FIPS 140-3 Non-Proprietary Security Policy

Oracle Corporation

Oracle Linux 9 libgcrypt Cryptographic Module

Software Version: 1.10.0-3957adb8de08b15a

Prepared by:

atsec information security corporation

4516 Seton Center Parkway, Suite 250

Austin, TX 78759

www.atsec.com

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

Title: Oracle Linux 9 libgcrypt Cryptographic Module Security Policy
Date: March 13th, 2025

Contributing Authors:
Oracle Linux Engineering
Security Evaluations – Global Product Security
atsec information security

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

Oracle is committed to developing practices and products that help protect the environment

Copyright © 2025, 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

Table of Contents ..... **ii**
List of Tables ..... **iv**
List of Figures ..... **v**
1 General **1**
1.1 Overview 1
1.1.1 How This Security Policy was Prepared 1
1.2 Security Levels 1
2 Cryptographic Module Specification **2**
2.1 Description 2
2.2 Tested and Vendor Affirmed Module Version and Identification 3
2.3 Excluded Components 3
2.4 Modes of Operation 3
2.5 Algorithms 4
2.6 Security Function Implementations 8
2.7 Algorithm Specific Information 13
2.7.1 AES XTS 13
2.7.2 Key Derivation using SP 800-132 PBKDF 13
2.8 RBG and Entropy 13
2.9 Key Generation 14
2.10 Key Establishment 14
2.11 Industry Protocols 14
3 Cryptographic Module Interfaces **15**
3.1 Ports and Interfaces 15
4 Roles, Services, and Authentication **16**
4.2 Roles 16
4.3 Approved Services 16
4.4 Non-Approved Services 19
5 Software/Firmware Security **21**
5.1 Integrity Techniques 21
5.2 Initiate on Demand 21
6 Operational Environment **22**
6.1 Operational Environment Type and Requirements 22
6.2 Configuration Settings and Restrictions 22
7 Physical Security **23**
8 Non-Invasive Security **24**
9 Sensitive Security Parameters Management **25**

9.1 Storage Areas 25
9.2 SSP Input-Output Methods 25
9.3 SSP Zeroization Methods......25
9.4 SSPs 26
9.5 Transitions 28
10 Self-Tests 29
10.1 Pre-Operational Self-Tests 29
10.2 Conditional Self-Tests......29
10.3 Periodic Self-Test Information......39
10.4 Error States......42
10.5 Operator Initiation of Self-Tests......42
11 Life-Cycle Assurance......43
11.1 Installation, Initialization, and Startup Procedures 43
11.2 Administrator Guidance 43
11.3 Non-Administrator Guidance 43
11.4 End of Life......43
12 Mitigation of Other Attacks 44
12.1 Attack List......44
12.2 Mitigation Effectiveness......44
Appendix A. Approved Public Key Flags 45
Appendix B. Glossary and Abbreviations 46
Appendix C. References 47

List of Tables

Table 1: Security Levels....1
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)....3
Table 3: Tested Operational Environments - Software, Firmware, Hybrid....3
Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid....3
Table 5: Modes List and Description....4
Table 6: Approved Algorithms....7
Table 7: Vendor-Affirmed Algorithms....8
Table 8: Non-Approved, Not Allowed Algorithms....8
Table 9: Security Function Implementations....13
Table 10: Entropy Certificates....13
Table 11: Entropy Sources....14
Table 12: Ports and Interfaces....15
Table 13: Roles....16
Table 14: Approved Services....18
Table 15: Non-Approved Services....19
Table 16: Storage Areas....25
Table 17: SSP Input-Output Methods....25
Table 18: SSP Zeroization Methods....25
Table 19: SSP Table 1....27
Table 20: SSP Table 2....28
Table 21: Pre-Operational Self-Tests....29
Table 22: Conditional Self-Tests....39
Table 23: Pre-Operational Periodic Information....39
Table 24: Conditional Periodic Information....42
Table 25: Error States....42

List of Figures

Figure 1: Block Diagram 2

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version 1.10.0-3957adb8de08b15a of the Oracle Linux 9 libgcrypt Cryptographic Module. It 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.

This Non-Proprietary Security Policy may be reproduced and distributed, but only whole and intact and including this notice. Other documentation is proprietary to their authors.

1.1.1 How This Security Policy was Prepared

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, wherein the Security Policy was submitted to the vendor, who would then edit, modify, and add technical contents. The vendor would also supply additional documentation, which the laboratory formatted into the existing Security Policy, and resubmitted to the vendor for their final editing.

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	1
	Overall Level	1

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The Oracle Linux 9 libgcrypt Cryptographic Module (hereafter referred to as “the module”) is a Software multi-chip standalone cryptographic module. The module is a software library implementing general purpose cryptographic algorithms. The module provides cryptographic services to applications running in the user space of the underlying operating system through an application program interface (API).

Module Type: Software

Module Embodiment: MultiChipStand

Cryptographic Boundary:

Figure 1 shows the cryptographic boundary of the module in orange, its interfaces with the operational environment and the flow of information between the module and operator (depicted through the arrows). The software component of the cryptographic module is listed in the *Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)* table.

Tested Operational Environment’s Physical Perimeter (TOEPP):

The TOEPP of the module is defined as the general-purpose computer on which the module is installed.

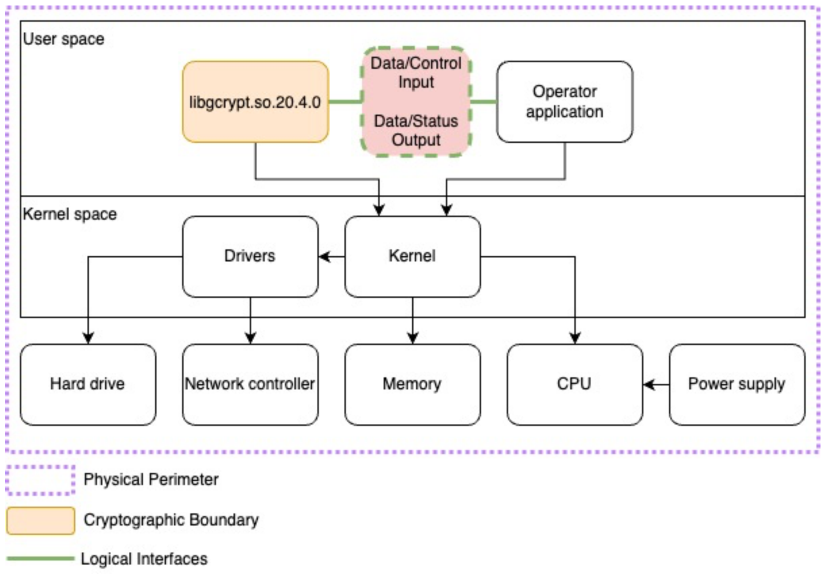

Figure 1: Block Diagram

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
libgcrypt.so.20.4.0 on Oracle Linux 9 with Intel(R) Xeon(R) Platinum 8358	1.10.0-3957adb8de08b15a	N/A	HMAC-SHA-256
libgcrypt.so.20.4.0 on Oracle Linux 9 with AMD EPYC 7J13	1.10.0-3957adb8de08b15a	N/A	HMAC-SHA-256
libgcrypt.so.20.4.0 on Oracle Linux 9 with Ampere(R) Altra(R) Q80-30	1.10.0-3957adb8de08b15a	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)
Oracle Linux 9	ORACLE SERVER X9-2c	Intel(R) Xeon(R) Platinum 8358	Yes	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a
Oracle Linux 9	ORACLE SERVER E4-2c	AMD EPYC 7J13	Yes	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a
Oracle Linux 9	ORACLE SERVER A1-2c	Ampere(R) Altra(R) Q80-30	Yes	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a
Oracle Linux 9	ORACLE SERVER X9-2c	Intel(R) Xeon(R) Platinum 8358	No	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a
Oracle Linux 9	ORACLE SERVER E4-2c	AMD EPYC 7J13	No	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a
Oracle Linux 9	ORACLE SERVER A1-2c	Ampere(R) Altra(R) Q80-30	No	KVM on Oracle Linux 8	1.10.0-3957adb8de08b15a

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

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

Operating System	Hardware Platform
Oracle Linux 9	Oracle X Series Servers
Oracle Linux 9	Oracle E Series Servers
Oracle Linux 9	Oracle A Series Servers
Oracle Linux 9	Marvell T93 LiquidIO III (ARM v8.x) SmartNIC
Oracle Linux 9	Pensando DSC-200-R (ARM v8.x) SmartNIC
Oracle Linux 9	Oracle Server X7-2L

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

There are no components within the cryptographic boundary excluded from the FIPS 140-3 requirements.

