NetLib® Encryptionizer® Platform (NEP)

Module Version 2018.1.1.0

Security Policy

Non-Proprietary

FIPS 140-2 Level 1 Validation

June 18, 2019
Version 0.9

NetLib SECURITY

Table of Contents

1 Introduction 3
1.1 Acronyms and Abbreviations 5
2 NetLib® Encryptionizer® 6
2.1 Functional Overview 6
2.2 Module Description 7
2.3 Module Ports and Interfaces 7
3 Security Functions 9
4 FIPS Approved Mode of Operation 10
5 Roles, Services and Authentication 10
6 Physical Security 11
7 Cryptographic Keys and CSPs 12
8 Access Control 14
9 EMI/EMC 14
10 Self-Tests 15
11 Design Assurance 16
12 Mitigation of Other Attacks 16
13 References 16
14 Document Revision History 17

List of Tables

Table 1 – Cryptographic Module Security Requirements 3
Table 2 – Physical Interfaces and Logical 140-2 Interfaces 8
Table 3 – FIPS 140-2 Logical Interfaces 8
Table 4 – Module Approved Security Functions. 9
Table 5 - Roles and Services 11
Table 6 - Cryptographic Keys and CSPs 12
Table 7 – Access Control 14
Table 8 – Self Tests. 15

List of Figures

Figure 1 – High Level Functional View of the Cryptographic Module 6

1 Introduction

This document is the Security Policy for NetLib® Encryptionizer® Platform (hereon in referred to as NEP) 2018.1.1.0 cryptographic module. This Security Policy specifies the security rules under which the module shall operate to meet the requirements of FIPS 140-2 Level 1. It describes how the module functions to meet the FIPS requirements, and the actions that operators must take to maintain the security of the module. Third party applications may use NEP to help towards achieving FIPS 140-2 operation. However use of NEP in and of itself does not make a Third Party Application FIPS 140-2 Validated.

NetLib Security has productized NEP in several applications, including Encryptionizer for SQL Server. However anyone can use the Module in a Third Party application under license from NetLib Security.

This Security Policy describes the features and design of the Encryptionizer® cryptographic module using the terminology contained in the FIPS 140-2 specification. *FIPS 140-2, Security Requirements for Cryptographic Modules* specifies the security requirements that must be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information. The NIST Cryptographic Module Validation Program (CMVP) validates cryptographic modules to the FIPS 140-2 standard. Validated products are accepted by the Federal agencies of both the USA and Canada for the protection of sensitive or designated information.

The FIPS 140-2 standard and information on the CMVP can be found at http://csrc.nist.gov/groups/STM/cmvp. More information describing the Encryptionizer® can be found at http://www.netlibsecurity.com.

In this document, the NetLib® Encryptionizer® Platform 2018.1.1.0 is also referred to as the “Encryptionizer®”, “the driver”, the “cryptographic module”, the “Library”, or “the module”.

This Security Policy contains only non-proprietary information. All other documentation submitted for FIPS 140-2 conformance testing and validation is “NetLib®’ - Proprietary” and is releasable only under appropriate non-disclosure agreements.

The NetLib® Encryptionizer® cryptographic module meets the overall requirements applicable to Level 1 security for FIPS 140-2.

Table 1 – Cryptographic Module Security Requirements

Security Requirements Section	*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	2
Mitigation of Other Attacks	N/A
Overall Security Level	1

The cryptographic module is also designed to mitigate certain non-crypto related attacks, such as Process Injection Attacks. In addition, data is always encrypted in Windows System cache. Lastly, startup self-tests are also applied even to critical non-crypto related modules to ensure integrity.

