# Ruckus Networks Virtual SmartZone (vSZ) Version 5.2.1.3

## FIPS 140-2 Level 1 Non-Proprietary Security Policy by CommScope Technologies LLC.

**Software Version: 5.2.1.3**

**Documentation Version Number: 1.6**

**May 16, 2023**

# Table of Contents

# LIST OF FIGURES

# LIST OF TABLES

# 1 Module Overview

### Ruckus Networks Virtual SmartZone (vSZ)

Ruckus virtual SmartZone (vSZ), is a Network Functions Virtualization (NFV) based WLAN Controller for customers who desire a carrier-class solution that runs in the cloud. It supports all of the WLAN Controller features of the industry leading physical controllers, while also enabling the rollout of highly scalable and resilient wireless LAN cloud services. The vSZ is a software module, which is defined as a multi-chip standalone cryptographic module by FIPS 140-2.

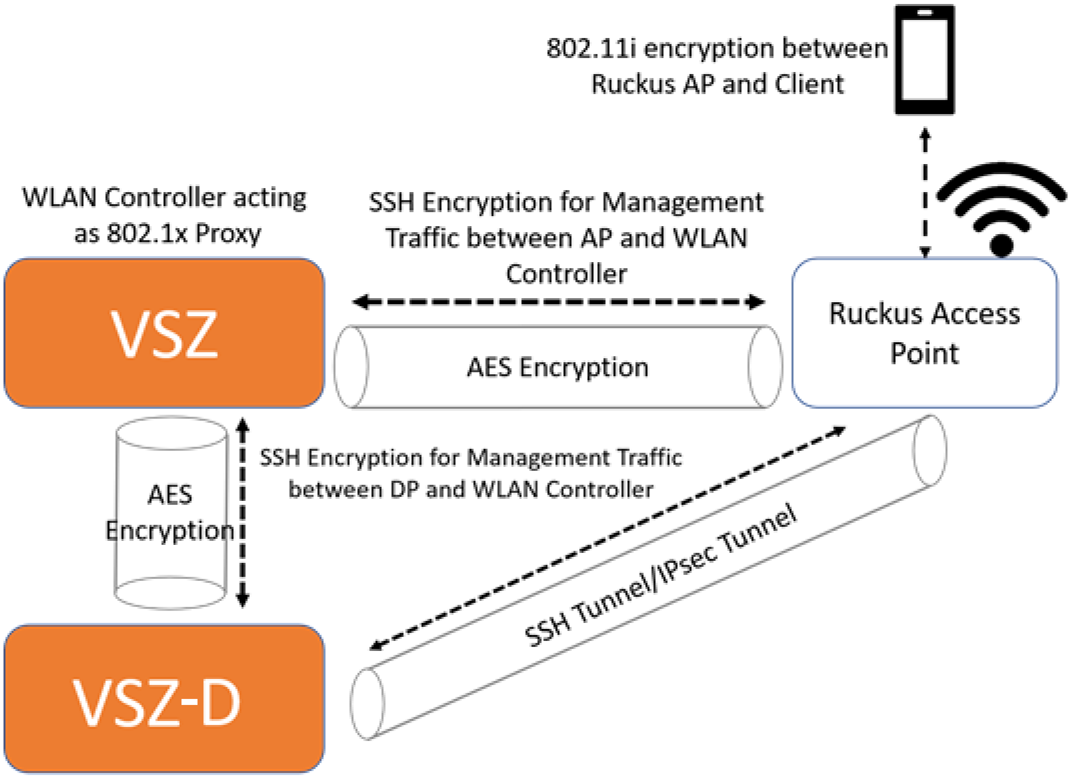
802.11i encryption between Ruckus AP and Client
WLAN Controller acting as 802.1x Proxy
SSH Encryption for Management Traffic between AP and WLAN Controller
VSZ
AES Encryption
Ruckus Access Point
AES Encryption
SSH Encryption for Management Traffic between DP and WLAN Controller
SSH Tunnel/IPsec Tunnel
VSZ-D

**Figure 1: Encryption between AP and Controller**

FIPS 140-2 conformance testing was performed at Security Level 1. The following configurations were tested by the lab.

**Table 1: Configuration tested by the lab**

| Module | Processors | Operating Systems | Tested Platform |
|---|---|---|---|
| Ruckus Networks Virtual SmartZone (vSZ)<br>SW Version: 5.2.1.3 | Intel XeonE5-2650 v2 with AES-NI; Intel Xeon E5-2650 v2 without AES-NI | CentOS 7.5 on VMware ESXi 6.5.0 | Dell PowerEdge R620 |

The Cryptographic Module meets FIPS 140-2 Level 1 requirements.

**Table 2: Module Security Level Statement**

| FIPS Security Area | Security Level |
|---|---|
| Cryptographic Module Specification | 1 |
| Module Ports and Interfaces | 1 |
| Roles, Services and Authentication | 2 |
| Finite State Model | 1 |
| Physical Security | N/A |
| Operational Environment | 1 |
| Cryptographic Key Management | 1 |
| EMI/EMC | 1 |
| Self-tests | 1 |
| Design Assurance | 2 |
| Mitigation of Other Attacks | N/A |

The module is defined as a multi-chip standalone software module, with the physical boundary being defined as the hard case enclosure around which everything runs. The logical cryptographic boundary is the vSZ cryptographic module, with the shared library file named /lib64/libcrypto.so.10. The module performs no communication other than with the calling application (the process that invokes the module services). The cryptographic boundary of the module is shown below.

