F5, Inc.

Cryptographic Module for BIG-IP

FIPS 140-3 Non-Proprietary Security Policy

Prepared by:
atsec information security corporation
4516 Seton Center Parkway, Suite 250
Austin, TX 78759
www.atsec.com

Table of Contents

1 General 7
1.1 Overview 7
1.2 Security Levels 7
1.3 Additional Information 7
2 Cryptographic Module Specification 8
2.1 Description 8
2.2 Tested and Vendor Affirmed Module Version and Identification 9
2.3 Excluded Components 10
2.4 Modes of Operation 10
2.5 Algorithms 10
2.6 Security Function Implementations 15
2.7 Algorithm Specific Information 17
2.8 RBG and Entropy 18
2.9 Key Generation 19
2.10 Key Establishment 19
2.11 Industry Protocols 19
3 Cryptographic Module Interfaces 20
3.1 Ports and Interfaces 20
4 Roles, Services, and Authentication 21
4.1 Authentication Methods 21
4.2 Roles 21
4.3 Approved Services 21
4.4 Non-Approved Services 28
4.5 External Software/Firmware Loaded 29
5 Software/Firmware Security 30
5.1 Integrity Techniques 30
5.2 Initiate on Demand 30
6 Operational Environment 31
6.1 Operational Environment Type and Requirements 31
6.2 Configuration Settings and Restrictions 31
7 Physical Security 32
8 Non-Invasive Security 33
9 Sensitive Security Parameters Management 34

9.1 Storage Areas 34
9.2 SSP Input-Output Methods 34
9.3 SSP Zeroization Methods 34
9.4 SSPs 35
9.5 Transitions 41
10 Self-Tests 42
10.1 Pre-Operational Self-Tests 42
10.2 Conditional Self-Tests 42
10.3 Periodic Self-Test Information 44
10.4 Error States 46
10.5 Operator Initiation of Self-Tests 46
11 Life-Cycle Assurance 47
11.1 Installation, Initialization, and Startup Procedures 47
11.2 Administrator Guidance 47
11.3 Non-Administrator Guidance 47
11.4 Design and Rules 47
11.5 End of Life 47
12 Mitigation of Other Attacks 48
Appendix A. Glossary and Abbreviations 49
Appendix B. References 50

List of Tables

Table 1: Security Levels 7
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 9
Table 3: Tested Operational Environments - Software, Firmware, Hybrid 9
Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid 10
Table 5: Modes List and Description 10
Table 6: Approved Algorithms 13
Table 7: Vendor-Affirmed Algorithms 13
Table 8: Non-Approved, Allowed Algorithms with No Security Claimed 14
Table 9: Non-Approved, Not Allowed Algorithms 15
Table 10: Security Function Implementations 17
Table 11: Entropy Certificates 18
Table 12: Entropy Sources 19
Table 13: Ports and Interfaces 20
Table 14: Roles 21
Table 15: Approved Services 27
Table 16: Non-Approved Services 29
Table 17: Storage Areas 34
Table 18: SSP Input-Output Methods 34
Table 19: SSP Zeroization Methods 34
Table 20: SSP Table 1 38
Table 21: SSP Table 2 41
Table 22: Pre-Operational Self-Tests 42
Table 23: Conditional Self-Tests 44
Table 24: Pre-Operational Periodic Information 44
Table 25: Conditional Periodic Information 45
Table 26: Error States 46

List of Figures

Figure 1: Block Diagram 8

Copyrights and Trademarks

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy that contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for an Overall Security Level 1 module.

1.2 Security Levels

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

Table 1: Security Levels

1.3 Additional Information

In preparing the Security Policy document, the laboratory formatted the vendor-supplied documentation for consolidation without altering the technical statements therein contained. The further refining of the Security Policy document was conducted iteratively throughout the conformance testing. The vendor reviewed the intermediate and final Security Policy and approved all of its content.

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The Cryptographic Module for BIG-IP (hereafter referred to as "the module") is a cryptographic library offering various cryptographic mechanisms to be used by OpenSSL application running on BIG-IP Virtual Edition.

The module provides cryptographic services to applications through an Application Program Interface (API). The module also interacts with the underlying operating system via system calls. The processors on which the module run include PAA functions.

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary: The software block diagram Figure 1 shows the module, its interfaces with the operational environment and the delimitation of its cryptographic boundary with bold black perimeter. The software components of the cryptographic module are listed in Table - Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets).

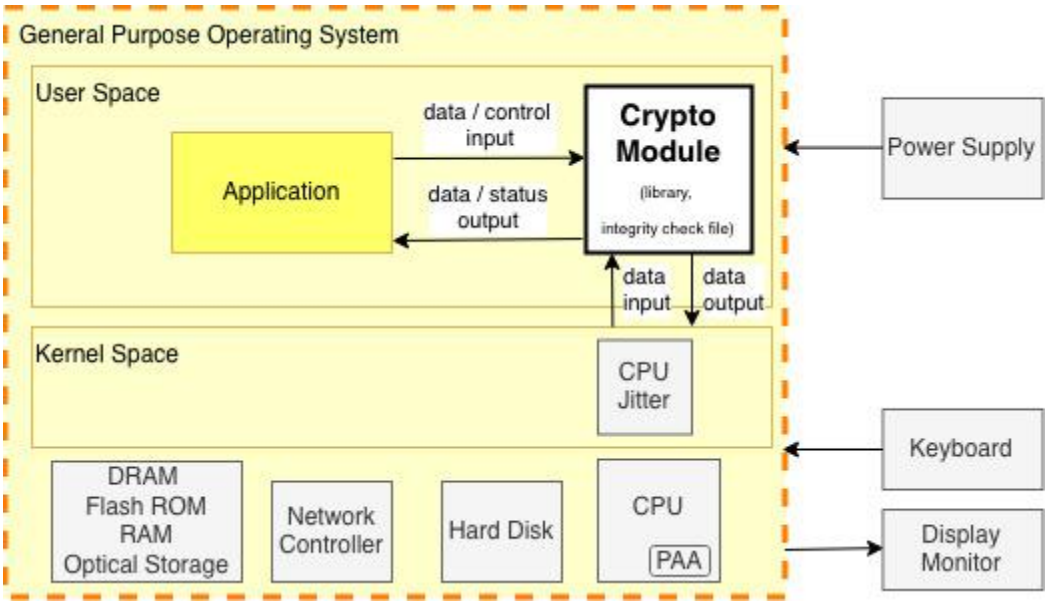

Figure 1: Block Diagram

Tested Operational Environment's Physical Perimeter (TOEPP): The module is aimed to run on a general-purpose computer; the physical perimeter is the surface of the case of the target platform, as shown with orange

dotted lines in the diagram Figure 1. The components of the TOEPP are listed in Table - Tested Operational Environments - Software, Firmware, Hybrid.

The entropy source located within the module's physical perimeter is outside of the module's cryptographic boundary (see Figure 1).

