Voltage IBE Cryptographic Module Version 4.0

Security Policy

Document Version 1.0

Voltage Security, Inc.

Revision History

The following table presents the history of changes to this document.

Document History

Date	Version	Changes
02-04-2012	1.0	Initial draft for public release.

TABLE OF CONTENTS

1. Module Overview

The Voltage IBE Cryptographic Module Version 4.0 is a FIPS 140-2 Level 1 compliant software module, which is also referred to by the acronym VIBECM, or the capitalized word Module. The Voltage IBE Developers Toolkit product includes the VIBECM along with supporting documentation. The VIBECM is a software-only cryptographic module packaged as a single run-time library (vibecryptofips.dll on a Windows system or vibecryptofips.so on a Unix system) and is supported on the operating systems listed in Table 1. The cryptographic boundary is defined by the single run-time library file and the signature file.

Table 1: Supported Operating Systems

Operating System	Version	Word Size
Windows 7 Professional	SP1	32-bit
Red Hat Enterprise Linux Server	5.3	32-bit

The primary purpose for this cryptographic module is to provide encrypt/decrypt and cryptographic signature services for Internet Protocol (IP) traffic.

The VIBECM provides status output via the "Show Status" service. The VIBECM provides program interfaces for data input and output. The diagram below illustrates these interfaces as well as defining the cryptographic boundary.