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	Equivalent to the indicator of the requested service as defined in section 4.3
Non-approved mode	Automatically entered whenever a non-approved service is requested	Non-Approved	Equivalent to the indicator of the requested service as defined in section 4.4

Table 5: Modes List and Description

Mode Change Instructions and Status:

After passing all pre-operational self-test and cryptographic algorithm self-tests (CASTs) executed on start-up, the module automatically transitions to the approved mode. No operator intervention is required to reach this point. The module automatically switches between the approved and non-approved modes.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A4773	-	SP 800-38A
AES-CBC	A4774	-	SP 800-38A
AES-CBC	A4776	-	SP 800-38A
AES-CBC	A4777	-	SP 800-38A
AES-CCM	A4773	-	SP 800-38C
AES-CCM	A4774	-	SP 800-38C
AES-CCM	A4776	-	SP 800-38C
AES-CCM	A4777	-	SP 800-38C
AES-CFB128	A4773	-	SP 800-38A
AES-CFB128	A4774	-	SP 800-38A
AES-CFB128	A4776	-	SP 800-38A
AES-CFB128	A4777	-	SP 800-38A
AES-CFB8	A4773	-	SP 800-38A
AES-CFB8	A4774	-	SP 800-38A
AES-CFB8	A4776	-	SP 800-38A
AES-CFB8	A4777	-	SP 800-38A
AES-CMAC	A4773	-	SP 800-38B
AES-CMAC	A4774	-	SP 800-38B
AES-CMAC	A4776	-	SP 800-38B
AES-CMAC	A4777	-	SP 800-38B
AES-CTR	A4773	-	SP 800-38A
AES-CTR	A4774	-	SP 800-38A
AES-CTR	A4776	-	SP 800-38A
AES-CTR	A4777	-	SP 800-38A
AES-ECB	A4773	-	SP 800-38A
AES-ECB	A4774	-	SP 800-38A
AES-ECB	A4776	-	SP 800-38A
AES-ECB	A4777	-	SP 800-38A
AES-KW	A4773	-	SP 800-38F
AES-KW	A4774	-	SP 800-38F
AES-KW	A4776	-	SP 800-38F
AES-KW	A4777	-	SP 800-38F
AES-OFB	A4773	-	SP 800-38A
AES-OFB	A4774	-	SP 800-38A
AES-OFB	A4776	-	SP 800-38A
AES-OFB	A4777	-	SP 800-38A
AES-XTS Testing Revision 2.0	A4773	-	SP 800-38E
AES-XTS Testing Revision 2.0	A4774	-	SP 800-38E
AES-XTS Testing Revision 2.0	A4776	-	SP 800-38E

Algorithm	CAVP Cert	Properties	Reference
AES-XTS Testing Revision 2.0	A4777	-	SP 800-38E
Counter DRBG	A4773	-	SP 800-90A Rev. 1
Counter DRBG	A4774	-	SP 800-90A Rev. 1
Counter DRBG	A4776	-	SP 800-90A Rev. 1
Counter DRBG	A4777	-	SP 800-90A Rev. 1
ECDSA KeyGen (FIPS186-4)	A4773	-	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A4774	-	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A4776	-	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A4777	-	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A4778	-	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4773	-	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4774	-	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4776	-	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4777	-	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4778	-	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4773	-	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4774	-	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4776	-	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4777	-	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4778	-	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4773	-	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4774	-	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4776	-	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4777	-	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4778	-	FIPS 186-4
Hash DRBG	A4773	-	SP 800-90A Rev. 1
Hash DRBG	A4774	-	SP 800-90A Rev. 1
Hash DRBG	A4776	-	SP 800-90A Rev. 1
Hash DRBG	A4777	-	SP 800-90A Rev. 1
Hash DRBG	A4778	-	SP 800-90A Rev. 1
HMAC DRBG	A4773	-	SP 800-90A Rev. 1
HMAC DRBG	A4774	-	SP 800-90A Rev. 1
HMAC DRBG	A4776	-	SP 800-90A Rev. 1
HMAC DRBG	A4777	-	SP 800-90A Rev. 1
HMAC DRBG	A4778	-	SP 800-90A Rev. 1
HMAC-SHA-1	A4772	-	FIPS 198-1
HMAC-SHA-1	A4773	-	FIPS 198-1
HMAC-SHA-1	A4774	-	FIPS 198-1
HMAC-SHA-1	A4775	-	FIPS 198-1
HMAC-SHA-1	A4776	-	FIPS 198-1
HMAC-SHA-1	A4777	-	FIPS 198-1
HMAC-SHA-1	A4778	-	FIPS 198-1
HMAC-SHA2-224	A4773	-	FIPS 198-1
HMAC-SHA2-224	A4774	-	FIPS 198-1
HMAC-SHA2-224	A4776	-	FIPS 198-1
HMAC-SHA2-224	A4777	-	FIPS 198-1
HMAC-SHA2-224	A4778	-	FIPS 198-1
HMAC-SHA2-256	A4773	-	FIPS 198-1
HMAC-SHA2-256	A4774	-	FIPS 198-1
HMAC-SHA2-256	A4776	-	FIPS 198-1
HMAC-SHA2-256	A4777	-	FIPS 198-1
HMAC-SHA2-256	A4778	-	FIPS 198-1
HMAC-SHA2-384	A4773	-	FIPS 198-1
HMAC-SHA2-384	A4774	-	FIPS 198-1
HMAC-SHA2-384	A4776	-	FIPS 198-1
HMAC-SHA2-384	A4777	-	FIPS 198-1
HMAC-SHA2-384	A4778	-	FIPS 198-1
HMAC-SHA2-512	A4773	-	FIPS 198-1
HMAC-SHA2-512	A4774	-	FIPS 198-1
HMAC-SHA2-512	A4776	-	FIPS 198-1
HMAC-SHA2-512	A4777	-	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-512	A4778	-	FIPS 198-1
HMAC-SHA2-512/224	A4773	-	FIPS 198-1
HMAC-SHA2-512/224	A4774	-	FIPS 198-1
HMAC-SHA2-512/224	A4776	-	FIPS 198-1
HMAC-SHA2-512/224	A4777	-	FIPS 198-1
HMAC-SHA2-512/224	A4778	-	FIPS 198-1
HMAC-SHA2-512/256	A4773	-	FIPS 198-1
HMAC-SHA2-512/256	A4774	-	FIPS 198-1
HMAC-SHA2-512/256	A4776	-	FIPS 198-1
HMAC-SHA2-512/256	A4777	-	FIPS 198-1
HMAC-SHA2-512/256	A4778	-	FIPS 198-1
HMAC-SHA3-224	A4773	-	FIPS 198-1
HMAC-SHA3-224	A4774	-	FIPS 198-1
HMAC-SHA3-224	A4778	-	FIPS 198-1
HMAC-SHA3-256	A4773	-	FIPS 198-1
HMAC-SHA3-256	A4774	-	FIPS 198-1
HMAC-SHA3-256	A4778	-	FIPS 198-1
HMAC-SHA3-384	A4773	-	FIPS 198-1
HMAC-SHA3-384	A4774	-	FIPS 198-1
HMAC-SHA3-384	A4778	-	FIPS 198-1
HMAC-SHA3-512	A4773	-	FIPS 198-1
HMAC-SHA3-512	A4774	-	FIPS 198-1
HMAC-SHA3-512	A4778	-	FIPS 198-1
PBKDF	A4773	-	SP 800-132
PBKDF	A4774	-	SP 800-132
PBKDF	A4776	-	SP 800-132
PBKDF	A4777	-	SP 800-132
PBKDF	A4778	-	SP 800-132
RSA KeyGen (FIPS186-4)	A4773	-	FIPS 186-4
RSA KeyGen (FIPS186-4)	A4774	-	FIPS 186-4
RSA KeyGen (FIPS186-4)	A4776	-	FIPS 186-4
RSA KeyGen (FIPS186-4)	A4777	-	FIPS 186-4
RSA KeyGen (FIPS186-4)	A4778	-	FIPS 186-4
RSA SigGen (FIPS186-4)	A4773	-	FIPS 186-4
RSA SigGen (FIPS186-4)	A4774	-	FIPS 186-4
RSA SigGen (FIPS186-4)	A4776	-	FIPS 186-4
RSA SigGen (FIPS186-4)	A4777	-	FIPS 186-4
RSA SigGen (FIPS186-4)	A4778	-	FIPS 186-4
RSA SigVer (FIPS186-2)	A4773	-	FIPS 186-4
RSA SigVer (FIPS186-2)	A4774	-	FIPS 186-4
RSA SigVer (FIPS186-2)	A4776	-	FIPS 186-4
RSA SigVer (FIPS186-2)	A4777	-	FIPS 186-4
RSA SigVer (FIPS186-2)	A4778	-	FIPS 186-4
RSA SigVer (FIPS186-4)	A4773	-	FIPS 186-4
RSA SigVer (FIPS186-4)	A4774	-	FIPS 186-4
RSA SigVer (FIPS186-4)	A4776	-	FIPS 186-4
RSA SigVer (FIPS186-4)	A4777	-	FIPS 186-4
RSA SigVer (FIPS186-4)	A4778	-	FIPS 186-4
SHA-1	A4772	-	FIPS 180-4
SHA-1	A4773	-	FIPS 180-4
SHA-1	A4774	-	FIPS 180-4
SHA-1	A4775	-	FIPS 180-4
SHA-1	A4776	-	FIPS 180-4
SHA-1	A4777	-	FIPS 180-4
SHA-1	A4778	-	FIPS 180-4
SHA2-224	A4773	-	FIPS 180-4
SHA2-224	A4774	-	FIPS 180-4
SHA2-224	A4776	-	FIPS 180-4
SHA2-224	A4777	-	FIPS 180-4
SHA2-224	A4778	-	FIPS 180-4
SHA2-256	A4773	-	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA2-256	A4774	-	FIPS 180-4
SHA2-256	A4776	-	FIPS 180-4
SHA2-256	A4777	-	FIPS 180-4
SHA2-256	A4778	-	FIPS 180-4
SHA2-384	A4773	-	FIPS 180-4
SHA2-384	A4774	-	FIPS 180-4
SHA2-384	A4776	-	FIPS 180-4
SHA2-384	A4777	-	FIPS 180-4
SHA2-384	A4778	-	FIPS 180-4
SHA2-512	A4773	-	FIPS 180-4
SHA2-512	A4774	-	FIPS 180-4
SHA2-512	A4776	-	FIPS 180-4
SHA2-512	A4777	-	FIPS 180-4
SHA2-512	A4778	-	FIPS 180-4
SHA2-512/224	A4773	-	FIPS 180-4
SHA2-512/224	A4774	-	FIPS 180-4
SHA2-512/224	A4776	-	FIPS 180-4
SHA2-512/224	A4777	-	FIPS 180-4
SHA2-512/224	A4778	-	FIPS 180-4
SHA2-512/256	A4773	-	FIPS 180-4
SHA2-512/256	A4774	-	FIPS 180-4
SHA2-512/256	A4776	-	FIPS 180-4
SHA2-512/256	A4777	-	FIPS 180-4
SHA2-512/256	A4778	-	FIPS 180-4
SHA3-224	A4773	-	FIPS 202
SHA3-224	A4774	-	FIPS 202
SHA3-224	A4778	-	FIPS 202
SHA3-256	A4773	-	FIPS 202
SHA3-256	A4774	-	FIPS 202
SHA3-256	A4778	-	FIPS 202
SHA3-384	A4773	-	FIPS 202
SHA3-384	A4774	-	FIPS 202
SHA3-384	A4778	-	FIPS 202
SHA3-512	A4773	-	FIPS 202
SHA3-512	A4774	-	FIPS 202
SHA3-512	A4778	-	FIPS 202
SHAKE-128	A4773	-	FIPS 202
SHAKE-128	A4774	-	FIPS 202
SHAKE-128	A4778	-	FIPS 202
SHAKE-256	A4773	-	FIPS 202
SHAKE-256	A4774	-	FIPS 202
SHAKE-256	A4778	-	FIPS 202

