Advanced Communications Concepts Inc (ACCI)
TUCrypt Cryptographic Module
FIPS 140-2 Validation Non-Proprietary Security Policy

November 18, 2009 Document Version: 1.8

Revision History

Date	Description
12-08-2008	First Draft
12-16-2008	Revision 1.0
03-24-2009	Revision 1.1
04-03-2009	Revision 1.2
05-05-2009	Revision 1.3
06-12-2009	Revision 1.4
06-18-2009	Revision 1.5
08-04-2009	Revision 1.6
11-09-2009	Revision 1.7
11-18-2009	Revision 1.8

Table of Contents

Table of Contents *3*
1.0 Introduction *4*
2.0 Cryptographic Module *6*
2.1 Cryptographic Module Interfaces **6**
3.0 Roles and Services *8*
3.1 Crypto Officer Role **8**
3.2 User Role **9**
4.0 Physical Security *10*
5.0 EMI/EMC *10*
6.0 Cryptographic Key Management *10*
6.1 Key Generation **13**
6.2 Key Transport **13**
7.0 Self-Tests *13*
8.0 Operational Environment *14*
9.0 Design Assurance *14*
10.0 Mitigation of other attacks *15*
11.0 Secure Operation *15*
11.1 Initial Setup **15**
Appendix A Acronyms *17*
Appendix B References *18*
Appendix C TUCrypt Module API *19*

1.0 Introduction

The non-proprietary security policy describes how the ACCI TUCrypt software cryptographic module version 2.32.0.0 will meet the security requirements of FIPS 140-2. This policy was prepared as part of the FIPS 140-2 Security Level 1 (SL1) validation of the ACCI TUCrypt. The ACCI TUCrypt is referred to in this document as the ACCI TUCrypt software cryptographic module, cryptographic module or module.

The cryptographic module meets the overall requirements applicable to Level 1 security for FIPS 140-2 as shown in the table below:

Table 1

Security Requirements Section	**Level**
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles and Services and Authentication	1
Finite State Machine 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

More information about the FIPS 140-2 standard and validation program is available on the NIST web site at:

http://csrc.nist.gov/groups/STM/cmvp/index.html

The security policy document is one document in a complete FIPS 140-2 submission package. In addition to this document, the complete submission package contains:

- Vendor Evidence Document
- Finite State Model
- Other supporting documentation as additional references

With the exception of this non-proprietary security policy, the FIPS 140-2 validation documentation is proprietary to ACCI and is releasable only under appropriate non-disclosure agreements. For access to these documents, contact ACCI through the information provided below.

Advanced Communication Concepts Inc (ACCI)
8834 N. Capital of Texas Highway, Suite 212
Austin, TX 78759
http://www.advcommcon.com

2.0 Cryptographic Module

The TUCrypt cryptographic module is classified as a multi-chip standalone cryptographic module for the purposes of FIPS 140-2. As such, TUCrypt cryptographic module must be tested on a specific operating system and computer platform. The cryptographic boundary includes TUCrypt cryptographic module running on selected operating systems while configured in "single user" mode. TUCrypt cryptographic module was validated as meeting all FIPS 140-2 Level 1 security requirements, including cryptographic key management and operating system requirements. The TUCrypt cryptographic module is packaged as a dynamically loaded library to perform operating system user space cryptography functions and a device driver to perform operating system kernel space cryptography functions, encompassing the module's executable code. The TUCrypt cryptographic module relies on the physical security provided by the host PC in which it runs on.

For FIPS 140-2 validation, TUCrypt cryptographic module is tested on the following platforms:

- Microsoft Windows® XP Professional, x86 (32-bit), built with Visual Studio 2008 SP1.
- Microsoft Windows Vista, x86 (32-bit), built with Visual Studio 2008 SP1.

2.1 Cryptographic Module Interfaces

TUCrypt cryptographic module is evaluated as a multi-chip, standalone module. The physical cryptographic boundary of the module is the case of the general-purpose computer, which encloses the hardware running the module. The physical interfaces for TUCrypt cryptographic module consist of the keyboard, mouse, monitor, CD-ROM drive, floppy drive, serial ports, USB ports, COM ports, and network adapter(s).

