GIRITECH

FIPS 140-2 Non-Proprietary Security Policy

for the

Giritech Cryptographic Support Library

CryptFacility, version 1.0.485

Level 1 Validation

Document Version: Version 1.18

July 1, 2007

REVISION HISTORY

The table below provides revision history of this document.

Version	Date	Author(s)	Comments
1.0	August 15, 2006	Apex Assurance Group	Template / draft Module Interfaces detail
1.1	August 21, 2006	Giritech	Added Product Overview
1.2	August 30, 2006	Giritech	Added Officer role descriptions
1.3	September 15, 2006	Giritech	Minor editorials
1.4	September 19, 2006	Giritech	Introduction updated, cryptographic boundary definition added, services added, reference to algorithm certificates added and Crypto++ use confirmed.
1.5	September 24, 2006	Apex Assurance Group	Reviewed content of document, inserted comments, inserted or reordered sections, added/modified text
1.6	September 26, 2006	Giritech	Added figures and updated tables
1.7	September 26, 2006	Giritech	Added section on cryptographic key management and self tests.
1.8	September 28, 2006	Apex Assurance Group	Clarified details for self tests, added general comments and added/modified text
1.9	September 29, 2006	Giritech	Updated with final design details
1.10	October 2, 2006	Apex Assurance Group	Final updates for submission to SAIC
1.11	October 3, 2006	Apex Assurance Group	Re-embedded block diagram image
1.12	November 3, 2006	Giritech	Updated with SAIC comments
1.13	November 7, 2006	Giritech	Further Clarifications added
1.14	November 30, 2006	Giritech / Apex Assurance Group	Clarifications
1.15	December 8, 2006	Apex Assurance Group	Address SAIC comments, update module digest value
1.16	December 14, 2006	Apex Assurance Group	Correct algorithm certificate table and minor formatting
1.17	June 7, 2007	Giritech	Minor updates according to NIST comments
1.18	July 1, 2007	Apex Assurance Group	Further clarification on NIST comments

Table 1 - Security Policy Revision History

TABLE OF CONTENTS

DOCUMENT INTRODUCTION

Background

Federal Information Processing Standards Publication 140-2 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic products to be deployed in a Sensitive but Unclassified environment. More information about the FIPS 140-2 standard and validation program is available on the NIST website at http://csrc.nist.gov/cryptval/ .

This non-proprietary Cryptographic Module Security Policy for the Cryptographic Support Library ("the Module" otherwise referred to as "CryptFacility") from Giritech provides an overview of the product line using the module and a high-level description of how the module meets the security requirements of FIPS 140-2. This document contains details on the module's cryptographic keys and critical security parameters. This Security Policy concludes with instructions and guidance on running the module in FIPS 140-2 mode of operation.

Further Information

The Giritech website (http://www.giritech.com) contains information on the full line of products from Giritech, including a detailed overview of the G/On product line that is using the CryptFacility library. The NIST Cryptographic Module Validation Program website (http://csrc.ncsl.nist.gov/cryptval/) contains Giritech contact information for answers to technical or sales-related questions.

Copyright Notice

All information available in the Giritech FIPS 140-2 documentation pertaining to the Giritech Cryptographic Support Library (the "Module" otherwise referred to as "CryptFacility") is the sole property of Giritech A/S and as such is confidential.

However it is specifically mentioned that this document (the Security Policy) may be freely reproduced and distributed in its entirety without modification (including this Copyright Notice).

Relationship to the Giritech G/On Product Line

The Cryptographic Support Library from Giritech is designed for use inside the full range of Giritech products, including G/On. As such the module implements the security relevant functions of Giritech's products. The typical "users" of the module are therefore other Giritech software processes referred to as "calling daemons" in the following text. The Cryptographic Support Library is also referred to as "CryptFacility" and is validated for FIPS 140-2.

