VMware Inc.
ACE Encryption Engine
(Software Version: 1.0)

FIPS 140-2
Non-Proprietary Security Policy

Level 1 Validation
Document Version 0.4

Prepared for:
VMware Inc.
3145 Porter Drive
Palo Alto, CA 94304
Phone: (650) 475-5000
Fax: (650) 475 5005
http://www.VMware.com

Prepared by:
Corsec Security, Inc.
10340 Democracy Lane, Suite 201
Fairfax, VA 22030
Phone: (703) 267-6050
Fax: (703) 267-6810
http://www.corsec.com

© 2007 VMware Inc.
This document may be freely reproduced and distributed whole and intact including this copyright notice.
## Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Modification Date</th>
<th>Modified By</th>
<th>Description of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2007-01-19</td>
<td>Rumman Mahmud</td>
<td>Initial draft.</td>
</tr>
<tr>
<td>0.2</td>
<td>2007-04-03</td>
<td>Rumman Mahmud</td>
<td>Modified “Cryptographic Key Management” section.</td>
</tr>
<tr>
<td>0.3</td>
<td>2007-04-19</td>
<td>Rumman Mahmud</td>
<td>Inserted Algorithm Certificate numbers</td>
</tr>
<tr>
<td>0.4</td>
<td>2007-09-04</td>
<td>Darryl Johnson</td>
<td>Updated to address CMVP comments.</td>
</tr>
</tbody>
</table>
Table of Contents

1 INTRODUCTION .............................................................................................................. 4
  1.1 PURPOSE ......................................................................................................................... 4
  1.2 REFERENCES .................................................................................................................. 4
  1.3 DOCUMENT ORGANIZATION ......................................................................................... 4

2 ACE ENCRYPTION ENGINE .............................................................................................. 5
  2.1 OVERVIEW .................................................................................................................... 5
  2.2 MODULE INTERFACES ................................................................................................... 6
  2.3 ROLES AND SERVICES ................................................................................................. 7
    2.3.1 Crypto Officer Role .................................................................................................... 7
    2.3.2 User Role ................................................................................................................... 7
  2.4 PHYSICAL SECURITY ..................................................................................................... 9
  2.5 OPERATIONAL ENVIRONMENT ..................................................................................... 9
  2.6 CRYPTOGRAPHIC KEY MANAGEMENT ......................................................................... 9
  2.7 SELF-TESTS .................................................................................................................... 11
  2.8 DESIGN ASSURANCE ................................................................................................... 12
  2.9 MITIGATION OF OTHER ATTACKS ............................................................................... 12

3 SECURE OPERATION ....................................................................................................... 13
  3.1 CRYPTO OFFICER GUIDANCE ...................................................................................... 13
    3.1.1 Initial Setup ............................................................................................................... 13
    3.1.2 Management ............................................................................................................. 13
    3.1.3 Zeroization ............................................................................................................... 13
  3.2 USER GUIDANCE .......................................................................................................... 13

4 ACRONYMS .................................................................................................................... 15

Table of Figures

FIGURE 1 – DEPLOYING VMware ....................................................................................... 5
FIGURE 2 – LOGICAL BOUNDARY OF THE ACE ENCRYPTION ENGINE.............................. 6
FIGURE 3 – STANDARD PC PHYSICAL BLOCK DIAGRAM .................................................. 7

Table of Tables

TABLE 1 – SECURITY LEVEL PER FIPS 140-2 SECTION ...................................................... 6
TABLE 2 – MAPPING OF LOGICAL TO PHYSICAL INTERFACES ........................................... 7
TABLE 3 – MAPPING OF CRYPTO OFFICER ROLE’S SERVICES TO INPUTS, OUTPUTS, CSPS, AND TYPE OF ACCESS ................................................................. 7
TABLE 4 – MAPPING OF USER ROLE’S SERVICES TO INPUTS, OUTPUTS, CSPS, AND TYPE OF ACCESS ........................................................................................................... 7
TABLE 5 – LIST OF CRYPTOGRAPHIC KEYS, KEY COMPONENTS, AND CSPS PRESENT IN LIBEAY.DLL ........................................................................................................... 10
TABLE 6 – LIST OF CRYPTOGRAPHIC KEYS, KEY COMPONENTS, AND CSPS PRESENT IN VMCRYPTOLIB.DLL ................................................................. 11
TABLE 7 – ACRONYMS ........................................................................................................ 15

