of the accelerators [60]. In contrast, our approach aims to provide a unified, trusted computing framework that supports multiple kinds of accelerators from different vendors and allows for the granular assignment of individual accelerators to user applications.

Distributed TEEs. HETEE [60] proposes a heterogeneous TEE implementation that has similarities to a distributed TEE. However, it does not scale well due to its requirement for a centralized security controller, which limits the domain to a single PCIe bus. A similar approach presented in [13] promises better scalability by allowing for multiple security controllers, where each rack containing devices without TEE hardware is equipped with one security controller. Both solutions do not allow spatial sharing of hardware components. Our design avoids relying on centralized security controllers and instead offers a peer-to-peer network across devices that can individually be incorporated into vTEEs.

5 Discussion

Previously attested device is compromised. Before a device executes user code, it has to be attested. If the device is malfunctioning or compromised during operation, we would like to detect it and immediately invalidate the attestation. This will require not only continuous validation of the state of the device but also a mechanism to inform other nodes that the corresponding device is no longer part of the virtual TEE. Can we achieve this functionality?

Scalability. By implementing a decentralized system of components communicating peer-to-peer, we can gain improved scalability compared to a system with a centralized entity mediating all communication. However, as all components share the same network, excessive data loads exchanged between one set of components could congest the network and affect the communication of another set of components. There may also be scenarios, like revoking an attestation, where information has to be sent from or to a large set of components at the same time.

Compatibility with existing APIs. Even though our solution will be custom-built for disaggregated heterogeneous architectures, we may be able to present established TEE APIs, like the GlobalPlatform APIs [20], to the user application. However, such APIs might prove to be too specifically tailored to traditional centralized TEEs.

WIU/DIU is compromised. While we treat the WIU and DIU as trusted, we should nevertheless explore strategies to handle a malfunctioning or compromised WIU/DIU. We want to avoid an authoritative centralized security controller, but WIUs/DIUs could, for example, have different levels of privilege to contain the damage a single component could cause. How would we determine which components should be assigned a higher privilege level? Would we inevitably need to introduce centralization to handle such a scenario?

Resource starvation. Because DEs are potentially shared by multiple vTEE applications, a malicious vTEE might attempt to perform a DoS attack by starving other genuine vTEEs of their access to the DE (see § 2.1). One way to mitigate such attacks is by having usage quotas for the DEs. For example, vTEEs would have a limit on the maximum memory size it can consume and a maximum number of expensive or inexpensive requests it can make to the DEs. It seems like a simple solution to the problem, but it comes at the cost of the flexibility of the vTEEs. What is the right balance between flexibility and protecting DEs from DoS attacks? Are there better solutions to this problem than the introduction of usage quotas?

Switching between trusted and untrusted modes. We will assess if a user application should be able to switch between a trusted and untrusted mode. There could also be different levels of trust or privilege at which a single component can run. This could allow for a minimal virtual TEE complementing a larger application running in a less privileged mode.

Acknowledgments

This work was partially supported by an ERC Starting Grant (ID: 101077577), and national funds through Fundação para a Ciência e a Tecnologia (FCT) under project UIDB/50021/2020 and the SmartRetail project (ref. C6632206063-00466847) financed by IAPMEI.

References

[1] M. Ammar, M. Washha, G. S. Ramachandran, and B. Crispo. slimIoT: Scalable Lightweight Attestation Protocol For the Internet of Things.

[2] N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann. SEDA: Scalable Embedded Device Attestation. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 964–975. Association for Computing Machinery.

[3] A. Baumann, P. Barham, P.-E. Dagand, T. Harris, R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania. The multikernel: A new os architecture for scalable multicore systems. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles*, SOSP '09, page 29–44, New York, NY, USA, 2009. Association for Computing Machinery.

[4] M. Boubakri, F. Chiatante, and B. Zouari. Towards a firmware TPM on RISC-V. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 647–650, Feb. 2021. ISSN: 1558-1101.

[5] F. Brasser, D. Gens, P. Jauernig, A.-R. Sadeghi, and E. Stapf. SANCTUARY: ARMing TrustZone with User-space Enclaves. In *Proceedings 2019 Network and Distributed System Security Symposium*, San Diego, CA, 2019. Internet Society.

