Whitepaper

CONFIDENTIAL COMPUTING

How to process data securely on third-party infrastructure

EDGELESS SYSTEMS

TABLE OF CONTENTS

1. WHAT IS CONFIDENTIAL COMPUTING AND WHY SHOULD YOU CARE?

Confidential computing is a technology that shields computer workloads from their environments and keeps data encrypted even during processing. The vast and previously unsolved problem that confidential computing addresses is the following: How to process data on a computer that is potentially compromised? This computer could be operated by yourself, your company, or a third party like a cloud provider.

The cloud setting is what gets most people excited about confidential computing. This is unsurprising because when running workloads in the cloud, you naturally have to trust the cloud provider with all data. In addition, you have to trust that the cloud provider actually runs the correct operations on your data. Trusting a cloud provider means trusting its employees and trusting that its systems have not been compromised by external parties or that they are subject to foreign legislation. In many cases, this trust requirement is acceptable. Still, for many industries and sensitive types of data it is not. This may be due to risk awareness or hard regulatory requirements. As a result, companies still hold on to their outdated and oftentimes expensive on-prem datacenters. And consumers refrain from using cloud-based services not deemed private enough.

Confidential computing solves this “trust problem” of the cloud. It also enables new forms of innovative applications on public cloud infrastructure. Confidential computing is thus poised to unlock large amounts of value in our global economies. It will also likely act as a catalyst for other disruptive technologies, like AI. This paper seeks to explain the basic concepts of confidential computing and help practitioners and decision makers understand the impact the technology has on their job and their business.

2. CORE CONCEPTS

So, what exactly is confidential computing? The basic idea of confidential computing is to have the processor of a system create a highly secure environment for data processing. Such execution environments are often referred to as trusted execution environments (TEEs) in the literature. In the following, we use the term “confidential computing environment” (CCE) to refer to TEEs with specific capabilities.

CCEs are shielded from the “rest of the system”. The rest of the system includes other apps, the host operating system, the hypervisor, and hardware other than the processor. This is depicted in *Figure 1*.

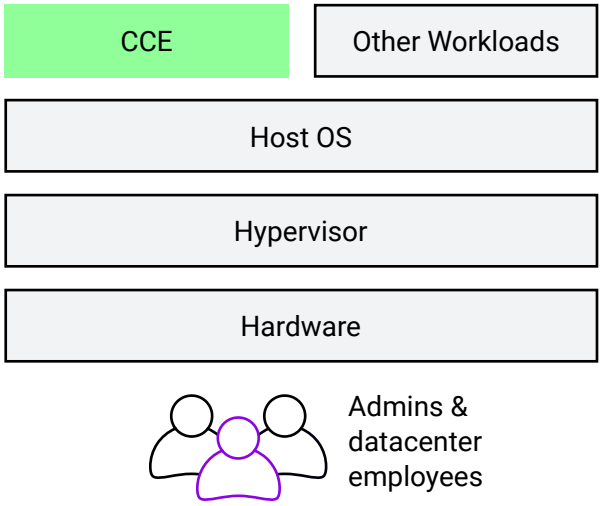

Figure 1: A confidential computing environment (CCE) is shielded from the rest of the system and privileged individuals like administrators or datacenter employees

It may help to think of CCEs as a diver's cage in a sea full of sharks or a fortress within enemy territory. In contrast to special-purpose security hardware like HSMs or smartcards, CCEs can typically run (almost) arbitrary software. CCEs have the following three defining properties:

- **Runtime encryption:** the processor keeps all of a CCE's data encrypted in main memory. Any system component or hardware attacker attempting to read a CCE's data directly from memory will only ever see encrypted data. Likewise, encryption prevents the modification of a CCE's data through direct access to memory.

- **Isolation:** the processor blocks software-based accesses to the CCE. Software like the operating system or other apps can only communicate with the CCE over specific interfaces. You can think of this as an API that the CCE exposes to the outside world.

- **Remote attestation:** on demand, the processor can issue a cryptographic certificate that proves the integrity and authenticity of a CCE and data it produced.