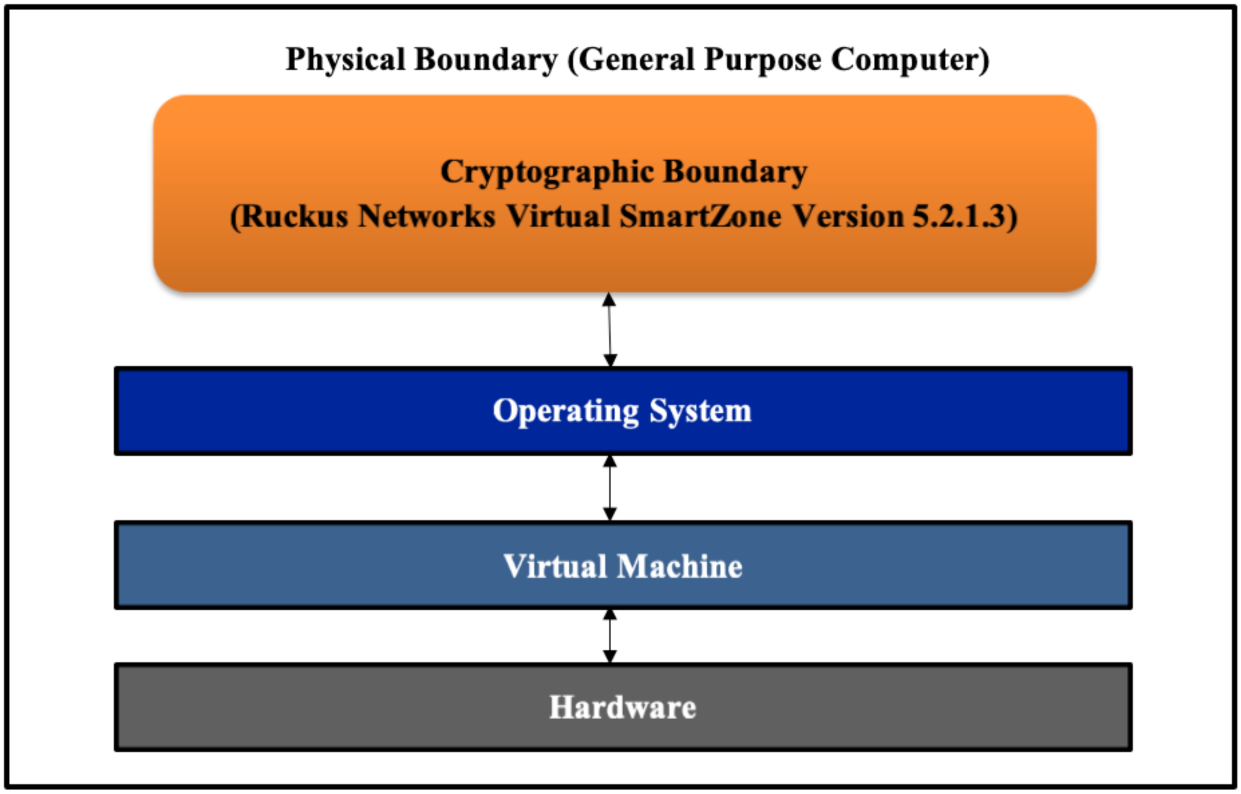
Physical Boundary (General Purpose Computer)
Cryptographic Boundary
(Ruckus Networks Virtual SmartZone Version 5.2.1.3)
Operating System
Virtual Machine
Hardware

**Figure 2: Block Diagram for Ruckus Networks Virtual SmartZone (vSZ)**

# 2 Modes of Operation

The module is intended to always operate in the FIPS Approved mode. A provision is made to disable/enable FIPS mode via configuration (Login CLI -> enabled mode -> fips enable/disable). In addition to running the 'fips enable' command, an operator must follow the procedural rules specified in Section 8 to remain in the Approved mode.

## 2.1 Approved and Allowed Cryptographic Algorithms

The following approved cryptographic algorithms are used in FIPS approved mode of operation. Note that in some cases, more algorithms/modes of operation have been tested than are utilized by the Module. Only implementations that are used are shown in the table below.

**Table 3: Approved Cryptographic Algorithms**

| CAVP Cert | Algorithm | Standard | Model/Method | Use |
|---|---|---|---|---|
| **Ruckus Virtual SmartZone (vSZ) Crypto – Kernel Algorithm Implementation** | | | | |
| C2078 | AES | FIPS 197,<br>SP 800-38A | CBC (128, 192, 256 bits) | Data Encryption/Decryption |
| C2078 | HMAC | FIPS 198-1 | HMAC-SHA-1<br>HMAC-SHA-256<br>HMAC-SHA-384<br>HMAC-SHA-512 | Message Authentication |
| C2078 | SHA | FIPS 180-4 | SHA-1<br>SHA-256<br>SHA-384<br>SHA-512 | Message Digest |
| **Ruckus Virtual SmartZone (vSZ) Crypto - OpenSSL/OpenSSH Algorithm Implementation** | | | | |
| C2083 | AES | FIPS 197,<br>SP 800-38A,<br>SP 800-38D | CBC, CFB128, CTR, GCM (128, 192, 256 bits) | Data Encryption/Decryption |
| Vendor affirmed | CKG | SP 800-133 | N/A | Key Generation |
| C2083 | CVL | SP 800-135 KDF | SSHv2<br>TLSv1.2,<br>IKEv2,<br>SNMPv3 | Key Derivation |
| A2457 | KAS-FFC-SSC | SP800-56Arev3 | dhEphem<br>ffdhe2048, ffdhe3072, MODP-2048, MODP-3072 | Key establishment methodology provides 112 or 128 bits of encryption strength |
| A2457 | KAS-ECC-SSC | SP800-56Arev3 | ephemeralUnified<br>P-256, P-384, P-521 | Key establishment methodology provides between 128 and 256 bits of encryption strength |