VMware ACE Encryption Engine
© 2007 VMware Inc.
1 Introduction

1.1 Purpose

This document is a non-proprietary Cryptographic Module Security Policy for the ACE Encryption Engine from VMware Inc. This Security Policy describes how the ACE Encryption Engine meets the security requirements of FIPS 140-2 (Federal Information Processing Standards Publication 140-2 – Security Requirements for Cryptographic Modules) and how to run the module in a secure FIPS 140-2 mode. This policy was prepared as part of the Level 1 FIPS 140-2 validation of the module.


The ACE Encryption Engine is referred to in this document as the cryptographic module or the module.

1.2 References

This document deals only with operations and capabilities of the module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the module from the following sources:

- The VMware website (www.VMware.com) contains information on the full line of products from VMware.
- The CMVP website (http://csrc.nist.gov/cryptval/) contains contact information for answers to technical or sales-related questions for the module.

1.3 Document Organization

The Security Policy document is one document in a FIPS 140-2 Submission Package. In addition to this document, the Submission Package contains:

- Vendor Evidence document
- Finite State Machine
- Other supporting documentation as additional references

This Security Policy and the other validation submission documentation were produced by Corsec Security, Inc. under contract to VMware. With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Validation Documentation is proprietary to VMware and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact VMware.
2 ACE Encryption Engine

2.1 Overview

VMware Inc. was founded in 1998 to bring virtual machine technology to industry-standard computers. VMware delivered its first product in 1999. Today, more than 4 million users and 20,000 corporate customers of all types and sizes use VMware software, including 99 of the Fortune 100 companies. The ACE Encryption Engine was created by VMware Inc. in 2004 to extend the capabilities of virtual infrastructure technology to the enterprise desktop.

The ACE Encryption Engine allows virtual machines to be encapsulated into files which can be saved, copied, and provisioned into a virtual machine. The virtual machine allows for fully configured applications, operating systems, BIOS, and virtual hardware, which can be moved from one physical computer to another without needing downtime or maintainance.

VMware’s ACE 2.0 is a software solution that delivers enhanced management, security, and usability to standard desktop virtualization products. Using ACE 2.0, an organization can rapidly provision a standardized, secure general purpose computer (GPC) environment — an ACE — to any device in the extended enterprise, regardless of whether it is managed by the ACE administrator. An ACE is a policy-protected virtual machine containing an operating system, applications, and data. Through virtual rights management technology, ACE 2.0 enables desktop administrators to control ACE lifecycles, protect data, and ensure compliance with IT policies including software lifecycle management and access to data and applications.

Unlike other products, ACE 2.0 is a hardware-independent solution that can be provisioned to any GPC and works either connected or disconnected from the enterprise network.

ACE 2.0 is used across an organization to:

- Ensure secure, controlled access to enterprise resources from a standardized PC environment called an ACE
- Provide a simplified end-user interface designed specifically for nontechnical Users
- Provide policy-based controls including access, network, and device rights

VMware’s ACE allows security administrators to package an entire system with an operating system, applications, and policies, and deploy it to an unmanaged GPC. Once VMware ACE is installed, it offers complete control of the hardware configuration and networking capabilities of the GPC. The VMware ACE application itself is a virtual machine that runs on top of a physical hardware such as a GPC, and simulates the operation of the virtualized system it mimicks.

![Figure 1 – Deploying VMware](image-url)
perform its cryptographic functions such as hashing, encryption, digital signing, etc. The ACE Encryption Engine is composed of three dynamic-link library (DLL) files: vmcryptolib.dll, ssleay32.dll, and libeay32.dll. The ACE Encryption Engine’s logical boundary is shown below in Figure 2.

