



**SUSE Linux Enterprise Kernel Crypto API  
Cryptographic Module**

**version 3.3 and 3.4**

**FIPS 140-3 Non-Proprietary Security Policy**

**Version 1.3**

**Last update: 2024-07-03**

Prepared by:

atsec information security corporation

4516 Seton Center Parkway, Suite 250

Austin, TX 78759

[www.atsec.com](http://www.atsec.com)

# 1 Table of Contents

---

<b>1</b>	<b>General</b> .....	<b>4</b>
<b>2</b>	<b>Cryptographic Module Specification</b> .....	<b>5</b>
2.1	Module Embodiment .....	5
2.2	Module Design, Components, versions .....	5
2.2.1	Module Name and Module Version Mapping.....	5
2.2.2	Module Components.....	5
2.3	Tested Operational Environments.....	7
2.4	Vendor-Affirmed Operational Environments .....	7
2.4.1	Version 3.3 Vendor Affirmed Operational Environments .....	9
2.4.2	Version 3.4 Vendor Affirmed Operational Environments .....	9
2.5	Modes of Operation of the Module .....	10
2.6	Security Functions .....	10
2.6.1	Approved Algorithms.....	10
2.6.2	Non-Approved Algorithms Allowed in the Approved Mode of Operation.....	18
2.6.3	Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed .....	18
2.6.4	Non-Approved Algorithms Not Allowed in the Approved Mode of Operation .....	18
<b>3</b>	<b>Cryptographic Module Ports and Interfaces</b> .....	<b>20</b>
<b>4</b>	<b>Roles, services, and authentication</b> .....	<b>21</b>
4.1	Roles .....	21
4.2	Authentication .....	22
4.3	Services .....	22
4.3.1	Service Indicator.....	22
4.3.2	Approved Services.....	22
4.3.3	Non-Approved Services .....	25
<b>5</b>	<b>Software/Firmware security</b> .....	<b>27</b>
5.1	Integrity Techniques .....	27
5.2	On-Demand Integrity Test.....	27
5.3	Executable Code .....	27
<b>6</b>	<b>Operational Environment</b> .....	<b>28</b>
6.1	Applicability .....	28
6.2	Policy .....	28
6.3	Requirements .....	28
<b>7</b>	<b>Physical Security</b> .....	<b>29</b>
<b>8</b>	<b>Non-invasive Security</b> .....	<b>30</b>

<b>9</b>	<b>Sensitive Security Parameter Management</b>	<b>31</b>
9.1	Random Number Generation	35
9.2	SSP Generation	35
9.3	Key Agreement	36
9.4	Key Transport	36
9.5	SSP Entry and Output	36
9.6	SSP Storage	36
9.7	SSP Zeroization	36
<b>10</b>	<b>Self-tests</b>	<b>37</b>
10.1	Pre-Operational Tests	37
10.2	Conditional Tests	37
10.2.1	Pairwise Consistency Test	38
10.3	Periodic/On-Demand Self-Test	38
10.4	Error States	38
<b>11</b>	<b>Life-cycle assurance</b>	<b>39</b>
11.1	Delivery and Operation	39
11.1.1	Module Installation	39
11.1.2	Operating Environment Configuration	39
11.1.3	Crypto Officer Guidance for Vendor Affirmed Operational Environments	39
11.2	Crypto Officer Guidance	40
11.2.1	AES XTS	40
11.2.2	AES GCM IV	40
11.2.3	Handling Self-Test Errors	41
11.2.4	SP 800-56Ar3 Assurances	41
11.2.5	End of Life Procedure	41
<b>12</b>	<b>Mitigation of other attacks</b>	<b>42</b>

# 1 General

This document is the non-proprietary FIPS 140-3 Security Policy for version 3.3 and 3.4 of the SUSE Linux Enterprise Kernel Crypto API Cryptographic Module. It has a one-to-one mapping to the [SP 800-140B] starting with section B.2.1 named “General” that maps to section 1 in this document and ending with section B.2.12 named “Mitigation of other attacks” that maps to section 12 in this document.

<b>ISO/IEC 24759 Section 6. [Number Below]</b>	<b>FIPS 140-3 Section Title</b>	<b>Security Level</b>
1	General	1
2	Cryptographic Module Specification	1
3	Cryptographic Module Interfaces	1
4	Roles, Services, and Authentication	1
5	Software/Firmware Security	1
6	Operational Environment	1
7	Physical Security	N/A
8	Non-invasive Security	N/A
9	Sensitive Security Parameter Management	1
10	Self-tests	1
11	Life-cycle Assurance	1
12	Mitigation of Other Attacks	N/A

*Table 1 - Security Levels*

## 2 Cryptographic Module Specification

### 2.1 Module Embodiment

The SUSE Linux Enterprise Kernel Crypto API Cryptographic Module (hereafter referred to as “the module”) is a Software multi-chip standalone cryptographic module.

### 2.2 Module Design, Components, versions

The software block diagram below shows the cryptographic boundary of the module, and its interfaces with the operational environment.

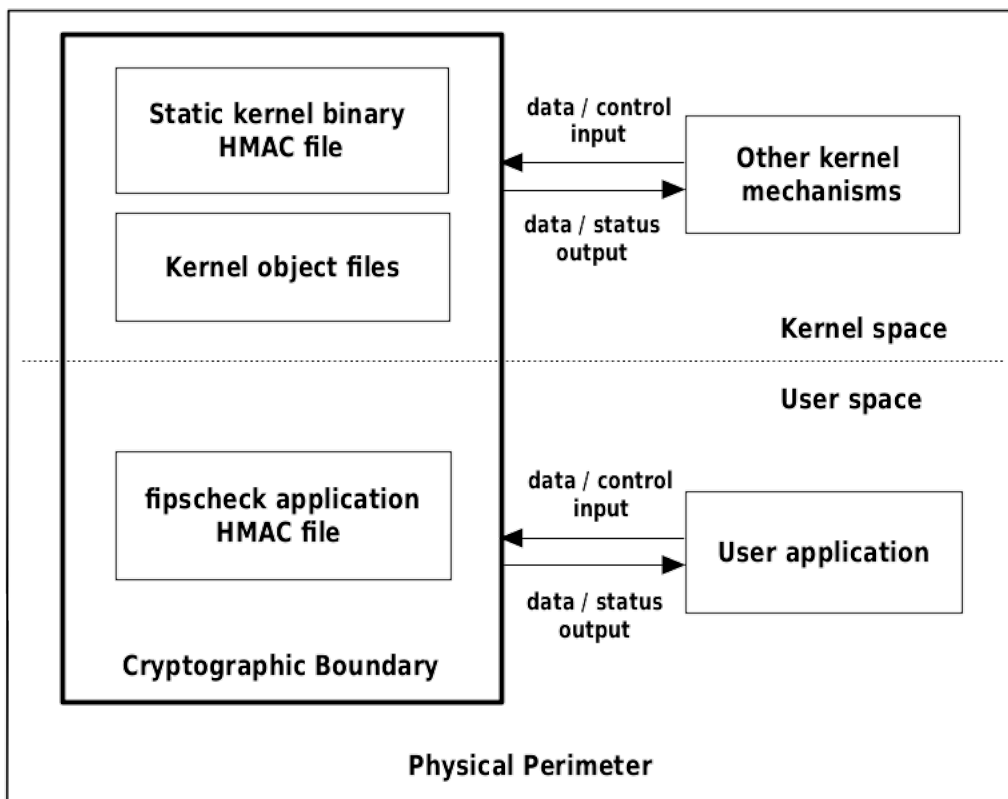


Figure 1 - Cryptographic Boundary

#### 2.2.1 Module Name and Module Version Mapping

The output of “uname -r” command will return the module identifier and version number. This command returns either “5.14.21-150400.24.46-default” that maps to module version 3.3 or “5.14.21-150400.15.11-rt” that maps to module version 3.4.

#### 2.2.2 Module Components

Table 2 lists the software components of the cryptographic module, which defines its cryptographic boundary.

