ORACLE®
Linux

FIPS 140-2 Non-Proprietary Security Policy

Oracle Linux 7 GnuTLS Cryptographic Module

FIPS 140-2 Level 1 Validation

Software Version: R7-4.0.0

Date: September 27th, 2021

©Oracle Corporation

This document may be reproduced whole and intact including the Copyright notice.

Title: Oracle Linux 7 Gnu TLS Cryptographic Module Security Policy
Date: September 27th, 2021
Author: Oracle Security Evaluations – Global Product Security
Contributing Authors:
Oracle Linux Engineering

Oracle Corporation
World Headquarters
2300 Oracle Way
Austin, TX 78741
U.S.A.
Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com

Copyright © 2021, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. Oracle specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may reproduced or distributed whole and intact including this copyright notice.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Hardware and Software, Engineered to Work Together

ORACLE

TABLE OF CONTENTS

List of Tables

List of Figures

1. Introduction

1.1 Overview

This document is the Security Policy for the Oracle Linux 7 GnuTLS Cryptographic Module by Oracle Corporation. Oracle Linux 7 GnuTLS Cryptographic Module is also referred to as “the Module or Module”. This Security Policy specifies the security rules under which the module shall operate to meet the requirements of FIPS 140-2 Level 1. It also describes how the Oracle Linux 7 GnuTLS Cryptographic Module functions in order to meet the FIPS requirements, and the actions that operators must take to maintain the security of the module.

This Security Policy describes the features and design of the Oracle Linux 7 GnuTLS Cryptographic Module using the terminology contained in the FIPS 140-2 specification. FIPS 140-2, Security Requirements for Cryptographic Module specifies the security requirements that will be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information. The NIST/CCCS Cryptographic Module Validation Program (CMVP) validates cryptographic module to FIPS 140-2. Validated products are accepted by the Federal agencies of both the USA and Canada for the protection of sensitive or designated information.

1.2 Document Organization

The FIPS 140-2 Submission Package contains:

- Oracle Linux 7 Gnu TLS Cryptographic Module Non-Proprietary Security Policy
- Other supporting documentation as additional references

With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Validation Documentation is proprietary to Oracle and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact Oracle.

2. Oracle Linux 7 Gnu TLS Cryptographic Module

2.1 Functional Overview

The Oracle Linux 7 Gnu TLS Cryptographic Module is a set of libraries implementing general purpose cryptographic algorithms and network protocols. The module supports the Transport Layer Security (TLS) Protocol defined in [RFC5246] and the Datagram Transport Layer Security (DTLS) Protocol defined in [RFC4347]. The module provides a C language Application Program Interface (API) for use by other calling applications that require cryptographic functionality or TLS/DTLS network protocols.

2.2 FIPS 140-2 Validation Scope

The following table shows the security level for each of the eleven sections of the validation.

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

Table 1: FIPS 140-2 Security Requirements

3. Cryptographic Module Specification

3.1 Definition of the Cryptographic Module

The Oracle Linux 7 GnuTLS Cryptographic Module is defined as a software-only multi-chip standalone module as defined by the requirements within FIPS PUB 140-2. The logical cryptographic boundary of the module consists of shared library files and their integrity check HMAC files, which are delivered through the Package Manager (RPM) as listed below:

Component	Description
libgnutls	This library provides the main interface which allows the calling applications to request cryptographic services. The Approved cryptographic algorithm implementations provided by this library include the TLS protocol, DRBG, RSA Key Generation, Diffie-Hellman and EC Diffie-Hellman.
libnettle	This library provides the cryptographic algorithm implementations, including AES, Triple-DES, SHA, HMAC, RSA Digital Signature, DSA and ECDSA.
libhogweed	This library includes the primitives used by libgnutls and libnettle to support the asymmetric cryptographic operations.
libgmp	This library provides the big number arithmetic operations to support the asymmetric cryptographic operations.
.hmac	The .hmac files contain the HMAC-SHA-256 values of its associated library for integrity check during the power-up.

Table 2: Oracle Linux 7 Gnu TLS Cryptographic Components

The module's logical boundary is the shared library files and their integrity check HMAC files, which are delivered through Oracle Linux Yum Public server listed in section 10.1.

All components of the module will be in the RPM versions gnutls-3.3.29-9.el7_6.x86_64.rpm, gmp-6.0.0-15.el7.x86_64.rpm, nettle-2.7.1-8.el7.x86_64.rpm. The binary files and the HMAC files within the module's logical boundary are listed below:

- libgnutls library:
  - /usr/lib64/libgnutls.so.28.43.3
  - /usr/lib64/.libgnutls.so.28.43.0.hmac (64 bits)

- libnettle library:
  - /usr/lib64/libnettle.so.4.7 (64 bits)
  - /usr/lib64/.libnettle.so.4.7.hmac (64 bits)

- libhogweed library:
  - /usr/lib64/libhogweed.so.2.5 (64 bits)
  - /usr/lib64/.libhogweed.so.2.5.hmac (64 bits)

- libgmp library:
  - /usr/lib64/libgmp.so.10.2.0 (64 bits)
  - /usr/lib64/fipscheck/libgmp.so.10.2.0.hmac (64 bits)

Figure 1 shows the logical block diagram of the module executing in memory on the host system.

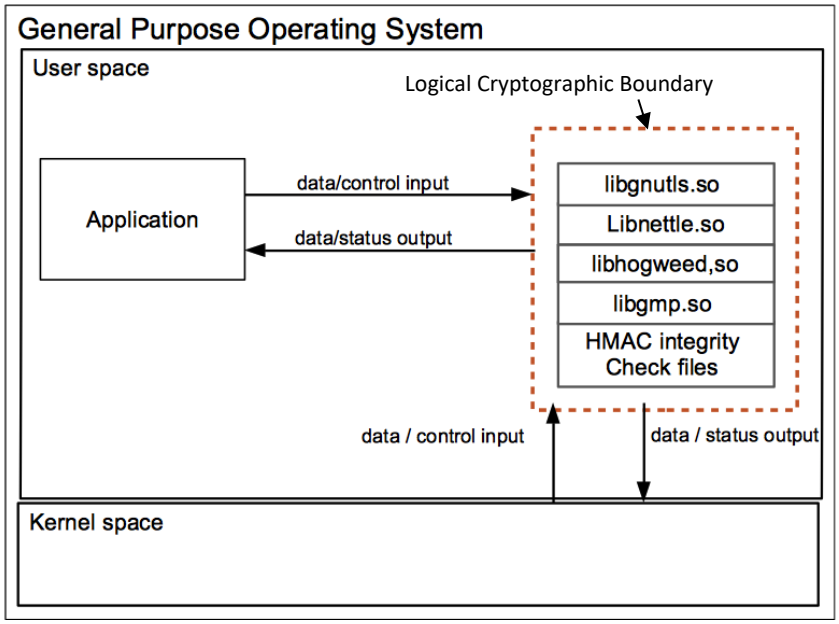

Figure 1: Oracle Linux 7 GnuTLS Logical Cryptographic Boundary

3.2 Definition of the Physical Cryptographic Boundary

The physical boundary of the module is the physical boundary of the test platform described by the dotted line which is a General Purpose Computer (GPC). No components are excluded from the requirements of FIPS PUB 140-2. The following block diagram shows the hardware components of a GPC.