The ACE Encryption Engine is validated at the following FIPS 140-2 Section levels:

<table>
<thead>
<tr>
<th>Section</th>
<th>Section Title</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cryptographic Module Specification</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Cryptographic Module Ports and Interfaces</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Roles, Services, and Authentication</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Finite State Model</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Physical Security</td>
<td>N/A</td>
</tr>
<tr>
<td>6</td>
<td>Operational Environment</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Cryptographic Key Management</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>EMI/EMC</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>Self-tests</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>Design Assurance</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>Mitigation of Other Attacks</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### 2.2 Module Interfaces

The ACE Encryption Engine is a multi-chip stand-alone module that meets overall level 1 FIPS 140-2 requirements. The logical cryptographic boundary of the ACE Encryption Engine is defined by the blue broken line that surrounds the vmcryptolib.dll, libeay32.dll and ssleay32.dll files in Figure 2.
The module is evaluated for use on a GPC. In addition to the binaries, the physical device consists of the integrated circuits of the motherboard, the central processing unit (CPU), random access memory (RAM), read-only memory (ROM), computer case, keyboard, mouse, video interfaces, expansion cards, and other hardware components included in the computer such as hard disk, floppy disk, CD-ROM drive, power supply, and fans. The physical cryptographic boundary of the module is the hard opaque metal and plastic enclosure of the computer. The block diagram for a standard personal computer (PC) is shown below in Figure 3.

![Figure 3 – Standard PC Physical Block Diagram](image)

All of these physical interfaces are separated into logical interfaces defined by FIPS 140-2. Each of these local interfaces are mapped to the corresponding physical interfaces in Table 2.

**Table 2 – Mapping of Logical to Physical Interfaces**

<table>
<thead>
<tr>
<th>Logical Interface</th>
<th>Physical Interface Mapping (Standard PC)</th>
<th>Module Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Input Interface</td>
<td>Keyboard, mouse, CD-ROM, floppy drive, and serial/USB/parallel/network ports</td>
<td>Arguments for a function that specify the data to be operated upon by that function.</td>
</tr>
<tr>
<td>Data Output Interface</td>
<td>Floppy drive, monitor, and serial/USB/parallel/network ports</td>
<td>Arguments for a function that specify where the result of the function is stored.</td>
</tr>
<tr>
<td>Control Input Interface</td>
<td>Keyboard, CD-ROM, floppy drive, mouse, and</td>
<td>Function calls utilized to initialize the module and the function calls used to control the</td>
</tr>
</tbody>
</table>
2.3 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 operator of the module assumes either of the roles based on the operations performed without any authentication. Both of the roles and their responsibilities are described below.

2.3.1 Crypto Officer Role

The Crypto Officer role has the ability to install, uninstall and manage the the ACE Encryption Engine module. The Crypto Officer monitors the ACE Encryption Engine by tracking the event log of the PC’s operating system. Descriptions of the services available to the Crypto Officer role are provided in the table below.

Table 3 – Mapping of Crypto Officer Role’s Services to Inputs, Outputs, CSPs, and Type of Access

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
<th>Input</th>
<th>Output</th>
<th>CSP and Type of Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation</td>
<td>Installing the ACE encryption engine module</td>
<td>Command</td>
<td>Module installed</td>
<td>None</td>
</tr>
<tr>
<td>Uninstallation</td>
<td>Uninstalling the ACE encryption engine module</td>
<td>Command</td>
<td>Module uninstalled</td>
<td>None</td>
</tr>
<tr>
<td>Management</td>
<td>Monitoring, managing and troubleshooting the event log of the operating system</td>
<td>Command</td>
<td>Status output</td>
<td>None</td>
</tr>
</tbody>
</table>

2.3.2 User Role

The User role has the ability to utilize the API’s available from the ACE Encryption Engine cryptographic services such as hashing, message authentication, encryption, pseudo-random number generation and disk sector encryption. Also, the module provides TLS services to Users. Descriptions of the services available to the User role are provided in the table below.

Table 4 – Mapping of User Role’s Services to Inputs, Outputs, CSPs, and Type of Access

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
<th>Input</th>
<th>Output</th>
<th>CSP and Type of Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hashing</td>
<td>Hashing operation</td>
<td>API call with input parameters</td>
<td>Status output, data hashed</td>
<td>None</td>
</tr>
<tr>
<td>Message authentication</td>
<td>Message authentication services</td>
<td>API call with input parameters</td>
<td>Status output, message authentication code (MAC) created</td>
<td>HMAC key – Read</td>
</tr>
<tr>
<td>Cipher operation</td>
<td>Cipher operation</td>
<td>API call with input parameters</td>
<td>Status output, data encrypted or decrypted</td>
<td>Symmetric key – Read/Write</td>
</tr>
</tbody>
</table>
### Service Description

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
<th>Input</th>
<th>Output</th>
<th>CSP and Type of Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Signing</td>
<td>Digital sign and verify operation</td>
<td>API call with input parameters</td>
<td>Status output, Signature</td>
<td>RSA key pair – Read</td>
</tr>
<tr>
<td>Data conversion</td>
<td>Convert entered data into another structure</td>
<td>API call with input parameters</td>
<td>Status output, data reformatted</td>
<td>None</td>
</tr>
<tr>
<td>Pseudo-random Number Generation</td>
<td>Generates random numbers</td>
<td>API call with input parameters</td>
<td>Status output, random number generated</td>
<td>Seed – Read</td>
</tr>
<tr>
<td>TLS services</td>
<td>Establish TLS channel</td>
<td>API call with input parameters</td>
<td>TLS connection</td>
<td>RSA key pair – Read/Write</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Symmetric key – Read/Write</td>
</tr>
<tr>
<td>Disk Sector Encryption</td>
<td>Encryption of disk sector</td>
<td>API call with input parameters</td>
<td>Status output, disk sector encrypted</td>
<td>AES disk sector encryption key – Read</td>
</tr>
</tbody>
</table>

#### 2.4 Physical Security

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

#### 2.5 Operational Environment

The module was tested for FIPS 140-2 validation on Windows XP Professional with SP2 and Windows Vista Ultimate operating systems (OS). VMware affirms that the module is binarily compatible for the general purpose Windows 2000 OS and later editions. The OS must be configured for single user mode, per NIST CMVP guidance for FIPS 140-2 compliance. Single user mode configuration instructions for the OS can be found in the “Secure Operation” section of this document.

#### 2.6 Cryptographic Key Management

The ACE Encryption Engine implements the following FIPS-approved algorithms:

*In Libeay.dll:*
- PRNG Appendix A.2.4 of ANSI X9.31 using 2-key Triple-DES – (certificate #306)
- AES encrypt/decrypt (ECB, CBC mode 128/256 bit) – (certificate #533)
- DSA signature generation and verification (1024 bit) – (certificate #220)
- HMAC SHA-1, HMAC SHA-224, HMAC SHA-256, HMAC SHA-384, HMAC SHA-512 (certificate #280)
- RSA sign/verify (1024, 2048, and 4096 bits) – (certificate #241)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 (certificate #603)
- TDES encrypt/decrypt (ECB, CBC mode KO 1,2,3) – (certificate #536)

*In VMCryptolib.dll:*
- PRNG – FIPS 186-2 General Purpose – (certificate #307)
- AES encrypt/decrypt (ECB, CBC, CTR mode 128/256 bit) – (certificate #534)
- HMAC SHA-1, HMAC SHA-256 (certificate #281)
- SHA-1, SHA-256 (certificate #604)
Additionally, the module utilizes the following non-FIPS approved algorithm implementation:

**In Libeay.dll:**
- Non-approved FIPS RNG used for seeding the FIPS approved deterministic RNG
- RSA (encrypt/decrypt 1024, 2048, and 4096 bits) (key transport methodology provides between 80-bits and 150-bits of encryption strength)
- DH (Key agreement 1024 bits) (key agreement methodology provides between 80-bits and 150-bits of encryption strength)
- RSA (sign/verify 512 bits)
- MD5

**In VMCryptolib.dll:**
- Non-approved FIPS RNG used for seeding the FIPS approved deterministic RNG

All secret keys and Critical Security Parameters (CSP) are protected against unauthorized disclosure, modification, and substitution. The module only allows access to CSPs through the module’s well-defined commands in the API. The module supports the following cryptographic keys and critical security parameters:

