PGP Corporation

PGP Cryptographic Engine FIPS 140-2 Security Policy

Document Version 1.0.2

Revision Date 5/4/10

Table of Contents

1 INTRODUCTION....2

2 MODULE SPECIFICATIONS....3

2.1 SUPPORTED ALGORITHMS....4
2.2 CRYPTOGRAPHIC BOUNDARY....5
2.3 PORTS AND INTERFACES....6
2.4 SECURITY LEVEL....7
2.5 OPERATIONAL ENVIRONMENT....8
2.6 APPROVED MODE OF OPERATION....9

3 SECURITY RULES....10

4 ROLES AND SERVICES....11

5 ACCESS CONTROL POLICY....14

5.1 CRITICAL SECURITY PARAMETERS....14
5.2 ACCESSES....14
5.3 SERVICE TO CSP ACCESS RELATIONSHIP....14

6 CRYPTOGRAPHIC KEY MANAGEMENT....16

6.1 KEY GENERATION AND ESTABLISHMENT....16

7 PHYSICAL SECURITY POLICY....16

8 SELF-TESTS....16

8.1 POWER-UP TESTS....17
8.2 ON-DEMAND TESTS....17

9 MITIGATION OF OTHER ATTACKS....17

GLOSSARY....18

1 Introduction

The PGP Cryptographic Engine (SW Version 4.0) (hereafter referred to as the “cryptographic module” or the “module”) is a software only cryptographic module validated to the standards set forth by the *FIPS PUB 140-2 Security Requirements for Cryptographic Modules* document published by the National Institute of Standards and Technology (NIST). The module is intended to meet the security requirements of FIPS 140-2 Level 1 overall.

This module is responsible for the cryptographic services used by PGP Corporation’s line of software products and is used as a building block to provide the secure storage of data.

This document, the *Cryptographic Module Security Policy* (CMSP), also referred to as the *Security Policy*, specifies the security rules under which the module must operate.

2 Module Specifications

PGP Cryptographic Engine (SW Version 4.0), is a software-only cryptographic module embodied as a shared library binary that executes on general-purpose computer systems and is available on a number of operating systems. The specific operating system and version to be validated is specified in the *Operational Environment* section of this document.

For the purpose of FIPS validation, this document will only be concerned with the core cryptographic APIs described under Section 4 of this document.

Table 1: Embodiment of the PGP Cryptographic Engine core cryptographic module on various OS in FIPS Mode

Operating System	**Typical Pathname**
Windows XP Professional SP2 (Kernel Mode)	PGPCryptoEng.dll
Mac OS X 10.5 (i386)	PGPCryptoEng.dylib

The PGP Cryptographic Engine cryptographic module is accessible to client applications through an application-programming interface (API). The API functions in the crypto module are enumerated in the Roles and Services Section of this document.

The module provides a FIPS mode of operation; which is described in the *Approved Mode of Operation* section of this document.

For the purposes of FIPS 140-2, the PGP Cryptographic Engine is classified as a multi-chip standalone module.

The module does not support multiple concurrent operators.

The module does not support a bypass mode.

2.1 Supported Algorithms

The PGP Cryptographic Engine implements the following algorithms in the FIPS mode of operation.

Table 2: Algorithms supported by the PGP Cryptographic Engine in FIPS mode

Type	Algorithm	FIPS Status
Symmetric Key	Triple-DES (3-Key) TECB, TCBC, TCFB	FIPS 46-3 (cert # 848, 895)
	AES (128,192,256) ECB, CBC and CFB128	FIPS-197 (cert # 1170, 1253)
Message Digest	SHA-1,256,384,512	FIPS 180-3 (cert # 1082, 1149)
Message Authentication	HMAC SHA-1, 256, 384, 512	FIPS-198 (cert # 670, 732)

Note: Algorithm validation certificate numbers listed in Table 2 are in order of OSX / WIN32 operating systems.

