



**MATEMATICKO-FYZIKÁLNÍ
FAKULTA**
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Peter Lakatoš

Analyzátor USB paketů

Katedra distribuovaných a spolehlivých systémů

Vedoucí bakalářské práce: Mgr. Pavel Ježek, Ph.D.

Studijní program: Informatika

Studijní obor: Programování a softwarové systémy

Praha 2021

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V dne

Podpis autora

Poděkování.

Název práce: Analyzátor USB paketů

Autor: Peter Lakatoš

Katedra: Katedra distribuovaných a spolehlivých systémů

Vedoucí bakalářské práce: Mgr. Pavel Ježek, Ph.D., Katedra distribuovaných a spolehlivých systémů

Abstrakt: USB zbernica je dnes jedným z najrozšírenejších spôsobov pripojenia periférií k počítaču. Cieľom práce bolo vytvoriť software, ktorý analyzuje zachytenú komunikáciu medzi zariadením pripojeným na danú zbernicu a počítačom.

Aplikácia následne rozumným spôsobom vizuálne zobrazuje zanalyzované dáta. Počiatočná verzia sa špecificky zameriava na HID triedu zariadení a ponúka aj sémantický význam jej úzkej podmnožiny do ktorej patria myš, klávesnica a joystick. Pri vizuálnej reprezentácii dát sa práca inšpiruje rôznymi dostupnými softwarmi, pričom rozlične kombinuje resp. dopĺňa ich vlastnosti a implementuje z nich tie, ktoré vníma ako najlepšie riešenie v danej situácii.

Ďalšie vlastnosti aplikácie sú napríklad parsovanie HID Report Descriptoru vďaka ktorému je jednoduchšie pridať sémantickú analýzu rôznym ďalším HID zariadeniam. Celkový návrh aplikácie by mal ponúknuť možnosť budúcej implementácie ďalších USB tried pre prípadné rozšírenie.

Klíčová slova: USB HID

Title: USB Packet Analyzer

Author: Peter Lakatoš

Department: Department of Distributed and Dependable Systems

Supervisor: Mgr. Pavel Ježek, Ph.D., Department of Distributed and Dependable Systems

Abstract: Abstract.

Keywords: key words

Obsah

1	Úvod	3
1.1	USB	3
1.2	Existujúce aplikácie	3
1.3	Požadované funkcie	6
1.4	Ciele práce	6
2	USB a Windows	7
2.1	USB zbernica	7
2.2	Device object a device stack	7
2.2.1	Drivery	7
2.3	Komunikacia s USB zariadením	7
2.4	USB descriptor	7
2.4.1	Rozloženie USB zariadenia z hľadiska descriptorov	7
2.5	HID zariadenia	8
2.5.1	Reporty	8
2.5.2	Report Descriptor	8
3	Analýza	9
3.1	Získanie USB packetov	9
3.1.1	Windows exclusive mód	9
3.1.2	Známe knižnice	9
3.1.3	Third-party aplikácie	9
3.2	Spracovávanie pcap súborov	9
3.3	Sémantická reprezentácia dát	9
3.4	Voľba frameworku	9
3.5	Zobrazenie základných informácií	9
3.6	Zobrazenie sémantického významu dát	10
3.7	Hexdump	10
4	Vývojová dokumentácia	11
4.1	Architektúra aplikácie	11
4.2	Jadro aplikácie	11
4.2.1	USB_Packet_Analyzer	11
4.2.2	Item Manager	11
4.2.3	DataViewer	11
4.2.4	TreeItem	11
4.3	Modely	11
4.3.1	AdditionaldataModel	11
4.3.2	ColorMapModel	11
4.3.3	DataViewerModel	11
4.3.4	TreeItemBaseModel	11
4.3.5	USBPcapHeaderModel	11
4.4	Interpretery	12
4.4.1	BaseInterpreter	12
4.4.2	Interpreter factory	12