GIRITECH CRYPTOGRAPHIC SUPPORT LIBRARY

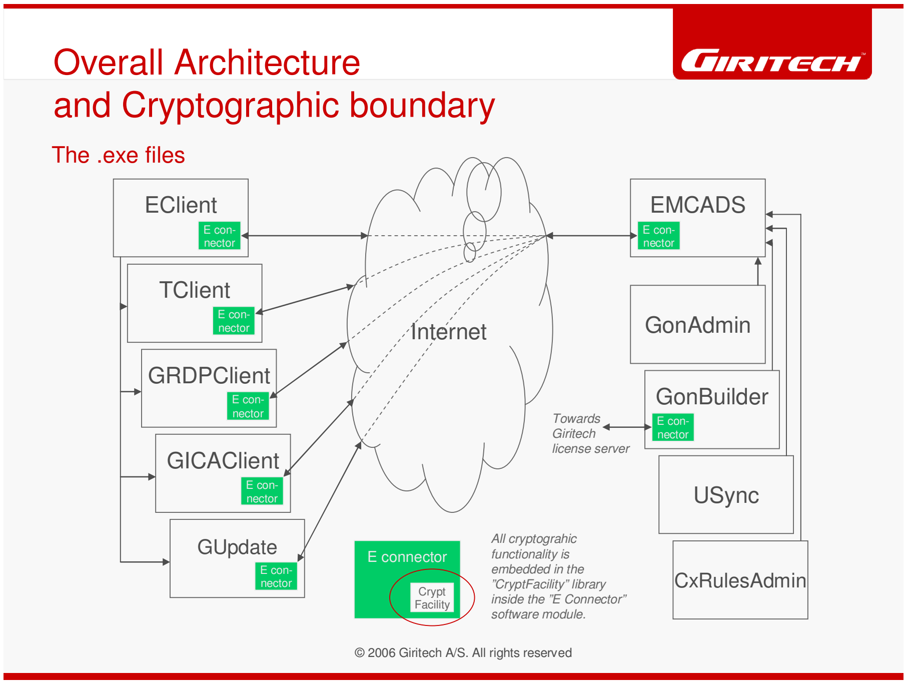
Giritech Product Line Overview

As a developer and manufacturer of remote connectivity solutions, Giritech has developed the G/On product line which covers most of the aspects needed to provide a complete all-in-one solution to typical connectivity needs of enterprises. G/On thus implements the connection itself, unique authentication of end-users and devices, limits dependence on endpoint state, control of access to the company and connection to individual applications, all in one integrated solution.

G/On combines:

- Encryption of data to make them inaccessible for everyone except the intended users.
- Mutual authentication of client *and* server, thus avoiding “man-in-the-middle” attacks.
- Authentication of users (username and password) integrated with existing Microsoft Active Directory to simplify user administration and to obtain single sign-on.
- An intelligent way of avoiding the security challenge from insecure computers when connecting from unknown IT infrastructures.
- Minimizes the risk for and potential consequences of attacks on the company’s internal IT infrastructure.

As an all-in-one remote connectivity solution, G/On contains a range of advanced cryptographic functions to enable secure connections to be established, maintained and removed. All executables in the G/On product line (see Figure 1 below for an overview) thus uses a software entity called “E connector” when connecting to each other through a network (e.g. the Internet). Inside the “E connector” all operations performing cryptographic tasks are using the standard Giritech Cryptographic Support library, “CryptFacility”, also referred to as the “module.”



Overall Architecture
and Cryptographic boundary

All cryptographic functionality is embedded in the ”CryptFacility” library inside the ”E Connector” software module.

© 2006 Giritech A/S. All rights reserved

Figure 1: G/On software architecture and CryptFacility cryptographic boundary

All these cryptographic tasks (key generation, encryption, etc.) of G/On are thus embedded in the module and it is this module that has been FIPS 140-2 validated. This document describes the details of the module and how it interfaces with and is used by other software solutions in the Giritech product-line, among others. A block diagram of the module can be seen on Figure 3.

Validation Level

The validation levels refer to the 4 levels of validation in the FIPS 140-2 standards. Giritech is targeting level 1 for all relevant parts of the CryptFacility library.

The following table lists the level of validation for each of the areas in FIPS 140-2 for the module:

Section No.	Area Title	Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services, and Authentication	1
4	Finite State Model	1
5	Physical Security	N/A
6	Operational Environment	1
7	Cryptographic Key management	1
8	Electromagnetic Interface/Electromagnetic Compatibility	1
9	Self-Tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A

Table 2 – Validation Level by Section

Physical security is not relevant as the module is a pure software-based solution. The “Mitigation of Other Attacks” section is also not relevant as the module does not implement any countermeasures towards special attacks.