In non-FIPS mode, the PGP Cryptographic Engine module also provides the following non-FIPS Approved algorithms:

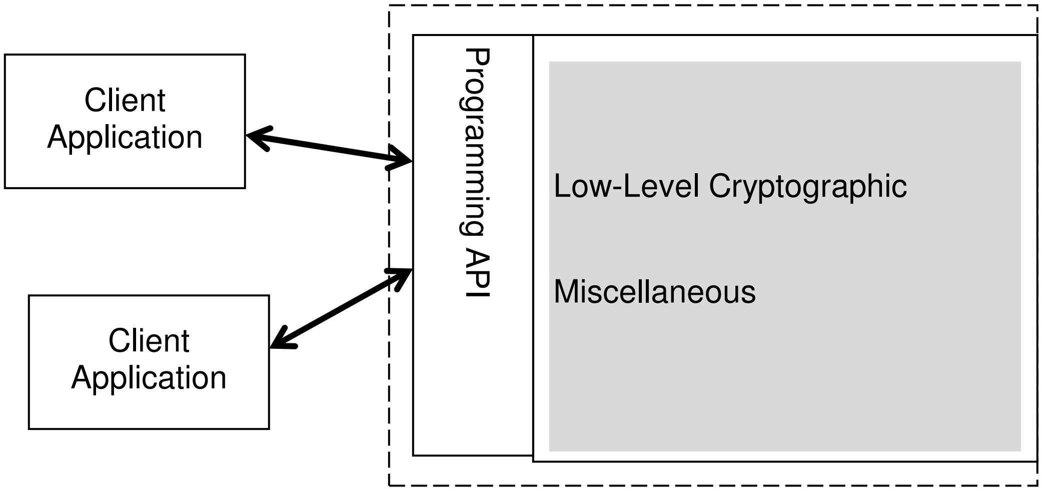
- AES (EME2 mode)

2.2 Cryptographic Boundary

The physical cryptographic boundary is defined as the computer's case that the PGP Cryptographic Engine is installed in and includes all the accompanying hardware. The module’s logical cryptographic boundary is defined to be a subset of the PGP Cryptographic Engine binary software library as defined by the Roles and Services Section of this document.

An operator is accessing (or using) the module whenever one of the library calls is executed through the API and thus the module logical interfaces are provided by the API calls.

Figure 1: Module Cryptographic Boundary



Note that the dashed line represents the PGP Cryptographic Engine crypto boundary.

2.3 Ports and Interfaces

The module restricts all access to its Critical Security Parameters (CSPs) through the API calls as enumerated in the Roles and Services Section of this document. This API acts as the logical interface to the module.

Although the computer's physical ports such as keyboards, mouse, displays, hard disks, smart card interfaces, etc. provide a means to access the cryptographic module, the actual interface is via the API itself.

For the purpose of FIPS 140-2, the logical interfaces can be modeled as described in the following table.

Table 3: PGP Cryptographic Engine Logical Ports

Data Input	Parameters passed to the module via API calls.
Data Output	Data returned by the module via API calls.
Control Input	Control Input – API function calls.
Status Output	Error and status codes returned by API calls.

The module does not support maintenance ports. The general purpose PC used receives power via a power supply.

Input and output data can consist of plain-text, cipher-text, and cryptographic keys as well as other parameters. The module does not support a cryptographic bypass mode.

All data output is prohibited whenever an error state occurs or during the self-test process.

2.4 Security Level

The PGP Cryptographic Engine Module meets the overall security requirements of FIPS 140-2 Level 1.

Table 4: Module Security Level Specification

Security Requirements Area	**Level**
Cryptographic Module Specification	1
Cryptographic 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	3
Mitigation of Other Attacks	N/A

2.5 Operational Environment

The following Operating Systems were used to operationally test and validate the PGP Cryptographic Engine to the requirements of FIPS-140-2. Therefore these Operating Systems provide the highest level of assurance that the module will correctly operate.

