Acronis SCS

Acronis SCS Cryptographic Module

Software Version: 1.0

FIPS 140-2 Non-Proprietary Security Policy

FIPS Security Level: 1
Document Version: 0.10

Prepared for:

Acronis SCS

Acronis SCS
6370 E. Thomas Road, Suite 250
Scottsdale, AZ 85251
United States of America

Phone: +1 781 782 9000
www.acronisscs.com

Prepared by:

Corsec

Corsec Security, Inc.
13921 Park Center Road, Suite 460
Herndon, VA 20171
United States of America

Phone: +1 703 267 6050
www.corsec.com

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1 Purpose

This is a non-proprietary Cryptographic Module Security Policy for the Acronis SCS Cryptographic Module (software version: 1.0) from Acronis SCS. This Security Policy describes how the Acronis SCS Cryptographic Module meets the security requirements of Federal Information Processing Standards (FIPS) Publication 140-2, which details the U.S. and Canadian government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the National Institute of Standards and Technology (NIST) and the Canadian Centre for Cyber Security (CCCS) Cryptographic Module Validation Program (CMVP) website at https://csrc.nist.gov/projects/cryptographic-module-validation-program.

This document also describes how to run the module in a secure FIPS-Approved mode of operation. This policy was prepared as part of the Level 1 FIPS 140-2 validation of the module. The Acronis SCS Cryptographic Module is referred to in this document as Acronis SCS Crypto Module or the module.

This Security Policy and the other validation submission documentation were produced by Corsec Security, Inc. under contract to Acronis SCS.

1.2 References

This document deals only with operations and capabilities of the module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the module from the following sources:

- The Acronis SCS website (https://www.acronis.com/en-us/) contains information on the full line of products from Acronis SCS.

- The search page on the CMVP website (https://csrc.nist.gov/Projects/cryptographic-module-validation-program/Validated-Modules/Search) can be used to locate and obtain vendor contact information for technical or sales-related questions about the module.

1.3 Document Organization

The Security Policy document is organized into two (2) primary sections. Section 2 provides an overview of the validated module. This includes a general description of the capabilities and the use of cryptography, as well as a presentation of the validation level achieved in each applicable functional area of the FIPS standard. It also provides high-level descriptions of how the module meets FIPS requirements in each functional area. Section 3 documents the guidance needed for the secure use of the module, including initial setup instructions and management methods and policies.

2. Acronis SCS Cryptographic Module

2.1 Overview

Acronis SCS is a leading backup software, disaster recovery, and secure data access provider to consumers, small-medium businesses, and enterprises. Acronis solutions include physical, virtual, and cloud server backup software, storage management, secure file sharing, and system deployment. Powered by the Acronis AnyData Engine, Acronis products provide easy, complete, and safe solutions for data in local, remote, cloud, and mobile devices and hybrid IT[1] environments.

The Acronis Cyber Backup SCS solution includes the following software components:

- Management Server – Acronis Management Server is composed of a number of services responsible for management functions of the Acronis Cyber Backup SCS solution and for providing the Web UI[2]. These services manage agents, groups, and backup plans; send notifications; collect data; and build and save reports.

- Backup agents – Acronis backup agents are installed as a number of services responsible for performing the specific backup, recovery, replication, and other data-manipulation tasks on the machines being protected. They are typically installed on each device that requires protection and then added to the Acronis Management Server. However, Acronis agents are able to work completely independently and do not require constant communication with the Management Server to run their scheduled backup operations. It is also possible to install an isolated agent and forego the Management Server entirely; in this case, management of the agent is performed via the command line.

  Different agent types are used to protect different data sources, but the agents all share the same architecture, communication protocols, and the vast majority of the functionality. There are agents currently available for the following data sources:
  - Windows
  - Linux
  - Mac
  - VMware
  - Hyper-V
  - Xen
  - Exchange
  - Office 365
  - Oracle

Figure 1 below shows a typical on-premise deployment for the Acronis Cyber Backup SCS solution software components.

[1] IT – Information Technology
[2] UI – User Interface

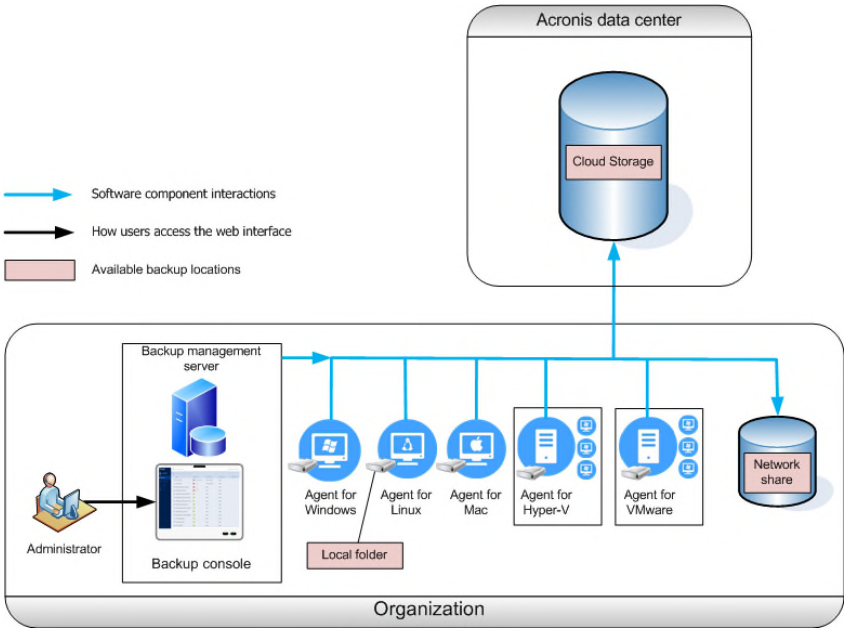

Figure 1 – Typical On-Premise Deployment

The Acronis SCS Cryptographic Module is a component of the Acronis Cyber Backup Advanced SCS Server and Acronis Cyber Backup Standard SCS Agent software (versions 12.5 and later). It provides the underlying cryptographic functionality necessary to support the use of secure communications protocols, encrypted backups, and secure file sharing.

