Operating System

Microsoft Kernel Mode Cryptographic Module

FIPS 140-1 Documentation: Security Policy

9/20/2000 11:25:51 AM

Abstract

This document specifies the security policy for the Kernel Mode Cryptographic Module (FIPS.SYS) as described in FIPS PUB 140-1.

CONTENTS

{ TC "INTRODUCTION" \F SP }INTRODUCTION

Microsoft Kernel Mode Cryptographic Module (FIPS.SYS) is a FIPS 140-1 Level 1 compliant, general-purpose, software-based, cryptographic module residing at the Kernel Mode level of the Windows Operating System. It runs as a kernel mode export driver (a kernel-mode DLL) and encapsulates several different cryptographic algorithms in an easy-to-use cryptographic module accessible by other kernel mode drivers. It can be linked into other kernel mode services to permit the use of FIPS 140-1 Level 1 compliant cryptography.

Cryptographic Boundary

The Kernel Mode Cryptographic Module (FIPS.SYS) consists of a single kernel mode export driver (SYS). The cryptographic boundary for FIPS.SYS is defined as the enclosure of the computer system on which the cryptographic module is to be executed. The physical configuration of the module, as defined in FIPS PUB 140-1, is Multi-Chip Standalone.

{ TC "SECURITY POLICY" \F SP }SECURITY POLICY

FIPS.SYS operates under several rules that encapsulate its security policy.

- FIPS.SYS is supported on Windows 2000 with Service Pack 2 or later.
- FIPS.SYS relies on Microsoft Windows 2000 for the authentication of users.
- FIPS.SYS enforces a single role, Authenticated User, which is a combination of the User and Cryptographic Officer roles as defined in FIPS PUB 140-1.
- All users authenticated by Microsoft Windows 2000 employ the Authenticated User role.
- All cryptographic services implemented within FIPS.SYS are available to kernel mode system services, which are a part of Windows operating system trusted computer base (TCB).
- Windows 2000 operating system requires each user to be successfully authenticated before any system services may act on behalf of that user.
- All services implemented within FIPS.SYS are available to the Authenticated User role.
- Keys created within FIPS.SYS for one user are not accessible to any other user via FIPS.SYS.
- FIPS.SYS performs the following self-tests upon power up:
  - DES ECB encrypt/decrypt
  -
  - 3DES (3 key) ECB encrypt/decrypt
  - DES CBC encrypt/decrypt
  -
  - 3DES (3 key) CBC encrypt/decrypt
  - 3DES ECB encrypt/decrypt
  - SHA-1 hash

{ TC "SPECIFICATION OF ROLES" \F SP }SPECIFICATION OF ROLES

FIPS.SYS combines the User and Cryptographic Officer roles (as defined in FIPS PUB 140-1) into a single role hereon called the Authenticated User role. The Authenticated User may access all services implemented in the cryptographic module. Windows 2000 operating system requires each user to be successfully authenticated before any system services may act on behalf of that user.
To use a DES or Triple DES function, a kernel mode system service needs to provide a DES or Triple DES key respectively to the crypto module. Keys are zeroized after FIPS.SYS completes a DES or Triple DES function with the keys.

Maintenance Roles

Maintenance roles are not supported by FIPS.SYS.

Multiple Concurrent Operators

FIPS.SYS is intended to run on Windows 2000 with Service Pack 2 or later in Single User Mode. When run in this configuration, multiple concurrent operators are not supported.

{ TC "SPECIFICATION OF SERVICES" \F SP }SPECIFICATION OF SERVICES

The following list contains all services available to an operator. All services are accessible by all Authenticated Users, the one and only role supported by FIPS.SYS.

Key Storage

FIPS.SYS does not store keys. DES and Triple DES keys are zeroized after used.

