IBM® z/VM® Version 6 Release 3 System SSL Cryptographic Module

FIPS 140-2

Non-Proprietary Security Policy

Policy Version 1.05

IBM Systems & Technology Group
System z Development
Endicott, New York

IBM Research
Zurich Research Laboratory

March 27, 2014

Table of Contents

Scope of Document

This document describes the services that the IBM® z/VM® Version 6 Release 3 System SSL Cryptographic Module (""z/VM System SSL library" or "z/VM System SSL" or "System SSL" or "module") provides to security officers and end users, and the policy governing access to those services. It complements official product documentation, which concentrates on server usage of the functionality, as well as environmental set-up.

Module Description The z/VM System SSL library in its FIPS 140-2 configuration consists of a set of loadable 31-bit modules. The deployed version consists of the following modules:

Table 1 z/VM System SSL Library Modules

Core	Auxiliary
GSKCMS31	GSKSUS31
GSKC31F	Side Decks
GSKSSL	Message Catalogs
GSKS31F	
GSKKYMAN	

The z/VM System SSL library consists of the core modules that provide FIPS 140-2 approved services, as well as some auxiliary modules and files. The auxiliary modules provide functionality that is not cryptographically relevant. The files consist of side decks and message catalogs.

The z/VM System SSL logical and physical boundaries are described in Figure 1 in the Operational Environment Section.

Note: Throughout this document, the CP Assist for Cryptographic Functions will also be referenced using the terms CP Assist and CPACF.

Cryptographic Module Specification

The z/VM System SSL module is classified as a *multi-chip standalone Software-Hybrid module* for **FIPS Pub 140-2** purposes. The actual cryptographic boundary for this FIPS 140-2 module validation includes the System SSL module running in configurations backed by hardware cryptography. The System SSL module consists of software-based cryptographic algorithms, as well as symmetric and hashing algorithms provided by the CP Assist for Cryptographic Function (CPACF). (See Figure 2.)

System SSL validation was performed using the z/VM Version 6 Release 3 operating system with the following platform configuration:

1.IBM System z10™ Enterprise Class (z10 EC) with CP Assist for Cryptographic Functions DES/TDES Enablement Feature 3863

The module running on the above platforms was validated as meeting all **FIPS Pub 140-2** Level 1 security requirements. The z/VM System SSL module is packaged as a set of DLLs and executables which contains all the code for the module.

In addition to the configurations tested by the laboratory, vendor-affirmed testing was performed using z/VM Version 6 Release 3 on the following platforms:

2. IBM System z196™ with CP Assist for Cryptographic Functions DES/TDES Enablement Feature 3863

3. IBM System zEC12™ with CP Assist for Cryptographic Functions DES/TDES Enablement Feature 3863

Security level This document describes the security policy for the z/VM System SSL with Level 1 overall security as defined in **FIPS Pub 140-2** [1].

Table 2 System SSL Module Components

Type	Name
Software (DLLs and executables)	5735FAL00: z/VM Version 6 Release 3 plus APAR PM95516, which updates the System SSL binary to meet Power-On Self Test requirements.
Documentation	Not applicable
Hardware components	z10 CP Assist for Cryptographic Functions DES/TDES Enablement Feature 3863

Cryptographic Module Security Level

The module is intended to meet requirements of Security Level 1 overall, with certain categories of security requirements not applicable (Table 3).

Table 3 Module Security Level Specification

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

Ports and Interfaces

As a multi-chip standalone Software-Hybrid module, the System SSL physical interfaces are the boundaries of the host running System SSL library code. The underlying logical interfaces of the module are self-controlled. Control inputs which control the mode of the module are provided. Data input and data output are provided in the variables passed through user-supplied buffers. Status output is provided in return codes and through messages.

Cryptographic bypass capability is not supported by System SSL.

Module Status The System SSL library communicates any error status synchronously through the use of its return codes. It is the responsibility of the calling application to handle exceptional conditions in a FIPS 140-2 appropriate manner.

System SSL is optimized for library use and does not contain any terminating assertions or exceptions. Any internal error detected by System SSL and not induced by user data will be reflected back to the application with an appropriate return code. The calling application must examine the return code and act in a FIPS 140-2 appropriate manner to such failures and reflect this error in a fashion consistent with this application.

User-induced or internal errors do not reveal any sensitive material to callers.

