NVIDIA DRIVE OS Virtualization Software Security Target

Version 2.7

Document History

Doc_Number

Version	Date	Authors	Description of Change
01	August 12, 2024	NVIDIA	Initial release
1.1	October 4, 2024	NVIDIA	Updated TOE scope
1.2	October 15, 2024	NVIDIA	Explicitly specified operations for SFRs
1.3	January 8, 2025	NVIDIA	Updated after Action item list processing
1.4	May 21, 2025	NVIDIA	Updates after code review and product version change
1.5	May 28, 2025	NVIDIA	Updated after Action item list processing
2.0	June 5, 2025	NVIDA	Accepted all changes after review
2.1	July 23, 2025	NVIDIA	Updates for Guidance documentation versions
2.2	July 24, 2025	NVIDIA	Updates for Guidance documentation versions
2.3	August 19, 2025	NVIDIA	Updates after fixing comments from EM1
2.4	August 26, 2025	NVIDIA	Updates after fixing comments from Action item list
2.5	October 16, 2025	NVIDIA	Updates after fixing comments from EM2
2.6	November 20, 2025	NVIDIA	Updates after fixing comments from EM3
2.7	December 1, 2025	NVIDIA	Updates after the Certificate Report review

Table of Contents

List of Figures

List of Tables

Chapter 1. Security Target Introduction

1.1 Security Target Reference

Table 1. Security Target reference

ST Title	NVIDIA DRIVE OS Virtualization Software Security Target
ST Version	2.7
ST Date	2025-12-01
ST Author	NVIDIA Corporation

1.2 TOE Reference

Table 2. TOE reference

TOE Name	NVIDIA DRIVE OS Virtualization Software
TOE Version	6.0.9.3.1
TOE Identification	NVIDIA DRIVE OS Virtualization Software v6.0.9.3.1

1.3 TOE Overview

Note: Nouns starting with capitals denote terms defined in Section 10 (Abbreviations and Glossary).

1.3.1 TOE Introduction

The TOE (Target of Evaluation) is the NVIDIA DRIVE® virtualization solution. It is a type 1 hypervisor developed by NVIDIA Corporation, intended for automotive applications.

TOE supports the execution of spatially isolated Virtual Machines with HV RTOS (HyperVisor Real Time Operating System) Processes in the operational environment provided by other components of NVIDIA DRIVE OS. The TOE uses parameters and resource configurations provided by PCD (Platform Configuration Data), as well as environmental parameters and executing entities supplied by the TCF (Tegra Core Firmware) that ensure the correctness, authenticity, and integrity of the provided data. The configuration is assigned to the TOE during initialization, and then it cannot be altered or updated.

The TOE is responsible for provision of hardware resources and communication channels to the Virtual Machines and HV RTOS Processes in accordance with the configuration. TOE also provides a set of statically configured capabilities to the TOE Logical Partitions to ensure its secure operation.

The TOE does not fully guarantee temporal isolation beyond normal time slicing and scheduling and does not guarantee availability of the TOE functionalities. Nevertheless, TOE guarantees that under every circumstance the TOE meets its secure objectives.

The TOE high-level functionalities include the following:

- Run Virtual Machines in virtualized isolated environments.
- Run auxiliary HV RTOS processes.
- Virtualize hardware resources and provide access to the virtualized hardware resources from both Virtual Machines and HV RTOS Processes.
- Provide communication mechanisms between TOE Logical Partitions.
- Support security functionalities (policy and configuration enforcement; environment separation and identification; and additional security measures such as PAC, stack canaries, interface robustness, and so on).
- Coordinate system-level state transitions.

1.3.2 Usage and Major Security Features

The TOE implements a set of functionalities with the common goal – run Virtual Machines controlled by guest operating systems in an isolated virtualized environment and provide these Virtual Machines with the communication channels and services supported by the TOE in a secure way.

The TOE provides the following security features:

- Spatial isolation between VMs and between VMs and TOE components.
- Identification of TOE components and VMs.
- Controllable access to the services and data provided by TOE.
- Controllable access to the hardware resources provided by NVIDIA DRIVE AGX Orin™ SoC.
- Controllable access to the communication channels provided by TOE.
- Secure access to the functionalities implemented in CCPLEX Secure World.

The TOE is built upon a proprietary microkernel (Hypervisor Kernel) and includes additional entities running outside the microkernel that support functioning of Virtual Machines.

1.3.3 TOE Type

The TOE is a type 1 software virtualization solution developed by NVIDIA Corporation that is intended for automotive applications.

1.3.4 Non-TOE Hardware/Software/Firmware

The following components are necessary for the correct operation of the TOE:

Table 3. Non-TOE Hardware/Software/Firmware

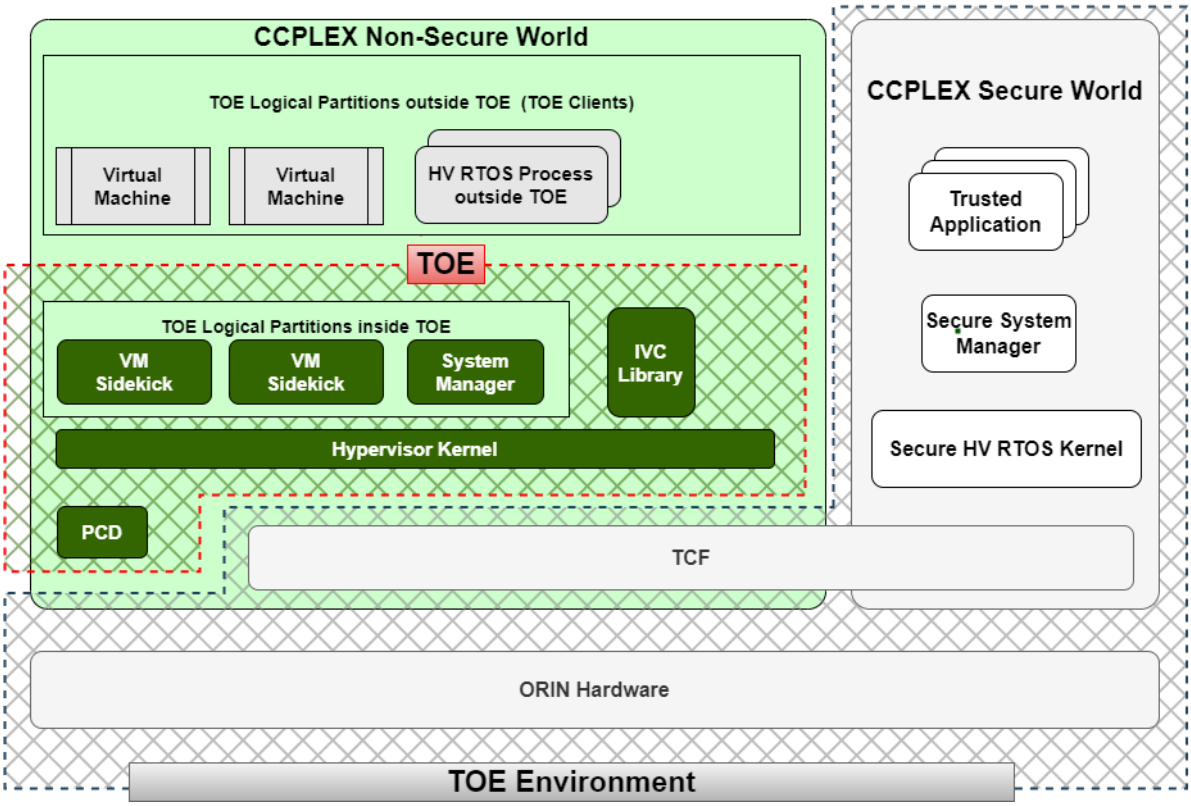
Component	Version	Description
Orin Hardware	T234	NVIDIA DRIVE Orin SoC (ARM v8) Tegra SoC architecture version.
Tegra Core Firmware (TCF)	6.0.9.3.1	This component is responsible for secure boot of the TOE. The component provides calibration data to TOE, specifies boot parameters for TOE, and implements EL3 monitor and PSCI functions.
CCPLEX Secure World Software	6.0.9.3.1	The software running in CCPLEX Secure World interacts both with the Hypervisor Kernel and TOE Logical Partitions.

1.4 TOE Description

1.4.1 TOE Architecture

The TOE uses Orin Hardware features to separate and protect its components and TOE Clients from unauthorised modifications, as well as to prevent potential data leakage. Hardware assisted virtualization enables high performance and provides reliable spatial isolation guarantees to TOE Clients.

Figure 1. TOE High-level architecture



TOE includes the components listed in the table below:

Table 4. Components of the TOE

Component	Description
Hypervisor Kernel	Hypervisor Kernel and platform virtualization drivers. This is part of the TOE that executes in Arm CPU Non-Secure EL2 security state.
VM Sidekick	Hypervisor Kernel extension deployed per VM to offload certain exception handling and VM state management from the Hypervisor Kernel for the VM, allowing the Hypervisor Kernel to be non-preemptible, lockless and SMP, which improves scalability and real-time performance. This is part of the TOE that executes in the Arm CPU Non-Secure EL1 security state.
System Manager	System Manager orchestrates VM and System State transitions. This is part of the TOE that executes in the Arm CPU Non-Secure EL0 security state.
Platform Configuration Data (PCD).	This component provides platform calibration (configuration) data for the TOE and its clients. For additional information, refer to Section 10 Abbreviations and Glossary.
Inter-VM Communication (IVC) Library	IVC Library implements a protocol for peer-to-peer communications between TOE Logical Partitions based on shared memory and notification mechanisms provided by the Hypervisor Kernel. The IVC Library is used both inside and outside the TOE, but only the inside TOE use cases are within the scope of this document.

TOE creates spatially isolated logical partitions using hardware mechanisms provided by Orin Hardware.

TOE Logical Partitions include the following:

- Virtual Machines
- VM Sidekicks
- HV RTOS Processes

Every logical partition runs in its own virtual address space, and the memory contents in this address space are inaccessible from other logical partitions unless the memory is specifically marked as shared (belonging to more than one virtual address space).

