Seminar

Applied Methods in Network and System Security

High Performance Computing: Trusted Execution Environments

FT 2023

Chair for IT-Security of Software and Data
Second lieutenant
Valentin Pfeil
valentin.pfeil@unibw.de
Representative
Computer Science 2021
Student Division Group 9/A

University of the German Federal Armed Forces in Munich
Department of Computer Science
Institute for Software Engineering

Chair for IT-Security of Software and Data (Eds.): Seminar Applied Methods in Network and System Security, FT 2023
Technical Report No. INF2-2023-3

University of the German Federal Armed Forces in Munich
Department of Computer Science
Institute for Software Engineering
https://www.unibw.de/inf2

Contents

Chapter 1

High-Performance Computing: Trusted Execution Environments

VALENTIN PFEIL

Abstract. *High-Performance Computing (HPC) offers fast processing of workloads for data analysis, simulation and other high-volume jobs. Via the public cloud, it makes supercomputing more accessible to users with limited resources. The viability, performance and usability of HPC were examined. Furthermore, computing in the cloud faces security-related challenges as data needs to be sent and processed to and on the cloud systems. Reliability comes with trustability as HPC platforms on the internet need to be secure. Either the stakeholders fully trust their cloud providers or they operate with their private, respectively hybrid systems.*
In recent times, Trusted Execution Environments (TEE) have become more practical as diverse implementations are already available for confidential HPC in the cloud facing security concerns. It also contributes to Multiparty Computation. The concept ensures secure and private data sharing and makes datasets more valuable for analysis. This paper covers the relevant technologies in this context. Their design and validation process, including the usage of certifications, will be discussed. Additionally, it demonstrates the corresponding threat model and gives a short analysis of the aspects of security, performance and limitations. A clear distinction between previous security solutions and Trusted Execution Environments will be made. Performance data considering the usage of TEE-technologies as AMD SEV and Intel SGX have been evaluated for a diverse collection of HPC benchmarks. This includes scientific computing, graph analytics and emerging scientific computing workloads. Furthermore, it will be identified what the symbioses of HPC and TEE in practical terms mean as the concepts of SCONE, QEMU/Kata and RESTful web application implemented in the Python programming language using Flask framework will be discussed. In the latter, Artificial Intelligence enhances the design of a confidential computing system. Finally, there will be an outlook on newer TEE concepts and their potential impact.

1.1 Introduction

Confidential Computing and, in particular, Confidential High-Performance Computing [31] (HPC) is a more frequently demanded service as the number of security threats to IT infrastructures has increased over the last years [29]. According to the Forrester study [30], IT infrastructure has been modernized after security issues and vulnerabilities have been found respectively by half of IT decision-makers. Refreshing on-premises hardware can be a challenge resulting in delays considering IT projects and priorities of the IT department. In comparison, modernization attracts less attention than new projects. Especially high expenses like these need to be properly addressed and justified. Not without reason there are terms such as "never change a running system". At first sight, reinvestment into the already existing infrastructure seems costly and lavish. In this case, it is crucial to modernize on-premise infrastructure to minimize security targets.
However, it might also be wiser to reconsider strategic decisions considering dynamic outsourcing as a beneficial tool to save money and effort in this case. It is crucial to distinguish the need for computing and data transfer. With the proper security concept in place, information is classified and needs to be properly handled. Each class of information needs its respective set of security measures implemented on the network. Depending on the requirements, it might be worth outsourcing sensitive data to the internet because otherwise, the whole infrastructure for sensitive data processing needs to be in place.
Therefore, there are solutions in the public cloud [32] available for example Infrastructure as a Service (IaaS) provides a platform to generate and interconnect virtual machines and respectively, their services. These machines can be prepared and scaled to solve large problems. Furthermore, considering those systems, a wide set of security features can be implemented.
Concluding, users do not have to invest in on-premise infrastructure to carry out their projects. Generally, cloud platforms on the internet are considered untrusted. Sensitive data processing is not per se to be considered feasible on the internet as it is the responsibility of the cloud provider to provide the respective service. The key issue for the users was and is to ensure trust between the service provider and himself. The provider is believed to ensure enough security measures to protect from all kinds of attacks and prevent failures of the IT operation.
To approach the privacy concerns with the recent developments, new security features have been introduced to the current processor generations. Trusted Execution Environments (TEE) such as Intel Software Guard Extensions (Intel SGX) and AMD Secure Encrypted Virtualization [18] (AMD SEV) are the key technologies to ensure data integrity and confidentiality on the hardware level, concluding that trust begins already with the production of the CPU. These features face several challenges regarding their usability, security and performance. For example, Intel SGX requires specific code implementations to ensure the confidential execution of workloads. During runtime, it is not possible to use OS functions and routines APIs used to access the file system or the user interface to ensure a strong security guarantee [18]. When OS API is accessed, data will be exchanged in the untrusted memory. Another example is the performance as TEE comes with performance penalties depending on the tasks and processing units.
Besides, there are several challenges regarding Single- and Multiuser Computing. Speaking of Single-User Cases, users may execute confidential computation jobs on the internet. Either the

hypervisor or the OS of the system can be compromised. Preservation of data and program integrity and confidentiality are crucial while availability has lower priority. On the other side, HPC workloads often require collaborative workflows. TEE implementations increase the attack surface, but not explicitly due to its HPC nature.
To illustrate Multiparty Computing (MPC), a directed graph [18] consisting of modules (data sources or processing modules) is conceptualized in Figure 1.1. These modules are common elements of the different participants, some of them are declared as *confidential components*. The illustration shows two of them in a collaborative workflow. P_1 consists a private dataset D_3 and a private algorithm C_1. The other participant P_2 has only a private algorithm C_2. All other components are public.

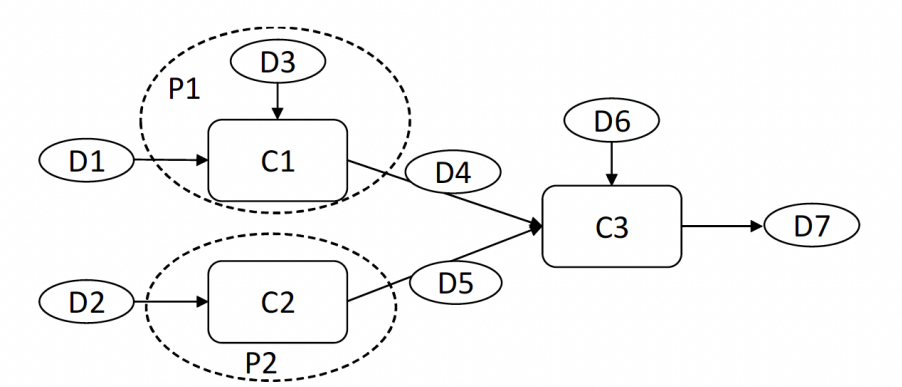

Figure 1.1: **Collaborative workflow [18]**
P_i: Participants who may own private components.
C_i: Processing component.
D_i: Data component.
P_1, P_2: Private components.
Rest: Public components.

In MPC, especially for scientific needs, reproducibility is mandatory. MPC workflows urge to have logging, provenance data analysis (for debugging and optimization et cetera) and reproducibility verification as their key feature set.
Furthermore, especially the private components need to be protected. Curious participants in the workflow and dishonest owners of private components might be assumed as potential adversaries. Considering the private components are running in a TEE, it minimizes the threat. However, the interaction between private and service components still leaves room for optimization. Additionally, Secure Multiparty Computation still has challenges such as Hardware Attestation, Secure Enclave Provisioning, Key and Secret Management and Code Writing Rules [34].

1.2 Preliminaries

1.2.1 Large Problem-Solving

High-Performance Computing [2] (HPC) is a technology that uses a cluster of powerful Central Processing Units. They are parallelly used to process large and multidimensional data sets (Big Data) and to solve complex problems at extremely high speeds. In comparison to default desktops, laptops or servers, HPC systems are significantly faster.
For decades, a supercomputer has been the typical model of an HPC system. It holds a huge amount of processors and their cores. However, this idea has not changed. According to IBM, with a processing speed of 1.102 Exaflops or 1 Trillion floating point operations per second (FLOPS), the US-based Frontier is the fastest supercomputer in the world [10]. HPC solutions as clusters of HPC systems are available on-premise or in the cloud while being used by enterprises or institutions.
The computers in this context handle large problems in scientific environments [1]. HPC supports significantly the development of human knowledge and establishes competitive advantage. There are many domains where HPC is being used, e.g. in the sequencing of DNA, stock trade, algorithm and simulations and processing of artificial intelligence [23] (AI). In the context of automated automobiles, embedded systems [4] such as IoT sensors, radars and GPS generate Big Data that is processed in real-time to induce split-second decisions.
HPC differs from common computing. Problems are divided and parallelly processed on two or more processing units of the HPC systems while default computer process them on their only multi-core processor. Distinctive characteristics of HPC are: *Massively Parallel Computing [7]*, *Clustering* and *High-Performance Components*.
Parallel Computing describes the parallel processing of tasks on two or more servers or CPUs simultaneously. *Massively Parallel Computing* differs via the usage of thousands to millions of CPUs and respective cores.
Clustering is used to interconnect High-Performance computers. A central scheduler manages the HPC workload to be executed in parallel. The computer as a component of these clusters is called *node*. They consist of high-performance components such as CPUs with several cores or Graphical Processing Units (GPUs). GPUs [13] [31] are especially beneficial for accurate mathematical operations, models for machine learning and graphic-intensive tasks. One only cluster potentially consists of more than 100.000 nodes [2].
High-Performance Components are provisioned to optimize the throughput of the cluster. Every component such as network-, memory-, storage- and file systems has high throughput with minimized latencies.
Until the last decade, for many companies, it had been difficult to access HPC [2]. The reason was its costs. The scope covers ownership and leasing of either a supercomputer or an HPC cluster in a local data center. Nowadays, HPC becomes more and more accessible as cloud providers extended their product range with another service: HPC as a Service (HPCaaS). For institutions and companies, it is a much faster, scalable and cost-efficient possibility to benefit from its characteristics. The service covers access to the HPC infrastructure of a cloud service provider, but also services such as analysis of AI or data and HPC knowledge.
HPC in the cloud engages with the following converging developments: *Increasing demand*,