The Acronis SCS Cryptographic Module is validated at the FIPS 140-2 section levels shown in Table 1.

Table 1 – Security Level per FIPS 140-2 Section

Section	Section Title	Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services, and Authentication	2
4	Finite State Model	1
5	Physical Security	N/A
6	Operational Environment	1
7	Cryptographic Key Management	1
8	EMI/EMC[3]	1
9	Self-tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A

[3] EMI/EMC – Electromagnetic Interference / Electromagnetic Compatibility

2.2 Module Specification

The Acronis SCS Cryptographic Module is a software module with a multiple-chip standalone embodiment. The overall security level of the module is 1. The Acronis SCS Cryptographic Module is based on the OpenSSL FIPS canister (OpenSSL FIPS Object Module version 2.0.16). The module is compiled into object form (*fipscanister.o* on Linux-based systems or *fipscanister.lib* on Windows-based systems) and then linked to an instance of the standard OpenSSL cryptographic library at build-time. This procedure creates a FIPS-capable OpenSSL library that can then be linked to a calling application.

The module was tested and found to be compliant with FIPS 140-2 requirements on the platforms and environments listed in Table 2.

Table 2 – Tested Configurations

Operating System	Processor	Platform	PAA[4]
Windows Server 2016	Intel Xeon E-2136	Dell PowerEdge R340	✓
Windows 10	Intel Core i7-8650U	Dell Latitude 7490	✓
RHEL[5] 7.6	Intel Core i5-8350U	Dell Latitude 7390	✓
RHEL 7.6	Intel Core i5-8350U	Dell Latitude 7390	

PAA testing was performed using the processor-resident AES-NI instruction set.

Additionally, per FIPS Implementation Guidance G.5, the CMVP allows for the porting of a validated software module to an operational environment which was not included as part of the validation testing. The vendor affirms that the module maintains its validation compliance when operating on a platform with one of the following operational environments[6]:

- RHEL 6.6 running on an Intel Core i3-8100
- RHEL 6.6 running on an Intel Core i5-8500
- RHEL 7.1 running on an Intel Core i3-8100
- RHEL 7.1 running on an Intel Core i5-8500
- RHEL 7.6 running on an Intel Core i3-8100
- RHEL 7.6 running on an Intel Core i5-8500
- 64-bit Windows 2008 R2 running on an Intel Core i3-8100
- 64-bit Windows 2008 R2 running on an Intel Core i5-8500
- 64-bit Windows 2012 R2 running on an Intel Core i3-8100
- 64-bit Windows 2012 R2 running on an Intel Core i5-8500
- 64-bit Windows 8.1 Pro running on an Intel Core i5-5300U
- 64-bit Windows 10 Pro running on an Intel Core i5-5300U
- 64-bit Windows 2016 running on an Intel Core i5-5300U
- 64-bit Windows 2019 running on an Intel Core i5-5300U

[4] PAA – Processor Algorithm Accelerators
[5] RHEL – Red Hat Enterprise Linux
[6] 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 environment not listed on the validation certificate.

Note that if porting to an untested platform, there is no assurance of the minimum strength of generated keys (as the NDRNG is outside the module's logical boundary).

2.2.1 Physical Cryptographic Boundary

As a software cryptographic module, the module has no physical components. Therefore, the physical boundary of the cryptographic module is defined by the hard enclosure around the host server on which it runs.

The host server hardware consists of a motherboard, a Central Processing Unit (CPU), random access memory (RAM), read-only memory (ROM), hard disk(s), hardware case, power supply, and fans. Other devices may be attached to the hardware appliance such as a monitor, keyboard, mouse, DVD[7] drive, printer, video adapter, audio adapter, or network adapter. Please see Figure 2 for the host server block diagram and physical cryptographic boundary.

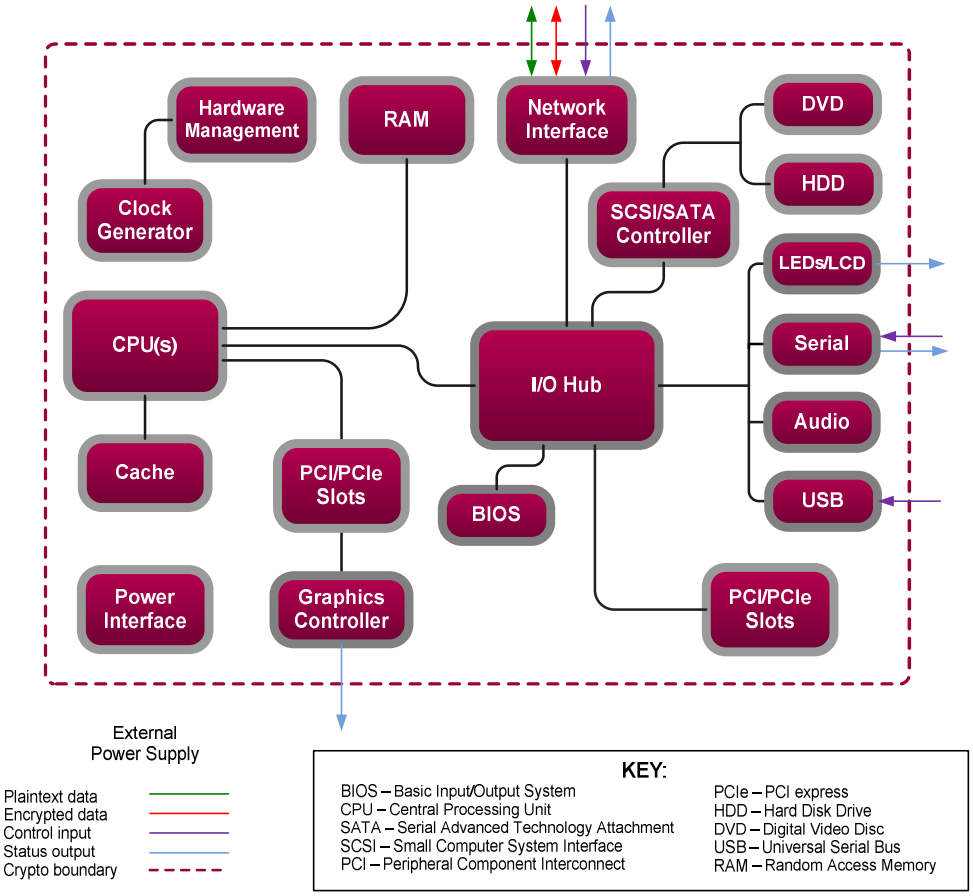

Figure 2 – Host Server Physical Block Diagram

2.2.2 Logical Cryptographic Boundary

The module provides cryptographic services for the Acronis Cyber Backup Advanced SCS Server and Acronis Cyber Backup Standard SCS Agent software (versions 12.5 and later). In this document, those applications will be

