

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

VYUŽITÍ PROGRAMU WIRESHARK V PŘEDMĚTU KOMUNIKAČNÍ TECHNOLOGIE

UTILIZATION OF WIRESHARK SOFTWARE IN COMMUNICATION TECHNOLOGY COURSE

BAKALÁŘSKÁ PRÁCE BACHELOR'S THESIS

AUTOR PRÁCE AUTHOR Jan Šíma

VEDOUCÍ PRÁCE SUPERVISOR

doc. Ing. Jan Jeřábek, Ph.D.

BRNO 2020



Bakalářská práce

bakalářský studijní program Informační bezpečnost

Ústav telekomunikací

Student: Jan Šíma *Ročník:* 3

ID: 203717 *Akademický rok:* 2019/20

NÁZEV TÉMATU:

Využití programu Wireshark v předmětu Komunikační technologie

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte fungování a možnosti využití programu Wireshark. Dále se zaměřte na problematiku základních komunikačních protokolů používaných v paketových sítích. Prostudujte dostupné laboratorní scénáře pro tento program a následně navrhněte a popište dva vlastní komplexní laboratorní scénáře. Cílem prvního scénáře bude seznámit vhodným způsobem studenty se základy používání programu Wireshark. U druhého scénáře pak již půjde o pokročilejší scénář využití tohoto programu v problematice protokolů spadajících do předmětu Komunikační technologie. Výstupem práce budou dva kompletní scénáře včetně podrobných návodů pro studenty v českém jazyce, prokonzultovaných s vedoucím práce, předpřipravených výchozích situací či souborů, doplňujících úkolů pro studenty a vzorového řešení. Předpokládá se, že délka realizace jedné úlohy bude pro studenta přibližně 2 hodiny času.

DOPORUČENÁ LITERATURA:

[1] KUROSE, J. F., ROSS, K. W., Computer networking: a top-down approach. 7th global ed. Essex: Pearson, 2017, 852 s. ISBN 978-1-292-15359-9.

[2] JEŘÁBEK, J. Komunikační technologie. Skriptum FEKT Vysoké učení technické v Brně, 2018. s. 1-172.

Termín zadání: 3.2.2020

Termín odevzdání: 8.6.2020

Vedoucí práce: doc. Ing. Jan Jeřábek, Ph.D.

doc. Ing. Jan Hajný, Ph.D. předseda rady studijního programu

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Fakulta elektrotechniky a komunikačních technologií, Vysoké učení technické v Brně / Technická 3058/10 / 616 00 / Brno

ABSTRAKT

Bakalářská práce se zabývá problematikou protokolové sady TCP/IP. Konkrétně jde především o protokoly ICMP, TCP, DNS, SIP a RTP. Hlavním cílem práce je vytvořit dva komplexní scénáře vhodné pro laboratorní cvičení předmětu Komunikační technologie. Scénáře obsahují návod s postupem řešení pro studenty, včetně doplňujících úkolů. První scénář seznámí studenty s prostředím programu Wireshark. Dále se tento program využívá k analýze a rozboru zachycených paketů, které byly popsány v teoretické části práce. Aby bylo možné zachytit pakety určitých protokolů, tak se ve zhotoveném scénáři pracuje také s příkazovým řádkem a s příkazy ping, traceroute a nslookup, včetně jejich parametrů. Druhý scénář se zaměřuje na analýzu VoIP hovoru, včetně rekonstrukce zachyceného hovoru. Druhá část scénáře porovnává rozdíly mezi protokoly TCP+TLS a QUIC při načítání zabezpečené webové stránky. Oba scénáře byly otestovány ve školní laboratoři, kde se řešení vypracovává ve virtualizovaném operačním systému. Zároveň každý scénář obsahuje doplňující otázky pro studenty a pro vyučujícího je připraven soubor, který obsahuje správné odpovědi.

KLÍČOVÁ SLOVA

Wireshark, TCP/IP, ICMP, DNS, RTP, VoIP, QUIC, analýza síťových protokolů, ping.

ABSTRACT

This bachelor thesis is focused on the issue with protocol set TCP/IP. Specifically it is about protocols ICMP, TCP, DNS, SIP and RTP. Main purpose of this thesis is to create two complex scenarios suitable for laboratory exercises for Communication technology course. The scenarios contain instructions with the solution procedure for students, including additional tasks. The first scenario introduces students to the Wireshark environment. Furthermore, this program is used to analyze captured packets, which were described in the theoretical part of the work. To capture packets of certain protocols, the command line and ping, traceroute, and nslookup commands, including their parameters, are also used in the scenario. The second scenario focuses on the VoIP call analysis, including the restoration of the captured call. The second part of this scenario compares differences between protocols TCP+TLS and QUIC while loading the secured web page. Both scenarios were tested in a school laboratory, where the scenarios are developed in a virtualized operating system. Also, each scenario contains additional questions for students and for teacher there is prepared a file, which contains the correct answers.

KEYWORDS

Wireshark, TCP/IP, ICMP, DNS, RTP, VoIP, QUIC, network protocols analysis, ping.

ŠÍMA, Jan. *Využití programu Wireshark v předmětu Komunikační technologie*. Brno, 2019, 122 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: doc. Ing. Jan Jeřábek, Ph.D.

Vysázeno pomocí balíčku thesis verze 3.05; http://latex.feec.vutbr.cz

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma "Využití programu Wireshark v předmětu Komunikační technologie" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu doc. Ing. Janu Jeřábkovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Ú	vod		9							
1	Pro	tokoly a funkce modelu Transmission Control Protocol/Internet								
Protocol										
	1.1	Model TCP/IP	10							
	1.2	Transmission Control Protocol (TCP)	11							
	1.3	User Datagram Protocol (UDP)	13							
	1.4	Internet Protocol (IP)	14							
	1.5	Internet Control Message Protocol (ICMP)	14							
	1.6	Time to live (TTL)	16							
	1.7	Systém doménových jmen (DNS)	17							
2	Uti	lity implementující funkce protokolů ICMP a DNS	19							
	2.1	Příkazový řádek	19							
	2.2	Ping	19							
		2.2.1 Průběh příkazu ping	21							
		2.2.2 Parametry pro ping	21							
	2.3	Traceroute	22							
	2.4	Nslookup	24							
	2.5	DNS resolver	25							
3	Pro	tokoly IP přenosu hlasu	26							
	3.1	Session Initiation Protocol (SIP)	26							
		3.1.1 Navázání a ukončení SIP spojení	26							
	3.2	Real-time Transport Protocol (RTP)	28							
4	Pro	tokoly zajišťující bezpečený přenos dat	29							
	4.1	Transport Layer Security (TLS)	29							
	4.2	Quick UDP Internet Connections (QUIC)	29							
5	Dal	ší použité programy	31							
	5.1	VM ware a virtualizovaný operační systém 	31							
	5.2	Wireshark	32							
	5.3	Linphone	33							
6	Prů	zkum dostupných úkolů a řešená problematika při vytváření								
	scéi	nářů	35							
	6.1	Hodnota TTL	35							

	6.2	Sítové	připojení virtualizovaného systému	36
	6.3	Výběr	programu pro uskutečnění VoIP hovoru	38
	6.4	Výběr	serveru pro zpřístupnění webové stránky	38
7	Náv	vrhy sc	enářů	40
	7.1	Návrh	prvního scénáře	40
	7.2	Návrh	druhého scénáře	41
	7.3	Komp	letní znění vytvořených scénářů	42
Zá	ávěr			43
Li	terat	ura		45
Se	eznan	n symł	oolů, veličin a zkratek	48
Se	eznan	n přílo	h	50
\mathbf{A}	Kor	npletn	í návod pro první vytvořený simulační scénář	52
	A.1	Teoret	tický úvod	53
		A.1.1	Protokoly transportní vrstvy - TCP a UDP	53
		A.1.2	Internet Control Message Protocol (ICMP)	54
		A.1.3	Systém doménových jmen (DNS)	54
	A.2	Prakti	ická část	55
		A.2.1	Popis prostředí programu Wireshark	55
		A.2.2	Základní zachycení ICMP paketů	57
		A.2.3	Filtry a jejich kombinace	58
		A.2.4	Úprava parametrů pro ping a zachycení	59
		A.2.5	Zachycení ICMP - zachycení chybového kódu	65
		A.2.6	Další prvky analýzy ICMP záhlaví	69
		A.2.7	DNS resolver	73
в	Kor	npletn	í návod pro druhý vytvořený simulační scénář	76
	B.1	Teoret	tický úvod 	77
		B.1.1	Protokoly realizující hovor skrze IP	77
		B.1.2	Protokoly zajišťující zabezpečený přenos dat	78
	B.2	Realiz	ace druhého scénáře	79
		B.2.1	Popis prostředí programu Linphone	79
		B.2.2	Linphone registrace a nastavení	80
		B.2.3	Rekonstrukce hovoru a jeho základní zabezpečení	83
		B.2.4	Představení funkce analýzy pomocí grafů	90
		B.2.5	Spojení zachycených souborů	93

		B.2.6	Srovnání protokolů TCP a UDP u DNS resolveru	. 95
		B.2.7	Nastavení prohlížeče a Wiresharku pro analýzu protokolu	
			QUIC	. 96
		B.2.8	Srovnání protokolů při načítání webových stránek	. 99
		B.2.9	Změna parametrů sítě a analýza změn	. 103
		B.2.10	Analýza QUICu v prostředí prohlížeče Chrome	. 107
С	Řeše	ení prv	vního simulačního scénáře	111
	C.1	Popis ₁	prostředí programu Wireshark	. 111
	C.2	Základ	ní zachycení ICMP paketů	. 111
	C.3	Filtry	a jejich kombinace	. 111
	C.4	Úprava	a parametrů pro Ping a zachycení	. 111
	C.5	Zachyc	ení ICMP - zachycení chybového kódu	. 113
	C.6	Další p	prvky ICMP záhlaví	. 114
	C.7	DNS r	esolver	. 114
D	Řeše	ení dru	ıhého simulačního scénáře	115
	D.1	Popis ₁	prostředí programu Linphone	. 115
	D.2	Linpho	one registrace a nastavení	. 115
	D.3	Rekons	strukce hovoru a jeho základní zabezpečení	. 115
	D.4	Předst	avení funkce analýzy pomocí grafů	. 116
	D.5	Spojen	í zachycených souborů	. 118
	D.6	Srovná	ní protokolů TCP a UDP u DNS resolveru	. 118
	D.7	Nastav	rení prohlížeče a Wiresharku pro ${\rm anal}\acute{\rm y}{\rm zu}$ protokolu QUIC $% {\rm e}{\rm z}$.	. 119
	D.8	Srovná	ní protokolů při načítání webových stránek	. 119
	D.9	Změna	parametrů sítě a analýza změn	. 120
	D.10	Analýz	za QUICu v prostředí prohlížeče Chrome	. 121
\mathbf{E}	Obs	ah pos	kytnutého odkazu se soubory	122

Úvod

S rapidním vývojem internetu se vyvíjejí a využívají stále novější protokoly a na nich založené programy a služby. Z těchto důvodů je nutné znát základní komunikační protokoly, ze kterých právě tyto nové protokoly vycházejí. Protokoly popisované v této práci jsou součástí rodiny modelu TCP/IP a jsou podrobně popsány ve veřejně přístupných RFC (Request for Comments) dokumentech. Právě to, že jsou tyto dokumenty veřejně přístupné, tak přispívá k rychlému vývoji nových nebo vylepšování stávajících protokolů. Užitečné je to i pro studenty, kteří nemusí hledat odborné články, ale stačí si najít oficiální dokument k danému protokolu. K jednoduššímu poznávání základních protokolů slouží jejich teoretický rozbor a praktická analýza, která je v této práci zahrnuta v rámci dvou vytvořených laboratorních scénářů. Ty se soustřeďují hlavně na komunikační protokoly TCP, UDP, DNS, ICMP, SIP a RTP. U těchto základních protokolů je nutné se seznámit s jejich možnostmi využití a také znát jejich výhody a nevýhody.

Mnoho běžných uživatelů si ani neuvědomuje, že lze zachytit a sledovat tok dat z daného zařízení. Kvůli tomu může vzniknout řada rizik, které mohou vést například ke ztrátě osobních údajů nebo finančních prostředků. Je tak dobré, aby si studenti toto zachycení dat vyzkoušeli a věděli, které protokoly se využívají u určitých programů a služeb. Také si ujasní kolik přenesených dat se skrývá za několika kliknutími uživatele, že tyto protokoly mají pevně dané záhlaví, které jim slouží k identifikaci při cestě po síti nebo na koncové stanici atd. U laboratorních úloh je důležité, aby si studenti dokázali spojit teorii probranou na přednáškách s praxí, kterou se jim vytvořené scénáře pokusí přiblížit.

1 Protokoly a funkce modelu Transmission Control Protocol/Internet Protocol

Ve vytvořených scénářích budeme pracovat s protokoly z rodiny modelu Transmission Control Protocol/Internet Protocol (TCP/IP). Proto se v této části nachází popis funkce tohoto modelu a také popis protokolů, které se následně budou analyzovat a je potřeba znát princip jejich fungování.

1.1 Model TCP/IP

TCP/IP je jeden ze dvou síťových modelů, které jsou zmiňovány v informačních sítích. Druhým modelem je referenční ISO/OSI, který obsahuje sedm vrstev. U obou těchto modelů spolu komunikují sousední vrstvy. Každá vrstva si bere informace z vrstvy nižší a následně výsledky své činnosti předává vrstvě vyšší. Jako příklad můžeme uvést vrstvu aplikační, která využívá protokoly TCP (Transmission Control Protocol) nebo UDP (User Datagram Protocol) pro přenos HTTP (Hypertext Transfer Protocol)stránek. Model TCP/IP však využívá pouze čtyři vrstvy, jak je vidět na obr. 1.1. Z toho vychází požadavek na jednoduchost a rychlost v rámci komunikační podsítě. TCP/IP na rozdíl od ISO/OSI přenechává otázku spolehlivosti přenosu až na koncových zařízeních, kdežto u ISO/OSI dochází zajištění kvality přenosu z části na každé vrstvě.

TCP/IP tedy přináší představu o tom jak by měly informační sítě vypadat. Definují jejich funkčnost pomocí čtyř vrstev, kdy každá vrstva má svůj úkol a specifikované protokoly, které používá. [1]



Obr. 1.1: Struktura modelu TCP/IP.

1.2 Transmission Control Protocol (TCP)

TCP je spojově orientovaný a plně duplexní protokol transportní vrstvy. Je označován jako spolehlivý protokol, protože se soustředí na garanci doručení dat a také doručení ve správném pořadí. Pro spolehlivé doručování je nutno nejdříve sestavit spojení. Následně probíhá přenos dat a to až do té doby, dokud nebude spojení řádně ukončeno. Protokol je využíván ve službách, kde nevyžadujeme nejrychlejší možné doručení, ale spolehlivé doručení všech segmentů. To může být například u emailu nebo přenosu souborů. Třeba u instalačního souboru požadujeme stažení všech dat a nevadí nám, že stahování bude trvat o pár milisekund déle. Mohlo by totiž dojít k poškození souboru a následné chybě při instalaci. Aby ovšem mohlo dojít k přenosu samotných dat, musí nejdříve dojít k vytvoření spojení. To je prováděno pomocí příznaků TCP protokolu. Vytvoření i ukončení spojení je znázorněno na obr. 1.2. [2]



Obr. 1.2: Schéma navázání a ukončení spojení protokolu TCP zobrazující komunikaci dvou stran pomocí TCP příznaků.

Pro navázání spojení je použito tří zpráv mezi klientem a serverem (odtud také anglický výraz "three-way handshake"). Popis komunikace mezi klientem a serverem je vysvětlen pomocí obr. 1.3. Při popisu bude vždy v závorce uvedeno číslo, které koresponduje s číslem řádku, respektive paketu na tomto obrázku. Výměnu začíná klient, který odesílá segment s příznakem SYN (139) s žádostí o navázání spojení (SYN - zkratka ze synchronization - synchronizace). Následně server posílá také SYN příznak a k tomu ještě potvrzovací příkaz ACK (140), kterým potvrzuje žádost klienta (ACK - zkratka z acknowledgement - potvrzení). Nakonec ještě klient pošle

potvrzení serveru příkazem ACK (141) a spojení je vytvořeno. Strany spolu mohou obousměrně komunikovat (full-duplex).

Samotný přenos dat je založen na spolehlivosti doručení všech těchto dat. V případě, že by došlo ke ztrátě některých segmentů při cestě po síti, tak musí být odeslány znovu. Kontrolu nad doručenými a případně nedoručenými segmenty zajišťuje číslování jednotlivých bajtů. Těchto pořadových čísel je také využito při nesprávném pořadí doručení segmentů. V příkladu na obr. 1.3 přenášíme data protokolu DNS (Domain Name System – Systém doménových jmen). Tento protokol odeslal dvě zprávy – DNS žádost (143) a DNS odpověď (149). Protože protokol TCP zajišťuje spolehlivé doručení tak oba tyto pakety musí potvrdit příznakem ACK (pakety na řádcích 146 a 158). DNS komunikace je hotová, klient získal IP adresu a můžeme ukončit TCP spojení.

	tcp.stream eq 1						X 🛋 🔻	Expression +	
No.	Source	Destination	Proto: Ler	ngt Info					٦
Г	139 147.229.197.30	147.229.190.143	TCP	66 3854 → 53	[SYN]	Seq=0 Win=64240 L	en=0 MSS=1460 Win=65535 Lon	WS=256 SACK_P	E
	140 147.229.190.145	147 229 190 143	TCP	$54 3854 \rightarrow 57$	EACK1	Seg=1 Ack=1 Win=1	051136 Len=0	-0 00+1-CCU 0-	1
	143 147 229 197 30	147 229 190 143	DNS	94 Standard	duecy	OXOOOO A Sernam cz	OPT		
	146 147,229,190,143	147.229.197.30	TCP	60 53 → 3854	ГАСКІ	Sea=1 Ack=41 Win=	65656 Len=0		
	149 147.229.190.143	147.229.197.30	DNS 2	31 Standard	query	response 0x0000 A	seznam.cz A 7	7.75.75.176 A	7
	158 147,229,197,30	147.229.190.143	TCP	54 3854 → 53	[FIN,	ACK] Seq=41 Ack=1	78 Win=105088	0 Len=0	
	159 147.229.190.143	147.229.197.30	ТСР	60 53 → 3854	[ACK]	Seq=178 Ack=42 Wi	n=65696 Len=0	,	
	160 147.229.190.143	147.229.197.30	TCP	60 53 → 3854	[FIN,	ACK] Seq=178 Ack=	42 Win=65696	Len=0	
L	163 147.229.197.30	147.229.190.143	ТСР	54 3854 → 53	[ACK]	Seq=42 Ack=179 Wi	n=1050880 Len	.=0	
<								2	>
\mathbf{v}	Transmission Control	Protocol, Src Por	rt: 3854	. Dst Port:	53. Se	a: 41, Ack: 178, L	en: 0	1	
	Source Port: 3854	,						_	
	Destination Port:	53							
	[Stream index: 1]								
	[TCP Segment Len:	0]							
	Sequence number: 4	1 (relative se	quence	number)					
	[Next sequence num	ıber: 41 (relat	ive seq	uence number)]				
	Acknowledgment num	ıber: 178 (rela	ative ac	k number)					
	0101 = Header	Length: 20 bytes	; (5)						
	 Flags: 0x011 (FIN, 	ACK)							
	000	= Reserved: Not s	et						
	0	= Nonce: Not set							
	0	= Congestion Wind	low Redu	ced (CWR): N	ot set				
		= ECN-Echo: Not s	et						
	0	= Urgent: Not set							
	1	= Acknowledgment:	Set						
	0	= Push: Not set							
		= Keset: Not set							
		= Syn: Not set						_	
	TCD Flags:	- 111; Set							
	Window size velue:	4105							
	[Calculated window	size: 10508801							
	[carculated window	SIZE. 1050800J							

Obr. 1.3: Průběh navázání a ukončení spojení a také přenos DNS dat pomocí protokolu TCP.

Ukončení spojení probíhá podobně jako vytváření spojení a to pomocí příznaků FIN a ACK (FIN - finish - ukončení). Může dojít k tomu, že jedna strana ukončí spojení a bude pouze přijímat data od strany druhé. Takže ukončení spojení nemusí mít pouze tři kroky, jako tomu bylo u sestavování spojení. Strana, která již nechce posílat další data, odešle segment s příznakem FIN (158), který dá druhé straně vědět, že požaduje ukončení spojení. Druhá strana potvrdí ukončení příznakem ACK (159). V případě, že i druhá strana chce ukončit spojení odešle také segment s příznakem FIN (160) a druhá strana tuto žádost potvrdí příznakem ACK (161).

1.3 User Datagram Protocol (UDP)

UDP je také protokol transportní vrstvy a může být označován jako protiklad k protokolu TCP. Není spojově orientovaný a nepotřebuje tedy navazovat spojení před přenosem dat. Posílá samostatné zprávy u kterých nezaručuje jejich doručení. Díky tomu také nese méně informací v záhlaví protokolu, protože nepotřebuje data číslovat, nebo si uchovávat informace o doručených a nedoručených datagramech.

Oproti TCP je vhodný pro aplikace, které potřebují rychlý přenos dat i za cenu ztráty některých datagramů. To může být například při přenosu dat streamovaných videí nebo multimédií, VoIP hovorů a online her. UDP protokol využívají také protokoly DNS (Domain Name System) a DHCP (Dynamic Host Configuration Protocol). U těchto služeb by bylo opětovné odesílání ztracených datagramů zbytečné, protože například u VoIP hovorů stačí i malé zpoždění v řádech několika set milisekund a již dochází k problémům v komunikaci. Stejné je to u streamování videa, kdy ztracený datagram již nemá cenu posílat znovu, protože přehraná část videa už je dávno nahrazena novými snímky. U těchto přenosů se využívá nedokonalostí lidských smyslů, kdy si některých chybějících pixelů ve snímcích videa nemáme šanci všimnout, takže některé chybějící datagramy ani nezaznamenáme.

UDP protokol tedy nepotřebuje tolik režijních dat pro přenos a na rozdíl od TCP protokolu mu také stačí menší šířka pásma, právě kvůli tomu, že nepotřebuje tolik režie a neposílá data vícekrát. Výběr transportního protokolu tedy záleží na typu aplikace a vlastnostech, které tato aplikace má splňovat. Oba protokoly obsahují informace o zdrojovém a cílovém portu, takže protokol může v jednu chvíli přenášet data pro více aplikací zároveň. Porovnání množství režijních dat v záhlavích je ukázáno na obr. 1.4. [3]

Transmission Control Protocol (TCP)										
1. bajt	2. bajt	3. bajt	4. bajt							
zdrojo	vý port	cílový port								
	pořadové číslo									
	potvrzovací číslo									
příznaky (ACK,) velikost TCP záha										
kontrolní součet urgent pointer										

User Datagram	Protocol	(UDP)
---------------	----------	-------

1. bajt	2. bajt	3. bajt 4. bajt					
zdrojo	vý port	cílový port					
Délka (UI)P + data)	kontrolr	ní součet				

Obr. 1.4: Srovnání záhlaví transportních protokolů TCP a UDP.

1.4 Internet Protocol (IP)

Internet Protocol, je protokol síťové vrstvy. Směrovače, které umí informace ze záhlaví tohoto protokolu zpracovat, zajišťují směrování paketů od zdroje k cíli. Cesta může obsahovat jednu nebo více sítí, kterými pakety projdou. K této funkci protokol definuje pojem IP adresy. Každé síťové rozhraní, každý uzel má tuto IP adresu, pomocí které se identifikuje v síti. Můžeme narazit na dvě verze IP adres, a to IPv4 a IPv6 adresy. IPv4 adresy mají 32 bitů z čehož můžeme odvodit celkový počet skoro čtyř miliard IPv4 adres. Přesto však počet těchto adres není dostatečný a i z tohoto důvodu dnes IPv4 adresy nahrazuje novější IPv6 protokol. Tyto adresy již mají 128 bitů a limit možných připojených zařízení tedy mnohonásobně přesahuje počet aktuálně připojitelných přístrojů k síti. IP protokol také provádí fragmentaci a následné složení fragmentů do původní podoby v případě, že je fragmentace nutná. [4] [5]

1.5 Internet Control Message Protocol (ICMP)

ICMP je jeden z protokolů síťové vrstvy modelu TCP/IP. Jak už nám anglická zkratka napovídá jedná se o protokol, který slouží k přenosu režijních dat, jako jsou informace o chybách nebo různá oznámení. Oznámení jsou využívána například při použití příkazu ping, který ke své funkci využívá právě zprávy protokolu ICMP. Konkrétně jde o echo request a echo reply, kdy nám právě odpověď oznamuje dostupnost požadované stanice a dobu odezvy. Příkladem pro informování o chybách může být třeba případ kdy využíváme HTTP pro zobrazení webových stránek, ale místo funkční stránky se nám objeví informace o chybě "*Destination network unreachable*". Právě toto chybové hlášení je přeneseno pomocí ICMP protokolu od místa, kde chyba vznikla, zpět k nám do webového prohlížeče. Obecná struktura ICMP paketu je zobrazena na obr. 1.5.

typ kód kontrolní součet proměnná část - liší se podle typu ICMP paketu (např. identifikátory, pořadová čísla…)	1. bajt	2. bajt	3. bajt	4. bajt					
proměnná část - liší se podle typu ICMP paketu (např. identifikátory, pořadová čísla…)	typ	kód	kontrolní součet						
	proměnná část - liší se podle typu ICMP paketu (např. identifikátory, pořadová čísla)								
ICMP data (množství přidaných dat se opět liší podle typu paketu)	ICMP da	ita (množství přidaných c	dat se opět liší podle typu	ı paketu)					

Obr. 1.5: Struktura zobrazující obecný ICMP paket. Je zde proměnná část, která může, ale nemusí obsahovat data identifikátoru a pořadového čísla. Velikost datové části se také mění podle aktuální chybové zprávy, kterou ICMP přenáší.

Chyba se identifikuje dvěma čísly. První určuje typ chyby a druhé číslo označuje kód chyby. Výčet několika možných chyb, včetně identifikátorů je uveden v tab. 1.1. [3]

Tab. 1.1: Význam chybových zpráv protokolu ICMP, které jsou označeny pomocí typu a kódu.

typ	kód	popis chyby nebo oznámení
0	0	ping - odpověď (echo reply)
8	0	ping - požadavek (echo request)
3	0	cílová síť není dostupná
11	0	hodnota TTL vypršela během přenosu
11	1	fragmenty se nepovedlo poskládat do původních dat

V následující části bude rozebrána struktura ICMP paketu. V tomto příkladu vybereme paket, který vrací chybovou zprávu ICMP Time-to-live. Paket obsahuje 70 bajtů dat a jeho struktura je zobrazena na obr. 1.6. Jsou v něm zapouzdřena data z vrstvy fyzického rozhraní obsahující záhlaví ethernetu, dále záhlaví IP a ICMP protokolu a také přenášená data. Ethernetový rámec obsahuje kromě 14 bajtového záhlaví a dat také ethernetové zápatí (trailer) o velikosti 4 bajty. Toto zápatí obsahuje hodnotu pro kontrolní součet, který může odhalit poškozená data v daném ethernetovém rámci. Avšak Wireshark tuto část ve výpisu vynechává, a tak není zápatí zobrazeno ani na obrázku 1.6 níže. Část ethernetu má tedy 14 bajtů a záhlaví IPv4, obsahující například hodnotu TTL (Time to live) a také cílové a destinační IP adresy, má 20 bajtů. Většinu z velikosti ICMP paketu obsahuje právě záhlaví ICMP a přenášená data. V záhlaví ICMP nalezneme dříve zmíněná dvě čísla označující typ a kód chyby. Dva bajty jsou využity na kontrolní součet a další 4 bajty záhlaví nejsou v tomto pakety využity. Data, která přenáší tento paket, jsou originální data pocházející z ICMP Echo request paketu včetně IPv4 záhlaví. Poznat to můžeme podle ICMP typu chyby, který má hodnotu 8, což odpovídá právě echo requestu. [6]



Obr. 1.6: Struktura paketu ICMP nesoucí chybu typu 11, kód 0 - vypršení hodnoty TTL na nula.

1.6 Time to live (TTL)

TTL (Time to live) je hodnota, která je zapouzdřená v záhlaví IP datagramu. Udává možný počet síťových prvků, kterými může paket projít při cestě po síti. S každým prvkem kterým projde se hodnota TTL dekrementuje o 1. V případě, že na směrovač dojde paket s hodnotou TTL 1, tak daný směrovač tuto hodnotu sníží na nula a paket zahodí a dále se s ním již nepracuje. Když dojde k zahození paketu směrovačem, tak směrovač dává vědět odesílateli (síťový prvek se zdrojovou IP adresou), že došlo k zahození pomocí ICMP chybové zprávy. Viz tabulka 1.1. Hodnota TTL byla zavedena, aby nedocházelo k tomu, že pakety uvíznou v síti v nekonečných smyčkách z důvodu špatných směrovacích tabulek. Pokud by TTL bylo nastaveno na maximální hodnotu 255 a došlo by k uvíznutí ve smyčce, tak paket bude za nějaký čas odstraněn a nebude zbytečně zahlcovat síť. Naopak se hodnoty TTL využívá například u příkazu traceroute, kdy se nejdříve posílá ICMP Echo request s hodnotou TTL 1. Příkaz traceroute je podrobněji popsán v samostatné podkapitole dále. [7]

1.7 Systém doménových jmen (DNS)

Domain Name System je systém, kterého je využíváno pro překlad doménových jmen a IP adres prvků v počítačových sítích. Tento systém je implementován na DNS serverech, které si uchovávají pomyslnou tabulku se sloupci IP adres a doménových jmen. V případě, že daný server dostane pomocí DNS protokolu žádost o zjištění určité adresy, tak tuto adresu vyhledá a poskytne ji dotazujícímu zařízení. V případě, že adresu nenalezne, musí se zeptat nadřízeného serveru, na který si adresu uchovává neustále. Když překlad zajistí a poskytne jej danému zařízení, tak si tyto informace po nějakou dobu uchovává v paměti (DNS cache). Tento časový údaj se označuje jako TTL v sekundách a určuje jej DNS odpověď od DNS serveru. Většinou se překlad nechává v paměti po dobu několika minut. Mohlo by totiž dojít k tomu, že se adresa hledaného serveru změní a k uživateli by se tato změna donesla až po vypršení doby TTL. Uživatel by tak k serveru nemohl přistoupit. DNS protokol využívá pro přenos transportní protokol UDP a je umístěn na portu číslo 53.

* [therne	t													_		×
File	Edit	View	Go Capture	Analyz	ze Statistic	s Telepho	ony W	/ireless	Tools	Help							
		•	🔝 🗙 🖾	۹ 🗢	⇒ 🕸 👔	· 🕹 📃			ə, 🎹								
dr	IS													X	- - - -	xpression.	
Source	e		Destination	Pi	Proto: Lenat	Info						~					
147.	229.19	97.30	147.229.19	0.143 D	NS 80	Standard	query	0x000	0 A se	znam.	cz OPT	ī.					
147.	229.19	97.30	147.229.19	0.143 D	NS 80	Standard	query	0x000	ο ΑΑΑΑ	sezn	am.cz	OPT					
147.	229.19	97.30	147.229.19	0.143 D	NS 80	Standard	query	0x000	0 CNAM	IE sez	nam.cz	: OPT					
147.	229.19	97.30	147.229.19	0.143 D	NS 96	Standard	query	0x000	0 PTR	172.7	5.75.7	7.in-addr.ar	pa OPT				
147.	229.19	97.30	147.229.19	0.143 D	NS 80	Standard	query	0x000	0 SOA	sezna	m.cz O)PT					
147.	229.19	90.143	147.229.19	7.30 D	NS 217	Standard	query	respo	nse Øx	0000	A sezn	am.cz A 77.7	5.75.176	A 77.75.7	5.172	RRSIG C)PT
147.	229.19	90.143	147.229.19	7.30 D	NS 241	Standard	query	respo	nse Øx	0000	AAAA s	eznam.cz AAA	A 2a02:59	8:4444:1:	:2 AAA	AA 2a02:	598:44
147.	229.19	90.143	147.229.19	7.30 D	ONS 429	Standard	query	respo	nse Øx	0000	CNAME	seznam.cz SO	A ans.sez	nam.cz RR	SIG RF	RSIG NSE	C3 OP1
147.	229.19	90.143	147.229.19	7.30 D	DNS 159	Standard	query	respo	nse Øx	0000	PTR 17	2.75.75.77.i	n-addr.ar	pa PTR ww	w.sezr	nam.cz N	IS ans
147.	229.19	90.143	147.229.19	7.30 D	ONS 236	Standard	query	respo	nse Øx	0000	SOA se	znam.cz SOA	ans.sezna	m.cz RRSI	G OPT		
<																	>
~	Answ	ens															~
	∨ s	eznam.	cz: type SO	A, class	s IN, mnam	ne ans.se	znam.c	z									
		Name	: seznam.cz														
		Type	: SOA (Star	t Of a z	zone of Au	thority)	(6)										
		Clas	s: IN (0×00	01)													
		Time	to live: 3	00													
		Data	length: 39														- 10
		Prim	ary name se	rver: an	ns.seznam.	cz											
		Resp	onsible aut	hority's	s mailbox:	hostmas	ter.se	znam.c	z								
	Serial Number: 2017276795																
	Refresh Interval: 60 (1 minute)																
		Retr	y Interval:	30 (30	seconds)												
	Expire limit: 604800 (7 days)																
		Mini	mum TTL: 30	0 (5 min	nutes)												¥
																	÷.
	🖌 Re	esponsibl	le authority's ma	ilbox (dns.s	soa.rname), 1	3 bytes						Packets: 1733	Displayed: 1	10 (0.6%)	P	Profile: Def	ault

Obr. 1.7: Zobrazení DNS žádostí a odpovědí několika typů záznamů. V dolní části jsou uvedeny podrobnosti SOA záznamu.

DNS servery mají svou určitou topologii. Jedná se o stromové uspořádání a vrchní servery jsou nazvány kořenové. Tyto servery jsou umístěny po celém světě a je jich více než 400. Další vrstva se anglicky popisuje jako top-level domain (TLD). V této vrstvě nalezneme DNS servery, které uchovávají záznamy například pouze o určité doméně nejvyšší úrovně. Doménou nejvyšší úrovně se rozumí koncovka, která se nachází na pravém konci doménového jména. Takže například DNS server, který uchovává některé doménové jména končící .com. Tyto domény bývají využívány pro komerční účely, podobně jako .org pro různé organizace. Dále známe koncovky státní, jako například .cz nebo .sk. Poslední vrstvou jsou DNS servery autoritativní. Sem můžeme zařadit servery společností jako třeba DNS server Facebooku nebo Googlu. DNS adresa Googlu bývá často využívána jako výchozí, IP adresa tohoto DNS serveru je 8.8.8.8. Doménové jméno může obsahovat celkem až 255 znaků, ale musí být děleno na subdomény o maximálním počtu znaků 63. [3]

DNS protokol používá několik typů záznamů, které si následně DNS servery uchovávají v databázích. Mezi nejběžnější patří A záznamy, což jsou IPv4 adresy a jejich jmenné překlady, dále jsou známy AAAA záznamy, které jsou zaměřeny na záznam doménových jmen a jejich překladů v podobě IPv6 adres. Dále třeba MX (Mail eXchangers) záznamy, které určují adresu pro příjem e-mailů. Záznam CNAME (Canonical NAME record) definuje takzvaný alias pro již používanou doménu. Dalším záznamem jsou PTR záznamy, které slouží ke zpětnému hledání v DNS databázi. Kdy známe IP adresu a chceme získat doménové jméno. Posledním záznamem, který bude uveden je SOA (Start Of Authority) záznam, který obsahuje administrativní informace, včetně jména primárního serveru, emailové adresy na administrátora tohoto serveru, sériové číslo a dále několik časových údajů jako je hodnota TTL a hodnota expirace záznamů. Tyto typy záznamů jsou zachyceny na obr. 1.7. [8]

2 Utility implementující funkce protokolů ICMP a DNS

2.1 Příkazový řádek

Příkazovým řádkem rozumíme uživatelské rozhraní, které neobsahuje grafickou nadstavbu, jak to známe u jiných programů. Není zde žádné menu nebo lišta s ikonami. Jedná se pouze o prostý řádek, do kterého píšeme své požadavky v podobě příkazů. Kromě případu, kdy chceme označit část textu na obrazovce, v příkazové řádce nepoužíváme myš pro interakci. K napsání příkazů nám stačí klávesnice. Je zde také možnost procházet předešlé příkazy pomocí šipky nahoru a dolů. Jediná možnost přizpůsobit si příkazovou řádku je skrze kliknutí na záhlaví aplikace, kde přes volbu vlastnosti je možné upravit základní vzhled prostředí. Konkrétně můžeme měnit barvy textu, pozadí a velikost textu. Jedná se tedy o velice jednoduchou aplikaci na pochopení, o to více má ale funkcí a možných příkazů. Ukázka prostředí příkazového řádku je na obr. 2.1.