Table 6: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
CKG	RSA:2048, 3072, 4096 (112, 128, 149 bits) ECDSA:P-224, P-256, P-384, P-512 (112, 128, 192, 256 bits)	Oracle Linux 9 libgcrypt Cryptographic Module (Full Acceleration)	FIPS 186-4, SP 800-133rev2 Section 5.1
CKG	RSA:2048, 3072, 4096 (112, 128, 149 bits) ECDSA:P-224, P-256, P-384, P-512 (112, 128, 192, 256 bits)	Oracle Linux 9 libgcrypt Cryptographic Module (No Acceleration)	FIPS 186-4, SP 800-133rev2 Section 5.1
CKG	RSA:2048, 3072, 4096 (112, 128, 149 bits) ECDSA:P-224, P-256, P-384. P-521 (112, 128, 192, 256 bits)	Oracle Linux 9 libgcrypt Cryptographic Module (AESNI AVX)	FIPS 186-4, SP 800-133rev2 Section 5.1
CKG	RSA:2048, 3072, 4096 (112, 128, 149 bits) ECDSA:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits)	Oracle Linux 9 libgcrypt Cryptographic Module (SHLD)	FIPS 186-4, SP 800-133rev2 Section 5.1

Name	Properties	Implementation	Reference
CKG	RSA:2048, 3072, 4096 (112, 128, 149 bits) ECDSA:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits)	Oracle Linux 9 libgcrypt Cryptographic Module (SSSE3)	FIPS 186-4, SP 800-133rev2 Section 5.1

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
MD5	Message digest
ECDH noncompliant with SP 800-56Arev3 assurances	Shared secret computation
AES GCM noncompliant with IG C.H.	Authenticated symmetric encryption, Authenticated symmetric decryption
AES GCM-SIV	Authenticated symmetric encryption, Authenticated symmetric decryption
AES OCB	Authenticated symmetric encryption, Authenticated symmetric decryption
AES-EAX	Authenticated symmetric encryption, Authenticated symmetric decryption
RSA	Signature generation primitive; Signature verification primitive; Encryption primitive; Decryption primitive
RSA with non-approved flags that are not listed in Appendix A	Key generation; Signature generation; Signature verification
ECDSA	Signature generation primitives, Signature verification primitives
ECDSA with non-approved flags that are not listed in Appendix A	Key generation; Key verification; Signature generation; Signature verification

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Key wrapping using AES CCM	KTS-Wrap	Key wrapping using AES CCM	Key:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM AES-CCM AES-CCM AES-CCM
Key unwrapping using AES CCM	KTS-Wrap	Key unwrapping using AES CCM	Key:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM AES-CCM AES-CCM AES-CCM
Key wrapping using AES KW	KTS-Wrap	Key wrapping using AES KW	Key:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-KW AES-KW AES-KW AES-KW
Key unwrapping using AES KW	KTS-Wrap	Key unwrapping using AES KW	Key:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-KW AES-KW AES-KW AES-KW
Encryption with AES	BC-UnAuth	Encryption using AES	Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC

Name	Type	Description	Properties	Algorithms
				AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0
Authenticated encryption with AES	BC-Auth	Authenticated encryption using AES	Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM AES-CCM AES-CCM AES-CCM AES-KW AES-KW AES-KW AES-KW
Decryption with AES	BC-UnAuth	Decryption using AES	Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0
Authenticated decryption with AES	BC-Auth	Authenticated decryption using AES	Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM AES-CCM AES-CCM AES-CCM AES-KW AES-KW AES-KW AES-KW

Name	Type	Description	Properties	Algorithms
Key Pair Generation with RSA	CKG	Key pair generation for RSA	Mode:B.3.3 Random Probable Primes Modulus:2048, 3072, 4096 bits (112, 128, 149 bits)	RSA KeyGen (FIPS186-4) RSA KeyGen (FIPS186-4) RSA KeyGen (FIPS186-4) RSA KeyGen (FIPS186-4) RSA KeyGen (FIPS186-4)
Key Pair Generation with ECDSA	CKG	Key pair generation for ECDSA	Mode:B.4.2 Testing Candidates Curves:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits)	ECDSA KeyGen (FIPS186-4) ECDSA KeyGen (FIPS186-4) ECDSA KeyGen (FIPS186-4) ECDSA KeyGen (FIPS186-4) ECDSA KeyGen (FIPS186-4)
Public Key Verification with ECDSA	AsymKeyPair-KeyVer	Verify public key for ECDSA	Curves:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits)	ECDSA KeyVer (FIPS186-4) ECDSA KeyVer (FIPS186-4) ECDSA KeyVer (FIPS186-4) ECDSA KeyVer (FIPS186-4) ECDSA KeyVer (FIPS186-4)
Signature Generation with RSA	DigSig-SigGen	Digital signature generation using RSA	Padding:PKCS#1 v1.5, PSS Keys:2048, 3072, 4096 bits (112, 128, 149 bits) Hashes:SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256	RSA SigGen (FIPS186-4) RSA SigGen (FIPS186-4) RSA SigGen (FIPS186-4) RSA SigGen (FIPS186-4) RSA SigGen (FIPS186-4)
Signature Verification with RSA	DigSig-SigVer	Digital signature verification using RSA	Padding:PKCS#1 v1.5, PSS Keys:1024, 1536, 2048, 3072, 4096 bits (80, 96, 112, 128, 149 bits) Hashes:SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256	RSA SigVer (FIPS186-4) RSA SigVer (FIPS186-4) RSA SigVer (FIPS186-4) RSA SigVer (FIPS186-4) RSA SigVer (FIPS186-4) RSA SigVer (FIPS186-2) RSA SigVer (FIPS186-2) RSA SigVer (FIPS186-2) RSA SigVer (FIPS186-2) RSA SigVer (FIPS186-2)
Signature Generation with ECDSA	DigSig-SigGen	Digital signature generation using ECDSA	Curves:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits) Hashes:SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	ECDSA SigGen (FIPS186-4) ECDSA SigGen (FIPS186-4) ECDSA SigGen (FIPS186-4) ECDSA SigGen (FIPS186-4) ECDSA SigGen (FIPS186-4)
Signature Verification with ECDSA	DigSig-SigVer	Digital signature verification using ECDSA	Curves:P-224, P-256, P-384, P-521 (112, 128, 192, 256 bits) Hashes:SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	ECDSA SigVer (FIPS186-4) ECDSA SigVer (FIPS186-4) ECDSA SigVer (FIPS186-4) ECDSA SigVer (FIPS186-4) ECDSA SigVer (FIPS186-4)

Name	Type	Description	Properties	Algorithms
Hashes	SHA	Compute a message digest using Secure Hash Algorithms		SHA-1 SHA-1 SHA-1 SHA-1 SHA-1 SHA-1 SHA-1 SHA2-224 SHA2-224 SHA2-224 SHA2-224 SHA2-224 SHA2-256 SHA2-256 SHA2-256 SHA2-256 SHA2-256 SHA2-384 SHA2-384 SHA2-384 SHA2-384 SHA2-384 SHA2-512 SHA2-512 SHA2-512 SHA2-512 SHA2-512 SHA2-512/224 SHA2-512/224 SHA2-512/224 SHA2-512/224 SHA2-512/224 SHA2-512/256 SHA2-512/256 SHA2-512/256 SHA2-512/256 SHA2-512/256 SHA3-224 SHA3-224 SHA3-224 SHA3-256 SHA3-256 SHA3-256 SHA3-384 SHA3-384 SHA3-384 SHA3-512 SHA3-512 SHA3-512
Extendable Output Function	XOF	Compute message digest from XOFs		SHAKE-128 SHAKE-128 SHAKE-128 SHAKE-256 SHAKE-256 SHAKE-256
Message Authentication Code	MAC	Compute MAC tags using AES-based CMAC or HMAC	Keys:112-256 bits	AES-CMAC AES-CMAC AES-CMAC AES-CMAC HMAC-SHA-1 HMAC-SHA-1 HMAC-SHA-1 HMAC-SHA-1 HMAC-SHA-1