Operating System configurations

- Mac OS X 10.5 (i386)
- Windows XP Professional SP2 (Kernel Mode)

As per FIPS Implementation guidance the PGP Cryptographic Engine module will remain compliant with the requirements of FIPS 140-2 when operating on the following Operating Systems provided that the general-purpose computer (GPC) uses the specified single user operating system/mode specified on the validation certificate, or another compatible single user operating system:

- Windows XP Professional SP2 (User Mode)
- Windows XP SP1
- Windows 2000 SP3
- Windows Vista
- Windows 7
- Mac OS X 10.4 (i386)
- Mac OS X 10.5 (x86_64)
- Mac OS X 10.6 (i386, x86_64)
- Unix
    - Linux, 32-bit: Fedora Core 4, 5, 6 Red Hat Enterprise Linux 3, 4
    - AIX5.2 and 5.3
    - Solaris 8 and 9
    - HPUX 11.11
    - Linux, 64-bit: Fedora Core 5

The operating system must be configured for single-user mode.

2.6 Approved Mode of Operation

The PGP Cryptographic Engine provides a FIPS 140-2 compliant mode of operation. Before any cryptographic operations can be performed, the client application must initialize (power-up) the module by invoking the `PGPSetFIPSModeHMAC() and PGPEnableFIPSMode()` functions and the following events will occur:

1. The module will perform a series of power on self-tests as detailed in the *Self-Tests* section of this document.
2. If a self-test fails, the module will return with a status or error indication.
3. If a self-test fails, the module will enter a FIPS persistent error state and no cryptographic functions will be allowed until re-initialization.
4. All data output will be prohibited whenever an error state occurs or during the self-test process.
5. Only FIPS Approved or allowed cryptographic algorithms as enumerated in the *Supported Algorithms* section of this documents can be used in FIPS mode.
6. Once in FIPS mode the module will maintain that mode until the module is shut down.

The client application can, at any time, test if the module is in FIPS compliant mode by performing the `PGPGetSDKErrorState()` and `PGPGetFeatureFlags()` API call.

An application can also check the module error state, reset the error state, and run all or any specific self-test through making the proper API calls.

3 Security Rules

The following is a list of security requirements that specify the Approved mode of operation and must be adhered to when complying with FIPS 140-2.

1. PGP Cryptographic Engine must be used as described in this document. Calling any function not listed under Section 4 of this document violates this security policy. For APIs that accept other functions as arguments, only those functions listed under Section 4 of this document are permitted.
2. All access to Critical Security Parameters (CSPs) must only be made through the API calls specified in Section 4 below.
3. Installation of the module is the responsibility of the Crypto-Officer.
4. The module must be installed on a host computer where the operating system is configured for single user mode.
5. The module provides a FIPS 140-2 compliant mode of operation. Before the module can be used, it must be initialized as described in the *Approved Mode of Operation* section of this document.
6. Only FIPS Approved or allowed cryptographic algorithms as enumerated in the *Supported Algorithms* section of this document are to be used.
7. The module inhibits data output during self-tests and error states. The data output interface is logically disconnected from the processes performing zeroization.
8. The zeroization process can be achieved the appropriate API function: `PGPFreeSymmetricCipherContext`, `PGPFreeCBCContext`, `PGPFreeCFBContext`, `PGPFreeHashContext`, `PGPFreeHMACContext` or `PGPWipeSymmetricCipher`.

4 Roles and Services

As mentioned earlier the only access to the cryptographic module is through the PGP Cryptographic Engine API. Thus, the module operator is defined as any client application that is linked to the PGP Cryptographic Engine shared library.

The cryptographic module supports two roles. An operator accesses both roles while using the module and the means of access is the same for both roles. A role is implicitly assumed based on the services that are accessed.

The roles are defined as the following:

- User: Shall be allowed to perform the module info service.
- Cryptographic Officer: Shall be allowed to perform all security relevant services provided by the module.

The following table lists the services in the cryptographic module organized by service category.