Module Definition

The Cryptographic Support Module is classified as a multi-chip standalone cryptographic module. As such, it has a well-defined physical and logical boundary. The physical boundary includes the generic, General Purpose Computer (GPC) upon which the module executes. The module’s logical cryptographic boundary includes the executable image of the module stored on the PC hard drive and running from Random Access Memory (RAM). There exists a distinct set of interfaces for each boundary definition.

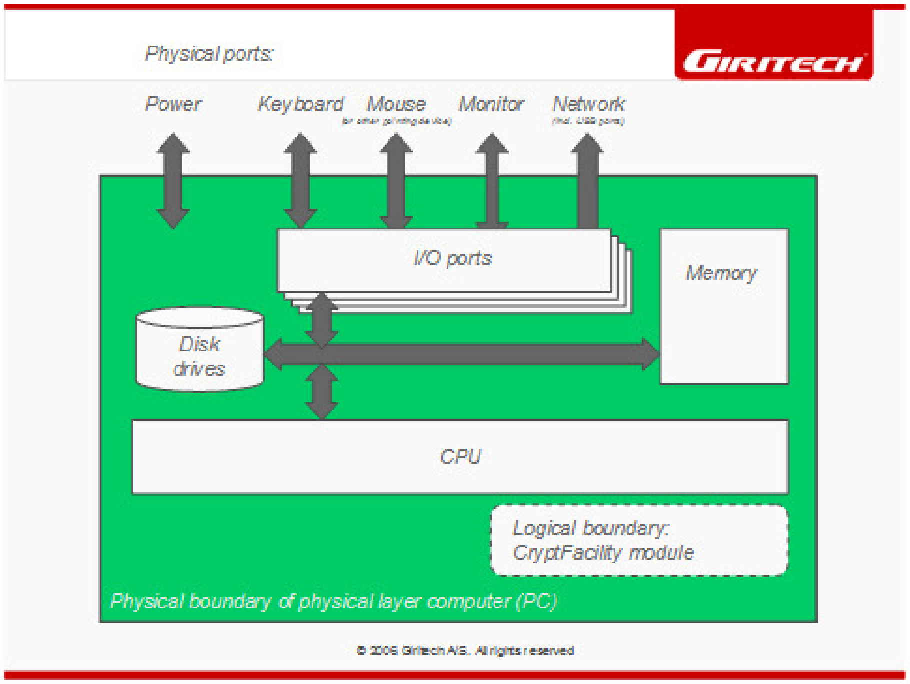

GIRITECH
Physical ports:
Power
Keyboard
Mouse
(or other pointing device)
Monitor
Network
(incl. USB ports)
I/O ports
Memory
Disk drives
CPU
Logical boundary: CryptFacility module
Physical boundary of physical layer computer (PC)
© 2006 Giritech A/S. All rights reserved

Figure 2: Physical Boundary

The overall block diagram of the module is very simple and basically summarizes the services (classes) available inside the CryptFacility module.