Adoption of Remote Direct Memory Access (RDMA) with better performance, *Wide adoption of HPCaaS*.
Increasing demand is determined as organizations of all kinds are getting more dependent on real-time analysis and competitive advantages resulting from large problem-solving via HPC. The detection of credit card fraud relies more and more on HPC to accelerate its recognition and minimize false alarms while fraudulent activities increase and tactics change. In this context, also collaborative workflows accelerate the search for findings in Big Data within HPC environments as Multiparty Computing accelerates this job.
Remote Direct Memory Access enables an interconnected computer to access the memory of another networking computer without involving its operating system or halting its processes. It contributes to minimizing latencies and maximizing throughput. Existing performant RDMA implementations, including Infiniband, Virtual Interface Architecture and RDMA over Converged Ethernet (RoCE) makes cloud-based HPC possible.
Wide adoption of HPCaaS is nowadays given as every leading public cloud service provider provides HPC services. Because a portion of institutions still has to locally work with HPC workloads, there are also private cloud HPC solutions available. The reasons might be strong regulations or a certain degree of data sensitivity.

1.2.2 Confidential Computing

Isolation of sensitive data in a protected Central Processing Unit (CPU) enclave during processing is called Confidential Computing [8]. It is a cloud computing technology. The protected area is only accessible to the authorized program. To anything else, including the service provider, the secured domain is unreachable. The protection includes contents such as the processed data and the code itself.
As the management of companies wants to rely more and more on flexible cloud solutions, data privacy is imperative. The idea of Confidential Computing is to assure leaders that their data in the cloud is protected and confidential [6] while data integrity is ensured. Ideally, the successful implementation of this technology encourages them to move sensitive data and workloads to the public cloud [5].
Traditional services of cloud providers are encryption services to protect static data, for example in storage and database systems. Also possible is the protection of dynamic data, e.g. while being transferred over the network. There is a security gap that needs to be addressed: data in use. For that, Confidential Computing has been developed which protects the data during processing.
The first state of data before being confidentially processed is unencrypted in the memory. This state leaves several threats open. Therefore it is possible to impose memory dumps just before, during and directly after the operation. Furthermore, root user exploits and other attacks are probable.
Trusted Execution Environments [25] (TEE) address these issues by adding a hardware-based [24] security layer to the CPU. This secured domain is identified as a secure enclave. The concept ensures protection by embedded mechanisms to authorize only privileged applications [26].

This includes the usage of encryption keys and attestation mechanisms. The CPU recognizes if malicious or hacked software is used to get credentials or to access security components. Then, it denies access and cancels further processing of the program.
This technology ensures the protection of sensitive data in memory until the program tells the CPU and its TEE to reveal he data. During runtime, the data is decrypted and not reachable by any other potential stakeholder, such as the operating system or hypervisor, other services and the cloud service provider.
The ultimate goal of Confidential Computing is to enable sensitive workloads on the cloud [18], not explicitly HPC ones. It helps to protect these sensitive workloads while in use. In combination with static and dynamic data encryption with key monitoring, Confidential Computing dismantles the only blocking point to transferring and processing sensitive and highly regulated Big Data and program workloads from a static and inefficient on-premise IT network to a modern, dynamically scalable public cloud platform.
To achieve this, several aspects need to be covered: *Protection of intellectual property*, *Secure Multiparty Computation* [34], *Elimination of cloud service provider concerns*, *Protection of Edge Computing*.
Protection of intellectual property is crucial to keep business intelligence (BI) as proprietary logic, algorithm and entire programs private.
Secure Multiparty Computation enables collaboration between stakeholders to process sensitive data and to create new solutions without exposing unwanted information.
Elimination of cloud service provider concerns encourages organizations to choose Confidential Computing over the cloud as a service as the best solution for one's technical and business needs. Worries about storing and processing business-related and proprietary data, technology and other sensitive assets need to be stopped. It further alleviates any concerns about competition if the service provider also provides competing services.
Protection of Edge Computing is a technology that moves enterprise software to embedded systems or edge servers as this framework uses distributed cloud technology. Confidential computing protects data and software at edge nodes.
The Confidential Computing Consortium [20] (CCC) was formed by a group of CPU manufacturers, cloud providers and software companies in 2019. These companies include Alibaba, AMD, Baidu, Fortanix, Google, IBM/Red Hat, Intel, Microsoft, Oracle, Swisscom, Tencent and VMware. Its intention is the definition of industry standards for Confidential Computing and the promotion of the development of open-source tools. TEE implementations are complex as programs have basically to be modified to benefit from its security features. Open-source projects such as Enclave SDK and Red Hat Enarx are the first projects of the Consortium.
Nonetheless, Confidential Computing technologies have already been used before the organization of the CCC. One of these key technologies is the Intel SGX which enables TEEs not only on the Intel Xeon CPU architecture but also on the Intel Core CPU family. However, its inception had been on workstations- and accordingly, server systems. Intel introduced its TEE implementation in 2016. IBM for example has introduced Confidential Computing in its product line since 2018 [8].

1.3 High-Performance Computing and Trusted Execution Environments Symbiosis

1.3.1 Trusted Execution Environments

Over the years, Digital Rights Management (DRM), mobile financial services, authentication, secure modular programming, organizations and their cloud emerged with the further need for confidentiality. Furthermore, the EU General Data Protection Regulation (GDPR) has put stricter legal policies in place when organizations process and transmit data from their clients [21]. Privacy has been increased with concepts such as homomorphic encryption. Unfortunately, these come with significant performance overhead [18].
Trusted Execution Environments (TEE) have been introduced to address the aspects of privacy, performance and practicability, considering a wider range of use cases at lower costs than pure software approaches. To consolidate findings about the definition, the CCC is quoted as follows:

The Confidential Computing Consortium, *Confidential Computing: Hardware-Based Trusted Execution for Applications and Data, 2021* [20] A Trusted Execution Environment (TEE) is commonly defined as an environment that provides a level of assurance of data integrity, data confidentiality, and code integrity. A hardware-based TEE uses hardwarebacked techniques to provide increased security guarantees for the execution of code and protection of data within that environment.

However, challenges such as security issues will be treated later in the paper. In our daily life, there are many examples given as Apple's Secure Enclave uses the concept of TEE for its Secure Enclave Processor [15] (SEP). It is one of Apple's key security features as it is implemented in their current SoC-based products such as their handhelds and accessory devices. Their iOS and Apps make use of encryption keys which are kept secure by the SEP.
Furthermore, business computers are regularly equipped with a Trusted Platform Module (TPM) which represents a form of TEE. The principle is to ensure the integrity of the hard- and software involved during boot. Occasionally, it is also used for other cases such as in the prevention of cheating on games. However, this technology lacks performance as it can not be used for large workloads. Newer Trusted Execution Environments enable organizations to securely process their data on the internet. Hard- and software-developer hope that this technology is a long-term solution for Confidential Computing on mobile devices, computers and cloud systems while having security threats minimized.
Standard groups such as the Internet Engineering Task Force (IETF) and its Trusted Execution Environment Provisioning working group develop standards to ensure interoperability between systems, software and workloads. For example, with Open-TEE, the use of virtual trusted execution environments is possible. This enables developers to build trusted programs while respecting GlobalPlatform's TEE specifications [22]. Cloud service providers did not hesitate to expand their services through confidential cloud computing. Amazon Web Services (AWS) introduced AWS Nitro Enclaves to minimize targets for their software by the provisioning of a secured computing environment. It is hardened, highly isolated and trusted according to AWS.

Most Intel- and AMD-based Amazon EC2 instance types built on the AWS Nitro System provide those enclaves. In contrast, Microsoft also rolled out its service for Confidential Computing with the offer of DCsv2-series virtual machines. The Intel Software Guard Extensions (SGX) are enabled on their Intel servers and by that, their security is enhanced. Azure confidential computing does not permit access to the data within the virtualized hardware-based TEEs to any unprivileged persons, including the cloud provider.
The concept of a Trusted Execution Environment makes it unique as hardware and software components are combined to establish a secure area within the memory. As illustrated in Figure 1.2, those components are built upon two distinct models for cloud computing [21]. The whole system memory of a virtual machine is encrypted in the virtual machine-based model (A). Only an encrypted memory area within the virtual machine is established in the process-based model. In comparison, in A, the whole VM is encrypted. In contrast, in B, the confidential code has to be destined by its software developer to be run in a Secure Enclave. It further means that in B it is needed to distinguish between encrypted- and unencrypted sections of the system memory [17].