Cryptographic Module Power Up and Power Down

DriverEntry

Each Windows 2000 driver must have a standard initialization routine DriverEntry in order to be loaded. The Windows 2000 Loader is responsible to call the DriverEntry routine. The DriverEntry routine must have the following prototype.

```
NTSTATUS
(*PDRIVER_INITIALIZE) (
        IN      PDRIVER_OBJECT      DriverObject,
        IN      PUNICODE_STRING     RegistryPath
        );
```

The input DriverObject represents the driver within the Windows 2000 system. Its pointer allows the DriverEntry routine to set an appropriate entry point for its DriverUnload routine in the driver object.

The RegistryPath input to the DriverEntry routine points to a counted Unicode string that specifies a path to the driver's registry key \Registry\Machine\System\CurrentControlSet\Services\FIPS.

DriverUnload

It is the entry point for the driver's unload routine. The pointer to the routine is set by the DriverEntry routine in the DriverUnload field of the DriverObject when the driver initializes. An Unload routine is declared as follows:

```
VOID
(*PDRIVER_UNLOAD) (
        IN      PDRIVER_OBJECT      DriverObject
        );
```

When the driver is no longer needed, the Windows 2000 Kernel is responsible to call the DriverUnload routine of the associated DriverObject.

Key Formatting

The following functions provide interfaces to the cryptomodule’s key formatting functions.

FipsDesKey

```
VOID
FipsDesKey(
          DESTable *      pDesTable,
          UCHAR *         pbKey
          )
```

The FipsDesKey function formats a DES cryptographic session key into the form of a DESTable struct. It fills in the DESTable struct with the decrypt and encrypt key expansions. Its second parameter points to the DES key of DES_BLOCKLEN (8) bytes. FipsDesKey zeroises its copy of the key before returning to the caller.

Fips3Des3Key

```
VOID
Fips3Des3Key(
          DES3TABLE *     pDES3Table,
          UCHAR *         pbKey
          )
```

The Fips3Des3Key function formats a Triple DES cryptographic session key into the form of a DES3Table struct. It fills in the DES3Table struct with the decrypt and encrypt key expansions. Its second parameter points to the Triple DES key of 3 * DES_BLOCKLEN (24) bytes. Fips3Des3Key zeroises its copy of the key before returning to the caller.

Random Number Generation

FipsGenRandom

```
BOOL
FIPSGenRandom(
          In OUT  UCHAR *         pb,
          IN      ULONG           cb
          );
```

The FipsGenRandom function fills the buffer pb with cb random bytes produced using a FIPS 140-1 compliant pseudo random number generation algorithm. The algorithm is the SHS based RNG from FIPS 186. Internally, the function compares each 160 bits of the buffer with the next 160 bits. If they are the same, the function returns FALSE. The caller may optionally specify the initial 160 bits in the pb buffer for the initiation of the comparison. This initial 160 bit sequence is used only for the comparison algorithm and it is not intended as caller supplied random seed.

Data Encryption and Decryption

The following functions provide interfaces to the cryptomodule's data encryption and decryption functions.

FipsDes

```
VOID
FipsDes(
        UCHAR *         pbOut,
        UCHAR *         pbIn,
        void *          pKey,
        int             iOp
        );
```

The FipsDes function encrypts or decrypts the input buffer pbIn using DES, putting the result into the output buffer pbOut. The operation (encryption or decryption) is specified with the iOp parameter. The pKey is a DESTable struct pointer returned by the FipsDesKey function. FipsDes zeroises its copy of the DESTable struct before returning to the caller.

FipsDes3

```
VOID
Fips3Des(
        UCHAR *         pbIn,
        UCHAR *         pbOut,
        void *          pKey,
        int             op)
```

The FipsDes3 function encrypts or decrypts the input buffer pbIn using Triple DES, putting the result into the output buffer pbOut. The operation (encryption or decryption) is specified with the op parameter. The pkey is a DES3Table struct returned by the Fips3Des3Key function. FipsDes3 zeroises its copy of the DES3Table struct before returning to the caller.

FipsCBC

```
BOOL FipsCBC(
        ULONG           EncryptionType,
    DWORD           dwBlockLen,
    BYTE   *        output,
    BYTE   *        input,
    void   *        keyTable,
    int             op,
    BYTE   *        feedback
)
```

The FipsCBC function encrypts or decrypts the input buffer input using CBC mode, putting the result into the output buffer output. The encryption algorithm (DES or Triple DES) to be used is specified with the EncryptionType parameter. The operation (encryption or decryption) is specified with the op parameter.