2.2 Tested and Vendor Affirmed Module Version and Identification

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
libcrypto.so.1.0.2za, .libcrypto.so.1.0.2za.hmac	1.0.2za-fips 0.66.4	N/A	HMAC-SHA-256

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

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
BIG-IP 17.1.0.1 EHF on VMWare	Dell PowerEdge R650	Intel® Xeon® Gold Ice Lake 6330N	Yes	VMware ESXi(TM) 8.0.0.10100 (Build: 20920323)	1.0.2za-fips 0.66.4
BIG-IP 17.1.0.1 EHF on Hyper-V	Dell PowerEdge R450	Intel® Xeon® Silver Ice Lake 4309Y	Yes	Hyper-V 10.0.20348.1 on Windows Server 2022	1.0.2za-fips 0.66.4
BIG-IP 17.1.0.1 EHF on KVM	Dell PowerEdge R450	Intel® Xeon® Silver Ice Lake 4309Y	Yes	KVM on Ubuntu 22.04.1 LTS	1.0.2za-fips 0.66.4
BIG-IP 17.1.0.1 EHF on VMWare	Dell PowerEdge R650	Intel® Xeon® Gold Ice Lake 6330N	No	VMware ESXi(TM) 8.0.0.10100 (Build: 20920323)	1.0.2za-fips 0.66.4
BIG-IP 17.1.0.1 EHF on Hyper-V	Dell PowerEdge R450	Intel® Xeon® Silver Ice Lake 4309Y	No	Hyper-V 10.0.20348.1 on Windows Server 2022	1.0.2za-fips 0.66.4
BIG-IP 17.1.0.1 EHF on KVM	Dell PowerEdge R450	Intel® Xeon® Silver Ice Lake 4309Y	No	KVM on Ubuntu 22.04.1 LTS	1.0.2za-fips 0.66.4

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

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

Operating System	Hardware Platform
BIG-IP 17.1.0.1 EHF	Azure CLI 2.48.1 with Intel Xeon Platinum 8272CL
BIG-IP 17.1.0.1 EHF	AWS CLI 2.11.19 with Intel Xeon Cascade Lake 8259CL

Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid

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.

2.3 Excluded Components

None

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved mode	Automatically entered whenever an approved service is requested	Approved	Output status corresponding to the calling service ("AES-ECB" is returned when AES-ECB encryption/decryption is called). See Section 4.3 "Indicator" column for each approved service.
Non-approved mode	Automatically entered whenever a non-approved service is requested	Non-Approved	No service indicator required for non-approved services per IG 2.4.C

Table 5: Modes List and Description

Mode Change Instructions and Status:

The module enters the approved mode after pre-operational self-tests succeed. The module automatically switches between the approved and non-approved modes depending on the services requested by the operator. The status indicator of the mode of operation is equivalent to the indicator of the service that was requested.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A3947, A3948	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CTR	A3947	Direction - Decrypt, Encrypt Key Length - 128, 192, 256 Payload Length - Payload Length: 8-128 Increment 8 Supports Counter larger than maximum value - No Incremental Counter - Yes Counter Tests Performed - Yes	SP 800-38A
AES-ECB	A3947, A3948	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A3947, A3948	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96 Payload Length - Payload Length: 128, 256, 104, 408 AAD Length - AAD Length: 128, 384, 160, 720, 0	SP 800-38D
AES-GMAC	A3947, A3948	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256 Tag Length - 104, 112, 120, 128, 32, 64, 96 IV Length - IV Length: 96 AAD Length - AAD Length: 0, 128, 160, 384, 720	SP 800-38D
Counter DRBG	A3947	Prediction Resistance - No, Yes Supports Reseed - Yes Mode - AES-256 Derivation Function Enabled - No, Yes Additional Input - Additional Input: 0 Entropy Input - Entropy Input: 256, Entropy Input: 384 Nonce - Nonce: 128 Personalization String Length - Personalization String Length: 0-256 Increment 8, Personalization String Length: 0-384 Increment 8 Returned Bits - 512	SP 800-90A Rev. 1
Counter DRBG	A3948	Prediction Resistance - Yes Supports Reseed - Yes Mode - AES-256 Derivation Function Enabled - Yes	SP 800-90A Rev. 1

Algorithm	CAVP Cert	Properties	Reference
		Additional Input - Additional Input: 0 Entropy Input - Entropy Input: 256 Nonce - Nonce: 128 Personalization String Length - Personalization String Length: 0-256 Increment 8 Returned Bits - 512	
ECDSA KeyGen (FIPS186-4)	A3947	Curve - P-256, P-384 Secret Generation Mode - Testing Candidates	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3947	Curve - P-256, P-384	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A3947	Component - No Curve - P-256, P-384 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A3947	Component - No Curve - P-256, P-384 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
HMAC-SHA-1	A3947, A3948	MAC - MAC: 160 Key Length - Key Length: 8, 16, 64, 128, 1024	FIPS 198-1
HMAC-SHA2-256	A3947	MAC - MAC: 256 Key Length - Key Length: 8, 16, 64, 128, 1024	FIPS 198-1
HMAC-SHA2-384	A3947	MAC - MAC: 384 Key Length - Key Length: 8, 16, 64, 128, 1024	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A3947	Domain Parameter Generation Methods - P-256, P-384 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A3947	Domain Parameter Generation Methods - ffdhe2048, ffdhe3072, ffdhe4096 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KDF SSH (CVL)	A3947	Cipher - AES-128, AES-256 Hash Algorithm - SHA2-256, SHA2-384	SP 800-135 Rev. 1
KDF TLS (CVL)	A3947	TLS Version - v1.0/1.1	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
RSA KeyGen (FIPS186-4)	A3947	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.2, Table C.3 Info Generated By Server - No Public Exponent Mode - Fixed Fixed Public Exponent - 010001 Private Key Format - Standard	FIPS 186-4
RSA SigGen (FIPS186-4)	A3947	Signature Type - PKCS 1.5 Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA2-256	FIPS 186-4
RSA SigVer (FIPS186-4)	A3947	Signature Type - PKCS 1.5 Modulo - 2048, 3072, 4096 Hash Pair - Hash Algorithm - SHA-1 Public Exponent Mode - Fixed Fixed Public Exponent - 010001	FIPS 186-4
Safe Primes Key Generation	A3947	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096	SP 800-56A Rev. 3
Safe Primes Key Verification	A3947	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096	SP 800-56A Rev. 3
SHA-1	A3947, A3948	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A3947	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A3947	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
TLS v1.2 KDF RFC7627 (CVL)	A3947	Hash Algorithm - SHA2-256, SHA2-384 Key Block Length - Key Block Length: 1024	SP 800-135 Rev. 1

Table 6: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
Cryptographic Key Generation (CKG)	Key Type:asymmetric	N/A	Random bit strings required for generating the cryptographic keys is compliant with section 4 example 1 of SP800-133r2

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms with No Security Claimed:

Name	Caveat	Use and Function
MD5	Allowed per IG 2.4.A	Message digest used in TLS 1.0 / 1.1 KDF only