[7] DVD – Digital Versatile Disc

referred to collectively as the "calling application". The module is used by the calling application to provide or support symmetric and asymmetric cipher operation, signature generation and verification, hashing, cryptographic key generation, random number generation, message authentication functions, and secure key agreement/key exchange protocols. The module is entirely contained within the physical cryptographic boundary described in Section 2.2.1.

Figure 3 below shows the logical block diagram of the module executing in memory and its interactions with surrounding software components, as well as the module's logical cryptographic boundary.

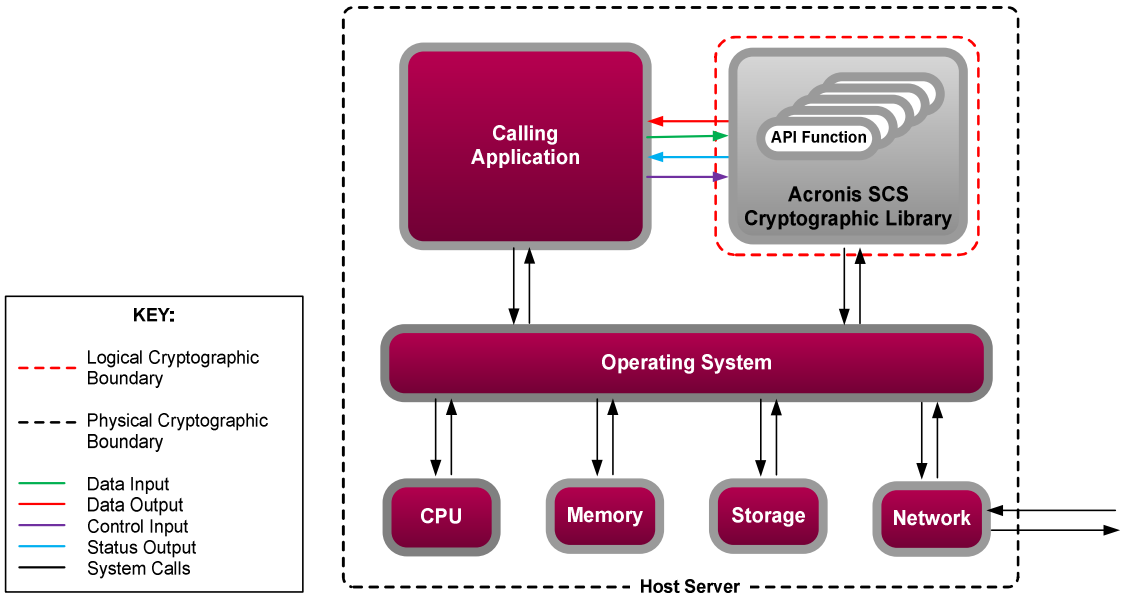

Figure 3 – Module Block Diagram (with Logical Cryptographic Boundary)

2.2.3 Algorithm Implementations

The module implements the FIPS-Approved algorithms listed in Table 3 below.

Table 3 – FIPS-Approved Cryptographic Algorithms

Certificate Number	Algorithm	Standard	Modes / Methods	Key Lengths / Curves / Moduli	Use
#C1351	AES[8]	FIPS PUB 197 NIST SP 800-38A	CBC[9], CFB1[10], CFB8, CFB128, ECB[11], OFB[12]	128, 192, 256	encryption/decryption
		FIPS PUB 197 NIST SP 800-38A	CTR[13]	128, 192, 256	encryption

[8] AES – Advance Encryption Standard
[9] CBC – Cipher Block Chaining
[10] CFB – Cipher Feedback
[11] ECB – Electronic Code Book
[12] OFB – Output Feedback
[13] CTR – Counter

Certificate Number	Algorithm	Standard	Modes / Methods	Key Lengths / Curves / Moduli	Use
		NIST SP[14] 800-38B	CMAC[15]	128, 192, 256	generation/verification
		NIST SP 800-38C	CCM[16]	128, 192, 256	encryption/decryption
		NIST SP 800-38D	GCM[17]	128, 192, 256	encryption/decryption
Vendor Affirmed	CKG[18]	NIST SP 800-133rev1	-	-	symmetric key generation
#C1351	CVL[19]	NIST SP 800-56Arev2	ECC CDH[20] primitive	P-256, P-384, P-521	shared secret computation
#C1351	DRBG[21]	NIST SP 800-90Arev1	CTR-based	128, 192, 256-bit AES	deterministic random bit generation
			Hash-based	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	deterministic random bit generation
			HMAC-based	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	deterministic random bit generation
#C1351	DSA[22]	FIPS PUB 186-4	-	2048/224, 2048/256, 3072/256	key pair generation
			SHA[23]-224	2048/224	PQG generation
			SHA2-256, SHA2-384, SHA2-512	2048/224, 2048/256, 3072/256	PQG generation
			SHA-1	1024/160	PQG verification
			SHA2-224	1024/160, 2048/224	PQG verification
			SHA2-256, SHA2-384, SHA2-512	1024/160, 2048/224, 2048/256, 3072/256	PQG verification
			SHA2-224, SHA2-256, SHA2-384, SHA2-512	2048/224, 2048/256, 3072/256	digital signature generation
			SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	1024/160, 2048/224, 2048/256, 3072/256	digital signature verification
#C1351	ECDSA[24]	FIPS PUB 186-4	-	B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	key pair generation
			-	B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	key pair verification

