



FIPS 140-3 Non-Proprietary Security Policy

Oracle Linux 9 NSS Cryptographic Module

FIPS 140-3 Level 1 Validation

Software Version: 4.35.0-381552536e763d0c

Prepared by:

atsec information security corporation

4516 Seton Center Pkwy, Suite 250

Austin, TX 78759

www.atsec.com



Title: Oracle Linux 9 NSS Cryptographic Module

Date: September 6th, 2024

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



Copyright © 2024, 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 be 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



Table of Contents

| | | |
|--------------|--|-----------|
| 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 Version and Identification | 3 |
| 2.3 | Excluded Components | 3 |
| 2.4 | Modes of Operation | 3 |
| 2.5 | Algorithms | 4 |
| 2.6 | Security Function Implementations | 10 |
| 2.7 | Algorithm Specific Information | 11 |
| 2.7.1 | AES GCM IV | 11 |
| 2.7.2 | Key Derivation using SP 800-132 PBKDF2 | 11 |
| 2.7.3 | SP 800-56Ar3 Assurances | 12 |
| 2.8 | RNG and Entropy | 12 |
| 2.9 | Key Generation | 12 |
| 2.10 | Key Establishment | 14 |
| 2.11 | Industry Protocols | 14 |
| 3 | Cryptographic Module Interfaces | 15 |
| 3.1 | Description | 15 |
| 3.2 | Trusted Channel Specification | 15 |
| 3.3 | Control Interface Not Inhibited | 15 |
| 4 | Roles, Services, and Authentication | 16 |
| 4.1 | Authentication Methods | 16 |
| 4.2 | Roles | 16 |
| 4.3 | Approved Services | 16 |
| 4.4 | Non-Approved Services | 19 |
| 4.5 | External Software/Firmware Loaded | 19 |
| 4.6 | Bypass Actions and Status | 19 |
| 4.7 | Cryptographic Output Actions and Status | 19 |
| 5 | Software/Firmware Security | 20 |
| 5.1 | Integrity Techniques | 20 |
| 5.2 | Initiate on Demand | 20 |
| 6 | Operational Environment | 21 |
| 6.1 | Operational Environment Type and Requirements | 21 |
| 6.2 | Configurable Settings and Restrictions | 21 |
| 7 | Physical Security | 22 |
| 8 | Non-Invasive Security | 23 |
| 9 | Sensitive Security Parameters Management | 24 |
| 9.1 | Storage Areas | 24 |
| 9.2 | SSP Input-Output Methods | 24 |



| | | |
|------|-----------------------------------|----|
| 9.3 | SSP Zeroization Methods | 24 |
| 9.4 | SSPs | 24 |
| 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-Tests..... | 30 |
| 10.4 | Error States..... | 30 |
| 10.5 | Operator Initiation..... | 31 |
| 11 | Life-Cycle Assurance | 32 |
| 11.1 | Startup Procedures | 32 |
| 11.2 | Administrator Guidance..... | 32 |
| 11.3 | Non-Administrator Guidance | 32 |
| 11.4 | Maintenance Requirements..... | 32 |
| 11.5 | End of Life..... | 32 |
| 12 | Mitigation of Other Attacks | 33 |
| 13 | Glossary and Abbreviations | 34 |
| 14 | References..... | 35 |

List of Tables

| | |
|--|----|
| Table 1 - Security Levels | 1 |
| Table 2 - Tested Module Identification | 3 |
| Table 3 - Tested Operational Environments..... | 3 |
| Table 4 - Vendor Affirmed Operational Environments | 3 |
| Table 5 - Modes List and Description | 3 |
| Table 6 - Approved Algorithms | 9 |
| Table 7 - Vendor Affirmed Algorithms | 9 |
| Table 8 - Non-Approved, Not Allowed Algorithms | 10 |
| Table 9 - Security Function Implementations..... | 11 |
| Table 10 – Entropy | 12 |
| Table 11 - Key Generation | 13 |
| Table 12 - Key Establishment..... | 14 |
| Table 13 - Ports and Interfaces | 15 |
| Table 14 – Roles | 16 |
| Table 15 – Approved Services..... | 18 |
| Table 16 - Non-Approved Services..... | 19 |
| Table 17 – Storage Areas | 24 |
| Table 18 – SSP Input-Output..... | 24 |
| Table 19 - SSP Zeroization Methods..... | 24 |
| Table 20 – SSP Information First..... | 26 |
| Table 21 – SSP Information Second | 28 |
| Table 22 - Pre-Operational Self-Tests | 29 |
| Table 23 - Conditional Self-Tests..... | 30 |
| Table 24 - Error States | 31 |

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 software version 4.35.0-381552536e763d0c of the Oracle Linux 9 NSS 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

Table 1 describes the individual security areas of FIPS 140-3, as well as the security levels of those individual areas.

| ISO/IEC 24759 Section 6 Subsections | FIPS 140-3 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 | Not Applicable |
| 8 | Non-invasive Security | Not Applicable |
| 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 NSS Cryptographic Module (hereafter referred to as “the module”) is defined as a software module in a multi-chip standalone embodiment. It provides a C language application program interface (API) designed to support cross-platform development of security-enabled client and server applications. Applications built with NSS can support SSLv3, TLS, IKEv2, PKCS#5, PKCS#7, PKCS#11, PKCS#12, S/MIME, X.509 v3 certificates, and other security standards supporting FIPS 140-3 validated cryptographic algorithms. It combines a vertical stack of Linux components intended to limit the external interface each separate component may provide.

Module Type: Software

Module Embodiment: Multi-chip standalone

Module Characteristics: N/A

Cryptographic Boundary: Figure 1 shows the cryptographic boundary of the module, its interfaces with the operational environment and the flow of information between the module and operator (depicted through the arrows).

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.

