White Paper

Secure Application and Web Service Developers

Orchestrating Confidential Workloads in Sovereign Clouds

T-Systems Open Sovereign Cloud helps secure data in use using Intel® Software Guard Extensions. Intel and T-Systems worked together on a memory-optimized solution designed to preserve data privacy for patient medical records

In response to German market requirements, T-Systems developed the Open Sovereign Cloud (OSC), a platform designed for insurance providers and medical companies. This platform was used to create one of the first Digital-Health-IDs for health insurance in Germany. OSC enables cloud services that allow consumers to use, view, and interact with their personal medical records and other sensitive data.

Because OSC manages sensitive and regulated data, T-Systems needed to design a system for data privacy at all phases, including when data is in use. Thanks to collaboration with Intel, T-Systems developed a system designed for data security and memory-optimized deployment in Kubernetes.

Prioritizing Privacy for Data in Use

A complete cyber security solution encrypts data at various stages to keep it secure from malicious actors. Numerous technologies exist to encrypt data in transit over the network, or data at rest on a storage drive. But not all systems protect data in use – when it is being read, accessed, erased, processed or updated. During this processing, data in use has traditionally not been encrypted. This can expose data in use to sophisticated theft or manipulation attempts.

For T-Systems and its customers to be able to use OSC, it needed to be approved by the German National Agency for Digital Medicine (gematik). The service must also meet the legal requirements of the European Union General Data Protection Regulation (GDPR), the professional secrecy requirements of the German Criminal Code, and the social data protection requirements of the German Social Code.

T-Systems met all these requirements in a couple of ways. The company stored all data on its own servers that are located in Germany. It also used Intel® Software Guard Extensions (Intel® SGX) to create trusted execution environments (TEE). Intel SGX is designed to protect data in use with isolation, encryption, and attestation capabilities to help guard against threats while also allowing users to maintain control of and use of their data.

This protection extends to privileged applications that would normally be able to access the data including the operating system, hypervisor or virtual machine manager, and other privileged processes. Intel SGX currently provides the smallest trust boundary in the data center compared to other confidential computing technologies. With Intel SGX, only the code or functions inside the TEE can access confidential data. Other software in the virtual machine, cloud tenants, the cloud stack, and admins are not allowed access.

Table of Contents

T-Systems Adopts Intel SGX for Open Sovereign Cloud

T-Systems is the IT and digitalization solution subsidiary of Deutsche Telekom, one of Germany's leading integrated telecommunications companies, and an Intel® Network Builders Gold Tier ecosystem member.

The technical foundation of OSC is based on a newly developed infrastructure as a service (IaaS) layer that orchestrates the platform's computing, networking, and storage resources. The open source cloud-native infrastructure provider is based on Kubernetes principles. For OSC, every part of the physical infrastructure is managed with Kubernetes including the servers, switches, routers, and network interfaces. Using Kubernetes for compute, networking, storage, and applications, results in a significant reduction in the complexity of the technology stack.

To automate Kubernetes, OSC uses Gardener, an open-source orchestration and management framework. This framework enables the deployment of Kubernetes as-a-Service to customers using the IaaS Kubernetes API. When a customer creates a new cluster, Gardener orchestrates not only the deployment of the Kubernetes clusters, but also the provisioning of all the infrastructure needed for the clusters.

Due to the high data protection requirements, dedicated compute node pools are provisioned for each tenant on the OSC with the required compute resources running all of the customers' workloads. All services on OSC are provided to the user as-a-service.

BARMER Digital-Health-ID Powered by OSC

An early customer for OSC is BARMER, one of Germany's largest health insurance companies. BARMER used a digital identity solution as-a-service from T-Systems to create a Digital-Health-ID that its customers can use for healthcare information. The Digital-Health-ID is one of the first to be launched in Germany. The Digital-Health-ID represents a significant advantage to BARMER customers who now have secure online access to records.

Intel SGX Helps Reduce the Trust Boundary

Data in use is vulnerable to sophisticated malware that attacks the operating system, BIOS, virtual machine manager (VMM), or system management mode (SMM) or other software or firmware that has special privileges. These hacks can gain access to personally identifiable information (PII) such as banking information, health records or the digital identity to access this data.

Intel SGX is an Intel® Security Engine, available on Intel® Xeon® processors, that can create a TEE, also known as a secure enclave. Intel SGX helps protect the contents of the enclave – including data and applications actively in use – from theft, deletion, or manipulation by inside or outside malicious actors.