2.2 Ping

Ping můžeme považovat za program nebo příkaz, kterým testujeme dostupnost a konektivitu mezi dvěma síťovými prvky, kterými mohou být například počítače nebo směrovače. Příkaz ke své funkci používá protokol ICMP. Obvykle příkaz zadáváme v příkazovém řádku a jako odpověď očekáváme zprávu, že spolu zařízení dokáží komunikovat. Jak můžeme vidět na obr. 2.1, je zde také zobrazena časová prodleva (latence), s jakou se nám dostalo odpovědi. Tento časový údaj je zobrazován v milisekundách.

Tento příkaz je možné zadávat ve většině operačních systémů, respektive v jejich příkazových řádcích. Existuje mezi nimi několik rozdílů a to hlavně v počtu dostupných volitelných parametrů. Zaměříme se na rozdíly mezi operačními systémy Windows a Linux. První rozdíl je ihned po napsání prostého příkazu ping bez volitelných parametrů. U OS Windows se vypíší 4 echo odpovědi, kdežto u Linuxu nám běží výpis do té doby, než jej manuálně přerušíme pomocí Ctrl+C. Dalším rozdílem je zaokrouhlování časové prodlevy. Windows nám tuto hodnotu zaokrouhluje dolů, na celá čísla. Linux tuto hodnotu ponechává v originálním formátu s přesností na tisíciny milisekund, stejně jako Wireshark, kde tuto informaci také můžeme vyhledat. V případě, že Wireshark zjistí časovou prodlevu například 0,637 ms, tak Windows vypíše pouze informaci o tom, že prodleva byla menší než 1 ms, jak je vidět na obr. 2.1. [9]

```
Administrator: Příkazový řádek
                                                         П
                                                             ×
Microsoft Windows [Version 10.0.18362.476]
(c) 2019 Microsoft Corporation. Všechna práva vyhrazena.
C:\WINDOWS\system32>ping vutbr.cz
Pinging vutbr.cz [147.229.2.90] with 32 bytes of data:
Reply from 147.229.2.90: bytes=32 time=2ms TTL=60
Reply from 147.229.2.90: bytes=32 time<1ms TTL=60
Reply from 147.229.2.90: bytes=32 time<1ms TTL=60
Reply from 147.229.2.90: bytes=32 time<1ms TTL=60
Ping statistics for 147.229.2.90:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = Oms, Maximum = 2ms, Average = Oms
C:\WINDOWS\system32>_
```

Obr. 2.1: Průběh příkazu ping v okně příkazové řádky operačního systému Windows, včetně čtyř řádků výpisu o přijetí ICMP zprávy echo reply a také shrnující statistika.

Další rozdíly jsou ve volitelných parametrech. Některé jsou podobné, jen se liší jejich zápis, ale Linux nabízí některé parametry, které u Windows nenajdeme. V příkazové řádce na Linuxu můžeme libovolně upravovat časový interval, mezi jednotlivými odeslanými žádostmi echo request pomocí parametru -i. U Windows je tato hodnota nastavena na 1 sekundu a nelze ji měnit. Naopak oba systémy disponují parametrem -w, který určuje dobu, po kterou se čeká na odpověď echo reply. Bohužel na OS Windows 10 se vyskytuje nespecifikovaná chyba, která tento parametr při zadání za příkaz ping ignoruje a výpis nijak neovlivňuje. Linux navíc umožňuje rozlišení tohoto parametru (-w, -W), kdy první z nich vypíše chybu ihned po vypršení nastaveného času, druhý vypíše chybu pouze v případě, že nedostane žádnou odpověď od cílové stanice.

Linux také umožňuje vyplnit přidaná data požadovanými hodnotami (-p ff vyplní data přidaná k pingu pouze jedničkami). Další volitelný parametr je -a, který určuje příkazové řádce, že při každém přijetí odpovědi echo reply má vydat zvuk na výstupu a -A upravuje interval mezi jednotlivými echo requesty podle celkové odezvy. [10]

Posledním rozdílem, který bude uveden je velikost dat, které se odesílají spolu se záhlavím ICMP. Windows má základní velikost stanovenou na 32 bajtů, spolu s osmi bajty ICMP záhlaví tedy 40 B. Linux má základní hodnotu nastavenou na 56 bajtů, celkem nese protokol ICMP 64 bajtů. U obou systémů lze pomocí parametru velikost měnit.

2.2.1 Průběh příkazu ping

Příkaz ping má jeden povinný parametr a tím je IP adresa, případně doménové jméno zařízení v síti, se kterým se chceme spojit. V případě že zadáme doménové jméno, provede se nejdříve překlad na IP adresu pomocí DNS a následně už je cesta shodná. Klient odešle výzvu v podobě ICMP echo request na adresu serveru. Server zprávu obdrží a obratem posílá odpověď echo reply protokolu ICMP. Když klient tuto odpověď obdrží, tak nám příkazový řádek vypíše na každý řádek jednu přijatou odpověď a časovou prodlevu. V případě, že nezadáme žádný další parametr, měly by se v případě úspěšného spojení zobrazit čtyři odpovědi. Na konec se vypíše statistika o průběhu příkazu s informace o minimální, maximální a průměrné době odezvy a také počet odeslaných a přijatých paketů. Z této informace se ještě vypočte ztrátovost (loss) na cestě k požadovanému serveru, která je uváděna v procentech. Ukázka průběhu příkazu ping je uvedena na obr. 2.2.

	*Etherne	t													_		×	<
Fi	ile Edit	View	Go	Captu	re Ar	nalyze	Statistic	s Te	lephon	y Wir	eless	Tools	Help					
		•	010	🗙 🖸	9	÷ =	2 🔮 👔	· 🔒		•	Θ. 🤅	1						
	ip.addr.ed	147.22	29.197.3	0 and (d	dns or id	:mp)								X	D T	xpression	ı	+
No	. Time		Source			Destin	ation		Proto	Length	Info							^
	11.0	85973	147.2	29.19	7.30	147.	229.190	.143	DNS	68	Stan	dard q	uery Øx	e423 A	vutbr.	cz		
	11.0	86343	147.2	29.19	7.30	147.	229.190	.143	DNS	68	Stan	dard q	uery 0x	1e75 AA	AAA vut	br.cz		
	11.0	86582	147.2	29.19	0.143	147.	229.197	.30	DNS	172	Stan	dard q	uery re:	sponse	0xe423	A vu		
	11.0	86739	147.2	29.19	0.143	147.	229.197	.30	DNS	125	Stan	dard q	uery re:	sponse	0x1e75	AAAA		
Г	11.0	93159	147.2	29.19	7.30	147.	229.2.9	0	ICMP	74	Echo	(ping) reque:	st id=	=0x0001	, seq		
	11.0	93780	147.2	29.2.9	90	147.	229.197	.30	ICMP	74	Echo	(ping) reply	id=	=0x0001	, seq		
-++	12.0	96721	147.2	29.19	7.30	147.	229.2.9	0	ICMP	74	Echo	(ping) reque	st id=	0x0001	, seq		
-	12.0	97238	147.2	29.2.9	90	147.	229.197	.30	ICMP	74	Echo	(ping) reply	id=	=0x0001	, seq	<u> </u>	
	13.10	30090	147.2	29.19	7.30	147.	229.2.9	0	ICMP	74	Echo	(ping) reque	st id=	=0x0001	, seq		
	13.10	00787	147.2	29.2.9	90	147.	229.197	.30	ICMP	74	Echo	(ping) reply	id=	=0x0001	, seq		
	14.10	04504	147.2	29.19	7.30	147.	229.2.9	0	ICMP	74	Echo	(ping) reque:	st id=	=0x0001	, seq		
	14.10	06169	147.2	29.2.9	90	147.	229.197	.30	ICMP	74	Echo	(ping) reply	id=	=0x0001	, seq		Υ.
~	Frame 1	12824:	74 b	/tes o	n wir	e (59)	2 bits)	, 74	bytes	captu	red (592 bi	ts) on	interfa	ace Ø			^
	> Inte	rface	id: 0) (\De	vice\	NPF_{F	04E56E	F-BC6	6-4E37	7-9509	-0778	9831D4	21})					
	Enca	psula	tion 1	ype:	Ether	net (1	L)											
	Anni	val T	ime: [)ec 1	, 201	9 21:5	53:41.0	54355	000 St	třední	Evro	pa (bě	žný čas)				
	[Tin	ne shi	ft for	• this	pack	et: 0	.000000	000 s	econds	5]								
	Epoc	h Tim	e: 157	52336	21.05	435500	00 seco	nds										
	[Tin	ne del	ta fro	om pre	vious	captı	uned fr	ame:	0.000	517000	seco	nds]						
	[Tin	ne del	ta fro	om pre	vious	displ	layed f	rame:	0.000	951700	0 sec	onds]						

Obr. 2.2: Zachycení procesu získání IP adresy pomocí DNS protokolu a následný ping a zobrazení ICMP paketů. Zároveň je v dolní části vidět latence v sekundách.

2.2.2 Parametry pro ping

K povinnému parametru IP adresy nebo doménového jména můžeme přidávat i parametry nepovinné. Tyto volitelné parametry se většinou píší za příkaz ping a pro jejich rozpoznání je píšeme s prefixem mínus (-) nebo lomítkem (/). V případě, že parametrů chceme použít více, oddělujeme je pouze mezerou. Parametry nám mohou pomoci při řešení problémů s konektivitou a v našem případě nám pomohou při analýze funkcí a chybových hlášení některých protokolů. Příklad zápisu příkazu ping můžeme vidět níže a volitelné parametry shrnuje tabulka 2.1. [11]

ping <volitelné parametry> <IP adresa nebo doménové jméno>
ping -4 -t vutbr.cz

	Ping se bude opakovat dokud nebude manuálně zastaven									
	(Ctrl+C), zároveň lze u každé odpovědi pomocí Ctrl+Break									
-t	zobrazit statistiky o počtu odeslaných a přijatých paketů, z toho									
	lze odvodit ztrátovost (packet loss) a také maximální, minimální									
	a průměrná doba odezvy.									
-n <číslo>	Kde číslo určuje počet odeslaných echo requestů.									
	V případě, že budeme provádět ping na IP adresu, tak nám tento									
-a	parametr vrátí i doménové jméno, ke kterému tato adresa patří.									
	Kromě ICMP protokolu se tak využije i protokol DNS.									
	Lze nastavit velikost dat, které se s pingem odešlou, toto můžeme									
1 (využít když řešíme problém se sítí. Pakety s malou velikostí (běžně									
-1 < venkost >	se posílají 32 bajtové) můžou sítí projít, ale s většími objemy dat									
	může být při přenosu problém nebo se může lišit doba odezvy.									
£	Parametr určí, že chceme aby se pakety nefragmentovaly (Do not									
-1	Fragment flag). Použitelné pouze pro IPv4.									
i căialas	Parametr omezí ping na zadaný počet přeskoků TTL (Time To									
-1 < cisio >	Live) viz kapitola o TTL.									
-4	Parametr vynutí použití IPv4.									
-6	Parametr vynutí použití IPv6.									
2	Parametr vypíše tyto a také další, méně používané parametry pro									
- !	ping.									

Tab. 2.1: Volitelné parametry pro příkaz ping.

2.3 Traceroute

Traceroute je nástroj pro síťovou diagnostiku. Dokáže zjistit cestu paketu od zdrojového zařízení až po koncovou stanici v síti. Ve výpisu se tedy objeví čas odezvy jednotlivých prvků v síti, kterými paket prochází, adresy a případně také doménová

jména prvků, které jsou zjistitelné přes DNS server podobně jako u pingu. Příkaz je dostupný na unixových systémech jako traceroute a také na operačním systému Windows jako příkaz tracert, který je zobrazen na obr. 2.3.

tracert <IP adresa nebo doménové jméno>

tracert vutbr.cz

```
Administrator: Příkazový řádek
                                                                          ×
C:\WINDOWS\system32>tracert -4 seznam.cz
Tracing route to seznam.cz [77.75.75.176]
over a maximum of 30 hops:
        6 ms
                 1 ms
                          1 ms irf-kn40.kn.vutbr.cz [147.229.196.1]
 2
       1 ms
                 1 ms
                          1 ms pe-meo.net.vutbr.cz [147.229.252.77]
 3
       1 ms
                <1 ms
                         <1 ms pe-ant2.net.vutbr.cz [147.229.253.58]
 4
                         <1 ms rt-ant.net.vutbr.cz [147.229.253.233]
       <1 ms
                <1 ms
 5
       <1 ms
                <1 ms
                         <1 ms r98-bm.cesnet.cz [147.229.252.17]
 6
       5 ms
                 7 ms
                          7 ms 195.113.235.99
 7
                          4 ms nix2.seznam.cz [91.210.16.194]
        4 ms
                 3 ms
 8
        4 ms
                 4 ms
                          4 ms n7k-ng-a-vdc-1-po1.seznam.cz [185.66.188.9]
 9
        4 ms
                 4 ms
                                n7k-ng-a-vdc-2-po3.seznam.cz [185.66.188.21]
                          4 ms
                          3 ms www.seznam.cz [77.75.75.176]
 10
        3 ms
                 4 ms
Trace complete.
```

Obr. 2.3: Zobrazení průběhu příkazu tracert s kompletním výpisem procházených prvků v síti.

Stejně jako ping i traceroute využívá ICMP zprávy, ovšem zde se pro každý směrovač upravuje hodnota TTL. Zdrojový počítač odešle ICMP echo request zprávu s hodnotou TTL 1. Tento paket se dostane na výchozí bránu, kde je hodnota TTL změněna na 0, takže nemůže pokračovat dále. Výchozí brána nám o tom dá vědět pomocí chybové zprávy *"Time to live exceeded"*. Z této zprávy zjistíme jak IP adresu, kterou vidíme ve výpisu, tak čas odezvy k tomuto síťovému prvku. Následně se odešle opět ICMP echo request, tentokrát s hodnotou TTL 2. Hodnota TTL se takto inkrementuje až do doby, kdy se nevrátí chybová zpráva TTL, ale ICMP echo reply. Zde již probíhá komunikace stejně jako u pingu a tím zdrojový počítač zjistil, že komunikuje s cílovým prvkem. Maximální hodnota, do které se TTL inkrementuje, je 30 přeskoků. Komunikace je znázorněna na obrázku 2.4 níže. Na unixových operačních systémech ve výchozím nastavení neprobíhá komunikace přes protokol ICMP, nýbrž přes UDP. [12]



Obr. 2.4: Znázornění komunikace zdrojového počítače a síťových prvků při zadání příkazu traceroute.

2.4 Nslookup

Nslookup je utilita implementující funkce protokolu DNS v příkazovém řádku. Umožňuje nám zajistit překlad IP adres na doménová jména a naopak pomocí protokolu DNS. Tento nástroj je podobně jako ping dostupný na známých operačních systémech Windwos i Linux. Příkaz se zadává pouze s jedním parametrem a tím je právě IP adresa nebo doménové jméno. V případě, že chceme nalézt IP adresu doménového jména, tak nám příkaz vypíše všechny dostupné adresy IPv4 i IPv6. Ve výpisu se také ukazuje použitá IP adresa DNS serveru, jak můžeme vidět na ukázce příkazu na obr. 2.5.

```
nslookup <IP adresa nebo doménové jméno>
nslookup seznam.cz
```

Obr. 2.5: Výpis příkazu nslookup v příkazové řádce při překladu doménového jména na dostupné IPv4 a IPv6 adresy.

2.5 DNS resolver

Program DNS resolver disponuje jednoduchým grafickým rozhraním, které je zobrazeno na obr. 2.6 Umožňuje nám provádět DNS dotazy, které následně umí zpracovat a přijatá data nám zobrazí. Jedná se vlastně o překladač IP adres s několika nastavitelnými parametry. Vstupem od uživatele je doménové jméno, adresa DNS serveru je již přednastavená. Dále může uživatel zvolit zda chce nalézt IPv4 nebo IPv6 adresu a může také zvolit typ DNS záznamu, který následně obdrží. Na výběr je také transportní protokol, pomocí kterého se budou zprávy přenášet, můžeme zvolit TCP nebo UDP protokol. [18]

DNS resolver				-
Menu				
seznam.cz	147.229.190.143	● IPv ²	 UDP TCP 	Send No_Error
IN Y	fec0:0:0:ffff::1%1] ⊖ IPv€	✓ Recursior ✓ DnsSec	
 ♦ Header Id: 0 Qdcount: 1 Ancount: 3 Nscount: 0 Arcount: 1 ♥ Flags ♦ Querry Qname: seznam.cz Qtype: A Qclass: IN 		Answer to: s Type: A Class: IN Time to live: 00:0 Data Length: 4 Data: 77.75.75.1 Answer to: s Type: A Class: IN Time to live: 00:0 Data Length: 4 Data: 77.75.75.1	eznam.cz a 04:44 72 eznam.cz a 04:44 76	Answer to: <root> Type: 41 Class: 4096 Time to live: 09:06:08 Data Length: 0</root>

Obr. 2.6: Jednoduché grafické prostředí programu DNS Resolver, který umožňuje překlad doménových jmen na IP adresy.

3 Protokoly IP přenosu hlasu

Ve druhém scénáři bude potřeba uskutečnit hovor mezi dvěma uživateli. Proto se v této části rozebere postup k uskutečnění hovoru. Hovor bude využívat internetovou síť a koncovými zařízeními budou pouze počítače. V pokročilejších řešeních je možné vytvořit celou síť, kdy spolu mohou komunikovat počítače a telefonní zařízení, například v jedné firmě. Výhodou je bezplatná komunikace uživatelů v této firemní síti. Ve vytvořeném scénáři si však vystačíme s komunikací dvou uživatelů přes počítač, respektive virtuální operační systém.

Hovory jsou nazývány VoIP (z anglické zkratky Voice over Internet Protocol - hlas přenášený Internet protokolem). Kromě Internet protokolu se využívají také transportní protokoly UDP nebo TCP pro sestavení hovoru. K sestavení VoIP hovoru, kdy volající čeká než volaný přijme hovor, se využívá SIP protokol (Session Initiation Protocol – protokol pro inicializaci relací). Poté přichází na řadu RTP protkol (Real-time Transport Protocol), který obstarává přenos dat se zaznamenaným hlasem. RTP protokol pro přenos využívá UDP protokol. Dvěma základním protokolům pro fungování VoIP hovorů se věnují následující podkapitoly č. 3.1 a 3.2.

3.1 Session Initiation Protocol (SIP)

Session Initiation Protocol je kontrolní protokol, pracující na aplikační vrstvě. Je určen k vytváření a ukončování relací, jako jsou VoIP hovory nebo multimediální konference, mezi jedním nebo několika uživateli. Účastníci si pomocí SIP protokolu dohodnou, jaký kodek budou při hovoru používat. V průběhu hovoru může SIP protokol přidat dalšího účastníka do hovoru, může dojít ke změně kodeku v průběhu hovoru a SIP také zprostředkovává přidržení hovoru. [13]

3.1.1 Navázání a ukončení SIP spojení

Pro sestavení hovoru, které je znázorněno na obr. 3.1, je nutné propojit obě komunikující strany. Ty jsou identifikovány podle jména a domény, kterou využívají (např. sip:bob@domain.com). Volající tedy vysílá požadavek INVITE na SIP proxy server, který ví že Bob má danou IP adresu a na tuto adresu přepošle INVITE zprávu. Tím se spustí zvonění na straně volaného a volající přijímá zprávu 180 Ringing (zvonící). Zpráva 180 indikuje úspěšný průběh komunikace s tím, že se ještě čeká na nějaký krok pro sestavení. Musí se vyčkat na přijetí hovoru druhou stranou. Poté je vyslána zpráva 200 OK, která znamená úspěšné spojení. Volající ještě potvrdí spojení (ACK) a může začít přenos hlasové komunikace.



Obr. 3.1: Průběh VoIP hovoru včetně sestavení a ukončení komunikace pomocí SIP protokolu a přenos multimediálních dat, většinou uskutečňovaný protokolem RTP.

To byla varianta, kdy vše probíhá bez problémů. V případě chyby na jedné ze stran, se vyšlou odlišné číselné kódy podle tabulky 3.1. Například, když volaný nedisponuje požadovaným kodekem, který navrhl volající, tak místo zprávy s indikací 200 OK odešle chybový kód 606 "Not Acceptable" (v překladu "nepřijatelné" - strana nemůže přijmout hovor, když nemá jak zakódovat přenášenou hlasovou komunikaci. Zároveň také pošle seznam kodeků, kterými disponuje. Další situace může nastat, když volaný hovor odmítne. Pro tuto situaci je hned několik kódů, označovaných jako "busy here". Například může být zaneprázdněn jiným hovorem (486) nebo je vyžadována platba pro spojení hovoru (402). [3]

Tab. 3.1: Tabulka zobrazující výčet možných číselných kódů protokolu SIP. Chybu znamenají kódy začínající číslicí 4,5 nebo 6.

kód	popis kódu
1XX	Požadavek se úspěšně zpracovává, ale ještě není ukončen.
2XX	Úspěšné vyřízení žádosti. (Nejčastěji 200)
3XX	Pro dokončení požadavku je nutné hovor přesměrovat.
4XX	Chyba na straně klienta.
5XX	Chyba na straně serveru.
6XX	Fatální chyba, kterou nelze zpracovat.

3.2 Real-time Transport Protocol (RTP)

RTP je protokol, který umožňuje přenášet v reálném čase data, jako jsou video a zvuk. Sám o sobě nezaručuje doručení přenášených paketů, stejně jako UDP protokol, na kterém je RTP postaven. Každý RTP paket ale obsahuje pořadové číslo a také časové razítko, které pomáhají při rekonstrukci hovoru ve správném pořadí na přijímající straně hovoru. Přijímacích stran může být víc, RTP totiž umožňuje přenos dat mezi více uživateli prostřednictvím IP multicastu. [14]

RTP protokol je využíván ve spojení s protokolem RTCP (RTP Control Protocol). RTP přenáší samotná multimediální data a RTCP zajišťuje kvalitu služeb (QoS) a také předává informace o účastnících aktuální relace. Záhlaví protokolu RTP obsahuje 12 bajtů dat a přenášená data jsou v jednom paketu zastoupena 160 bajty. Těmto 160 bajtům se říká "chunky", což jsou zakódované kusy hlasových dat, které jsou po určitém časovém intervalu (několik milisekund) rozsekány na části - chunky, které se distribuují pomocí RTP protokolu. [3]

4 Protokoly zajišťující bezpečený přenos dat

4.1 Transport Layer Security (TLS)

TLS je protokol zajišťující zabezpečenou komunikaci mezi klientem a serverem. Konkrétně protokol zajišťuje autentizaci serveru a to pomocí asymetrických šifer, jako je například RSA (iniciály autorů - Rivest, Shamir, Adleman). Dále TLS zajišťuje důvěrnost a integritu, takže posílaná data jsou přístupná pouze koncovým prvkům, které jsou autentizovány a integrita znamená odhalení případné změny dat během přenosu. [21]

Na začátku komunikace se musí klient a server domluvit na tom, který typ šifrování využijí. Následně si bezpečnou cestou vymění klíče a také se provede autentizace pomocí certifikátů. Poté již probíhá zabezpečená komunikace, která se šifruje pomocí dříve vyměněného symetrického klíče.

TLS je nejvíce využíváno pro bezpečnou komunikaci v rámci webových stránek, kdy se z nezabezpečené stránky (HTTP - Hypertext Transfer Protocol) stane stránka zabezpečená (HTTPS - HTTP Secure). TLS může být také využito u mailových klientů (SMTP - Simple Mail Transfer Protocol) nebo pro zabezpečení protokolu SIP u VoIP hovorů.

4.2 Quick UDP Internet Connections (QUIC)

QUIC je relativně nový protokol, který byl na veřejnost uveden společností Google v roce 2013. Ten je dnes označován jako gQUIC a je součástí protokolu HTTP/2. Od té doby se očekává, že QUIC bude novým webovým standardem, který bude spravován organizací IETF (Internet Engineering Task Force) a stane se součástí nového protokolu HTTP/3.

Jedná se o zabezpečený protokol a úroveň bezpečnosti je srovnatelná s protokolem TLS. Hlavním cílem je zrychlit přístup k webovým službám. Na rozdíl od protokolu TLS, který využívá TCP je QUIC postaven na protokolu UDP. Výhoda QUICu se uplatní hlavně při opětovném připojení klienta na server, který již někdy využíval. Při komunikaci se použijí klíče, které si strany vyměnily dříve a není tak nutné tento proces opakovat a tak se požadovaná data přenášejí hned v prvním paketu. V případě, že jde o prvotní komunikaci, tak se prvními pakety navazuje spojení a také se přenášejí klíče k ustanovení zabezpečeného kanálu. Požadovaná data se pak přenášejí od třetího paketu dále.

Další výhodou QUICu je možnost reagovat na změnu adres při již navázaném spojení. To je možné díky identifikaci spojení podle tzv. "Connection ID", které je

součástí záhlaví protokolu. QUIC nabízí dvě varianty záhlaví, dlouhou a krátkou. Rozboru záhlaví se věnuje jedna z podkapitol druhého scénáře. [23]

Protokol je již několik let využíván v prohlížeči Chrome a to především při komunikaci se servery Google jako je Youtube nebo samotný Google vyhledávač. Postupně se s podporou QUICu přidávají také další webové prohlížeče. [22]

5 Další použité programy

V rámci připravovaných počítačových cvičení bude využita virtualizace operačního systému. Hlavní výhodou tohoto systému bude, že studenti budou mít plná práva pro ovládání tohoto systému. Na systému budou nainstalovány všechny potřebné programy, které se budou v rámci cvičení používat. Tyto programy jsou popsány v následujících podkapitolách.

5.1 VMware a virtualizovaný operační systém

Virtualizace je pojem, kdy na jednom fyzickém zařízení a na jednom hostovském operačním systému může zároveň fungovat několik dalších virtuálních zařízení. Na jednom hostujícím zařízení můžeme současně spustit virtualizovaný operační systém Windows a třeba i druhý virtualizovaný operační systém Linux. Virtualizace v připravených počítačových cvičeních bude provedena virtualizačním programem VMware Workstation a vytvořený virtualizovaný operační systém bude 64-bitový Windows 10 Pro, verze 1809 (build 17763). Scénáře budou vytvářeny a testovány v programu VMware Workstation 15 Player (verze 15.5.0). Vytvořeným virtuálním operačním systémům můžeme v nastavení VMware přerozdělit hardwarové prostředky fyzického počítače. V průvodci vytvoření systému si zvolíme, kolik operační paměti a místa na pevném disku si může VMware "ukousnout" pro daný virtualizovaný systém. Program také nabízí více možností pro různé typy připojení k síti v rámci vytvořeného systému. Virtualizace umožňuje bez omezení pracovat na systému, kde můžeme mít plná práva pro ovládání tohoto systému a zároveň můžeme testovat různá řešení, která nám systém dovolí. Bohužel používaná verze VMware neumožňuje vytváření snapshotů, což jsou snímky, které fungují jako záloha a v případě potřeby se můžeme vrátit do předchozího uloženého stavu. Program je vhodný pro výuku nových operačních systémů nebo testování a zjištování možností různých programů, jako v případě této práce. Student bude mít připravený obraz s výchozím stavem pro danou úlohu a následně po spuštění může se systémem pracovat jako s běžným počítačem, s možností kdykoliv začít opět z výchozího stavu. Zároveň budou na předem připraveném operačním systému dostupné všechny potřebné programy pro zvládnutí vytvořeného scénáře. Ukázka virtualizovaného systému včetně používaných programů je na obr. 5.1. [15]

😼 Windows 1	0 x64 ·	- VMv	vare Workstation	n 15 P	Player (Non-commercial	use or	nly)							- 0	×
Player 👻	•	4	99										≫ [a 📀 🗟 🖨 🍕	0 🗗
Koś	▲ * File	Etherr Edit	view Go	Cap	ture Analyze Statisti	s ⊺ ₹ <u>₹</u>	Telephony Wire	less Q. Q.	Tools Help	p				- 0	×
	Ap	oply a c	display filter <	Strl-/:	>			-						Expression.	
Microsoft Edge	No.	49 50 51 52 53	Tme 15.694091 16.724659 16.725652 17.739627 17.740578	S 1 1 1 1 1	ource 47.229.2.90 92.168.98.134 47.229.2.90 92.168.98.134 47.229.2.90	Des 192 147 192 147 192	tination 2.168.98.134 7.229.2.90 2.168.98.134 7.229.2.90 2.168.98.134		Protocol L ICMP ICMP ICMP ICMP ICMP	ength Info 74 Echo 74 Echo 74 Echo 74 Echo 74 Echo 74 Echo	(ping) reply (ping) request (ping) reply (ping) request (ping) reply	id=0x0001, id=0x0001, id=0x0001, id=0x0001, id=0x0001,	, seq=2/512, ttl=128 (request in 48) , seq=3/768, ttl=128 (reqly in 51) , seq=3/768, ttl=128 (request in 59) , seq=4/1024, ttl=128 (reqly in 53) , seq=4/1024, ttl=128 (request in 52)		
Wireshark	III Men	DNS n	esolver								-	D X	Microsoft Windows [Version 10.0.1776]	-	
[©] >− [™]	vut IN	br.cz	A	v	8.8.8.8 fec0:0:0:ffff::1%1		 IPv4 IPv6 		 UDP TCP Recursion DnsSec 	Send	No_Error		<pre>(c) 2018 Microsoft Corporation. vsect C:\Users\student>ping -4 vutbr.cz Pinging vutbr.cz [147.229.2.90] with</pre>	32 bytes of data	a:
Pinkazový řádek DNSresolver	Id:1 Qdi Ano Nso Arc) Head 0 loount count: count: ount:) Flags	der :1 :2 0 1				Answer Type: A Class: IN Time to live Data Lengt Data: 147.2 Answer	to: vut : 00:01: h: 4 29.2.90 to: vut	br.cz a 48 br.cz msig	Answ Type: 41 Class: 51 Time to I Data Len	verto: <root> 2 ive:09:06:08 gth:0</root>		Reply From 147.229.2.90: bytes=32 tir Reply from 147.229.2.90: bytes=32 tir Reply from 147.229.2.90: bytes=32 tir Reply from 147.229.2.90: bytes=32 tir Ping statistics for 147.229.2.90: Packets: Sent - 4, Received - 4, Approximate round trip times in milli	ne-ins TTL-128 ne-ins TTL-128 ne-ins TTL-128 ne-ins TTL-128 Lost - 0 (0% los L-seconds:	55),
Google Chrome	() Qni Qtj Qcl) Quer ame: \ /pe: A lass: IN	ny vutbr.cz N				Type: RRSIC Class: IN Time to live Data Lengti Type cover Algorithm: Labels: 2 Original T	: 00:01: h: 156 kd: A RSA_SH	48 IA1	Done			runumum = ims, Neximum = ims, Ave C:\Users\student>	rage = 1ms	
		e	= 9		🦲 🖻 🗖								^ †	⊒ ¢») CES 11:25 18/12/20	19 🖵

Obr. 5.1: Okno programu VMware se spuštěným virtualizovaným operačním systémem Windows 10 s programy, které se využívají ve vytvořeném scénáři.

5.2 Wireshark

Wireshark je síťový protokolový analyzér a paketový sniffer. Pomocí tohoto programu můžeme zachytávat provoz na síti a následně jej prohlížet. A to vše přehledně, díky podbarvení jednotlivých protokolů ve výpisu, čímž dojde k oddělení a zpřehlednění zachyceného síťového provozu. Můžeme také provoz filtrovat nebo ukládat pro pozdější analýzu a to v mnoha formátech, například tcpdump, Pcap NG, Microsoft Network Monitor a mnoho dalších. V případě, že máme uložené zachycené relace paketů, můžeme několik těchto relací spojit do jedné relace v jednom okně pro jednodušší procházení podrobností o paketech. Wireshark také umožňuje stažení různých profilů z internetu. Profily upravují zobrazované informace o různých paketech. Takže můžeme najít profily, které vylepšují procházení a usnadňují pochopení funkce určitých protokolů. Díky těmto funkcím se jedná o jeden z nejpoužívanějších programů ve své kategorii. Uplatnění najde jak u expertů na síťový provoz, tak u vývojářů nebo jako v našem případě pro výuku a názorné ukázky funkce jednotlivých protokolů. Program je volně dostupný na operačních systémech Windows, Linux a macOS jako open source pod licencí GNU General Public License verze 2. Popisu samotného prostředí programu se věnuje jedna z částí prvního scénáře, včetně obr. A.3, který popisuje jeho hlavní obrazovku. [16]

5.3 Linphone

Linphone je open source program, který umí realizovat vzdálené hovory prostřednictvím internetu (VoIP). Kromě přenosu hlasu je v programu také možnost psát textové zprávy a k hlasovému hovoru je umožněno také přidat přenos videa z webkamery. Tyto dvě možnosti ale ve scénáři nebudou využity a bude se přenášet jen zvuková stopa v kodeku PCMU (Pulse-code modulation μ -law – pulzně kódová modulace μ -law).

Program je možné získat zdarma v původní podobě na oficiálních stránkách Linphone¹ pod GNU GPLv2 licencí. Původní podobou se myslí kompletní program, kde je možná také registrace uživatele. Dále vývojáři nabízí program v distribuci přes Microsoft Store. Tam je program nabízen v omezenější podobě, v jednodušším grafickém provedení a není zde možnost registrace uživatele. Registrace musí být provedena přes oficiální stránky Linphone. Jinak ale tato podoba nabízí stejné možnosti nastavení kodeků a zabezpečení hovorů jako prvně zmiňovaná podoba programu. Poslední možností je mobilní aplikace, která je také zdarma ke stažení jak na Google Play, tak v App store. V té je umožněna registrace a také široká škála nastavení, stejně jako v desktopové verzi programu. [17]

Pro účely vypracovávání scénáře je také vhodné, že program nabízí možnost hovor šifrovat nebo nešifrovat. Bude tak možné porovnat obě varianty, kdy v prvním případě ve Wiresharku nepůjde hovor rekonstruovat a ve druhém případě půjde zachycená zvuková stopa přehrát.

Popisu programu a jeho prostředí se věnuje jedna z částí druhého scénáře.

¹http://www.linphone.org/technical-corner/linphone?qt-technical_corner=2#qt-technical_corner

🖸 Linphone			— (⊐ ×		
xsimaj01 sip:xsimaj01@sip.linphone.org	Search contact, start a call or a chat		a 🙀			
🖒 номе		 Linphone Registered 			- 0	I X
A CONTACTS		Number or address				$\langle \times$
Previously						
xsimaj01 sipxsimaj01@sip.linph		1 ∞	<mark>2</mark> АВС		3 Def	
		4 GHI	5 JKL		6 MNO	
		7	8		9	
	INVITE YOUR FRIENDS	PQRS	TUV		WXYZ	
	Invite your friends on Linphone.	*	0		#	
		· · ·	÷		;	
	INVITE		چ Call			
			©	8		

Obr. 5.2: Úvodní obrazovky dvou dostupných podob programu Linphone. Vlevo originální podoba a vpravo adaptace programu Linphone dostupná přes Microsoft Store.