TOE Logical Partitions do not include Hypervisor Kernel. Although Hypervisor Kernel runs in a separate address space, it is considered as an outstanding entity responsible for the provisioning of TOE Logical Partitions. Hypervisor Kernel may also have shared memory regions with TOE Logical Partitions.

TOE Logical Partitions outside the TOE are denoted as TOE Clients, which are either Virtual Machines or HV RTOS Processes.

1.4.2 TOE Physical Scope

1.4.2.1 Binary files

The following items compose the physical scope of the TOE:

Table 5. Physical components

TOE components	Description	Version	Format and Distribution method
kernel.bin	Binary image of Hypervisor Kernel	6.0.9.3.1	Direct binary delivery to the DRIVE OS package, available to be downloaded in NVIDIA portal.
sidekick	Binary image of VM Sidekick	6.0.9.3.1	Direct binary delivery to the DRIVE OS package available to be downloaded in NVIDIA portal.
pct.bin	Platform Configuration Table binary	6.0.9.3.1	Direct binary delivery to the DRIVE OS package available to be downloaded in NVIDIA portal.
sysmgr	Binary image of System Manager	6.0.9.3.1	Direct binary delivery to the DRIVE OS package available to be downloaded in NVIDIA portal.

1.4.2.2 Guidelines

The following guidelines are provided:

Table 6. Guidance documentation

Documentation	Version	Format and Distribution Method
NVIDIA DRIVE OS Virtualization Software AGD	1.1	Provided as a part of signed package
NVIDIA DRIVE OS 6.0 QNX Cybersecurity Manual	1.2.1 DRIVE OS 6.0.9.3.1	Provided as a part of signed package, available to be downloaded in NVIDIA portal.
NVIDIA DRIVE OS 6.X SEOOC SPECIFICATION	1.4 DRIVE OS 6.0.9.3.1	Provided as a part of signed package
NVIDIA DRIVE OS 6.0 Safety Manual	1.2 DRIVE OS 6.0.9.3	Provided as a part of signed package
NVIDIA DRIVE OS 6.0 QNX PDK Developer Guide	DRIVE OS 6.0.9.3.1 14/02/2025	Provided as a part of signed package
NVIDIA DRIVE OS 6.0 QNX Installation Guide	DRIVE OS 6.0.9.3.1 02/03/2025	Installation guide for DRIVEOS product: DRIVE-OS-6.0.9.3.1-QNX-Installation-Guide
Interface Control Document of Virtualization System	DRIVE OS 6.0.9.3.1 08/10/2024	Provided via dedicated server within internal NVIDIA infrastructure
DRIVE OS SEooC SWADS	DRIVE OS 6.0.9.3.1 23/03/2025	Provided via dedicated server within internal NVIDIA infrastructure
Software Project Cybersecurity Plan for Virtualization System	DRIVE OS 6.0.9.3.1 12/11/2024	Provided via dedicated server within internal NVIDIA infrastructure

1.4.3 TOE Logical Scope

This section outlines the logical boundaries of the security functionality of the TOE.

The TOE includes the elements of the architecture that support the TOE security functionalities.

In particular, the TOE consists of the components described in Table 4.

TOE provides the following security services:

1.4.3.1 VS_SPATIAL_ISOLATION

The TOE creates and manages TOE Logical Partitions. The spatial isolation is based on the hardware memory access control mechanism provided by configuring Orin Hardware (MMU , SMMU and MSSNVLINK Protection) and TOE security functionalities.

TOE guarantees that the private resources assigned to the given TOE Logical Partition are not accessible from other TOE Logical partitions unless the given TOE Logical Partition explicitly grants access to these.

1.4.3.2 VS_IDENTIFICATION

TOE provides a mechanism to identify the TOE Logical Partitions, which is supported at the Hypervisor Kernel level and is independent of the TOE Logical Partitions themselves.

The PCD provides the Platform Configuration Table (PCT) and Device Trees that specify the calibration data for every TOE Logical Partition based on the identification of the logical partition.

1.4.3.3 VS_SECURE_COMMUNICATIONS

TOE provides communications channels:

> Communication channels between TOE clients and TOE (Hypervisor Kernel):
  - System calls (such as SVC, HVC, and SMC instructions) allow sending a synchronous exception that is trapped at the higher exception level together with data passed through the General-Purpose Registers.
  - Synchronous memory exceptions (Data Aborts) are also used by Virtual Machines to interact with the Hypervisor Kernel, e.g. for emulation of hardware or to communicate events between Logical Partitions.
  - Access to certain ARM System Registers may be emulated by the Hypervisor Kernel and trigger exceptions to be processed.
  - Hypervisor Kernel uses shared memory regions provided to other TOE Logical Partitions to perform data exchange that happens along with exceptions or independently.
> Communications between TOE Logical Partitions:
  - IVC Queue is a peer-to-peer communication channel organized via shared memory regions available to both ends (TOE Logical Partitions) of the IVC Queue and only for them. It works together with a notification mechanism provided by Hypervisor Kernel and communication protocol implemented by the IVC Library.
  - Mempool is a communication channel for TOE Logical Partitions provided in a form of peer-to-peer shared memory regions (that are generally used for DMA transfers)
> Communications between TOE Clients and TOE environment:
  - TOE creates IVC Queues between TOE Clients and services provided by CCPLEX Secure World.

TOE protects the data transmitted through the communication channels from unauthorized access. Only the peers of the peer-to-peer communication channels are authorized to access the data.

1.4.3.4 VS_ACCESS_CONTROL

The TOE provides the following access control mechanisms to managed resources and services:

> The security policies defined by calibration data provided by PCD:

Based on the flags set in PCT, the TOE grants access to its services for TOE Clients.

> Restricted access to hardware resources:

Hypervisor Kernel contains modules that provide access to hardware resources; either as specified in PCT or enforced by Hypervisor Kernel itself.

> Restricted access to the services provided via IVC Queues, Mempools:

The TOE sets up the configuration of IVC Queues and Mempools during initialization. The IVC Queues are either pre-configured or specified through PCT. Mempools are specified through PCT. The information that identifies IVC Queues and Mempools for every TOE Client is provided to the client by the Hypervisor Kernel. Thus, services provided through IVC Queues and Mempools are restricted to the end points of the IVC Queues and Mempools.

Note: TOE logical partitions can share resources on their own behalf and implement their own access control policies. This behavior is out of scope TOE Security functionality.

1.4.3.5 VS_SECURE_STATE

The TOE provides functions to preserve its secure state and properly reacts the violation of SFPs.

- Violation of access control policies results in exceptions that are processed by the TOE.
- Upon detection of a security critical failures, the TOE undertakes measures to preserve the TOE security state.

Chapter 2. Conformance Claims

2.1 Conformance Claim

The TOE and Security Target (ST) claim conformance to the CC:2022 Revision 1 [CC:2022-P1], [CC:2022-P2], [CC:2022-P3], [CC:2022-P4], [CC:2022-P5] and [CC:2022-Errata1].

The ST claims conformance to CC Part 2 conformant and CC Part 3 conformant.

2.2 Package Claim

The ST claims conformance to assurance package EAL4 package-augmented ALC_FLR.1.

2.3 Protection Profile Claims

This ST does not claim conformance with any Protection Profiles.

2.4 Conformance Rationale

No conformance is claimed to any Protection Profile.

Chapter 3. Security Problem Definition

This chapter identifies the following:

- Assets the TOE protects against threats.
- Significant assumptions about the TOE's operational environment.
- Threats that must be countered by the TOE or its environment.

3.1 Assets

Table 7. List of TOE protected security assets

Assets	Description
VM Execution	The code and control flow of the installed Virtual Machine (VM). The TOE guarantees the integrity and authenticity of the initial code of the VM. Though the TOE does not protect the VM code from any activity from within the VM, TOE protects the VM code from modification performed by other TOE Logical Partitions, as well as from injection of malicious code.
VM Private Data	VM Private Data stored in RAM is protected from modification from other TOE Logical Partitions and managed.
Identification	Identification of TOE Logical Partitions that is used to determine the capabilities of these logical partitions are statically configured during initialization of TOE and cannot be altered or spoofed.
TOE Data	TOE data, including execution variables, runtime context, etc. This data is stored in volatile memory and protected from modifications or eavesdropping.
TOE Execution	Integrity and authenticity of TOE code shall be protected from modifications, as well as the control flow integrity shall be maintained to prevent malicious code injections into and of the TOE components.
HV RTOS Process Data	The data, including execution variables, runtime context, and other data, that belongs to HV RTOS processes outside the TOE. This data is stored in volatile memory and protected from modifications or leakage.
HV RTOS Process Execution	The code of HV RTOS processes outside the TOE shall be guaranteed for integrity and authenticity at every point in time as well as the control flow integrity and absence of any possibility to inject a malicious code into any of HV RTOS processes outside the TOE.
TOE Configuration	TOE configuration includes: Assignment of capabilities to access communication channels and TOE services

Assets	Description
	Calibration data provided to TOE including the binary images (PCT, DTB, executables)
Communication Channels	The data sent via communication channels (see Section 1.4.3.3. VS_SECURE_COMMUNICATIONS)
Hardware Resources	Hardware Resources identified through calibration or configuration that are assigned to TOE Logical Partitions. Examples include, SMMU Stream IDs, interrupts, hardware peripherals (MMIO).

Note: Execution and TOE Configuration assets are protected in two stages. First TCF prepares these assets for TOE, ensures its integrity and authenticity using cryptography; and then TOE preserves this property by restricting the access to the assets preventing its modification.

3.2 Assumptions

The specific conditions listed in the following subsections are assumed to exist in the TOE's environment. These assumptions include both practical realities in the development of the TOE security requirements and the essential environmental conditions on the use of the TOE. This document identifies assumptions as *A.assumption* with "assumption" specifying a unique name. (Similar abbreviations are used elsewhere in this document, such as *T* for "threat" and *O* for "security objective.")