| CAVP Cert | Algorithm | Standard | Model/Method | Use |
|---|---|---|---|---|
| A2457 | KAS (FFC)<br>KAS (ECC) | SP 800-56rev3<br>SP 800-135rev1 | Diffie-Hellman dhEphem ffdhe2048, ffdhe3072, MODP-2048, MODP-3072;<br>EC Diffie-Hellman P-256, P-384 and P-521; with SSHv2, TLSv1.2, IKEv2 and SNMPv3 KDF | Key Agreement Scheme per SP 800-56Arev3 with key derivation per SP 800-135rev1 |
| C2083 | DRBG | SP 800-90A | CTR_DRBG (AES-256) | Deterministic Random Bit Generation |
| C2083 | ECDSA | FIPS 186-4 | Key Generation:<br>- Curves: P-256/384/521<br>SigGen/SigVer:<br>- Curves: P-256/384/521 with SHA-256/384/512 | Key Generation, Digital Signature Generation and Verification |
| C2083 | HMAC | FIPS 198-1 | HMAC-SHA1<br>HMAC-SHA256<br>HMAC-SHA384<br>HMAC-SHA512 | Message Authentication |
| C2083 | KTS | SP 800-38F | AES (128, 192, 256 bits) with HMAC-SHA-1/256/384/512 | Key Transport |
| C2083 | KTS | SP 800-38F | AES-GCM (128, 256 bits) | Key Transport |
| C2083 | RSA | FIPS 186-2<br>FIPS 186-4<br>Note: only FIPS 186-2 RSA 4096 bits was used in FIPS mode | FIPS 186-4 RSA Key Generation:<br>- Key Generation Mode: B.3.3<br>- 2048/3072-bits<br>FIPS 186-4 RSA SigGen/SigVer:<br>- PCKSv1.5<br>- 2048/3072-bits with SHA-256/384/512<br>FIPS 186-2 RSA SigVer:<br>- PCKSv1.5<br>- 4096-bits with SHA-1/256/384/512 | Key Generation, Digital Signature Generation and Verification |
| A2457 | Safe Primes | SP800-56Arev3 | KeyGen/KeyVer;<br>ffdhe2048, ffdhe3072, MODP-2048, MODP-3072 | KAS-FFC-SSC Domain Parameters Generation with SafePrimes groups |
| C2083 | SHS | FIPS 180-4 | SHA1<br>SHA-256<br>SHA-384<br>SHA-512 | Message Digest |

Notes:

- There are some algorithm modes that were tested but not used by the module. Only the algorithms, modes, and key sizes that are implemented by the module are shown in this table.

- The module's AES-GCM implementation conforms to IG A.5 scenario #1 following RFC 5288 for TLS and RFCs 4252, 4253 and RFC 5647 for SSHv2. The module is compatible with TLSv1.2 and provides support for the acceptable GCM cipher suites from SP 800-52 Rev1, Section 3.3.1. The operations of one of the two parties involved in the TLS key establishment scheme were performed entirely within the cryptographic boundary of the module being validated. The counter portion of the IV is set by the module within its cryptographic boundary. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition will trigger a handshake to establish a new encryption key. In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption shall be established. The module is also compatible with SSHv2 and provides support for the acceptable GCM cipher suites from Section 7.1 of RFC 5647. The IV consist of a 4-byte fixed field and an 8-byte invocation counter. If the invocation counter reaches its maximum value 2^64 – 1, the next AES GCM encryption is performed with the invocation counter set to 0. No more than 2^64 – 1 AES GCM encryptions may be performed in the same session. The SSH session is reset for both the client/server after one GB of data (2^23 block encryptions) or one hour whichever comes first. When a session is terminated for any reason, a new key and a new initial IV are derived.

- No parts of the SSH, TLS, SNMP and IPsec protocols, other than the KDFs, have been tested by the CAVP and CMVP.

- In accordance with FIPS 140-2 IG D.12, the cryptographic module performs Cryptographic Key Generation as per section 6 in SP800-133rev2. The resulting generated seed used in the asymmetric key generation is the unmodified output from SP800-90A DRBG.

## 2.2 Non-FIPS Approved but Allowed Cryptographic Algorithms

The following non-FIPS approved but allowed cryptographic algorithms are used in FIPS approved mode of operation.

**Table 4: Non-FIPS Approved but Allowed Cryptographic Algorithms.**

| Algorithm | Caveat | Use |
|---|---|---|
| NDRNG | | Used to seed the SP 800-90A DRBG. |

## 2.3 Non-FIPS Approved Cryptographic Algorithms

The following non-FIPS approved cryptographic algorithms are used only in the non-Approved mode of operation.

**Table 5: Algorithms/Services in Non-Approved Mode**

| Algorithm | Use |
|---|---|
| chacha20-poly1305@openssh.com, umac-64@openssh.com, hmac-sha1-etm@openssh.com, umac-64-etm@openssh.com, umac-128-etm@openssh.com, hmac-sha2-256-etm@openssh.com, hmac-sha2-512-etm@openssh.com, hmac-ripemd160-etm@openssh.com, umac-64@openssh.com, umac-128@openssh.com, hmac-ripemd160,<br>DSA, ED25519 | OpenSSH |
| MD5, DES | SNMP |
| MD5, DES, TDES | OpenSSL |

**Note**

- In addition to the FIPS mode of operation, the cryptographic module can also be operated in a non-FIPS mode of operation. Table 5 lists the non-approved/non-allowed the algorithms and services are available to both the User role and CO role in the module. Prior to using any of the Non-Approved services with the associated non-approved/non-allowed algorithms listed in Table 5 above, the Crypto Officer must zeroize all CSPs, which would put the module into the non-FIPS mode of operation.