If the EncryptionType parameter specifies Triple DES, the keyTable is a DES3Table struct returned by the Fips3Des3Key function. If the EncryptionType parameter specifies DES, the keyTable is a DESTable struct returned by the FipsDesKey function.

This function encrypts just one block at a time and assumes that the caller knows the algorithm block length and the buffers are of the correct length. Every time when the function is called, it zeroises its copy of the DES3Table or DESTable struct before returning to the caller.

Hashing

The following functions provide interfaces to the cryptomodule's hashing functions.

FipsSHAInit

```
void
FipsSHAInit(
        A_SHA_CTX *    hash_context
        )
```

The FipsSHAInit function initiates the hashing of a stream of data. The output hash_context is used in subsequent hash functions.

FipsSHAUpdate

```
void FipsSHAUpdate(
        A_SHA_CTX *    hash_context,
        UCHAR *        pb,
        unsigned int   cb
        )
```

The FipsSHAUpdate function adds data pb of size cb to a specified hash object associated with the context hash_context. This function can be called multiple times to compute the hash on long data streams or discontinuous data streams. The FipsSHAFinal function must be called before retrieving the hash value.

FipsSHAFinal

```
void FipsSHAFinal (
        A_SHA_CTX *                            hash_context,
        unsigned char [A_SHA_DIGEST_LEN]       hash)
```

The FipsSHAFinal function computes the final hash of the data entered by the FipsSHAUpdate function. The hash is an array char of size A_SHA_DIGEST_LEN (20).

Acquiring a Table of Pointers to FipsXXX Functions

A kernel mode user of the FIPS.SYS driver must be able to reference the FipsXXX functions before using them. The user needs to acquire the table of pointers to the FipsXXX functions from the FIPS.SYS driver. The user accomplishes the table acquisition by building a Fips function table request irp (I/O request packet) and then sending the irp to the FIPS.SYS diver via the IoCallDriver function. Further information on irp and IoCallDriver can be found on Microsoft Windows 2000 Driver Development Kit.

{ TC "CRYPTOGRAPHIC KEY MANAGEMENT" \F SP }CRYPTOGRAPHIC KEY MANAGEMENT

The FIPS.SYS cryptomodule manages keys in the following manner.

Key Material

FIPS.SYS use keys provided by the caller for the following algorithms: DES, 3DES and 3DES 112.

Key Generation

Random keys can be generated by calling the FipsGenRandom() function. DES key generated in this way meet the requirements described in FIPS PUB 46-2 and FIPS PUB 81.

Key Entry and Output

DES keys can be imported into FIPS.SYS via FipsDesKey(). DESTable struct can be exported out of FIPS.SYS via FipsDesKey(). DESTable struct can be imported into FIPS.SYS via FipsDes() or FipsCBC().

Triple DES keys can be imported into FIPS.SYS via Fips3Des3Key(). DES3Table struct can be exported out of FIPS.SYS via Fips3Des3Key(). DES3Table struct can be imported into FIPS.SYS via Fips3Des3() or FipsCBC().

Key Storage

FIPS.SYS does not store keys. DES and Triple DES keys and their associated DESTable and DES3Table struct are zeroized after used.

Key Archival

FIPS.SYS does not archive cryptographic keys. All key copies inside FIPS.SYS are destroyed and their memory location zeroized after used. It is the caller’s responsibility to maintain the security of DES and Triple DES keys when the keys are outside FIPS.SYS.

Key Destruction

All DES and Triple DES key copies and their associated DESTable and DES3Table struct copies inside FIPS.SYS are destroyed and their memory location zeroized after they have been used in FipsDes, FipsDes3, or FipsCBC.