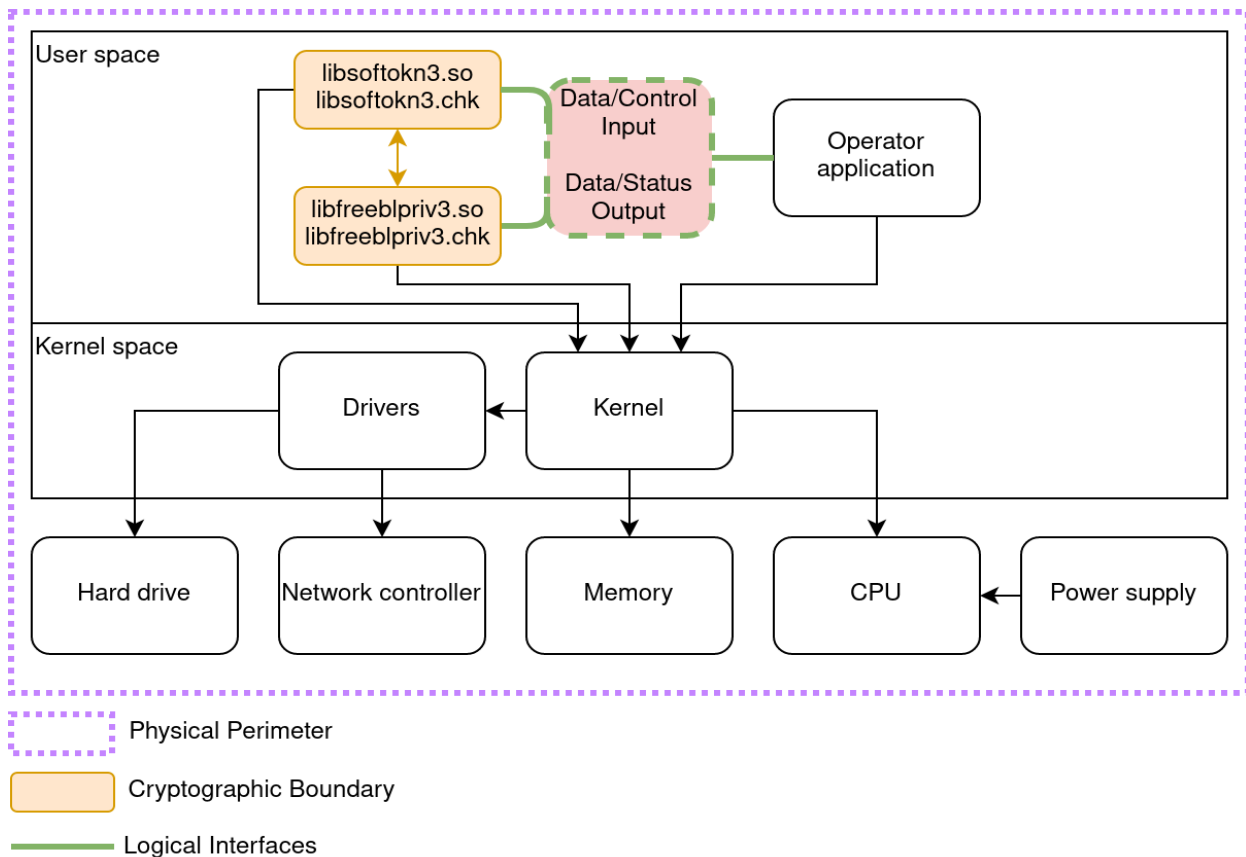


Figure 1 – Block Diagram



2.2 Tested and Vendor Affirmed Version and Identification

Hardware Operating Environments: N/A

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

| Package or File Names | Software/ Firmware Versions | Features | Hybrid Hardware Version | Integrity Test |
|------------------------------------|-----------------------------|----------|-------------------------|----------------|
| libsoftkn3.so libfreeblpriv3.so | 4.35.0-381552536e763d0c | N/A | N/A | HMAC-SHA-256 |

Table 2 - Tested Module Identification

Software, Firmware, Hybrid Tested Operating Environments:

| Operating System | Hardware Platform | Processor(s) | PAA/PAI | Hypervisor and Host OS |
|------------------|---------------------|----------------------------|--|------------------------|
| Oracle Linux 9 | ORACLE SERVER X9-2c | Intel® Xeon® Platinum 8358 | With and without AES-NI and SHA Extensions | KVM on Oracle Linux 8 |
| | ORACLE SERVER E4-2c | AMD EPYC 7J13 | With and without AES-NI and SHA Extensions | |
| | ORACLE SERVER A1-2c | Ampere® Altra® Q80-30 | With and without NEON and Cryptography Extensions (CE) | |

Table 3 - Tested Operational Environments

Vendor Affirmed Operating Environments:

| Operating Systems | Hardware Platforms | Virtual Platforms |
|-------------------|---|---------------------------------|
| Oracle Linux 9 | Oracle X Series Servers Oracle E Series Servers Oracle A Series Servers Marvell T93 LiquidIO III (ARM v8.x) SmartNIC Pensando DSC-200-R (ARM v8.x) SmartNIC | Oracle Linux KVM VmWare ESXi |

Table 4 - Vendor Affirmed Operational Environments

Note: the CMVP makes no statement as to the correct operation of the module or the security strengths of the generated SSPs 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:

| Name | Description | Type | Status Indicator |
|-------------------|--|--------------|--|
| Approved mode | Automatically entered whenever an approved service is requested | Approved | Equivalent to the indicator of the requested service |
| Non-approved mode | Automatically entered whenever a non-approved service is requested | Non-approved | Equivalent to the indicator of the requested service |

Table 5 - Modes List and Description

After passing all pre-operational self-tests and conditional 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.



Mode change instructions and status indicators:

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

Degraded Mode Description:

The module does not implement a degraded mode of operation.

2.5 Algorithms

Approved Algorithms:

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|----------------|--|---|---|---|
| SHA2-224 | #A4760 | N/A | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | FIPS 180-4 |
| SHA2-256 | #A4760 | N/A | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | |
| SHA2-384 | #A4760 | N/A | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | |
| SHA2-512 | #A4760 | N/A | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | |
| AES-ECB | #A4760 , #A4767 , #A4769 | Encryption, Decryption using 128, 192, 256-bit keys | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C, CE, AESNI</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C, CE, AESNI</p> | FIPS 197 SP 800-38A SP 800-38A Addendum |

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|----------------|--|---|--|---|
| | | | Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C, CE, AESNI | |
| AES-CBC | #A4760 , #A4767 , #A4769 | Encryption, Decryption using 128, 192, 256-bit keys | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C, CE, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C, CE, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C, CE, AESNI | FIPS 197 SP 800-38A SP 800-38A Addendum |
| AES-CBC-CS1 | #A4765 | Encryption, Decryption using 128, 192, 256-bit keys | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C CTS Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C CTS Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C CTS | FIPS 197 SP 800-38A Addendum |
| AES-CTR | #A4760 , #A4769 | Encryption, Decryption using 128, 192, 256-bit keys | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C, AESNI | FIPS 197 SP 800-38A SP 800-38A Addendum |
| AES-CMAC | #A4762 | Message Authentication using 128, 192, 256-bit keys | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C CMAC Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C CMAC Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C CMAC | FIPS 197 SP 800-38B |
| AES-GCM | #A4760 , #A4767 , #A4769 | Authenticated Encryption (internal IV), Authenticated Decryption (external IV) using 128, 192, 256-bit keys | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C, CE, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C, CE, AESNI Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C, CE, AESNI | FIPS 197 SP 800-38D |

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|----------------|--|--|---|--------------------------|
| | | IV Generation: Internal & External IV Generation Mode: 8.2.1 and 8.2.2 | | |
| AES-KW | #A4761 , #A4766 , #A4768 | Key Wrapping, Key Unwrapping using 128, 192, 256-bit keys | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C KW, AESNI KW</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C KW, AESNI KW</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C KW, AESNI KW</p> | FIPS 197 SP 800-38F |
| AES-KWP | #A4761 , #A4766 , #A4768 | Key Wrapping, Key Unwrapping using 128, 192, 256-bit keys | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C KW, AESNI KW</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C KW, AESNI KW</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C KW, AESNI KW</p> | FIPS 197 SP 800-38F |
| HMAC | #A4760 | SHA-224, SHA-256, SHA-384, SHA-512 112-524288-bit keys | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | FIPS 198-1 FIPS 180-4 |
| KBKDF | #A4763 | Modes: counter, feedback, double pipeline CMAC and HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C KBKDF</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C KBKDF</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C KBKDF</p> | SP 800-108r1 |
| HKDF | #A4759 | Key Derivation with HMAC SHA-224, SHA-256, SHA-384, SHA-512 | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: TLS v1.3</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: TLS v1.3</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: TLS v1.3</p> | SP 800-56Cr1 |

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|----------------------------|------------------------|--|---|--------------|
| TLS 1.0/1.1 KDF (CVL) | #A4760 | Key Derivation with SHA1 | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | SP 800-135r1 |
| TLS 1.2 KDF (CVL RFC 7627) | #A4760 | Hashes: SHA-256, SHA-384, SHA-512 | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | SP 800-135r1 |
| IKEv2 KDF (CVL) | #A4764 | Hashes: SHA-1, SHA-256, SHA-384, SHA-512 | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: IKE KDF</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: IKE KDF</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: IKE KDF</p> | SP 800-135r1 |
| PBKDF2 | #A4760 | <p>Option 1a</p> <p>Password length: 7-128 characters</p> <p>Salt length: 128-4096 bytes</p> <p>Iteration count: 1000-10000</p> <p>Hashes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512</p> | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | SP 800-132 |
| Hash_DRBG | #A4760 | Hashes: SHA-256 With/without prediction resistance | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | SP 800-90Ar1 |
| KAS-FFC-SSC | #A4760 | <p>Scheme: dhEphem</p> <p>Roles: initiator, responder</p> <p>Groups: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072,</p> | <p>Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C</p> <p>Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C</p> | SP 800-56Ar3 |