- Neither the User nor the Crypto Officer are allowed to operate any of these services listed in table 5 above while in FIPS mode of operation.

- To put the module back into the FIPS mode from the non-FIPS mode, the CO must zeroize all Keys/CSPs used in non-FIPS mode, and then strictly follow up the steps in section 8 of this document to put the module into the FIPS mode.

- In addition, addition, all available services supported by the module can be found at RUCKUS FIPS and Common Criteria Configuration Guide for SmartZone and AP, 5.2.1.3, Published on 2021-04-14 with the documentation Part Number 800-72735-001 RevA, https://support.ruckuswireless.com/documents/3509.

# 3 Ports and interfaces

The following table describes logical interfaces of the module.

**Table 6: FIPS 140-2 Logical Interfaces**

| Logical Interface | Description |
|---|---|
| Data Input | Input parameters that are supplied to the API commands |
| Data Output | Output parameters that are returned by the API commands |
| Control Input | API commands |
| Status Output | Return status provided by API commands |

# 4 Roles and Services

The module supports role-based authentication mechanism. Each role is authenticated by the module upon initial access to the module. There are three roles supported by the module: Crypto Officer role, User role and AP (Access Point) role. The Crypto Officer installs and administers the module. The Users and APs use the cryptographic services provided by the module.

The User role or Crypto Officer role password as well as all other shared secrets must each be at least eight (8) characters long, including at least one alphabet, one numeric character, one special character (note: The special character ` cannot be used in the password and the special characters combination '$(' cannot be used in the password). Given these restrictions, we have 52 x 10 x 31 x 93 ^ 5 = 112,144,965,131,160 password combinations. If the '$(' combination was chosen in the password, then it would have 1 x 52 x 10 x 93^4 = 38,898,704,520 combinations, resulting the final correct password combinations are 112,144,965,131,160 - 38,898,704,520 = 112,106,066,426,640. Thus, the probability of a successful random attempt is approximately is one (1) in 112,106,066,426,640, which is less than the 1 in 1,000,000 required by FIPS 140-2. This calculation is based on the assumption that the typical standard American QWERTY computer keyboard has 10 Integer digits, 52 alphabetic characters, and 32 special characters providing 94 characters to choose from in total.

In addition, for multiple attempts to use the authentication mechanism during a one-minute period, under the optimal modern network condition, if an attacker would only get 60,000 guesses per minute. Therefore, the associated probability of a successful random attempt during a one-minute period is 60,000/112,106,066,426,640 = 1/1,868,434,440, which is less than 1 in 100,000 required by FIPS 140-2.

Additionally, when using RSA based authentication (AP Role), RSA key pair has modulus size of 3072 bits, thus providing 128 bits of strength, which means an attacker would have a 1 in 2^128 chance of randomly obtaining the key, which is much stronger than the one in a million chances required by FIPS 140-2. To exceed a one in 100,000 probability of a successful random key guess in one minute, an attacker would have to be capable of approximately 2.04x10^40 (2^128 /60 = 2.04 x 10^40) attempts per second, which far exceeds the operational capabilities of the module to support.

Table 7 below lists the complete services and the associated types of access to the Keys/CSPs access supported by each role.

**Table 7: Approved Mode Roles and Services**

| Service | Corresponding Roles | Types of Access to Cryptographic Keys and CSPs<br>R – Read or Execute<br>W – Write or Create<br>Z – Zeroize |
|---|---|---|
| Reboot/Self-test | Crypto Officer<br>User | All (not including instances in NVRAM): Z |
| Zeroization | Crypto Officer | All: Z |
| Software update | Crypto Officer | Software update key: R |
| Show status | Crypto Officer<br>User<br>AP | N/A |
| Login | Crypto Officer<br>User | Password: R<br>SSHv2 Keys: R, W<br>TLSv1.2 Keys: R, W<br>DRBG related Keys: R, W |
| SSHv2 Functions | Crypto Officer<br>User<br>AP | Password: R, W<br>SSHv2 Keys: R, W<br>DRBG related Keys: R, W |
| Configuration | Crypto Officer | Password: R, W<br>SSHv2 Keys: R, W<br>TLSv1.2 Keys: R, W<br>DRBG related Keys: R, W |
| RadSec (RADIUS over TLS) | AP | TLSv1.2 Keys: R, W<br>DRBG related Keys: R, W<br>Radius Secret: R, W |
| NTP | Crypto Officer | NTP Keys: R, W |
| HTTPS/TLSv1.2 Functions | Crypto Officer<br>User<br>AP | TLSv1.2 Keys: R, W<br>DRBG related Keys: R, W |
| IPsec/IKEv2 Functions | Crypto Officer | IPsec/IKEv2 Keys: R, W<br>DRBG related Keys: R, W |
| EAP authenticator<br>(EAP-TLS, EAP-TTLS, EAP-PEAP) | AP | SSHv2 Keys: R, W<br>DRBG related Keys: R, W<br>TLSv1.2 Keys: R, W |
| SNMPv3 Functions | Crypto Officer<br>User | Password: R, W<br>SNMPv3 Keys: R, W |
| FIPS mode enable/disable | Crypto Officer | ALL: Z |

For the services and algorithms supported by the module while in non-approved mode of operation, please refer to section 2.3 in this document for more information.