The logical boundary of the cryptographic module is the set of binary files (TUCrypt.dll v2.32.0.0, TUCrypt.sys v2.32.0.0) that makes up the module. The cryptographic module provides for Control Input through the API calls. Data Input and Output are provided in the variables passed with the API calls, and Status Output is provided through the returns and error codes that are documented for each call. This is illustrated in Figure 1 below.

Figure 1

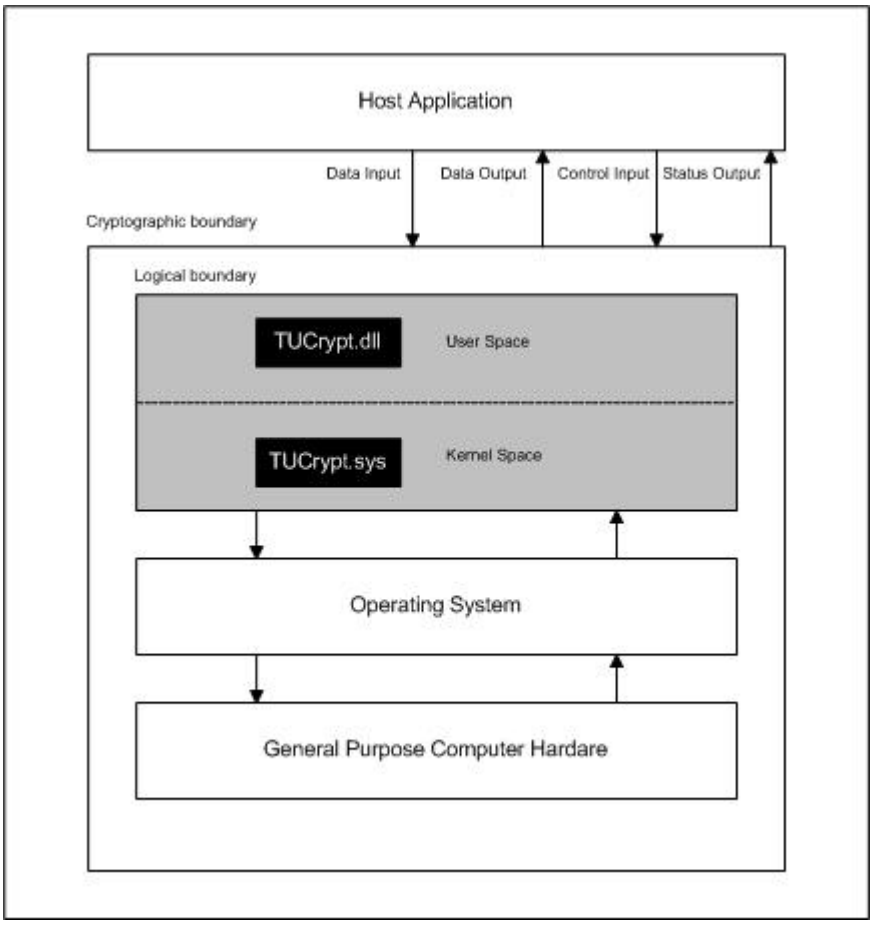
Host Application
Data Input
Data Output
Control Input
Status Output
Cryptographic boundary
Logical boundary
TUCrypt.dll
User Space
TUCrypt.sys
Kernel Space
Operating System
General Purpose Computer Hardare

Table 1

FIPS 140-2 Logical Interfaces	General Purpose Computer Physical Interfaces	TUCrypt Module
Data Input Interface	Keyboard, mouse, CD-ROM, floppy drive, and serial/USB/parallel/network ports	Arguments for a module API call that specify the data to be operated upon by that function.
Data Output Interface	Floppy drive, monitor, and serial/USB/parallel/network ports	Arguments for a module API call that specify where the result of the function is stored.
Control Input Interface	Keyboard, mouse, CD-ROM, floppy drive, and serial/USB/parallel/network ports	API calls utilized to initialize the module and the function calls used to control the operation of the module.
Status Output Interface	Floppy drive, monitor, and serial/USB/parallel/network ports	Return values for API calls.
Power Interface	Power Switch	Not Applicable