Runtime encryption is what most people associate with confidential computing. It's also what typically creates the most excitement. It's fair to say that confidential computing is the first practical technology that can keep data encrypted while running arbitrary programs on it. With confidential computing, data can now be kept encrypted in all three of its fundamental "aggregate states": at rest (i.e., on disk), in motion (i.e., on the wire), and during processing (i.e., in memory). Before confidential computing, only the former two were possible.[1]

Still, runtime encryption and isolation alone wouldn't be very useful in most use cases. (We'll discuss concrete use cases further down in section 4.) To understand why, consider a CCE running on a remote computer/cloud. How would you know that the CCE was indeed a real CCE and that it was running the intended software? The computer could be compromised and just pretend that it was running a CCE and steal your data as soon as you've sent it over. This fundamental problem is addressed by the remote attestation feature. Remote attestation allows software running inside a CCE to re-

1 Other technologies exist that allow for the processing of encrypted data. Fully homomorphic encryption (FHE) is an approach that does not require hardware support and is purely based on math. To date, FHE does not scale to complex workloads. It is for example virtually impossible to run a real-world database with FHE.

quest cryptographic certificates for itself from the processor at runtime. With this certificate, the CCE can prove to anyone that it is a "good" CCE with a certain configuration and that it is running a certain piece of software. These cryptographic certificates look and work much like website certificates (i.e., X.509 certificates used for TLS).[2] Each confidential computing-enabled processor has a unique cryptographic key associated with it, called a private key. Only the processor can access this key. When a CCE requests a certificate for itself, the processor uses its private key to issue the certificate. The certificate includes a cryptographic hash of the CCE's code and configuration. This hash is also called measurement. A CCE certificate may include additional information. Typically, analogously to web certificates, this information includes the CCE's public key. Based on this, secure TLS connections with a CCE can be bootstrapped.

By inspecting a CCE's certificate, we can learn that we're talking to a specific piece of software running encrypted and in isolation on a trustworthy processor. But how can we trust the processor's key? Processor keys also come with their own certificates. These are issued by the processor manufacturers. The processor manufacturers act as certificate authorities (CAs) similar to how Let's Encrypt or VeriSign act as CAs for the web. The root CAs, i.e., the processor manufacturers, act as trust anchors. Without them, no remote attestation is possible.

Does this mean that we must talk to, say, Intel every time we want to verify a CCE? No, it is sufficient to obtain a manufacturer CA's certificate once and then use it for an unlimited number of verifications.

SUMMARY

Confidential computing is all about running workloads in hardware-enforced secure execution environments. We refer to them as confidential computing environments (CCEs).

One can load arbitrary programs into CCEs. The hardware ensures that all data (and code) of the CCE remain encrypted at runtime. This feature is what most people associate with confidential computing.

Apart from that, confidential computing has one additional defining feature: remote attestation. Remote attestation allows anyone to establish trust in a CCE and bootstrap a secure channel to it. So, confidential computing is about keeping data encrypted during processing and making this verifiable remotely. The verification/remote attestation feature is important, because without such a mechanism, a malicious actor could just claim to be running a CCE and then access data once it is sent over.

2 Website certificates are ubiquitous on the WWW. Website certificates are required to create secure (=authenticated and encrypted) connections between your browser and a website. For websites with valid certificates, your browser displays the well-known lock symbol in its address bar.

3. FLAVORS OF CONFIDENTIAL COMPUTING ENVIRONMENTS

Different implementations of CCEs exist. The two primary categories are secure enclaves and confidential virtual machines (CVMs).

3.1 SECURE ENCLAVES

The most widely known secure enclaves platform is Intel's SGX, which is short for Software Guard Extensions. Intel started adding SGX to its processors in 2015. By now, SGX is available in many Xeon server processors. On an SGX-enabled system, enclaves can be created at runtime within any process. (Such a system could, for example, be an Intel processor running Linux.) The original intent of SGX was to enable programmers to selectively isolate and protect sensitive parts of an application. This could, for example, be the cryptographic library in a web server or the password manager within a browser. However, today, the most common approach is to run entire applications inside enclaves.

Distinctive for SGX is its programming model. It requires a programmer to split an application into "enclave" and "host". The enclave part has all the CCE features. The enclave contains all sensitive code, whereas the host contains non-sensitive code such as basic networking or file I/O. The enclave is protected from the rest of the system, whereas the host is not.