## Unauthenticated Services

The module also supports the unauthenticated services, including the view to the status output from the module’s LED, and power cycling.

# 5 Operational Environment

The operating system is restricted to a single operator mode of operation wherein concurrent operators are explicitly excluded.

This software cryptographic module is implemented in client/server architecture and is intended to be used on both the client and the server. It will be used to provide cryptographic functions to the client and server applications. Since this module is implemented in a server environment, the server application is the user of the cryptographic module. The server application makes the calls to the cryptographic module. Therefore, the server application is the single user of the module, even when the server application is serving multiple clients. Please refer to Table 1 above for the tested platform information.

# 6 Cryptographic Keys and CSPs

The entropy source (NDRNG) within the module provides at least 256 bits of entropy to seed SP800-90a DRBG for use in key generation. The table below describes cryptographic keys and CSPs used by the module.

**Table 8: Cryptographic Keys and CSPs**

| Name | CSP Type | Size | Description/Usage | Storage | Zeroization |
|---|---|---|---|---|---|
| DRBG Entropy Input | SP800-90A CTR_DRBG (AES-256) | 384 bits | This is the entropy for SP 800-90A CTR_DRBG, used to construct the seed. | DRAM (plaintext) | Power cycle the device |
| DRBG Seed | SP800-90A CTR_DRBG (AES-256) | 384 bits | Input to the DRBG that determines the internal state of the DRBG. Generated using DRBG derivation function that includes the entropy input from the entropy source. | DRAM (plaintext) | Power cycle the device |
| DRBG V | SP800-90A CTR_DRBG (AES-256) | 128 bits | The DRBG V is one of the critical values of the internal state upon which the security of this DRBG mechanism depends. Generated first during DRBG instantiation and then subsequently updated using the DRBG update function. | DRAM (plaintext) | Power cycle the device |
| DRBG Key | SP800-90A CTR_DRBG (AES-256) | 256 bits | Internal critical value used as part of SP 800-90A CTR_DRBG. Established per SP 800-90A CTR_DRBG. | DRAM (plaintext) | Power cycle the device |
| User Password | Password | At least eight characters | Password used to authenticate the User (at least eight (8) characters). | NVRAM (cyphertext) | Procedurally erase the password |
| Crypto Officer Password | Password | At least eight characters | Password used to authenticate the Crypto Officer (at least eight (8) characters) | NVRAM (cyphertext) | Procedurally erase the password |
| Software Upgrade Verification Key | RSA (FIPS 186-2) | 4096 bits | RSA public key used to verify the signature for Software Update/Load Test. The key was pre-installed on the system for signature verification. Note | NVRAM (plaintext) | Zeroized by erasing the software image |

| Name | CSP Type | Size | Description/Usage | Storage | Zeroization |
|---|---|---|---|---|---|
| | | | that the public key is a cryptographic key, but not considered as CSP. | | |
| Software Integrity Test Key | RSA (FIPS 186-2) | 4096 bits | RSA public key used to verify the signature for Software Integrity Test. The key was pre-installed on the system for signature verification. Note that the public key is a cryptographic key, but not considered as CSP. | NVRAM<br>(plaintext) | Zeroized by erasing the software image |
| NTP Secret | Shared Secret | 40 characters | Used to authenticate with the NTP server (40 characters in Approved mode, no restriction in non-Approved mode). The key is configured by user. | NVRAM (plaintext) | Procedurally erase the shared secret |
| RADIUS Secret | Shared Secret | At least 8 characters | Used to authenticate with the RadSec server (at least eight (8) characters). The key is configured by the Crypto Officer. | NVRAM (plaintext) | Procedurally erase the shared secret |
| **TLSv1.2 Protocol Keys and CSPs** | | | | | |
| TLS DH/ECDH Private Key | KAS-FFC/ECC-SSC (DH /ECDH)<br>[SP800-56Arev3] | 256 bits/P-384 curve | DH or ECDH private key used to establish the TLSv1.2 DH/ECDH shared secret. This key was generated by calling FIPS approved DRBG. | DRAM (plaintext) | Automatically when TLS session is terminated. |
| TLS DH/ECDH Public Key | KAS-FFC/ECC-SSC (DH /ECDH)<br>[SP800-56Arev3] | 3072 bits/P-384 curve | DH or ECDH public key used in TLSv1.2 handshakes. Note that the public key is a cryptographic key, but not considered a CSP | DRAM (plaintext) | Automatically when TLS session is terminated. |
| TLS DH/ECDH Shared Secret | KAS-FFC/ECC-SSC (DH/ECDH)<br>[SP800-56Arev3] | 3072 bits | The shared secret used in TLSv1.2 DH/ECDH exchange. This key was derived per the DH/ECDH key agreement scheme. | DRAM (plaintext) | Automatically when TLS session is terminated |
| TLS RSA Private Key | RSA (FIPS 186-4) | 3072 bits | RSA private key, used to sign the authentication certificate during the TLSv1.2 handshakes. This key was generated by calling FIPS approved DRBG | NVRAM (plaintext) | Zeroization by RSA Keypair delete command |
| TLS RSA Public Key | RSA (FIPS 186-4) | 3072 bits | RSA public key, used for authentication during the TLSv1.2 handshakes. This key is derived in compliance with FIPS 186-4 RSA key pair generation method in the module. Note that the public key is a cryptographic key, but not considered a CSP. | NVRAM (plaintext) | Zeroization by RSA Keypair delete command |
| TLS Pre-Master Secret | keying material | At least eight characters | Keying material used in TLSv1.2 handshakes. This key was used to derive TLSv1.2 Master Secret. | DRAM (plaintext) | Automatically when TLS session is terminated. |
| TLS Master Secret | keying material | 48 bytes | Keying material, used to derive TLS Encryption Key and TLS Authentication Key. The master | DRAM (plaintext) | Automatically when TLS |