Roles, Services and Authentication

Roles

The module supports two roles: a cryptographic officer (Officer) role and a User role (Table 4). The module does not support user identification or authentication that would allow the module to distinguish between the two supported roles. Each of the roles is authenticated through the operating system implicitly prior to using any system services.

The Officer role is a purely administrative role that does not involve the use of cryptographic services. The role is not explicitly authenticated but assumed implicitly on implementation of the module’s installation and usage sections defined in the security rules section.

The User role has access to all of the module’s services. The role is not explicitly authenticated, but assumed implicitly on access of any of the non-Officer services. An operator is implicitly in the User or Officer role based upon the service(s) chosen. If any of the User-specific services are called, then the operator is in the User role; otherwise the operator is in the Officer role.

Table 4 Roles and Authentication Mechanisms

Role	Type of Authentication	Authentication Data	Strength of Mechanism
Officer	None(Automatic)	None	N/A
User	None(Automatic)	None	N/A

Services

The module provides queries (Table 6) and commands. Queries return status of commands or command groups; commands exercise cryptographic functions. Officers perform queries; Users may perform both queries and commands. While most test queries are not executed automatically as part of regular operations, certain test queries are executed automatically; these special cases are parenthesized as (yes) in Table 6.

Services are accessed only by an authorized calling application. These applications are defined in Section 6.

Certificate management services (CMS) perform both non-cryptographic and cryptographic PKI management activities, as well as general cryptographic operations, such as signature verification. Functions in this group parse and categorize X.509 certificates and transport certificates, and also handle standard encodings (such as PKCS#7). Cryptographic operations, such as signature verification, are delegated to lower-level crypto core functions.

SSL protocol implementation is split into infrastructure and protocol functions. Lower-level functions implement SSL message formatting and other primitives. SSL protocol operations extend SSL primitives with handshake state machines, session caching, and attribute parsing to provide a full SSL/TLS implementation. Both System SSL layers use cryptographic cores indirectly. SSL 3.0 functionality is disallowed by FIPS 140-2: all other compliance checks are implemented at lower levels. Cipher suites are restricted to those built with approved algorithms only.

Format conversions, labeled as “other operations", are other non-cryptographic commands that change the representation of data. Format converters read and write, among others, the following formats:

- Various protocols based on ASN.1/BER encoded data (PKI-related and similar standard formats)
- Conversions between industry-standard object identifiers and internal symbolic constants (mainly intrinsic, not externalized).

Protocol-level format conversions generally package data without modification, treating output or input of lower-level crypto primitives as opaque data. The purpose of these conversions is to bridge protocols with predefined formats with cryptographic primitives, which are oriented around raw byte streams or blocks, but generally not standard encapsulation methods:

- Base-64 encoding (“ASCII armor"), generating and reading printable representation of binary data, for example, encountered in certificates
- Conversions between ASCII and non-ASCII data (such as EBCDIC), non-cryptographic but potentially modifying security-relevant data.

Format conversion services do not provide cryptographic functionality, but may use other services if the transport mechanism requires them. As an example, if signed data is represented as a standard ASN.1 structure, it implicitly uses one of the sign calls. Similarly, certificate management or processing of PKCS#7 data may involve signature verifications, for example.

Table 5 Services and Access

Service	**Notes**	**Is FIPS-Approved? If yes, Cert #**	**CSPs and their Access**	
Software				
Symmetric Algorithms				
AES CBC •Encryption •Decryption	Compliant to FIPS 197, FIPS SP 800-38A *Input*: plaintext, IV, key *Output*: ciphertext *Input*: ciphertext, IV, key *Output*: plaintext	Yes Cert. #2627	128 or 256 bit AES keys	Read
Triple-DES CBC •Encryption •Decryption		Yes Cert. #1577	168-bit Triple-DES keys	
DSA Key/Parameter Generation[1]	Compliant to FIPS 186-2 *Input*: supported modulus size 1024-bit *Output*: domain parameters, p, q, g, k private key x and public key y pair	Yes Cert. #792	DSA public and private key pair: L=1024, N=160	Write
DSA Sign[1]	Compliant to FIPS 186-2 *Input*: message m, domain parameters p, q, g, k and sender's private key x *Output*: signature of m as a pair of r and s	Yes Cert. #792	DSA private key L= 1024, N = 160	Read
DSA Verify[2]	Compliant to FIPS 186-2 *Input*: domain parameters p, q, g, k and sender's public key y received message m', and signature in form of r' and s' pair *Output*: pass if the signature is valid fail if the signature is invalid	Yes Cert. #792	DSA public key L = 1024, N = 160	Read
RSA Key Generation[1]	Compliant to ANSI X9.31 *Input*: modulus size, the public key random numbers: X_{p1}, X_{p2}, X_{q1} and X_{q2} *Output*: the private prime factor p, the private prime factor q the value of the modulus n the value of the private signature exponent d	Yes Cert. #1344	RSA public and private key pair with modulus sizes 1024, 2048, 3072, and public key value 65,537.	Write
RSA Sign[1]	Compliant to PKCS#1 *Input*: the module n, the private key d the SHA algorithm (SHA-1/SHA-224/SHA-256/SHA-384/SHA-512) a message m to be signed *Output*: the signature s of the message	Yes Cert #1344	RSA private key with modulus sizes 1024, 2048, 3072, 4096	Read