"The rest of the system" from which the enclave is protected includes:

- the enclave's own host
- other applications
- the operating system
- the hypervisor and host operating system
- the bootloader and other firmware such as the BIOS
- hardware other than the processor

Enclave and host talk to each other over an interface (or API) that is defined in advance. The host can call into the enclave, e.g., to have it perform a cryptographic operation. These calls are also referred to as *ecalls*. The enclave relies on the host to interact with the rest of the system. For example, to write to disk or to receive data over the network, the enclave needs to call into the host. These calls are referred to as *ocalls*. Ocalls are required because the enclave cannot talk to the operating system and the hardware directly. The host needs to do this for the enclave.

Splitting an application into enclave and host such that the result runs securely and stably can be challenging. The bigger the application, the bigger the challenge. Hence, a common approach today is to move all functional parts of an application into the enclave and use a boilerplate host for I/O. Modern SDKs and frameworks often even hide the split between host and enclave from the programmer. We discuss SDKs, frameworks, and other tooling in section 5.

Another implementation of the enclave concept is Nitro Enclaves, which are only available in AWS. As with SGX, programmers are supposed to split their apps into an untrusted host and a trusted enclave. In the case of Nitro Enclaves, host and enclave are two separate VMs. These

are normal VMs running on standard Intel or AMD processors. The enclave can only directly talk to the host. It cannot access the network or the storage by itself. The communication between enclave and host happens via a so-called VSOCK.

The AWS Nitro platform, which includes a custom hypervisor and a PCIe card, provides the full set of attestation features for the enclave. Depending on the underlying Intel or AMD processor, a Nitro Enclave's memory may also be encrypted at runtime.

It is important to note that the intended use case for Nitro Enclaves is separation of privilege within an app, e.g., for more secure key management, and not protection against the cloud operator or infrastructure. In fact, AWS states that a "Nitro enclave has the same level of protection from the cloud operator as a normal Nitro-based EC2 instance, but adds the capability for customers to divide their own systems into components with different levels of trust." Consequently, there is some debate in the community whether Nitro Enclaves provide "real" confidential computing or not.

Today, people are also using Nitro Enclaves to run entire applications, and they have built runtime frameworks to enable transparent network and file I/O over the VSOCK. This approach enables one to apply the attestation features of Nitro Enclaves to entire apps. The downside is that the entire network and file I/O of the "enclaved" app needs to go through the VSOCK, which has performance implications.

3.2 CONFIDENTIAL VMS (CVMS)

The concept of CVMs postdates that of secure enclaves. It's actually a pretty simple concept which can be summarized as follows: "Take the three defining CCE properties and apply them to entire virtual machines." Thus, other than secure enclaves, CVMs can basically run any workload without requiring modifications. Ease-of-use is the key advantage of CVMs. On the other hand, at least when compared to SGX, CVMs have a larger attack surface because they run an operating system[3] and directly interact with a potentially complex set of hardware. Still, other than Nitro Enclaves, CVMs are explicitly designed to shield workloads from the infrastructure, including the cloud operator. These trade-offs are visualized in *Figure 2*.

CVMs were pioneered by AMD with their Secure Encrypted Virtualization (SEV) feature. The latest instalment of SEV, called SEV-SNP (for Secure Nested Paging), introduced general purpose attestation. Prior to this, SEV only supported attestation for certain use cases. SEV-SNP is available in AMD EPYC server processors of the "Milan" generation. Intel and Arm have both announced similar features for their server processors. In Intel's case, the feature is called Trust Domain Extensions (TDX). Abstractly, TDX offers the same features as SEV-SNP. Its attestation infrastructure is based on that of SGX. Arm's CVM feature is called Realms. It was announced in 2021 for the Arm v9 processor architecture. Conceptually, it is like SEV and TDX and aimed at server processors. To date, still no corresponding hardware

3 To be precise, AWS Nitro Enclaves also run an entire OS. From an attack surface perspective, they thus resemble more a CVM.

appears to be available in the market. Given the support from the three major processor vendors and designers, it seems fair to predict that the CVM concept will dominate the confidential computing landscape going forward.