| Name | CSP Type | Size | Description/Usage | Storage | Zeroization |
|---|---|---|---|---|---|
| | | | secret was derived from TLS pre-master secret during the TLS session establishment. | | session is terminated. |
| TLS Encryption Key | AES-CBC or AES-GCM | AES 128/256 bits | This key is used to encrypt/decrypt the data throughout the TLSv1.2 session. This key was derived via key derivation function defined in SP800-135 KDF (TLSv1.2). | DRAM (plaintext) | Automatically when TLS session is terminated. |
| TLS Authentication Key | HMAC-SHA256<br>HMAC-SHA384 | 256 bits<br>384 bits | This key is used to protect the data integrity throughout the TLSv1.2 session. This key is derived via key derivation function defined in SP800-135 KDF (TLSv1.2). | DRAM (plaintext) | Automatically when TLS session is terminated. |
| **SSHv2 protocol Keys/CSPs** | | | | | |
| SSHv2 DH/ECDH Private Key | KAS-FFC/ECC-SSC (DH /ECDH)<br>[SP800-56Arev3] | 224 bits/P-256, P-384 and P-521 curves | DH/ECDH private key, used to derive SSHv2 DH/ECDH Shared Secret during the SSHv2 handshakes. This key was generated by calling FIPS approved DRBG. | DRAM (plaintext) | Automatically when SSH session is terminated. |
| SSHv2 DH/ECDH Public Key | KAS-FFC/ECC-SSC (DH /ECDH)<br>[SP800-56Arev3] | 2048 bits/P-256, P-384 and P-521 curves | DH/ECDH public key, used in SSHv2 ECDH exchange. This key is established per the DH/ECDH key agreement. Note that the public key is a cryptographic key, but not considered a CSP. | DRAM (plaintext) | Automatically when SSH session is terminated. |
| SSHv2 DH/ECDH Shared Secret | KAS-FFC/ECC-SSC (DH/ECDH)<br>[SP800-56Arev3] | 2048 bits/P-256, P-384, P521 curves | The shared secret used in SSHv2 DH/ECDH exchange. This key was derived per the DH/ECDH key agreement scheme. | DRAM (plaintext) | Power cycle the device. |
| SSHv2 RSA/ECDSA Private Key | RSA/ECDSA | 3072 bits/P-384 curve | RSA or ECDSA private key, used to sign the authentication certificate during the SSHv2 handshakes. The key was generated by calling SP800-90A DRBG. | NVRAM (plaintext) | Zeroization by RSA Keypair delete command |
| SSHv2 RSA/ECDSA Public Key | RSA/ECDSA | 3072 bits/P-384 curve | RSA or ECDSA public key, used for authentication during the SSHv2 handshake. This key is derived in compliance with FIPS 186-4 RSA/ECDSA key pair generation method in the module. Note that the public key is a cryptographic key, but not considered a CSP. | NVRAM (plaintext) | Zeroization by RSA Keypair delete command |
| SSHv2 Session Key | AES-CTR or AES-GCM | CTR mode:<br>128/256 bits<br>GCM mode:<br>256 bits | This key is used to encrypt/decrypt the data throughout the SSHv2 session. This key is derived from key derivation function defined in SP800-135 KDF (SSHv2). | DRAM (plaintext) | Automatically when SSH session is terminated. |
| SSHv2 Authentication Key | HMAC-SHA1<br>HMAC-SHA256<br>HMAC-SHA512 | 160 bits<br>256 bits<br>512 bits | This key is used to protect the data integrity throughout the TLSv1.2 session. This key is derived from key derivation | DRAM (plaintext) | Automatically when SSH session is terminated. |