1.1 Acronyms and Abbreviations

AES	Advanced Encryption Standard
APP	Application, such as MS SQL Server or MS Access
CBC	Cipher Block Chaining
CMVP	Cryptographic Module Validation Program
CSE	Communications Security Establishment
CSP	Critical Security Parameter
CTR	Counter Mode
ECB	Electronic Code Book
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
EK	Encryption Key
FCC	Federal Communication Commission
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
HMAC	Keyed-Hash Message Authentication Code
I/O	Input/Output
KMK	Key Management Key
MS	Microsoft
NIST	National Institute of Standards and Technology
PC	Personal Computer
POST	Power On Self Test
PUB	Publication
RAM	Random Access Memory
RFC	Request for Comment
SHA	Secure Hash Algorithm
SQL	Structured Query Language
UKMK	User-entered Key Management Key
USB	Universal Serial Bus

2 NetLib® Encryptionizer®

2.1 Functional Overview

The NetLib® Encryptionizer® Platform provides encryption of data stored in Application databases and backups. It can be deployed without programming and without adding any administrative overhead. The purpose of whole database encryption is to make a database unusable if it is stolen, copied, downloaded, lost, or otherwise improperly accessed. Data is never decrypted on disk, only in memory as requested by the Application (such as MS SQL Server or MS Access). Access permissions to the decrypted data are controlled by Encryptionizer®'s Administration Wizard. In addition, data is automatically encrypted before being written back to disk.

Features of the software include:

- Strong AES data encryption
- FIPS 140-2 power-on self tests and conditional tests.

Figure 1 shows how Encryptionizer® operates within the Database Access Layers. Data at rest (on disk) is encrypted. If stolen and moved to another machine without the decryption key, the data cannot be read. Only authorized human operators can freely access data with encryption being totally transparent to applications. The cryptographic module consists of only those libraries within the application that perform key management and encrypt and decrypt data.

