3e Technologies International, Inc.

FIPS 140-2

Non-Proprietary Security Policy

3e Cryptographic Kernel Library

(Version 1.0)

September 25, 2007

Table of contents

GLOSSARY OF TERMS

AP	Access Point
CO	Cryptographic Officer
DH	Diffie Hellman
IP	Internet Protocol
EAP	Extensible Authentication Protocol
FIPS	Federal Information Processing Standard
HTTPS	Secure Hyper Text Transport Protocol
LAN	Local Area Network
MAC	Medium Access Control
PRNG	Pseudo Random Number Generator
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SRDI	Security Relevant Data Item
SSID	Service Set Identifier
TLS	Transport Layer Security
WAN	Wide Area Network
WLAN	Wireless Local Area Network

1. Introduction

1.1. Purpose

This document describes the non-proprietary cryptographic module security policy for 3e Technologies International's 3e Cryptographic Kernel Library (Version 1.0), hereafter known as the CKL. This software is intended to run on Microsoft Windows based computers. This software was created to perform cryptographic operation at the kernel level within Microsoft Windows environment. This policy was created to satisfy the requirements of FIPS 140-2 Level 1. This document defines 3eTI's security policy and explains how 3eTI CKL meets the Level 1 FIPS 140-2 requirements. This security policy document also serves the purpose of allowing independent evaluation of CKL with respect to this policy and for the needs of potential users.

The cryptographic module security policy consists of a specification of the security rules, under which the cryptographic module shall operate, including the security rules derived from the requirements of the standard. Please refer to FIPS 140-2 (Federal Information Processing Standards Publication 140-2) — *Security Requirements for Cryptographic Modules* available on the NIST website at http://csrc.nist.gov/cryptval/.

1.2. Definition

The 3e Cryptographic Kernel Library (CKL) is a software module that implements a set of cryptographic algorithms for use by a software application which usually is in the form of a Windows device driver. The 3eTI CKL is a binary dynamic link library that is compiled from source code written in C, and C++ . This binary library resides in Windows kernel space. It runs on PCs under Windows OS. This product is tested on Windows XP (32-bit) using an x86-compatilbe PC. This library runs in FIPS-mode only and does not implement a non-Approved mode of operation.

The Operating System must be configured to run single-user mode (See Section 3). For purposes of FIPS 140-2, the module is considered to be a multi-chip standalone product. The physical cryptographic boundary is the machine on which the software is loaded. The logical boundary consists of the CCL.sys and cclib.dat files.