Name	Type	Description	Properties	Algorithms
				HMAC-SHA-1 HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-224 HMAC-SHA2-224 HMAC-SHA2-224 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-256 HMAC-SHA2-256 HMAC-SHA2-256 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-384 HMAC-SHA2-384 HMAC-SHA2-384 HMAC-SHA2-384 HMAC-SHA2-512 HMAC-SHA2-512 HMAC-SHA2-512 HMAC-SHA2-512 HMAC-SHA2-512 HMAC-SHA2-512/224 HMAC-SHA2-512/224 HMAC-SHA2-512/224 HMAC-SHA2-512/224 HMAC-SHA2-512/224 HMAC-SHA2-512/256 HMAC-SHA2-512/256 HMAC-SHA2-512/256 HMAC-SHA2-512/256 HMAC-SHA2-512/256 HMAC-SHA3-224 HMAC-SHA3-224 HMAC-SHA3-224 HMAC-SHA3-256 HMAC-SHA3-256 HMAC-SHA3-256 HMAC-SHA3-384 HMAC-SHA3-384 HMAC-SHA3-384 HMAC-SHA3-512 HMAC-SHA3-512 HMAC-SHA3-512
Random Number Generation with DRBG	DRBG	Random number generation using DRBG	Compliance:Compliant with SP800-90Arev1	Counter DRBG Counter DRBG Counter DRBG Counter DRBG Hash DRBG Hash DRBG Hash DRBG Hash DRBG Hash DRBG HMAC DRBG HMAC DRBG HMAC DRBG HMAC DRBG HMAC DRBG
Key Derivation with PBKDF	PBKDF	Key derivation using PBKDF	Derived keys:112-256 bits HMAC modes:SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-	PBKDF PBKDF PBKDF PBKDF PBKDF

Name	Type	Description	Properties	Algorithms
			512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES XTS

The length of a single data unit encrypted or decrypted with AES XTS shall not exceed 2^{20} AES blocks, that is 16MB, of data per XTS instance. An XTS instance is defined in Section 4 of SP 800-38E.

To meet the requirement stated in IG C.I, the module implements a check to ensure that the two AES keys used in AES XTS mode are not identical.

The XTS mode shall only be used for the cryptographic protection of data on storage devices. It shall not be used for other purposes, such as the encryption of data in transit.

2.7.2 Key Derivation using SP 800-132 PBKDF

The module provides password-based key derivation (PBKDF), compliant with SP 800-132. The module supports option 1a from Section 5.4 of SP 800-132, in which the Master Key (MK) or a segment of it is used directly as the Data Protection Key (DPK).

In accordance with SP 800-132 and FIPS 140-3 IG D.N, the following requirements shall be met.

- Derived keys shall only be used in storage applications. The Master Key (MK) shall not be used for other purposes. The module accepts length of the MK or DPK of 112 bits or more.
- A portion of the salt shall be generated randomly using the SP 800-90Arev1 DRBG provided by the module. The minimum length required is 128 bits.
- The iteration count is selected as large as possible, as long as the time required to generate the key using the entered password is acceptable for the users. The minimum value accepted by the module is 1000.
- Passwords or passphrases, used as an input for the PBKDF, shall not be used as cryptographic keys.
- The minimum length of the password or passphrase accepted by the module is 8 characters, and may consist of lowercase, uppercase, and numeric characters. Assuming the worst-case scenario of all digits, the probability is estimated to be at most 10^{-8}. Combined with the minimum iteration count as described above in the third bullet, this provides an acceptable trade-off between user experience and security against brute-force attacks.

The calling application shall also observe the rest of the requirements and recommendations specified in SP 800-132.

2.8 RBG and Entropy

Cert Number	Vendor Name
E99	Oracle Corporation

Table 10: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Oracle Userspace CPU Time Jitter RNG Entropy Source	Non-Physical	Oracle Linux 9 on Oracle SERVER X9-2c; Oracle Linux 9 on ORACLE SERVER E4-2c; Oracle Linux 9 on ORACLE SERVER A1-2c	256 bits	256 bits	HMAC_DBRG with SHA-512

Table 11: Entropy Sources

The module provides an SP 800-90Arev1-compliant Deterministic Random Bit Generator (DRBG) for creation of key components of asymmetric keys, and random number generation. This entropy source is located within the module's physical perimeter, but outside of the module's cryptographic boundary. The module obtains 384 bits to seed the DRBG, and 256 bits to reseed it.

The seeding (and automatic reseeding) of the DRBG is done with getrandom().

The DRBG supports the Hash_DRBG, HMAC_DRBG, and CTR_DRBG mechanisms. The DRBG is initialized during module initialization; the module loads by default the DRBG and using the HMAC_DRBG mechanism with SHA-256 and without prediction resistance. A different DRBG mechanism can be chosen by invoking the gcry_control(GCRYCTL_DRBG_REINIT) function.

The module performs the DRBG health tests as defined in Section 11.3 of SP 800-90Arev1.

2.9 Key Generation

The module provides an SP 800-90Arev1-compliant DRBG for the creation of the key components of asymmetric keys, and random number generation.

The Cryptographic Key Generation (CKG) methods implemented in the module for Approved Services in the approved mode are compliant with Section 5.1 of SP 800-133rev2.

For generating RSA and ECDSA keys, the module implements asymmetric key generation services compliant with FIPS 186-4. A seed (i.e. the random value) used in asymmetric key generation is directly obtained from the SP 800-90Arev1 DRBG.

Additionally, according to section 6.2 of SP 800-133rev2, the module implements the PBKDF2 key derivation method compliant with option 1a of SP 800-132. This implementation shall only be used to derive keys for use in storage applications.

2.10 Key Establishment

As permitted by IG D.G, the module provides key transport either by using an approved authenticated encryption mode or by a combination of any approved symmetric encryption mode and an approved authentication method. The SSP transport methods are specified in the Security Function Implementations table.

2.11 Industry Protocols

The module does not support any industry protocols listed within the publication of SP 800-135rev1. Therefore, this section is not applicable.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API input parameters for data
N/A	Data Output	API output parameters for data
N/A	Control Input	API function calls, API input parameters for control input
N/A	Status Output	API return codes, API output parameters for status output

Table 12: Ports and Interfaces

The logical interfaces are the APIs through which the applications request services. These logical interfaces are logically separated from each other by the API design. The module does not implement a control output interface.

4 Roles, Services, and Authentication

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	CO	None

Table 13: Roles

The module supports the Crypto Officer role only. This sole role is implicitly and always assumed by the operator of the module. The module does not support multiple concurrent operators.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Symmetric encryption	Perform AES encryption	gcry_control() returns 0	AES key, Plaintext	Ciphertext	Encryption with AES	Crypto Officer - AES key: W,E
Symmetric decryption	Perform AES decryption	gcry_control() returns 0	AES key, Ciphertext	Plaintext	Decryption with AES	Crypto Officer - AES key: W,E
Authenticated symmetric encryption	Authenticate and encrypt a plaintext using AES	gcry_control() returns 0	AES key, Plaintext, IV	Ciphertext, MAC tag	Authenticated encryption with AES	Crypto Officer - AES key: W,E
Authenticated symmetric decryption	Authenticate and decrypt a plaintext using AES	gcry_control() returns 0	AES key, Ciphertext, MAC tag	Plaintext	Authenticated decryption with AES	Crypto Officer - AES key: W,E
RSA Key generation	Generate RSA key pairs	gcry_control() returns 0	Key size	RSA public key, RSA private key	Key Pair Generation with RSA	Crypto Officer - RSA public key: G,R - RSA private key: G,R
ECDSA Key generation	Generate ECDSA key pairs	gcry_control() returns 0	Key size	ECDSA public key, ECDSA private key	Key Pair Generation with ECDSA	Crypto Officer - ECDSA public key: G,R - ECDSA private key: G,R
RSA Digital signature generation	RSA signature generation	gcry_control() returns 0	RSA private key, Message, Hash algorithm	Signature	Signature Generation with RSA	Crypto Officer - RSA private key: W,E
ECDSA Digital signature generation	ECDSA signature generation	gcry_control() returns 0	ECDSA private Key, Message, Hash algorithm	Signature	Signature Generation with ECDSA	Crypto Officer - ECDSA private key: W,E
RSA Digital signature verification	RSA signature verification	gcry_control() returns 0	Signature, Hash algorithm, RSA public key	Signature verification result	Signature Verification with RSA	Crypto Officer - RSA public key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
ECDSA Digital signature verification	ECDSA signature verification	gcry_control() returns 0	Signature, Hash algorithm, ECDSA public key	Signature verification result	Signature Verification with ECDSA	Crypto Officer - ECDSA public key: W,E
Public key verification	Verify ECDSA public key	gcry_mpi_ec_curve_point() returns 0	ECDSA public key, ECDSA private key	Return codes/log messages	Public Key Verification with ECDSA	Crypto Officer - ECDSA public key: W,E
Random number generation	Generate random bitstrings	gcry_randomize(), gcry_random_bytes(), gcry_random_bytes_secure() returns 0	Size	Random number	Random Number Generation with DRBG	Crypto Officer - Entropy input: W,E - DRBG seed: G,E - DRBG internal state: (V value, C value): G,W,E - DRBG internal state: (V value, key): G,W,E
Message digest	Compute SHA hashes	gcry_control() returns 0	Message	Message digest	Hashes Extendable Output Function	Crypto Officer
Message authentication code (MAC)	Compute HMAC or AES-based CMAC	gcry_control() returns 0	Message, Key	MAC tag	Message Authentication Code	Crypto Officer - HMAC key: W,E - AES key: W,E
Key wrapping	Perform AES-based key wrapping	gcry_control() returns 0	Key wrapping key, key to be wrapped	Wrapped key	Key wrapping using AES CCM Key wrapping using AES KW	Crypto Officer - AES key: W,E
Key unwrapping	Perform AES-based key unwrapping	gcry_control() returns 0	Wrapped key, key unwrapping key	Unwrapped key	Key unwrapping using AES CCM Key unwrapping using AES KW	Crypto Officer - AES key: W,E
Key derivation	Perform key derivation	gcry_control() returns 0	Password/passphrase; Derived key	Derived key	Key Derivation with PBKDF	Crypto Officer - Password or passphrase: W,E - Derived key: G,R
Show status	Show module status	N/A	None	Current status of the module	None	Crypto Officer
Zeroization	Zeroize SSPs	N/A	N/A	N/A	None	Crypto Officer - AES key: Z - HMAC

