ViPNet Common Crypto Core

Non-Proprietary Security Policy

FIPS 140-2 Level 1 Validation

Software Version: 2.0

Document Version 1.4

July 3rd, 2018

Table of Contents

1. Introduction

This document is the non-proprietary security policy for the ViPNet Common Crypto Core (Version 2.0). This document was prepared as part of the Federal Information Processing Standard (FIPS) 140-2 Level 1 validation process. This document contains a specification of the rules under which the module must operate and describes how this module meets the requirements as specified in the Federal Information Processing Standards Publication (FIPS PUB) 140-2 for a Security Level 1 multi-chip standalone module.

The purpose of this security policy is:

- To provide a specification of the cryptographic functionality that will allow individuals and organizations to determine whether the cryptographic module, as implemented, satisfies a stated security policy
- To describe to individuals and organizations the capabilities, protection, and access rights provided by the cryptographic module, thereby allowing an assessment of whether the module will adequately serve the individual or organizational security requirements

2. Cryptographic Module Specification

2.1.Cryptographic Module Description

The ViPNet Common Crypto Core (hereinafter referred to as the cryptographic module or module) is a software-only module supporting FIPS-approved cryptographic algorithms. It is available in user space and kernel driver implementations on a wide range of operational systems, including Windows user space (vccc64.dll), kernel space (kvccc64.sys), Linux user space (libvccc.so), kernel space (itcsvccc.ko), macOS, iOS (vccc.lib), and Android (libvcccfips.so). User space library and kernel library use the same base source code. The module is accordingly implemented and tested as a kernel driver library, a dynamic link library, and a shared-object file library. It is designed with a default entry point (DEP) which ensures that the power-up tests are initiated automatically when the module is loaded.

This module provides a C-language application program interface (API) for use by ViPNet applications that require cryptographic functionality.

The module contains the following cryptographic functionality:

- Symmetric key encryption and decryption
- Cryptographic hash function
- Message authentication code functions
- Key derivation and key wrapping functions

The cryptographic module is designed to reside in a number of ViPNet applications, including:

- ViPNet Administrator operating on Windows
- ViPNet Network Manager operating on Windows
- ViPNet Network Security Management System (NSMS) operating on Windows
- ViPNet Client and ViPNet Coordinator operating on Windows and Linux
- ViPNet Client for macOS
- ViPNet Client for Android
- ViPNet Client for iOS
- ViPNet SoftToken operating on macOS X, Android, iOS, Windows, and Linux

The products comprise essential part of the ViPNet technology, which power almost all ViPNet solutions and products. The ViPNet technology is designed for deployment over global and local networks, and fully leverages the existing network infrastructure without decreasing network performance. It facilitates the transparent interaction of protected hosts, independently from their location or the way they are connected to a network. The interaction can be established on the basis of client-to-client, client-to-site and site-to-site (VPN tunnel) schemes.

Management ViPNet software (ViPNet Administrator, ViPNet Network Manager, ViPNet NSMS) is used to manage the logical structure of the ViPNet network as well as cryptographic keys. ViPNet Coordinator performs functions of a VPN gateway. ViPNet Client provides a comprehensive network protection for VPN users and includes secure communication applications, such as instant messaging and file exchange.

More information on ViPNet solutions and products, see http://www.infotecs.us/products/.

For FIPS 140-2 validation, the module has been tested by an accredited Cryptographic Module Testing Laboratory (CSTL) on the following operating environments:

Table 1 - Tested Operating Systems and Platforms

Operating System	Hardware Platform	Acceleration
Windows 10 x64 (user space)	ASUS Zenbook UX51VZA	No
Linux Debian 7.11 (user space)	Acer Aspire s7-191	No
Linux Debian 7.11 (kernel space)	Acer Aspire s7-191	No
Android 7.0	Samsung Galaxy A5	No
iOS 10.3.3	iPhone 7+	No
macOS X 10.12	MacBook Pro	No
Windows 10 x64 (kernel space)	ASUS Zenbook UX51VZA	No
Linux Debian 7.11 (kernel space)	Acer Aspire s7-191	Yes (AES-Ni)

Infotecs affirms that the module (compiled with the same source code) remains compliant with FIPS 140-2 requirements when operated in user space and kernel space on Windows 7 32/64-bit, Windows 8 32/64-bit, Debian Linux 6 32-bit and Debian Linux 7 64-bit. The following table lists the security level for each of the sections of FIPS validation.

The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when ported to an operational environment which is not listed on the validation certificate (see IG G.5)

Table 2: FIPS validation security levels

Security Component	Security Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	3
Mitigation of Other Attacks	N/A

2.2.Approved Mode of Operation

The cryptographic module supports only a FIPS 140-2 approved mode of operation. When the cryptographic module is loaded and initialized, it is set in the FIPS 140-2 approved mode.

The cryptographic module supports the following FIPS approved algorithms.

Table 3: Approved Algorithms

Algorithm	Sizes	Validation Certificate	Usage
[FIPS 197, NIST SP 800-38A] AES (ECB, CFB 128 and CTR modes)	256-bits	AES Cert. #5033 and #5035	Encryption/decryption
[NIST SP 800-38B] AES-CMAC	256-bits	CMAC Cert. #5033 and #5035	Message authentication
[FIPS 198-1] HMAC	HMAC- SHA-256	HMAC Cert. #3347 and #3349	Keyed hash message authentication
[FIPS 180-4] SHS	SHA-256	SHS Cert. #4092 and #4094	Hashing
[NIST SP 800-90A] CTR_DRBG	AES 256-bits	DRBG Cert. #1846 and #1848	Random bit generation
[SP 800-108] KDF in Counter Mode, AES-CMAC as PRF	AES 256-bits	KDF Cert. #168 and #170	Key derivation function
[NIST SP 800-38F] AES key-wrap (KTS)	256-bits	KTS Cert. #5033 and #5035	Key wrapping; key establishment methodology provides 256 bits of encryption strength

2.3.Cryptographic Module Boundary

For FIPS 140-2 purposes the cryptographic module is classified as a multi-chip standalone module. The same module (same algorithmic functionality, same services and interfaces) can be compiled as a dll, kernel driver or shared-object file. Therefore the logical cryptographic boundary of the module consists of:

- either a Driver file for the operation in Windows kernel mode
- or a Dynamic Link Library file for the operation in Windows user space
- or a Shared Library file for the operation in macOS/iOS/Android user space

The physical cryptographic boundary of the module is the enclosure of the general purpose computer system on which it is executing. The module performs no communications other than with the process that calls it and does not store any persistent information.

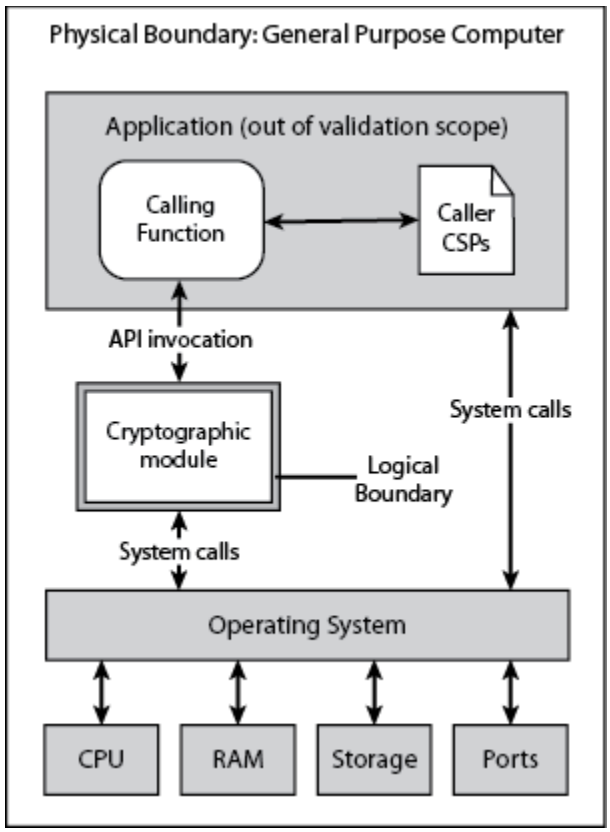

Figure 1: Cryptographic module logical diagram

3. Cryptographic Module Ports and Interfaces

The physical ports of the module are the same as the general purpose computer system on which it is executing. The logical interface is a C-language application program interface (API).

The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the API functions. The Status Output interface includes the return values of the API functions.

Table 4 : Ports and interfaces

FIPS Interface	Physical Ports	Logical Ports
Data Input	Physical ports of a GPC	API input parameters
Data Output	Physical ports of a GPC	API output parameters and return values
Control Input	Physical ports of a GPC	API input parameters
Status Output	Physical ports of a GPC	API return values
Power Input	Physical ports of a GPC	N/A

As a software module, control of the physical ports is outside module scope. However, when the module is performing self-tests, or is in an error state, all output on the logical data output interface is inhibited.

4. Roles, Services, and Authentication

The cryptographic module implements both User and Crypto Officer (CO) roles. As allowed by FIPS 140-2 at Security Level 1, the module does not support user authentication for those roles. Only one role may be active at a time and the module does not allow concurrent operators.

The User and CO roles are implicitly assumed by the entity accessing services implemented by the module:

- User role: Loading the module and calling of any API function. This role has access to all of the services provided by the module.
- CO role: Installation of the module on the host computer system. Loading the module and calling of any API function. This role is assumed implicitly when the system administrator installs the module.

The services supported by the module and access rights within services are listed in the table below.

Table 5: Services and access rights

Service	Approved security functions	Cryptographic keys and/or CSPs	Roles	Access rights to keys and/or CSPs
Symmetric encryption/decryption	AES 256 bits (ECB, CFB128, CTR mode)	AES encrypt/decrypt keys	User, CO	Execute
Keyed hashing (HMAC)	HMAC-SHA256	HMAC key	User, CO	Execute
Hashing	SHA-256	None	User, CO	N/A
Message authentication	AES-CMAC	AES CMAC key	User, CO	Execute
Random number generation	CTR_DRBG	CTR_DRBG CSPs	User, CO	Write/Execute
Key derivation	KDF in Counter Mode using AES-CMAC as PRF	Input: AES encrypt/decrypt key, Output: KDF keying material	User, CO	Execute
Key wrapping	AES KW	AES encrypt/decrypt key, Key-encryption key	User, CO	Execute
Self-test	HMAC-SHA256	Integrity key, HMAC digest	User, CO	Execute
Zeroization	None	All keys	User, CO	N/A
Show status	None	None	User, CO	N/A

5. Physical Security

The cryptographic module is comprised of software only and thus does not claim any physical security.

6. Operational Environment

The cryptographic module operates under Windows, Linux, Android, macOS, iOS

Windows, Linux, Android, macOS, and iOS are defined as modifiable operational environment per the FIPS 140-2. These operational systems segregate user processes into separate process spaces. Each process space is an independent virtual memory area that is logically separated from all other processes by the operating system software and hardware.

The module functions entirely within the process space of the application that invokes it. The calling application is the single user of the module even when the operating system supports multiple user sessions. Therefore, the module satisfies the FIPS 140-2 requirement for a single user mode of operation.

7. Cryptographic Key Management

The table below provides a complete list of Keys and CSPs used within the module:

Table 6 - Keys and CSPs

Keys and CSPs	Description
AES encrypt/decrypt key	Used for symmetric encryption/decryption
HMAC key	256 bits; used for HMAC-SHA256 keyed hash
Integrity key	HMAC key used to perform software integrity test at power-on. This key is embedded within the module.
HMAC digest	Pre-calculated HMAC-SHA256 digest used for software integrity check
AES CMAC key	256 bits; used for CMAC generate and verify
CTR_DRBG CSPs	CSPs for CTR_DRBG as per NIST SP 800-90A: seed (384 bits), internal state V (128 bits) and Key (AES-256) values
Key-encryption key	256 bits; key-encryption key as per NIST SP 800-38F
KDF keying material	Keying material (min. 16 bytes, max. 256 bytes) resulting from a key derivation function in Counter Mode using AES-CMAC as pseudorandom function

7.1.Key Generation

The cryptographic module does not perform key generation. It implements a NIST SP 800-90A compliant random number generator to generate random bits which are used by the calling application for key generation. The calling application is responsible for generation and storage of keys.

The module passively receives a minimum of 264-bits of entropy from its operational environment per FIPS 140-2 IG 7.14 2 (b). No assurance of the minimum strength of generated keys.

7.2.Key Entry and Output

The cryptographic module is passed keys and CSPs as API parameters, associated by memory location. The application using the cryptographic module passes keys and CSPs to the module in plaintext within the physical boundary.

The module does not output CSPs, other than as explicit results of the DRBG and KDF services. However, none cross the physical boundary.

7.3.Key Storage

The cryptographic module does not perform persistent storage of keys. Keys and CSPs are passed to the module by the calling application. The keys and CSPs are stored in memory in plaintext. Keys and CSPs residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the module defined API. The operating system protects memory and process space from unauthorized access.

7.4.Zeroization Procedure

The module is passed keys as part of a function call from a calling application and does not store keys persistently. The calling application is responsible for parameters passed in and out of the module. Upon uninstantiation of the DRBG temporary or ephemeral keys utilized by the module which reside in memory space will be overwritten with random bits by the Approved DRBG.

All CSPs can be zeroized by power-cycling the module (with the exception of the Integrity key and HMAC digest).

8. Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

The Module is software only and shall be installed on general purpose computing devices complying with FCC standards.

9. Self-Tests

The power-up self-tests are executed automatically when the cryptographic module is loaded. The kernel module initialization function returns a status code indicating whether self-tests succeeded or failed.

If the self-tests are successful, then the module operates in the FIPS Approved mode.

If the power-up self-tests fail, the cryptographic module enters an error state. No further cryptographic operations are possible when the module is in an error state. The module can be recovered from a power-up self-test failure only by reload. If the failure persists the ViPNet application encompassing the cryptographic module must be reinstalled.

The power-up self-tests may be performed on-demand at any time by reloading the module.

9.1.Integrity Test

The module initialization function verifies the integrity of the runtime executable using a HMAC-SHA-256 digest computed at build time. If the digest computed at self-test matches the stored known digest then the algorithm specific Known Answer tests are performed.

9.2.Cryptographic Algorithm Tests

Power-up self-tests include the following cryptographic algorithm tests which are conducted for all cryptographic functions using known answers.

Table 7: Algorithm specific tests

Algorithm	Type
AES	Separate KAT for encrypt and decrypt. CTR, CFB128 and ECB modes
AES-CMAC	Separate KAT for generate and verify
HMAC with SHA-256	KAT

Algorithm	Type
CTR_DRBG	KAT
KDF in Counter Mode	KAT, AES CMAC as Pseudo Random Function
Key wrap	KAT

9.3. Conditional Tests

A Continuous Random Number Generator Test is performed for the CTR_DRBG.

9.4. Critical Function Tests

The module implements the health tests described in Section 11.3 of SP 800-90A:

- Instantiate Test
- Generate Test
- Reseed Test
- Uninstantiate Test

10. Design Assurance

10.1. Configuration Management

Infotecs performs configuration management of the cryptographic module. A complete revision history of the source code from which the module was generated is maintained in a version control database.

10.2. Delivery and Operation

10.2.1. Delivery

The cryptographic module is never released outside of Infotecs as a source code distribution. It is contained within Infotecs source code management repository that can be accessed by engineers to download a copy of the code. It is not possible to make changes to the code and replace it within the repository. When a developer downloads code for integration in a ViPNet product, the code gets integrated into the configuration management structure for that product. The module code is then linked as part of an application build process that is configured to operate in FIPS approved mode.

10.2.2. Initialization

The cryptographic module is initialized by loading the module before any cryptographic functionality is available. Loading the module in the user space is performed by the encompassing ViPNet application. Loading the module in the kernel is performed by invoking the module in the system boot sequence.

10.2.3. Recovering from a Self-Test Failure

If the FIPS 140-2 power-up self-tests fail, the module must be reloaded. The ViPNet application encompassing the module is responsible for reloading the module. If the failure persists the ViPNet application encompassing the cryptographic module must be reinstalled.

11. Mitigation of Other Attacks

The cryptographic module does not contain additional security mechanisms beyond the requirements for FIPS 140-2 level 1 cryptographic modules.