Table 8. List of TOE security assumptions

Assumption	Description
A.Secure_Initialization	The TOE shall reach a safe and secure state after the TCF has successfully verified the integrity of the TOE, initialized the hardware required by the TOE, performed the hand-over to the TOE and after the TOE is successfully initialized.
A.Hardware_Platform	The hardware platform where the TOE runs is assumed to correctly operate under normal working conditions. It is assumed that the hardware platform on which the Target of Evaluation (TOE) operates is physically secure and resistant to tampering. The platform must be protected from unauthorized physical access, modifications, and direct attacks, ensuring the integrity and confidentiality of the hardware components and their embedded security features.
A.OE_Security	TCF and CCPLEX Secure World software are trusted by TOE and cannot be used to attack the TOE.
A.No_Internal_Attacker	The TOE configuration or TOE code or other code provided by PCD – everything that is in the scope of secure boot protection performed by TCF does not intentionally or by mistake introduce malicious functionalities (such as backdoors). Authorized internal users or developers are competent and do not intentionally introduce potentially insecure settings to the TOE.

3.3 Threats

TOE considers the following threat agents:

- External attacker that exploits an existing vulnerability in TOE Clients or in the TOE itself.
- External attacker that injects a malicious functionality into code that is not covered by Secure Boot, such as in the application supplied together with user VM.

The following table lists the threats addressed by the TOE and its environment. The assumed level of expertise of the attacker for all the threats identified below is Enhanced-Basic.

Table 9. List of threats considered for TOE

Threat	Description
T.Abuse_Functionality	An attacker abuses the functionality provided by the TOE by using its interfaces incorrectly compared to the Interface Control Document Specification, thus violating the isolation of TOE Client's (assets: VM Private Data, VM Execution, HV RTOS Process Data, HV RTOS Process Execution) or TOE's assets (TOE Data, TOE Execution), with the potential impact in data integrity (VM Private Data, HV RTOS Process Data, TOE Data) and TOE execution state affecting the control flow integrity.

Threat	Description
T.Data_Leakage	An attacker partially or totally recovers the content of the memory, thus disclosing sensitive TOE Clients (VM Private Data, HV RTOS Process Data) or TOE data and potentially allowing the attacker to mount other attacks.

Threat	Description
T.Unauthorised_Access	An attacker gains unauthorized access to TOE services (asset : Communication Channels, TOE Execution, TOE Data, Identification, TOE Configuration) HW resources (asset : Hardware Resources) Any data (VM Private Data, HV RTOS Process Data, TOE Data) or Communication Channels that were not specifically assigned to the TOE Client that the attacker acts from. By gaining this access, the attacker is able to affect the integrity or the confidentiality of the protected assets.
T.Remote_Code_Execution	An attacker injects code/data into TOE (TOE Execution) to execute malicious instructions or use TOE to inject code/data to other TOE Clients (VM Execution, HV RTOS Process Execution)(RCE attack).  **Note**: The situation when the attacker makes a VM to execute malicious instructions from within the VM itself is out of scope. TOE does not provide security guarantees for the internals of VM.

3.4 Organisational Security Policies

Table 10. Organization Security Policies

Assumption	Description
P.System_Integrator	The system integrator shall: Verify that the identification of the hardware platform is compatible before integration; Ensure that the environment (either software, hardware, the OSP for the product in field, or by a mix of these) ensures the following properties: The bootloader shall initialize the hardware so the TOE starts in a safe and secure state. The integrity of the product binary image is ensured when the TOE is loaded and starts running. The correct performance of the integration process according to the TOE guidelines (see Section 1.4.2.2. Guidelines). Installation of the integrated product binary image onto the hardware.
P.Secure_Development	During the TOE development and verification: The architecture and design shall pass through security and safety analysis to exclude potential issues and vulnerabilities. The code shall: Pass peer review to ensure the absence of potential vulnerabilities Pass verification that includes: Static analysis to exclude potential vulnerabilities Unit tests and integration tests (including requirements tests and interface tests) with at least 100% explained branch coverage. In a case when code is not covered, it shall be either excluded or explained to avoid undocumented functionalities. Fuzzing tests to identify potential vulnerabilities.

Chapter 4. Security Objectives

4.1 Security Objectives for the TOE

Table 11. Security Objectives for the TOE

Objective	Definition
O.Spatial_Isolation	The TOE shall enforce measures to protect the private resources that belong to TOE Clients from unauthorized modifications or private data disclosure.
O.Access_Control	The TOE shall prevent the TOE Client from accessing the data, services provided by TOE, communication channels, and HW resources unless they are explicitly assigned to the TOE Client.
O.Identification	The TOE shall provide an identification mechanism for TOE Clients (independent of the TOE Client). This identification information cannot be spoofed or tampered. The Identification Information shall be used to assign resources and capabilities to the TOE Clients as well as to enforce access control policies.
O.Secure_Communications	The TOE shall ensure that data that is passed through the communication channels is not read or modified by an unauthorized entity.

4.2 Security Objectives for the Operational Environment

Table 12. Security Objectives for the Operational Environment

Objective	Definition
OE.Secure_Initialization	The TCF performs correct initialization of the operating environment and securely boots the TOE.
OE.Hardware_Platform	The Orin Hardware operates correctly and does not expose any security-critical side effects on the functionalities of the TOE.
OE.Secure_World_Correctness	CCPLEX Secure World Software operates correctly and does not expose any security-critical side effects of the functionalities of the TOE.
OE.TCF_Correctness	TCF operates correctly and does not expose any security-critical side effects of the functionalities of the TOE.
OE.Trustworthy_Personnel	The Authorized internal users or developers are trustworthy, act according to the TOE user manuals, and are sufficiently qualified for this task.

Chapter 5. Extended Component Definition

This Security Target does not include any extended components.

Chapter 6. Security Functional Requirements

Notes:

- Selections and assignments are highlighted in italic.
- Iterations are indicated by adding a slash (“/”) and an iteration identifier to the SFR, as well as by appending an additional specification to the requirement name (in brackets). For instance, an iteration of the FDP_ACC.2 SFR named ‘Complete Access Control’ may appear as ‘FDP_ACC.2/Memory Complete Access Control (Memory Access) ’.

The user of TOE is a software component running inside a TOE Client. In this case, the TOE Clients act as subjects on behalf of the TOE user. TOE does not distinguish between the user and the TOE Client, which acts as a subject on behalf of the user. If there are many users inside a TOE Client, TOE considers them as a single combined user.

6.1 Identification and Authentication

6.1.1 FIA_UID.2 User identification before any action

FIA_UID.2.1	The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

6.1.2 FIA_ATD.1 User attribute definition

FIA_ATD.1.1	The TSF shall maintain the following list of security attributes belonging to individual users: > *ID: Unique ID assigned to the TOE Client:* > *TOE Client’s access control attributes* > *List of physical memory pages owned by the TOE Client that acts on behalf of the user* > *Set of IVC Queues assigned to the TOE Client* > *Set of Mempools assigned to the TOE Client*

6.1.3 FIA_USB.1 User-subject binding

FIA_USB.1.1	The TSF shall associate the following user security attributes with subjects acting on behalf of that user: > *ID* > *The TOE Client's access control attributes* > *Physical memory page ownership* > *The set of IVC Queues* > *The set of Mempools*
FIA_USB.1.2	The TSF shall enforce the following rules about the initial association of user security attributes with subjects acting on behalf of users: > *The association between the user's identity and the security attributes is made statically during build time or initialization (for physical memory page ownership) and cannot be changed. The only way to change the association is to update the entire DRIVE OS installation or reboot TOE (for physical memory page ownership).* > *The TOE assigns a Unique ID to every TOE Client* > *The TOE assigns access control attributes based on the calibration data provided by PCD (via PCT and DTBs)* > *For every physical memory page used by a TOE Client, the TOE explicitly assigns this page to this TOE Client* > *For every TOE Client, the TOE assigns a specific set of IVC Queues. Each IVC Queue assigned to a TOE Client is connected either to TOE or another TOE Client or to an entity in CCPLEX Secure World. Based on the IVC information, provided by TOE, the TOE Client unambiguously identifies IVC Queue peer. IVC Queues assignment is either enforced by the Hypervisor Kernel (these IVC Queues are always present for TOE Clients), or specified by PCT.* > *For every TOE Client, the TOE assigns a specific set of Mempools. Each Mempool assigned to the TOE Client is connected either to TOE or to another TOE Client. Based on the Mempool information, provided by TOE the TOE Client unambiguously identifies Mempool peer. Mempool assignment is specified by PCT.*
FIA_USB.1.3	The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: > *None*

6.2 User data protection

Access control policies used by TOE include the following:

1. Memory Access Control Policies

   To implement spatial isolation, the TOE implements Memory Access Control Policies. The TOE Clients can access only the physical memory regions that are specifically assigned to them. The TOE uses hardware-based mechanisms to enforce this access control policy.

2. Hardware Resource Access Control Policies

   The TOE Clients utilize hardware resources, including MMIO regions, and Interrupts and Stream IDs to perform their intended functionalities. TOE enforces access control policies. Some of the permissions are configured by TOE and other permissions are defined in PCT. The access to these resources is granted during the initialization of the TOE and cannot be modified.

3. Communications Access Control Policies

TOE assigns IVC Queues and Mempools to the TOE Clients that use these communications mechanisms to access the required services provided by other entities in DRIVE OS. If a communication channel is not established between the TOE Client and the entity that provides a service in DRIVE OS, the TOE Client does not have access to the service. Some communication channels are statically configured in the TOE and others are specified by PCD (through PCT). The communication channels use shared memory to transfer data. To avoid unauthorized access to the data transmitted through the communications channels, the TOE ensures that the shared memory region is available only to the two ends of the communication channel (such as the producer and the consumer of the shared memory).

4. TOE core Access Control Policies

TOE may provide access to the certain TOE services for TOE clients by discretion. In addition to the IVC Queues based access control mechanisms mentioned previously, the TOE grants or denies access to the services based on the TOE Client's security attributes specified by PCD.

6.2.1 Memory access control policy

The TOE manages physical memory pages assigned to the TOE during initialization.

To make the physical memory page available to the TOE Clients, the access to the memory page shall be explicitly granted by TOE.