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|------------------|------------------------|---|---|--------------------------|
| | | ffdhe4096, ffdhe6144, ffdhe8192 | | |
| KAS-ECC-SSC | #A4760 | Scheme: Ephemeral Unified Model Roles: initiator, responder Curves: P-256, P-384, P-521 | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | SP 800-56Ar3 |
| DSA ¹ | #A4760 | Signature Verification Hashes: SHA-224, SHA-256, SHA-384, SHA-512 Keys: (L = 1024, N = 160); (L = 2048, N = 224); (L = 2048, N = 256); (L = 3072, N = 256) | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-4 |
| RSA | #A4760 | Signature Generation & Verification Padding: PKCS#1 v1.5 and PSS Hashes: SHA-224, SHA-256, SHA-384, SHA-512 Modulus: 2048-4096 bits | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-4 |
| RSA | #A4760 | Signature Verification Padding: PKCS#1 v1.5 and PSS Hashes: SHA-224, SHA-256, SHA-384, SHA-512 Modulus: 1024, 1280, 1536, 1792 bits | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-2 FIPS 186-4 |
| ECDSA | #A4760 | Signature Generation & Verification Hashes: SHA-224, SHA-256, SHA-384, SHA-512 Curves: P-256, P-384, P-521 | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-4 |
| Safe Primes | #A4760 | Key Pair Generation | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C | SP 800-56Ar3 |

¹ The following are allowed for legacy use only; DSA signature verification with L=1024 and N=224.

| Algorithm Name | CAVP Numbers | Algorithms Capabilities | OE (Implementation) | Reference |
|----------------|------------------------|---|---|------------|
| | | Mode: Testing Candidates (Appendix 5.6.1.1.4) Groups: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192 | Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | |
| RSA | #A4760 | Key Pair Generation Mode: Probable Primes (Appendix B.3.3) Modulus: 2048-4096 bits | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-4 |
| ECDSA | #A4760 | Key Pair Generation Mode: Extra Random Bits (Appendix B.4.1) Curves: P-256, P-384, P-521 | Oracle Linux 9 on KVM on Oracle Linux 8 on AMD EPYC™ 7001 Series AMD EPYC 7J13: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Ampere® Altra® Q80-30: Generic C Oracle Linux 9 on KVM on Oracle Linux 8 on Intel® Xeon® Platinum 8358: Generic C | FIPS 186-4 |

Table 6 - Approved Algorithms

Vendor Affirmed Algorithms:

| Algorithm Name | Algorithm Capabilities | OE (Implementation) | References |
|------------------------------------|--|---------------------|--|
| Cryptographic Key Generation (CKG) | Symmetric Key Generation using SP 800-90Ar1 Hash_DRBG : 112-256 bits of key strength Safe Primes: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192 groups (112-200 bits of key strength) RSA: 2048, 3072, 4096-bit keys (112, 128, 150 bits of key strength) ECDSA: P-256, P-384, P-521 elliptic curves (128-256 bits of key strength) | Same as in Table 6 | SP 800-133Rev2 section 4 example 1 and section 6.1 |

Table 7 - Vendor Affirmed Algorithms

Non-Approved, Allowed Algorithms:

The module does not implement non-approved algorithms allowed in the approved mode of operation.

Non-Approved, Allowed Algorithms with No Security Claimed:

| Algorithm Name | Caveat | Use/Function |
|----------------|--|---|
| MD5 | Only allowed as the PRF in TLSv1.0 and v1.1 per IG 2.4.A | Message digest used in TLS 1.0/1.1 KDF only |

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

Non-Approved, Not Allowed Algorithms:

| Name | Use and Function |
|--|---|
| MD2, MD5, SHA-1 | Message Digest |
| RC2, RC4, DES, Triple-DES, CDMF, Camellia, SEED, ChaCha20 (-Poly1305) | Encryption Decryption |
| AES GCM (external IV) | Encryption |
| CBC-MAC, AES XCBC-MAC, AES XCBC-MAC-96 | Message Authentication |
| HMAC-SHA-1, HMAC (MD2, MD5; < 112-bit keys) | |
| HMAC/SSLv3 MAC (constant-time implementation) | |
| RSA OAEP | Key Encapsulation/Decapsulation |
| SEED, ANS X9.63 KDF, SSL 3 PRF, 40v1 PRF | Key Derivation |
| KBKDF, HKDF, TLS 1.0/1.1 KDF, TLS 1.2 KDF, IKEv2 PRF (< 112-bit keys) | |
| KBKDF (MD2, MD5) | |
| TLS 1.2 KDF (without extended master secret) | |
| IKEv1 KDF | |
| IKEv2 PRF (MD2, MD5) | |
| PKCS#5 PBE, PKCS#12 PBE | Password-Based Key Derivation |
| PBKDF2 (short password; short salt; insufficient iterations; < 112-bit keys) | |
| J-PAKE | Shared Secret Computation |
| KAS-FFC-SSC (FIPS 186-type groups) | |
| KAS-ECC-SSC (P-192) | |
| DSA | Parameter Generation Parameter Verification Key Pair Generation Signature Generation |
| RSA with SHA-1, RSA (primitive; PKCS#1 v1.5 or PSS with MD2, MD5) | Signature Generation Signature Verification |
| ECDSA (P-192) | |
| RSA | Asymmetric Encryption Asymmetric Decryption |
| DH (FIPS 186-type groups) | Key Pair Generation |
| RSA (< 2048 bits) | |
| ECDSA (P-192) | |
| Symmetric key generation (< 112 bits) | Secret Key Generation |

Table 9 - Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

| Name | Type | Description | SF Capabilities | Algorithms |
|------------------------------|------|---------------------------|--|-------------------------------------|
| KAS-ECC-SSC [SP 800-56ARev3] | KAS | Shared Secret Computation | Ephemeral Unified scheme Curves: P-256, P-384, P-521 elliptic curves with 128-256 bits of key strength Compliant with IG D.F scenario 2(1) | KAS-ECC-SSC: #A4760 |
| KAS-FFC-SSC [SP 800-56ARev3] | | | Ephemeral Unified scheme Keys: 2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of key strength | KAS-FFC-SSC: #A4760 |

| | | | | |
|------------------------------|-----|--|--|---|
| | | | Compliant with IG D.F scenario 2(1) | |
| AES-GCM [SP 800-38D] | KTS | Key Wrapping (internal IV), Key Unwrapping (external IV) | 128, 192, 256 bits with 128-256 bits of key strength IV Generation: Internal & External IV Generation Mode: 8.2.1 and 8.2.2 Compliant with IG D.G | AES-GCM: #A4760 , #A4767 , #A4769 |
| AES-KW, AES-KWP [SP 800-38F] | | Key Wrapping, Key Unwrapping | 128, 192, 256 bits with 128-256 bits of key strength Compliant with IG D.G | AES-KW, AES-KWP: #A4761 , #A4766 , #A4768 |

Table 10 - Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES GCM IV

The Crypto Officer shall consider the following requirements and restrictions when using the module.

For TLS 1.2, the module offers the AES GCM implementation and uses the context of Scenario 1 of FIPS 140-3 IG C.H. NSS is compliant with SP 800-52r2 Section 3.3.1 and the mechanism for IV generation is compliant with RFC 5288 and 8446.

The module does not implement the TLS protocol. The module’s implementation of AES GCM is used together with an application that runs outside the module’s cryptographic boundary. The design of the TLS protocol implicitly ensures that the counter (the nonce_explicit part of the IV) does not exhaust the maximum number of possible values for a given session key.