Service	Notes	Is FIPS-Approved? If yes, Cert #	CSPs and their Access	
RSA Verify[2]	Compliant to PKCS#1 Input: the module n, the public key e the SHA algorithm (SHA-1/SHA -224/SHA-256/SHA-384/SHA-512) a message m a signature for the message Output: pass if the signature is valid fail if the signature is invalid	Yes Cert. #1344	RSA public key	Read
RSA Wrapping[1] RSA Unwrapping[1]	PKCS#1 Input: key to be wrapped RSA public key Output: encrypted key using RSA public key Input: encrypted key using RSA public key RSA private key Output: key in plaintext	No	RSA public and private key pair with modulus sizes 1024, 2048, 3072, 4096	Read
Diffie-Hellman (DH)	NIST SP 800-56A Input: prime number (p), base (g), secret integers(a,b), Output: shared secret	No	DH public and private key pair with 2048-bit modulus	Read/ write
Hash Functions				
SHA-1[3] SHA-224 SHA-256 SHA-384 SHA-512	Compliant to FIPS 180-4 *Input*: message *Output*: message digest	Yes Cert. #2203	None	N/A
Message Authentication Codes (MACs)				
HMAC-SHA1 HMAC-SHA256 HMAC-SHA384	Compliant to FIPS 198-1 *Input*: HMAC key message *Output*: HMAC value of message	Yes Cert. #1624	HMAC key with minimum length 112 bits	Read
HMAC-MD5[4]	Used as a pseudo-random function	No	HMAC-MD5 key	Read
Random Number Generation				
RNG[5]	FIPS 186-2 *Input*: seed and seed key *Output*: random bits	Yes Cert. #1241	Seed Seed key RNG output	Read Write
TRNG	Non-deterministic RNG to provide suitable entropy for the approved FIPS 186-2 RNG	No		Read
Key Derivation Function				
TLS 1.0/1.1/1.2	Compliant to SP 800-135 *Input*: pre-master secrets *Output*: derived key	Yes Cert # 110		Read
CP Assist for Cryptographic functions				
Symmetric Algorithms				

Service	Notes	Is FIPS-Approved? If yes, Cert #	CSPs and their Access	
AES CBC	Compliant to FIPS 197, FIPS SP 800-38A *Input*: plaintext, IV, key *Output*: ciphertext *Input*: ciphertext, IV, key *Output*: plaintext	Yes Cert. #976	128 or 256 bit AES keys	Read
Triple-DES CBC	Compliant to FIPS SP 800-67, FIPS SP 800-38A *Input*: plaintext, IV, key *Output*: ciphertext *Input* ciphertext, IV, key *Output*: plaintext	Yes Cert. #769	168-bit Triple-DES keys	Read
Hash Functions				
SHA-1[3] SHA-224 SHA-256 SHA-384 SHA-512	Compliant to FIPS 180-4 *Input*: message *Output*: message digest	Yes Cert. #946	None	N/A

Notes:
1.Per NIST SP 800-131a recommendations, use of RSA or DSA private keys less than 2048 bits in length for digital signature generation is not allowed and should not be used after 2013.
2.Per NIST SP 800-131a recommendations, use of RSA or DSA private keys less than 2048 vits in length for digital signal verification is supported in legacy mode only after 2013.
3.Per NIST SP 800-131a recommendations, use of SHA-1 for RSA or DSA signature generation is not allowed and should not be used after 2013.
4.HMAC-MD5 is allowed to be used in the FIPS mode in the context of TLS 1.0 and 1.1 as part of pseudorandom function.
5.Per NIST SP 800-131a recommendations, RNG based on X9.31 standards (FIPS 186-2) is deprecated after 2013 and should not be used after 2015.