3.0 Roles and Services

There are two roles in the module (as required by FIPS 140-2) that operators may assume: a Crypto Officer (CO) role and a User role. The module does not provide any identification or authentication means of its own. The Crypto Officer and the User roles are implicitly assumed based on the service requested. Both of the roles and their responsibilities are described below.

3.1 Crypto Officer Role

The crypto officer role has the ability to install, uninstall and manage the TUCrypt module. The crypto officer monitors the TUCrypt module by reviewing event output generated by the host application in the form of log messages into a file. Table 2 provides a mapping of the services to CSPs and Type of Access.

Table 2

Services	Description	CSP/Access
Module Installation	Installing the TUCrypt cryptographic module.	None
Module Un-Installing	Uninstalling the TUCrypt cryptographic module.	None
Module Management	Monitoring the log file of the host application	None

3.2 User Role

The user role has the ability to utilize the API's available from the TUCrypt Module such as hashing, message authentication, encryption and decryption. The hosting application that is using the cryptographic module is considered a user. Table 3 provides a mapping of the services to CSPs and Type of Access.

Table 3

Services	Description	CSP/Access
Hashing	Hashing operation.	None
Message Authentication	HMAC authentication operation.	HMAC Integrity Key - Read
Encryption	Symmetric key encryption operation.	AES Key – Read
Decryption	Symmetric key decryption operation.	AES Key – Read
KEK Unwrap	Using the module's KEK to unwrap symmetric keys inputted in the module.	AES Key Encryption Key (KEK) – Read/Write

Show Status	Show Status Service	None
Module Initialization	Host application initializing the cryptographic module. Invokes the FIPS 140-2 self-tests (Power-Up and On-Demand) for the cryptographic module.	None
Module Un-Initialization	Host application un-initializing the cryptographic module	None

4.0 Physical Security

The TUCrypt module is a multi-chip standalone module, which is purely a software module and thus physical security requirements do not apply.

5.0 EMI/EMC

Although the TUCrypt module consists entirely of software, the hardware is a standard IBM compatible PC, which has been tested for and meets applicable Federal Communication Commission (FCC) Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined in Subpart B of FCC Part 15, Class B.

6.0 Cryptographic Key Management

The TUCrypt module's user mode dynamic link library (dll) implements the following FIPS 140-2 approved algorithms:

For Windows XP (TUCrypt.dll):

- AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit) – certificate #1057
- HMAC SHA-512 - certificate #595
- SHA-512 - certificate #1003

For Windows VISTA (TUCrypt.dll):

- AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit) – certificate #1058
- HMAC SHA-512- certificate #596

- SHA-512 - certificate #1004

The TUCrypt module's kernel mode sys driver implements the following FIPS 140-2 approved algorithms:

For Windows XP (TUCrypt.sys):

- AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit) – certificate #1102
- HMAC SHA-512 – certificate #616
- SHA-512 – certificate #1025

For Windows VISTA (TUCrypt.sys):

- AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit) – certificate #1102
- HMAC SHA-512 – certificate #616
- SHA-512 – certificate #1025

The TUCrypt module currently does not implement non-approved FIPS 140-2 algorithms.

All Critical Security Parameters (CSP) are protected against unauthorized disclosure, modification, and substitution. The TUCrypt module only allows access to CSPs through the module's well-defined API commands.