6 Průzkum dostupných úkolů a řešená problematika při vytváření scénářů

V rámci přípravy pro vytváření scénářů byla provedena rešerše již existujících úloh, které se nachází na internetu. Většina těchto návodů na internetu se vyskytuje v podobě videí a z hlediska obsahu se jedná o kratší a jednoduché návody. Mezi nejoblíbenější kategorie patří ukázka funkce protokolu TCP, kde je vysvětleno, jak se každý TCP paket potvrzuje paketem s příznakem ACK atp. [24]

Další návody jsou zaměřeny například na získání přihlašovacího jména a hesla při přihlašování na FTP server. Počítač při komunikaci se serverem využívá protokol FTP, který přenáší tyto údaje v otevřené, nezašifrované podobě. [25] [26] A další variantou jsou úlohy, kde je pouze úkol zjistit IP adresy serveru a vlastní adresu při použití protokolu DNS. [28] Každopádně žádné z nalezených návodů nedosahují požadované složitosti, rozmanitosti použitých protokolů ani rozsahu dvou hodin práce pro studenty. Tyto návody také neobsahují žádné doplňující otázky, které by navazovaly na vysvětlenou problematiku.

U vypracovávání a testování postupu se vyskytly určité problémy, které bylo nutné překonat. Popis a použité řešení těchto problémů jsou popsány dále.

6.1 Hodnota TTL

U prvního scénáře se vyskytl problém s tím, že DNS záznamy byly uchovávány v dočasné paměti pouze po velmi krátkou dobu (přibližně 5 sekund). Což znamenalo, že se po zadání příkazu **ipconfig** -displaydns nevypsaly požadované záznamy. Zobrazí se pouze záhlaví výpisu a chyba "could not display the DNS Resolver Cache", tak jako na obr. 6.1. Problém způsobovalo to, že nebyla nastavena pevná IP adresa pro DNS dotazy v nastavení síťového adaptéru. Operační systém tedy získal adresu automaticky a použil adresu výchozí brány, kterou je v našem případě hostitelský operační systém. Ten nám poskytl doménové jméno v paketu DNS response s nízkou hodnotou TTL. V případě, že IP adresu DNS serverů nastavíme ručně (například na DNS servery od Googlu s adresami 8.8.8.8 a 8.8.4.4), tak se problém vyřeší. Operační systém použije jednu z námi definovaných adres a obdržené DNS záznamy mají běžnou hodnotu TTL (v řádu několika minut). Výpis příkazu **ipconfig** -displaydns lze použít opakovaně v rámci několika minut a studenti stále uvidí uložené DNS záznamy.

d Capturing from Ethernet0	- 🗆	х	💷 Vybrat Příkazový řádek
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools	Help		C:\Users\student>ping seznam.cz
			Pinging seznam.cz [77.75.75.172] with 32 bytes of data:
R dns	Expression	. +	Reply from 77.75.75.172: bytes=32 time=4ms TTL=128
No. Time Source Destination Protocol Length	Info	^	
1 0.000000 192.168.171.129 192.168.171.2 DNS 69	Standard query Øxe942 A seznam.cz		Ping statistics for 77.75.75.172:
	Standard query response 0xe942 A sezna	in_	Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
9 57.400176 192.168.171.129 192.168.171.2 DNS 73	Standard query Øxdd10 A fp.msedge.net	¥	Approximate round trip times in milli-seconds:
Internet Protocol Version 4, Src: 192.168.171.2, Dst: 192.168.171.1	29	~	Minimum = 4ms, Maximum = 4ms, Average = 4ms
> User Datagram Protocol, Src Port: 53, Dst Port: 50174			Control-C
🝸 Domain Name System (response)			^C
Transaction ID: @xe942			C:\Users\student>ipconfig -displaydns
> Flags: 0x8180 Standard query response, No error			
Questions: 1			Windows IP Configuration
Authority DBc: 0			
Additional BBs: Ø			
✓ Oueries			C:\Users\student\inconfigdisplaydns
✓ seznam.cz: type A, class IN			a fore a face and the analysis
Name: seznam.cz			Windows IP Configuration
[Name Length: 9]			
[Label Count: 2]			Could not display the DMS Receiven Cache
Type: A (Host Address) (1)			courd not display the bus Resolver Cache.
Class: IN (0x0001)			(:) Users) student)
Y Answers			c. (osers (students)
 Serham.cz: type A, class IN, addr 77.75.75.172 			
Type: A (Host Address) (1)			
(lass: IN (0x0001)			
Time to live: 5			
Data length: 4			
Address: 77.75.75.172			
∨ seznam.cz: type A, class IN, addr 77.75.75.176			
Name: seznam.cz			
Type: A (Host Address) (1)			
Liass: IN (0X0001) Time to live: 5			
Data length: 4			
Address: 77.75.176			
[Request In: 1]			
[Time: 0.001026000 seconds]		~	1
🔵 🎽 Response TTL (dns.resp.ttl), 4 bytes	Packets: 73 · Displayed: 8 (11.0%) Profile: Defa	ault	

Obr. 6.1: Nízká hodnota TTL pro uložené DNS záznamy, v důsledku které se po několika sekundách nevypíše žádný DNS záznam.

S tímto problémem úzce souvisí i poznatek, ke kterému došlo v rámci vytváření scénářů. Příkaz nslookup sice využívá služby DNS protokolu pro zjištění IP adres nebo doménových jmen, ale své výsledky neukládá do stejné dočasné paměti jako příkaz ping. Když tedy odešleme požadavek s příkazem nslookup a následně chceme zobrazit výpis ipconfig -displaydns, ukáže se nám stejná chyba jako v předchozím případě výše. V rámci scénáře se záznam do paměti uloží tím, že studenti zobrazí webovou stránku v internetovém prohlížeči. To záznam uloží do paměti stejně jako v případě použití příkazu ping.

6.2 Síťové připojení virtualizovaného systému

U testování prvního scénáře se objevil problém, který se vyskytoval pouze na virtualizovaném operačním systému na školních počítačích. Při zadání příkazu tracert se nevypisovala doba odezvy jednotlivých prvků v síti a také se nezobrazují IP adresy těchto prvků, viz obr. 6.2. K tomuto problému se přidal další, při testování maximálního možné velikosti přidaných dat, které lze poslat spolu s pingem pomocí parametru -l 65500.

Oba problémy jsou způsobeny síťovým nastavením virtualizovaného operačního systému. V programu VMware je pro tento systém nastaven síťový adaptér na NAT (Network address translation), kdy se všechny požadavky z virtualizovaného systému
4	TEthernet0													
File	Edit	View Go	Capture	e Analyze	Statistics Telepho	ony Wire	less Ti	ools	Help					
	1	Θ	h 🗙 🗅	۹ 👳	+ 2 T 🕹 📃	≣ ©,	ର୍ ଭ୍	種						
	cmp													
No.		Time	Source		Destination	Protocol	Length	Info						
	5640	6.531668	192.168	.152.128	77.75.75.176	ICMP	74	Echo	(ping)	request	id=0x0001,	seq=111/28416,	ttl=128 (reply in 6264)	
4	6264	7.076539	77.75.7	5.176	192.168.152.128	ICMP	74	Echo	(ping)	reply	id=0x0001,	seq=111/28416,	ttl=128 (request in 5640)	
	7347	7.546646	192.168	.152.128	77.75.75.176	ICMP	74	Echo	(ping)	request	id=0x0001,	seq=112/28672,	ttl=128 (reply in 7363)	
	7363	7.577592	77.75.7	5.176	192.168.152.128	ICMP	74	Echo	(ping)	reply	id=0x0001,	seq=112/28672,	ttl=128 (request in 7347)	
	8369	8.562317	192.168	.152.128	77.75.75.176	ICMP	74	Echo	(ping)	request	id=0x0001,	seq=113/28928,	ttl=128 (reply in 8370)	
	8370	8.638118	77.75.7	5.176	192.168.152.128	ICMP	74	Echo	(ping)	reply	id=0x0001,	seq=113/28928,	ttl=128 (request in 8369)	
	9386	9.594000	192.168	.152.128	77.75.75.176	ICMP	74	Echo	(ping)	request	id=0x0001,	seq=114/29184,	ttl=128 (reply in 9789)	
L	9789	9.781352	77.75.7	5.176	192.168.152.128	ICMP	74	Echo	(ping)	reply	id=0x0001,	seq=114/29184,	ttl=128 (request in 9380)	
64	Příkazo	vý řádek						-	- 0		Příkazový řá	idek		
Tra	cing	route to	seznam	.cz [77.	.75.75.176]						C:\Users\stu	udent>ping se	znam.cz	
ove	r a m	aximum o	f 30 ho								Pinging seznam.cz [77.75.75.176] with 32 bytes of data:			
											Reply from	77.75.75.176:	bytes=32 time=545ms TTL=128	
1		1 ms	<1 ms	<1 ms	192.168.152.2						Reply from	77.75.75.176:	bytes=32 time=31ms TTL=128	
2		*			Request timed	out.					Reply from .	77.75.75.176:	bytes=32 time=75ms IIL=128	
3			2		Request timed	out.					Reply from a	//./5./5.1/0:	Dytes=32 time=18/ms (11=128	
4		*			Request timed	out.					Ping statist	tics for 77.7	5.75.176:	
6					Request timed	out.					Packets	: Sent = 4. R	eceived = 4. Lost = 0 (0% loss).	
7					Request timed	out.					Approximate	round trip t	imes in milli-seconds:	
8					Request timed	out.					Minimum	= 31ms, Maxi	mum = 545ms, Average = 209ms	
9		*			Request timed	out.								
10					Request timed	out.					C:\Users\stu	udent>		
11					Request timed	out.					C:\Users\stu	udent≻		
12	34	0 ms 2:	15 ms	255 ms	www.seznam.cz	[77.75	.75.17	[6]			C:\Users\stu	udent>		
Tra	Trace complete.								C:\Users\stu C:\Users\stu	udent> udent>				
c:\	rate complete. C:\Users\student> :\Users\student> :\Users\student>													

Obr. 6.2: tracert a ping na seznam na mém pc s virtuálkou nastavenou na NAT.

překládají v hostovském systému. Možnosti nastavení síťového adaptéru v programu VMware je zobrazen na obr. 6.3. VMware a virtualizovaný systém si ovšem nedokáží poradit s požadavky v podobě příkazů tracert a ping s velkým množstvím přidaných dat.

Řešením by bylo přepnout nastavení síťového adaptéru na "Bridged", kdy se virtualizovaný systém připojí přímo do sítě a komunikace se nepřekládá. To ale není možné realizovat ve školních laboratořích, kde by byla komplikovaná správa IP adres těchto virtualizovaných systémů, nehledě na nedostatek těchto IPv4 adres. Ve vytvořených scénářích se tak bude tato část vypracovávat přímo v hostovském operačním systému.

dura oracia			
options Device Memory Processors Hard Disk (SCSI) Co/DVD (SATA) Network Adapter US8 Controller US8 Controller Display	Summary 4 GB 2 30 GB Auto detect Bridged (Automatic) Present Using device Reproduktory (Present Auto detect	Device status Connected Connect at power on Network connection Bridged: Connected directly to the physical network Replicate physical network connection state NAT: Used to share the host's IP address Host-only: A private network shared with the host Custom: Specific virtual network Minet0 LAN segment: LAN Segments Advanced	

Obr. 6.3: Nastavení síťového adaptéru virtualizovaného operačního systému v programu VMware.

6.3 Výběr programu pro uskutečnění VoIP hovoru

Cílem programu bylo uskutečnit nešifrovaný VoIP hovor pro využití ve druhém scénáři. Naštěstí dnes již mnoho programů nešifrovanou variantu nevyužívá, což ovšem není ideální pro sestavování plánované laboratorní úlohy. Jako první se objevil program ZoiPer. Ten sice nabízí verzi programu pro nekomerční využití, ale po instalaci programu se v nastavení ukáže, že spousta těchto nastavení je pouze pro placenou verzi. A to včetně registrace více uživatelů nebo možností nastavení šifrování (v našem případě hlavně nešifrování) hovorů.

Poté se objevila lepší varianta v podobě programu Linphone. Program po bezplatném stažení a instalaci nabídl registraci, po které již můžeme uskutečňovat VoIP hovory. Zároveň je možná registrace několika uživatelů a to jak přes hlavní okno programu nebo webové stránky programu. Bohužel při testování se párkrát stalo, že se program neočekávaně ukončil. Při testování na virtualizovaném operačním systému se již program neukončoval, ale docházelo k problému se špatnou průhledností hlavního okna programu na světlém pozadí. To značně komplikovalo jeho používání. Při hledání řešení tohoto problému se ovšem objevila druhá varianta programu Linphone. Jde o variantu, která je ke stažení zdarma v Microsoft store. Obsahuje všechny potřebné nastavení jako nastavení audio kodeků nebo nastavení šifrování. Registrace se provede během dvou minut na oficiálních stránkách programu. Jde o odlehčenou verzi programu, ale je zde vše pro zvládnutí vytvořených laboratorních úloh. Detailnějšímu popisu programu Linphone se věnuje kapitola 5.3 a také kapitola B.2.1, která je součástí druhého scénáře.

6.4 Výběr serveru pro zpřístupnění webové stránky

Ve druhém scénáři se nachází úloha, která zobrazuje rozdíly mezi protokolem TCP a UDP. UDP je využito prostřednictvím protokolu QUIC. V jedné z částí úlohy se zachytí pakety obou těchto protokolů při načítání webové stránky. Ve webovém prohlížeči Chrome, lze v nastavení přepínat využití experimentálního protokolu QUIC. Tím můžeme jednu stránku načíst přes jeden nebo druhý typ transportního protokolu. Pro využití v této úloze ale bylo potřeba najít vhodnou webovou stránku na které tento postup testovat. Na internetu se ještě nenachází mnoho stránek, které QUIC podporují. Podporu mají hlavně stránky od společnosti Google, jako Youtube nebo samotný vyhledávač. Tyto stránky ale obsahují mnoho analytických a reklamních prvků, které by mohly porovnání protokolů zkreslovat. Další variantou byla možnost vytvořit vlastní webový server, kde ale není varianta vytvoření serveru s podporou protokolu QUIC pro Windows. Nakonec se tedy využila varianta, kde se vytvoří vlastní webová stránka přes platformu Google sites. Jelikož stránky se nachází na serverech Googlu, tak je zde podpora pro protokol QUIC. Zároveň v nastavení stránky lze vypnout analytické nástroje a na vytvořených stránkách se nenacházejí reklamy.

QUIC můžeme analyzovat ve Wiresharku jako všechny ostatní pakety. Bohužel v základním nastavení se pakety s QUIC záhlavím zobrazují pouze jako UDP pakety a nelze zobrazit podrobnosti QUIC záhlaví (testováno na verzi Wiresharku 3.0.6). Pro umožnění analýzy tohoto protokolu je tak nutné v nastavení Wiresharku definovat, že má všechny UDP pakety přicházející z portu 443 identifikovat jako QUIC pakety. Následně již zobrazí záhlaví protokolu, včetně informace jestli jde o krátkou nebo dlouhou verzi záhlaví, podobně jako na obr. 6.4. Stále si ale Wireshark neporadí s detekováním verze protokolu QUIC. Pro částečnou analýzu je to ale dostačující a popis na zprovoznění detekce protokolu QUIC je součástí druhého scénáře.

	0_2_10	0_UDP.pca	png								_		×
Fil	e Edit	t View	Go Capture	Analyz	ze Statis	tics Te	lephony	Wireles	s Tools Hel	p			
			🗟 🔀 🛅	۹ 🗢	2 🖘	î 🕹		€, €,	Q. 🎹				
	Apply a	display filte	er <ctrl-></ctrl->										- +
No.	1	Time	Source		Destinatio	1	Protocol	Length	Info				^
	1 (000000	192.168.17	1.129	172.217	.23.2	QUIC	1392	65480 → 443	3 Len=1350[Malformed P	acket]		
Г	2 (0.001549	192.168.17	1.129	216.58.	201.1	QUIC	1392	65481 → 443	3 Len=1350[Malformed P	acket]		
	3 (0.002068	192.168.17	1.129	216.58.	201.1	QUIC	729	0-RTT	Les 12505N-1 General D			
	4 6	0.003342 3 336173	192.168.17	234	1/2.21/	171	QUIC	1392	49525 → 44: 0-PTT DCT	3 Len=1350[Maitormed P)-1557546500	аскетј		
Li -	6 (0.226172	216.58.201	.110	192.168	.171	OUTC	1392	0-RTT, DCI)=cb608eaf8a			
	7 (0.227037	192.168.17	1.129	216.58.	201.1	OUIC	70	Protected F	Payload (KP0)			
	8 (0.227751	192.168.17	1.129	172.217	.23.2	QUIC	70	Protected F	Payload (KP0)			
	9 (0.227927	172.217.23	195	192.168	.171	QUIC	1392	0-RTT, DCI)=c525fb06fc			
	10 (0.231902	192,168,17	1.129	172.217	.23.2	QUIC	1392	50851 → 443	3 Len=1350[Malformed P	acket]		
	11 (0.232680	192.168.17	1.129	172.217	.23.1	QUIC	70	Protected P	Payload (KP0)			Y
<												>	
 ~	QUIC	IETF											^
	> QU	IC Conne	ction inform	ation									
	1	аскет Le	ngtn: 263j - Hoodon For		a Hoodor	(1)							
	.1		= Fixed Bit:	True	g neauer	(1)							
		01 ·	= Packet Type	e: 0-R	TT (1)								
	Ver	rsion: U	nknown (0x51	303436)								
	Des	stinatio	n Connection	ID Le	ngth: 5								
	Des	stinatio	n Connection	ID: c	b608eaf8	a							
	SOL	urce Con	nection ID L	ength:	212	0770	c20-0-00	070471	0				- 11
	500	arce com agth: 38	nection iD:	/05000	00000148	956472	02969690	50/94/1	000046001601	ru4			
	> TE	xpert In	fo (Warning/	Decrvp	tion): F	ailed	to decry	/pt pac	ket numberl				
	Rer	maining	Payload: 40f	14568c	2333d25b	f20dea	ebbb210	5120a17	9922b6bcd46.				
~	QUIC :	IETF											
	[Pa	acket Le	ngth: 1087]										
	Rer	maining	Payload: 00d	449182	539cec50	638530	758166b 1	Fc89c02	39e07695a99.				\sim
<													>
	2	0_2_100_U	IDP.pcapng							Packets: 1880 • Displayed:	1880 (100.0%)	Profile: Def	ault 🔡

Obr. 6.4: Detekce protokolu QUIC ve Wiresharku, včetně zobrazeného záhlaví.

7 Návrhy scénářů

Cílem bakalářské práce je navrhnout dva scénáře, kdy první má obsahovat seznámení studentů s programem Wireshark a ukázku základních funkcí. Druhý bude obsahovat složitější úkoly, které budou vysvětlovat funkce a využití protokolů z předmětu Komunikační technologie a zejména pak i pokročilejší funkce programu Wireshark.

7.1 Návrh prvního scénáře

První scénář bude obsahovat vše, co je potřebné pro seznámení se s programem Wireshark. Tím se rozumí popis prostředí, kde se nachází daný výpis a kam se píší parametry pro filtrování paketů. Dále také jak spustit první zachytávání paketů a jak z velkého množství zachycených paketů efektivně vyfiltrovat potřebné protokoly pro zadané úkoly.

Po úvodním seznámení si studenti vyzkouší funkce základních příkazů, které byly vysvětleny v teoretické části a jejich zachycení ve Wiresharku. Nejdříve tedy proběhne testování konektivity pomocí příkazové řádky a příkazu ping. Ping je použit z důvodu názornosti a jednoduchosti, kdy pomocí jednoho základního příkazu můžeme demonstrovat úvodní zachycení paketů a následně se seznámit s prostředím. Kde se nachází zdrojová a cílová adresa nebo port a také další podrobnosti, které nalezneme v detailnějším rozboru vybraného paketu. Pomocí pingu lze také ukázat základní filtrování pouze určitého typu protokolu nebo IP adresy. Po vyfiltrování se může provést úvodní rozbor protokolu ICMP, konkrétně echo žádostí a odpovědí.

Protokol ICMP byl navrhnut do scénáře, protože je úzce spjat s příkazem ping a hlavně s jeho parametry. Kdy pomocí parametrů můžeme vyvolat chybové zprávy ICMP a ty následně analyzovat ve Wiresharku. Při změně parametrů budou studenti sledovat změny v záhlaví protokolů a také zjistí jak fungují další příkazy v příkazové řádce jako například traceroute. Zjistí také co znamená a k čemu se využívá fragmentace, při použití parametru pro změnu velikosti zasílaných dat spolu s pingem. Na úpravu těchto parametrů navazuje jedna z dalších částí, která bude zaměřena na zachycení a analýzu chybového označení ICMP zpráv pomocí kódu a typu těchto zprávy u protokolu ICMP, kdy bude úkolem zjistit jaké označení nese jakou chybovou zprávu.

Dále se studenti také seznámí s prací a výhodami, které nám přináší protokol DNS a DNS servery. Při analýze studenti uvidí vše, co obsahuje záhlaví DNS protokolu a také, které věci si Wireshark umí dopočítat sám i když nejsou součástí záhlaví. K ukázce DNS protokolu bude využit příkaz nslookup a konec cvičení bude zaměřen na program DNS resolver, který studentům ukáže s jakými daty pracují DNS servery. Bude zde poukázáno na rozdíly mezi různými typy DNS záznamů, protože DNS resolver umožňuje přepínání mezi těmito typy DNS záznamů. DNS resolver také umí přepínat mezi tím, který transportní protokol využije, takže lze vyzdvihnout rozdíly mezi transportními protokoly TCP a UDP, čehož bude využito ve druhém scénáři.

7.2 Návrh druhého scénáře

Druhý scénář bude taktéž řešen ve virtualizovaném prostředí operačního systému Windows 10. Systém bude stejně jako v případě prvního scénáře obsahovat všechny potřebné programy pro úspěšné zvládnutí cvičení. Část cvičení bude zaměřena na analýzu VoIP hovoru (Voice over Internet Protocol) a druhá část ukáže možnosti využití grafů ve Wiresharku a také se porovnají protokoly TCP+TLS s protokolem QUIC při načítání webové stránky.

První část bude využívat program pro realizaci VoIP hovorů Linphone. Nejdříve bude program popsán, včetně postupu jednoduché registrace, budou popsány jeho funkce a možnosti nastavení hovorů. Pomocí programu se uskuteční hovor, který se zachytí ve Wiresharku a následně dojde na analýzu paketů a protokolů, které jsou využívány při sestavování a uskutečňování hovoru. Rozebrány budou hlavně protokoly SIP a RTP. Zároveň po zachycení těchto paketů dojde k rekonstrukci hovoru přímo ve Wiresharku, který umí zpětně sestavit VoIP hovor a přehrát ho. Samozřejmě v případě, že hovor je uskutečňován v nešifrované podobě. Naproti tomu program Linphone nabízí také možnost hovorů v šifrované podobě, takže studenti uvidí a vyzkouší si analyzovat obě varianty.

Druhá část se bude věnovat rozdílům mezi transportními protokoly TCP a UDP. A to například při přepínání těchto transportních protokolů v programu DNS resolver, který byl využit již v prvním scénáři. Zároveň bude vysvětlena a představena funkce Wiresharku, kdy můžeme ze zachycených paketů sestavovat grafy pro zobrazení například množství paketů v čase. Do grafu můžeme umístit pakety různých protokolů a porovnávat jejich množství mezi sebou. Při tom se využije funkce filtrování, která byla vysvětlena v prvním scénáři. Dále budou porovnány protokoly TCP+TLS a QUIC při načítání vytvořené webové stránky. Ve webovém prohlížeči Chrome totiž můžeme přepínat a vybrat si pomocí kterého protokolu se stránka načte. Ve výchozím nastavení jde o transportní protokol TCP a druhou možností je zapnout v nastavení načítání stránky přes experimentální protokol QUIC, který je postaven na základech protokolu UDP. Srovnání protokolů bude provedeno nejprve při běžných podmínkách a následně se budou měnit parametry jako je maximální šířka pásma nebo latence pomocí nastavení síťového adaptéru v programu VMware Workstation Player. V poslední části se využije utilita Netlog Viewer pro analýzu webového provozu.

7.3 Kompletní znění vytvořených scénářů

Kompletní scénáře jsou z důvodu přehlednosti a rozsáhlosti uvedeny jako příloha této práce. První scénář včetně teoretického úvodu pro studenty je uveden v příloze A. Stejně tak se celý druhý scénář nachází v příloze B. Řešení scénářů v podobě odpovědí na průběžné otázky ze scénářů je uvedeno v přílohách C a D.

Závěr

V této práci byl popsán základní síťový model TCP/IP a také základní síťové protokoly jako jsou TCP, UDP a DNS. Velká pozornost byla věnována také protokolu ICMP, protože ten byl nejvíce analyzován v následné praktické části v rámci prvního laboratorního scénáře. Kromě protokolů byly definovány také programy a příkazy, které se v těchto vytvořených scénářích uplatnily. Před vytvořením scénářů byl proveden průzkum některých volně dostupných scénářů, které využívají program Wireshark. Z tohoto průzkumu byly vytvořeny návrhy scénářů, které byly průběžně konzultovány s vedoucím práce. Dle návrhů byly vytvořeny dva samostatné laboratorní scénáře, které mohou být využity v rámci předmětu Komunikační technologie.

Oba scénáře obsahují návod pro studenty s postupem řešení a také doplňující otázky a úkoly k probraným protokolům, příkazům nebo funkcím programu. K doplňujícím otázkám byly vytvořeny soubory pro vyučujícího, které shrnují správné odpovědi a postupy. Ke každému scénáři byl vytvořen teoretický úvod, který shrnuje teorii obsaženou v této bakalářské práci. Oba scénáře také obsahují předpřipravené soubory se zachycenými pakety, je tak možné odpovědět na otázky i v případě, že není aktuálně možné pakety zachytit díky neočekávaným výpadkům apod. Pro vypracování scénářů byl vytvořen obraz virtualizovaného operačního systému, který obsahuje všechny potřebné programy a nastavení. Pro potřeby druhého scénáře byla vytvořena jednoduchá webová stránka přes platformu Google Sites. Oba scénáře byly otestovány a upraveny pro správnou funkčnost ve školní učebně. V jedné z kapitol jsou také popsány některé překážky, které bylo nutné překonat při vytváření a testování scénářů.

První scénář je zaměřen na úvodní seznámení s paketovým analyzerem Wireshark a také na vyzkoušení základních příkazů v příkazové řádce, jako je ping, traceroute a další a to včetně jejich parametrů. Tyto příkazy vyvolaly použití protokolů, které byly popsány v teoretické části práce a následně byly analyzovány v prostředí programu Wireshark. Jedná se hlavně o protokoly ICMP, IP a DNS. Na protokolu ICMP byly vysvětleny základní funkce programu Wireshark a také možnosti příkazu ping. Bylo analyzováno záhlaví protokolu a studenti zjistili, jak fungují chybové kódy tohoto protokolu. IP protokol byl nejvíce vidět při rozdělení dat na jednotlivé fragmenty a DNS protokol byl využit při objasnění funkce DNS serverů za pomocí programu DNS resolver, který dokáže generovat požadavky na překlad IP adres nebo doménových jmen.

Druhý scénář je již pokročilejší a obsahuje dvě hlavní části, které však využívají prvky, které se studenti naučili při vypracovávání prvního scénáře, jako například filtrování paketů nebo zobrazení záhlaví jednotlivých protokolů. První část se

věnovala realizaci VoIP hovoru, který využívá protokoly SIP a RTP. Hovor byl následně analyzován a taktéž byly využity různé možnosti šifrování hovoru, které zvolený program Linphone podporuje. Dále byla použita funkce Wiresharku, kdy ze zachycených paketů můžeme vytvářet grafy. Například první vytvořený graf studentům ukazuje maximální přenosovou rychlost použitého kodeku u VoIP hovoru. Druhá část scénáře je také zaměřena na srovnávání. Porovnávají se zde dva protokoly, které jsou využity pro načtení zabezpečené webové stránky. Jedná se o protokoly TCP+TLS a QUIC. Využije se zde funkce Wiresharku, která spojí více souborů do jednoho a také určité přehledy Wiresharku, které zobrazí počet použitých bajtů a paketů jednotlivých protokolů při načítání stránky. Srovnání bude provedeno za ideálních podmínek a následně se podmínky upraví změnou sítových parametrů. Změna parametrů, jako je maximální šířka pásma, ztrátovost paketů nebo zvýšená latence, bude simulována programem VMware Workstation Player. Poslední část scénáře studentům ukáže možnost zachycení a zobrazení webového provozu přímo v prostředí prohlížeče Chrome pomocí utility Netlog viewer.

Literatura

[1] JEŘÁBEK, J. Komunikační technologie: Skriptum FEKT Vysoké učení technické v Brně [online]. Brno: Vysoké učení technické v Brně Fakulta elektrotechniky a komunikačních technologií Ústav telekomunikací, 2013, poslední aktualizace 7.1.2019 [cit. 2019-11-18]. ISBN 978-80-214-4713-4. Dostupné z URL:
<https://www.vutbr.cz/studenti/predmety/detail/183986>.

 [2] KRISTOFF, John. The Transmission Control Protocol. DePaul University [online]. Chicago, 2000, April 24, 2000 [cit. 2019-12-11]. Dostupné z URL:

<https://condor.depaul.edu/~jkristof/technotes/tcp.html>.

- [3] KUROSE, James F. a Keith W. ROSS. *Computer networking: a top-down approach*. Seventh edition. Essex: Pearson, 2017. ISBN 1-292-15359-8.
- [4] RFC 791 Internet Protocol. IETF Tools [online]. [cit. 2019-12-11]. Dostupné z URL: https://tools.ietf.org/html/rfc791.
- [5] RFC 1883 Internet Protocol, Version 6 (IPv6) Specification. IETF Tools [online]. [cit. 2019-12-11]. Dostupné z URL:
 <https://tools.ietf.org/html/rfc1883>.
- [6] RFC 792 Internet Control Message Protocol. IETF Tools [online]. [cit. 2019-12-11]. Dostupné z URL: <https://tools.ietf.org/html/rfc792#ref-1>.
- [7] MARGARET, Rouse. Time-to-live (TTL). Techtarget.com: whatis.com [online].
 2015 [cit. 2019-11-18]. Dostupné z URL:
 https://searchnetworking.techtarget.com/definition/time-to-live.
- [8] DNS: Types of DNS Records, DNS Servers and DNS Query Types. NS1 [online]. 2019 [cit. 2019-12-11]. Dostupné z URL: <https://nsl.com/resources/dns-types-records-servers-andqueries>.
- [9] ARORA, Himanshu. Linux ping Command Tutorial for Beginners. HowtoForge [online]. Jun 12, 2018 [cit. 2019-12-01]. Dostupné z URL: <https://www.howtoforge.com/linux-ping-command/>
- [10] Ping Linux man page. Die.net [online]. [cit. 2019-12-11]. Dostupné z URL: <https://linux.die.net/man/8/ping>.