[14] SP – Special Publication
[15] CMAC – Cipher-based Message Authentication Code
[16] CCM – Counter with CBC-MAC
[17] GCM – Galois/Counter Mode
[18] CKG – Cryptographic Key Generation
[19] CVL – Component Validation List
[20] ECC CDH – Elliptic Curve Cryptography Cofactor Diffie-Hellman
[21] DBRG – Deterministic Random Bit Generator
[22] DSA – Digital Signature Algorithm
[23] SHA – Secure Hash Algorithm
[24] ECDSA – Elliptic Curve Digital Signature Algorithm

Certificate Number	Algorithm	Standard	Modes / Methods	Key Lengths / Curves / Moduli	Use
			SHA-224, SHA-256, SHA-384, SHA-512	B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	digital signature generation
			SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	digital signature verification
#C1351	HMAC[25]	FIPS PUB 198-1	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	KS<BS, KS=BS, KS>BS	message authentication
#C1351	RSA[26]	FIPS PUB 186-2	ANSI[27] X9.31	1024, 1536, 2048, 3072, 4096 (SHA-1, SHA2-256, SHA2-384, SHA2-512)	digital signature verification
			PKCS1.5, PSS	1024, 1536, 2048, 3072, 4096 (SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512)	digital signature verification
		FIPS PUB 186-4	-	2048, 3072	key pair generation
			ANSI X9.31	2048, 3072 (SHA2-256, SHA2-384, SHA2-512)	digital signature generation
				1024, 2048, 3072 (SHA-1, SHA2-256, SHA2-384, SHA2-512)	digital signature verification
			PKCS1.5, PSS	2048, 3072 (SHA2-224, SHA2-256, SHA2-384, SHA2-512)	digital signature generation
				1024, 2048, 3072 (SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512)	digital signature verification
#C1351	SHS[28]	FIPS PUB 180-4	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	-	message digest
#C1351	Triple-DES[29]	NIST SP 800-67rev2	CBC, CFB1, CFB8, CFB64, ECB, OFB	Keying option 1	encryption/decryption
		NIST SP 800-38B	CMAC	Keying option 1	generation/verification

***NOTE**: Per NIST SP 800-131Arev2, SHA-1 and 1024-bit keys for DSA and RSA digital signature verification operations are allowed for legacy use only. Likewise, the curves B-163, K-163, and P-192 for ECDSA digital signature verification operations are allowed for legacy use only.*

The vendor affirms the following cryptographic security methods:

- Cryptographic key generation – As per *NIST SP 800-133rev1*, the module uses its FIPS-Approved DRBGs to generate cryptographic keys. The resulting symmetric key or generated seed is an unmodified output from

[25] HMAC – (keyed-) Hashed Message Authentication Code
[26] RSA – Rivest Shamir Adleman
[27] ANSI – American National Standards Institute
[28] SHS – Secure Hash Standard
[29] DES – Data Encryption Standard

the DRBG. When built, linked, and installed as described in section 3.1 below, the module's DRBG is seeded via entropy generated from the RDRAND instruction on the host server's Intel processor, which is external to the module.

The module implements the non-Approved but allowed algorithms shown in Table 4 below.

Table 4 – Allowed Algorithms

Algorithm	Caveat	Use
RSA	key establishment methodology provides between 112 and 256 bits of encryption strength	key wrapping

The module implements the non-Approved algorithms listed in Table 5 below (these algorithms shall not be used in the Approved mode of operation).

Table 5 – Non-Approved Cryptographic Algorithms

Algorithm	Standard	Modes / Curves / Moduli
AES-XTS[30,31,32] (non-compliant)	NIST SP 800-38E	encryption/decryption (128 and 256)
RNG[33]	ANSI X9.31	-
DSA (non-compliant)	FIPS PUB 186-2	PQG Gen, KeyGen, SigGen
	FIPS PUB 186-4	PQG Gen (1024 with all SHA sizes, 2048, 3072 with SHA-1)
		KeyGen (1024 with all SHA sizes, 2048, 3072 with SHA-1)
		SigGen (1024 with all SHA sizes, 2048, 3072 with SHA-1)
ECC CDH primitive (non-compliant)	NIST SP 800-56A	Curves P-192, K-163, and B-163
EC Diffie-Hellman (non-compliant)	NIST SP 800-56A	Curves P-192, K-163, and B-163
ECDSA (non-compliant)	FIPS PUB 186-2	PKG and SigGen
	FIPS PUB 186-4	PKG (P-192, K-163, and B-163)
		SigGen (P-192, K-163, B-163 with SHA-1 and all SHA-2 sizes)
		SigVer (K-163, B-163 with SHA-1 and all SHA-2 sizes)
RSA (non-compliant)	FIPS PUB 186-2	KeyGen, SigGen
	FIPS PUB 186-4	KeyGen (1024, 1536 with all SHA sizes)
		SigGen (1024, 1536 with all SHA sizes)
		SigGen (4096 and greater)

[30] XTS – XEX-Based Tweaked-Codebook Mode with Ciphertext Stealing
[31] XEX – XOR-Encrypt-XOR
[32] XOR – Exclusive Or
[33] RNG – Random Number Generator

2.2.4 Modes of Operation

The module supports two modes of operation: Approved and Non-approved. The module will be in FIPS-Approved mode when all power up self-tests have completed successfully, and only Approved algorithms are invoked. See Table 3 and Table 4 above for a list of the Approved and allowed algorithms.

The module can alternate service-by-service between Approved and non-Approved modes of operation. The module will switch to the non-Approved mode upon execution of a non-Approved service. The module will switch back to the Approved mode upon execution of an Approved service.

Table 6 below lists the services available in the non-Approved mode of operation.

