August 2023

# A Trusted Execution Enclave within Dual-Confidential Computing Hardware

Jerome Glisse



# A Trusted Execution Enclave within Dual-Confidential Computing Hardware

ABSTRACT

In confidential computing (CC), the program and the data are expected to be provided by the same user. However, there has been a rapid rise in the number of workloads where the program and the data come from distinct users wanting to maintain confidentiality from each other. This disclosure describes techniques to create trusted execution enclaves (TEE) within dual-confidential computing (DCC) hardware. In DCC-TEE, the program is encrypted using a key controlled by the program owner. The user data is encrypted using a key controlled by the user. The encrypted program and the encrypted user data are both sent to the enclave, where the program can access user data in plaintext but is prevented from sending user data outside the enclave. The program can only send back encrypted data to the user with the key owned by the user. The program and the data are thus isolated from each other.

KEYWORDS

- Confidential computing
- Trusted execution enclave
- Trusted computing
- Confidential virtual machine (CVM)
- Cloud service provider (CSP)
- Model security
- Machine learning

## BACKGROUND

In confidential computing (CC), the program and the data are expected to be provided by the same user. For example, in confidential computing (CC), a CPU can enable a confidential

virtual machine (CVM) to run with encrypted CVM data and code, and protects the data and code from the cloud service provider (CSP) or other entity that hosts the virtual machine. However, the underlying assumption in CC is that the CVM is spawned by the owner of the data, such that the owner of the data has clear and legitimate access to program binaries within the CVM.

The emergence of machine learning (ML) has seen a rapid rise in the number of workloads where the program (e.g., the ML model) and the data are from distinct users (e.g., the ML model provider who may also control the model training data) who want to maintain confidentiality from each other. The conventional CC paradigm is inapplicable in situations where a program or algorithm vendor such as an ML model provider wishes to sell their program as a service while keeping the program confidential.

A trusted execution enclave (TEE) is an area on the processor of a computing device that is separated from the main operating system (OS) of the device, and ensures that data is stored, processed, and protected in a secure environment with no access to such data to entities outside the TEE.

## DESCRIPTION

This disclosure describes techniques to create trusted execution enclaves (TEE) within dual-confidential computing (DCC) hardware, such as DCC CPUs. In a TEE within DCC hardware (DCC-TEE), the program is provided by one entity (the program vendor) while the data is provided by a different entity (the user or the customer). A DCC-TEE guarantees that the program is confidential such that no one can access the program binary, not even the user or the cloud service provider (CSP). A DCC-TEE guarantees to the user that the user data cannot be accessed by the program vendor or by the CSP.

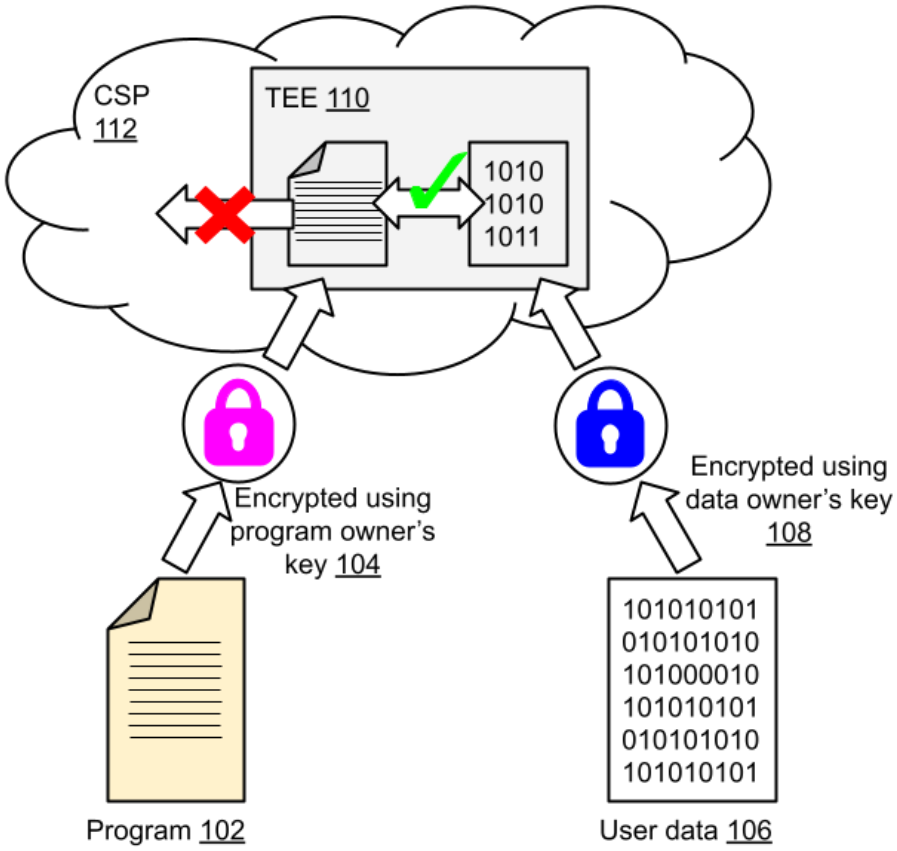


**Fig. 1: Trusted execution enclave (TEE) within dual-confidential computing (DCC-TEE)**

In DCC-TEE, the program and the data are isolated from each other as illustrated in Fig. 1. The program (102), provided by a program vendor, is encrypted using a key (104) controlled by the program owner. User data (106) is encrypted using a key (108) controlled by the user. The encrypted program and the encrypted user data are both sent to the enclave (110), which can be instantiated in a cloud computing facility operated by a cloud service provider (112). In the DCC enclave, the program can access the user data in plaintext, but the enclave prevents the program from sending user data outside the enclave. The program can send back encrypted data to the user which can only be decrypted using a key owned by the user.