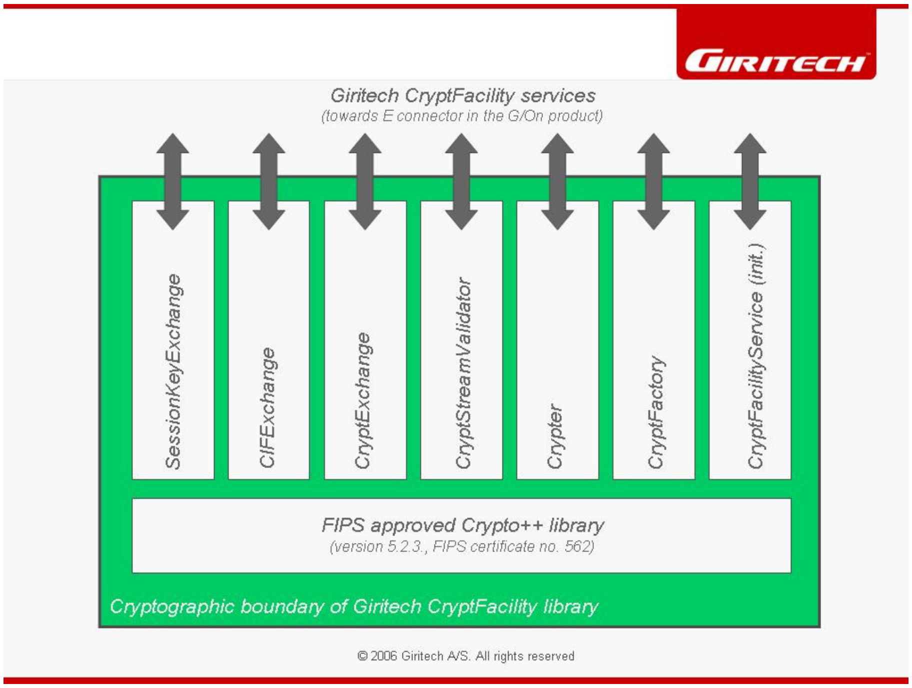

GIRITECH
Giritech CryptFacility services
(towards E connector in the G/On product)
SessionKeyExchange
CIFExchange
CryptExchange
CryptStreamValidator
Crypter
CryptFactory
CryptFacilityService (init.)
FIPS approved Crypto++ library
(version 5.2.3., FIPS certificate no. 562)
Cryptographic boundary of Giritech CryptFacility library
© 2006 Giritech A/S. All rights reserved

Figure 3: CryptFacility Block Diagram

Note that the Giritech CryptFacility module is using a FIPS validated (certificate number 562, http://csrc.nist.gov/cryptval/140-1/140crt/140crt562.pdf) encryption/decryption library called “Crypto++” from Wei Dai. CryptFacility will ensure correct version and correct configuration and setup of Crypto++ according to the Crypto Officer guidelines in the Security Policy of Crypto++ to ensure FIPS approved mode of operation.

Module Ports and Interfaces

The interfaces (ports) for the physical boundary include the computer keyboard port, CDROM drive, floppy disk, mouse, network port, parallel port, USB ports, monitor port and power plug. When operational, the module does not transmit any information across these physical ports because it is a software cryptographic library. Therefore, the module’s interfaces are purely logical and are provided through the Application Programming Interface (API) that a calling daemon can operate. The logical interfaces expose services that applications directly call, and the API provides functions that may be called by a referencing application (see *Roles and Services* for the list of available functions).

The API provided by the module is mapped onto the FIPS 140- 2 logical interfaces: data input, data output, control input, and status output. Each of the FIPS 140- 2 logical interfaces relates to the Module's callable interface, as follows:

FIPS 140-2 Logical Interfaces	Module Logical Interfaces	GPC Physical Ports
Data Input	Input arguments to API functions	Standard GPC input ports (e.g., keyboard)
Data Output	Return values or output parameters from API functions	Standard GPC output ports (e.g., monitor)
Control Input	Calls to API functions	Standard GPC input ports (e.g., keyboard)
Status Output	Information returned via exceptions (such as return or exit codes)	Standard GPC output ports (e.g., monitor)
Power	N/A	Standard GPC power port (e.g., power connector)

Table 3 – Cryptographic Support Library Logical Interface Mapping

Roles and Services

The module supports a *Crypto Officer* and a *User* role. The Crypto Officer can access all services in the module and perform initialization while the User role can only access the services of the module. The module supports no *Maintenance* role.

The Module does not implement authentication of the two roles. Roles are implicitly selected via the services being called and the situation in which they are called as described below. There are also no internal audit trails tracking any of the events or data generated or used by the module during FIPS approved mode of operation.

Crypto Officer Role

The Crypto Officer role is responsible only for the initialization of the module. The Crypto Officer role has no special access to keys or data *inside* the module except for generating the shared secret at initialization time. The Crypto Officer role is implicitly selected or assumed when initializing module.

User Role

The User role includes all the calling daemons (other software modules outside CryptFacility) using the Module as part of their normal operation. The user role can access all services in the module, but should not be allowed to set the mode of operation (initializing the service “CryptFacilityService”), generate the preshared secret (using the sub-service “generateClientServerKnownSecret” of the service “SessionKeyExchange”) or setting the preshared secret(using the sub-services “createFromMessage” and constructor of the service “SessionKeyExchange”) when the module is operating in FIPS approved mode.

The User role is implicitly selected or assumed when calling any services on the module API when using the module during normal operations.