Table 6 – Non-Approved Services

Service	Operator		Security Function(s)
	CO	User	
Perform symmetric encryption	✓	✓	AES-XTS (non-compliant)
Perform symmetric decryption	✓	✓	AES-XTS (non-compliant)
Generate random number	✓	✓	ANSI X9.31 RNG
Generate domain parameters	✓	✓	DSA (non-compliant)
Generate asymmetric key pair	✓	✓	DSA (non-compliant) ECDSA (non-compliant) RSA (non-compliant)
Generate signature	✓	✓	DSA (non-compliant) ECDSA (non-compliant) RSA (non-compliant)
Compute shared secret	✓	✓	ECC CDH primitive (non-compliant)
Perform key agreement	✓	✓	Elliptic Curve Diffie-Hellman (non-compliant)

2.3 Module Interfaces

The module isolates communications to logical interfaces that are defined in the software as an API[34]. The API interface is mapped to the following four logical interfaces:

- Data Input
- Data Output
- Control Input
- Status Output

The module's physical boundary features the physical ports of a host server. The module's manual controls, physical indicators, and physical, logical, and electrical characteristics are those of the host server. The module's logical interfaces are at a lower level in the software. The physical data and control input through physical

[34] API – Application Programming Interface

mechanisms is translated into the logical data and control inputs for the software module. A mapping of the FIPS 140-2 logical interfaces, the physical interfaces, and the module interfaces can be found in Table 7.

Table 7 – FIPS 140-2 Logical Interface Mappings

FIPS Interface	Physical Interface	Module Interface (API)
Data Input	USB[35] ports (keyboard, mouse, data), network interface, serial ports	The API calls that accept input data for processing through their arguments.
Data Output	Graphics controller, USB ports, network interface, serial ports	The API calls that return, by means of their return codes or arguments, generated or processed data back to the caller.
Control Input	USB ports (keyboard, mouse), network interface, serial ports	The API calls that are used to initialize and control the operation of the module.
Status Output	Graphic controller, network interface, serial ports, Audio ports, LCD[36]/LED[37]s	Return values for API calls.
Power Input	AC[38] power socket	-

2.4 Roles, Services, and Authentication

The sections below describe the module's authorized roles, services, and operator authentication methods.

2.4.1 Authorized Roles

There are two authorized roles that module operators may assume: Crypto Officer (CO) role and a User role. Since all services are available to each role, operators explicitly assume both roles upon successful authentication. The module does not allow multiple concurrent operators in the FIPS-Approved mode of operation. As per section 6.1 of the *Implementation Guidance for FIPS PUB 140-2 and the CMVP*, the calling application that loads the module is its only operator.

2.4.2 Operator Services

Descriptions of the services available are provided in Table 8 below. Please note that the keys and Critical Security Parameters (CSPs) listed in the table indicate the type of access required using the following notation:

- R – Read: The CSP is read.
- W – Write: The CSP is established, generated, modified, or zeroized.
- X – Execute: The CSP is used within an Approved or Allowed security function or authentication mechanism.

Table 8 – Mapping of Operator Services to Inputs, Outputs, CSPs, and Type of Access

[35] USB – Universal Serial Bus
[36] LCD – Liquid Crystal Display
[37] LED – Light Emitting Diode
[38] AC – Alternating Current

Service	Operator		Description	Input	Output	CSP and Type of Access
	CO	User				
Initialize	✓	✓	Perform initialization of the module	API call parameters	Status	Operator password – RX
Run self-test on demand	✓	✓	Performs power-up self-tests	API call parameters	Status	None
Show status[39]	✓	✓	Returns the current mode of the module	None	Status	None
Zeroize	✓	✓	Zeroizes and de-allocates memory containing sensitive data	API call parameters; power cycle; unload module	None	AES key – W AES CMAC key – W AES GCM key – W AES GCM IV – W Triple-DES key – W Triple-DES CMAC key – W HMAC key – W RSA private/public key – W DSA private/public key – W ECDSA private/public key – W DRBG seed – W DRBG entropy – W DRBG 'C' value – W DRBG 'V' value – W DRBG 'Key' value – W
Generate random number	✓	✓	Returns the specified number of random bits to the calling application	API call parameters	Status, random bits	DRBG seed – WRX DRBG entropy – RX DRBG 'C' value – WRX DRBG 'V' value – WRX DRBG 'Key' value – WRX
Generate message digest	✓	✓	Compute and return a message digest using SHS algorithms	API call parameters, message	Status, hash	None
Generate keyed hash	✓	✓	Compute and return a message authentication code	API call parameters, key, message	Status, hash	HMAC key – RX
Generate symmetric digest	✓	✓	Compute and return a cipher message authentication code	API call parameters, key, message	Status, hash	AES CMAC key – RX Triple-DES CMAC key – RX
Generate symmetric key	✓	✓	Generate and return the specified type of symmetric key (Triple-DES or AES)	API call parameters	Status, key	AES key – W Triple-DES key – W
Perform symmetric encryption	✓	✓	Encrypt plaintext using supplied key and algorithm specification (Triple-DES or AES)	API call parameters, key, plaintext	Status, ciphertext	AES key – RX Triple-DES key – RX

[39] Running the 'show status' command on the module will, along with its operating status, return the version of the module. The value returned by the module will be version "2.0.16". This version is equivalent to version "1.0" of the module.

Service	Operator		Description	Input	Output	CSP and Type of Access
	CO	User				
Perform symmetric decryption	✓	✓	Decrypt ciphertext using supplied key and algorithm specification (Triple-DES or AES)	API call parameters, key, ciphertext	Status, plaintext	AES key – RX Triple-DES key – RX
Perform authenticated symmetric encryption	✓	✓	Encrypt plaintext using supplied AES GCM key and IV	API call parameters, key, plaintext	Status, ciphertext	AES GCM key – RX AES GCM IV – RX
Perform authenticated symmetric decryption	✓	✓	Decrypt ciphertext using supplied AES GCM key and IV	API call parameters, key, ciphertext	Status, plaintext	AES GCM key – RX AES GCM IV – RX
Generate asymmetric key pair	✓	✓	Generate and return the specified type of asymmetric key pair (RSA, DSA, or ECDSA)	API call parameters	Status, key pair	RSA public key – W RSA private key – W DSA public key – W DSA private key – W ECDSA public key – W ECDSA private key – W
Verify asymmetric key pair	✓	✓	Verify the asymmetric key pair (RSA, DSA, or ECDSA)	API call parameters	Status, key pair	RSA public key – RX RSA private key – RX DSA public key – RX DSA private key – RX ECDSA public key – RX ECDSA private key – RX
Compute shared secret	✓	✓	Compute shared secret (ECDH)	API call parameter	Status, key components	ECDH public component – WX ECDH private component – WX
Generate signature	✓	✓	Generate a signature for the supplied message using the specified key and algorithm (RSA, DSA, or ECDSA)	API call parameters, key, message	Status, signature	RSA private key – RX DSA private key – RX ECDSA private key – RX
Verify signature	✓	✓	Verify the signature on the supplied message using the specified key and algorithm (RSA, DSA, or ECDSA)	API call parameters, key, signature, message	Status	RSA public key – RX DSA public key – RX ECDSA public key – RX