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						key: Z - Password or passphrase: Z - Derived key: Z - Entropy input: Z - DRBG internal state: (V value, key): Z - DRBG internal state: (V value, C value): Z - DRBG seed: Z - ECDSA public key: Z - ECDSA private key: Z - RSA public key: Z - RSA private key: Z
Self-tests	Perform self-tests	N/A	Booting the module	N/A	None	Crypto Officer
Show module name and version	Show module name and version	N/A	N/A	Display module name and version	None	Crypto Officer

Table 14: Approved Services

For all approved services, GPG_ERR_NO_ERROR (i.e., “0”) return code indicates the service is approved. In case the above-mentioned controls are used in conjunction, the operator is responsible to check that all the called functions return GPG_ERR_NO_ERROR (i.e., “0”). For all non-approved services, a "non-zero" return code indicates the service is not approved.

The table above lists the approved services. For each service, the table lists the associated cryptographic algorithm(s), the role to perform the service, the cryptographic keys or CSPs involved, and their access type(s). The following convention is used to specify access rights to SSPs:

- **Generate (G)**: The module generates or derives the SSP.
- **Read (R)**: The SSP is read from the module (e.g. the SSP is output).
- **Write (W)**: The SSP is updated, imported, or written to the module.
- **Execute (E)**: The module uses the SSP in performing a cryptographic operation.
- **Zeroize (Z)**: The module zeroizes the SSP.
- **N/A**: the calling application does not access any CSP or key during its operation.

The details of the approved cryptographic algorithms including the CAVP certificate numbers can be found in Section 2.5. In order to check whether it utilizes an approved security function or not, the operator is responsible to invoke the gcry_control() API along with dedicated controls in the form of API input parameters.

The module implements the following controls depending on the requested service:

1. GCRYCTL_FIPS_SERVICE_INDICATOR_CIPHER - For symmetric algorithms and the related modes.
2. GCRYCTL_FIPS_SERVICE_INDICATOR_KDF - For KDF operations.
3. GCRYCTL_FIPS_SERVICE_INDICATOR_PK_FLAGS - For asymmetric operations.[1]
4. GCRYCTL_FIPS_SERVICE_INDICATOR_MD - For digest operations.
5. GCRYCTL_FIPS_SERVICE_INDICATOR_MAC - For MAC operations.

In addition to that, for the below-mentioned services, the approved service indicator corresponds to the GPG_ERR_NO_ERROR returned from listed functions in the indicator column below. They don't use gcry_control() API:

1. *Random number generation* service: gcry_randomize(), gcry_random_bytes(), gcry_random_bytes_secure().
2. *Public key validation* service: gcry_mpi_ec_curve_point().

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Authenticated symmetric encryption	AES encryption using non-approved AES modes	AES GCM noncompliant with IG C.H. AES GCM-SIV AES OCB AES-EAX	CO
Authenticated symmetric decryption	AES decryption using non-approved AES modes	AES GCM noncompliant with IG C.H. AES GCM-SIV AES OCB AES-EAX	CO
Message digest using non-approved algorithms	Message digest	MD5	CO
Shared secret computation	ECDH Shared secret computation	ECDH noncompliant with SP 800-56Arev3 assurances	CO
Key generation	Generate RSA/ECDSA key pairs with public key flags not listed in Appendix A	RSA with non-approved flags that are not listed in Appendix A ECDSA with non-approved flags that are not listed in Appendix A	CO
Digital signature generation	RSA/ECDSA signature generation with public key flags not listed in Appendix A	RSA with non-approved flags that are not listed in Appendix A ECDSA with non-approved flags that are not listed in Appendix A	CO
Digital signature verification	RSA/ECDSA signature verification with public key flags not listed in Appendix A	RSA with non-approved flags that are not listed in Appendix A ECDSA with non-approved flags that are not listed in Appendix A	CO
Asymmetric encryption and decryption primitives	RSA encryption and decryption primitives	RSA	CO
Signature generation/verification primitives	RSA/ECDSA signature generation/verification primitives	RSA ECDSA	CO

Table 15: Non-Approved Services

The table above lists the non-approved services. The details of the non-approved cryptographic algorithms available in non-approved mode can be found in Section 2.5. For the services listed above, the module implements an additional service indicator in the form of a control named GCRYCTL_FIPS_SERVICE_INDICATOR_FUNCTION. The

[1] The list of public key flags allowed in approved mode of operation is described in Section 4.4.

operator is responsible to invoke the gcry_control() API along with the following input parameters: GCRYCTL_FIPS_SERVICE_INDICATOR_FUNCTION control; the name of the API[2] representing the service.

[2] The list of APIs is supported by the module can be found in the documentation included in the optional *libgcrypt-devel* package.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by comparing a HMAC SHA-256 values calculated at run time with the HMAC SHA-256 value embedded within the module binary. If the HMAC values do not match, the test fails, and the module enters the Error state.

5.2 Initiate on Demand

The integrity test is performed as part of the pre-operational self-test, which is executed when the module is initialized.

In addition, the module provides the Self-Test service to perform self-tests on demand which includes the pre-operational test (i.e., integrity test) and cryptographic algorithm self-tests (CASTs). This service can be invoked relying on the gcry_control(GCRYCTL_SELFTEST) API function call or by powering-off and reloading the module. During the execution of the on-demand self-tests, services are not available, and data output or input is not possible.

In order to verify whether the self-tests have succeeded and the module is in the Operational state, the calling application may invoke the gcry_control(GCRYTCL_OPERATIONAL_P) API. The function will return TRUE if the module is in the Operational state and FALSE if the module is in the Error state.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The module shall be installed as stated in Section 11. 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 shall be installed as stated in Section 11.1.

Instrumentation tools like the ptrace system call, gdb and strace, as well as other tracing mechanisms offered by the Linux environment such as ftrace or systemtap, shall not be used in the operational environment. The use of any of these tools implies that the cryptographic module is running in a non-validated operational environment.

7 Physical Security

The module is comprised of software only, and therefore this section is not applicable.

8 Non-Invasive Security

This module does not implement any non-invasive security mechanism, and therefore this section is not applicable.

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 16: Storage Areas

The module does not perform persistent storage of SSPs. The SSPs are temporarily stored in the RAM in plaintext form. SSPs are provided to the module by the calling process and are destroyed when released by the appropriate zeroization function calls.

9.2 SSP Input-Output Methods

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

Table 17: SSP Input-Output Methods

The module does not support manual SSP input or intermediate SSP generation output. The SSPs are provided to the module via API input parameters in plaintext form and output via API output parameters in plaintext form within the physical perimeter of the operational environment. This is allowed by FIPS 140-3 IG 9.5.A, according to the "CM Software to/from App via TOEPP Path" entry in the table above.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Free cipher handle	Zeroizes the SSPs contained within the provided cipher handle	Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable. The completion of a zeroization routine will indicate that a zeroization procedure succeeded.	By calling the appropriate zeroization functions: AES key: gcry_cipher_close gcry_free() HMAC key: gcry_mac_close, gcry_free Key-derivation key: gcry_free Derived key: gcry_free RSA keys: gcry_mpi_release, gcry_sexp_release, gcry_free EC keys: gcry_mpi_release, gcry_free, gcry_mpi_point_release, gcry_sexp_release, gcry_ctx_release Entropy input: gcry_ctrl(GCRYCTL_TERM_SECMEM) Internal state: gcry_ctrl(GCRYCTL_TERM_SECMEM)
Remove power from the module	De-allocates the volatile memory used to store SSPs	Volatile memory used by the module is overwritten within nanoseconds when power is removed. Module power off indicates that the zeroization procedure succeeded.	By unloading the module

Table 18: SSP Zeroization Methods

All data output is inhibited during zeroization.