Available Services

The services available to the User and Crypto Officer roles in the Module consist of the following:

Service	Description	Roles	Class name
Session key exchange	The *session-key-exchange* exchange the public key of a public/private session key pair. The key's are to be used later in the key exchange process. Both server and client generate a key pair and distribute the public key to the other part.	User	SessionKeyExchange
Client Information and Facility exchange	The *CIF-exchange* exchange client facility and identification information and key/iv to use for the upstream cipher. The facility information contains a list of available ciphers, key lengths and hashids (see Table 6). The identification information contain G/On-usb-key id and other information to identify the client. Notice that all identification information is seen as data and is not used inside the module.	User	CIFExchange
Cryptexchange	The *crypt-exchange* exchange the cipher for up- and down-stream, the symmetric downstream key and holds a hash (SHA-1) of the server-key message to test up- and downstream ciphers.	User	CryptExchange
Cryptvalidator	This class contains methods to validate the up- and down-stream ciphers.	User	CryptStreamValidator
Message encryption and decryption	This class handles encryption and decryption of individual messages.	User	Crypter
Collection of basic services	This service includes: • GenerateRandomNumber (This call generates a random number based on the PRNG function inside the Crypto++ library) • Hashing of messages (using SHA1 based on Crypto++ service)	User	CryptFactory
Initialization of module	Initializes and tests the module upon power-on and include calls that implement on-demand status verification of the module: • initialize (this service runs the built-in selftests, see page 14, and specifies what mode of operation the module is operating in: FIPS approved or non-FIPS approved) • getIntegrityHash (this call verifies the integrity of the module by calculating a HMAC SHA-1 hash on the .dll) • isReady, isModeOfOperationFIPS, getVersionInfo (provides module status information)	Crypto Officer	CryptFacilityService

Table 4 – User Services and Descriptions

The services accessing the Critical Service Parameters (CSPs), the type of access and which role accesses the CSPs are listed in Table 6. Note that all keys and ciphers are directly derived from the embedded Crypto++ module. However the Crypto++ library is not directly available on any CryptFacility APIs.

Note that all conditional self-tests are not separate service calls on the external API but are embedded in the relevant other services and performed as part of calling the above mentioned services.

Physical Security

The Giritech CryptFacility module is a software-only module and does not provide any physical security mechanisms. Therefore, this section is not applicable.

Operational Environment

The Giritech CryptFacility module operates on a general purpose computer (GPC) running on a modern version of the Microsoft Windows general purpose operating system (GPOS), including Microsoft Windows XP and Microsoft Windows 2003 Server. For FIPS purposes, Giritech CryptFacility is running on Microsoft Windows in single user mode and does not require any additional configuration to meet the FIPS requirements.

The Giritech CryptFacility module was tested on the Microsoft Windows XP Professional. The GPC(s) used during testing met Federal Communications Commission (FCC) FCC Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part15, Subpart B. While the module was tested on Microsoft Windows XP Professional, FIPS 140-2 validation compliance is maintained when the module is operated on other versions of the Microsoft Windows GPOS running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

Cryptographic Algorithms

The module implements the FIPS-approved algorithms as listed in Table 5. As the module uses an already FIPS approved encryption library (Crypto++, refer to www.cryptopp.com from Wei Dai, ver. 5.2.3[1]) the Giritech module's cryptographic algorithm implementations leverage the same certifications as this module. In general all approved algorithms and modes from the Crypto++ library are supported in the CryptFacility module, the list below only summarizes the by Giritech recommended modes. The module does not use any non-FIPS approved algorithms.

[1] FIPS certificate 562, obviously used according to guidelines and recommendations in the Crypto++ Security Policy (http://csrc.nist.gov/cryptval/140-1/140sp/140sp562.pdf) to ensure it is only used in FIPS validated mode.

Algorithm	Recommended Mode	Standard	Certificate Number
AES	CBC	FIPS 197	216
TDES	CBC	FIPS 46-3	309
Skipjack	CBC	FIPS 185	14
ECDSA	EC2N-SHA	FIPS 186-2	5
DSA	n/a	FIPS 186-2	79
SHA-1	SHA	FIPS 180-2	134
HMAC-SHA-1	SHA	FIPS 198	26
RNG	DES-EDE3[2]	ANSI X9.31-1998-App.A	61
RSAES	OAEP	ANSI X9.31-1998-App.A	562[3]

Table 5 – FIPS-approved Cryptographic Algorithms

Please note that the listed symmetric algorithms (AES, TDES and Skipjack) are all available for generic crypto services offered by the "Crypter" service. However during "CIFExchange"/"CryptExchange" AES is always selected for both up- and downstream ciphers.