Components	Description
<p><b>Version 3.3</b></p> <p>/boot/Image-5.14.21-150400.24.46-default (For ARM Ampere Altra platform)  /boot/vmlinux-5.14.21-150400.24.46-default (For IBM Power platform)  /boot/image-5.14.21-150400.24.46-default (For IBM z/15 platform)  /boot/vmlinux-5.14.21-150400.24.46-default (For Intel Xeon and AMD EPYC platforms)</p>	<p>Static kernel binary.</p>
<p><b>Version 3.4</b></p> <p>/boot/vmlinux-5.14.21-150400.15.11-rt (For Intel Xeon and AMD EPYC platforms)</p>	
<p><b>Version 3.3</b></p> <p>/boot/.vmlinuz-5.14.21-150400.24.46-default.hmac /boot/.vmlinuz-5.14.21-150400.24.46-default.hmac (For Intel Xeon and AMD EPYC platforms)  /boot/.Image-5.14.21-150400.24.46-default.hmac (For ARM Ampere Altra platform)  /boot/.vmlinux-5.14.21-150400.24.46-default.hmac (For IBM Power platform)  /boot/.image-5.14.21-150400.24.46-default.hmac (For IBM z/15 platform)</p>	<p>Integrity check HMAC value for Linux kernel static binary (HMAC file).</p>
<p><b>Version 3.4</b></p> <p>/boot/.vmlinuz-5.14.21-150400.15.11-rt.hmac (For Intel Xeon and AMD EPYC platforms)</p>	
<p><b>Version 3.3</b></p> <p>/lib/modules/5.14.21-150400.24.46-default/kernel/crypto/*.ko  /lib/modules/5.14.21-150400.24.46-default/kernel/arch/x86/crypto/ *.ko (For Intel Xeon and AMD EPYC platforms)  /lib/modules/5.14.21-150400.24.46-default/kernel/arch/arm64/ crypto/*.ko (For ARM Ampere Altra platform)  /lib/modules/5.14.21-150400.24.46-default/kernel/arch/powerpc/ crypto/*.ko (For IBM Power platform)  /lib/modules/5.14.21-150400.24.46-default/kernel/arch/s390/crypto/*.ko (For IBM z/15 platform)</p> <p><b>Version 3.4</b></p> <p>/lib/modules/5.14.21-150400.15.11-rt/kernel/crypto/*.ko (For Intel Xeon and AMD EPYC platforms)  /lib/modules/5.14.21-150400.15.11-rt/kernel/arch/x86/crypto/ *.ko (For Intel Xeon and AMD EPYC platforms)</p>	<p>Cryptographic kernel object files.</p>

Components	Description
/usr/lib64/libkcapifipscheck	Integrity test utility (fipscheck application).
/usr/lib64/libkcapifipscheck.hmac	Integrity check HMAC file for integrity test utility (HMAC file).

Table 2 - Cryptographic Module Components

## 2.3 Tested Operational Environments

The module has been tested on the following platforms with the corresponding module variants and configuration options:

#	Operating System	Hardware Platform	Processor	PAA/Acceleration	Module Version Tested
1	SUSE Linux Enterprise Server 15 SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)	3.3 and 3.4
2	SUSE Linux Enterprise Server 15 SP4	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)	3.3 and 3.4
3	SUSE Linux Enterprise Server 15 SP4	GIGABYTE G242-P32-QZ	ARM Ampere® Altra® Q80-30	With and without Cryptography Extensions (PAA)	3.3
4	SUSE Linux Enterprise Server 15 SP4	IBM z/15	z15	With and without CPACF (PAI)	3.3
5	SUSE Linux Enterprise Server 15 SP4 on PowerVM (VIOS 3.1.4.00)	IBM Power E1080 (9080-HEX)	Power10	With and without ISA (PAA)	3.3

Table 3 - Tested Operational Environments

## 2.4 Vendor-Affirmed Operational Environments

In addition to the platforms listed in Table 3, SUSE has also tested the module on the platforms shown in Table 4 and Table 5, and claims vendor affirmation on them.

Note: the 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.





### 2.4.1 Version 3.3 Vendor Affirmed Operational Environments

#	Operating system	Hardware platform	Processor	PAA/Acceleration
1	SUSE Linux Enterprise Server 15SP4	IBM LinuxONE III LT1	z15	With and without CPACF (PAI)
2	SUSE Linux Enterprise Micro 5.3	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)
3	SUSE Linux Enterprise Micro 5.3	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)
4	SUSE Linux Enterprise Micro 5.3	GIGABYTE G242-P32-QZ	ARM Ampere® Altra® Q80-30	With and without Cryptography Extensions (PAA)
5	SUSE Linux Enterprise Micro 5.3	IBM z/15	z15	With and without CPACF (PAI)
6	SUSE Linux Enterprise Micro 5.3	IBM LinuxONE III LT1	z15	With and without CPACF (PAI)
7	SUSE Linux Enterprise Server for SAP 15SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)
8	SUSE Linux Enterprise Server for SAP 15SP4	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)
9	SUSE Linux Enterprise Server for SAP 15SP4 on PowerVM (VIOS 3.1.4.00)	IBM Power E1080 (9080-HEX)	Power10	With and without ISA (PAA)
10	SUSE Linux Enterprise Desktop 15SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)
11	SUSE Linux Enterprise Desktop 15SP4	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)

Table 4 - Vendor Affirmed Operational Environments for version 3.3

### 2.4.2 Version 3.4 Vendor Affirmed Operational Environments

#	Operating System	Hardware platform	Processor	PAA/Acceleration
1	SUSE Linux Enterprise Micro 5.3	Supermicro Super Server	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)

#	Operating System	Hardware platform	Processor	PAA/Acceleration
		SYS-6019P-WTR		
2	SUSE Linux Enterprise Micro 5.3	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)
3	SUSE Linux Enterprise Real Time 15SP4	Supermicro Super Server SYS-6019P-WTR	Intel® Xeon® Silver 4215R	With and without AES-NI (PAA)
4	SUSE Linux Enterprise Real Time 15SP4	GIGABYTE R181-Z90-00	AMD EPYC™ 7371	With and without AES-NI (PAA)

Table 5 - Vendor Affirmed Operational Environments for version 3.4

## 2.5 Modes of Operation of the Module

After following the instructions for installation and configuration provided in section 11 the module is pre-configured to be operating in the Approved mode. When the module starts up successfully, after passing all the pre-operational self-test and conditional cryptographic algorithm self-tests (CASTs), the module is operating in the Approved mode of operation. Please see section 4 for the details on service indicator provided by the module that identifies when an approved service is called. If any service from the non-approved services list is called, the module transitions to non-approved mode automatically.

## 2.6 Security Functions

### 2.6.1 Approved Algorithms

Table 6 below lists all security functions of the module, including specific key strengths employed for approved services.

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3061</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3131</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3106</a> , <a href="#">A3109</a>	AES FIPS197, SP800-38A	CBC	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3050</a> , <a href="#">A3059</a> , <a href="#">A3070</a> , <a href="#">A3084</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3095</a> , <a href="#">A3104</a> , <a href="#">A3115</a>	AES SP800-38A- addendum	CBC-CS3	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3109</a>	AES SP800-38C	CCM	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption
Version 3.3: <a href="#">A3048</a> , <a href="#">A3057</a> , <a href="#">A3068</a> , <a href="#">A3082</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3093</a> , <a href="#">A3102</a> , <a href="#">A3113</a>	AES FIPS197, SP800-38A	CFB128	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3109</a>	AES SP800-38B	CMAC	128, 192, 256-bit keys with 128-256 bits of security strength	Message authentication code (MAC)
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3061</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3131</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3106</a> , <a href="#">A3109</a>	AES FIPS197, SP800-38A	CTR	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3075</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3129</a> , <a href="#">A3130</a> , <a href="#">A3131</a>	AES FIPS197, SP800-38A	ECB	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption
Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a>				
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3061</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3128</a>	AES SP800-38D	GCM with external IV	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric decryption
Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3106</a> , <a href="#">A3109</a>				
Version 3.3: <a href="#">A3051</a> , <a href="#">A3055</a> , <a href="#">A3062</a> , <a href="#">A3065</a> , <a href="#">A3076</a> , <a href="#">A3079</a> , <a href="#">A3129</a>	AES SP800-38D RFC4106	GCM with internal IV (RFC4106) (IV Gen Mode 8.2.1)	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption
Version 3.4: <a href="#">A3096</a> , <a href="#">A3100</a> , <a href="#">A3107</a> , <a href="#">A3110</a>				
Version 3.3: <a href="#">A3052</a> , <a href="#">A3056</a> , <a href="#">A3063</a> , <a href="#">A3066</a> , <a href="#">A3077</a> , <a href="#">A3080</a> , <a href="#">A3130</a>	AES SP800-38D RFC4106	GCM with external IV (RFC4106)	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric decryption
Version 3.4: <a href="#">A3097</a> , <a href="#">A3101</a> , <a href="#">A3108</a> , <a href="#">A3111</a>				