4.4.3	Interpretery descriptorov	12
4.4.4	Interrupt transfer interpretery	12
4.5	Delegáti	12
4.6	HID	12
4.6.1	HIDDevices	12
4.7	Práca so súbormi	12
4.7.1	FileReader	12
4.8	Globálne dáta	13
4.8.1	ConstDataHolder	13
4.8.2	PacketExternStructs	13
5	Možnosti rozšírenia	14
5.1	Ukladanie výstupu do súboru	14
5.2	Iná vizuálna reprezentácia dát	14
5.3	Pridávanie nových interpreterov pre descriptorov	14
5.4	Pridanie interpreteru na interrupt transfer	14
5.4.1	Pridanie nových HID zariadení	14
5.5	Pridanie analýzy pre isochronous a bulk transfer	14
5.6	?Možnosť rozšírenia na iné platformy?	14
6	Užívateľská dokumentácia	15
6.1	Inštalácia	15
6.2	Orientácia v GUI aplikácie	15
6.3	Používanie aplikácie	15
7	Záver	16
7.1	Zhrnutie	16
7.2	Budúce plány	16
	Seznam obrázků	17
	Seznam tabulek	18
	Seznam použitých zkratk	19
	Přílohy	20
.1	První příloha	21

1. Úvod

Možnosti pripojenia rôznych periférií k zariadeniu sú v dnešnej dobe rozsiahle. Aj napriek tomu, že technológia každým dňom napreduje a svet sa uberať viac bezdrôtovým smerom, je USB stále najrozšírenejší sériový spôsob prenášania dát. Už z názvu „Universal Serial Bus“ je jasné, že rozpätie zariadení, ktoré možno k tejto zbernici pripojiť je obrovské. Práve preto je USB protokol jeden z najkomplexnejších protokolov ktoré sa využívajú na komunikáciu.

V tejto práci sa pozrieme na užšiu podmnožinu USB protokolu a na komunikáciu s dopredu vybratými perifériami, konkrétne myšou, klávesnicou a joystickom.

1.1 USB

vysvetlenie základných pojmov spojených USB: historia, usb port/conector, plug and play(<https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/introduction-to-plug-and-play>), host-master, low/full/high speed zariadenia

1.2 Existujúce aplikácie

Momentálne existuje niekoľko známych aplikácií ktoré slúžia na analýzu USB paketov. V tejto kapitole si ich zopár ukážeme, pričom mnohé z nich nám poslúžili ako inšpirácia pri písaní našej práce a riešení konkrétnych problémov na ktoré sa pozrieme v nasledujúcich kapitolách.

Je nutné upozorniť, že väčšina dnešných analyzátorov sú platené aplikácie, prípadne majú odomknuté len základné vlastnosti s možnosťou dokúpenia si plnej verzie. Práve preto pri ich prípadnom porovnávaní budeme brať do úvahy len funkcionality, ktorá je dostupná zadarmo.

Wireshark

Pravdepodobne najznámejšia third-party aplikácia na analýzu paketov. Jeho funkcionality je veľmi rozsiahla, a vzhľadom na to, že sa jedná o open-source projekt, neustále rastie. Zameriava sa hlavne na analýzu sieťových paketov. Napriek tomu podporuje spoluprácu s rôznymi inými sniffermi. Jeden z takýchto sniffrov je USBPcap ktorý zachytáva USB komunikáciu, a tak nie je prekvapivé, že Wireshark podporuje analýzu paketov aj nad touto zbernicou.

Vzhľadom na obľúbenosť a rozsiahlosť programu, nám Wireshark poslúžil ako referenčná aplikácia, z ktorej sme čerpali celkovú inšpiráciu na funkcie, ktoré by mal bežný analyzátor paketov spĺňať. Medzi tie úplne základné patrí napríklad hexdump dát nad ktorými prebieha analýza, ale napríklad aj spôsob vyobrazenia rôznych deskriptorov, ktoré je riešené cez stromovú štruktúru. Medzi viac špecifické funkcie ktoré sme neskôr implementovali aj v našom programe patrí napríklad detailnejšie vyobrazenie jednotlivých bytov a ich význam, ako je možné vidieť nižšie na obrázku 1.1. Túto vlastnosť aj napriek jej využitiu mnohé konkurenčné aplikácie postrádajú.


```

▼ ENDPOINT DESCRIPTOR
    bLength: 7
    bDescriptorType: 0x05 (ENDPOINT)
    ▼ bEndpointAddress: 0x81 IN Endpoint:1
        1... .... = Direction: IN Endpoint
        .... 0001 = Endpoint Number: 0x1
    ▼ bmAttributes: 0x03
        .... ..11 = Transfertype: Interrupt-Transfer (0x3)
    ▼ wMaxPacketSize: 4
        ...0 0... .... .... = Transactions per microframe: 1 (0)
        .... ..00 0000 0100 = Maximum Packet Size: 4
    bInterval: 10