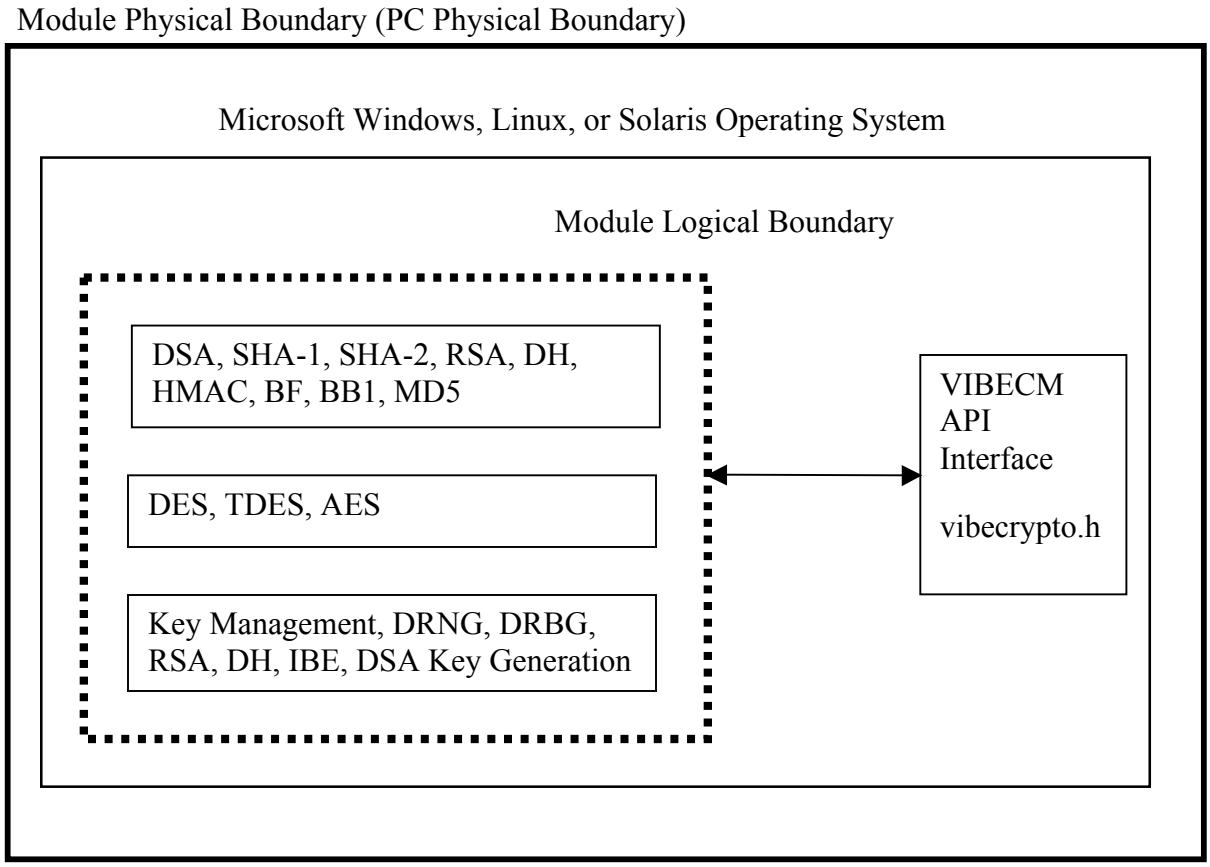

Figure 1 – Image of the Cryptographic Module

The physical cryptographic boundary for the Module is defined as the enclosure of the computer system on which the cryptographic module is to be executed. The physical embodiment of the Module, as defined in FIPS 140-2, is Multi-Chip Standalone.

Persistent storage of keys is not supported by the VIBECM.

2. Security Level

The VIBECM meets the overall requirements applicable to Level 1 security of FIPS 140-2.

Table 2: Module Security Level Specification

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

3. Modes of Operation

The VIBECM supports two modes of operation; a FIPS approved mode and a non-Approved mode. The Approved mode of operation is invoked by creating a FIPS Library context, which can be done through the function, VtCreateLibCtxFips.

3.1 FIPS approved mode of operation

In FIPS mode, the VIBECM supports FIPS Approved algorithms as follows:

Table 3: FIPS Approved Algorithms with Modes of Operation

Algorithm	Modes of Operation	Certificate #
DSA with 1024 bit keys	Sign/Verify	#547
TDES – 3 key mode	ECB, CBC, OFB, CFB	#1135
SHS (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)	Byte Oriented	#1539
AES - 128 and 192 and 256 key sizes are supported	ECB, CBC, OFB, CFB	#1752
DRNG – FIPS 186-2	X-Change Notice (SHA-1) K-Change Notice (SHA-1)	#934
HMAC – 112, 128, 192 and 256 bit keys are supported	Byte Oriented	#1029
RSA	Sign/Verify	#871
DRBG	Hash-DRBG	#115

In FIPS mode, the VIBECM also supports non-FIPS Approved algorithms in limited uses as follows:

Table 4: Non-FIPS Approved Algorithms Available in FIPS mode.

Algorithm	Limitations to Use
RSA (1024-bit and 2048-bit keys) (key wrapping; key establishment methodology provides 80 or 112 bits of encryption strength)	Use limited to the encryption and decryption of symmetric keys
MD5	Use limited to that required for generation of TLS keys

3.2 Non-Approved mode (non-FIPS mode)

In addition to all of the algorithms available in Approved mode, the VIBECM also supports the use of the additional algorithms listed in Table 5 in non-Approved mode.

Table 5: Non-FIPS Approved Algorithms Available in non-Approved Mode.

Algorithm	Limitations to Use
BF IBE, BB1 IBE	Identity-based encryption public-key algorithms – only available in non-Approved mode
AES FFX (128-, 192- and 256-bit keys)	Format-preserving mode of AES – only available in non-Approved mode
Diffie-Hellman (DLC primitive only)	Only available in non-Approved mode
DES	Only available in non-Approved mode

The VIBECM includes a deterministic random number generator (DRNG) that is compliant with FIPS 186-2 with 256-bit XKEY and underlying G function constructed from SHA-1 for generation of all cryptographic keys.

FIPS-186-2 requires that the DRNG for DSA X values is slightly different from the algorithm for DSA K values (Appendix 3.1 and 3.2 respectively). The VIBECM implements both of these algorithms and they are used appropriately.

The VIBECM also includes a deterministic random bit generator that is compliant with SP 800-90. This DRBG only operates in Hash-DRBG mode.