- [11] Ping. Microsoft.com [online]. Redmond (Washington): Microsoft Corporation, 2018 [cit. 2019-11-18]. Dostupné z URL: <https://docs.microsoft.com/en-us/windows-server/administration/ windows-commands/ping>.
- [12] CHANDEL, RAJ. Working of Traceroute using Wireshark. Hackingarticles.in: Raj Chandel's Blog [online]. 2018 [cit. 2019-11-18]. Dostupné z URL: <https://www.hackingarticles.in/working-of-traceroute-usingwireshark/>
- [13] RFC 3261 SIP: Session Initiation Protocol. IETF Tools [online]. June 2002
 [cit. 2020-02-15]. Dostupné z URL: <https://tools.ietf.org/html/rfc3261>.
- [14] RTP: A Transport Protocol for Real-Time Applications. IETF Tools [online]. July 2003 [cit. 2020-02-23]. Dostupné z URL: <https://tools.ietf.org/html/rfc3550>.
- [15] VMware Workstation Player. VMware [online]. 2019 [cit. 2019-12-04]. Dostupné z URL: <https://www.vmware.com/products/workstation-player.html>
- [16] Wireshark [online]. 2019 [cit. 2019-11-18]. Dostupné z URL: <https://www.wireshark.org/>.
- [18] SOUSEDÍK, Tomáš. Klient protokolu DNS s grafickým rozhraním vhodný pro demonstrativní účely. Brno, 2014. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Vedoucí práce Jan Jeřábek.
- [20] RTP payload formats. Wikipedia: the free encyclopedia [online]. 26 October 2019 [cit. 2020-03-04]. Dostupné z URL: <https://en.wikipedia.org/wiki/RTP_payload_formats>
- [21] RFC 8446 The Transport Layer Security (TLS) Protocol Version 1.3. IETF Tools [online]. August 2018 [cit. 2020-05-03]. Dostupné z URL: <https://tools.ietf.org/html/rfc8446>

- [22] KRČMÁŘ, Petr. HTTP/3 nebude postavené na TCP, základem bude QUIC používající UDP. *Root.cz* [online]. 14.11.2018 [cit. 2020-05-03]. Dostupné z URL: <https://www.root.cz/clanky/http-3-nebude-postavene-na-tcpzakladem-bude-quic-pouzivajici-udp>
- [23] JEŘÁBEK, J. Pokročilé komunikační techniky: Skriptum FEKT Vysoké učení technické v Brně [online]. Brno: Vysoké učení technické v Brně Fakulta elektrotechniky a komunikačních technologií Ústav telekomunikací, 2019, poslední aktualizace 28.1.2020 [cit. 2020-05-13]. Dostupné z URL: <https://www.vutbr.cz/studenti/predmety/detail/211991>.
- [24] DANSCOURSES. Using Wireshark to capture a 3 way handshake with TCP.
 In: Youtube [online]. 25. 5. 2019 [cit. 2020-05-05]. Dostupné z URL:
 <https://www.youtube.com/watch?v=4dSaAMZsPvw>
- [25] ALPINE SECURITY. How to Capture Passwords with Wireshark. In: Youtube [online]. 22. 4. 2019 [cit. 2020-05-05]. Dostupné z URL: <https://www.youtube.com/watch?v=_m-9QXzVHkg&ab_channel= AlpineSecurity>
- [26] CANFIELD, Jeremy. View FTP usernames and passwords. FreeKB [online]. 2020 [cit. 2020-05-05]. Dostupné z URL: <http://www.freekb.net/Article?id=133>
- [27] YODER. Exercise: TCP Handshake: TCP Handshake and Closing of TCP Connection. In: Milwaukee School of Engineering faculty web [online]. 2014 [cit. 2020-05-05]. Dokument ve formátu PDF. Dostupné z URL: <https://faculty-web.msoe.edu/yoder/a/cs2910/15q1/slides/ Exercise.8-2.TCPHandshake.pdf>

Seznam symbolů, veličin a zkratek

ACK	acknowledgement - potvrzení		
ASCII American Standard Code for Information Interchange			
\mathbf{BE}	Big endian		
CNAME	Canonical name		
DHCP	Dynamic Host Configuration Protocol		
DNS Domain Name System – Systém doménových jmen			
Domain Name System			
DNSSEC Domain Name System Security Extensions			
FIN finish – ukončení			
\mathbf{FTP}	File Transfer Protocol		
\mathbf{HTTP}	Hypertext Transfer Protocol		
\mathbf{HTTPS}	Hypertext Transfer Protocol Secure		
ICE	Interactive Connectivity Establishment		
ICMP	Internet Control Message Protocol		
IETF	Internet Engineering Task Force		
IP	Internet Protocol		
ISO	International Organization for Standardization		
LE Little endian			
MTU	Maximum transmission unit		
MX	Mail exchangers		
NAT	Network address translation		
OS	Operating sytem – operační systém		
\mathbf{PCMA}	Pulse-code modulation A-law – pulzně kódová modulace A-law		
\mathbf{PCMU}	Pulse-code modulation $\mu\text{-law}-$ pulzně kódová modulace $\mu\text{-law}$		
QUIC	Quick UDP Internet Connections		
RFC	Request for Comments		
\mathbf{RSA}	iniciály autorů - Rivest, Shamir, Adleman		
RTCP	RTP Control Protocol		
RTP	Real-time Transport Protocol		
SIP	Session Initiation Protocol – protokol pro inicializaci relací		
SMTP	Simple Mail Transfer Protocol		
SOA	Start of authority		
SRTP	Secure Real-time Transport Protocol		
SYN	synchronization – synchronizace		
TCP	Transmission Control Protocol		
TCP/IP	Transmission Control Protocol/Internet Protocol		
TLD	Top-level domain		

\mathbf{TLS}	Transport Layer Security
\mathbf{TTL}	Time to live
UDP	User Datagram Protocol
VoIP	Voice over Internet Protocol

Seznam příloh

\mathbf{A}	Kon	npletní	návod pro první vytvořený simulační scénář	52
	A.1	Teoreti	ický úvod	53
		A.1.1	Protokoly transportní vrstvy - TCP a UDP	53
		A.1.2	Internet Control Message Protocol (ICMP)	54
		A.1.3	Systém doménových jmen (DNS)	54
	A.2	Praktie	cká část	55
		A.2.1	Popis prostředí programu Wireshark	55
		A.2.2	Základní zachycení ICMP paketů	57
		A.2.3	Filtry a jejich kombinace	58
		A.2.4	Úprava parametrů pro ping a zachycení	59
		A.2.5	Zachycení ICMP - zachycení chybového kódu	65
		A.2.6	Další prvky analýzy ICMP záhlaví	69
		A.2.7	DNS resolver	73
В	Kon	npletní	í návod pro druhý vytvořený simulační scénář	76
	B.1	Teoret	ický úvod 	77
		B.1.1	Protokoly realizující hovor skrze IP $\ .\ .\ .\ .\ .\ .$.	77
		B.1.2	Protokoly zajišťující zabezpečený přenos dat	78
	B.2	Realiza	ace druhého scénáře	79
		B.2.1	Popis prostředí programu Linphone	79
		B.2.2	Linphone registrace a nastavení $\hfill\h$	80
		B.2.3	Rekonstrukce hovoru a jeho základní zabezpečení $\ . \ . \ .$.	83
		B.2.4	Představení funkce analýzy pomocí grafů $\hfill\$	90
		B.2.5	Spojení zachycených souborů	93
		B.2.6	Srovnání protokolů TCP a UDP u DNS resolveru $\ . \ . \ . \ .$	95
		B.2.7	Nastavení prohlížeče a Wiresharku pro analýzu protokolu	
			QUIC	96
		B.2.8	Srovnání protokolů při načítání webových stránek $\ .\ .\ .\ .$	99
		B.2.9	Změna parametrů sítě a analýza změn $\ . \ . \ . \ . \ . \ .$	103
		B.2.10	Analýza QUICu v prostředí prohlížeče Chrome	107
С	Řeš	ení prv	vního simulačního scénáře	111
	C.1	Popis :	prostředí programu Wireshark	111
	C.2	Základ	lní zachycení ICMP paketů	111
	C.3	Filtry	a jejich kombinace	111
	C.4	Úprava	a parametrů pro Ping a zachycení	111
	C.5	Zachyo	cení ICMP - zachycení chybového kódu	113

	C.6	Další prvky ICMP záhlaví	. 114
	C.7	DNS resolver	. 114
_			
D	Reš	ení druhého símulačního scénáře	115
	D.1	Popis prostředí programu Linphone	. 115
	D.2	Linphone registrace a nastavení \ldots \ldots \ldots \ldots \ldots \ldots	. 115
	D.3	Rekonstrukce hovoru a jeho základní zabezpečení $\ \ldots \ \ldots \ \ldots \ \ldots$. 115
	D.4	Představení funkce analýzy pomocí grafů	. 116
	D.5	Spojení zachycených souborů	. 118
	D.6	Srovnání protokolů TCP a UDP u DNS resolver u $\ \ldots\ \ldots\ \ldots\ \ldots$. 118
	D.7	Nastavení prohlížeče a Wiresharku pro analýzu protokolu QUIC $$. 119
	D.8	Srovnání protokolů při načítání webových stránek $\ .\ .\ .\ .\ .$. 119
	D.9	Změna parametrů sítě a analýza změn	. 120
	D.10) Analýza QUICu v prostředí prohlížeče Chrome	. 121
\mathbf{E}	Obs	ah poskytnutého odkazu se soubory	122

A Kompletní návod pro první vytvořený simulační scénář

VYPRACOVANÁ PRVNÍ ÚLOHA

Úvod do programu Wireshark s pomocí utility ping, dalších funkcí ICMP protokolu a DNS resolveru.

A.1 Teoretický úvod

V tomto cvičení proběhne seznámení s programem Wireshark. Bude vysvětleno jak zachytit provoz na síti, jak tato data filtrovat a dále zpracovávat. Následující část se bude věnovat zachycení paketů s ICMP protokolem. V těchto paketech bude prakticky ukázána funkce zpráv echo request a echo reply. Dále si studenti vyzkouší různé parametry pro příkaz ping a také uvidí jejich následky ve Wiresharku. Využijí se také příkazy traceroute a nslookup a v poslední části se bude pracovat s programem DNS resolver, pomocí kterého můžeme zjišťovat IP adresy zvolených doménových jmen. Celé cvičení bude realizováno ve virtualizovaném operačním systému skrze VMware Workstation Player.

A.1.1 Protokoly transportní vrstvy - TCP a UDP

Tyto dva protokoly jsou využívány pro přenos všech dále popisovaných aplikačních protokolů v síti, proto je důležité znát jejich odlišnosti.

TCP je spojově orientovaný protokol. Je označován jako spolehlivý protokol, protože se soustředí na garanci doručení dat a také doručení ve správném pořadí. Pro spolehlivé doručování je nutno nejdříve sestavit (a po přenosu ukončit) spojení, které je zobrazeno na obr. A.1. Protokol je využíván ve službách, kde nevyžadujeme nejrychlejší možné doručení, ale spolehlivé doručení všech segmentů.



Obr. A.1: Navazování a ukončování spojení u protokolu TCP.

Oproti tomu UDP není spojově orientovaný a nepotřebuje tedy navazovat spojení před přenosem dat. Posílá samostatné zprávy, u kterých nezaručuje jejich doručení. Díky tomu také nese méně informací v záhlaví protokolu, protože nepotřebuje data číslovat, nebo si uchovávat informace o doručených a nedoručených datagramech. Porovnání hlaviček protokolů je zobrazeno na obr. A.2. UDP je vhodný pro aplikace, které potřebují rychlý přenos dat i za cenu ztráty některých datagramů. Případně aplikace, které si spolehlivost řeší jiným způsobem na vyšší vrstvě.

Transmission Control Protocol (TCP)							
1. bajt	2. bajt	3. bajt	4. bajt				
zdrojo	vý port	cílový port					
	pořadové číslo						
	potvrzovací číslo						
příznaky	(ACK,)	velikost TCP záhaví					
kontrolr	ní součet	urgent	pointer				

1. bajt	2. bajt	3. bajt	4. bajt		
zdrojo	vý port	cílový port			
Délka (UI	OP + data)	kontrolní součet			

Obr. A.2: Srovnání záhlaví transportních protokolů TCP a UDP.

A.1.2 Internet Control Message Protocol (ICMP)

ICMP je jeden z protokolů síťové vrstvy modelu TCP/IP. Protokol slouží k přenosu servisních zpráv, jako jsou informace o chybách nebo různá oznámení. Oznámení jsou využívána například při použití příkazu ping, který ke své funkci využívá právě zprávy protokolu ICMP. Konkrétně jde o echo request a echo reply, kdy nám právě odpověď oznamuje dostupnost požadované stanice a dobu odezvy. Informace o chybě se v protokolu identifikuje dvěma čísly. První určuje typ chyby a druhé číslo označuje kód chyby. Výčet několika možných chyb, včetně identifikátorů je uveden v tab. A.1.

typ	kód	popis chyby nebo oznámení
0	0	ping - odpověď (echo reply)
8	0	ping - požadavek (echo request)
3	0	cílová sít není dostupná
11	0	hodnota TTL vypršela během přenosu
11	1	fragmenty se nepovedlo poskládat do původních dat

Tab. A.1: Chybové zprávy protokolu ICMP, které jsou označeny pomocí typu a kódu.

A.1.3 Systém doménových jmen (DNS)

Domain Name System je systém, kterého je využíváno při vzájemném překladu doménových jmen a IP adres prvků v počítačových sítích. Překlad probíhá na DNS serverech, které si uchovávají pomyslnou tabulku se sloupci IP adres a doménových jmen. V případě, že daný server dostane pomocí DNS protokolu žádost o zjištění určité adresy, tak tuto adresu vyhledá a poskytne ji dotazujícímu zařízení. K vytvoření DNS komunikace budeme ve cvičení využívat program DNS resolver.

A.2 Praktická část

A.2.1 Popis prostředí programu Wireshark

Spustíme program Wireshark. Jeho zástupce se nachází na ploše. Po spuštění nás Wireshark vyzve k výběru rozhraní, pomocí kterého budeme sledovat sítový provoz. Vybereme Ethernet dvojitým kliknutím. Volba lze následně změnit v okně Capture interfaces, do kterého se dostaneme přes ikonu Capture options v horním menu ikon nebo přes položku Capture > Options... v textovém menu. Funguje také klávesová zkratka Ctrl+K.

Následně se nám otevře hlavní obrazovka programu. Ta je rozdělená do několika částí, jak je vidět na obrázku A.3 níže.

📕 *Ethe	met				– 🗆 X				
File Ec	File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help 🖌								
	2 💿 📙 🛅 🔀 🖸	९ ⇔ ⇒ ≌ । ₫	a a 🖩						
Apply	a display filter <ctrl-></ctrl->				Expression +				
No.	Time	Source	Destination	Protocol Length Info					
	75 0.024804	147.229.197.50	52.225.200.110	TCP 54 2151 - 443	[ACK] Seq=				
	74 0.024960	52.225.200.116	147.229.197.50	TCP 1514 445 - 2151	[ACK] Seq=				
1	75 0.024961	52.223.200.116	14/.229.19/.30	TCP 1514 443 → 2151	[ACK] Seq=				
	76 0.025025	147.229.197.30	52.225.200.110	TCP 54 2151 → 443	NR RPH0452				
	77 0.025040	147.229.197.177	14/.229.199.255	MBNS 92 Name query	ND DRW9455				
	78 0.025040	147.229.197.177	224.0.0.251	MDNS 81 Standard d	uery 0x0000 3				
	/90.025549	Te80::2180:3dd2:a	TT02::TD	MDNS 101 Standard d	uery 0x0000				
	80 0.025//9	52.223.200.116	147.229.197.30	TLS 749 Applicatio	on Data				
	81 0.025/80	52.225.200.116	14/.229.19/.30	TLS 78 Applicatio	on Data				
	82 0.025829	147.229.197.30	52.223.200.116	TCP 54 2151 → 443	E [ACK] Seq=				
	83 0.025927	Te80::0667:6ee8:e	TT02::1:TT00:0	ICM 86 Neighbor S	Solicitation				
1	84 0.028160	Te80::218D:3dd2:a	TT02::1:3	LLM 95 Standard d	uery 0xet03 🗸				
> Fra	ame 76: 54 b	ytes on wire (432 bi	its), 54 bytes ca	ptured (432 bits) on	interface 0 📫				
> Etł	nernet II, S	rc: WistronI_c7:59:3	37 (f0:de:f1:c7:5	9:37), Dst: HewlettP_	07:ca:1e (94:3f				
> Int	ternet Proto	col Version 4, Src:	147.229.197.30,	Dst: 52.223.200.116					
Y Tra	ansmission C	ontrol Protocol, Sro	: Port: 2151, Dst	Port: 443, Seq: 1, A	ck: 73394, Len:				
S	ource Port:	2151							
D	estination H	Port: 443							
	Stream index	x: 0]							
	TCP Segment	Len: 0]			4				
S	equence num	ber: 1 (relative	sequence number)						
	Next sequence	ce number: 1 (rel	ative sequence nu	umber)]					
А	cknowledgmen	nt number: 73394	(relative ack num	mber)					
e	101 = H	Header Length: 20 by	tes (5)						
> F	> Flags: 0x010 (ACK)								
¢					>				
0000	94 3f c2	07 ca 1e f0 de f1 d	c7 59 37 08 00 45	00 ·?···· ·· Y7··	E' K				
001	00 28 52	14 40 00 40 06 00 0	00 93 e5 c5 1e 34	df (R-@-@	4. 5				
0020	a c8 74 08	67 01 bb 0b 61 17 :	25 c3 80 8b 68 50	10 👥 t g · · a ·%· · · h	P. 7				
0030	20 1+ 56		- dute 2020 - Diseland, 2020 (real and)		Deafler Dafault				
	Transmission Control Protoc	cor (tcp), 20 bytes	ackets: 2929 • Displayed: 2929 (100.0%)	· Dropped: 0 (0.0%)	Profile: Default				

Obr. A.3: Základní prostředí programu Wireshark s vyznačením významných částí.

(1) Menu ikon a textové menu. Vlevo jsou dvě nejdůležitější ikony pro práci s Wiresharkem. První z nich po stisku začíná zachytávat pakety. Druhé tlačítko

je pro pozastavení v případě, že již nechceme pokračovat v zachytávání.

- (2) Pole pro zadání požadovaného filtru. Filtry se skládají z názvu protokolů, který chceme hledat. Případně můžeme také zadat, že požadujeme jen pakety s konkrétní zdrojovou nebo cílovou IP adresou nebo portem. S filtrováním paketů se také úzce pojí různé možnosti podbarvení různých protokolů případně lze jednou barvou podbarvit celou konverzaci s daným serverem (komunikace 2 IP adres se celá podbarví stejně).
- (3) Hlavní pracovní pole Wiresharku. Zde se objeví pakety zachycené během zaznamenávání sítového provozu. Sloupce nám dávají informace o každém paketu: pořadí paketu od spuštění, čas ve kterém byl paket zachycen, zdrojová a cílová IP adresa, název protokolu, délka - objem dat které paket nese (v bajtech) a v posledním sloupci se nachází další informace, například čísla portů.
- (4) Pod výpisem paketů se nachází další okno, kde se zobrazí podrobnosti o paketu v případě, že si nějaký vybereme kliknutím ve vrchním okně s výpisem. V jednotlivých odrážkách se zobrazí struktura paketu tak, jak byla data zapouzdřena. Můžeme vidět například zdrojové a cílové IP adresy a mac adresy, verzi IP protokolu, hodnoty TTL a následně podrobnosti toho protokolu, který zrovna paket nese.
- (5) V dolním okně se zobrazují bajty daného paketu tak, jak byly přijaty. Pomocí pravého tlačítka můžeme přepínat zobrazení z hexadecimálního zápisu na binární.
- (6) Na dolní liště vlevo můžeme vidět dodatečné informace o právě vybrané části paketu. Například když v okně číslo 4 klikneme na User Data Protocol..., tak se nám v okně 5 označí 8 oktetů přijatých dat a v na dolní liště vidíme, že část paketu, která obsahuje UDP protokol má opravdu velikost 8 bajtů, tak jak uvádí okno číslo 5 v binárním zápisu.
- (7) V pravém dolním rohu můžeme přepínat profily zobrazení. Výchozí profil jsme si teď popsali, ale na internetu můžeme dohledat a importovat další profily, které nám přidají nové sloupce do třetího okna. To se hodí například, když potřebujeme analyzovat více paketů jednoho protokolu. Profily nám usnadní a zpřehlední výpis síťového provozu.
- (8) Na spodní liště uprostřed máme informaci o celkovém počtu zachycených paketů a také počet právě zobrazovaných paketů. Čísla se mohou lišit když nastavíme nějaký filtr zobrazených paketů. Vidíme zde také přepočet těchto čísel na procenta a také informaci o zahozených paketech Wiresharkem. To znamená, že Wireshark ví, že byly pakety přijaty, ale třeba z důvodu pomalého disku počítače nebyly zapsány a nejsou zobrazeny ve výpisu.

Úkoly:

- (1) Prostudujte hlavní prostředí Wiresharku, nahlédněte jaké možnosti nabízí hlavní menu a jeho položky.
- (2) Vyzkoušejte si základní operace programu a prozkoumejte pár zachycených paketů.

A.2.2 Základní zachycení ICMP paketů

Nyní již přejdeme k používání programu Wireshark. Cílem bude zachytit ICMP pakety, které můžeme vyvolat například při zaslání příkazu ping do příkazové řádky. Následně rozebereme odeslané a přijaté pakety.

Spustíme tedy program Wireshark, zvolíme si rozhraní Ethernet pro zachytávání a dále spustíme příkazovou řádku operačního systému Windows. Do vyhledávacího pole ve Start menu v levém dolním rohu napíšeme "cmd" a následně spustíme nalezený program. Ve Wiresharku spustíme zachytávání a následně se přesuneme do konzole příkazového řádku. Napíšeme příkaz na otestování konektivity se vzdáleným sítovým prvkem pomocí IPv4:

ping <volitelné parametry> <IP adresa nebo doménové jméno> ping -4 seznam.cz

A po dokončení výpisu zastavíme zachytávání v okně programu Wireshark. Měli bychom vidět podobný výpis jako na obr. A.4. V případě, že je ve výpisu více zachycených protokolů, můžeme vyfiltrovat požadované pakety zadáním filtru "icmp or dns". Nejprve se klient dotáže na IP adresu domény seznam.cz. Ve sloupci source na prvním řádku vidíme naši IP adresu a v cílové adrese máme adresu DNS serveru. Tento server zná IP adresy požadovaných domén a následně nám ji zašle v DNS odpovědi. V případě, že by adresu neznal, tak se dotáže dalších DNS serverů. Zjištěnou IP adresu můžeme vidět ve třetím řádku a sloupci destination. Následně na řadu přichází ICMP protokol, který zajišťuje zasílání paketů echo request a echo reply. Po každé přijaté ICMP odpovědi nám příkazová řádka vypíše nejpodstatnější informace, a to: zjištěnou IP adresu, velikost přijatých dat, dobu odezvy a hodnotu TTL. Hodnota TTL u zprávy echo request bývá většinou 128 nebo 64, ale může se lišit podle typu a nastavení operačního systému.

	*Ethernet0										_		\times
Fil	e Edit View	Go Capture Ana	alyze Statistics Teleph	iony Wi	reless To	ools Help							
	■ ⊿ 💿	🔄 🔀 🖾 🔍	e e 🕾 T 🛓 📃	≣ @	ର୍ବ୍	壅							
	icmp or dns										× → •	Expression.	
No.	Time	Source	Destination	Protocol	Length	Info							^
	14 4.021	804 192.168.0.11	2 8.8.8.8	DNS	69	Standard qu	ery Øx589	b A seznam.	cz				
	15 4.144	502 8.8.8.8	192.168.0.112	DNS	101	Standard qu	ery respo	inse Øx589b	A seznam.cz A	77.75.75.176	A 77.75	.75.172	
7*	16 4.162	401 192.168.0.11	2 77.75.75.176	ICMP	74	Echo (ping)	request	id=0x0001,	seq=81/20736,	ttl=128 (re	ply in 17	7)	
4	17 4.192	934 77.75.75.176	192.168.0.112	ICMP	74	Echo (ping)	reply	id=0x0001,	seq=81/20736,	ttl=54 (req	uest in 1	16)	
	19 5.182	195 192.168.0.11	2 77.75.75.176	ICMP	74	Echo (ping)	request	id=0x0001,	seq=82/20992,	ttl=128 (re	ply in 20	0)	
	20 5.374	128 77.75.75.176	192.168.0.112	ICMP	74	Echo (ping)	reply	id=0x0001,	seq=82/20992,	ttl=54 (req	uest in 1	19)	
	22 6.198	115 192.168.0.11	2 77.75.75.176	ICMP	74	Echo (ping)	request	id=0x0001,	seq=83/21248,	ttl=128 (re	ply in 23	3)	
	23 6.267	895 77.75.75.176	192.168.0.112	ICMP	74	Echo (ping)	reply	id=0x0001,	seq=83/21248,	ttl=54 (req	uest in 2	22)	
	24 7.213	596 192.168.0.11	2 77.75.75.176	ICMP	74	Echo (ping)	request	id=0x0001,	seq=84/21504,	ttl=128 (re	ply in 25	5)	~
	75 7 770	456 77 75 75 176	192 168 0 112	TCMP	7/	Echo (ping)	renty	14-920001	EAG-84/21504	++1-54 (red	lert in 1	2/11	
	Distancel i	TCMD (1)											^
	Header ch	iche (1) ocksum: 0x0741 [u	alidation disabled]										
	[Header ch	ecksum status: U	nvecified]		Příka	azový řádek				_		×	
	Source: 7	.75.75.176	inter 111200 j		Pinging	g seznam.cz	[77.75	.75.176] w	ith 32 bytes	of data:		~	
	Destinatio	on: 192.168.0.112			Reply +	from 77.75.	75.176:	bytes=32	time=30ms TT	L=54			
\sim	Internet Con	trol Message Prot	ocol		Reply +	from 77.75,	75.176:	bytes=32	time=192ms T	TL=54			
	Type: 0 (1	cho (ping) reply)		Reply +	from 77.75.	75.176:	bytes=32	time=69ms TT	L=54			
	Code: 0	1 07 T 7	·		Reply +	from 77.75.	75.176:	bytes=32	time=65ms TT	L=54			
	Checksum:	0x550a [correct]											
	[Checksum	Status: Good]			Ping st	tatistics f	or 77.7	5.75.176:	4 Last 0	(0%])			
	Identifie	(BE): 1 (0x0001)		Apppov	imata nound	= 4, R	eceived = /	4, LOST = 0	(0% 1055),			
	Identifie	(LE): 256 (0x01	00)		Mi	nimum – 30m	i trip t. ic Mavii	mum - 102m	s Average -	80mc			
	Sequence i	number (BE): 81 (0x0051)			12.000 - 500	o, novi	1920	o, Archage -	00110			
	Sequence r	umber (LE): 2073	6 (0x5100)		C:\User	rs\student;	_						
	[Request 1	rame: 16]					-				_	¥	
	[Response	time: 30.533 ms]											
	> Data (32	oytes)											¥
C) 📝 🛛 Time to liv	e (ip.ttl), 1 byte						Packets:	29 · Displayed: 10 (34.5%) · Dropped	: 0 (0.0%)	Profile: Def	ault 🔡

Obr. A.4: Základní zachycení protokolu ICMP a DNS po zadání příkazu ping v příkazové řádce.

Úkoly:

- (3) Zjistěte typ použitého transportního protokolu a také číslo cílového portu DNS serveru.
- (4) Kolik bajtů má segment ICMP protokolu bez části nesoucí informace o IP protokolu, včetně 32 B přidaných dat?

A.2.3 Filtry a jejich kombinace

Jak již bylo zmíněno, do pole filtru můžeme psát mnoho různých sekvencí příkazů a Wireshark nám následně zobrazí jen požadované pakety, které odpovídají filtraci. Níže můžete vidět několik příkladů pro filtraci pomocí IP adres, portů nebo protokolů a také kombinace těchto sekvencí. Při zadávání filtrů vidíme, že se pole barví podle textu který píšeme. Zelené podbarvení značí, že sekvence je napsána správně a po stisknutí klávesy enter se nám zobrazí pakety splňující pravidlo zadaného filtru. V případě, že je podbarvení žluté, tak není sekvence napsána úplně jednoznačně a je třeba ji upravit. To může být například delší sekvencí bez závorek, kdy Wireshark zobrazí pakety podle zadaného filtru, ale zápis není jednoznačný a tak nemusí být zobrazeny pouze ty pakety, které požadujeme. Červené podbarvení značí chybu a sekvenci je nutné opravit. Lze používat známé operátory nebo jejich ekvivalenty jako jsou například: == (eq), && (and), || (or), != (ne), > (gt), < (lt)... Všechny zkratky pochází z anglických výrazů, například gt - "greater than" nebo lt - "less than".

Výpis všech paketů kde zdrojová nebo cílová adresa je 192.168.0.1. ip.src==192.168.0.1 or ip.dst==192.168.0.1 Nebo kratší způsob: ip.addr==192.168.0.1

Zobrazí pakety, které obsahují protokol ICMP nebo DNS a zároveň je jejich zdrojová adresa 192.168.0.1.

(ip.src==192.168.0.1) and (icmp or dns)

Tento výpis je dost podobný tomu předchozímu, ale závorka je ukončena jinde a tedy i výpis bude odlišný. Proto je potřeba dávat na umístění závorek pozor. Zde dojde k vypsání ICMP paketů se zdrojovou adresou 192.168.0.1 a všech DNS paketů. (ip.src==192.168.0.1 and icmp) or dns

Další možnost jak zobrazit pouze DNS pakety. Ty totiž využívají právě port 53. Tento filtr tedy zobrazí UDP pakety, kde zdrojový nebo cílový port má číslo 53. udp.port eq 53

A.2.4 Úprava parametrů pro ping a zachycení

V další části budeme upravovat parametry pro příkaz ping a následně budeme analyzovat co vše se s upravenými parametry mění ve Wiresharku.

V příkazové řádce otestujte funkci DNS serveru. První možností, jak využít funkci DNS serveru, je použití jednoho z parametrů pro ping. V tabulce A.2 nalezněte vhodný parametr pro provedení pingu a zjištění doménového jména. V příkazové řádce zadejte ping s nalezeným parametrem na IP adresu 77.75.75.176 a celý proces komunikace protokolů ICMP a DNS zachyťte ve Wiresharku. Následně zjistěte adresu použitého DNS serveru.

Tab. A.2: Volitelné parametry pro příkaz ping.

	Ping se bude opakovat dokud nebude manuálně zastaven								
	(Ctrl+C), zároveň lze u každé odpovědi pomocí Ctrl+Break								
-t	zobrazit statistiky o počtu odeslaných a přijatých paketů, z toho								
	lze odvodit ztrátovost (packet loss) a také maximální, minimální								
	a průměrná doba odezvy.								
-n <číslo>	Kde číslo určuje počet odeslaných echo requestů.								
	V případě, že budeme provádět ping na IP adresu, tak nám tento								
-a	parametr vrátí i doménové jméno, ke kterému tato adresa patří.								
	Kromě ICMP protokolu se tak využije i protokol DNS.								
	Lze nastavit velikost dat, které se s pingem odešlou, toto můžeme								
1 <rrelilroat></rrelilroat>	využít když řešíme problém se sítí. Pakety s malou velikostí (běžně								
-1 < venkost>	se posílají 32 bajtové) můžou sítí projít, ale s většími objemy dat								
	může být při přenosu problém nebo se může lišit doba odezvy.								
f	Parametr určí, že chceme aby se pakety nefragmentovaly (Do not								
-1	Fragment flag). Použitelné pouze pro IPv4.								
i cătalos	Parametr omezí ping na zadaný počet přeskoků TTL (Time To								
-1 < cisio >	Live) viz kapitola o TTL.								
-4	Parametr vynutí použití IPv4.								
-6	Parametr vynutí použití IPv6.								
2	Parametr vypíše tyto a také další, méně používané parametry pro								
- !	ping.								

V případě, že jste příkaz v příkazové řádce zadali vícekrát, počítač si IP adresu a k ní odpovídající doménu uchová v paměti, takže se ve Wiresharku příslušná DNS komunikace pravděpodobně již neukáže a nemůžete tedy najít adresu použitého DNS serveru. Pro tento případ použijte příkaz pro vymazání DNS záznamů z paměti:

ipconfig -flushdns

Následný příkaz ping s vhodným parametrem opět vyvolá funkci DNS resolveru.

Další možností zjištění doménového jména z IP adresy a naopak je příkaz nslookup. Výhoda tohoto příkazu je, že při vyhledávání IP adres podle doménového jména dostaneme v odpovědi všechny dostupné adresy, které server uchovává a to včetně IPv6 adres.

nslookup <ip adresa nebo doménové jméno>

Opět spusťte zachytávání ve Wiresharku, zadejte příkaz nslookup do příkazové řádky a zjistěte přes jaké adresy je možné se dostat na webové stránky mapy.cz. Zastavte zachytávání ve Wiresharku a zjistěte kolik DNS odpovědí (response) bylo potřeba pro přenesení všech IP adres po našem dotazu v příkazové řádce. Měli bychom dostat odpověď podobně jako na obrázku A.5.

```
Administrator Příkazový řádek - □ ×
C:\WINDOWS\system32>nslookup seznam.cz
Server: arekol.kn.vutbr.cz
Address: 147.229.190.143
Non-authoritative answer:
Name: seznam.cz
Addresses: 2a02:598:4444:1::2
2a02:598:4444:1::1
77.75.75.172
77.75.75.176
```

Obr. A.5: Zobrazení příkazu nslookup v příkazové řádce při překladu doménového jména.

Nyní si vyzkoušíme příkaz **ipconfig** -displaydns, který vytvoří výpis právě uložených domén, včetně podrobnějších informací. V paměti se těchto informací nachází mnoho, proto nebudeme výpis příkazu zobrazovat v příkazové řádce, ale uložíme jej do textového souboru na plochu.

Otevřete libovolný webový prohlížeč. Následně vymažte dosud uložené DNS informace v paměti příkazem **ipconfig -flushdns**. V prohlížeči zobrazte webovou stránku seznam.cz. Tím dojde k využití DNS serveru a do paměti počítače bude uložen požadovaný DNS záznam. Nyní zkontrolujte, že se v příkazové řádce nacházíte ve složce student a uložte výpis -displaydns do textového souboru pomocí následujícího příkazu:

C:\Users\student>ipconfig -displaydns > desktop\soubor.txt

Zobrazte si soubor uložený na ploše pomocí poznámkového bloku. Přes klávesovou zkratku Ctrl+F vyhledejte záznam na doménu seznam.cz. Výpis by měl vypadat podobně, jako na obr. A.6 a A.7.



Obr. A.6: Příkazy pro výpis DNS cache do textového souboru a zobrazení vytvořeného souboru s podrobnostmi o uložených doménách.

Jak si můžete povšimnout, tak kromě záznamu o seznam.cz se ve výpisu nachází mnoho dalších záznamů. Je to tím, že se u prohlížení webové stránky předpokládá, že budete pokračovat na stránku, na kterou je odkazováno na úvodní stránce seznam.cz. Takže se rovnou pošlou adresy pro přidružené stránky seznam.cz, jako jsou mapy.cz, sauto.cz, firmy.cz a další.

V záznamech je vždy položka record name, ta obsahuje název domény. Record type nejčastěji uvádí, zda se se serverem komunikuje pomocí IPv4 (type 1) nebo IPv6 (type 28). Další záznam je Time to Live, nejedná se však o počet možných přeskoků, jako u pingu. Je to časový údaj v sekundách, kdy se po vypršení času maže daný záznam z paměti. Data length opět označuje, které adresy jsou IPv4 (4 bajty) nebo IPv6 (16 bajtů). Další položka section může obsahovat dvě hodnoty. Answer, v případě, že daná adresa je odpovědí na DNS request a druhá možnost additional, kdy DNS response přidá tyto adresy navíc, protože by se mohly hodit v budoucnu. Poslední záznam je samotná IP adresa, která odpovídá doméně. Zobrazení záznamu po uložení do souboru je na obr. A.7.

 Record Name
 : www.seznam.cz

 Record Type
 . . . : 1

 Time To Live
 . . . : 198

 Data Length
 . . . : 4

 Section
 . . . : Answer

 A (Host) Record
 . . : 77.75.75.176

Obr. A.7: Výpis DNS cache v textovém souboru včetně hodnoty TTL a dalších podrobností.

Úkoly:

- (5) Zjistěte název domény, která se nachází na IP adrese 93.185.98.100.
- (6) Kolik bajtů dat je potřeba na přenesení názvu domény v DNS response paketu z předchozího úkolu?

Další část se bude věnovat kontrole času, za který se nám od pingu vrátí echo reply. Vyzkoušejte a nalezněte časovou jednotku latence (response time) protokolu ICMP v paketech echo reply ve Wiresharku. Jak se liší hodnota udávaná v příkazové řádce od té ve Wiresharku?

Nyní vyzkoušíme další z parametrů pro ping. Parametrem -l <číslo 0-65500> můžeme měnit velikost zaslaných dat společně s pingem.

ping -4 -l <číslo 0-65500> <ip adresa>

V případě, že na vutbr.cz chceme spolu s ICMP protokolem odeslat více než určitý počet bajtů dat, tak se data rozfragmentují. Fragmentace slouží k rozdělení větších paketů na menší části (fragmenty) tak, aby se daly přenést po síti. Této maximální velikosti paketu říkáme MTU (z anglického Maximum transmission unit), tedy maximální jednotka, která lze po síti Ethernet II přenést. MTU pro rozhraní počítače můžete zkontrolovat pomocí příkazu:

netsh interface ipv4 show interface



Obr. A.8: Výpis příkazu pro kontrolu hodnoty MTU v příkazovém řádku.

V testovacím příkladu je hranice fragmentování někde nad 1400 bajty, ale toto číslo se může lišit podle aktuální situace (umístění v síti, testovaný server, operační systém...). Zároveň by ale hranice neměla překročit hodnotu MTU (většinou 1500 B). Spusťte zachytávání ve Wiresharku a zadejte ping s parametrem -l a velikost 1400 B. Ve Wiresharku vidíme, že jsou data pohromadě v jednom ICMP paketu. Celková velikost ICMP paketu je 1442 bajtů. Z toho 1400 bajtů jsou námi požadovaná data, 8 bajtů má záhlaví ICMP, 20 bajtů část IPv4 a 14 bajtů zapouzdřená část pro Ethernet.

Nyní zadejte za parametr -l 1600 B, spusťte zachytávání paketů ve Wiresharku a spusťte ping v příkazovém řádku. Po dokončení zastavíme zachytávání a vložíme filtr pro zobrazení pouze ICMP paketů. Zobrazte si podrobnosti paketu echo request > Internet Protocol Version 4 > 2 IPv4 Fragments, stejně jako na obr. A.9. Vidíme, že se data již musela rozfragmentovat, aby mohla projít k cíli.

V případě, že máme zadaný filtr pro zobrazení pouze ICMP paketů, tak se nám další fragmenty ve výpisu neukazují. To můžeme změnit kliknutím pravým tlačítkem myši na ICMP paket > Conversation filter > IPv4. Nyní již vidíme vše, co se odesílalo s příkazem ping v příkazové řádce, podobně jako na obr. A.9. Paketům ICMP předchází fragmenty s přebývajícími daty, ty již neobsahují ICMP záhlaví a tak jsou podbarveny bíle a nesou označení IPv4 protokolu.