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES key	AES key used for encryption, decryption, and computing MAC tags	XTS: 256, 512 bits; Other modes: 128, 192, 256 bits - XTS: 128, 256 bits; Other modes: 128, 192, 256 bits	Symmetric key - CSP			Key wrapping using AES CCM Key wrapping using AES KW Key unwrapping using AES CCM Key unwrapping using AES KW Encryption with AES Decryption with AES
HMAC key	HMAC key	112-256 bits - 112-256 bits	Authentication key - CSP			Message Authentication Code
Password or passphrase	PBKDF2 password or passphrase	At least 8 characters - N/A	Password or passphrase - CSP			Key Derivation with PBKDF
Derived key	PBKDF2 derived key	112-256 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with PBKDF		Key Derivation with PBKDF
Entropy input	Obtained from the entropy source, used to seed the DRBGs	128-448 bits (128-256 bits) - 256 bits	Entropy input - CSP			Random Number Generation with DRBG
DRBG internal state: (V value, key)	Internal state of CTR_DRBG and HMAC_DRBG	CTR_DRBG: 256, 320, 384 bits; HMAC_DRBG: 320, 512, 1024 bits - CTR_DRBG: 128, 192, 256 bits; HMAC_DRBG: 128, 256 bits	Internal state - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
DRBG internal state: (V value, C value)	Internal state of Hash_DRBG	880, 1776 bits - 128, 256 bits	Internal state - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
DRBG seed	DRBG seed derived from entropy input	CTR_DRBG: 256, 320, 384 bits; Hash_DRBG: 440, 888 bits; HMAC_DRBG: 440, 888 bits - CTR_DRBG: 128, 192, 256 bits; Hash_DRBG: 128, 256 bits; HMAC_DRBG: 128, 256 bits	Seed - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
ECDSA public key	Public key used for ECDSA signature verification	P-224, P-256, P-384, P-521 - 112, 128, 192, 256 bits	Public key - PSP	Key Pair Generation with ECDSA		Key Pair Generation with ECDSA Public Key Verification with ECDSA Signature Verification with ECDSA
ECDSA private key	Private key used for ECDSA signature generation	P-224, P-256, P-384, P-521 - 112, 128, 192, 256 bits	Private key - CSP	Key Pair Generation with ECDSA		Key Pair Generation with ECDSA Public Key Verification with ECDSA Signature

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						Generation with ECDSA
RSA public key	Public key used for RSA signature verification	1024, 1536, 2048, 3072, 4096 bits - 80, 96, 112, 128, 140 bits	Public key - PSP	Key Pair Generation with RSA		Key Pair Generation with RSA Signature Verification with RSA
RSA private key	Private key used for RSA signature generation	2048, 3072, 4096 bits - 112, 128, 140 bits	Private key - CSP	Key Pair Generation with RSA		Key Pair Generation with RSA Signature Generation with RSA

Table 19: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES key	API input parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	
HMAC key	API input parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	
Password or passphrase	API output parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	Derived key:Generates
Derived key	API input parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	Password or passphrase:Derived From
Entropy input		RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	DRBG internal state: (V value, key):Generates DRBG internal state: (V value, C value):Generates DRBG seed:Generates
DRBG internal state: (V value, key)		RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	Entropy input:Derived From DRBG seed:Derived From
DRBG internal state: (V value, C value)		RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	Entropy input:Derived From DRBG seed:Derived From
DRBG seed		RAM:Plaintext		Free cipher handle Remove power from the module	Entropy input:Derived From DRBG internal state: (V value, key):Generates DRBG internal state: (V value, C value):Generates
ECDSA public key	API input parameters API output parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	DRBG internal state: (V value, key):Derived From ECDSA private key:Paired With
ECDSA private key	API input parameters API output parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	DRBG internal state: (V value, key):Derived From ECDSA public key:Paired With
RSA public key	API input parameters API output parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	DRBG internal state: (V value, key):Derived From RSA private key:Paired With
RSA private key	API input parameters	RAM:Plaintext	From service invoked to service completed	Free cipher handle Remove power from the module	RSA public key:Paired With DRBG internal state: (V value, key):Derived From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
	API output parameters				

Table 20: 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.

The RSA algorithm as implemented by the module conforms to FIPS 186-4, which has been superseded by FIPS 186-5. FIPS 186-4 has been withdrawn since February 3, 2024.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A4773)	Key size: 376-bit key	Message Authentication	SW/FW Integrity	The module becomes operational and the services are available for use	Integrity test for libgcrypt.so.20.4.0

Table 21: Pre-Operational Self-Tests

The pre-operational software integrity test is performed automatically when the module is powered on before the module transitions into the Operational state. While the module is executing the self-tests, services are not available, and data output (via the data output interface) is inhibited until the tests are successfully completed. The module transitions to the Operational state only after the pre-operational self-test passes successfully.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A4773)	AES ECB mode with 128, 192, 256-bit keys, encryption, and decryption (separately tested)	KAT	CAST	The module becomes operational and the services are available for use	Encryption, Decryption	Module initialization or on demand through API function call
AES-ECB (A4774)	AES ECB mode with 128, 192, 256-bit keys, encryption, and decryption (separately tested)	KAT	CAST	The module becomes operational and the services are available for use	Encryption, Decryption	Module initialization or on demand through API function call
AES-ECB (A4776)	AES ECB mode with 128, 192, 256-bit keys, encryption, and decryption (separately tested)	KAT	CAST	The module becomes operational and the services are available for use	Encryption, Decryption	Module initialization or on demand through API function call
AES-ECB (A4777)	AES ECB mode with 128, 192, 256-bit keys, encryption, and decryption (separately tested)	KAT	CAST	The module becomes operational and the services are available for use	Encryption, Decryption	Module initialization or on demand through API function call
AES-CMAC (A4773)	AES CMAC with 128-bit key, MAC generation	KAT	CAST	The module becomes operational and the services are available for use	Message Authentication	Module initialization or on demand through API function call
AES-CMAC (A4774)	AES CMAC with 128-bit key, MAC generation	KAT	CAST	The module becomes operational and the services are available for use	Message Authentication	Module initialization or on demand through API function call
AES-CMAC (A4776)	AES CMAC with 128-bit key, MAC generation	KAT	CAST	The module becomes operational and the services are available for use	Message Authentication	Module initialization or on demand through API function call
AES-CMAC (A4777)	AES CMAC with 128-bit key, MAC generation	KAT	CAST	The module becomes operational and the services are available for use	Message Authentication	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
Counter DRBG (A4773)	CTR_DRBG with AES with 128-bit key with DF, with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Counter DRBG (A4774)	CTR_DRBG with AES with 128-bit key with DF, with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Counter DRBG (A4776)	CTR_DRBG with AES with 128-bit key with DF, with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Counter DRBG (A4777)	CTR_DRBG with AES with 128-bit key with DF, with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4773)	SHA-1 without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4774)	SHA-1 without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4776)	SHA-1 without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4777)	SHA-1 without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4778)	SHA-1 without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4773)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4774)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4776)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
Hash DRBG (A4777)	SHA-256 with and without PR	KAT	CAST	The module becomes	Generate, Reseed	Module initialization or on

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				operational and the services are available for use		demand through API function call
Hash DRBG (A4778)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
HMAC DRBG (A4773)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
HMAC DRBG (A4774)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
HMAC DRBG (A4776)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
HMAC DRBG (A4777)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
HMAC DRBG (A4778)	SHA-256 with and without PR	KAT	CAST	The module becomes operational and the services are available for use	Generate, Reseed	Module initialization or on demand through API function call
ECDSA SigGen (FIPS186-4) (A4773)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
ECDSA SigGen (FIPS186-4) (A4774)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
ECDSA SigGen (FIPS186-4) (A4776)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
ECDSA SigGen (FIPS186-4) (A4777)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
ECDSA SigGen (FIPS186-4) (A4778)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
ECDSA SigVer (FIPS186-4) (A4773)	P-256 with SHA-256	KAT	CAST	The module becomes operational and	Signature verification	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				the services are available for use		
ECDSA SigVer (FIPS186-4) (A4774s)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
ECDSA SigVer (FIPS186-4) (A4776)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
ECDSA SigVer (FIPS186-4) (A4777)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
ECDSA SigVer (FIPS186-4) (A4778)	P-256 with SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
HMAC-SHA-1 (A4772)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA-1 (A4773)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA-1 (A4774)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA-1 (A4776)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA-1 (A4777)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA-1 (A4778)	HMAC-SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-224 (A4773)	HMAC-SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-224 (A4774)	HMAC-SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC-SHA2-224 (A4776)	HMAC-SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-224 (A4777)	HMAC-SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-224 (A4778)	HMAC-SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-256 (A4773)	HMAC-SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-256 (A4774)	HMAC-SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-256 (A4776)	HMAC-SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-256 (A4777)	HMAC-SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-256 (A4778)	HMAC-SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-384 (A4773)	HMAC-SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-384 (A4774)	HMAC-SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-384 (A4776)	HMAC-SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-384 (A4777)	HMAC-SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-384 (A4778)	HMAC-SHA-384	KAT	CAST	The module becomes	Message authentication	Module initialization or on