The TOE Clients may use different types of accesses to the memory:

- Access from CCPLEX (controlled by the Hardware Memory Management Unit)
- DMA access (controlled by the System Memory Management Unit)
- GPU access (controlled by the MSSNVLINK Unit)

A physical memory page is available to a TOE Client if and only if the virtual memory page from the TOE Client's address space is mapped to the physical memory page.

Each virtual memory page can be mapped with the following attributes that can be used in combination:

- MA.READ – Memory page content may be read. If the memory page does not have this attribute and read, the memory content is not read, but an exception is generated.
- MA.WRITE – Memory page content may be written. If the memory page does not have this attribute and access to write to the page is generated, the memory content is not written, and an MMU exception (Data Abort) is generated.
- MA.EXECUTE – Memory page content may be executed as a program. If the memory page does not have this attribute and access to write to the page is generated, the memory content is not executed, and an MMU exception (Data Abort) is generated.

The following derived virtual memory page attributes are defined.

- MA.MEMORY = (MA.READ + MA.WRITE + MA.EXECUTE)
- MA.DATA = (MA.READ + MP.WRITE)

Physical memory pages have the following types depending on the intended usage:

- MT.PRIVATE

- In a case where a physical memory page can be only used by the TOE Client that owns the memory and cannot be used from the TOE Logical Partitions or TOE Environment. For VM - MA.MEMORY. For HV RTOS Process Memory Attributes can have different set of memory access policies (MA.DATA or MA.EXECUTE).

> MT.SHARED (Communication)

- In the case where a physical memory page besides the TOE Client can be used by other TOE Logical Partitions, or the Hypervisor Kernel can read and write data or the TOE Environment (MA.DATA)

> MT.CALIBRATION

- In the case where the memory is used to provide data only in one direction: from provider (MA.DATA) to consumer (MA.READ)

TOE Logical Partitions (as subjects) are identified by their Unique IDs.

6.2.1.1 FDP_ACC.2/Memory Complete access control (Memory Access)

FDP_ACC.2.1/Memory	The TSF shall enforce the *Memory Access Control SFP* on: > *Subjects:* • *TOE Clients* ○ *S.Virtual_Machine* ○ *S.HVRTOS_Process* > *Objects:* • *OBJ.Physical_Memory_pages* and all operations among subjects and objects covered by the SFP.
FDP_ACC.2.2/Memory	The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

6.2.1.2 FDP_ACF.1/Memory Security attribute based access control (Memory Access)

FDP_ACF.1.1/Memory	The TSF shall enforce the *Memory Access Control SFP* to objects based on the following: > *Subjects:* • *TOE Clients* ○ *S.Virtual_Machine* ○ *S.HVRTOS_Process* > *Objects:* • *OBJ.Physical_Memory_Page* • *OBJ.Virtual_Memory_Page* > *Operations:* • *OP.READ* • *OP.WRITE* • *OP.EXECUTE* > *Security attributes:*

	• *Subjects: ID*
	• *Objects*
	  ○ *OBJ.Physical_Memory_Page*
	  ▪ *Memory Page Type: MT.PRIVATE, MT.SHARED, MT.CALIBRATION*
	  ▪ *Ownership (linked to Subject ID)*
	  ○ *OBJ.Virtual_Memory_Page*
	  ▪ *Memory Page Attributes: MA.READ, MA.WRITE, MA.EXECUTE, MA.MEMORY, MA.DATA*

FDP_ACF.1.2/Memory	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: > *OP.READ access to physical memory page is allowed for TOE Client if and only if:* *Read.Policy.1:* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.Virtual_Machine* *AND (Memory Page Type is MT.PRIVATE* *OR (Memory Page Type is MT.SHARED and S.Virtual_Machine is one of two peers of the communication channel)* *OR Memory Page Type is MT_CALIBRATION)* *OR* *Read.Policy.2:* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.HVRTOS_Process* *AND ((Memory Page Type is MT.PRIVATE AND Memory Page Attribute is MA.DATA OR MA.READ)* *OR (Memory Page Type is MT.SHARED AND S.HVRTOS_Process is one of two peers of the communication channel)* *OR Memory Page Type is MT_CALIBRATION)* > *OP.WRITE access to physical memory page is allowed for TOE Client if and only if:* *Write.Policy.1:* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.Virtual_Machine* *AND (Memory Page Type is MT.PRIVATE* *OR (Memory Page Type is MT.SHARED and S.Virtual_Machine is one of two peers of the communication channel)* *AND Memory Page Attribute is MA.DATA)* *OR* *Write.Policy.2:* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.HVRTOS_Process* *AND Memory Page Type is MT.PRIVATE AND Memory Page Attribute is MA.DATA* *OR Memory Page Type is MT.SHARED AND S.HVRTOS_Process is one of two peers of the communication channel* *AND Memory Page Attribute is MA.DATA*

	> *OP.EXECUTE access to physical memory page is allowed for TOE Client if and only if:* *Execute_Policy.1* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.Virtual_Machine* *AND Memory Page Type is MT.PRIVATE* *AND Memory Attribute is MA.MEMORY* *OR* *Execute.Policy.2:* *OBJ.Physical_Memory_Page ownership is linked to the Subject ID* *AND subject is S.HVRTOS_Process* *AND Memory Page Type is MT.CALIBRATION AND Memory Page Attribute is MA.EXECUTE* In all other cases any operations are prohibited.
FDP_ACF.1.3/Memory	The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: > *None*
FDP_ACF.1.4/Memory	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: > *None*

Application Note:

TOE applies the following limitation on configuration of Memory Page Type and Memory Attributes:

1. Memory types cannot be combined.
2. There are only the following memory attributes: MA.READ, MA.WRITE, MA.EXECUTE, MA.MEMORY, MA.DATA. Other combinations are not allowed.
3. MT.SHARED memory is assigned only to two peer endpoints of the communications channel.
4. Each MT.CALIBRATION memory page is assigned as MA.READ to one and only TOE Client.
5. MT.PRIVATE memory can be assigned to S.Virtual_Machine as MA.MEMORY or MA.DATA.
6. MT.PRIVATE memory can be assigned to S.HVRTOS_Process as MA.DATA only.
7. For S.HVRTOS_Process, MT.CALIBRATION may contain both code and data.For MT.CALIBRATION memory pages that contain code, the memory is assigned to S.HVRTOS_Process as MA.EXECUTE, otherwise the memory is assigned to S.HVRTOS_Process as MA.READ.

Memory Page Types and Memory Attributes are determined by TOE during initialization and cannot be modified during runtime.

6.2.2 HW Resources Access Control Policies

TOE assigns hardware resources to TOE Clients to enable clients to perform their functionalities.

The hardware resources can be assigned directly to a TOE Client (pass-through resources) or provided by TOE via emulation. In the last case the TOE may control certain properties of the accessing resources or share the HW resource between several TOE Clients.

Hardware resources (*OBJ.HW_Resource*) include the following:

- MMIO regions
- Interrupts
- SMMU Stream IDs

Each resource is identified by its unique ID (for example, the interrupt ID, SMMU Stream ID or the physical address of the MMIO region).

The hardware resource assignment is either pre-configured in TOE or provided by TOE based on security attributes stored in PCT.

If a TOE Client is granted the access to hardware resources, it may perform an OP.Access operation, which enables the operation with the hardware resources.

By default, the TOE Client does not have access to hardware resources. To enable access, the TOE shall explicitly grant it.

6.2.2.1 FDP_ACC.2/HW_Resources Complete access control (HW resources access)

FDP_ACC.2.1/ HW_Resources	The TSF shall enforce the *HW Resources Access Control SFP* on:
	- *Subjects:*
	  - *TOE Client*
	- *Objects:*
	  - *OBJ.HW_Resource*
	and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2/ HW_Resources	The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

6.2.2.2 FDP_ACF.1/HW_Resources Security attribute based access control (HW resources access)

FDP_ACF.1.1/HW_Resources	The TSF shall enforce the *HW Resources Access Control SFP* to objects based on the following:
	- *Subjects:*
	  - *TOE Client*
	- *Objects:*
	  - *OBJ.HW_Resource*
	- *Operations:*
	  - *OP.Access*
	- *Security attributes*
	  - *Subject:*
	    - *TOE Client ID*
	    - *Allowed HW resources list (Stored in PCT or enforced by TOE itself)*

	• *Object:* ○ *HW resource ID*
FDP_ACF.1.2/HW_Resources	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: > *OP.Access for OBJ.HW_Resource is allowed to a subject if:* *a. The TOE Client Id to get the allowed HW resources from PCT (within PCD) matches the requested HW resource ID against the allowed ones in the PCT.* *b. The access is granted by TOE because the permission is hardcoded in TOE.* *Otherwise OP.Access is denied.*
FDP_ACF.1.3/HW_Resources	The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: > *None*
FDP_ACF.1.4/HW_Resources	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: > *None*

6.2.3 Communications

The communication channels are described in the Section 1.4.3.3. VS_SECURE_COMMUNICATIONS of the current document.

TOE meets the **O.Secure_Communications** security objective. To meet the security objective, TOE controls access to the shared memory regions used for communications to ensure that no other entity, except the two peer endpoints of the communication channel that are data producer and data consumer, have access to the communication channel. The respective access control policies are provided in 6.2.1. Memory access control policy.

6.2.4 Access to the services via communication channels

The TOE Logical Partitions inside and outside TOE as well as services provided by CCPLEX Secure World may provide services to TOE Clients.

The access to the services is performed via IVC Queues and Mempools.

TOE Client has access to services provided via a communication channel if and only if the TOE Client has a communication channel to the service. Some of the services are equally available for of TOE Clients, thus TOE always establishes a communication channel between the service and each TOE Client, though other services are available if the communication channel is explicitly defined in PCT between the TOE Client and the Service (OBJ.RSC - RSC stands for “Restricted Service provided via Communication channel”) that is associated with HV RTOS Process Data. PCT binds the TOE Client ID with RSC ID and specifies the Communication Channel ID. If the binding is present in PCT, the RSC is accessible for TOE Client.