```

Obrázek 1.1: Ukážka vyobrazenia jednotlivých bytov.

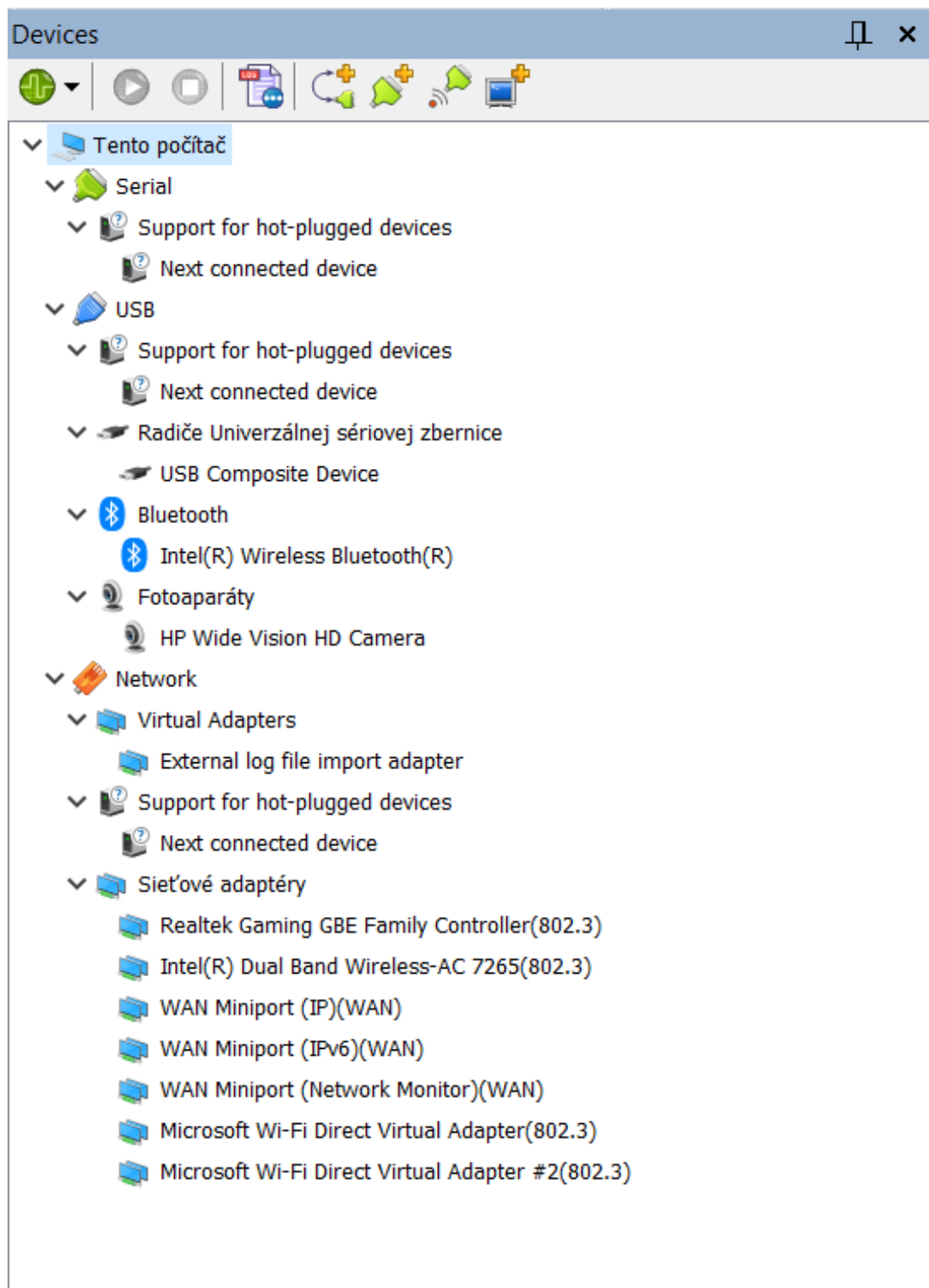
Jeho výhoda je hlavne v tom, že podporuje širokú škálu deskriptorov a plná verzia programu je dostupná úplne zadarmo. Z pohľadu užívateľa je až prekvapivé, že aj napriek rozsiahlosti programu je aplikácia veľmi user-friendly orientovaná a dopĺňa ju veľmi intuitívne užívateľské rozhranie.