**Table 5 – List of Cryptographic Keys, Key Components, and CSPs Present in Libeay.dll**

<table>
<thead>
<tr>
<th>Key</th>
<th>Key Type</th>
<th>Generation/Input</th>
<th>Output</th>
<th>Storage</th>
<th>Zeroization</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA public key</td>
<td>1024, 2048, and 4096 bit public key</td>
<td>Generated internally or enters the module in plaintext</td>
<td>Exits the module in plaintext</td>
<td>Held in volatile memory in plaintext.</td>
<td>Erased when session is closed or on reset</td>
<td>Used for encryption (key transport) and signature verification</td>
</tr>
<tr>
<td>RSA private key</td>
<td>1024, 2048, and 4096 bits private key</td>
<td>Generated internally or enters the module in plaintext</td>
<td>Exits the module in plaintext</td>
<td>Held in volatile memory in plaintext.</td>
<td>Erased when session is closed or on reset</td>
<td>Used for decryption (key transport) and signing</td>
</tr>
<tr>
<td>AES Symmetric key</td>
<td>128 and 256 bits ECB, CBC key</td>
<td>Generated internally or enters the module in plaintext or in encrypted form</td>
<td>Exits the module in plaintext</td>
<td>Held in volatile memory in plaintext.</td>
<td>Erased when session is closed or on reset</td>
<td>Used to perform cipher operation on data for TLS channel</td>
</tr>
<tr>
<td>TDES Symmetric key</td>
<td>TDES ECB, CBC 1, 2, or 3 key</td>
<td>Generated internally or enters the module in encrypted form</td>
<td>Exits the module in plaintext</td>
<td>Held in volatile memory in plaintext.</td>
<td>Erased when session is closed or on reset</td>
<td>Used to perform cipher operation on data for TLS channel</td>
</tr>
<tr>
<td>HMAC key</td>
<td>HMAC key</td>
<td>Generated internally or enters the module in plaintext</td>
<td>Exits the module in plaintext</td>
<td>Held in volatile memory in plaintext.</td>
<td>Erased when session is closed or on reset</td>
<td>Used for message authentication and integrity for TLS channel</td>
</tr>
<tr>
<td>PRNG seed</td>
<td>8 bytes of seed value</td>
<td>Generated internally</td>
<td>Never exits the module</td>
<td>Held in volatile memory only in plaintext</td>
<td>Erased when session is closed or on reset</td>
<td>Used to generate pseudo-random number</td>
</tr>
<tr>
<td>PRNG seed key</td>
<td>16 bytes of key</td>
<td>Generated internally</td>
<td>Never exits the module</td>
<td>Held in volatile memory only in plaintext</td>
<td>Erased when session is closed or on reset</td>
<td>Used to generate pseudo-random number</td>
</tr>
</tbody>
</table>
The module generates an HMAC keys internally or enters in plaintext during Transport Layer Security (TLS) handshaking; an Advanced Encryption Standard (AES) or Triple Data Encryption Standard (TDES) Symmetric key is generated internally using FIPS approved PRNG (ANSI X9.31 Appendix A.2.4) during TLS handshaking. Both the ephemeral keys are zeroized when the session is closed or the module is reset/reloaded. The Pseudorandom Number Generator (PRNG) seed and seed key are generated internally and are zeroized when the session is closed or the module is reset or reloaded. The Rivest Shamir and Adleman (RSA) public/private key is generated when the caller provides the certificate containing the key and is zeroized when the TLS context is destroyed. The DSA public key is hardcoded in the source code and is zeroized when the module is uninstalled.

**Table 6 – List of Cryptographic Keys, Key Components, and CSPs Present in VMCryptolib.dll**

<table>
<thead>
<tr>
<th>Key</th>
<th>Key Type</th>
<th>Generation/Input</th>
<th>Output</th>
<th>Storage</th>
<th>Zeroization</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA public key</td>
<td>1024 bit public key</td>
<td>Hardcoded in source code</td>
<td>Never exits the module</td>
<td>Stored in non-volatile memory in plaintext</td>
<td>Erased when module is uninstalled</td>
<td>Used for power-up software integrity test</td>
</tr>
<tr>
<td>HMAC key</td>
<td>HMAC key</td>
<td>Generated internally or enters the module in plaintext</td>
<td>Exits the module in plaintext</td>
<td>Held in non-volatile memory in plaintext</td>
<td>Erased when session is closed or on reset</td>
<td>Used for message authentication and integrity validation</td>
</tr>
<tr>
<td>PRNG seed</td>
<td>20 bytes of seed value</td>
<td>Generated internally</td>
<td>Never exits the module</td>
<td>Held in volatile memory only in plaintext</td>
<td>Erased when session is closed or on reset</td>
<td>Used to generate pseudo-random number</td>
</tr>
</tbody>
</table>

HMAC keys enter into the module in plaintext. These keys are used for message authentication and integrity checking. VMCryptolib.dll generates an AES Symmetric key internally using FIPS 186-2 Appendix 3.1 PRNG in order to encrypt information stored on the workstation. The AES Disk Sector Encryption key resides only in volatile memory and is zeroized after session is over or on reset. The Pseudorandom Number Generator (PRNG) seed is generated internally and is zeroized when the session is closed or the module is reset or reloaded.

### 2.7 Self-Tests

The ACE Encryption Engine performs the following self-tests at power-up:

**For VMCryptolib.dll:**

- Software integrity check (using DSA)
- Known Answer Tests (KATs)
  - AES KAT (ECB, CBC, CTR modes 128/256 bit)
  - HMAC SHA-1 KAT
  - HMAC SHA-256 KAT
  - PRNG KAT (FIPS 186-2)

**For Libeay.dll:**

- Software integrity check (using DSA)
• Know Answer Tests (KATs)
  o AES KAT (ECB, CBC, modes 128/256 bit)
  o HMAC SHA-1, HMAC SHA-224, HMAC SHA-256, HMAC SHA-384, and HMAC SHA-512 KATs
  o PRNG KAT – Appendix A.2.4 of ANSI X9.31
  o RSA KAT (encrypt/decrypt and sign/verify)
  o SHA-1 KAT
  o TDES KAT (ECB mode)

The ACE Encryption Engine performs the following conditional self-tests:

For VMCryptolib.dll:

• CRNGT for FIPS 186-2 Appendix 3.1 PRNG
• CRNGT for entropy gathering for FIPS 186-2 Appendix 3.1 PRNG

For Libeay.dll:

• CRNGT for ANSI X9.31 Appendix A.2.4 PRNG
• CRNGT for entropy gathering for ANSI X9.31 Appendix A.2.4 PRNG
• RSA pairwise Consistency Test

The Status Output from the self-tests is stored in RAM\(^1\) (unless the Crypto Officer configures the module to record the Status Output into a local file), and it can be accessed whenever the CO needs to troubleshoot a startup failure.

2.8 Design Assurance

Microsoft Visual SourceSafe (VSS) version 6.0 is used to provide configuration management for the ACE Encryption Engine’s FIPS documentation. This software provides access control, versioning, and logging. VSS also maintains an internal revision history of each of the module’s files and uniquely labels these revisions.

The VMware source code is maintained in Perforce, which maintains an internal revision history of the module’s source files. The Perforce server manages the master file repository that contains every revision of every file under Perforce control. The server maintains a database to track change logs, user permissions, and which users have which files checked out at any time.

2.9 Mitigation of Other Attacks

This section is not applicable. The module does not claim to mitigate any attacks beyond the FIPS 140-2 level 1 requirements for this validation.

---

\(^1\) RAM – random access memory
3 Secure Operation

The ACE Encryption Engine meets Level 1 requirements for FIPS 140-2. The section below describes how to place and keep the module in FIPS-approved mode of operation.