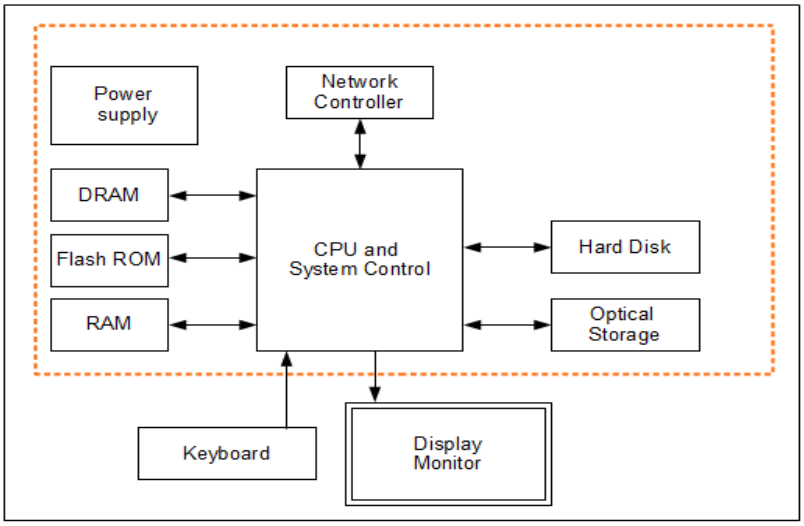

Figure 2: Oracle Linux 7 GnuTLS Hardware Block Diagram

3.3 Description of the Approved Modes of Operation

The module supports two modes of operation:

- In "FIPS mode" (the FIPS Approved mode of operation) only approved or allowed security functions with sufficient security strength can be used.

- In "non-FIPS mode" (the non-Approved mode of operation) only non-approved security functions can be used.

When the module is powered on, after the power-up self-tests are completed successfully, the module will be in FIPS Approved mode by default. Then the mode will be implicitly assumed depending on the services and security functions invoked.

3.4 Approved or Allowed Security Functions

The Oracle Linux 7 Gnu TLS Cryptographic Module contains the following FIPS Approved Algorithms:

Approved or Allowed Security Functions		Certificate
Symmetric Algorithms		
AES	**C Implementation:** **CBC** (e/d; 128 , 192 , 256); **GCM** (KS: AES_128, AES_256) (e/d) Tag Length(s): 128 120 112 104 96 64 32) **PT Lengths Tested:** (0 , 128 , 256 , 120 , 248) ; **AAD Lengths tested:** (0 , 128 , 256 , 120 , 248) ; **96BitIV_Supported GMAC_Supported**	C 1018
	AES-NI Implementation: **CBC** (e/d; 128 , 192 , 256); **GCM** (KS: AES_128, AES_256 (e/d) Tag Length(s): 128 120 112 104 96 64 32) **PT Lengths Tested:** (0, 128, 256, 120, 248) ; **AAD Lengths tested:** (0, 128, 256, 120, 248) ; **96BitIV_Supported** **GMAC_Supported**	C 1001
	SSSE3 Implementation: **CBC** (e/d; 128 , 192 , 256); **GCM** (KS: AES_128, AES_256) (e/d) Tag Length(s): 128 120 112 104 96 64 32) **PT Lengths Tested:** (0 , 128 , 256 , 120 , 248) ; **AAD Lengths tested:** (0 , 128 , 256 , 120 , 248) ; **96BitIV_Supported** **GMAC_Supported**	C 1019
	Nettle Implementation[1]: **ECB** (e/d : 128, 192, 256)	C 929
Triple DES	**C Implementation:** **TCBC** (KO 1 e/d,)	C 1018
Secure Hash Standard (SHS)		
SHS	**C Implementation:** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)	C 1018
	SSSE3 Implementation: **SHA-1** (BYTE-only)	C 1019

[1] *This AES implementation is used internally by the module and is not available externally to the module users.*

Approved or Allowed Security Functions		Certificate
	SHA-224 (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)	
Data Authentication Code		
HMAC	**C Implementation:** **HMAC-SHA1** (Key Size Ranges Tested: KS<BS KS=BS KS>BS) **HMAC-SHA224** (Key Size Ranges Tested: KS<BS KS=BS KS>BS) **HMAC-SHA256** (Key Size Ranges Tested: KS<BS KS=BS KS>BS) **HMAC-SHA384** (Key Size Ranges Tested: KS<BS KS=BS KS>BS) **HMAC-SHA512** (Key Size Ranges Tested: KS<BS KS=BS KS>BS)	C 1018
Asymmetric Algorithms		
RSA	**C Implementation:** **186-4KEY(gen):** FIPS186-4_Random_e **ALG[RSASSA-PKCS1_V1_5]** SIG(gen) (2048 SHA(224, 256, 384, 512)) (3072 SHA(224, 256, 384, 512)) SIG(Ver) (1024 SHA(1, 224, 256, 384, 512)) (2048 SHA(1, 224, 256, 384, 512)) (3072 SHA(1, 224, 256, 384, 512))	C 1018
DSA	**C Implementation:** **FIPS186-4:** **PQG(gen)PARMS TESTED:** [(2048, 224)SHA(384); (2048,256)SHA(384); (3072,256) SHA(384)] **PQG(ver)PARMS TESTED:** [(2048,224) SHA(384); (2048,256) SHA(384); (3072,256) SHA(384)] **KeyPairGen:** [(2048,224) ; (2048,256) ; (3072,256)] **SIG(gen)PARMS TESTED:** [(2048,224) SHA(224 , 256 , 384 , 512); (2048,256) SHA(256 , 384 , 512); (3072,256) SHA(256 , 384 , 512);] **SIG(ver)PARMS TESTED:** [(1024,160) SHA(224, 256, 384, 512); (2048,224) SHA (224, 256, 384, 512); (2048,256) SHA(256, 384, 512); (3072,256) SHA(256, 384, 512)]	C 1018
ECDSA	**FIPS186-4:** **PKG:** CURVES (P-256 P-384 P-521 Testing Candidates) **PKV:** CURVES (P-256 P-384 P-521) **SigGen:** CURVES (P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) **SigVer:** CURVES (P-256: (SHA-1, 224, 256, 384, 512) P-384: (SHA-1, 224, 256, 384, 512) P-521: (SHA-1, 224, 256, 384, 512))	C 1018
Random Number Generation		
DRBG	**C Implementation:** **CTR_DRBG**: [Prediction Resistance Tested: Not Enabled; BlockCipher_No_df: (AES-256)	C 1018
Cryptographic Key Generation		
CKG	NIST SP 800-133	Vendor Affirmed
Key Establishment (All of NIST SP 800-56A Except KDF)		

Approved or Allowed Security Functions		Certificate
Diffie-Hellman and EC Diffie-Hellman	**C Implementation:** **FFC:** (FUNCTIONS INCLUDED IN IMPLEMENTATION: DPG DPV KPG Full Public Key Validation Public Key Regeneration) **SCHEMES: dhEphem:** (KARole: Initiator / Responder) FB FC **ECC:** (FUNCTIONS INCLUDED IN IMPLEMENTATION: DPG DPV KPG Full Public Key Validation Public Key Regeneration) **SCHEMES: EphemUnified:** (KARole: Initiator / Responder) EC: P-256 ED: P-384 EE: P-521 ECDSA	C 1018
Key Derivation (NIST SP 800-135 Section 4.2 in TLS 1.0, 1.1, and 1.2)		
TLS	(TLS1.0/1.1, TLS 1.2 (SHA 256, 384, 512)	C 1018

Table 3: FIPS Approved or Allowed Security Functions

The module supports different AES and SHA implementations based on the underlying platform's capability. The module supports the use of AES-NI and SSSE3 when it is operated in an Intel® x86- 64 architecture environment. When the AES-NI is enabled in the operating environment, the module performs the AES operations using the supports from the AES-NI instructions; when the AES-NI is disabled in the operating environment, the module performs the AES operations using the supports from the Supplemental Streaming SIMD Extensions 3 (SSSE3). The module also performs SHA operations using the supports from the SSSE3. The AES and SHA implementations that uses the AES-NI and SSSE3 supports and their related algorithms have been CAVS tested and functional tested. Although the module implements different implementations for AES and SHA, only one implementation for one algorithm will ever be available for AES, SHA and HMAC cryptographic services at run-time.