| Name | CSP Type | Size | Description/Usage | Storage | Zeroization |
|---|---|---|---|---|---|
| | | | function defined in SP800-135 KDF (SSHv2). | | |
| **IPsec/IKEv2 Keys and CSPs** | | | | | |
| IKEv2 ECDH Private Key | KAS-ECC-SSC (ECDH)<br>[SP800-56Arev3] | P-384 curve | ECDH private key, used to sign the authentication certificate signature verification Used during the IKEv2 handshakes. This key was generated by calling FIPS approved DRBG. | DRAM (plaintext) | Automatically when IPsec session is terminated. |
| IKEv2 ECDH Public Key | KAS-ECC-SSC (ECDH) | P-384 curve | ECDH public key, used in IKEv2 EC Diffie-Hellman (DH) exchange. This key is established per the ECDH key agreement. Note that the public key is a cryptographic key, but not considered a CSP | DRAM (plaintext) | Automatically when IPsec session is terminated. |
| IKEv2 ECDH Shared Secret | KAS-ECC-SSC (ECDH) | P-384 curve | The shared secret used to in IKEv2 ECDH exchange. This key was derived per the ECDH key agreement scheme. | DRAM (plaintext) | Power cycle the device. |
| IKEv2 RSA/ECDSA Private Key | RSA/ECDSA | RSA:3072 bits<br>ECDSA: P-256, P384, P-521 curves | RSA or ECDSA private key used for authentication during the IKEv2 protocol handshake. This key was generated by calling FIPS approved DRBG. | NVRAM (plaintext) | Zeroization by RSA/ECDSA Keypair delete command |
| IKEv2 RSA/ECDSA Public Key | RSA/ECDSA | RSA: 3072 bits<br>ECDSA: P-256, P384, P-521 curves | RSA or ECDSA public key used for authentication during the IKEv2 protocol handshake. The key is derived in compliance with FIPS 186-4 RSA/ECDSA key pair generation method in the module. Note that the public key is a cryptographic key, but not considered a CSP. | NVRAM (plaintext) | Zeroization by RSA/ECDSA Keypair delete command |
| IKEv2 Pre-Shared Key | Shared Secret | 8-63 characters | Used to authenticate IPsec peers to each other. This key is configured by the Crypto Officer. | NVRAM (plaintext) | Configuration changes or zeroization by mode change |
| SKEYSEED | Keying material | 160 bits | Keying material used to derive the IKEv2 session key. It was derived via key derivation function defined in SP800-135 KDF (IKEv2). | DRAM (plaintext) | Automatically when IPSec/IKE session is terminated |
| IKEv2 Encryption Key | AES-CBC | 128/192/256 bits | This key is used to encrypt/decrypt the data throughout the IKEv2 session. This key was derived by key derivation function defined in SP800-135 KDF (IKEv2). | DRAM (plaintext) | Automatically when IPsec session is terminated. |
| IKEv2 Authentication Key | HMAC-SHA256<br>HMAC-SHA384<br>HMAC-SHA512 | 256 bits<br>384 bits<br>512 bits | This key is used to protect the data integrity of data throughout the IKEv2 session. This key is derived by key derivation function defined in SP800-135 KDF (IKEv2). | DRAM (plaintext) | Automatically when IPsec session is terminated. |
| IPsec Encryption Key | AES-CBC | 128/192/256 bits | This key is used to encrypt/decrypt the data | DRAM (plaintext) | Automatically when IPsec |

| Name | CSP Type | Size | Description/Usage | Storage | Zeroization |
|---|---|---|---|---|---|
| | | | throughout the IPsec session. This key is derived by key derivation function defined in SP800-135 KDF (IKEv2). | | session is terminated. |
| IPsec Authentication Key | HMAC-SHA256<br>HMAC-SHA384<br>HMAC-SHA512 | 256 bits<br>384 bits<br>512 bits | This key is used to protect the data integrity of data throughout the IPsec session. This key is derived by key derivation function defined in SP800-135 KDF (IKEv2). | DRAM (plaintext) | Automatically when IPsec session is terminated. |
| **SNMPv3 Keys and CSPs** | | | | | |
| SNMPv3 Passphrase | Shared Secret | 8-63 characters | Shared secret, used for SNMPv3 authentication. The key is configured by the Crypto Officer. | NVRAM (plaintext) | Procedurally erase the shared secret |
| SNMPv3 Authentication Key | HMAC-SHA-1 | 160 bits | This key is used to protect the data integrity of data throughout the SNMPv3 session. This key is derived by key derivation function defined in SP800-135 KDF (SNMPv3). | DRAM (plaintext) | Automatically when SNMPv3 session is terminated. |
| SNMPv3 Session key | AES-CFB-128 | 128 bits | This key is used to encrypt/decrypt the data throughout the SNMPv3 session. This key is derived by key derivation function defined in SP800-135 KDF (SNMPv3). | DRAM (plaintext) | Automatically when SNMPv3 session is terminated. |

# 7 Self-tests

The module performs the following power-up and conditional self-tests. Running power up self-tests does not involve action from the operator. Upon failure or a power-up or conditional self-test the module halts its operation and enters a quarantine state. The following tables describe each self-test implemented by the module.

**Table 9: Power-Up Self-Tests**

| Algorithm | Test |
|---|---|
| **Virtual SmartZone (vSZ) Crypto - Linux Kernel** | |
| AES | AES-CBC KATs (encryption/decryption) |
| HMAC | HMAC-SHA256/384/512 KATS |
| SHA | SHA-1/256/384/512 KATs |
| **Virtual SmartZone (vSZ) Crypto - OpenSSL/OpenSSH** | |
| AES | AES-CBC KATs (encryption/decryption) |
| AES-GCM | AES-GCM KATs (encryption/decryption) |
| SHS | SHA-1/256/512 KATs |
| HMAC | HMAC-SHA-1/224/256/384/512 KATs |
| SP800-90A DRBG | AES-256 CTR DRBG KAT (DRBG health tests per SP 800-90A Section 11.3) |
| RSA (FIPS 186-4) | RSA KATs (separate KAT for signing; separate KAT for verification) |
| KAS-FFC-SSC | KAS-FFC-SSC Primitive “Z” computation KAT |
| KAS-ECC-SSC | KAS-ECC-SSC Primitive “Z” computation KAT |
| ECDSA | ECDSA Pairwise Consistency Test (Sign and Verify) |
| Software Integrity Test | FIPS 186-2 RSA 4096 bits with SHA-384 for signature verification |

| | |
|---|---|
| SP800-135rev1 SSH-KDF | KAT for SSHv2 KDF |
| SP800-135rev1 TLS-KDF | KAT for TLSv1.2 KDF |
| SP800-135rev1 IKE-KDF | KAT for IKEv2 KDF |
| SP800-135rev1 SNMP-KDF | KAT for SNMPv3 KDF |

**Table 10: Conditional Self-Tests**

| Algorithm | Test |
|---|---|
| SP800-90A DRBG | Continuous Random Number Generator test |
| NDRNG | Continuous Random Number Generator test |
| RSA | Pairwise Consistency Test |
| ECDSA | Pairwise Consistency Test |
| Software Load Test | FIPS 186-2 RSA 4096 bits with SHA-384 for signature verification. |

# 8 Procedural Rules

The module meets all the Level 1 requirements for FIPS 140-2. The module is shipped only to authorized operators by the vendor. Follow the instructions provided below to place the module in FIPS-approved mode. Operating this module without maintaining the following settings prevents the module from being placed into FIPS approved mode of operation. The module was validated with software version 5.2.1.3 in FIPS-approved mode of operation.