2.4.3 Authentication

The module supports role-based authentication. As the module's only operator, the calling application explicitly assumes the CO or User role by passing the appropriate password to the module. Valid password values are specified and converted to HMAC digests at build-time by the Acronis SCS development team. The password digests and associated HMAC key are then built into the module firmware image.

At runtime, the module's default entry point invokes the `FIPS_module_mode_set()` function with a password argument. Internally, this function generates the HMAC digest for the specified password and checks to see if it matches either of the pre-built digests. If there is no match, then the authentication attempt fails.

Any attempt to authenticate with an invalid password will result in an immediate and permanent failure condition, rendering the module unable to enter the FIPS mode of operation, even with subsequent use of a correct password.

Authentication data is loaded into the module in digest form during the module build process and otherwise cannot be accessed.

The minimum password length is 16 characters. The password character space comprises the full ASCII character set of 256 characters. The strength of the authentication mechanism is as follows:

- For each attempt to use the authentication mechanism, the probability that a random attempt will succeed or a false acceptance will occur is $1/256^{16}$ (which is less than 1/1,000,000).

- While the module permanently disables further authentication attempts after a single failure, the host server can be restarted, and authentication can be retried. Assuming a restart time of 45 seconds, a maximum of two authentication attempts are possible within one minute. For multiple attempts to use the authentication mechanism during a one-minute period, the probability that a random attempt will succeed or a false acceptance will occur is $2/256^{16}$ (which is less than 1/100,000).

2.5 Physical Security

The cryptographic module is a software module and does not include physical security mechanisms. Therefore, per section G.3 of the FIPS Implementation Guidance, requirements for physical security are not applicable.

2.6 Operational Environment

The process and memory management functionality of host device's operating system prevents unauthorized access to plaintext private and secret keys, CSPs, and intermediate key generation values by external processes during module execution. The module only allows access to CSPs through its well-defined API. Processes that are spawned by the firmware module are owned by the module and are not owned by external processes.

2.7 Cryptographic Key Management

The module supports the CSPs listed below in Table 9.

Table 9 – Cryptographic Keys, Cryptographic Key Components, and CSPs

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
AES key	128, 192, 256-bit AES key (CBC, CFB1, CFB8, CFB128, CTR, ECB, OFB modes)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Encryption, decryption
AES CCM key	128, 192, 256-bit AES key (CCM mode)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Encryption, decryption
AES GCM key	128, 192, 256-bit AES key (GCM mode)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Encryption, decryption
AES GCM IV[40]	96-bit value	Generated internally randomly via Approved DRBG[41]	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Initialization vector for AES GCM
AES CMAC key	128, 192, 256-bit AES key (CMAC mode)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	MAC generation and verification

[40] IV – Initialization Vector

[41] The IV generation method complies with technique #2 in section A.5 of the *Implementation Guidance for FIPS PUB 140-2 and the CMVP*.

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
Triple-DES key	168-bit Triple-DES key (CBC, CFB1, CFB8, CFB64, ECB, OFB modes)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Encryption, decryption
Triple-DES CMAC key	168-bit Triple-DES key (CMAC mode)	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature generation and verification
HMAC key	variable-bit HMAC key	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Message authentication with SHS
RSA private key	2048 or 3072-bit RSA key	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature generation, decryption
RSA public key	1024, 2048, or 3072-bit RSA key	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature verification, encryption

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
DSA private key	2048 or 3072-bit DSA key	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature generation
DSA public key	1024, 2048, or 3072-bit DSA key	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature verification
ECDSA private key	NIST-defined P/B/K-curves	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature generation

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
ECDSA public key	NIST-defined P/B/K-curves	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Signature verification
ECDH private component	NIST-defined P/B/K-curves	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Shared secret computation in key agreement

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
ECDH public component	NIST-defined P/B/K-curves	Internally generated via Approved DRBG OR Input in plaintext via API call parameter	Output in plaintext	Keys are not persistently stored by the module	Unload module; API call; Remove power	Shared secret computation in key agreement
DRBG entropy[42]	256-bit value	Externally generated and input in plaintext via API call parameter	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Entropy material for SP 800-90A DRBGs
DRBG seed	Random data – 440 or 880 bits	Generated internally using nonce along with DRBG entropy input	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Seeding material for DRBGs
DRBG 'C' value	Internal DRBG state value	Internally generated	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Used for Hash_DRBG
DRBG 'V' value	Internal DRBG state value	Internally generated	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Used for Hash_DRBG, HMAC_DRBG, and CTR_DRBG
DRBG 'Key' value	Internal DRBG state value	Internally generated	Never	Keys are not persistently stored by the module	Unload module; API call; Remove power	Used for HMAC_DRBG and CTR_DRBG
Operator password	16-character (minimum) string	Loaded into the module at build-time	Never	Authentication data is stored in digest form as part of the module's binary image	Uninstall module from host server and overwrite storage media	Used for authenticating to the module

[42] As entropy is provided to the module via the calling application, there is no assurance of the minimum strength of generated keys.