6.2.4.1 FDP_ACC.1/RSC Subset access control (RSC)

FDP_ACC.1.1/RSC	The TSF shall enforce the *RSC SFP* on: > *Subjects:*

	• *TOE Client* > *Objects:* • *OBJ.RSC* > *Operations:* • *OP.Access*

6.2.4.2 FDP_ACF.1/RSC Security attribute based access control (RSC)

FDP_ACF.1.1/ RSC	The TSF shall enforce the *RSC SFP* to objects based on the following: > *Subjects:* • *TOE Client* > *Objects:* • *OBJ.RSC* > *Operations:* • *OP.Access* > *Security Attributes:* • *Subject* ○ *TOE Client ID* • *Objects* ○ *RSC ID* ○ *Communication Channel ID*
FDP_ACF.1.2/ RSC	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: > *TOE grants the TOE Client access to OBJ.RSC, if the definition of the communication channel with Communication Channel ID between TOE Client specified by its ID and the OBJ.RSC specified by RSC ID is present in PCT.*
FDP_ACF.1.3/ RSC	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: > *None*
FDP_ACF.1.4/RSC	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: > *None*

6.2.5 TOE core services Access Control Policies

The TOE provides services to TOE Clients available via System Calls (SVC, SMC, HVC) or via communication channels by discretion depending on security attributes assigned to TOE Clients. The security attributes are assigned to TOE Clients during initialization and cannot be updated in runtime.

SVC are used by HV RTOS Processes. HVC and SMC are used by Virtual Machines.

Each system call has its unique ID (which together with the type of the system call make up the TOE Service ID).

Capabilities are either pre-configured in TOE or specified in PCT.

6.2.5.1 FDP_ACC.1/TOE_Services Subset access control (TOE Services)

FDP_ACC.1.1/ TOE_Services	The TSF shall enforce the *TOE Services Access Control SFP* on: > *Subjects:* • *TOE Client* > *Objects:* • *OBJ.TOE_Service* > *Operations:* • *OP.Access*

6.2.5.2 FDP_ACF.1/TOE_Services Security attribute based access control (TOE Services)

FDP_ACF.1.1/ TOE_Services	The TSF shall enforce the *TOE Services Access Control SFP* to objects based on the following: > *Subjects:* • *TOE Client* > *Objects:* • *OBJ.TOE_Service* > *Operations:* • *OP.Access* > *Security Attributes:* • *Subject* ○ *Capability* • *Objects* ○ *TOE Service ID*
FDP_ACF.1.2/ TOE_Services	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: > *TOE grants TOE Clients access to OBJ.TOE_Service if the respective Capability (to allow access to the OBJ.TOE_Service with TOE Service ID) is assigned to the TOE Client either by TOE directly or by TOE using the data from PCT.*
FDP_ACF.1.3/ TOE_Services	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: > *None*
FDP_ACF.1.4/TOE_Services	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: > *None*

6.3 Security audit

6.3.1 FAU_ARP.1 Security alarms

FAU_ARP.1.1	The TSF shall take *the following list of actions in the table below* upon detection of a potential security violation:

Security Violation	***Action***
TOE Client accesses an unassigned memory or HW resource	*TOE blocks execution of the TOE Client on the violated CPU core and reports an error to TCF.*
TOE Client accesses an unauthorized service	*TOE reports an error to the TOE Client or reports an error to TCF.*

6.4 Protection of TSF

6.4.1 FPT_FLS.1 Failure with preservation of secure state

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: > *Failure during initialization: Implausible configuration data; Failure to initialize VMs or HV RTOS;* Fail to configure TOE. > *Spatial Isolation violation attempts* > *TOE control flow integrity violation* > *Implausible input submitted to TOE component* > *TOE Client violates access permission to a system call interface provided by the TOE* > *TOE Client attempts to execute data*

Application Note: The potential security violations are explicitly defined in the FAU_ARP.1 requirement. There is no audited event defined in the SFR of this Security Target.

Chapter 7. Security Assurance Requirements

This Security Target claims conformance to EAL4, augmented with ALC_FLR.1. This assurance level was chosen to ensure the following:

- The TOE has a moderate level of assurance in enforcing its security functions when instantiated in its intended environment, which imposes no restrictions on assumed activity on applicable networks.
- Any remaining security flaws in the TOE that are brought to the notice of the Developer will be remediated.

The requirements are summarized in the following table:

Table 13. EAL4 requirements description augmented with ALC_FLR.1

Assurance Class	Component	Component Title
ADV: Development	ADV_TDS.3	Basic modular design
	ADV_ARC.1	Security architecture description
	ADV_FSP.4	Complete functional specification
	ADV_IMP.1	Implementation representation of the TSF
AGD: Guidance documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative procedures
ALC:_ Life-cycle support	ALC_CMC.4	Production support, acceptance procedures and automation
	ALC_CMS.4	Problem tracking CM coverage
	ALC_DEL.1	Delivery procedures
	ALC_DVS.1	Identification of security measures
	ALC_LCD.1	Developer-defined life-cycle model
	ALC_TAT.1	Well-defined development tools
	ALC_FLR.1	Basic flaw remediation
ASE: Security Target evaluation	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	Security Target introduction

Assurance Class	Component	Component Title
	ASE_OBJ.2	Security objectives
	ASE_REQ.2	Derived security requirements
	ASE_SPD.1	Security problem definition
	ASE_TSS.1	TOE summary specification
ATE: Tests	ATE_COV.2	Analysis of coverage
	ATE_DPT.1	Testing: basic design
	ATE_FUN.1	Functional tests
	ATE_IND.2	Independent testing
AVA: Vulnerability assessment	AVA_VAN.3	Focused vulnerability analysis

Chapter 8. Rationales

8.1 Security Functional Requirements Rationale

Table 14. Mapping between TOE security objectives and SFRs

Functional Security Requirements	O.Spatial_Isolation	O.Access_Control	O.Identification	O.Secure_Communications
FIA_UID.2	X	X	X	X
FIA_USB.1	X	X	X	X
FIA_ATD.1	X	X	X	X
FDP_ACC.2/Memory	X	X		X
FDP_ACF.1/Memory	X	X		X
FDP_ACC.2/HW_Resources	X	X		
FDP_ACF.1/HW_Resources	X	X		
FDP_ACC.1/RSC		X		
FDP_ACC.1/RSC		X		
FDP_ACC.1/TOE_Services		X		
FDP_ACF.1/TOE_Services		X		
FAU_ARP.1	X	X	X	X

Functional Security Requirements	O.Spatial_Isolation	O.Access_Control	O.Identification	O.Secure_Communications
FPT_FLS.1	X	X	X	X

8.1.1 O.Spatial_Isolation

The TOE shall enforce measures to protect the private resources that belong to TOE, VMs, or HV RTOS processes outside TOE from unauthorized modifications or eavesdropping.

Every TOE Client runs in its own virtual address space. The identification of the TOE Clients is performed according to FIA_UID.2, FIA_USB.1 and FIA_ATD.1.

TOE allocates private memory (MT.PRIVATE) for TOE Clients and this guarantees that content of the private memory cannot be read or modified by other TOE Clients.

Access control policies regarding the memory operations are defined in FDP_ACC.2/Memory and FDP_ACF.1/Memory. Besides the private data in memory the TOE protects, Hardware Resources are assigned to TOE Clients (via FDP_ACC.2/HW_Resources and FDP_ACF.1/HW_Resources). The list of private resources (Memory, HW Resources) is complete, so no other access controls are needed.

The security properties shall be preserved under any conditions (as specified by FPT_FLS.1). Attempts to violate the spatial isolation are detected and processed as mandated by FAU_ARP.1.

8.1.2 O.Access_Control

The TOE shall prevent the TOE Client from accessing the data, services provided by the TOE, communication channels, and hardware resources, unless they are explicitly assigned to the TOE Client.

Access control uses identification data as it is specified by FIA_UID.2.1 and security attributes as specified in FIA_UID.2, FIA_USB.1 and FIA_ATD.1.

The access control policies that manage access to data, services provided by TOE, communication channels and hardware resources, are defined in the following requirements:

- FDP_ACC.2/Memory
- FDP_ACF.1/Memory
- FDP_ACC.2/HW_Resources
- FDP_ACF.1/HW_Resources
- FDP_ACC.1/RSC
- FDP_ACF.1/RSC

- FDP_ACC.1/TOE_Services
- FDP_ACF.1/TOE_Services

The security properties shall be preserved under any conditions (as specified by FPT_FLS.1). Attempts to violate the spatial isolation are detected in a timely way and processed as mandated by FAU_ARP.1.

8.1.3 O.Identification

The TOE Logical Partition identification mechanism is based on security attributes assigned to the user of TOE and the subject acting on behalf of the user as defined in

- FIA_UID.2
- FIA_USB.1
- FIA_ATD.1

The security properties shall be preserved under any conditions (as specified by FPT_FLS.1). Attempts to violate the spatial isolation are detected in a timely way and processed as mandated by FAU_ARP.1

8.1.4 O.Secure_Communications

The communication channels between the TOE components, between the components and operational environment and between TOE and VMs and HV RTOS Processes outside the TOE use identification information specified in the user security attributes and the security attributes of the subjects acting on behalf of users as specified in:

- FIA_UID.2
- FIA_USB.1
- FIA_ATD.1

TOE uses the following communication channels:

- IVC Queues
- Mempools
- Kernel calls to get access to the services provided by Hypervisor Kernel and VM Sidekick.

IVC Queues and Mempools use shared memory regions to perform data exchange.

TOE guarantees the absence of modifications of the shared memory by meeting the requirements, given below:

- FDP_ACC.2/Memory
- FDP_ACF.1/Memory

Kernel calls use General Purpose Registers to pass data along with shared memory regions established between VM Sidekick and Hypervisor Kernel, and other interacting parties.

The protection of the shared memory regions is guaranteed by applying the memory access control as defined in

- FDP_ACC.2/Memory

The security properties shall be preserved under every condition (as specified by FPT_FLS.1). Attempts to violate the spatial isolation are detected and processed as mandated by FAU_ARP.1.