The following procedural rules must be maintained by the operator in order to remain in the Approved mode.

- An operator shall zeroize all keys/CSPs when switching between the Approved and non-Approved mode (or vice versa).
- Approved key sizes are used by default, however the operator is capable of loading their own TLS certificates containing non-Approved RSA key lengths. Only Approved RSA key lengths specified in Table 3 shall be used.
- An operator shall not attempt to access the module’s BIOS. In particular, an operator shall not change the port configurations specified in Section 3 of this Security Policy.
- The module does not enforce a limit on the number of authentication attempts without first being configured to do so. The User and Cryptographic Officer shall have an authentication try limit configured between the range of 1-100.
- An operator shall not authorize access to the Diagnostics service while in the Approved mode.
- The module’s validation to FIPS 140-2 is no longer valid once a non-validated software version is loaded. Any software not identified in this Security Policy does not constitute the Module defined by this Security Policy or covered by this validation.

## 8.1 Module Initialization

The Crypto Officer shall follow the steps below to configure and initialize the module.

- Establish a command Line Interface (CLI) connection between the module and the CO management PC.
- Integrity Check: Before deployment, the image’s checksum should be checked and compared with the one announced on Ruckus website[1]

[1] https://support.ruckuswireless.com/software?format=coveo#q=virtual%20smartzone%205.2.1.3&layout=table

- Deployment: Deploy the Virtual SmartZone (vSZ) image on a hypervisor by following up the steps specified in section "vSZ Installation with FIPS Image" of "*RUCKUS FIPS and Common Criteria Configuration Guide for SmartZone and AP, 5.2.1.3"*, Published on 2021-04-14 with the documentation Part Number 800-72735-001 RevA https://support.ruckuswireless.com/documents/3509
- Controller Configuration with FIPS Image:
  - Power on the module and access the CLI via its virtual console port from hypervisor.
  - At the login prompt, login with the administrator username and password. And then, issue 'enable' (en) command with the privileged mode password to promote the authorization.
  - If the system is first time boot up, issue 'setup' command and follow the virtual console's instructions to configure the system fundamental parameters, such as the mode of FIPS, the network and change the default login and privileged mode passwords. The system will reboot when FIPS mode is being changed. The CO needs to make sure the passwords and all other shared secrets used by the module must each be at least eight (8) characters long, including at least one alphabet, one numeric character, one special character (note: The special character ` cannot be used in the password and the special characters combination '$(' cannot be used in the password).
  - At the command prompt enter 'fips?' to display the list of available FIPS commands.
  - Enter 'fips status' to verify whether FIPS mode is enabled or disabled. If the FIPS mode is enabled, user should be able to observe "FIPS compliance is Enable" from the console. On the other hand, if the FIPS mode is disabled, "FIPS compliance is Disable" will be shown on the console.
  - User can issue 'fips enable' or 'fips disable' to enable or disable FIPS mode, then enter 'yes' to confirm.
  - Configure RadSec Service (RADIUS over TLS). Please refer to Configuring RadSec section in "*RUCKUS FIPS and Common Criteria Configuration Guide for SmartZone and AP, 5.2.1.3"*, Published on 2021-04-14 with the documentation Part Number 800-72735-001 RevA https://support.ruckuswireless.com/documents/3509
  - Enter 'fips showlog' to display the results of self-tests and verify all are passing.
  - Follow steps in Section 8, above, while operating the module.

# 9 References

**Table 11: References**

| Reference | Specification |
|---|---|
| [ANS X9.31] | Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA) |
| [FIPS 140-2] | Security Requirements for Cryptographic modules, May 25, 2001 |
| [FIPS 180-4] | Secure Hash Standard (SHS) |
| [FIPS 186-2/4] | Digital Signature Standard |
| [FIPS 197] | Advanced Encryption Standard |
| [FIPS 198-1] | The Keyed-Hash Message Authentication Code (HMAC) |
| [FIPS 202] | SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions |
| [PKCS#1 v2.1] | RSA Cryptography Standard |
| [PKCS#5] | Password-Based Cryptography Standard |

| | |
|---|---|
| [PKCS#12] | Personal Information Exchange Syntax Standard |
| [SP 800-38A] | Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode |
| [SP 800-38B] | Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication |
| [SP 800-38C] | Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality |
| [SP 800-38D] | Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC |
| [SP 800-38F] | Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping |
| [SP 800-56A] | Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography |
| [SP 800-56B] | Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography |
| [SP 800-56C] | Recommendation for Key Derivation through Extraction-then-Expansion |
| [SP 800-67R1] | Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher |
| [SP 800-89] | Recommendation for Obtaining Assurances for Digital Signature Applications |
| [SP 800-90A] | Recommendation for Random Number Generation Using Deterministic Random Bit Generators |
| [SP 800-108] | Recommendation for Key Derivation Using Pseudorandom Functions |
| [SP 800-132] | Recommendation for Password-Based Key Derivation |
| [SP 800-135] | Recommendation for Existing Application –Specific Key Derivation Functions |