TLDR -

Without adequate constraints or attestation, binary provisioning within TEEs can compromise confidentiality by replacing the legitimate provisioning service with a rogue one.

Problem Description

Imagine a data provider who would like to co-locate the business logic of partners as a way to allow low latency access to confidential data while at the same time restricting the business logic from leaking accessed confidential data. On the other hand the partners would want to prevent the data provider to extract their business logic via reverse engineering or execution time analysis.

To protect the business logic the data provider could provide co-located shell TEE that can be provisioned with the proprietary binary. It will start up initially and provide a TLS-attested input channel. Through this channel, the binary can be provided, effectively replacing the initial provisioning service. The co-locator will then impose whatever restrictions are necessary (i.e. firewall) outside of the TEE on the network level.

The challenge arises when provisioning a proprietary binary within a TEE without imposing constraints or attesting to parts of the provisioning process. Without proper constraints or attestation, the binary itself could potentially replace the initial provisioning service with a rogue one. This rogue binary could masquerade as if the TEE hasn't been provisioned yet. Consequently, the security of the TEE becomes compromised.

When the binary lacks constraints, it gains access to attestation reports and can open network services. As a result, it becomes difficult to determine whether the TEE has been provisioned with a legitimate binary or a rogue one. The absence of constraints undermines the integrity of the provisioning process and jeopardizes the overall security of the TEE.

Possible solutions

Solution Type

Description

Pros

Cons

Modify TCB

Add the provisioning user's public key to the TCB, which would allow only the user to initiate the binary provisioning.

This solution can be highly secure as it allows for customization of the TCB.

We need to make use of dynamic runtime attestation (RTMR or vTPM), otherwise VM images would be per single use case

Intermediate TEE

Provision the binary via an intermediate TEE, which controls and attests to the provisioning process.

Provides an additional layer of security that sufficently controls the provisioning process

Adds complexity and overhead to the provisioning process.

Sandboxing (Hardware-Level)

Utilize features like SEV's Virtual Machine Privilege Levels (VMPLs) or potential TDX features. (VMPLs allow a guest VM to divide its address space into four levels for additional security controls and assistance with managing communication with the hypervisor)

Constraining the VM at the hardware level gives very strong security guarantees

It may require specific hardware support. Not all environments may support such features. R&D effort.

Sandboxing (Software-Level)

Utilize software like LXC, bubblewrap, or OCI runtimes to restrict the binary in network access and ability to generate attestation reports

This solution is generally more flexible and easier to implement than hardware-level sandboxing.

It does not provide as strong security guarantees as hardware-level sandboxing.

Sandboxing (SDK-Level)

Provide a Software SDK that requires applications to build atop and sufficiently restricts the logic

Fine grained control over business logic privileges

R&D effort and significant limitation of use cases

References

- bubblewrap - GitHub - containers/bubblewrap: Low-level unprivileged sandboxing tool used by Flatpak and similar projects
- OCI runtimes (GitHub - containers/youki: A container runtime written in Rust, GitHub - containers/crun: A fast and lightweight fully featured OCI runtime and C library for running containers, GitHub - opencontainers/runc: CLI tool for spawning and running containers according to the OCI specification)
- VMPLs (https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/solution-briefs/amd-secure-encrypted-virtualization-solution-brief.pdf - page 13)