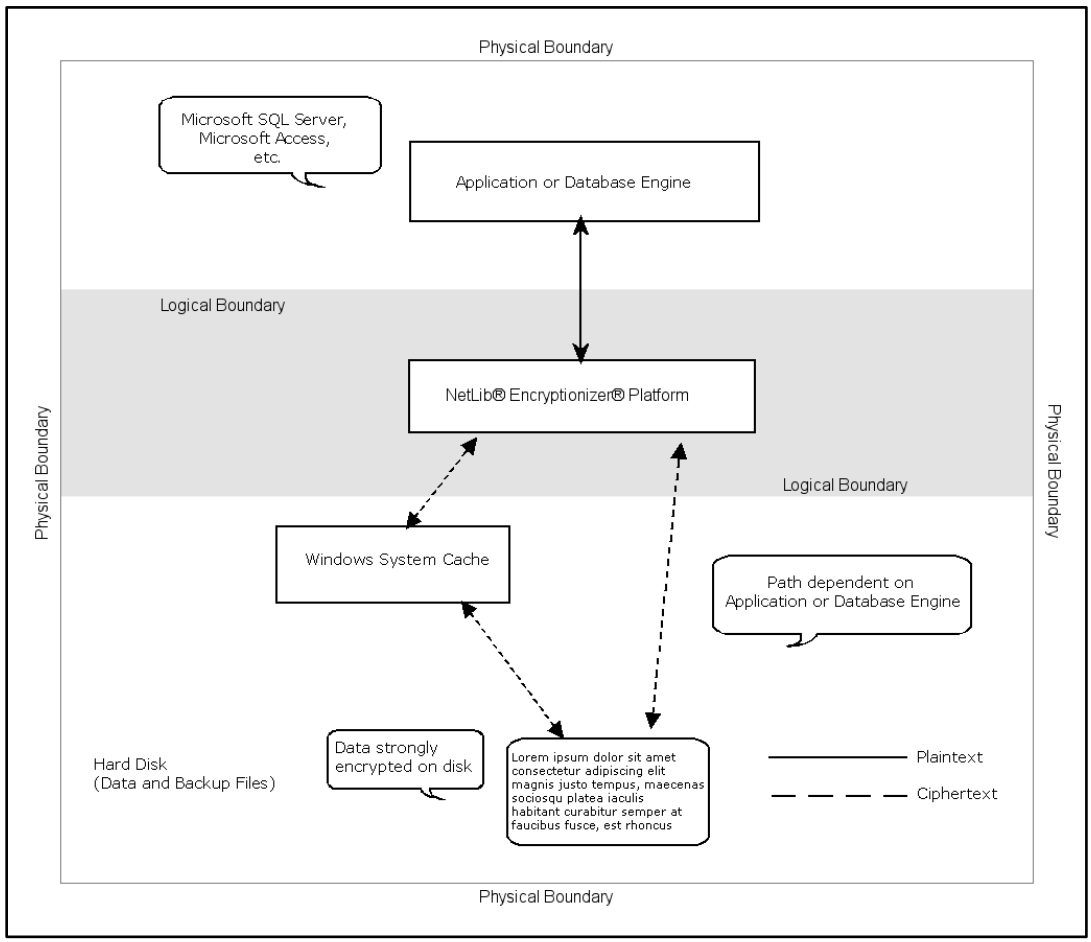

Figure 1 – High Level Functional View of the Cryptographic Module

2.2 Module Description

The Encryptionizer® cryptographic module is a FIPS level 1 multi-chip standalone cryptographic module consisting of a Software Library that executes on a general-purpose computer configured in single-user mode. Only the administrator account is enabled. All other user and guest accounts are disabled. Only a single user (e.g., MS Access or MS SQL Server) may access the module at any point in time. Multiple concurrent users (e.g., multiple SQL Server or Access processes) are not supported. The Application provides a Process ID Hash with each command that the module uses to restrict access to the single Application.

The module was tested with the following operational environments:

- Windows Server 2016 64-bit on Dell PowerEdge R210 II Server with Intel Xeon ES-2640 with/without AES-NI. (Single-user mode)
- Windows 10 64-bit on HP Pavilion Notebook PC with Intel Core I7-6500U with/without AES-NI. (Single-user mode)
- Windows 10 32-bit on HP Pavilion Notebook PC with Intel Core I7-6500U with/without AES-NI. (Single-user mode)

The module will also run in the following environments but was not tested by labs:

- Server Operating Systems: Windows 2003 Server and later, both 32-bit and 64-bit
- Desktop Operating Systems: Windows XP and later, both 32-bit and 64-bit

CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

The module provides data encryption and decryption services, key management services, software integrity services, a power up self-test and a conditional test assuring operators of a valid firmware state within the module.

The module consists of the following files for Windows 32-bit Operating Systems:

File	*File Version*
`C:\Windows\System32\nlem1402.dll`	2018.101.11.0
`C:\Windows\System32\drivers\nlem1402.sys`	2018.101.11.0

The module consists of the following files for Windows 64-bit Operating Systems:

File	*File Version*
`C:\Windows\System32\nlem1402.dll`	2018.501.11.0
`C:\Windows\System32\drivers\nlem1402.sys`	2018.501.11.0

2.3 Module Ports and Interfaces

The cryptographic module has four physical interfaces and four FIPS 140-2 logical interfaces. The physical ports have the functions described in Table 2. Where distinct logical interfaces share the same physical port, the system timing, software and hardware protocols, software APIs, and other controls logically separate and isolate these distinct categories of data from one another. The internal system bus acts as the physical path for clocking data into and out of the module. System synchronization and timing controls, and the protocol of the data ensure that logically distinct categories of data do not occupy the data path at the same time.

Table 2 – Physical Interfaces and Logical 140-2 Interfaces

Physical Interface	*FIPS 140-2 Logical Interface*
PC USB ports, PC network port, keyboard interface, mouse port, hard drive, floppy drive (optional), CDROM drive (optional), internal I/O ports.	Data input interface
PC USB ports, PC network port, hard drive, CDROM drive, internal I/O ports.	Data output interface
PC keyboard port, mouse port, network port, PC power button, internal I/O ports.	Control input interface
PC monitor.	Status output interface

The physical interfaces map to logical interfaces as described in Table 3.