Table 8: Non-Approved, Allowed Algorithms with No Security Claimed

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
AES with OFB, CCM, CFB, XTS, KW modes and Blowfish, Camellia, CAST5, DES, IDEA, RC2, RC4, SEED, SM2, SM4, Triple-DES	Symmetric encryption / decryption (not CAVP tested)
HMAC-SHA2-224, HMAC-SHA2-512, AES CMAC, Triple-DES CMAC	Message authentication (not CAVP tested)
ECDSA using P-224, P-521 curves, DSA with all key sizes, RSA with 1024 and greater than 4096 up to 16384 modulus	Key generation (not CAVP tested)
ECDSA using P-224, P-521 curves	Key verification (not CAVP tested)
ECDSA using SM2 algorithm, Probabilistic Signature Scheme (PSS), ANSI X9.31 schemes, PKCS #1 v1.5 scheme with 1024 and greater than 4096 up to 16384 modulus for all SHA sizes, DSA with all key and SHA sizes, ECDSA using curves P-256, P-384 with SHA-1, SHA2-224, SHA2-512, RSA PKCS #1 v1.5 scheme with modulus size 2048, 3072, 4096 bits with SHA-1, SHA2-224, SHA2-512	Digital Signature Generation (not FIPS186-4 compliant/ not CAVP tested)
ECDSA using SM2 algorithm, Probabilistic Signature Scheme (PSS), ANSI X9.31 schemes, PKCS #1 v1.5 scheme with 1024 and greater than 4096 up to 16384 modulus for all SHA sizes, DSA with all key and SHA sizes, ECDSA using curves P-256, P-384 with SHA-1, SHA2-224, SHA2-512, RSA PKCS #1 v1.5 scheme with modulus size 2048, 3072, 4096 bits with SHA2-224, SHA2-512	Digital Signature Verification (not FIPS186-4 compliant/ not CAVP tested)
RSA with modulus sizes up to 16384 bits	Asymmetric encryption / decryption
DSA with all key sizes	Domain parameter generation, domain parameter verification
Diffie-Hellman with groups other than ffdhe2048, ffdhe3072, ffdhe4096, EC Diffie-Hellman Ephemeral Unified without KDF with curves other than P-256, P-384.; onePassDh and StaticUnified without KDF	Shared secret computation (not CAVP tested)

Name	Use and Function
HMAC_DRBG and Hash_DRBG using all SHA sizes, CTR_DRBG with AES-128, AES-192, ANSI X9.31 RNG	Random number generation (not CAVP tested)
SHA2-224, SHA2-512, SM3, MD4, MD5 (outside of TLS), MDC2, RIPEMD, Whirlpool	Message digest (not CAVP tested)
TLS using SHA2-224 / SHA2-512	TLS KDF (not CAVP tested)
SSH using SHA-1 / SHA2-224 / SHA2-512	SSH KDF (not CAVP tested)

Table 9: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Symmetric Encryption and Decryption	BC-UnAuth BC-Auth	Symmetric Encryption and Decryption		AES-CBC: (A3947, A3948) AES-CTR: (A3947) AES-ECB: (A3947, A3948) AES-GCM: (A3947, A3948) IV: internal
MAC Generation/ Verification	MAC	MAC Generation/ Verification		AES-GMAC: (A3947, A3948)
Random Number Generation	DRBG	Random Number Generation		Counter DRBG: (A3947) Derivation Function Enabled: yes / no Prediction Resistance: yes / no Counter DRBG: (A3948) Derivation Function Enabled: Yes Prediction Resistance: Yes

Name	Type	Description	Properties	Algorithms
ECDSA Key Pair Generation	AsymKeyPair-KeyGen CKG	ECDSA Key Pair Generation		ECDSA KeyGen (FIPS186-4): (A3947) Cryptographic Key Generation (CKG): ()
ECDSA public key Verification	AsymKeyPair-KeyVer	ECDSA public key Verification		ECDSA KeyVer (FIPS186-4): (A3947)
ECDSA Signature Generation	DigSig-SigGen	ECDSA Signature Generation		ECDSA SigGen (FIPS186-4): (A3947)
ECDSA Signature Verification	DigSig-SigVer	ECDSA Signature Verification		ECDSA SigVer (FIPS186-4): (A3947)
Message Authentication	MAC	Message Authentication		HMAC-SHA-1: (A3947, A3948) HMAC-SHA2-256: (A3947) HMAC-SHA2-384: (A3947)
KAS-ECC-SSC (EC Diffie-Hellman shared secret computation)	KAS-SSC	Shared Secret Computation	IG:D.F Scenario 2 (path 1)	KAS-ECC-SSC Sp800-56Ar3: (A3947)
Key derivation	KAS-135KDF	Key derivation		KDF SSH: (A3947) KDF TLS: (A3947) TLS v1.2 KDF RFC7627: (A3947)
KAS-FFC-SSC (Diffie-Hellman shared secret computation)	KAS-SSC	Shared Secret Computation	IG:D.F Scenario 2 (path 1)	KAS-FFC-SSC Sp800-56Ar3: (A3947)
RSA Key Generation	AsymKeyPair-KeyGen CKG	RSA Key Generation		RSA KeyGen (FIPS186-4): (A3947) Cryptographic Key Generation (CKG): ()

Name	Type	Description	Properties	Algorithms
RSA Signature Generation	DigSig-SigGen	RSA Signature Generation		RSA SigGen (FIPS186-4): (A3947)
RSA Signature Verification (legacy)	DigSig-SigVer	RSA Signature Verification using SHA-1		RSA SigVer (FIPS186-4): (A3947)
RSA Signature Verification	DigSig-SigVer	RSA Signature Verification using SHA-256, SHA-384		RSA SigVer (FIPS186-4): (A3947)
Safe Primes Key Generation	AsymKeyPair-KeyGen CKG	Key Generation		Safe Primes Key Generation: (A3947) Cryptographic Key Generation (CKG): ()
Public Key Verification	AsymKeyPair-KeyVer	Public Key Verification		Safe Primes Key Verification: (A3947)
Message Digest	SHA	Message Digest		SHA-1: (A3947, A3948) SHA2-256: (A3947) SHA2-384: (A3947)

Table 10: Security Function Implementations

2.7 Algorithm Specific Information

KAS-SSC: The module does not establish SSPs using an approved key agreement scheme (KAS).

However, it does offer some or all of the underlying KAS cryptographic functionality to be used by an external operator/application as part of an approved KAS.

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

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

AES-GCM IV: The IV for AES-GCM is constructed in compliance with IG C.H scenario 1a (TLS 1.2) and scenario 1d (SSHv2).

- For TLS 1.2, the module offers the AES-GCM implementation and uses the context of Scenario 1 of IG C.H. The module is compliant with SP800-52Rev2 section 3.3.1 and the mechanism for IV generation is compliant with RFC5288.
  The module does not implement the TLS protocol. The module's implementation of AES-GCM is used together with an application that runs outside the module's cryptographic boundary. The design of the TLS

protocol implicitly ensures that the counter (the nonce_explicit part of the IV) does not exhaust the maximum number of possible values for a given session key.

- For SSHv2, the IV for the module AES-GCM implementation is only used in the context of the AES-GCM mode encryptions. The module is compliant with RFCs 4252, 4253 and 5647.
  The module does not implement SSH protocol and the module's implementation of AES-GCM is used together with an application that runs outside the module's cryptographic boundary. The design of the SSH protocol implicitly ensures that the counter does not exhaust the maximum number of possible values for a given session key and that the no more than 2^{64} -1 AES-GCM encryptions are performed.
  When a session is terminated, as new key and a new initial IV shall be derived.
- For both TLSv1.2 and SSHv2 protocols, in the event the module's power is lost and restored, the consuming application must ensure that a new key for use with the AES-GCM key encryption or decryption under this scenario shall be established.

RSA modulus size (IG C.F): The module implements FIPS 186-4 RSA KeyGen, SigVer and RSA SigGen with modulus lengths of 2048, 3072, 4096 bits. All these modulus lengths have been CAVP tested.

SP800-56Ar3 assurances (IG D.F): To comply with the assurances found in Section 5.6.2 of SP800-56Ar3, the keys for KAS-FFC-SSC and KAS-ECC-SSC must be generated using the approved key generation services specified in section 9.2. For KAS-FFC-SSC the module generates keys using Safe Primes Key Generation with Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096. For KAS-ECC-SSC, the module generates keys using ECDSA KeyGen, with curves P-384 and P-256. The module performs full public key validation on the generated public keys. Additionally, the module performs full public key validation on the received public keys.

SHA-1 (Table 8 of SP 800-131A rev2): The SHA-1 algorithm is offered "as is" in approved mode: Usage of SHA-1 is only approved when used for Message Authentication HMAC, Message Digest, TLS Key Derivation, and RSA Signature Verification (Legacy). Algorithms designated as "Legacy" can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M.
The module doesn't offer signature generation with SHA-1 in approved mode."

HMAC (IG C.L): In order to comply with IG C.L, the user shall ensure that HMAC keys are generated as specified in SP800-133r2.

2.8 RBG and Entropy

Cert Number	Vendor Name
E74	F5, Inc

Table 11: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
CPU Jitter 3.4.0	Non-Physical	BIG-IP 17.1.0.1 EHF on Hyper-V version 10.0.20348.1 on Windows Server 2022 Standard on Intel Ice Lake 4309Y; BIG-IP 17.1.0.1 EHF on KVM on Ubuntu 22.04.1 LTS on Intel Ice Lake 4309Y; BIG-IP 17.1.0.1 EHF on VMware ESXi	256	full entropy	SHA-3 #A2621

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
		8.0.0.10100 (Build 20920323) on Intel Ice Lake 6330N			

Table 12: Entropy Sources

The entropy source used by the module is specified in Table Entropy Sources uses jitter variations caused by executing instructions and memory accessed. The operator does not have the ability to modify the F5 entropy source (ES) configuration settings (see details in Public Use Document referenced in section 11.2).

The module employs a Deterministic Random Bit Generator (DRBG) based on [SP800-90ARev1] for the generation of random value used in asymmetric keys, and for providing a RNG service to calling applications. The approved DRBG provided by the module is the CTR_DRBG with AES-256.

The output of entropy sources provides 256-bits of entropy to seed and reseed SP800-90ARev1 DRBG during initialization (seed) and reseeding (reseed).

2.9 Key Generation

The module implements asymmetric key generation methods according to SP 800-133r2 section 5. The key generation methods are specified in the *Security Function Implementations* table.

The module does not implement symmetric key generation.

2.10 Key Establishment

The module implements SSP agreement, compliant with IG D.F scenario 2 (path 1). The Key Establishment methods are specified in the *Security Function Implementations* table.

2.11 Industry Protocols

GCM with internal IV generation in the approved mode is compliant with version 1.2 of the TLS protocol (RFC 5288) and shall only be used in conjunction with the TLS protocol.

Additionally, the module implements the TLS 1.2 and SSH key derivation functions for use in the TLS protocol and SSH protocol (RFC 4253 and RFC 6668).

No parts of the TLS 1.2 and SSHv2 protocols, other than the KDF, have been tested by the CAVP or CMVP.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	Data inputs are provided in the variables passed in the API and callable service invocations, generally through caller-supplied buffers
N/A	Data Output	Data outputs are provided in the variables passed in the API and callable service invocations, generally through caller-supplied buffers
N/A	Control Input	Control inputs which control the mode of the module are provided through dedicated parameters.
N/A	Status Output	Status output is provided in return codes and through messages. Documentation for each API lists possible return codes. A complete list of all return codes returned by the C language APIs within the module is provided in the header files and the API documentation. Messages are also documented in the API documentation.

Table 13: Ports and Interfaces

The logical interfaces are the API through which the applications request services.

The module does not implement Control Output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

FIPS 140-3 does not require an authentication mechanism for level 1 modules. Therefore, the module does not implement an authentication mechanism for Crypto Officer. The Crypto Officer role is implicitly assumed when accessing all services provided by the module (see Table - Approved Services and Table - Non-Approved Services below).

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	Crypto Officer	None

Table 14: Roles

No support is provided for multiple concurrent operators.

4.3 Approved Services

The status output from the FIPS_set_indicator_status function indicator's call is provided in Indicator column in Table – Approved Services. To read this indicator, the calling application must register a callback function using 'FIPS_register_indicator_callback'. The callback function shall take the input of the form "char *" which is the form of the indicator being output by the module.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Encryption / Decryption	Executes AES- mode encrypt or decrypt operation	AES-ECB, AES-CBC, AES-CTR, AES-GCM	Plaintext and key / ciphertext and key	Ciphertext / plaintext	Symmetric Encryption and Decryption	Crypto Officer - AES key: W,E
Random number generation	Generate Random number	CTR-DRBG-AES-256	Number of bits	Random numbers	Random Number Generation	Crypto Officer - Entropy input string: G,E,Z - DRBG seed : G,E - DRBG internal state (V and Key values): G,W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
RSA key pair generation	Generate RSA Key Pair	RSA-KEY-GEN-2048, RSA-KEY-GEN-3072, RSA-KEY-GEN- 4096	Key size	Key pair	RSA Key Generation	Crypto Officer - RSA public key: G,R - RSA private key: G,R
RSA signature generation	Sign a message with a specified RSA private key.	RSA-SIG	Private key, message, hashing algorithm	computed signature	RSA Signature Generation	Crypto Officer - RSA private key: W,E
RSA signature verification	Verify the signature of a message with a specified RSA public key.	RSA-VER	Public key, digital signature, message, hashing algorithm	Pass / fail result of digital signature verification	RSA Signature Verification RSA Signature Verification (legacy)	Crypto Officer - RSA public key: W,E
ECDSA / ECDH key pair generation	Generate a key pair for a requested elliptic curve	EC-KEYGEN-P-256, EC-KEYGEN-P-384	Key size	Key pair	ECDSA Key Pair Generation	Crypto Officer - ECDSA public key: G,R - ECDSA private key: G,R - EC Diffie-Hellman private key: G,R - EC Diffie-Hellman public key : G,R

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
ECDSA / ECDH public key verification	Test that an ECC public key is a point on the specified elliptic curve	EC-KEY-VERIFY-P-256, EC-KEY-VERIFY-P-384	Key pair	Pass / fail result of public key verification	ECDSA public key Verification	Crypto Officer - ECDSA public key: W,E - EC Diffie-Hellman public key : W,E
ECDSA signature generation	Sign a message with a specified ECDSA private key.	ECDSA-SIGN-P-256, ECDSA-SIGN-P-384	Private key, message, hashing algorithm	Computed signature	ECDSA Signature Generation	Crypto Officer - ECDSA private key: W,E
ECDSA signature verification	Verify the signature of a message with a specified ECDSA public key	ECDSA-VERIFY-P-256, ECDSA-VERIFY-P-384	Public key, digital signature, message, hashing algorithm	Pass / fail result of digital signature verification	ECDSA Signature Verification	Crypto Officer - ECDSA private key: W,E
EC Diffie-Hellman shared secret computation IG D.F scenario 2, path 1	Calculate a shared secret via the ECDH algorithm.	ECDH-COMPUTE-KEY-P-256, ECDH-COMPUTE-KEY-P-384	Received public key and possessed private key	Shared secret	KAS-ECC-SSC (EC Diffie-Hellman shared secret computation)	Crypto Officer - EC Diffie-Hellman public key : W,E - EC Diffie-Hellman private key: W,E - EC Diffie-Hellman Shared Secret: G,R

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- ECC Domain Parameters (SP800-56Ar3) : R,E
Safe primes key generation	Generate a key pair	FFDHE2048-KEYGEN, FFDHE3072-KEYGEN, FFDHE4096-KEYGEN	Key size	Generated Diffie-Hellman key pair	Safe Primes Key Generation	Crypto Officer - Diffie-Hellman private key: G,R - Diffie-Hellman public key : G,R
Safe primes key verification	Verify public key	FFDHE2048-COMPUTE, FFDHE3072-COMPUTE, FFDHE4096-COMPUTE	Key pair	Pass / fail result of public key verification	Public Key Verification	Crypto Officer - Diffie-Hellman private key: W,E - Diffie-Hellman public key : W,E
Diffie-Hellman Shared Secret Computation IG D.F scenario 2, path 1	Calculate a shared secret via the DH algorithm.	FFDHE2048-COMPUTE, FFDHE3072-COMPUTE, FFDHE4096-COMPUTE	Received public key and possessed private key	Shared secret	KAS-FFC-SSC (Diffie-Hellman shared secret computation)	Crypto Officer - Diffie-Hellman public key : R,E - Diffie-Hellman private key: E - Diffie-Hellman shared secret: G - FFC Domain Parameter

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						s (SP800-56Ar3): R,E
Message digest	Generate a digest for the requested algorithm	MESSAGE-DIGEST-SHA-1, MESSAGE-DIGEST-SHA-256, MESSAGE-DIGEST-SHA-384	Message, Hashing algorithm	Hashed message	Message Digest	Crypto Officer
MAC Generation / verification	Generate / verify an HMAC or GMAC digest using the requested SHA algorithm or AES algorithm as appropriate	MSG-AUTH-HMAC-SHA-1, MSG-AUTH-HMAC-SHA-256, MSG-AUTH-HMAC-SHA-384, AES-GMAC	MAC generation: Message, HMAC key or GMAC key, MAC algorithm, MAC length. MAC verification: Authenticated message, HMAC key or GMAC key, MAC algorithm	MAC generation: Authenticated message. MAC verification: Message	MAC Generation/ Verification Message Authentication	Crypto Officer - AES key: W,E - HMAC key: W,E
TLS key derivation	Deriving TLS keys	TLS-P-HASH-DERIVATION-SHA-1 TLS-P-HASH-DERIVATION-SHA-256 TLS-P-HASH-DERIVATION SHA-384	PRF algorithm, TLS pre-primary secret, TLS primary secret	Derived key	Key derivation	Crypto Officer - TLS pre-primary secret : W,E - TLS primary secret: G,E - TLS derived key (AES HMAC): G
SSH key derivation	Deriving SSH keys	SSH-KEY-HASH-	PRF algorithm,	Derived key	Key derivation	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		DERIVATION-SHA-256 SSH-KEY-HASH-DERIVATION SHA-384	SSH shared secret			- SSH shared secret: W,E - SSH derived key (AES, HMAC): G
Show version	Return the SW version and the module's name	None	None	Name and Version information	None	Crypto Officer
Show Status	Return the module status	None	None	Status output	None	Crypto Officer
Self-tests	Execute integrity test, Execute the CASTs	None	Power	Pass / fail results of self-tests	None	Crypto Officer
Zeroization	Zeroize all non-protected SSPs	None	Destruction functions provided in the module's API. Platform on which the module runs is powered down.	None	None	Crypto Officer - AES key: Z - HMAC key: Z - RSA public key: Z - ECDSA public key: Z - EC Diffie-Hellman public key : Z - EC Diffie-Hellman

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Shared Secret: Z - Diffie-Hellman public key : Z - Diffie-Hellman shared secret: Z - TLS pre-primary secret : Z - TLS primary secret: Z - TLS derived key (AES HMAC): Z - SSH shared secret: Z - SSH derived key (AES, HMAC): Z - EC Diffie-Hellman private key: Z - Diffie-Hellman private key: Z - DRBG seed : Z - DRBG internal state (V and Key values): Z

Table 15: Approved Services

For the above table, the convention below applies when specifying the access permissions (types) that the service has for each SSP.

G = **Generate**: The module generates or derives the SSP.

R = **Read**: The SSP is read from the module (e.g. the SSP is output).

W = **Write**: The SSP is updated, imported, or written to the module.

E = **Execute**: The module uses the SSP in performing a cryptographic operation.

Z = **Zeroise**: The module zeroises the SSP.

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Symmetric Encryption / Decryption (not CAVP tested)	Encrypt a plaintext / Decrypt a ciphertext	AES with OFB, CCM, CFB, XTS, KW modes and Blowfish, Camellia, CAST5, DES, IDEA, RC2, RC4, SEED, SM2, SM4, Triple-DES	CO
Message authentication (not CAVP tested)	Generate and verify a MAC tag	HMAC-SHA2-224, HMAC-SHA2-512, AES CMAC, Triple-DES CMAC	CO
Key generation (not CAVP tested)	Generate a key pair	ECDSA using P-224, P-521 curves, DSA with all key sizes, RSA with 1024 and greater than 4096 up to 16384 modulus	CO
Key verification (not CAVP tested)	Verify a key pair	ECDSA using P-224, P-521 curves	CO
Signature generation (not FIPS186-4 compliant/ not CAVP tested)	Generate a signature	ECDSA using SM2 algorithm, Probabilistic Signature Scheme (PSS), ANSI X9.31 schemes, PKCS #1 v1.5 scheme with 1024 and greater than 4096 up to 16384 modulus for all SHA sizes, DSA with all key and SHA sizes, ECDSA using curves P-256, P-384 with SHA-1, SHA2-224, SHA2-512, RSA PKCS #1 v1.5 scheme with modulus size 2048, 3072, 4096 bits with SHA-1, SHA2-224, SHA2-512	CO
Signature verification (not FIPS186-4 compliant/ not CAVP tested)	Verify a signature	ECDSA using SM2 algorithm, Probabilistic Signature Scheme (PSS), ANSI X9.31 schemes, PKCS #1 v1.5 scheme with 1024 and greater than 4096 up to 16384 modulus for all SHA sizes, DSA with all key and SHA sizes, ECDSA using curves P-256, P-384 with SHA-1, SHA2-224, SHA2-512, RSA PKCS #1 v1.5 scheme with modulus size 2048, 3072, 4096 bits with SHA2-224, SHA2-512	CO
Asymmetric encryption /	Encrypt a plaintext /	RSA with modulus sizes up to 16384 bits	CO