In the event the module’s power is lost and restored, the consuming application must ensure that a new key for use with the AES GCM key encryption or decryption under this scenario shall be established.

Alternatively, the Crypto Officer can use the module’s API to perform AES GCM encryption using internal IV generation compliant with Scenario 2 of FIPS 140-3 IG C.H. These IVs are always 96 bits and generated using the approved DRBG internal to the module’s boundary.

Additionally, the module offers an internal deterministic IV generation mode compliant with Scenario 3 of FIPS 140-3 IG C.H. The size of the fixed (name) field used by this IV generation mode is at least 32 bits. The module then internally generates a 32 bit or longer deterministic non-repetitive counter. The module explicitly ensures that this counter is monotonically increasing at each invocation of the AES-GCM for the same encryption key, and that this counter does not exhaust all its possible values. The generated GCM IV is at least 96 bits in length.

Finally, for TLS 1.3, the AES GCM implementation uses the context of Scenario 5 of FIPS 140-3 IG C.H. The protocol that provides this compliance is TLS 1.3, defined in RFC8446 of August 2018, using the cipher-suites that explicitly select AES GCM as the encryption/decryption cipher (Appendix B.4 of RFC8446). The module supports acceptable AES GCM cipher suites from Section 3.3.1 of SP800-52r2. TLS 1.3 employs separate 64-bit sequence numbers, one for protocol records that are received, and one for protocol records that are sent to a peer. These sequence numbers are set at zero at the beginning of a TLS 1.3 connection and each time when the AES-GCM key is changed. After reading or writing a record, the respective sequence number is incremented by one. The protocol specification determines that the sequence number should not wrap, and if this condition is observed, then the protocol implementation must either trigger a re-key of the session (i.e., a new key for AES-GCM), or terminate the connection.

2.7.2 Key Derivation using SP 800-132 PBKDF2

The module provides password-based key derivation (PBKDF2), 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 MK shall not be used for other purposes. The length of the MK or DPK shall be of 112 bits or more.



- Passwords or passphrases, used as an input for the PBKDF2, shall not be used as cryptographic keys.
- The length of the password or passphrase shall be at least 7 characters, and shall consist of lowercase, uppercase, and numeric characters. In a worst-case scenario where the user selects a password consisting of only digits, the guessing probability is estimated to be 10^{-7} . Combined with the minimum iteration count as described below, this provides an acceptable trade-off between user experience and security against brute-force attacks.
- A portion of the salt, with a length of at least 128 bits, shall be generated randomly using the SP 800-90Ar1 DRBG provided by the module.
- The iteration count shall be 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 is 1000.

2.7.3 SP 800-56Ar3 Assurances

The module offers DH and ECDH shared secret computation services compliant to the SP 800-56Ar3 and meeting IG D.F scenario 2 path (1). To meet the required assurances listed in section 5.6 of SP 800-56Ar3, the module shall be used together with an application that implements the “TLS protocol” and the following steps shall be performed.

- The entity using the module, must use the module's "Key pair generation" service for generating DH/ECDH ephemeral keys. This meets the assurances required by key pair owner defined in the section 5.6.2.1 of SP 800-56Ar3.
- As part of the module's Shared Secret Computation (SSC) service, the module internally performs the public key validation on the peer's public key passed in as input to the SSC function. This meets the public key validity assurance required by the sections 5.6.2.2.1/5.6.2.2.2 of SP 800-56Ar3.
- The module does not support static keys therefore the "assurance of peer's possession of private key" is not applicable.

2.8 RNG and Entropy

Entropy Information:

| Name | Type | Operational Environment | Sample Size | Entropy Per Sample | Conditioning Component |
|--|--------------|-------------------------|-------------|--------------------|---|
| Oracle Userspace CPU Time Jitter RNG Entropy Source (Cert. # E99) | Non-physical | See Table 3 | 256 bits | 256 bits | HMAC-SHA2-512-DRBG (CAVP cert A4162) |

Table 11 – Entropy

RNG Information:

The module implements a Deterministic Random Bit Generator (Hash_DRBG) implementation compliant with SP 800-90Ar1. This DRBG is used internally by the module (e.g., to generate symmetric keys, seeds for asymmetric key pairs, and random numbers for security functions). It can also be accessed using the specified API functions. The module DRBG implemented is a SHA-256 Hash_DRBG, seeded by the entropy source specified in Table 11.

2.9 Key Generation

| Name | Type | Properties |
|-------------------------------------|------|---|
| Direct Generation of Symmetric Keys | CKG | Key type: Symmetric key Security strength: 112-256 bits Method: Direct Generation Compliant to SP 800-133r2, Section 6.1 |
| Safe Primes Key Pair Generation | | Key type: DH key pair Groups: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192 Security strength: 112-200 bits Method: SP 800-56Ar3 (safe primes) Section 5.6.1.1.4 Testing Candidates Compliant to SP 800-133r2, Section 5.2 |
| ECDSA Key Pair Generation | | Key type: EC key pair Curves: P-256, P-384, P-521 |

| | | |
|-------------------------|----------------|--|
| | | <p>Security strength: 128-256 bits Method: FIPS 186-4 Appendix B.4.1 Extra Random Bits Compliant to SP 800-133r2, Section 4</p> |
| RSA Key Pair Generation | | <p>Key type: RSA key pair Modulus: 2048-4096 bits Security strength: 112-150 bits Method: FIPS 186-4 Appendix B.3.3 Probable Primes Compliant to SP 800-133r2, Section 4</p> |
| KBKDF | Key Derivation | <p>Key type: Symmetric key Security strength: 112-256 bits Method: Counter, feedback and double pipeline mode, using CMAC and HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512</p> |
| HKDF | | <p>Key type: Symmetric key Security strength: 112-256 bits Method: (HMAC) SHA-224, SHA-256, SHA-384, SHA-512 Compliant to SP 800-56Cr1</p> |
| TLS 1.0/1.1 KDF | | <p>Key type: Symmetric key Security strength: 112-256 bits Method: SHA1 Compliant to SP 800-135r1</p> |
| TLS 1.2 KDF (RFC 7627) | | <p>Key type: Symmetric key Security strength: 112-256 bits Method: SHA-256, SHA-384, SHA-512 Compliant to SP 800-135r1</p> |
| IKEv2 KDF | | <p>Key type: Symmetric key Security strength: 112-256 bits Method: SHA-1, SHA-256, SHA-384, SHA-512 Compliant to SP 800-135r1</p> |
| PBKDF2 | | <p>Key type: Symmetric key Security strength: 112-256 bits Method: Option 1a with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 Compliant to option 1a of SP 800-132</p> |

Table 12 - Key Generation

2.10 Key Establishment

| Name | Type | Properties |
|------------------------------|---|--|
| KAS-FFC-SSC [SP800-56Arev3] | KAS (Shared Secret Computation) | Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Security strength: 112-200 bits Compliant with: Scenario 2 (1) of FIPS 140-3 IG D.F: Shared Secret Computation |
| KAS-ECC-SSC [SP800-56Arev3] | | Curves: P-256, P-384, P-521 Security strength: 128-256 bits Compliant with: Scenario 2 (1) of FIPS 140-3 IG D.F: Shared Secret Computation |
| AES GCM [SP 800-38D] | KTS-Wrap (Key Wrapping) | Keys: 128, 192, or 256 bits IV generated internally Security strength: 128, 192, or 256 bits Compliant with IG D.G |
| | KTS-Wrap (Key Unwrapping) | Keys: 128, 192, or 256 bits IV provided externally Security strength: 128, 192, or 256 bits Compliant with IG D.G |
| AES KW, AES KWP [SP 800-38F] | KTS-Wrap (Key Wrapping, Key Unwrapping) | Keys: 128, 192, or 256 bits Security strength: 128, 192, or 256 bits Compliant with IG D.G |

Table 13 - Key Establishment

2.11 Industry Protocols

For DH, the module supports the use of the safe primes defined in RFC 3526 (IKE) and RFC 7919 (TLS) as listed in Table 13. Note that the module only implements domain parameter generation, key pair generation and verification, and shared secret computation.