One of the challenges this design overcomes is the various trust boundaries in cloud services (see Figure 1). Cloud providers have the control of trust they create for their server configurations (OS, VMM, firmware, hardware, etc.) and employees, but their tenant customers have to rely on the trust created by the third party – the cloud providers – to assure their applications are protected. Other tenants could exist on the same server that may execute vulnerabilities not known to the cloud provider. Using a TEE in the cloud can help isolate and address user concerns over trust boundaries.

Memory Allocation for Enclaves

The enclave's code and data are stored in a subset of DRAM called Processor Reserved Memory (PRM), the size of the enclave can be scaled from an entire application to a single function in order to share the memory with other non-sensitive workflows. The CPU is instructed to reject any direct memory access targeting the PRM in order to protect the enclave.

Intel SGX utilizes an Enclave Page Cache (EPC) to store memory pages that are associated with an enclave. Each EPC has numerous 4kb pages that can be assigned to different enclaves supporting multiple enclaves, a requirement for multi-process systems.

These EPC memory allocations are separate from normal RAM allocations and are managed solely by the Linux Kernel SGX subsystem. The challenge is that the subsystem does not provide any insight into the consumption of EPC memory by the workloads. As described later in the paper, this restriction has been overcome by an open-source initiative from Intel and T-Systems.

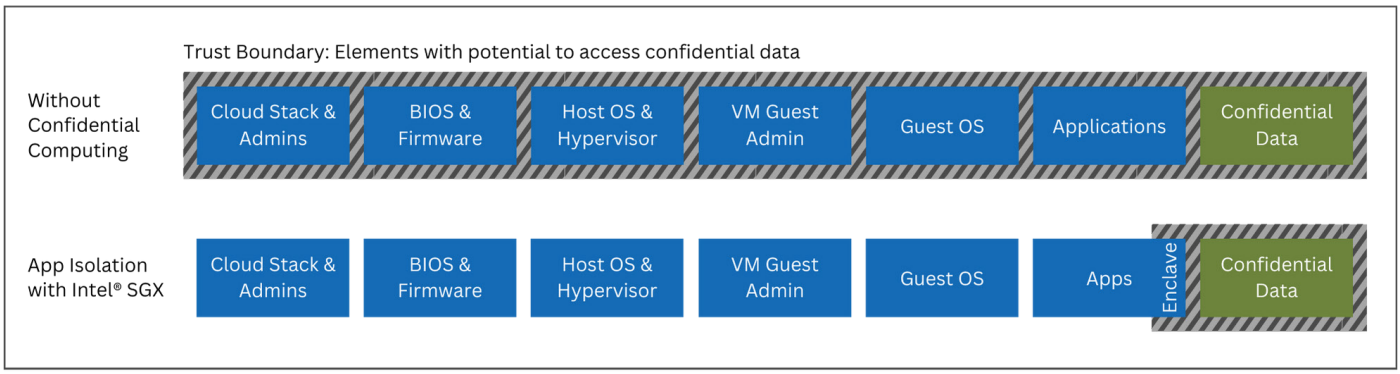

Figure 1. Trust boundaries are shown for two levels of confidential computing starting with no confidential computing and moving to Intel SGX for app isolation. The boxes in the shaded area represent user functions that have access to confidential data.

Intel SGX provides a variety of software development kit (SDK) options that can be used for new application development. Existing applications can be used in an Intel SGX environment with few modifications if the server is running a commercial or open-source library operating system (LibOS). Once the application is ready, it can be uploaded to Intel SGX enclaves offered by a cloud service provider.

Developing a New Resource Management Function

As T-Systems was using Intel SGX to build out OSC, it discovered the need for a resource management function to manage memory consumption within an Intel SGX enclave.

By adding an open-source memory resource management capability to Intel SGX, T-Systems and Intel removed resource uncertainty allowing DevOps teams to effectively manage, control, and monitor the use of encrypted memory for their Intel SGX workloads using the same toolchain that is used for non-confidential workloads and without the need for training.

Working together, Intel and T-Systems contributed to several open-source communities to create a resource management function to measure memory within the enclave.

This approach utilizes the concepts for non-encrypted main memory and uses the node-feature-discovery (NFD) add-on for enabling request-based scheduling, and the node resource interface (NRI) plugin framework supported by the *containerd* and *cri-o* runtimes for enforcing upper limits on encrypted memory using the Linux *misc cgroup*. Both *runc* and *cAdvisor* are used to provide the current resource consumption.