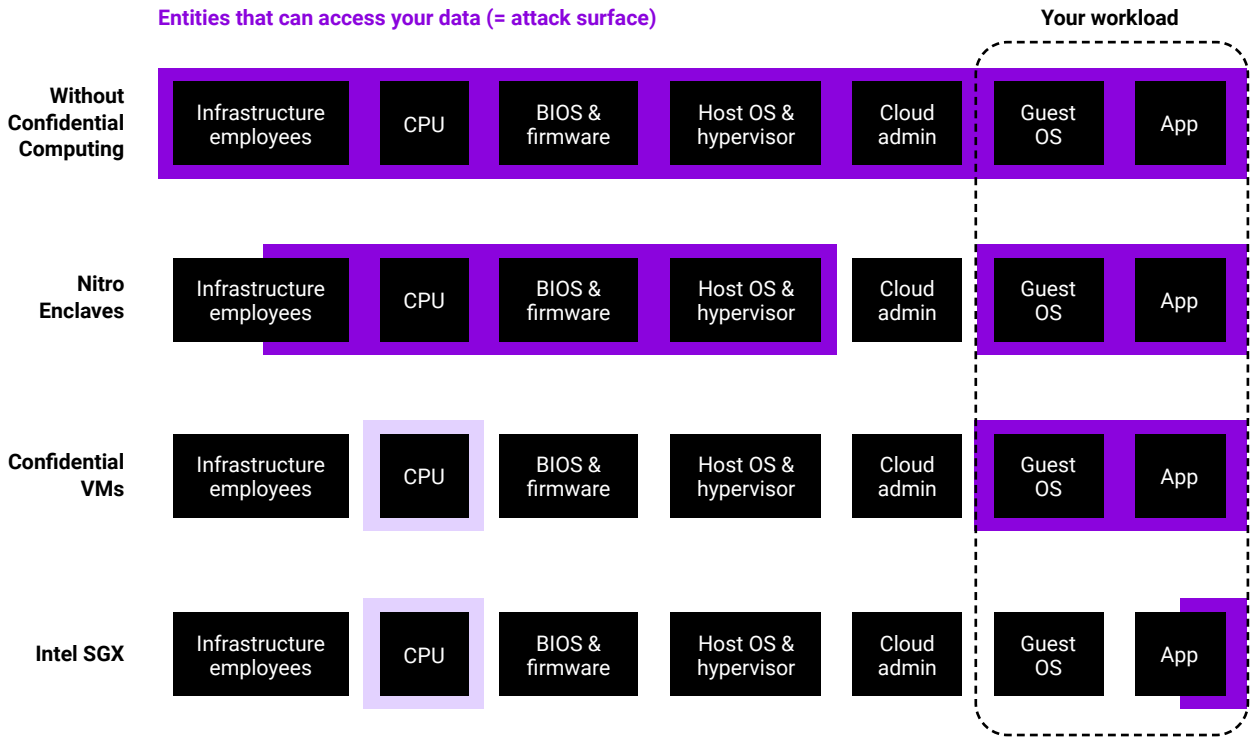

Figure 2: Simplified comparison of the trusted computing base (TCB) of the different confidential computing technologies. "Infrastructure employees" includes cloud provider and datacenter employees. For Intel SGX and CVMs, the CPU enforces the secure execution context. It is thus marked with lighter purple.

3.3 SECURE ENCLAVES VS. CVMS, WHICH ONE TO USE?

So, which one should you use, secure enclaves or CVMs? The simple answer is that secure enclaves make the most sense for use cases where fine-grained separation and a minimized attack surface are required. CVMs make the most sense for cases that involve more complex applications that may be difficult to port to a restricted enclave environment.

Further, hardware availability is an important point to consider: to date, Intel SGX is only available in Azure and some smaller clouds. Nitro Enclaves are only available on AWS. CVMs are available in Azure and GCP, but not in AWS.

4. USE CASES

Confidential computing has a large range of potential applications. These can be roughly divided into the following two classes:

1. Making existing applications more secure
2. Building new privacy-preserving applications

We discuss these in the following.

4.1 MAKING EXISTING APPLICATIONS MORE SECURE VIA LIFT AND SHIFT

The idea is straightforward: running existing applications inside CCEs increases security. If implemented correctly, this can protect applications from attacks from the infrastructure layer. This includes a compromised host operating system (e.g., through malware or rootkits), malicious system administrators, and physical attackers. The latter could, for example, be a datacenter employee who attaches a probe to a memory bus. Fundamentally, it is possible to construct CCE-based applications that are verifiably shielded from the infrastructure. This yields two key benefits: First, it prevents data breaches. Second, it helps with compliance.