The TUCrypt module supports the following critical security parameters:

Table 4 – List of CSP's present in TUCrypt.dll

CSP	Generation/Key Size	Storage	Zeroization	Usage
AES Key	Generated externally of the cryptographic module/ 256 bits.	Held in volatile memory in plaintext.	Erased from memory by overwriting with NULL bytes when TUCrypt is un-initialized by the host application or reboot.	Used for encryption and decryption of data. Entry: Inputted as ciphertext. Module decrypts with AES Key Encryption Key. Output: Never Outputted.
HMAC Integrity Key	Hardcoded in source code / 512 bits.	Stored in non-volatile memory in plaintext	Erased from the hard drive when module is uninstalled and hard drive is reformatted.	Used for Power-Up Software Integrity test. Entry: Never Entered Output: Never Outputted.
AES Key Encryption Key (KEK)	Generated externally of the cryptographic module/ 256 bits.	Held in volatile memory in plaintext.	Erased from memory by overwriting with NULL bytes when TUCrypt is un-initialized by the host application or reboot.	Used by the module to unwrap encrypted AES key that is inputted. Entry: Inputted as plaintext. Output: Never Outputted.

Table 5 – List of CSP's present in TUCrypt.sys

CSP	Generation/Key Size	Storage	Zeroization	Usage
AES Key	Generated externally of the cryptographic module/ 256 bits.	Held in volatile memory in plaintext.	Erased from memory by overwriting with NULL bytes when TUCrypt is un-initialized by the host application or reboot.	Used for encryption and decryption of data. Entry: Inputted as ciphertext. Module decrypts with AES

				Key Encryption Key. Output: Never Outputted.
HMAC Integrity Key	Hardcoded in source code / 512 bits.	Stored in non-volatile memory in plaintext	Erased from the hard drive when module is uninstalled and hard drive is reformatted	Used for power-up software integrity test. Entry: Never Entered Output: Never Outputted.
AES Key Encryption Key (KEK)	Generated externally of the cryptographic module/ 256 bits.	Held in volatile memory in plaintext.	Erased from memory by overwriting with NULL bytes when TUCrypt is un-initialized by the host application or reboot.	Used by the module to unwrap encrypted AES key that is inputted. Entry: Inputted as plaintext. Output: Never Outputted.

6.1 Key Generation

The TUCrypt module does not generate any cryptographic keys internally.

6.2 Key Transport

The AES Key may enter the TUCrypt module AES wrapped with the AES Key Encryption Key (KEK). This mechanism is compliant to RFC standard 3394.

7.0 Self-Tests

The TUCrypt module performs the following self-tests on power-up and on-demand:

TUCrypt.dll:

- Software integrity check (HMAC SHA-512)
- Known Answer Tests
    - AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit)
    - HMAC SHA-512
    - SHA-512

TUCrypt.sys:

- Software integrity check (HMAC SHA-512)
- Known Answer Tests
    - AES encrypt/decrypt (ECB, CTR, CFB mode 256 bit)
    - HMAC SHA-512
    - SHA-512

The TUCrypt module currently does not perform any conditional self-tests.

The integrity of the TUCrypt module in (TUCrypt.dll, TUCrypt.sys) is checked at runtime when the TC_SetMode_FIPS () function call is made. At runtime the TC_SetMode_FIPS() function uses the embedded HMAC-SHA-512 digest to check the integrity of the memory mapped contents module prior to executing the suite of power-up self tests. The function call also executes the module's known answer tests (KAT) after the integrity of module has been verified. The TC_SetMode_FIPS() function returns OK (==0) if all self-test functions pass.

Upon failure of a self-test, an error message indicating the failure is sent to the calling application and the module enters the Error state where no operations are permitted. To transition out of the Error state, the module must be uninstalled and installed by the crypto officer only.

The module does not provide a direct means for executing an on-demand self-test, though every time the calling application is restarted, the module is also restarted, and the self-tests are automatically executed. To run self-tests on request; restart the application which is using the module.