EPC-based Scheduling in Kubernetes

The steps below show how EPC-based scheduling in Kubernetes can be achieved using the existing upstream Kubernetes Node-Feature-Discovery (NFD) add-on:

1. The NFD add-on discovers available EPC memory

2. Adds the available EPC memory as an "extended resource" to the Kubernetes node resources

   a. The NFD add-on must be configured using a custom resource (*nodefeaturerules.nfd.k8s-sigs.io*)

   b. This function is part of the Intel SGX device plugin

3. Extended resources are supported by the Kubernetes default scheduler. No additional components or modifications are required

   a. User defines EPC requests as they do for main memory

Building a Resource Management Feature that Helps Protect Security

Because of their commitment to open-source communities, Intel and T-Systems developed the resource management solution as native code which meant working with various open-source communities to gain acceptance for the changes and test and refine the code before it could be included in future Linux and Kubernetes distributions. Specifically, the components and code that was developed or patched included:

- Container runtime (newly developed NRI plugin)

- runc (upstream PR)

- Linux Kernel (upstream patch)

- cAdvisor (upstream PR)

These contributions provide both monitoring of consumed EPC memory and introducing the enforcement of user-defined limits on the EPC memory consumption by applications.

Figure 2 shows the Kubernetes block diagram including the functionality that is added to the code. The process flows for the resource management solution are as follows:

Process Flow for Defining an Upper Limit of Maximum EPC Memory Consumption:

1. Users declare EPC limit in Pod spec

   a. Currently this declaration is dropped by Kubernetes and not forwarded to the container runtime (e.g. *containerd*)

   b. But Kubernetes is forwarding the annotations of the container

   c. Therefore, the user can declare the desired EPC limit either as an annotation or use the Intel SGX device plugin for translating the declared limits as an annotation (webhook component of the device plugin)

   d. Future versions may not require the webhook. For information, see: https://github.com/kubernetes/enhancements/pull/4113

2. Kubernetes (*kubelet* component) requests declared container from the high-level container runtime (e.g. *containerd* – would also work with *cri-o*)

3. The container runtime forwards the request to the sgx-epc NRI plugin, which transforms the EPC limit annotation into an actual limit definition (adding it to the cgroups v2 unified configuration). Afterwards the plugin returns the request back to the container runtime

4. High level container runtime (e.g. *containerd*) instructs low level container runtime (e.g. runc) to create the actual container (including all cgroup v2 configurations – defining the upper EPC limit in the *misc.max* configuration file)

Misc cgroup controller of the Linux Kernel enforces the defined upper limit and reflects the current EPC consumption in the *misc.current* file. The controller also enforces the defined upper limit (kills the container in case it wants to consume more EPC memory than allowed)

Process Flow for Monitoring EPC Memory Consumption:

1. *Misc cgroup* controller in the Linux Kernel reflects current EPC consumption in *misc.current* file

2. *runc* is a container runtime and it monitors value from *misc* controller for consumed EPC memory and exports this information

3. *cAdvisor* component of the Kubelet fetches the information from runc and exports the information as a Prometheus metric