[6] F. Brasser, U. Müller, A. Dmitrienko, K. Kostiainen, S. Capkun, and A.-R. Sadeghi. Software grand exposure: SGX cache attacks are practical. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC, Aug. 2017. USENIX Association.

[7] X. Carpent, K. ElDefrawy, N. Rattanavipanon, and G. Tsudik. Lightweight Swarm Attestation: A Tale of Two LISA-s. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pages 86–100. Association for Computing Machinery.

[8] X. Carpent, N. Rattanavipanon, and G. Tsudik. ERASMUS: Efficient Remote Attestation via Self- Measurement for Unattended Settings.

[9] D. Cerdeira, N. Santos, P. Fonseca, and S. Pinto. Sok: Understanding the prevailing security vulnerabilities in trustzone-assisted tee systems.

In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1416–1432, 2020.

[10] G. Chen and Y. Zhang. MAGE: Mutual Attestation for a Group of Enclaves without Trusted Third Parties. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4095–4110, 2022.

[11] V. Costan and S. Devadas. Intel sgx explained. Cryptology ePrint Archive, Paper 2016/086, 2016. https://eprint.iacr.org/2016/086.

[12] H. Cui, H. Zhang, G. R. Ganger, P. B. Gibbons, and E. P. Xing. Geeps: Scalable deep learning on distributed gpus with a gpu-specialized parameter server. In *Proceedings of the Eleventh European Conference on Computer Systems*, EuroSys '16, New York, NY, USA, 2016. Association for Computing Machinery.

[13] A. Dhar, S. Sridhara, S. Shinde, S. Capkun, and R. Andri. Empowering data centers for next generation trusted computing, 2022.

[14] J. Domingo-Ferrer, O. Farràs, J. Ribes-González, and D. Sánchez. Privacy-preserving cloud computing on sensitive data: A survey of methods, products and challenges. *Comput. Commun.*, 140(C):38–60, may 2019.

[15] P. Faraboschi, K. Keeton, T. Marsland, and D. Milojicic. Beyond processor-centric operating systems. In *15th Workshop on Hot Topics in Operating Systems (HotOS XV)*, Kartause Ittingen, Switzerland, May 2015. USENIX Association.

[16] A. Ferraiuolo, A. Baumann, C. Hawblitzel, and B. Parno. Komodo: Using verification to disentangle secure-enclave hardware from software. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, page 287–305, New York, NY, USA, 2017. Association for Computing Machinery.

[17] A. Forencich, A. C. Snoeren, G. Porter, and G. Papen. Corundum: An open-source 100-gbps nic. In *2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 38–46, 2020.

[18] D. Ghimire, D. Kil, and S.-h. Kim. A survey on efficient convolutional neural networks and hardware acceleration. *Electronics*, 11(6), 2022.

[19] A. Gholami and E. Laure. Security and privacy of sensitive data in cloud computing : A survey of recent developments. In *Computer Science & Information Technology (CS & IT)*. Academy & Industry Research Collaboration Center (AIRCC), dec 2015.

[20] GlobalPlatform Technology. TEE System Architecture v1.3 | GPD_spe_009. https://globalplatform.org/specs-library/tee-system-architecture/.

[21] J. Götzfried, M. Eckert, S. Schinzel, and T. Müller. Cache attacks on intel sgx. In *Proceedings of the 10th European Workshop on Systems Security*, EuroSec'17, New York, NY, USA, 2017. Association for Computing Machinery.

[22] J. Gu, M. Chowdhury, K. G. Shin, Y. Zhu, M. Jeon, J. Qian, H. Liu, and C. Guo. Tiresias: A GPU cluster manager for distributed deep learning. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, pages 485–500, Boston, MA, Feb. 2019. USENIX Association.

[23] J.-Y. Gu, H. Li, Y.-B. Xia, H.-B. Chen, C.-G. Qin, and Z.-Y. He. Unified Enclave Abstraction and Secure Enclave Migration on Heterogeneous Security Architectures. *Journal of Computer Science and Technology*, 37(2):468–486, Apr. 2022.