{ TC "SELF-TESTS" \F SP }SELF-TESTS

Mandatory

Software tests via a DES MAC of library image

- DES ECB encrypt/decrypt KAT
- 3DES ECB encrypt/decrypt KAT
- DES CBC encrypt/decrypt KAT
- 3DES CBC encrypt/decrypt KAT
- SHA-1 hash KAT

{ TC "MISCELLANEOUS" \F SP }MISCELLANEOUS

The following items address requirements not addressed above.

Cryptographic Bypass

Cryptographic bypass is not support in FIPS.SYS.

Operation Authentication

FIPS.SYS inherits all authentication from the Microsoft Windows 2000 operating system upon which it runs. Microsoft Windows 2000 requires authentication from a trusted control base (TCB) before a user is able to access system services. Once a user is authenticated from the TCB, a process is created bearing the Authenticated User's security token. All subsequent processes and threads created by that Authenticated User are implicitly assigned the parent's (thus the Authenticated User's) security token. Every user that has been authenticated by Microsoft Windows 2000 is naturally assigned the Authenticated User role.

Operating System Security

The FIPS.SYS cryptomodule is intended to run on Windows 2000 with Service Pack 2 or later in the Single User Mode.

When the Windows 2000 operating system Loader loads the cryptomodule into memory, the cryptomodule runs a DES MAC on the cryptomodule's disk image of FIPS.SYS, excluding the DES MAC, checksum, and export signature resources. This MAC is compared to the value stored in the DES MAC resource. Initialization will only succeed if the two values are equal.

{ TC "FOR MORE INFORMATION" \F SP }FOR MORE INFORMATION

For the latest information on Windows 2000 Server, check out our World Wide Web site at http://www.microsoft.com/windows2000.

Operating System

Microsoft Kernel Mode Cryptographic Module

FIPS 140-1 Documentation: Finite State Machine

9/20/2000 11:25:51 AM

Abstract

This document specifies the finite state machine for the Kernel Mode Cryptographic Module (FIPS.SYS) as described in FIPS PUB 140-1.

CONTENTS

{ TC "INTRODUCTION" \F FSM }INTRODUCTION

Microsoft Kernel Mode Cryptographic Module (FIPS.SYS) is a FIPS 140-1 Level 1 compliant, general-purpose, software-based, cryptographic module residing at the Kernel Mode level of the Windows Operating System. It runs as a kernel mode export driver (a kernel-mode DLL) and encapsulates several different cryptographic algorithms in an easy-to-use cryptographic module accessible by other kernel mode drivers. It can be linked into other kernel mode services to permit the use of FIPS 140-1 Level 1 compliant cryptography.

{ TC "FINITE STATE MACHINE" \F FSM }FINITE STATE MACHINE

The FIPS.SYS cryptomodule can be in exactly one of the following states at any given moment. Transitions between states can be automatic or result from user intervention.

States

See Appendix A and B for more information.

Power Up

The Power Up state is entered when Windows 2000 Loader calls the FIPS.SYS driver entry point function DriverEntry() during system boot.

Power Down

The Power Down state is entered when Windows 2000 Kernel calls the FIPS.SYS driver's unload function which was set in DriverUnload field of the DriverObject representing FIPS.SYS during the Power Up state.

Init Error

The Init Error State is entered when FIPS.SYS's DriverEntry() fails as a result of either configuration errors (i.e. not enough memory, etc.) or errors resulting from the power up self-tests.

Initialized

The Initialized state is entered when FIPS.SYS's DriverEntry() returns successfully and the Windows Loader completes the loading of FIPS.SYS.

Key Initialized

The Key Initialized state is entered after keys are formatted into a DESTable or DES3Table struct with FipsDesKey(), Fips3Des3Key() .

Operation Error

The Operation Error state is entered whenever an error occurs as a result of a cryptographic operation. FIPS.SYS will automatically transition back to either the Initialized or Key Initialized state depending on whether or not keys have been successfully formatted into a DESTable or DES3Table struct.

State Transitions

See Appendix A.

State Diagrams

See Appendix B.

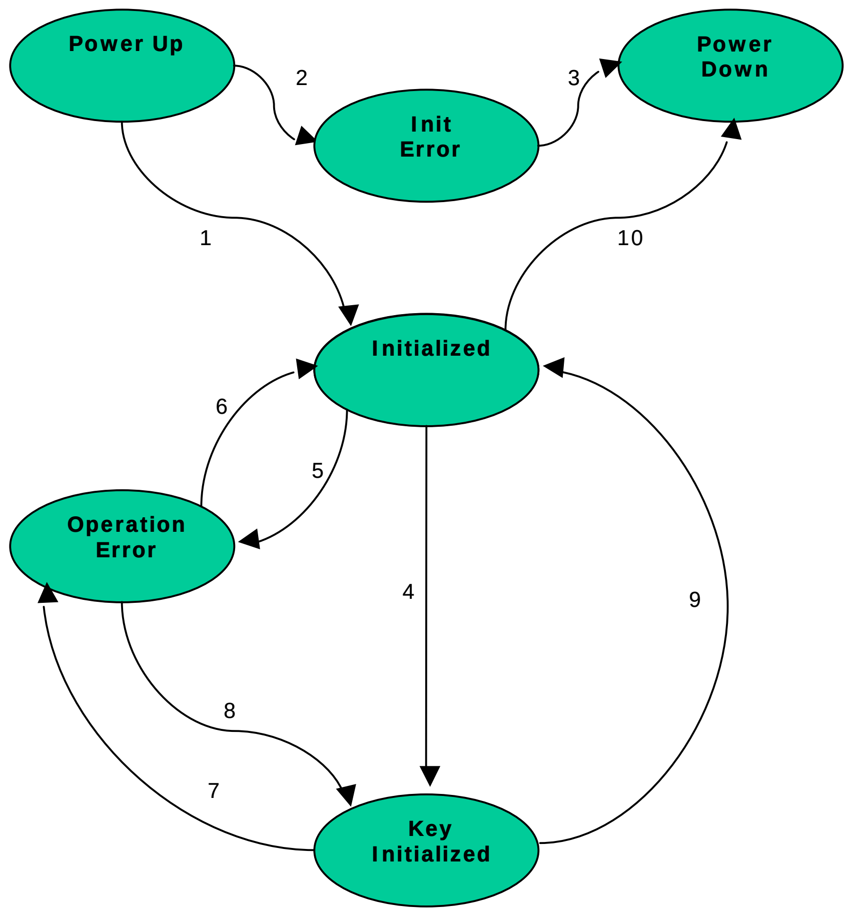
{ TC "APPENDIX A" \F FSM }APPENDIX A

The following table describes the state transitions possible within the FIPS.SYS cryptomodule during operation.

	Current State	Input	Output	Next State
1	Power Up	FIPS.SYS loads	NO_ERROR	Initialized
2	Power Up	FIPS.SYS not found	STATUS_UNSUCCESSFUL	Init Error
2	Power Up	DES MAC check on cryptographic provider fails	STATUS_UNSUCCESSFUL	Init Error
2	Power Up	One or more power-on cryptographic self-tests fail	STATUS_UNSUCCESSFUL	Init Error
2	Power Up	System error	STATUS_UNSUCCESSFUL	Init Error
3	Init Error	Automatic transition	No output	Power Down
4	Initialized	Key formatting operation (i.e. FipsDesKey(), Fips3Des3Key()) requested	No output	Key Initialized
5	Initialized	Key formatting operation failure	Operation specific error message	Operation Error
6	Operation Error	Automatic transition when keys have not yet been initialized	No output	Initialized
7	Key Initialized	Generic cryptographic operation failure	Operation specific error message	Operation Error
8	Operation Error	Automatic transition when keys have already been initialized	No output	Key Initialized
9	Key Initialized	Generic cryptographic operation (i.e. FipsDes(), Fips3Des(), or FipsCBC ()) completed	NO_ERROR	Initialized
10	Initialized	Automatic transition when Windows 2000 Kernel calls the FIPS.SYS driver's unload function	NO_ERROR	Power Down

{ TC "APPENDIX B" \F FSM }APPENDIX B

The following diagram illustrates the finite state machine of the FIPS.SYS cryptomodule.



