



# *SharkFest'20 Virtual*



## **USB Analysis 101**

Using Wireshark to analyze USB traffic

 **Tomasz Moń**  
**Etteplan**



# Table of contents



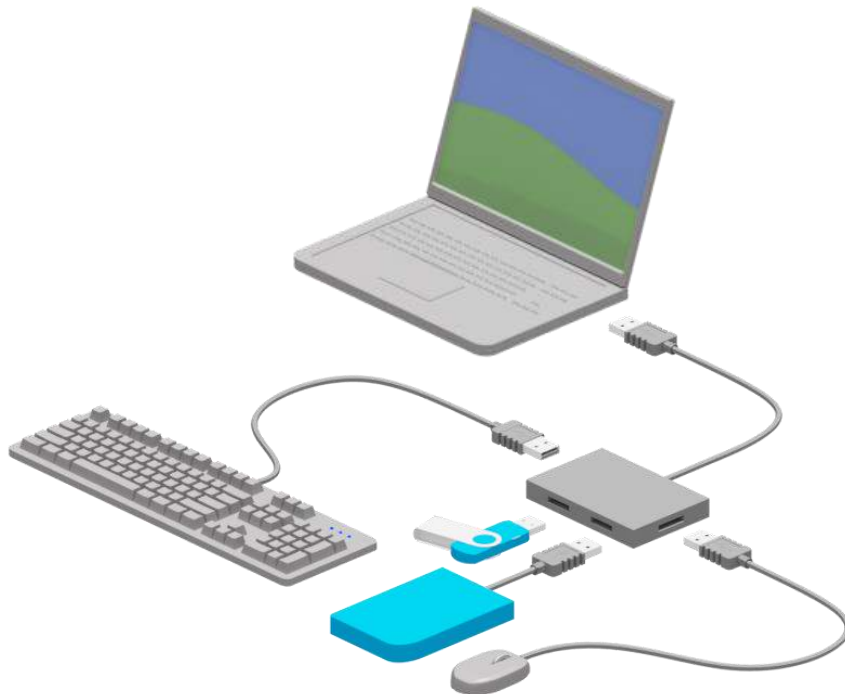
- Introduction
  - Terminology, speeds, connectors overview
  - Why USB 2.0 is still relevant?
  - USB transfer types and device classes
- USB traffic capture
  - USB “Packets”
    - Software vs hardware sniffers
  - Example traffic: USB Mass Storage
- Summary
  - Questions & Answers



# Basic USB terminology



USB	Analogous to
Host	Requester, DHCP server
Device	Responder
Port	Physical port connector
Hub	Switch, Hub
Address	Local IP address
Endpoint	Buffer, TCP/UDP Port
Class	Communication Protocol
Descriptor	Datasheet
VID	OUI
PID	Product code





# USB connectors



Pin	Mini/Micro Pin	Name
1	1	VBUS
2	2	D-
3	3	D+
N/A	4	ID
4	5	GND

**As there is just a single differential pair in USB 2.0, only Half-Duplex communication is possible**

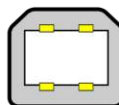
5	6	SSTx-
6	7	SSTx+
7	8	GND
8	9	SSRx-
9	10	SSRx+

**USB 3.0 is dual-simplex**

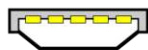
## USB 1.1 – 2.0



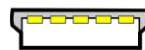
A



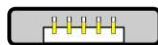
B



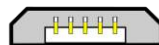
Mini-A



Mini-B

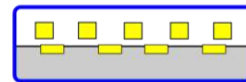


Micro-A

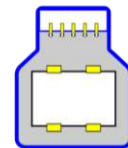


Micro-B

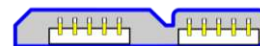
## USB 3.0



A



B



Micro-B

Image from [Wikipedia](https://en.wikipedia.org/wiki/USB) licensed under [CC BY-SA 3.0](https://creativecommons.org/licenses/by-sa/3.0/).



# USB speeds



USB 2.0 features three transmission speeds:

- Low speed (1.5 Mbps)
- Full speed (12 Mbps)
- High speed (480 Mbps)

USB 3.x is a bit more complex:

- SuperSpeed 3.2 Gen 1x1 (5 Gbps) (formerly USB 3.0)
- SuperSpeed+ 3.2 Gen 2x1 (10 Gbps; 1 lane) (formerly USB 3.1)
- SuperSpeed+ 3.2 Gen 1x2 (10 Gbps; 2 lanes) (USB-C required)
- SuperSpeed+ 3.2 Gen 2x2 (20 Gbps; 2 lanes) (USB-C required)



# USB Type C connector

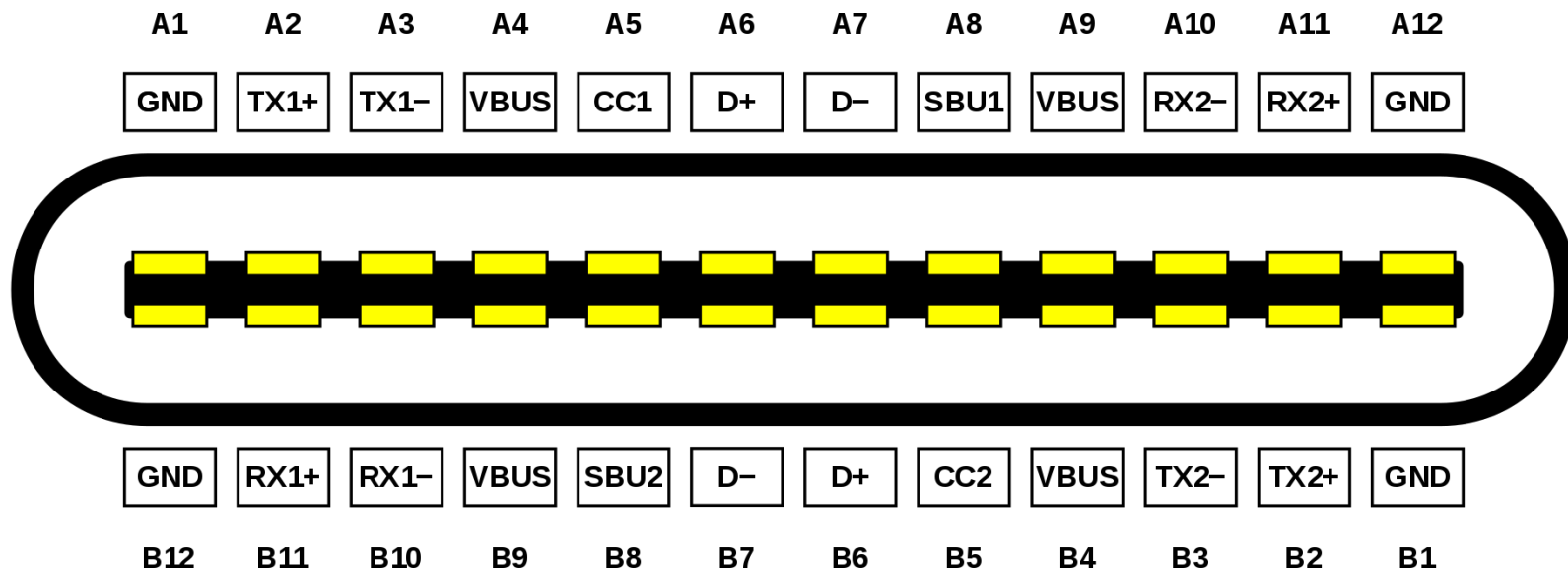


Image from [Wikipedia](https://en.wikipedia.org/wiki/USB_Type-C) licensed under [CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/).



# Speed detection



- Low speed devices feature 1.5 k $\Omega$  pull-up resistor on D-
- Full speed devices feature 1.5 k $\Omega$  pull-up resistor on D+
- High speed devices starts as Full speed, and switch to High speed after chirping sequence (D+ pull-up is disabled after chirp to balance the lines)
- USB 3.x link negotiation is much more complex, utilizing side band communication called Low Frequency Periodic Signaling (LFPS)



# Why USB 2.0 is still relevant?



- USB 3.x and USB 4.0 are not replacing USB 2.0  
Backwards compatibility is achieved by dual bus  
The upper layers are pretty much the same
- Every USB 3.x hub contains both USB 2.0 and USB 3.x hub inside
- USB 3.x and USB-C connectors contain dedicated USB 2.0 D+/D-  
All USB 2.0 rules apply on D+/D- signals
- There's a lot of devices that are fine with USB 2.0 speeds:
  - Keyboard
  - Mouse
  - Controllers





# USB Transfer Types



USB generalizes all possible transfers into 4 types:

- **Control**  
Used for handling commands, e.g. GET\_DESCRIPTOR  
Class and vendor commands possible, e.g. volume adjustment
- **Interrupt**  
Periodic, guaranteed latency, retry on errors, e.g.: keyboard, mouse
- **Isochronous**  
Periodic, guaranteed bandwidth, no retry or guarantee of delivery, e.g.: audio
- **Bulk**  
Transfer large data, retry on errors, e.g.: mass storage



# USB Classes

## the communication protocols



USB specific classes, e.g.:

- Hub
- Human Interface Device (HID)

Protocol wrappers, e.g.:

- Mass Storage Class (MSC)
- Communications Device Class (CDC)
- Printer

Vendor specific:

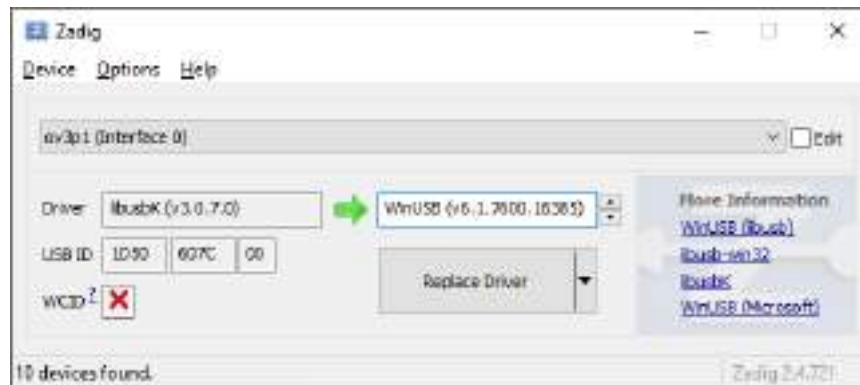
- Quite a few 😊



# libusb – a cross-platform library to access USB devices



- Works out-of-the-box on Linux, on Windows needs generic USB driver



- Automated WinUSB installation is possible if device implements WCID
  - String descriptor 0xEE with "MSFT100" + Vendor Code (1 byte) + Padding (0x00)
  - Two vendor feature descriptors: Microsoft Compatible ID and Microsoft Extended Properties