[24] K. Guo, S. Han, S. Yao, Y. Wang, Y. Xie, and H. Yang. Software-hardware codesign for efficient neural network acceleration. *IEEE Micro*, 37(2):18–25, 2017.

[25] Z. Guo, Y. Shan, X. Luo, Y. Huang, and Y. Zhang. Clio: A hardware-software co-designed disaggregated memory system. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS 2022, page 417–433, New York, NY, USA, 2022. Association for Computing Machinery.

[26] M. Hille, N. Asmussen, P. Bhatotia, and H. Härtig. SemperOS: A distributed capability system. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 709–722, Renton, WA, July 2019. USENIX Association.

[27] M. Hille, N. Asmussen, H. Härtig, and P. Bhatotia. A heterogeneous microkernel os for rack-scale systems. In *Proceedings of the 11th ACM SIGOPS Asia-Pacific Workshop on Systems*, APSys '20, page 50–58, New York, NY, USA, 2020. Association for Computing Machinery.

[28] A. Ibrahim, A.-R. Sadeghi, G. Tsudik, and S. Zeitouni. DARPA: Device Attestation Resilient to Physical Attacks. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, WiSec '16, pages 171–182. Association for Computing Machinery.

[29] Intel. Intel infrastructure processing unit (ipu) and smartnics. https://www.intel.de/content/www/de/de/products/network-io/smartnic.html, 2022.

[30] I. Jang, A. Tang, T. Kim, S. Sethumadhavan, and J. Huh. Heterogeneous isolated execution for commodity gpus. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19, page 455–468, New York, NY, USA, 2019. Association for Computing Machinery.

[31] D. Ji, Q. Zhang, S. Zhao, Z. Shi, and Y. Guan. Microtee: Designing TEE OS based on the microkernel architecture. In *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 13th IEEE International Conference On Big Data Science And Engineering, TrustCom/BigDataSE 2019, Rotorua, New Zealand, August 5-8, 2019*, pages 26–33. IEEE, 2019.

[32] L. Kang, Y. Xue, W. Jia, X. Wang, J. Kim, C. Youn, M. J. Kang, H. J. Lim, B. Jacob, and J. Huang. Iceclave: A trusted execution environment for in-storage computing. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, page 199–211, New York, NY, USA, 2021. Association for Computing Machinery.

[33] K. Katrinis, D. Syrivelis, D. Pnevmatikatos, G. Zervas, D. Theodoropoulos, I. Koutsopoulos, K. Hasharoni, D. Raho, C. Pinto, F. Espina, S. Lopez-Buedo, Q. Chen, M. Nemirovsky, D. Roca, H. Klos, and T. Berends. Rack-scale disaggregated cloud data centers: The dredbox project vision. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 690–695, 2016.

[34] L. M. Kaufman. Data security in the world of cloud computing. *IEEE Security & Privacy*, 7(4):61–64, 2009.

[35] G. Klein, J. Andronick, K. Elphinstone, G. Heiser, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. Sel4: Formal verification of an operating-system kernel. *Commun. ACM*, 53(6):107–115, jun 2010.

[36] A. Klimovic, C. Kozyrakis, E. Thereska, B. John, and S. Kumar. Flash storage disaggregation. In *Proceedings of the Eleventh European Conference on Computer Systems*, EuroSys '16, New York, NY, USA, 2016. Association for Computing Machinery.

[37] F. Kohnhäuser, N. Büscher, S. Gabmeyer, and S. Katzenbeisser. SCAPI: A scalable attestation protocol to detect software and physical attacks. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '17, pages 75–86. Association for Computing Machinery.

[38] F. Kohnhäuser, N. Büscher, and S. Katzenbeisser. SALAD: Secure and Lightweight Attestation of Highly Dynamic and Disruptive Networks. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, ASIACCS '18, pages 329–342. Association for Computing Machinery.

[39] D. Korolija, T. Roscoe, and G. Alonso. Do OS abstractions make sense on FPGAs? In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pages 991–1010. USENIX Association, Nov. 2020.

[40] G. Lacey, G. W. Taylor, and S. Areibi. Deep learning on fpgas: Past, present, and future, 2016.