To operate the VIBECM in the FIPS approved mode, operators must run VIBECM on one of the operating systems specified in Table 1 and access only the service listed in Table 6 below.

4. Ports and Interfaces

The logical interface of the Module is accessed through the API (VIBECM API) as defined in the header file vibecrypto.h.

Control Input is provided by the API function calls, which represent the services provided by the Module. Data Input is provided by the variables passed with the function calls. These variables are passed on the program stack either directly on the stack or as a pointer on the stack that points to memory allocated in a heap. Both stack and heap are located in RAM. Data Output is provided by variables returned from a function call. As with Data Input, these variables are located either on the program stack or in a heap. The Status Output is provided in the return values and error codes provided by a function.

All data output is inhibited during the self-test process, and during key generation.

Only limited data processing will be allowed when the module is in a FIPS error state. During this limited processing no cryptographic operations are allowed. The only FIPS services available during limited process are Zeroize and Show Status. The only operations available during limited processing that output data provide Base64 encoding and decoding. There is no output of any critical security parameters during limited data processing. The only way to reset the Module from this limited processing state is to unload the Module or to power down the computer.

5. Identification and Authentication Policy

This section describes the identification and authentication policy of the Module.

The VIBECM supports two distinct operator roles (User and Cryptographic-Officer).

The User role provides the basic services to process data (encryption, decryption, and key management), whereas the Crypto Officer role provides the services to perform integrity checking self-tests, and zeroize.

VIBECM does not support a Maintenance role.

The role of the operator of VIBECM is identified implicitly on the library function being called, as shown in Table 6 in the next section. There is no operator authentication.

6. Access Control Policy

This section describes the access control policy of the Module.

6.1 Roles and Services

The services available to each role are described in the following table.

Table 6: Services Authorized for Roles

Role	Authorized Services
User This role shall provide all of the services necessary to: • Examine and set the attributes of the Voltage IBE Cryptographic Module. • Support data encryption and decryption operations. • Compute hashes • Create and verify digital signatures. • Generate DSA, RSA key pairs • Compute HMAC message authentication code	• Create Algorithm Object: Creates a new algorithm object. • Destroy Algorithm Object: Destroys an algorithm object. • Create Random Object: Creates a new random number generator object. • Destroy Random Object: Destroys a random number generator object. • Create Key Object: Creates a new key object. • Destroy Key Object: Destroys a key object. • Set Key Object: Sets the key object with information. • Get Key Info: Returns key information. • Create Parameter Object: Creates a new parameter object. • Destroy Parameter Object: Destroys a parameter object. • Set Parameter Object: Sets a parameter object with information. • Get Parameter Info: Returns parameter information. • Generate Parameters: Generates DSA parameters. • Digest Data: Initializes an object for digesting. Finishes the digest process, generating the final digest output. • Encrypt Data: Initializes an object for encrypting. Encrypts a data stream. • Decrypt Data: Initializes the object for decrypting. Decrypts

Role	Authorized Services
	a data stream. • Seed Random: Add seed material to a DRNG or DRBG object. • Generate Random Bytes: Generates bytes of random data. • Sign: Creates a DSA or RSA signature. • Verify: Verifies a DSA or RSA signature. • Generate Key Pair: Generates a DSA, RSA key pair • MAC Data: Calculates a HMAC of input data. • Show Status: Returns the current status of the Module. • Other non-security relevant functions.
Cryptographic-Officer Role: This role shall provide the services necessary for: Performing module Self-Tests Zeroizing and destroying CSPs	• Perform Self-Tests: Executes the suite of self-tests required by FIPS 140-2. • Zeroize: Actively destroys all plaintext critical security parameters. • Other non-security relevant functions.

The Perform Self-Tests service is automatically run when the VIBECM is powered on/initialized. The operator can cause this service to be run by calling the VtCreateLibCtxFips function in the C language API vibecrypto.h.