8.2 SFR Dependencies

Table 15. SFR dependencies

SFRs	CC Dependencies	Satisfied Dependencies
FIA_UID.2	No Dependencies	No Dependencies
FIA_USB.1	FIA_ATD.1	FIA_ATD.1
FIA_ATD.1	No Dependencies	No Dependencies
FDP_ACC.2/Memory	FDP_ACF.1	FDP_ACF.1/Memory
FDP_ACF.1/Memory	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2/Memory FMT_MSA.3 discarded *See justification below
FDP_ACC.2/HW_Resources	FDP_ACF.1	FDP_ACF.1/HW_Resources
FDP_ACF.1/HW_Resources	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2/HW_Resources FMT_MSA.3 discarded *See justification below
FDP_ACC.1/RSC	FDP_ACF.1	FDP_ACF.1/RSC
FDP_ACF.1/RSC	FDP_ACC.1	FDP_ACC.1/RSC FMT_MSA.3 discarded *See justification below
FDP_ACC.1/TOE_Services	FDP_ACF.1	FDP_ACF.1/TOE_Services
FDP_ACF.1/TOE_Services	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1/TOE_Services FMT_MSA.3 discarded *See justification below
FAU_ARP.1	FAU_SAA.1	FAU_SAA.1 discarded *See justification below
FPT_FLS.1	No Dependencies	No Dependencies

The dependency FMT_MSA.3 of FDP_ACF.1/* is discarded. There is no management of security attributes by authorized users for this information flow control SFP as all security attributes are exclusively managed by the TSF, therefore the dependency FMT_MSA.3 is not applicable.

The dependency FAU_SAA.1 of FAU_ARP.1 is discarded. The potential security violations are explicitly defined in the FAU_ARP.1 requirement. There is no audited event defined in the SFR of this Security Target.

8.3 Security Objectives Rationale

8.3.1 Threats, Assumptions and OSPs to Security Objectives Mapping

Table 16. Threats and Assumptions to Security Objectives Mapping

Threats, Assumptions and OSPs **Objectives**	**T.Abuse_Functionality**	**T.Data_Leakage**	**T.Unauthorized_Access**	**T.Remote Code_Execution**	**A.Secure_Initialization**	**A.Hardware_Platform**	**A.OE_Security**	**A.No_Internal_Attacker**	**P.System_Integrator**	**P.Secure_Development**
O.Spatial_Isolation	X	X	X	X						X
O.Access_Control	X	X	X	X					X	X
O.Identification	X	X	X						X	X
O.Secure_Communications	X	X	X							X
OE.Secure_Initialization	X	X	X	X	X				X	
OE.Hardware_Platform	X	X	X	X		X			X	
OE.Secure_World_Correctness							X		X	X
OE.TCF_Correctness	X	X	X				X		X	X
OE.Trustworthy_Personnel								X		

8.3.2 Threats to security objectives rationale

Table 17. Threats to security objectives rationale

Threat	Rationale
T.Abuse_Functionality	**O.Spatial_Isolation** confines private data inside TOE and TOE Clients that make unauthorized access impossible, so the request shall meet **O.Access_Control** security objective. **O.Access_Control** security objective makes sure that the TOE Client gets access only to the resources and functionalities if it is explicitly allowed to access, thus the TOE Client is not able to produce impact on data integrity and the TOE execution state. **O.Identification** is required for proper functioning of access control mechanisms. If identification is spoofed or tampered with, it may result in elevation of privilege and facilitate the threat. **O.Secure_Communications** security objective guarantees that a malicious TOE Client cannot affect TOE and other clients by spoofing and/or tampering with the data passing through communication channels. **OE.Secure_Initialization** ensures that TOE is properly configured, including correct specification of identification data and correctly configured access control policies. **OE.Hardware_Platform** guarantees that hardware-based access control mechanisms, including MMU, SMMU and MSSNVLINK protection, work properly and malicious TOE Client cannot use hardware vulnerabilities to impact operations either of TOE or other TOE Clients **OE.TCF_Correctness** ensures that malicious TOE Clients cannot affect the operation of either a TOE or other TOE Clients via a vulnerability inside TCF.
T.Data_Leakage	**O.Spatial_Isolation** confines private data inside the TOE and TOE Clients that make unauthorized access impossible.. **O.Access_Control** security objective makes sure that the TOE Client only gets access to the resources and functionalities it is explicitly allowed to access, thus the TOE Client is not able to access data that it is not allowed to access. **O.Identification** is required for proper functioning of access control mechanisms. If identification is spoofed or tampered with, it may result in elevation of privilege and result in access to data that the TOE prohibits access to. **O.Secure_Communications** security objective guarantees that a malicious TOE Client cannot access data passing through communication channels. **OE.Secure_Initialization** ensures that the TOE is properly configured, including correct specification of identification data and correctly configured access control policies. Incorrectly set access control policies may result in elevation of privileges and access to the data that it is not permitted to access. **OE.Hardware_Platform** guarantees that hardware-based access control mechanisms, including MMU, SMMU and MSSNVLINK protection, work properly and malicious TOE Clients cannot use hardware vulnerabilities to get unauthorized access to data. **OE.TCF_Correctness** ensures that malicious TOE Clients cannot use a vulnerability inside TCF to get unauthorized access to data.

Threat	Rationale
T.Unauthorised_Access	**O.Spatial_Isolation** prevents bypassing access control mechanisms **O.Access_Control** ensures that the TOE Client has access only to the resources and services that it is permitted to access in accordance with security policies. **O.Identification** enables correct operation of access control mechanisms. By tampering with or spoofing of identification, a malicious TOE Client may elevate its privileges and overcome the access control mechanisms. **O.Secure_Communications** guarantees that a malicious TOE Client is not able to tamper with or spoof the communication protocols and get access to the resources or services that it is not permitted to access. **OE.Secure_Initialization** ensures that the TOE is properly configured, including correct specification of identification data and correctly configured access control policies. **OE.Hardware_Platform** guarantees that hardware-based access control mechanisms, including MMU, SMMU and MSSNVLINK protection, work properly and a malicious TOE Client cannot use hardware vulnerabilities to impact operations either of TOE or other TOE Clients **OE.TCF_Correctness** ensures that malicious TOE Clients cannot bypass access control mechanisms via a vulnerability inside TCF.
T.Remote_Code_Execution	**O.Spatial_Isolation** restricts the possibility of injecting and executing code remotely, bypassing memory access control security policies that grant execution capabilities to read-only memory regions. **O.Access_Control** - restricts capabilities of TOE Clients to perform a Remote Code Execution attack (for example, via Return Oriented Programming techniques). **OE.Secure_Initialization** ensures that the TOE is properly configured, including correct specification of identification data and correctly configured access control policies. **OE.Hardware_Platform** guarantees that hardware-based access control mechanisms, including MMU, SMMU and MSSNVLINK protection, work properly and malicious TOE Clients cannot use hardware vulnerabilities to impact operations either of TOE or other TOE Clients

8.3.3 OSPs vs Objectives for the TOE and Objectives for the Environment Mapping

Table 18. Organizational Security Policies rationale

OSP	Rationale
P.Secure_Development	**P.Secure_Development** supports **A.No_Internal_Attacker** assumption. A potential internal attacker, if not restricted, may embed a vulnerability (or backdoor) in TOE code that may affect the following security objectives: **O.Spatial_Isolation** - this security objective is potentially affected because the respective memory access control policies may be modified or even disabled (for example, read-only code sections)

OSP	Rationale
	O.Access_Control - this security objective is potentially affected if there is an intended vulnerability or backdoor embedded by an internal attacker. By using a backdoor or vulnerability, the malicious TOE Client may elevate its privileges. **O.Identification** - this security objective is potentially affected if there is an intended vulnerability or backdoor embedded by an internal attacker. Using a backdoor or vulnerability, the malicious TOE Client may spoof or tamper with the identification information. **O.Secure_Communications** - this security objective is potentially affected if there is an intended vulnerability or backdoor embedded by an internal attacker. Using a backdoor or vulnerability, the malicious TOE Client may get unauthorised access to the data passing through communication channels to read or modify it. **OE.Secure_World_Correctness** - the attacker may embed a vulnerability or backdoor into CCPLEX Secure World software that may affect TOE security properties. **OE.TCF_Correctness** - the attacker may embed a vulnerability or backdoor into TCF software that may affect TOE security properties.
P.System_Integrator	**P.System_Integrator** supports **A.No_Internal_Attacker** assumption. Unqualified or malicious integration may affect the following security objectives: **O.Access_Control** policies are configured by PCD and stored in PCT, so that it can be corrupted during integration. **O.Identification** - Identification information is stored in PCT and may be set incorrectly, affecting the identification of TOE Clients. **OE.Secure_Initialization** - the entire TOE may be spoofed during integration, which may affect its security properties. **OE.Hardware_Platform** - if TOE is used on a hardware platform which is incompatible with the TOE, it may affect security properties of the TOE (for example, O.Spatial_Isolation). **OE.Secure_World_Correctness** - incompatible CCPLEX Secure World software may impact TOE security properties. **OE.TCF_Correctness** - incompatible TCF software may impact TOE security properties.