TLS 1.0/1.1 KDF, TLS 1.2 KDF (RFC 7627), IKEv2 implementations shall only be used to generate secret keys in the context of the TLS 1.0/1.1, TLS 1.2, IKE protocols respectively. No parts of this protocol, other than the approved cryptographic algorithms and the KDFs, have been tested by the CAVP and CMVP.

3 Cryptographic Module Interfaces

3.1 Description

| Physical Port | Logical Interface | Data That Passes Over the Port/Interface |
|--|-------------------|--|
| As a software-only module, the module does not have physical ports. Physical Ports are interpreted to be the physical ports of the hardware platform on which it runs. | Data Input | API data input parameters |
| | Data Output | API output parameters |
| | Control Input | API function calls, API control input parameters |
| | Status Output | API return code, error queue |

Table 14 - 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.

3.2 Trusted Channel Specification

The module does not implement a trusted channel.

3.3 Control Interface Not Inhibited

The module does not implement a control output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module does not implement authentication.

4.2 Roles

| Name | Type | Operator Type | Authentication Methods |
|----------------|------|---------------|--------------------------|
| Crypto Officer | Role | CO | N/A (Implicitly assumed) |

Table 15 – Roles

The module supports the Crypto Officer role only. This sole role is implicitly and always assumed by the operator of the module. No support is provided for multiple concurrent operators.

4.3 Approved Services

| Name | Description | Indicator | Inputs | Outputs | Security Functions | Roles | SSP Access |
|-------------------------------------|--------------------------|----------------------------|------------------------------|--|---|-----------------------|--|
| Message Digest | Compute a message digest | CKS_NSS_FI PS_OK | Message | Digest value | SHA2-224, SHA2-256, SHA2-384, SHA2-512 | CO | N/A |
| Encryption | Encrypt a plaintext | | AES Key, plaintext | Ciphertext | AES-ECB, AES-CBC, AES-CBC-CS1, AES-CTR, AES-GCM (internal IV) | | AES Key: W, E |
| Decryption | Decrypt a ciphertext | | AES Key, ciphertext | Plaintext | AES-ECB, AES-CBC, AES-CBC-CS1, AES-CTR, AES-GCM (external IV) | | |
| Key Wrapping | Wrap a key | | AES Key, Key To Be Wrapped | Wrapped key | AES-GCM, AES-KW, AES-KWP | | AES Key: W, E Wrapped Key: R Key To Be Wrapped: W, E |
| Key Unwrapping | Unwrap a key | | AES Key, Key To Be Unwrapped | Unwrapped key | | | AES Key: W, E Unwrapped Key: R Key To Be Unwrapped: W, E |
| Message Authentication | Compute a MAC tag | | AES Key, message | MAC tag | AES-CMAC | | AES Key: W, E |
| | | | HMAC Key, message | | HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 | | HMAC Key: W, E |
| Message Authentication Verification | Verify a MAC tag | AES Key, message, MAC tag | Pass/fail | AES-CMAC | | AES Key: W, E | |
| | | HMAC Key, message, MAC tag | | HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 | | HMAC Key: W, E | |
| Shared Secret | Compute a | DH Private Key, | Shared | KAS-FFC-SSC | | DH Private Key: W, E; | |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | Roles | SSP Access |
|-------------------------------|------------------------------|--------------------|----------------------------------|---------------------------------|---|-------|---|
| Computation | shared secret | | DH Public Key | Secret | | | DH Public Key: W, E; Shared Secret: G, R |
| | | | EC Private Key, EC Public Key | | KAS-ECC-SSC | | EC Private Key: W, E; EC Public Key: W, E; Shared Secret: G, R |
| Signature Generation | Generate a digital signature | | Message, private key | Signature | RSA PKCS#1 v1.5 and PSS with SHA-224, SHA-256, SHA-384, SHA-512, ECDSA with SHA-224, SHA-256, SHA-384, SHA-512 | | RSA Private Key: W, E; ECDSA Private Key: W, E |
| Signature Verification | Verify a digital signature | | Message, public key, signature | Pass/fail | RSA PKCS#1 v1.5 and PSS with SHA-224, SHA-256, SHA-384, SHA-512, ECDSA with SHA-224, SHA-256, SHA-384, SHA-512 DSA with SHA-224, SHA-256, SHA-384, SHA-512 | | RSA Public Key: W, E; ECDSA Public Key: W, E; DSA Public Key: W, E |
| Key Pair Generation | Generate a key pair | | DH Group | DH Private Key, DH Public Key | Safe Primes | | DH Private Key: G, R; DH Public Key: G, R; Intermediate Key Generation Value: G |
| | | | Curve | EC Private Key, EC Public Key | ECDSA | | EC Private Key: G, R; EC Public Key: G, R; Intermediate Key Generation Value: G |
| | | | Modulus | RSA Private Key, RSA Public Key | RSA | | RSA Private Key: G, R; RSA Public Key: G, R; Intermediate Key Generation Value: G |
| Secret Key Generation | Generate a secret key | | Key size | Symmetric Key | CKG (Symmetric Key Generation) | | Symmetric Key (AES Key, HMAC Key Key-Derivation Key): G |
| Key Derivation | Derive a key | | Shared Secret | Derived Key | HKDF, TLS 1.0/1.1 KDF, TLS 1.2 KDF, IKEv2 KDF | | Shared Secret: W, E; Derived Key: G, R |
| Key-Based Key Derivation | Derive a key from a key | Key-Derivation Key | | | KBKDF | | Key-Derivation Key: W, E; Derived Key: G, R |
| Password-Based Key Derivation | Derive a key from a password | Password | | | PBKDF2 | | Password: W, E; Derived Key: G, R |
| Random Number Generation | Generate random bytes | CKR_OK | Output length | Random bytes | Hash_DRBG, | | Entropy Input: W, E; DRBG Seed: E, G; Internal State (V, C): W, E, G |
| Show Version | Return the | None | N/A | Module | N/A | | N/A |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | Roles | SSP Access |
|-------------|---------------------------------------|-----------|---------|------------------|---|-------|-------------|
| | module name and version information | | | name and version | | | |
| Show Status | Return the module status | None | N/A | Module status | N/A | | N/A |
| Self-Test | Perform the CASTs and integrity tests | None | N/A | Pass/fail | SHA-224, SHA-256, SHA-384, SHA-512 AES-GCM, AES-ECB, AES-CBC, AES-CMAC, HMAC, KBKDF, HKDF, TLS 1.0/1.1 KDF, TLS 1.2 KDF, IKEv2 KDF, PBKDF2, Hash_DRBG, KAS-FFC-SSC, KAS-ECC-SSC, RSA, ECDSA, DSA See Table 24 for specifics | | N/A |
| Zeroization | Zeroize all SSPs | None | Any SSP | N/A | N/A | | All SSPs: Z |

Table 16 – Approved Services

Table 16 lists the approved services. 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.

To interact with the module, a calling application must use the FIPS token APIs provided by Softoken. The FIPS token API layer can be used to retrieve the approved service indicator for the module. This indicator consists of four independent service indicators.