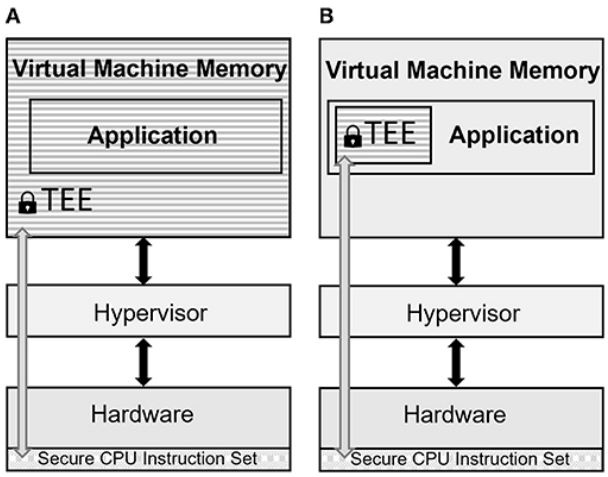

Figure 1.2: **TEE cloud computing [21]**
A: **Virtual machine-based model**, the whole memory of the virtual machine is encrypted.
B: **Process-based model**, only the memory of the enclave is encrypted.

It is crucial to know that the hardware vendor, e.g. Intel or AMD represents the certificate authority (CA) as the CA provides the pair of private and public keys to identify the unique hardware. During production, the private key is installed into the hardware. This constitutes the so-called root of trust [27]. In contrast, the CA signs with its private key the public key of the hardware. This ensures the reverse lookup to the CA while building trust.
TEE is implemented via platform-specific microcode instructions which come with the hardware.

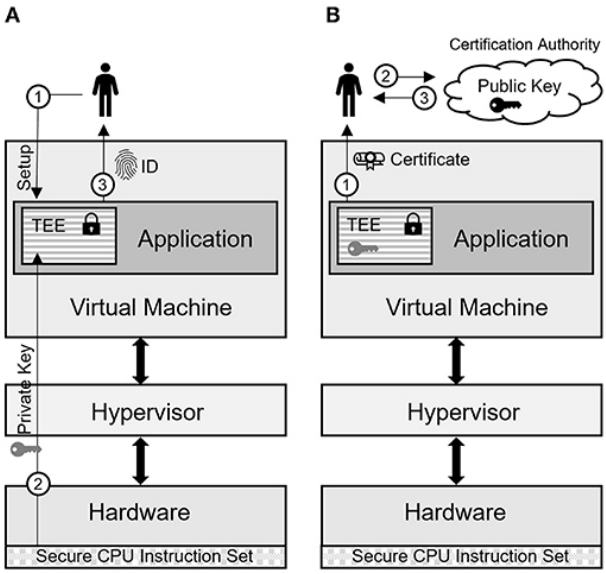

Figure 1.3: **Process-based model - TEE creation and validation [21]**
A: **Creation**, system memory is encrypted using a symmetric encryption schema.
B: **Validation**, a remote CA validation service uses the private validation key with metadata to send a certificate for validation to the CA.

As shown in Figure 1.3 (A), the system memory is user-defined and only for this encryption process, access to the private key is granted. This is the crucial part where confidentiality over the cloud service provider or other stakeholders is built up. After encryption, a portion of the application is loaded into the encrypted memory and finally, a unique identifier of the TEE is determined and sent to the user for validation. As shown in Figure 1.3 (B), validation needs to be done as the user responsible for the TEE on a cloud system does not have access to the physical hardware. There is a necessity of trust that the TEE source is the vendor's infrastructure as only this guarantees the integrity of encryption and the expected program code lies in the secure area. The remote CA validation service is responsible for this procedure. Firstly, with the private key and the metadata, a certificate will be generated which will be retrieved by the user. Then, the certificate is signed by the TEE corresponding private validation key. This key contains the unique ID from A, and supplementary details about the considered program code and its underlying hardware. The anticipation of the user concerning the software can then be confirmed as the certificate is sent to the CA. With successful validation, the promise of privacy is fulfilled.

In contrast, the virtual machine-based model is being used by AMD with their AMD Secure Encrypted Virtualization (SEV) technologies. These technologies were introduced in 2016 [14] for x86 architecture to enable isolation between VMs and corresponding hypervisors. Originally, hypervisors had been trusted components in the virtualization security model. But it has its limits when it comes to Confidential Computing as the cloud service provider manages the hypervisor and has access to the machines. This fact leaves users to desire further isolation of their VMs at a hardware level from the hypervisor and other software.

To counter the issue, AMD implemented Secure Memory Encryption (SME). VMs can be assigned a unique AES encryption key to automatically encrypt the in-use data. Hypervisors have only access to the encrypted bytes.
Furthermore, in 2017, Encrypted State (ES) has been added to the SEV portfolio. Before, the CPU register state had been exposed to the hypervisors. Now, with this feature, it is possible to encrypt these on each hypervisor transition so that the hypervisor can not read data while being processed within the VM. This feature enhances VM protection as the data in memory is additionally protected.
In 2020, Secure Nested Paging (SNP) has been introduced as the next-generation SEV technology. It builds upon other technologies. It adds supplementary hardware-level security features. Known threats on that level are data replay, memory remapping and more to have an isolated execution environment. To counter those malicious hypervisor-based attacks, SEV-SNP ensures strong memory integrity. Further virtualization-based use cases are supported and the protection around interrupt behavior has been fortified. SNP faces current threats by side-channel attacks. They will be discussed later.
When it comes to encryption, AES is in place [14]. It provides trust by protecting the memory. Without the corresponding key, for unprivileged individuals, it is not possible to decrypt the in-use data of the VM as it makes use of SME. A hardware random number generator creates the key stores in dedicated hardware registers. Software is not permitted to read it. Furthermore, by design, identical plaintext at different memory locations is encrypted differently by the hardware.
Besides, attempts to change memory values can not be excluded, even without the encryption secret. Those attacks are called integrity attacks as RAM is manipulated. Without the secret, proper data placement seems difficult. However, the VM could see random values which might throw exceptions. Also, replay attacks could be conducted. In such a case, an attacker records ciphertext at one point in time and later replaces memory with the earlier captured data [14]. The impact would be much higher if the attacker knew the semantics of the data.
In contrast, attacks that impact the integrity of the VM do not directly control a VM as the VM is treated as a black box. Patterns of behavior are being interpreted, as incorrect data shall compromise the VM or disclose information. Success is determined by how well the machine and its behavior had been analyzed and malicious data accurately implanted.
With AMD SEV-SNP in place, the risk associated with integrity attacks is significantly reduced. The main principle of SNP is the virtual machines' only permission to read a private and encrypted page of memory is given when it reads the last value it wrote. Assumed that a value A to RAM location X was written by the VM. Whenever it later reads it, it sees either the value or the process throws an exception which indicates an access failure. By design, the VM can not see a different value at that location.
These technologies impede default virtualization features as an integrity guarantee has to hold in any case. So features and hardware need to be designed around. If memory pages are being transferred between disks or entire virtual machines migrated to new hosts or clusters, this guarantee has to be upheld. This requires state-of-the-art hardware. Within and without virtual machines need to run their jobs, in the latter with their corresponding interfaces. This might involve network communication, storage systems or other components. For external communication, unencrypted memory is used. Outgoing information is moved to a shared page of memory, respectively for incoming data. It is recommended to use at least secure communica-

tion protocols to transfer the information.

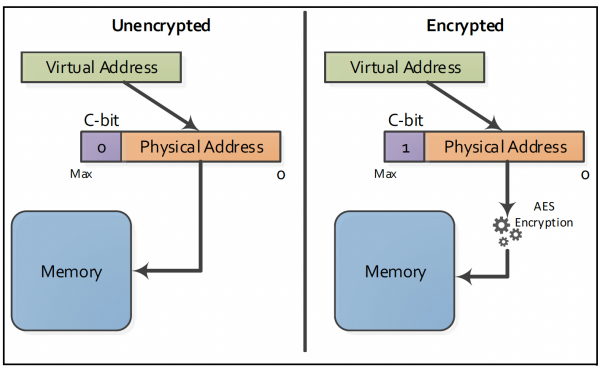

Figure 1.4: **Virtual machine-based model - Encryption Control [14]**
Unencrypted: **Shared memory**, C-bit is 0. Encryption is deactivated.
Encrypted: **Private memory**, C-bit 1. AES Encryption is active.