2.8 EMI / EMC

The Acronis SCS Crypto Module was tested on the servers listed in Section 2.6 above. These servers have been found conformant to the following EMI/EMC requirements specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital Devices:

- Dell PowerEdge R340 rack server – Class A (business use)
- Dell Latitude 7390 laptop – Class B (home use)
- Dell Latitude 7490 laptop – Class B (home use)

2.9 Self-Tests

Cryptographic self-tests are performed by the module when the module is first powered up and loaded into memory as well as when a random number or asymmetric key pair is created. The following sections list the self-tests performed by the module, their expected error status, and the error resolutions.

2.9.1 Power-Up Self-Tests

The module performs the following self-tests at power-up:

- Software integrity check (using HMAC SHA-1)
- Algorithm implementation tests
    - AES-ECB encrypt and decrypt KATs[43]
    - AES-CCM encrypt and decrypt KATs
    - AES-GCM encrypt and decrypt KATs
    - AES-CMAC KAT
    - Triple-DES encrypt and decrypt KATs
    - Triple-DES CMAC KAT
    - HMAC SHA-1/224/256/384/512 KATs
    - RSA signature generation/verification KAT
    - DSA pairwise consistency test
    - ECDSA pairwise consistency test
    - CTR_DRBG KAT
    - Hash DRBG KAT
    - HMAC DRBG KAT
    - ECC CDH Primitive “Z” Computation KAT

Note that the HMAC KATs utilize (and thus test) the full functionality of the with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 algorithm implementations; thus, per section 9.2 of the *Implementation Guidance for FIPS PUB 140-2 and the CMVP*, no independent KATs for SHA-1 and SHA-2 implementations are required.

2.9.2 Conditional Self-Tests

The module performs the following conditional self-tests:

- Continuous RNG[44] Test for the DRBG

[43] KAT – Known Answer Test
[44] RNG – Random Number Generator

- Continuous RNG Test for the NDRNG
- RSA pairwise consistency test for sign/verify
- RSA pairwise consistency test for encrypt/decrypt
- DSA pairwise consistency test
- ECDSA pairwise consistency test

2.9.3 Critical Functions Self-Tests

The module performs the following critical functions tests at module power-up:

- DRBG (Hash, HMAC, CTR) Instantiate Test
- DRBG (Hash, HMAC, CTR) Generate Test
- DRBG (Hash, HMAC, CTR) Reseed Test
- DRBG (Hash, HMAC, CTR) Uninstantiate Test

2.9.4 Self-Test Failure Handling

If any self-test fails, the module will enter a critical error state and an internal global error flag `FIPS_selftest_fail` is set which prevents the invocation of any cryptographic function calls. The module can only recover from the critical error state by restarting the calling application or rebooting the host server (thus restarting the module) and passing all power-up self-tests.

If these recovery methods do not result in the successful execution of the power-up self-tests, then the module will not be able to resume normal operations, and the CO should contact Acronis SCS for assistance.

2.10 Mitigation of Other Attacks

This section is not applicable. The module does not claim to mitigate any attacks beyond the FIPS 140-2 Level 1 requirements for this validation.

3. Secure Operation

The sections below describe how to ensure that the module is operating in its validated configuration. **Operating the module without following the guidance herein (including the use of undocumented services) will result in non-compliant behavior and is outside the scope of this Security Policy**.

3.1 Module Setup

The following paragraphs describe the steps necessary to ensure that the Acronis SCS Cryptographic Module is running in its validated configuration.

3.1.1 Building and Linking

Please note that the Acronis SCS Cryptographic Module is not delivered to end-users as a standalone offering. Rather, it is an integrated component of the Acronis Cyber Backup SCS software. Acronis SCS does not provide end-users with source code to build the module.

As it is based on the OpenSSL FIPS Object Module 2.0.16, Acronis SCS developers build the cryptographic module as described in the *OpenSSL FIPS Object Module User Guide v2.0*. Note that the cryptographic module is not intended to be linked directly to calling applications. Rather, the cryptographic module shall first be linked to a "FIPS compatible" version of the OpenSSL library (1.0.1 and 1.0.2 baselines) to create a "FIPS capable" OpenSSL library which can then be referenced in calling applications.

- Environment variables – The build process for the FIPS module references three environment variables:

  FIPS_AUTH_CRYPT_OFFICER
  FIPS_AUTH_ CRYPT_USER
  FIPS_AUTH_KEY

  The values for these environment variables are set by Acronis SCS developers and specify the CO password HMAC digest, the User password HMAC digest, and the associated HMAC key. These values are stored in the firmware image at build-time.

- Linux environments – When building the cryptographic module from source, the following command set is issued from the cryptographic module's root directory:

```
$./config
$ make
$ sudo make install
```

  The FIPS-capable OpenSSL library is then built and linked to the cryptographic module using the following command set from the OpenSSL library's root directory:

```
$./config fips shared
$ make depend
```

```
$ make
$ sudo make install_sw
```

- Windows environments – When building the cryptographic module from source, the following command set is issued from the cryptographic module's root directory:

  ```
  ms\do_fips.bat
  ```

  The FIPS-capable OpenSSL library is then built and linked to the cryptographic using the following command set from the OpenSSL library's root directory:

  ```
  $ perl Configure VC-WIN64A fips –-with-fipslibdir=C:\usr\local\ssl\fips-2.0
  $ ms\do_win64a.bat

  $ nmake -f ms\nt.mak clean
  $ nmake -f ms\nt.mak
  $ nmake -f ms\nt.mak install

  $ nmake -f ms\ntdll.mak clean
  $ nmake -f ms\ntdll.mak
  $ nmake -f ms\ntdll.mak install
  ```

These build procedures ensure that the cryptographic module performs all power-up self-tests automatically when the module is loaded into memory for execution.

3.1.2 Installation

As the module is an integrated component of the Acronis Cyber Backup SCS software, module operators have no ability to independently load the module onto the target platform. The module and its calling application are to be installed on a platform specified in section 2.6 or one where portability is maintained. Acronis SCS does not provide any mechanisms to directly access the module, its APIs, or any information sent to/from the Acronis Cyber Backup SCS software.