3.5 Non-Approved but Allowed Security Functions

The following are considered non-Approved but allowed security functions:

Algorithm	Usage
RSA Key Wrapping	Key wrapping, key establishment methodology provides between 112 and 256 bits of encryption strength.
Diffie-Hellman	Key agreement, key establishment methodology provides between 112 and 256 bits of encryption strength.
EC Diffie-Hellman	Key agreement, key establishment methodology provides between 128 and 256 bits of encryption strength.
NDRNG	Used for seeding NIST SP 800-90A DRBG.
MD5	Message digest used in TLS only.

Table 4: Non-Approved but Allowed Security Functions

3.6 Non-Approved Security Functions

The following services are non-Approved and use of these algorithms will put the module in the non-approved mode of operation implicitly. The services associated with these algorithms are specified in section 7.3:

Algorithm	Usage
AES-CTR Mode	Encrypt/Decrypt
Blowfish	Encrypt/Decrypt
Camellia	Encrypt/Decrypt
CAST-128	Encrypt/Decrypt
DES	Encrypt/Decrypt
Diffie-Hellman	Key agreement using keys less than 2048 bits
RSA	FIPS 186-2 Key Generation
RSA	FIPS 186-4 Key generation, Signature generation, key wrapping with keys less than 2048 bits
RSA	FIPS 186-4 Signature Verification with keys smaller than 1024 bits modulus size
DSA	Parameter /Key generation/Signature generation with keys not listed in Table 3
GOST	GOST Hash R 34.11-94 (RFC4357)
Lagged Fibonacci Pseudo-randomness Generator	Generating random numbers
MD2	Hashing
MD4	Hashing
MD5	Hashing
PBKDF 2	Password based key derivation function
RC2	Encrypt/Decrypt
RC4	Encrypt/Decrypt
RIPEMD-160	Hashing
Salsa-20	Encrypt/Decrypt
Serpent	Encrypt/Decrypt
SHA-1	Signature Generation
SHA-3	Hashing
Twofish	Encrypt/Decrypt
UMAC	Authenticated data integrity of a message
Yarrow RNG	Random number generation

Table 5: Non-Approved Functions

4. Module Ports and Interfaces

As a software-only module, the module does not have physical ports. For the purpose of FIPS 140-2 validation, the physical ports of the module are interpreted to be the physical ports of the hardware platform on which it runs. The logical interface is a C-language Application Program Interface (API) through libgnutls library.

The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the actual API functions. The Status Output interface includes the return values of the API functions. The module can be accessed by utilizing the API it exposes. Table below, shows the mapping of ports and interfaces as per FIPS 140-2 Standard.

FIPS 140 Interface	Physical Port	Module Interfaces
Data Input	Ethernet Ports	API input parameters, kernel I/O – network or files on file system, TLS protocol
Data Output	Ethernet Ports	API output parameters, kernel I/O – network or files on file system, TLS protocol
Control Input	Management Ethernet Port, USB for Keyboard/Mouse, Serial Port	API function calls, TLS protocol
Status Output	Management Ethernet Port, Serial Port	API return codes, error message, TLS protocol

Table 6: Mapping of FIPS 140 Logical Interfaces to Logical Ports

Note: The module is an implementation to support the TLS protocol defined in [RFC5246] and TLS is a port networking interface to provide secure channel between entities. When the calling application sends the data to the module, the module packages the data according to the TLS standard and send to other entity confidentially and integrity. The module is considered as a user interface to use the TLS protocol to communicate with other remote entities securely through the network.

5. Physical Security

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

6. Operational Environment

6.1 Tested Environments

The Module was tested on the following modifiable environment with and without PAA (i.e. AES-NI):

Operating Environment	Processor	Hardware
Oracle Linux 7.6 64-bit	Intel® Xeon® Silver 4114	Oracle Server X7-2
Oracle Linux 7.6 64-bit	AMD® EPYC® 7551	Oracle Server X7-2

Table 7: Tested Operating Environment

6.2 Vendor Affirmed Environments

The following platforms have not been tested as part of the FIPS 140-2 level 1 certification however Oracle "vendor affirms" that these platforms are equivalent to the tested and validated platforms. Additionally, Oracle affirms that the module will function the same way and provide the same security services on any of the systems listed below.

Operating Environment	Processor	Hardware
Oracle Linux 7.8 64-bit	Intel® Xeon® Platinum 8167M	Oracle Server X7-2c
Oracle Linux 7.8 64-bit	AMD® EPYC® 7551	Oracle Server E1
Oracle Linux 7.8 64-bit	Ampere Altra A1	Oracle Server A1
Oracle Linux 7.8 64-bit	Intel® Xeon® Ice Lake-SP	Oracle Server X9
Oracle Linux 7.8 64-bit	Intel® Xeon® Silver 4114	Oracle Server X7-2
Oracle Linux 7.8 64-bit	AMD® EPYC® 7551	Oracle Server X7-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600/E5-2600 v2	Cisco UCS B200 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Cisco UCS B200 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Cisco UCS B200 M5
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2400/E5-2400 v2	Cisco UCS B22 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-2800/E7-8800	Cisco UCS B230 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-2800/E7-8800 v3	Cisco UCS B260 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600/E5-4600 v2	Cisco UCS B420 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600 v3 & v4	Cisco UCS B420 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-2800/E7-8800	Cisco UCS B440 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-2800 v2/E7-4800 v2/E7-8800 v2/E7-4800 v3/E7-8800 v3	Cisco UCS B460 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Cisco UCS B480 M5
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2400/E5-2400 v2	Cisco UCS C22 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600/E5-2600 v2	Cisco UCS C220 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Cisco UCS C220 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Cisco UCS C220 M5
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2400/E5-2400 v2	Cisco UCS C24 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600/E5-2600 v2	Cisco UCS C240 M3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Cisco UCS C240 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Cisco UCS C240 M5
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-2800 v2/E7-4800 v2, v3 & v4/E7-8800 v2 & v4	Cisco UCS C460 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Cisco UCS C480 M5
Oracle Linux 7.6 64-bit	Intel® Xeon® D-1500	Cisco UCS E1120D-M3/K9