AMD SEV-enabled virtual machines have control over the state of being private or shared using the enCrypted bit [14] (C-bit) in the guest page tables. The location of this bit depends on the implementation. It may be the top physical address as shown in Figure 1.4.
Shared memory needs to be unencrypted, so the C-bit is 0 and encryption is deactivated. Private memory needs to be encrypted, so the C-bit is 1 and AES encryption is activated. In most cases, the majority of memory pages are marked private and only a careful selection need external communication which then, needs to be marked as shared. SEV-SNP integrity guarantees come only into effect when private memory is used.
As mentioned before, there are many use cases for Trusted Execution Environments such as artificial intelligence, Secure Multiparty Computation (SMPC), Internet of Things (IoT) and cloud computing. Currently, the principal one is cloud computing. It is especially promising regarding its capability to add the same security properties to mobile and cloud systems that organizations strive for their on-premises environments. It fulfills the requirements regarding security and trust and with that, it shall also allow cloud computing in sensitive areas [18].
Speaking of HPC use cases, TEE enhances Confidential HPC in the public cloud. It enables secure data processing, the establishment of Secure Enclaves for processes and collaboration with stakeholders. Although not perfectly secured, Trusted Execution Environments enable a high level of security that is not accessible by the hardware producers and software developers [22].
Big Data processing and intensive workloads with the need for split-second response latencies signify the difference between ordinary cloud computing and high-performance computing in the cloud. Furthermore, also MPC is being done on HPC infrastructures, also with the need for enough security to process sensitive data. Data analysis on sensitive data, as mentioned before, of financial services can be a use case. MPC faces the challenge that it often relies on trusted third parties or legal contracts. There, data exploits are still possible.
Trusted Execution Environments can increase the trust in MPC as it offers sufficient security and efficiency. Those analytics can be run within the TEE. Each stakeholder can validate the

code run within the TEE. The benefit is by putting raw data in and collecting aggregated output data.

1.3.2 Threat Modeling

As the HPC cloud environment differs from ordinary HPC infrastructure, unique challenges will be demonstrated. In any way, HPC runs in a cluster of nodes. Regularly, virtualization software is used to build up a dynamic and scalable network. Regarding the security challenges, Single-User and Collaborative-Multiparty Cases have been discussed in the introductive section of this paper. The before-mentioned TEEs provide hardware-protected memory areas. Intel Software Guard Extensions establish the Secure Enclave, while AMD Secure Encrypted Virtualization encrypts entire virtual machines. However, side-channel attacks may still occur and threaten these technologies as access patterns and partially, CPU caches are readable.
Therefore, manufacturers are demanded to provide patches, upgrades to the software of their technologies, or even newer hardware architectures. However, as the Secure Enclave and the encrypted virtual machines also need to communicate with untrusted areas of the memory, this gives the attack surface. Nonetheless, it is reasonable to assume that physical access to the cloud server is denied to the attacker. So it is not possible to attach any malicious device to a server or access its technical components. This excludes attacks that depend on physical access. In the context of collaboration, there are *curious participants* and *dishonest owners* [18] who might have malicious intentions. The *curious participants* generate results, while they might also be interested in collecting information about sensitive data and programs, while *dishonest owners* of private components might also use their Confidential Computing technology to shroud their devious intentions.

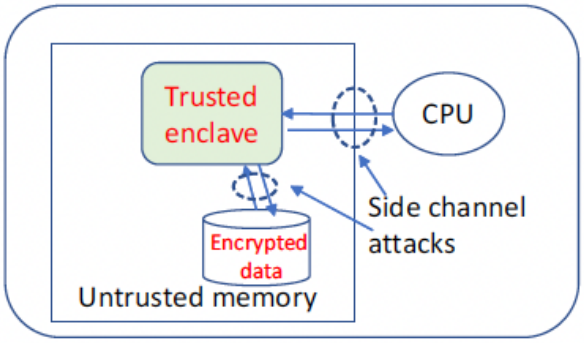

Figure 1.5: **Generic Threat Model - Untrusted Server [18]**
Intel SGX: Trusted enclave corresponds to Secure Enclave and includes encrypted data.
AMD SEV: The entire virtual machine(s) are encrypted.

For HPC, there are clusters of nodes that are in practice generated via one or many hypervisor(s) [33]. That means that virtualization technology is being used. AMD System-On-Chip (SOC) hardware, the AMD Secure Processor (AMD-SP) and the previously introduced and

integrated SEV technologies such as Secure Encrypted Virtualization (SEV), Encrypted State (ES), Secure Nested Paging (SNP) and the entire encrypted virtual machine itself are considered fully trusted. It is the VMs responsibility to ensure its full protection, considering I/O and its data, e.g. network- and storage traffic. SEV technologies protect only data in use. To protect data at rest, AMD recommends Full Disk Encryption (FDE) solutions as they exist on the market [14].
Considering SEV-SNP, all other components are considered untrusted. This includes the BIOS, hypervisor, device driver, other virtual machines, et cetera. They are considered as malicious as they probably interact with other untrustable components. This can harm the security guarantees of a virtual machine with SEV-SNP enabled.

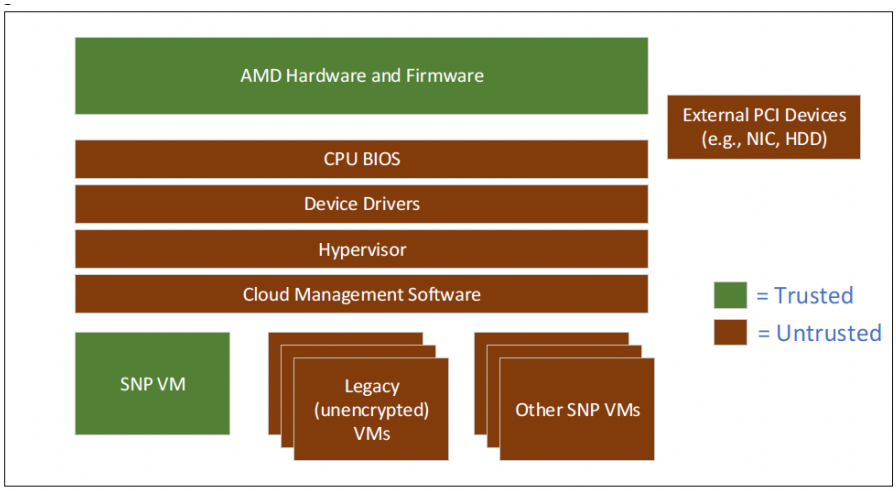

Figure 1.6: **SEV-SNP Threat Model - Server Components [14]**
Trusted: Components in green.
Untrusted: Components in red.

This threat model is an improvement in comparison to models of older technologies of that technology stack [14]. The precedent model of SEV and its addition ES described a "benign but vulnerable" hypervisor. It was not treated as fully protected and so far trusted with benevolent intentions. Considering that while it does not try to take control of the virtual machines, it stays exposed to its vulnerabilities. Nonetheless, there are measures to minimize the threat. SEV and SEV-ES contribute significantly to that by reducing corresponding bugs or keeping up a certain level of protection. The SNP technology prevents supplementary threats to VM security. More details will follow later in the Security Analysis section.

1.3.3 Security Analysis

Regarding AMD technologies, there are specific threats that are not in the scope of SEV, SEV-ES and SEV-SNP. Hardware means do not prevent any kind of side-channel attacks on CPU communication [14]. Sensitive data and code need to be written in a way that confidentiality, in the end, is guaranteed. Also, Fingerprinting can not be protected with the current development. By monitoring of metadata of virtual machines, such as performance numbers and access patterns, corresponding attacks attempt to retrieve information about the contents of the VM. In most cases, data in use by the code is the most sensitive data which can be of interest. Therefore, the current SEV stack concentrates principally on the protection of sensitive data within the virtual machine. Furthermore, SEV-SNP supports the restriction of interrupt- and exception-handling of a VM. Branch Target Buffer (BTB) protection against specific side-channel attacks can also be implemented.
The following aspects and corresponding threats of SEV, SEV-ES and SEV-SNP will be discussed: *Confidentiality*, *Integrity*, *Availability*, *Physical Access Attacks* and *Miscellaneous*.
Confidentiality: All SEV technologies support Secure Memory Encryption. Unprivileged components such as DMA-capable devices and the hypervisor will not get any access to the contents of a VM. Encrypted State added the protection of register states of VMs. These states get encrypted when a VM exits back to the hypervisor. In Secure Nested Pages, this is included.
Integrity: SNP can protect against integrity attacks. These attacks are data replay, corruption, re-mapping and aliasing-based attacks. All these malicious forms are hindered by the guarantee that a VM always sees the data it last wrote.
Availability: Availability needs to be distinguished into two parts. Firstly, the hypervisor supervises its system. Secondly, the guest VM obeys the operation commands of its hypervisor and it can not harm the physical system. The SEV stack supports this degree of availability as a whole. From the first perspective, by interrupt handling or command enforcement, the hypervisor has a guaranteed possibility to control its guest systems. In the second regard, by exceeding the scope of AMD technologies, minimum run-time can be ensured by the hypervisor which is in contrast to malicious host systems which might never want to run at least one its guests.
Physical Access Attacks: Mostly memory cold boot attacks can be prevented by AMD technologies. As online memory-integrity attacks are very complex and exhausting considering needed access and resources, they are currently not addressed by AMD.
Miscellaneous: Trusted Computing Base (TCB) rollback attacks are also prevented by SEV-SNP. The technology enables a confidential verification of compliance to the VM by trusted components such as the AMD-SP firmware or other.

Figure 1.7: **AMD - Threat Model [14]**

The following threats of SEV-SNP will be discussed: *Replay Protection*, *Data Corruption*, *Memory Aliasing* and *Memory Re-Mapping*.
Replay Protection and *Data Corruption* rely on having unprivileged programs being able to access a VM by writing into is memory. SEV-SNP counters this issue by exclusively granting permission to the owner of a memory page to do so, e.g. VM with SEV-SNP enabled itself. This behavior is enforced by a mechanism called Reverse Map Table (RMP).