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3046</a> , <a href="#">A3054</a> , <a href="#">A3067</a> , <a href="#">A3081</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3091</a> , <a href="#">A3099</a> , <a href="#">A3112</a>	AES SP800-38F	KW	128, 192, 256-bit keys with 128-256 bits of security strength	Key wrapping; Key unwrapping
Version 3.3: <a href="#">A3049</a> , <a href="#">A3058</a> , <a href="#">A3069</a> , <a href="#">A3083</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3094</a> , <a href="#">A3103</a> , <a href="#">A3114</a>	AES SP800-38A	OFB	128, 192, 256-bit keys with 128-256 bits of security strength	Symmetric encryption; Symmetric decryption
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3061</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3131</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3106</a> , <a href="#">A3109</a>	AES SP800-38E	XTS	128, 256-bit keys with 128 and 256-bits of security strength	Symmetric encryption and Symmetric decryption (for data storage)
Vendor Affirmed	CKG SP800-133rev2	FIPS 186-4	EC: P-256, P384 keys with 128 and 192 bits of security strength; Safe Primes Key Generation with 2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of security strength	Key generation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3075</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3128</a> , <a href="#">A3129</a> , <a href="#">A3130</a>	DRBG SP800- 90Arev1	CTR_DRBG: AES-128, AES-192, AES-256 with DF, with/without PR	128, 192, 256-bit keys with 128, 192 and 256 bits of security strength	Deterministic random bit generation
Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a>		Hash_DRBG: SHA-1, SHA2-256, SHA2-384, SHA2-512 with/without PR	N/A	
Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3129</a> , <a href="#">A3130</a>		HMAC_DRBG: SHA-1, SHA2-256, SHA2-384, SHA2-512 with/without PR	≥ 112-bit keys with 112-256 bits of security strength	
Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a>				
Version 3.3: <a href="#">A3044</a> Version 3.4: <a href="#">A3089</a>	ECDSA FIPS186-4	B.4.2 Testing candidates	P-256, P-384 keys with 128 and 192 bits of security strength	EC Diffie-Hellman key generation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">E19</a> Version 3.4: <a href="#">E20</a>	ESV SP800-90B	CPU Jitter Source	N/A	Random number generation
Version 3.3: <a href="#">A3086</a>	HMAC FIPS198-1	SHA-1	≥ 112-bit keys with 112-256 bits of security strength	Message authentication code (MAC) Integrity Test (with SHA2-256)
Version 3.3: <a href="#">A3074</a> , <a href="#">A3087</a> , <a href="#">A3125</a> Version 3.4: <a href="#">A3119</a>		SHA-1, SHA2-224, SHA2-256		
Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3078</a> Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a>		SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512		
Version 3.3: <a href="#">A3131</a>		SHA2-224, SHA2-256		
Version 3.3 <a href="#">A3132</a>		SHA2-224, SHA2-256, SHA2-384, SHA2-512		
Version 3.3 <a href="#">A3127</a>		SHA2-384, SHA2-512		
Version 3.3: <a href="#">A3047</a> , <a href="#">A3085</a> , <a href="#">A3126</a> Version 3.4: <a href="#">A3092</a>		SHA3-224, SHA3-256, SHA3-384, SHA3-512		
Version 3.3: <a href="#">A3044</a> Version 3.4: <a href="#">A3089</a>		KAS ECC- SSC SP800- 56Arev3 IG D.F 2 (1)		
Version 3.3: <a href="#">A3043</a> Version 3.4: <a href="#">A3088</a>	KAS FCC- SSC SP800- 56Arev3 IG D.F 2 (1)	dhEphem with safe prime groups	2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of security strength	Diffie-Hellman shared secret computation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3109</a>	KTS SP800-38F	AES-CCM	128, 192, 256-bit keys with 128, 192 and 256 bits of security strength	Key Wrapping and Unwrapping
Version 3.3: <a href="#">A3051</a> , <a href="#">A3055</a> , <a href="#">A3062</a> , <a href="#">A3065</a> , <a href="#">A3076</a> , <a href="#">A3079</a> , <a href="#">A3129</a> Version 3.4: <a href="#">A3096</a> , <a href="#">A3100</a> , <a href="#">A3107</a> , <a href="#">A3110</a>		AES-GCM	128, 192, 256-bit keys with 128, 192 and 256 bits of security strength	
Version 3.3: <a href="#">A3046</a> , <a href="#">A3054</a> , <a href="#">A3067</a> , <a href="#">A3081</a> , <a href="#">A3128</a> Version 3.4: <a href="#">A3091</a> , <a href="#">A3099</a> , <a href="#">A3112</a>		AES-KW	128, 192, 256-bit keys with 128-256 bits of security strength	



CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
(AES) Version 3.3: <a href="#">A3045</a> , <a href="#">A3053</a> , <a href="#">A3061</a> , <a href="#">A3064</a> , <a href="#">A3075</a> , <a href="#">A3078</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3131</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3098</a> , <a href="#">A3106</a> , <a href="#">A3109</a>	KTS SP800-38F FIPS140-3 IG D.G	AES-CBC and HMAC- SHA-1 or HMAC-SHA2	128, 256-bit keys with 128, 256 bits of security strength	
(HMAC) Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3074</a> , <a href="#">A3078</a> , <a href="#">A3086</a> , <a href="#">A3087</a> , <a href="#">A3125</a> , <a href="#">A3131</a> , <a href="#">A3132</a> , <a href="#">A3127</a> Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a> , <a href="#">A3119</a>				
Version 3.3: <a href="#">A3045</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3078</a> Version 3.4: <a href="#">A3090</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a>	RSA FIPS186-4	PKCS#1v1.5: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	2048, 3072, 4096- bit keys with 112- 149 bits of security strength	Integrity test using digital signature verification  (usage of SHA-1 is considered Legacy Use)
Version 3.3: <a href="#">A3043</a> Version 3.4: <a href="#">A3088</a>	Safe Primes Key Generation	Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP- 3072, MODP-4096, MODP-6144, MODP- 8192	2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of security strength	Diffie-Hellman shared secret computation
Version 3.3: <a href="#">A3047</a> , <a href="#">A3085</a> , <a href="#">A3126</a> Version 3.4: <a href="#">A3092</a>	SHA-3 FIPS202	SHA3-224, SHA3-256, SHA3-384, SHA3-512	N/A	Message Digest
Version 3.3: <a href="#">A3086</a>	SHS FIPS180-4	SHA-1	N/A	Message digest

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use
Version 3.3: <a href="#">A3074</a> , <a href="#">A3087</a> , <a href="#">A3125</a> Version 3.4: <a href="#">A3119</a>		SHA-1, SHA2-224, SHA2-256		
Version 3.3: <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3078</a> Version 3.4: <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a>		SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512		
Version 3.3: <a href="#">A3131</a>		SHA2-224, SHA2-256		
Version 3.3 <a href="#">A3132</a>		SHA2-224, SHA2-256, SHA2-384, SHA2-512		
Version 3.3 <a href="#">A3127</a>		SHA2-384, SHA2-512		

Table 6 - Approved Algorithms

## 2.6.2 Non-Approved Algorithms Allowed in the Approved Mode of Operation

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

## 2.6.3 Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed

The module does not implement non-approved algorithms allowed in the approved mode of operation with no security claimed..

## 2.6.4 Non-Approved Algorithms Not Allowed in the Approved Mode of Operation

Table 7 lists non-approved algorithms that are not allowed in the approved mode of operation. These algorithms are used by the non-approved services listed in Table 11.