Name	Description	Algorithms	Role
decryption (not CAVP tested)	Decrypt a ciphertext		
Domain parameter generation and verification	Generate and verify a domain parameter	DSA with all key sizes	CO
Shared secret computation (not CAVP tested)	Compute a shared secret	Diffie-Hellman with groups other than ffdhe2048, ffdhe3072, ffdhe4096, EC Diffie-Hellman Ephemeral Unified without KDF with curves other than P-256, P-384.; onePassDh and StaticUnified without KDF	CO
Random number generation	Generate a random number	HMAC_DRBG and Hash_DRBG using all SHA sizes, CTR_DRBG with AES-128, AES-192, ANSI X9.31 RNG	CO
Message digest (not CAVP tested)	Compute a message digest	SHA2-224, SHA2-512, SM3, MD4, MD5 (outside of TLS), MDC2, RIPEMD, Whirlpool	CO
TLS Key Derivation (not CAVP tested)	Derive a key from a key-derivation key or a shared secret via TLS	TLS using SHA2-224 / SHA2-512	CO
SSH Key Derivation (not CAVP tested)	Derive a key from a key-derivation key or a shared secret via SSH	SSH using SHA-1 / SHA2-224 / SHA2-512	CO

Table 16: Non-Approved Services

4.5 External Software/Firmware Loaded

The section is not applicable for the module.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by comparing a HMAC value calculated at run time on the libcrypto.so.1.0.2za file, with the HMAC-SHA2-256 value stored in the module file .libcrypto.so.1.0.2za.hmac that was computed at build time. The HMAC key used for integrity verification is 256 bits in length and is stored as part of the module binary.

Integrity tests are performed as part of the Pre-Operational Self-Tests.

5.2 Initiate on Demand

The on-demand integrity test is performed by unloading and reloading the module.

6 Operational Environment

6.1 Operational Environment Type and Requirements

BIG-IP consists of a Linux based operating system customized for performance that runs directly on the hardware or in virtual environment.

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The module shall be installed as stated in Section 11.1. If properly installed, the operating system provides process isolation and memory protection mechanisms that ensure appropriate separation for memory access among the processes on the system. Each process has control over its own data, and uncontrolled access to the data of other processes is prevented.

6.2 Configuration Settings and Restrictions

The module runs on a BIG-IP 17.1.0.1 EHF operating system executing on the hardware and hypervisor specified in Table - Tested Operational Environments. The module should be installed as stated in section 11.1. The operator should confirm that the module is installed and configured correctly by section 11.2. The restrictions to the configuration are listed in sections 11.4.

7 Physical Security

The module is a software and therefore this section is Not Applicable (N/A).

8 Non-Invasive Security

Per IG 12.A: Until requirements of SP 800-140F are defined, non-invasive mechanisms fall under ISO / IEC 19790:2012 Section 7.12 Mitigation of other attacks.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Temporary storage for SSPs used by the module as part of service execution.	Dynamic

Table 17: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API input parameters	Operator calling application (TOEPP)	Cryptographic module	Plaintext	Manual	Electronic	
API output parameters	Cryptographic module	Operator calling application (TOEPP)	Plaintext	Manual	Electronic	

Table 18: SSP Input-Output Methods

The module only supports SSP entry and output to and from the calling application running on the same operational environment. This corresponds to manual distribution, electronic entry/output ("CM Software to/from App via TOEPP Path") per FIPS 140-3 IG 9.5.A Table 1.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Wipe and Free memory block allocated	Zeroizes the SSPs contained within the cipher handle.	Memory occupied by SSPs is overwritten with zeroes and then it is released, which renders the SSP values irretrievable. The completion of the zeroization routine indicates that the zeroization procedure succeeded.	By calling the cipher related zeroization API
Module Reset	De-allocates the volatile memory used to store SSPs	Volatile memory used by the module is overwritten within nanoseconds when power is removed.	By unloading and reloading the module

Table 19: SSP Zeroization Methods

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES key	AES key used for symmetric encryption / decryption and for MAC Generation and for Key Transport	Key length: 128 to 256 bits - 128 to 256 bits	Symmetric - CSP			
HMAC key	HMAC key used for Message authentication	Key length: 112 to 192-bits - 112 to 192-bits	Symmetric - CSP			
RSA public key	RSA public key used for digital signature verification	Modulus N: 2048, 3072, 4096 bits - 112 to 150 bits	Asymmetric - PSP	RSA Key Generation		
ECDSA public key	ECDSA public key used for Digital signature verification	Curve size: P-256 and P-384 bits - 128 and 192 bits	Asymmetric - PSP	ECDSA Key Pair Generation		
EC Diffie-Hellman public key	EC Diffie-Hellman public key used for ECDH SSC	Curve size: P-256, P-384-bits - 128 and 192-bits	Asymmetric - PSP	ECDSA Key Pair Generation		
EC Diffie-Hellman Shared Secret	EC Diffie-Hellman Shared Secret	Curve size: P-256, P-384 bits - 128 and 192 bits	Asymmetric - CSP	KAS-ECC-SSC (EC Diffie-Hellman shared secret computation)		
Diffie-Hellman public key	Diffie-Hellman public key used for KAS-FFC-SSC	Domain parameters: ffdhe2048, ffdhe3072, ffdhe4096 -	Asymmetric - PSP	Safe Primes Key Generation		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		112 to 150 bits				
Diffie-Hellman shared secret	Diffie-Hellman shared secret	Domain parameters: ffdhe2048, ffdhe3072, ffdhe4096 - 112 to 150 bits	Asymmetric - CSP	KAS-FFC-SSC (Diffie-Hellman shared secret computation)		
Entropy input string	Entropy obtained from the non-physical entropy source	1/3 bits - 256 bits	Entropy Input - CSP			Random Number Generation
DRBG seed	DRBG seed used for Random number generation	384 bits - 256 bits	Seed - CSP	Random Number Generation		Random Number Generation
DRBG internal state (V and Key values)	DRBG internal state (V and Key values) / CSP used for: Use: Random number generation	256 bits - 256 bits	Internal state - CSP	Random Number Generation		Random Number Generation
TLS pre-primary secret	TLS pre-primary secret used for TLS KDF	EC Diffie-Hellman Curves: P-224, P-256, P-384 - Diffie-Hellman: 112, 128, 150-bits EC Diffie-Hellman: 128-bits or 192-bits	Asymmetric - CSP			