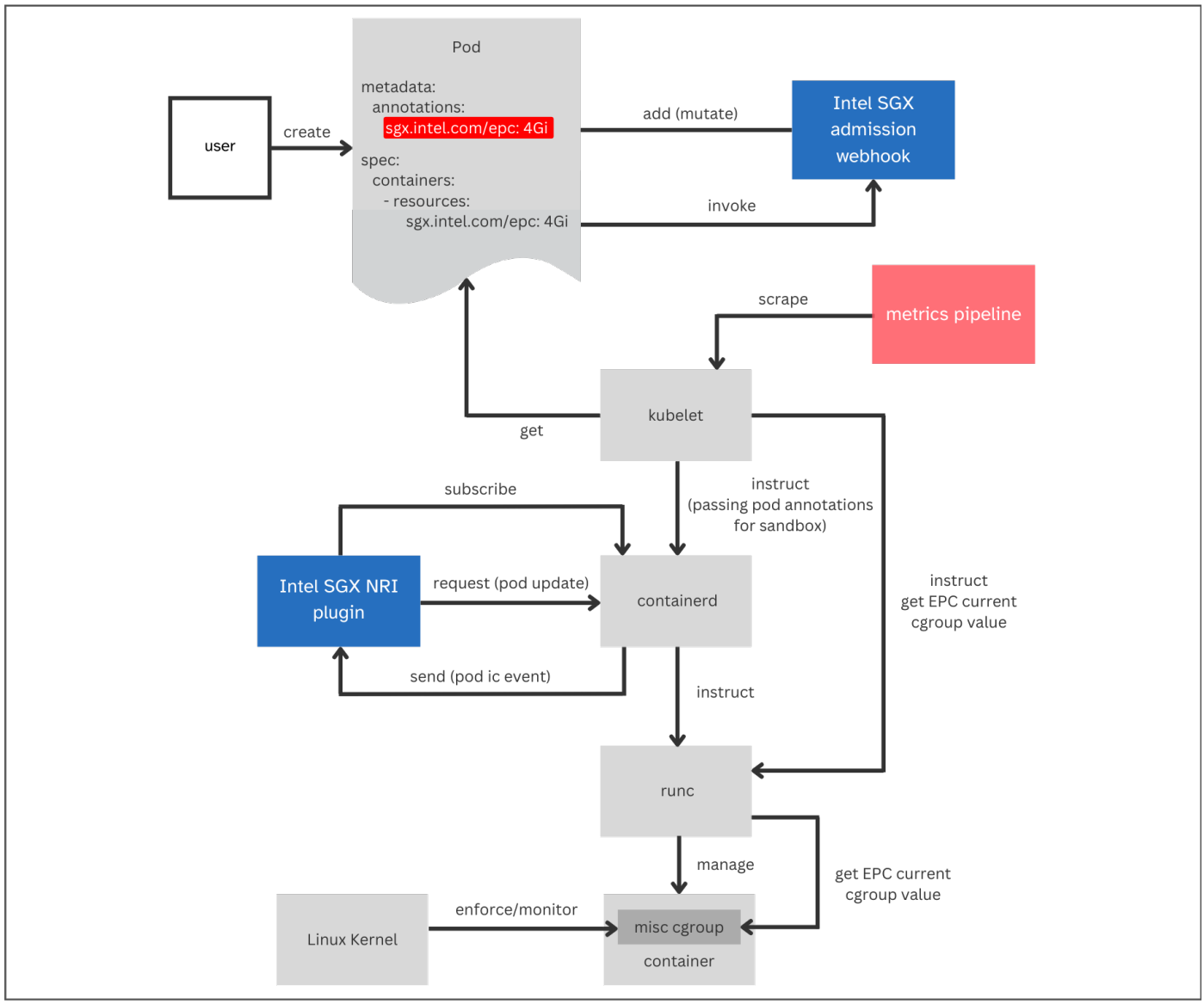

Figure 2. Block diagram showing how an upper limit for the maximum consumed EPC memory can be declared by users and how it is configured for the Linux *misc cgroup* controller.

4. Existing monitoring stacks/solutions for Kubernetes can fetch the metric automatically from *cAdvisor*. It is also possible to deploy *cAdvisor* as a dedicated workload in a Kubernetes cluster. In this way it would be also possible to use the proposed solution, while the Kubernetes community did not implement the "patched" *cAdvisor* release in the kubelet component

5. Users/operators need to adapt their queries/dashboards/alerting rules accordingly based on the new *misc* EPC metrics

Conclusion

Creating a TEE with Intel SGX helps provide security for data in use, which is a key enabler for T-Systems' OSC – a new breed of cloud services that helps customers meet regulations and requirements for hosting applications that rely on PII or other sensitive data.

With Intel SGX, other software in the virtual machine, cloud tenants, the cloud stack, and admins present on the same server are not allowed access to the enclave. The potential for this technology is demonstrated by OSC being used to create one of the first Digital-Health-ID for BARMER, a leading German health insurance company.

In developing OSC, T-Systems engineers, working with Intel, created a resource management function to guarantee stable operations in a production deployment by better managing encrypted memory consumption. Intel and T-Systems worked together on an open-source solution and developed a series of upstream contributions to enable the technology, simplifying operations without changing the de-facto standard tool chain.

The resource management function concept can easily be used for all resources that are supported by the Linux *misc cgroup* controller. While the NFD add-on can be used for supporting resource requests, a new NRI plugin must be developed to support the enforcement of resource limits. Monitoring the function would be covered automatically, because *runc* and *cAdvisor* supporting the Linux *misc cgroup*.

Learn More

BARMER case study: ID and health insurance card in the app

OnMetal API GitHub

NRI GitHub

Intel SGX

Intel Confidential Computing

Intel Network Builders

GitHub contributions:

- Main epic: https://github.com/intel/intel-device-plugins-for-kubernetes/issues/1567
- Runc: https://github.com/opencontainers/runc/pull/3972
- cAdvisor: https://github.com/google/cadvisor/pull/3420
- https://github.com/containers/nri-plugins/pull/156

0426/TM/DJA/PDF Please Recycle 359930-001US