Algorithm/Functions	Use/Function
AES-GCM with external IV	Symmetric encryption
AES-GMAC	Message authentication code (MAC)

<b>Algorithm/Functions</b>	<b>Use/Function</b>
RSA	RSA encryption primitive; RSA decryption primitive; RSA signature generation primitive; RSA signature verification primitive
ECDSA	ECDSA signature verification primitive

*Table 7 - Non-Approved Algorithms Not Allowed in the Approved Mode of Operation*

### 3 Cryptographic Module Ports and Interfaces

As a software-only module, the module does not have physical ports. The operator can only interact with the module through the API provided by the module. Thus, the physical ports are interpreted to be the physical ports of the hardware platform on which the module runs.

All data output via data output interface is inhibited when the module is performing pre-operational test or zeroization or when the module enters error state.

Logical Interface <sup>1</sup>	Data that passes over port/interface
Data Input	API data input parameters from kernel system calls, kernel command line
Data Output	API output parameters from kernel system calls
Control Input	API function calls, API control input parameters from kernel system calls, kernel command line
Status Output	API return values, kernel logs

*Table 8 - Ports and Interfaces*

---

<sup>1</sup> The control output interface is omitted on purpose because the module does not implement it.

## 4 Roles, services, and authentication

### 4.1 Roles

The module supports the Crypto Officer role only. This sole role is implicitly assumed by the operator of the module when performing a service. The module does not support authentication.

Role	Service	Input	Output
Crypto Officer (CO)	Symmetric encryption	Key, plaintext	Ciphertext
	Symmetric decryption	Key, ciphertext	Plaintext
	Random number generation	Number of bits	Random number
	Message digest	Message	Digest of the message
	Message authentication code (MAC)	Message, key	Message authentication code
	Encrypt-then-MAC operation	Message, AES Key, HMAC key	Authenticated message
	RSA encryption primitive	Key, plaintext	Ciphertext
	RSA decryption primitive	Key, ciphertext	Plaintext
	RSA signature generation primitive	Key, hashed message	Signature
	RSA signature verification primitive	Key, signature	Hashed message
	Key wrapping	Key wrapping key, key to be wrapped	Wrapped key
	Key unwrapping	Wrapped key, key unwrapping key	Unwrapped key
	EC Diffie-Hellman shared secret computation	Private key, public key from peer	Shared secret
	Diffie-Hellman shared secret computation	Private key, public key from peer	Shared secret
	EC Diffie-Hellman key generation	Domain parameters	Generated key
	Diffie-Hellman key generation using safe primes	Domain parameters	Generated key
	Error detection code	None	Code
	Data compression	Data to compress	Compressed data
	Memory copy operation	Source, destination, offset, amount	Return codes and/or log messages
	Show status	None	Return codes and/or log messages
Zeroization	Context containing SSPs	None	
Self-tests	None	Return codes and/or log	

Role	Service	Input	Output
	Digital signature verification	Message, hash algorithm, public key	Verification result
	ECDSA digital signature verification primitive	Key, signature	Hashed message
	Module installation and configuration	Configuration parameters	Return codes and/or log
	Module initialization	None	None
	Show module name and version	None	Name and version information

Table 9 - Roles, Service Commands, Input and Output

## 4.2 Authentication

The module does not support authentication.

## 4.3 Services

The module provides services to the users that assume one of the available roles. All services are shown in Table 10 and Table 11.

### 4.3.1 Service Indicator

The module provides an approved service indicator as specified in the “Indicator” column in Table 10.

### 4.3.2 Approved Services

Table 10 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). No support of intermediate key generation is provided. The following convention is used to specify access rights to a CSP:

- **G = Generate:** The module generates or derives the SSP.
- **R = Read:** The SSP is read from the module (e.g., the SSP is output).
- **W = Write:** The SSP is updated, imported, or written to the module.
- **E = Execute:** The module uses the SSP in performing a cryptographic operation.
- **Z = Zeroize:** 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 Table 6. Having the module configured properly in the approved mode as detailed in section 11, restricts the Crypto Officer to only access the approved services listed in Table 10.

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
<b>Cryptographic Services</b>						
Symmetric encryption; Symmetric decryption	Perform AES encryption and decryption	AES-CBC, AES-CBS-CS3, AES-CFB128, AES-CTR, AES-ECB, AES-OFB, AES-XTS	AES key	CO	W, E	crypto_skcipher_setkey returns 0
		AES-CCM				crypto_aead_setkey returns 0
		AES-GCM				crypto_tfm_get_flags has the CRYPTO_TFM_FIPS_COMPLIANCE flag set
Random number generation	Generate random numbers	DRBG	Entropy input		W, E G, E	crypto_rng_get_bytes returns 0
			DRBG seed, DRBG internal state			
Message digest	Compute SHA hashes	SHA-1, SHA2, SHA3	None		N/A	crypto_shash_init returns 0
Message authentication code (MAC)	Compute HMAC	HMAC	HMAC key		W, E	crypto_shash_init returns 0
	Compute AES-based CMAC	CMAC with AES	AES key		W, E	crypto_cmac_digest_init returns 0
Encrypt-then-MAC operation	Perform AES encryption then compute MAC	AES-CBC, HMAC- [SHA-1, SHA2]	AES key, HMAC key		W, E	crypto_shash_init returns 0
Key wrapping	Perform AES-based key wrapping	AES-KW, AES-CCM, AES-GCM	AES key		W, E	crypto_skcipher_setkey returns 0
	Perform AES-based key wrapping and HMAC	AES-CBC and HMAC	AES key, HMAC key		W, E	crypto_skcipher_setkey returns 0; crypto_shash_init returns 0
Key unwrapping	Perform AES-based key unwrapping	AES-KW, AES-CCM, AES-GCM	AES key		W, E	crypto_skcipher_setkey returns 0
	Perform AES-based key unwrapping and HMAC	AES-CBC and HMAC	AES key, HMAC key		W, E	crypto_skcipher_setkey returns 0; crypto_shash_init returns 0

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
EC Diffie-Hellman shared secret computation	Perform ECDH shared secret computation	EC Diffie-Hellman	EC Diffie-Hellman public, EC Diffie-Hellman private key		W, E	crypto_kpp_compute_shared_secret returns 0
			EC Diffie-Hellman shared secret		G, R	
Key generation for EC Diffie-Hellman shared secret computation <sup>2</sup>	Perform ECDH key generation	Generation per Section 5.6.1.2 of SP800-56Arev3 exclusively for EC Diffie-Hellman	Module-generated EC Diffie-Hellman public key, Module-generated EC Diffie-Hellman private key		E, G, R	crypto_kpp_set_secret and crypto_kpp_generate_public_key return 0
Diffie-Hellman shared secret computation	Perform DH shared secret computation	Diffie-Hellman	Diffie-Hellman public key, Diffie-Hellman private key		W, E	crypto_kpp_compute_shared_secret returns 0
			Diffie-Hellman shared secret		G, R	
Diffie-Hellman key generation using safe primes	Perform DH key generation	Generation per Section 5.6.1.1 of SP800-56Arev3 exclusively for Diffie-Hellman	Module-generated Diffie-Hellman public key, Module-generated private key		E, G, R	crypto_kpp_set_secret and crypto_kpp_generate_public_key return 0
<b>Other FIPS-related Services</b>						
Error detection code	Detect, report, correct memory errors with crc32c <sup>3</sup> , crct10dif <sup>3</sup>	N/A	None	CO	N/A	Implicit (always approved)
Data compression	Compress data with deflate <sup>3</sup> , lz4 <sup>3</sup> , lz4hc <sup>3</sup> , lzo <sup>3</sup> , zlib <sup>3</sup> , 842 <sup>3</sup>	N/A	None		N/A	Implicit (always approved)

<sup>2</sup> The module performs a pairwise consistency test for EC Diffie-Hellman as outlined in section 5.6.2.1.4 of SP 800-56Arev3. This key generation service is for exclusive use of the EC Diffie-Hellman shared secret computation service.