8.3.4 Assumptions to security objectives rationale

Table 19. Assumptions to security objectives rationale

Assumption	Rationale
A.Secure_initialization	**OE.Secure_Initialization** security objective for TOE environment corresponds to **A.Secure_initialization** assumption. If the assumption is not met, the TOE security objectives are impacted: > **O.Spatial_Isolation** > **O.Access_Control** > **O.Identification** > **O.Secure_Communications**

Assumption	Rationale
	The impact occurs because it is impossible to guarantee the integrity and authenticity of TOE software as well as calibration (configuration) data.
A.Hardware_Platform	**OE.Hardware_Platform** security objective for TOE environment corresponds to **A.Hardware_Platform** assumption. If the assumption is not met, the TOE security objectives are impacted: > **O.Spatial_Isolation**
A.OE_Security	**OE.Secure_World_Correctness** and **OE.TCF_Correctness** security objectives for TOE environment correspond to **A.OE_Security.** If the assumption is not met, the TOE security objectives are impacted: > **O.Spatial_Isolation** > **O.Access_Control** > **O.Identification** > **O.Secure_Communications** > **OE.Secure_Initialization** Incorrect operation of TOE environment may hit every aspect of TOE operation
A.No_Internal_Attacker	**OE.Trustworthy_Personnel** security objective directly upholds the **A.No_Internal_Attacker** assumption. If the assumption is not met, the TOE security objectives are impacted: > **O.Spatial_Isolation** > **O.Access_Control** > **O.Identification** > **O.Secure_Communications**

Chapter 9. TOE Summary Specification

9.1 Implementation of Security Functional Requirements

TOE meets the following Security Functional Requirements:

- FIA_UID.2
- FIA_USB.1
- FIA_ATD.1
- FDP_ACC.2/Memory
- FDP_ACF.1/Memory
- FDP_ACC.2/HW_Resources
- FDP_ACF.1/HW_Resources
- FDP_ACC.1/RSC
- FDP_ACC.1/RSC
- FDP_ACC.1/TOE_Services
- FDP_ACF.1/TOE_Services
- FAU_ARP.1
- FPT_FLS.1

9.1.1 FIA_ATD.1 and FIA_UID.2

For every TOE Logical Partition, the Hypervisor Kernel stores the TOE Logical Partition's context, including:

- Unique identification information of the TOE Logical Partition
- Security attributes specified for the logical partition
- Mapping between TOE Logical Partitions to IVC Queues and Mempools
- Memory Translation Tables that contain lists of all physical memory pages assigned to the TOE Client (including memory and MMIO resources)

TOE statically assigns IVC Queues and Mempools to the TOE Logical Partition, so that the TOE Client always identifies its communication peers.

The TOE supports multitasking, and every time the execution context is switched on a given CPU Core TOE has access to the context of the currently executing TOE Logical Partition.

For asynchronous events, the TOE can identify the physical memory address of the access or Stream ID and, therefore, identify the TOE Client that caused the asynchronous event.

The TOE maintains the list of TOE Logical Partitions, so it can get access to the TOE Logical Partition context by its identification information at any time to perform a requested operation.

The TOE Client is always identified before TOE applies access control SFP to any TSF-mediated actions on behalf of that user.

9.1.2 FIA_USB.1

The TOE supports the following security attributes specified for the subject that acts on behalf of the TOE user:

- ID: Unique ID assigned to the TOE Client
- VM attributes (only for VMs: Server VM, Privileged VM, System Halt Capability attribute)
- HV RTOS Process attributes (Error Reporting Capability attribute, System State Transition Initiation Capability attribute)
- Physical memory page ownership
- The set of IVC Queues assigned to the TOE Client
- The set of Mempools assigned to the TOE Client

The Unique ID assigned to the TOE Client is specified in the PCT table. The TOE parses PCT and for every TOE Client defined in PCT, creates the client with the ID and other properties specified for this TOE Client including:

- VM attributes (only for VMs: Server VM, Privileged VM, System Halt Capability attribute)
- HV RTOS Process attributes (Error Reporting Capability attribute, System State Transition Initiation Capability attribute)
- The set of IVC Queues assigned to the TOE Client
- The set of Mempools assigned to the TOE Client

In addition, the information provided in the PCT TOE may create the IVC Queues between the TOE Client and the TOE to support coordination between TOE Clients.

Physical memory pages are assigned to the client during initialization. The TOE initialization algorithm creates TOE Clients and allocates physical memory pages to the TOE Client. The configuration of memory pages allocated to the client is based on data provided in PCT. The TOE performs mapping between TOE virtual memory pages and allocated memory pages as prescribed in FDP_ACC.2/Memory and FDP_ACF.1/Memory.

The TOE Client that accesses the virtual memory address actually accesses the physical address with certain restrictions specified in the mapping attributes (e.g. read-only access).

9.1.3 FDP_ACC.2/Memory and FDP_ACF.1/Memory

FDP_ACC.2/Memory and FDP_ACF.1/Memory SFRs implement spatial isolation between TOE Logical Partitions.

The SFPs defined in these SFRs are enforced by different hardware mechanisms

- MMU
- SMMU
- MSSNVLINK Protection

Under the assumption A.OE_Security, these mechanisms guarantee the provision of spatial isolation.

9.1.4 FDP_ACC.2/HW_Resources and FDP_ACF.1/HW_Resources

The hardware resources include passthrough resources and emulated resources.

For the passthrough resources, the access control SFPs defined in FDP_ACC.2/HW_Resources and FDP_ACF.1/HW_Resources are implemented using hardware assistance in the same way as in 9.1.3.

For emulated resources, the TOE provides virtual MMIO regions and intercepts the access to this MMIO region. These regions are not mapped to any physical memory, hence when TOE Client accesses this memory, the access is trapped and processed by TOE according to the emulation algorithm.

Then the TOE algorithm processes the request accordingly.

For the resources with limited access (controlled via access control SFPs), the TOE provides access to the resources via configuring MMU translation tables, SMMU translation tables, and MSSNVLINK protection tables based on the information provided in PCT.

9.1.5 FDP_ACC.1/RSC and FDP_ACF.1/RSC

Restricted Services provided via communications used in a situation when the TOE Client uses services provided by other TOE Logical Partitions or TOE environment.

These services use communication channels (IVC Queues or Mempools) created on top of shared memory regions.

The TOE allocates the shared memory regions and assigns them to the communicating agents. It ensures that only 2 endpoints have access to this shared memory region - the ends of the peer-to-peer communication channel.

Thus, the TOE Client may access the service provided via the communication channel if the TOE Client is an endpoint of this communication channel and TOE assigned the respective shared memory region to the TOE Client. The assignment of shared memory to the TOE Client is performed by TOE based on the information stored in PCT.

9.1.6 FDP_ACC.1/TOE_Services and FDP_ACF.1/TOE_Services

Core TOE Services are provided to TOE Clients by the Hypervisor Kernel, VM Sidekick (for the respective VM), or by another TOE Logical Partition belonging to the TOE.

In the case where the service is provided by VM Sidekick, the request is first trapped by the Hypervisor Kernel, which redirects it to VM Sidekick. This approach allows keeping a minimal attack surface and a code base for the microkernel as well as get other benefits to safety and security of TOE.

In both cases, the request results in a hardware synchronous exception, which is trapped by the Hypervisor Kernel. The fact the exception is trapped synchronously allows using the context of currently executing the TOE Logical Partition to retrieve the security attributes associated with it.

Based on these security attributes and the type of the request, the Hypervisor kernel decides to do one of the following:

- Accept the request and provide the service
- Reject the request
- Redirect the request to VM Sidekick

If redirected, the VM Sidekick, based on the TOE Client's security attributes shared by Hypervisor Kernel, decides to do one of the following:

- Accept the request and provide the service
- Reject the request

The security attributes are assigned to the TOE Client as defined in Section 6.1. Identification and authentication.

In cases when a TOE Service is provided by the TOE Logical Partition by discretion, the TOE Logical Partition reads security attributes provided in the calibration data for this service, identifies the TOE client and makes a verdict if the access is allowed. Then the TOE Logical Partition enforces it.

Note: For the TOE case, the security policy decision is made during the integration of the TOE and recorder in calibration data, thus TOE only enforces it.

9.1.7 FAU_ARP.1

The violation of access control policies results in exceptions that must be processed in the same way as specified in 9.1.8. Then either the Hypervisor Kernel or VM Sidekick analyses the exception context and make one of the following decisions:

- Stop execution of the TOE Client on the current CPU core
- Report the error to the TOE Client.

When the TOE halts execution on the current CPU core, it also sends an error report to the TCF. Based on this report, the TCF can restart the system to restore the functionalities that became unavailable due to the halted core. In this case, the previous state is not preserved, which prevents any impact on the system's operation caused by the error.

9.1.8 FPT_FLS.1

TOE implements a set of self-protecting mechanisms to detect security critical failures quickly and undertake measures to preserve the TOE security state.

The following failures are considered:

1. Failure during initialization: Implausible configuration data; Fail to initialize VMs or HV RTOS Processes; Fail to configure TOE.

   TOE initialization is single threaded. During the initialization, the TOE controls the status of operations; implements plausibility checks and if it detects unrecoverable failure, the TOE stops the execution before the threat may become active.

2. Spatial isolation violation attempts

   The attempts to bypass hardware-based mechanisms that enforce Spatial Isolation result in synchronous exceptions that are trapped by Hypervisor Kernel in the same way as specified in 9.1.7, or asynchronous exceptions where the culprit is identified. Then algorithms implemented in the Hypervisor Kernel or VM Sidekick evaluate risks to TOE security, and if the risk is present, stop the execution of the TOE Client that causes the violation on the CPU core where the violation was detected.

3. TOE control flow integrity is guaranteed by a set of the following measures:

> FDP_ACC.2/Memory and FDP_ACF.1/Memory that enforce read-only instruction memory and data execution prevention.

> Pointer Authentication Code

  The Hypervisor Kernel implements Pointer Authentication Code (PAC) for itself, HV RTOS Processes, and VM Sidekicks to protect them from Return-Oriented-Programming (ROP) attacks. PAC uses keys generated from a trusted RNG source provided by TCF to sign the return addresses at function entry and verify them at function exit to ensure they were not tampered during the execution of the function, thereby preserving the control flow integrity (CFI).

> Stack Canaries

  The Hypervisor Kernel implements Stack Canaries for itself, HV RTOS Processes, and VM Sidekicks by providing a stack canary per instance to protect against CFI attacks resulting from the corruption of return address as a result of smashing the function stack frame.

> Kernel Stack Overflow Protection

  The Hypervisor Kernel Stacks used for initialization and runtime are protected from overflow by implementing a detection mechanism where one guard page of memory is left unmapped before and after CPU stack allocation. When a program results in an unintended overflow of the allocated stack, the kernel shall detect the unintended access to the unmapped guard page.

  When the TOE detects CFI violation, it stops the execution of the TOE Client or the TOE itself before it hits TOE security.

4. Implausible input submitted to a TOE component

   TOE performs plausibility checks on input data to ensure that it does not affect the proper operation of the requested functionalities. If TOE detects a violation, it reacts in accordance with FAU_ARP.1.