# USB traffic capture



USB traffic can be captured in software:

- On Linux using usbmon module
- On Windows using USBPcap

There are Open Source hardware USB 2.0 sniffers available:

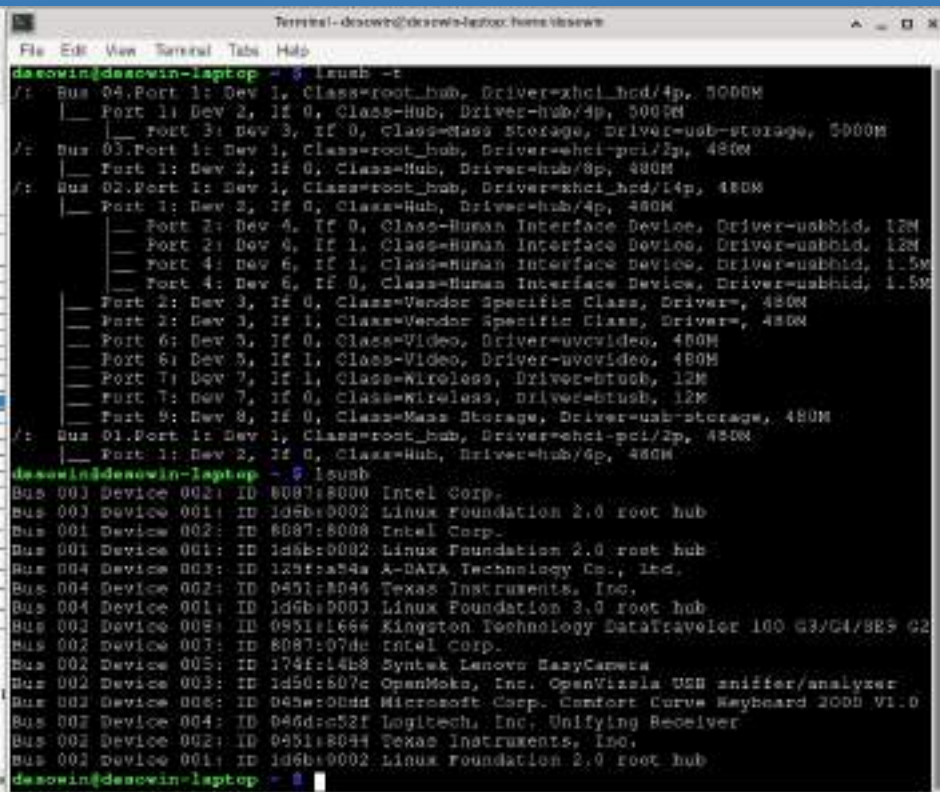
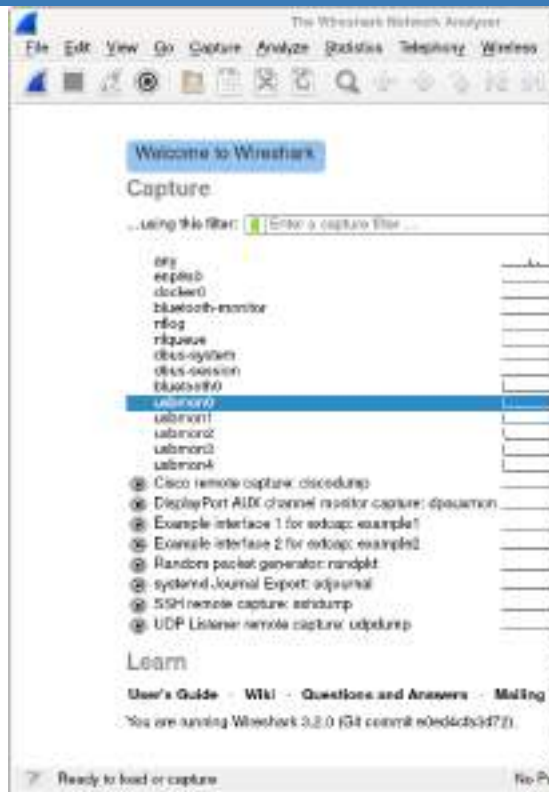
- OpenVizsla
- LambdaConcept USB2Sniffer

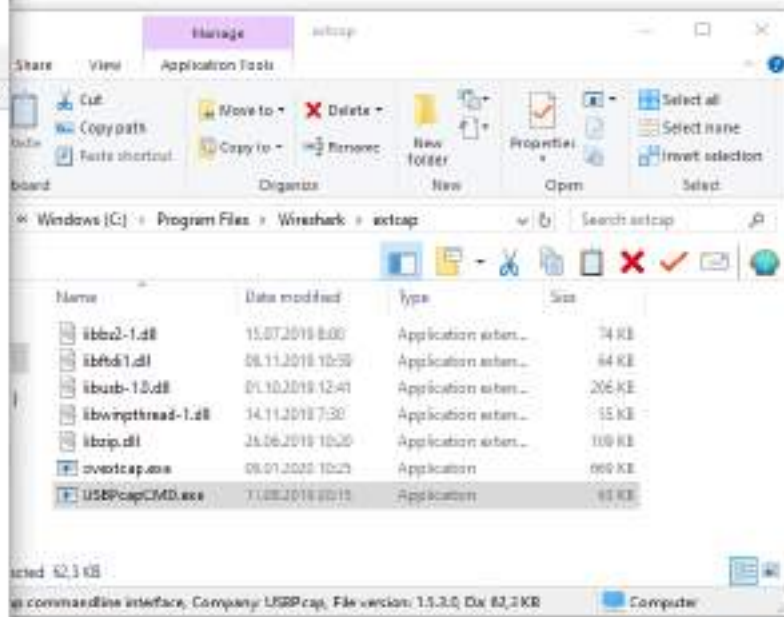
Sigrok can decode Low and Full speed signaling (capture with logic analyzer)

To my best knowledge, there are no Open Source USB 3.x hardware sniffers.



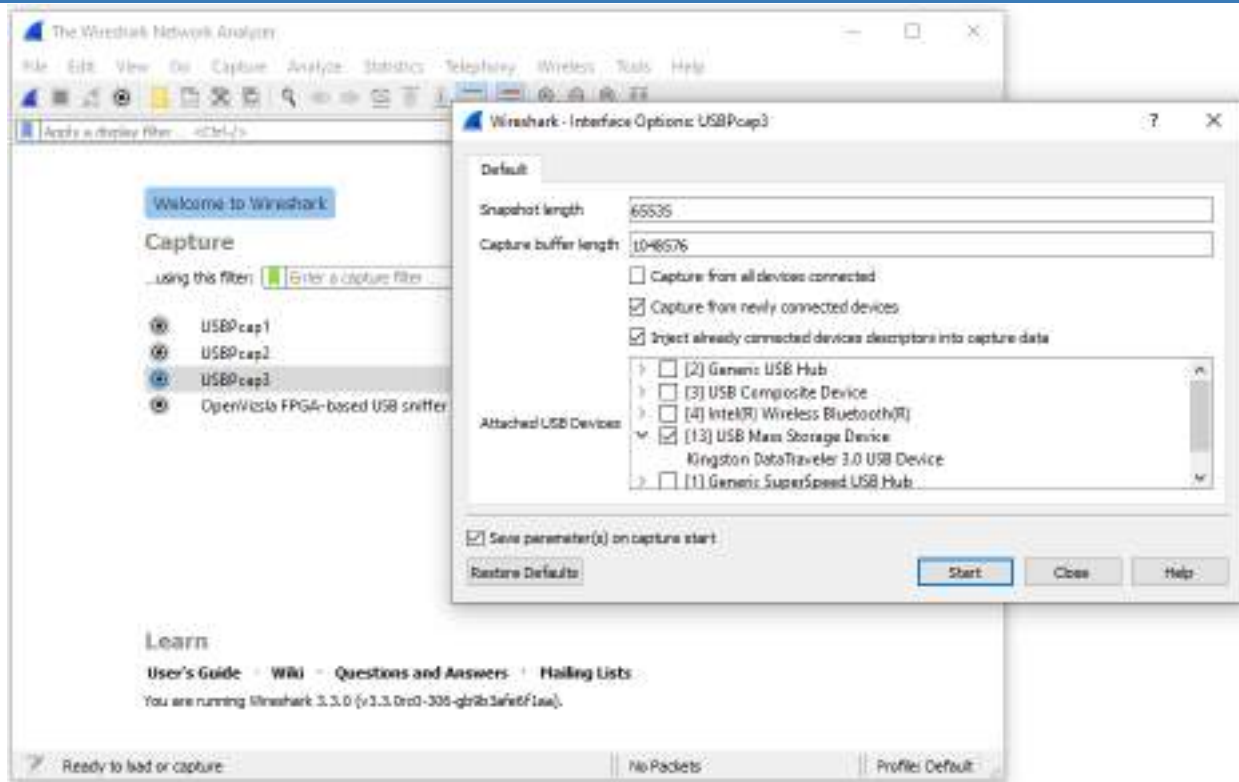
# sudo modprobe usbmon







# USBPcap extcap options





# OpenVizsla extcap options



OpenVizsla PCBA photo from [sysmocom webshop](https://www.sysmocom.com/).





# USB „Packets“



Wireshark shows what the capture engine provided, e.g.:

- libpcap (usbmon) provides "USB packets with Linux header and padding"
- USBPcap provides "USB packets with USBPcap header"
- OpenVizsla provides "USB 2.0/1.1/1.0 packets"

The "USB 2.0/1.1/1.0 packets" are described in USB 2.0 Specification, Chapter 8.

Software sniffers capture USB Request Blocks submitted to the host controller driver.

The "Linux header" and "USBPcap header" contain OS specific URB information.