Operating Environment	Processor	Hardware
Oracle Linux 7.6 64-bit	Intel® Xeon® D-1500	Cisco UCS E180D-M3/K9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge FC630
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600 v3	Dell PowerEdge FC830
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge M630 Blade
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600 v4	Dell PowerEdge M830 Blade
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge R630
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge R730
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge R730xd
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4	Dell PowerEdge R930
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Dell PowerEdge T630
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2/E7-8800 v2	Fujitsu PRIMEQUEST 2400E
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2400E2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2400E3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2	Fujitsu PRIMEQUEST2400L
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST2400L2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2400L3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2	Fujitsu PRIMEQUEST 2400S
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2	Fujitsu PRIMEQUEST 2400S Lite
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2400S2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2400S2 Lite
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2400S3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2400S3 Lite
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Fujitsu PRIMEQUEST 2800B
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2800B2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2800B3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Fujitsu PRIMEQUEST 2800E
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2800E2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQUEST 2800E3
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Fujitsu PRIMEQUEST 2800L
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Fujitsu PRIMEQUEST 2800L2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v4	Fujitsu PRIMEQEST 2800L3
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Fujitsu PRIMEQUEST 3800B
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Fujitsu PRIMERGY BX2580 M1
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Fujitsu PRIMERGY BX2580 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Fujitsu PRIMERGY CX2560 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Fujitsu PRIMERGY RX2530 M1
Oracle Linux 7.6 64-bit	Intel® Xeon®E5-2600 v4	Fujitsu PRIMERGY RX2530 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Fujitsu PRIMERGY RX2530 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Fujitsu PRIMEGY RX2540 M1
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Fujitsu PRIMERGY RX2540 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Fujitsu PRIMERGY RX2540 M4
Oracle Linux 7.6 64-bit	Intel® Xeon®E7-4800 v2/E7-8800 v2	Fujitsu PRIMERGY RX4770 M1
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v3/E7-8800 v3	Fujitsu PRIMERGY RX4770 M2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	Fujitsu PRIMERGY RX4770 M3

Operating Environment	Processor	Hardware
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Fujitsu PRIMERGY RX4770 M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Hitachi Compute Blade 2500 CB520H B4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Hitachi Compute Blade 2500 CB520X B2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Hitachi Compute Blade 2500 CB520X B3
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Hitachi Compute Blade 500 CB520H B4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Hitachi Compute Blade 500 CB520X B2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Hitachi QuantaGrid D51B-2U
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Hitachi QuantaPlex T41S-2U
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Hitachi Vantara Hitachi Advanced Server DS120
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Hitachi Vantara Hitachi Advanced Server DS220
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Hitachi Vantara Hitachi Advanced Server DS240
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	HPE Integrity MC990 X
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v2	HPE ProLiant BL460c Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	HPE ProLiant BL460c Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600 v3	HPE ProLiant BL660c Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant DL160 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant DL180 Gen9
Oracle Linux 7.6 64-bit	Intel ® Pentium® G2120 & Intel® Xeon® E3-1200 v2	HPE ProLiant DL320e Gen8
Oracle Linux 7.6 64-bit	Intel® Pentium® G3200-series/G3420, Core i3-4100-series/Intel® Xeon® E3-12 v3	HPE ProLiant DL320e Gen8 v2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant DL360 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2400/E5-2400 v2	HPE ProLiant DL360e Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant DL360p Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant DL380 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2400/E5-2400 v2	HPE ProLiant DL380e Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-4600/E5-4600 v2	HPE ProLiant DL560 Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon®E5-4600 v3 & v4	HPE ProLiant DL560 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2/E7-8800 v2	HPE ProLiant DL580 Gen8
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v3/E7-8800 v3	HPE ProLiant DL580 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	HPE ProLiant ML350 Gen9
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	HPE Synergy 480 Gen9 Compute Module
Oracle Linux 7.6 64-bit	Intel® Xeon®E7-4800 v4/E7-8800 v4	HPE Synergy 620 Gen9 Compute Module
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	HPE Synergy 680 Gen9 Compute Module
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer 1288H V5
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer 2288H V5
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer CH121 V5
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer CH121L V5
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer CH242 V5
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Huawei FusionServer RH2288H V3
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable Processors	Huawei FusionServer XH321 V5
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Inspur Yingxin NF5170M4

Operating Environment	Processor	Hardware
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Inspur Yingxin NF5180M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Inspur Yingxin NF5240M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Inspur Yingxin NF5270M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Inspur Yingxin NF5280M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Inspur Yingxin NF5460M4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v3 & v4/E7-8800 v3 & v4	Inspur Yingxin NX8480M4
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100/4100/3100 Processors	Lenovo ThinkSystem SD530
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100/4100/3100 Processors	Lenovo ThinkSystem SN550
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100 Processors	Lenovo ThinkSystem SN850
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100 Processors	Lenovo ThinkSystem SR850
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100 Processors	Lenovo ThinkSystem SR860
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/5100 Processors	Lenovo ThinkSystem SR950
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC Express 5800/A1040d
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC Express 5800/A2010d
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC Express 5800/A2020d
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC Express 5800/A2040d
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC NX7700x/A4010M-4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC NX7700x/A4012L-1
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800/4800 v4	NEC NX7700x/A4012L-1D
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v4	NEC NX7700x/A4012L-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800/4800 v4	NEC NX7700x/A4012L-2D
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v3/E7-8800 v3	NEC NX7700x/A4012M-4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Oracle Netra Server X5-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Oracle Server X5-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Oracle Server X5-2L
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Oracle Server X5-4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v3	Oracle ServerX5-8
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Oracle Server X6-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Oracle Server X6-2L
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v4	Oracle Server X6-2M
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/4100 Processors	Oracle Server X7-2
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100/4100 Processors	Oracle Server X7-2L
Oracle Linux 7.6 64-bit	Intel® Xeon® Scalable 8100/6100 Processors	Oracle Server X7-8
Oracle Linux 7.6 64-bit	Intel® Xeon® x7500-series	Oracle Sun Fire X4470
Oracle Linux 7.6 64-bit	Intel® Xeon® x7500-series	Oracle Sun Fire X4800
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800	Oracle Sun Server X2-8
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800	Oracle Sun Server X2-4
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600	Oracle Sun Server X3-2

Operating Environment	Processor	Hardware
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600	Oracle Sun Server X3-2L
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v2	Oracle Sun Server X4-2
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v2	Oracle Sun Server X4-2L
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Oracle Sun Server X4-4
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-8800 v2	Oracle Sun Server X4-8
Oracle Linux 7.6 64-bit	Intel® Xeon®E7-8800 v3 & v4	SGI UV 300RL
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4/E7-8800 v3 & v4	SGI UV 300
Oracle Linux 7.6 64-bit	AMD Opteron™ 6000	Sugon A840-G10
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Sugon CB50-G20
Oracle Linux 7.6 64-bit	AMD Opteron™ 6000	Sugon A840-G10
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Sugon CB50-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2	Sugon CB80-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v4	Sugon CB80-G25
Oracle Linux 7.6 64-bit	AMD Opteron™ 6300	Sugon CB85-G10
Oracle Linux 7.6 64-bit	Intel® Xeon® 6100, 5100, 4100, 3100	Sugon I420-G30
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Sugon I610-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3	Sugon I620-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v3 & v4	Sugon I840-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2	Sugon I840-G25
Oracle Linux 7.6 64-bit	Intel® Xeon® E7-4800 v2 & v3/E7-8800 v2 & v3	Sugon I980-G20
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Sugon TC4600T
Oracle Linux 7.6 64-bit	Intel® Xeon® E5-2600 v3 & v4	Supermicro SuperServer SYS-6018U-TR4T+

Table 8: Vendor Affirmed Operational Environments

Note: 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.