THREAT	DESIRED SECURITY PROPERTY	SEV-SNP ENFORCEMENT MECHANISM
REPLAY PROTECTION	Only the owner of a memory page can write that page	Reverse Map Table (RMP)
DATA CORRUPTION	Only the owner of a memory page can write that page	Reverse Map Table (RMP)
MEMORY ALIASING	Every physical memory page can map only to a single guest page at one time	Reverse Map Table (RMP)
MEMORY RE-MAPPING	Every guest page can map only to a single physical memory page at one time	Page Validation

Figure 1.8: **SEV-SNP - Integrity Threats [14]**

Memory Aliasing exists when the host system maliciously maps different guest pages to the same physical memory page. This can lead to data corruption as the VM assumes that different pages lead to a different memory. By the RMP mechanism it is enforced that page mapping can only be distinct, which means that there is only one relation between a physical page and a guest page at a time. *Memory Re-Mapping* involves a malicious host system that remaps a single guest page to at least more than one different physical memory page. Inconsistencies might occur in the memory. The 1:1 relation between the physical and guest page counters this issue. Only privileged entities such as the AMD-SP can change any mapping. Page Validation enables this measure and is part of SEV-SNP. It also relies on RMP.
The detailed treatment of further security mechanisms such as Reverse Map Table, Page Validation, Virtual Machine Privilege Levels, Interrupt- and Exception-Handling, et cetera are out of the scope of this paper.

1.3.4 Performance Analysis

Regarding AMD SEV and Intel SGX, performance-related data of HPC applications could be collected. Ordinary HPC workloads as well as modern applications were used. NAS Parallel Benchmark (NPB) is a software suite that provides different pseudo applications and kernels to test the performance of parallel supercomputers. It is an established suite and still up-to-date. Different data sizes had been configured and profiled as classes for the benchmarks. NPB Class C was used for SEV and SGX to solve standard test problems. NPB Class D was only for SEV to solve large test problems. Here, relevant statistics and characteristics will be shown and evaluated. Only core issues will be further treated either still in this or the following sections depending on the use case. Apart from this software suite, modern HPC workloads had been used. Each of them has its characteristics:

GAPBS: Graph workloads, with the input of a graph of road networks in the US.
Kripke: Particle transport simulation.
Livermore Unstructured Lagrange Explicit Shock Hydro (LULESH): Solves "Full-featured" hydrodynamics simulation problem.
LightGBM: Machine learning training, decision tree workload, characterized by irregular memory accesses, using Microsoft's Learning to Rank (MSLR) data set.
Mobiliti: Transportation system simulator (based on parallel discrete event simulation).

Basic Local Alignment Search Tool Nucleotide (BLASTN): Bio-informatics tool to search sequence similarities, more specifically BLASTN was used to search a nucleotide sequence against a nucleotide database.

Concerning the following findings, the relevant architectures and system configurations are illustrated in Figure 1.9 and 1.10. The evaluations had been done without hyperthreading. The number of cores corresponds to the number of threads on each system. Significant performance issues of Confidential Computing is minimized because cache contention is reduced. This affects threads with large jobs.

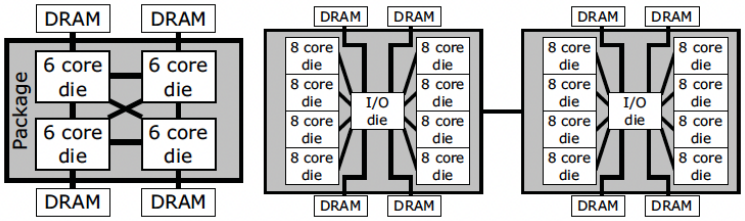

Figure 1.9: **AMD - NUMA-architecture [11]**
Left: **AMD EPYC 7401P (Naples)**
Right: **AMD EPYC 7702 (Rome)**

Three AMD-based servers were used to test SEV. The CPU scheme of AMD EPYC 7401P (Naples) and AMD EPYC 7702 (Rome) are shown in Figure 1.9. The Naples system consists of 24 CPU cores with 6 cores on each of 4 dies in a single CPU. It is single-socketed but has 4 NUMA nodes in total. A multi-chip CPU behaves similarly to a multi-socket system considering latency and bandwidth between separate dies. Depending on the location management of NUMA nodes, memory performance varies highly. The system with the Rome architecture had a more uniform memory design, so it was also evaluated. It had 64 cores with 8 cores on each of 8 dies. It constitutes a multi-chip package. The dual-socket system consists of a total of 128 cores. It has more chips per package. The memory design is more uniform because each die has the same distance to the corresponding I/O die with the memory controllers. Per socket, there is only one NUMA node. In this case, considering the 2 sockets, there are 2 NUMA nodes in total.
Considering the Intel platform, a desktop-class CPU with 6 cores and 1 NUMA node was used to perform SGX simulations. In many cases, the size of secure memory was still significantly smaller than the job units of most studied HPC workloads, e.g. only the benchmark ep has a job unit smaller than 256 MB [11]. It generates independent Gaussian random variates using the Marsaglia polar method [16]. QEMU was used as a hypervisor for virtualization and performance could be improved by interleaving which is discussed later.

Feature	AMD SEV 1	AMD SEV 2	AMD SEV 3	Intel SGX
CPU	EPYC 7401P	EPYC 7702	EPYC 7402P	Core i7-8700
Sockets	1	2	1	1
Cores	24	128	24	6
NUMA	4 Nodes	2 Nodes	1 Node	1 Node
RAM	64GB	1TB	64GB	32GB

Figure 1.10: **AMD EPYC - System configuration [11]**

Concerning the performance penalties of HPC workloads in Trusted Execution Environments, the following core reasons could be extracted by Akram et. al [11]:

1. Correct NUMA allocation policy implies small overhead with SEV enabled.
2. Virtualization dependencies (e.g. QEMU) imply significant performance cuts regarding irregular workloads, intensive I/O and CPU usage where a significant number of CPU threads are involved.
3. Initialization of SEV implies poor performance depending on the memory characteristics of the application.
4. Limited secure memory pages and partially, scalability and programming challenges in the usage of SGX imply high-performance overhead.

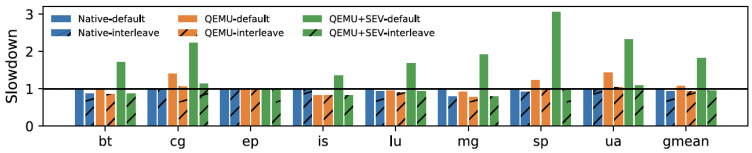

Figure 1.11: **AMD Naples (24 Threads) - SEV for NPB C [11]**
Evaluation: SEV performance overhead caused by default NUMA memory allocation. Solved by interleaving.

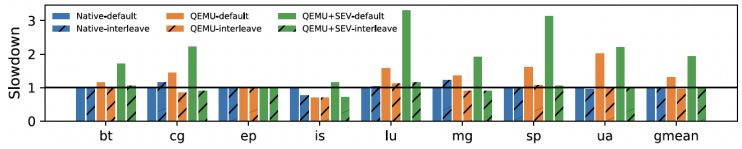

Figure 1.12: **AMD Naples (24 Threads) - SEV for NPB D [11]**
Evaluation: SEV performance overhead caused by default NUMA memory allocation. Solved by interleaving.

The following observations could be done regarding the results of Figures 1.11 and 1.12:

1. SEV activation causes performance penalties beyond virtualization.
2. SEV performance depends on NUMA design.

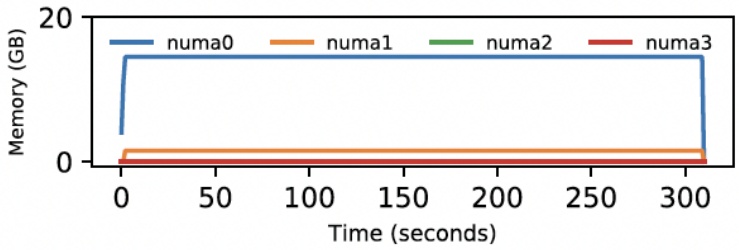

Figure 1.13: **AMD SEV1 (Figure 1.10) - SEV Default Allocation [11]**
Evaluation: VM (16 GB RAM) launch. Performance throttling by data allocation to one only NUMA node.

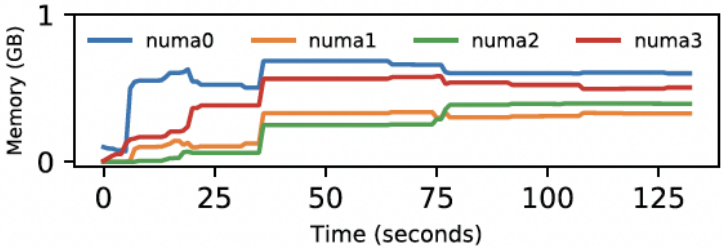

Figure 1.14: **AMD SEV1 (Figure 1.10) - No SEV Default Allocation [11]**
Evaluation: VM (16 GB RAM) launch. Data allocation to all four NUMA nodes following the on-demand paging scheme.

After the observations of the NUMA placement correlation, the assumptions were further compacted after the usage of the AMD SEV3-configuration. There, the NUMA design issues did not occur as the platform has a uniform memory architecture.
The conclusion was that the penalties came with the NUMA allocation policy which is configurable.