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
TLS primary secret	TLS primary secret used for TLS KDF	Key length: 384-bits - 128 or 192 bits	Symmetric - CSP	Key derivation		
TLS derived key (AES HMAC)	TLS derived key (AES HMAC) used for TLS protocol	Key length: 128 and 256-bits (AES) 112 and 256-bits (HMAC) - 112 to 192-bits	Symmetric - CSP	Key derivation		
SSH shared secret	SSH shared secret used for: SSH protocol	Curve size: P-256 and P-384-bits - 128 and 192-bits	Symmetric - CSP			
SSH derived key (AES, HMAC)	SSH derived key (AES, HMAC) used in SSH protocol	Key length: 128 and 256-bits (AES), 112 and 256-bits (HMAC) - 112 to 192-bits	Symmetric - CSP	Key derivation		
RSA private key	RSA private key pair used for digital signature generation	Modulus N: 2048, 3072, 4096 bits - 112-150 bits	Asymmetric - CSP	RSA Key Generation		
ECDSA private key	ECDSA private key pair used for digital signature generation	Curve size: P-256, P-384 bits - 128 and 192-bits	Asymmetric - CSP	ECDSA Key Pair Generation		
EC Diffie-Hellman private key	ECDH private key or digital signature generation	Curve size: P-256, P-384 bits - 128 and 192-bits	Asymmetric - CSP	ECDSA Key Pair Generation		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Diffie-Hellman private key	DH private key used for KAS-FFC-SSC	Domain parameters: ffdhe2048, ffdhe3072, ffdhe4096 - 112 to 150 bits	Asymmetric - CSP	Safe Primes Key Generation		
ECC Domain Parameters (SP800-56Ar3)	SP800-56Ar3 ECC Domain Parameters	P-256, P-384 - 128-bits or 192-bits	Asymmetric - PSP			KAS-ECC-SSC (EC Diffie-Hellman shared secret computation)
FFC Domain Parameters (SP800-56Ar3)	SP800-56Ar3 FFC Domain Parameters	ffdhe2048, ffdhe3072, ffdhe4096 - 112 to 150 bits	Asymmetric - PSP			KAS-FFC-SSC (Diffie-Hellman shared secret computation)

Table 20: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES key	API input parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
HMAC key	API input parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
RSA public key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	RSA private key:Paired With
ECDSA public key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	ECDSA private key:Paired With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
EC Diffie-Hellman public key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC Diffie-Hellman private key:Paired With
EC Diffie-Hellman Shared Secret	API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC Diffie-Hellman public key :Derived From EC Diffie-Hellman private key:Derived From
Diffie-Hellman public key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	
Diffie-Hellman shared secret	API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Diffie-Hellman public key :Established Diffie-Hellman private key:Established
Entropy input string	API input parameters	RAM:Plaintext	From service invocation to service completion	N/A	DRBG seed / CSP:Used With
DRBG seed		RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Entropy input string:Derived From DRBG internal state (V and Key values) / CSP:Derives
DRBG internal state (V and Key values)		RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	DRBG seed :Derived From
TLS pre-primary secret	API input parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	TLS primary secret:Derives

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
TLS primary secret		RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	TLS pre-primary secret :Derived From
TLS derived key (AES HMAC)	API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	TLS primary secret:Derived From
SSH shared secret	API input parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	SSH derived key (AES, HMAC):Derives
SSH derived key (AES, HMAC)	API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	SSH shared secret:Derived From
RSA private key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	RSA public key:Paired With
ECDSA private key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	ECDSA public key:Paired With
EC Diffie-Hellman private key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC Diffie-Hellman public key :Paired With ECC Domain Parameters (SP800-56Ar3) :Used With EC Diffie-Hellman Shared Secret:Derives
Diffie-Hellman private key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Diffie-Hellman public key :Paired With FFC Domain Parameters (SP800-56Ar3):Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					Diffie-Hellman shared secret:Derives
ECC Domain Parameters (SP800-56Ar3)		RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	EC Diffie-Hellman public key :Used With EC Diffie-Hellman private key:Used With EC Diffie-Hellman Shared Secret:Derives
FFC Domain Parameters (SP800-56Ar3)		RAM:Plaintext	From service invocation to service completion	Wipe and Free memory block allocated Module Reset	Diffie-Hellman public key :Used With Diffie-Hellman private key:Used With Diffie-Hellman shared secret:Derives

Table 21: SSP Table 2

9.5 Transitions

The SHA-1 algorithm as implemented by the module will be non-approved for all purposes, starting January 1, 2031.

10 Self-Tests

Pre-operational self-tests, and conditional algorithm self-tests are performed automatically when the module is loaded into memory.

While the module is executing the pre-operational self-tests, and conditional algorithm self-tests, services are not available, and input and output are inhibited. The module does not return control to the calling application until the tests are completed. On successful completion of the pre-operational self-tests, and conditional algorithm self-tests the module enters operational mode and cryptographic services are available.

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A3947)	HMAC key: 256-bits	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity of the module is verified by comparing the HMAC-SHA2-256 value calculated at runtime with the HMAC-SHA2-256 value stored in the module that was computed at build time

Table 22: Pre-Operational Self-Tests

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM-encrypt	256-bit key, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM-decrypt	256-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB-encrypt	128-bit key encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB-decrypt	128-bit key decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
Counter DRBG	AES 256-bits	KAT	CAST	Module becomes operational	instantiate, generate and reseed health test per section 11.3 of the SP 800-90Ar1	Test runs at power-on before the integrity test
HMAC-SHA2-256	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA-1	SHA-1	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
HMAC-SHA2-384	SHA2-384	KAT	CAST	Module becomes operational	Message authentication	Test runs at power-on before the integrity test
RSA KeyGen (FIPS186-4)	SHA2-256 and respective keys	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA SigGen (FIPS186-4)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-4)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA KeyGen (FIPS186-4)	SHA2-256 and respective keys	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA SigGen (FIPS186-4)	P-256 with SHA-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-4)	P-256 with SHA-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KAS-ECC-SSC Sp800-56Ar3	P-256 curve	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-FFC-SSC Sp800-56Ar3	ffdhe2048	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
Safe Primes Key Generation	ffdhe2048	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
KDF SSH	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Test runs at power-on before the integrity test
KDF TLS	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Test runs at power-on before the integrity test

Table 23: Conditional Self-Tests

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A3947)	Message authentication	SW/FW Integrity	On Demand	Manually

Table 24: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-GCM-encrypt	KAT	CAST	On Demand	Manually
AES-GCM-decrypt	KAT	CAST	On Demand	Manually
AES-ECB-encrypt	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB-decrypt	KAT	CAST	On Demand	Manually
Counter DRBG	KAT	CAST	On Demand	Manually
HMAC-SHA2-256	KAT	CAST	On Demand	Manually
HMAC-SHA-1	KAT	CAST	On Demand	Manually
HMAC-SHA2-384	KAT	CAST	On Demand	Manually
RSA KeyGen (FIPS186-4)	PCT	PCT	On Demand	Manually
RSA SigGen (FIPS186-4)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-4)	KAT	CAST	On Demand	Manually
ECDSA KeyGen (FIPS186-4)	PCT	PCT	On Demand	Manually
ECDSA SigGen (FIPS186-4)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-4)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3	KAT	CAST	On Demand	Manually
KAS-FFC-SSC Sp800-56Ar3	KAT	CAST	On Demand	Manually
Safe Primes Key Generation	PCT	PCT	On Demand	Manually
KDF SSH	KAT	CAST	On Demand	Manually
KDF TLS	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627	KAT	CAST	On Demand	Manually

Table 25: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Halt Error	Module is no longer operational. The data output is inhibited.	HMAC-SHA2-256 - integrity technique- KAT failure, HMAC-SHA2-256 integrity test failure, Failure of any of the CASTs, Failure of any of the PCTs	The module must be re-loaded with a fresh image	Integrity test failure: the module will not load. KAT failure: flag fips_selftest_fail set to 1 and the module will not load, PCT failure: flag fips_selftest_fail set to 1 and module is aborted confirming it entered the error state

Table 26: Error States

The module must be re-loaded in order to clear the error condition.

