TASK DESCRIPTION

Family name, Given name:	Jing, Yaoxin
Degree Program:	Diplom Information Systems Engineering
Matriculation Number:	4838882
Topic:	***Secure Orchestration of Confidential Process Virtual Machine Enclaves***

Confidential computing has provides guarantees of confidentiality, integrity and consistency properties to application data and code during computation. Presently, there are two hardware approaches to achieve a trusted execution environment(TEE) providing the aforementioned properties: (1) process based enclaves, like SGX, and (2) virtual machine based enclaves, like AMD SEV[1][2] and Intel TDX. There are existing and emerging frameworks targeting each of the different TEEs. Hereafter, the focus will be on VM based enclaves. For secure VMs, the software stack can consist of a full operating system and a user applications like in traditional computing. One alternative approach is to make a software stack with a minimal OS and a single user application process, hereafter referred to as a process VM (pVM). A good example of a pVM is Quark Container framework[3] which will be used for this thesis. This approach helps to reduce the trusted computing base, introduce fast boot times for applications and as well as performance in comparison to the traditional approach. As such, it forms a good ground for further research with respect to secure process VMs in cloud environments.

In Cloud computing, a tenant delegates the management of a host, virtual machine, Kubernetes and application to a cloud provider. In this situation, a TEE protects the user application or virtual machine from direct attack from a malicious cloud operator, hypervisor or operating system. However, the application still needs to be managed by the untrusted cloud provider, even when it is running inside a TEE. For the purposes of orchestration, a cloud provider may use Kubernetes to manage an Open Container Iniative (OCI)[4][5] runtime spec compliant software. This interface, while providing a means to orchestrate an application, opens a new attack surface which can be used to leak data from an application running inside a TEE.

This thesis is aimed at enabling secure orchestration of confidential applications by analyzing the OCI runtime interface and providing mechanisms that effect policies that mitigate emergent risks, in the context of a secure pVM, while still allowing the untrusted cloud provider to continue managing the deployment as before.

Threat Model

There are three sources of threats in consideration.

1. **A malicious supervisor, hypervisor or physical attacker**. This threat is mitigated by running the application in a secure virtual machine like AMD SEV.
2. **Hypercall or system call interface** (see Figure 0.1). With pVMs, guest system calls may be redirected from the guest OS and sent directly to the host through hypercalls or a different interface. This approach could result in use of compromised code (in case of loading libraries during runtime) or leak of secrets, if sensitive data is shared unprotected through the interfaces.

3. **OCI runtime specifications interface**[5] . The OCI specifications interface offers a means to create and manage container images, creation and running of sandboxes and containers, a mechanism to issue any instructions to gather an applications statistics or manage system resources. The latter provides a risk for data and application secrets leakage from within the pVM, and could lead to compromise of an application even from a benign Kubernetes not seeking to access secrets. Securing this interface allows protection from a malicious or Trojan Kubernetes.

Implementation and Evaluation

The work shall be based on Quark Containers [3]. This is a framework written in Rust that turns a VM into a pVM and its architecture is as shown in Figure 0.1.

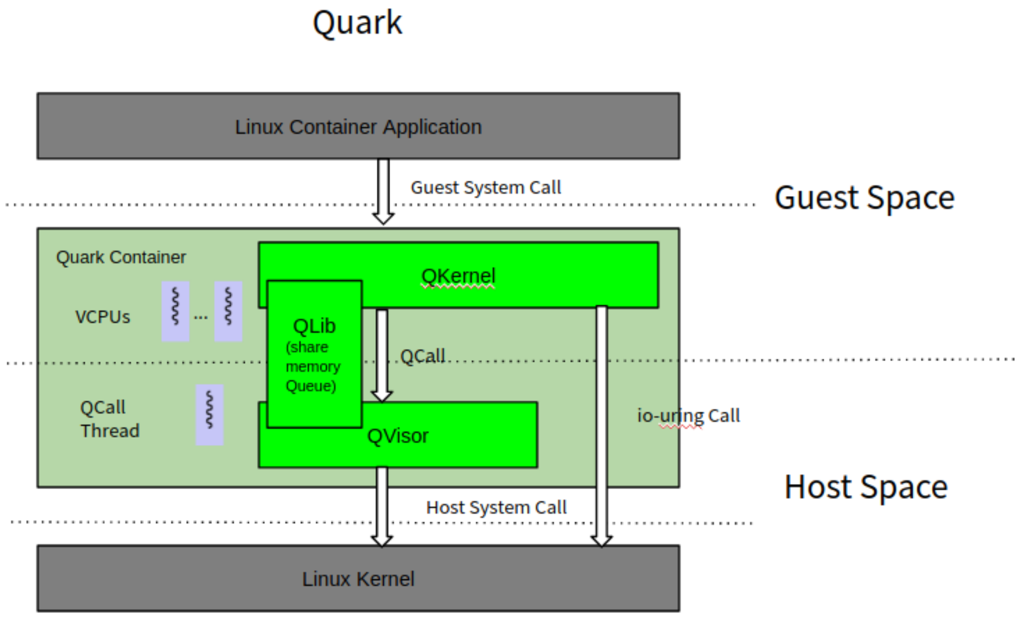

Figure 0.1: The architecture of Quark Container.[3]

Specifically, the student shall carry out the following tasks.

1. Do a security analysis of the hypercall and external system call interface of Quark Containers.
2. Do a security a security analysis of the OCI runtime interface as implemented to interact with QVisor, QKernel and QLib.
3. Analyze the extension of the above interfaces to support specific confidential computing operations like attestation and provisioning.
4. Develop a mechanism to apply policy to protect an application and its secrets from being compromised i.e. a shielding interface, for the above interfaces.
5. Evaluate the developed solution with respect to security and performance.

With regard to evaluation, the following shall be considered.

1. The overhead introduced by the developed interfaces.
2. A list of permissible and non-permissible operations.
3. The impact of allowing and disallowing (2) above.
4. Demonstrate orchestration of a Quark Container application with above changes.

The described task of this thesis shall be undertaken in the period specified for a Diploma thesis. At the end, a written Diploma Thesis and code for the developed solution shall be submitted together.

Advisor:	MSc. Pamenas Kariuki
Supervising Professor:	Prof. Christof Fetzer
Second Reviewer:	Dr.Thomas Knauth
Institute:	System Architecture
Chair:	Systems Engineering
Start:	____________________
End:	____________________

Declaration: Software developed as part of this work can be used by the Systems Engineering Group for further research and teaching purposes.

Dresden, ________________________________
Student

________________________________ ________________________________
Advisor Professor

Bibliography

[1] "SEV Secure Nested Paging Firmware ABI Specification," p. 124, 2022.

[2] D. Kaplan, "AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More," p. 20.

[3] Y. Sun, "Quark Container," Jul. 2022, original-date: 2021-03-04T22:47:16Z. [Online]. Available: https://github.com/QuarkContainer/Quark

[4] "About the Open Container Initiative - Open Container Initiative." [Online]. Available: https://opencontainers.org/about/overview/

[5] "Open Container Initiative Runtime Specification," Aug. 2022, original-date: 2015-06-05T23:30:10Z. [Online]. Available: https://github.com/opencontainers/runtime-spec