1. The session indicator, which must be used for all cryptographic services except the key derivation service. It can be accessed by invoking the NSC_NSSGetFIPSStatus function with the CKT_NSS_SESSION_LAST_CHECK parameter. If the output parameter is set to CKS_NSS_FIPS_OK (1), the service was approved.
2. The object indicator, which must be used for the key derivation service. It can be accessed by invoking the NSC_NSSGetFIPSStatus function with the CKT_NSS_OBJECT_CHECK parameter and the output derived key. If the output parameter is set to CKS_NSS_FIPS_OK (1), the service was approved.
3. The DRBG service indicator, which must be used for the DRBG service. It can be accessed by invoking the C_SeedRandom or C_GenerateRandom functions. If any of these functions returns CKR_OK, the service was approved.

4.4 Non-Approved Services

| Name | Description | Security Functions | Role |
|---------------------------------|-------------------------------|--|------|
| Message Digest | Compute a message digest | MD2, MD5, SHA-1 | CO |
| Encryption | Encrypt a plaintext | RC2, RC4, DES, Triple-DES, CDMF, Camellia, SEED, ChaCha20(-Poly1305), AES GCM (external IV) | |
| Decryption | Decrypt a ciphertext | RC2, RC4, DES, Triple-DES, CDMF, Camellia, SEED, ChaCha20(-Poly1305) | |
| Message Authentication | Compute a MAC tag | CBC-MAC, AES XCBC-MAC, AES XCBC-MAC-96; HMAC-SHA-1, HMAC (MD2, MD5; < 112-bit keys); HMAC/SSLv3 MAC (constant-time implementation) | |
| Key Encapsulation/Decapsulation | Encapsulate/Decapsulate a key | RSA OAEP | |
| Key Derivation | Derive a key | SEED, ANS X9.63 KDF, SSL 3 PRF, IKEv1 KDF, KBKDF, HKDF, TLS 1.0/1.1 KDF, TLS 1.2 KDF, IKEv2 KDF (< 112-bit keys), KBKDF (MD2, MD5), TLS 1.2 KDF (without extended master secret), IKEv2 KDF (MD2, MD5) | |
| Password-Based Key Derivation | Derive a key from a password | PKCS#5 PBE, PKCS#12 PBE, PBKDF2 (short password; short salt; insufficient iterations; < 112-bit keys) | |
| Shared Secret Computation | Compute a shared secret | J-PAKE, KAS-FFC-SSC (FIPS 186-type groups), KAS-ECC-SSC (P-192) | |
| Signature Generation | Generate a signature | DSA, RSA with SHA-1, RSA (primitive; PKCS#1 v1.5 or PSS with MD2, MD5), ECDSA (P-192) | |
| Signature Verification | Verify a signature | RSA with SHA-1, RSA (primitive; PKCS#1 v1.5 or PSS with MD2, MD5), ECDSA (P-192) | |
| Asymmetric Encryption | Encrypt a plaintext | RSA | |
| Asymmetric Decryption | Decrypt a plaintext | | |
| Parameter Generation | Generate domain parameters | DSA | |
| Parameter Verification | Verify domain parameters | | |
| Key Pair Generation | Generate a key pair | DH (FIPS 186-type groups), RSA (< 2048 bits), DSA, ECDSA (P-192) | |
| Secret Key Generation | Generate a secret key | CKG (< 112 bits) | |

Table 17 - Non-Approved Services

4.5 External Software/Firmware Loaded

The module does not load external software or firmware.

4.6 Bypass Actions and Status

The module does not implement a bypass capability.

4.7 Cryptographic Output Actions and Status

The module does not implement a self-initiated cryptographic output capability.



5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by comparing a HMAC-SHA-256 value calculated at run time with the HMAC-SHA-256 value embedded in the module that was computed at build time. If the integrity test fails, the module enters the Power-On Error state.

5.2 Initiate on Demand

Integrity tests are performed as part of the pre-operational self-tests, which are executed when the module is initialized. The integrity test may be invoked on-demand by unloading and subsequently re-initializing the module.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operating Environment: modifiable: the module executes on a general-purpose operating system (Oracle Linux 9), which allows modification, loading, and execution of software that is not part of the validated module.

How Requirements are Satisfied: 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 Configurable Settings and Restrictions

The module shall be installed as stated in Section 11.1.

There are no concurrent operators.

The module does not have the capability of loading software or firmware from an external source.

Instrumentation tools like the ptrace system call, gdb and strace, userspace live patching, 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. The module does not perform persistent storage of SSPs. | Dynamic |

Table 18 – Storage Areas

9.2 SSP Input-Output Methods

| Name | From | To | Format Type | Distribution Type | Entry Type | Related SFI |
|-----------------------------------|-----------------------------|-----------------------------|-------------------------------------|-------------------|-----------------|--|
| API input parameters (plaintext) | Calling application (TOEPP) | Cryptographic module | Plaintext (P) | Manual (MD) | Electronic (EE) | AES-KW, AES-KWP [SP 800-38F]; AES-GCM [SP 800-38D] |
| API input parameters (encrypted) | | | Encrypted (E) with AES GCM, KW, KWP | | | |
| API output parameters (plaintext) | Cryptographic module | Calling application (TOEPP) | Plaintext (P) | | | |
| API output parameters (encrypted) | | | Encrypted (E) with AES GCM, KW, KWP | | | |

Table 19 – SSP Input-Output

9.3 SSP Zeroization Methods

| Zeroization Method | Description | Rationale | Operator Initiation |
|------------------------------|--|---|---|
| Calling the zeroization API | Zeroizes the SSPs | Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable. | By calling the C_DestroyObject function |
| Automatic | Automatically zeroized by the module when no longer needed | | N/A |
| 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 | By removing power |

Table 20 - SSP Zeroization Methods

All data output is inhibited during zeroization.

9.4 SSPs

| Name | Description | Size | Strength | Type | Generated By | Established By |
|---------|--|--------------|--------------|---------------|---------------------------------|----------------|
| AES Key | AES key used for Encryption, Decryption, Key Wrapping, Key Unwrapping, Message Authentication, | 128-256 bits | 128-256 bits | Symmetric key | CKG (SP 800-133r2, Section 6.1) | N/A |

| Name | Description | Size | Strength | Type | Generated By | Established By |
|---|---|-----------------------------------|--------------|---------------------|---|--|
| | Message Authentication Verification | | | | | |
| Wrapped Key | Wrapped key | 128-8192 bits | 112-256 bits | Wrapped key | N/A | AES-GCM, AES-KW, AES-KWP |
| Unwrapped Key | Unwrapped key | 128-8192 bits | 112-256 bits | Unwrapped key | N/A | AES-GCM, AES-KW, AES-KWP |
| Key To Be Wrapped | Key To Be Wrapped | 128-256 bits | 112-256 bits | Key To Be Wrapped | N/A | AES-GCM, AES-KW, AES-KWP |
| Key To Be Unwrapped | Key To Be Unwrapped | 128-256 bits | 112-256 bits | Key To Be Unwrapped | N/A | AES-GCM, AES-KW, AES-KWP |
| HMAC Key | HMAC key used for Message Authentication, Message Authentication Verification | 112-524288 bits | 112-256 bits | Authentication key | CKG (SP 800-133r2, Section 6.1) | N/A |
| Key-Derivation Key | Key for key derivation with KBKDF | 112-4096 bits | 112-256 bits | Key-derivation key | CKG (SP 800-133r2, Section 6.1) | N/A |
| Shared Secret | Shared secret established by KAS-ECC-SSC/KAS-FFC-SSC | 256-8192 bits | 112-256 bits | Shared secret | N/A | KAS-ECC-SSC, KAS-FFC-SSC (according to SP 800-56Ar3) |
| Password | PBKDF2 password | 64-1024 bits | N/A | Password | N/A | N/A |
| Derived Key | KBKDF derived key | 128-4096 bits | 112-256 bits | Derived key | KBKDF | N/A |
| | PBKDF2 derived key | 128-4096 bits | | | PBKDF2 | |
| | KDA HKDF derived key | 2048 bits | | | KDA HKDF | |
| | TLS 1.0/1.1 KDF derived key | 384 bits | | | TLS 1.0/1.1 KDF | |
| | TLS v1.2 KDF (RFC7627) derived key | 1024 bits | | | TLS v1.2 KDF | |
| | IKEv2 KDF derived key | 1056 and 3072 bits | | | IKEv2 KDF | |
| Entropy Input | Entropy input used to seed the DRBG | 256 bits | 256 bits | Entropy | ENT (NP) See Table 11 | N/A |
| DRBG Seed IG D.L compliant | DRBG seed derived from entropy input as defined in SP 800-90Ar1 | 256 bits | 256 bits | Seed | Hash_DRBG (according to SP800-90Ar1) | N/A |
| Internal State (V, C) IG D.L compliant | Internal state of Hash_DRBG | Hash_DRBG: 128, 256 bits | 256 bits | Internal state | Hash_DRBG (derived from DRBG Seed as defined in SP800-90Ar1) | N/A |
| DH Public Key | Public key used by DH | 2048, 3072, 4096, 6144, 8192 bits | 112-200 bits | Public key | Safe Primes (SP 800-56Ar3 section 5.6.1.1.4 Testing Candidates) Hash_DRBG (for generation of random values per SP 800- | N/A |
| DH Private Key | Private key used by DH | | | Private key | | |