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				operational and the services are available for use		demand through API function call
HMAC-SHA2-512 (A4773)	HMAC-SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-512 (A4774)	HMAC-SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-512 (A4776)	HMAC-SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-512 (A4777)	HMAC-SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA2-512 (A4778)	HMAC-SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-224 (A4773)	HMAC-SHA3-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-224 (A4774)	HMAC-SHA3-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-224 (A4773)	HMAC-SHA3-224	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-256 (A4773)	HMAC-SHA3-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-256 (A4774)	HMAC-SHA3-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-256 (A4778)	HMAC-SHA3-256	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-384 (A4773)	HMAC-SHA3-384	KAT	CAST	The module becomes operational and	Message authentication	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				the services are available for use		
HMAC-SHA3-384 (A4774)	HMAC-SHA3-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-384 (A4778)	HMAC-SHA3-384	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-512 (A4773)	HMAC-SHA3-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-512 (A4774)	HMAC-SHA3-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
HMAC-SHA3-512 (A4778)	HMAC-SHA3-512	KAT	CAST	The module becomes operational and the services are available for use	Message authentication	Module initialization or on demand through API function call
RSA SigGen (FIPS186-4) (A4773)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
RSA SigGen (FIPS186-4) (A4774)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
RSA SigGen (FIPS186-4) (A4776)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
RSA SigGen (FIPS186-4) (A4777)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
RSA SigGen (FIPS186-4) (A4778)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature generation	Module initialization or on demand through API function call
RSA SigVer (FIPS186-4) (A4773)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
RSA SigVer (FIPS186-4) (A4774)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA SigVer (FIPS186-4) (A4776)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
RSA SigVer (FIPS186-4) (A4777)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
RSA SigVer (FIPS186-4) (A4778)	PKCS #1 v1.5 with 2048-bit key and SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Signature verification	Module initialization or on demand through API function call
SHA-1 (A4772)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4773)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4774)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4775)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4776)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4777)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA-1 (A4778)	SHA-1	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-224 (A4773)	SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-224 (A4774)	SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-224 (A4776)	SHA-224	KAT	CAST	The module becomes	Message digest	Module initialization or on

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				operational and the services are available for use		demand through API function call
SHA2-224 (A4777)	SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-224 (A4778)	SHA-224	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-256 (A4773)	SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-256 (A4774)	SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-256 (A4776)	SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-256 (A4777)	SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-256 (A4778)	SHA-256	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-384 (A4773)	SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-384 (A4774)	SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-384 (A4776)	SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-384 (A4777)	SHA-384	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-384 (A4778)	SHA-384	KAT	CAST	The module becomes operational and	Message digest	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				the services are available for use		
SHA2-512 (A4773)	SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-512 (A4774)	SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-512 (A4776)	SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
SHA2-512 (A4778)	SHA-512	KAT	CAST	The module becomes operational and the services are available for use	Message digest	Module initialization or on demand through API function call
PBKDF (A4773)	SHA-1 password length 24 characters, master key length of 200 bits, iteration count of 4096, and salt length of 288 bits; SHA-256 password length 24 characters, master key length of 320 bits, iteration count of 4096, and salt length of 288 bits	KAT	CAST	The module becomes operational and the services are available for use	Key derivation	Module initialization or on demand through API function call
PBKDF (A4774)	SHA-1 password length 24 characters, master key length of 200 bits, iteration count of 4096, and salt length of 288 bits; SHA-256 password length 24 characters, master key length of 320 bits, iteration count of 4096, and salt length of 288 bits	KAT	CAST	The module becomes operational and the services are available for use	Key derivation	Module initialization or on demand through API function call
PBKDF (A4776)	SHA-1 password length 24 characters, master key length of 200 bits, iteration count of 4096, and salt length of 288 bits; SHA-256 password length 24 characters, master key length of 320 bits, iteration count of 4096, and salt length of 288 bits	KAT	CAST	The module becomes operational and the services are available for use	Key derivation	Module initialization or on demand through API function call
PBKDF (A4777)	SHA-1 password length 24 characters, master key length of 200 bits, iteration count of 4096, and salt length of 288 bits; SHA-256 password length 24 characters, master key length of 320 bits, iteration count of 4096, and salt length of 288 bits	KAT	CAST	The module becomes operational and the services are available for use	Key derivation	Module initialization or on demand through API function call
PBKDF (A4778)	SHA-1 password length 24 characters, master key length of 200 bits, iteration count of 4096, and salt length of 288 bits; SHA-256 password length 24 characters, master key length of 320 bits, iteration count of 4096, and salt length of 288 bits	KAT	CAST	The module becomes operational and the services are available for use	Key derivation	Module initialization or on demand through API function call

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA KeyGen (FIPS186-4) (A4773)	Signature generation and verification with SHA-256	PCT	PCT	Successful key generation	Key generation	EC key pair generation
ECDSA KeyGen (FIPS186-4) (A4774)	Signature generation and verification with SHA-256	PCT	PCT	Successful key generation	Key generation	EC key pair generation
ECDSA KeyGen (FIPS186-4) (A4776)	Signature generation and verification with SHA-256	PCT	PCT	Successful key generation	Key generation	EC key pair generation
ECDSA KeyGen (FIPS186-4) (A4777)	Signature generation and verification with SHA-256	PCT	PCT	Successful key generation	Key generation	EC key pair generation
ECDSA KeyGen (FIPS186-4) (A4778)	Signature generation and verification with SHA-256	PCT	PCT	Successful key generation	Key generation	EC key pair generation
RSA KeyGen (FIPS186-4) (A4773)	Signature generation of verification with SHA-256	PCT	PCT	Successful key generation	Key generation	RSA key pair generation
RSA KeyGen (FIPS186-4) (A4774)	Signature generation of verification with SHA-256	PCT	PCT	Successful key generation	Key generation	RSA key pair generation
RSA KeyGen (FIPS186-4) (A4776)	Signature generation of verification with SHA-256	PCT	PCT	Successful key generation	Key generation	RSA key pair generation
RSA KeyGen (FIPS186-4) (A4777)	Signature generation of verification with SHA-256	PCT	PCT	Successful key generation	Key generation	RSA key pair generation
RSA KeyGen (FIPS186-4) (A4778)	Signature generation of verification with SHA-256	PCT	PCT	Successful key generation	Key generation	RSA key pair generation

Table 22: Conditional Self-Tests

The module performs self-tests on all approved cryptographic algorithms as part of the approved services supported in the approved mode of operation, using the tests shown in table above. Services are not available, and data output (via the data output interface) is inhibited during the self-tests. If any of these tests fails, the module transitions to the Error state.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A4773)	Message Authentication	SW/FW Integrity	Whenever module is powered on	Upon every power on

Table 23: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A4773)	KAT	CAST	On demand	Manually
AES-ECB (A4774)	KAT	CAST	On demand	Manually
AES-ECB (A4776)	KAT	CAST	On demand	Manually
AES-ECB (A4777)	KAT	CAST	On demand	Manually
AES-CMAC (A4773)	KAT	CAST	On demand	Manually
AES-CMAC (A4774)	KAT	CAST	On demand	Manually
AES-CMAC (A4776)	KAT	CAST	On demand	Manually
AES-CMAC (A4777)	KAT	CAST	On demand	Manually
Counter DRBG (A4773)	KAT	CAST	On demand	Manually
Counter DRBG (A4774)	KAT	CAST	On demand	Manually
Counter DRBG (A4776)	KAT	CAST	On demand	Manually
Counter DRBG (A4777)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
Hash DRBG (A4773)	KAT	CAST	On demand	Manually
Hash DRBG (A4774)	KAT	CAST	On demand	Manually
Hash DRBG (A4776)	KAT	CAST	On demand	Manually
Hash DRBG (A4777)	KAT	CAST	On demand	Manually
Hash DRBG (A4778)	KAT	CAST	On demand	Manually
Hash DRBG (A4773)	KAT	CAST	On demand	Manually
Hash DRBG (A4774)	KAT	CAST	On demand	Manually
Hash DRBG (A4776)	KAT	CAST	On demand	Manually
Hash DRBG (A4777)	KAT	CAST	On demand	Manually
Hash DRBG (A4778)	KAT	CAST	On demand	Manually
HMAC DRBG (A4773)	KAT	CAST	On demand	Manually
HMAC DRBG (A4774)	KAT	CAST	On demand	Manually
HMAC DRBG (A4776)	KAT	CAST	On demand	Manually
HMAC DRBG (A4777)	KAT	CAST	On demand	Manually
HMAC DRBG (A4778)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A4773)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A4774)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A4776)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A4777)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-4) (A4778)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A4773)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A4774s)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A4776)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A4777)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-4) (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4772)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4776)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4777)	KAT	CAST	On demand	Manually
HMAC-SHA-1 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA2-224 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA2-224 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA2-224 (A4776)	KAT	CAST	On demand	Manually
HMAC-SHA2-224 (A4777)	KAT	CAST	On demand	Manually
HMAC-SHA2-224 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A4776)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A4777)	KAT	CAST	On demand	Manually
HMAC-SHA2-256 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A4776)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A4777)	KAT	CAST	On demand	Manually
HMAC-SHA2-384 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A4776)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A4777)	KAT	CAST	On demand	Manually
HMAC-SHA2-512 (A4778)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA3-224 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA3-224 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA3-224 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA3-256 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA3-256 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA3-256 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA3-384 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA3-384 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA3-384 (A4778)	KAT	CAST	On demand	Manually
HMAC-SHA3-512 (A4773)	KAT	CAST	On demand	Manually
HMAC-SHA3-512 (A4774)	KAT	CAST	On demand	Manually
HMAC-SHA3-512 (A4778)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A4773)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A4774)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A4776)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A4777)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-4) (A4778)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4773)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4774)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4776)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4777)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4778)	KAT	CAST	On demand	Manually
SHA-1 (A4772)	KAT	CAST	On demand	Manually
SHA-1 (A4773)	KAT	CAST	On demand	Manually
SHA-1 (A4774)	KAT	CAST	On demand	Manually
SHA-1 (A4775)	KAT	CAST	On demand	Manually
SHA-1 (A4776)	KAT	CAST	On demand	Manually
SHA-1 (A4777)	KAT	CAST	On demand	Manually
SHA-1 (A4778)	KAT	CAST	On demand	Manually
SHA2-224 (A4773)	KAT	CAST	On demand	Manually
SHA2-224 (A4774)	KAT	CAST	On demand	Manually
SHA2-224 (A4776)	KAT	CAST	On demand	Manually
SHA2-224 (A4777)	KAT	CAST	On demand	Manually
SHA2-224 (A4778)	KAT	CAST	On demand	Manually
SHA2-256 (A4773)	KAT	CAST	On demand	Manually
SHA2-256 (A4774)	KAT	CAST	On demand	Manually
SHA2-256 (A4776)	KAT	CAST	On demand	Manually
SHA2-256 (A4777)	KAT	CAST	On demand	Manually
SHA2-256 (A4778)	KAT	CAST	On demand	Manually
SHA2-384 (A4773)	KAT	CAST	On demand	Manually
SHA2-384 (A4774)	KAT	CAST	On demand	Manually
SHA2-384 (A4776)	KAT	CAST	On demand	Manually
SHA2-384 (A4777)	KAT	CAST	On demand	Manually
SHA2-384 (A4778)	KAT	CAST	On demand	Manually
SHA2-512 (A4773)	KAT	CAST	On demand	Manually
SHA2-512 (A4774)	KAT	CAST	On demand	Manually
SHA2-512 (A4776)	KAT	CAST	On demand	Manually
SHA2-512 (A4778)	KAT	CAST	On demand	Manually
PBKDF (A4773)	KAT	CAST	On demand	Manually
PBKDF (A4774)	KAT	CAST	On demand	Manually
PBKDF (A4776)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
PBKDF (A4777)	KAT	CAST	On demand	Manually
PBKDF (A4778)	KAT	CAST	On demand	Manually
ECDSA KeyGen (FIPS186-4) (A4773)	PCT	PCT	Upon generation of an ECDSA key pair	Upon generation of an ECDSA key pair
ECDSA KeyGen (FIPS186-4) (A4774)	PCT	PCT	Upon generation of an ECDSA key pair	Upon generation of an ECDSA key pair
ECDSA KeyGen (FIPS186-4) (A4776)	PCT	PCT	Upon generation of an ECDSA key pair	Upon generation of an ECDSA key pair
ECDSA KeyGen (FIPS186-4) (A4777)	PCT	PCT	Upon generation of an ECDSA key pair	Upon generation of an ECDSA key pair
ECDSA KeyGen (FIPS186-4) (A4778)	PCT	PCT	Upon generation of an ECDSA key pair	Upon generation of an ECDSA key pair
RSA KeyGen (FIPS186-4) (A4773)	PCT	PCT	Upon generation of an RSA key pair	Upon generation of an RSA key pair
RSA KeyGen (FIPS186-4) (A4774)	PCT	PCT	Upon generation of an RSA key pair	Upon generation of an RSA key pair
RSA KeyGen (FIPS186-4) (A4776)	PCT	PCT	Upon generation of an RSA key pair	Upon generation of an RSA key pair
RSA KeyGen (FIPS186-4) (A4777)	PCT	PCT	Upon generation of an RSA key pair	Upon generation of an RSA key pair
RSA KeyGen (FIPS186-4) (A4778)	PCT	PCT	Upon generation of an RSA key pair	Upon generation of an RSA key pair