During installation, the Acronis Cyber Backup SCS software verifies that the target platform's operational environment supports entropy generation via the RDRAND instruction. All platforms that were tested and vendor-affirmed during this validation provide such support. When installed on a platform that does not provide such support, the module may use entropy sources that are insufficient for generating cryptographic keys in a FIPS-Approved mode of operation. **Operation of the module when installed on a platform that does not provide entropy via the RDRAND instruction will result in non-compliant behavior and is outside the scope of this Security Policy**.

3.1.3 Initialization

This module is designed to support Acronis SCS applications, and these applications are the sole consumers of the cryptographic services provided by the module. No end-user action is required to initialize the module for operation; the calling applications perform all initialization actions required to place the module into FIPS mode. The power-up integrity test and cryptographic algorithm self-tests are then performed automatically via a default entry point (DEP) when the module is loaded for execution by the calling application, without any specific action

from the calling application or the end-user. End-users have no means to short-circuit or bypass these actions. Failure of any of the initialization actions will result in a failure of the module to load for execution.

3.1.4 Configuration

No configuration steps are required to be performed by end-users.

3.2 Operator Guidance

The following sections provide guidance to module operators for the correct and secure operation of the module.

3.2.1 Crypto Officer Guidance

There are no specific management activities required of the CO role to ensure that the module runs securely. However, if any irregular activity is noticed or the module is consistently reporting errors, then Acronis SCS Customer Support should be contacted.

3.2.2 User Guidance

Although the User role provides no ability to modify the configuration of the module, they should notify the CO if any irregular activity is noticed.

3.2.3 General Operator Guidance

The following provide further guidance for the general operation of the module:

- With the exception of the Operator password, the module does not store any CSPs persistently. Further, only the DRBG state values used for the module's key generation service are stored beyond the lifetime of the associated API call. Zeroization of temporarily stored CSPs is performed automatically by API function calls.

  As the operator password is stored as part of the module's binary image, procedural methods of zeroization shall also be used. While in the module operator's control, the module image shall be uninstalled from the host server, and the media upon which the module was stored shall be overwritten (at least once) or reformatted.

- The module stores DRBG state values for the lifetime of the DRBG instance. The `FIPS_drbg_uninstantiate()`API can be used to explicitly destroy CSPs related to random number generation services.

- To determine the module's operational status, the `FIPS_mode()` API can be used. A non-zero return value indicates that the module is running in its FIPS-Approved mode.

- To execute the module's power-up self-tests on-demand, the module's host server can be rebooted/power-cycled. Additionally, the `fips_selftests()` API can be used to execute the tests on an on-demand basis.

3.3 Additional Guidance and Usage Policies

The notes below provide additional guidance and policies that must be followed by the calling application (acting as the module's sole authorized operator):

- When built with a software library, the cryptographic module's services are intended to be provided to a calling application. Excluding the use of the NIST-defined elliptic curves as trusted third-party domain parameters, all other assurances from FIPS 186-4 (including those required of the intended signatory and the signature verifier) are outside the scope of the module and are the responsibility of the calling application.

- The seeds and entropy input are provided to the module by the calling application. The minimum number of bits of entropy loaded by the calling application is 112 bits.

  The calling application shall use entropy sources that meet the security strength required for the random number generation mechanism as shown in *NIST SP 800-90Arev1*, Table 2 (Hash DRBG and HMAC DRBG) and Table 3 (CTR DRBG). The entropy is supplied by means of callback functions. Those functions shall return an error if the collected entropy cannot meet the minimum security strength requirement.

- The calling application is responsible for ensuring that the module performs no more than 2^{16} 64-bit data block encryptions under the same three-key Triple-DES key.

- The calling application is responsible for the storage and zeroization of keys and CSPs passed into and out of the module.

- In the event that power to the module is lost and subsequently restored, the calling application must ensure that any AES-GCM keys used for encryption or decryption are re-distributed.

- In the event that the module encounters a DRBG self-test failure, the calling application must uninstantiate and reinstantiate the DRBG per the requirements found in *NIST SP 800-90Arev1*.

4. Acronyms

Table 10 provides definitions for the acronyms used in this document.

Table 10 – Acronyms

Acronym	Definition
AC	Alternating Current
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
BIOS	Basic Input/Output System
CBC	Cipher Block Chaining
CCM	Counter with CBC-MAC
CKG	Cryptographic Key Generation
CMVP	Cryptographic Module Validation Program
CO	Cryptographic Officer
CPU	Central Processing Unit
CCCS	Canadian Centre for Cyber Security
CSP	Critical Security Parameter
CTR	Counter
CVL	Component Validation List
DEP	Default Entry Point
DES	Data Encryption Standard
DRBG	Deterministic Random Bit Generator
DVD	Digital Video Disc
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECC CDH	Elliptic Curve Cryptography Cofactor Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
EMI/EMC	Electromagnetic Interference /Electromagnetic Compatibility
FFC DH	Finite Field Cryptography Diffie-Hellman
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
HDD	Hard Disk Drive
HMAC	(keyed-) Hash Message Authentication Code
I/O	Input/Output

Acronym	Definition
IT	Information Technology
KAS	Key Agreement Scheme
KAT	Known Answer Test
LCD	Liquid Crystal Display
LED	Light Emitting Diode
NDRNG	Non-Deterministic Random Number Generator
NIST	National Institute of Standards and Technology
OS	Operating System
PAA	Processor Algorithm Accelerators
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PKCS	Public Key Cryptography Standard
PKG	Key Pair Generation
PKV	Key Pair Verification
PSS	Probabilistic Signature Scheme
RAM	Random Access Memory
RHEL	Red Hat Enterprise Linux
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SP	Special Publication
SSH	Secure Shell
TDES	Triple Data Encryption Standard
USB	Universal Serial Bus
XEX	XOR Encrypt XOR
XOR	Exclusive OR
XTS	XEX Tweakable Block Cipher with Ciphertext Stealing

Prepared by:

Corsec Security, Inc.

13921 Park Center Road, Suite 460
Herndon, VA 20171
United States of America

Phone: +1 703 267 6050
Email: info@corsec.com
http://www.corsec.com