| Name | Description | Size | Strength | Type | Generated By | Established By |
|-----------------------------------|--|--|------------------------|--------------------|---|----------------|
| | | | | | 90Ar1) | |
| EC Private Key | Private key used for ECDSA Signature Generation and Shared Secret Computation | P-256, P-384, P-521 | 128-256 bits | Private key | ECDSA (FIPS 186-4 Appendix B.4.1 Extra Random Bits) Hash_DRBG (for generation of random values per SP 800-90Ar1) | N/A |
| EC Public Key | Public key used for ECDSA Signature Verification and Shared Secret Computation | | | Public key | | |
| RSA Private Key | Private key used for RSA signature generation | 2048, 3072, 4096 bits | 112, 128, 150 bits | Private key | RSA (FIPS 186-4 Appendix B.3.3 Probable Primes) | N/A |
| RSA Public Key | Public key used for RSA signature verification | 1024, 2048, 3072, 4096 bits | 80, 112, 128, 150 bits | Public key | Hash_DRBG (for generation of random values per SP 800-90Ar1) | |
| DSA Public Key | Public key used for DSA signature verification | (1024, 160), (2048, 224), (2048, 256), (3072, 256) | 80, 112, 128 bits | Public key | N/A | N/A |
| Intermediate Key Generation Value | Intermediate key generation value | 112-8192 bits | 112-256 bits | Intermediate Value | CKG (SP 800-133r2 Section 4, 6.1) | N/A |

Table 21 – SSP Information First

| Name | Used By | Inputs/Outputs | Storage | Storage Duration | Zeroization | Type | Related SSPs |
|---------------------|---|---|---------|---------------------------------------|--|------|---|
| AES Key | Encryption, Decryption, Key Wrapping, Key Unwrapping, Message Authentication, Message Authentication Verification | API input parameters (encrypted); API output parameters (encrypted) | RAM | Until explicitly zeroized by operator | Destroy object, remove power from the module | CSP | N/A |
| Wrapped key | Key wrapping | | | | | | Wrapped using Symmetric key (AES Key) |
| Unwrapped key | Key unwrapping | No input; API output parameters (plaintext) | | | | | Unwrapped using Symmetric key (AES Key) |
| Key To Be Wrapped | Key wrapping | API input parameters (plaintext); No output | | | | | Key to be wrapped using Symmetric key (AES Key) |
| Key To Be Unwrapped | key unwrapping | API input parameters (encrypted); No output | | | | | Key to be unwrapped using Symmetric key (AES Key) |
| HMAC Key | Message Authentication, | API input parameters | | | | | N/A |

| Name | Used By | Inputs/Outputs | Storage | Storage Duration | Zeroization | Type | Related SSPs |
|---|---|--|---|--|--|--|---|
| | Message Authentication Verification | (encrypted); API output parameters (encrypted) | | | | | |
| Key-Derivation Key | Key-Based Key Derivation | API input parameters (encrypted); API output parameters (encrypted) | | | | | Used for deriving a key (i.e., Derived Key) |
| Shared Secret | Shared Secret Computation, Key Derivation | API input parameters (encrypted); API output parameters (encrypted) | | | | | Established using DH public, DH Private Key, EC Public Key, EC Private Key, Derived Key |
| Password | Password-Based Key Derivation | API input parameters (plaintext); No output | | | | | Used for deriving a key (i.e., Derived Key) |
| Derived Key | Key Derivation, Key-Based Key Derivation, Password-Based Key Derivation | No input; API output parameters (encrypted) | | | | | Derived from Key-Derivation Key, Shared Secret or Password |
| Entropy Input | Random Number Generation | No input; No output | | From generation until DRBG Seed is created | Automatic, remove power from the module | | Used for deriving DRBG Seed |
| DRBG Seed IG D.L compliant | | | | While the DRBG is instantiated | | | Derived from Entropy Input; used for the generation of Internal State (V, C) |
| Internal State (V, C) IG D.L compliant | | | | While the module is operational | | Remove power from the module | Generated from DRBG Seed |
| DH Public Key | Shared Secret Computation, Key Pair Generation | API input parameters (encrypted); API output parameters (encrypted) | | For the duration of the service | Destroy object, remove power from the module | PSP | Paired with DH Private Key; used for establishing Shared Secret |
| DH Private Key | | | CSP | | | Paired with DH Public Key, used for establishing Shared Secret | |
| EC Public Key | Shared Secret Computation, Signature Verification | API input parameters (encrypted); API output parameters (encrypted) | | | | PSP | Paired with EC Private Key; used for establishing Shared Secret |
| EC Private Key | | | Shared Secret Computation, Signature Generation | | | CSP | Paired with EC Public Key; used for establishing Shared Secret |
| RSA Public Key | Signature Verification | API input parameters | | | | PSP | Paired with RSA Private Key |

| Name | Used By | Inputs/Outputs | Storage | Storage Duration | Zeroization | Type | Related SSPs |
|-----------------------------------|------------------------|---|---------|------------------|---------------|------|--|
| RSA Private Key | Signature Generation | (encrypted); API output parameters (encrypted) | | | | CSP | Paired with RSA Public Key |
| DSA Public Key | Signature Verification | API input parameters (encrypted); No output | | | | PSP | N/A |
| Intermediate Key Generation Value | Key Pair Generation | N/A | | | Automatically | CSP | Generated during the generation of RSA Public Key, RSA Private Key, DH Public Key, DH Private Key, EC Public Key, EC Private Key |

Table 22 – SSP Information Second

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, ECDSA algorithm as implemented by the module conforms to FIPS 186-4, which has been superseded by FIPS 186-5. FIPS 186-4 will be withdrawn on February 3, 2024.

| Algorithm | Implementation | Test Properties | Test Method | Type | Indicator | Details | Conditions |
|-------------|----------------|---|-------------|---|--------------------------------------|---|----------------------------|
| | | SHA-512 80, 128, 144-bit input secret | | | | | |
| PBKDF2 | | SHA-256 14-character password 128-bit salt Iteration count: 5 | | | | | |
| Hash_DRBG | | SHA-256 without prediction resistance | | | | Instantiate Generate Reseed Generate (compliant with SP 800- 90A Section 11.3) | Freebl initialization |
| KAS-FFC-SSC | | 2048-bit key | | | | Shared Secret Computation | |
| KAS-ECC-SSC | | P-256 | | | | | |
| RSA | | PKCS#1 v1.5 with SHA-256, SHA-384, SHA-512 2048-bit key | | | | Signature Generation Signature Verification | Softoken initialization |
| DSA | | 1024-bit key | | | | Signature Verification | Freebl initialization |
| ECDSA | | SHA-256 P-256 | | | | Signature Generation Signature Verification | Freebl initialization |
| Safe Primes | | N/A | PCT | Conditional Pairwise Consistency Self-Test | Key Pair Generation is successful | SP 800-56Ar3 Section 5.6.2.1.4 | Key Pair Generation |
| ECDH | | | | | | | |
| RSA | | PKCS#1 v1.5 with SHA-256 | | | | Signature Generation & Signature Verification | |
| ECDSA | | SHA-256 | | | | | |

Table 24 - 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 24. Data output through the data output interface is inhibited during the self-tests. If any of these tests fails, the module transitions to the Power-On Error state.