Table 24: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error State	The module immediately stops functioning due to a self-test failure; PCT failure	Software integrity test failure CAST failure PCT failure	Restarting the module	Module will not load; Module stops functioning for PCT failure
Fatal Error State	The module immediately enters a non-recoverable error state and automatically transits to shutdown	Random numbers are requested in the error state or cipher operations are requested on a deallocated handle	Restarting the module	Module is aborted and is not available for use

Table 25: Error States

The table above shows the error codes and the corresponding condition. When the module fails any pre-operational self-test or conditional test, the module returns an error code to indicate the error and will enter the Error state. Any further cryptographic operation is inhibited. The calling application can obtain the module state by calling the gcry_control(GCRYCTL_OPERATIONAL_P) API function. The function returns FALSE if the module is in the Error state and TRUE if the module is in the Operational state. In the Error state, all data output is inhibited, no cryptographic operation is allowed, and the module accepts no more inputs or requests (as the module is no longer running). Recovery from the Error state includes restarting (i.e., powering off and powering on) of the module or running self-tests. Recovery from the Fatal Error state can only be done by restarting the module.

10.5 Operator Initiation of Self-Tests

The software integrity tests, cryptographic algorithm self-tests, and entropy source start-up tests can be invoked on demand by unloading and subsequently re-initializing the module. The pair-wise consistency tests can be invoked on demand by requesting the key pair generation service.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The module is distributed as part of the Oracle Linux 9 (OL9) RPM package in the form of libgcrypt-1.10.0-10.0.1.el9_2_fips RPM package that is located in the "Oracle Linux 9 Security Validation (Update 3)" yum repository (ol9_u3_security_validation).

The operational environment needs to be set up in the FIPS validated configuration by installing the module as follows:

- For installation, add the fips=1 option to the kernel command line during the system installation. During the software selection stage, do not install any third-party software.
- Switching the system into the FIPS validated configuration after the installation, execute the fips-mode-setup --enable command. Restart the system using the reboot command.
- For verifying that the FIPS validated configuration is enabled, execute the fips-mode-setup --check command as specified in Section 11.2.

11.2 Administrator Guidance

The binaries of the module are contained in the RPM packages for delivery. The Crypto Officer shall follow Section 11.1 to configure the operational environment and install the module to be operated as a FIPS 140-3 validated module.

The following RPM packages contain the FIPS validated module:

- libgcrypt-1.10.0-10.0.1.el9_2_fips.x86_64.rpm for Intel and AMD 64-bit processors
- libgcrypt-1.10.0-10.0.1.el9_2_fips.aarch64.rpm for Ampere 64-bit processor

The Crypto Officer must verify the Oracle Linux 9 system operates in the FIPS validated configuration by executing the fips-mode-setup --check command, which should output "FIPS mode enabled".

After installation of the libgcrypt-1.10.0-10.0.1.el9_2_fips RPM packages, the Crypto Officer must check the output of the gcry_get_config() API. From the output, the Crypto Officer must ensure that the proper name is listed in the output as follows:

Oracle Linux 9 libgcrypt Cryptographic Module 1.10.0-3957adb8de08b15a

Once libgcrypt has been put into the FIPS validated configuration, it is not possible to switch back to the non-validated configuration without terminating the process first. If the logging verbosity level of libgcrypt has been set to at least 2, the state transitions and self-tests are logged.

11.3 Non-Administrator Guidance

There is no non-administrator guidance.

11.4 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. Then, if desired, the libgcrypt-1.10.0-10.0.1.el9_2_fips RPM packages can be uninstalled from the Oracle Linux 9 system.

12 Mitigation of Other Attacks

12.1 Attack List

The module implements blinding against RSA Timing Attacks.

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

12.2 Mitigation Effectiveness

By default, the module uses the following blinding technique: instead of using the RSA decryption directly, a blinded value $y = x\ r^e \bmod n$ is decrypted and the unblinded value $x' = y'\ r^{-1} \bmod n$ returned.

The blinding value r is a random value with the size of the modulus n.

Appendix A. Approved Public Key Flags

Listed below are the approved public key flags for an input s-expression:

curve	d	data	e	ecdsa	flags	sig-val
genkey	hash	n	nbits	pkcs1	private-key	value
pss	public-key	q	r	raw	rsa	salt-length
rsa-use-e	s					

Appendix B. Glossary and Abbreviations

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
API	Application Programming Interface
CAST	Cryptographic Algorithm Self-Test
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
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
HMAC	Keyed-Hash Message Authentication Code
KAT	Known Answer Test
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
PAA	Processor Algorithm Acceleration
PBKDF2	Password-based Key Derivation Function v2
PKCS	Public-Key Cryptography Standards
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SSP	Sensitive Security Parameter
XTS	XEX-based Tweaked-codebook mode with cipher text Stealing

Appendix C. References

FIPS 140-3	FIPS PUB 140-3 - Security Requirements For Cryptographic Modules March 2019 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf
FIPS 140-3 IG	Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements
FIPS 180-4	Secure Hash Standard (SHS) March 2012 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
FIPS 186-4	Digital Signature Standard (DSS) July 2013 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
FIPS 186-5	Digital Signature Standard (DSS) February 2023 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf
FIPS 197	Advanced Encryption Standard November 2001 https://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
FIPS 198-1	The Keyed Hash Message Authentication Code (HMAC) July 2008 https://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf
FIPS 202	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions August 2015 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf
SP 800-38A	Recommendation for Block Cipher Modes of Operation Methods and Techniques December 2001 https://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf
SP 800-38B	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication May 2005 https://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf
SP 800-38F	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping December 2012 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf
SP 800-56Ar3	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography April 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf
SP 800-90Ar1	Recommendation for Random Number Generation Using Deterministic Random Bit Generators June 2015 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf
SP 800-90B	Recommendation for the Entropy Sources Used for Random Bit Generation January 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf
SP 800-108r1	NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions

	August 2022 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-108r1.pdf
SP 800-132	Recommendation for Password-Based Key Derivation - Part 1: Storage Applications December 2010 https://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf
SP 800-133r2	Recommendation for Cryptographic Key Generation June 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf
SP 800-135r1	Recommendation for Existing Application-Specific Key Derivation Functions December 2011 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf