Vulnerability to Flash Controller for Secure USB Drives

Jaein Kim, Youngjun Lee, Kyungroul Lee,
Taeyoung Jung, Dmitry Volokhov, and Kangbin Yim*
Soonchunhyang University, Asan, Republic of Korea
{boxbop, dogehk, carpedm, jtyworld, dmitry, yim}@sch.ac.kr

Abstract

This paper analyzes a vulnerability in the flash controller for secure USB drives, which is meant to enable secure USB drives to prevent unauthorized access to the data stored on them. This controller divides a driver into several partitions, one of which is configured as a secure area to store secret information. Generally, secure USB drives supporting multiple partitions may have three different areas, such as a CD-ROM area, a secure area, and a hidden area. The CD-ROM area stores the application software that manages security functions, the secure area stores the data that users wish to protect, and the hidden area stores secure information for user authentication. In this environment, it is a requirement that no one can access the data stored in the secure area when the user authentication fails. Nevertheless, attackers can access the secure area if they manipulate a vulnerability in the flash controller within the USB flash drive. In this paper, we analyze and verify this vulnerability. We expect our results will provide manufacturers with useful information for making a more secure USB flash controller.

Keywords: Vulnerability Analysis, Secure USB drive, Flash Controller

1 Introduction

USB flash drives have become prevalent in the last decade due to their portability and availability. However, the USB specification on which USB flash drives are based does not support data protection by itself, since the protocol did not consider security applications at the outset [6]. Thus, the data stored in a USB flash drive should be protected by hardware manufacturers or software developers through additional security technologies. This type of protected drive is commonly called a Secure USB Flash Drive and it has been recommended to use these secure USB drives to protect secret data stored on them.

Functions for protecting data in a USB flash drive consist of user authentication and data encryption/decryption. The user authentication function is responsible for access control to the stored data. The encryption/decryption function is responsible for transforming the secret data. In realizing these functions, various implementations have been found, though they are categorized into three different approaches; one is software-based, another one is hardware-based partitioning, and another one is hardware-based. The software-based approach performs both functions for user authentication and encryption/decryption within the application software, outside the drive. The hardware-based partitioning method supports multiple partitions in one physical drive, divided by a hardware device. The hardware-based method encrypts or decrypts the stored data inside the USB flash drive by generating the key within it.

Even though they incorporate these security functions explained above, some secure USB memory devices have various vulnerabilities due to design and implementation mistakes. These vulnerabilities can leak the stored data [8, 5, 7, 1, 2]. Most such vulnerabilities can be detected through reverse

*Corresponding author: Dept. of Information Security Engineering, Soonchunhyang Univeristy, 646 Eupnae, Shinchang, Asan, Republic of Korea, Tel +82-(0)41-5301741, Web: `http://lisa.sch.ac.kr` engineering that is one of the analyzing methods of security functions, and attackers can bypass user authentication and steal decryption keys by using the results of that analysis. Published research reports on hardware modules have not included detailed explanations of these kinds of attacks and there has not been much recent research on this in any case, so new studies are needed. Therefore, in this paper, we select the SM3254 chipset that is supported by Silicon Motion Corporation [9], as a commonly used secure USB flash drive controller that utilizes dividing partitions in USB memory, and analyze the operating process of it. We illustrate the results of our vulnerability-based analysis.

2 Implementations of secure USB memories

2.1 Incorporated functional elements

Secure storage technology basically involves two different functions for authenticity, integrity and confidentiality of protected data: user authentication and data encryption [11], although the encryption may serve the dual purposes of authentication and data protection. Authentication is a verification process of which role is to prevent unauthorized access to the data stored in the device by applying some user input to an oracle that was priorly registered for an authenticated user [3].

Data encryption is required for confidentiality, which requires a cryptographic algorithm and a dedicated key. Only users who have the dedicated key can decrypt the secret information. Meanwhile, it is feasible to check the validity and integrity of the decrypted data once the user provides the appropriate key for the decryption. If the data does not decrypt properly, it is assumed that an incorrect key has been provided. Preserving the integrity and confidentiality of the data are the key aspects of encryption.

2.2 Implementation methodologies

Commercial secure USB flash drives can be classified into different types due to the methodologies that they use to implement the above functional elements. Some drives implement the security functions only by the host software. In this case, a normal USB memory can be used to implement a secure USB flash drive. Because the host software only has the responsibility to protect the drive, anti reverse-engineering on the software is a critical challenge.

Some other drives can support hardware-based partitions to separate hidden or secure areas from normal sections of memory in an USB memory device. Some of them have facility to encrypt data by themselves inside of the device. This approach uses a specific hardware chip such as SM3xxx from Silicon Motion to generate multiple partitions, for example. User authentication and, optionally, data encryption, are delegated to the incorporated security chip [4] and the management software only has a role of controlling the transactions.

3 Attack surface of USB flash drives

Besides the vulnerabilities to the host functions of the management software, an experienced attacker can utilize different weak points to neutralize the security mechanisms incorporated in the controller of the USB flash drive. Generally speaking, these security mechanisms are dependent of the controller's internal architecture and it may not be easy to figure out the details. However, analysis on the controllers from the inside out usually can give chances for a possible attack due to their generalized fittings that are implemented with popular core architecture and interfaces. Consequently, even a limited set of documents is sometimes enough finding considerable vulnerabilities. Figure 1 shows the block diagram of SM3524, which is the most popular USB flash controller.

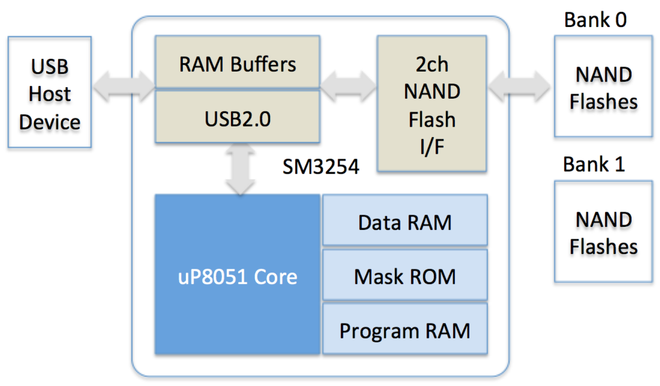


Figure 1: Block diagram of SM3254 [10]

3.1 Fishing on the bridge

Sniffing and Snooping the boundary of the controller is an essential step for research on vulnerability. Sniffing on the USB bus transactions, pin signals and memory interfaces is helpful for understanding the protocols and the internal states between the pair of the software. Stateful reconstruction between the control codes from the host driver and the commands into the NAND flash may display an outline of the implementation. Engineers need to apply some randomizing algorithms for practical implementation.

3.2 Fishing over the bridge

Dumping and Interpreting the control contexts inside the host driver and the NAND flashes is another step. Because the core architecture is based on Intel MCS51, the firmware is easily reversible when it is exposed. Because the device has Program RAM, it is assumed that an additional or upgraded code image can be downloaded into the Program RAM from the up-to-dated host driver. In this case, the program code might be exposed and reversed through scanning the host driver. Any scrambling algorithm should be applied between the host driver and the firmware of the controller to protect the context of the downloadable firmware.

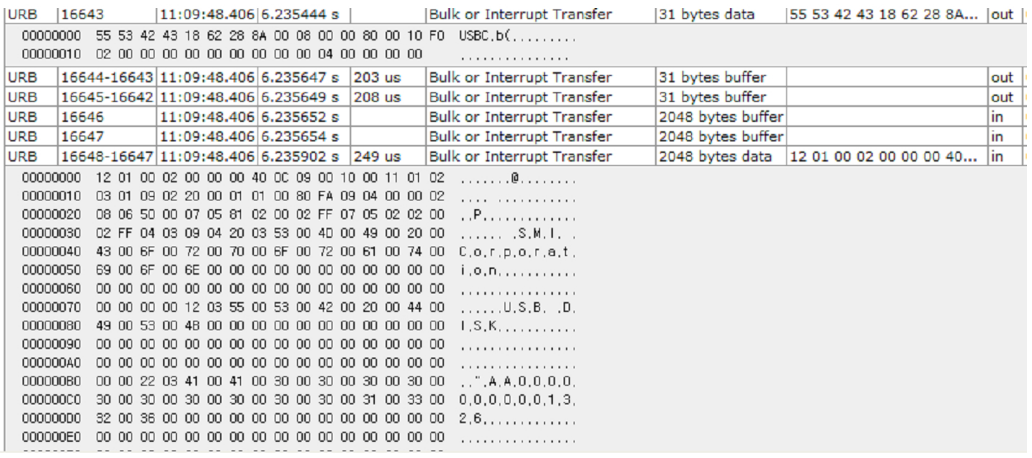
Achieving critical information such as encryption keys from inside NAND flash is another problem. No flash or NVRAM is found within the block diagram, which means non-volatile information required for the security functions should be stored into the external NAND flash block. Dynamically or statically retrieving or tracing the contents of the NAND flash can provide many hints to get the critical information. Usage of pages and blocks inside the flash devices can be understandable through a time consuming analysis.

4 Analysis on USB flash drive controller

In this paper, we have prepared an environment for analyzing vulnerabilities: Microsoft Windows XP for the operating system, OllyDbg for reverse engineering, Microsoft Visual Studio for implementation, USBlyzer for checking USB communication, and DeviceTree for checking organization of device drivers.

4.1 Structure of USB flash drive controller

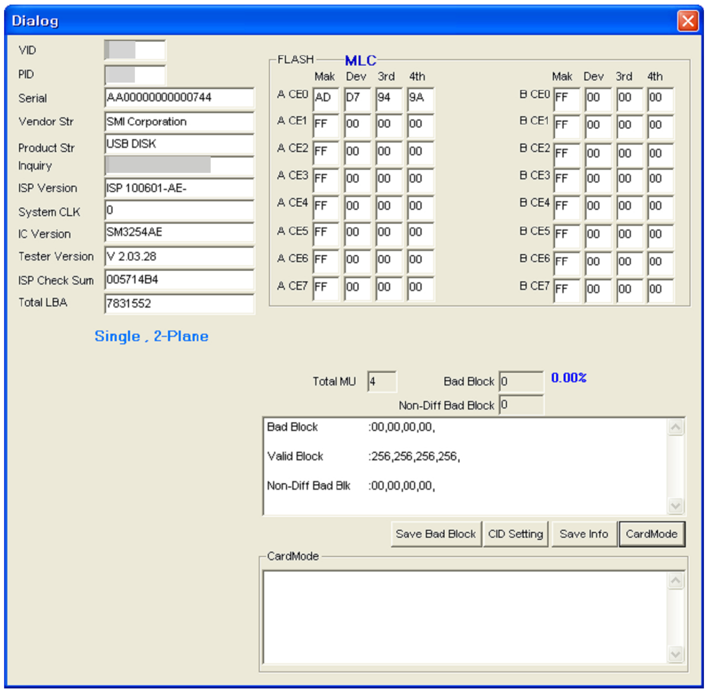
To confirm the manufacturer and chipset of the USB flash drive controller, we use USB sniffing and the result is shown in Figure 2.



```
URB |16643      |11:09:48.406|6.235444 s |       |Bulk or Interrupt Transfer |31 bytes data    |55 53 42 43 18 62 28 8A... |out |
  00000000  55 53 42 43 18 62 28 8A 00 08 00 00 80 00 10 F0  USBC.b(.........
  00000010  02 00 00 00 00 00 00 00 00 00 04 00 00 00 00      ...............
URB |16644-16643|11:09:48.406|6.235647 s |203 us |Bulk or Interrupt Transfer |31 bytes buffer  |                           |out |
URB |16645-16642|11:09:48.406|6.235649 s |208 us |Bulk or Interrupt Transfer |31 bytes buffer  |                           |out |
URB |16646      |11:09:48.406|6.235652 s |       |Bulk or Interrupt Transfer |2048 bytes buffer|                           |in  |
URB |16647      |11:09:48.406|6.235654 s |       |Bulk or Interrupt Transfer |2048 bytes buffer|                           |in  |
URB |16648-16647|11:09:48.406|6.235902 s |249 us |Bulk or Interrupt Transfer |2048 bytes data  |12 01 00 02 00 00 00 40...  |in  |
  00000000  12 01 00 02 00 00 00 40 0C 09 00 10 00 11 01 02  .......@........
  00000010  03 01 09 02 20 00 01 01 00 80 FA 09 04 00 00 02  .... ...........
  00000020  08 06 50 00 07 05 81 02 00 02 FF 07 05 02 02 00  ..P.............
  00000030  02 FF 04 03 09 04 20 03 53 00 4D 00 49 00 20 00  ...... .S.M.I. .
  00000040  43 00 6F 00 72 00 70 00 6F 00 72 00 61 00 74 00  C.o.r.p.o.r.a.t.
  00000050  69 00 6F 00 6E 00 00 00 00 00 00 00 00 00 00 00  i.o.n...........
  00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  00000070  00 00 00 00 12 03 55 00 53 00 42 00 20 00 44 00  ......U.S.B. .D.
  00000080  49 00 53 00 4B 00 00 00 00 00 00 00 00 00 00 00  I.S.K...........
  00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  000000B0  00 00 22 03 41 00 41 00 30 00 30 00 30 00 30 00  ..".A.A.0.0.0.0.
  000000C0  30 00 30 00 30 00 30 00 30 00 30 00 31 00 33 00  0.0.0.0.0.0.1.3.
  000000D0  32 00 36 00 00 00 00 00 00 00 00 00 00 00 00 00  2.6.............
  000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Figure 2: Verification of the vendor for the controller by sniffing

The sniffing result shows that the manufacturer is “SMI Corportation“, but there is no chipset information. Therefore we use a special tool that is able to check SM series chipsets, and Figure 3 shows the results. This tool is called sm32xtest and it can be downloaded from the web. This result means the USB flash drive controller is SM3254AE, as shown in the IC Version field. The result is from a special tool so we cannot prove that it is correct. Therefore, we should find the actual chipset by reverse engineering.



VID	[redacted]
PID	[redacted]
Serial	AA00000000000744
Vendor Str	SMI Corporation
Product Str	USB DISK
Inquiry	[redacted]
ISP Version	ISP 100601-AE-
System CLK	0
IC Version	SM3254AE
Tester Version	V 2.03.28
ISP Check Sum	005714B4
Total LBA	7831552

Single , 2-Plane

FLASH MLC

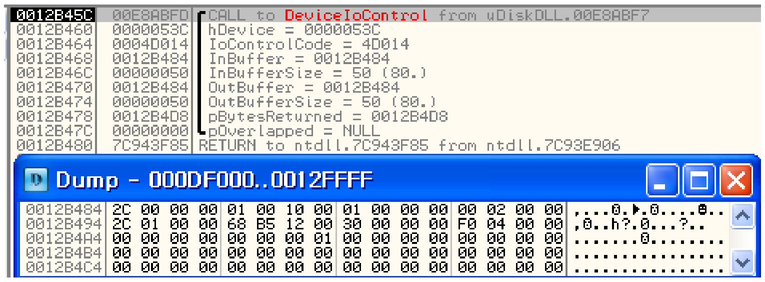
	Mak	Dev	3rd	4th		Mak	Dev	3rd	4th
A CE0	AD	D7	94	9A	B CE0	FF	00	00	00
A CE1	FF	00	00	00	B CE1	FF	00	00	00
A CE2	FF	00	00	00	B CE2	FF	00	00	00
A CE3	FF	00	00	00	B CE3	FF	00	00	00
A CE4	FF	00	00	00	B CE4	FF	00	00	00
A CE5	FF	00	00	00	B CE5	FF	00	00	00
A CE6	FF	00	00	00	B CE6	FF	00	00	00
A CE7	FF	00	00	00	B CE7	FF	00	00	00

Total MU 4 Bad Block 0 0.00%
Non-Diff Bad Block 0

```
Bad Block        :00,00,00,00,
Valid Block      :256,256,256,256,
Non-Diff Bad Blk :00,00,00,00,
```

Figure 3: Verification of the vendor by a dedicated tool

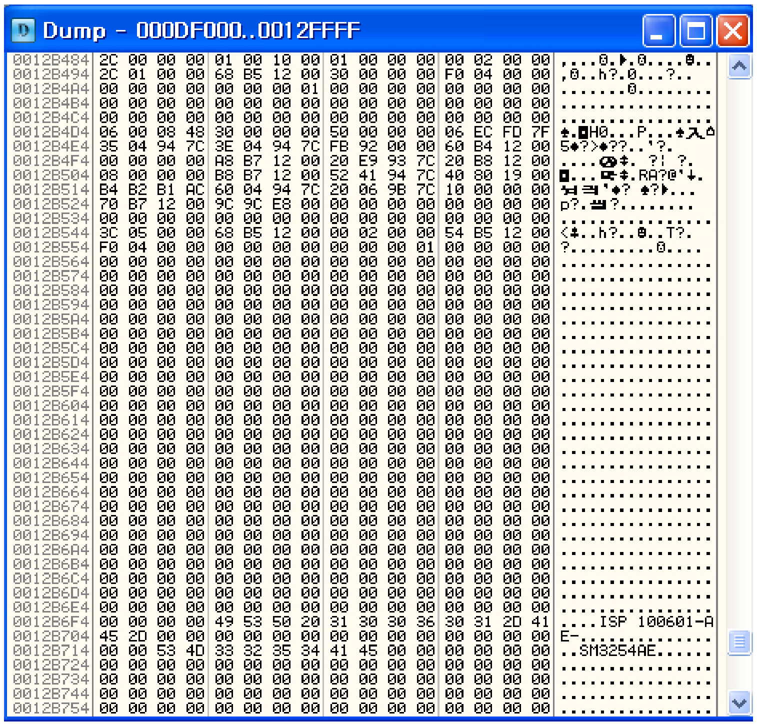
First of all, we assume that the USB flash drive controller communicates with the host through specific commands, such as requesting the chipset name, connection with the secure area and so on. To verify this assumption, we analyze the communication process between application and device driver using the DeviceIoControl function. The DeviceIoControl function is called from uDiskDLL and information about the chipset is stored at offset 0x292. This result is SM3254AE and it is equal to the result we got by using the tool we already mentioned. Figure 4 shows the command requesting the chipset name and Figure 5 shows the result when receiving the chipset name.



```
0012B45C  00E8ABFD  CALL to DeviceIoControl from uDiskDLL.00E8ABF7
0012B460  0000053C  hDevice = 0000053C
0012B464  0004D014  IoControlCode = 4D014
0012B468  0012B484  InBuffer = 0012B484
0012B46C  00000050  InBufferSize = 50 (80.)
0012B470  0012B484  OutBuffer = 0012B484
0012B474  00000050  OutBufferSize = 50 (80.)
0012B478  0012B4D8  pBytesReturned = 0012B4D8
0012B47C  00000000  pOverlapped = NULL
0012B480  7C943F85  RETURN to ntdll.7C943F85 from ntdll.7C93E906
```

```
0012B484 2C 00 00 00 01 00 10 00 01 00 00 00 00 02 00 00 ,...0.▶.0....☻..
0012B494 2C 01 00 00 68 B5 12 00 30 00 00 00 F0 04 00 00 ,0..h?.0...?..
0012B4A4 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 .......0........
0012B4B4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B4C4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

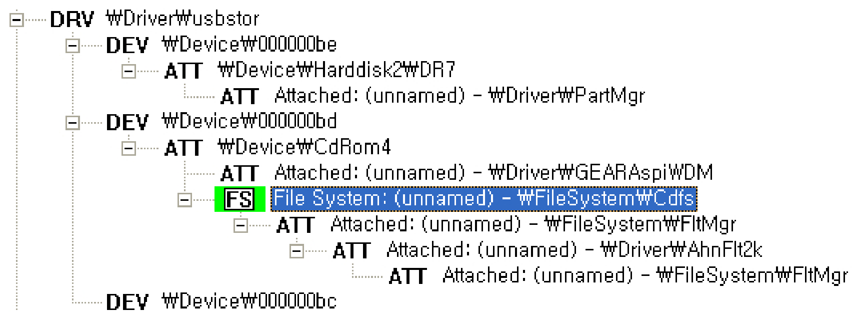
Figure 4: Reverse Engineering to find Controller Information



```
0012B484 2C 00 00 00 01 00 10 00 01 00 00 00 00 02 00 00 ,...0.▶.0....☻..
0012B494 2C 01 00 00 68 B5 12 00 30 00 00 00 F0 04 00 00 ,0..h?.0...?..
0012B4A4 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 .......0........
0012B4B4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B4C4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B4D4 06 00 08 48 30 00 00 00 50 00 00 00 06 EC FD 7F ♠.◘H0...P...♠?⌂
0012B4E4 35 04 94 7C 3E 04 94 7C FB 92 00 00 60 B4 12 00 5♦?>♦??..'?.
0012B4F4 00 00 00 00 A8 B7 12 00 20 E9 93 7C 20 B8 12 00 ....?↕. ?| ?.
0012B504 08 00 00 00 B8 B7 12 00 52 41 94 7C 40 80 19 00 ◘...?↕.RA?@'↓.
0012B514 B4 B2 B1 AC 60 04 94 7C 20 06 9B 7C 10 00 00 00 ??'♦? ♠?▶...
0012B524 70 B7 12 00 9C 9C E8 00 00 00 00 00 00 00 00 00 p?.?........
0012B534 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B544 3C 05 00 00 68 B5 12 00 00 02 00 00 54 B5 12 00 <♣..h?..☻..T?.
0012B554 F0 04 00 00 00 00 00 00 00 00 00 01 00 00 00 00 ?.........0....
0012B564 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B574 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B584 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B594 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5A4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5B4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5C4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5D4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5E4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B5F4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B604 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B614 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B624 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B634 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B644 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B654 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B664 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B674 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B684 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B694 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6A4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6B4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6C4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6D4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6E4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B6F4 00 00 00 00 49 53 50 20 31 30 30 36 30 31 2D 41 ....ISP 100601-A
0012B704 45 2D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E-..............
0012B714 00 00 53 4D 33 32 35 34 41 45 00 00 00 00 00 00 ..SM3254AE......
0012B724 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B734 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B744 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0012B754 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Figure 5: Controller Information in Memory

The commands used between host and USB flash drive controller are generated and transmitted by SCSICommandProcess, SCSICommandRead, and SCSICommandWrite functions inside uDiskDll, and the commands are finally sent by the DeviceIoControl function and transferred to the USB protocol. Namely, command itself comprises SCSI command format but it translates USB protocol and sends USB flash drive controller because USB flash drive controller connects to the USB interface at the host. Therefore, in order to support this communication method, the host has to insert a specific driver that it is usually called the filter driver. We check device driver structure using the DeviceTree tool and its result is shown in Figure 6.



```
DRV  ₩Driver₩usbstor
  DEV  ₩Device₩000000be
    ATT  ₩Device₩Harddisk2₩DR7
      ATT  Attached: (unnamed) - ₩Driver₩PartMgr
  DEV  ₩Device₩000000bd
    ATT  ₩Device₩CdRom4
      ATT  Attached: (unnamed) - ₩Driver₩GEARAspiWDM
      FS  File System: (unnamed) - ₩FileSystem₩Cdfs
        ATT  Attached: (unnamed) - ₩FileSystem₩FltMgr
          ATT  Attached: (unnamed) - ₩Driver₩AhnFlt2k
            ATT  Attached: (unnamed) - ₩FileSystem₩FltMgr
  DEV  ₩Device₩000000bc
```

Figure 6: Device Tree for the Storage Driver

The result shows that, in order to support CDFS (CD File System), a CDFS driver is attached and we consider that GEARAspiWDM driver processes SCSI commands, and other filter drivers transform from SCSI commands to USB commands and then send those USB commands using the USB interface. Figure 7 shows the procedure for processing commands.

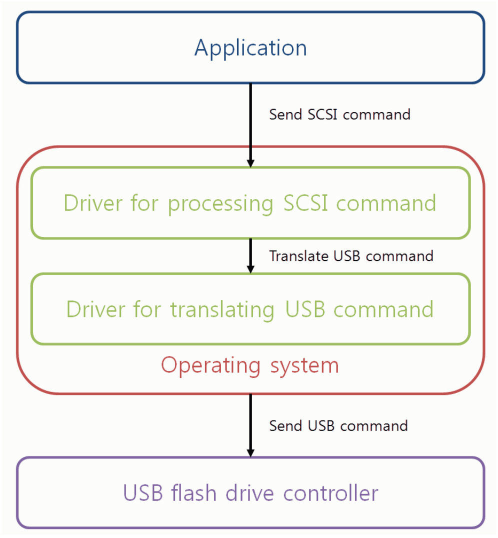
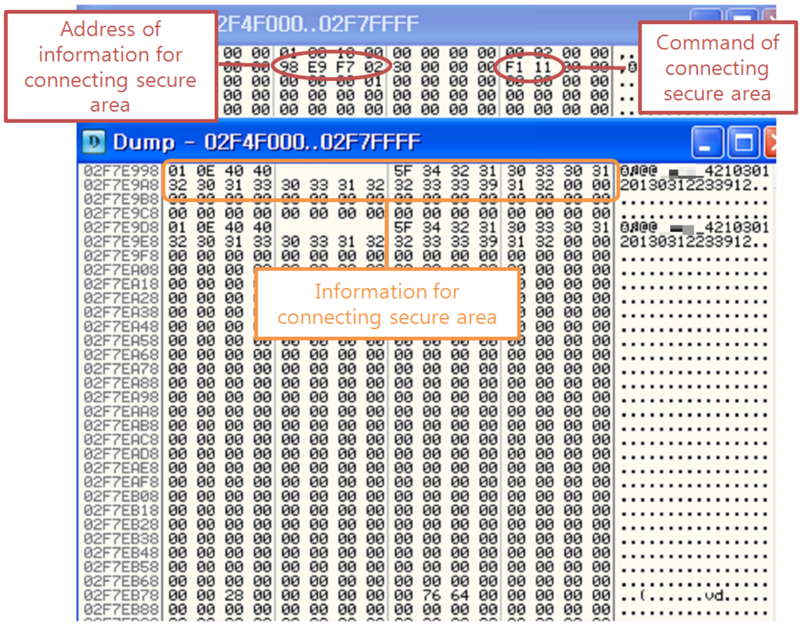

Application
Send SCSI command
Driver for processing SCSI command
Translate USB command
Driver for translating USB command
Operating system
Send USB command
USB flash drive controller

Figure 7: Command Flow for a Storage Device

4.2 Secure Area of the USB Flash Drive

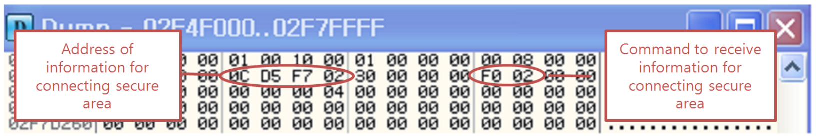
Figure 7 shows the command for connecting to the secure area. The command for connecting to the secure area is 0xF111 and the address of stored information for connecting to the secure area is 0x02F7E998. If an attacker does not have information for connecting to the secure area, he or she cannot connect to it. However the information can be obtained by sending the received information for connecting to the secure area. The vulnerability in the USB flash drive controller is caused by this command. This information is necessary for user authentication and only correct users can generate this information. Due to this command, incorrect users can get this information and bypass user authentication and access stored data inside the secure area. This command is shown in Figure 8 and Figure 9 shows information received after the Send command.


Address of information for connecting secure area
Command of connecting secure area
Information for connecting secure area

```
2F4F000..02F7FFFF
00 00 01 00 10 00 00 00 00 00 00 02 00 00 ..
00 00 98 E9 F7 02 30 00 00 00 F1 11 00 00 .0
00 00 00 00 00 01 00 00 00 00 00 00 00 00 ..
00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..
00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..

Dump - 02F4F000..02F7FFFF
02F7E998 01 0E 40 40             5F 34 32 31 30 33 30 31 0ᴀ@@ [redacted] 4210301
02F7E9A8 32 30 31 33 30 33 31 32 32 33 33 39 31 32 00 00 2013031223391 2..
02F7E9B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7E9C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7E9D8 01 0E 40 40             5F 34 32 31 30 33 30 31 0ᴀ@@ [redacted]_4210301
02F7E9E8 32 30 31 33 30 33 31 32 32 33 33 39 31 32 00 00 2013031223391 2..
02F7E9F8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EA08 00 00 00                            00 00 00 00 ................
02F7EA18 00 00 00                            00 00 00 00 ................
02F7EA28 00 00 00                            00 00 00 00 ................
02F7EA38 00 00 00                            00 00 00 00 ................
02F7EA48 00 00 00                            00 00 00 00 ................
02F7EA58 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EA68 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EA78 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EA88 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EA98 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAA8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAB8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAC8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAD8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAE8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EAF8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB28 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB38 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB48 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB58 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB68 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7EB78 00 00 28 00 00 00 00 00 00 76 64 00 00 00 00 00 ..(......vd.....
02F7EB88 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Figure 8: Command to connect the secure area


Address of information for connecting secure area
Command to receive information for connecting secure area

```
Dump - 02F4F000..02F7FFFF
         00 01 00 10 00 01 00 00 00 00 08 00 00
         00 0C D5 F7 02 30 00 00 00 F0 02 00 00
         00 00 00 00 04 00 00 00 00 00 00 00 00
         00 00 00 00 00 00 00 00 00 00 00 00 00
02F7D260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

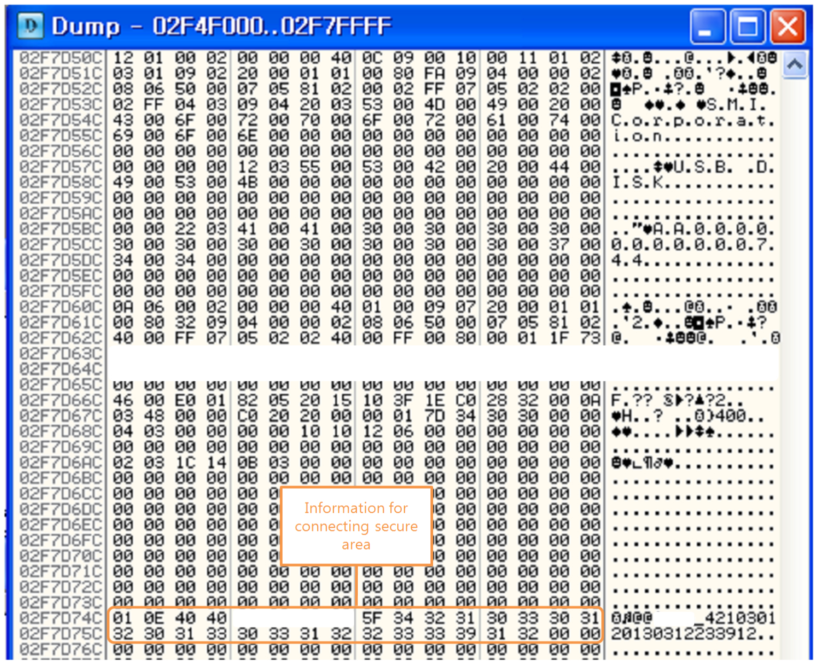
Figure 9: Subsequent command to connect the secure area

The command to receive information for connecting to the secure area is 0xF002 and information for connecting to the secure area is stored at offset 0x240 and is 32 bytes in size. If an attacker wants to connect to the secure area, the attacker sends command 0xF111 with this information. Even though this information can be different depending on the configuration of the secure USB, a third party can obtain this information by sending this command. As a result, a third party can access stored data inside the secure area without user authentication. To prove this vulnerability, we implement attack software using these commands and Figure 10 shows the result of the attack.

As shown in Figure 10, when we select the drive letter that communicates with the USB flash drive controller and then click the “Send Command“ button, we can access stored data in the “I“ drive as a secure area.

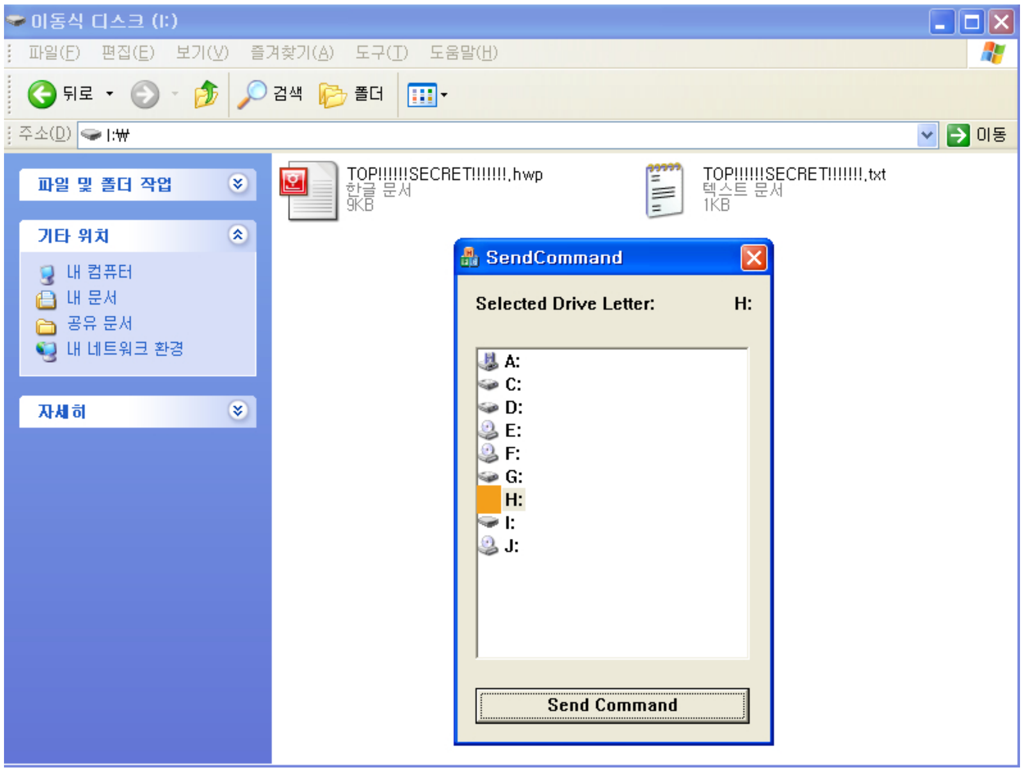
5 Conclusions

This paper analyzes a vulnerability in the SM3254AE chipset that is used to divide partitions for hardware-based partitioning method, which is one of the methods to protect data inside USB flash drive. As a result of analyzing this vulnerability, we verify that an unauthorized user can access data stored inside the secure area by analyzing for vulnerabilities. This vulnerability is caused by a specific command that responses with critical information for connecting secure area. Therefore, if manufacturers want to make stronger secure USB devices, this command must be forbidden.


Dump - 02F4F000..02F7FFFF
Information for connecting secure area

```
02F7D50C 12 01 00 02 00 00 00 40 0C 09 00 10 00 11 01 02 ‡0.0...@...▸.◂00
02F7D51C 03 01 09 02 20 00 01 01 00 80 FA 09 04 00 00 02 ♥0.0 .00.'?♦...0
02F7D52C 08 06 50 00 07 05 81 02 00 02 FF 07 05 02 02 00 ◘♠P.·‡?.0  ·‡00.
02F7D53C 02 FF 04 03 09 04 20 03 53 00 4D 00 49 00 20 00 0  ♦♥.♦ ♥S.M.I.
02F7D54C 43 00 6F 00 72 00 70 00 6F 00 72 00 61 00 74 00 C.o.r.p.o.r.a.t.
02F7D55C 69 00 6F 00 6E 00 00 00 00 00 00 00 00 00 00 00 i.o.n...........
02F7D56C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D57C 00 00 00 00 12 03 55 00 53 00 42 00 20 00 44 00 ....‡♥U.S.B. .D.
02F7D58C 49 00 53 00 4B 00 00 00 00 00 00 00 00 00 00 00 I.S.K...........
02F7D59C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D5AC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D5BC 00 00 22 03 41 00 41 00 30 00 30 00 30 00 30 00 .."♥A.A.0.0.0.0.
02F7D5CC 30 00 30 00 30 00 30 00 30 00 30 00 30 00 37 00 0.0.0.0.0.0.0.7.
02F7D5DC 34 00 34 00 00 00 00 00 00 00 00 00 00 00 00 00 4.4.............
02F7D5EC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D5FC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D60C 0A 06 00 02 00 00 00 40 01 00 09 07 20 00 01 01 .♠.0...@0..· .00
02F7D61C 00 80 32 09 04 00 00 02 08 06 50 00 07 05 81 02 .'2.♦..0◘♠P.·‡?
02F7D62C 40 00 FF 07 05 02 02 40 00 FF 00 80 00 01 1F 73 @.  ·‡00@.  .'.0
02F7D63C [redacted]
02F7D64C
02F7D65C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D66C 46 00 E0 01 82 05 20 15 10 3F 1E C0 28 32 00 0A F.?? §▸?▲?2..
02F7D67C 03 48 00 00 C0 20 20 00 00 01 7D 34 30 30 00 00 ♥H..?  ..0}400..
02F7D68C 04 03 00 00 00 00 10 10 12 06 00 00 00 00 00 00 ♦♥....▸▸‡♠......
02F7D69C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D6AC 02 03 1C 14 0B 03 00 00 00 00 00 00 00 00 00 00 0♥∟¶♂♥..........
02F7D6BC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D6CC 00 00 00 00 00 0               0 00 00 00 00 00 ................
02F7D6DC 00 00 00 00 00 0               0 00 00 00 00 00 ................
02F7D6EC 00 00 00 00 00 0               0 00 00 00 00 00 ................
02F7D6FC 00 00 00 00 00 0               0 00 00 00 00 00 ................
02F7D70C 00 00 00 00 00 0               0 00 00 00 00 00 ................
02F7D71C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D72C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D73C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02F7D74C 01 0E 40 40 [redacted] 5F 34 32 31 30 33 30 31 0♫@@ [redacted] _4210301
02F7D75C 32 30 31 33 30 33 31 32 32 33 33 39 31 32 00 00 20130312233912..
02F7D76C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Figure 10: Parameter Block for connecting the Secure Area



TOP!!!!!!SECRET!!!!!!!.hwp
한글 문서
9KB

TOP!!!!!!SECRET!!!!!!!.txt
텍스트 문서
1KB

Selected Drive Letter: H:

A:
C:
D:
E:
F:
G:
H:
I:
J:

Figure 11: Implemented Software

References

[1] H. Jeong, Y. Choi, W. Jeon, F. Yang, Y. Lee, S. Kim, and D. Won. Vulnerability analysis of secure usb flash drives. In *Proc. of the 2007 IEEE International Workshop on Memory Technology, Design and Testing*

(MTDT'07), Taipei, Taiwan, pages 61–64. IEEE, December 2007.

[2] S. L. Jewan Bang, ByeongYeong Yoo. Secure usb bypassing tool. *Journal of Digital Investigation*, 7(Supplement):S114–S120, August 2010.

[3] T. Jung and K. Yim. Countermeasures to the vulnerability of the keyboard hardware. *Journal of the Korea Information Security and Cryptology*, 18(4):187–194, August 2008.

[4] Kingpin. Attacks on and countermeasures for usb hardware token devices. In *Proc. of the 5th Nordic Workshop on Secure IT Systems Encouraging Co-operation (NORDSEC'00), Reykjavik, Iceland*, pages 35–57. Reykjavik University, October 2000.

[5] K. Lee, Hyeungjun, Y. Choi, S. Pho, I. You, and K. Yim. Safe authentication protocol for secure usb memories. *Journal of Wireless Mobile Networks, Ubiquitous Computing and De-pendable Application (JoWUA)*, 1(1):46–55, June 2010.

[6] K. Lee, W. Kom, K. Bae, and K. Yim. A solution to protecting usb keyboard data. In *Proc. of the 5th International Conference on Broadband, Wireless Computing, Communication and Applications(BWCCA '10), Fukuoka, JAPAN*, pages 108–111. IEEE, November 2010.

[7] K. Lee, S. Pho, and K. Yim. Reversability assessment on secure usb memories. In *Proc. of the 5th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing(IMIS'11), Seoul, Korea*, pages 6–8. IEEE, 2011.

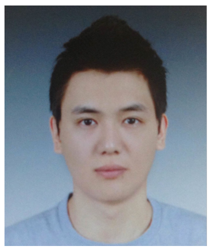
[8] K. Lee, K. Yim, and E. H. Spafford. Reverse-safe authentication protocol for secure usb memories. *Journal of Security and Communication Networks (SCN)*, 5(8):834–845, August 2012.

[9] Silicon Motion Technology Corp. Silicon for mobile information. `http://www.siliconmotion.com`.

[10] Silicon Motion Technology Corp. SM3254 Product Brief, 2009. `http://www.sabreadv.com/wp-content/uploads/SM3254-Product-Brief.pdf`.

[11] A. Vasudevan, B. Parno, N. Qu, and A. Perrig. Lockdown: A Safe and Practical Environment for Security Applications. Technical Report 11, CyLab, Carnegie Mellon University, 2009.

Author Biography



Jaein Kim is currently a senior student in the Department of Information Security Engineering, Soonchunhyang University, Asan, Republic of Korea and he is working as a researcher in the Lab. of Information Systems Security Assurance towards the Graduate School of Soonchunhyang University. His research interests include hardware security, kernel security and CPS security.

Youngjun Lee received his B.S. in 2013 from the Dept. of Information Security Engineering, Soonchunhyang University, Asan, Republic of Korea and he is currently a master student in the Lab. of Information Systems Security Assurance, Graduate School of Soonchunhyang University. His research interests include CPS security, malware analysis and vulnerability assessment.

Kyungroul Lee received his B.S. and M.S. from the Department of Information Security Engineering, Soonchunhyang University, Asan, Korea in 2008 and 2010 respectively and he is currently towards his Ph.D. degree. His research interests include vulnerability analysis, kernel mode root kit, binary obfuscation, hardware security, reverse engineering and insider threats.

Taeyoung Jung received his B.S. and M.S. from the Department of Information Security Engineering, Soonchunhyang University, Asan, Korea in 2007 and 2009 respectively. He worked on developing video surveillance servers in XRPLUS INC. and crypto libraries in INITECH co., ltd. and he has been towards his Ph.D. degree since 2013. His research interests include vulnerability analysis, secure coding, reverse engineering, CPS security and security hardware design.

Dmitry Volokhov received his B.S. from University of Ottawa, Canada in 2005. He is currently an assistant professor in the Department of Information Security Engineering, Soonchunhyang University, Asan, Republic of Korea and he is working as a researcher in the Lab. of Information Systems Security Assurance. His research interests include security issues on the cyber physical systems and the cognitive sensor network.

Kangbin Yim received his B.S. M.S. and Ph.D. from Ajou University, Suwon, Republic of Korea in 1992, 1994 and 2001, respectively. He has joined the Department of Information Security Engineering, Soonchunhyang University as a professor since 2003 and he is currently running the Lab. of Information Systems Security Assurance. His research interests include security protocols and access control mechanism, secure hardware design, vulnerability assessment, offensive security analysis and security issues on cyber physical systems and embedded realtime systems.