Device Monitoring Studio

Plná verzia aplikácie je platená. Verzia zadarmo ponúka analýzu sieťových a USB paketov, tak ako aj analýzu komunikácie prebiehajúcej cez sériový port.

Ako prvé na aplikácii zaujme spôsob zvolenia si zariadenia s ktorým bude sledovaná komunikácia. Je implementovaný štýlom stromovej štruktúry ako je ukázané na obrázku 1.2 nižšie. Už základná verzia programu poskytuje veľmi rozsiahlu funkcionality. Používateľovi je umožnené poslať zariadeniu požiadavky definované v USB špecifikácii(ODKAZ). Aplikácia taktiež zobrazuje sémantickú analýzu niektorých HID zariadení, ale nie v práve najlepšej podobe.

Na základe výstupu sa dá povedať, že sémantická analýza je naimplementovaná skôr obecné a pri niektorých položkách je namiesto ich významu napísané „Unknown“. Taktiež nie je veľmi jasné odkiaľ sa dané hodnoty berú, keďže k nim chýba ich dátová reprezentácia. Medzi zachytenými paketami sa prvotne nezobrazujú tie, ktoré označujú nakonfigurovanie daného zariadenia (je nutné ho odpojiť a znova napojiť počas monitorovania). Užívateľské rozhranie je veľmi chaotické a chvíľu trvá, kým človek nájde čo i len základné informácie ako napríklad hlavičky ku jednotlivým paketom. Nepoteší ani fakt, že verzia zadarmo nedovoľuje monitorovanie dlhšie ako 10 minút a maximálny počet monitorovaní za jeden deň je taktiež 10.



Obrázek 1.2: Ukážka stromovej štruktúry na zvolenie si zariadenia s ktorým bude zachytávaná komunikácia.

1.3 Požadované funkcie

Základne požiadavky kladene na aplikáciu, čo by mala splnať.

1.4 Ciele práce

2. USB a Windows

2.1 USB zbernica

Plug and Play device tree(sposob akym si windows udrziava strom zariadeni na zbernici)(<https://docs.microsoft.com/sk-sk/windows-hardware/drivers/gettingstarted/device-nodes-and-device-stacks>)

2.2 Device object a device stack

PDO,FDO, Device object(<https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/creating-a-device-object>) <https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/creating-a-device-object>

2.2.1 Drivery

windows driver model(WDM) : <https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/types-of-wdm-drivers> bus driver(<https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/bus-drivers>), function driver(<https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/function-drivers>) a filter driver(<https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/filter-drivers>)

2.3 Komunikacia s USB zariadenim

sposob komunikacie operacneho systemu so zariadenim pripojenym na USB zbernicu : IRP(<https://docs.microsoft.com/en-us/windows-hardware/drivers/gettingstarted/irp-overview>) , URB (<https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/urb-overview>) a pod. <https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/irps>

2.4 USB descriptory

opis zakladnych USB descriptorov, hlavne tych ktore neskor aj vyuzivam v program(Device, Interface, Endpoint, Configuration, String, Setup) : <https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/usb-descriptors> <https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/usb-control-transfer>

2.4.1 Rozloženie USB zariadenia z hladiska descriptorov

<https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/usb-device-layout>

2.5 HID zariadenia

hid zariadenie obecne, priklady <https://docs.microsoft.com/en-us/windows-hardware/drivers/hid/>

2.5.1 Reporty

Input/Output/Feature reporty.