Note that the listed symmetric algorithms (AES, TDES and Skipjack) are the ones available for selection in the "CIFExchange" service. "CIFExchange" allows a client implementation of the module to list these three algorithms for the server implementation of the module to select from. The actual selection of ciphers is however outside the scope of the CryptFacility module, but CryptFacility must offer all three modes for selection and ensures that only FIPS approved algorithms (as listed) can be used.

Cryptographic Key Management

The module is using a FIPS validated library (Crypto++), therefore please refer to the documentation on this library for the further details on Crypto++ key management. Besides the functionality described in that module, the CryptFacility module implements additional key management procedures as described below. Table 6 provides is a complete list of critical security parameters used within the module.

[2] The RNG algorithm is seeded using the CryptGenRandom API provided by the Windows OS.

[3] Note that, according to the security policy for Crypto++ page 3, http://csrc.nist.gov/cryptval/140-1/140sp/140sp562.pdf, there is no FIPS approved key exchange method. However the CMVP allows the use of RSAES in FIPS approved mode of operation. Refer to FIPS 140-2 annex D.

Key / CSP Name	Type	Use	Generation	Storage	Zeroization	User Privileges	Crypto Officer Privileges
PRNG Seed	System Entropy	Seed the X9.31 PRNG	Internal to Crypto++ library	RAM (plaintext)	Generating a new seed	R	R
Client/Server known secret	ECDSA (EC2N-SHA)	For signing of public key from Server key pair and signing of CIF from client ("SessionKeyExchange" and "CIFExchange")	Pre-defined shared secret[4]	RAM (plaintext)	At termination of service ("SessionKeyExchange" and "CIFExchange")	R	R W D
Client key pair	RSAES	For en/de-crypting the CIF ("CIFExchange")	Generated by the module using RNG service	RAM (plaintext)	At termination of service ("CIFExchange")	R	R
Server key pair	RSAES	Selection of cipher ("CryptExchange")	Generated by the module using RNG service	RAM (plaintext)	At termination of service ("CryptExchange ")	R	R
Upstream key/iv pair	Selected cipher (AES 256)	En/de-crypting messages ("Crypter", "CryptValidator")	Generated by the module using RNG service	RAM (plaintext)	At termination of service ("Crypter")	R	R
Downstream key/iv pair	Selected cipher (AES 256)	En/de-crypting messages ("Crypter", "CryptValidator")	Generated by the module using RNG service	RAM (plaintext)	At termination of service ("Crypter")	R	R
Recipe	SHA-1	Hash used to validate up- and down-stream ciphers ("CryptStreamValidator")	Generated by hashing the CryptExchange message	RAM (plaintext)	At termination of service ("CryptExchange ")	R	R
Key/IV pair	TDES and Skipjack	En/de-crypting"("Crypter" , "CryptValidator")	Generated by the module using RNG service	RAM (plaintext)	At termination of service ("Crypter")	R	R

Table 6 –Critical Security Parameters

Zeroization has been implemented to ensure no traces are left in memory (RAM, HardDrives or elsewhere) of any CSP's upon termination of the service using the CSP. Zeroization has been implemented by overwriting the allocated memory buffer with zeros before freeing the memory to other uses. Any service using a CSP will zeroize the CSP upon normal termination and when transitioning into error states. Note

[4] Shared secret key pair is generated by the CryptoOfficer *at install time as part of the basic initialization process* as part of the "SessionKeyExchange" service. As there are no authentication implemented in the module, the module itself does not limit the "User" role from gaining write privileges to the shared secret, but as matter of definition only the CryptoOfficer role should be allowed to change and/or generated the shared secret when operating in FIPS approved mode.

that the CryptFacility module does not implement key establishment. Zeroization is initiated by terminating the process and powering off the module.

Self-Tests

Two basic sets of tests are run by the module: Power-on self-tests and conditional self-tests.

Power-on self-tests are run upon every initialization of the module and if any of the tests fail, the module will not initialize. The module will enter an error state and no services can be accessed by the users. Conditional self-tests are on-demand tests and tests run continuously during operation of the module. If any of these tests fail, the module will enter and error state and will power off. No services can be accessed by the users.