6.2 Service Inputs and Outputs

The following table specifies the inputs and output for each service.

Table 7: Specification of Service Inputs & Outputs

Service	Control Input	Data Input	Data Output	Status Output
Create Algorithm Object	VtCreateAlgorithmObject function call	Algorithm Specific Information	Algorithm Object	Succeed / Fail
Destroy Algorithm Object	VtDestroyAlgorithmObject function call	Pointer to Algorithm Object	None	Succeed / Fail
Create Random Object	VtCreateRandomObject function call	XKEY	DRNG Object	Succeed / Fail
Destroy Random Object	VtDestroyRandomObject function call	Pointer to DRNG Object	None	Succeed / Fail
Create Key Object	VtCreateKeyObject function call	None	Key Object	Succeed / Fail
Destroy Key Object	VtDestroyKeyObject function call	Pointer to Key Object	None	Success / Fail
Set Key Object	VtSetKeyParam function call	Key Data	None	Succeed / Fail
Get Key Info	VtGetKeyParam function call	Pointer to Key Object	Key data	Succeed / Fail
Create Parameter Object	VtCreateParameterObject function call	None	Parameter Object	Succeed / Fail
Destroy Parameter Object	VtDestroyParameterObject function call	Pointer to Parameter Object	None	Succeed / Fail
Set Parameter Object	VtSetParameterParam function call	Parameter Data.	None	Succeed / Fail

Service	**Control Input**	**Data Input**	**Data Output**	**Status Output**
Get Parameter Info	VtGetParameterParam function call	Pointer to Parameter Object	Parameter Data	Succeed / Fail
Generate Parameters	VtGenerateParameters function call	Size of DSA Prime; Size of DH Prime and Subprime; or Size of IBE Prime and Subprime	Parameter Object Filled with Generated Parameters	Succeed / Fail
Digest Data	VtDigestInit, VtDigestUpdate, and VtDigestFinal function calls	Data to Hash	Hashed data	Succeed / Fail
Encrypt Data	VtEncryptInit, VtEncryptUpdate, and VtEncryptFinal function calls	Data to Encrypt, Key Object, Random Number Generator Object	Encrypted Data	Succeed / Fail
Decrypt Data	VtDecryptInit, VtDecryptUpdate, and VtDecryptFinal function calls	Encrypted Data, Key Object, Random Number Generator Object	Decrypted Data	Succeed / Fail
Seed Random	VtSeedRandom function call	DRNG Seed Data	None	Succeed / Fail
Generate Random Bytes	VtGenerateRandomBytes function call	None	Random Data	Succeed / Fail

Service	**Control Input**	**Data Input**	**Data Output**	**Status Output**
Sign	VtSign function call	Key Object, Hashed Data, Random Number Generator Object	Cryptographic Signature	Succeed / Fail
MAC Data	VtMACInit, VtMACUpdate, VtMACFinal	Key Object, Algorithm Object, Data to MAC	MAC Data	Succeed/Fail
Generate Shared Secret	VtGenerateSharedSecret	Key Object (Sender), Key Object (Recipient), Random Number Generator Object	Shared Secret	Succeed/Fail
Verify	VtVerifySignature function call	Cryptographic Signature, Key Object, Hashed Data, Random Number Generator Object	Verification Result	Succeed / Fail
Generate Key Pair	VtGenerateKeyPair function call	Algorithm Object, Random Number Generator Object	DSA, RSA Public and Private Key Objects	Succeed / Fail
Show Status	VtGetFipsError function call	None	Cryptographic Module Status	Succeed / Fail
Perform Self Test	VtCreateLibCtxFips function call	None	None	Succeed / Fail

Service	**Control Input**	**Data Input**	**Data Output**	**Status Output**
Zeroize	VtDestroyLibCtxFips function call Win32FipsDestroyMemoryCtx function call LintelFipsDestroyMemoryCtx function call	None	None	Succeed / Fail

6.3 Definition of Critical Security Parameters (CSPs)

The following are the critical security parameters contained in the Module:

- AES Keys: These keys are imported into the Module by the operator using VIBECM services.
- Triple-DES Keys: These keys are imported into the Module by the operator using VIBECM services.
- DSA Private Keys. These keys are generated by, or imported into the Module by the operator using VIBECM services.
- DRNG XKEY: This key is imported into the Module by the operator using VIBECM services and is the initial XKEY value for the FIPS 186-2 DRNG.
- DRBG Seed and Entropy Input: These values are imported into the Module by the operator using VIBECM services.
- DRBG V and C: These values are imported into the Module by the operator using VIBECM services.
- RSA Private Keys: These keys are generated by, or imported into the Module by the operator using VIBECM services.
- HMAC Secret Key: These keys are generated by, or imported into the Module by the operator using VIBECM services.

Definition of Public Keys:

The following are the public keys contained in the Module:

- DSA Software Signing Public Key: This key is the DSA public key associated with the DSA private key used to sign the Module DLL or shared library for software integrity checking.
- DSA Public Keys. These keys are generated by, or imported into the Module by the operator using VIBECM services.
- RSA Public Keys. These keys are generated by, or imported into the Module by the operator using VIBECM services.

6.4 Definition of CSPs Modes of Access

Table 8 defines the relationship between access to CSPs and the different Module services. The modes of access shown in the table are defined as follows:

- Create: Creates two key objects, and then fills the key objects with cryptographic keys, using previously created random object as input. One key object contains the private key, and the other key object contains the public key.

- Zeroize: Destroys a cryptographic key object, freeing memory allocated for this object.
- Write: Sets a cryptographic key object with key data.
- Read: Accesses a CSP to obtain information about the CSP.

The following table describes how the services performed by each role access the CSP. An "X" means that the service is allowed in that mode.

Table 8: CSP Access Rights within Roles & Services

Role		Service	Cryptographic Keys and CSPs Access Operation
Cryptographic Officer	**User**		
	X	Create Algorithm Object	None
	X	Destroy Algorithm Object	None
	X	Create Random Object	Writes a DRNG XKEY key to a Random Number Generator Object
	X	Destroy Random Object	None
	X	Create Key Object	None
	X	Destroy Key Object	Zeroizes AES Key, Triple-DES Key, HMAC key or DSA Key
	X	Set Key Object	Writes to a key object with key data
	X	Get Key Info	Reads key data
	X	Create Parameter Object	None
	X	Destroy Parameter Object	None
	X	Set Parameter Object	None

Role		Service	Cryptographic Keys and CSPs Access Operation
Crypto graphic Officer	**User**		
	X	Get Parameter Info	None
	X	Generate Parameters	None
	X	Digest Data	None
	X	Encrypt Data	Reads key for selected algorithm (AES Key or Triple-DES Key)
	X	Decrypt Data	Reads key for selected algorithm (AES Key or Triple-DES Key)
	X	Seed Random	Writes DRNG seed into a Random Number Generator Object
	X	Generate Random Bytes	None
	X	Sign	Reads key for the selected algorithm (DSA, RSA)
	X	Verify	Reads key for the selected algorithm (DSA, RSA)
	X	Generate Key Pair	Creates key pair for the selected algorithm (DSA, RSA, DH)
	X	MAC Data	Reads HMAC key
	X	Show Status	None
X		Perform Self Test	None
X		Zeroize	Zeroize all CSPs

7. Operational Environment

The operating environment for the Module is a "modifiable operational environment".

The FIPS 140-2 Area 6 Operational Environment requirements for Security Level 1 are satisfied in the following ways:

When the Module is operated in FIPS approved mode, the environment is restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded).

The Module prevents access by other processes to plaintext private and secret keys, CSPs, and intermediate key generation values during the time the cryptographic Module is executing/operational; using address space separation mechanisms of the operational environment. Processes that are spawned by the Module are owned by the Module and are not owned by external processes/operators. Non-cryptographic processes shall not interrupt the Module during execution.