6.3 Operational Environment Policy

The operating system is restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded).

The application that makes calls to the module is the single user of the module, even when the application is serving multiple clients.

In operational mode, the ptrace(2) system call, the debugger (gdb(1)), and strace(1) shall not be used. In addition, other tracing mechanisms offered by the Linux environment, such as ftrace or systemtap, shall not be used.

7. Roles, Services and Authentication

7.1 Roles

The module supports the following roles:

- **User Role**: performs all services (in both FIPS mode and non-FIPS mode of operation), except module installation.
- **Crypto Officer Role**: performs module installation.

The User and Crypto Officer roles are implicitly assumed by the entity accessing services implemented by the module.

7.2 FIPS Approved Operator Services and Descriptions

The below table provides a full description of FIPS Approved services provided by the module and lists the roles allowed to invoke each service. In the table below, the "U" represents a User Role, and "CO" denotes a Crypto Officer role.

U	CO	Service Name	Service Description	Keys and CSP(s)	Access Type(s)
X		Symmetric Encryption/Decryption	Encrypts or decrypts a block of data using 3-Key Triple-DES or AES	AES or 3-Key Triple-DES Key	W, X
X		Asymmetric Key Generation	Generate RSA, DSA, ECDSA keys	RSA, DSA, ECDSA	W, X
X		Digital Signature Generation and Verification	Sign and verify operations with X509 certificates	RSA, DSA, and ECDSA keys	W, X
X		Public Key Verification	Verifies a public key is valid	RSA, DSA, ECDSA public key	R, X
X		Diffie-Hellman Parameters Generation, Import and Export	Generate, import, and export Diffie-Hellman parameters	Diffie-Hellman domain parameters	R, W, X
X		Import and Export Public Key	Import and export public key components	RSA, DSA, ECDSA public key	R, W, X
X		Import and Export Private Key	Import and export private key components	RSA, DSA, ECDSA private key	R, W, X
X		Keyed Hash (HMAC)	HMAC-SHA services	HMAC keys > 112 bits	R, W, X
X		Hash	SHA hashing services	None	N/A
X		Random Number Generation	Generate random numbers using SP 800-90A DRBG	Entropy input string and seed, DRBG Internal V and Key	R, W, X
X		TLS/DTLS Network Protocol	Provide data encryption and authentication over TLS network protocol	AES, Triple-DES, and HMAC keys.	X
X		TLS/DTLS Key Agreement	Negotiate a TLS key agreement secure channel	AES or Triple-DES key, RSA, DSA or ECDSA private key, HMAC Key, shared secret (pre-master	X, W

U	CO	Service Name	Service Description	Keys and CSP(s)	Access Type(s)
				secret), master secret, Diffie-Hellman and EC Diffie-Hellman Private keys	
X		Show Status	Show status of the module state	None	N/A
X		Self-Test	Initiate power-on self-tests	None	N/A
X		Zeroization	Zeroize all critical security parameters	All keys and CSP's	Z
	X	Module Installation	Installation of the module	None	N/A

R – Read, W – Write, X – Execute, Z - Zeroize

Table 9: FIPS Approved Services

7.3 Non-FIPS Approved Services and Descriptions

The following table lists the non-Approved services available in non-FIPS mode.

U	CO	Service Name	Service Description	Keys	Access Type(s)
X		Asymmetric Encryption/Decryption	Encrypts or decrypts using non-Approved RSA key size	RSA key wrapping with keys less 2048 bits	R, W, X
X		Symmetric Encryption/Decryption	Encrypts or decrypts using non-Approved algorithms	AES-CTR, Blowfish, Camellia, CAST-128, DES, RC2, RC4, Salsa-20, Serpent, Twofish keys	R, W, X
X		Digital Signature Generation	Sign operations with non-Approved keys	RSA or DSA key lengths of < 2048 or with SHA-1	R, W, X
X		Digital Signature Verification	Verify operations with RSA keys < 2048 or verify operations with DSA keys < 1024	RSA keys < 2048 or DSA keys < 1024 bits	R, W, X
X		TLS/DTLS Key Agreement	Negotiate a TLS key agreement with non-Approved key sizes	Diffie-Hellman key lengths < 2048	R, W, X
X		Asymmetric Key Generation	FIPS 186-4 key generation of non-Approved RSA and DSA keys	RSA or DSA keys < 2048	R, W, X
			FIPS 186-2 RSA key generation	RSA, DSA keys	R, W, X
X		Random Number Generation	Generation of random numbers using the lagged Fibonacci PRNG and yarrow RNG	None	N/A
X		Hash	Hashing using non-Approved hash functions that include MD2, MD4, MD5, GOST, RIPEMD-160, and SHA-3	None	N/A

U	CO	Service Name	Service Description	Keys	Access Type(s)
X		UMAC	MAC generation using UMAC	MAC Key	R, W, X
X		PBKDF v2	Creating keys derived from password	Password based generated keys	R, W, X
X		Support to use DANE Certificate	DNS-based Authentication of Named Entities (DANE) is a protocol to allow X.509 certificates, commonly used for TLS.	RSA, DSA and ECDSA public/private keys	R, W, X
X		Support to use OpenPGP Certificate	Use of OpenPGP certificates for a TLS session	RSA, DSA and ECDSA public/private keys	R, W, X
X		Support to use PKCS#11 Certificate	Use of PKCS#11 certificates in GnuTLS	RSA, DSA and ECDSA public/private keys	R, W, X
X		Support to use the Secure RTP (SRTP) defined in RFC5764	a SRTP extension for DTLS	AES and HMAC keys	R, W, X
X		Support to use Trusted Platform Module (TPM)	TPM operations supported and used by GnuTLS limited to TPM 1.2.	RSA, DSA and ECDSA public/private keys	R, W, X

Table 10: Non-FIPS Approved Services

Note: The module does not share CSPs between FIPS mode of operation and a non-FIPS mode of operation. All cryptographic keys used in the FIPS mode of operation must be generated in the FIPS mode or imported while running in the FIPS mode. The DRBG shall not be used for key generation for non-Approved services in non-FIPS mode.

7.4 Operator Authentication

The module does not support operator authentication mechanisms.

8. Key and CSP Management

The following keys, cryptographic key components and other critical security parameters are contained in the module.