The compliance aspect is particularly interesting in a cloud or SaaS context. Internal policies and external regulation prevent companies and institutions from using cloud or SaaS offerings with certain types of data. This is unsurprising because using these offerings effectively means entrusting the respective provider and their infrastructure with the data. In essence, if a SaaS company gets hacked, all of their customers' data are at risk. With confidential computing, this changes. Regulated institutions and companies may now for the first time be able to benefit from the efficiency and scale of the cloud and "as-a-Service" offerings. This could create tremendous amounts of value and could turn out disruptive for solutions focused on "on-premises" and "virtual private cloud" deployments. However, to realize this value, more will be required than simply running applications inside CCEs. Attestation is always required. Without it, most benefits of confidential computing collapse as "counterfeiting attacks" become possible. In such an attack, a compromised or malicious infrastructure would claim to be running a CCE and would convince a client to send sensitive data which could then be leaked or manipulated.

A large in-production example of an app that got additional security from confidential computing is the German eRezept. The eRezept, which translates to "e-prescription", is a nationwide digital prescription service commissioned by the German government and implemented and operated by IBM with the help of Intel SGX. The eRezept is an end-to-end service that connects doctors, pharmacies, and patients. Its purpose is to digitalize the existing paper-based drug prescription system while keeping the patients' data secure.

Since CVMs offer better compatibility with existing software than secure enclaves (i.e., CVMs don't require splitting between enclave and host), we expect CVMs to become the dominant approach for making existing apps more secure.

CASE STUDY: CONFIDENTIAL AI PIPELINE

Bosch works at the forefront of artificial intelligence and autonomous driving. As data privacy and compliance are key requirements when gathering and training neural networks, Bosch wanted to leverage confidential computing for these applications. Based on EGo and EdgelessDB, Bosch and Edgeless Systems together built an end-to-end confidential AI pipeline. With the solution, Bosch strengthened AI data compliance and security. Learn more about the project in this video.

4.2 BUILDING NEW PRIVACY-PRESERVING APPLICATIONS

Confidential computing enables new types of apps for data sharing or pooling that weren't possible before. As an example, consider two companies wishing to identify certain shared customers without revealing their customer databases to one another (or a third party). Solving this is virtually impossible with technologies other than confidential computing, in particular, if the databases are large. In contrast, with confidential computing, it's straightforward to sketch a solution:

- **Step 1:** Create an app that implements the desired functionality, that is, intersect two given customer databases without leaking data.
- **Step 2:** Run the app in a CCE.
- **Step 3:** Send the CCE certificate (see section 2) to both companies.
- **Step 4:** The companies verify that the measurement reflected in the certificate corresponds to the expected app.
- **Step 5:** The companies establish encrypted connections to the app based on the CCE certificate, send their data, and receive their results. The companies know that their data will only be used for the intended purpose and that no one will get access to the raw input data.

Essentially, all new types of apps enabled by confidential computing are variants and extensions of this simple example. One can easily come up with more examples: joint training of AI models, benchmarking of business metrics, analysis of census data, joint identification of money laundering activities, or discovery of shared contacts between the users of a messaging app (we describe a corresponding real-world case further down). Essentially, every multi-party computation scenario can be securely and efficiently implemented using confidential computing. Conceptually, this has much in common with smart contracts from the blockchain space. Blockchain-based smart

contracts also have fixed functionality that can be examined prior to invocation. However, there are also important differences. Blockchain-based smart contracts typically cannot provide confidentiality and aren't suitable for larger amounts of data and frequent invocations. On the other hand, blockchain-based smart contracts are typically also decentralized and don't require trust in a particular processor vendor.

A popular example of a privacy-preserving app that was made possible by confidential computing is the Signal Messenger. Signal uses Intel SGX in the backend to implement privacy-preserving contact discovery. In a nutshell, for contact discovery, the Signal app on a user's phone contacts the SGX-based backend, verifies the backend's CCE certificate, and then sets up a secure connection based on that. It then sends over the phone's address book entries in hashed form. The enclaved backend checks which of the entries correspond to existing Signal users and lets the user's Signal app know. The user can then message these contacts. Neither employees of Signal nor those of the cloud provider running the backend can ever access the users' contacts. The same is true for potential hackers with root access to the backend servers. These are very strong guarantees that are virtually impossible to achieve with technology other than confidential computing. These guarantees give Signal a clear USP over competing messengers like WhatsApp.