<sup>3</sup> This algorithm does not provide any cryptographic attribute, i.e., its purpose in the module is not security relevant



Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
Memory copy operation	Copy memory	N/A	None		N/A	Implicit (always approved)
Show status	Show module status	N/A	None		N/A	Implicit (always approved)
On-Demand Self-tests	Perform self-tests on demand	AES, CMAC, SHS, SHA-3, HMAC, DRBG, EC Diffie-Hellman, Diffie-Hellman, RSA, DRBG	None		N/A	Implicit (always approved)
Module installation and configuration	Install and configure module	N/A	None		N/A	Implicit (always approved)
Module initialization	Initialize module	N/A	None		N/A	Implicit (always approved)
Show status	Show module status	N/A	None		N/A	Implicit (always approved)
Zeroization	Zeroize SSPs	N/A	All SSPs		Z	Implicit (always approved)
Show module name and version	Show module name and version	N/A	None		N/A	Implicit (always approved)

Table 10 - Approved Services

### 4.3.3 Non-Approved Services

Table 11 lists the non-approved services. The details of the non-approved cryptographic algorithms available in non-approved mode can be found in Table 7.

Service	Description	Algorithms Accessed	Roles
<b>Cryptographic Services</b>			
Symmetric encryption	Perform AES-GCM encryption using external IV	AES	CO
Message authentication code (MAC)	Perform message authentication code (MAC) using AES-GMAC		
RSA encryption primitive	Compute RSA cipher	RSA	
RSA decryption primitive	Compute plaintext from RSA cipher		
RSA signature generation primitive	Sign using RSA		
RSA signature verification primitive	Verify RSA-based signatures		

Service	Description	Algorithms Accessed	Roles
ECDSA signature verification primitive	Verify ECDSA-based signatures	ECDSA	

*Table 11 - Non-Approved Services*

## 5 Software/Firmware security

### 5.1 Integrity Techniques

The module uses the HMAC-SHA2-256 algorithm for the integrity test of the static kernel binary and the fipscheck application. The HMAC calculation is performed by the fipscheck application itself. Additionally, the module uses RSA signature verification with a SHA2-256 message digest and a 2048-bit key for the integrity test of each of the kernel object files loaded during boot-up time. If the integrity values do not match the expected values, the test fails, and the module enters the error state.

### 5.2 On-Demand Integrity Test

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

The module provides the Self-Test service to perform self-tests on demand. This service performs the same cryptographic algorithm tests executed during power-up. Pre-Operational Self-Tests can also be invoked by calling the `kernel_restart()` which effectively powers off the module and then reloads the module. During the execution of the on-demand self-tests, services are not available, and no data output or input is possible.

### 5.3 Executable Code

The module consists of executable code in the form of the files listed in Table 2.

## 6 Operational Environment

### 6.1 Applicability

This module operates in a modifiable operational environment per the FIPS 140-3 level 1 specifications. The SUSE Linux Enterprise Server operating system is used as the basis of other products. Compliance is maintained for SUSE products whenever the binary is found unchanged per the vendor affirmation from SUSE based on the allowance FIPS 140-3 management manual [FIPS140-3\_MM] section 7.9.1 bullet 1 a i).

Note: The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when supported if the specific operational environment is not listed on the validation certificate.

### 6.2 Policy

Instrumentation tools like the `ptrace` system call, `gdb` and `strace` utilities, 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-tested operational environment.

### 6.3 Requirements

The module shall be installed as stated in section 11. 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.

## 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 Parameter Management

Table 12 summarizes the Sensitive Security Parameters (SSPs) that are used by the cryptographic services implemented in the module.

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import/Export	Establishment	Storage	Zeroization	Use & related keys
AES key	128, 192, 256	AES-CBC, AES-CBC-CS3, AES-CFB128, AES-CMAC, AES-CTR, AES-ECB, AES-CCM, AES-CMAC, AES-GCM, AES-KW, AES-OFB, AES-XTS, CTR_DRBG <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3049</a> , <a href="#">A3050</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3054</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3057</a> , <a href="#">A3058</a> , <a href="#">A3059</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3068</a> , <a href="#">A3069</a> , <a href="#">A3070</a> , <a href="#">A3075</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3082</a> , <a href="#">A3083</a> , <a href="#">A3084</a> , <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3093</a> , <a href="#">A3094</a> , <a href="#">A3095</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3099</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3102</a> , <a href="#">A3103</a> , <a href="#">A3104</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a> , <a href="#">A3113</a> , <a href="#">A3115</a> , <a href="#">A3125</a> , <a href="#">A3128</a> , <a href="#">A3129</a> , <a href="#">A3130</a> , <a href="#">A3131</a>	N/A	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Symmetric encryption; Symmetric decryption; Message authentication code (MAC) <b>Related keys:</b> N/A
HMAC key	112 to 256	HMAC <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3047</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3074</a> , <a href="#">A3078</a> , <a href="#">A3085</a> , <a href="#">A3086</a> , <a href="#">A3087</a> , <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3092</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a> , <a href="#">A3119</a> , <a href="#">A3125</a> , <a href="#">A3126</a> , <a href="#">A3127</a> , <a href="#">A3131</a> , <a href="#">A3132</a>	N/A	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Message authentication code (MAC) <b>Related keys:</b> N/A

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import/Export	Establishment	Storage	Zeroization	Use & related keys
Entropy input  IG D.L compliant	192 to 384	ESV <a href="#">E19</a> , <a href="#">E20</a>	Obtained from the SP800-90B entropy source	<b>Import/Export:</b> N/A; it remains within the cryptographic boundary.	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Random number generation <b>Related keys:</b> DRBG seed
DRBG seed  IG D.L compliant	192 to 384	CTR_DRBG <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3075</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a> , <a href="#">A3128</a> , <a href="#">A3129</a> , <a href="#">A3130</a>	Derived from the entropy input as defined in SP800-90Arev1	<b>Import/Export:</b> N/A; it remains within the cryptographic boundary.	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Random number generation <b>Related keys:</b> Entropy input, DRBG internal state
DRBG internal state (V, C or key)  IG D.L compliant	128 to 256	Hash_DRBG, HMAC_DRBG <a href="#">A3043</a> , <a href="#">A3044</a> , <a href="#">A3045</a> , <a href="#">A3051</a> , <a href="#">A3052</a> , <a href="#">A3053</a> , <a href="#">A3055</a> , <a href="#">A3056</a> , <a href="#">A3061</a> , <a href="#">A3062</a> , <a href="#">A3063</a> , <a href="#">A3064</a> , <a href="#">A3065</a> , <a href="#">A3066</a> , <a href="#">A3071</a> , <a href="#">A3072</a> , <a href="#">A3073</a> , <a href="#">A3076</a> , <a href="#">A3077</a> , <a href="#">A3078</a> , <a href="#">A3079</a> , <a href="#">A3080</a> , <a href="#">A3088</a> , <a href="#">A3089</a> , <a href="#">A3090</a> , <a href="#">A3096</a> , <a href="#">A3097</a> , <a href="#">A3098</a> , <a href="#">A3100</a> , <a href="#">A3101</a> , <a href="#">A3106</a> , <a href="#">A3107</a> , <a href="#">A3108</a> , <a href="#">A3109</a> , <a href="#">A3110</a> , <a href="#">A3111</a> , <a href="#">A3116</a> , <a href="#">A3117</a> , <a href="#">A3118</a> , <a href="#">A3129</a> , <a href="#">A3130</a>	Computed as defined in SP800-90Arev1	<b>Import/Export:</b> N/A; it remains within the cryptographic boundary	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Random number generation <b>Related keys:</b> DRBG seed
Module-generated EC Diffie-Hellman public key	128, 192	KAS ECC-SSC <a href="#">A3044</a> , <a href="#">A3089</a>	Generated using the FIPS 186-4 key generation method, random values are obtained from the SP800 90Arev1 DRBG	<b>Import:</b> N/A  <b>Export:</b> CM to TOEPP Path. Passed from the module via API parameters in plaintext (P) format.	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> EC Diffie-Hellman key generation <b>Related keys:</b> Module-generated EC Diffie-Hellman private key



Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import/Export	Establishment	Storage	Zeroization	Use & related keys
Module-generated EC Diffie-Hellman private key	128, 192	KAS ECC-SSC <a href="#">A3044</a> , <a href="#">A3089</a>	Generated using the FIPS 186-4 key generation method, random values are obtained from the SP800 90Arev1 DRBG	<b>Import:</b> N/A  <b>Export:</b> CM to TOEPP Path. Passed from the module via API parameters in plaintext (P) format.	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> EC Diffie-Hellman key generation, EC Diffie-Hellman shared secret computation  <b>Related keys:</b> Module-generated EC Diffie-Hellman public key
EC Diffie-Hellman public key	128, 192	KAS ECC-SSC <a href="#">A3044</a> , <a href="#">A3089</a>	N/A	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> EC Diffie-Hellman shared secret computation  <b>Related keys:</b> EC Diffie-Hellman shared secret
EC Diffie-Hellman private key	128, 192	KAS ECC-SSC <a href="#">A3044</a> , <a href="#">A3089</a>	N/A	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> EC Diffie-Hellman shared secret computation  <b>Related keys:</b> EC Diffie-Hellman shared secret
Module-generated Diffie-Hellman public key	128 to 200	KAS FFC-SSC <a href="#">A3043</a> , <a href="#">A3088</a>	Public and private keys are generating using the SP 800-56Arev3 Safe Primes key generation method, random values are obtained from the SP800-90Arev1 DRBG.	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Key generation  <b>Related keys:</b> Module-generated Diffie-Hellman shared secret

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import/Export	Establishment	Storage	Zeroization	Use & related keys
Module-generated Diffie-Hellman private key	128 to 200	KAS FFC-SSC <a href="#">A3043</a> , <a href="#">A3088</a>	Public and private keys are generating using the SP 800-56Arev3 Safe Primes key generation method, random values are obtained from the SP800-90Arev1 DRBG.	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Key generation  <b>Related keys:</b> Module-generated Diffie-Hellman shared secret
Diffie-Hellman public key	128 to 200	KAS FFC-SSC <a href="#">A3043</a> , <a href="#">A3088</a>	Public and private keys are generating using the SP 800-56Arev3 Safe Primes key generation method, random values are obtained from the SP800-90Arev1 DRBG.	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Diffie-Hellman shared secret computation  <b>Related keys:</b> Diffie-Hellman shared secret
Diffie-Hellman private key	128 to 200	KAS FFC-SSC <a href="#">A3043</a> , <a href="#">A3088</a>	Public and private keys are generating using the SP 800-56Arev3 Safe Primes key generation method, random values are obtained from the SP800-90Arev1 DRBG.	<b>Import:</b> CM from TOEPP Path. Passed to the module via API parameters in plaintext (P) format.  <b>Export:</b> N/A	N/A	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Diffie-Hellman shared secret computation  <b>Related keys:</b> Diffie-Hellman shared secret
EC Diffie-Hellman Shared secret	112 to 200	KAS ECC-SSC <a href="#">A3044</a> , <a href="#">A3089</a>	N/A	<b>Import:</b> N/A  <b>Export:</b> CM to TOEPP Path. Passed from the module via API parameters in plaintext (P) format.	Computed during the EC Diffie-Hellman shared secret computation according to SP 800-56Arev3.	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> EC Diffie-Hellman shared secret computation  <b>Related keys:</b> EC Diffie-Hellman public key, EC Diffie-

Key/SSP Name/Type	Strength	Security Function and Cert. Number	Generation	Import/Export	Establishment	Storage	Zeroization	Use & related keys
								Hellman private key
Diffie-Hellman Shared secret	112 to 200	KAS FFC-SSC <a href="#">A3043</a> , <a href="#">A3088</a>	N/A	<b>Import:</b> N/A  <b>Export:</b> CM to TOEPP Path. Passed from the module via API parameters in plaintext (P) format.	Computed during the Diffie Hellman shared secret computation according to SP 800-56Arev3.	RAM	Zeroized when freeing the cipher handler	<b>Use:</b> Diffie-Hellman shared secret computation  <b>Related keys:</b> Diffie-Hellman public key, Diffie-Hellman private key

Table 12 - SSPs

## 9.1 Random Number Generation

The module employs the Deterministic Random Bit Generator (DRBG) based on [SP800-90Arev1] for the random number generation. The DRBG supports the Hash\_DRBG, HMAC\_DRBG and CTR\_DRBG mechanisms. The module performs the DRBG health tests as defined in section 11.3 of [SP 800-90Arev1]. The module uses an SP800-90B compliant entropy source specified in Table 13. The entropy source is tested with RCT and APT Health tests as required by section 4 of [SP 800-90B]. The DRBG is seeded with (DRBG\_security\_strength \* 1.5) bits of random data from the CPU jitter RNG containing at least 256 bits of entropy. Therefore, the module ensures that during initialization (seed) and reseeding, the entropy source provides the required amount of entropy to meet the security strength of the respective DRBG methods.

The module uses the entropy source specified in Table 13.

Entropy Sources	Minimum number of bits of entropy	Details
ESV certs. <a href="#">version 3.3: E19</a> <a href="#">version 3.4: E20</a>	At least 256 bits of entropy in the 384-bit output	The CPU jitter is used as a SP800-90B-compliant entropy source.

Table 13 - Non-Deterministic Random Number Generation Specification

## 9.2 SSP Generation

The public and private keys used in the EC Diffie-Hellman shared secret computation are generated internally by the module using the ECDSA key generation method compliant with [FIPS186-4], [SP800-56Arev3] and section 4 and 5.1 of [SP800-133rev2]. The random value used in asymmetric key generation is directly obtained from the [SP800-90Arev1] DRBG. This key generation method is used exclusively by the EC Diffie-Hellman algorithm and provides support for the required assurances of [SP800-56Arev3]. The Diffie-Hellman shared secret computation is also compliant with [SP 800-56Arev3] and generates keys using safe primes defined in RFC7919 and RFC3526.

In accordance with FIPS140-3 IG D.H, the cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys per SP800-133rev2 section 4 and 5 (vendor affirmed).

## 9.3 Key Agreement

The module provides Diffie-Hellman and EC Diffie-Hellman shared secret computation compliant with [SP800-56Arev3] in accordance with scenario 2 (1) of IG D.F, CAVP-tested compliance with the derivation of a shared secret Z for Diffie-Hellman and EC Diffie-Hellman (section 6 of SP 800-56Arev3).

- Diffie-Hellman: Shared secret computation provides between 112 and 200 bits of encryption strength.
- EC Diffie-Hellman: Shared secret computation provides 128 or 192 bits of encryption strength.

## 9.4 Key Transport

The module provides approved key transport methods as permitted by IG D.G. These key transport methods are provided either by using an approved key wrapping algorithm, an approved authenticated encryption mode, or a combination method of approved symmetric encryption (AES) and HMAC-SHA-1/SHA-2.

Table 6 specifies the key sizes allowed in the approved mode of operation. According to “Table 2: Comparable strengths” in [SP800-57rev5], the key sizes of AES key wrapping provide the following security strengths:

- AES-KW key wrapping; key establishment methodology provides between 128 and 256 bits of encryption strength.
- AES-GCM and AES-CCM authenticated encryption key wrapping; key establishment methodology provides between 128 and 256 bits of encryption strength.
- Combination method of approved AES-CBC block mode and message authentication code (HMAC-SHA-1/SHA-2) key wrapping; key establishment methodology provides 128 or 256 bits of encryption strength.

## 9.5 SSP Entry and Output

The module does not support manual SSP entry 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. This is allowed by [FIPS140-3\_IG] 9.5.A, according to the “CM Software to/from App via TOEPP Path” entry on the Key Establishment Table.

## 9.6 SSP Storage

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.

## 9.7 SSP Zeroization

The application is responsible for calling the appropriate destruction functions from the Kernel Crypto API. When a calling application calls the appropriate API function, that function overwrites the memory with zeros and deallocates the memory when the cipher handler is freed. The completion of a zeroization and freeing the cipher handler routines will indicate that a zeroization procedure succeeded.

## 10 Self-tests

The module performs the pre-operational and conditional cryptographic algorithms self-tests automatically when the module is loaded into memory. These self-tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected. While the module is executing the pre-operational and the conditional cryptographic algorithms self-tests, services are not available, and input and output are inhibited. The module is not available for use by the calling application until the self-tests are completed successfully. If any of the self-tests fails, an error message is returned and the module transitions to error state.

### 10.1 Pre-Operational Tests

The module performs a pre-operational software integrity test automatically when the module is powered on before the module transitions into the operational state. The details on the integrity test are specified in section 5.1.