CSP Name	Generation	Entry/Output	Storage	Zeroization
AES (CBC & ECB) Keys (128, 192, 256 bits)	The Key is passed into the module via API input parameter or are generated as part of protocol negotiation by applying Key derivation on the shared secret generated during the Diffie-Hellman and EC Diffie- Hellman key agreement.	The key is passed into the module via API input parameters. No output mechanism provided.	RAM in plaintext	gnutls_cipher_deinit()
AES (GCM key) (128, 256 bits)				
Triple-DES Keys (192 bits)			RAM in plaintext	gnutls_cipher_deinit()
HMAC Key (≥ 112 bits)			RAM in plaintext	gnutls_hmac_deinit()
DSA Private Key (2048 and 3072 bits)	Keys are generated using FIPS 186-4 and the random value used in the key generation is generated using SP800-90A DRBG	The key is passed into the module to and from calling application via API parameters.	RAM in plaintext	gnutls_privkey_deinit() or gnutls_x509_privkey_deinit()
ECDSA Private Key (P-256, P-384, P-521)		The key is passed into the module to and from calling application via API parameters.	RAM in plaintext	gnutls_privkey_deinit() Or gnutls_x509_ privkey_deinit()
RSA Private Keys (2048, 3072 bits for Signatures or => 2048 bits as allowed for key wrapping/key establishment)		The key is passed into the module to and from calling application via API parameters.	RAM in plaintext	gnutls_rsa_params_deinit(, gnutls_privkey_deinit() or gnutls_x509 _ privkey_deinit()
Entropy Input String for DRBG Seed	Obtained from NDRNG	The module does not import or export the key or CSP.	RAM in plaintext	gnutls_global_deinit()
DRBG Internal V and Key	Generated internally in the DRBG		RAM in plaintext	gnutls_global_deinit()
Shared Secret (pre-master secret)	Generated using in the Diffie-Hellman or EC Diffie-Hellman key agreement or RSA key transport during handshake.	Output encrypted with RSA only when TLS with RSA key wrapping is used.	RAM in plaintext	gnutls_deinit()
Master secret	Derived from pre-master secret.	Generated inside the module. No output mechanism provided.	RAM in plaintext	gnutls_deinit()
Diffie-Hellman key with at least 2048 bits	Keys are generated using FIPS 186-4 and the random value used in the key	The key is passed into the module to and from calling application via API parameters.	RAM in plaintext	gnutls_deinit() or gnutls_dh_params_deinit()

CSP Name	Generation	Entry/Output	Storage	Zeroization
EC Diffie-Hellman private keys P-256, P-384 and P-521 curves	generation is generated using SP800-90A DRBG	The key is passed into the module to and from calling application via API parameters.	RAM in plaintext	gnutls_deinit() or gnutls_ecdh_params_deinit()

Table 11: Keys/CSPs Table

8.1 Random Number Generation

The module employs a Deterministic Random Bit Generator (DRBG) based on [SP800-90A] for the creation of key components of asymmetric keys and random number generation.

The module implements the CTR_DRBG with AES-256 without derivation function and without prediction resistance. The CTR_DRBG is implemented in the libgnutls library and provides at least 128 bits of output data per each request.

The module uses the output of an NDRNG (i.e. /dev/urandom) as the entropy source for seeding the CTR_DRBG. The NDRNG is implemented in the O/S which is outside of the module's logical boundary within the module's physical boundary. The continuous self-tests on the output of NDRNG is performed by the underlying operating system. The NDRNG provides at least 130 bits of entropy to the DRBG.

The module generates cryptographic keys whose strengths are modified by available entropy.

8.2 Key Generation

The Key Generation methods implemented in the module for Approved services in FIPS mode is compliant with example 1 in Section 4 of [SP 800-133]. Direct output from the DRBG is used as keying material ie K = U.

For generating RSA, DSA and ECDSA keys the module implements asymmetric key generation services compliant with [FIPS186-4] and [SP800-90A].
A seed (i.e. the random value) used in asymmetric key generation is obtained from [SP800-90A] DRBG. The module does not offer a dedicated service for generating keys for symmetric algorithms or for HMAC. However, during TLS protocol negotiation, the symmetric keys are generated by applying key derivation on the shared secret generated during the Diffie-Hellman or EC Diffie-Hellman key agreement. In accordance with FIPS 140-2 IG D.12, the cryptographic module performs Cryptographic Key Generation (CKG) as per SP800-133 (vendor affirmed).

The public and private key pairs used in the Diffie-Hellman and EC Diffie-Hellman KAS are generated internally by the module using the same DSA and ECDSA key generation compliant with [FIPS186-4] which is compliant with [SP800-56A].

8.3 Key Establishment / Key Derivation

The module supports the [SP800-56A] Diffie-Hellman with at least 2048 bits key size and EC Diffie-Hellman with P-256, P-384 or P-521 curve in FIPS mode. The module also supports RSA key wrapping using encryption and decryption primitives with the modulus size of at least 2048. In addition, module provides approved key transport method according to IG D.9 which is used in the TLS protocol context. The key transport method is provided either by using approved authenticated encryption mode i.e. AES GCM or a combination method which includes approved symmetric encryption mode i.e. AES/Triple-DES CBC together with approved authentication method i.e. HMAC-SHA.

- RSA key wrapping provides between 112 and 256 bits of encryption strength;
- Diffie-Hellman key agreement provides between 112 and 256 bits of encryption strength;
- EC Diffie-Hellman key agreement provides between 128 and 256 bits of encryption strength.
- Approved authenticated encryption mode i.e. AES GCM - KTS (AES Certs. #C 1001, #C 1018 and #C 1019; key establishment methodology provides 128 or 256 bits of encryption strength)
- Combination of approved AES encryption and HMAC authentication method - KTS (AES Certs. #C 1001, # 1018 and #C 1019 and HMAC Cert. #C 1018; key establishment methodology provides between 128 and 256 bits of encryption strength)
- Combination of approved Triple-DES encryption and HMAC authentication method - KTS (Triple-DES Cert. #C 1018 and HMAC Cert. #C 1018; key establishment methodology provides 112 bits of encryption strength)

8.4 Key Entry and Output

The module does not support manual key entry or intermediate key generation key output. For symmetric algorithms or for HMAC, the keys are provided to the module via API input parameters for the cryptographic operations. For asymmetric algorithms, the keys are also provided to the module via API input parameters. The module also provides the services to import and export public and private keys to and from calling application only.

8.5 Key / CSP Storage

The module does not support persistent key storage. The key and CSPs are stored as plaintext in the RAM. The keys are provided to the module via API input parameters, and are destroyed by the module using appropriate API function calls before they are released in the memory.
The HMAC key used for integrity test is stored in the .hmac file and relies on the operating system for protection.

8.6 Key / CSP Zeroization

The memory occupied by keys is allocated by regular libc malloc/calloc() calls. The application that uses the module is responsible for calling the appropriate destruction functions from the GnuTLS API to zeroize the keys or keying material. The destruction functions then overwrite the

memory occupied by keys with pre-defined values and deallocates the memory with the free() call. In case of abnormal termination, or swap in/out of a physical memory page of a process, the keys in physical memory are overwritten by the Linux kernel before the physical memory is allocated to another process.

9. Self-Tests

FIPS 140-2 requires that the module perform power-up tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. In addition, some functions require continuous testing of the cryptographic functionality, such as the asymmetric key generation. If any self-test fails, the module returns an error code and enters the error state. No data output or cryptographic operations are allowed in error state. See section 10.3 for descriptions of possible self-test errors and recovery procedures.

9.1 Power-Up Self-Tests

The module performs power-up self-tests automatically when the module is loaded into memory; power-up tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected. Input, output, and cryptographic functions cannot be performed while the module is in a self-test state because the module is single-threaded and will not return to the calling application until the power-up self-tests are completed. If any power-up self-test fails, the module returns the error code listed in section 10.3 and displays “Error in GnuTLS initialization” and then enters error state. The subsequent calls to the module will also fail - thus no further cryptographic operations are possible. If the power-up self-tests complete successfully, the module will return 0 and accepts cryptographic operation services request.