Table 6 Queries and other services

Service	**Notes**	**Roles**	
Module Status		**Officer**	**User**
Query mode	Check if module is in FIPS 140-2 mode	Yes	Yes
Integrity Checks			
Power-up Tests	Automatic before first use	(yes)	No
Self-Tests	DLL verification self-test. Self-tests can be done by stopping and restarting the appropriate server.	Yes	Yes
Operational Correctness Checks			
RNG Tests	Continuously performed (automatic)	Yes	Yes
Pair-wise consistency	Continuously performed (automatic)	Yes	Yes
Other services			
Installation and configuration		Yes	No

Operational Environment

Installation and Invocation

System SSL is installed as part of the z/VM Version 6 Release 3 SDO. The evaluated version of System SSL additionally requires APAR PM95516..

The cryptographic modules are invoked via specific IBM programs.   All other programs attempting to access the cryptographic modules will abend. These authorized programs are the z/VM SSL Server, the z/VM LDAP server and client utilities, the gskkyman utility (certificate management) and the gsktrace utility (debugging utility).

Module Operation

The System SSL security module is written in C, with certain functionality contained within assembler, such as functions that utilize the CPACF. Extensive internal consistency checks verify both user input and library configuration, terminating early if errors are encountered. Internal errors are externalized and do not terminate execution, since the code has been developed mainly for library use.

Since System SSL can access certain platform-specific functionality, which is not represented in higher-level languages, System SSL uses a mixture of high-level and platform-specific native code.

Using z/VM System SSL in a FIPS 140-2 approved manner assumes that the following defined criteria are followed:

- The Operating System enforces authentication method(s) to prevent unauthorized access to Module services.

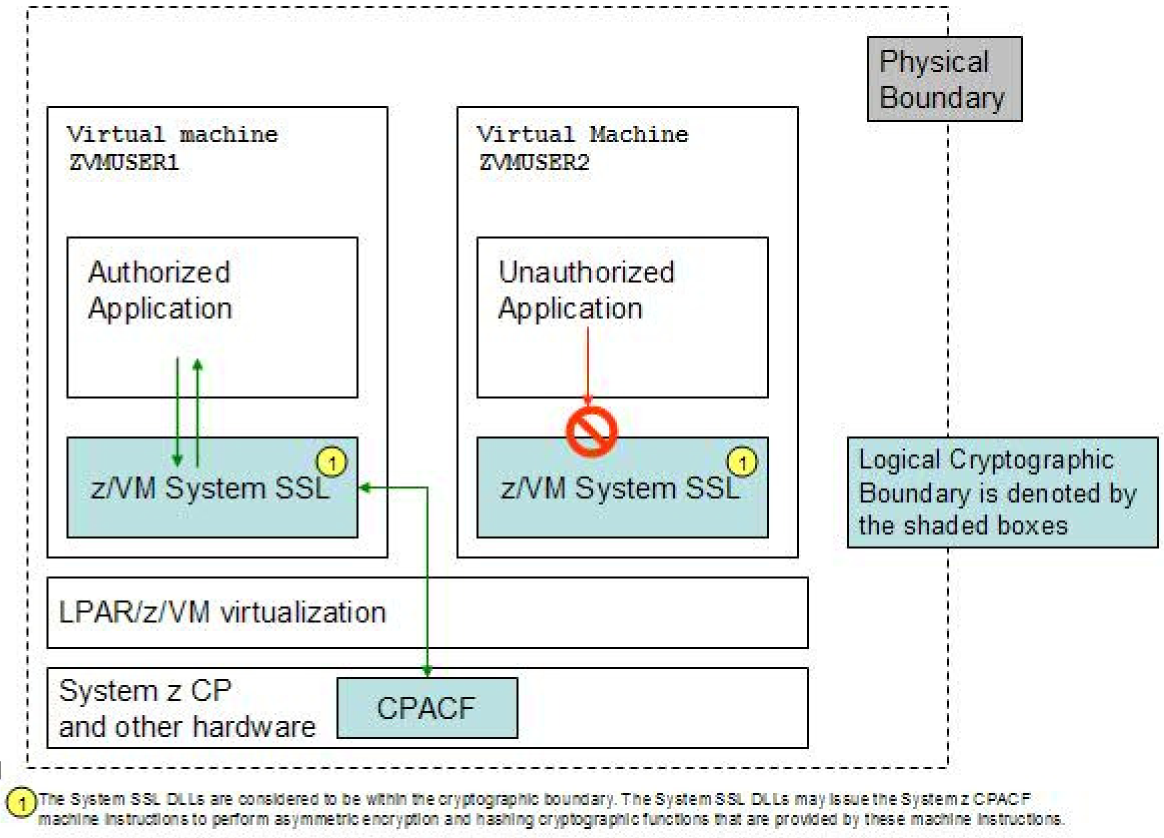
- All host system components that can contain sensitive cryptographic data (main memory, system bus, disk storage) must be located within a secure environment.
- The application using the module services must consist of one or more processes in which each process is utilizing a separate copy of the executable code.
- The unauthorized reading, writing or modification of the virtual machine which contains the System SSL instance is not allowed.
- An instance of the System SSL Library DLLs must be accessed only by a single process (virtual machine). This means that each process has it own instance of the System SSL Library DLLs.
- The System SSL module must be configured to execute in the FIPS 140-2 mode of operation.
- The CP Assist for Cryptographic Functions DES/TDES Enablement Feature 3863 must be installed and enabled.