Table 3 – FIPS 140-2 Logical Interfaces

Logical Interface	*Description*
Data input	The data input is: • All plaintext data entering the Encryptionizer® for the purpose of being encrypted and stored in the database. The Application provides these logical interfaces. • All ciphertext data entering the Encryptionizer® from the database for the purpose of being decrypted and output. The database API provides these logical interfaces.
Data output	The data output is: • All ciphertext data exiting the Encryptionizer® to the database. The database API provides these logical interfaces. • All plaintext data exiting the Encryptionizer® for use by the Application. The Application database API provides these logical interfaces.
Control input	The Encryptionizer® accepts control input from the following sources: • Commands passed from the Encryptionizer® user interface. • Parameters provided by the profile. • Parameters provided from the database header.
Status output	The status output consists of all messages either logged by or returned from the module and status messages returned to the Encryptionizer® user interface for viewing by the operator.

3 Security Functions

The Encryptionizer® cryptographic module implements the security functions described in Table 4:

Table 4 – Module Approved Security Functions.

CAVP Cert	*Algorithm*	*Standard*	*Mode/Method*	*Key Lengths*	*Use*
5529	AES	FIPS 197, SP 800-38A	CBC[1], ECB, CTR (internally derived)	128, 256 bits	Data Encryption / Decryption
4437	SHS	*FIPS 180-4*	SHA-1: Byte-oriented one-way hashing.	N/A	Message Digest: Used with HMAC for POST and Software Integrity Tests.
3682	HMAC	*FIPS 198-1*	HMAC-SHA1	512 bits	Message Authentication: POST and Software Integrity Tests.

Note 1: When encrypting a file with AES-CBC, the file is broken down into 512 byte blocks with each block being encrypted separately with an internally generated IV being partly reinitialized at the start of each block based on the offset in the file. This is done for performance purposes.

4 FIPS Approved Mode of Operation

The Encryptionizer® has only a FIPS Approved mode of operation that is restricted to performing only FIPS-approved cryptographic algorithms and security functions. The module does not have a non-approved mode.

The approved mode becomes active after the module has powered up and has passed the power-on self-test. The approved mode lets the user read and write data from and to an Application database. If the module is operational (meaning it has passed the power-on self-test), the module is operating in FIPS approved mode. The status message "FIPS ON" is displayed on any Encryptionizer® wizard splash screen. There is no Non-FIPS Mode.

The module must run on one of the platforms described in section 2.2. The system is configured in single-user mode by configuring the OS to provide only an Administrator (Windows) account. Disable any other administrative, guest, and user accounts during setup of the server.

When the module is installed, confirm the Encryptionizer® has the correct version number by clicking "**About"** on any Encryptionizer® wizard splash screen and checking that the version is 2018.1.1.0. Alternatively check the File Version of any of the files that are included in the Module.

5 Roles, Services and Authentication

The module supports two roles, as follows.

- crypto officer role
- user role

The crypto officer role is implicitly assumed by the operator installing the module and configuring it for use. Crypto officer operations consist of running the installation program to install NetLib® Encryptionizer® application software, setting up the initial user account, entering encryption keys, and uninstalling the software. Crypto officers must authenticate to the Windows operating system to install and configure the module for use.

The user role is implicitly assumed by the software Application. User operations consist of read and write operations from and to the database. Human users authenticate to the Application that is outside the boundary of the cryptographic module.

Multiple concurrent users are not supported. Only a single user may access the cryptographic module at any given point in time.

The module does not require any physical maintenance.

The module does not support authentication or identification for the Crypto-Officer or User role.

The module supports services that are available to crypto officers and users (the user is the Application accessing the module). All of the services are described in detail in the module's user documentation.

The crypto officer role is established by the operational rules of the module. The crypto officer role is assumed implicitly.