# Device Descriptor Request



Wireshark

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	host	3.13.0	USB	36	GET_DESCRIPTOR Request DEVICE
2	0.000284	3.13.0	host	USB	46	GET_DESCRIPTOR Response DEVICE
3	0.000530	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
4	0.025938	3.13.0	host	USB	37	GET_DESCRIPTOR Response CONFIGURATION
5	0.030188	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
6	0.030218	5.13.0	host	USB	68	GET_DESCRIPTOR Response CONFIGURATION
7	0.030258	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
8	0.040155	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING[Malformed Packet]
9	0.040226	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
10	0.041380	3.13.0	host	USB	32	GET_DESCRIPTOR Response STRING
11	0.041641	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING

Frame 1: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on Interface 0  
USB USB

Setup Data

- bRequestType: 0x80
  - 1... .... = Direction: Device-to-host
  - ..00. .... = Type: Standard (0x0)
  - ...0 0000 = Recipient: Device (0x00)
- bRequest: GET\_DESCRIPTOR (6)
- Descriptor Index: 0x00
- bDescriptorType: DEVICE (0x01)
- Language Id: no language specified (0x0000)
- wLength: 12

0000 1c 02 f0 8a b6 20 05 0d ff ff 00 00 00 00 0b 00  
0016 00 03 00 0d 00 00 02 00 00 00 00 00 00 00 01  
0028 00 00 12 00

Text item (text), 8 bytes

Packets: 452 · Displayed: 452 (100.0%)

Profile: Default



# Device Descriptor Response



Wireshark

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	host	3.13.0	USB	36	GET_DESCRIPTOR Request DEVICE
2	0.000204	3.13.0	host	USB	46	GET_DESCRIPTOR Response DEVICE
3	0.000530	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
4	0.025938	3.13.0	host	USB	37	GET_DESCRIPTOR Response CONFIGURATION
5	0.030188	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
6	0.030210	3.13.0	host	USB	60	GET_DESCRIPTOR Response CONFIGURATION
7	0.030500	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION

Frame 2: 46 bytes on wire (368 bits), 46 bytes captured (368 bits) on interface 0  
USB USB

DEVICE\_DESCRIPTOR

- bLength: 18
- bDescriptorType: 0x01 (DEVICE)
- bcdUSB: 0x0210
- bDeviceClass: Device (0x00)
- bDeviceSubClass: 0
- bDeviceProtocol: 0 (Use class code info from Interface Descriptors)
- bMaxPacketSize0: 64
- IdVendor: Kingston Technology (0x0951)
- IdProduct: DataTraveler 100 G3/G4/SE9 G2 (0x1666)
- bcdDevice: 0x0001
- iManufacturer: 1
- iProduct: 2
- iSerialNumber: 3
- bNumConfigurations: 1

Text item (text), 18 bytes

Packets: 452 - Displayed: 452 (100.0%) Profile: Default



# Configuration Descriptor Request

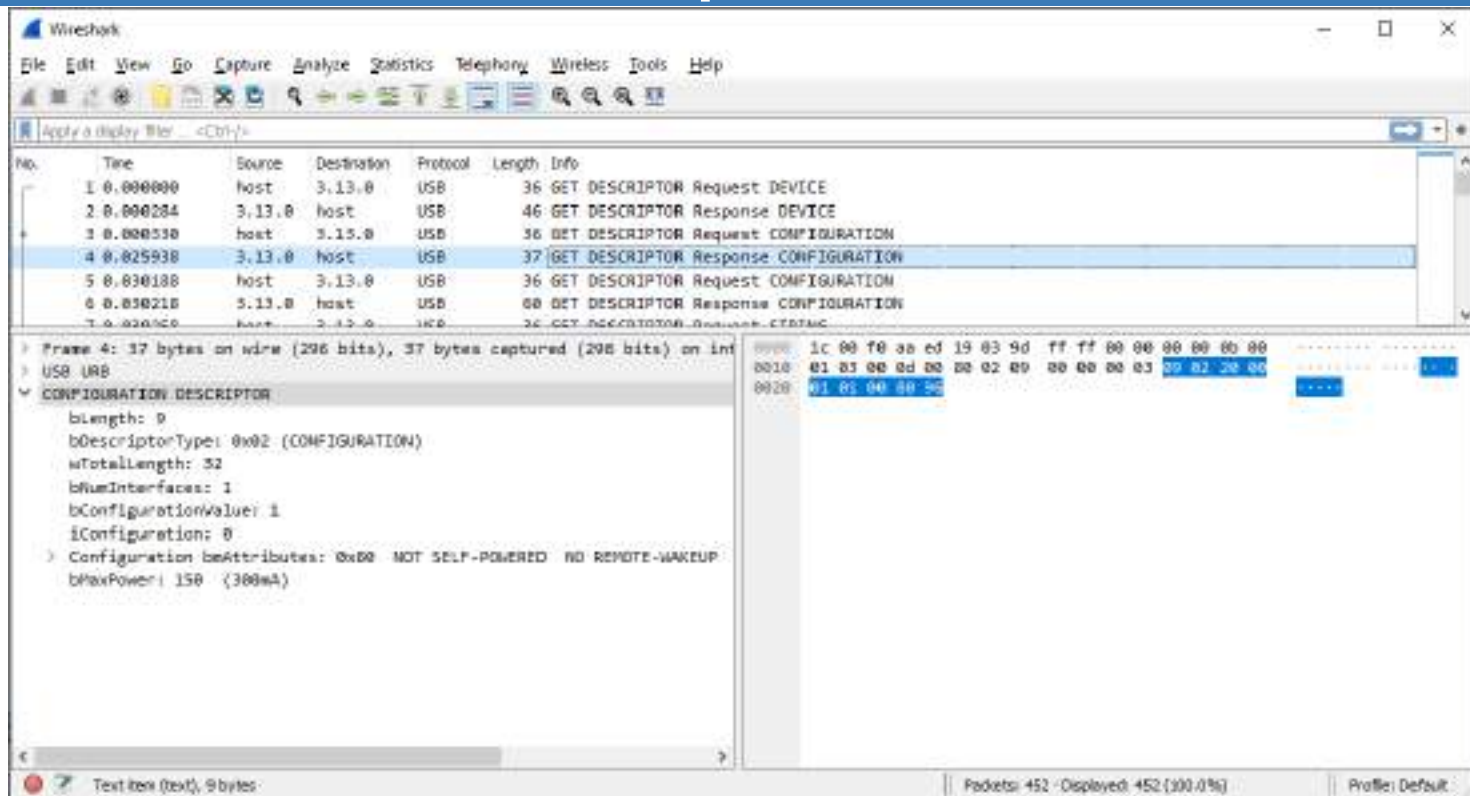


The image shows a Wireshark network packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. The main display area shows a list of captured packets. The selected packet is number 3, which is a GET\_DESCRIPTOR Request for the CONFIGURATION descriptor. The packet details pane on the left shows the setup data for this request, including the direction (Device-to-host), type (Standard), recipient (Device), request type (GET\_DESCRIPTOR), descriptor index (0x00), descriptor type (CONFIGURATION), language ID (no language specified), and length (9). The packet bytes pane on the right shows the raw data of the packet, with the first 9 bytes highlighted in blue.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	host	3.13.0	USB	36	GET_DESCRIPTOR Request DEVICE
2	0.000284	3.13.0	host	USB	46	GET_DESCRIPTOR Response DEVICE
3	0.000330	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
4	0.025938	3.13.0	host	USB	37	GET_DESCRIPTOR Response CONFIGURATION
5	0.030188	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
6	0.030210	3.13.0	host	USB	68	GET_DESCRIPTOR Response CONFIGURATION
7	0.030250	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION

Frame 3: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface  
USB URB  
Setup Data  
bRequestType: 0x00  
1... .. = Direction: Device-to-host  
.00. .... = Type: Standard (0x00)  
...0 0000 = Recipient: Device (0x00)  
bRequest: GET\_DESCRIPTOR (6)  
Descriptor Index: 0x00  
bDescriptorType: CONFIGURATION (0x02)  
Language Id: no language specified (0x0000)  
wLength: 9

1c 00 f0 3a ed 19 03 9d ff ff 70 00 00 00 00 00  
00 00 00 00 00 00 02 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





# Configuration Descriptor Request



The image shows a Wireshark network packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. The main display area shows a list of captured packets. The selected packet is number 5, a GET\_DESCRIPTOR Request CONFIGURATION, with a length of 36 bytes. The packet details pane on the left shows the USB URB and Setup Data. The Setup Data section includes fields for bmRequestType (0x00), bRequest (GET\_DESCRIPTOR (6)), Descriptor Index (0x00), bDescriptorType (CONFIGURATION (0x02)), Language Id (no language specified (0x0000)), and wLength (32). The packet bytes pane on the right shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	host	3.13.0	USB	36	GET_DESCRIPTOR Request DEVICE
2	0.000284	3.13.0	host	USB	46	GET_DESCRIPTOR Response DEVICE
3	0.000330	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
4	0.025938	3.13.0	host	USB	37	GET_DESCRIPTOR Response CONFIGURATION
5	0.030188	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION
6	0.030210	3.13.0	host	USB	68	GET_DESCRIPTOR Response CONFIGURATION
7	0.030300	host	3.13.0	USB	36	GET_DESCRIPTOR Request CONFIGURATION

Frame 5: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface 0  
USB URB  
Setup Data  
bmRequestType: 0x00  
1... .. = Direction: Device-to-host  
.00. .... = Type: Standard (0x00)  
...0 0000 = Recipient: Device (0x00)  
bRequest: GET\_DESCRIPTOR (6)  
Descriptor Index: 0x00  
bDescriptorType: CONFIGURATION (0x02)  
Language Id: no language specified (0x0000)  
wLength: 32

0000 1c 00 f0 3a ed 19 03 9d ff ff 00 00 00 00 00 00  
0010 00 03 00 0d 00 00 02 00 00 00 00 00 00 00 02  
0020 00 00 20 00

Text box (text), 8 bytes  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default



# Configuration Descriptor Response



The image shows a Wireshark packet capture of a USB transaction. The packet list pane shows a packet at time 0.000218 from source 3,13,0 to host, protocol USB, length 68, labeled "GET\_DESCRIPTOR Response: CONFIGURATION".

The packet details pane shows the following structure:

- CONFIGURATION\_DESCRIPTOR**
  - bLength: 9
  - bDescriptorType: 0x02 (CONFIGURATION)
  - wTotalLength: 32
  - bNumInterfaces: 1
  - bConfigurationValue: 1
  - iConfiguration: 0
  - Configuration bAttributes: 0x00 NOT SELF-POWERED NO REMOTE-WAKEUP
    - 1... .. = Must be 1: Must be 1 for USB 1.1 and higher
    - .0... .. = Self-Powered: This device is powered from the USB bus
    - ..0... .. = Remote Wakeup: This device does NOT support remote wakeup
  - bMaxPower: 150 (300mA)
- INTERFACE\_DESCRIPTOR (0.0): class Mass Storage**
  - bLength: 9
  - bDescriptorType: 0x04 (INTERFACE)
  - bInterfaceNumber: 0
  - bAlternateSetting: 0
  - bNumEndpoints: 2
  - bInterfaceClass: Mass Storage (0x08)
  - bInterfaceSubClass: SCSI transparent command set (0x06)
  - bInterfaceProtocol: Bulk-Only (BBB) Transport (0x50)
  - iInterface: 0
- ENDPOINT\_DESCRIPTOR**

The packet bytes pane shows the raw data of the descriptor, with the first 9 bytes highlighted in blue:

```
0000 1c 00 f0 aa ed 19 03 9d ff ff 00 00 00 00 00 00
0010 01 05 00 0d 00 00 02 20 00 00 00 03 00 02 20 00
0020 01 01 00 00 96 09 04 00 00 02 05 05 50 90 07 05
0030 01 02 00 02 ff 07 05 02 02 00 02 ff
```

The status bar at the bottom indicates "Text item (text), 32 bytes" and "Packets: 452 - Displayed: 452 (100.0%)".





# Configuration Descriptor Response



The image shows a Wireshark packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains various icons for packet capture and analysis. The packet list pane shows a single packet at time 6.000218, source 3.13.0, destination host, protocol USB, and length 68. The packet details pane shows the configuration descriptor response, which includes two endpoint descriptors. The first endpoint descriptor is for endpoint 1, and the second is for endpoint 2. Both are bulk-transfer endpoints with a maximum packet size of 512 and an interval of 255. The packet bytes pane shows the raw data of the response, which is a 68-byte configuration descriptor.

No.	Time	Source	Destination	Protocol	Length	Info
6	0.000218	3.13.0	host	USB	68	GET_DESCRIPTOR Response: CONFIGURATION

▼ ENDPOINT DESCRIPTOR  
bLength: 7  
bDescriptorType: 0x05 (ENDPOINT)  
▼ bEndpointAddress: 0x01 IN Endpoint:1  
1... .. = Direction: IN Endpoint  
.... 0001 = Endpoint Number: 0x1  
▼ bAttributes: 0x02  
.... ..10 = TransferType: Bulk-Transfer (0x2)  
▼ wMaxPacketSize: 512  
.... ..10 0000 0000 = Maximum Packet Size: 512  
bInterval: 255  
▼ ENDPOINT DESCRIPTOR  
bLength: 7  
bDescriptorType: 0x05 (ENDPOINT)  
▼ bEndpointAddress: 0x02 OUT Endpoint:2  
0... .. = Direction: OUT Endpoint  
.... 0010 = Endpoint Number: 0x2  
▼ bAttributes: 0x02  
.... ..10 = TransferType: Bulk-Transfer (0x2)  
▼ wMaxPacketSize: 512  
.... ..10 0000 0000 = Maximum Packet Size: 512  
bInterval: 255

0000 1c 00 f0 aa ed 19 03 9d ff ff 00 00 00 00 00 00  
0010 01 05 00 0d 00 00 02 20 00 00 00 03 00 02 20 00  
0020 01 01 00 00 95 09 04 00 00 02 05 05 50 90 07 05  
0030 01 02 00 02 ff 07 05 02 02 00 02 ff

Text item (text), 32 bytes  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default





# String Descriptor 0 Request



The image shows a Wireshark capture of USB traffic. The packet list on the left shows a sequence of USB messages. Packet 7 is selected, showing a GET\_DESCRIPTOR Request for String Descriptor 0. The packet details pane on the right shows the USB protocol structure, including the bRequestType, bRequest, Descriptor Index, bDescriptorType, Language Id, and wLength. The packet bytes pane on the right shows the raw data of the packet.

No.	Time	Source	Destination	Protocol	Length	Info
7	0.030258	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
8	0.040135	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING [Malformed Packet]
9	0.040226	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
10	0.041300	3.13.0	host	USB	32	GET_DESCRIPTOR Response STRING
11	0.041641	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
12	0.043762	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING
13	0.044049	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
14	0.045772	3.13.0	host	USB	78	GET_DESCRIPTOR Response STRING
15	0.048368	host	3.13.0	USB	36	SET_CONFIGURATION Request
16	0.067722	3.13.0	host	USB	28	SET_CONFIGURATION Response
17	0.067979	host	3.13.0	USB	36	SET_INTERFACE Request

Frame 7: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface wi  
USB USB  
Setup Data  
bRequestType: 0x00  
1... .... = Direction: Device-to-host  
.00. .... = Type: Standard (0x0)  
...0 0000 = Recipient: Device (0x00)  
bRequest: GET\_DESCRIPTOR (0)  
Descriptor Index: 0x00  
bDescriptorType: STRING (0x03)  
Language Id: no language specified (0x0000)  
wLength: 2

0000 1c 00 f0 aa ed 19 03 9d ff ff 00 00 00 00 00 00  
0010 00 03 00 0d 00 00 02 00 00 00 00 00 30 96 00 03  
0020 50 00 01 00

Text box (text), 8 bytes

Packets: 452 - Displayed: 452 (100.0%)

Profile: Default



# String Descriptor 0 Response

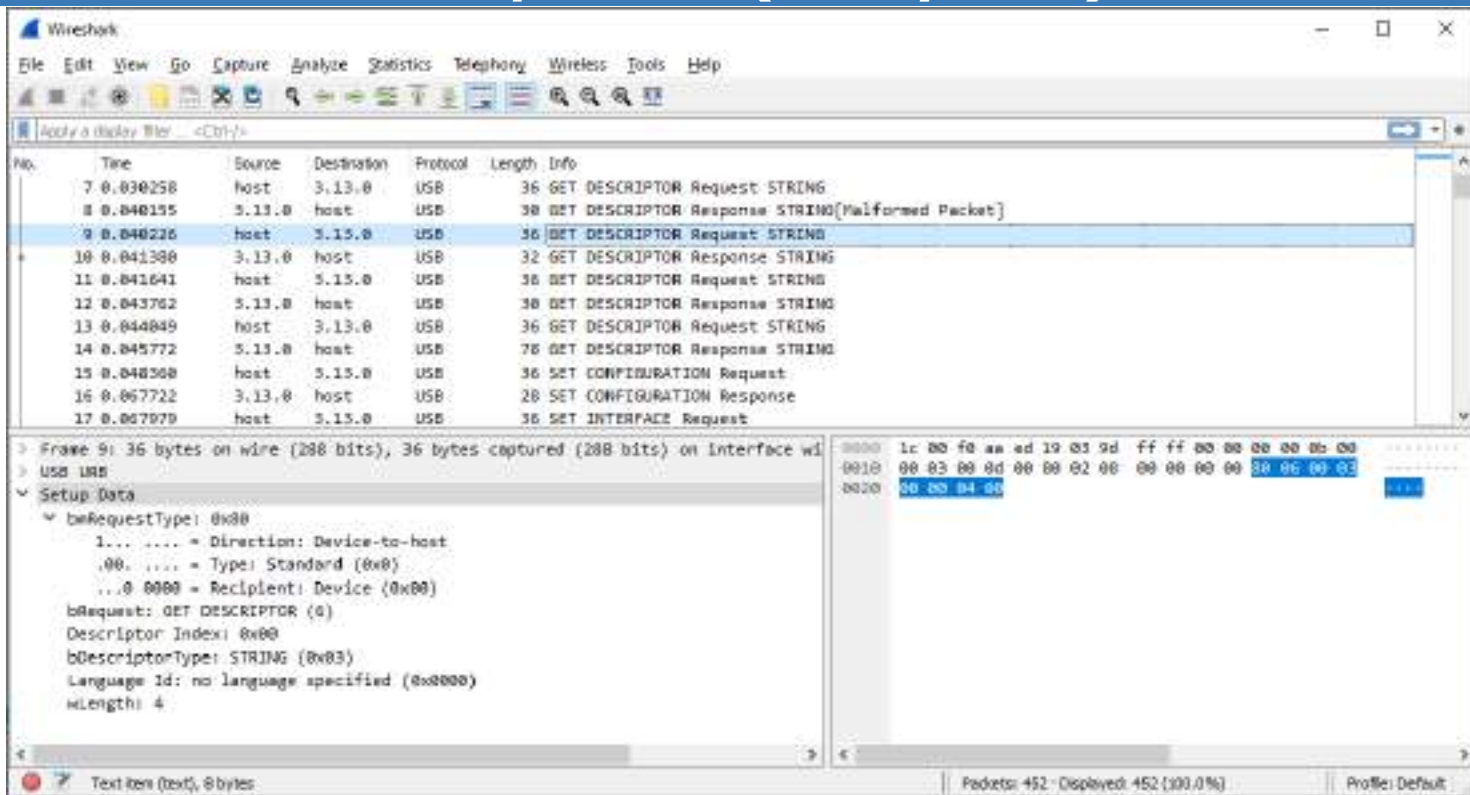


The image shows a Wireshark network packet capture window. The top pane displays a list of captured packets. Packet 8, at time 0.040155, is a USB GET\_DESCRIPTOR Response STRING (38 bytes) from host 3.13.0 to host 5.15.0, which is highlighted in blue and labeled as a 'Malformed Packet'. The bottom pane shows the details of this packet, indicating it is a STRING\_DESCRIPTOR with a length of 4 and a descriptor type of 0x03 (STRING). It is marked as '[Malformed Packet: USB]' and '[Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]'. The packet bytes are shown in the bottom right pane, with the first few bytes being 1c 00 fe aa ed 19 03 9d ff ff 00 00 00 00 05 00.

No.	Time	Source	Destination	Protocol	Length	Info
7	0.030258	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
8	0.040155	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING (Malformed Packet)
9	0.040226	host	3.15.0	USB	36	GET_DESCRIPTOR Request STRING
10	0.041300	3.13.0	host	USB	32	GET_DESCRIPTOR Response STRING
11	0.041641	host	5.15.0	USB	38	GET_DESCRIPTOR Request STRING
12	0.043762	5.13.0	host	USB	38	GET_DESCRIPTOR Response STRING
13	0.044049	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
14	0.045772	5.13.0	host	USB	78	GET_DESCRIPTOR Response STRING
15	0.048368	host	5.15.0	USB	36	SET_CONFIGURATION Request
16	0.067722	3.13.0	host	USB	28	SET_CONFIGURATION Response
17	0.067979	host	5.15.0	USB	36	SET_INTERFACE Request

Frame 8: 38 bytes on wire (240 bits), 38 bytes captured (240 bits) on interface wi  
USB USB  
STRING\_DESCRIPTOR  
length: 4  
bDescriptorType: 0x03 (STRING)  
[Malformed Packet: USB]  
[Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]  
[Malformed Packet (Exception occurred)]  
[Severity level: Error]  
[Group: Malformed]

Text item (text), 2 bytes  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default





# String Descriptor 0 Response (4 bytes)



The image shows a Wireshark network packet capture window. The packet list on the left shows a series of USB messages. Packet 10 is selected, which is a 'GET\_DESCRIPTOR Response STRING' of 32 bytes. The packet details pane on the right shows the structure of this response:

- Frame 10: 32 bytes on wire (256 bits), 32 bytes captured (256 bits) on interface w
- USB USB:
- STRING DESCRIPTOR
  - bLength: 4
  - bDescriptorType: 0x03 (STRING)
  - wLANGID: English (United States) (0x0409)

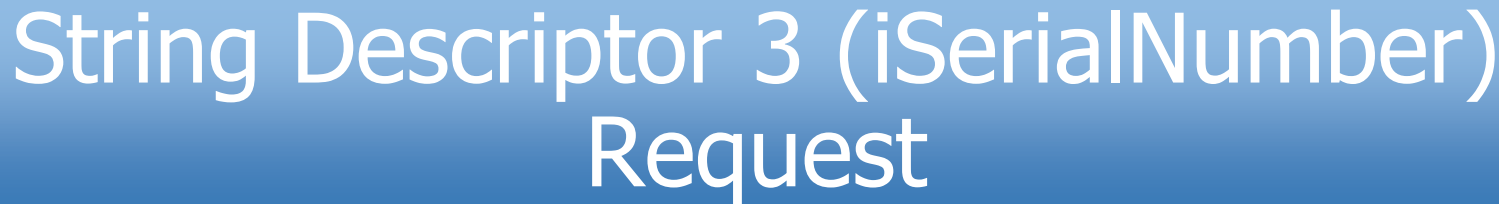
The packet bytes pane on the right shows the raw data in hexadecimal and ASCII. The first four bytes of the response are highlighted in blue, corresponding to the 'bLength' field value of 4.

No.	Time	Source	Destination	Protocol	Length	Info
7	0.030258	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
8	0.040195	3.13.0	host	USB	36	GET_DESCRIPTOR Response STRING [Malformed Packet]
9	0.040226	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
10	0.041300	3.13.0	host	USB	32	GET_DESCRIPTOR Response STRING
11	0.041641	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
12	0.043762	3.13.0	host	USB	36	GET_DESCRIPTOR Response STRING
13	0.044049	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
14	0.045772	3.13.0	host	USB	76	GET_DESCRIPTOR Response STRING
15	0.048368	host	3.13.0	USB	36	SET_CONFIGURATION Request
16	0.067722	3.13.0	host	USB	28	SET_CONFIGURATION Response
17	0.067979	host	3.13.0	USB	36	SET_INTERFACE Request

Text item (text), 4 bytes

Packets: 452 - Displayed: 452 (100.0%)

Profile: Default





# String Descriptor 3 (iSerialNumber) Response



Wireshark

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>/

No.	Time	Source	Destination	Protocol	Length	Info
7	0.030258	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
8	0.040155	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING[Malformed Packet]
9	0.040226	host	3.13.0	USB	38	GET_DESCRIPTOR Request STRING
10	0.041380	3.13.0	host	USB	32	GET_DESCRIPTOR Response STRING
11	0.041641	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
12	0.043762	3.13.0	host	USB	38	GET_DESCRIPTOR Response STRING
13	0.044049	host	3.13.0	USB	36	GET_DESCRIPTOR Request STRING
14	0.045772	3.13.0	host	USB	78	GET_DESCRIPTOR Response STRING
15	0.048368	host	3.13.0	USB	36	SET_CONFIGURATION Request
16	0.067722	3.13.0	host	USB	28	SET_CONFIGURATION Response
17	0.067878	host	3.13.0	USB	36	SET_INTERFACE Request

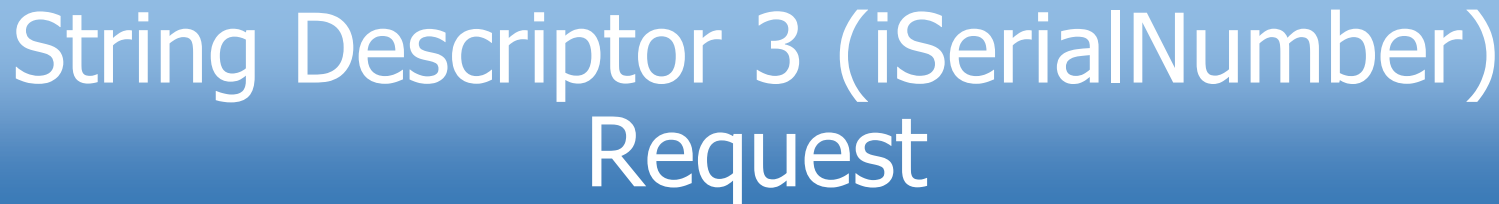
> Frame 12: 38 bytes on wire (240 bits), 38 bytes captured (240 bits) on interface 0  
> USB URB  
▼ STRING\_DESCRIPTOR  
  bLength: 50  
  bDescriptorType: 0x03 (STRING)  
  bString:

0000 1c 00 f0 aa ed 10 05 0d ff ff 00 00 00 00 00 00  
0010 01 03 00 01 00 00 02 02 00 00 00 03 02 03

Text box (text), 2 bytes

Packets: 452 - Displayed: 452 (100.0%) Profile: Default







# String Descriptor 3 (iSerialNumber) Response

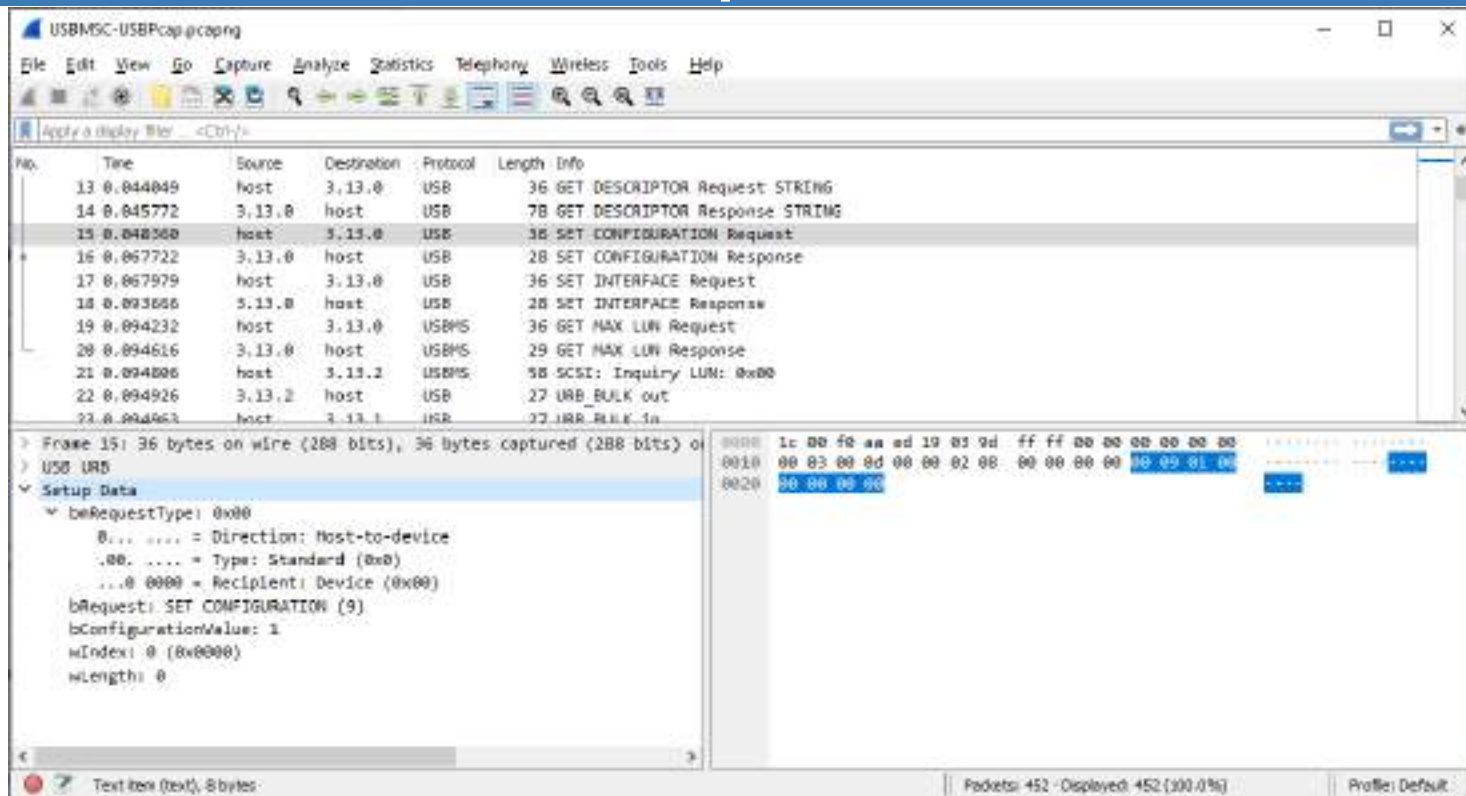


The image shows a Wireshark network traffic capture. The packet list on the left shows a series of USB messages between a host and a device (3.13.0). Packet 14 is selected, showing a 'GET\_DESCRIPTOR Response STRING' of 78 bytes. The packet details pane on the right shows the structure of this string descriptor:

- Frame 14: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on USB URD
- String Descriptor
  - bLength: 50
  - bDescriptorType: 0x03 (STRING)
  - bString: 68A44C42583CF268B90B67F

The packet bytes pane on the right shows the raw data in hexadecimal and ASCII. The ASCII column shows the string '2-6-4-4-2-5-8-0-C-F-2-0-0-8-9-0-F-8-7-F'.