This module has only a FIPS 140-2 mode of operation. The module allows approved algorithms to be used in addition to RSA/Diffie-Hellman for key establishment and exchange; however, the module does not enforce a minimum key strength of 112 bits. The configuration automatically inhibits certain parameter combinations that are technically possible, but not permitted by FIPS 140-2. In cases where module prohibition is not enforced, the user of the module must not utilize those algorithms which are disallowed by FIPS 140-2.

The System SSL DLLs and gskkyman certificate utility represent the logical boundary of the module. The physical cryptographic boundary for the module is defined as the enclosure of the host on which the cryptographic module is to be executed.

As shown in Figure 1, System SSL Cryptographic Module, the cryptographic module's DLLs are instantiated within an application's virtual machine. Each operating system component that utilizes the System SSL cryptographic module will create a new instance of the z/VM System SSL DLLs.

Figure 1: System SSL Cryptographic Module



The System SSL DLLs are considered to be within the cryptographic boundary. The System SSL DLLs may issue the System z CPACF machine instructions to perform asymmetric encryption and hashing cryptographic functions that are provided by these machine instructions.

Key Management

Key Storage The System SSL library retains key material within its virtual machine. In a typical SSL/TLS setting, private keys would be imported from a different key store. Public keys (certificates) would be distributed through other channels, such as out-of-band PKI messages.

The module provides key import and export routines to applications such that key material can be used in conjunction with cryptographic services. It is the responsibility of applications using library services to ensure that these services are used in a FIPS 140-2 compliant manner. Keys managed or generated by applications or libraries may be passed from applications to the module in the clear, provided that the sending application or library exists within the physical boundary of the host computer.

Key material resides in memory as clear data or in a standard key store format. The most frequently used standard formats, using passphrase-derived keys such as PKCS#12, are classified as clear-key storage according to **FIPS Pub 140-2** guidelines.

Key Generation Key Generation uses an approved RNG algorithm (specified in **FIPS Pub 186-2**) which is based on SHA-1. The DRNG has a maximum number of internal states of 2^{160}, this maximum number reflecting the limitation of the compression function in SHA-1. RSA DSA and Diffie-Hellman key generation algorithms use the DRNG engine seeded with 20 bytes of true random data. This true random number generator extracts entropy from time measurement jitter (minute variations of clock edges). The internal TRNG engine feeds entropy on demand into the DRNG; the TRNG itself maintains a running pool of samples, and provides seed if the pool passes basic entropy content checks. The statistical check on the entropy content guarantees that the entropy pool has at least 0.75 entropy per bit. Therefore, the DRNG's internal state has at least 120 bits entropy.

DSA key generation is done according to **FIPS Pub 186-2**. RSA key generation only implements the ANSI X9.31 key generation method [2].

Caveat: The module generates cryptographic keys whose strengths are modified by available entropy. In particular, when operating in FIPS mode, 4096-bit RSA keys generated by the module may have slightly less than 150-bit of security strength.

Key Establishment The module provides support for asymmetric key establishment methods as allowed by Annex D in the **FIPS Pub 140-2**. The supported asymmetric key establishment methods are RSA Key Wrapping and Diffie-Hellman (DH) key agreement.

When using Diffie-Hellman in FIPS 140-2 mode, the allowed modulus length is 2048 bits, which provides 112 bits of encryption strength.