### 10.2 Conditional Tests

Table 14 lists the cryptographic algorithm self-tests (CASTs). The CASTs include the KATs for the integrity mechanism that is run prior to performing the integrity test. The details of the integrity test are provided in section 5.1.

Each KAT includes comparison of the calculated output with the expected known answer, hard coded as part of the test vectors used in the test. If the values do not match, the KAT fails. After the pre-operational and conditional cryptographic algorithms self-tests succeed, a success message is recorded in the dmesg log, and the module becomes operational.

Algorithm	Condition	Test
AES	Power up	KATs for AES in ECB, CBC, CTR, GCM, CCM and XTS modes using 128- 192- and 256-bit key size; encryption and decryption are performed separately.
CMAC	Power up	KATs AES CMAC with 128-, 192- and 256-bits keys, MAC generation.
SHS	Power up	KATs SHA-1, SHA2-224, SHA2-256, SHA2-384 and SHA2-512.
SHA-3	Power up	KATs SHA3-224, SHA3-256, SHA3-384 and SHA3-512.
HMAC	Power up	KATs HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC2-SHA-512. KATs HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512.
DRBG	Power up	KAT CTR_DRBG with AES with 128-bit key with DF, with and without PR. KAT CTR_DRBG with AES with 192, 256-bit key with DF, without PR. KAT Hash_DRBG with SHA-256 with and without PR. KAT HMAC_DRBG with SHA-256 with and without PR. KAT HMAC_DRBG with SHA-512 without PR.

Algorithm	Condition	Test
EC Diffie-Hellman	Power up	KAT EC Diffie-Hellman primitive "Z" computation with P-256 curve.
Diffie-Hellman	Power up	KAT Diffie-Hellman primitive "Z" computation with MODP-2048 and ffdhe3072.
RSA	Power up	KAT RSA signature verification with 2048-bit key and SHA-256
DRBG	Power up	DRBG health tests as specified in Section 11.3 of SP 800-90Arev1.

Table 14 - Cryptographic Algorithms Self-Tests

### 10.2.1 Pairwise Consistency Test

Conditional pair-wise consistency tests are performed during operational state of the module when the respective cryptographic functions are used. If any of the conditional Pair-Wise Consistency (PCT) tests fails, the module transitions to the error state. The module performs PCT tests for Diffie-Hellman and EC Diffie-Hellman as outlined in section 5.6.2.1.4 of SP 800-56Arev3.

### 10.3 Periodic/On-Demand Self-Test

On-demand self-tests can be invoked by powering-off and reloading the module which cause the module to run the pre-operational and conditional cryptographic algorithms self-tests. On-demand self-tests can also be invoked by calling `kernel_restart()` function which effectively powers off the module and then reloads the module. During the execution of the on-demand self-tests, services are not available, and no data output or input is possible.

### 10.4 Error States

When the module fails any pre-operational self-test or conditional test, the module will indicate an error has occurred and will enter the Error state. Any further cryptographic operation is inhibited. In the Error state, all data output is inhibited, and no cryptographic operations are allowed. The error can be recovered by a restart (i.e., powering off and powering on) of the module.

The following table shows the error codes and the corresponding condition:

Error State	Cause of Error	Status Indicator
Error state	Failure of pre-operational tests or conditional tests.	Kernel panic
	Failure of Entropy source Health Tests	
	Failure of PCT tests.	

Table 15 - Error States

# 11 Life-cycle assurance

## 11.1 Delivery and Operation

### 11.1.1 Module Installation

The Crypto Officer can install the RPM packages containing the module as listed in Table 17 using the zypper tool. The integrity of the RPM package is automatically verified during the installation, and the Crypto Officer shall not install the RPM package if there is any integrity error.

### 11.1.2 Operating Environment Configuration

The operating environment needs to be configured to support FIPS, so the following steps shall be performed with the root privilege:

1. Install the dracut-fips RPM package:

```
# zypper install dracut-fips
```

2. Recreate the INITRAMFS image:

```
# dracut -f
```

3. After regenerating the initrd, the Crypto Officer must append the following parameter in the `/etc/default/grub` configuration file in the `GRUB_CMDLINE_LINUX_DEFAULT` line:

```
fips=1
```

4. After editing the configuration file, please run the following command to change the setting in the boot loader depending on if the system being used uses UEFI boot or legacy boot:

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

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

If `/boot` or `/boot/efi` resides on a separate partition, the kernel parameter `boot=<partition of /boot or /boot/efi>` must be supplied. The partition can be identified with the command `"df /boot"` or `"df /boot/efi"` respectively. For example:

```
# df /boot
```

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

The partition of `/boot` is located on `/dev/sda1` in this example. Therefore, the following string needs to be appended in the aforementioned grub file:

```
"boot=/dev/sda1"
```

5. Reboot to apply these settings.

Now, the operating environment is configured to support FIPS operation. The Crypto Officer should check the existence of the file `/proc/sys/crypto/fips_enabled`, and verify it contains a numeric value "1". If the file does not exist or does not contain "1", the operating environment is not configured to support FIPS and the module will not operate as a FIPS validated module properly.

### 11.1.3 Crypto Officer Guidance for Vendor Affirmed Operational Environments

Table 16 below includes the installation process for the Vendor Affirmed Operational Environments found in section 2.4.1 and section 2.4.2 above.

Operating System	Product Link
SUSE Linux Enterprise Micro 5.3	<a href="https://documentation.suse.com/sle-micro/5.3/single-html/SLE-Micro-security/#sec-fips-slemicro-install">https://documentation.suse.com/sle-micro/5.3/single-html/SLE-Micro-security/#sec-fips-slemicro-install</a>
SUSE Linux Enterprise Server for SAP 15SP4	<a href="https://documentation.suse.com/sles/15-SP4/html/SLES-all/book-security.html">https://documentation.suse.com/sles/15-SP4/html/SLES-all/book-security.html</a>
SUSE Linux Enterprise Desktop 15SP4	<a href="https://documentation.suse.com/sled/15-SP4/html/SLED-all/book-security.html">https://documentation.suse.com/sled/15-SP4/html/SLED-all/book-security.html</a>
SUSE Linux Enterprise Real Time 15SP4	<a href="https://documentation.suse.com/sle-rt/15-SP4/">https://documentation.suse.com/sle-rt/15-SP4/</a>

Table 16 - Installation Process for Vendor Affirmed Operational Environments

Note: Per section 7.9 in the FIPS 140-3 Management Manual [FIPS140-3\_MM], the Cryptographic Module Validation Program (CMVP) makes no statement as to the correct operation of the module or the security strengths of the generated keys when this module is ported and executed in an operational environment not listed on the validation certificate.

## 11.2 Crypto Officer Guidance

The binaries of the module are contained in the RPM packages for delivery. The Crypto Officer shall follow sections 11.1.1 and 11.1.2 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:

Processor Architecture	RPM Packages
x86_64	dracut-fips-055+suse.252.g4988b0bf-150400.1.8.rpm
aarch64	kernel-default-5.14.21-150400.24.46.1.rpm
z15	kernel-rt-5.14.21-150400.15.11.1.rpm (for x86_64 processors architecture)
Power10	libkcapi-tools-0.13.0-1.114.rpm

Table 17 - RPM packages

### 11.2.1 AES XTS

The AES algorithm in XTS mode can be only used for the cryptographic protection of data on storage devices, as specified in [SP800-38E]. The length of a single data unit encrypted with the XTS-AES shall not exceed  $2^{20}$  AES blocks, that is 16MB of data.

To meet the requirement stated in IG C.I, the module implements a check that ensures, before performing any cryptographic operation, that the two AES keys used in AES XTS mode are not identical.

Note: AES-XTS shall be used with 128 and 256-bit keys only. AES-XTS with 192-bit keys is not an Approved service.

### 11.2.2 AES GCM IV

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



The GCM with internal IV generation in the approved mode is in compliance with RFC4106 and shall only be used in conjunction with the IPsec stack of the kernel to be compliant with IG C.H scenario 1. Any other usage of GCM encryption is considered as non-Approved.

The `nonce_explicit` part of the IV does not exhaust the maximum number of possible values for a given session key. The design of the IPsec protocol ensures that the `nonce_explicit`, or counter portion, of the IV will not exhaust all of its possible values.