DCC-TEE can be based on a container model, wherein the container can run directly on a host without virtualization or can run within a virtual machine. The CPU provides a container

context that is unique to one user and one program, anchored by a certificate and managed by the enclave. The user is provided with a runtime memory that can only be read by the CPU instruction scheduler. An API is provided that enables programming within the user's runtime memory. The API is usable to control the flow of data in and out of the user's runtime memory. The main program is forbidden from creating network connections or making system calls to the host, while such calls or connections can be made from the user runtime memory via the API. No program on the CPU can copy or modify the runtime memory. Like CC, code that implements DCC can be made open source while maintaining security and confidentiality.

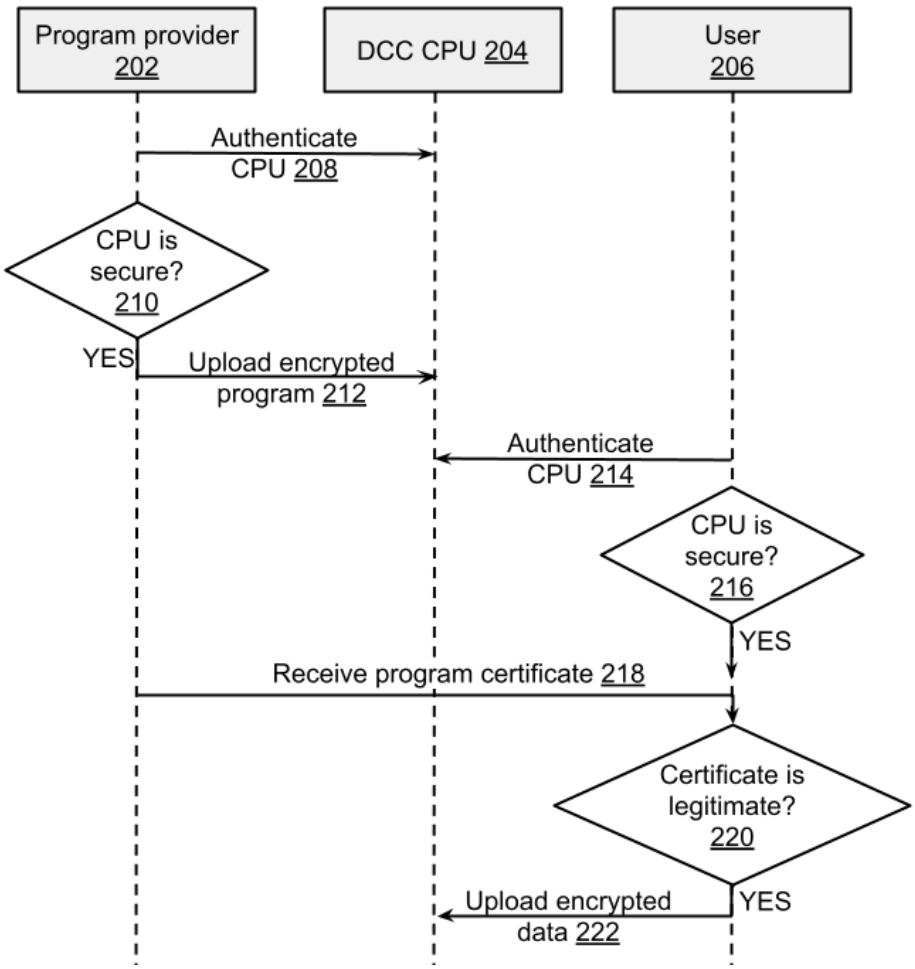


**Fig. 2: Creating a TEE within a DCC CPU**

A DCC-enabled CPU can offer the context of a trusted execution enclave as illustrated in Fig. 2. A program provider (202) authenticates and assesses (208) the CPU (204). If the CPU is deemed secure (210), the program provider uploads an encrypted program to the CPU (212). The program can be decrypted only within the TEE, such that it is kept secure from and inaccessible by the operating system of the CSP. The program provider also provides a certificate for its program such that the user can verify that the program is legitimate and is the one the user intended to run.

The user (206) authenticates and assesses the CPU (214), and, if the CPU is deemed secure (216), the user receives and queries the program certificate (218). Once the user accepts the program certificate (220), the user can upload their data to their runtime memory, which is also the memory region in which the program from the program provider runs (222).

The APIS as described herein can be used to move data in and out of the container over a network. However, the program from the program provider cannot use any CPU or operating system (OS) functions to create any kind of communication channel. The program must go through the user's runtime memory for all of its operations, and only the user's runtime memory can interact with the OS.

Furthermore, the user’s runtime memory can call functions or APIs from the program of the program vendor, but it cannot access data that belongs to the program or program binaries. For example, in the machine-learning use case, the user’s runtime memory cannot access the weights of an ML model.

In this manner, a DCC-TEE enables a user to keep control of their data even as the data is provided to a program whose code is inaccessible to (not inspectable by) them. The DCC-TEE also protects the program vendor from the user and from the CSP. An asset such as a program,

algorithm, or ML model developed and owned by the program vendor can be provided on CSP infrastructure for use by a user without the CSP or the user being able to inspect the program.

## CONCLUSION

This disclosure describes techniques to create trusted execution enclaves (TEE) within dual-confidential computing (DCC) hardware. In DCC-TEE, the program is encrypted using a key controlled by the program owner. The user data is encrypted using a key controlled by the user. The encrypted program and the encrypted user data are both sent to the enclave, where the program can access user data in plaintext but is prevented from sending user data outside the enclave. The program can only send back encrypted data to the user with the key owned by the user. The program and the data are thus isolated from each other.