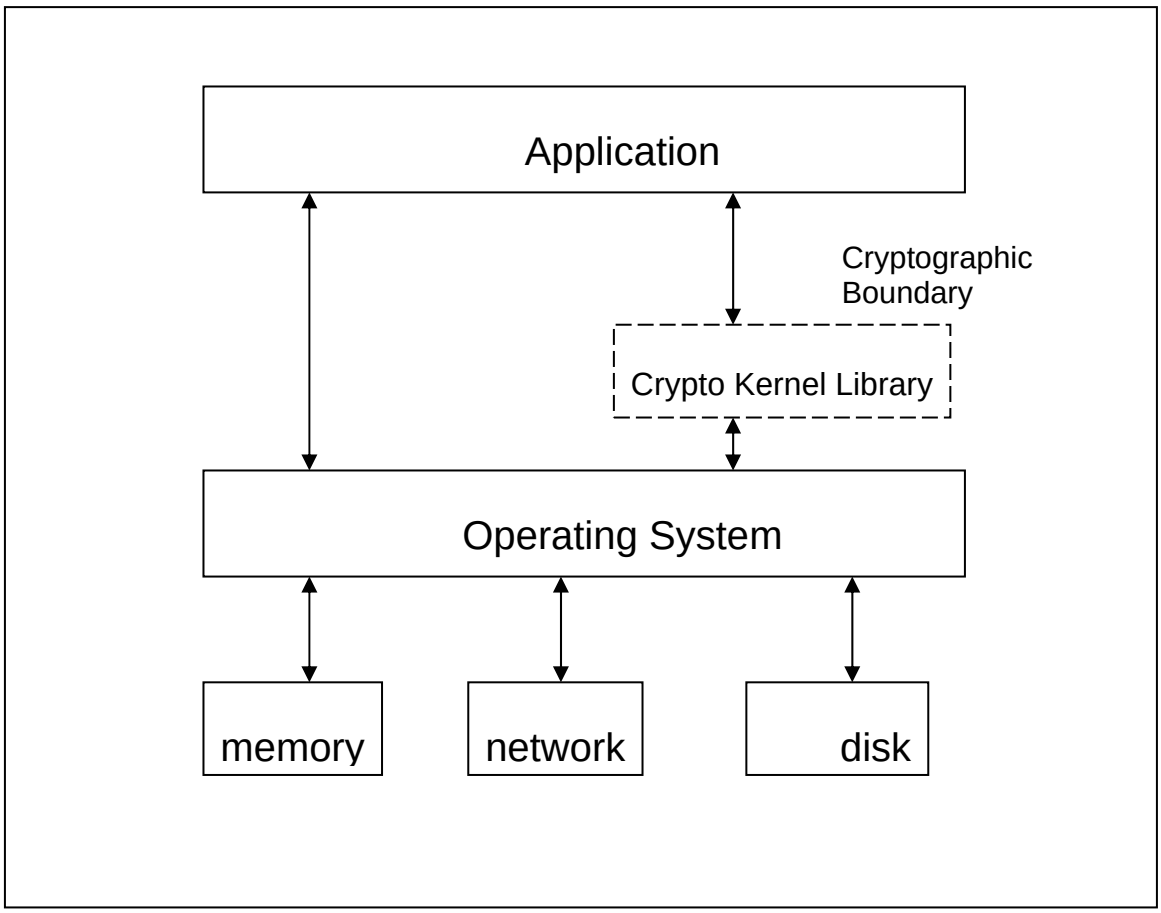

Figure 1 Cryptographic Boundary

1.3. Cryptographic Module Description

3eTI Cryptographic Kernel Library provides the following major services:

- Symmetric Encryption/Decryption

  AES: ECB with key size 128, 192, 256 bit

  AES_CCM: with key size 128 bit

  Triple-DES: ECB with key size 112 and 168 bit

- Key Zeroization

1.4. Scope

This document will cover the secure operation of the 3eTI Cryptographic Kernel Library. It includes the initialization, roles and responsibilities of operating the product in a secure, FIPS-compliant manner, and also describe the Security Relevant Data Items (SRDIs).

2. Roles, Services, and Authentication

2.1. Roles

The User and Crypto Officer roles are implicitly assumed by any entity that can access services implemented by the CKL. In addition, the Crypto Officer role can install and initialize the CKL module; this role is implicitly entered when installing the CKL or performing system administration functions on the host operating system:

Role	Authorized Services	Type of Authentication	Authentication Data
User role	All services except installation and initialization	N/A	N/A
Crypto Officer role	All services including installation and initialization	N/A	N/A

Table 1: Roles, Authorized Services & Authentication

This implementation of roles, services and authentication meets the FIPS 140-2 level 1 requirements for Roles and Services. As a library and as allowed by FIPS 140-2, the CKL does not support user identification or authentication for those roles listed above.

2.2. Services

The 3eTI CKL provides the following major services listed in the following table:

Roles	Services	Critical Security Parameters	Algorithm	API Functions	Access
User, Crypto Officer	Symmetric Encryption/Decryption	Symmetric key	AES AES_CCM 3DES	CCL_AES_ecb_encrypt CCL_AES_ecb_decrypt CCL_AES_CCM_Encrypt	Read Write Execute

				CCL_AES_CCM_Decrypt CCL_TDES_ecb_encrypt CCL_TDES_ecb_decrypt CCL_AES_CCM_ConstructNonce CCL_AES_CCM_ConstructAdata CCL_ZeroizeKey	
User, Crypto Officer	Non-Cryptographic header control and status functions	none	none	CCL_Is_3eti_Header CCL_Add_3eti_Header	Read, Write, Execute

Table 2: Authorized Services

2.3. Authentication Mechanisms and Strength

The following table identifies the strength of authentication for each authentication mechanism supported:

Authentication Mechanism	Strength of Mechanism
None	N/A

Table 3: Strengths of Authentication Mechanisms

3. Secure Operation and Security Rules

3.1. Security Rules

The following security rules must be followed by the operator in order to ensure secure operation:

1. The Operating System enforces authentication method(s) to prevent unauthorized access to CKL services.

2. All Critical Security Parameters are verified as correct and are securely generated, stored, and destroyed.

3.2. FIPS mode of operation

The following steps must be performed to place the Operating System in single user mode:

1. The operating system must be configured to prevent remote login and access to the workstation as a server. The operating system must be configured to run in single-user mode.

2. The paging file (Virtual memory) must be configured to reside on a local drive on the workstation, not a network drive.

3.3. FIPS Policy

The following policies must always be followed in order to achieve a FIPS 140-2 mode of operation:

1. The single user mode enforces the rule that the module can be used by only one human operator at a time, and cannot be actively shared among different operators concurrently.

2. None of the files belonging to the module that are installed on the machine should be moved from their original location.

3. The 3eTI Cryptographic Kernel Library's Crypto Officer must have Administrative privileges on the computer on which the module is run. This will allow the CO to install and uninstall the software if needed.

