GENERAL DYNAMICS
Mission Systems

General Dynamics Mission Systems

GD SecureComm Crypto

FIPS 140-3 Non-Proprietary Security Policy

Document Version: 1.2
August 10th, 2026

General Dynamics Mission Systems

8201 E McDowell Rd
Scottsdale, AZ
85257
USA
www.gdmissionsystems.com

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document defines the Non-Proprietary Security Policy for the GD SecureComm Crypto, hereafter referred to as the Module. The Module conforms to FIPS 140-3 overall Level 1 requirements, with individual security levels detailed in Section 1.2. In accordance with AS02.05 and ISO/IEC 19790:2012 §7.7, Physical Security is optional and not applicable to the Module.

1.2 Security Levels

The GD SecureComm Crypto meets the overall requirements for Level 1 security under FIPS 140-3. The applicable security categories and their corresponding levels are detailed below:

Section	Title	Security Level
1	General	1
2	Cryptographic module specification	1
3	Cryptographic module interfaces	1
4	Roles, services, and authentication	1
5	Software/Firmware security	2
6	Operational environment	1
7	Physical security	N/A
8	Non-invasive security	N/A
9	Sensitive security parameter management	2
10	Self-tests	1
11	Life-cycle assurance	2
12	Mitigation of other attacks	N/A
	Overall Level	1

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The module is a standalone software cryptographic module implemented as a C language shared library. It provides FIPS 140-3 Level 1 protection and implements FIPS 140-3-approved cryptographic algorithms. The library is dynamically linked to the calling application and loaded into memory for execution by the operating system loader. It exposes a C language Application Programming Interface (API) that supports key wrap/unwrap, keystream generation, and data integrity functions.

This module is identified as GD SecureComm Crypto, Version 1.1.0. The shared library is only available on Linux and consists of the shared library, *libgdcrypto.so*.

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary:

The module is a software cryptographic module and, as such, has no physical components. Its TOEPP is defined by the enclosure of the general-purpose computer on which it executes. The module operates entirely within the TOEPP. The cryptographic boundary is defined by module itself.

Tested Operational Environment's Physical Perimeter (TOEPP):

The General-Purpose Computer on which the module executes is the TOEPP.

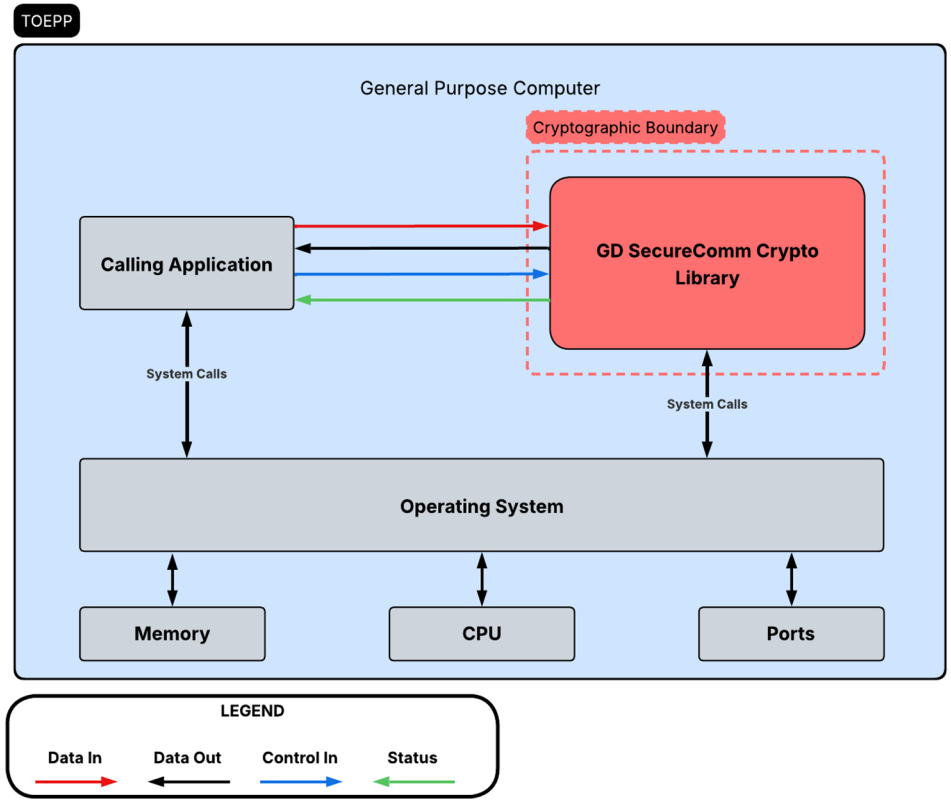

Figure 1 - Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Hardware:

N/A for this module.

Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets):

Package or File Name	Software/ Firmware Version	Features	Integrity Test
Libgdcrypto.so	1.1.0		HMAC-SHA2-256 Cert. #A7487 over the complete software binary.

Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)

Tested Module Identification – Hybrid Disjoint Hardware:

N/A for this module.

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Red Hat Enterprise Linux 9	Crystal Group RS2301S18N	Intel Xeon Platinum 8461V	No	N/A	1.1.0
Red Hat Enterprise Linux 9	Crystal Group RS2301S18N	Intel Xeon Platinum 8558U	No	N/A	1.1.0
Red Hat Enterprise Linux 9	Trenton Systems TRS21-18	AMD EPYC 7643P	No	N/A	1.1.0

Table 3: Tested Operational Environments - Software, Firmware, Hybrid

Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid:

N/A for this module.

2.3 Excluded Components

N/A for this Module

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	Approved Mode of Operation	Approved	As per IG 2.4.C Option 2, a global indicator is used implicitly through the successful completion of a service.

Table 4: Modes List and Description

The GD SecureComm Crypto supports only a single mode of operation, which is designated as the Approved Mode. This mode is entered exclusively by invoking the 'GD_FIPS_mode_set' API. Successful transition into the Approved Mode requires pre-operational and conditional self-tests to have successfully completed during module startup as well as the 'GD_FIPS_mode_set' API to be called in order to transition to the Approved Mode and utilize any of the module's services or security functions.

The GD SecureComm Crypto does not support or provide a 'non-approved' mode of operation. The module remains inactive and does not perform any cryptographic functions or services unless the Approved mode is explicitly set and all associated self-tests have passed.

2.5 Algorithms

Approved Algorithms:

The following cryptographic library and associated CAVP certificate is used by the cryptographic module:

- GD Secure Comm Crypto Library (Cert. #A7298)

Algorithm	CAVP Cert	Properties	Reference
AES-CTR	A7487	Direction - Encrypt Key Length - 256	SP 800-38A
AES-ECB	A7487	Direction - Decrypt Key Length - 256	SP 800-38A
AES-KW	A7487	Direction - Decrypt, Encrypt Key Length - 256	SP 800-38F
HMAC-SHA2-256	A7487	Key Length - Key Length: 256	FIPS 198-1
SHA2-256	A7487	Message Length - Message Length: 256	FIPS 180-4

Table 5: Approved Algorithms

Vendor-Affirmed Algorithms:

N/A for this module.

Non-Approved, Allowed Algorithms:

N/A for this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

Non-Approved, Not Allowed Algorithms:

N/A for this module.

2.6 Security Function Implementations

The table below lists the security function implementations for this module.

Name	Type	Description	Properties	Algorithms
AES-CTR Encryption	BC-UnAuthEncrypt	Encryption using AES-CTR	Publication:SP 800-38A	AES-CTR: (A7487) Key Size: 256 bits
AES-KW Encryption/Decryption	BC-Auth	Authenticated Encryption/Decryption using AES-KW	Publication:SP 800-38F	AES-KW: (A7487) Key Size: 256 bits AES-ECB: (A7487) Key Size: 256 bits AES-CTR: (A7487) Key Size: 256 bits
HMAC for Software Integrity	MAC	Message Authentication	Publication:FIPS 198-1	HMAC-SHA2-256: (A7487) Key Sizes: 256 bits MAC Length: 256 bits SHA2-256: (A7487)
Message Digest	SHA	Compute message digest using SHA	Publication:FIPS 180-4	SHA2-256: (A7487)

Table 6: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES-KW

The module does not establish SSPs using an approved key transport scheme (KTS).

However, it does offer approved authenticated algorithms that can be used by an external operator/application as part of an approved KTS.

2.8 RBG and Entropy

N/A for this module.

N/A for this module.

2.9 Key Generation

N/A for this module.

2.10 Key Establishment

N/A for this module.

2.11 Industry Protocols

N/A for this module.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API Input Parameters
N/A	Data Output	API Output Parameters
N/A	Control Input	API function calls
N/A	Status Output	Error Log API (API status parameters)

Table 7: Ports and Interfaces

The physical ports of the cryptographic module include those of the underlying computing platform on which the module executes. These physical ports are outside the scope of the FIPS 140-3 validation.

The logical interface to the module is defined as a C language Application Program Interface (API), through which all input, output, control, and status information is communicated with the cryptographic module.

The module does not implement a control output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

N/A for this module.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	CO	None

Table 8: Roles

4.3 Approved Services

All services listed in the table below can be accessed in approved mode and when in this mode exclusively use the security functions listed in Cryptographic Algorithms.

- In the 'Access Rights to Keys and/or SSPs' column:
  - G = Generate: The module generates or derives the SSP.
  - R = Read: The module exports the SSP.
  - W = Write: The SSP is imported or updated
  - E = Execute: The module uses the SSP in performing a cryptographic operation.
  - Z = Zeroize: The module zeroizes the SSP.

- In the 'Roles SSP Access' column:
  - For a complete description of SSP referenced from the table, see Section 9 Sensitive Security Parameters Management.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
AES Key Unwrap	AES-256 Key Unwrap	API Return Value	Ciphertext, AES key	Plaintext	AES-KW Encryption/Decryption	Crypto Officer - Key Wrapping Key (KWK): W,E,Z
AES Key Wrap	AES-256 Key Wrap	API Return Value	Plaintext, AES key	Ciphertext	AES-KW Encryption/Decryption	Crypto Officer - Key Wrapping Key (KWK): W,E,Z
Error Log	Retrieves the current error log, if there are any errors reported	API Return Value	API Call Parameters	Log Results	None	Crypto Officer
Keystream Generation	Generate keystream data via AES-256 CTR	API Return Value	Plaintext, AES key	Ciphertext	AES-CTR Encryption	Crypto Officer - Data Encryption Key (DEK) : W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Message Digest	Generates a SHA-256 message digest	API Return Value	Message	Digest Value	Message Digest	Crypto Officer
Perform Self-Tests	Performs self-test upon power-up or command.	API Return Value	API Call Parameters	Pass/Fail results of self-tests	HMAC for Software Integrity	Crypto Officer
Show Status	Provide Success or failure code to the calling application.	API Return Value	API Call Parameters	Return Code, Status	None	Crypto Officer
Show Version	Return the module name and version information	API Return Value	API Call Parameters	Module name and version	None	Crypto Officer
Zeroization	Zeroization occurs implicitly when the host machine is powered off or when any of the services that use approved security functions.	None	Powering the Module Off, Use of Approved Security Function	None	None	Crypto Officer - Data Encryption Key (DEK) : Z - Key Wrapping Key (KWK): Z

Table 9: Approved Services

4.4 Non-Approved Services

N/A for this module.

4.5 External Software/Firmware Loaded

N/A for this Module.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the GD SecureComm Crypto (executable file "*libgdcrypto.so*") is verified at runtime using an HMAC-SHA-256 (Cert. # A7487) based integrity check. During initialization, the module calculates an HMAC-SHA-256 value over its own binary image and compares this result to a known-good HMAC value that is embedded within the module at build time.

If the computed and stored HMAC values do not match, the integrity test fails, and the module transitions into the Error State, thereby preventing any further cryptographic operations.

5.2 Initiate on Demand

The integrity test is performed as part of the pre-operational self-tests, which are automatically executed during module initialization. This test ensures the authenticity and integrity of the modules binary prior to allowing any cryptographic operations.

In addition to execution at startup, the integrity test may also be invoked on demand in one of two ways:

- By reloading the module, which triggers re-initialization and re-execution of the pre-operational self-tests.

If the integrity test fails during either automatic or on-demand invocation, the module transitions to the Error State and ceases all cryptographic functionality.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The cryptographic module maintains exclusive control over its own SSPs, specifically any AES Keys created during cryptographic operations.

The process and memory management mechanisms of the host operating system prevent unauthorized access to secret keys by external processes during module execution. These protections ensure process isolation and memory confidentiality.

The module only permits access to SSPs through its API, maintaining strict control over all cryptographic material within the module's boundary.

6.2 Configuration Settings and Restrictions

The module is designed to operate in a single-user mode. Only one operator is permitted to utilize the module at any given time, and concurrent operators are explicitly excluded. This restriction is enforced by the operational environment in which the module executes.

7 Physical Security

N/A for this module.

8 Non-Invasive Security

N/A for this Module.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Random access memory	Dynamic

Table 10: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input Parameters	Outside Cryptographic Boundary	Cryptographic Module	Plaintext	Manual	Electronic	

Table 11: SSP Input-Output Methods

9.3 SSP Zeroization Methods

The module is designed such that zeroisation occurs as part of the approved services that process unprotected SSPs. Therefore, invoking any of the approved services in Section 4.3 serves as an implicit indication that zeroisation of the SSPs has completed.

Zeroization Method	Description	Rationale	Operator Initiation
Reboot	Upon rebooting the host device, the SSPs in memory are zeroized by being overwritten with zeroes.	Rebooting the host device causes SSPs in volatile memory to become irretrievable and unusable, thereby effectively zeroizing them.	Operator reboots the host device.
Remove Power from Module	Power is removed from the module and the memory containing SSPs is overwritten with zeroes.	Removing power from the host device renders SSPs in memory irretrievable and unusable, effectively zeroizing them.	Operator removes power from the host device.
Zeroization After Use	Immediately after use, the SSPs are zeroized by being overwritten with zeros.	Calling any of the approved services that utilizes an SSP will automatically call "SCRUB_AND_FREE_MEM".	Operator uses a service that utilizes an SSP.