Upon generation of a DH, EC or RSA key pair, the module will perform a pair-wise consistency test (PCT) as shown in Table 24, which provides some assurance that the generated key pair is well formed. The test for DH and ECDH consists of the PCT described in Section 5.6.2.1.4 of SP 800-56Ar3. For ECDSA or RSA key pairs, the tests consist of performing signature generation and verification using the generated key pairs. Services are not available, and data output (via the data output interface) is inhibited during execution of the PCT. If a PCT test fails, the module transitions to the PCT Error state.

10.3 Periodic Self-Tests

The module does not implement any periodic self-tests.

10.4 Error States

| Name | Description | Conditions | Recovery Method | Indicator |
|----------------|--|---------------------------------|-----------------------|----------------------|
| Power-On Error | An error occurred during the self-tests executed | Software integrity test failure | Restart of the module | Module will not load |

| Name | Description | Conditions | Recovery Method | Indicator |
|-----------|--------------------------------|--------------|-----------------|---|
| | on power-on | CAST failure | | |
| PCT Error | An error occurred during a PCT | PCT failure | | Module stops functioning (sftk_fatalError is set to TRUE) |

Table 25 - Error States

In any error state, the output interface is inhibited, and the module accepts no more inputs or requests (as the module is no longer running).

10.5 Operator Initiation

The software integrity tests and CASTs can be invoked on demand by unloading and subsequently re-initializing the module. The PCTs can be invoked on demand by requesting the key pair generation service.



11 Life-Cycle Assurance

11.1 Startup Procedures

The module is distributed as part of the Oracle Linux 9 (OL9) RPM package in the form of `nss-softokn-3.90.0-3.0.1.el9_2_fips` and `nss-softokn-freebl-3.90.0-3.0.1.el9_2_fips` RPM packages that are located in the “Oracle Linux 9 Security Validation (Update 3)” yum repository (`ol9_u3_security_validation`).

The module can achieve the FIPS validated configuration by:

- 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 approved mode configuration after the installation. Execute the `fips-mode-setup --enable` command. Restart the system.

In both cases, the Crypto Officer must verify the Oracle Linux 9 system operates in the approved mode by executing the `fips-mode-setup --check` command, which should output “FIPS mode is enabled.”

For more information on Oracle Linux 9 system approved mode, please see Oracle Linux 9 product [documentation](#).

After installation of the `nss-softokn-3.90.0-3.0.1.el9_2_fips` and `nss-softokn-freebl-3.90.0-3.0.1.el9_2_fips` RPM packages, the Crypto Officer must execute the “Show Version” service by accessing the `CKA_NSS_VALIDATION_MODULE_ID` attribute of the `CKO_NSS_VALIDATION` object in the default slot. The object attribute must contain the value:

Oracle Linux 9 NSS Cryptographic Module 4.35.0-381552536e763d0c

Alternatively, the `/usr/lib64/nss/unsupported-tools/validation` tool is provided as a convenience by the `nss-tools-3.90.0-3.0.1.el9_2` RPM package. This tool performs the same steps, and outputs the FIPS module identifier as above. The cryptographic boundary consists only of the Softoken and Freebl libraries along with their associated integrity check values as listed in Table 2. If any other NSS API outside of these two libraries is invoked, the user is not interacting with the module specified in this Security Policy.

11.2 Administrator Guidance

See section 2.7 for algorithm-specific information.

11.3 Non-Administrator Guidance

There is no non-administrator guidance.

11.4 Maintenance Requirements

There are no maintenance requirements.

11.5 End of Life

As the module does not persistently store SSPs, secure sanitization of the module consists of unloading the module. This will zeroize all SSPs in volatile memory. Then, if desired, the `nss-softokn-3.90.0-3.0.1.el9_2_fips` and `nss-softokn-freebl-3.90.0-3.0.1.el9_2_fips` RPM packages can be uninstalled from the Oracle Linux 9 system.

12 Mitigation of Other Attacks

Timing attacks on RSA:

RSA blinding: Timing attack on RSA was first demonstrated by Paul Kocher in 1996, who contributed the mitigation code to our module. Most recently Boneh and Brumley showed that RSA blinding is an effective defense against timing attacks on RSA.

Specific Limit: None

Cache-timing attacks on the modular exponentiation operation used in RSA:

Cache invariant modular exponentiation: This is a variant of a modular exponentiation implementation that Colin Percival showed to defend against cache-timing attacks.

Specific Limit: This mechanism requires intimate knowledge of the cache line sizes of the processor. The mechanism may be ineffective when the module is running on a processor whose cache line sizes are unknown.

Arithmetic errors in RSA signatures:

Double-checking RSA signatures: Arithmetic errors in RSA signatures might leak the private key. Ferguson and Schneier recommend that every RSA signature generation should verify the signature just generated.

Specific Limit: None

13 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 |
| CTS | Ciphertext Stealing |
| DH | Diffie-Hellman |
| DRBG | Deterministic Random Bit Generator |
| DSA | Digital Signature Algorithm |
| ECB | Electronic Code Book |
| ECC | Elliptic Curve Cryptography |
| ECDH | Elliptic Curve Diffie-Hellman |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| ENT (NP) | Non-physical Entropy Source |
| FFC | Finite Field Cryptography |
| FIPS | Federal Information Processing Standards |
| GCM | Galois Counter Mode |
| HKDF | HMAC-based Key Derivation Function |
| HMAC | Keyed-Hash Message Authentication Code |
| KAT | Known Answer Test |
| KBKDF | Key-based Key Derivation Function |
| 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 |
| SSC | Shared Secret Computation |
| SSP | Sensitive Security Parameter |
| TOEPP | Tested Operational Environment's Physical Perimeter |

14 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 |
| PKCS#1 | Public Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 February 2003 https://www.ietf.org/rfc/rfc3447.txt |
| RFC 3526 | More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE) May 2003 https://www.ietf.org/rfc/rfc3526.txt |
| RFC 5288 | AES Galois Counter Mode (GCM) Cipher Suites for TLS August 2008 https://www.ietf.org/rfc/rfc5288.txt |
| RFC 7919 | Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS) August 2016 https://www.ietf.org/rfc/rfc7919.txt |
| RFC 8446 | The Transport Layer Security (TLS) Protocol Version 1.3 August 2018 https://www.ietf.org/rfc/rfc8446.txt |
| 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 800-38A Addendum | Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode October 2010 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a-add.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 800-52r2 | Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations August 2019 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.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 800-56Cr1 | Recommendation for Key-Derivation Methods in Key-Establishment Schemes August 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr1.pdf |
| SP 800-56Cr2 | Recommendation for Key-Derivation Methods in Key-Establishment Schemes August 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.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 800-131Ar2 | Transitioning the Use of Cryptographic Algorithms and Key Lengths March 2019 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.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 |
| SP 800-140B | CMVP Security Policy Requirements March 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-140B.pdf |