{ TC "FOR MORE INFORMATION" \F FSM }FOR MORE INFORMATION

For the latest information on Windows 2000 Server, check out our World Wide Web site at http://www.microsoft.com/windows2000.

Operating System

Microsoft Kernel Mode Cryptographic Module

FIPS 140-1 Documentation: Master Component List

9/20/2000 11:25:51 AM

Abstract

This document specifies the master component list for the Kernel Mode Cryptographic Module (FIPS.SYS) as described in FIPS PUB 140-1.

CONTENTS

{ TC "MASTER COMPONENT LIST" \F MCS }MASTER COMPONENT LIST

The FIPS.SYS cryptomodule is a software cryptomodule and is intended to operate on a PC running Windows 2000 with Service Pack 2 or later. Several components of the base PC are also to be considered components of the cryptomodule.

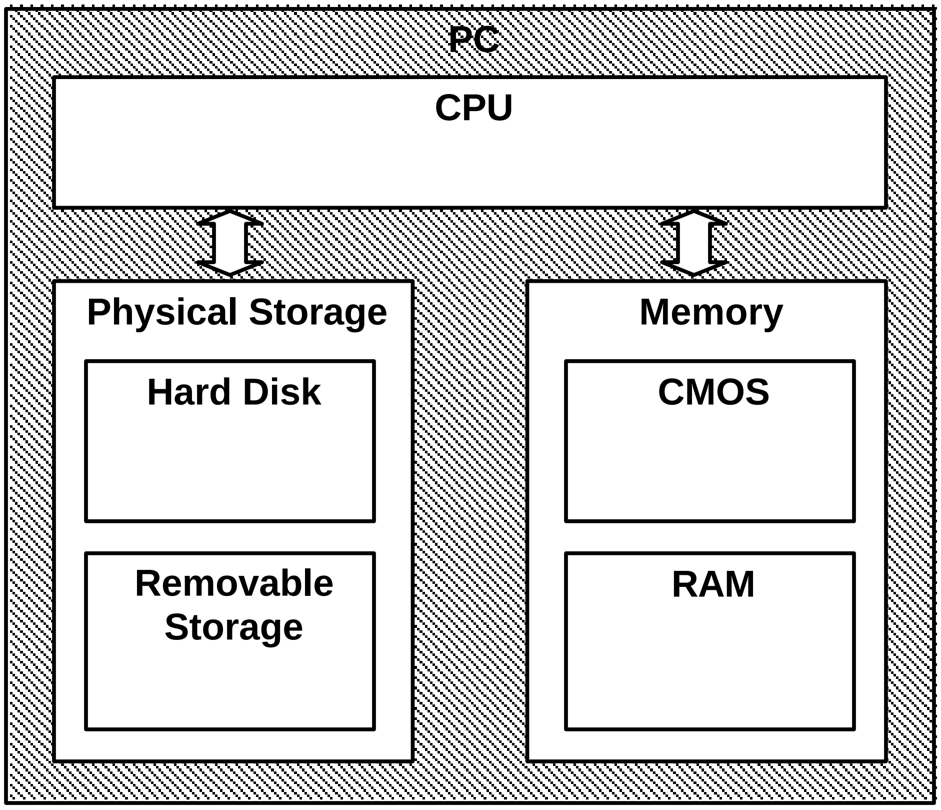
Components

The following components are to be considered components of the cryptomodule (see Appendix A below):

- PC Enclosure
- Central Processing Unit (CPU)
- Physical Storage (Hard Drives and Removable Storage)
- Memory (RAM and CMOS)

{ TC "APPENDIX A" \F MCS }APPENDIX A

The following diagram illustrates the master components of the RSAENH cryptomodule.



{ TC "FOR MORE INFORMATION" \F MCS }FOR MORE INFORMATION

For the latest information on Windows 2000 Server, check out our World Wide Web site at http://www.microsoft.com/windows2000.