3.1 Crypto Officer Guidance

3.1.1 Initial Setup

The Crypto Officer is responsible for installing/uninstalling, configuring, and managing the module. The ACE Encryption Engine is available on a Compact Disc (CD) to the Crypto Officer using a standard carrier (i.e. FedEx or UPS). The Crypto Officer is responsible for inspecting the CD and its packaging upon receipt for signs of tampering. If evidence of tampering is found, the Crypto Officer should contact the vendor immediately and should not use the module. The software module will be provided to the Crypto Officer by VMware Inc. and its marketing affiliates. The module becomes installed during the process of installing the host application, VMware ACE. With the delivered software, the Crypto Officer also receives detailed documentation on installing, uninstalling and configuring VMware ACE. Before installing the module, the CO must configure the PC for single user mode as instructed below:

To configure the Windows platforms for single user mode, the CO must ensure that all remote guest accounts are disabled in order to ensure that only one operator can log into the OS at a time. Guest accounts can be disabled via the Users and Passwords selection on the Control Panel window.

Services that should be disabled are:

- Server services
- Terminal services
- Remote registry service
- Remote desktop and remote assistance service

For specific information regarding the Microsoft operating system security and administration procedures, please refer to Microsoft’s online technical and product information repository at http://technet.microsoft.com/en-us/default.aspx.

3.1.2 Management

The Crypto Officer should monitor the module’s status by regularly checking the Windows Event log. If any irregular activity is noticed or the module is consistently reporting errors, then VMware’s customer support should be contacted. The CO is also responsible to monitor that FIPS mode of operation is maintained by using only FIPS approved functions. The module should not be used for any FIPS non-approved functions, such as MD5 or DH.

3.1.3 Zeroization

The RSA key pair is erased when the TLS session is terminated. The AES key, TDES key, HMAC keys, and PRNG seed are erased when the session is closed or when the module is reset. The DSA public key is erased when the module is uninstalled.

3.2 User Guidance

The User accesses the module’s cryptographic functionalities only. Although the User does not have any ability to modify the configuration of the module, they should check that the host application is enabled and providing cryptographic protection.

The module offers ‘Data conversion’ service to the Users via the following APIs:
CryptoPass2key_FromString
CryptoPass2key_ToString
CryptoPass2key_Compute
CryptoPass2key.Makekey
Crypto.PasswordWrapData
Crypto.PasswordUnwrapData
Crypto.GetPassword
Crypto.ManglePassphrase
Crypto.EncryptPassword
Crypto.DecryptPassword
Crypto.ClearEncryptedPassword
Crypto.InitializeEncryptedPassword

The keys derived from the APIs mentioned above are not to be used for data encryption/decryption. Otherwise, the resultant values will be considered as plaintext in FIPS mode of operation.
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>CD</td>
<td>Compact Disc</td>
</tr>
<tr>
<td>CD-ROM</td>
<td>Compact Disc Read-Only Memory</td>
</tr>
<tr>
<td>CMVP</td>
<td>Cryptographic Module Validation Program</td>
</tr>
<tr>
<td>CO</td>
<td>Crypto Officer</td>
</tr>
<tr>
<td>CSP</td>
<td>Critical Security Parameter</td>
</tr>
<tr>
<td>DLL</td>
<td>Dynamic-Link Library</td>
</tr>
<tr>
<td>DSA</td>
<td>Digital Signature Algorithm</td>
</tr>
<tr>
<td>EMC</td>
<td>Electromagnetic Compatibility</td>
</tr>
<tr>
<td>EMI</td>
<td>Electromagnetic Interference</td>
</tr>
<tr>
<td>FIPS</td>
<td>Federal Information Processing Standard</td>
</tr>
<tr>
<td>HMAC</td>
<td>(Keyed-) Hash Message Authentication Code</td>
</tr>
<tr>
<td>KAT</td>
<td>Known Answer Test</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PRNG</td>
<td>Pseudorandom Number Generator</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RNG</td>
<td>Random Number Generator</td>
</tr>
<tr>
<td>ROM</td>
<td>Read Only Memory</td>
</tr>
<tr>
<td>RSA</td>
<td>Rivest Shamir and Adleman</td>
</tr>
<tr>
<td>SHA</td>
<td>Secure Hash Algorithm</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>VSS</td>
<td>Visual SourceSafe</td>
</tr>
</tbody>
</table>