-																		_	
-	10 - ping	j -l 1600	na Vu	itbr.pcapn	9												_		×
File	Edit	View	Go	Capture	Analyz	e Statis	tics Te	lephony	Wireless	Tools	Help								
⊿	I	۲	1 110	🗙 😂	۹ 👳	2 🖘	1		ΘΘ	Q. 🔢									
	p.addre	q 192. 16 8	3.0.11	12 and ip.ad	dr eq 147.	229.2.90												$X \rightarrow$	+
No.	Ti	ne	1	Source		Destinat	ion	Protocol	Length	Info									1
	6 1.	591719	• i	192.168.0	9.112	147.22	9.2.90	IPv4	1514	Fragm	ented	IP protoc	ol (prot	D=ICMP 1, of	F=0, I	D=5ab0)	[Rease	embled	
Г	7 1.	591720)	192.168.0	3.112	147.22	9.2.90	ICMP	162	Echo	(ping)	request	id=0x00	01, seq=69/1	7664,	tt1=128	(reply	/ in 9)	
	8 1	623951	1	147.229.3	2.90	192.16	8.0.11	2 IPv4	1490	Fragm	ented	IP protoc	ol (prot	D=ICMP 1, of	F=0, I	0=4449)	[Reass	embled	
	91.	623952		147.229.3	2.90	192.16	8.0.11	2 ICMP	186	Echo	(ping)	reply	id=0x00	01, seq=69/1	7664,	tt1=52 ((reques	t in 7	
1	12 2	625057		192.168.0	9.112	147.22	9.2.90	IPv4	1514	Fragm	ented	IP protoc	of (prot	D=ICMP 1, 0+	F=0, 1	U=5aD1)	[Reass	embled	
*	15 2.	650243		192.108.0	2.112	147.22	9.2.90	2 TD4	162	EC	Mark	Unmark Da	-kat(r)	Ctrl=M	20,	0=4442)	(Pepi)	in 15	
	15 2	650343		147 229	2 90	192.16	8 0 11	2 1FV4	186	Fr					10	++1=52 (reques	t in 1	
	18 3	640599		192.168.0	3.112	147.22	9.2.90	TPv4	1514	Er	Ignor	e/Unignore	Packet(s)	Ctrl+D	. 1	D=5ab2)	(Reass	embled	
	19 3.	640600		192.168.0	9.112	147.22	9.2.90	ICMP	162	EC	Set/U	nset Time Re	eference	Ctrl+T	6,	tt1=128	(repl)	in 21	
	20 3	680144		147.229.3	2.90	192.16	8.0.11	2 IPv4	1490	Fr	Time	Shift		Ctrl+Shift+T), I	D=444b)	[Reass	embled	×
<											Packe	t Comment		Ctrl+Alt+C				>	
>	Frame :	13: 162	2 byt	tes on wi	ire (12	96 bits	. 162	bytes ca	ptured	(12					28-	385F-466	57 - 8D33	-4C9AD	C633F4
>	Ethern	t II,	Src	VMware	e5:42:1	4 (00:0	dc:29:e	5:42:b4)	, Dst:	Tp-	Edit R	esolved Nar	ne						
¥	Intern	et Prot	loco]	L Version	1 4, Sr	:: 192.1	168.0.1	12, Dst:	147.22	9.2					- -				
	0100		= Ve	ersion: 4							Apply	as Filter		•					
		0101	= He	ader Ler	igth: 20	0 bytes	(5)				Prepa	re as Filter		•					
	> Dift Tota	Ferenti 1 Leng	ated	i Service 148	s Field	d: 0x00	(DSCP:	CSØ, EC	N: Not-	ECT	Conv	ersation Filte	2r	•		CIP Conn	ection		1
	Ider	ntifica	itior	: 0x5ab1	(23217	7)					Color	ize Conversa	ation	•		Ethernet			
	> Flag	s: 0x0	0b9								SCTP			•		F5 TCP			
	Fnag	ment o	ffse	et: 1480							Enllow ESTIDE								
	Time	to li	ve:	128							10101					55.10			
	Heat	er che	1CMP	(1) 		dation	dicabl	lbe			Сору			•		FOIP			
	FHeat	der che	ecks	um statu	is: Unw	erified)	015801	euj								IEEE 802.1	5.4		
	Sour	ce: 19	2.16	8.0.112							Proto	col Preferen	ces	•		IPv4			
	Dest	inatio	n: 1	47.229.2	.90						Deco	de As				IPv6			
	¥ [2]	Pv4 Fr	agme	ents (160	8 bytes	s): #12(1480),	#13(128)]		Show	Packet in N	ew Windov			TCD			
	[Frame: 12, payload: 0-1479 (1480 bytes)]									_					-	ICP			
	[Frame: 13, pavload: 1480-1607 (128 bytes)]															UDP			L
		Fragme	nt c	ount: 2]												ZigBee N	etwork L	ayer	
	[Keassembled IPV4 Length: 1608] [Reassembled IPv4 data: 08004309000100466163636465666768										che a car					PN-IO AR			L
s i	Intern	Reasse	mole	Message	Brotocy	0043096 51	001004	001026364	+00066/	000968	000060	реот/0]			1.1	DNLIO AR	(with d	atal	
é	incerin	ic cont		nessage	FIOLUCI											ENHO AN	(with a	ataj	,
-	· ·			Manager	a har and the		hu dan (n)						II nadat	er an e Dieslauede		PN-CBA			-
0	1	INFINET CO	DINITO	PIESSAGE M	DIDCOLUCT	nnu. (608	DVD (S)						Packet	stau tushavent	10 10 1.	1701	_	TIME: 12	TOUR

Obr. A.9: Zobrazení celé konverzace ve Wiresharku po zadání příkazu ping s parametrem -l 1600 B.

První fragment obsahuje 1472 bajtů přidaných dat a 8 bajtů ICMP záhlaví a druhý 128 bajtů přidaných dat. Tyto čísla se mohou lišit, jak bylo zmíněno výše. Celkem tedy 1472 bajtů a 128 bajtů přidaných dat dává dohromady našich požadovaných 1600 bajtů (1472 + 128 = 1600).

Také si povšimněte označení nalevo od sloupce zdrojové IP adresy. Máme označený paket echo request. Šipkou se nám zvýrazní odpověď v podobě echo reply a tečkou jsou označeny fragmenty, které souvisí s označeným ICMP paketem.

Úkoly:

- (7) Na kolik fragmentů se rozdělí přidaná data v případě odeslání maximálního možného objemu dat příkazem ping na vutbr.cz? Bohužel virtualizovaný OS nedokáže přijmout zprávy ICMP echo reply s takovou velikostí přidaných dat. Využijte tedy příkazovou řádku i Wireshark z hostovského operačního systému.
- (8) Pro ping na doménu vutbr.cz nalezněte maximální hodnotu parametru -l, kdy ještě nedojde k fragmentaci a minimální hodnotu -l, kdy už ke fragmentaci dojde. Čím jsou tyto hodnoty dány?

U úkolu (8) si všimněte, že u ICMP odpovědi echo reply se liší velikost paketů oproti požadavku echo request. Větší velikost je způsobena tím, že je potřeba zvětšit velikost paketu na minimální možnou délku (většinou 60 bajtů). Když si zobrazíme jednotlivé bajty tohoto paketu, tak uvidíme v ethernetovém záhlaví výplň (padding), stejně jako na obr. A.10. Výplň je tvořena pouze nulami. U původního ICMP požadavku echo request se tato výplň přidává také, ale až potom co paket zachytil Wireshark.

A.2.5 Zachycení ICMP - zachycení chybového kódu

V další části zkombinujeme příkazy tracert a ping. Tracert nám v příkazové řádce vypíše cestu k danému síťovému prvku a to včetně doby odezvy a IP adresy jednotlivých uzlů. Bohužel v otevřené virtuálce má příkazová řádka problém vypsat jak odezvu, tak IP adresy uzlů, kromě prvního a posledního uzlu. Proto si příkaz **tracert** vyzkoušejte v hostovském operačním systému. Pozor na to, že počet uzlů zde bude o jeden menší, než kdybychom příkaz zadali ve virtualizovaném OS. Tracert můžeme využít v případě problému s delší odezvou, kdy pomocí odezvy každého prvku určíme problematický uzel. Také můžeme zkontrolovat, zda vyšší latence byla jen jednorázová, nebo se v síti vyskytuje nějaký větší problém.

Pomocí tracert jsme zjistili počet směrovačů na trase směrem k cíli. Nyní zkuste upravit příkaz ping v příkazové řádce v hostovském operačním systému tak, aby první nastavená hodnota (X) způsobila, že echo request dorazí do cíle a vrátí

	*Intel(R) Ethernet Con	nection (2) I219-V: Internet			- 0	×
File	Edit View Go	Capture Analyze Statis	tics Telephony Wireless	Tools	Help	
	🔳 🔬 🕥 📙 💼	🖹 🔄 ९ 👄 🔿 😫	T 🕹 📃 🗏 Q Q	् 🎹		
	o.addr eq 147.229.149.	190 and ip.addr eq 77.75.75.17	72		🔀 📼 💌 Expression	. +
No.	Time	Source	Destination	Protoco	Length Info	~
	97 0.968617	147.229.149.190	77.75.75.172	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID	
>	98 0.968618	147.229.149.190	77.75.75.172	ICMP	35 Echo (ping) request id=0x0001, seq=9/2304, ttl	
1	99 0.972482	77.75.75.172	147.229.149.190	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID	
	100 0.972483	77.75.75.172	147.229.149.190	ICMP	60 Echo (ping) reply id=0x0001, seq=9/2304, ttl	
	194 1.980690	147.229.149.190	77.75.75.172	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID	
	195 1.980692	147.229.149.190	77.75.75.172	ICMP	35 Echo (ping) request id=0x0001, seq=10/2560, tt	
	196 1.984511	77.75.75.172	147.229.149.190	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID	
	197 1.984512	77.75.75.172	147.229.149.190	ICMP	60 Echo (ping) reply id=0x0001, seq=10/2560, tt	
	315 2.996445	147.229.149.190	77.75.75.172	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, IC	
	316 2.996447	147.229.149.190	77.75.75.172	ICMP	35 Echo (ping) request id=0x0001, seq=11/2816, tt	
	317 3.000430	77.75.75.172	147.229.149.190	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, IC	
<					>	
> *	rame 100: 60 byt thernet II, Src: > Destination: M	es on wire (480 bits), HewlettP_09:0f:c7 (d8 icro-St_9a:e6:83 (30:9	, 60 bytes captured (4 3:94:03:09:0f:c7), Dst 0c:23:9a:e6:83)	80 bits) : Micro-	on interface 0 St_9a:e6:83 (30:9c:23:9a:e6:83)	^
	> Source: Hewlet	tP_09:0f:c7 (d8:94:03:	09:0f:c7)			
	Type: IPv4 (0x	0800)				
	Padding: 00000	000000000000000000000000000000000000000	000000000000000000000000000000000000000			
$ $ \vee	Internet Protocol	Version 4, Src: 77.75	5.75.172, Dst: 147.229	.149.190		
	0100 = Ve	rsion: 4				
	0101 = He	ader Length: 20 bytes	(5)			
	> Differentiated	Services Field: 0x00	(DSCP: CS0, ECN: Not-	ECT)		
	Total Length.	21				-
000	0 30 9c 23 9a et	6 83 d8 94 03 09 0† d	7 08 00 45 00 0.#	· · · · · · · ·	··E·	
001	0 00 15 50 75 0	0 D9 36 01 6D 11 40 4		0 K MK		
003	0 00 00 00 00 00	0 00 00 00 00 00 00 00 00 00 00 00 00 0	0			
Fra	me (60 bytes) Reas	sembled IPv4 (1481 bytes)	Not dissected data bytes (147:	B bytes)		
\bigcirc	Z Ethernet Padding	(eth.padding), 25 bytes			Packets: 17234 · Displayed: 32 (0.2%) · Dropped: 0 (0.0%) Profile: Def	ault 🔡

Obr. A.10: Přidaná výplň v ethernetovém záhlaví pro naplnění minimální velikosti paketu.

obvyklé hodnoty. Následně zvolte druhou hodnotu (Y) tak, aby vznikla chyba při přenosu paketu **echo request** a v konzoli jsme dostali výpis "TTL expired in transit", stejně jako na obr. A.11.

ping -4 -i <číslo X nebo Y> vutbr.cz

Úkoly:

- (9) Druhý případ zachyťte ve Wiresharku a vyhledejte typ a kód chyby ICMP protokolu, který souvisí se zprávou "TTL expired in transit".
- (10) Ve výpisu paketů vidíme chybové protokoly ICMP (v základním nastavení podbarveny černě), které v informační části mají popisek Time-to-Live exceeded. Kdo je odesilatelem těchto paketů z hlediska síťové vrstvy?

Dále vyzkoušíme ping request s upravenou velikostí dat, podobně jako u úkolu (7) a (8). Nyní ovšem pošleme větší objem dat. Otestujte, jaké z velikostí dat (15000 B, 25000 B a 48000 B) lze nebo nelze poslat při pingu na seznam.cz a vutbr.cz.

```
Administrator Pfikazovy idek
C:\WINDOWS\system32>ping -4 -i 7 seznam.cz
Pinging seznam.cz [77.75.75.172] with 32 bytes of data:
Reply from 91.210.16.194: TTL expired in transit.
Ping statistics for 77.75.75.172:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

Obr. A.11: Chybový výpis při zadání malého počtu dovolených přeskoků. Tudíž došlo k výpisu chyby, že hodnota TTL vypršela při přenosu sítí.

Na některé servery nám ping bude fungovat i s maximální možnou velikostí zasílaných dat (vutbr.cz), ale od jiných (seznam.cz) se nám nedostane odpovědi. Dotazovaný server se tím chrání před možným zahlcením v případě, že bychom příkaz poslali mnohokrát v krátkém časovém úseku. Takové zahlcení by mohlo mít za následek zpomalení nebo zastavení dotazovaného systému. Zadejte příkaz ping s parametrem -l 25000 a zachytte pakety ve Wiresharku.