3.4. Installation and Initialization

To install and initialize the 3eTI CKL for FIPS 140-2 compliant operation, the following steps must be performed:

The operating system must be configured to operate in single user mode. The CKL kernel mode driver (CCL.sys) must be installed into the running operating system as a boot loadable driver.

4. Physical Security

The 3eTI Cryptographic Kernel Library is software that can be run on Windows Operating Systems and it's tested on Windows XP SP2 (32 bit). The FIPS 140-2 physical security requirements do not apply.

5. Security Relevant Data Items

This section specifies the CKL's Security Relevant Data Items (SRDIs) as well as the access control policy enforced by the 3eTI Cryptographic Kernel Library.

5.1. Cryptographic Algorithms

The 3eTI CKL supports the following FIPS-approved cryptographic algorithms:

- AES (ECB mode; 128, 192, 256-bit key sizes)

- Triple-DES (ECB mode; 112 and 168 bit key size)

- AES_CCM (128 bit key size)
- SHA-1 (software integrity test)
- HMAC SHA-1 (software integrity test)

5.2. Self-tests

5.2.1. Power-up Self-tests

- AES ECB - encrypt/decrypt KAT
- Triple-DES ECB - encrypt/decrypt KAT
- AES_CCM – encrypt/decrypt KAT
- SHA-1 KAT
- HMAC SHA-1 KAT
- Software Integrity Test

5.3. Critical Security Parameters

Critical Security Parameter is information such as symmetric keys, and asymmetric private keys that must be protected from unauthorized access. Since 3eTI CKL is accessed via an API from a referencing application, the CKL does not manage CSPs.

The application designer and the end user of the CKL are responsible for ensuring that CSPs are always protected from unauthorized access. This protection will generally make use of the security features of the host hardware and software which is outside the cryptographic boundary defined for this Module.

5.4. Access Control Policy

As stated under section 2.1, the CKL supports the following 2 roles: User and Crypto Officer.

An operator in either role can perform the following operations on CSPs: read, write and execute. Each services API indicates the type of access to CSPs defined by that API. When a CSP is used by the API call to perform a specific service, read and execute access is indicated. When a CSP is generated, modified or deleted by the API call, write access is indicated.

Approved Service	CSPs	Certificate Number	Accessible Roles	Type of Access

AES	AES Key, stored in plaintext in RAM	#640	User, Crypto Officer	Read, execute. Zeroization via CCL_ZeroizeKey()
TDES	Triple DES Key, stored in plaintext in RAM	#593	User, Crypto Officer	Read, execute. Zeroization via CCL_ZeroizeKey()
AES_CCM	AES_CCM Key, stored in plaintext in RAM	#640	User, Crypto Officer	Read, execute. Zeroization via CCL_ZeroizeKey()
Run Self Tests	HMAC Software integrity key, stored in plaintext on the hard drive	#329	Crypto Officer	Read, Zeroization via uninstalling the module.

Table 4: FIPS 140-2 Approved Services Authorized for Roles

5.5. API Specification

The following is a complete list of exported API functions and their prototypes:

```
int CCL_AES_CCM_ConstructNonce(unsigned char *phdr, int hdrLen,
uint32_t iv, uint32_t eIV, unsigned char *pNonce, unsigned char
*pCcmpHdr);

int CCL_AES_CCM_ConstructAdata (unsigned char *phdr, int hdrLen,
unsigned char *pAData);

int CCL_AES_CCM_Encrypt(unsigned char *nonce, unsigned int nlen,
unsigned char *adata, unsigned int alen, unsigned char
*pSourcePayload, unsigned int uiSourcePayloadLength,  unsigned
char *pKey, unsigned char *pEncryptedPayload, unsigned char
*pMic);

int CCL_AES_CCM_Decrypt(unsigned char *nonce, unsigned int nlen,
unsigned char *adata, unsigned int alen, unsigned char
*pSourcePayload, unsigned int uiSourcePayloadLength,  unsigned
char *pKey);

int CCL_AES_ecb_encrypt(int dlen, unsigned char *in, int klen,
unsigned char *key, unsigned char *out);

int CCL_AES_ecb_decrypt(int dlen, unsigned char *in, int klen,
unsigned char *key, unsigned char *out);
```

```
int CCL_TDES_ecb_crypt(u8 * inbuf,int len, u8 * outbuf,u8 *key,
int enc, int lastBlockBytes);

NTSTATUS CCL_ZeroizeKey(char *pKey, int length);



int CCL_Is_3eti_Header(unsigned char *pbuf, int eType);

int CCL_Add_3eti_Header(unsigned char *p3etiHdr,unsigned int
payLoadLen, int encType);
```

6. Mitigation of Other Attacks

The module does not provide mitigation against any commonly known attacks. FIPS 140-2 Level 1 does not require a specific security policy for mitigation of other attacks, except those for which testable requirements are defined in the standard.