Table 5: Low-Level Cryptographic

Service	API	Description
Hash data	PGPContinueHash PGPCopyHashContext PGPFinalizeHash PGPFreeHashContext PGPNewHashContext PGPResetHash PGPGetHashSize	Create a hash value based on the provided data.
Compute HMAC on data	PGPContinueHMAC PGPFinalizeHMAC PGPFreeHMACContext PGPNewHMACContext PGPResetHMAC	Compute the message authentication code on the provided data using the provided key.
Encrypt data with symmetric key Encrypt data	PGPCBCEncrypt PGPCFBEncrypt PGPSymmetricCipherEncrypt PGPCBCGetSymmetricCipher PGPCFBGetSymmetricCipher PGPCFBSync PGPCopyCBCContext PGPCopyCFBContext PGPCopySymmetricCipherContext PGPFreeCBCContext PGPFreeCFBContext PGPFreeSymmetricCipherContext	Encrypt the provided data with the provided key using a symmetric cipher algorithm.

Table 5: Low-Level Cryptographic

Service	API	Description
with symmetric key (cont'd)	PGPGetSymmetricCipherSizes PGPCFBRandomCycle PGPCFBGetRandom PGPInitCBC PGPInitCFB PGPInitSymmetricCipher PGPNewCBCContext PGPNewCFBContext PGPNewSymmetricCipherContext PGPWashSymmetricCipher PGPWipeSymmetricCipher	
Decrypt data with symmetric key	PGPCBCDecrypt PGPCFBDecrypt PGPSymmetricCipherDecrypt	Decrypt the provided data with the provided key using a symmetric cipher algorithm.

Table 6: Miscellaneous

Service	API	Description
Data storage management	PGPFreeDataExternal PGPNewDataExternal PGPNewSecureDataExternal PGPReallocDataExternal PGPNewMemoryMgrPosix PGPNewMemoryMgrExternal PGPNewMemoryMgrInternal PGPValidateMemoryMgrExternal	Create and free memory for holding various.

Table 6: Miscellaneous

Service	**API**	**Description**
Run self-tests	PGPRunAllSDKSelfTests PGPRunSDKSelfTest	Run the required self-tests.
Enable FIPS compliant mode	PGPEnableFIPSMode PGPSetFIPSModeHMAC	Power-Up the module into FIPS 140-2 mode
Module Info	PGPGetVersionString	Obtain module version number information.

5 Access Control Policy

In the PGP Cryptographic Engine, access to critical security parameters is controlled. A module User or Cryptographic Officer can only read, modify, or otherwise access the security relevant data through the cryptographic module services provided by the module API interface. This section details the Critical Security Parameters (CSPs) in the cryptographic module that a User or Cryptographic Officer can access, how the CSPs can be accessed in the cryptographic module, and which services are used for access to the data item.

A PGP Cryptographic Engine operator using the module in the Cryptographic Officer role can access all of the module services, but an operator in the User role can only use a subset of those functions. More information on this can be found in the *Roles and Services* section of this document.

5.1 Critical Security Parameters

The Critical Security Parameters (CSPs) used by the PGP Cryptographic Engine module are protected from unauthorized disclosure, modification, and substitution.

Definition of CSPs:

- TDES Encrypting Key - used to TDES encrypt/decrypt data.
- AES Encrypting Key - used to AES encrypt/decrypt data.
- HMAC Key - used for message authentication of data.

5.2 Accesses

The types of access to CSPs in the PGP Cryptographic Engine module are listed in the following table.

Table 7: CSP Access Types

Access	Description
create	The item is created.
destroy	The item is destroyed, in other words the data is cleared from any memory in the cryptographic module and then that memory is released.
read	The item is accessed for reading and use.
write	The item is modified or changed.

5.3 Service to CSP Access Relationship

The following table shows which CSPs are accessed by each service, the role(s) the operator must be in for access, and how the CSP is accessed on behalf of the operator when the service is performed.

Several services provided by the PGP Cryptographic Engine module do not access any CSPs and are included here for completeness.