```
ping -4 -1 25000 seznam.cz
```

Pro vyfiltrování potřebných paketů zvolte tento filtr: ip.addr == <IP adresa seznam.cz zobrazená v příkazového řádku> například: ip.addr == 77.75.75.172

Odešlou se 4 echo requesty, příkazový řádek nás informuje o tom, že požadavek vypršel a nelze kontaktovat cílovou stanici. Ve Wiresharku vidíme pouze echo requesty a fragmenty, které dávají dohromady našich 25000 B dat, tak jako na obr. A.12. Zprávy echo request zůstávají bez odpovědi od dotazovaného serveru domény seznam.cz.

Úkoly:

(11) Kolik bajtů je potřeba pro přenos hodnoty TTL a zdrojové a cílové IP adresy v IP záhlaví?

Nyní vyzkoušejte parametr -f u pingu. Parametr nám nastaví to, aby v sobě pakety obsahovaly informaci, že nechceme aby byly fragmentovány. Otestujte parametr -f u pingu na vutbr.cz v kombinaci s volitelnou velikostí přidaných dat

4	'Wi-Fi								- 0	×
File	Edit	View	Go Capture	e Analyz	e Statistics	Telep	hony	Wireless	s Tools Help	
4		6	- E E	Q (m)	⇒ 🖘 ō	. =		ΘΘ		
	= <u>as</u>	e L		• •	~ = 0	~ -		~ ~	м <u>т</u>	_
	p.addr ==	= 77.75.7	5.172						X	<u> </u>
No.	Tim	e	Source		Destination		Protocol	Length	Info	^
	68 1.6	645025	192.168	0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=13320, ID=6644) [Reassembled	i
	69 1.0	645026	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=14800, ID=6644) [Reassembled	i
	70 1.0	645026	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=16280, ID=6644) [Reassembled	i
	71 1.0	645026	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=17760, ID=6644) [Reassembled	i
1	72 1.4	645026	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=19240, ID=6644) [Reassembled	i –
	73 1.6	645027	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=20720, ID=6644) [Reassembled	i
	74 1.4	645027	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=22200, ID=6644) [Reassembled	i
	75 1.0	645027	192,168	.0.100	77.75.75	.172	ICMP	1362	Echo (ping) request id=0x0001, seq=195/49920, ttl=64 (no response four	10
	91 6.0	645071	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=6645) [Reassembled in #	/1
	92 6.0	645072	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=6645) [Reassembled i	in in
	93 6.0	645072	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=6645) [Reassembled i	in
	94 6.0	645075	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=4440, ID=6645) [Reassembled i	.n
	95 6.0	645075	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=5920, ID=6645) [Reassembled i	.n
	96 6.0	645075	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=7400, ID=6645) [Reassembled i	.n
	97 6.0	645076	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=8880, ID=6645) [Reassembled i	.n
	98 6.0	645076	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=10360, ID=6645) [Reassembled	i
	99 6.0	645076	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=11840, ID=6645) [Reassembled	1
	100 6.0	645076	192.168	.0.100	//./5./5	.1/2	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=13320, ID=6645) [Reassembled	1
	101 6.0	645077	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=14800, ID=6645) [Reassembled	1
	102 6.0	645077	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=16280, ID=6645) [Reassembled	1
	103 6.0	645077	192.168	.0.100	//./5./5	.1/2	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1//60, ID=6645) [Reassembled	1
	104 6.4	645078	192.168	.0.100	//./5./5	.1/2	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=19240, ID=6645) [Reassembled	1
	105 6.0	645078	192.168	.0.100	77.75.75	.172	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=20/20, ID=6645) [Reassembled	1
	105 6.0	645078	192.168	.0.100	//./5./5	.1/2	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=22200, ID=6645) [Reassembled	1
	10/ 6.	645079	192.168	.0.100	//.75.75	.1/2	TCMP	1362	Ecno (ping) request id=0x0001, seq=196/50176, tt1=64 (no response four	IC
	124 11	. 64414	5 192.168	.0.100	//./5./5	.1/2	IPv4	1514	Fragmented IP protocol (proto=1LMP 1, off=0, ID=6646) [Reassembled in # Fragmented ID protocol (proto=1LMP 1, off=0, ID=6646) [Reassembled in #	1
	125 11	.04414	5 192.168	0.100	77.75.75	.172	1PV4	1514	Fragmented IP protocol (proto=itmP 1, 0ff=1480, ID=6646) [Reassembled 1]	- v
<	120 11	.04414	/ 192.168	.0.100	11.15.15	.172	rPv4	1514	Fragmented in protocol (proto=itme 1, 0ff=2960, 1D=6646) [Reassembled 1]	.n
-					****		_			C 10
	📧 Wi	esnark \	///-=1 2020040.	2114003 al	HUGO, DCaping				Packets; 92041 * Displayed; 68 (0, 1%) * Dropped; 0 (0,0%) Profile; D	Jerault .:

Obr. A.12: Zobrazení echo požadavků s objemnou datovou částí rozdělenou do více fragmentů v podobě paketů bez ICMP záhlaví.

parametrem -l, vyzkoušejte data o délce 1400 B a 1600 B. Oba případy zachyťte ve Wiresharku a zodpovězte níže uvedené úkoly.

V prvním případě, kdy se odesílá 1400 bajtů dat, tak se ve Wiresharku zdá vše stejné jako u běžného pingu. Jedinou změnou je nastavení bitu "Don´t fragment" z 0 na 1, jak je ukázáno i na obr. A.13. Tuto změnu vyvolává právě námi nastavený parametr -f. Při zadání hodnoty 1600 bajtů u parametru -l a zároveň při vypnuté fragmentaci dochází k okamžitému vypsání chyby. K vypsání chyby dochází ihned, protože paket se ani neodešle ze síťové karty našeho počítače, protože je překročena hodnota MTU.

Úkoly:

- (12) Jaká je maximální velikost přidaných dat u pingu na vutbr.cz, která lze nastavit při vypnuté fragmentaci a zároveň nevypíše chybu v příkazové řádce?
- (13) Jaká chyba se vypíše v příkazové řádce v případě, že chceme bez fragmentace odeslat větší množství dat než je hodnota MTU? Hodnotu MTU zjistěte přes již dříve uvedený příkaz netsh interface ipv4 show interface.
- (14) Kolik ICMP paketů bylo zachyceno Wiresharkem v případě odesílání 1600 bajtů přidaných dat u pingu na vutbr.cz s parametrem -f?

	*Wi-F	i													-		×	
Fil	e Ec	lit View	Go Capture	Analyze	e Statistics	Tele	phony	Wireless	Tools	Help								
		10		۹ 😓	€ 😤 🗧	. €		ଇ୍ପ୍	Q 🎚									
	icmp															×→	• +	
No		Time	Source		Destination		Protocol	Length	Info								^	
7	6	0.152701	192.168.0	.100	147.229.3	2.90	ICMP	1442	Echo	(ping)	request	id=0x0001,	seq=168/43008	3, ttl=64	(reply	/in 7)		
*	7	0.177986	147.229.2	.90	192.168.0	0.100	ICMP	1442	Echo	(ping)	reply	id=0x0001,	seq=168/43008	3, tt1=52	(reque	est in		
	10	1.157011	192.168.0	.100	147.229.	2.90	ICMP	1442	Echo	(ping)	request	id=0x0001,	seq=169/43264	1, ttl=64	(reply	/in 11		
	11	1.178524	147.229.2	.90	192.168.0	0.100	ICMP	1442	Echo	(ping)	reply	id=0x0001,	seq=169/43264	1, ttl=52	(reque	est in		
	14	2.160703	192.168.0	.100	147.229.3	2.90	ICMP	1442	Echo	(ping)	request	id=0x0001,	seq=170/43520	0, ttl=64	(reply	/in 15		
	15	2.185856	147.229.2	.90	192.168.0	0.100	ICMP	1442	Echo	(ping)	reply	id=0x0001,	seq=170/43520	0, tt1=52	(reque	est in		
	16	3.164710	192.168.0	.100	147.229.3	2.90	ICMP	1442	Echo	(ping)	request	id=0x0001,	seq=171/43776	5, ttl=64	(reply	/in 17	×	1
<																>		
>	Fram	e 6: 1442	bytes on wi	re (115	36 bits),	1442	bytes	capture	d (119	536 bit	s) on int	terface \Dev	ice\NPF_{C93B0	A0A-6D8F-	458E-9	605-E61	209 🗚	Ī
>	Ethe	rnet II,	Src: IntelCo	r_ab:8c	:f2 (74:e	5:0b:a	ab:8c:f	2), Dst	: Tp-L	.inkT_3	1:62:fc	(68:ff:7b:31	:62:fc)					
~	Inte	rnet Prot	ocol Version	4, Src	: 192.168	.0.100), Dst:	147.22	9.2.90	3								
	0	100	= Version: 4															
		0101	= Header Len	gth: 20	bytes (5)											- 5	
	∨ D	ifferenti	ated Service	s Field	: 0x00 (D	SCP: 0	50, EC	N: Not-	ECT)									
		0000 00	= Differe	ntiated	Services	Codep	oint: I	Default	(0)									
			00 = Explici	t Conge	stion Not:	ificat	ion: No	ot ECN-	Capabl	e Tran	sport (0))						
	T	otal Leng	th: 1428															
	I	dentifica	tion: 0xf9d8	(63960)													
	Y ₽	lags: 0x4	000, Don't f	ragment														
		0		= Rese	rved bit:	Not s	et											
		.1		= Don'	t fragmen	t: Set												
		0		= More	fragment	s: Not	set											
	F	ragment o	ffset: 0															
	Т	ime to li	ve: 64															
	P	rotocol:	ICMP (1)														~	1
<																	>	
	2	Internet Pr	otocol Version 4 (i	p), 20 byt	te(s)						Pa	ickets: 18 · Displa	ayed: 8 (44.4%) · D	ropped: 0 (0.0	0%) P	Profile: De	ault	

Obr. A.13: Záhlaví IP protokolu informující o tom, že byl použit parametr -f. Bit "don´t fragment" je nastaven na 1 a přidaná data tak nebudou fragmentována na více částí.

A.2.6 Další prvky analýzy ICMP záhlaví

Jak je vidět na obr. A.14, tak v ICMP záhlaví se kromě čísla typu a kódu uvádějí také další hodnoty, kterými se nyní budeme zabývat.

Jako první po typu a kódu v ICMP záhlaví následuje "checksum" - kontrolní součet. Pomocí této hodnoty, se kontroluje na straně příjemce, zda k němu dorazila všechna data daného paketu. Nejdříve se na straně odesílatele rozdělí prvky ICMP záhlaví a přidaná data (payload) do 16 bitových čísel (čísel v hexadecimálním zápisu). Na místo kontrolního součtu je vložena hodnota 0x0000. Následně se vypočítá součet jedničkových doplňků (one's complement sum) prvních dvou čísel. Jedničkový doplněk binárního čísla se dostane znegováním všech bitů (jedničky se nahradí nulami a naopak). Toto číslo se následně sčítá s dalším (třetím) číslem a tak se postupuje dále, až se dojde k poslednímu prvku přidaných dat. Nakonec se ještě výsledek opět bitově invertuje. Výsledná hodnota se uloží do pole "checksum" a paket se odešle. Příjemce provede stejný postup a hodnota se porovná s kontrolním součtem, který byl vypočítán na straně odesílatele. Pokud jsou obě hodnoty stejné, tak nám Wireshark tuto informaci doplní do hranatých závorek s popisem "correct". Všechny položky v podrobnostech o paketu, které jsou uzavřeny v hranatých závorkách, tak jsou dopočítány Wiresharkem. Jako další příklad můžeme uvést

1	Wi-Fi													_		×
File	Edit	View	Go Capture	Analyze	Statistics	Telephon	y Wireles	s Tools	Help							
		0	🛅 🔀 🚨	۹ 👄 ۱	⇒ ≊ 🖗	₺ 🔳 🗉	• • •	0. 🏢								
ici	mp														$\times \rightarrow$	*]+
No	Tie	~ ^	Source		Dectination	Prot	ocol Lenat	h Tofo								
	5 0	612668	192 168	a 100	77 75 75	172 TC	MD 7/	Echo	(ning)	request	id=0x0001	seg=1/256	tt1=64	(ceply	in 7)	
-	7 0.	629452	77.75.75	.172	192.168.0	.100 TO	MP 74	Echo	(ping)	renly	id=0x0001	seg=1/256	tt1=54	(reque	st in 5	0
	15 1.	617398	192.168.	0.100	77.75.75.	172 10	MP 74	Echo	(ping)	request	id=0x0001	seg=2/512	tt1=64	(reply	in 16)	<u> </u>
	16 1.	634376	77.75.75	.172	192.168.0	.100 10	MP 74	Echo	(ping)	reply	id=0x0001	seq=2/512	tt1=54	(reque	st in 1	5)
	20 2.	620419	192.168.	0.100	77.75.75.	172 IC	MP 74	Echo	(ping)	request	id=0x0001	seq=3/768	tt1=64	(reply	in 21)	
	21 2.	637566	77.75.75	.172	192.168.0	.100 IC	MP 74	Echo	(ping)	reply	id=0x0001	seq=3/768	, ttl=54	(reque	st in 2	0)
	22 3.	624265	192.168.	0.100	77.75.75.	172 IC	MP 74	Echo	(ping)	request	id=0x0001	seq=4/1024	4, ttl=64	l (repl	y in 23)
L	23 3.	641582	77.75.75	.172	192.168.0	.100 IC	MP 74	Echo	(ping)	reply	id=0x0001	seq=4/1024	↓, ttl=54	l (requ	est in	22)
<																>
> E > I ~ I	<pre>> Frame 7: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{C93BCA0A-6D8F-458E-9605-E612C9CB5B ^ > Ethernet II, Src: Tp-LinkT_31:62:fc (68:ff:7b:31:62:fc), Dst: IntelCor_ab:8c:f2 (74:e5:0b:ab:8c:f2) > Internet Protocol Version 4, Src: 77.75.75.172, Dst: 192.168.0.100 V Internet Control Message Protocol Type: 0 (Echo (ping) reply) Code: 0 Checksum: 0x555a [correct] [Checksum Status: Good] Identifier (EE): 1 (0x0001) Gequence number (EE): 1 (0x0001) Sequence number (EE): 256 (0x0100) [Request frame: 51 Resonse time: 16.784 ms1</pre>															
	Data	(32 b)	ytes)				_							_		, ĭ
																-
0000 0010 0020 0030 0040	74 00 00 67 77	e5 0b a 3c 0d 0 64 00 0 68 69 0 61 62 0	ab 8c f2 68 02 00 00 36 00 55 5a 00 5a 6b 6c 6d 53 64 65 66	ff 7b 3 01 1d 7 01 00 0 6e 6f 7 67 68 0	31 62 fc 0 7c 4d 4b 4 31 61 62 6 70 71 72 7 59	8 00 45 4 b ac c0 a 3 64 65 6 3 74 75 7	10 t 18 .< 16 .d. 76 ghij wabo		b···E@ MKK··· abcdef oqrstuv							
\bigcirc	Z Id	lentifier (b	oig endian repres	entation) (i	cmp.ident), 2	byte(s)				Packets:	28 · Displayed:	8 (28.6%) • Dro	pped: 0 (0.	0%) P	rofile: Def	ault

Obr. A.14: Zobrazení záhlaví protokolu ICMP, včetně hexadecimálního a ASCII zápisu.

poslední položku zobrazovanou v ICMP záhlaví, kterou je "response time". Tuto hodnotu dopočítá sám Wireshark tak, že odečte čas odeslání zprávy echo request od času přijetí zprávy echo reply.

Další položkou v ICMP záhlaví je identifikátor. U Linuxových systémů má každý samostatný proces pingu vlastní identifikátor. U Windows je tomu jinak. Identifikátor je pevné číslo, které se nemění ani v případě, že budeme provádět ping z více příkazových řádků najednou. Číslo se liší pouze u různých verzí operačního systému.

Identifikátor se ve Wiresharku vyskytuje ve dvou zápisech. Prvním je BE (Big endian), který ponechává zápis tak, jak je uveden v hexadecimálním zápise v dolní části programu. Druhý zápis - LE (Little endian) - reprezentuje zápis bajtů tak, že nejdříve uvádí nejméně významný bajt a poté následují bajty významnější. Zápis tedy prohazuje bajty mezi sebou a z toho poté vychází jiný identifikátor pro jiný typ zápisu. Viz obr. A.14.

Podstatnější hodnotou uváděnou v ICMP záhlaví je pro nás sekvenční číslo. Sekvenční číslo je vždy stejné pro pár, který tvoří zprávy echo request a echo reply. Číslování sekvencí se resetuje až při opětovném zavádění operačního systému. První pár zpráv po restartu počítače tedy bude začínat se sekvenčním číslem 1 (0x0001) v zápise BE a každý další se zvětšuje o 1. Jak je vidět na obr. A.14, stejně jako u identifikátorů jsou dostupné opět oba typy zápisu (BE, LE).

Po sekvenčním čísle následuje řádek s odkazem, kdy po dvojkliku pomocí myši přeskočíme na druhý z paketů v páru (echo request nebo echo reply). Jako poslední prvek zobrazovaný v záhlaví je "response time", který byl vysvětlen výše. Poté již následují přidaná data, které zasíláme spolu s pingem (v našem příkladě se jedná o výchozí hodnotu 32 bajtů). Tato data už nejsou součástí ICMP záhlaví. Přidaná data ICMP paketu ve zprávě echo reply jsou totožná jako ve zprávě echo request.

Nyní otestujeme funkci sekvenčního čísla za pomocí dvou příkazových řádků. Otevřete si dvě okna příkazového řádku. V první příkazové řádce si nachystejte příkaz ping na seznam.cz s parametrem -4, ale ještě jej nespouštějte. Ve druhém konzolovém okně přichystejte příkaz ping na vutbr.cz s parametrem -4 -l 2000 bajtů. Spusťte zachytávání paketů ve Wiresharku a rovnou zadejte filtr pro zobrazení pouze ICMP paketů.

V jeden moment potvrďte příkaz ping v obou příkazových řádcích. Poté co obdržíte všechny 4 odpovědi echo reply v obou oknech, tak zastavte zachytávání paketů ve Wiresharku. Hodnota sekvenčního čísla ICMP záhlaví je sice uvedena v informacích o paketu, my si ale vytvoříme nový sloupec pouze pro tuto hodnotu. Klikněte pravým tlačítkem na záhlaví jednoho ze sloupců (Time, Source, Destination...) a ze zobrazeného menu vyberte Column preferences.... Otevře se nové okno, které ukazuje všechny dosud zobrazované sloupce, tak jako na obr. A.15. Tlačítkem plus "+" v levé dolní části otevřeného okna přidáme nový sloupec. Pro zobrazení sekvenčního čísla vložte do sloupce Fields textový řetězec icmp.seq. Do tohoto sloupce lze vkládat různé hodnoty, stejně jako u filtrování zachycených paketů. Do sloupce Title u naší nové položky vepište název sloupce Sekvenční číslo. Ostatní položky položky nově vytvořeného řádku ponechejte tak jak jsou (viz obr. A.15). Nový sloupec máme vytvořený, ještě jej myší přetáhněte o pár řádků výše, ať se nezobrazuje jako poslední (v hlavním okně Wiresharku by se zobrazoval úplně napravo). Okno zavřete kliknutím na tlačítko OK.

A jak tedy první příkazová řádka pozná, že ICMP pakety patří jí a ne druhé příkazové řádce? Seřadte sekvenční čísla vzestupně, kliknutím na nově vytvořený sloupec, respektive jeho název "Sekvenční číslo", stejně jako na obr. A.16.

Vidíme, že číslo se vždy u paketu echo request zvýší o jedna oproti předchozímu paketu. Důležité ale je, že v páru echo request a echo reply je číslo stejné. Když tedy odešleme příkaz ping, příkazová řádka si zjistí, že další volné sekvenční číslo je například 1. Odešle tedy paket echo request se sekvenčním číslem 1 v ICMP záhlaví a následně čeká pouze na ICMP pakety se stejným sekvenčím číslem u zpráv echo reply. Všechny ostatní čísla ignoruje, respektive je v našem případě přijímá druhý proces, který svůj výstup zobrazuje v druhé příkazové řádce.

	*Ethernet0 (icmp)										
File	e Edit View Go	Capture Analyze	Statistics Tel	ephony Wireless Tools	Help						
4	🔳 🔬 🛞 📙 詰	🔀 😂 🔍 🗢 🗢	🕾 🗿 👲 🛛	କୁ 📃 ପ୍ର୍ର୍ୟୁ							
	icmp										
No.	Time	Source	Destination	4 11 1 1 1 1 1	-					2	
	1 0.000200 2 0.004178 4 0.347116 6 0.348144 7 1.012223 10 1.355437 12 1.356458 13 2.028093 14 2.022890 16 2.371604 18 2.373226 19 3.043362 22 3.402822 24 3.404198	192.168.171.129 77.75.75.172 192.168.171.129 147.229.2.90 192.168.171.129 147.229.2.90 192.168.171.129 77.75.75.172 192.168.171.129 77.75.75.75.172 192.168.171.129 147.229.2.90 192.168.171.129 192.168.171.129 192.168.171.129	77, 75, 75, 1 192, 166, 1 147, 229, 2 190, 168, 1 177, 75, 75, 1 192, 166, 1 147, 229, 2 190, 168, 1 147, 229, 2 190, 168, 1 147, 229, 2 192, 166, 1 147, 229, 2 192, 168, 1	 Wireshark - Preference: Appearance Columns Font and Colors Layout Capture Expert Filter Buttons Name Resolution Protocols RSA Keys Statistics Advanced 	Displayed VV VV VV VV	Title No. Time Source Destination Setvenční Číslo Protocol Length Info	Type Number Time (format as specified) Source address Custom Protocol Protocol Packet length (bytes) Number	Fields	Field Oc	? currenc	×
								OK	Cancel	Help	

Obr. A.15: Vytvoření nového sloupce pro zobrazení sekvenčního čísla v samostatném sloupci.

	Wi-Fi										Administrator: Příkazový řádek	
File	Edit View	Go Capture Ana	lyze Statistics Te	lephony	Wireless Tools	Help						
4	60			ĖĖ								
			= 0 - 2								Pinging seznam.cz [77.75.75.176] with 32 bytes of dat	ta:
	mp -	-									Reply from 77.75.75.176: bytes=32 time=14ms TTL=54	
No.	Time	Source	Destination	Protocol	Sekvenční čislo	Length	Info			1.0.0004	Damla from 77 75 75 176, but a 20 time 14me TTL 54	
1	16 0 591580	77 75 75 176	192 168 0 100	TCMP	1 (0x0001) 1 (0x0001)	74	Echo (ping) requ ping) repl	Jest Iv	id=0x0001	Reply from //./5./5.1/6: Dytes=32 time=14ms L=54	
	22 0.980338	192.168.0.100	147.229.2.90	ICMP	2 (0×0002)	562	Echo (ping) requ	iest	id-0x0001	Reply from 77.75.75.176: bytes=32 time=13ms TTL=54	
	25 1.014232	147.229.2.90	192.168.0.100	ICMP	2 (0×0002)	586	Echo (ping) repl	ly .	id=0x0001	Bonly from 77 75 75 176, bytos-32 timo-19ms TTL-54	
	26 1.580389	192.168.0.100	77.75.75.176	ICMP	3 (0x0003)	74	Echo (ping) requ	lest	id=0x0001	Reply 110m 77.75.75.176. Dytes-52 time-18ms 11L-54	
	2/ 1.594809	//./5./5.1/6	192.168.0.100	ICMP	3 (0×0003)	74	Echo (ping) repl	Ly .	1d=0x0001		
	31 2.023229	147.229.2.90	192.168.0.100	ICMP	4 (0x0004)	586	Echo (ping) requ	lv	id=0x0001	Ping statistics for 77 75 75 176.	
	33 2.583951	192.168.0.100	77.75.75.176	ICMP	5 (0x0005)	74	Echo (ping) requ	iest	id=0x0001		
	34 2.597233	77.75.75.176	192.168.0.100	ICMP	5 (0x0005)	74	Echo (ping) repl	Ly:	id=0x0001	Packets: Sent = 4, Received = 4, Lost = 0 (0% los	5S),
	36 2.988562	192.168.0.100	147.229.2.90	ICMP	6 (0×0006)	562	Echo (ping) requ	Jest	id=0x0001	Approximate round trip times in milli-seconds:	
	39 3, 586574	192.168.0.100	77.75.75.176	TCMP	5 (0x0005) 7 (0x0007)	586	Echo (ping) repi	uest.	id=0x0001	Minimum 10ms Maulimum 10ms Ausses 14ms	
	40 3.604807	77.75.75.176	192.168.0.100	ICMP	7 (0x0007)	74	Echo (ping) requ	lv	id=0x0001	Minimum = 13ms, Maximum = 18ms, Average = 14ms	
	42 3.992331	192.168.0.100	147.229.2.90	ICMP	8 (0x0008)	562	Echo (ping) requ	est	id-0x0001		
	44 4.036887	147.229.2.90	192.168.0.100	ICMP	8 (0×0008)	586	Echo (ping) repl	Ly .	id=0x0001	vome sekvenchi cisio je napriklad 1. Odesie	
											al Administrator: Příkazový řádek –	
											Pinging withr.cz [147.229.2.90] with 2000 bytes of da	ata:
¢											Pangang fucor for [14, feesterse] with 2000 bytes of at	ucu.
> F	rame 14: 74 b	ytes on wire (5	92 bits), 74 byt	es capt	ured (592 bit:	s) on i	nterfac	<pre>> \Device\</pre>	NPF_{	C93BCAØA-6	Reply from 147.229.2.90: bytes=2000 time=34ms L=52	
> E	thernet II, S	inc: IntelCor_ab	:8c:f2 (74:e5:0b	:ab:8c:	f2), Dst: Tp-I	LinkT_3	1:62:fc	(68:ff:7b	:31:6	2:fc)	Reply from 147.229.2.90: bytes=2000 time=38ms TTL=52	
> 1	nternet Proto	col Version 4,	Src: 192.168.0.1	00, Dst	: 77.75.75.17	5					Bonly from 147 220 2 00, bytoc-2000 time-20mc TTL-52	
V I	nternet Contr	ol Message Prot	ocol								Reply from 147.229.2.90: Dytes=2000 time=38ms iit=52	
	Code: 0	no (pring) reque									Reply from 147.229.2.90: bytes=2000 time=44ms TTL=52	
	Checksum: 0	x4d5a [correct]										
	[Checksum S	tatus: Good]										
	Identifier	(BE): 1 (0x0001)								Ping statistics for 147.229.2.90:	
	Identifier	(LE): 256 (0x010 mbac (PE): 1 (0:	30) (0001)								Packets: Sent = / Received = / Lost = 0 (0% los	(2 2
	Sequence number (BE): 1 (0x0001) Sequence number (LE): 256 (0x0100)										Packets. Sent = 4, Received = 4, Lost = 0 (0.0 10.	<i>ss</i> ,
	[Response frame: 16]										Approximate round trip times in milli-seconds:	
3	Data (32 by	tes)									Minimum = 34ms, Maximum = 44ms, Average = 38ms	
۲											C:\WINDOWS\system32>	
0	Sequence n.	umber (big endian rer	ntation) (icmp.seq), 2 b	yte(s) Pi	ackets: 52 · Display	ed: 16 (30).8%) · Dro	pped: 0 (0.09	5) Pr	ofile: Default		

Obr. A.16: Zachycení ICMP paketů z obou příkazových řádek a seřazení paketů podle nově vytvořeného sloupce.
Nyní si zobrazte ICMP pakety včetně fragmentů, které jsou součástí komunikace se serverem vutbr.cz. Pomůže vám k tomu tento filtr:

ip.addr == <IP adresa na vutbr.cz z příkazové řádky> or icmp

Seřaďte si zobrazený výpis paketů opět vzestupně podle hodnoty **Time** a zodpovězte následující otázky.

Úkoly:

- (15) Jak Wireshark respektive příkazová řádka poznají, které IPv4 fragmenty bez ICMP záhlaví a ICMP pakety patří k sobě? (Nápověda: hledejte v IPv4 záhlaví.)
- (16) Jaká data vkládá ICMP protokol do přidaných dat, které se odesílají spolu s pingem. Hledejte v dolní části programu Wireshark, kde se data paketu zobrazují v ASCII znacích.

A.2.7 DNS resolver

V poslední části tohoto cvičení využijeme program DNS Resolver. Tento program využívá podobné funkce jako námi dříve použitý příkaz nslookup. Umí přeložit doménové jméno na IPv4 i IPv6 adresu. Navíc nabízí grafickou nadstavbu a také možnost přepínat protokoly TCP a UDP podle toho, který zrovna chceme použít pro přenos DNS protokolu.

Otevřete program (ikona umístěna na ploše). Do prvního pole zadejte libovolné doménové jméno a do druhého zvolte adresu DNS serveru, například 8.8.8.8 (volně dostupný Google DNS server). Dále v první nabídce vyberte třídu IN (internet), pro použití běžných internetových adres. Ve druhé nabídce zvolte typ A pro využití IP verze 4. Transportní protokol ponechejte UDP a vypněte DNSSEC, podobně jako je to na obr. A.17. Zapněte zachytávání Wiresharku a odešlete požadavek tlačítkem Send.

Následně proces zopakujte pro doménové jméno seznam.cz. Zaznamenejte proces získání IP adresy pomocí dotazů typu A i AAAA. Vyhledejte v DNS odpovědi v podrobnostech protokolu DNS životnost TTL přijaté IP adresy pro oba typy záznamu. (Standard query response - A seznam.cz > Domain Name System (response) > Answers > seznam.cz: type A... > Time to live). Ve Wiresharku a na obr. A.18 vidíme, že životnost DNS záznamu je v odpovědi uložena v sekundách a Wireshark nám hodnotu přepočítá na minuty. Také úplně dole máme možnost zjistit dobu odezvy DNS serveru. Podobně jako u ICMP odpovědi i zde Wireshark tuto hodnotu vypočítá odečtením času kdy byl odeslán dotaz od času přijaté odpovědi.



Obr. A.17: Prostředí programu DNS resolver a výpis pro vyhledávané doménové jméno vutbr.cz.

🚄 *Wi-Fi		- 🗆 X
File Edit View Go Capture Analyze Statistics Telephony	Wireless Tools Help	
🚄 🔳 🔬 🕒 🔜 🛣 🗳 🔍 🗢 🗢 🕾 🕢 📃 📃 🗉	B C C II	
dns		▼→+
No. Time Source Destination Protocol Le	ngth Info	
2 0.564159 192.168.0.100 8.8.4.4 DNS	68 Standard query	0x289b A vutbr.cz
- 3 0.600511 8.8.4.4 192.168.0.100 DNS	84 Standard query	response 0x289b A vutbr.cz A 147.229.2.90
19 2.235523 192.168.0.100 8.8.8.8 DNS	85 Standard query	0x0000 PTR 172.75.75.77.in-addr.arpa
20 2.266858 8.8.8.8 192.168.0.100 DNS	112 Standard query	response 0x0000 Pik 1/2./5./5.//.in-addr.arpa Pik www.sezham.cz
<		>
Transaction ID: 0x289b		^
> Flags: 0x8180 Standard query response, No error		
Questions: 1		
Answer RRS: 1 Authority PRs: 0		
Additional RRs: 0		
✓ Queries		
✓ vutbr.cz: type A, class IN		
Name: vutbr.cz		
[Name Length: 8]		
[Label Count: 2] Type: A (Host Address) (1)		
Class: IN (0x0001)		
✓ Answers		
✓ vutbr.cz: type A, class IN, addr 147.229.2.90		
Name: vutbr.cz		
Type: A (Host Address) (1)		
Liass: IN (0X0001) Time to live: 255 (4 minutes 15 seconds)		
Data length: 4		
Address: 147.229.2.90		
[Request In: 2]		
[Time: 0.036352000 seconds]		v
0000 74 e5 0b ab 8c f2 68 ff 7b 31 62 fc 08 00 45 40	th. {1bE@	
0010 00 46 22 4c 00 00 76 11 55 03 08 08 04 04 c0 a8	-F"Lv- U	
0030 00 01 00 00 00 00 05 76 75 74 62 72 02 63 7a 00	·····v utbr·cz·	
0040 00 01 00 01 c0 0c 00 01 00 01 00 00 ff 00 04	· · · · · · · · · · · · · · · · · · ·	
0050 93 e5 02 5a	· · · Z	
Response TTL (dns.resp.ttl), 4 byte(s)		Packets: 52 · Displayed: 4 (7.7%) · Dropped: 0 (0.0%) Profile: Default

Obr. A.18: Zobrazení podrobností DNS paketu a v horní části se nachází dotaz typu PTR pro zpětný překlad z IP adresy na doménové jméno.

Nyní si vyzkoušíme zpětné převedení z IP adresy na doménové jméno pomocí PTR záznamu. U tohoto záznamu dochází k převrácení IP adresy (viz obr. A.18),

protože IP adresa má v levé části adresu sítě a vpravo část adresy, která určuje konkrétní počítače. U doménových jmen je to naopak a proto se adresa převrací. Proto při PTR dotazu na adresu 77.75.75.172 (seznam.cz) uvidíme ve Wiresharku zápis adresy opačně a s připojenou doménou (172.75.75.77.in-addr.arpa), avšak se správným výsledkem v DNS odpovědi.

Zapněte zachytávání ve Wiresharku, do DNS resolveru zadejte IP adresu 217.31.205.50 do pole pro doménové jméno, zvolte typ záznamu PTR a odešlete požadavek. Pomocí Wiresharku vyhledejte odpověď na následující otázku.

Úkoly:

- (17) Jaké doménové jméno se skrývá za hledanou IP adresou 217.31.205.50?
- (18) Jaké hlavní činnosti má na starosti sdružení, které se nachází na dané adrese?
- (19) Pomocí kolika IPv4 a IPv6 adres lze ve webovém prohlížeči Chrome zobrazit stránku sport.cz?

B Kompletní návod pro druhý vytvořený simulační scénář

VYPRACOVANÁ DRUHÁ ÚLOHA

Pokročilejší možnosti programu Wireshark: protokoly IP přenosu hlasu (VoIP, RTP), tvorba grafů, srovnání protokolů TCP a UDP, protokol QUIC, úprava parametrů sítě a analýza webového provozu v prostředí prohlížeče Chrome.

B.1 Teoretický úvod

V tomto cvičení uskutečníme VoIP hovor (Voice over Internet Protocol), jehož pakety zachytíme v programu Wireshark. Hovor zanalyzujeme a pokusíme se jej zpětně rekonstruovat. Následně vyzkoušíme dvě formy šifrování hovoru, které jsou součástí programu Linphone, pomocí kterého bude hovor realizován. Dále se studenti naučí vytvářet grafy v programu Wireshark ze zachycených paketů. Druhá čast se věnuje srovnání protokolů TCP a QUIC. K tomu bude využita testovací webová stránka a také prostředí webového prohlížeče Chrome. Testování bude probíhat jednak v ideálních podmínkách, následně ale také v upravených podmínkách. Ty budou simulovány v rámci virtualizovaného operačního systému skrze program VMware Workstation Player a jeho nastavení.

B.1.1 Protokoly realizující hovor skrze IP

Pro uskutečnění VoIP hovoru jsou využity především protokoly **SIP** (Session Initiation Protocol – protokol pro inicializaci relací) a **RTP** (Real-time Transport Protocol). První z protokolů zajišťuje spojení účastníků hovoru. V případě, že je hovor spojen, tak přebírá komunikaci protokol RTP, který má na starosti přenos multimediálních dat, jako je zvuk nebo video.

SIP je kontrolní protokol, pracující na aplikační vrstvě. Je určen k vytváření a ukončování relací, jako jsou VoIP hovory nebo multimediální konference, mezi jedním nebo několika uživateli. Účastníci si pomocí SIP protokolu dohodnou například to, jaký kodek budou při hovoru používat. Pro sestavení hovoru je nutné propojit obě komunikující strany. Ty se kontaktují požadavkem INVITE a po přijetí hovoru druhou stranou je vyslána zpráva 200 OK, která znamená úspěšné spojení. V případě chyby na jedné ze stran, se vyšlou odlišné číselné kódy podle tabulky B.1. Například, když volaný hovor odmítne, odešle se některá ze zpráv označovaných jako "busy here". Volaný může být zaneprázdněn jiným hovorem (486) nebo je vyžadována platba pro spojení hovoru (402).

RTP je protokol, který umožňuje přenášet data v reálném čase. Sám o sobě nezaručuje doručení přenášených paketů, stejně jako UDP protokol, na kterém je RTP postaven. Každý RTP paket ale obsahuje pořadové číslo a také časové razítko, které pomáhají při rekonstrukci hovoru ve správném pořadí na přijímající straně hovoru. Záhlaví protokolu obsahuje 12 bajtů dat a přenášená data jsou v jednom paketu zastoupena 160 bajty. Těmto 160 bajtům se říká "chunky", což jsou zakódované kusy hlasových dat, které jsou po určitém časovém intervalu (několik milisekund) rozděleny na části - chunky, které se distribuují pomocí RTP protokolu.

Tab. B.1: Tabulka zobrazující výčet možných číselných kódů protokolu SIP. Chybu znamenají kódy začínající číslicí 4,5 nebo 6.

kód	popis kódu
1XX	Požadavek se úspěšně zpracovává, ale ještě není ukončen.
2XX	Úspěšné vyřízení žádosti. (Nejčastěji 200)
3XX	Pro dokončení požadavku je nutné hovor přesměrovat.
4XX	Chyba na straně klienta.
5XX	Chyba na straně serveru.
6XX	Fatální chyba, kterou nelze zpracovat.

B.1.2 Protokoly zajišťující zabezpečený přenos dat

V rámci cvičení se budou porovnávat dva protokoly, které umožňují zabezpečený přenos webové stránky. Prvním je **TLS** (Transport Layer Security) a druhým je **QUIC** (Quick UDP Internet Connections).

TLS je jeden z protokolů, které zajišťují zabezpečenou komunikaci mezi klientem a serverem. Konkrétně protokol zajišťuje autentizaci serveru a to pomocí asymetrických šifer, jako je například RSA (iniciály autorů - Rivest, Shamir, Adleman). Dále TLS zajišťuje důvěrnost a integritu, takže posílaná data jsou přístupná pouze koncovým prvkům, které jsou autentizovány a integrita znamená odhalení případné změny dat během přenosu.

Na začátku komunikace se musí strany domluvit na tom, který typ šifrování využijí. Následně si bezpečnou cestou vymění klíče a také se provede autentizace pomocí certifikátů. Poté již probíhá zabezpečená komunikace, která se šifruje pomocí dříve vyměněného symetrického klíče. TLS je využíváno pro bezpečnou komunikaci v rámci webových stránek nebo právě u zabezpečení protokolu SIP u VoIP hovorů.

QUIC je relativně nový protokol a jeho úroveň bezpečnosti je srovnatelná s protokolem TLS. Hlavním cílem je zrychlit přístup k webovým službám. Na rozdíl od protokolu TLS, který využívá TCP je QUIC postaven na protokolu UDP. Výhoda QUICu se uplatní hlavně při opětovném připojení klienta na server, který již někdy využíval. Při komunikaci se použijí klíče, které si strany vyměnily dříve a není tak nutné tento proces opakovat a tak se požadovaná data přenášejí hned v prvním paketu. V případě, že jde o prvotní komunikaci, tak se prvními pakety navazuje spojení a také se přenášejí klíče k ustanovení zabezpečeného kanálu. Požadovaná data se pak přenášejí od třetího paketu dále. QUIC identifikuje jednotlivá spojení takzvaným "Connection ID", které je součástí záhlaví protokolu. Protokol je již několik let využíván v prohlížeči Chrome a to především při komunikaci se servery Google

B.2 Realizace druhého scénáře

Tento scénář se bude v první části zabývat rozborem VoIP hovoru. Studenti se ve virtuálním prostředí pokusí pomocí programu Linphone kontaktovat své spolužáky, případně lze zvolit variantu kdy student volá sám sobě a nepotřebuje vytvářet 2 účty pro správnou funkčnost komunikačního programu Linphone, který bude pro uskutečnění hovoru využit. Po zachycení hovoru si studenti ve Wiresharku zobrazí dostupné podrobnosti o hovoru a zkusí si zpětně rekonstruovat zachycený hovor v případě nešifrovaného i šifrovaného spojení s druhou stranou.

Další část se bude věnovat analýze zachycených dat pomocí grafů, které lze ve Wiresharku vytvořit. Úvodní seznámení s možnostmi sestavení grafů bude vysvětleno právě na zachyceném VoIP hovoru, kde studenti uvidí maximální hranici přenosové rychlosti použitého kodeku. Kromě vytvoření křivek v grafu pro protokoly TCP a UDP bude také využita možnost vytvořit křivky s pomocí filtrů, které byly vysvětleny v první scénáři.

Následně bude vysvětlen postup uložení a spojení dvou souborů se zachycenými pakety do jednoho souboru. To se využije při porovnání dvou protokolů použitých v totožné situaci. První situací bude použití DNS resolveru, kde se u grafu ze spojených souborů ukáže rozdíl počtu potřebných paketů mezi protokoly TCP a UDP pro provedení DNS požadavku. Druhá situace obsahuje srovnání protokolů TCP+TLS a QUIC. K porovnání poslouží vytvořená testovací webová stránka a prohlížeč Chrome, který umožňuje přepínání mezi těmito protokoly. Srovnání bude provedeno za ideálních podmínek a následně se podmínky upraví změnou síťových parametrů. Změna parametrů, jako je šířka pásma, ztrátovost paketů nebo zvýšená latence, bude simulována programem VMware Workstation Player.

Poslední část studentům ukáže možnost zachycení a zobrazení webového provozu přímo v prostředí prohlížeče Chrome pomocí utility Netlog viewer. Utilita nám zobrazí podrobnosti o protokolu QUIC a také obsah paketů, který se nedá zobrazit ve Wiresharku díky použitému šifrování.

B.2.1 Popis prostředí programu Linphone

Jako první se seznámíme s prostředím programu. Ve výchozím scénáři budeme využívat odlehčenou verzi programu, která je zdarma dostupná z Microsoft Store. Verze je jednodušší na pochopení, avšak plně dostačuje k využití pro naše účely. Ikona programu Linphone je umístěna na ploše a hlavním panelu, jedná se o verzi s písmenem "L" na ikoně. Spustte program a projděte si jeho možnosti nastavení.

Na úvodní obrazovce, která je také zobrazena na obr. B.1, se nachází adresní řádek, kam budeme psát adresu volaného. Ještě nad tímto řádkem je zobrazen stav aplikace. Ideální případ je zobrazen na obrázku, kdy je uživatel přihlášen "Registered" a indikátor je zbarven zeleně. Registraci a přihlášení se bude více věnovat následující kapitola B.2.2.

Pod adresním řádkem je klávesnice pro zápis adresy. V našem případě budeme využívat běžnou hardwarovou klávesnici, protože bude potřeba psát adresu i s textem, nikoliv pouze čísla. Pod klávesnicí je tlačítko "Call", kterým zahájíme hovor, případně jím lze do adresního řádku vyvolat poslední volaný kontakt pro urychlení práce.