When a GCM IV is used for decryption, the responsibility for the IV generation lies with the party that performs the AES-GCM encryption. The module merely receives the GCM IV and performs the operation. It is not responsible for generating the IV.

### 11.2.3 Handling Self-Test Errors

Self test failure within the kernel crypto API module will panic the kernel and the operating system will not load and/or halt immediately.

Error recovery and return to operational state can be accomplished by rebooting the system. If the failure continues, the Crypto Officer must re-install the software package and make sure to follow all instructions. If the software was downloaded, the package hash value must be verified to confirm a proper download. Please contact SUSE if these steps do not resolve the problem.

The kernel dumps self-test success and failure messages into the kernel message ring buffer. Post boot, the messages are moved to `/var/log/messages`.

Use **dmesg** to read the contents of the kernel ring buffer. The format of the ringbuffer (**dmesg**) output is:

```
alg: self-tests for %s (%s) passed
```

Typical messages are similar to "alg: self-tests for xts(aes) (xts(aes)) passed" for each algorithm/sub-algorithm type.

### 11.2.4 SP 800-56Ar3 Assurances

To comply with the assurances found in Section 5.6.2 of SP 800-56Ar3, the operator must use the Diffie-Hellman and Elliptic Curve Diffie-Hellman shared secret computation algorithms in the context of IETF protocols. Additionally, the module's approved key pair generation service must be used to generate ephemeral Diffie-Hellman or EC Diffie-Hellman key pairs, or the key pairs must be obtained from another FIPS-validated module. As part of this service, the module will internally perform the full public key validation of the generated public key.

The module's shared secret computation service will internally perform the full public key validation of the peer DH public key, and the partial public key validation of the peer EC public key, complying with Section 5.6.2.2.2 of SP 800-56Ar3.

### 11.2.5 End of Life Procedure

As a first step for the secure sanitization, the module needs to be powered off which will erase the SSPs in the volatile memory. Then, the files listed in Table 2 must be deleted using the command "`shred -zu <file_name>`". Then, for the actual deprecation, the module will be upgraded to a newer version that is approved.

## **12 Mitigation of other attacks**

The module does not offer mitigation of other attacks.

## Appendix A. Glossary and Abbreviations

<b>AES</b>	Advanced Encryption Standard
<b>AES-NI</b>	Advanced Encryption Standard New Instructions
<b>CAVP</b>	Cryptographic Algorithm Validation Program
<b>CBC</b>	Cipher Block Chaining
<b>CCM</b>	Counter with Cipher Block Chaining-Message Authentication Code
<b>CFB</b>	Cipher Feedback
<b>CMAC</b>	Cipher-based Message Authentication Code
<b>CMVP</b>	Cryptographic Module Validation Program
<b>CPACF</b>	Central Processor Assist for Cryptographic Function
<b>CSP</b>	Critical Security Parameter
<b>CTR</b>	Counter Mode
<b>DF</b>	Derivation Function
<b>DRBG</b>	Deterministic Random Bit Generator
<b>ECB</b>	Electronic Code Book
<b>ECC</b>	Elliptic Curve Cryptography
<b>FFC</b>	Finite Field Cryptography
<b>FIPS</b>	Federal Information Processing Standards Publication
<b>FSM</b>	Finite State Model
<b>GCM</b>	Galois Counter Mode
<b>HMAC</b>	Hash Message Authentication Code
<b>ISA</b>	Instruction Set Architecture
<b>KAS</b>	Key Agreement Schema
<b>KAT</b>	Known Answer Test
<b>KW</b>	AES Key Wrap
<b>KWP</b>	AES Key Wrap with Padding
<b>MAC</b>	Message Authentication Code
<b>NIST</b>	National Institute of Science and Technology
<b>OFB</b>	Output Feedback
<b>PAA</b>	Processor Algorithm Acceleration
<b>PR</b>	Prediction Resistance
<b>RNG</b>	Random Number Generator
<b>RSA</b>	Rivest, Shamir, Adleman
<b>SSC</b>	Shared Secret Computation
<b>SHA</b>	Secure Hash Algorithm
<b>SHS</b>	Secure Hash Standard

**SSP** Sensitive Security Parameter  
**XTS** XEX-based Tweaked-codebook mode with cipher text Stealing

## Appendix B. References

- FIPS140-3**      **FIPS PUB 140-3 - Security Requirements For Cryptographic Modules**  
March 2019  
<https://csrc.nist.gov/csrc/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf>
- FIPS140-3\_IG**      **Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program**  
October 2022  
<https://csrc.nist.gov/csrc/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf>
- FIPS140-3\_MM**      **FIPS 140-3 Cryptographic Module Validation Program - Management Manual**  
April 2024  
<https://csrc.nist.gov/csrc/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS-140-3-CMVP%20Management%20Manual.pdf>
- FIPS180-4**      **Secure Hash Standard (SHS)**  
March 2012  
<https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>
- FIPS186-4**      **Digital Signature Standard (DSS)**  
July 2013  
<https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>
- FIPS197**      **Advanced Encryption Standard**  
November 2001  
<https://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>
- FIPS198-1**      **The Keyed Hash Message Authentication Code (HMAC)**  
July 2008  
[https://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1\\_final.pdf](https://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf)
- FIPS202**      **SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions**  
August 2015  
<https://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>
- RFC3394**      **Advanced Encryption Standard (AES) Key Wrap Algorithm**  
September 2002  
<https://www.ietf.org/rfc/rfc3394.txt>

- RFC5649**      **Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm**  
September 2009  
<https://www.ietf.org/rfc/rfc5649.txt>
- SP800-38A**      **NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques**  
December 2001  
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf>
- SP800-38B**      **NIST Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication**  
May 2005  
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38B.pdf>
- SP800-38C**      **NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality**  
May 2004  
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf>
- SP800-38D**      **NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC**  
November 2007  
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf>
- SP800-38E**      **NIST Special Publication 800-38E - Recommendation for Block Cipher Modes of Operation: The XTS AES Mode for Confidentiality on Storage Devices**  
January 2010  
<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38e.pdf>
- SP800-38F**      **NIST Special Publication 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>
- SP800-38G**      **NIST Special Publication 800-38G - Recommendation for Block Cipher Modes of Operation: Methods for Format - Preserving Encryption**  
March 2016  
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38G.pdf>
- SP800-56Arev3**      **NIST Special Publication 800-56A Revision 3 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**  
April 2018  
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf>

<b>SP800-56B</b>	<b>Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography</b> March 2019 <a href="https://csrc.nist.gov/publications/detail/sp/800-56b/rev-2/final">https://csrc.nist.gov/publications/detail/sp/800-56b/rev-2/final</a>
<b>SP800-56Crev2</b>	<b>Recommendation for Key Derivation through Extraction-then-Expansion</b> August 2020 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf</a>
<b>SP800-57rev5</b>	<b>NIST Special Publication 800-57 Part 1 Revision 5 - Recommendation for Key Management Part 1: General</b> May 2020 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf</a>
<b>SP800-90Arev1</b>	<b>NIST Special Publication 800-90A - Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators</b> June 2015 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf</a>
<b>SP800-90B</b>	<b>NIST Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation</b> January 2018 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf</a>
<b>SP800-108</b>	<b>NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions (Revised)</b> October 2009 <a href="https://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf">https://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf</a>
<b>SP800-131Arev2</b>	<b>NIST Special Publication 800-131A Revision 1- Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths</b> March 2019 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf</a>
<b>SP800-132</b>	<b>NIST Special Publication 800-132 - Recommendation for Password-Based Key Derivation - Part 1: Storage Applications</b> December 2010 <a href="https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-132.pdf">https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-132.pdf</a>
<b>SP800-133rev2</b>	<b>NIST Special Publication 800-133 - Recommendation for Cryptographic Key Generation</b> June 2020 <a href="https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf">https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf</a>

**SP800-135rev1**    **NIST Special Publication 800-135 Revision 1 - Recommendation for Existing Application-Specific Key Derivation Functions**

December 2011

<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf>**SP800-140B**    **NIST Special Publication 800-140B - CMVP Security Policy Requirements**

March 2020

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-140B.pdf>