Table 12: SSP Zeroization Methods

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Data Encryption Key (DEK)	Keystream generation via AES-CTR	256 bits - 256 bits	Symmetric Key - CSP			AES-CTR Encryption
Key Wrapping Key (KWK)	Wrap or unwrap key via AES-KW	256 bits - 256 bits	Symmetric Key - CSP			AES-KW Encryption/Decryption

Table 13: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Data Encryption Key (DEK)	API Input Parameters	RAM:Plaintext	For the duration of the service	Remove Power from Module Reboot Zeroization After Use	
Key Wrapping Key (KWK)	API Input Parameters	RAM:Plaintext	For the duration of the service	Remove Power from Module Reboot Zeroization After Use	

Table 14: SSP Table 2

The module does not generate keys or SSPs.

10 Self-Tests

FIPS 140-3 requires that the cryptographic module perform self-tests to ensure the integrity of the cryptographic module at start up. During the execution of the self-tests, cryptographic services are not available and data input and output is not possible.

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A7487)	SHA2-256	Software Integrity	SW/FW Integrity	Module becomes operational and services are available for use	Software Integrity Test

Table 15: Pre-Operational Self-Tests

The integrity of the GD SecureComm Crypto's text and data segments is verified using HMAC SHA-256. A digest is calculated at build time and embedded within the software image. When the linking application is launched at OS loader load-time, the digest is recalculated and compared against the stored digest. If the digests match, the Cryptographic Algorithm Self-Tests (CASTs) are initiated.

The HMAC-SHA-256 algorithm undergoes its CAST prior to executing the software integrity tests.

Only upon successful completion of all CASTs and pre-operational integrity tests does the module transition into the operational state. No operator intervention is required to reach this state.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-CTR (A7487)	256 bit	KAT	CAST	Module becomes operational and services are available for use	Encrypt	Power-Up
AES-KW Decrypt (A7487)	256 bit	KAT	CAST	Module becomes operational and services are available for use	Decrypt	Power-Up
AES-KW Encrypt (A7487)	256 bit	KAT	CAST	Module becomes operational and services are available for use	Encrypt	Power-Up
HMAC-SHA2-256 (A7487)	SHA2-256; Key Length: 256 bits	KAT	CAST	Module becomes operational and services are available for use	Message Authentication	Power-Up and prior to integrity test
SHA2-256 (A7487)	SHA2-256	KAT	CAST	Module becomes operational and services are available for use	Secure Hash	Power-Up and prior to integrity test

Table 16: Conditional Self-Tests

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A7487)	Software Integrity	SW/FW Integrity	On-Demand	Programmatically

Table 17: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-CTR (A7487)	KAT	CAST	On-Demand	Programmatically
AES-KW Decrypt (A7487)	KAT	CAST	On-Demand	Programmatically
AES-KW Encrypt (A7487)	KAT	CAST	On-Demand	Programmatically
HMAC-SHA2-256 (A7487)	KAT	CAST	On-Demand	Programmatically
SHA2-256 (A7487)	KAT	CAST	On-Demand	Programmatically

Table 18: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error State	All cryptographic functionalities will stop upon self-test failure.	Module fails any of the self-tests.	Re-initialization of the module.	Module becomes unresponsive

Table 19: Error States

During the execution of self-tests, all services are disabled, and data output via the module's data output interface is suppressed until the tests pass successfully. If any self-test fails, an error message is issued, and the module enters an error state.

10.5 Operator Initiation of Self-Tests

The module provides a Self-Test service to perform self-tests on demand which includes the pre-operational self-test. The Self-Test service can be called on demand by invoking the “GD_FIPS_on_demand_selftest” API call.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

General Dynamics will provide the following items to the application vendor:

- Libgdcrypto.so

The vendor will link the GD SecureComm Crypto to the application. Upon the first and subsequent instantiation of the host application, at power up, the GD SecureComm Crypto runs the HMAC SHA-256 to validate the integrity of the library by running a load time software integrity test. If successful, the library will perform the rest of the CAST self-tests. If the software integrity test fails, the library will enter an error state. If the first-time software integrity test fails, the application vendor should contact General Dynamics.

11.1 Administrator Guidance

After installing the GD SecureComm Crypto package, the Crypto Officer must verify the module name and version by executing the GD_retrieve_module_information() command. The Crypto Officer must confirm that the module version and status are listed in the output as follows:

- GD SecureComm Crypto, Version (1.1.0)

11.2 Non-Administrator Guidance

None

11.3 End of Life

Since the module does not persistently store SSPs, secure sanitization involves simply unloading the module, which zeroizes all SSPs in volatile memory.

12 Mitigation of Other Attacks

N/A for this Module.