5. TOE Client violates access permission to a system call interface provided by the TOE

   The behaviour is described in 9.1.9. FAU_ARP.1.

6. TOE Client attempts to execute data

   TOE implements the following measures:

- FDP_ACC.2/Memory and FDP_ACF.1/Memory that enforce read-only instruction memory and data execution prevention.
- Pointer Authentication Code.

  The Hypervisor Kernel implements Pointer Authentication Code (PAC) for itself, HV RTOS Processes and VM Sidekicks to protect them from Return-Oriented-Programming (ROP) attacks. PAC uses keys generated from a trusted RNG source provided by TCF to sign the return addresses at function entry and verify them at function exit to ensure they were not tampered with during the execution of the function, thereby preserving the control flow integrity (CFI).

9.2 Implementation of TOE security services

TOE provides the following security services as specified in Section 1.4.3. TOE Logical Scope:

- VS_SPATIAL_ISOLATION
- VS_IDENTIFICATION
- VS_SECURE_COMMUNICATIONS
- VS_ACCESS_CONTROL

9.2.1 VS_SPATIAL_ISOLATION

TOE creates spatially isolated logical partitions to meet O.Spatial_Isolation security objective. Private resources of TOE logical partitions are not accessible by other TOE Logical Partitions. Private resources include memory and hardware resources (including virtualized hardware resources). Other TOE Clients shall not be able to affect the private resources of the TOE Logical Partitions except the legitimate operation performed via the interfaces exposed by the TOE Logical Partition. Each logical partition has its unique ID (the information that allows unambiguously identifying the TOE Logical Partition).

The respective SFRs are as follows:

- FIA_USB.1
- FIA_UID.2
- FIA_ATD.1
- FDP_ACC.2/Memory
- FDP_ACF.1/Memory
- FDP_ACC.2/HW_Resources
- FDP_ACF.1/HW_Resources

9.2.2 VS_IDENTIFICATION

Each logical partition has its unique ID (the information that allows unambiguously identifying the TOE Logical Partition). This information is defined according to the following:

- FIA_USB.1
- FIA_UID.2

- FIA_ATD.1

9.2.3 VS_SECURE_COMMUNICATIONS

Unauthorised access to the data passing through communication channels is prohibited as specified by the following SFRs:

- FDP_ACC.2/Memory
- FDP_ACF.1/Memory

with the support of the following SFRs to facilitate correct identification of TOE Logical Partitions:

- FIA_USB.1
- FIA_UID.2
- FIA_ATD.1

9.2.4 VS_ACCESS_CONTROL

Access control TSF is supported by the following SFRs:

- FDP_ACC.2/Memory
- FDP_ACF.1/Memory
- FDP_ACC.2/HW_Resources
- FDP_ACF.1/HW_Resources
- FDP_ACC.1/RSC
- FDP_ACF.1/RSC
- FDP_ACC.1/TOE_Services
- FDP_ACF.1/TOE_Services

9.2.5 VS_SECURE_STATE

To ensure that TOE preserves its secure state and properly reacts the violation of SFPs the following requirements shall be met:

- FAU_ARP.1
- FPT_FLS.1

Chapter 10. Abbreviations and glossary

Table 20. Glossary and abbreviations

AES	Advanced Encryption Standard
Calibration data	Calibration data provides parameters that define TOE behavior and is applied in runtime. It is different from configuration data that also specifies TOE behavior, but the configuration data is applied in compile/build time. For TOE, the difference between calibration and configuration data is not important so the terms "calibration data" and "configuration data" may be used interchangeably.
CC	Common Criteria - the cybersecurity standard
CCPLEX	CCPLEX - CPU complex. The part of Orin SoC that includes all ARMv8 CPU cores.
CCPLEX Non-Secure World	The functionalities running on CCPLEX in Arm CPU Non-Secure EL2/EL1/EL0 security state. Security state is a concept-defined ARMv8 architecture.
CCPLEX Secure World	The functionalities running on CCPLEX in Arm CPU Secure EL1/EL0 security state. Security state is a concept-defined ARMv8 architecture.
CPU Core	Atomic processing unit compliant to ARMv8 specification.
Data Abort	ARM architecture.
Device Tree	Hierarchical data structure that describes hardware configuration for its user. Device Tree is used for the TOE component as calibration data.
DTB	Device Tree Binary. A binary format that represents Device Tree data structure.
DMA	Direct memory access (DMA) is a feature of computer systems that allows certain hardware subsystems to access main system memory independently of the CCPLEX.
Drive Update VM	The component of DRIVE OS - TOE Client that is responsible for downloading software updates for TOE and its environment.
EAL	An Evaluation Assurance Level (EAL) is a category ranking assigned to an IT product or system after a Common Criteria security evaluation.
EL[0..2]	See Exception Levels.
Exception Levels	Exception Level, a concept defined for ARMv8 architecture.
FSI	Functional Safety Island - Hardware unit in Orin SoC.
GOP	Goto Oriented Programming - hacking technique that allows use of arbitrary data as a program to implement Remote Code Execution attack.

GPR	General Purpose Register. Registers X0..X31 in ARMv8 architecture.
GPU	Graphics Processing Unit - high performance processor in Orin SoC.
HV RTOS Processes	Process created and managed by Hypervisor Kernel.
HVC	HVC - a dedicated instruction to implement hypercalls (exceptions to EL2). HVC is a part of ARMv8 architecture.
HW	Hardware.
Hypervisor Kernel	Nvidia proprietary microkernel that supports execution of HV RTOS Processes and Virtual Machines.
IVC library	Software component that implements communication protocol over shared memory region to provide IVC Queue communication channel.
IVC Mempool	The region of shared memory used for communications between TOE Logical Partitions. This type of communication does not use any notification mechanisms.
IVC Queue	The communication channel that uses shared memory and the IVC library to implement the communication protocol. IVC Queues support notifications.
MMIO	Memory Mapped Input/Output. Hardware or emulated interface intended for interaction with hardware.
MMU	Memory Management Unit.
MSSNVLINK	The NVLink (NVIDIA High Performance Data Transfer Technology) interface between the MSS (Memory SubSystem) and an external GPU.
PAC	Pointer Authentication Code
PCD	Platform Configuration Data, TOE element that stores calibration data for NVIDIA DRIVE OS. Calibration data is represented in a form of a collection of binary blobs in including: PCT Device Tree
PCT	Platform Configuration Table. The binary blob provided by PCD that stores calibration data for TOE.
Physical Memory Page	A fixed-size block of physical RAM used in memory management.
PSCI	Arm Power State Coordination Interface.
ROP	Return Oriented Programming -- a hacking technique that allows use of arbitrary data as a program to implement Remote Code Execution attack.
RTOS	Real-Time Operating System.
Safety	The term “safety” refers to the one defined by ISO 26262 – Road Vehicles – Functional Safety
SFP	Security Function Policy.
SFR	Security Functional Requirement.
SHA256	256-bit Secure Hash Algorithm.

Shared memory	Physical memory which is common for two or more Virtual Address Spaces.
SMC	SMC - a dedicated instruction to implement Secure Monitor Calls (exceptions to EL3). SMC is a part of ARMv8 architecture.
SMCCC	Secure Monitor Calling Convention.
SMMU	System Memory Management Unit.
SMP	Symmetric Multi-Processing.
Spatial Isolation	Two or more entities are spatially isolated if these entities have private resources that cannot be read or modified from other entities.
ST	Security Target.
TA	Trusted Application.
TCF	Tegra Core Firmware.
TOE	Target Of Evaluation.
TOE Client	TOE Logical Partition that is not a part of TOE.
TOE Logical Partition	Virtual Machine or HV RTOS process running under governance of Hypervisor Kernel.
TOE User	Software running inside TOE Client.
Trusted Application	The application running in CCPLEX Secure World.
TrustZone	TrustZone is ARM CPU-specific technology that introduces hardware support to reliably isolate security-related functionalities. This technology utilizes a concept of Secure (trusted) and Non-Secure (non-trusted) worlds that extends beyond the processor to encompass memory, software, bus transactions, interrupts, and peripherals within an SoC.
TSF	TOE Security Functionality.
Virtual Memory Page	A fixed-size unit within a process's virtual address space, linked to a physical memory page by the operating system.
VM	Virtual Machine.
VM Sidekick	Virtual Machine Sidekick.

Chapter 11. References

Table 21. References

[CC:2022-P1]	CCMB-2022-11-001 Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and General Model, CC:2022, Revision 1, November 2022.
[CC:2022-P2]	CCMB-2022-11-002 Common Criteria for Information Technology Security Evaluation. Part 2: Security functional components, CC:2022, Revision 1, November 2022.
[CC:2022-P3]	CCMB-2022-11-003 Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components, CC:2022, Revision 1, November 2022.
[CC:2022-P4]	CCMB-2022-11-004 Common Criteria for Information Technology Security Evaluation. Part 4: Framework for the specification of evaluation methods and activities, CC:2022, Revision 1, November 2022.
[CC:2022-P5]	CCMB-2022-11-005 Common Criteria for Information Technology Security Evaluation. Part 5: Pre-defined packages of security requirements, CC:2022, Revision 1, November 2022.
[CC:2022-Errata1]	Errata and Interpretation for CC:2022 (Release 1) and CEM:2022 (Release 1), 2024-02-01

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation (“NVIDIA”) makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer (“Terms of Sale”). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer’s own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA’s aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA, the NVIDIA logo, are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

Arm

Arm, AMBA, and ARM Powered are registered trademarks of Arm Limited. Cortex, MPCore, and Mali are trademarks of Arm Limited. All other brands or product names are the property of their respective holders. ”Arm” is used to represent ARM Holdings plc; its operating company Arm Limited; and the regional subsidiaries Arm Inc.; Arm KK; Arm Korea Limited.; Arm Taiwan Limited; Arm France SAS; Arm Consulting (Shanghai) Co. Ltd.; Arm Germany GmbH; Arm Embedded Technologies Pvt. Ltd.; Arm Norway, AS, and Arm Sweden AB.

OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

Copyright

© 2024 NVIDIA Corporation. All rights reserved.

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
http://www.nvidia.com