USB Hacking

mongii@grayhash

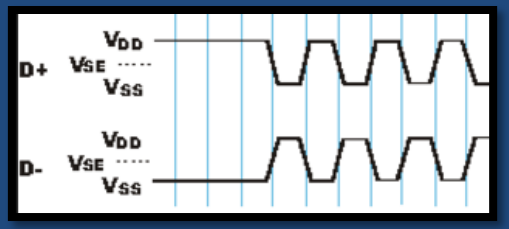
Summary

- About USB protocol
- USB Packet Analysing
- USB Stack Fuzzing
- File System Fuzzing
- Multi-media File Fuzzing

USB(Universal Serial Bus) 기초

USB(Universal Serial Bus) 소개

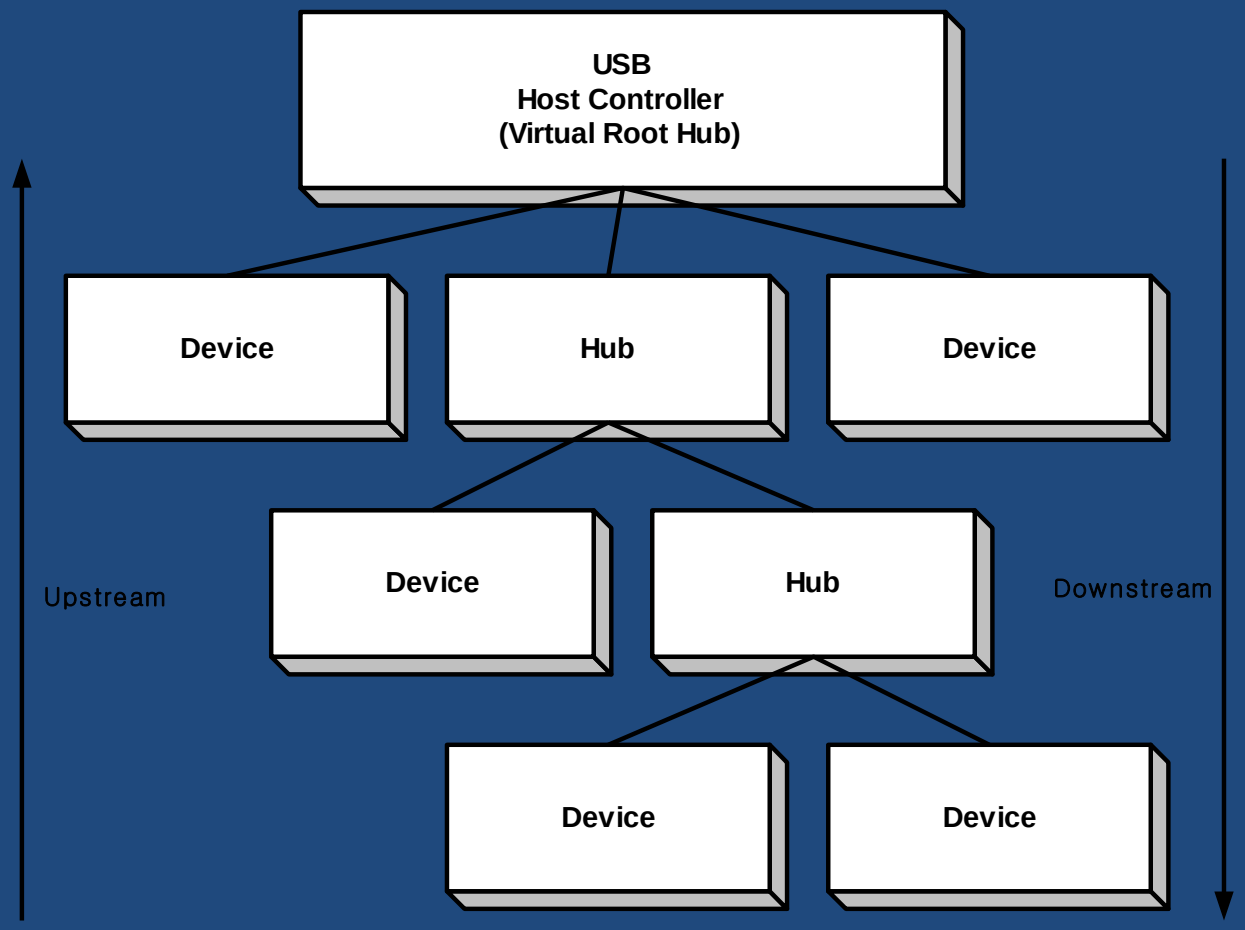
- Host-Device architecture
- Host driven communication
- Half-Duplex (1.1, 2.0 기준)
- Speeds
  - Low Speed: 1.5Mbits/s
  - Full Speed: 12Mbits/s
  - High Speed: 480Mbits/s
- Differential signaling (D+, D-)
- Up to 127 devices can be connected
- Power: 5V



USB 시스템의 구조

- Host
  - USB 통신의 중심
  - USB 네트워크에 오직 한 개의 호스트만 가짐
  - 루트 허브를 포함
- Device
  - USB Host에 연결되는 장치
  - USB Hub 혹은 Function

USB bus topology



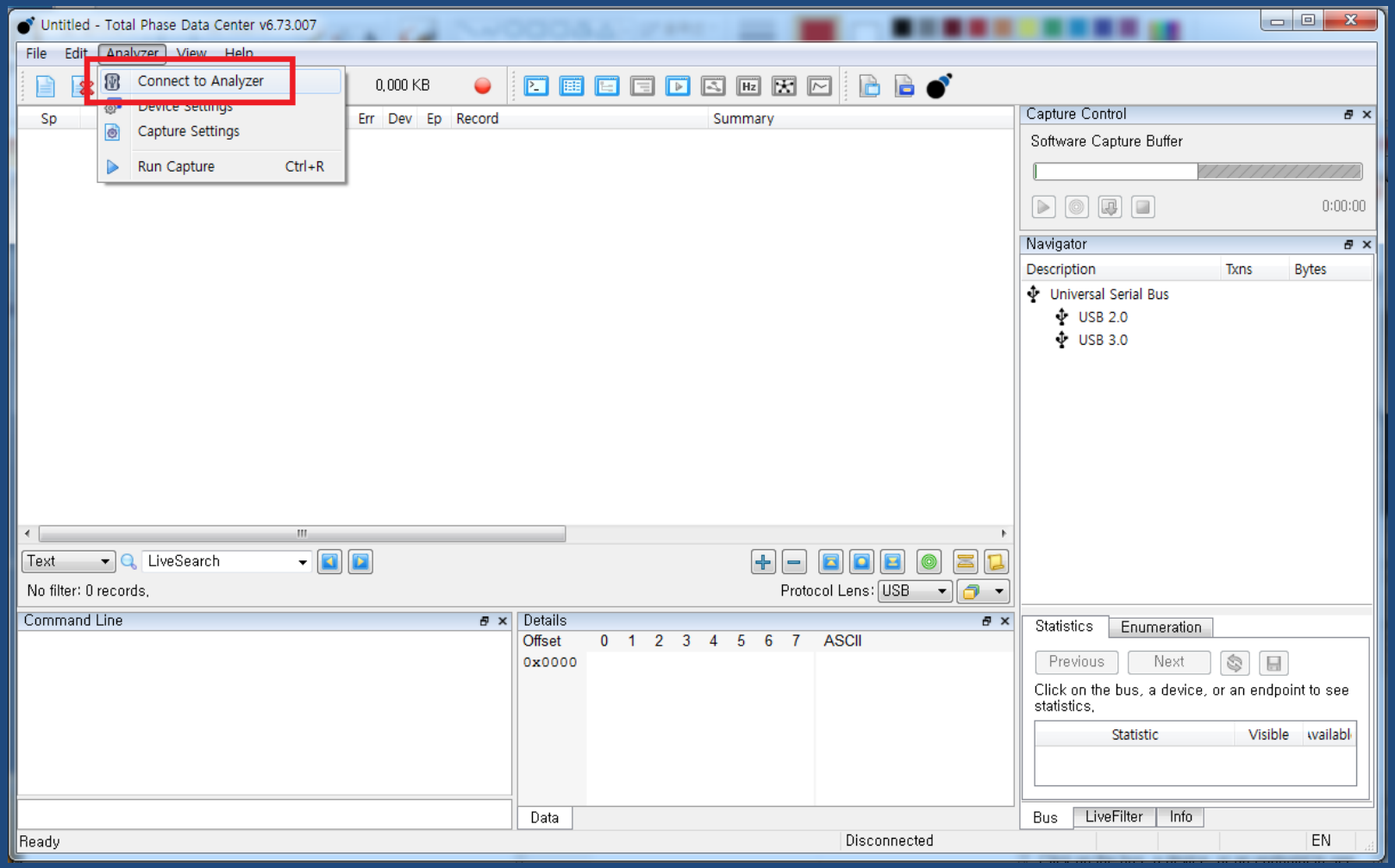
USB Packet 분석

USB Packet 분석

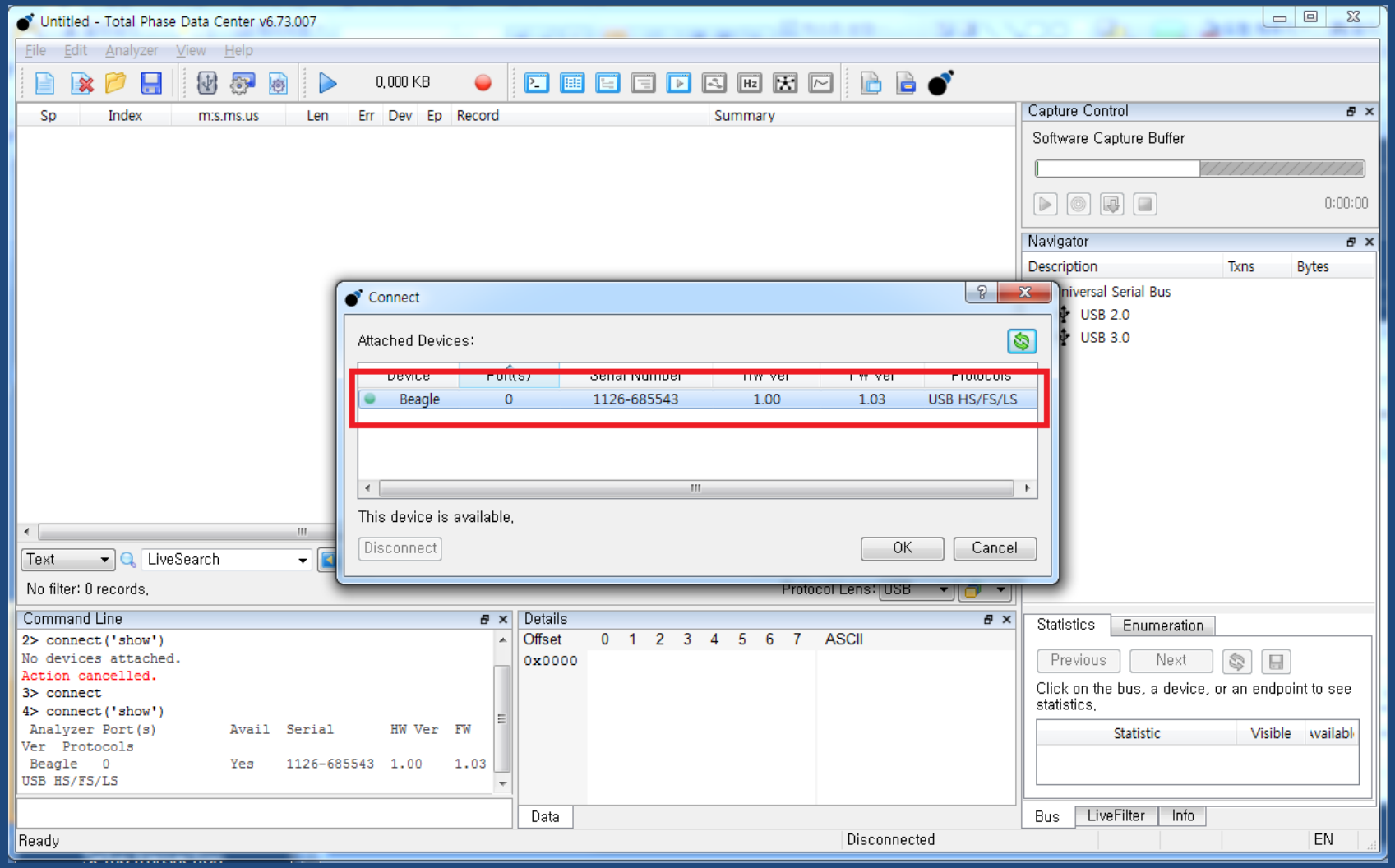
Beagle USB 480

- USB Packet Analyzer
- 가격 : 약 $1,400
- Software : Total Phase Data Center
  USBpcap과의 차이점
  - Low Level의 USB 패킷들을 볼 수 있음

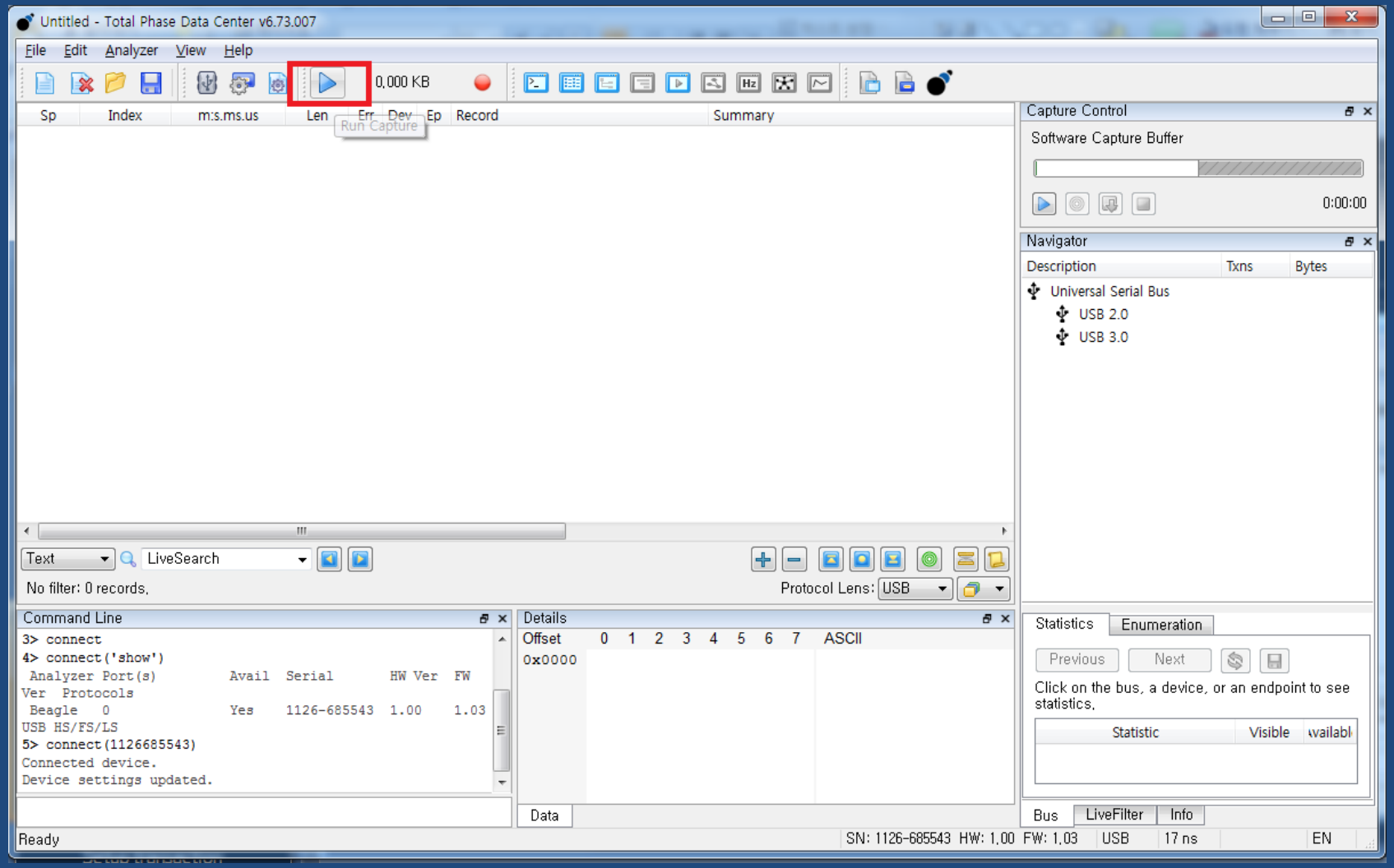
USB Packet Capture



USB Packet Capture



USB Packet Capture



USB Packet Capture

- 대상 USB 장치 연결 (ex> USB 키보드)

USB Packet Capture

USB_키보드_연결_패킷 - Total Phase Data Center v6.73.007

File Edit Analyzer View Help

127.1 KB

Sp	Index	m:s.ms.us	Len	Err	Dev	Ep	Record	Summary
	0	0:00.000.000					Capture started (Aggregate)	[07/27/17 12:16:36]
	1	0:00.000.000					<Host connected>	
LS	2	0:25.314.354					<Low-speed>	
LS	3	0:25.317.354	114 ms				<Suspend>	
LS	4	0:25.431.996	54.8 ms				<Reset> / <Chirp K> / <Tiny K>	
LS	5	0:25.486.878					<Low-speed>	
LS	6	0:25.487.766	37.0 ms				[38 KEEP-ALIVE]	
LS	7	0:25.524.781	18 B		00	00	Get Device Descriptor	Index=0 Length=64
LS	31	0:25.525.771	1.30 us				[1 KEEP-ALIVE]	
LS	32	0:25.526.672	54.8 ms				<Reset> / <Chirp K> / <Tiny K>	
LS	33	0:25.581.554					<Low-speed>	
LS	34	0:25.581.777	36.0 ms				[37 KEEP-ALIVE]	
LS	35	0:25.618.266	0 B		00	00	Set Address	Address=17
LS	44	0:25.618.781	18.0 ms				[19 KEEP-ALIVE]	
LS	45	0:25.636.357	18 B		17	00	Get Device Descriptor	Index=0 Length=18
LS	68	0:25.637.783	2.00 ms				[3 KEEP-ALIVE]	
LS	69	0:25.638.405	59 B		17	00	Get Configuration Descriptor	Index=0 Length=255
LS	118	0:25.640.783	1.00 ms				[2 KEEP-ALIVE]	
LS	119	0:25.641.793	4 B		17	00	Get String Descriptor	Index=0 Length=255

Text LiveSearch

No filter: 871 records.

Protocol Lens: USB

Capture Control

Software Capture Buffer

0:00:00

Navigator

Description	Txns	Bytes
Universal Serial Bus		
USB 2.0	152	845
USB 3.0	0	0

Command Line

```
7> stop
Capture stopped.
8> save
9> save(u'F:/\u2605 GRAYHASH/\u2605
\ud2b8\ub808\uc774\ub2dd/\uce74\ud574\ud0b9
\ud2b8\ub808\uc774\ub2dd/USB Packet
Sniffer/USB_\ud0a4\ubcf4\ub4dc_\uc5f0\uacb0_\ud328\ud0b7.t
dc', {'no_timing': False, 'filtered_only': False}, True)
File saved.
```

Details

Offset	0	1	2	3	4	5	6	7	ASCII
0x0000									

Data

Statistics Enumeration

Previous Next

Click on the bus, a device, or an endpoint to see statistics.

Statistic	Visible	Availabl

Bus LiveFilter Info

Ready SN: 1126-685543 HW: 1.00 FW: 1.03 USB 17 ns EN

USB Packet 요약

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

주요 키워드

Device Descriptor
Configuration Descriptor
String Descriptor
Report Descriptor

Get String Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor

-총 여섯 번 반복

-필요할 때 마다 재요청하기 때문 (USB스택&드라이버의 구현마다 다름, 한 번 요청 후 정보를 저장해놓는 경우도 있음)

-헤더를 먼저 요청 후 길이, 인코딩 정보를 파악 => 전체 정보 요청

Get String Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor

Details

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
0x0000	1A	03	55	00	53	00	42	00	20	00	4B	00	65	00	79	00	··U·S·B· ·K·e·y·
0x0010	62	00	6F	00	61	00	72	00	64	00							b·o·a·r·d·

Data

Length : 0x1A(26)
Type : 0x03 (string descriptor)

Offset	Field	Size	Value	Description
0	*bLength*	1	N+2	Size of this descriptor in bytes
1	*bDescriptorType*	1	Constant	STRING Descriptor Type
2	*wLANGID[0]*	2	Number	LANGID code zero
...	...	...	...	...
N	*wLANGID[x]*	2	Number	LANGID code *x*

혹은 문자열 데이터
...
...
...

Descriptor Type

- 0x01 : DEVICE
- 0x02 : CONFIGURATION
- 0x03 : STRING
- 0x04 : INTERFACE
- 0x05 : ENDPONT

Device Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

-셋 모두 동일(string과 같은 이유)
-장치에 대한 기본적인 정보들 제공
-Vendor, Product ID 정보 제공
-Configuration의 개수 정보 제공

Device Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

Details

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
0x0000	12	01	10	01	00	00	00	08	2C	1A	2A	0B	10	01	01	02	·····,·*·····
0x0010	00	01															··

Data

옵셋	필드	크기	값	내용
0	bLength	1	숫자	Descriptor의 크기
1	bDescriptorType	1	상수	디바이스 Descriptor 형태
2	bcdUSB	2	BCD	BCD형태로 이루어진 발표 번호
4	bDeviceClass	1	클래스	클래스 코드
5	bDeviceSubClass	1	서브클래스	서브클래스 코드
6	bDeviceProtocol	1	프로토콜	프로토콜 코드
7	bMaxPacketSized()	1	숫자	Endpoint 0를 위한 최대 패킷의 크기
8	idVendor	2	ID	Vendor ID
10	idProduct	2	ID	Product ID
12	bcdDevice	2	BCD	BCD으로 나타낸 디바이스 릴리즈 번호
14	iManufacturer	1	인덱스	제조자를 나타내는 스트링 Descriptor의 인덱스
15	iProduct	1	인덱스	생산자를 나타내는 스트링 Descriptor의 인덱스
16	iSerialNumber	1	인덱스	디바이스의 시리얼 번호를 나타내는 스트링 Descriptor의 인덱스
17	bNumConfigurations	1	숫자	가능한 설정의 번호

Device Descriptor

옵셋	필드	크기	값	내용
0	bLength	1	숫자	Descriptor의 크기
1	bDescriptorType	1	상수	디바이스 Descriptor 형태
2	bcdUSB	2	BCD	BCD형태로 이루어진 발표 번호
4	bDeviceClass	1	클래스	클래스 코드
5	bDeviceSubClass	1	서브클래스	서브클래스 코드
6	bDeviceProtocol	1	프로토콜	프로토콜 코드
7	bMaxPacketSized()	1	숫자	Endpoint 0를 위한 최대 패킷의 크기
8	idVendor	2	ID	Vendor ID
10	idProduct	2	ID	Product ID
12	bcdDevice	2	BCD	BCD으로 나타낸 디바이스 릴리즈 번호
14	iManufacturer	1	인덱스	제조자를 나타내는 스트링 Descriptor의 인덱스
15	iProduct	1	인덱스	생산자를 나타내는 스트링 Descriptor의 인덱스
16	iSerialNumber	1	인덱스	디바이스의 시리얼 번호를 나타내는 스트링 Descriptor의 인덱스
17	bNumConfigurations	1	숫자	가능한 설정의 번호

Size	12
Type	01
Release num	10 01
Class Code	00
Sus-Class //	00
Protocol //	00
Packet Size	08
Vendor ID	2C 1A
Product ID	2A 0B
Release ver	10 01
String index	01
String index	02
String index	00
Config num	01

Vendor ID & Product ID

```
1a1d  Veho
        0407  Mimi WiFi speakers
1a25  Amphenol East Asia Ltd.
1a2a  Seagate Branded Solutions
1a2c  China Resource Semico Co., Ltd
        0021  Keyboard
        0024  Multimedia Keyboard
1a32  Quanta Microsystems, Inc.
        0304  802.11n Wireless LAN Card
1a34  ACRUX
        0802  Gamepad
```

- http://www.linux-usb.org/usb.ids

이 값이 무엇이냐에 따라 OS에 인식되는 장치명과 사용되는 Device Driver가 달라짐

buy a VID?
You can buy the right to use a single VID from the usb.org. They charge $2000 for this, but this is a one-time fee. This gives you 65536 PID numbers, more than enough for the rest of your life. Check Getting a Vendor ID on the usb.org website for this option.

Vendor ID & Product ID

USB ID Database

Search for USB devices with Vendor ID, Product ID and/or Name:

Vendor ID: 0x1A2C

Product ID: Product

Name: Name

Search

Search Results:

Vendor ID	Product ID	Name		Comment
0x1A2C		China Resource Semico Co., Ltd		?
0x1A2C	0x0021	China Resource Semico Co., Ltd	Keyboard	
0x1A2C	0x0024	China Resource Semico Co., Ltd	Multimedia Keyboard	
0x1A2C	0x0C21	China Resource Semico Co., Ltd	USB Keyboard	

- http://www.the-sz.com/products/usbid/index.php?v=0x1A2C

Device Descriptor

Navigator

Description	Txns	Bytes
Unconfigured Device (0) (BusIdx:0)	7	34
USB Keyboard (17) (BusIdx:1)	145	811
Default Endpoint (EP 0)	143	795
Cfg 1, Bus Powered, 98mA	2	16
IF 0 (alt 0), HID, Boot Interface, Keyboard	2	16
IF 1 (alt 0), HID, None, None	0	0

Statistics | Enumeration

Device Descriptor — Radix: auto

bLength	18 (0x12)
bDescriptorType	DEVICE (0x01)
bcdUSB	1.1 (0x0110)
bDeviceClass	Defined in Interface (0x00)
bDeviceSubClass	0x00
bDeviceProtocol	0x00
bMaxPacketSize0	8 (0x08)
idVendor	6700 (0x1a2c)
idProduct	2858 (0x0b2a)
bcdDevice	1.1 (0x0110)
iManufacturer	1 (0x01)
iProduct	USB Keyboard (0x02)
iSerialNumber	(0x00)
bNumConfigurations	1 (0x01)

Configuration Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

-다섯 모두 동일(string과 같은 이유)
-인터페이스의 개수 정보 제공
-파워 공급 방법을 기술함

Configuration Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

Details

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
0x0000	09	02	3B	00	02	01	00	A0	31	09	04	00	00	01	03	01	··;·····1·····
0x0010	01	00	09	21	10	01	00	01	22	36	00	07	05	81	03	08	···!····"6·····
0x0020	00	0A	09	04	01	00	01	03	00	00	00	09	21	10	01	00	·····!···
0x0030	01	22	32	00	07	05	82	03	08	00	0A						·"2·····

옵셋	필드	크기	값	내용
0	bLength	1	숫자	Descriptor의 크기
1	bDescriptorType	1	상수	설정 Descriptor 형태
2	wTotalLength	2	숫자	이 설정에서의 반환되는 데이터의 전체크기
4	bNumberInterfaces	1	숫자	이 설정에서 지원되는 인터페이스의 수
5	bConfigurationValue	1	숫자	SetConfiguration() 요청에 대해 인자로서 사용되는 값
6	iConfiguration	1	인덱스	이 설정을 나타내는 스트링 Descriptor의 인덱스
7	bmAttribute	1	비트맵	설정 특성
8	MaxPower	1	mA	이 설정에서의 버스로부터의 USB 디바이스의 최대 파워 소비량

Configuration Descriptor

Navigator

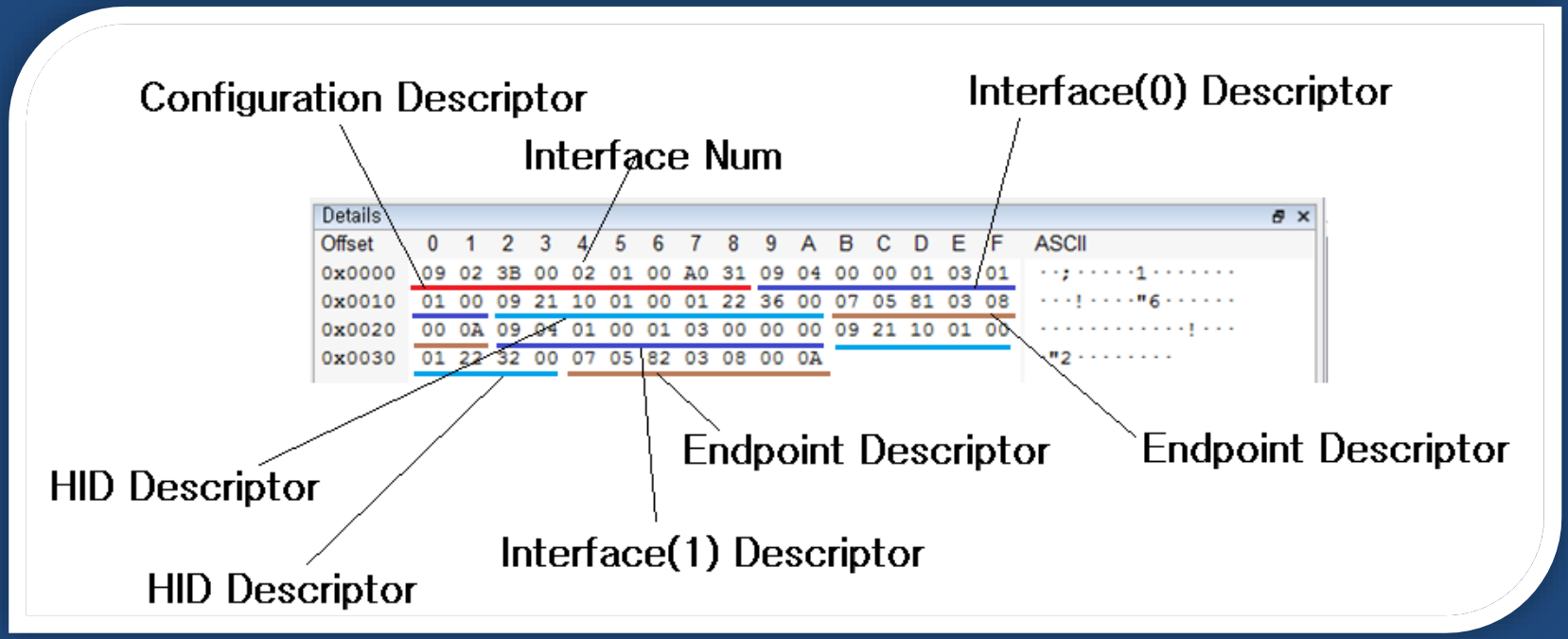
Description	Txns	Bytes
Unconfigured Device (0) (BusIdx:0)	7	34
USB Keyboard (17) (BusIdx:1)	145	811
Default Endpoint (EP 0)	143	795
Cfg 1, Bus Powered, 98mA	2	16
IF 0 (alt 0), HID, Boot Interface, Keyboard	2	16
IF 1 (alt 0), HID, None, None	0	0

Statistics | Enumeration

Configuration Descriptor — Radix: auto

bLength	9 (0x09)
bDescriptorType	CONFIGURATION (0x02)
wTotalLength	59 (0x3b)
bNumInterfaces	2 (0x02)
bConfigurationValue	1 (0x01)
iConfiguration	(0x00)
bmAttributes.Reserved1	0x1
bmAttributes.SelfPowered	Bus Powered (0x0)
bmAttributes.RemoteWakeup	Supported (0x1)
bmAttributes.Reserved0	0x00
bMaxPower	98mA (0x31)

그렇다면 뒤 쪽의 데이터 들은?



Interface Descriptor

- Class & SubClass 정보 제공
  - 해당 장치가 어떤 역할을 하는지 나타냄
- 한 장치 안에 여러 개의 Interface 존재 가능
  - 스마트폰 : 이동식 저장장치, adb 디버깅, 설치 CD 등

옵셋	필드	크기	값	내용
0	bLength	1	숫자	바이트로 나타낸 Descriptor의 크기
1	bDescriptorType	1	상수	인터페이스 Descriptor 형태
2	bInterfaceNumber	1	숫자	인터페이스의 수
3	bAlternateSetting	1	숫자	이전 필드에서 인터페이스 확인을 위해서 다른 셋팅을 선택하기 위해서 사용한 값
4	bNumEndpoints	1	숫자	이 인터페이스에 의해서 사용되는 Endpoint 수
5	bInterfaceClass	1	클래스	클래스 코드
6	bInterfaceSubClass	1	서브클래스	서브클래스 코드
7	bInterfaceProtocol	1	프로토콜	프로토콜 코드
8	iInterface	1	인덱스	이 인터페이스를 나타내는 스트링 Descriptor의 인덱스

Class & SubClass

Interface(0) Descriptor : 09 04 00 00 01 03 01 01 00

Base Class	Descriptor Usage	Description
00h	Device	Use class information in the Interface Descriptors
01h	Interface	Audio
02h	Both	Communications and CDC Control
03h	Interface	HID (Human Interface Device)
05h	Interface	Physical
06h	Interface	Image
07h	Interface	Printer
08h	Interface	Mass Storage
09h	Device	Hub
0Ah	Interface	CDC-Data
0Bh	Interface	Smart Card
0Dh	Interface	Content Security
0Eh	Interface	Video
0Fh	Interface	Personal Healthcare
10h	Interface	Audio/Video Devices
11h	Device	Billboard Device Class
12h	Interface	USB Type-C Bridge Class
DCh	Both	Diagnostic Device
E0h	Interface	Wireless Controller
EFh	Both	Miscellaneous
FEh	Interface	Application Specific
FFh	Both	Vendor Specific

http://www.usb.org/developers/defined_class/

Subclass Codes

Subclass Code	Description
0	No Subclass
1	Boot Interface Subclass
2 - 255	Reserved

http://www.rennes.supelec.fr/ren/fi/elec/docs/usb/hid1_11.pdf

0x03 : HID
0x01 : Boot Interface

Interface Descriptor

Interface Descriptor	Radix: auto
bLength	9 (0x09)
bDescriptorType	INTERFACE (0x04)
bInterfaceNumber	0x00
bAlternateSetting	0x00
bNumEndpoints	1 (0x01)
bInterfaceClass	Human Interface Device (0x03)
bInterfaceSubClass	Boot Interface (0x01)
bInterfaceProtocol	Keyboard (0x01)
iInterface	(0x00)

Interface(0) Descriptor

Interface Descriptor	Radix: auto
bLength	9 (0x09)
bDescriptorType	INTERFACE (0x04)
bInterfaceNumber	1 (0x01)
bAlternateSetting	0x00
bNumEndpoints	1 (0x01)
bInterfaceClass	Human Interface Device (0x03)
bInterfaceSubClass	None (0x00)
bInterfaceProtocol	None (0x00)
iInterface	(0x00)

Interface(1) Descriptor

Endpoint Descriptor

- 실질적인 데이터가 오가는 통로
- Endpoint Descriptor 역시 여러 개가 될 수 있음
- Transfer Type 정보 제공
  - Control, Interrupt, Bulk, Isochronous
- 파이프라고 부르기도 함

옵셋	필드	크기	값	내용
0	bLength	1	숫자	Descriptor의 크기
1	bDescriptorType	1	상수	Endpoint Descriptor의 형태
2	bEndpointAddress	1	Endpoint	Endpoint의 주소
3	bmAttribute	1	비트맵	Endpoint의 특성
4	wMaxPacketSize	2	숫자	설정이 선택될 때 송수신 최대 패킷 크기
6	bInterval	1	숫자	데이터 전송을 위한 폴링 Endpoint를 위한 간격

Transfer Type

- Interrupt
  - Function에서 Host에 주기적으로 소량의 데이터를 입력하는 경우에 적합
  - 키보드/마우스 등
- Contol
  - 디바이스가 설정 정보 등을 호스트에 전송할 때 사용
  - 혹은 호스트가 디바이스로 새로운 설정 정보 전송
- Bulk
  - 대량의 데이터 고속 전송
  - 신뢰성이 요구되는 경우에 적합
  - EX> USB 이동식 저장장치

Transfer Type

- Isochronous
  - 등시성 전송
  - 일정 주기에 일정량의 데이터를 전송하고자 할 때 적합
  - 다른 전송모드에 비해 높은 우선 순위
  - 데이터 전송 폭과 전송 시간을 보장
  - 실시간 어플리케이션에 적합
    - 예> CCTV 영상, 오디오 스트리밍
  - 데이터 오류 보장은 X → 오류 시 재전송 요청 불가

Interface & Endpoint Descriptor

Endpoint Descriptor	Radix: auto
bLength	7 (0x07)
bDescriptorType	ENDPOINT (0x05)
bEndpointAddress	1 IN (0x81)
bmAttributes.TransferType	Interrupt (0x3)
bmAttributes.Reserved0	0x00
wMaxPacketSize	8 bytes (1 transaction per microframe if HS) (0x0008)
bInterval	LS/FS:10ms HS:64ms (0x0a)

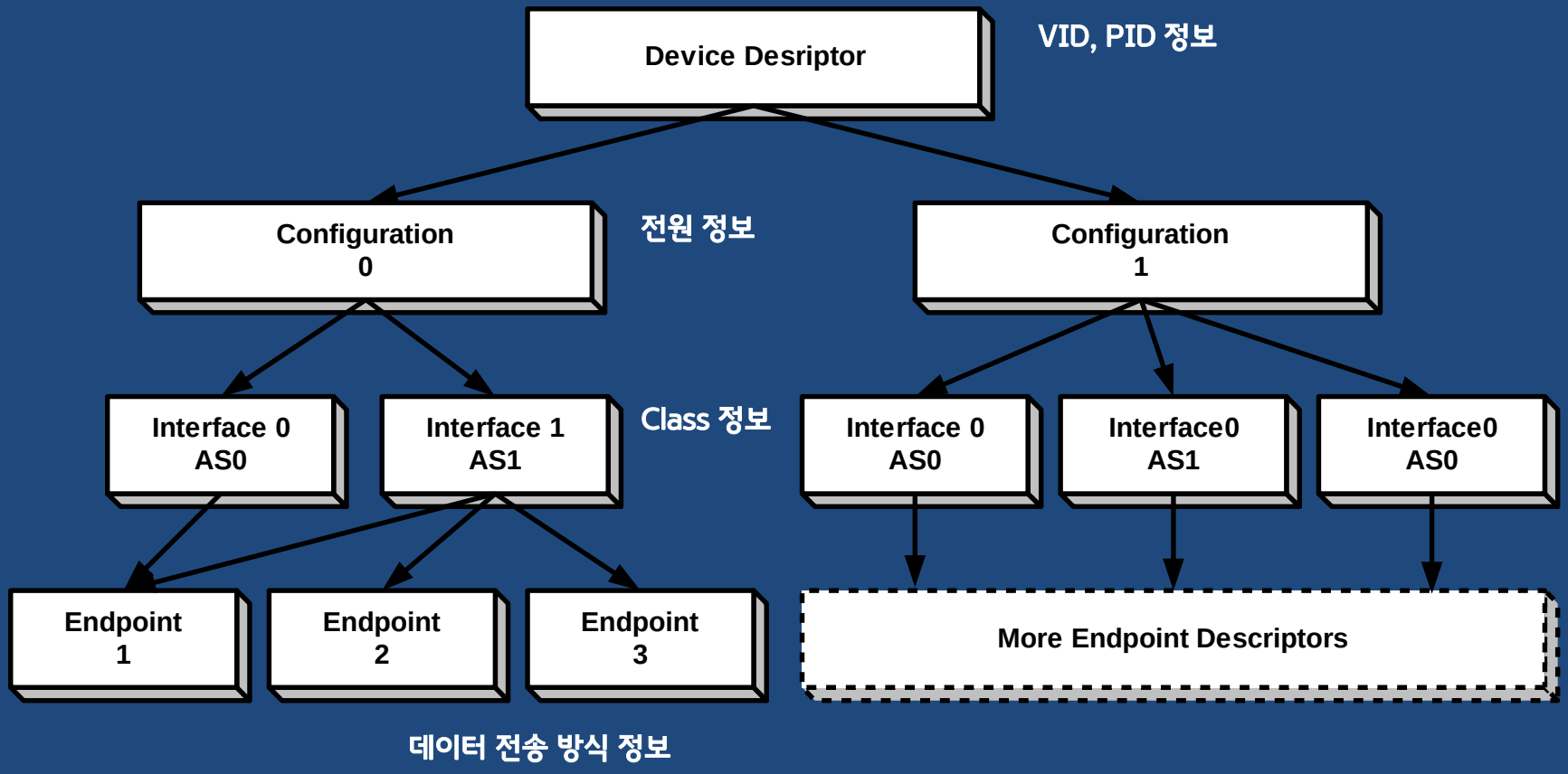
Endpoint(1) Descriptor

Endpoint Descriptor	Radix: auto
bLength	7 (0x07)
bDescriptorType	ENDPOINT (0x05)
bEndpointAddress	2 IN (0x82)
bmAttributes.TransferType	Interrupt (0x3)
bmAttributes.Reserved0	0x00
wMaxPacketSize	8 bytes (1 transaction per microframe if HS) (0x0008)
bInterval	LS/FS:10ms HS:64ms (0x0a)

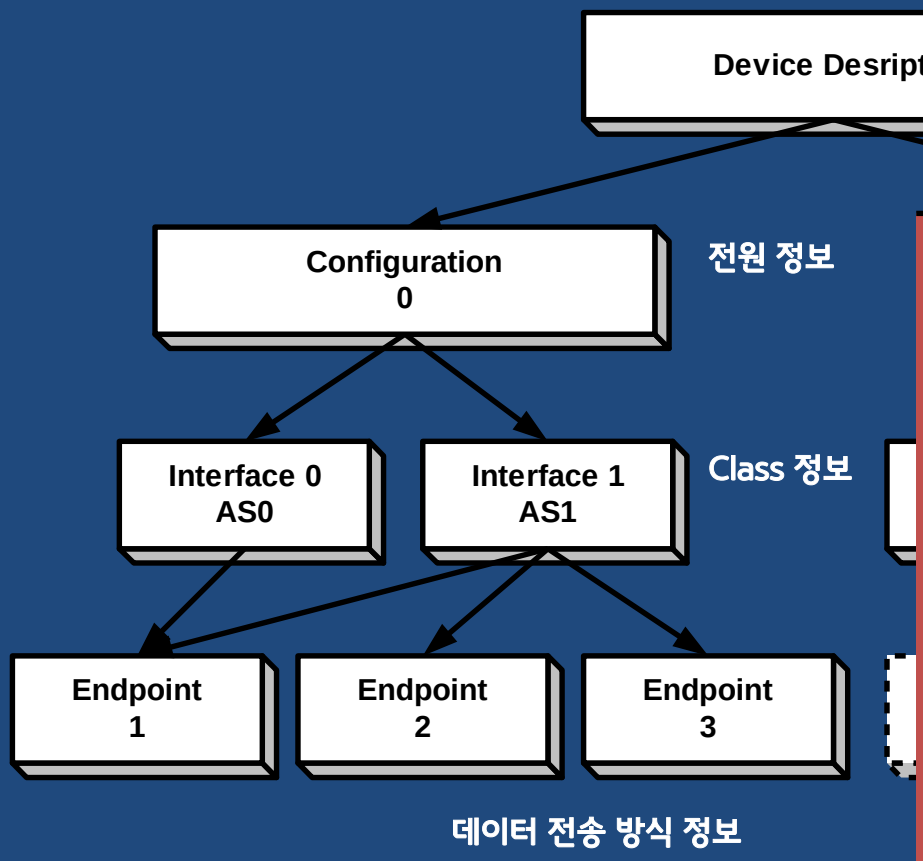
Endpoint(2) Descriptor

* Endpoint(0)은 기본으로 존재하는 Endpoint로서, Control packet들을 처리하는 역할을 함

USB Descriptor Hierarchy



USB Descriptor Hierarchy



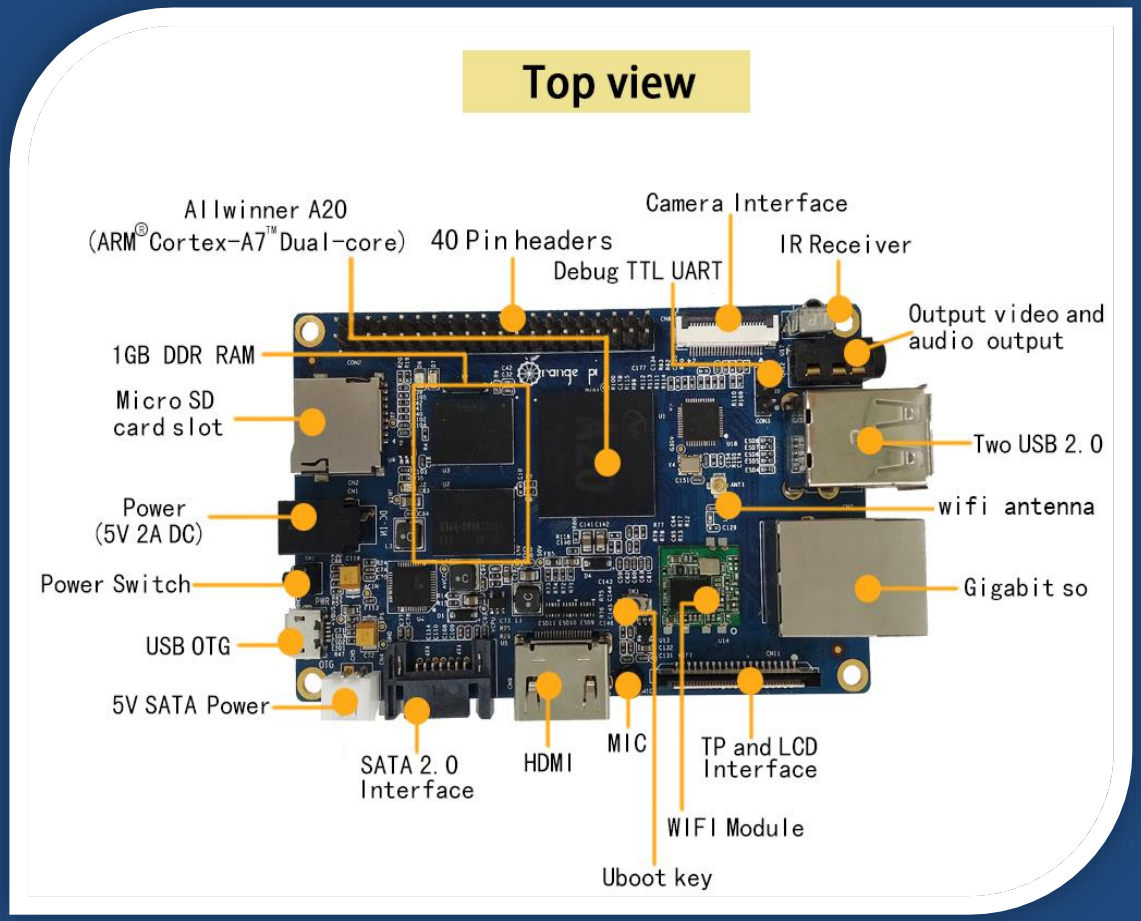
-VID : 0x1a2c (China Resource..)
-PID : 0x0b2a (USB Keyboard)
-전원 : Bus Powered, 98mA
-CLASS : HID keyboard
-전송방식 : Interrupt

USB Stack Fuzzing

USB Fuzzing 위한 준비물

- Fuzzing 대상
  - USB Host stack (OS kernel)
  - File system parser (OS kernel)
- Hardware requirement : USB OTG port
- Software requirement : Customizable USB Device source code

Hardware : OrangePi series



Hardware : OrangePi series

- Support USB Host & Device port (OTG)
- Support UART port for debug console
- Support Linux and Android software
- Support WIFI
- ETC...

Software : Linux USB Gadget

- USB Device function들을 구현해 놓은 커널 모듈들
- Supports USB Serial, USB Ethernet, USB Printer, USB Mass-storage and etc...
- Kernel/drivers/usb/gadget/*

```
root@orangepi:~# ls -al /lib/modules/3.4.103/kernel/drivers/usb/gadget
total 5108
drwxr-xr-x 2 root root   4096 Mar 11  2016 .
drwxr-xr-x 8 root root   4096 Mar 11  2016 ..
-rw-r--r-- 1 root root 410739 Mar 11  2016 g_acm_ms.ko
-rw-r--r-- 1 root root 247163 Mar 11  2016 g_audio.ko
-rw-r--r-- 1 root root 382175 Mar 11  2016 g_cdc.ko
-rw-r--r-- 1 root root 183304 Mar 11  2016 g_dbgp.ko
-rw-r--r-- 1 root root 406915 Mar 11  2016 g_ether.ko
-rw-r--r-- 1 root root 488028 Mar 11  2016 g_ffs.ko
-rw-r--r-- 1 root root 213017 Mar 11  2016 g_fuzz.ko
-rw-r--r-- 1 root root 224254 Mar 11  2016 g_hid.ko
-rw-r--r-- 1 root root 331832 Mar 11  2016 g_mass_storage.ko
-rw-r--r-- 1 root root 206815 Mar 11  2016 g_midi.ko
-rw-r--r-- 1 root root 604670 Mar 11  2016 g_multi.ko
-rw-r--r-- 1 root root 331664 Mar 11  2016 g_ncm.ko
-rw-r--r-- 1 root root 188169 Mar 11  2016 g_printer.ko
-rw-r--r-- 1 root root 291227 Mar 11  2016 g_serial.ko
-rw-r--r-- 1 root root 280383 Mar 11  2016 g_webcam.ko
-rw-r--r-- 1 root root 200463 Mar 11  2016 g_zero.ko
-rw-r--r-- 1 root root 194569 Mar 11  2016 gadgetfs.ko
root@orangepi:~#
```

Fuzzing Target

- 디바이스가 호스트로 전송하는 정보들

- USB Descriptors Fuzzing
  - Device descriptors
  - Configuration descriptors
  - Interface descriptors
  - Endpoint descriptors
  - Etc...

- File System Fuzzing
  - Using Mass-storage gadget
  - Mutation variety file-system image

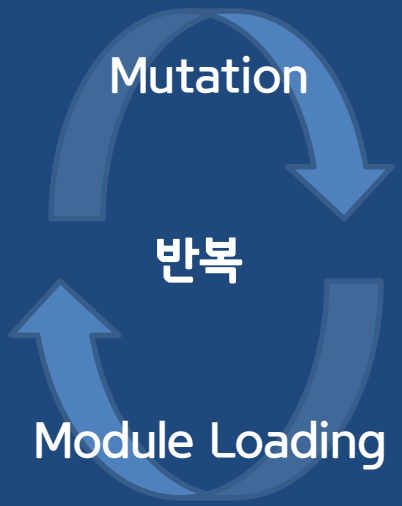
Descriptor Fuzzer 구현 방법

- Build kernel for orange-pi mini
- Modify Kernel-Level USB gadget source code for fuzzing
- Implement User-Level Fuzzer using python
  - Make mutated image or descriptors for fuzzing
  - Load Gadget module using ‘modprobe’ command
  - Wait for enumeration done
  - Unload gadget module using ‘modprobe -r’ command
  - Repeat these
- Mutation methods
  - Evil payloads DB for fuzzing
  - Using radamsa (mutation tool by google, need to cross-compile)

USB Fuzzer의 구성

- USB descriptor Fuzzing
  - descfuzz.py
  - g_fuzz.ko
  - usbfuzz.c
- File System Fuzzing
  - fsfuzz_radamsa.py
  - fsfuzz_fuzzdb.py
  - g_mass_storage.ko

Implement : Descriptor Fuzzer




```python
#!/usr/bin/python

import sys
import os
import time

def main(argv):
    while True:
        # create descriptors
        os.system('./usbfuzz')
        descFile = '/tmp/fuzz.desc'
        strDescFile = '/tmp/strFuzz.desc'

        os.system('modprobe g_fuzz descFile=' + descFile + ' strDescFile=' + strDescFile)
        time.sleep(5)
        os.system('modprobe -r g_fuzz')

if __name__ == "__main__":
    main(sys.argv)
```

descfuzz.py

Implement : Descriptor Fuzzer

- g_fuzz.ko
  - USB descriptor Fuzzing을 위한 특수 gadget
  - User level의 descriptor 파일을 Parsing하여 usb gadget의 descriptor 데이터를 생성
  - Usage:
    modprobe g_fuzz.ko descFile=<descriptor binary file> strDescFile=<string descriptor binary file>

Implement : Descriptor Fuzzer

- g_fuzz.ko – user defined descriptor parsing

```c
static void
DescriptorParsingAndSetting(char *buffer, int size)
{
    int len, functionIndex, configIndex, interfaceIndex, endpointIndex, offset = 0;
    int functionSize = 0, num;

    /* parsing device descriptor */
    //if (buffer[0] != sizeof(device_desc)) return;
    //if (buffer[1] != USB_DT_DEVICE) return;
    memcpy(&device_desc, &buffer[0], sizeof(device_desc));
    offset += sizeof(struct usb_device_descriptor);

    if (device_desc.bNumConfigurations == 0)
    {
        descFile = NULL;
        return;
    }

    num = device_desc.bNumConfigurations + 1;
    //printk("configs: %d\n", num-1);

    config_desc = vzalloc(sizeof(struct usb_config_descriptor)*num);
    intf_desc = vzalloc(sizeof(struct usb_interface_descriptor*)*num);
    fs_ep_desc = vzalloc(sizeof(struct usb_endpoint_descriptor**)*num);
#ifdef CONFIG_USB_GADGET_DUALSPEED
    hs_ep_desc = vzalloc(sizeof(struct usb_endpoint_descriptor**)*num);
#endif
    fs_fuzzer_function = vzalloc(sizeof(struct usb_descriptor_header**)*num);
#ifdef CONFIG_USB_GADGET_DUALSPEED
    hs_fuzzer_function = vzalloc(sizeof(struct usb_descriptor_header**)*num);
#endif

    /* parsing configuration descriptor */
    for (configIndex = 0 ; configIndex < device_desc.bNumConfigurations; configIndex++)
    {
        memcpy(&config_desc[configIndex], &buffer[offset], sizeof(struct usb_config_descriptor));
        offset += sizeof(struct usb_config_descriptor);

        if (config_desc[configIndex].bNumInterfaces == 0) continue;

        num = config_desc[configIndex].bNumInterfaces + 1;
        intf_desc[configIndex] = vzalloc(sizeof(struct usb_interface_descriptor)*num);
        fs_ep_desc[configIndex] = vzalloc(sizeof(struct usb_endpoint_descriptor*)*num);
#ifdef CONFIG_USB_GADGET_DUALSPEED
        hs_ep_desc[configIndex] = vzalloc(sizeof(struct usb_endpoint_descriptor*)*num);
#endif

        functionSize = config_desc[configIndex].bNumInterfaces;
        //printk("interfaces: %d\n", functionSize);

        /* parsing interface descriptor */
        for (interfaceIndex = 0; interfaceIndex < config_desc[configIndex].bNumInterfaces; interfaceIndex++)
        {
            /* copy interface descriptor */
            memcpy(&intf_desc[configIndex][interfaceIndex], &buffer[offset], sizeof(struct usb_interface_descriptor));
            offset += sizeof(struct usb_interface_descriptor);

            if (intf_desc[configIndex][interfaceIndex].bNumEndpoints == 0) continue;
            num = intf_desc[configIndex][interfaceIndex].bNumEndpoints + 1;
            //printk("endpoints: %d\n", num-1);

            fs_ep_desc[configIndex][interfaceIndex] = vzalloc(sizeof(struct usb_endpoint_descriptor)*num);
#ifdef CONFIG_USB_GADGET_DUALSPEED
            hs_ep_desc[configIndex][interfaceIndex] = vzalloc(sizeof(struct usb_endpoint_descriptor)*num);
#endif

            functionSize += intf_desc[configIndex][interfaceIndex].bNumEndpoints;

            /* parsing endpoint descriptor */
            for (endpointIndex = 0; endpointIndex < intf_desc[configIndex][interfaceIndex].bNumEndpoints; endpointIndex++)
            {
                /* copy endpoint descriptor */
                memcpy(&fs_ep_desc[configIndex][interfaceIndex][endpointIndex], &buffer[offset], sizeof(struct usb_endpoint_descriptor));
#ifdef CONFIG_USB_GADGET_DUALSPEED
                memcpy(&hs_ep_desc[configIndex][interfaceIndex][endpointIndex], &buffer[offset], sizeof(struct usb_endpoint_descriptor));
#endif
                offset += sizeof(struct usb_endpoint_descriptor);
            }
        }

        /* set configuration function */
        functionIndex = 0;
        functionSize += 2;
        //printk("function size: %d\n", functionSize);
```

Implement : Descriptor Fuzzer

- g_fuzz.ko - Load String Descriptors

```
static void
StringDescriptorLoading(char *buffer, int size)
{
    int offset, len, idx;
    char *pbuf;

    pbuf = buffer;
    for (idx = 0; (idx < 255) && ((int)pbuf < (int)buffer + size); idx++)
    {
        strings[idx].id = pbuf[0];
        len = strlen(pbuf+1);
        strings[idx].s = vzalloc(len+1);
        strcpy(strings[idx].s, pbuf+1);
        pbuf += len+2;
    }
    strings[idx].id = 0;
    strings[idx].s = NULL;
}
```

Implement : Descriptor Fuzzer

- usbfuzz.c
  - Make abnormal usb descriptors
  - Storing random values to descriptor fields

```c
            for (k = 0; k < tmpInterface.bNumEndpoints; k++)
            {
                /* make endpoint descriptor */
                copyFillRandom(&tmpEndpoint, &epDesc, sizeof(tmpEndpoint), 254);

                write(fd, &tmpEndpoint, sizeof(tmpEndpoint));
```

```c
void copyFillRandom(void *d, void *s, int size, int max)
{
    int i;
    unsigned char *dst = (unsigned char*)d, *src = (unsigned char *)s;

    for (i = 0; i < size; i++)
    {
        if (src[i] == 0)
            dst[i] = rand() % max + 1;
        else
            dst[i] = src[i];
    }
}
```

File System Fuzzing

Implement : fsfuzz_radamsa.py

```
#!/usr/bin/python

import sys
import os
import time

def main(argv):
    if len(sys.argv) != 2 :
        print argv[0], ' <image file>\n'
        sys.exit(1)

    if os.path.isfile(argv[1]) == False :
        print argv[1], ' file is not exist!\n'
        sys.exit(2)

    while True:
        # file mutation
        os.system('cat ' + argv[1] + '| radamsa > ' + argv[1] + '.fuzz')
        filefullpath = os.path.abspath(argv[1] + '.fuzz')
        os.system('modprobe g_mass_storage file=' + filefullpath + ' stall=0')
        time.sleep(3)
        os.system('modprobe -r g_mass_storage')
        #time.sleep(1)
        os.system('cp ' + filefullpath + ' fsbackup.fuzz')

if __name__ == "__main__":
    main(sys.argv)
```

Implement : fsfuzz_fuzzdb.py

```
#!/usr/bin/python

import sys
import os
import time
import fuzz_db

def main(argv):
    index = 0

    if len(sys.argv) != 2 :
        print argv[0], ' <image file>\n'
        sys.exit(1)

    if os.path.isfile(argv[1]) == False :
        print argv[1], ' file is not exist!\n'
        sys.exit(2)

    payload = fuzz_db.get_payload()
    while True:
        # file mutation
        #os.system('cat ' + argv[1] + '| radamsa > ' + argv[1] + '.fuzz')
        os.system('cp ' + argv[1] + ' ' + argv[1] + '.fuzz')

        for fuzzstr in payload :
            file = open(argv[1], 'rb')
            content = file.read()
            file.close()
            file = open(argv[1] + '.fuzz', 'wb')
            content = content[:index] + fuzzstr + content[index+4:]
            file.write(content)
            file.close()
            filefullpath = os.path.abspath(argv[1] + '.fuzz')
            os.system('modprobe g_mass_storage file=' + filefullpath + ' stall=0')
            time.sleep(3)
            os.system('modprobe -r g_mass_storage')
            #time.sleep(1)
            os.system('cp ' + filefullpath + ' fsbackup.fuzz')
        index = index + 4

if __name__ == "__main__":
    main(sys.argv)
```

USB Fuzzer 실행 방법

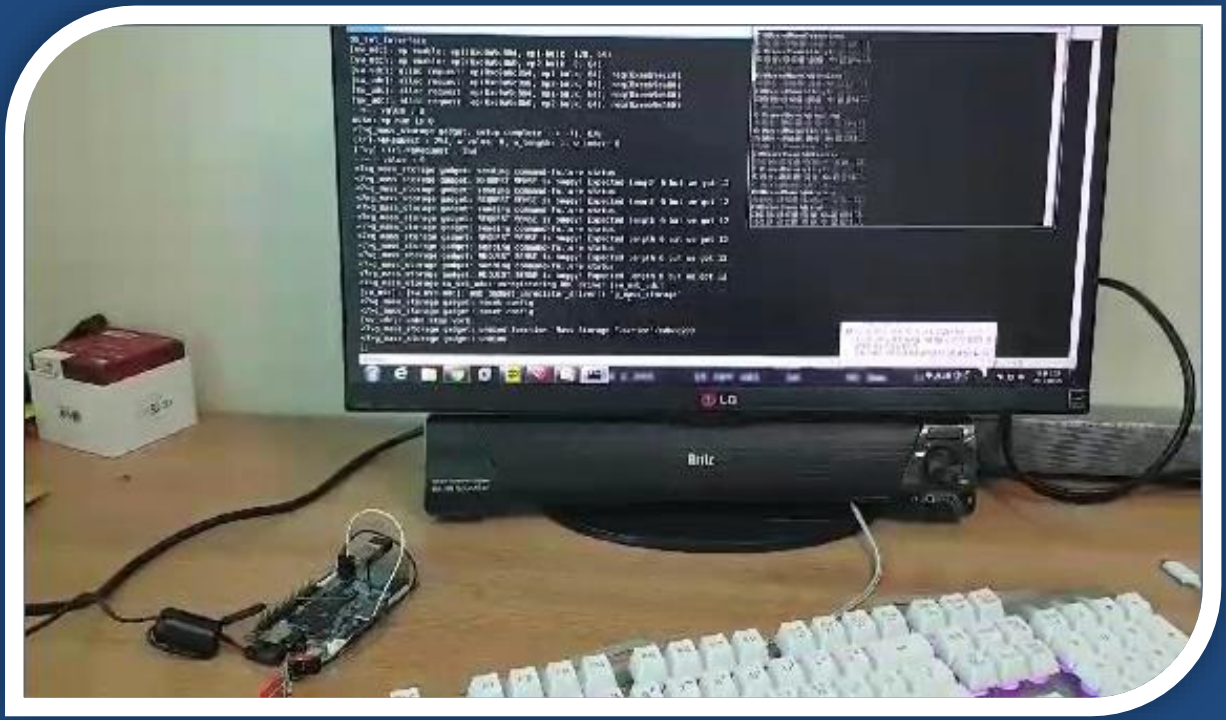
- USB descriptor Fuzzer
  - **just run "python descfuzz.py"**
    - g_fuzz.ko : auto loading by descfuzz.py
    - usbfuzz : auto running by descfuzz.py

- File System Fuzzer
  - Radamsa ver : **python fsfuzz_radamsa.py**
  - Fuzz DB ver : **python fsfuzz_fuzzdb.py**

Crash Detection

- Host로부터의 응답이 살아 있는지를 체크
  - 오류 발생 시 Kernel Panic이 발생하기 때문
- 방법1 : USB packet 응답을 체크
  - Kernel level에서 구현
- 방법2 : Host OS에 간단한 echo TCP Server를 가동한 후, 응답이 오는지를 체크
  - User level에서 구현

Fuzzer 실행 화면



Multi-Media File Fuzzing

Fuzzing 방법

- Radamsa
  - Mutation tool by google
  - https://github.com/aoh/radamsa
- Dumb Fuzzing
  - 무작위 변조
  - Insert, Delete, Edit(Overwriting)
- Fuzz Payloads
  - Format String Bug
  - Buffer Overflow
  - Command Injection
  - ETC

Fuzzing 주요 코드

- Fuzzing 과정
  - 샘플 파일 선택
  - 샘플 파일 mutation
    - Radamsa
    - Dumb Fuzzing
    - Fuzz Payloads
  - 파일 재생
    - 재생 커멘드 이용
  - Crash 발생 확인
    - Crash 발생 파일 보관
    - Logging

```
def fuzz(output_folder, prefix, cases):
    cnt = 1
    len_chk = ""
    fileList = os.listdir("./sample")
    n = len(fileList)
    print "[+] Sample List : " + str(fileList)
    print ""

    while cnt <= int(cases):
        samplefile = fileList[random.randrange(n)]
        fname, ext = os.path.splitext(samplefile)

        print "["+str(cnt)+"] Fuzzing Format : " + ext
        print "[+] Choose Sample File : " + samplefile
        case = Case(samplefile, output_folder, prefix)

        cmd_getPid = "pgrep kodi.bin"
        old_pid = subprocess_open(cmd_getPid)

        try:
            print "[+] Mutate Sample File (Custom Mutate) "
            filename = case.generateCaseAndWriteFile(cnt)

            print "[+] Play Media File "
            os.system('kodi-send --action="PlayMedia(`pwd`/' + filename + ')"')
            time.sleep(4)
        except:
            print "[-] Mutate Error. Retry!!!! "
            continue

        new_pid = subprocess_open(cmd_getPid)
        print "[+] pid : " + old_pid + "->" + new_pid
        if old_pid != new_pid :
            print "[+] Crash!!!! : " + filename
            cnt += 1
            old_pid = new_pid
            print "[-] Move Crash file \"./crash/\" Directory "
            os.system("mv " + filename + " ./crash/")
            time.sleep(8)
        else :
            cnt += 1
            print "[-] Remove File : " + filename
            os.system("rm " + filename )
        print ""
```

Fuzzing 주요 코드

- Dumb Fuzzing

```
def _mod_editBytes(self, offset=None, to_bytes=None):
    if offset == None: offset = self.__randOffset()
    if to_bytes == None:
        to_bytes = [self.random.randint(0,255) for _ in range( self.random.randint(0,255) )]
    for i in range(len(to_bytes)):
        self.modified[(offset+i) % self.lengthOfMod()] = to_bytes[i]
    return True

def _mod_deleteBytes(self, offset=None, length=None):
    if offset == None: offset = self.__randOffset()
    if length == None: length = self.random.randint(0,255)
    if length + offset > self.lengthOfMod():
        length = (- offset) + self.lengthOfMod()
    for i in range(length):
        self.modified.pop(offset)
    return True

def _mod_insertBytes(self, offset=None, new_bytes=None):
    if offset == None: offset = self.__randOffset()
    if new_bytes == None:
        new_bytes = ''.join(chr(self.random.randint(0,255)) for _ in range(
            self.random.randint(0,255) ))
    for b in new_bytes[::-1]:
        self.modified.insert(offset,b)
    return True
```

```
def _mod_increaseBytes(self, offset=None, length=None, decint=None):
    if offset == None: offset = self.__randOffset()
    if length == None: length = self.random.randint(0,255)
    if decint == None: decint = self.random.randint(-255,255)

    if (offset + length) > self.lengthOfMod():
        length = self.lengthOfMod() - offset
    for i in range(length):
        self.modified[offset+i] = (self.modified[offset+i] + decint + 256 ) % 256
    return True

def _mod_mutate(self, to_offset=None, from_offset=None, size=None):
    if to_offset == None: to_offset = self.__randOffset()
    if from_offset == None: from_offset = self.__randOffset()
    if size == None: size = self.random.randint(0,255)

    if to_offset > from_offset:
        to_offset -= from_offset

    tmp = self.modified[from_offset:from_offset+size]

    for i in range(size):
        self.modified.pop(from_offset)
    for b in tmp[::-1]:
        self.modified.insert(to_offset,b)
```

- Fuzz Payloads

```
def _mod_useFuzzPayloads(self, offset=None, payload=None):
    print "[-] Use Fuzz Payloads "
    if offset == None:
        offset = self.__randOffset()
    if payload == None:
        payload_file = os.listdir("./FuzzPayloads/")
        fp = open("./FuzzPayloads/" + payload_file[ self.random.randint(0, len(payload_file)-1) ], "rb")
        fuzzpayloads = fp.readlines()
        fp.close()
        payload = fuzzpayloads[ self.random.randint( 0, len(fuzzpayloads)-1) ]

    for b in payload[::-1]:
        self.modified.insert(offset, b)

    return True
```

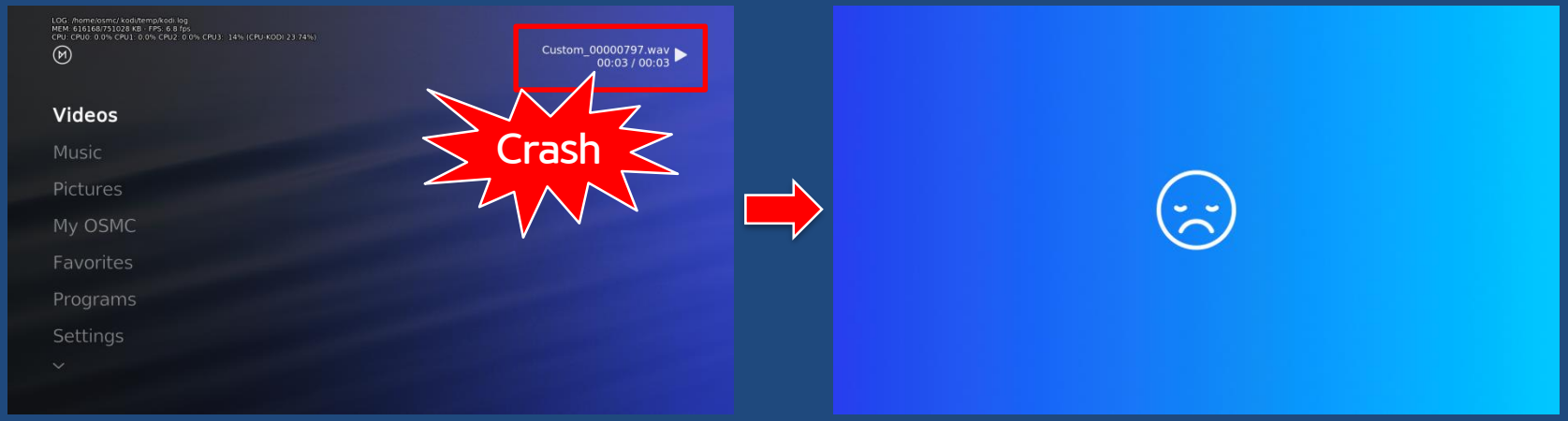
Fuzzer 실행

- python MediaFuzzRadamsa.py 1000

```
osmc@osmc06:~/MediaFuzz$ python MediaFuzzRadamsa.py 1000
[+] Total count : 1000
[+] Sample List : ['sample.amr', 'sample.ogg', 'sample.au', 'Enrique-Iglesias-Bailando-feat-Sean-Paul-
', 'sample.m4a', 'sample.mka', 'Southern_All_Stars_-_I_AM_YOUR_SINGER.m4a', 'SampleAudio_0.5mb.mp3',
mp3', 'sample.wma', 'sample.aiff']

[1] Fuzzing Format : .m4a
[+] Choose Sample File : sample.m4a
[+] Mutate Sample File (use Radamsa)
[+] Play Media File
Sending action: PlayMedia(/home/osmc/MediaFuzz/payloads/Radamsa-00000001.m4a)
[+] pid : 723
        ->723

[-] Remove File : Radamsa-00000001.m4a

[2] Fuzzing Format : .m4a
[+] Choose Sample File : sample.m4a
[+] Mutate Sample File (use Radamsa)
[+] Play Media File
Sending action: PlayMedia(/home/osmc/MediaFuzz/payloads/Radamsa-00000002.m4a)
[+] pid : 723
        ->723
```

Crash 확인 방법

- Crash 발생



- 재생 전후 플레이어의 pid 확인

```
root     18089   292  0 17:13 ?        00:00:00 sudo -u osmc /usr/lib/kodi/kodi.bin --standalone -fs --lircdev /var/run/lirc/lircd
osmc     18090 18089 60 17:13 ?        00:00:15 /usr/lib/kodi/kodi.bin --standalone -fs --lircdev /var/run/lirc/lircd
```

```
[7] Fuzzing Format : .m4a
[+] Choose Sample File : sample.m4a
[-] Read sample.m4a
[-] OK. ext m4a, out payloads
[+] Mutate Sample File (Custom Mutate)
[-] Generated payloads/Custom_00000007.m4a.
[+] Play Media File
Sending action: PlayMedia(/home/osmc/MediaFuzz/payloads/Custom_00000007.m4a)
[+] pid : 18090
        ->18260

[+] Crash!!!! : payloads/Custom_00000007.m4a
[-] Move Crash file "./crash/" Directory
```

Fuzzing 결과

- Target Formats
  - aac, ac3, aiff, amr, au, flac, m4a, mid, mka, mp3, ogg, ra, voc, wav, wma

- Crash 발생 Formats
  - flac, m4a, mka, ra, wma

```
osmc@osmc03:~/MediaFuzz/crash$ ls
Custom_00000015.ra     Radamsa-00001335.m4a   Radamsa-00003051.m4a   Radamsa-00005600.m4a   Radamsa-00008380.m4a   Radamsa-00011745.m4a
Custom_00000157.ra     Radamsa-00001358.m4a   Radamsa-00003095.wma   Radamsa-00005903.m4a   Radamsa-00008563.m4a   Radamsa-00011996.m4a
Custom_00000315.wma    Radamsa-00001437.m4a   Radamsa-00004034.m4a   Radamsa-00005963.m4a   Radamsa-00008872.m4a   Radamsa-00012289.m4a
Custom_00000360.mka    Radamsa-00001448.flac  Radamsa-00004508.m4a   Radamsa-00006502.wma   Radamsa-00008994.m4a   Radamsa-00012516.wma
Radamsa-00000078.m4a   Radamsa-00001468.wma   Radamsa-00004529.m4a   Radamsa-00007035.m4a   Radamsa-00009432.m4a   Radamsa-00012626.mka
Radamsa-00000291.m4a   Radamsa-00002209.m4a   Radamsa-00004733.m4a   Radamsa-00007058.m4a   Radamsa-00010071.wma
Radamsa-00000322.m4a   Radamsa-00002374.m4a   Radamsa-00004994.m4a   Radamsa-00007259.m4a   Radamsa-00010251.m4a
Radamsa-00000428.m4a   Radamsa-00002383.m4a   Radamsa-00005232.m4a   Radamsa-00007547.m4a   Radamsa-00010851.m4a
Radamsa-00000590.m4a   Radamsa-00002677.m4a   Radamsa-00005524.m4a   Radamsa-00007621.m4a   Radamsa-00011444.m4a
Radamsa-00001197.m4a   Radamsa-00003011.m4a   Radamsa-00005590.m4a   Radamsa-00008005.m4a   Radamsa-00011606.m4a
```

결론

- USB 포트를 이용하여 대상 장비를 장악하는 것이 가능함
- USB를 공격하기 위한 방법은 USB Stack, File System, Multi-media file 등 다양함
- Fuzzing을 통해 USB 공격을 자동화할 수 있음

QNA

감사합니다!

기타 참고자료

btsnoop_hci.log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

```
Bluetooth HCI H4
    [Direction: Sent (0x00)]
    HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
    .... 0000 0000 1011 = Connection Handle: 0x00b
    ..10 .... .... .... = PB Flag: First Automatically Flushable Packet (2)
    00.. .... .... .... = BC Flag: Point-To-Point (0)
    Data Total Length: 21
    [Connect in frame: 1732]
    [Source BD_ADDR: 00:00:00_00:00:00 (00:00:00:00:00:00)]
    [Source Device Name: Galaxy Note5]
    [Source Role: Master (1)]
    [Destination BD_ADDR: 20:17:01:03:46:89 (20:17:01:03:46:89)]
    [Destination Device Name: GOOHONG]
    [Destination Role: Slave (2)]
    [Last Role Change in Frame: 1730]
    [Current Mode: Active Mode (0)]
    [Last Mode Change in Frame: 1959]
Bluetooth L2CAP Protocol
    Length: 17
    CID: Dynamically Allocated Channel (0x0041)
    [Connect in frame: 1771]
    [Disconnect in frame: 2044]
    [PSM: RFCOMM (0x0003)]
Bluetooth RFCOMM Protocol
    > Address: E/A flag: 1, C/R fla
    > Control: Frame type: Unnumber
    Payload length: 12
    Credits: 4
    Frame Check Sequence: 0x86
Bluetooth SPP Packet
    Data: 2a6161616161616161616161
0000  02 0b 20 15 00 11 00 41  00
0010  61 61 61 61 61 61 61 61  61 86                aaaaaaaa a.
```

Control (btrfcomm.control), 1 byte

HCI Packet Type

- HCI Command : 0x01 – 명령 전송 (from host)
- HCI Event : 0x04 – 결과, 상태값 전송 (to host)
- ACL Data : 0x02 – 데이터 전송 (both)

* HCI : Host-Controller Interface

btsnoop_hci.log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ··· <Ctrl-/>

```
Bluetooth HCI H4
    [Direction: Sent (0x00)]
    HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
    .... 0000 0000 1011 = Connection Handle: 0x00b
    ..10 .... .... .... = PB Flag: First Automatically Flushable Packet (2)
    00.. .... .... .... = BC Flag: Point-To-Point (0)
    Data Total Length: 21
    [Connect in frame: 1732]
    [Source BD_ADDR: 00:00:00_00:00:00 (00:00:00:00:00:00)]
    [Source Device Name: Galaxy Note5]
    [Source Role: Master (1)]
    [Destination BD_ADDR: 20:17:01:03:46:89 (20:17:01:03:46:89)]
    [Destination Device Name: GOOHONG]
    [Destination Role: Slave (2)]
    [Last Role Change in Frame: 1730]
    [Current Mode: Active Mode (0)]
    [Last Mode Change in Frame: 1959]
Bluetooth L2CAP Protocol
    Length: 17
    CID: Dynamically Allocated Channel (0x0041)
    [Connect in frame: 1771]
    [Disconnect in frame: 2044]
    [PSM: RFCOMM (0x0003)]
Bluetooth RFCOMM Protocol
  > Address: E/A flag: 1, C/R flag: 1, Direction: 0, Channel: 1
  > Control: Frame type: Unnumbered Information with Header check (UIH) (0xef), P/F flag: 1
    Payload length: 12
    Credits: 4
    Frame Check Sequence: 0x86
Bluetooth SPP Packet
    Data: 2a6161616161616161616161
```

```
0000  02 0b 20 15 00 11 00 41  00 0b ff 19 04 2a 61 61   .. ....A ....*aa
0010  61 61 61 61 61 61 61 61  61 86                     aaaaaaaa a.
```

Control (btrfcomm.control), 1 byte

btsnoop_hci.log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ··· <Ctrl-/>

```
Bluetooth HCI H4
    [Direction: Sent (0x00)]
    HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
    .... 0000 0000 1011 = Connection Handle: 0x00b
    ..10 .... .... .... = PB Flag: First Automatically Flushable Packet (2)
    00.. .... .... .... = BC Flag: Point-To-Point (0)
    Data Total Length: 21
    [Connect in frame: 1732]
    [Source BD_ADDR: 00:00:00_00:00:00 (00:00:00:00:00:00)]
    [Source Device Name: Galaxy Note5]
    [Source Role: Master (1)]
    [Destination BD_ADDR: 20:17:01:03:46:89 (20:17:01:03:46:89)]
    [Destination Device Name: GOOHONG]
    [Destination Role: Slave (2)]
    [Last Role Change in Frame: 1730]
    [Current Mode: Active Mode (0)]
    [Last Mode Change in Frame: 1959]
Bluetooth L2CAP Protocol
    Length: 17
    CID: Dynamically Allocated Channel (0x0041)
    [Connect in frame: 1771]
    [Disconnect in frame: 2044]
    [PSM: RFCOMM (0x0003)]
Bluetooth RFCOMM Protocol
  > Address: E/A flag: 1, C/R flag: 1, Direction: 0, Channel: 1
  > Control: Frame type: Unnumbered Information with Header check (UIH) (0xef), P/F flag: 1
    Payload length: 12
    Credits: 4
    Frame Check Sequence: 0x86
Bluetooth SPP Packet
    Data: 2a6161616161616161616161
```

```
0000  02 0b 20 15 00 11 00 41  00 0b ff 19 04 2a 61 61   .. ....A ....*aa
0010  61 61 61 61 61 61 61 61  61 86                     aaaaaaaa a.
```

Control (btrfcomm.control), 1 byte

btsnoop_hci.log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ··· <Ctrl-/>

```
Bluetooth HCI H4
    [Direction: Sent (0x00)]
    HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
    .... 0000 0000 1011 = Connection Handle: 0x00b
    ..10 .... .... .... = PB Flag: First Automatically Flushable Packet (2)
    00.. .... .... .... = BC Flag: Point-To-Point (0)
    Data Total Length: 21
    [Connect in frame: 1732]
    [Source BD_ADDR: 00:00:00_00:00:00 (00:00:00:00:00:00)]
    [Source Device Name: Galaxy Note5]
    [Source Role: Master (1)]
    [Destination BD_ADDR: 20:17:01:03:46:89 (20:17:01:03:46:89)]
    [Destination Device Name: GOOHONG]
    [Destination Role: Slave (2)]
    [Last Role Change in Frame: 1730]
    [Current Mode: Active Mode (0)]
    [Last Mode Change in Frame: 1959]
Bluetooth L2CAP Protocol
    Length: 17
    CID: Dynamically Allocated Channel (0x0041)
    [Connect in frame: 1771]
    [Disconnect in frame: 2044]
    [PSM: RFCOMM (0x0003)]
Bluetooth RFCOMM Protocol
  > Address: E/A flag: 1, C/R flag: 1, Direction: 0, Channel: 1
  > Control: Frame type: Unnumbered Information with Header check (UIH) (0xef), P/F flag: 1
    Payload length: 12
    Credits: 4
    Frame Check Sequence: 0x86
Bluetooth SPP Packet
    Data: 2a6161616161616161616161
```

```
0000  02 0b 20 15 00 11 00 41  00 0b ff 19 04 2a 61 61   .. ....A ....*aa
0010  61 61 61 61 61 61 61 61  61 86                     aaaaaaaa a.
```

Control (btrfcomm.control), 1 byte

btsnoop_hci.log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ··· <Ctrl-/>

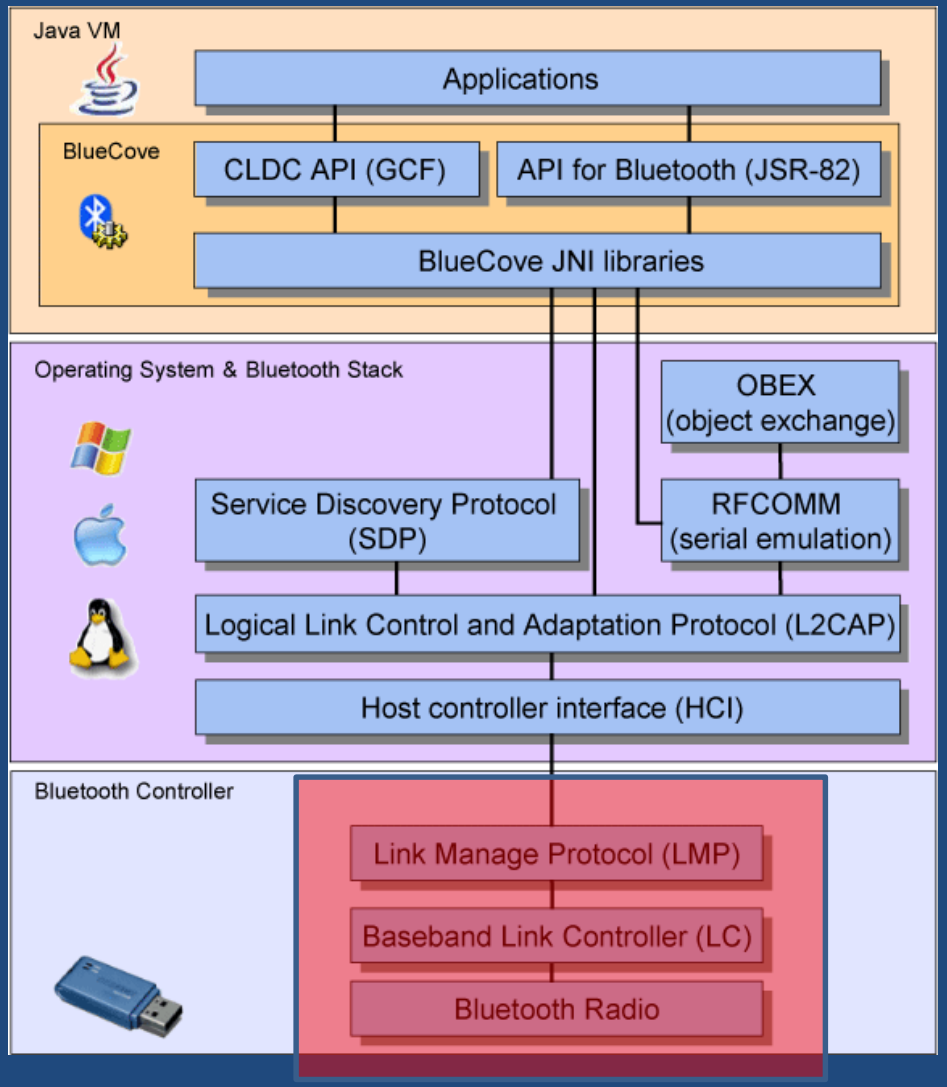
```
Bluetooth HCI H4
    [Direction: Sent (0x00)]
    HCI Packet Type: ACL Data (0x02)
Bluetooth HCI ACL Packet
    .... 0000 0000 1011 = Connection Handle: 0x00b
    ..10 .... .... .... = PB Flag: First Automatically Flushable Packet (2)
    00.. .... .... .... = BC Flag: Point-To-Point (0)
    Data Total Length: 21
    [Connect in frame: 1732]
    [Source BD_ADDR: 00:00:00_00:00:00 (00:00:00:00:00:00)]
    [Source Device Name: Galaxy Note5]
    [Source Role: Master (1)]
    [Destination BD_ADDR: 20:17:01:03:46:89 (20:17:01:03:46:89)]
    [Destination Device Name: GOOHONG]
    [Destination Role: Slave (2)]
    [Last Role Change in Frame: 1730]
    [Current Mode: Active Mode (0)]
    [Last Mode Change in Frame: 1959]
Bluetooth L2CAP Protocol
    Length: 17
    CID: Dynamically Allocated Channel (0x0041)
    [Connect in frame: 1771]
    [Disconnect in frame: 2044]
    [PSM: RFCOMM (0x0003)]
Bluetooth RFCOMM Protocol
  > Address: E/A flag: 1, C/R flag: 1, Direction: 0, Channel: 1
  > Control: Frame type: Unnumbered Information with Header check (UIH) (0xef), P/F flag: 1
    Payload length: 12
    Credits: 4
    Frame Check Sequence: 0x86
Bluetooth SPP Packet
    Data: 2a6161616161616161616161
```

```
0000  02 0b 20 15 00 11 00 41  00 0b ff 19 04 2a 61 61   .. ....A ....*aa
0010  61 61 61 61 61 61 61 61  61 86                     aaaaaaaa a.
```

Control (btrfcomm.control), 1 byte

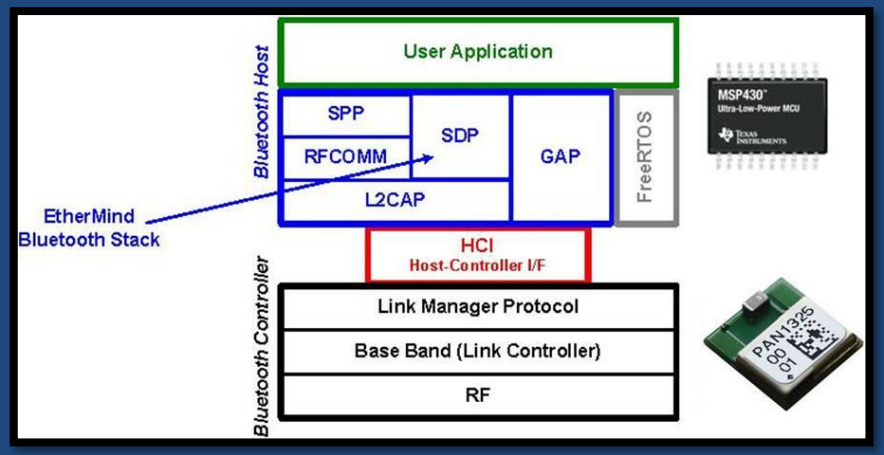
Bluetooth Stack

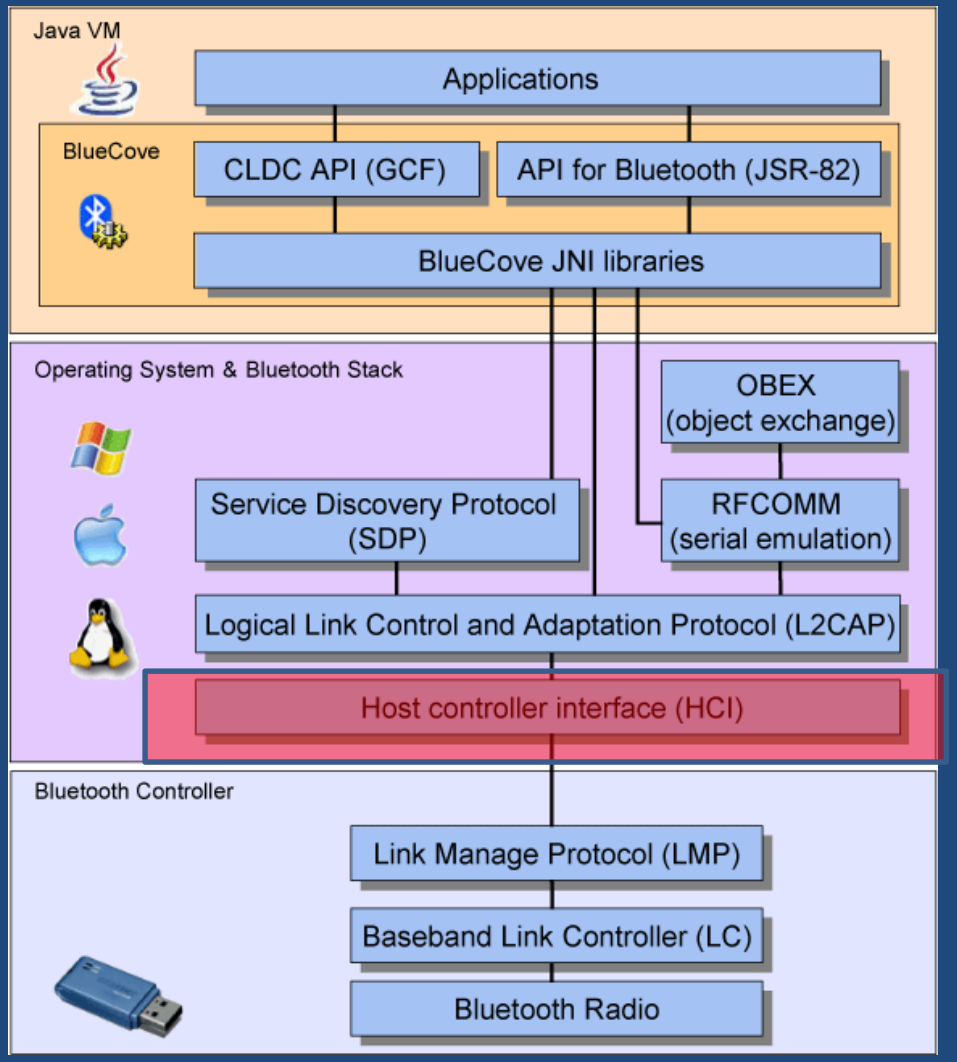
- Bluetooth Radio
  - 무선 주파수 통신 구간
- LC (Link Control)
  - 흐름 제어, 확인 응답(ACK), 재전송 요청
- LMP (Link Manager Protocol)
  - 장치간 링크 생성 및 해제
  - 인증, 암호화
  - 전원 관리



Bluetooth Stack

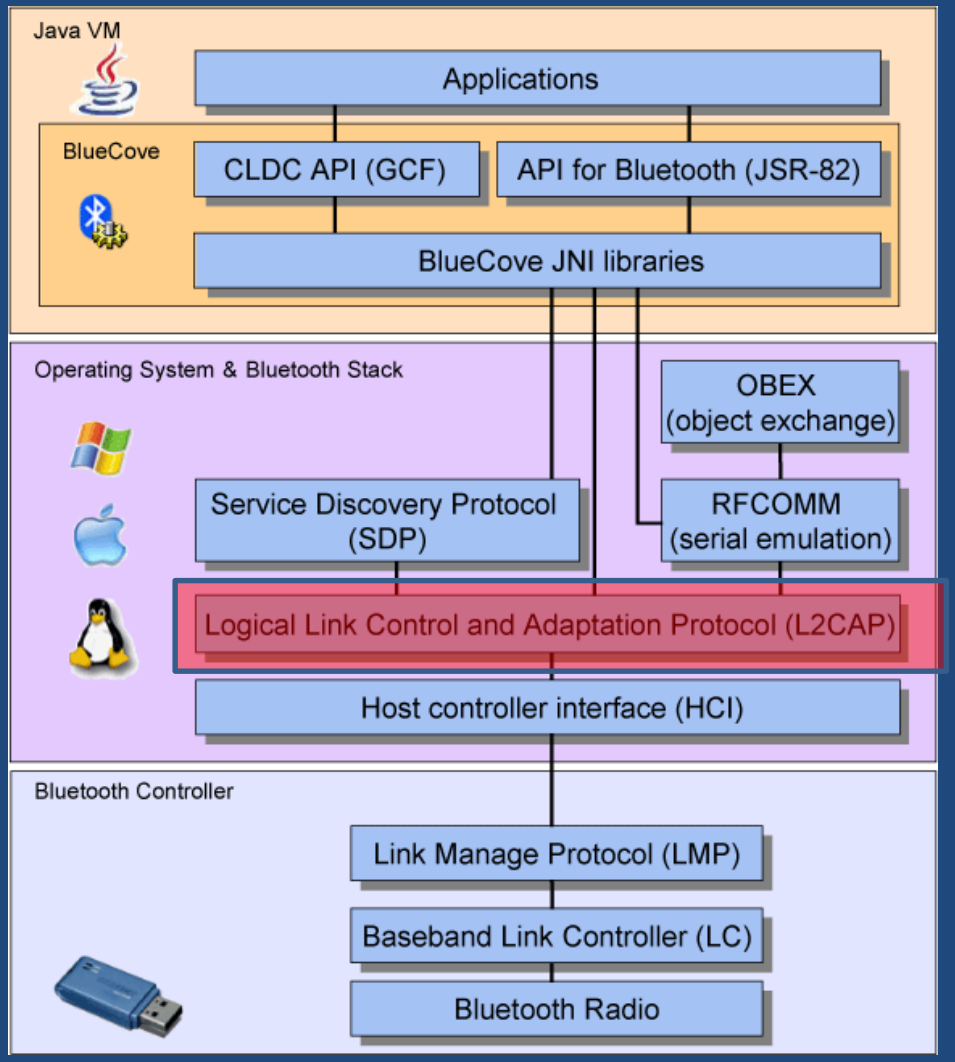
- HCI : Host Controller Interface
  - CPU와 Bluetooth IC 사이를 연결
  - host stack (CPU, OS)
  - the controller (Bluetooth 모듈)
  - UART, USB 및 PCMCIA로 연결





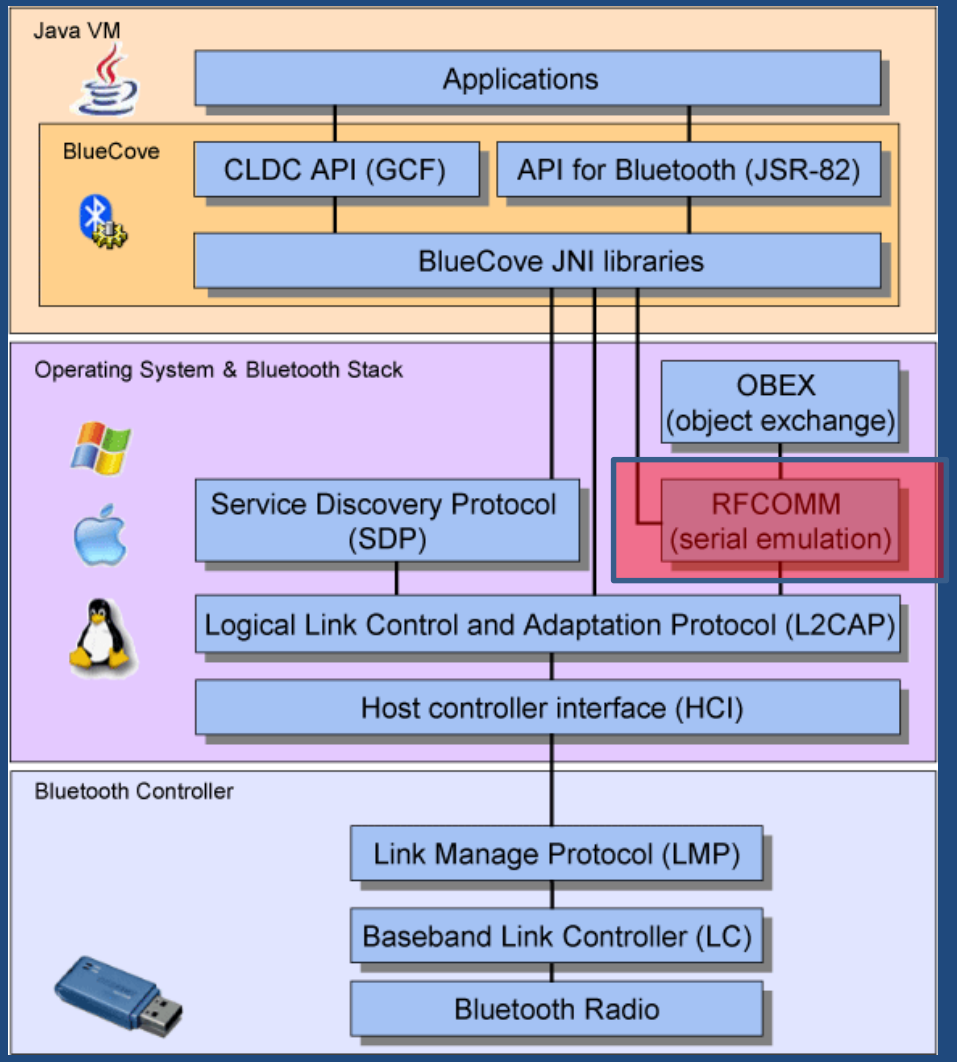
Bluetooth Stack

- L2CAP
  - 논리적인 연결 생성
  - Multiplexing (다중화)
    - 데이터의 용도 구분
  - Segmentation and reassembly
    - 패킷 조각화/복구
  - QoS management
  - TCP 레이어의 역할



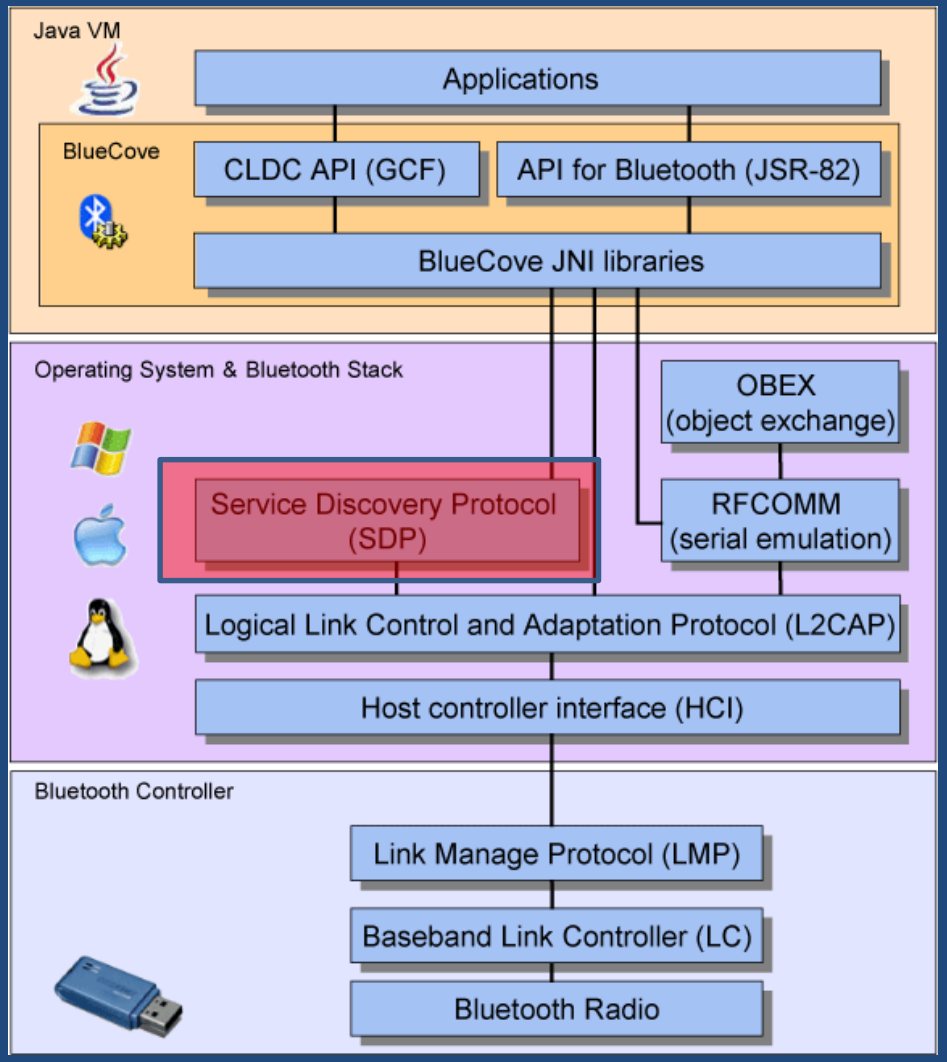
Bluetooth Stack

- RFCOMM
  - 시리얼 프로토콜 에뮬레이팅
  - Data stream 전달



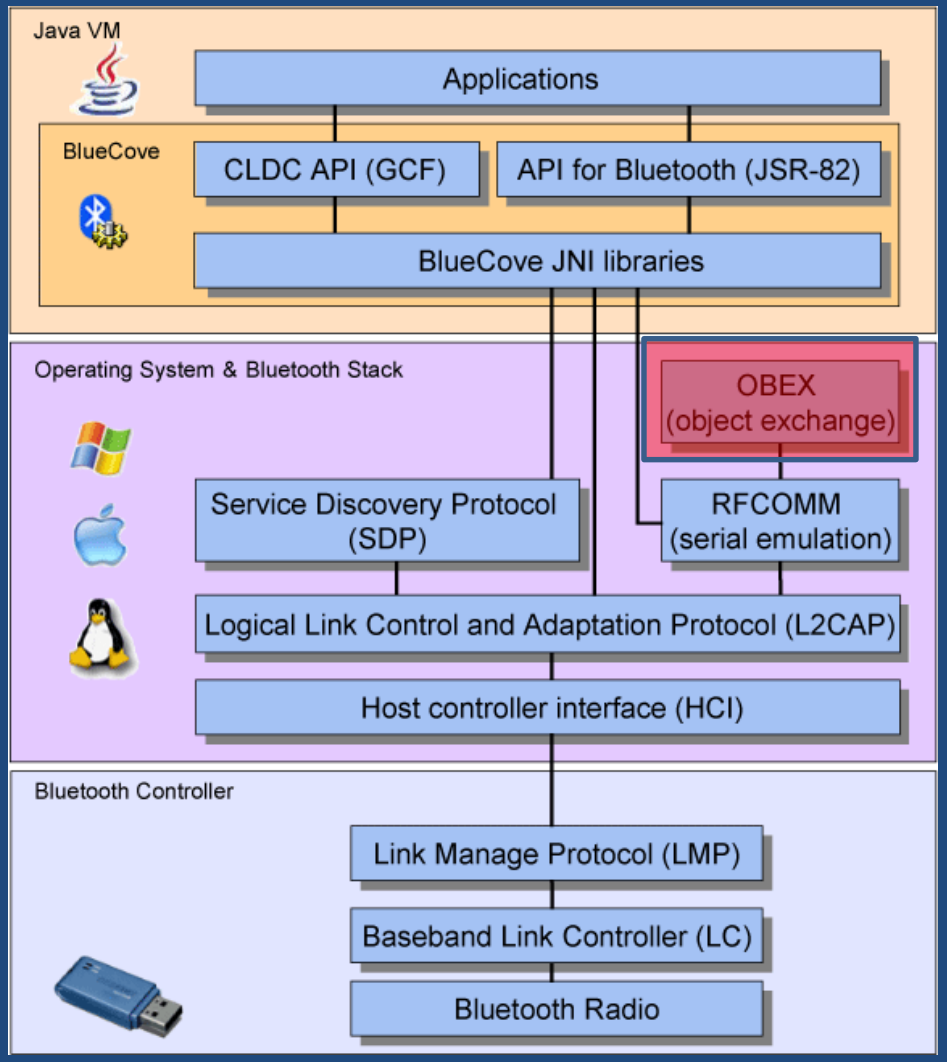
Bluetooth Stack

- SDP
  - Service discovery protocol
  - 장치에서 제공하는 기능(프로파일)들에 대한 정보 제공



Bluetooth Stack

- OBEX
  - Object Exchange
  - Data Object 교환
  - 블루투스 프로파일 중 하나



SDP Packet

btsnoop_hci (3).log

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ··· <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
71	5.116957	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
72	5.154534	20:17:01:03:46:89 (…	localhost ()	L2CAP	19	Rcvd Configure Response - Success (SCID: 0x004f)
73	5.174329	20:17:01:03:46:89 (…	localhost ()	L2CAP	21	Rcvd Configure Request (DCID: 0x004f)
74	5.176741	localhost ()	20:17:01:03:46:89 (…	L2CAP	19	Sent Configure Response - Success (SCID: 0x0043)
75	5.176983	localhost ()	20:17:01:03:46:89 (…	SDP	29	Sent Service Search Attribute Request : L2CAP: Attribute Range (0x0000 - 0xffff)
76	5.180748	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
77	5.217241	20:17:01:03:46:89 (…	localhost ()	SDP	57	Rcvd Service Search Attribute Response (fragment)
78	5.219803	localhost ()	20:17:01:03:46:89 (…	SDP	31	Sent Service Search Attribute Request : L2CAP: Attribute Range (0x0000 - 0xffff)
79	5.273926	20:17:01:03:46:89 (…	localhost ()	SDP	42	Rcvd Service Search Attribute Response
80	5.274941	localhost ()	20:17:01:03:46:89 (…	L2CAP	17	Sent Disconnection Request (SCID: 0x004f, DCID: 0x0043, PSM: 0x0001, Service: SDP)
81	5.293432	controller	host	HCI_EVT	8	Rcvd Number of Completed Packets
82	5.330836	20:17:01:03:46:89 (…	localhost ()	L2CAP	17	Rcvd Disconnection Response (SCID: 0x004f, DCID: 0x0043, PSM: 0x0001, Service: SDP)
83	9.333077	host	controller	HCI_CMD	7	Sent Disconnect
84	9.336034	controller	host	HCI_EVT	7	Rcvd Command Status (Disconnect)
85	9.418455	controller	host	HCI_EVT	7	Rcvd Disconnect Complete

```
    Data Element: Unsigned Integer 4 bytes
        0000 1... = Data Element Type: Unsigned Integer (1)
        .... .010 = Data Element Size: 4 bytes (2)
    Data Value
        Service Record Handle: 0x00010000
Service Attribute: Service Class ID List (0x1), value = Serial Port
  Attribute ID: Service Class ID List
  Value
    Data Element: Sequence uint8 3 bytes
        0011 0... = Data Element Type: Sequence (6)
        .... .101 = Data Element Size: uint8 (5)
        Data Element Var Size: 3
    Data Value
```

```
0000  36 00 3c 36 00 39 09 00  00 0a 00 01 00 00 09 00   6.<6.9.. ........
0010  01 35 03 19 11 01 09 00  04 35 0c 35 03 19 01 00   .5...... .5.5....
0020  35 05 19 00 03 08 01 09  00 06 35 09 09 65 6e 09   5....... ..5..en.
0030  00 6a 09 01 00 09 01 00  25 05 44 65 76 20 42      .j...... %.Dev B
```

Frame (42 bytes) | Reassembled SDP (63 bytes)

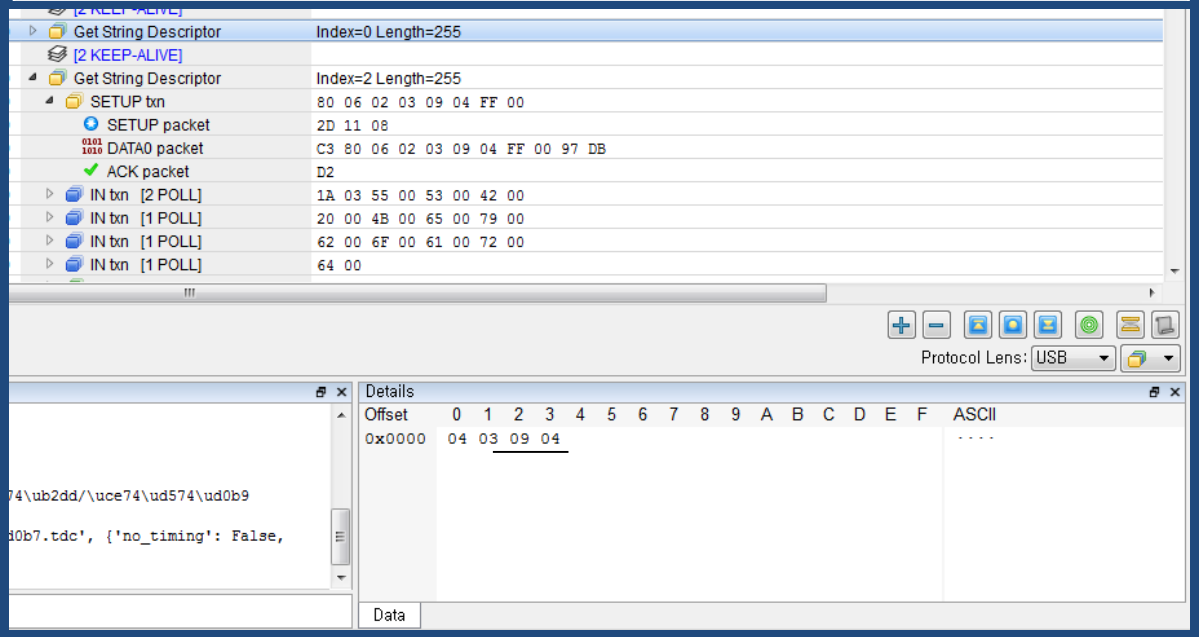
Service Attribute (btsdp.service_attribute), 8 bytes

```
> Frame 79: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
> Bluetooth
> Bluetooth HCI H4
> Bluetooth HCI ACL Packet
> Bluetooth L2CAP Protocol
> Bluetooth SDP Protocol
```

Get String Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor : Get Header
- Get String Descriptor : “USB Keyboard”
- Get Report Descriptor

The First “Get String Descriptor” Packet



Identifier	Language
0x1404	Chinese (Macau SAR)
0x041a	Croatian
0x0405	Czech
0x0406	Danish
0x0413	Dutch (Netherlands)
0x0813	Dutch (Belgium)
0x0409	English (United States)
0x0809	English (United Kingdom)
0x0c09	English (Australian)

http://www.usb.org/developers/docs/USB_LANGIDs.pdf

기타 : Report Descriptor

- Get Device Descriptor
- Set Address
- Get Device Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Device Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Set Configuration
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Configuration Descriptor
- Get Configuration Descriptor
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor
- Set Output Report
- Input Report
- Get String Descriptor
- Get String Descriptor
- Get Report Descriptor

HID 관련 정보 제공
자세한 정보 :
http://www.rennes.supelec.fr/ren/fi/elec/docs/usb/hid1_11.pdf

Details

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
0x0000	05	0C	09	01	A1	01	85	01	19	00	2A	3C	02	15	00	26	······*<···&
0x0010	3C	02	95	01	75	10	81	00	C0	05	01	09	80	A1	01	85	<···u·····
0x0020	02	19	81	29	83	25	01	75	01	95	03	81	02	95	05	81	···)·%·u·····
0x0030	01	C0															··