When using RSA Encrypt/decrypt in FIPS 140-2 mode, the allowed modulus lengths must be between 2048 and 4096 bits which provides between 112 and 150 bits of encryption strength. Note that **NIST SP 800-131a** requires a modulus length of at least 2048 bits, which provides at least 112 bits of encryption strength.

Key Entry and Key Exit The module does not support manual key entry or intermediate key generation key output.

The module does not output or input keys outside of the physical boundary, with the exception of secret keys that are used for key establishment. The secret keys are wrapped with RSA.

Key Protection To enforce compliance with **FIPS Pub 140-2** key management requirements on the System SSL library itself, code issuing calls must manage keys in a **FIPS Pub 140-2** compliant method. Keys managed or generated by applications may be passed from the application to the module in the clear in the **FIPS Pub 140-2** validated configuration.

The management and allocation of memory is the responsibility of the operating system. It is assumed that a unique process is allocated for each request, and that the operating system and the underlying hardware control access to the virtual machine which contains the process that uses the module. Each instance of the cryptographic module is self-contained within a process; the library relies on such process separation and address separation to maintain confidentiality of secrets. All platforms used during **FIPS Pub 140-2** validation provide per-process protection for user data. Keys stored internally within the address range of System SSL are similarly separated logically (even if they reside in the same virtual machine).

All keys are associated with the User role. It is the responsibility of application program developers to protect keys exported from the System SSL module.

Key Destruction Applications must destroy persistent key objects and similar sensitive information using **FIPS Pub 140-2** compliant procedures. The System SSL library itself does not destroy externally stored keys and secrets, as it does not own or discard persistent objects. Objects, when released on behalf of a caller, are erased before they are released.

Physical Security

The System SSL installation inherits the physical characteristics of the host running it. The System SSL library has no physical security characteristics of its own.

The CP Assist for Cryptographic Function (CPACF) offers the full complement of the Triple-DES algorithm, Advanced Encryption Standard (AES) algorithm and Secure Hash Algorithm (SHA).

CPACF Physical Design: Each two microprocessors (cores) on the quad-core chip share a Co-Processor Unit (CoP), which implements the crypto instructions and also provides the hardware compression function. The compression unit is integrated with the CP Assist for Cryptographic Function (CPACF), benefiting from combining (sharing) the use of buffers and interfaces. The CoP is located on the processor die and is connected to two cores and to L2 cache with dedicated buses.

The CP Assist for Cryptographic Function (CPACF) accelerates the encrypting and decrypting of SSL transactions and VPN-encrypted data transfers and data-storing applications. The assist function uses a special instruction set for symmetrical clear key cryptographic encryption and decryption operations. Five special instructions are used with the cryptographic assist function.

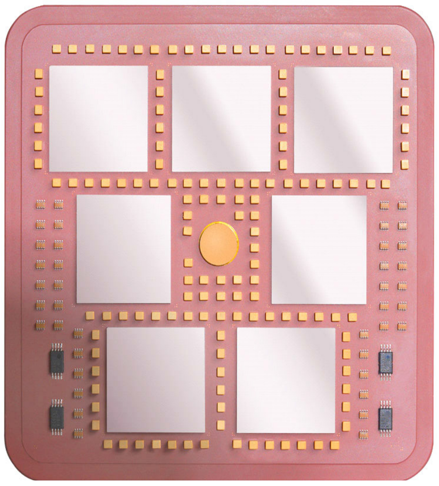


Figure 2: z10 EC Processor Unit MCM with CPACF COP chips

Figure 3: IBM System z10 EC Mainframe Computer.

EMI/EMC

EMI/EMC properties of System SSL are not meaningful for the library itself. Systems utilizing the System SSL library services have their overall EMI/EMC ratings determined by the host system. The validation environments have FCC Class A ratings.

Self-Tests

The System SSL library implements a number of self-tests to check proper functioning of the module including power-up self-tests and conditional self-tests. Conditional tests are performed when symmetric or asymmetric keys are generated. These tests include a continuous random number generator test and pair-wise consistency tests of the generated DSA or RSA keys.

Startup Self-Tests “Power-up" self-tests consist of software integrity test(s) and known-answer tests of algorithm implementations listed below. The module integrity test is automatically performed during loading. If any of these tests fail, the module will terminate the loading process. The module cannot be used in this state.

The integrity of the module is verified by checking a HMAC-SHA-256-based hash value of each module binary prior to being utilized. Initialization will only succeed if all utilized module hash values are verified successfully. Module hash values are generated during the final phase of the build process.