8.0 Operational Environment

The FIPS 140-2 operational environment requirements are applicable because the cryptographic module operates in a modifiable operational environment. The following operational environments are supported:

- Microsoft Windows XP (single-user mode)
- Microsoft Windows Vista (single-user mode)

9.0 Design Assurance

ACCI uses Subversion 1.6 as a source and document control with Tortoise SVN (windows shell extension) as the client. SVN is used for software and document version control, code sharing and build management. The configuration management system is used for software lifecycle modeling. Software life-cycle modeling is the business of tracking source code as it goes through various stages throughout its life, from

development, to testing, release, reuse, and retirement. ACCI also uses the best practices for configuration management to perform the following processes:

- Workspaces - where developers build, test, and debug
- Codelines - the canonical sets of source files
- Branches - variants of the codeline
- Change propagation - getting changes from one codeline to another
- Builds - turning source files into products

ACCI follows best software engineering principles in designing, developing, tracking and documenting software modules. The FIPS submission documentation is maintained and tracked using SVN.

10.0 Mitigation of other attacks

The TUCrypt module does not claim to mitigate other attacks.

11.0 Secure Operation

The TUCrypt module meets Level 1 requirements for FIPS 140-2. The section below describes how to place and keep the module in the FIPS-approved mode of operation. The cryptographic module is designed to operate with other ACCI host software; installation of the cryptographic module is dependent on installation of the host software.

- The replacement or modification of the module by unauthorized users is prohibited.
- It is the responsibility of the calling application to properly and securely generate, store and destroy all critical security parameters.
- The unauthorized reading, writing, or modification of the address space of the module is prohibited.

11.1 Initial Setup

- The module must explicitly be placed in FIPS mode with the function call TC_SetMode_FIPS (..). TC_SetMode_FIPS() returns OK(==0) if success and 1 otherwise. TC_GetMode_FIPS() returns FIPS_MODE (==2) if FIPS mode was enabled. No cryptographic services will be provided until the module is placed in FIPS mode. However, even though the cryptographic module currently supports only FIPS 140-2 approved algorithms, future versions of the cryptographic module may incorporate non-approved algorithms.

- Before FIPS mode is enabled, the KEK is entered in plaintext. After FIPS mode is enabled, all imported keys are encrypted. To enable FIPS mode, call TC_SetMode_FIPS(KEK).

Appendix A Acronyms

ACCI	Advanced Communications Concept Inc
API	Application Programming Interface
CBC	Cipher Block Chaining
CFB	Cipher Feedback
CMT	Cryptographic module Testing
CMVP	Cryptographic module Validation Program
CO	Crypto Officer
CSE	Communications Security Establishment (Canada)
CSP	Critical Security Parameter
CTR	Counter Mode
DES	Digital Encryption Standard
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
ECB	Electronic Codebook
FIPS	Federal Information Processing Standard
FSM	Finite State Machine
HMAC	Hash Based Message Authentication Code
IV	Initialization Vector
KAT	Known Answer Test
NIST	National Institute of Standards and Technology (United States)
OS	Operating System
RSA	Rivest, Shamir and Adleman
SHA	Secure Hash Algorithm
TUC	Tactically Unbreakable COMSEC
XOR	Exclusive Or

Appendix B References

FIPS PUB 140-2 , Security Requirements for Cryptographic modules, May 2001, National Institute of Standards and Technology

Implementation Guidance for FIPS PUB 140-2 and the Cryptographic module Validation Program, May 2008,National Institute of Standards and Technology

NIST Special Publication 800-90, Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised) March 2007

NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation, 2001 Edition

AES KEY Wrap Specification, http://csrc.nist.gov/groups/ST/toolkit/documents/kms/AES_key_wrap.pdf

Appendix C TUCrypt Module API

The following list contains the functions exported by TUCrypt to its callers.

Symmetric Ciphers:

- TC_AES_ECB_Start
- TC_AES_ECB_Process
- TC_AES_CTR_Start
- TC_AES_CTR_Process
- TC_AES_CFB_Start
- TC_AES_CFB_Process

Hashes:

- TC_SHA512
- TC_hmac_SHA512

Show Status

- TC_GetMode_FIPS

Initialization/Self Tests:

- TC_SetMode_FIPS