Table 8: Module Services vs. Role Access

Service	CO	User	CSP	create	destroy	read	write
Encrypt data with symmetric key	X		TDES Encrypt Key AES Encrypt Key	●	●	●	●
Decrypt data with symmetric key	X		TDES Encrypt Key AES Encrypt Key	●	●	●	●
Hash data	X		N/A				
Compute HMAC on data	X		HMAC Key	●	●	●	●
Data storage management	X		N/A				
Module status	X		N/A				
Run self-tests	X		N/A				
Enable FIPS compliant mode	X		N/A				
Module Info		X	N/A				

6 Cryptographic Key Management

The PGP Cryptographic Engine takes a number of steps to protect secret keys and CSPs from unauthorized disclosure, modification, and substitution throughout the key life cycle. The module receives the keys from calling application via a pointer to memory with the function invocation. The module does not output keys in any form.

6.1 Key Generation and Establishment

Not Applicable.

7 Physical Security Policy

The PGP Cryptographic Engine is implemented as a software module, and as such the physical security section of FIPS 140-2 is not applicable.

8 Self-Tests

The PGP Cryptographic Engine provides for two forms of self-tests: power-on, and on-demand. Since the PGP Cryptographic Engine module provides a special FIPS Approved mode of operation, the module is defined as powering-up into FIPS mode when the `PGPSetFIPSModeHMAC() and PGPEnableFIPSMode()` call are made by the client application.

All data output is prohibited during the self-test process.

If any of these test fail, the module will enter an error state, which can only be cleared by a deliberate call to `PGPResetSDKErrorState()` or by powering down the module. Once in an error state, all further cryptographic operations and data output is disabled.

The resultant test status will be also returned by the `PGPEnableFIPSMode()` call. A client application can also ascertain module at anytime by using the `PGPGetSDKErrorState()` function. Possible error codes returned by the self-test routines include:

- `kPGPError_NoErr` – Self-test was successful.
- `kPGPError_SelfTestFailed` – Self Test Failed.

8.1 Power-Up Tests

When the client application invokes the `PGPSetFIPSModeHMAC() and PGPEnableFIPSMode()` API call, the module will start the self-test process. The following self-tests will run and in the following order until all the tests have been completed or one of the tests fail.

- Triple-DES – Runs known answer test of a 64-byte block in ECB, CBC and CFB modes, then decrypts the block checking for consistency with original plain-text.
- AES - Runs known answer test of a 16 byte block in ECB mode for 128, 192 and 256 bit keys then decrypts the block checking for consistency with original plain-text
- SHA-1, 256, 384, 512 – Runs a series of known answer tests using test vectors derived from the FIPS-180-3 document.
- HMAC/SHA-1, 256, 384, 512 – Runs a series of known answer tests
- Software Integrity – The module uses HMAC-SHA-256 used for module software integrity verification.

8.2 On-Demand Tests

After the `PGPEnableFIPSMode()` call is complete, the client application can optionally initiate a specific test or all tests on demand by using the `PGPRunSDKSelfTest()` or `PGPRunAllSDKSelfTests()` functions respectively. Note that if the on-demand tests fail, the module will enter an error state in a manner identical to the power-on self-tests.

9 Mitigation of Other Attacks

The Mitigation of Other attacks security section of FIPS 140-2 is not applicable to the PGP Cryptographic Engine module.

Glossary

API: Application Programming Interface.

Context: a reference value used to accept (or refer to) an internal PGP Cryptographic Engine state for an ongoing operation. Examples of contexts include “symmetric cipher context” or “hash context.”

CSP: Critical Security Parameters.

Cipher: a cryptographic algorithm used for encryption and decryption.

Client: Application or program calling into the PGP Cryptographic Engine API.

FIPS: Federal Information Processing Standards.

FIPS 140-2: FIPS for cryptographic modules.

FIPS Mode: FIPS 140-2 compliant mode of operation for PGP Cryptographic Engine.

Low-level cryptographic: a low-level CAPI that includes the intimate details for specific cryptographic algorithms.

MAC: Message Authentication Code.

NIST: National Institute of Standards and Technology.

Symmetric Key: key material used for symmetric ciphers. A symmetric key can be provided by the operator or created as needed by the PGP Cryptographic Engine.