# Set Configuration Response



The image shows a Wireshark network packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. The packet list pane shows three packets:

No.	Time	Source	Destination	Protocol	Length	Info
16	0.007722	3.13.0	host	USB	28	SET_CONFIGURATION Response
17	0.007979	host	3.13.0	USB	36	SET_INTERFACE Request
18	0.009366	3.13.0	host	USB	26	SET_INTERFACE Response

The packet details pane for the selected packet (Frame 16) shows the following structure:

- Frame 16: 28 bytes on wire (224 bits), 28 bytes captured (224 bits) on 1
- USB URB
  - [Source: 3.13.0]
  - [Destination: host]
  - USB/cap pseudoheader length: 28
  - IRP ID: 0xffff0d0319edaeaf8
  - IRP USB0\_STATUS: USB0\_STATUS\_SUCCESS (0x00000000)
  - URB Function: URB\_FUNCTION\_SELECT\_CONFIGURATION (0x0000)
  - IRP information: 0x01, Direction: PDO -> PDO
    - 0000 0000 = Reserved: 0x00
    - .... 1 = Direction: PDO -> PDO (0x1)
  - URB bus id: 3
  - Device address: 13
  - Endpoint: 0x00, Direction: OUT
    - 0... 0000 = Direction: OUT (0)
    - .... 0000 = Endpoint number: 0
  - URB transfer type: URB\_CONTROL (0x02)
  - Packet Data Length: 0
  - [Request in 15]
  - [Time from request: 0.019362000 seconds]
  - Control transfer stage: Complete (3)

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 1c 00 f0 0e ed 10 03 0d ff ff 00 00 00 00 00 00
0010 01 03 00 01 00 00 02 00 00 00 00 02
```

The status bar at the bottom indicates: USB (usb), 28 bytes | Packets: 452 - Displayed: 452 (100.0%) | Profile: Default



# Set Interface Request



The image shows a Wireshark network traffic capture. The top pane displays a list of captured packets. Packet 17, at time 0.007979, is a USB SET INTERFACE Request from host 3.13.0 to device 5.13.0. The bottom pane shows the details of this packet, including the Setup Data section with fields like bmRequestType, bRequest, and wInterface. The packet bytes pane on the right shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
16	0.007722	3.13.0	host	USB	28	SET CONFIGURATION Response
17	0.007979	host	5.13.0	USB	36	SET INTERFACE Request
18	0.008066	5.13.0	host	USB	28	SET INTERFACE Response
19	0.004232	host	3.13.0	USBMS	36	GET MAX LUN Request
20	0.004616	5.13.0	host	USBMS	20	SET MAX LUN Response
21	0.004888	host	5.13.2	USBMS	58	SCSI: Inquiry LUN: 0x00
22	0.004926	3.13.2	host	USB	27	URB_BULK out
23	0.004993	host	5.13.1	USB	27	URB_BULK in
24	0.005084	5.13.1	host	USBMS	63	SCSI: Data In LUN: 0x00 (Inquiry Response Data) [SCSI transfer limited due to allocation_
25	0.005117	host	3.13.1	USB	27	URB_BULK in
26	0.005191	5.13.1	host	USBMS	48	SCSI: Response LUN: 0x00 (Inquiry) (Good)

Frame 17: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface 0  
USB URB  
Setup Data  
bmRequestType: 0x00  
0... .. = Direction: Host-to-device  
..00. .... = Type: Standard (0x0)  
...0 0000 = Recipient: Device (0x00)  
bRequest: SET INTERFACE (11)  
bAlternateSetting: 0  
wInterface: 0  
wLength: 0

0000 1c 00 f0 aa ed 19 03 9d ff ff 00 00 00 00 01 00  
0010 00 05 00 0d 00 00 02 03 00 00 00 00 00 0b 00 00  
0020 00 00 00 00

Text item (text), 8 bytes  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default



# Set Interface Response



The image shows a Wireshark network packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. The packet list pane shows two packets:

No.	Time	Source	Destination	Protocol	Length	Info
18	0.093666	3.13.0	host	USB	28	SET_INTERFACE Response
19	0.094252	host	3.13.0	USBMS	36	GET_MAX_LUN Request

The packet details pane for packet 18 shows the following information:

- Frame 18: 28 bytes on wire (224 bits), 28 bytes captured (224 bits) on 1
- USB URB
  - [Source: 3.13.0]
  - [Destination: host]
  - USBcap pseudohdr Length: 28
  - IRP ID: 0xffff9d8319edaef0
  - IRP USB0 STATUS: USB0\_STATUS\_SUCCESS (0x00000000)
  - URB Function: URB\_FUNCTION\_SELECT\_INTERFACE (0x0001)
  - IRP information: 0x01, Direction: PDO -> FDO
    - 0000 000. = Reserved: 0x00
    - .... 001 = Direction: PDO -> FDO (0x1)
  - URB bus id: 3
  - Device address: 13
  - Endpoint: 0x00, Direction: OUT
    - 0... .. = Direction: OUT (0)
    - .... 0000 = Endpoint number: 0
  - URB transfer type: URB\_CONTROL (0x02)
  - Packet Data Length: 0
  - [Request id: 17]
  - [Time from request: 0.025687000 seconds]
  - Control transfer stage: Complete (3)

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 1c 00 70 a5 ed 19 83 9d ff ff 00 00 00 00 01 00 .....
0010 01 03 00 0d 00 00 02 02 00 02 00 00 00 00 00 00 .....
```

The status bar at the bottom indicates: USB (usb), 28 bytes | Packets: 452 - Displayed: 452 (100.0%) | Profile: Default



# MSC Class GET MAX LUN Request



The image shows a Wireshark network traffic capture. The main pane displays a list of captured packets. Packet 19 is selected, showing a USB GET MAX LUN Request from host 3.13.0 to device 3.13.0. The packet details pane on the left shows the structure of the USB request, including the request type (0x01), direction (Device-to-host), and the specific request (GET MAX LUN). The packet bytes pane on the right shows the raw data of the request, including the request type (0x01), direction (0x00), and the request value (0x0000).

No.	Time	Source	Destination	Protocol	Length	Info
18	0.093666	3.13.0	host	USB	28	SET INTERFACE Response
19	0.094232	host	3.13.0	USBMS	36	GET MAX LUN Request
20	0.094616	3.13.0	host	USBMS	29	GET MAX LUN Response
21	0.094886	host	3.13.2	USBMS	58	SCSI: Inquiry LUN: 0x00
22	0.094926	3.13.2	host	USB	27	URB_BULK out
23	0.094983	host	3.13.1	USB	27	URB_BULK in
24	0.095084	3.13.1	host	USBMS	63	SCSI: Data In LUN: 0x00 (Inquiry Response Data) [SCSI transfer limited due to allocation_
25	0.095117	host	3.13.1	USB	27	URB_BULK in

Frame 19: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on 1  
USB URB  
Setup Data  
bRequestType: 0x01  
1... .. = Direction: Device-to-host  
01. .... = Type: Class (0x01)  
...0 0001 = Recipient: Interface (0x01)  
USB Mass Storage  
bRequest: GET MAX LUN (0xfe)  
wValue: 0x0000  
wIndex: 0  
wLength: 1

1c 00 f0 84 5d 1d 03 0d ff ff 00 00 00 00 1b 00  
0010 00 03 00 0d 00 00 02 08 00 00 00 00 01 fe 00 00  
0020 00 00 01 00

Text item (text), 8 bytes  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default



# MSC Class GET MAX LUN Response



The image shows a Wireshark network packet capture window. The top pane displays a list of captured packets. The bottom pane shows the details of the selected packet (Frame 20), including the USB Mass Storage protocol and the raw data bytes.

No.	Time	Source	Destination	Protocol	Length	Info
18	0.093666	3.13.0	host	USB	28	SET INTERFACE Response
19	0.094232	host	5.15.0	USBMS	36	GET MAX LUN Request
20	0.094618	5.15.0	host	USBMS	29	GET MAX LUN Response
21	0.094886	host	3.13.2	USBMS	58	SCSI Inquiry LUN: 0x00
22	0.094926	5.15.2	host	USB	27	URB_BULK out
23	0.094983	host	5.15.1	USB	27	URB_BULK in
24	0.095084	3.13.1	host	USBMS	63	SCSI Data In LUN: 0x00 (Inquiry Response Data) [SCSI transfer limited due to allocation_
25	0.095117	host	5.15.1	USB	27	URB_BULK in

Frame 20: 29 bytes on wire (232 bits), 29 bytes captured (232 bits) on 1  
USB URB  
USB Mass Storage  
Max LUN: 0

Raw data (hex):  
1c 00 f0 84 5d 1d 03 0d ff ff 00 00 00 00 00 00  
0010 01 03 00 0d 00 00 02 01 00 00 00 03 00

USB Mass Storage (usbms), 1 byte  
Packets: 452 - Displayed: 452 (100.0%)  
Profile: Default



# SCSI command on Bulk OUT endpoint



The image shows a Wireshark packet capture of a SCSI command on a Bulk OUT endpoint. The packet list shows four packets: a SCSI Read(10) command (packet 215), a USB\_BULK out packet (packet 216), a USB\_BULK in packet (packet 217), and a SCSI Data In packet (packet 218). The packet details pane shows the structure of the SCSI command, including the LUN (0x00), the command set (Direct Access Device), the opcode (Read(10)), the logical block address (2048), and the transfer length (16). The packet bytes pane shows the raw data of the command, including the signature, tag, and data transfer length.

Wireshark

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter: <Ctrl>F

No.	Time	Source	Destination	Protocol	Length	Info
215	0.329042	host	3.13.2	USBMS	58	SCSI: Read(10) LUN: 0x00 (LBA: 0x00000000, Len: 16)
216	0.329110	3.13.2	host	USB	27	USB_BULK out
217	0.329128	host	3.13.1	USB	27	USB_BULK in
218	0.329602	3.13.1	host	USBMS	8219	SCSI: Data In LUN: 0x00 (Read(10) Response Data)

> Frame 215: 58 bytes on wire (464 bits), 56 bytes captured (464 bits) on  
> USB: USB

USB Mass Storage

- Signature: 0x43425355
- Tag: 0x20ad2a60
- DataTransferLength: 8192
- Flags: 0x00
  - .000 .... = Target: 0x0 (0)
  - ... 0000 = LUN: 0x0
  - ... 0 1010 = CDB Length: 0x0a

SCSI CDB: Read(10)

- [LUN: 0x0000]
- [Command Set: Direct Access Device (0x00) ]
- [Response in: 220]
- Opcode: Read(10) (0x28)
- Flags: 0x00
- Logical Block Address (LBA): 2048
  - ... 0 0000 = Group: 0x00
- Transfer Length: 16
- Control: 0x00

0000 1b 00 60 2a ad 20 03 9d ff ff 00 00 00 00 00 00  
0010 00 03 00 0d 00 02 03 1f 00 00 00 55 53 42 43 60  
0020 2a ad 20 00 20 00 00 00 00 0a 20 00 00 00 00 00  
0030 00 00 10 00 00 00 00 00 00 00

USB

USB Mass Storage (usbms), 31 bytes

Packets: 452 - Displayed: 452 (100.0%)

Profile: Default





# SCSI command on Bulk OUT endpoint



The image shows a Wireshark network packet capture. The top pane displays a list of packets. Packet 216 is selected, showing it is a USB\_BULK out packet from host 3.13.2 to host 3.13.1. The bottom pane shows the details of this packet, including the USB\_PCAP pseudoheader, IRP information, and the USB transfer type (USB\_BULK). The packet data is displayed in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
215	0.329042	host	3.13.2	USBMS	58	SCSI: Read(10) LUN: 0x00 (LBA: 0x00000000, Len: 16)
216	0.329110	3.13.2	host	USB	27	USB_BULK out
217	0.329128	host	3.13.1	USB	27	USB_BULK in
218	0.329602	3.13.1	host	USBMS	8219	SCSI: Data In LUN: 0x00 (Read(10) Response Data)

Frame 216: 27 bytes on wire (216 bits), 27 bytes captured (216 bits) on interface 0

USB\_BULK

- [Source: 3.13.2]
- [Destination: host]
- USB\_PCAP pseudoheader length: 27
- IRP ID: 0xfffff0d0120ed2a0
- IRP USB\_STATUS: USB\_STATUS\_SUCCESS (0x00000000)
- USB Function: USB\_FUNCTION\_BULK\_OR\_INTERRUPT\_TRANSFER (0x0000)
- IRP information: 0x01, Direction: PDO -> PDO
  - 0000 0000 = Reserved: 0x00
  - .... 0001 = Direction: PDO -> PDO (0x01)
- USB bus id: 3
- Device address: 13
- Endpoint: 0x02, Direction: OUT
- USB transfer type: USB\_BULK (0x03)
- Packet Data Length: 0
- [\[Request in: 215\]](#)
- [Time from request: 0.00000000 seconds]
- [bInterfaceClass: Mass Storage (0x08)]

USB (usb), 27 bytes

Packets: 452 - Displayed: 452 (100.0%)

Profile: Default





# What software sniffers show?



Device driver **submits URB**, HCD handles URB and **reports back** to Device driver.