The crypto officer may install, configure (enter keys), and uninstall the module. The crypto officer may also view version information, run the power-up self-test (by starting the Encryptionizer®), and view status information (event logs).

The user role is also established by the operational rules of the module. The user role is implicitly assumed by the Application (sending read and write commands) when the module is running.

Table 5 shows the services available to the various roles. Encrypt and decrypt services delete the encryption key from memory (RAM) when the operation completes without modifying, disclosing, or substituting the key in any manner.

Table 5 - Roles and Services

Service	*Crypto Officer*	*User*
Install the module	●	
Power On Self Tests	●	
Generate Logs	●	●
Zeroization (Uninstall the module, reformat and overwrite, at least once, the hard drive.)	●	
Configure the module (Optionally enter a UKMK and / or set a policy rule)	●	
Enter UKMK from outside the module	●	
Enter EK from outside the module	●	
Encrypt data being written to the file		●
Decrypt data being read from the file		●
View version information	●	
Run Self-Test	●	
Show Status	●	

6 Physical Security

The FIPS 140-2 Physical Security requirements are not applicable to this module because the NetLib® Encryptionizer® Platform is a software only module.

7 Cryptographic Keys and CSPs

The following table identifies the cryptographic keys employed within the module.

Table 6 - Cryptographic Keys and CSPs

Keys/CSPs	*Generation*	*Storage*	*Input/Output*	*Zeroization*	*Description*
Encryption Key (EK)	N/A	A copy is stored in encrypted form (under the KMK) within the module profile that is outside the module boundary. In subsequent uses, this key is passed automatically from the profile to the module for use.	Manually entered by CO	Deleted from RAM after use. Zeroize this key according to IG 7.9 described in Note 1 below.	The EK (AES) key length depends on the options chosen by the crypto officer when entering the key. Key lengths are 128 and 256-bits. A crypto officer must enter the key twice to confirm the correct key is used. This key is used to encrypt and decrypt data written to or read from the database.
Key Management Key (KMK)	N/A	Hard coded in module by manufacturer	N/A	Deleted from RAM immediately after use. Zeroize this key according to IG 7.9 described in Note 1 below.	The KMK is an AES 256-bit key that the Encryptionizer® uses by default to encrypt and decrypt the module profile (the encryption key store). If the crypto officer enters a User-Entered Key Management Key (UKMK), the KMK encrypts the UKMK that is stored in the registry. In this case the KMK is not used to encrypt and decrypt the module profile. Note that this is not a FIPS key and only provides obfuscation of the EK that it encrypts. In order to claim true FIPS encryption of the profile and EK, user must assign a UKMK (see below).

Keys/CSPs	*Generation*	*Storage*	*Input/Output*	*Zeroization*	*Description*
User-Entered Key Management Key (UKMK)	N/A	Stored in the registry encrypted under the KMK.	Manually entered by CO	Deleted from RAM immediately after use. Zeroize this key according to IG 7.9 described in Note 1 below.	The UKMK is a 128-bit or 256-bit AES key that may be entered by a crypto officer. A crypto officer must enter the key twice to confirm the correct key is used. The UKMK is used only to encrypt and decrypt the module profile (the encryption key store).
Initialization Vector (IV)	N/A	Stored in plaintext in the header of file.	N/A	When user shreds the file	Used to initialize the first block during encryption in AES-CBC and AES-CTR modes.
HMAC Key	N/A	Hard coded in module by manufacturer	N/A	Deleted from RAM immediately after use. Zeroize this key according to IG 7.9 described in Note 1 below.	64 bytes HMAC key used for the Software Integrity Test.

Note 1: Perform zeroization by uninstalling the cryptographic module, reformatting the platform's hard drive and overwriting the platform's hard drive, at least once.

8 Access Control

Table 7 shows services that use or affect cryptographic keys or CSPs. For each service, the key or CSP is indicated along with the type of access.

R - The item is **read** or referenced by the service.
W - The item is **written** or updated by the service.
E - The item is **executed** by the service. (The item is used as part of a cryptographic service.)
D - The item is **deleted** by the service.
Z - The item is **zeroized** by the service.