[41] B. Lampson, M. Abadi, M. Burrows, and E. Wobber. Authentication in distributed systems: Theory and practice. *ACM Trans. Comput. Syst.*, 10(4):265–310, nov 1992.

[42] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanović, and D. Song. Keystone. *Proceedings of the Fifteenth European Conference on Computer Systems*, 2020.

[43] S.-s. Lee, Y. Yu, Y. Tang, A. Khandelwal, L. Zhong, and A. Bhattacharjee. Mind: In-network memory management for disaggregated data centers. In *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, SOSP '21, page 488–504, New York, NY, USA, 2021. Association for Computing Machinery.

[44] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt, and T. F. Wenisch. Disaggregated memory for expansion and sharing in blade servers. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, page 267–278, New York, NY, USA, 2009. Association for Computing Machinery.

[45] lowRISC contributors. The opentitan project. https://opentitan.org, 2022.

[46] V. Nitu, B. Teabe, A. Tchana, C. Isci, and D. Hagimont. Welcome to zombieland: Practical and energy-efficient memory disaggregation in a datacenter. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, New York, NY, USA, 2018. Association for Computing Machinery.

[47] M. S. Riazi, B. Darvish Rouani, and F. Koushanfar. Deep learning on private data. *IEEE Security & Privacy*, 17(6):54–63, 2019.

[48] Y. Shan, Y. Huang, Y. Chen, and Y. Zhang. LegoOS: A disseminated, distributed OS for hardware resource disaggregation. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 69–87, Carlsbad, CA, Oct. 2018. USENIX Association.

[49] S. Shinde, Z. L. Chua, V. Narayanan, and P. Saxena. Preventing page faults from telling your secrets. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, page 317–328, New York, NY, USA, 2016. Association for Computing Machinery.

[50] L. Vilanova, L. Maudlej, S. Bergman, T. Miemietz, M. Hille, N. Asmussen, M. Roitzsch, H. Härtig, and M. Silberstein. Slashing the disaggregation tax in heterogeneous data centers with fractos. In *Proceedings of the Seventeenth European Conference on Computer Systems*, EuroSys '22, page 352–367, New York, NY, USA, 2022. Association for Computing Machinery.

[51] S. Volos, K. Vaswani, and R. Bruno. Graviton: Trusted execution environments on GPUs. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 681–696, Carlsbad, CA, Oct. 2018. USENIX Association.

[52] Y. E. Wang, G.-Y. Wei, and D. Brooks. Benchmarking tpu, gpu, and cpu platforms for deep learning, 2019.

[53] R. N. M. Watson, J. Anderson, B. Laurie, and K. Kennaway. A taste of capsicum: Practical capabilities for unix. *Commun. ACM*, 55(3):97–104, mar 2012.

[54] K. Xia, Y. Luo, X. Xu, and S. Wei. Sgx-fpga: Trusted execution environment for cpu-fpga heterogeneous architecture. *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021.

[55] A. Xilinx. Alveo sn1000 smartnic accelerator card. https://www.xilinx.com/products/boards-and-kits/alveo/sn1000.html, 2022.

[56] Y. Xu, W. Cui, and M. Peinado. Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *2015 IEEE Symposium on Security and Privacy*, pages 640–656, 2015.

[57] M. Zhao, M. Gao, and C. Kozyrakis. *ShEF: Shielded Enclaves for Cloud FPGAs*, page 1070–1085. Association for Computing Machinery, New York, NY, USA, 2022.

[58] R. Zhao, W. Luk, X. Niu, H. Shi, and H. Wang. Hardware acceleration for machine learning. In *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 645–650, 2017.

[59] S. Zhao, Q. Zhang, Y. Qin, W. Feng, and D. Feng. Sectee: A software-based approach to secure enclave architecture using tee. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 1723–1740, New York, NY, USA, 2019. Association for Computing Machinery.

[60] J. Zhu, R. Hou, X. Wang, W. Wang, J. Cao, B. Zhao, Z. Wang, Y. Zhang, J. Ying, L. Zhang, and D. Meng. Enabling rack-scale confidential computing using heterogeneous trusted execution environment. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1450–1465, 2020.