10.5 Operator Initiation of Self-Tests

The on demand self-tests can be invoked by unloading and subsequently reloading the module. This service performs the same cryptographic algorithm tests executed during power-on and module loading. During the execution of the on demand self-tests, crypto services are not available, and no data output or input is possible.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

Startup Procedures: Before the Crypto Officer can configure and use the BIG-IP system, the Crypto Officer must activate a valid license on the system. The Crypto Officer chooses the license to buy by selecting the hypervisor from Table 'Tested Operational Environments - Software, Firmware, Hybrid". The following procedures described in "K7752 Licensing the BIG-IP system" on my.F5.com (https://my.f5.com/manage/s/article/K7752#reg) are performed:

- Obtaining a registration key
- Obtaining a dossier
- Activating the license

Installation Process: The Crypto Officer downloads the BIG-IP VE image (ie the module) and deploys it. After the FIPS validated module license is installed, the command prompt will change to 'REBOOT REQUIRED'. The Crypto Officer must reboot the BIG-IP for all FIPS-compliant changes to take effect.

11.2 Administrator Guidance

The Crypto Officer should verify the validity of the BIG-IP software license by running the command: *'tmsh show sys license*' which should output 'FIPS 140, BIG-IP VE-1G to 10G,' under the 'Active Modules' list.

On the BIG-IP product the Crypto Officer should call the dedicated Show version API, fips_get_f5fips_module_version, to ensure that the module identifier and version are shown as: Cryptographic Module for BIG-IP OpenSSL 1.0.2za-fips 0.66.4.

If the module has passed all self-tests then it is operating in the Approved mode. The Approved mode of operation can only transition into the non-Approved mode by calling one of the non-Approved services listed in Table - Non-Approved Services.

The ESV Public Use Document (PUD) reference for non-physical entropy source is as follows: https://csrc.nist.gov/projects/cryptographic-module-validation-program/entropy-validations/certificate/74

11.3 Non-Administrator Guidance

The section is not applicable for the module.

11.4 Design and Rules

The Crypto Officer shall consider the requirements and restrictions in above section 2.7 when using the module.

11.5 End of Life

As the module does not persistently store SSPs, secure sanitization of the module consists of unloading the module. This will zeroize all SSPs in volatile memory.

12 Mitigation of Other Attacks

The module does not implement security mechanisms to mitigate other attacks.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cipher Feedback
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CTR	Counter Mode
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ESV	Entropy Source Validation
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
KAS	Key Agreement Schema
KAT	Known Answer Test
KDF	Key Derivation Function
KW	AES Key Wrap
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
OFB	Output Feedback
PAA	Processor Algorithm Acceleration
PCT	Pairwise Consistency Test
PR	Prediction Resistance
PSS	Probabilistic Signature Scheme
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SSH	Secure Shell
XTS	XEX-based Tweaked-codebook mode with cipher text Stealing

Appendix B. References

FIPS140-3	**FIPS PUB 140-3 - Security Requirements for Cryptographic Modules** March 2019 https://doi.org/10.6028/NIST.FIPS.140-3
FIPS140-3_IG	**Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program** April 2025 https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements
FIPS180-4	**Secure Hash Standard (SHS)** March 2012 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
FIPS186-4	**Digital Signature Standard (DSS)** July 2013 https://doi.org/10.6028/NIST.FIPS.186-4
FIPS197	**Advanced Encryption Standard** November 2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
FIPS198-1	**The Keyed Hash Message Authentication Code (HMAC)** July 2008 http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf
FIPS202	**SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions** August 2015 https://dx.doi.org/10.6028/NIST.FIPS.202
PKCS#1	**Public Key Cryptography Standards (PKCS) #1: RSA Cryptography** Specifications Version 2.1 February 2003 http://www.ietf.org/rfc/rfc3447.txt
RFC3394	**Advanced Encryption Standard (AES) Key Wrap Algorithm** September 2002 http://www.ietf.org/rfc/rfc3394.txt
RFC5649	**Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm** September 2009 http://www.ietf.org/rfc/rfc5649.txt
SP800-38A	**NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques** December 2001 https://doi.org/10.6028/NIST.SP.800-38A

SP800-38B	**NIST Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication** May 2005 https://doi.org/10.6028/NIST.SP.800-38B
SP800-38C	**NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality** May 2004 http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf
SP800-38D	**NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC** November 2007 https://doi.org/10.6028/NIST.SP.800-38D
SP800-38E	**NIST Special Publication 800-38E - Recommendation for Block Cipher Modes of Operation: The XTS AES Mode for Confidentiality on Storage Devices** January 2010 https://doi.org/10.6028/NIST.SP.800-38E
SP800-38F	**NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping** December 2012 http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf
SP800-38G	**NIST Special Publication 800-38G - Recommendation for Block Cipher Modes of Operation: Methods for Format - Preserving Encryption** March 2016 https://dx.doi.org/10.6028/NIST.SP.800-38G
SP800-56ARev3	**NIST Special Publication 800-56A Revision 3 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography** April 2018 https://doi.org/10.6028/NIST.SP.800-56Ar3
SP800-56CRev2	**Recommendation for Key Derivation through Extraction-then-Expansion** August 2020 https://doi.org/10.6028/NIST.SP.800-56Cr2
SP800-57	**NIST Special Publication 800-57 Part 1 Revision 54 - Recommendation for Key Management Part 1: General** May 2020 https://doi.org/10.6028/NIST.SP.800-57pt1r5
SP800-90ARev1	**NIST Special Publication 800-90A - Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators** June 2015 http://dx.doi.org/10.6028/NIST.SP.800-90Ar1

SP800-90B	**NIST Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation** January 2018 https://doi.org/10.6028/NIST.SP.800-90B
SP800-131A	**NIST Special Publication 800-131A Revision 2- Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths** March 2019 https://doi.org/10.6028/NIST.SP.800-131Ar2
SP800-132	**NIST Special Publication 800-132 - Recommendation for Password-Based Key Derivation - Part 1: Storage Applications** December 2010 https://doi.org/10.6028/NIST.SP.800-132
SP800-133Rev2	**NIST Special Publication 800-133 - Recommendation for Cryptographic Key Generation** June 2020 https://doi.org/10.6028/NIST.SP.800-133r2
SP800-135Rev1	**NIST Special Publication 800-135 Revision 1 - Recommendation for Existing Application-Specific Key Derivation Functions** December 2011 http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf
SP800-140Br1	**NIST Special Publication 800-140Br1 - CMVP Security Policy Requirements** March 2020 https://doi.org/10.6028/NIST.SP.800-140Br1