Table 7 – Access Control

Authentication Data (Key or CSP)	*Service*	*Role[1]*	*Access Control*
Encryption Key (EK)	Encrypt data being written to the file	U	R,E,D
	Decrypt data being read from the file	U	R,E,D
	Enter EK from outside the module	CO	W
	Zeroization (Uninstall the module, reformat and overwrite, at least once, the hard drive).	CO	Z
	Configure the module (Optionally enter a UKMK and / or set a policy rule)	CO	R,E,D
Key Management Key (KMK) [2]	Configure the module (Optionally enter a UKMK and / or set a policy rule)	CO	R,E,D
	Encrypt data being written to the file	U	R,E,D
	Decrypt data being read from the file	U	R,E,D
	Zeroization (Uninstall the module, reformat and overwrite, at least once, the hard drive.	CO	Z
	Enter UKMK from outside the module	CO	W
User-Entered Key Management Key (UKMK) [3]	Configure the module (Optionally enter a UKMK and / or set a policy rule)	CO	R,E,D
	Encrypt data being written to the file	CO	R,E,D
	Decrypt data being read from the file	CO	R,E,D
	Zeroization (Uninstall the module, reformat and overwrite, at least once, the hard drive.	CO	Z
	Run Self-Test	CO	R,E
HMAC Key	Run Self-Test	CO	R,E
	Zeroization (Uninstall the module, format and overwrite at least once, the hard drive).	CO	Z
Initialization Vector (IV)	Encrypt data being written to the file	CO	R,E,D
	Decrypt data being read from the file	CO	R,E,D

[1] U indicates the service is available to a user role. CO indicates the service is available to a crypto officer role.
[2] If a UKMK is used, the KMK is used only to encrypt or decrypt the UKMK that is stored in the registry. The access control for these encryption and decryption operations is R, E, D.
[3] These associated services are available only if the crypto officer uses the UKMK.

9 EMI/EMC

The module is software that is intended to run on a GPC that conforms to the EMI/EMC requirements specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital devices, Class A. The module was tested on a GPC having a FCC DoC (Declaration of Conformity) meeting these requirements.

10 Self-Tests

The module performs power-on self-tests (POSTs) to verify the integrity and correct operational functioning of the cryptographic module. If the system fails a POST, it reports status indicating which failure occurred and transitions to an error state. The module does not contain any user data before or during the POST so it is impossible for the module to output user data in this state or a subsequent error state that halts module operation. The module outputs its status to the Windows Application Event log file in the event of a passed or failed power on self-test. Operators can run the POST on demand by stopping and restarting the containing application.

The Power On Self Tests (POSTs) follow the guidelines specified in the "FIPS Implementation Guidelines" (IG) section 9.10: "Power-Up Tests for Software Module Libraries".

When an Application or the Operating System attempts to load the Module into memory, Windows automatically transfers control to a startup procedure: DllMain for DLLs, and DriverMain for Kernel Mode Drivers. This transfer is automatic and does not require any user intervention. The POST routine performs the following tasks:

- Algorithm KAT: this is performed by testing encrypt, decrypt vector tests to know values stored in the Registry at Module installation.
- Module Integrity Test: this is performed by computing the HMAC of the Module files and comparing them to the known values stored in the Registration at Module installation.

If either of these tests fails, the Module outputs an entry to the Windows Application Event Log and then halt. The application attempting to load the Module will fail.

If both tests succeed, the Module outputs a "Success" entry to the Windows Application Event Log. The Application attempting to load the Module will be able to use the operations of the Module.

Table 8 summarizes the system self-tests.

Table 8 – Self Tests.