All software sniffer “packets” contain OS specific metadata (URB ID, endpoint, ...)

	OUT (send to device)		IN (receive from device)	
	Host→Device	Device→Host	Host→Device	Device→Host
Control	SETUP Data (8 bytes) + Payload (if wLength > 0)	Indicates that URB handling is done (result code is OS specific)	SETUP Data (8 bytes)	Payload (if wLength > 0)*
Interrupt	Payload		Indicates that device driver requested host to start read attempts	Payload*
Bulk	Payload			Payload*
Isochronous	Payload			Payload*

\* Metadata only if the URB has failed/was cancelled, e.g. device was disconnected, STALL occurred, ...



# SCSI response on Bulk IN endpoint



The image shows a Wireshark network packet capture window. The top pane displays a list of captured packets. Packet 217, at time 0.329128, is a USB packet from host 3.13.1 to host 3.13.1, identified as a USB\_BULK in packet. The bottom pane shows the detailed view of this packet, which is a USB URB (USB Request Block) of 27 bytes. The URB information indicates it is a Bulk IN transfer from endpoint 0x81. The packet data is shown in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
215	0.329042	host	3.13.2	USBMS	58	SCSI: Read(10) LUN: 0x00 (LBA: 0x00000000, Len: 16)
216	0.329110	3.13.2	host	USB	27	USB_BULK out
217	0.329128	host	3.13.1	USB	27	USB_BULK in
218	0.329602	3.13.1	host	USBMS	8219	SCSI: Data In LUN: 0x00 (Read(10) Response Data)

Frame 217: 27 bytes on wire (216 bits), 27 bytes captured (216 bits) on interface 0

USB URB

- [Source: host]
- [Destination: 3.13.1]
- USBpcap pseudoheader length: 27
- IRP ID: 0xfffff0d0120ad2a00
- IRP USB\_STATUS: USB\_STATUS\_SUCCESS (0x00000000)
- URB Function: URB\_FUNCTION\_BULK\_OR\_INTERRUPT\_TRANSFER (0x0000)
- IRP information: 0x00, Direction: FDO -> PDO
  - 0000 0000 = Reserved: 0x00
  - .... 0000 = Direction: FDO -> PDO (0x0)
- URB bus id: 3
- Device address: 13
- Endpoint: 0x81, Direction: IN
- URB transfer type: URB\_BULK (0x03)
- Packet Data Length: 0
- [\[Response in: 218\]](#)
- [bInterfaceClass: Mass Storage (0x08)]

0000 1b 00 00 2a ad 20 03 0d ff ff 00 00 00 00 00 00  
0018 00 03 00 0d 00 81 03 00 00 00 00

USBMS: USBpcap.pcapng

Packets: 452 - Displayed: 452 (100.0%)

Profile: Default



# SCSI response on Bulk IN endpoint



The image shows a Wireshark network packet capture. The packet list on the left shows four packets. Packet 218, at time 0.329602, is a SCSI Data In LUN 0x00 (Read(10) Response Data) of length 8219 bytes. The packet details pane on the left shows the following structure:

- Frame 218: 8219 bytes on wire (65752 bits), 8219 bytes captured (65752) on interface 0
- USB URB
- USB Mass Storage
- SCSI Payload (Read(10) Response Data)
  - [LUN: 0x0000]
  - [Command Set: Direct Access Device (0x00)]
  - [SBC Opcode: Read(10) (0x20)]
  - [Request In: 223]
  - [Response In: 220]

The packet bytes pane on the right shows the raw data in hexadecimal and ASCII. The data starts with 00 10 01 03 00 0d 00 01 03 00 20 00 00 eb 76 90 45 50. The ASCII column shows the word 'FAT' in the first few lines of the data.

SCSI (scsi), 8,192 bytes

Packets: 452 - Displayed: 452 (100.0%)