All the power-up self-tests are performed automatically without any operator intervention when a system administrator configures System SSL to auto-load in FIPS mode by setting the environment variable GSK_DEFAULT_FIPS_STATE to 'GSK_FIPS_STATE_ON'. On successful completion of the power-up self-tests, the module enters a FIPS approved mode. All cryptographic services are available only in the FIPS mode.

The module performs the following power-up self-tests :
- Integrity test using HMAC-SHA-256
- AES (KATs for encryption/decryption tested separately)
- Triple-DES (KATs for encryption/decryption tested separately)
- RSA (KATS for sign/verify, wrapping/unwrapping tested separately),
- DSA (a pair-wise consistency test for sign/verify is performed as a part of power-on self-test),
- SHA-1,SHA-224,SHA-256, SHA-384, SHA-512 (Separate KAT for each SHA),
- HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 (Separate KAT for each HMAC-SHA)
- RNG (KAT)

Self-tests are performed in logical order, verifying library integrity incrementally:
1. Integrity test on library, using HMAC-SHA-256
2. Known-answer tests on algorithms, from integrity-verified binary.

The integrity check process covers all constituent DLLs. DLLs are individually hashed and verified.

Startup Recovery If any of the start up self-tests fail, System SSL will terminate the loading of the module needed for the FIPS 140-2 processing.

Conditional Self-Test Conditional self-testing includes continuous RNG test and a pairwise consistency test for RSA/DSA .
Continuous RNG testing involves comparing every newly-generated RNG block with the previously-generated one. The first output block generated by RNG is used only for the purpose of initiating the continuous RNG test. The test fails if the RNG outputs the same value twice subsequently.

If the RNG outputs identical, subsequent pseudo-random blocks, it enters an error state and returns the corresponding status. The calling application must recognize this error and handle it in a FIPS 140-2 appropriate manner, for example, by reinitializing the library instance.

Similar to the RNG, high-entropy seed extracted by the TRNG is checked for repeated blocks, before seeding the RNG. If blocks of entropy repeat, the TRNG reports a failure, which caller applications must also handle as an error.

Pair-wise Consistency Checks This test is run whenever the module generates a private-public key-pair. The private key structure always contains either the data of the corresponding public key or information sufficient for computing the corresponding public key.

If the pair-wise consistency check fails, the module enters an error state and returns an error status code. The calling application must recognize this error and handle it in a FIPS 140-2 appropriate manner, for example, by reinitializing the library instance.

Invoking FIPS 140-2 self-tests on demand. If a user can access System SSL services, the library has passed its integrity and power-up self tests. On-demand self-tests can be invoked by reloading the module.

If a KAT failure is encountered, the module enters an error state and returns an error status code. The calling application must recognize this error and handle it in a FIPS 140-2 appropriate manner, for example, by reinitializing the library instance. The error state is indicated by a return code describing the error. No cryptographic services are available in the error state.

Operational Requirements (Officer/User Guidance)

Module Configuration for FIPS 140-2 Compliance

To verify FIPS 140-2 compliant usage, the following requirements must be observed:

•The Operating System (OS) hosting the library must be set up in accordance with **FIPS Pub 140-2** rules. It must provide sufficient separation between processes to prevent inadvertent access to data of different processes. (This requirement was met for all platforms tested during validation.)

•An instance of the module must not be used by multiple callers simultaneously such that they might interfere with each other. Note that for keys retained in caller-provided storage, this requirement is automatically met if the OS provides sufficient process separation (since the ownership of each memory region, therefore, each object, is uniquely determined.)

•Applications using System SSL services must verify that ownership of keys is not compromised, and keys are not shared between different users of the calling application.

Note that this requirement is not enforced by the System SSL library itself, but by the application providing the keys to System SSL.

•Applications utilizing System SSL services must avoid using key sizes and algorithms that are disallowed after the transition period as stated in the NIST SP800-131A.

•To be in a FIPS 140-2 compliant state, the System SSL installation must run on a host with commercial grade components and must be physically protected as prudent in an enterprise environment.

- **Physical assumptions**
  - The module is intended for application in user areas that have physical control and monitoring.

It is assumed that the following physical conditions will exist:
- **LOCATION**
  - The processing resources of the module will be located within controlled access facilities that will prevent unauthorized physical access.
- **PROTECTION**
  - The module hardware and software critical to security policy enforcement will be protected from unauthorized physical modification.