9.1.1 Integrity Tests

The integrity of the module is verified by comparing an HMAC-SHA-256 value calculated at run time with the HMAC value stored in the .hmac file that was computed at build time for each component of the module. If the HMAC values do not match, the test fails and the module enters the error state.

9.1.2 Cryptographic Algorithms Tests

The module performs self-tests on all FIPS-Approved cryptographic algorithms supported in the approved mode of operation, using the known answer tests (KAT) and pair-wise consistency test (PCT), shown in the following table:

Algorithm	Test
AES	CBC KAT with 128 bit key, and GCM KAT with 256 bit key, encryption and decryption are tested separately.
Triple-DES	CBC KAT, encryption and decryption are tested separately.
HMAC	(SHA-1, SHA-224, SHA-256, SHA-384, SHA-512) KAT
SHA	Tested as part of HMAC KAT
DSA	sign and verify KAT with 2048 bit keys and SHA-256.
RSA	sign and verify KAT with 2048 bit keys and SHA-256.
ECDSA	sign and verify KAT with curves P-256 and SHA-256
Diffie-Hellman	Primitive "Z" Computation KAT
EC Diffie-Hellman	Primitive "Z" Computation KAT with P-256 curve
DRBG	CTR_DRBG KAT with AES-256
Module Integrity	HMAC-SHA-256

Table 12: Power-On Self-Tests

For the KAT, the module calculates the result and compares it with the known value. If the answer does not match the known answer, the KAT is failed and the module returns the error code and enters the error state. As described in section 3.3, only one AES or SHA implementation from libnettle library written in C language or using

the support from AES-NI or SSSE3 instructions is available at run-time. The KATs cover different implementations depending on the implementations availability in the operating environment.

9.2 On Demand Self-Tests

The on-demand self-tests is invoked by powering-off and reloading the module which cause the module to run the power-up tests again. During the execution of the on-demand self-tests, services are not available and no data output or input is possible.

9.3 Conditional Self-Tests

The module performs conditional tests on the cryptographic algorithms, using the pair-wise consistency test (PCT) and Continuous Random Number Generator Test (CRNGT), shown in the following table:

Algorithm	Test
DSA	Pairwise Consistency Test: signature generation and verification
ECDSA	Pairwise Consistency Test: signature generation and verification
RSA	Pairwise Consistency Test: signature generation and verification, encryption and decryption
DRBG NIST SP 800-90A	CRNGT is not required per IG 9.8
NDRNG	CRNGT is implemented in the UEK Kernel

Table 13: Conditional Self-Tests

10. Crypto-Officer and User Guidance

This section provides guidance for the Cryptographic Officer and the User to maintain proper use of the module per FIPS 140-2 requirements.

10.1 Crypto-Officer Guidance

The binaries of the module are delivered via Oracle Package Manager (RPM) packages. The Crypto Officer shall follow this Security Policy to configure the operational environment and install the module to be operated as FIPS 140-2 validated module. The version of the RPM packages containing the FIPS validated module are listed in section 3.1.

For proper operation of the in-module integrity verification, the prelink has to be disabled. This can be done by setting PRELINKING=no in the /etc/sysconfig/prelink configuration file. If module were already prelinked, the prelink should be undone on all the system files using the 'prelink -u -a' command.

To configure the operating environment to support FIPS perform the following steps:

1. Insure that the system is registered with the unbreakable Linux Network (ULN) and that the OL7_X86_64_latest channel is enabled
   # yum-config-manager –enable ol7_latest
2. Install the dracut-fips package:
   # yum install dracut-fips
3. Install the dracut-fips-aesni package (if AES-NI is supported):
   To check if AES-NI is supported run:
   # grep aes /proc/cpuinfo
   If it is supported, run:
   # yum install dracut-fips-aesni
4. Recreate the INITRAMFS image:
   # dracut -f
5. Perform the following steps to configure the boot loader so that the system boots into FIPS mode:

   a) Identify the boot partition and the UUID of the partition. If /boot or /boot/efi resides on a separate partition, the kernel parameter boot=<partition of /boot or /boot/efi> must be supplied. The partition can be identified with the command:

      # df /boot or df /boot/efi

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	233191	30454	190296	14%	/boot

      # blkid /dev/sda1

   /dev/sda1: UUID="6046308a-75fc-418e-b284-72d8bfad34ba" TYPE="xfs"

   b) As the root user, edit the /etc/default/grub file as follows:

      i. Add the fips=1 option to the boot loader configuration.
         GRUB_CMDLINE_LINUX="vconsole.font=latarcyrheb-sun16

rd.lvm.lv=ol/swap rd.lvm.lv=ol/root crashkernel=auto
vconsole.keymap=uk rhgb quiet fips=1"

ii. If the contents of /boot reside on a different partition to the root partition, you must use the boot=UUID=boot_UUID line to the boot loader configuration to specify the device that should be mounted onto /boot when the kernel loads.

```
GRUB_CMDLINE_LINUX="vconsole.font=latarcyrheb-sun16
    rd.lvm.lv=ol/swap rd.lvm.lv=ol/root crashkernel=auto
    vconsole.keymap=uk rhgb quiet
    boot=UUID=6046308a-75fc-418e-b284-72d8bfad34ba fips=1"
```

iii. Save the changes.

This is required for FIPS to perform kernel validation checks, where it verifies the kernel against the provided HMAC file in the /boot directory.

Note:
On systems that are configured to boot with UEFI, /boot/efi is located on a dedicated partition as this is formatted specifically to meet UEFI requirements. This does not automatically mean that /boot is located on a dedicated partition.

Only use the boot= parameter if /boot is located on a dedicated partition. If the parameter is specified incorrectly or points to a non-existent device, the system may not boot.

If the system is no longer able to boot, you can try to modify the kernel boot options in grub to specify an alternate device for the boot=UUID=boot_UUID parameter, or remove the parameter entirely.

6. Rebuild the GRUB configuration as follows:

On BIOS-based systems, run the following command:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

On UEFI-based systems, run the following command:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

To ensure proper operation of the in-module integrity verification, prelinking must be disabled on all system files. By default, the prelink package is not installed on the system. However, if it is installed, disable prelinking on all libraries and binaries as follows:

Set PRELINKING=no in the /etc/sysconfig/prelink configuration file.

If the libraries were already prelinked, undo the prelink on all of the system files as follows:

```
# prelink –u –a
```

7. Reboot the system

8. Verify that FIPS Mode is enabled by running the command:

    # cat /proc/sys/crypto/fips_enabled

    The response should be “1”

The Crypto Officer shall check the file /proc/sys/crypto/fips_enabled if the file exists and contains “1”. If the file does not exist or does not contain “1”, the operating environment is not configured to support FIPS and the module will not operate properly. Once the operating environment has been configured to support FIPS, it is not possible to switch back to standard mode without terminating the module first.