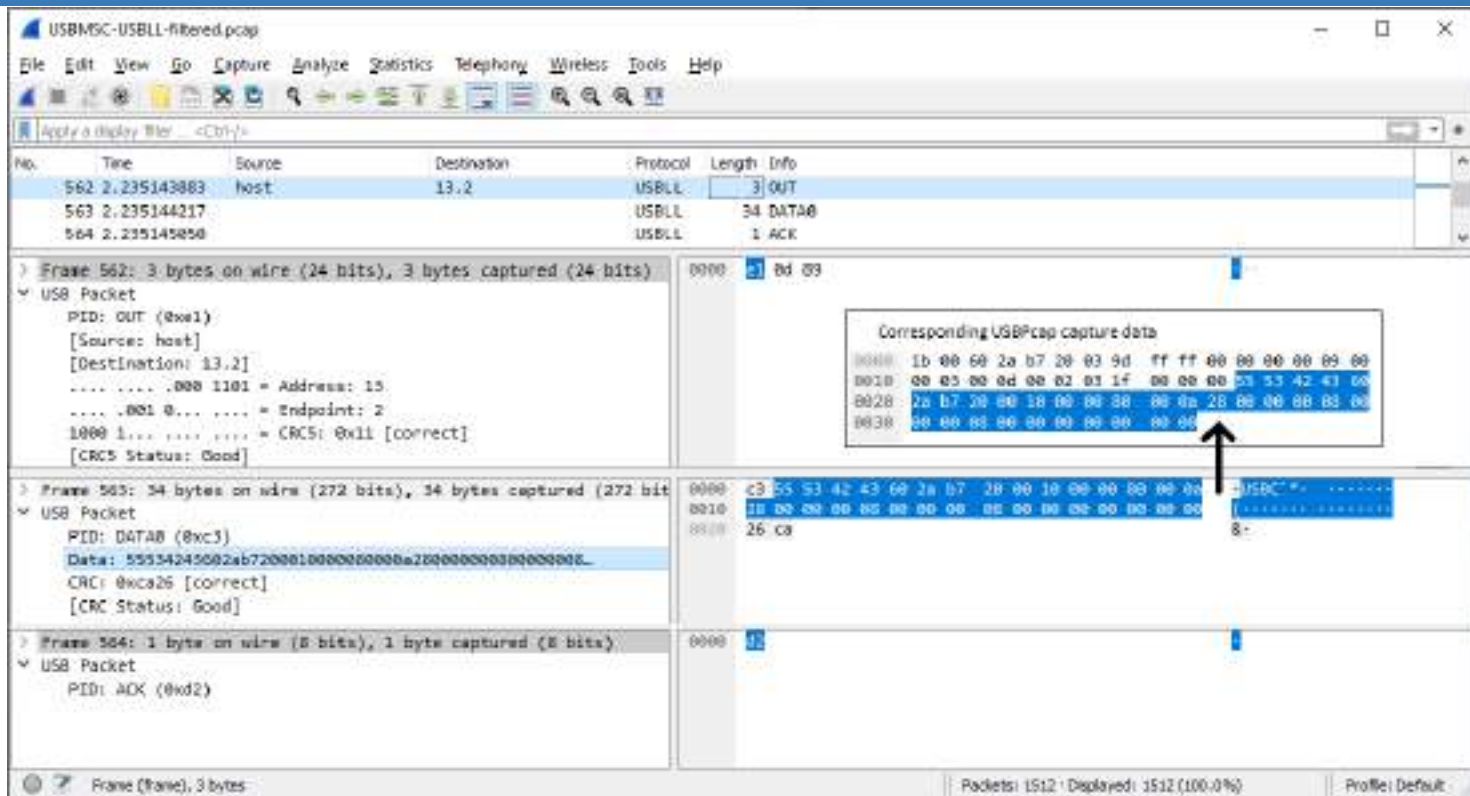
Profile: Default

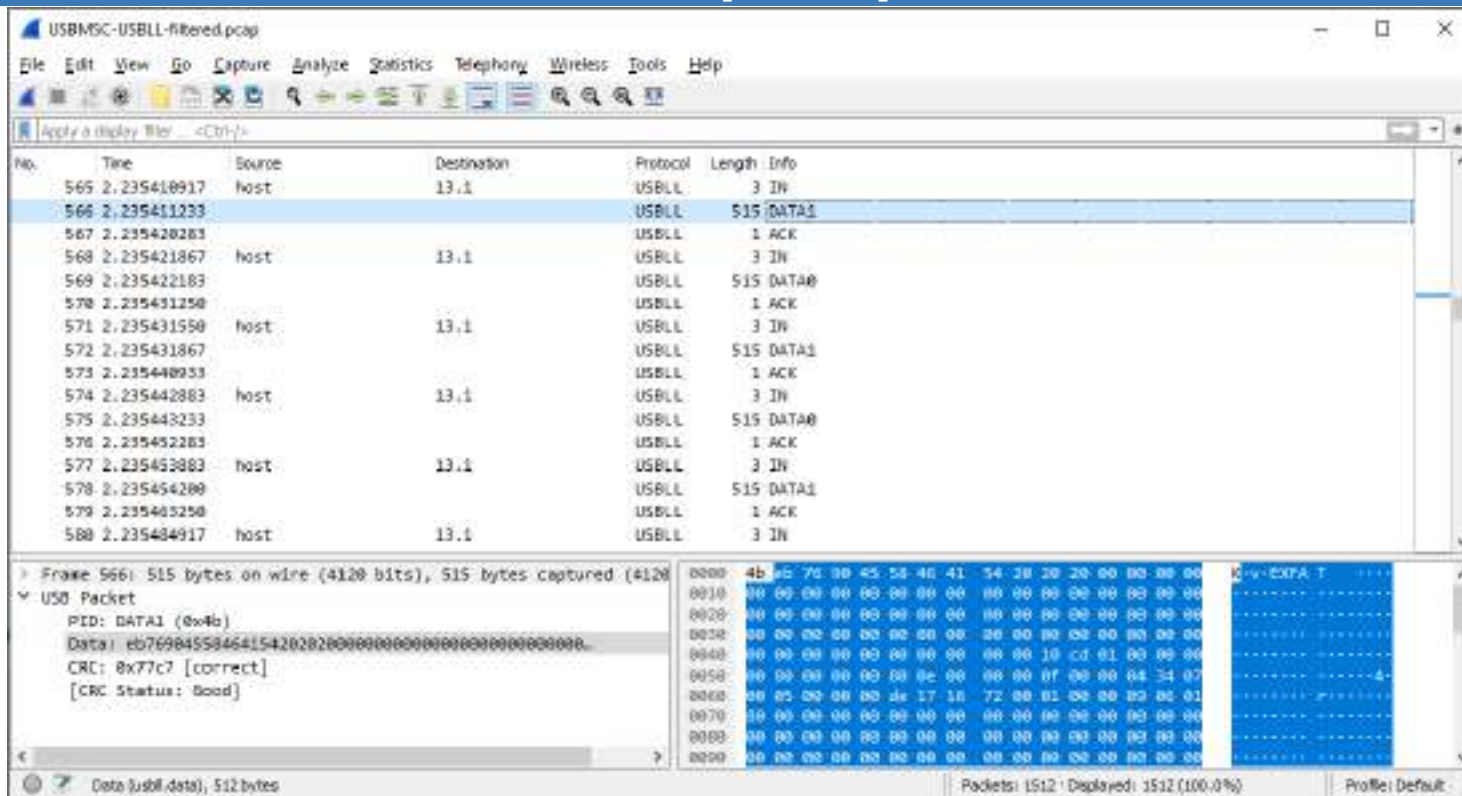


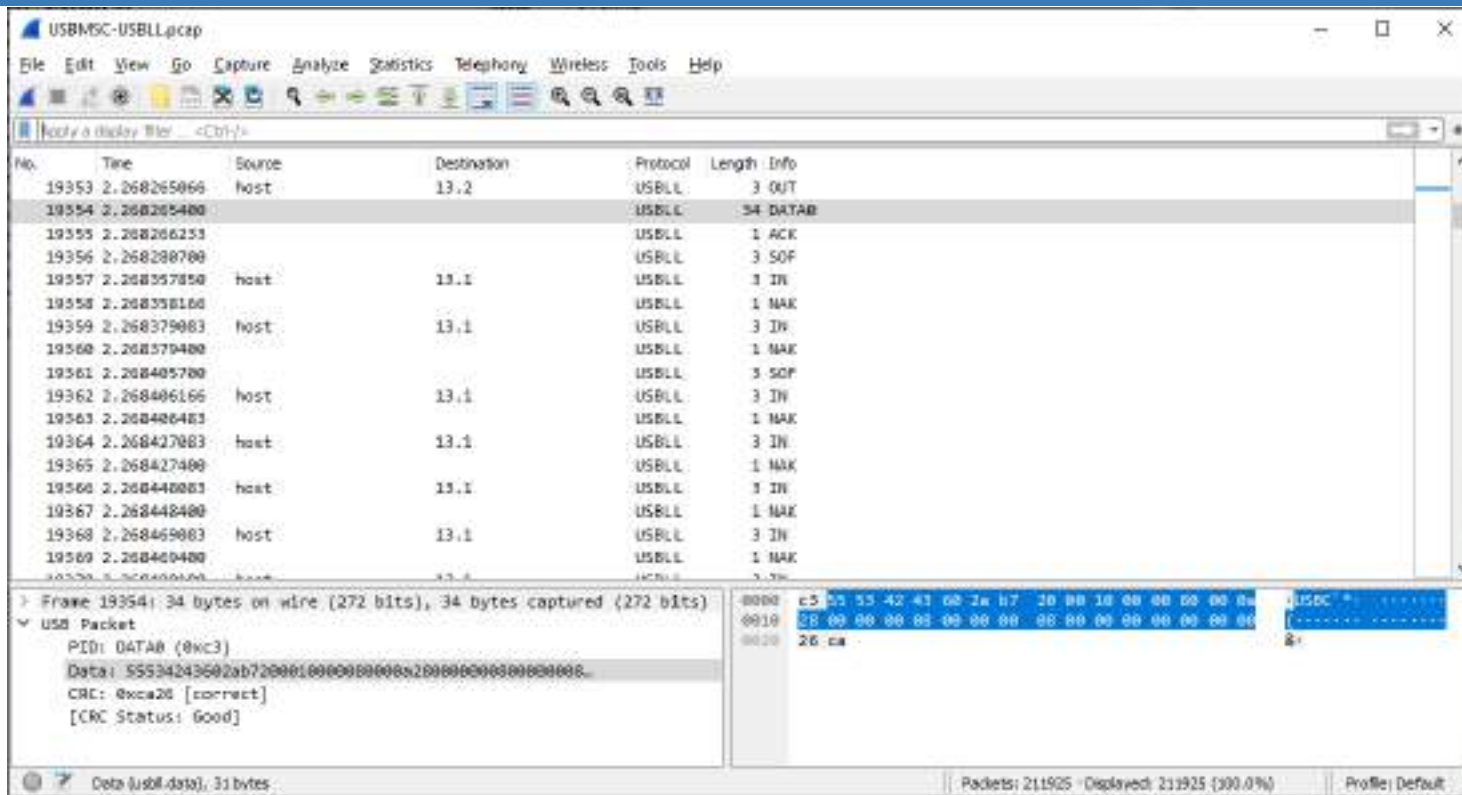
# What about Max Packet Size?



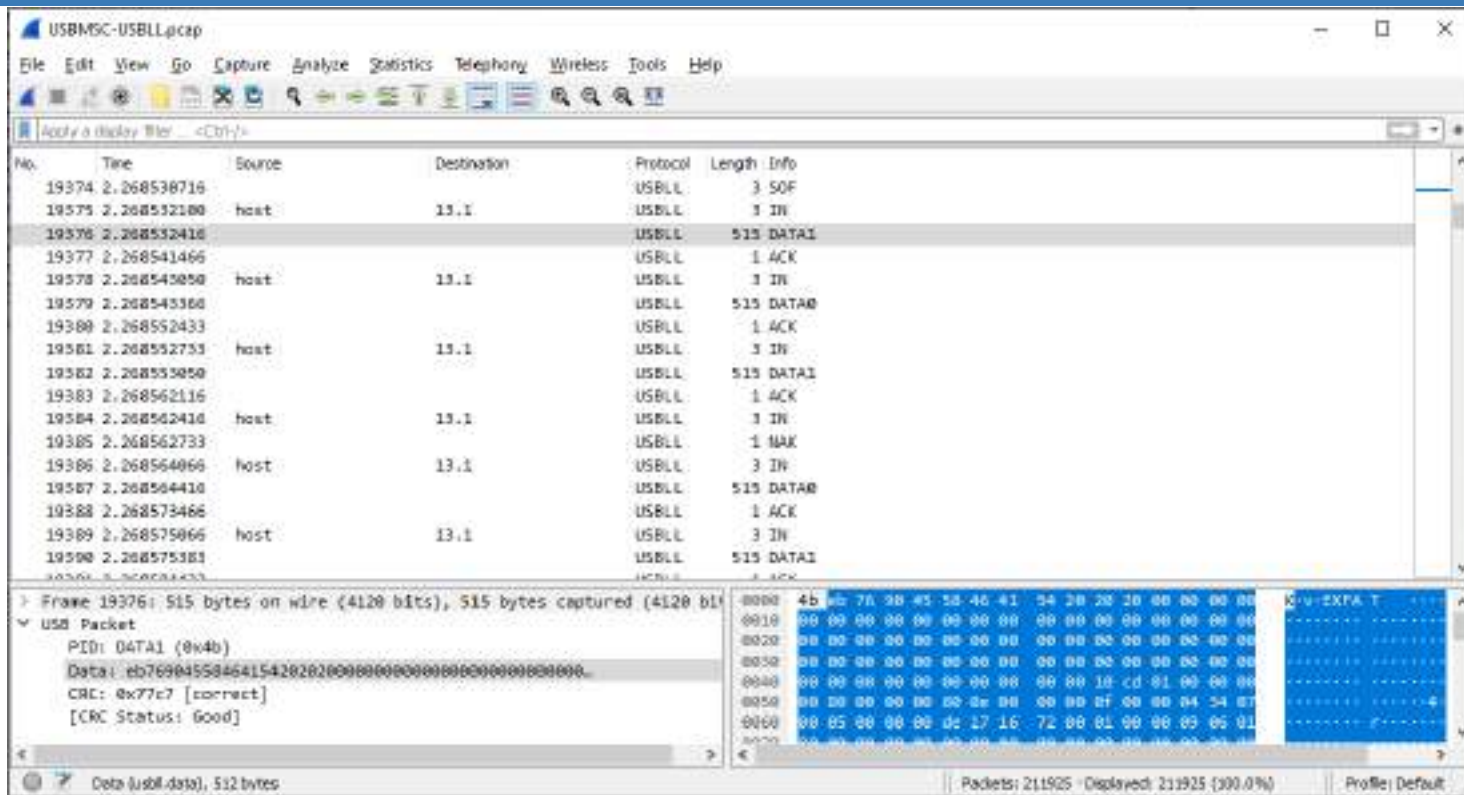
- Endpoint 1 wMaxPacketSize is 512 bytes
- SCSI Reponse in „USBPcap packet“ was 8192 bytes
- Software USB sniffers capture USB Request Blocks (URBs)
  - USB host converts URBs into USB packets
- To see USB packets we have to use hardware sniffer















# Summary



- USB 2.0 is still relevant today
  - USB 3.x backwards compatibility with USB 2.0 is achieved by dual bus
- Host initiates all communication
  - IN and OUT is always from Host perspective
  - Device cannot send data unless host asks for it (driver submits "IN" URB)
- Software sniffers capture URBs
  - Every URB is captured as 2 "URB packets"
    - Driver to HCI includes data payload from host to device (if any)
    - HCI to driver includes data payload from device to host (if any)
  - URB level capture is sufficient for general use
    - Understanding USB at packet level helps make sense out of the "URB packets"



# Q & A