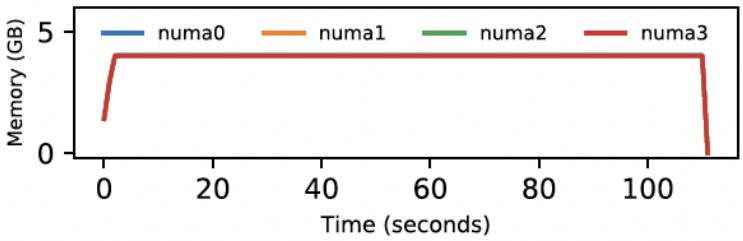

Figure 1.15: **AMD SEV1 (Figure 1.10) - SEV Interleaved Allocation [11]**
Evaluation: VM (16 GB RAM) launch. Data allocation to all four NUMA nodes with significantly increased performance.

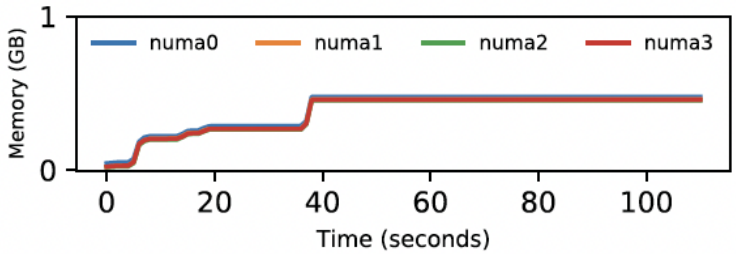

Figure 1.16: **AMD SEV1 (Figure 1.10) - No SEV Interleaved Allocation [11]**
Evaluation: VM (16 GB RAM) launch. Data allocation to all four NUMA nodes with poor performance.

Regarding the results of Figures 1.15 and 1.16, it showed: Performance penalty mitigation was achieved by explicit interleaving of data across NUMA nodes using numactl. Numactl is a NUMA policy control of Linux and allows running processes with a specific NUMA scheduling or memory placement policy. It was used to assign memory pages across NUMA nodes. In that context, AMD SEV2 (Figure 1.10) was used to find out whether the improved uniformity regarding its NUMA design improves performance. It was found out that NUMA design has still a significant role as memory management is crucial for the overall performance while running benchmarks.

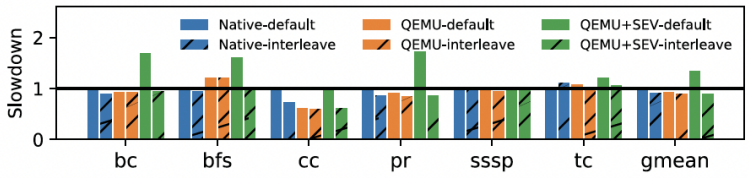

Figure 1.17: **AMD SEV1 (Figure 1.10) - SEV for GAPBS (road network) [11]**
Evaluation: Interleaving works for graph and other HPC workloads except for BLASTN.

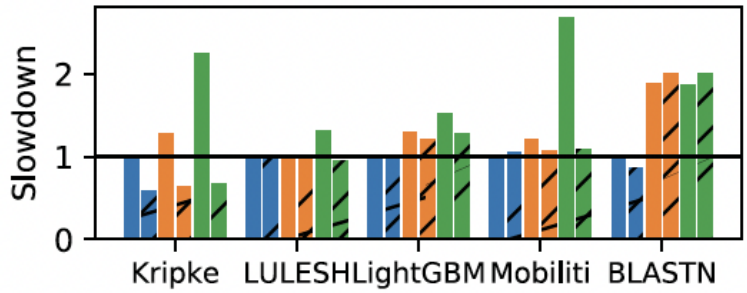

Figure 1.18: **AMD SEV1 (Figure 1.10) - SEV for Real world HPC workloads [11]**
Evaluation: Interleaving works for graph and other HPC workloads except for BLASTN.

BLASTN shows a significant slowdown mainly due to virtualized disk I/O operations. It uses a nucleotide database of approximately 245 GB in size which is larger than the memory size of 64 GB of the used system. This caused the slowdown under virtualization. In contrast, Figures 1.20 and 1.21 show that there is unsignificant overhead due to virtualization with SEV enabled.

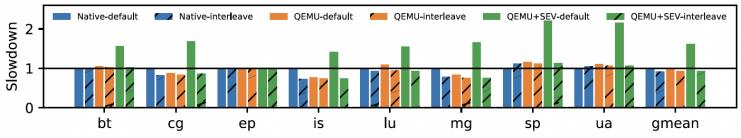

Figure 1.19: **AMD SEV2 (Figure 1.10) - SEV for NPB D [11]**

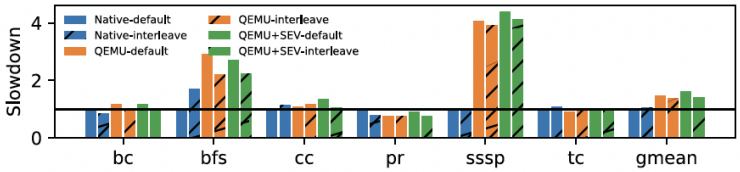

Figure 1.20: **AMD SEV2 (Figure 1.10) - SEV for GAPBS (road network) [11]**
Evaluation: NUMA placement still matters on more uniform memory designs.

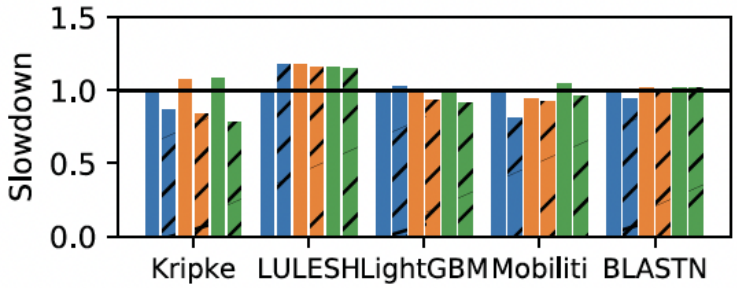

Figure 1.21: **AMD SEV2 (Figure 1.10) - SEV for Real world HPC workloads [11]**
Evaluation: NUMA placement still matters on more uniform memory designs.

In summary, with SEV activation comes performance penalty within and without the virtualization nature. Modification of the interleave policy is the key to optimization depending on NUMA design.

In the following, other, but for the scope of this paper less relevant findings will be summarized as follows [11]:

1. Remaining performance penalties with SEV activated due to virtualization overheads.
2. VM bootup time: Poor performance due to SEV and memory footprint of VM.
3. SGX: Poor performance and compatibility is only given by modified scientific software.

1.3.5 Constraint Analysis

Trusted Execution Environments come with limitations. They are to be distingushed between their characteristics in the aspects of *performance*, *nature*, *security* and *usability*.

Especially in High-Performance Computing, TEEs come with performance penalties. As discussed before, it could be derived from benchmarks with ordinary and modern HPC workloads

that AMD SEV outperforms Intel SGX in significant way. Simulation, graph algorithms and other emerging scientific computing workloads were used.
Virtualization costs performance by 1x - 4x slowdown [11]. The technology has its challenges with irregular memory accesses and the processing of large amounts of data.
Secure Encrypted Virtualization comes with 1x - 3.4x slowdown without NUMA-adjusted configuration, 1x - 1.15x slowdown with the proper adjustments. SEV requires optimized memory management on large-scale HPC clusters. In contrast, the Software Guard Extensions have a much wider range as they come with a 1.2x - 126x slowdown. These performance numbers indicate the differences between secure and insecure executions.

The following Figures 1.17, 1.18 and 1.19 further explain the nature of Trusted Execution Environments. There are three states regarding data: in transit, at rest and in use. Trusted Execution Environments are allocated to the data in use components and additionally, to be distinguished from the other ones. Within the data in use concept, TEEs coexist with Homomorphic Encryption (HE) and Trusted Platform Modules (TPM).

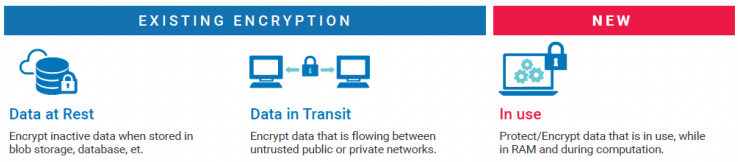

Figure 1.22: **States of data [20]**

In Figure 1.18, isolation is differenced between CPU Addressability Isolation and Memory Isolation. Further, their components as Access Control Validation, Address Translation, Paging Control and RAM Encryption are being used by some example platforms. These are acknowledged by the CC and show how they ensure isolation and confidentiality.

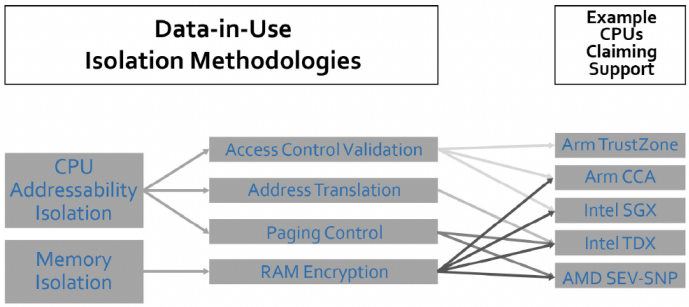

Figure 1.23: **Data in use [20]**

In Figure 1.19, the characteristics of the data in use technologies are shown. Essential categories are confidentiality, integrity and usability. Each of those has its specifics and is divided into further details listed in the matrix. Programmability means the requirement of modification or the possibility of customization.