After performing the above configuration, the Crypto Officer should proceed for module installation. The RPM package of the Module can be installed by standard tools recommended for the installation of Oracle packages on an Oracle Linux system (for example, yum, RPM, and the RHN remote management tool). The integrity of the RPM is automatically verified during the installation of the Module and the Crypto Officer shall not install the RPM file if the Oracle Linux Yum Server indicates an integrity error. The RPM files listed in section 3 are signed by Oracle and during installation; Yum performs signature verification which ensures as secure delivery of the cryptographic module. If the RPM packages are downloaded manually, then the CO should run ‘rpm –K <rpm-file-name>’ command after importing the builder’s GPG key to verify the package signature. In addition, the CO can also verify the hash of the RPM package to confirm a proper download.

10.2 User Guidance

The applications must be linked dynamically to run the module. Only the services listed in Table 9 are allowed to be used in FIPS mode.

The libraries of GMP and Nettle provides the support of cryptographic operations to the GnuTLS library. The operator shall use the API provided by the GnuTLS library for the services. Invoking the APIs provided by the supporting libraries are forbidden.

10.2.1 TLS and Diffie-Hellman

The TLS protocol implementation provides both, the server and the client sides. As required by SP800-131A, Diffie-Helllman with keys smaller than 2048 bits must not be used any more. The TLS protocol lacks the support to negotiate the used Diffie-Hellman key sizes. To ensure full support for all TLS protocol versions, the TLS client implementation of the cryptographic Module accepts Diffie-Hellman key sizes smaller than 2048 bits offered by the TLS server.

For complying with the requirement to not allow Diffie-Hellman key sizes smaller than 2048 bits, the Crypto Officer must ensure that:

- in case the module is used as TLS server, the Diffie-Hellman domain parameters must be 2048 bits or larger;
- in case the module is used as TLS client, the TLS server must be configured to only offer Diffie-Hellman domain parameters of 2048 bits or larger.

10.2.2 AES GCM IV Guidance

AES GCM encryption and decryption are used in the context of the TLS protocol version 1.2 (compliant to Scenario 1 in [FIPS140-2_IG] A.5). The module is compliant with [NIST SP 800-52] and the mechanism for IV generation is compliant with [RFC 5288]. The operations of one of the two parties involved in the TLS key establishment scheme are performed entirely within the cryptographic boundary of the module, including the setting of the counter portion of the IV.

When the nonce_explicit part of the IV exhausts the maximum number of possible values for a given session key, the module (acting as server or client) triggers a handshake to establish a new encryption key per Section 7.4.1.1 and Section 7.4.1.2 in [RFC 5246] and compliant to [FIPS140-2_IG] A.5.

In case the module's power is lost and then restored, the key used for AES GCM encryption or decryption shall be re-distributed.

10.2.3 RSA and DSA Keys

The module allows the use of 1024 bit RSA and DSA keys for legacy purposes, including signature generation.

As per SP 800-131A, RSA and DSA must be used at least 2048 bit keys in FIPS mode. To comply with the requirements of [FIPS 140-2], the operator must therefore only use keys with at least 2048 bits in FIPS mode.

10.2.4 Symmetric Key Generation

The API function gnutls_key_generate() shall not be used in FIPS mode of operation. The caller shall call gnutls_rnd() which calls the DRBG compliant to [SP800-90A] to generate the key materials for symmetric keys or HMAC keys.

10.3 Handling Self-Test Errors

When the module fails any self-test, it will return an error code to indicate the error and enters error state that any further cryptographic operations is inhibited. Here is the list of error codes when the module fails any self-test or in error state:

Error Events	Error Codes	Error Messages
When the KAT or Integrity fails at the power-up	GNUTLS_E_SELF_TEST_ERROR (-400)	"Error while performing self checks."
When the KAT of DRBG fails at the power-up	GNUTLS_E_RANDOM_FAILED (-206)	"Error while performing self checks."
When the new generated RSA, DSA or ECDSA key pair fails the PCT	GNUTLS_E_PK_GENERATION_ERROR (-403)	"Error in public key generation."
When the module is in error state and caller requests cryptographic operations	GNUTLS_E_LIB_IN_ERROR_STATE (-402)	"An error has been detected in the library and cannot continue operations."

Table 14: Error Events and Error Messages

Self-test errors transition the module into an error state that keeps the module operational but prevents any cryptographic related operations. The module must be restarted and perform power-up self-test to recover from these errors. If failures persist, the module must be re-installed.

A completed list of the error codes can be found in Appendix C “Error Codes and Descriptions” in the gnutls.pdf provided with the module's code.

11. Mitigation of Other Attacks

RSA is vulnerable to timing attacks. In a setup where attackers can measure the time of RSA decryption or signature operations, blinding is always used to protect the RSA operation from that attack.

The internal API function of _rsa_blind() and _rsa_unblind() are called by the module for RSA signature generation and RSA decryption operations. The module generates a random blinding factor and include this random value in the RSA operations to prevent RSA timing attacks.

Acronyms, Terms and Abbreviations

Term	Definition
AES	Advanced Encryption Standard
CAVP	Cryptographic Algorithm Validation Program
CCCS	Canadian Centre for Cyber Security
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DH	Diffie-Hellman
DHE	Diffie-Hellman Ephemeral
DRBG	Deterministic Random Bit Generator
DTLS	Datagram Transport Layer Security
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
EDC	Error Detection Code
HMAC	(Keyed) Hash Message Authentication Code
IKE	Internet Key Exchange
KAT	Known Answer Test
KDF	Key Derivation Function
NDF	No Derivation Function
NDRNG	Non Deterministic Random Number Generator
NIST	National Institute of Standards and Technology
PBKDF	Password Based Key Derivation Function
PAA	Processor Algorithm Acceleration
POST	Power On Self-Test
PR	Prediction Resistance
PSS	Probabilistic Signature Scheme
PUB	Publication
SHA	Secure Hash Algorithm
TLS	Transport Layer Security

Table 15: Acronyms

References

The FIPS 140-2 standard, and information on the CMVP, can be found at http://csrc.nist.gov/groups/STM/cmvp/index.html. More information describing the module can be found on the Oracle web site at https://www.oracle.com/linux/ .

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

Document	Author	Title
FIPS PUB 140-2	NIST	FIPS PUB 140-2: Security Requirements for Cryptographic Modules
FIPS IG	NIST	Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program
FIPS PUB 140-2 Annex A	NIST	FIPS 140-2 Annex A: Approved Security Functions
FIPS PUB 140-2 Annex B	NIST	FIPS 140-2 Annex B: Approved Protection Profiles
FIPS PUB 140-2 Annex C	NIST	FIPS 140-2 Annex C: Approved Random Number Generators
FIPS PUB 140-2 Annex D	NIST	FIPS 140-2 Annex D: Approved Key Establishment Techniques
DTR for FIPS PUB 140-2	NIST	Derived Test Requirements (DTR) for FIPS PUB 140-2, Security Requirements for Cryptographic Modules
NIST SP 800-67	NIST	Recommendation for the Triple Data Encryption Algorithm TDEA Block Cipher
FIPS PUB 197	NIST	Advanced Encryption Standard
FIPS PUB 198-1	NIST	The Keyed Hash Message Authentication Code (HMAC)
FIPS PUB 186-4	NIST	Digital Signature Standard (DSS)
FIPS PUB 180-4	NIST	Secure Hash Standard (SHS)
NIST SP 800-131A	NIST	Recommendation for the Transitioning of Cryptographic Algorithms and Key Sizes
PKCS#1	RSA Laboratories	PKCS#1 v2.1: RSA Cryptographic Standard

Table 16: References