Posledním prvkem jsou ikony na dolní straně okna programu. Zleva jde o historii volání, uložené kontakty, chat a nastavení.



Obr. B.1: Zobrazení úvodní obrazovky programu Linphone.

B.2.2 Linphone registrace a nastavení

Nastavení obsahuje položku "SIP account", kde je možné po registraci zadat přihlašovací údaje, pro úspěšnou komunikaci s dalšími uživateli. "Audio settings" umožňuje výběr z několika kodeků, které program podporuje. My všechny možnosti odškrtneme a povolíme pouze PCMU (Pulse-code modulation μ -law – pulzně kódová modulace μ -law) a PCMA (Pulse-code modulation A-law – pulzně kódová modulace A-law), stejně jako na obr. B.2. Tyto dva kodeky volíme z toho důvodu, že jsou to jediné dva, které dokáže Wireshark následně přehrát již ve výchozím stavu. V případě zvolení odlišných kodeků by se hovor detekoval, ale nešlo by jej ze zachycených paketů přehrát. V nastavení videa "Video settings" ponecháme vše ve výchozím stavu. V poslední položce "Advanced settings" pouze zkontrolujeme, že Media encryption je nastaveno na None, zbytek ponecháme tak jak je.

← Linphone	-	×
AUDIO SETTINGS		
Codecs		
OPUS		
Off Off		
Speex 16		
Off Off		
Speex 8		
Off Off		
PCMU		
On On		
PCMA		
On On		
G729		
Off Off		
GSM		
Off Off		

Obr. B.2: Nastavení zvukových kodeků v aplikaci Linphone (light).

Posledním krokem k uskutečnění hovoru je registrace uživatele. Tu je nutné provést buď přes oficiální stránky Linphone¹ nebo přes druhou verzi aplikace, která je na ploše označena "Linphone 2". V obou případech je potřeba si určit SIP jméno uživatele (nejlépe v podobě VUT loginu - xjmeno00). Následně je třeba zadat email, ke kterému máte přístup (na něj dojde potvrzovací email). Pak už jen heslo a registrace je hotová po potvrzení emailu.

Nyní je čas na upravení informací v nastavení "SIP account". Vyplníme informace tak jak je uvedeno na obr. B.3. Uživatelské jméno a heslo zadáme podle toho, co jsme si zvolili při registraci a dále změníme transportní protokol z TLS na TCP. Zbytek ponecháme tak jak je a úpravy uložíme tlačítkem dole vpravo. Změna transportního protokolu mění protokol, který obsluhuje úvodní spojení uživatelů, nikoliv samotný přenos hlasových dat. Ten je ve většině případů obsloužen protokolem UDP.

¹https://www.linphone.org/freesip/home

← Linphone		_		х
SIP account				
xjmeno00				\times
Auth userid (not mandatory)				
Password				_
•••••				
Domain				
sip.linphone.org				
Transport				
TCP 🗸				
Proxy (not mandatory)				
sip:sip.linphone.org;transport=tcp				
Outbound proxy				
On On				
ICE				
On On				
AVPF				
On				
Expire				
28800				
Ĩ	Q		9	•••

Obr. B.3: Možnosti přihlášení a dalších parametrů v nastavení SIP account programu Linphone.

Po uložení by se měla při správně vyplněných údajích zbarvit indikační tečka na úvodní obrazovce do zelena. V tom případě můžete otestovat komunikaci s některým z dalších studentů. Pro vyvolání hovoru je nutno zadat adresu volaného. Adresa je psána ve tvaru:

sip:xjmeno00@sip.linphone.org

Prefix **sip** definuje, že chceme použít protokol SIP. Zbytek adresy je podobný jako zápis emailové adresy. Před znakem zavináče je uživatelské jméno volaného a za ním je doména, kterou jsme zvolili v nastavení. Tyto domény mohou být placené, Linphone svou doménu poskytuje po registraci zdarma.

Úkol:

 $(1)\,$ Kontaktujte některého ze svých spolužáků a ověřte funkčnost spojení.

÷	Linphone				_		×
			8				
		XS	imaj01@sip.linphone.or	g			
			00:01:29				
			\triangleright				
	⊐ı Video		ھِ) Toggle camera		Sp	⊲») eaker	
	ll Pause		U Micro		S	() tats	
			Di	iji alpad			

Obr. B.4: Probíhající hovor v programu Lipnhone (light).

B.2.3 Rekonstrukce hovoru a jeho základní zabezpečení

V případě, že hovor probíhá v pořádku (obr. B.4), tak můžeme zobrazit jeho statistiky (tlačítko Stats na obrazovce hovoru). Dostaneme podobné informace jako na obr. B.5. V prvním řádku vidíme informace o zvoleném kodeku. V našem případě jde o kodek PCMU (Pulse-code modulation μ -law – pulzně kódová modulace μ -law), který využívá standard G.711. Každý kodek má určitou vzorkovací frekvenci, která určuje počet odebíraných vzorků při přeměně analogového signálu (hlasu) na signál digitální za jednu sekundu. Jednotkou této hodnoty je hertz [Hz] a v našem případě se jedná o vzorkovací frekvenci 8000 Hz. Maximální bitrate (datový tok) tohoto kodeku pro přenos hlasu je ve výchozím stavu 80 kb/s. Tuto hodnotu nám potvrzují následující dva řádky, kdy ani upload ani download nepřesahují 80 kb/s. [20]

Dále zkontrolujte čtvrtý řádek, který nám říká, zda je nebo není použito šifrování přenášených dat. Nyní si zkusíme zachytit tato přenášená data a následně rekonstruovat hovor, tudíž potřebujeme aby se data nešifrovala.

Pokud probíhá hovor, tak jej ukončete. Zapněte Wireshark a spusťte zachytávání paketů. Uskutečněte krátký hlasový hovor, stačí několik sekund. Hovor ukončete a poté zastavte zachytávání síťového provozu ve Wiresharku. Ve Wiresharku by měl být vidět výpis podobně jako na obr. B.6. Pokud na pozadí operačního systému probíhá nějaká další rušivá komunikace, tak zadejte filtr: (tcp or udp) and (ip.src or ip.dst == 54.37.202.229). Kde zadaná IP adresa odpovídá adrese serveru, přes který komunikujeme s druhou osobou. IP adresu naleznete

například u RTP paketů a každý student může mít tuto adresu odlišnou. Zachycené pakety uložte do souboru přes File > Save As... pro pozdější analýzu.

← Linphone		_		×
	8			
StatPayloadType	PCMU/8000			
StatDownloadBandwidth	75,43 kb/s			
StatUploadBandwidth	78,68 kb/s			
StatMediaEncryption	None			
StatICE	NotActivated	ł		
ll Pause	ل Micro		① Stats	
ر Hang	[iji Dialpad		

Obr. B.5: Statistiky přenosu v programu Linphone (light) u právě probíhajícího hovoru, včetně informace o použitém kodeku a informace zda je hovor šifrován nebo ne.

	*Ethernet0					×
File	e Edit View	Go Capture Analy	ze Statistics Telep	hony Wir	eless Tools Help	
				I≡I⊕	A A #	
_			~~~~~		vv ±	
	Apply a display filte	r <ctrl-></ctrl->			Expression	+
No.	Time	Source	Destination	Protocol	Length Info	^
Г	1 0.000000	192.168.98.134	54.37.202.229	SIP/SDP	954 Request: INVITE sip:xsimaj01@sip.linphone.org	
	2 0.011274	54.37.202.229	192.168.98.134	тср	60 5228 → 49816 [ACK] Seq=1 Ack=901 Win=64240 Len=0	
	3 0.047904	54.37.202.229	192.168.98.134	SIP	566 Status: 407 Proxy Authentication Required	
	4 0.099234	192.168.98.134	54.37.202.229	TCP	54 49816 → 5228 [ACK] Seq=901 Ack=513 Win=63728 Len=0	
	5 0.231865	192.168.98.134	54.37.202.229	SIP	472 Request: ACK sip:xsimaj01@sip.linphone.org	
	6 0.232170	192.168.98.134	54.37.202.229	SIP/SDP	1239 Request: INVITE sip:xsimaj01@sip.linphone.org	
	7 0.233523	54.37.202.229	192.168.98.134	TCP	60 5228 → 49816 [ACK] Seq=513 Ack=1319 Win=64240 Len=0	
	8 0.233524	54.37.202.229	192.168.98.134	TCP	60 5228 → 49816 [ACK] Seq=513 Ack=2504 Win=64240 Len=0	
	9 0.351828	54.37.202.229	192.168.98.134	SIP	441 Status: 100 Trying	
	10 0.361838	54.37.202.229	192.168.98.134	SIP/SDP	1161 Request: INVITE sip:xsimaj01@46.174.33.147:11120;transport	
	11 0.365752	54.37.202.229	192.168.98.134	SIP/SDP	1162 Request: INVITE sip:xsimaj01@46.174.33.147:11258;transport	
	12 0.365811	192.168.98.134	54.37.202.229	TCP	54 49816 → 5228 [ACK] Seq=2504 Ack=2008 Win=64240 Len=0	
	13 0.375998	192.168.98.134	54.37.202.229	SIP	405 Status: 100 Trying	
	14 0.380776	192.168.98.134	54.37.202.229	SIP	405 Status: 100 Trying	
	15 0.384238	192.168.98.134	54.37.202.229	SIP	518 Status: 486 Busy here	
	16 0.385187	54.37.202.229	192.168.98.134	TCP	60 5228 → 49725 [ACK] Seq=1108 Ack=352 Win=64240 Len=0	
	17 0.385188	54.37.202.229	192.168.98.134	TCP	60 5228 → 49816 [ACK] Seq=2008 Ack=2855 Win=64240 Len=0	
	18 0.385188	54.37.202.229	192.168.98.134	TCP	60 5228 → 49816 [ACK] Seq=2008 Ack=3319 Win=64240 Len=0	
	19 0.499091	54.37.202.229	192.168.98.134	SIP	375 Request: ACK sip:xsimaj01@46.174.33.147:11258;transport=tc	
	20 0.574945	192.168.98.134	54.37.202.229	TCP	54 49816 → 5228 [ACK] Seq=3319 Ack=2329 Win=63919 Len=0	
	21 0.669135	192.168.98.134	54.37.202.229	SIP	562 Status: 180 Ringing	
	22 0.670480	54.37.202.229	192.168.98.134	TCP	60 5228 → 49725 [ACK] Seq=1108 Ack=860 Win=64240 Len=0	
	23 0.724098	54.37.202.229	192.168.98.134	SIP	477 Status: 180 Ringing	¥
<					>	
	🛛 wireshark_E	Ethernet0_202002221849	016_a07292.pcapng		Packets: 2000 · Displayed: 2000 (100.0%) · Dropped: 0 (0.0%) Profile: Default	t 🔒

Obr. B.6: Zobrazení výpisu programu Wireshark, kde můžeme vidět pozadí komunikace dvou uživatelů při hlasovém hovoru.

Vidíme, že pro sestavení hovoru je využit SIP protokol, který využívá transportní protokol TCP tak, jak jsme si to vyžádali v nastavení. Dochází tak k výměně informací pro sestavení hovoru. Volající vysílá požadavek INVITE na SIP proxy server, který ví že volaný má danou IP adresu a na tuto adresu přepošle INVITE zprávu. Tím se spustí zvonění na straně volaného a volající přijímá zprávu 180 Ringing (zvonící). Zpráva 180 indikuje úspěšný průběh komunikace s tím, že se ještě čeká na nějaký krok pro sestavení. Musí se vyčkat na přijetí hovoru druhou stranou. Poté je vyslána zpráva 200 OK, která znamená úspěšné spojení. Volající ještě potvrdí spojení (ACK) a může začít přenos hlasové komunikace protokolem RTP.

Pro přehlednější ukázku komunikace můžeme využít zobrazení "toku hovoru", které je dostupné přes **Telephony > VoIP calls**. Zobrazí se nám automaticky detekovaný hovor, jako na obr. B.7. Vybereme detekovaný hovor a následně **Flow sequence**. Zobrazí se nám okno podobné tomu na obr. B.8. Vidíme IP adresy dvou komunikujících stran a také časový údaj uváděný s velkou přesností. Z něj můžeme odečíst například dobu, která uplynula, než volaný hovor přijal. To jsou v našem případě zhruba dvě sekundy, které odpovídají reakčnímu času osoby přijímající hovor. Dále si strany vyměnily potvrzující zprávy 200 OK a hovor byl spojen.



Obr. B.7: Hovor, který automaticky detekoval program Wireshark.

Úkoly:

- (2) Projděte si výpis paketů zachyceného hovoru jak pro sestavení spojení (SIP), tak pro samotný přenos dat (RTP).
- (3) Jaký port je využíván při RTP komunikaci na našem zařízení?
- (4) Zobrazte si bajty jednotlivých RTP paketů v dolní části programu. Je možné z takovýchto dat zpětně rekonstruovat hovor?

Pro zpětné sestavení hovoru ve zvukové podobě ze zachycených paketů je potřeba opět vyhledat hovor přes **Telephony > VoIP calls** a následně **Play Streams**. Otevře se nám okno s jednoduchým přehrávačem jako na obr. B.9. V náhledovém okně vidíme 2 zvukové stopy. Jedna zachycuje námi vysílaný hlas a druhá stopa nese hlas volaného. Stopy můžeme přehrát tlačítky v levém dolním rohu. Dále můžeme vidět podrobnosti o každé zvukové stopě, jako dobu trvání hovoru, použitou vzorkovací frekvenci nebo kodek.

Time	192.168	.98.134	54.37.202.229	Comment
0.000000	49816	INVITE SDP (g711U g711A telephone-event)	5228	SIP INVITE From: <sip::simaj01@sip.linphone.or< td=""></sip::simaj01@sip.linphone.or<>
0.047904	49816	 407 Proxy Authentication Required 	5228	SIP Status 407 Proxy Authentication Required
0.231865	49816	ACK	5 228	SIP ACK From: <sip:xsimaj01@sip.linphone.org></sip:xsimaj01@sip.linphone.org>
0.232170	49816	INVITE SDP (g711U g711A telephone-event)	► 5228	SIP INVITE From: <sip:xsimaj01@sip.linphone.or< td=""></sip:xsimaj01@sip.linphone.or<>
0.351828	49816	a 100 Trying	5228	SIP Status 100 Trying
0.361838	49725	INVITE SDP (g711U g711A telephone-event)	5228	SIP INVITE From: <sip:xsimaj01@sip.linphone.or< td=""></sip:xsimaj01@sip.linphone.or<>
0.365752	49816	INVITE SDP (g711U g711A telephone-event)	5228	SIP INVITE From: <sip:xsimaj01@sip.linphone.or< td=""></sip:xsimaj01@sip.linphone.or<>
0.375998	49725	100 Trying	 5228 	SIP Status 100 Trying
0.380776	49816	100 Trying	5228	SIP Status 100 Trying
0.384238	49816	486 Busy here	► 5228	SIP Status 486 Busy here
0.499091	49816	ACK	5228	SIP ACK From: <sip:xsimaj01@sip.linphone.org></sip:xsimaj01@sip.linphone.org>
0.669135	49725	180 Ringing	5228	SIP Status 180 Ringing
0.724098	49816	180 Ringing	5228	SIP Status 180 Ringing
2.214995	49725	200 Ok SDP (g711U g711A telephone-event)	5228	SIP Status 200 Ok
2.263489	49816	200 Ok SDP (g711U g711A telephone-event)	5228	SIP Status 200 Ok
2.521048	49816	ACK	5228	SIP Request INVITE ACK 200 CSeq:21
2.526155	7078	RTP (g711U)	7766	RTP, 479 packets, Duration: 9,627s SSRC: 0x3BF
				`
< 1 (()	Ann (51			>
2 node(s), 25	item(s)			Court 4a

Obr. B.8: Přehlednější zobrazení komunikace uživatele se SIP serverem.



Obr. B.9: RTP přehrávač integrovaný ve Wiresharku, který umožňuje přehrát zachycený hovor.

V případě, že chceme přehrát pouze jednotlivou stopu a ne "rozhovor", který přehrává obě stopy současně, tak zvolíme Telephony > RTP > RTP Streams. Zde máme stopy oddělené a po výběru jedné z nich ji můžeme kliknutím na Analyze > Play Streams přehrát, stejně jako v předchozím případě.

Zároveň můžeme zkoumat podrobnosti daného hovoru v okně RTP Stream Analysis, jako na obr. B.10. Můžeme vidět pořadové číslo (sequence), které označuje pořadí dané části hovoru. Těmto částem hovoru se říká "chunky", což jsou zakódované kusy hlasových dat, které jsou po určitém časovém intervalu (několik milisekund) rozsekány na části - chunky, které se distribuují pomocí RTP protokolu.

Každé pořadové číslo je přenášeno v určitém paketu, který je očíslován tak, jak byl zachycen Wiresharkem (packet). Další sloupec označuje časový úsek (Delta v milisekundách). Jedná se o čas, který uplynul od posledního přijatého RTP paketu. Jitter je další sledovaná hodnota, která obsahuje srovnání dvou po sobě přijatých paketů daného RTP streamu. Porovnává se rozdíl časů, ve kterém byly pakety přijaty a také rozdíl časů z časového razítka, které bylo vytvořeno na straně odesílatele.

Nalevo od výpisu ve sloupcích se nachází obecný přehled obou streamů, které byly zaznamenány. Uvádí se zde maximální hodnoty, které jsou součástí sloupcového výpisu, dále také celkový počet RTP paketů, který byl zapotřebí pro přenos těchto streamů. Poté je zobrazen počet ztracených paketů a z nich vypočtené procento ztrátovosti (lost) a také počet chyb při přenosu streamu (seq errs). Zaznamenán je i časový údaj o začátku hovoru od startu zachytávání paketů a také celková doba trvání hovoru.

Tento případ nám ukázal zcela nezabezpečený hovor kdy každý, kdo zachytí komunikaci v sítí, může hovor rekonstruovat a přehrát. V programu Linphone máme možnost nastavit a následně si porovnat také zabezpečený hovor. Zabezpečení proběhne ve dvou fázích. První fáze nám zajistí, že data zasílaná protokolem RTP se budou posílat v šifrované podobě protokolem SRTP (Secure Real-time Transport Protocol). Toho docílíme přepnutím položky Media encryption v nastavení Advanced settings na SRTP, tak jako na obr. B.11. Další úroveň zabezpečení nám pomůže skrýt informace, které jsou uvedeny při navazování hovoru protokolem SIP. To mohou být například adresy volaného i volajícího, jejich stav a možnosti spojení jako je kodek nebo přenosová rychlost. Tuto část zabezpečíme protokolem TLS (Transport Layer Security), který také zvolíme v nastavení SIP accounts > Transport > TLS, podobně jako na obr. B.12. TLS pro přenos využívá transportní protokol TCP, takže ve výpisu uvidíme kromě TLS paketů také potvrzovací TCP pakety s příznakem ACK, jako na obr. B.13.

🚄 Wireshark · RTP Stream Analysis · graf.pcapng

 \Box \times

—

54.37.202.229:46206 ↔ 192.168.152.128:7078		Forward	Reverse	e Graph						
	20.7070	Packet	Sequence	Delta (ms)	Jitter (ms)	Skew	Bandwidth	Marker	Status	^
Forward		48	. 0	0.00	0.00	0.00	1.60		1	
SSRC	0x95f32d8f	54	1	21.15	0.07	-1.15	3.20		1	
Max Delta	87.48 ms @ 128	57	2	19.08	0.12	-0.23	4.80		1	
Max Jitter	13.12 ms	62	3	21.06	0.18	-1.28	6.40		1	
Mean Jitter	7.21 ms	65	4	28.61	0.71	-9.89	8.00		1	
Max Skew	-131.63 ms	70	5	28.68	1.21	-18.58	9.60		1	
RTP Packets	\$ 305	73	6	13.09	1.56	-11.66	11.20		1	
Expected	305	83	7	7/ 30	4.87	-66.05	12.80		·	
Lost	0 (0.00 %)	24	, 0	0.55	5 70	-46.60	14.40		¥	
Seq Errs	0	04	0	20.91	5.70	57.41	14.40		*	
Duration	5.20+200 S@+10 6.18 c	00	10	50.61	6.09	-37,41	17.60		×	
Clock Drift	-313 ms	69	10	0.00	0.90	-37.41	17.00		¥.,	
Frea Drift	7595 Hz (-5.06 %)	93	11	4.06	7.52	-21.47	19.20		¥	
••••	(99	12	31.41	7.76	-32.88	20.80		1	
Reverse		101	13	12.89	1.72	-25.77	22.40		1	
cene	ouch o course	104	14	23.49	7.46	-29.26	24.00		1	
SSKL Max Dalta	0X6D85848C	110	15	32.55	7.78	-41.80	25.60		1	
Max Jittor	32.27 ms @ 66	113	16	9.76	7.93	-31.56	27.20		1	
Mean litter	1.10 ms	128	17	87.48	11.65	-99.04	28.80		4	
Max Skew	-99.23 ms	129	18	0.00	12.18	-79.05	30.40		1	
RTP Packets	306	131	19	1.99	12.54	-61.04	32.00		1	
Expected	306	132	20	0.00	13.01	-41.04	33.60		1	
Lost	0 (0.00 %)	139	21	34.85	13.12	-55.89	35.20		1	
Seq Errs	0	141	22	14.21	12.66	-50.10	36.80		1	
Start at	5.226089 s @ 35	146	23	22.13	12.00	-52.23	38.40		1	
Duration	6.19 s	151	24	26.64	11.67	-58.87	40.00		1	
Clock Drift	-308 ms	155	25	21.66	11.04	-60.53	41.60		1	
Freq Drift	7602 Hz (-4.98 %)	158	26	9.86	10.00	-50.30	43.20		·	
Forward to r start diff -0.0	everse 038117 s @ -13	< 10		00.70	10.55	50.55	11.00		>	~
2 streams found.										
				Save		Close	▶ Play St	reams	Help	

Obr. B.10: Zobrazení podrobností RTP streamu, jako jsou ztrátovost nebo pořadové číslo.

← Linphone		_	×
ADVANCED SETTING	5		
Network Call	Debug		
Media encryption			
SRTP ~			
IPV6			
Off			
StunServer			
stun.linphone.org			

Obr. B.11: Nastavení šifrování SRTP v programu Linphone (light).

← Linphone		-	-		×
SIP account					
Username					_
xjmeno00					×
Auth userid (not mandatory)					_
Password					
•••••					_
Domain					
sip.linphone.org					
Transport					
TLS 🗸					
Proxy (not mandatory)					
sip:sip.linphone.org;transport=tls					_
Outbound proxy					
On					
ICE					
On					
AVPF					
On					
Expire					
28800					
	Ŵ	Q	Ę	3	

Obr. B.12: Nastavení TLS protokolu pro přenos hovoru v programu Linphone (light).

	TLS_a_None.pcap	ng				- 0	×
Fi	le Edit View	Go Capture Ana	lyze Statistics Tele	phony Wir	reless 1	Tools Help	
		. 🗈 🗙 🖻 🖉 «	- 	= = @	00	TT .	
					~ ~	· #	_
	Apply a display filte	r <ctrl-></ctrl->					· +
No	. Time	Source	Destination	Protocol	Length	Info	- ^
Г	1 0.000000	192.168.152.128	54.37.202.229	TLSv1.2	983	Application Data	
	2 0.005860	54.37.202.229	192.168.152.128	TCP	60	5223 → 50156 [ACK] Seg=1 Ack=930 Win=64240 Len=0	
	3 0.049175	54.37.202.229	192.168.152.128	TLSv1.2	594	Application Data	
	4 0.095470	192.168.152.128	54.37.202.229	TCP	54	50156 → 5223 [ACK] Seq=930 Ack=541 Win=63166 Len=0	
	5 0.215985	192.168.152.128	54.37.202.229	TLSv1.2	499	Application Data	
4	6 0.230512	192.168.152.128	54.37.202.229	TLSv1.2	1267	Application Data	
	7 0.234242	54.37.202.229	192,168,152,128	TCP	60	5223 → 50156 [ACK] Seq=541 Ack=1375 Win=64240 Len=	
	8 0.234243	54.37.202.229	192.168.152.128	TCP	60	5223 → 50156 [ACK] Seq=541 Ack=2588 Win=64240 Len=	
	9 0.353536	54.37.202.229	192.168.152.128	TLSv1.2	460	Application Data	
	10 0.363059	54.37.202.229	192.168.152.128	TLSv1.2	1212	Application Data, Application Data	
	11 0.402169	192.168.152.128	54.37.202.229	TCP	54	50156 → 5223 [ACK] Seq=2588 Ack=947 Win=64240 Len=	
	12 0.423036	192.168.152.128	54.37.202.229	TCP	54	49992 → 5223 [ACK] Seq=1 Ack=1159 Win=64240 Len=0	
	13 0.424398	192.168.152.128	54.37.202.229	TLSv1.2	460	Application Data	
	14 0.428430	54.37.202.229	192.168.152.128	TCP	60	5223 → 50007 [ACK] Seq=1 Ack=407 Win=64240 Len=0	
	15 0.628986	192.168.152.128	54.37.202.229	TLSv1.2	608	Application Data	
	16 0.629526	54.37.202.229	192.168.152.128	TCP	60	5223 → 50007 [ACK] Seq=1 Ack=961 Win=64240 Len=0	
	17 0.667439	54.37.202.229	192.168.152.128	TLSv1.2	496	Application Data	
	18 0.707219	192.168.152.128	54.37.202.229	TCP	54	50156 → 5223 [ACK] Seq=2588 Ack=1389 Win=63798 Ler	
	19 2.116848	192.168.152.128	54.37.202.229	TLSv1.2	1129	Application Data	
	20 2.130127	54.37.202.229	192.168.152.128	TCP	60	5223 → 50007 [ACK] Seq=1 Ack=2036 Win=64240 Len=0	
	21 2.159152	54.37.202.229	192.168.152.128	TLSv1.2	1054	Application Data, Application Data	
	22 2.165145	192.168.152.128	54.37.202.229	STUN	62	Binding Request	V
<		100 100 100 100	F4 37 000 000		~		_
) 🝸 TLS_a_Non	e.pcapng				Packets: 1159 · Displayed: 1159 (100.0%) Profile: Defau	lt _

Obr. B.13: Zachycené pakety protokolu TLS pro zamezení zachycení údajů ze SIP protokolu při navazování hovoru.

Úkoly:

- (5) Vyberte typ šifrování SRTP a opakujte předešlé kroky.
- (6) Lze hovor zachytit ve Wiresharku? Lze jej přehrát?
- (7) Ve které části RTP paketu jsou uložena šifrovaná data? Kolik bajtů obsahují?
- (8) Změňte protokol pro navázání hovoru na TLS a opakujte předešlé kroky.
- (9) Lze hovor zachytit ve Wiresharku? Lze jej přehrát?

B.2.4 Představení funkce analýzy pomocí grafů

Nyní si na našem VoIP hovoru vysvětlíme další z funkcí Wiresharku. Tím je možnost sestavit grafy z paketů, které byly zachyceny, jako na obr. B.14 a B.15. Grafy budeme vytvářet pro první zachycený hovor, který nebyl šifrován. Otevřete dříve uložený soubor přes File > Open. Vytváření grafů se nachází ve Statistics > I/O Graph. Otevře se nám nové okno s grafem, který odpovídá výchozímu nastavení. To obsahuje graf zobrazující všechny zachycené pakety v závislosti na čase. V dolní části okna se nachází prvky pro přizpůsobení a zobrazení vlastních grafů. Tlačítky plus a minus můžeme přidávat nebo odstraňovat grafy, které se mohou zobrazovat i přes sebe, tzn. více křivek zobrazených v jednom grafu, které jsou barevně odlišeny. Další tlačítka slouží pro kopírování nebo odstraňení všech vytvořených grafů z menu nad tlačítky. Dalším zajímavým prvkem je možnost měnit časový interval. Pro krátký hovor do

několika sekund je ideální interval nastavit na 100 ms. Pozor na tlačítko "Reset" v dolním pravém rohu, neslouží k resetování přiblížení grafu, tak aby byl graf vidět celý. Tlačítko smaže všechny vytvořené křivky v grafu. K resetování přiblížení do původní polohy musíme kliknout pravým tlačítkem na graf a zvolit "Reset Graph".

Naším cílem bude pro seznámení se s grafy ve Wiresharku vytvořit dva grafy, které budou zobrazovat množství TCP a UDP paketů, které byly zachyceny v průběhu hovoru. Přidejte tedy nový graf resp. křivku kliknutím na "+". Pro jeho zobrazení zaškrtněte položku enabled. Graph name zvolte stejný jako Display filter, tedy například tcp. Display filter funguje stejně jako filtrování zachycených paketů v hlavním okně Wiresharku. Lze tedy kombinovat filtry na protokoly, IP adresy, porty... Interval nastavte na 100 ms.





Obr. B.14: Vytvořené grafy pro protokoly TCP a UDP ze zachyceného VoIP hovoru.

Z grafu na obr. B.14 vyplývá, že TCP protokol je využit pro sestavení a ukončení hovoru. Pakety jsou přeneseny při stisknutí tlačítka pro spuštění volání, následně se vyčkává až volaný hovor přijme, takže nejsou přenášeny žádné pakety. Informace o přijetí hovoru ještě taktéž přenese protokol TCP a následný přenos dat s hlasem již realizuje UDP protokol.

Úkoly:

- (11) Pomocí filtru IP adresy vytvořte další 2 křivky v grafu, kdy každá bude zobrazovat buď příchozí nebo odchozí množství UDP paketů.
- (12) Přepněte zobrazení osy Y z paketů na bity a interval změňte na 1s.
- (13) Odpovídají hodnoty bitů v grafu hodnotám z teoretického úvodu o PCMU kodeku?



Obr. B.15: Vytvořený graf analyzující VoIP hovor přes I/O Graph ve Wiresharku. Vidíme rozdělený tok UDP pro příchozí a odchozí hovory, kde počet bitů přenesených za 1 sekundu odpovídá udávanému bitratu kodeku PCMU 80 kbit/s.

B.2.5 Spojení zachycených souborů

Pro porovnání transportních protokolů bude potřeba najít situaci, kde můžeme mezi těmito protokoly přepínat a zvolit, který z nich zrovna využijeme. Z toho plyne, že ve Wiresharku budeme provádět zachycení paketů dvakrát, jednou pro každý protokol. Následně musíme tyto zachycené soubory spojit do jednoho, abychom mohli provádět srovnání v jednom grafu přes funkcionalitu I/O Graph.

Pro vyzkoušení si zkusíme spojit dohromady dva předpřipravené soubory. Ty se nacházejí v připraveném virtualizovaném operačním systému v: C:\BKOM\Druhý scénář\Spojení souborů. Ve Wiresharku otevřete jeden z těchto souborů. Druhý soubor pak otevřete přes File > Merge..., nikoliv přes běžnou cestu File > Open. Tím jsme spojily soubory dohromady.

Úkoly:

(14) Vytvořte ze spojených souborů graf, který zobrazí počet použitých paketů TCP a UDP pro zobrazení webové stránky.

Na vytvořeném grafu vidíme, že soubory byly vytvořeny v delším časovém rozestupu od sebe (osa x) a tudíž nám graf nezobrazuje mnoho užitečných informací.

Je tedy potřeba upravit hodnotu "Time" tak, aby začátek obou souborů byl ve stejném čase. Potřebujeme najít první paket ze souboru, který byl vytvořen později. Všechny pakety později vytvořeného souboru budou mít ve sloupci "Time" o několik desítek sekund více než pakety prvního souboru. Pakety seřaďte vzestupně kliknutím na sloupec "Time" a najděte první paket druhého souboru, kde bude skokově vyšší čas. Na paket klikněte pravým tlačítkem a zvolte **Set/Unset Time Reference**. Zkontrolujte, že se hodnota "time" prvního paketu druhého souboru změnila na ***REF***, tak jako na obr. B.16. Tím bude tento paket a všechny za ním posunuty a budou počítány opět od začátku. To nám pomůže lépe zobrazit požadovaný graf, nyní by měl vypadat jako na obr. B.17.

	*(Unt	itled)					- 🗆 X
File	Ec	lit View G	o Capture Analyz	e Statistics Tele	phony 1	Wireless	Tools Help
		10	🗟 🗙 🖸 🔍 🥌	⇒ ≤2 A 4 5		e e	
							,
	Apply	a display filter	. <ctrl-></ctrl->	1			+
No.		Time	Source	Destination	Protocol	Length	Info
	91	4.102554	192.168.0.100	8.8.4.4	TCP	66	1404 → 53 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
	92	4.129300	8.8.4.4	192.168.0.100	TCP	66	53 → 1404 [SYN, ACK] Seq=0 Ack=1 Win=60720 Len=0 MSS=1380 SACK_PER
	93	4.129414	192.168.0.100	8.8.4.4	TCP	54	1404 → 53 [ACK] Seq=1 Ack=1 Win=131072 Len=0
	94	4.129556	192.168.0.100	8.8.4.4	DNS	83	Standard query 0x0000 A seznam.cz
	95	4.150293	8.8.4.4	192.168.0.100	TCP	54	53 → 1404 [ACK] Seq=1 Ack=30 Win=60928 Len=0
	96	4.150349	8.8.4.4	192.168.0.100	DNS	115	Standard query response 0x0000 A seznam.cz A 77.75.75.176 A 77.75.
	97	4.150455	192.168.0.100	8.8.4.4	TCP	54	1404 → 53 [FIN, ACK] Seq=30 Ack=62 Win=130816 Len=0
	98	4.170009	8.8.4.4	192.168.0.100	TCP	54	53 → 1404 [FIN, ACK] Seq=62 Ack=31 Win=60928 Len=0
	99	4.170099	192.168.0.100	8.8.4.4	TCP	54	1404 → 53 [ACK] Seq=31 Ack=63 Win=130816 Len=0
	100	4.440779	192.168.0.172	255.255.255	UDP	217	49154 → 6666 Len=175
7	101	*REF*	192.168.0.100	8.8.4.4	DNS	69	Standard query 0x0000 A seznam.cz
+⊢	102	0.026241	8.8.4.4	192.168.0.100	DNS	101	Standard query response 0x0000 A seznam.cz A 77.75.75.176 A 77.75.
	103	0.040752	192.168.0.172	255.255.255	UDP	217	49154 → 6666 Len=175
	104	0.419538	172.217.23.234	192.168.0.100	TLSv1	99	[TCP Previous segment not captured] , Application Data
	105	0.459109	192.168.0.100	172.217.23.2	TCP	54	[TCP ACKed unseen segment] 1552 → 443 [ACK] Seq=1 Ack=316 Win=511
	106	0.781174	192.168.0.100	8.8.4.4	DNS	69	Standard query 0x0000 A seznam.cz
	107	0.816684	8.8.4.4	192.168.0.100	DNS	101	Standard query response 0x0000 A seznam.cz A //./5./5.1/2 A //./5.
	108	1.524096	192.168.0.100	8.8.4.4	DNS	69	Standard query 0x0000 A seznam.cz
	109	1.546067	0.0.4.4	192.108.0.100	DNS	101	Standard query response 0x0000 A seznam.cz A //./5./5.1/6 A //./5.
	110	2.062109	192.100.0.100	0.0.4.4	DNS	69	Standard query 0x0000 A Seznam.cz
	112	2.115091	102 162 0 100	192.100.0.100	DNS	101	Standard query response 0x0000 A seznam.cz A //./5./5.1/2 A //./5.
	112	2.447205	192.100.0.100	0.0.4.4	DNS		
<							>
0	7	wireshark_202	200408152448_a10268.j	ocapng			Packets: 152 · Displayed: 152 (100.0%) Profile: Default

Obr. B.16: Spojení dvou souborů ve Wiresharku a určení referenčního paketu pro synchronizování časové osy.



Obr. B.17: Graf ve Wiresharku po správném nastavení časové reference zobrazující počet paketů TCP a UDP potřebných pro načtení webové stránky.

B.2.6 Srovnání protokolů TCP a UDP u DNS resolveru

Teď již umíme spojit soubory dohromady. Nyní tedy vytvoříme dva soubory pro porovnání množství paketů, které se použijí pro přenesení DNS dotazů. Otevřeme programy Wireshark a DNS resolver, který jsme používali již v minulé úloze. DNS resolver nám vygeneruje DNS dotazy, které zachytíme Wiresharkem jak pro TCP, tak následně pro UDP do dvou souborů.

V DNS resolveru vyplníme první pole doménovým jménem, například seznam.cz. Dále zvolíme adresu DNS adresu, například 8.8.8.8 a ještě ve druhém řádku zvolíme IN a A, pro využití IPv4 adres (viz obr. B.18). Odškrtneme položku DnsSec a vybereme transportní protokol TCP. Tím je program připraven.

DNS resolver					_	×
Menu						
seznam.cz	8.8.8.8	● IPv4	UDP O TCP	Send		
IN Y A Y	fec0:0:0:ffff::1%1	⊖ IPv€	✓ Recursior DnsSec			
IN ¥ A ¥	Tecu:u:u:tttt::1%1	O IPvť	DnsSec			

Obr. B.18: Nastavení DNS resolveru pro vygenerování DNS dotazů.

Ve Wiresharku zapneme zachytávání síťového provozu a v DNS resolveru odešleme 10 DNS požadavků za sebou tlačítkem **Send**. Větší počet odeslaných DNS dotazů volíme, aby byl v grafu lépe vidět rozdíl mezi protokoly. Zachytávání zastavíme a soubor uložíme, do názvu souboru vepište použitý transportní protokol pro lepší orientaci. Následně změníme v DNS resolveru transportní protokol na UDP a postup opakujeme. Opět vygenerujeme 10 DNS požadavků a soubor uložíme. Snažte se dobu zachytávání paketů držet stejnou pro oba soubory (10 až 20 s).

Po uložení druhého souboru s použitým protokolem UDP, zvolíme File > Merge... a vybereme soubor s použitým protokolem TCP. Soubory spojíme a upravíme referenční paket pro změnu času stejně jako v minulé podkapitole B.2.5.

Nyní máme soubor připraven k vytvoření grafu pro názorné srovnání. Otevřete Statistics > I/O Graph a vytvořte dvě křivky podobně jako na obr. B.19, zobrazující množství paketů, které se použije pro přenos DNS dat u jednotlivých protokolů.

Úkoly:

- (15) Vytvořte popsané grafy s použitím filtru IP adresy DNS serveru a použitého transportního protokolu.
- (16) Který z transportních protokolů potřebuje více paketů pro přenos těchto dat? Proč tomu tak je.



Obr. B.19: Porovnání počtu použitých paketů TCP a UDP pro přenos deseti DNS dotazů.

B.2.7 Nastavení prohlížeče a Wiresharku pro analýzu protokolu QUIC

V této části vyzkoušíme načítání této webové stránky² přes oba transportní protokoly. Pro načtení webové stránky pomocí protokolu QUIC, který pro přenos dat využívá transportní protokol UDP, využijeme možnosti v nastavení webového prohlížeče Chrome. Do tohoto nastavení se dostaneme tak, že zadáme do adresního řádku prohlížeče Chrome tento text: chrome://flags/#enable-quic. V případě, že máme povoleno ("enabled") použití experimentálního protokolu QUIC, jako na obr. B.20, tak se bude využívát při načtení stránky protokol UDP. V případě opačném ("disabled"), se použije TCP. V případě, že provedeme nějakou změnu, je nutné prohlížeče vypnout a opětovně zapnout, pro provedení změn.

Povolte používání protokolu QUIC a restartujte prohlížeč.³ Načtěte testovací webovou stránku⁴. V pravém horním rohu prohlížeče si všimněte ikony rozšíření

²https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka

³Testováno na prohlížeči Chrome, verze 81.0.4044.138

⁴https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka

pro prohlížeč, viz obr. B.21. Jedná se o aplikaci QUIC indicator, která je zdarma dostupná v internetovém obchodě Chrome⁵. V případě, že je ikona zelená, tak se využívá QUIC protokol. Najeďte na ikonu myší. Rozšíření nás informuje o použití QUIC protokolu a také uvádí jeho verzi.



Obr. B.20: Nastavení experimentálních funkcí v prohlížeči Chrome, včetně povolení QUIC protocolu.



Obr. B.21: Testovací webová stránka a ikona rozšíření prohlížeče, která indikuje použití protokolu QUIC.

⁵https://chrome.google.com/webstore/detail/quic-indicator/ dghiicffobcmljfmlaedpmpmgpnjcabc?hl=cs

Bohužel Wireshark sám od sebe nerozpozná pakety s protokolem QUIC, musíme tedy ručně upravit zobrazování tohoto protokolu. V horním textovém menu zvolte Edit > Preferences... V nově otevřeném okně v menu vlevo otevřete menu Protocols > QUIC. Seznam protokolů je dlouhý, proto můžete klávesou "Q" přeskočit až na protokoly začínající tímto písmenem. Při kliknutí na protokol QUIC se zobrazí pouze jedno volné pole, jako na obr. B.22. Do pole vepište hodnotu 443. Tím se všechny UDP pakety, které přichází z portu 443 přemění na QUIC pakety. V názvu protokolu uvidíme místo UDP QUIC a také můžeme zobrazit QUIC záhlaví a přenášená šifrovaná data.

0_2_100_UDP.pcapng	_		\times
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help			
◢ ■ ∅ ® 📴 🖾 🕱 🖆 ९ ⇔ ∞ 🕾 🖗 💆 🚍 🖲 ९, ९, ९, छ			
Apply a display filter <ctrl-></ctrl->	Т т Бх	pression	. +
No. Time Source Destination Protocol Length Info			^
No. Time Source Destination Protocol Length Info 1 0.000000 192.168.171.129 172.217.23.234 QUIC 1392 Initial, DCID=1b57f46f90486f13[Dissect 3 0.002068 192.166.171.129 216.58.201.110 QUIC 1392 Initial, DCID=cb608eaf8ad4765d[Dissect 4 0.003342 192.168.171.129 172.217.23. Wireshark · Preferences 5 0.226172 172.217.23.234 192.168.171.129 172.217.23. 6 0.226173 216.58.201.110 192.168.171.129 172.217.23. 9 0.227037 192.168.171.129 172.217.23. ProtoBuf PIP 9 0.227751 192.168.171.129 172.217.23.23 Quic UDP port 443 9 0.227037 192.168.171.129 172.217.23.23 Quic UDP port 443 9 0.227751 192.168.171.129 172.217.23.23 Quic UDP port 443 9 UIS PYFS Q.331 Quic UDP port 443 9 UIC Connection information QUAKE3 QUAKE3 QUAKE3 0 UIC Connection information RADIUS RADIUS RADIUS	or bug,	pr pr	
0000 00000000 00001100 00100100 00100100 00100100 Riemann 0010 00000100 11110010 10001100 0000100 0000100 0 0010 00000100 11110010 10001100 00000000 0 > 0010 00000100 10001110 0000000 0 > 0010 00000100 10001110 0000000 >		OK	

Obr. B.22: Nastavení programu Wireshark pro správnou detekci protokolu QUIC. V pozadí je již ukázka správné detekce protokolu a také část záhlaví.

QUIC protokol má 2 varianty záhlaví, krátké "short header" a dlouhé "long header". Dlouhé záhlaví je využíváno hlavně na začátku přenosu dat protokolem QUIC, než se strany dohodnou na verzi QUICu, kterou budou využívat a také při výměně klíčů pro zabezpečený přenos. Delší typ záhlaví uvádí více informací v nezašifrované podobě, jako například verze protokolu nebo zdrojové a cílové ID spojení. Wireshark si v aktuální verzi moc neumí poradit s detekcí verze QUIC protokolu. Stejně jako na obr. B.23 nám Wireshark vypíše že verze je neznámá. Nicméně v zobrazení dat v ASCII kódu můžeme vidět, že je zde verze uvedená, jedná se o verzi 46. Toto číslo také koresponduje s verzí, která je uvedena v rozšíření QUIC indicator (viz začátek kapitoly B.2.7). Krátká verze záhlaví je využívána u většiny zachycených QUIC paketů. Tento typ záhlaví obsahuje pouze jeden bajt, kde nás jeden bit informuje například o tom, který typ záhlaví paket obsahuje. Pro krátké záhlaví je použita hodnota 0 a dlouhé záhlaví je indikováno hodnotou 1.