	HW TEE	Homomorphic Encryption	Secure Element e.g., TPM
Data Integrity	Y	Y (subject to code integrity)	Keys only
Data confidentiality	Y	Y	Keys only
Code integrity	Y	No	Y
Code confidentiality	Y (may require work)	No	Y
Programmability	Y	Partial ("circuits")	No
Unspoofability/Recoverability	Y	No	Y
Attestability	Y	No	Y

Figure 1.24: **Comparison data in use technologies [20]**

In this paper, only a limited set of platforms and their TEE implementations are considered, e.g. AMD SEV and Intel SGX. Their common and specific security-related challenges will be discussed. Currently, most of the research regarding TEE and its security focus on side-channel attacks as these constitute its largest threat. E.g. access patterns are one of the major side effects of data-intensive processing. These can be detected, analyzed and abused. The critical moment during data processing is the data transfer between the unprotected and the confidential memory area. In the context of Intel SGX, the area outside the Secure Enclave is the untrusted part, while the encrypted Secure Enclave itself is the trusted part. Whereas in the context of AMD SEV, the area outside the VM is the untrusted part, while the encrypted VM itself is the trusted part.
However, regardless of the specific TEE being used, the interplay between protected and unprotected area is of the essence. Also, this interplay is of interest to adversaries and their exploits. To counteract, measures of concealment have been introduced, for example, Oblivious RAM (ORAM) to hide block-level access patterns, such as ZeroTrace, Obliviate and Oblix [18]. Furthermore, another major side-channel is the CPU cache which concerns both platforms. Meltdown and Spectre are still threats to pierce through the TEEs protection. Nonetheless, with the manufacturer's ongoing architecture developments and firm- and software fixes, these attacks can be mitigated.
Additionally, regarding Secure Multiparty Computation, there are challenges such as *Owner's Attacks*, *Conflict between Confidentiality and Provenance Analysis* and *Reproducibility Verification*.
Owner's Attacks constitute the case where the owner of a private component (see Figure 1.1) has access to the internals whereas the logging service is limited to external details of the algorithm, e.g. parameter, input and output. Further, considering the owner's actions are concealed, he can manipulate data and code while ensuring that metadata stays similar, disabling the effectiveness of integrity protection measures.
Conflict between Confidentiality and Provenance Analysis exists when the latter needs to access log files that hold information, including metadata about the workloads. Private in- and output data from a known processing algorithm can be captured (model-inversion attack). Also, with sample in- and output data, a private code can be rebuilt (model-stealing attack) [18].
Reproducibility Verification is supposed to be done by a third-party authority. Considering the

private components, data and code are controlled and run by their owners. They also pass secret keys to the enclave which is one of the security requirements. This prevents any unauthorized verification, although it is unacceptable that all owners stay online for verification.
Usability depends on the desired security level [18]. By its design, Intel SGX splits directly between the trusted and untrusted area of the memory which increases the achieved protection and resilience against side-channel attacks, while application modification is mandatory [12]. There are workarounds to avoid the change of code but they can not guarantee the stability and support of HPC applications. However, AMD SEV does natively not require redesigning programs.

1.4 Common Frameworks

1.4.1 SCONE and Graphene-SGX

Although SGX implementations naturally consider adjusted programs as mandatory, there is the possibility to run unmodified ones within a secure container system. One solution is called SCONE. It statically compiles and links applications against a modified standard C library. Also, it performs asynchronous system calls and uses a kernel module while running. This minimizes the overhead due to the enclave transitions. Because of its usability, SCONE is among many others, one of the most popular SGX interfaces [18]. Considering performance data, SCONE seems to be a valuable solution as figures show that it has 0.6 - 1.22x throughput compared to native execution of services, such as Apache, Redis, NGINX and Memcached. Benchmarks with Memcached delivered figures with better performances than those derived from native execution [28]. However, Graphene-SGX had been used as an HPC application to evaluate MPC characteristics considering SCONE [19]. Figures show that in most cases, it is not efficient enough, also considering better performances than these legacy approaches.
Furthermore, SCONE does not protect from access pattern attacks and it is difficult to implement security measures on the application level.
Nonetheless, one the one hand, SCONE increases the confidentiality and integrity of containerized services using Intel SGX. It improves significantly the usability of Intel SGX. On the other hand, it cuts performance and decreases the potential degree of security.

1.4.2 QEMU and Kata

With the means of AMD SEV, QEMU had been used to perform most of the previously mentioned benchmarks. It was also used in combination with Kata [9]. Kata Containers are a container system to deliver secure services within virtual machines. The goal is to achieve stronger workload isolation by using hardware virtualization technology while maintaining lightweight virtual machines. However, the technology is still considered poorly conceived as figures with SEV activated show inconsistencies regarding its performance. Future developments of Kata are

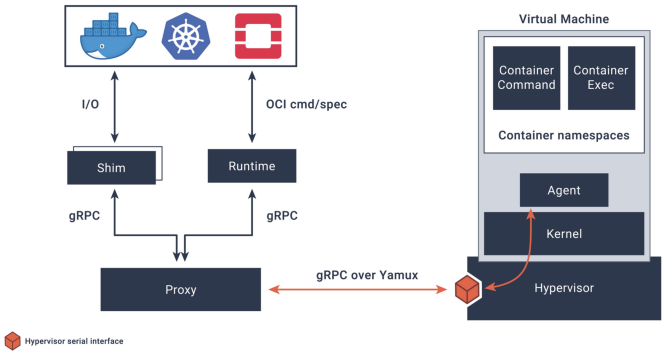

Figure 1.25: **Kata Containers [9]**

assumed to perform as well as when running VMs directly on QEMU since they both use hardware support for virtualization [11]. Kata containers address HPC challenges such as mobility, compatibility and scientific collaboration [3]. As this and similar technologies, e.g. Singularity are still preliminary, they need to be further developed and explored.

1.4.3 RESTful web application in Python using Flask

For the exploration of Secure Multiparty Computation based on Trusted Execution Environments in HPC, a secure virtual machine was used to implement a RESTful web application in Python language while using Flask framework [34]. The Flask framework allows building web applications. The REST API services within the Python language support HTTP(s) and enable interaction with the database by submitting HTTP requests.
The goal of the VM is to securely load the code and input data, aid in the computation of that code and securely deliver the output data to its corresponding parties.

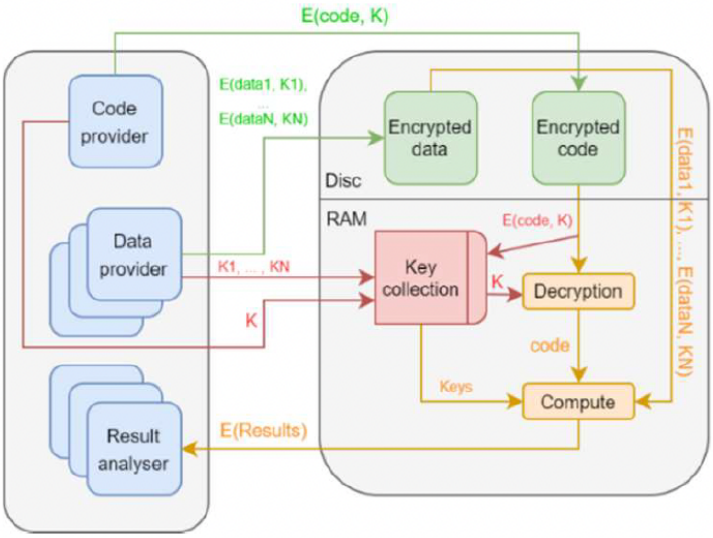

Figure 1.26: **SMPC - Computation process [34]**

Figure 1.21 presents the essential stakeholders of the MPC process. The illustration contains the following components and steps:

1. Load input resources from code and data provider and store them in a predetermined location.

2. Code provider (CP) submits one only encrypted code file using his secret key (K). Data providers (DP) submit an arbitrary amount of encrypted data files using their secret key (K_1,..., K_n). The secret keys which were used are collected by the application. While code and data are encrypted, the supplied keys are in raw form. During runtime, they are stored in the memory. Then they are mapped to their corresponding files. Pythons dictionary data structure was used for this purpose.

3. Supplied code needs to be decrypted and run within the application memory. To accomplish that, the symmetric encryption algorithm of the origin is used. After computation, for the encryption of the results, the program generates a one-time key (KA), encrypts the results and delivers both to the parties of interest.

4. The parties collect, decrypt and process the results locally.

A Marshal module was used to convert an array of bytes to an executable code object. It contains functions to read and write Python values in a binary format while providing cross-platform compatibility. The program expects the code in a marshal binary structure. Therefore, the code provider needs to prepare the code for its structure of byte strings before the encryption. To do so, he uses the *dumps* function. Then, during runtime, the *loads* function reconstructs the code from the decrypted array of bytes that can be directly run using *exec* function. The data structure used for Marshal is specific to the Python implementation. That means compatibility between the parties depends on the Python release. It should be ensured that every stakeholder

uses the same version of Python under which the string of bytes was originally generated with the *dumps* function.