2.5.2 Report Descriptor

Opis report descriptoru, k comu sluzi, pripadne ako z neho vycitat zaujimave data (neskor vyuzite v programe pri parsovani HID Report Descriptoru na naslednu semanticku analyzu dat ktore posielala zariadenie)

3. Analýza

3.1 Získanie USB packetov

3.1.1 Windows exclusive mód

opisat co to je, a dolezite je spomenut, ze windows otvara v exclusive mode zakladne HID zariadenia ako mys a klavesnica

3.1.2 Známe knižnice

opisat zakladne kniznice na sledovanie USB zbernice a preco som ich nemohol pouzit : libUSB, hidAPI, moufiltr, SetupAPI, WinUSB

3.1.3 Third-party aplikácie

opisat odkial nakoniec ziskavam packety - USBPcap a Wireshark

3.2 Spracovávanie pcap súborov

moznosti ako citat pcap subory : bud pouzit uz existujucu kniznicu : na linuxe Libpcap, windows NPcap(deprecated WinPcap), alebo citat subory manualne : std::istream alebo QFile

3.3 Sémantická reprezentácia dát

ako si z dat vytiahnut udaje ktore su potom pouzite na semanticku analyzu implementovanych HID zariadeni : HID Report parser, InputValues a EndpointDevice struct. Nasledne sparovanie - ako vybrat spravny report pre konkretny input

3.4 Voľba frameworku

obecne co by som od toho GUI priblizne chcel, potom opisat preco som si vybral prave Qt a v nasledujucich kapitolach opisat rozhodnutia uz v Qt dovod preco som si zvolil qt namiesto inych c++ GUI frameworkov(napriklad sfml)

3.5 Zobrazenie základných informácií

ako zobrazovat zakladne info o packete : pouzit QListWidget alebo QTableWidget (pripadne nieco ine ako nejaky abstract viewmodel), narok na zakladne funkcionality : lahka rozsiritelnost o dalsie "stlpceky" , moznost jednoduchej interakcie(doubleClick na polozku). Mat vsetky info na jednom okne / mat pop-up okna.

3.6 Zobrazenie sémantického významu dát

ako vyzobrazit semanticky vyznam roznych dat - descriptory, usb header, vyznam input dat roznych HID zariadeni

3.7 Hexdump

ako v qt urobiť hexdump - do čoho zobrazovať data (vytvoriť si vlastný viewer dedený od QAbstractScrollArea, prípadne niečoho iného) vs najst niečo čo už v qt je a upraviť to aby to sedelo požiadavkám. Vziať do úvahy bežné funkcie hexdumpu : selection módy (označiť naraz hexa a im odpovedajúce printable), logické oddelenie dát (napríklad farbami)

4. Vývojová dokumentácia

4.1 Architektúra aplikácie

4.2 Jadro aplikácie

4.2.1 USB_Packet_Analyzer

riadi celkový beh programu, reaguje na input od užívateľa

4.2.2 Item Manager

spracovanie samostatného packetu a uloženie dát o ňom

4.2.3 DataViewer

trieda ktorá má na starosti vyskakovacie okno po dvojkliku a item a následne reaguje na input od užívateľa v okne

4.2.4 TreeItem

reprezentuje jednotlivé nody v stromovej štruktúre ktorá sa potom využíva na zobrazenie dát v QTreeView

4.3 Modely

4.3.1 AdditionaldataModel

model na spravovanie zvyšných dát (dát ktoré nie sú súčasťou hlavicej packetu)

4.3.2 ColorMapModel

vyobrazenie pomocnej mapy na lepšie sa zorientovanie v zvyraznenom hex-dumpe

4.3.3 DataViewerModel

model na hexdump - prenáša hex/printable a zároveň o čo vlastne ide (konkrétny descriptor, interrupt data, ...)

4.3.4 TreeItemBaseModel

model na QTreeView ktorým využíva TreeItem

4.3.5 USBPcapHeaderModel

model na QTreeView ale špeciálne pre USBPcap hlavicku packetu

4.4 Interpretery

4.4.1 BaseInterpreter

abstractna trieda od ktorej dedia vsetkz interpretery

4.4.2 Interpreter factory

factory trieda na pridelenie konkretného interpreteru za runtimu kvoli jednoduchosti na lepsie rozsirenie programu do buducnosti

4.4.3 Interpretery descriptorov

Config,Device,Setup,String,...

4.4.4 Interrupt transfer interpretery

obecne interrupt transfer interpreter - sluzi skor ako factory na rozne doteraz implementovane HID zariadenia

Joystick interpreter

Mouse interpreter

Keyboard interpreter

4.5 Delegáti

DataViewerDelegate

Qt delegat - stara sa o highlight hexdumpu

4.6 HID

4.6.1 HIDDevices

staticka trieda, drzi vsetky rozpoznane HID zariadenia a obsahuje funkcie specificke nich - parsovanie HID Report descriptoru

4.7 Práca so súbormi

4.7.1 FileReader

praca zo suborom a predavanie precitanych dat, offline/online capture, QFile vs std::istream

4.8 Globálne dáta

4.8.1 ConstDataHolder

staticka trieda na drzanie si konstant ktore su potrebne napriec celym programom. Mapovanie z enumu do jeho stringovej reprezentacie

4.8.2 PacketExternStructs

obsahuje definiciu vsetkych dolezitych USBPcap structov, pcap structov, enumov a vsetkych structov ktore pouzivam v aplikacii

5. Možnosti rozšírenia

Rozobrať čo všetko sa dá urobiť s tými dátami, ktoré už mám uložené v pamäti, ale momentálne sa s nimi nič nedeje

5.1 Ukladanie výstupu do súboru

výstup analýzy do súboru(textového)

5.2 Iná vizuálna reprezentácia dát

Momentálne vyzobrazujem dáta prevažne v QTreeView alebo QTableView, ale vďaka tomu ako ich mám uložené + to že nad nimi operuje nejaký model ktorý vie vrátiť dáta na základe indexu, by nemuselo byť taká zložitá pridať inú vizualizáciu dát(napríklad obrázkovú ako tu : https://www.usbmadesimple.co.uk/ums_5.htm)

5.3 Pridávanie nových interpreterov pre descriptor

pridanie nových druhov descriptorov - pridať nový interpreter do factory

5.4 Pridanie interpreteru na interrupt transfer

pridanie analýzy interrupt transferu aj pre ine ako hid zariadenia

5.4.1 Pridanie nových HID zariadení

nové HID zariadenie - pridanie do interrupt "factory"

5.5 Pridanie analýzy pre isochronous a bulk transfer

semantická analýza aj iných ako interrupt alebo control transferov - momentálne sú rozpoznávané len v hexdumpe

5.6 ?Možnosť rozšírenia na iné platformy?

uprava aplikácie aby bola prenositeľná aj na iné platformy, čo všetko by tam bolo treba upraviť(pravdepodobne nie veľa, keďže Qt je prenosné, a prakticky jediné čo používam spojené s Windowsom sú jeho structy na rôzne descriptor)

6. Užívateľská dokumentácia

6.1 Inštalácia

nastavenie celkovej aplikácie, ale aj nainštalovanie USBPcap + wireshark a ich kombinácia pre live capture

6.2 Orientácia v GUI aplikácie

popis k jednotlivým tlačidlám gui

6.3 Používanie aplikácie

ako spustiť live/offline capture, a celkovo ako pracovať s aplikáciou (popis funkcií - doubleClick na item => zobrazí sa pop-up okno s bližšou analýzou)

7. Záver

7.1 Zhrnutie

celkove zhrnutie prace, ?praca s Qt?

7.2 Budúce plány

Seznam obrázků

1.1	Ukážka vyobrazenia jednotlivých bytov.	4
1.2	Ukážka stromovej štruktúry na zvolenie si zariadenia s ktorým bude zachytávaná komunikácia.	5

Seznam tabulek

Seznam použitých zkratek

Přílohy

.1 První příloha