Self Test	*Description*
Mandatory power-up tests performed at power-up and on demand:	
Cryptographic Algorithm (Known Answer Tests) KATs	The AES cryptographic algorithms are tested using a “known answer” test to verify the operation of the function. The AES known answer tests perform both encryption and decryption. AES-ECB and AES-CBC are tested separately. HMAC and SHA-1 are tested using the Software Integrity Test.
Software Integrity Test	The module verifies the integrity of the software by generating an HMAC-SHA1 message authentication code for the Encryptionizer® and comparing the code against the expected values stored in the registry.
Conditional tests	
Manual Key Entry Test	Manually entered keys are entered twice and are compared to ensure the correct key is entered.

Known answer tests for encryption/decryption or hashing, function by encrypting (or hashing) a string for which the calculated output is known and stored within the cryptographic module. An encryption or hashing test passes when the freshly calculated output matches the expected (stored) value. A test fails when the calculated output does not match the expected value. The test then decrypts the ciphertext string. A decryption test passes when the freshly calculated output matches the plaintext value. A test fails when the calculated output does not match the plaintext value. Success and failure messages are written to the log file.

The software integrity test performs an HMAC/SHA1 calculation over each file comprising the cryptographic module. If the freshly calculated values match the pre-calculated stored values, the test passes. The module writes the success message to the log file and the module startup continues. If any freshly calculated value does not match the associated pre-calculated stored value, the test fails and the module startup exits. The module writes the failure message to the log file.

The manual key entry test compares key values that must be entered twice. If the values match, the key value is accepted. If the values do not match, the key value is rejected and the key must again be entered twice. Success and failure messages are written to the log file.

11 Design Assurance

Configuration Management – Source code and associated documentation and files are managed using a configuration management system. Each modification requires using a unique version identifier.

Delivery and Operation – Delivery and first time operation are controlled. Organizations purchasing the product must register to receive a unique authorization key to use the product.

Development – The module design follows a High Level Design specification that functionally defines the module, ports and interfaces and the purpose of each. Guidance is provided in the Security Policy and User Guide.

12 Mitigation of Other Attacks

The cryptographic module is not designed to mitigate specific crypto-related attacks such as differential power analysis or timing attacks. See introduction for mitigation of non-crypto-related attacks.

13 References

The following references are available at URL:
https://csrc.nist.gov/groups/computer-security-division/security-testing-validation-and-measurement.

- *FIPS PUB 140-2: Security Requirements for Cryptographic Modules*
- *FIPS 140-2 Annex A: Approved Security Functions*
- *Derived Test Requirements (DTR) for FIPS PUB 140-2, Security Requirements for Cryptographic Modules*
- *Secure Hash Standard (SHS)*, *FIPS Publication 180-4*
- *Advanced Encryption Standard (AES)*, *FIPS 197*
- *Recommendation for Block Cipher Modes of Operation: Methods and Techniques, SP 800-38A*
- *Keyed-Hash Message Authentication Code (HMAC)*, *FIPS 198-1*

14 Document Revision History

2018-03-09	Revised Figure 1, Updated module version numbers, Updated Tables 4, 6 and 8.
2018-03-15	Moved Document Revision History to end of document. Now referring to Module as NetLib Encryptionizer Platform Library.
2018-06-18	Replaced High Level Functional View (Figure 1)
	Added Initialization Vector (IV) description to Table 5
2018-07-24	Added other operational environments to Section 2.2
	Added statement as to how POSTs adhere to "Implementation Guidance for FIPS 140-2 and the Cryptographic Module Validation Program" section 9.10.
2018-09-02	Added IV details to Table 4.
	Replaced Table 5 with new format.
	Clarified Section 9: Self-Tests
2018-09-20	Updated Tables 6 and 7 with additional key information.
	Rearranged sections to improve readability.
2019-04-04	Updated processor listings in Section 2.2.
	Added a sentence about IG G.5 porting to the vendor affirmed environments section in Section 2.2.
	Table 4 hyperlinks updated.
2019-06-07	Referring to NetLib Encryptionizer Platform Library just as NetLib Encryptionizer Platform (NEP).