5. THE SOFTWARE ECOSYSTEM

Without the right tools, frameworks, and apps, confidential computing is mostly useless. There are different types of software for confidential computing, which we discuss in the following. The discussion focuses on open-source software.

5.1 SOFTWARE DEVELOPMENT KITS (SDKS) AND FRAMEWORKS

SDKs and frameworks enable programmers to build or port confidential apps. There are three core features that SDKs and frameworks for confidential computing typically provide: (1) abstraction of CCE features, in particular remote attestation, (2) packaging and signing of apps, which is also crucial for remote attestation, and (3) providing of a POSIX-like runtime environment with support for network and file I/O, multi-threading, and more.

(3) is primarily important in the enclave context, as both Intel SGX and AWS Nitro Enclaves don't provide direct access to the underlying OS. Open-source frameworks such as EGo (for Go) from Edgeless Systems and OpenEnclave (for C/C++) or Gramine, Occlum, Enarx, and Mystikos (for multiple languages, including Java, C#, and Python) address this in the case of Intel SGX. With their help, enclave programmers can keep using standard libraries to, for example, access the network or files. Language-focused frameworks like EGo map fea-

tures and concepts of a specific programming language to the enclave context and are typically more optimized in terms of performance, compatibility, stability, security, or ease-of-use. Generic frameworks like Gramine typically aim to provide as much system-interface functionality (e.g., POSIX) as possible inside the enclave without the help of the outside world. This is also referred to as the "library OS" approach. This approach is largely, but not entirely, programming-language agnostic. In the ideal case, it allows to port existing applications with little or no effort, regardless of the programming language. However, in many cases re-compilation is still required, and library OSes often also do not cover all edge cases. For Nitro Enclaves, AWS provides tooling to convert and run standard Docker images. At least in theory, apps written in any programming language can be ported to Nitro Enclaves this way. However, as discussed, Nitro Enclaves by design don't offer things like network and file I/O. To address this and enable the lift and shift of existing applications that rely on these, some commercial tooling exists.

For CVMs, a compatible (or "enlightened") OS kernel that knows how to provide (3) is required. As of the current version 5.18, the Linux Kernel largely does this for AMD SEV-SNP and will likely have the same support for Intel TDX and Arm Realms once these are available in the market. While (1) and (2) are also crucial in the CVM context, there still only exists a small number of corresponding tools exists. One of them is the Kata Containers open-source project which boots single Docker containers inside CVMs with a minimal runtime environment. This concept is also referred to as "confidential containers". The concept is somewhat similar to that of Nitro Enclaves. However, a key difference is that AMD SEV-SNP-based confidential containers are protected against the cloud provider.

EGO: BUILD APPS FOR SGX ENCLAVES WITH EASE

EGo is an open-source SDK for building confidential apps in the Go programming language. It comprises a modified Go compiler, additional tooling, and a Go library.

In short, EGo enables developers to run virtually any Go program inside SGX enclaves without modifying existing code.

5.2 ORCHESTRATION AND DEPLOYMENT TOOLS

When running confidential apps in production, typically the following questions arise:

- How to securely provision confidential apps with parameters and secrets?
- How to migrate the apps from one physical host to another without losing keys?
- How to recover the data of a failed confidential app?
- How to set up secure (=attested) connections between apps?

- How to attest a collection of connected confidential apps from the outside?
- How to do software updates without breaking trust relationships?

These questions become even more acute when one considers a modern scalable microservice architecture running on Kubernetes.

To date, there exists only a small number of tools that addresses these questions. In the context of secure enclaves, possibly the only available open-source tool is MarbleRun. It addresses the above questions for apps and services built with EGo, Gramine, or Occlum for Intel SGX. MarbleRun is an enclaved service (built with EGo) that exposes a REST API. For a given deployment of confidential services, MarbleRun is provisioned with a so-called "manifest" file. The manifest defines the core properties of a so-called "confidential deployment", including which types of services are allowed to run, which parameters and secrets exists and who has access to them, and who may recover a failed deployment. MarbleRun enforces the manifest for the entire deployment. It uses remote attestation to identify services and to bootstrap secure connections within the deployment. For an external user of a deployment, it is sufficient to check the CCE certificate of the MarbleRun service, which includes a hash of the manifest, to verify that a deployment adheres to a given manifest. Subsequently, the user can establish a secure connection to any exposed service in the deployment. In sum, MarbleRun extends the remote attestation and encryption features of confidential computing to entire deployments of confidential services. A deployment may consist of a single confidential service or several hundred. MarbleRun is designed to be run on Kubernetes in conjunction with existing service meshes. It handles dynamic scaling of confidential services. MarbleRun primarily aims at enabling new privacy-preserving apps (use-case class 2) and only secondarily aims at making existing apps more secure (use-case class 1).