The Module software is installed in a form that protects the software and executable code from unauthorized disclosure and modification.

Cryptographic algorithm integrity tests are performed using Power-Up Self-Tests, Software Integrity Tests, and Conditional Self Tests. (See Section 8 - Security Rules)

8. Security Rules

1. The Module design corresponds to the VIBECM security rules. This section documents the security rules enforced by the Module to implement the security requirements of this FIPS 140-2 Level 1 module.

2. The VIBECM performs all of the tests listed below.

   A. Power up Self-Tests: These are performed without any operator intervention.

      1. Cryptographic algorithm tests
         a. DSA, Sign/Verify Pairwise Consistency Test
         b. RSA Sign/Verify Known Answer Test
         c. RSA Encrypt/Decrypt Pairwise Consistency Test
         d. DH Pairwise Consistency Test
         e. MD5 Known Answer Test
         f. HMAC SHA-1 Known Answer Test
         g. HMAC SHA-256 Known Answer Test
         h. HMAC SHA-512 Known Answer Test
         i. AES, CBC mode, 128 bit key size Known Answer Test
         j. TDES, CBC mode, Known Answer Test
         k. DRNG, X values, Known Answer Test
         l. DRNG, K values, Known Answer Test
         m. DRBG, V values Known Answer Test
         n. DRBG, C values Known Answer Test
      2. Software Integrity Test
         a. DSA Signature verification of the vibecryptofips.dll or the vibecryptofips.so.

   B. Conditional Self-Tests: These tests are performed during the appropriate services.

      1. Continuous Random Number Generator (DRNG) tests – initiated at random number generation and performed by both the FIPS 186-2 appendix 3.1 (DRNG, X values) and the FIPS 186-2 appendix 3.2 (DRNG, K values) random number generators
      2. Continuous Random Bit Generator (DRBG) tests – initiated at random byte generation and performed for both the V and C values as defined in SP 800-90.
      2. Pairwise consistency test for newly generated DSA key pairs
      3. Pairwise consistency test for newly generated RSA signature key pairs.

4. Pairwise consistency test for newly generated RSA encryption key pairs
5. Pairwise consistency test for newly generated DH key pairs

9. Physical Security Policy

VIBECM is a software module and the physical security requirements are not applicable.

10. Mitigation of Other Attacks Policy

The Module is not designed to mitigate any other attacks.

11. References

This section contains informative references that provide helpful background information.

[FIPS-140-2] "Security Requirements for Cryptographic Modules," Version 2, May 25, 2001.
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

[FIPS-180-2] "Secure Hash Standard," Version 2, August 1, 2002.
http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf

[FIPS-186-2] "Digital Signature Standard (DSS)," Version 2, January 27, 2000.
http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf

[FIPS-197] "Advanced Encryption Standard (AES)," November 26, 2001.
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[FIPS-46-3] "Data Encryption Standard," October 25, 1999.
http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[FIPS-198] "The Keyed-Hash Message Authentication Code (HMAC)," April 8, 2002.
http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf

12. Definitions and Acronyms

The following paragraphs define the acronyms used in this document.

AES. Advanced Encryption Standard secret key algorithm. See [FIPS-197].
API. Application Program Interface
CBC. Cipher Block Chaining mode
CFB. Cipher Feed Back mode
CSP. Critical Security Parameters
DES. Data Encryption Standard. See [FIPS-46-3].
DRNG. Deterministic Random Number Generator.
DSS. Digital Signature Standard. See [FIPS-186-2]
ECB. Electronic Codebook mode
EMI. Electromagnetic Interference
EMC. Electromagnetic Compatibility
FIPS. Federal Information Processing Standards of NIST.
IV. Initialization Vector
NIST. National Institute of Standards and Technologies.
OFB. Output Feed Back mode
SHA-1. Secure Hash Algorithm revision 1. See [FIPS-180-2].
TDES. Triple DES. See [FIPS-43-3].