The available power-on self-tests are as follows:

Power-on Self Test	Description
AES Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
TDES Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
SkipJack Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
RSAES Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
ECDSA Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
SHA-1 Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
HMAC Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
Random Number Generator Known Answer Test	Executed via the associated Crypto++ self-tests. If the test fails, CryptFacility will fail and enter an error state
Integrity Check	The Crypto++ integrity is verified as part of above mentioned power-on test while the CryptFacility module performs its own integrity check using HMAC-SHA-1and entering into error state when/if failing. The integrity test verifies the integrity of the executable code

Table 7 – Power-on Self Tests

The Power-on self-tests can be run on demand by reinitializing the module in FIPS approved Mode of Operation or by calling the corresponding service on the API.

The available conditional self-tests are as follows:

Conditional Self Test	Description
RSAES Pairwise Consistency Check	Test generated each time a key is generated. Executed via CryptFacility.
ECDSA Pairwise Consistency Check	Test generated each time a key is generated. Executed via CryptFacility.
Continual Random Number Generator Test	If the test fails, CryptFacility will fail and enter an error state. Executed via the associated Crypto++ self-tests.

Table 8 – Conditional Self Tests

The status of all tests are output from the module towards the calling daemons via the standard API to enable users to verify results and status and to initiate tests on-demand where relevant. Note that the complete module halts when a test fails.

Mitigation of Other Attacks

The module does not claim to mitigate any other attacks in FIPS-approved mode of operation.

SECURE OPERATION OF THE CRYPTOGRAPHIC SUPPORT LIBRARY

This section describes how to configure the module for FIPS-approved mode of operation. Operating the module without maintaining the following settings will remove the module from the FIPS-approved mode of operation.

FIPS Mode of Operation

The module allows operation in FIPS approved mode of operation via initialization of the service "CryptFacilityService", as described below, and by adhering to the guidelines in the section on *Operational Environment*.

Crypto Officer Guidance – Module Initialization and Configuration

The Crypto Officer must configure and enforce the following initialization procedures in order to operate in FIPS approved mode of operation:

1. Ensure that the service `CryptFacilityService` is initialized for FIPS approved mode of operation, which ensures self-tests are run upon module initialization.
    a. The module is initialized by using the sub-service `Giritech::CryptFacility::CryptFacilityService::getInstance().initialize(CryptFacilityService:: modeofoperation_fips)` of the of the `CryptFacilityService`.
2. Verify that the software version of the CryptFacility module is 1.0.485, using the sub-service `Giritech::CryptFacility::CryptFacilityService::getInstance().getVersionInfo` of the service `CryptFacilityService`.
3. Verify the hash of the module using the sub-service `Giritech::CryptFacility::CryptFacilityService::getInstance().getIntegrityHash` of the service `CryptFacilityService`. The hash value should be:
   `A00E2C493591FE3A58CD320C43DFF70691EA5E30`
4. Verify that the module is ready and operating in FIPS approved mode using the sub-services `Giritech::CryptFacility::CryptFacilityService::getInstance().isReady` and `Giritech::CryptFacility::CryptFacilityService::getInstance().isModeOfOperationFIPS` of the service `CryptFacilityService`.

By ensuring the calling daemons conforms to these requirements during the initialization procedure, the Crypto Officer can unambiguously verify that the complete module is operating in FIPS approved mode of operation.

Note that the Crypto Officer explicitly performs each of the steps above.

User Guidance

CryptFacility is not distributed as a standalone library and is only used in conjunction with the Giritech G/On solution. As such, there is no direct User Guidance.

DEFINITION LIST

AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
CIF	Client Identity Facility
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
D	Delete
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
EMC	ElectroMagnetic Compatibility
EMI	ElectroMagnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
G/On	The primary product line from Giritech that deploys the CryptFacility module
GPC	General Purpose Computer
GPOS	General Purpose Operating System
HMAC	Hashed MAC (see MAC)
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
PRNG	Pseudo-Random Number Generator
R	Read
RAM	Random Access Memory
RNG	Random Number Generator
SHA	Secure Hash Algorithm
TDES	Triple DES
USB	Universal Serial Bus
VPN	Virtual Private Network
W	Write