MARBLERUN: SCALE SGX-BASED APPS IN THE CLOUD

MarbleRun is an open-source framework that simplifies managing, scaling, and verifying SGX-based apps. It is designed to run on unmodified Kubernetes alongside your existing tooling. Think of MarbleRun as a service mesh for secure enclaves.

Constellation from Edgeless Systems is an open-source framework that shields entire Kubernetes deployments, including Kubernetes itself, from the infrastructure. It shares core concepts with MarbleRun but builds upon CVMs instead of enclaves and focuses on use-case class 1 and ease-of-use. For DevOps engineers, a Constellation deployment works like a normal Kubernetes deployment. They can launch and scale unmodified containerized services as usual. However, from the outside, the whole deployment is shielded from the infrastructure and all data is always encrypted. Similar to MarbleRun, a Constellation deployment can also be verified end-to-end based on

a single CCE certificate, regardless of the size of the deployment.

5.3 OTHER BUILDING BLOCKS

Some higher-level confidential apps and tools exist that can be used as building blocks in other apps. This section introduces two of them.

EdgelessDB from Edgeless Systems is an open-source SQL database that is optimized for Intel SGX. EdgelessDB is a MariaDB fork and comes with built-in confidential-computing features. Most notably, EdgelessDB's attestation reflects the initial state of the database in the CCE certificate. Further, EdgelessDB ensures that no plaintext data ever leaves the enclave. EdgelessDB encrypts data on disk in such a way that integrity of the database state is guaranteed as a whole, even in the face of an active attacker. This makes EdgelessDB a versatile tool for securely storing and analyzing data. A common use case is to use EdgelessDB as secure storage backend in confidential apps.

Confidential Consortium Framework (CCF) from Microsoft is an open-source tool for running smart contract-style applications on a distributed set of Intel SGX enclaves. CCF focuses on high transaction throughput, accountability, and verifiability. CCF writes a signed and partly encrypted ledger, which reflects the state of a deployment.

EDGELESSDB: THE WORLD'S FIRST CONFIDENTIAL SQL DATABASE

EdgelessDB is an open-source SQL database, tailor-made for confidential computing. As it runs entirely inside a secure enclave, EdgelessDB's data, cryptographic keys, and code are always protected. Thus, EdgelessDB prevents even privileged attackers, like malicious administrators or rootkits, from accessing a database's memory. In this way, there is no need to worry about the server machine being compromised.

6. WHERE WE'RE HEADED

There is a growing chorus of industry experts who see confidential computing as one of the computer industry's next big trends, following a decade of industry players working to putting crucial pieces in place. For instance, Mark Russinovich (CTO, Microsoft Azure) stated at the OC3 2022 conference that "confidential computing is the future of computing in general."

While in earlier years confidential computing could realistically only be used by developers willing to stitch together a range of low-level tools, confidential computing is now increasingly becoming a push-button feature. Correspondingly, we expect confidential computing to become a must-have for most cloud and SaaS workloads. People will simply expect CISOs to protect their companies' workloads with confidential computing, regardless of the specific sensitivity of a workload. Much like it is expected and required to mitigate risk with firewalls and antivirus in today's on-prem scenarios.

Correspondingly, it is unsurprising that according to a recent study by the Everest Group, the confidential computing market will enjoy a compound annual growth rate of as much as 90%-95% over the next five years, topping $52 billion by 2026. This will be driven by sales of hardware, software and emerging services within regulated industries and the need to meet privacy regulations and incidences of cyber threats, with emerging paradigms such as multi-party computing and blockchain acting as further accelerators.