- **Personnel assumptions**
  - It is assumed that the following personnel conditions will exist:
    - **MANAGE**
      - There will be one or more competent individuals assigned to manage the module and the security of the information it contains.
    - **NO EVIL ADMINISTRATOR**
      - The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the administrator documentation.
    - **CO-OPERATION**
      - Authorized users possess the necessary authorization to access at least some of the information managed by the module and are expected to act in a cooperative manner in a benign environment.

Determining Mode of Operation

The module provides a query function which will start indicating FIPS 140-2 mode after all self tests are successfully completed.

Applications utilizing services must enforce key management compliant with **FIPS Pub 140-2** requirements. This should be indicated in an application-specific way that is directly observable by administrators and end-users.

While such application-specific details are outside the scope of the validation, they are mentioned here for completeness.

The module automatically restricts algorithms usage to approved or allowed algorithms and inhibits parameter combinations that are technically legal but outside standardized range (such as nonstandard DSA key sizes, short HMAC keys, etc.), except in cases where the minimum key sizes of previously approved algorithms have been deprecated or disallowed. Those parameter combinations (such as nonstandard RSA key sizes) must not be exercised by the user of the module.

Testing/Physical Security Inspection Recommendations

In addition to automatic tests, which are described elsewhere in this document, System SSL users may invoke FIPS 140-2 mode self-tests at any time. These self-tests are initiated through a dedicated function which is invoked automatically at startup. Continuous tests reside within their respective functions and are called implicitly during the function processing. These tests are not observable unless a failure is detected.

Apart from prudent security practice of server applications and those of security-critical embedded systems, no further restrictions are placed on hosts utilizing these services.

Mitigation of Other Attacks

The Mitigation of Other attacks security section of FIPS 140-2 is not applicable to the System SSL cryptographic module.

Glossary

CP	Control Program. A component of z/VM that manages the resources of a single computer so that multiple computing systems appear to exist. Each apparent system, or virtual machine, is the functional equivalent of the real computer, and CP simulates the real machine architecture in the virtual machine.
CPACF	CP Assist for Cryptographic Function, clear key on-chip accelerator integrated into mainframe processors. CPACF functionality is restricted to symmetric and hashing operations.
DLL	Dynamic Link Library, shared program library instantiated separately from binaries using it. FIPS 140-2 configurations of System SSL DLLs are never statically linked.
DRNG	Deterministic Random Number Generator, a deterministic function expanding a “true random” seed to a pseudo-random sequence.
Enclave	In the z/VM Language Environment, a collection of routines, one of which is named as the main routine. The enclave contains at least one thread. Multiple enclaves may be contained within a process.
KAT	Known Answer Test
OS	Operating System
Process	A collection of resources; both program code and data, consisting of at least one enclave.
SDO	Prepackaged version of the z/VM Operating System
Side deck	The functions and variables that can be imported by DLL applications.
Thread	An execution construct that consists of synchronous invocations and terminations of routines. The thread is the basic run_time path within the z/VM Language Environment program management model, and is dispatched by the operating system with its own run-time stack, instruction counter and registers. A thread may exist concurrently with other threads within a virtual machine.
TRNG	True Random Number Generator, a service that extracts cryptographically-useful random bits from non-deterministic (physical) sources. The “random seed” bits are post-processed by a DRNG.
Virtual Device	The simulation of a device by CP.
Virtual Machine	The virtual processors, virtual storage, and virtual devices that CP allocates to a single

	user. A virtual machine also includes any expanded storage dedicated to it.
Virtual Processor	A representation of a processor that is dispatched by CP on a real processor. It includes the contents of all registers and the state of the processor.
Virtual Storage	Storage space that can be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses.

References

[1] National Institute of Standards and Technology, Security Requirements for Cryptographic Modules (FIPS 140-2), 2002

[2] American National Standard Institute, Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (X9.31), 1998

[3] National Institute of Standards and Technology, Digital Signature Standard (DSS) (FIPS 186-2), 2000

[4] National Institute of Standards and Technology, Digital Signature Standard (DSS) (FIPS 186-3), 2009

[5] National Institute of Standards and Technology, Secure Hash Standard (FIPS 180-4), 2012

[6] National Institute of Standards and Technology, Special Publication 800-131a: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, 2011

[7] National Institute of Standards and Technology, Advanced Encryption Standard (FIPS 197), 2001

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:
•IBM
•z10
•z196
•zEC12
•z/VM