This example takes advantage of key and secret management to ensure that data in use is being processed confidentially. It constitutes one of the essential issues regarding SMPC. The related paper mentions and confirms that Intel SGX performances are poor in comparison to its AMD competitor. The bottleneck was traced back to the Enclave Page Cache (EPC) which is limited to its size of 128 MB (on desktop CPU) [34]. In contrast, entire encrypted virtual machines have secure areas with a size analogous to their memory configuration.

1.5 Conclusion

1.5.1 Summary

High-Performance Computing in combination with Trusted Execution Environments extends Confidential Computing capabilities by adding HPC workloads with their specifics and limitations to its portfolio. Secure Multiparty Computation Based on Trusted Execution Environments initialized the discussion about HPC and Confidential Computing.
Considering the most popular use cases, it was found that the major use case of HPCaaS in the public cloud is only properly enabled by the use of Trusted Execution Environments. Significant concepts such as AMD SEV and Intel SGX were introduced. Their characteristics were investigated and evaluated by the aspects of Security, Performance and Constraints.
Also, Common Frameworks were discussed and delivered insights about the major issues and trends in the application of the concept of Confidential HPC. The usability and performance differences between Intel SGX and AMD SEV applications were distinct.
Furthermore, containerization addresses HPC challenges and adds more flexibility. Its implementation is feasible but needs to be further explored to be beneficial and to make clear statements about quantity and quality.
Finally, it could be shown that SMPC is possible by discussing its theory and giving an example with a RESTful web application in Python language using the Flask framework to see its implementation and coverage.

1.5.2 Evaluation

Trusted Execution Environments contribute significantly to the effective and efficient protection of data in use whereas conventional methods such as Homomorphic Encryption and Trusted Platform Modules lose relevance. However, their combination is not fully excluded to enhance security.
Currently, the domain of Cloud Computing benefits most from TEE in the context of HPC as

HPCaaS was introduced. High throughput with minimized latencies needs to be ensured in the HPC context. And therefore, with the current situation, AMD SEV is the technology that delivers in comparison to Intel SGX pragmatic security with efficiency and effectiveness in one package. By design, it is possible to concentrate on the actual software build and not on security measures regarding the modification needs of Intel SGX.
Nonetheless, it still has weaknesses and it is not fully immune to side-channel attacks.

1.5.3 Outlook

Currently, Intel stays with its technology behind AMD. Nonetheless, it introduced its still relatively new technology of Intel Trust Domain Extensions [34] (Intel TDX) to have an approach similar to AMDs. For now, there is no processor which implements this concept.
With a lot of security research, AMD focuses on closing side-channel gaps. The latest releases enhance certification mechanisms and foster interoperability between hypervisors and SEV-enabled guests. Further exploration is needed to make further statements, especially regarding its difficulties with NUMA design dependencies and side-channel gaps.
As of March 2023, Frontier is the world's fastest supercomputer with 9,472 AMD Epyc 7453s in its system [10]. It can be expected that it also accelerates the debate about AMD architectures in Confidential HPC use cases.

References

[1] AMD Solutions for Supercomputing and HPC, 2022-11-17. URL https://www.amd.com/en/solutions/supercomputing-and-hpc.

[2] What is HPC? Introduction to High-Performance Computing, 2023. URL https://www.ibm.com/topics/hpc.

[3] Linux Containers for HPC, 2023-01-16. URL https://www.tuebix.org/2019/programm/holger-gantikow-linux-containers-for-hpc-container-technology-engine-architecture-101/.

[4] What is High-Performance Computing, 2023-02-13. URL https://www.netapp.com/data-storage/high-performance-computing/what-is-hpc/.

[5] Azure Confidential Computing – Protect Data In Use, 2023-02-13. URL https://azure.microsoft.com/en-us/solutions/confidential-compute/#overview.

[6] Confidentiality, 2023-02-13. URL https://dictionary.cambridge.org/dictionary/english/confidentiality.

[7] Was ist High-Performance Computing?, 2023-02-21. URL https://www.oracle.com/de/cloud/hpc/what-is-hpc/.

[8] What is Confidential Computing?, 2023-02-23. URL https://www.ibm.com/topics/confidential-computing.

[9] Kata Containers - Open Source Container Runtime Software, 2023-02-23. URL https://katacontainers.io/.

[10] TOP500 Release, November 2022, 2023-03-07. URL https://www.top500.org/lists/top500/2022/11/.

[11] Ayaz Akram, Anna Giannakou, Venkatesh Akella, Jason Lowe-Power, and Sean Peisert. Performance Analysis of Scientific Computing Workloads on General Purpose TEEs. pages 1066–1076. doi: 10.1109/IPDPS49936.2021.00115.

[12] Ayaz Akram, Venkatesh Akella, Sean Peisert, and Jason Lowe-Power. SoK: Limitations of Confidential Computing via TEEs for High-Performance Compute Systems. In *International Symposium on Secure and Private Execution Environment Design (SEED)*, pages 121–132, 2022. doi: 10.1109/SEED55351.2022.00018.

[13] Amazon Web Services, Inc. High Performance Computing (HPC), 2023-02-22. URL https://aws.amazon.com/hpc/.

[14] AMD. AMD Secure Encrypted Virtualization (SEV) - AMD, 2023-02-01. URL https://developer.amd.com/sev/.

[15] Apple Support. Secure Enclave, 2023-03-03. URL https://support.apple.com/en-gb/guide/security/sec59b0b31ff/web.

[16] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagum, R. A. Fatoohi, P. O. Frederickson, T. A. Lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, and S. K. Weeratunga. The Nas Parallel Benchmarks. 5(3):63–73, 1991. ISSN 0890-2720. doi: 10.1177/109434209100500306.

[17] Nicolas Buchner. Survey on Trusted Execution Environments. 2022. doi: 10.2313/NET-2022-07-1_05.

[18] Keke Chen. Confidential High-Performance Computing in the Public Cloud. pages 1–10, 2023. ISSN 1089-7801. doi: 10.1109/MIC.2022.3226757.

[19] Chia-Che Tsai, Donald E. Porter, and Mona Vij. Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX. pages 645–658. URL https://www.usenix.org/conference/atc17/technical-sessions/presentation/tsai.

[20] Confidential Computing Consortium. Confidential Computing Whitepapers, 2023-01-26. URL https://confidentialcomputing.io/white-papers-reports/.

[21] Tim Geppert, Stefan Deml, David Sturzenegger, and Nico Ebert. Trusted Execution Environments: Applications and Organizational Challenges. 4:78, 2022. doi: 10.3389/fcomp.2022.930741. URL https://www.frontiersin.org/articles/10.3389/fcomp.2022.930741/full.

[22] HPE. Boosting security with trusted execution environments, 2021. URL https://www.hpe.com/us/en/insights/articles/boosting-security-with-trusted-execution-environments-2102.html.

[23] Intel. What Is High Performance Computing (HPC)?, 2023-02-20. URL https://www.intel.com/content/www/us/en/high-performance-computing/what-is-hpc.html.

[24] Intel. Confidential Computing, 2023-02-23. URL https://www.intel.com/content/www/us/en/security/confidential-computing.html.

[25] Jelsag. Was ist Confidential Computing? URL https://www.cloudcomputing-insider.de/was-ist-confidential-computing-a-32a437cc8b9cfd10b7828f583df59a9a/.

[26] Fahmida Y. Rashid. What Is Confidential Computing? URL https://spectrum.ieee.org/what-is-confidential-computing.

[27] Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. Trusted Execution Environment: What It is, and What It is Not. In *Trustcom/BigDataSE/ISPA*, number 1, pages 57–64, 2015. doi: 10.1109/Trustcom.2015.357.

[28] Sergei Arnautov, Bohdan Trach, Franz Gregor, Thomas Knauth, Andre Martin, Christian Priebe, Joshua Lind, Divya Muthukumaran, Dan O'Keeffe, Mark L. Stillwell, David Goltzsche, Dave Eyers, Rüdiger Kapitza, Peter Pietzuch, and Christof Fetzer. SCONE: Secure Linux Containers with Intel SGX. pages 689–703. URL https://www.usenix.org/conference/osdi16/technical-sessions/presentation/arnautov.

[29] Servers and Storage. The hidden danger of outdated infrastructure: security risk, 2021. URL https://www.ibm.com/blogs/systems/the-hidden-danger-of-outdated-infrastructure-security-risk/.

[30] Servers and Storage. Forrester study: hybrid cloud strategy and the importance of on-premises infrastructure, 2021. URL https://www.ibm.com/blogs/systems/forrester-study-hybrid-cloud-strategy-and-the-importance-of-on-premises-infrastructure/.

[31] SMBrook. High-performance computing (HPC) on Azure, 2023-02-23. URL https://learn.microsoft.com/en-us/azure/architecture/topics/high-performance-computing.

[32] VMware. What is a Public Cloud?, 2023. URL https://www.vmware.com/topics/glossary/content/public-cloud.html.

[33] VMware. Virtualizing High Performance Computing, 2023. URL https://www.vmware.com/uk/solutions/high-performance-computing.html.

[34] Maja Vukasovic, Danko Miladinovic, Adrian Milakovic, Pavle Vuletic, and Zarko Stanisavljevic. Programming Applications Suitable for Secure Multiparty Computation Based on Trusted Execution Environments. In *Telecommunications Forum (TELFOR)*, pages 1–4, 2022. doi: 10.1109/TELFOR56187.2022.9983726.