CONSTELLATION: END-TO-END CONFIDENTIALITY ON ANY PUBLIC CLOUD

Constellation is a Kubernetes engine that leverages confidential VMs to isolate entire Kubernetes deployments from the infrastructure. It enables end-to-end encryption (even in use) and verifiability on all major clouds without requiring changes to existing containerized applications.

> ***"Confidential computing is the future of computing in general."***
>
> Mark Russinovich, CTO at Microsoft Azure

7. WHERE TO LEARN MORE AND TO GET INVOLVED

Confidential computing is still an emerging field. The community of developers and practitioners is growing and there's a range of online resources and events for both beginners and experts.

Probably the largest online event in the space is the free Open Confidential Computing Conference (OC3), which is organized annually by Edgeless Systems. In 2022, the event featured 19 talks and more than 500 active participants. The recordings of the talks are available online.

The talks cover applications of confidential computing (e.g., the eRezept) as well as low-level technical aspects. The focus is on open-source projects.

The Confidential Computing Consortium (CCC) is the central industry consortium for confidential computing. Its members include big tech companies as well as startups. The CCC has published its own whitepapers and a market study for confidential computing.

OPEN CONFIDENTIAL COMPUTING CONFERENCE

OC3 is an annual conference that brings together the confidential computing community. In 2022, OC3 speakers included the CTO of Microsoft Azure and thought leaders from companies like Apple, Google, Baidu and Intel. Aimed at security engineers, cloud architects, and IT decision makers, the event features a broad range of topics from confidential computing use cases to cloud-native to "low-level magic". All talks are available on YouTube. The next OC3 will be in Q1-2023. If you want to join the event, sign up for the Edgeless Systems newsletter to get the latest updates.

GLOSSARY/ABBREVIATIONS

CCE	A confidential computing environment (CCE) is shielded from the rest of a system and privileged individuals. In contrast to special-purpose security hardware like HSMs or smartcards, CCEs can typically run (almost) arbitrary software. CCEs have three defining properties: runtime encryption, isolation, and remote attestation.
CVM	A confidential virtual machine (CVM) takes the defining properties of a CCE and applies them to an entire virtual machine. Thus, in contrast to secure enclaves, CVMs can basically run any workload without requiring modifications.
Enclave	An enclave is a finer-grained form of a CCE. The most widely known enclave platform are Intel's Software Guard Extensions (SGX).
OC3	The Open Confidential Computing Conference (OC3) is an annual event that brings together leading experts in confidential computing and IT security enthusiasts.
Remote attestation	Remote attestation means that, on demand, the processor can issue a cryptographic certificate that proves the integrity and authenticity of a CCE and data it produced.
Runtime encryption	Runtime encryption means that a processor keeps all of a CCE's data encrypted in main memory. Any system component or hardware attacker attempting to read a CCE's data directly from memory will only ever see encrypted data. Likewise, encryption prevents the modification of a CCE's data through direct access to memory.
SEV	AMD's Secure Encrypted Virtualization (SEV) is designed to isolate virtual machines (VMs) from the hypervisor, available in the latest AMD processor generations like Milan.
SGX	Intel's Software Guard Extensions (SGX) are the most widely known implementation of secure enclaves, available in many Intel Xeon server processors.
TDX	Intel's Trust Domain Extensions (TDX) are made available with next-generation Xeon processors that enable confidential VMs, similar to AMD-SEV.
TEE	Trusted execution environments (TEEs) are secure environments for data processing that are created by the processor of a system, which is the basic underlying concept of confidential computing.

ABOUT THE AUTHOR

Felix Schuster is an academic turned startup founder. After his PhD in computer security, he joined Microsoft Research to work on the foundations of Azure Confidential Computing. There, he was one of the first to work with confidential computing and authored a range of foundational scientific papers and patents in that field. He left Microsoft to start Edgeless Systems in 2019. The company's mission is to make confidential computing accessible to everyone, at scale and across clouds and infrastructures.

ABOUT EDGELESS SYSTEMS

Edgeless Systems is a cybersecurity company based in Germany that turns the public cloud into the safest place for your data. Leveraging confidential computing technology, our solutions elevate the security of your applications and workflows to unprecedented levels. Edgeless Systems is a member of the Confidential Computing Consortium that includes companies like Microsoft, Google, Red Hat and Intel, and is backed by renown investors including the Swiss Stock Exchange.