Obr. B.23: Podrobnosti paketu včetně záhlaví protokolu QUIC a zobrazení verze v ASCII kódu.

B.2.8 Srovnání protokolů při načítání webových stránek

Zapněte zachytávání paketů přes Capture > Options..., vložte filtr udp port https, vyberte Ethernet a spusťte zachytávání tlačítkem Start, viz obr. B.24. Obnovte webovou stránku⁶ pomocí klávesové zkratky Ctrl+F5. Kombinace s klávesou "control" zajistí, že se webová stránka načte kompletně znovu a nepoužije se obsah uložený v paměti z předešlého načítání. Zastavte zachytávání paketů a soubor uložte.

Následně v kartě nastavení experimentálních funkcí zakažte použití protokolu QUIC a restartujte prohlížeč. Spusťte nové zachytávání paketů přes Capture > Options..., vložte filtr tcp port https, vyberte Ethernet a spusťte zachytávání tlačítkem Start. Opět obnovte testovací webovou stránku pomocí zkratky Ctrl+F5. Opět zastavte zachytávání paketů a soubor uložte.

⁶https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka

iterrace	Traffic	Link-layer Header	Promis	Snaplen	Buffer (N	Monit	Capture Filter	
Připojení k místní síti* 7		Ethernet	\checkmark	default	2	_		
Připojení k místní síti* 6		Ethernet	\checkmark	default	2	_		
Wi-Fi		Ethernet	\checkmark	default	2			
VMware Network Adapter VMn	et8	Ethernet	\checkmark	default	2			
Připojení k místní síti* 8		Ethernet	\sim	default	2			
Připojení k místní síti* 9		Ethernet	\sim	default	2	_		
VirtualBox Host-Only Network #	#2	Ethernet	\sim	default	2	_		
Adapter for loopback traffic cap	ture	BSD loopback	\sim	default	2	_		
Ethernet		Ethernet	\checkmark	default	2	—	udp port https	
USBPcap1		USBPcap	_	_	_	_		
USBPcap2		USBPcap	_	_	_	—		
USBPcap2		USBPcap	_	_	_	_		

Obr. B.24: Zachytávání paketů s předem vloženým filtrem, aby se zachytily pouze UDP pakety přenášející HTTPS.

Spojte tento soubor s předchozím souborem, kde byla stránka načtena pomocí protokolu QUIC, přes File > Merge.... Nastavte časovou referenci později vytvořeného souboru, podobně jako v podkapitole B.2.5. Ve Wiresharku nebo na obr. B.25 vidíme, že stejně jako u UDP je využit protokol QUIC, tak u TCP se využívá protokol TLS, který zajišťuje zabezpečenou komunikaci.

	🚺 *Ethern	et0								_		×
E	le <u>E</u> dit	<u>V</u> iew <u>G</u> o	<u>C</u> apture <u>A</u> na	alyze <u>S</u> tatisti	cs Telephony	<u>v</u> ireless	Tools	<u>H</u> elp				
		🛞 📙 🚮	🗙 🖨 🤇 🖉	⇔ ⇒ 😤 (F 🕹 📃 🔳	€. 0,	Q. 🎹					
	tls or quic										Expression.	. +
No		Time	Source	Destin	ation	Protocol	Length	Info				^
	1259	0.952606	172.217.23	.225 192.	168.171.129	TLS∀1.3	1424	Application	Data			
	1260	0.952606	172.217.23	.225 192.	168.171.129	TLS∀1.3	1514	Application	Data			
	1261	0.952606	172.217.23	.225 192.	168.171.129	TLSv1.3	1454	Application	Data			
	1262	0.952606	172.217.23	.225 192.	168.171.129	TLS∀1.3	1484	Application	Data			
	1263	0.952607	172.217.23	.225 192.	168.171.129	TLS∀1.3	1373	Application	Data			
	1265	0.952716	172.217.23	.225 192.	168.171.129	TLS∀1.3	85	Application	Data			
	1266	0.952716	172.217.23	.225 192.	168.171.129	TLS∀1.3	93	Application	Data			
	1268	0.953046	192.168.17	1.129 172.	217.23.225	TLS∀1.3	93	Application	Data			
	1273	*REF*	192.168.17	1.129 172.	217.23.234	QUIC	1392	Initial, DC	ID=9fee	2fe41b013	542[Diss	
	1274	0.003415	192.168.17	1.129 216.	58.201.110	QUIC	123	Protected F	ayload	(KPØ)		
	1275	0.019046	172.217.23	.234 192.	168.171.129	QUIC	1392	Initial, SC	ID=9fee	2fe41b013	542[Diss	
1	1276	a angaza	172 217 23	234 192	168 171 129	OUTC	1392	Initial SC	TD=9fee	2fe41h017	S42[Died. ≯	•
			-						-			
12	Frame 1	1265: 85 by	tes on wire	(680 bits),	, 85 bytes c	aptured ((680 bit	s) on inter	face Ø	· -		
2	Etherne	et II, Src:	Vmware_t2:t	d:94 (00:50	0:56:†2:†d:9	4), Dst:	Vmware_	at:79:61 (0	0:0c:29	:at:79:bt)	
2	Interne	et Protocol	Version 4,	Src: 172.21	L7.23.225, D	st: 192.1	168.171.	129				
12	Transmi	LSSION LONT	roi Protocol	, She Port:	: 443, DST ⊢	ort: 4982	24, Seq:	515647, AC	K: 1164	, Len: 31		
ľ	- inanspo	ort Layer S	ecurity	ication Dat	- Dector-1-	btto eve						
	* 1134	ri.s kecora	Layer: Appi	reation Dat n Doto (02)	a Protocol:	urrh-ove	201-115					
		paque Type	: MPDIICation : 1 0 (Avaga:	n Dala (20) 2)								
	ν I	ength: 26	3 I.Z (0X030.	-)								
	E	incrysted A	nnlication D	ata: fr@Qa7	ch676f4422h	5ar5f60de	0-05408	65607240573	21 od			
		легурсеа м	ppireaction D	aca. (coba/	20070144220	Jaconosue	00000400	000070400701	JICU.			
	🕽 🏹 ра	ayload is encryp	oted application da	ata (tis.app_dat	:a), 26 bytes		Pac	ets: 2514 · Disp	layed: 225	64 (89.7%)	Profile: Def	ault

Obr. B.25: Spojené soubory se zachyceným protokolem TLS a QUIC.

Pro zobrazení požadovaných statistik nejdříve zvolíme vhodný filtr pro zobrazení pouze těch paketů, které se podílejí na přenosu webové stránky. Pro nezabezpečené stránky bychom zvolili filtr s portem 80, naše testovací stránka je zabezpečená, tudíž zvolíme port 443 a filtr bude vypadat následovně:

tcp.port == 443 or udp.port == 443

Nejdříve zobrazíme tabulkové statistiky porovnávající obě vyzkoušené metody pro přenos webové stránky. Otevřete Statistics > Protocol Hierarchy. Otevře se nové okno, podobné jako na obr. B.26. Nejdříve porovnejte počet paketů protokolu QUIC oproti TLS. V testovaném případě je vidět, že protokolu TLS stačilo méně paketů pro přenos stejné webové stránky. Když se ovšem k TLS přičtou ostatní TCP pakety, jako jsou například potvrzovací pakety s příznakem ACK, tak jsou na tom obě strany podobně a rozdíly jsou pouze několik desetin procenta.

0.01	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
rame	100.0	3383	100.0	3974791	403 k	0	0	0
🗸 Ethernet	100.0	3383	1.2	47362	4806	0	0	0
✓ Internet Protocol Version 4	100.0	3383	1.7	67660	6865	0	0	0
👻 User Datagram Protocol	50.7	1716	0.3	13728	1393	0	0	0
QUIC IETF	50.7	1716	48.0	1905916	193 k	1716	1905916	193 k
 Transmission Control Protoco 	49.3	1667	48.8	1939817	196 k	278	20266	2056
Transport Layer Security	42.9	1452	48.9	1945145	197 k	1389	1857439	188 k

Obr. B.26: Hierarchické statistiky zachycených protokolů při zadaném filtru pro zobrazení paketů s použitým portem 443.

Další sledovanou statistikou bude množství přenesených bajtů u jednotlivých metod. Jedná se o sloupec Bytes v již otevřeném okně. Opět jsou na tom obě metody podobně, což je logické vzhledem k tomu, že se z pohledu běžného uživatele nic nezměnilo a načítá se jedna a ta samá webová stránka. Jediný rozdíl je tak ve velikosti záhlaví porovnávaných protokolů a také ve způsobu šifrování dat, kdy každý z protokolů přenáší data (payload) v jinak velkých blocích.

V hostovském operačním systému vytvořte tabulku (jako na obr. B.27), do které budete postupně zaznamenávat počet paketů, počet bajtů a čas potřebný ke stažení webové stránky. Následně budeme měnit parametry sítě a hodnoty porovnávat. Počet paketů a bajtů pro výchozí nastavení doplňte do prvních dvou řádků tabulky z otevřeného okna **Protocol Hierarchy**.

	А	В	С	D	E	F	G	Н	1
1		bandwidth [kbps]	packet loss [%]	latency [ms]	počet bajtů	počet paketů	čas - Wireshark	čas - Chrome	
2	TCP+TLS		0	0					TCP+TLS
3	QUIC	-	0	0					QUIC
4	TCP+TLS	0		200					TCP+TLS
5	QUIC	-	0	200					QUIC
6	TCP+TLS			0					TCP+TLS
7	QUIC	-	2	0					QUIC
8	TCP+TLS	4000	0	0					TCP+TLS
9	QUIC	4000	0	0					QUIC

Obr. B.27: Vytvořená tabulka pro záznam hodnot jako je počet paketů, počet bajtů a čas u obou protokolů při různých nastaveních síťového adaptéru v programu VMware.

Nyní se přepněte do okna pro vytváření grafů přes Statistics > I/O Graph. Pro užitečnější formu zobrazení přepněte Interval na 10ms. Úkolem bude vytvořit dva grafy, zobrazující počet bajtů potřebných k přenosu dat webové stránky pomocí obou metod. Pro graf protokolu QUIC můžete využít filtr quic, případně udp.port == 443. Obě varianty by měly zobrazit stejnou křivku. U grafu s využitím protokolu TCP použijte filtr tcp.port == 443. V grafu totiž požadujeme zobrazení i TCP paketů, které slouží k potvrzování, případně sestavování a ukončování spojení. Nelze tudíž použít pouze filtr tls. Uložte si spojené soubory do jednoho souboru pomocí File > Save as... pro pozdější porovnání s dalšími grafy a zpracujte následující úkoly.

Úkoly:

- (17) Sestavte graf zobrazující počet celkově přenesených bajtů protokolu UDP (QUIC) a TCP (TCP+TLS).
- (18) Pomocí metody, která je popsána v následujícím odstavci, určete celkovou dobu přenosu u obou protokolů.

Dobu, kterou trvalo načítání stránky můžeme také zjistit přesněji pomocí sloupce **Time** v hlavním okně programu Wireshark. Při zadaném filtru tcp.port == 443 or udp.port == 443 zkontrolujte, že první zobrazovaný paket má hodnotu **Time** 0,0 s. V případě, že se zachytil nějaký provoz ještě před prvním zobrazovaným paketem, tak nastavte referenční hodnotu času pro první zobrazovaný paket a pak také pro první zobrazovaný paket ze druhého souboru. Při zadaném filtru tak budeme mít dva pakety s hodnotou **Time** 0,0 s. Požadovaný čas nyní zjistíme tak, že najdeme poslední zobrazovaný paket daného souboru. Čas, který je u něj zobrazen, odpovídá času potřebnému ke stažení webových stránek a k řádnému ukončení spojení, doplňte jej do vytvořené tabulky k oběma protokolům. Ani jeden z protokolů však není obecně lepší nebo rychlejší, vždy záleží na aktuální situaci. Poslední sledovanou hodnotou bude hodnota odečítaná z webového prohlížeče Chrome. Jedná se o hodnotu, která nám ukáže dobu, potřebnou k načtení a zobrazení stránky, není zde tedy oproti předchozí hodnotě započten čas potřebný k ukončení spojení. Zobrazte si testovací webovou stránku⁷. Pomocí klávesy F12 se přesuňte do vývojářského prostředí prohlížeče. V horním menu vyberte položku Network. Pro zaznamenání načítání webu stačí stránku obnovit pomocí zkratky Ctrl+F5, následně uvidíme výpis načtených prvků, stejně jako na obr. B.28. Po načtení stránky zastavte zaznamenávání logu červeným tlačítkem nahoře vlevo.

V jednotlivých sloupcích vidíme zleva název prvku stránky, metodu pomocí které byl prvek vyžádán ke stažení (na výběr máme z metod GET a POST). Dále se zobrazuje status stažení (nejčastější hodnota 200 značí úspěšný HTTP požadavek). Následující sloupec uvádí použitou verzi HTTP protokolu a v našem případě také verzi protokolu QUIC. Dále jsou uvedeny podrobnosti jako typ (script, font, jpeg...), velikost prvku a také čas potřebný ke stažení tohoto prvku. Nás bude nejvíce zajímat hodnota FINISH v dolní části obrazovky. Ta nám oznamuje dobu, za kterou se stránka načetla. Hodnotu zaznamenejte do vytvořené tabulky k oběma protokolům.

\leftarrow \rightarrow C \triangle $($ sites.google.com	/vutbr.cz/xsimaj01/domovská-stránk	а								\$	😁 Anonym	of) :
	🕞 🖬 Elements Console	Sources	Networ	k Performance	Memory	Application Se	curity A	udits				: ×
	🔸 🛇 🛛 🖓 🔍 📄 Preserve log	Dis	able cache	Online 🔻	± ±							-
	200 ms 400 ms		600 m	800	ms	1000 ms	1200 ms		1400 ms	1600 ms	1800 ms	2000
1 incrotaci												
lusiovaci												
wabavá				1								
webova	Name	Meth	Status	Protocol	Туре	Initiator	Size	Time	Waterfall			*
	domovsk%C3%A1-str%C3%A1nka	GET	200	http/2+quic/46	document	Other	10.2 kB	271 ms				
stránka	css?family=Lato%3A300%2C300ita	GET	200	http/2+quic/46	stylesheet	domovská-stránka	640 B	174 ms				
Juanka	css?family=Google+Sans:400,500	GET	200	http/2+quic/46	stylesheet	domovská-stránka	1.2 kB	173 ms				
	rs=AGEqA5kTh4JmIJaOPwUZ-y3Cr	GET	200	http/2+quic/46	stylesheet	domovská-stránka	64.2 kB	197 ms				
	client.js?onload=gapiLoaded	GET	200	http/2+quic/46	script	domovská-stránka	5.1 kB	160 ms	_			
	m=view	GET	200	http/2+quic/46	script	domovská-stránka	179 kB	282 ms				
	ntQ323r5519W2vlcvcU2fOxTaEx91	GET	200	http/2+quic/46	jpeg	domovská-stránka	168 kB	336 ms				
odkaz na nastavení vložte do	TDDNEvtsKHpqtDsx0EFHc60sc9-4t	GET	200	http/2+quic/46	jpeg	domovská-stránka	159 kB	419 ms				
adresního řádku v novém	olusZC50loUroFVQ5TID25R-VS8K	GET	200	http/2+quic/46	jpeg	domovská-stránka	112 kB	419 ms				
alrað wohového problížočo	 VjTs246mEgJGc0btykRSG4NTu3R 	GET	200	http/2+quic/46	jpeg	domovská-stránka	124 kB	571 ms				
Chromos	cb=gapiloaded_0	GET	200	http/2+quic/46	script	client.js?onload=ga	101 k8	386 ms				
chrome:	simple-header-blended-small.png	GET	200	http/2+quic/46	png	domovská-stránka	297 kB	578 ms				
chrome://flags/#enable-quic	KFOmCnqEu92Fr1Mu4mxK.woff2	GET	200	http/2+quic/46	font	domovská-stránka	15.8 kB	187 ms				
	S6u9w48MUTPHh7USSwiPGQ.woff2	GET	200	http/2+quic/46	font	domovská-stránka	23.3 kB	258 ms				
	logImpressions?authuser=0	POST	200	http/2+quic/46	xhr	m=view:895	113 B	326 ms		•		
	m=A4UTCb,xiqF3,sy2m,owcnme,q	GET	200	http/2+quic/46	script	m=view:960	24.2 kB	42 ms				
	m=EGNJFf	GET	200	http/2+quic/46	script	m=view:960	2.7 kB	38 ms				
	m=Ae65rd,CuaHnc,sy1y,sy20,fgj8R	GET	200	http/2+quic/46	script	m=view:960	8.5 kB	128 ms				
Harris .	m=pB6Zqd,sy11,sy12,IZT63,SF3gs	GET	200	http/2+quic/46	script	m=view:960	8.8 kB	90 ms				
	m=fuVYe,vVEdxc,sy32,sy36,sy3m,C	GET	200	http/2+quic/46	script	m=view:960	7.7 kB	81 ms				
Hims more strength and a second	94YVGiyYMVet95WQBv_JPXd5id6	GET	200	http/2+quic/46	png	Other	501 kB	473 ms				
CO HILL HE REPORT												
	21 requests 1.8 MB transferred 2.	9 MB reso	ources F	inish: 1.80 s DC	MContentLo	aded: 997 ms Load:	1.33 s					

Obr. B.28: Vývojářské prostředí webových stránek v prohlížeči Chrome zobrazující podrobnosti týkající se načítání stránky.

B.2.9 Změna parametrů sítě a analýza změn

V následující části budeme upravovat parametry používané komunikační sítě, respektive sítového adaptéru. Prostředí programu VMware nám totiž umožňuje

⁷https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka

měnit parametry jako je maximální šířka pásma, ztrátovost paketů a latence a to jak pro odchozí, tak i příchozí komunikaci. Můžeme tak uměle zhoršit vlastnosti přenosové trasy, aby bylo dosaženo vlastností, které předpokládáme u určité trasy, přes kterou pak nemusíme testovat. V takové situaci můžeme sledovat změny v účinnosti u jednotlivých protokolů a tím poukázat na výhody nebo nedostatky těchto protokolů.

Vypněte virtualizovaný operační systém a přejděte do nastavení síťového adaptéru v programu VMware. To lze provést buď kliknutím pravým tlačítkem myši na virtuální systém a poté Settings... nebo výběrem systému myší a následně Edit virtual machine settings, viz obr. B.29. Otevře se nové okno s nastavením, kde přejdeme do Network Adapter. Zde můžeme upravovat jakým způsobem bude náš virtuální systém přistupovat k internetu, ponecháme zaškrtnutý NAT, stejně jako na obr. B.30. Dále klikneme na Advanced... tím se otevře další okno s nastavením zmíněných parametrů.



Obr. B.29: Úvodní obrazovka programu VMware s volbami pro zvolený virtuální systém.



Obr. B.30: Nastavení virtualizovaného systému v programu VMware, konkrétně nastavení síťového adaptéru včetně nastavení parametrů jako maximální šířka pásma, ztrátovost paketů nebo latence.

Šířku pásma ponechte neomezenou, stejně tak ztrátovost paketů ("Packet Loss"). Změňte pouze latenci na 100 ms. Parametr nastavte pro oba směry ("Incoming Transfer, Outgoing Transfer"). Nastavení potvrďte tlačítkem OK pro obě okna s nastavením. Spusťte virtualizovaný operační systém.

Nejdříve ověříme nastavení parametrů v příkazové řádce. Proveďte příkaz ping na vutbr.cz. Zkontrolujte, že zpoždění paketů se pohybuje kolem 200 ms, stejně jako na obr. B.31. Přidaných sto milisekund simuluje zpoždění paketu echo request při cestě po síti. Ve skutečnosti toto zpoždění vytváří VMware, který paket pozdrží před odesláním. Stejně tak paket echo reply, který dorazí v běžném čase a VMware jeho zpoždění nasimuluje až po jeho přijetí. Wireshark nám ale časy ukazuje přesně tak, jako by se zpoždění vytvářelo při cestě po síti. Zbytek přesahující nad 200 ms je běžná latence, která se v rámci sítě VUT pohybuje v jednotkách milisekund.

V běžném provozu by zpoždění 100 až 200 ms odpovídalo přenosu dat mezi kontinenty. Touto změnou chceme zjistit, jak se budou protokoly chovat v jiných podmínkách.

Nyní provedeme stejné zachycení paketů při načítání webové stránky, stejně jako před restartováním virtualizovaného OS. Zapněte tedy zachytávání paketů ve Wiresharku a obnovte webovou stránku⁸ pomocí klávesové zkratky Ctrl+F5. Vyčkejte na úplné načtení stránky a se zpožděním pár sekund zastavte zachytávání paketů. Doba zachytávání by se měla pohybovat kolem 15 sekund. Soubor vhodně pojmenujte a uložte.

Následně v kartě nastavení experimentálních funkcí (chrome://flags/#enable-quic) povolte použití protokolu QUIC a restartujte

⁸https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka

prohlížeč. Spusťte nové zachytávání ve Wiresharku a opět obnovte testovací webovou stránku pomocí zkratky Ctrl+F5. Zhruba po 15 sekundách zastavte zachytávání paketů a soubor uložte.

Prikazový řádek - □ × C:\Users\student>ping vutbr.cz Pinging vutbr.cz [147.229.2.90] with 32 bytes of data: Reply from 147.229.2.90: bytes=32 time=205ms TTL=128 Reply from 147.229.2.90: bytes=32 time=205ms TTL=128 Reply from 147.229.2.90: bytes=32 time=203ms TTL=128 Ping statistics for 147.229.2.90: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), Approximate round trip times in milli-seconds: Minimum = 203ms, Maximum = 206ms, Average = 204ms C:\Users\student>

Obr. B.31: Příkaz ping v příkazové řádce s vyšší latencí po úpravě parametrů v nastavení programu VMware.

Úkoly:

- (19) Vytvořte nový sloupec v hlavním okně programu Wireshark, který nám u každého QUIC paketu vypíše použitý typ záhlaví. Využijte řetězec quic.header_form.
- (20) Kolik bajtů dat je potřeba pro přenos informace o používané verzi protokolu QUIC u delší verze záhlaví?

Spojte tento soubor se zachyceným protokolem QUIC s předchozím souborem, kde byla stránka načtena pomocí protokolu TCP+TLS, přes File > Merge... Nastavte časovou referenci později vytvořeného souboru, podobně jako v podkapitole B.2.5. Spojený soubor také uložte. Následně otevřete Statistics > Protocol Hierarchy, hodnoty zapište do vytvořené tabulky a zodpovězte následující úkoly.

Úkoly:

- (21) Jak se změnil procentuální poměr výskytu obou protokolů při vyšší latenci?
- (22) Jak se vložení umělého zpoždění projevilo na celkové době přenosu u jednotlivých protokolů?

Celý postup nyní zopakujte s tím, že latenci vrátíme na výchozí hodnotu (0 ms)a nastavte ztrátovost paketů na 2%. Poté co zapíšete hodnoty o zachycených paketech do tabulky, vraťte ztrátovost paketů na 0% a nastavte umělé omezení hodnoty šířky pásma na 4000 kbps. Opět opakujte celý proces zachycení paketů u obou protokolů, vždy vyčkejte na úplné načtení stránky. Spojte soubory, odečtěte hodnoty z tabulky **Statistics > Protocol Hierarchy** a také odečtěte čas z Wiresharku a následně z prohlížeče Chrome. Hodnoty zapište do tabulky a odpovězte na následující otázky.

Úkoly:

- (23) Okomentujte změnu počtu využitých paketů a také změnu počtu bajtů, které využili oba protokoly dle hodnot zapsaných v tabulce.
- (24) Který z testovaných parametrů měl největší vliv na dobu, která byla potřeba pro stažení a zobrazení webové stránky?

Po zápisu hodnot vypněte virtualizovaný systém a všechny parametry (ztrátovost, latence...) vratte na výchozí, nulovou hodnotu.

B.2.10 Analýza QUICu v prostředí prohlížeče Chrome

Prohlížeč Chrome nabízí utilitu pro zachycení webového provozu přímo v okně prohlížeče. Před spuštěním zachytávání ještě povolte použití protokolu QUIC při načítání webových stránek přes chrome://flags/#enable-quic. Následně vložte text: chrome://net-export/ do adresního řádku prohlížeče. Ukáže se nám jednoduché rozhraní, jako na obr. B.32.



Obr. B.32: Utilita pro zachycení webového provozu přímo v okně prohlížeče.

Vybereme Include raw bytes pro zachycení všech bajtů dat, včetně HTML kódu načítané stránky. Tlačítkem Start Logging to Disk a výběrem místa na disku pro uložení souboru zahájíme zachytávání webového provozu. Zároveň zapněte zachytávání síťového provozu ve Wiresharku pro následné porovnání. Poté v nové kartě prohlížeče načtěte testovací webovou stránku⁹. Po načtení stránky zastavte zachytávání tlačítkem **Stop Logging** a také zachytávání ve Wiresharku. Pro zobrazení obsahu souboru na disku využijeme utilitu Netlog viewer¹⁰. Úvodní obrazovka Netlog vieweru je zachycena na obr. B.33.



Obr. B.33: Utilita Netlog viewer pro zobrazení zachyceného webového provozu v prohlížeči Chrome.

Vyberte vytvořený soubor a nahrajte jej do Netlog vieweru. Po načtení se nám zobrazí úvodní obrazovka (viz obr. B.34), která nás informuje o podrobnostech nahraného souboru, včetně data, verze prohlížeče Chrome i verze operačního systému Windows.



Obr. B.34: Úvodní obrazovka utility Netlog viewer s podrobnostmi o souboru.

⁹https://sites.google.com/vutbr.cz/xsimaj01/domovska-stranka ¹⁰https://netlog-viewer.appspot.com/#import
Pro analýzu protokolu QUIC se přepněte do položky QUIC v levém menu. Zobrazí se nám dvě tabulky, jako na obr. B.35. První shrnuje vlastnosti zachyceného souboru vzhledem k protokolu QUIC, včetně podporované verze protokolu a také maximální možnou velikost paketů. Další možnosti nastavení protokolu QUIC v této tabulce nejsou pro účely úlohy důležité. Druhá tabulka zobrazuje podrobnosti o každém QUIC spojení zvlášť, jako například počet přijatých a odeslaných paketů nebo Connection ID, na které se zaměříme v rámci následujících úkolů.

\leftrightarrow \rightarrow	C netlog-viewer.appspot.com/#quic									(Q &	• I e
rome-net-e	xport-log.json											(
mport	QUIC Option	Value										
Proxy	Supported Versions	Q046										
i i i i i i i i i i i i i i i i i i i	Connection options											
vents	Max Packet Length	1350										
imeline	Idle Connection Timeout In Seconds	30										
INS	Reduced Ping Timeout In Seconds	15										
iockets	Packet Reader Yield After Duration in Milliseconds											
It-Svc	Mark QUIC Broken When Network Blackholes	false										
ITTD/2	Do Not Mark QUIC Broken on Network Changes	faise										
11119/2	Retry without Alt-Svc on QUIC Errors	true										
2UIC	Do Not Fragment	false										
Reporting	Allow Server Migrations	false										
lache	Migrate Sessions Early V2	false										
loduloe	Migrate Sessions on Network Change V2	false										
nouules	Retransmittable on Wire Timeout in Milliseconds	false										
rerender	Disable Bidirectional Streams	false										
	Race Cert Verification	false										
	Race Stale DNS On Connection	false										
	Estimate Initial RTT	false										
	Force Head of Line Blocking	false										
	Max Server Conflos Stored in Properties	0										
	Origins To Force QUIC On											
	Server Push Cancellation	false										
	QUIC sessions View live QUIC sessions											
	Host		Version	Peer address	Connection ID	Active stream count	Active streams	Total stream count	Packets Sent	Packets Lost	Packets Received	Conne
	apis.google.com:443		QUIC_VERSION_46	172.217.22.110:443	05a836f54f35754c	0	None	2	47	0	88	true
	fonts.googleapis.com:443		QUIC_VERSION_46	172.217.23.234:443	ef955b5e16ce06d6	0	None	2	8	0	9	true
	fonts.gstatic.com:443		QUIC_VERSION_46	216.58.201.67:443	86e16b9da14238f7	0	None	0	3	0	4	true
	fonts.gstatic.com:443		QUIC VERSION 46	216.58.201.67:443	70838a1a88f60572	0	None	5	41	0	69	true

Obr. B.35: Zobrazení vlastností QUIC paketů po jednotlivých QUIC spojeních.

Úkoly:

- (25) Ve Wiresharku vytvořte nový sloupec zobrazující hodnoty "Destination Connection ID" a "Source Connection ID". Využijte řetězec quic.dcid a quic.scid.
- (26) Porovnejte Connection ID zachycená v prohlížeči s těmi ve Wiresharku. Shodují se?

Dále nám Netlog viewer umožňuje zobrazit obsah jednotlivých paketů kliknutím na View live QUIC sessions. Následně vybereme první QUIC relaci, jako na obr. B.36. V pravé části obrazovky se zobrazí obsah záhlaví paketů, které jsou součástí vybrané relace. Obsah záhlaví není šifrovaný jako ve Wiresharku a obsahuje například ustanovení certifikátů. V případě zájmu můžete prozkoumat i následující QUIC relace.

Dále můžeme zobrazit HTML kód přenášené stránky. Vymažte filtr, který nám zobrazuje pouze QUIC relace ("type:QUIC_SESSION is:active") a vložte filtr vutbr. Zobrazí se nám relace odpovídající filtru. Vybereme relaci typu URL REQUEST, kde v popisu je url načítané stránky tak jako na obr. B.37. V pravé části vidíme HTTP odpověď, která obsahuje jednotlivé příznaky a tok dat. Pro zobrazení HTML kódu potřebujeme vyhledat (klávesová zkratka Ctrl+F5) příznak URL_REQUEST_JOB_FILTERED_BYTES_READ. Pod tímto příznakem máme v levé části jednotlivé bajty v hexadecimálním zápise a v pravé části už je samotný HTML kód v čitelné formě. Můžeme tak zobrazit například titulní název ("title") nebo nadpis stránky úrovně h1 ("testovaci webova stranka").

chrome-net-	exp	ort-log.j:	son					
Import Proxy	(?)	type:Q	UIC_SESSION is:act	live	4 of 98			
Events		ID	Source Type	Description		50273: QUIC_SE	SSION	
Timeline		50273	QUIC_SESSION	Ih4.googleusercor	ntent.com	Start Time: 2020-0)5-15 1	17:19:56.640
DNS		50290	QUIC_SESSION	Ih6.googleusercor	itent.com			
Sockets		50293	QUIC_SESSION	Ih5.googleusercor	ntent.com	t=25257 [st=	0]	+QUIC_SESSION [dt=5340+]
Alt-Svc		50302	QUIC_SESSION	fonts.gstatic.com				> host = "lh4.googleusercontent.com"
HTTP/2								> port = 443
QUIC								> privacy_mode = false
Cacho						+ 05050 F-+	21	> require_confirmation = false
Modules						t=25259 [st=	2]	SIGNED_CERTIFICATE_TIMESTAMPS_RECEIVED > ombaddad_cete_=_"APAAduCyHaYMi6LNiiBOb2b5K7mK15Bpa9p6cOpySVMt74u0YaA
Prerender								> scts from ocsp response = ""
Tratenda								> scts from tls extension = ""
						t=25259 [st=	2]	SIGNED_CERTIFICATE_TIMESTAMPS_CHECKED
								> scts = [{"extensions":"","hash_algorithm":"SHA-256","log_id":"sh4FzI
						t=25259 [st=	2]	CERT_CT_COMPLIANCE_CHECKED
								> certificate =
								BEGIN CERTIFICATE MTTPADCCDeigAwTRAgTRAKc6HdgKgHftCAAAAA90HowDOVTKoZTbycNAOELROAw
								UTTLADCCDCTRAMIDARINAKCOURRVBULFCAAAAAA2OUOMDQ13KOZIUVCUAQEEDQAM

Obr. B.36: Obsah paketů v nešifrované podobě zobrazený v Netlog vieweru.

(?) withr (B of 184) 3D 22 68 A4 44 7/mt core Core at a core	-export-log.json			tectovari	1/1	Ι.	. v	~		
64 69 76 3E 3C 2F 64 69 76 3E 3C 2F 64 69 76 3E 3C 2F 64 69 76 3E div×/div×/div×/div	export-log ison (?) wutbr (D) Source Type 42 QUIC_SESSION 222 URL_REQUEST 223 URL_REQUEST 224 HTTP_STREAM_JOB_CONTROLLER 275 URL_REQUEST 275 URL_REQUEST 297 URL_REQUEST 352 URL_REQUEST	6 of 184 Description https://sites.google.com/vutbr.cz/xsimaj01idomovska-stranka https://sites.google.com/vutbr.cz/xsimaj01idomovska-stranka https://sites.google.com/_view/logimpressions?authuser=0 https://apis.google.com/_isevalaps-static/_j6k-roz.gapi.cs.Gr3 https://sites.google.com/_view/logimpressions?authuser=0 https://sites.goog	30 22 68 4A 44 7 75 51 53 43 68 6 65 20 47 4E 7A 5 6C 61 73 73 30 2 41 79 62 22 3E 3 22 74 79 4A 43 7 61 5A 70 4A 63 7 63 6E 65 75 22 2 63 6E 65 75 22 2 63 6E 65 75 22 2 63 6E 65 73 74 6 20 51 64 67 73 74 72 64 69 76 3E 37 77 2 64 69 76 3E 30 77 72	testovaci 4 20 6A 4 20 6A 4 37 5 4E 63 12 6A 58 12 6A 58 12 6A 58 12 6A 58 12 6A 22 10 6C 68 12 68 2E 10 64 69 10 22 7A 13 74 79 16 74 79 16 76 66 17 76 66 16 68 17 76 66 16 68 17 76 66 17 76 66 17 76 66 17 76 66 18 77 76 19 77 76 10 62 68 19 77 76 10 62 68 10 64 69 10 65 10 66 10 66 10 66 10 69 10 65 10	1/1 58 48 39 51 64 65 20 77 48 22 35 3C 64 69 4B 39 61 64 20 70 20 35 62 64 4B 79 76 32 27 7A 44 63 62 62 74 35 42 76 32 74 34 64 53 22 74 34 64 53 22 74 34 65 22 34 72 30 22 67 4 36 56 67 46 5 36 56 67 46 5 63 36 56 67 46 5 63 69 76 83 2 76 36 32 27 64 46<	200 200 200 200 200 200 200 200	32 75 63 4B 3D 62 68 62 68 62 68 62 20 52 74 22 74 22 76 2F 3E	<pre>x uQSCk fZ2 0 e GNz lass= Ayb">x tyJC aZpAe 1 id= cneu" class jpb" -alig >test a str div>< </pre>	wNd-AH d jXKS jCsFc UNc">< "jXK9a <div o<br="">td mGz lkHyy "h.z4J dir=" zfr: style= n: cer ovaci anka<, /div></div>	iqUyc- ad D2 wHaqu (div c ad-SmK :lass= caTb b yc"> <h 19shs1 "lsr" 3Q duR ="text nter;" weboy /h1><!--/</th--></h

Obr. B.37: Relace zobrazující HTML kód načítané stránky

Úkoly:								
(27) Zobrazte si	QUIC	pakety	zachycené	ve	Wiresharku.	Můžeme	zde	také
zobrazit HTM	[L kód?	,						

C Řešení prvního simulačního scénáře

C.1 Popis prostředí programu Wireshark

- (1) Prostudujte hlavní prostředí Wiresharku, nahlédněte jaké možnosti nabízí hlavní menu a jeho položky.
- (2) Vyzkoušejte si základní operace programu a prozkoumejte pár zachycených paketů.

C.2 Základní zachycení ICMP paketů

(3) Zjistěte typ použitého transportního protokolu a také číslo cílového portu DNS serveru.

O: DNS používá ve výchozím nastavení transportní protokol UDP a servery využívají port číslo 53.

(4) Kolik bajtů má segment ICMP protokolu bez části nesoucí informace o IP protokolu, včetně 32 B přidaných dat?

O: ICMP záhlaví obsahuje 8 bajtů a přidaná data mají velikost 32 bajtů. Celkem tedy 40 bajtů.

C.3 Filtry a jejich kombinace

Tato kapitola neobsahuje kontrolní otázky.

C.4 Úprava parametrů pro Ping a zachycení

(5) Zjistěte název domény, která se nachází na IP adrese 93.185.98.100.

O: Na této adrese se nachází doména csfd.cz.

(6) Kolik bajtů dat je potřeba na přenesení názvu domény v DNS response paketu z předchozího úkolu?

O: Je potřeba 13 bajtů dat. Informaci nalezneme v paketu csfd.cz DNS response - Domain Name System (response) - Answers - csfd.cz - Domain Name = 13 bajtů (7) Na kolik fragmentů se rozdělí přidaná data v případě odeslání maximálního možného objemu dat příkazem ping na vutbr.cz? Bohužel virtualizovaný OS nedokáže přijmout zprávy ICMP echo reply s takovou velikostí přidaných dat. Využijte tedy příkazovou řádku i Wireshark z hostovského operačního systému.

O: Data se rozdělí na 45 fragmentů ((1472+(43*1480)+388) = 65500 bajtů). Viz obrázek C.1. První fragment obsahuje 1472 bajtů přidaných dat a 8 bajtů ICMP záhlaví. 43 fragmentů má maximální velikost a poslední fragment obsahuje zbytek dat do onoho maximálního objemu dat 65500 bajtů. Wireshark však k velikosti prvního fragmentu doplňuje i velikost ICMP záhlaví a tak v jedné z částí uvádí součet 65508 bajtů.

(8) Pro ping na doménu vutbr.cz nalezněte maximální hodnotu parametru -l, kdy ještě nedojde k fragmentaci a minimální hodnotu -l, kdy už ke fragmentaci dojde. Čím jsou tyto hodnoty dány?

O: Hranice je 1472 a 1473 bajtů. Hodnoty jsou dány jednotkou MTU (Maximum transmission unit).

11 (h-17)		
The ukon 7 - ping -1 00000 na vutbr (tragmenty), pcap	ng	
File Edit View Go Capture Analyze Statisti	cs leiephony wireless loois help	
🧸 🗏 🖉 🕒 🔚 🖾 🖉 🔍 🗢 🕾 🖞	/ ୬ 🚍 🔳 ལ, ལ, ལ, 뱊	
Apply a display filter <ctrl-></ctrl->		+ 💌 🗧
No. Time Source Destination	Protocol Length Info	^
 38 9.527200 192.168.0.112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto=	=ICMP 1, off=45880, ID=5ab8) [Reassembled in #5
 39 9.527200 192.168.0.112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto=	=ICMP 1, off=47360, ID=5ab8) [Reassembled in #5
 40 9.527201 192.168.0.112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto-	=ICMP 1, off=48840, ID=5ab8) [Reassembled in #5
 41 9.527201 192.168.0.112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto=	=ICMP 1, off=50320, ID=5ab8) [Reassembled in #5
42 9.527201 192.168.0.112 147.229.2	.90 IPv4 1514 Fragmented IP protocol (proto=	=ICMP 1, off=51800, ID=5ab8) [Reassembled in #5
 43 9.527202 192.168.0.112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto=	=ICMP 1, off=53280, ID=5ab8) [Reassembled in #5
 44 9.52/202 192.168.0.112 14/.229.2 45 0 527203 102 168 0 112 147.229.2 	.90 IPv4 1514 Fragmented IP protocol (proto=	-ICMP 1, Off=54/60, ID=5ab8) [Reassembled in #5
 45 5.527205 152.100.0.112 46 0 527202 102 162 0 112 147.225.2 	99 TPv4 1514 Pragmented IP protocol (proto-	-ICMP 1, off=50240, ID=5ab0) [Reassembled in #5
A7 9 527263 192.108.0.112 147.225.2	90 TBv4 1514 Fragmented TP protocol (proto-	-ICMP 1, off-50200 ID-5ab0) [Reassembled in #5
 48 9.527204 192.168.0.112 147.229.2 	.90 TPv4 1514 Fragmented TP protocol (proto-	=ICMP 1, off=60680, ID=5ab8) [Reassembled in #5
49 9.527204 192.168.0.112 147.229.2	.90 TPv4 1514 Fragmented IP protocol (proto-	=ICMP 1. off=62160, ID=5ab8) [Reassembled in #5
50 9.527204 192.168.0.112 147.229.2	.90 IPv4 1514 Fragmented IP protocol (proto-	=ICMP 1, off=63640, ID=5ab8) [Reassembled in #5
51 9.527205 192.168.0.112 147.229.2	.90 ICMP 422 Echo (ping) request id=0x0001	1, seq=77/19712, ttl=128 (reply in 96)
<		>
Screen AG and S3330 50100	1480 (0.000)	
[Frame: 40, pavload: 57720-59199	(1480 bytes)]	^
[Ename: 48 _ navload: 50200 00075]	(1480 bytes) $(1480 \pm 10 / 200) \pm 1$	Ethormot $(44P) = 4E44P$
[Ename: 49, payload: 62160-63639]	1480 bytes)1 (1400 + IP (20D) + I	Ethernet (146) = 1514 6
[Frame: 50, payload: 63640-65119	1480 bytes)]	
[Frame: 51, payload: 65120-65507	388 bytes)] 388 + IP (2	20B) + Ethernet (14B) = 422 B
[Fragment count: 45]		
[Reassembled IPv4 length: 65508]		
[Reassembled IPv4 data: 0800732f00	01004d6162636465666768696a6b6c6d6e6f70]	
 Internet Control Message Protocol 		
Type: 8 (Echo (ping) request)		
Code: 0		
Checksum: 0x/32+ [correct]	+ ICMD zéhleví	(8B) v prvním fragmotnu
[Checksum Status: Good] Identifier (RE): 1 (0x0001)	+ ICMF Zamavi	(OB) v prvnin nagmetnu
Identifier (LE): 255 (0x0001)		$\mathbf{\Lambda}$
Sequence number (BE): 77 (0x004d)		
Sequence number (LE): 19712 (0x4d00).		
[Response frame: 96]	1472 ± 42×1480 ± 280 = 6E	500 B
✓ Data (65500 bytes)	14/2 T 43X 1400 T 300 - 03	5500 B
Data: 6162636465666768696a6b6c6d6	6f707172737475767761	
[Length: 65500]		v
<		>
🔵 🌋 Internet Control Message Protocol (icmp), 65 508	byte(s)	Packets: 379 · Displayed: 379 (100.0%) Profile: Default

Obr. C.1: Zobrazení podrobností ICMP paketu s vysvětlivkami k zodpovězení otázky číslo (7).

C.5 Zachycení ICMP - zachycení chybového kódu

(9) Druhý případ zachyťte ve Wiresharku a vyhledejte typ a kód chyby ICMP protokolu.

O: ICMP type: 11 (Time to live exceeded), Code: 0 (Time to live exceeded in transit).

(10) Ve výpisu paketů vidíme chybové protokoly ICMP (v základním nastavení podbarveny černě), které v informační části mají popisek Time-to-Live exceeded. Kdo je odesilatelem těchto paketů z hlediska síťové vrstvy?

O: Adresa patří desátému (předposlednímu) síťovému prvku. Tento prvek sníží hodnotu echo request z 1 na 0 a tím pádem request nedojde do cíle. Tento prvek odešle chybovou zprávu (černě podbarvený paket hned pod echo requestem). Zdrojová adresa tedy patří desátému (předposlednímu) prvku na cestě k cíli. Počet prvků lze přesně určit přes příkaz tracert, který prvky vypíše i s IP adresami.

(11) Kolik bajtů je potřeba pro přenos hodnoty TTL a zdrojové a cílové IP adresy v IP záhlaví?

O: Pro přenos hodnoty TTL je potřeba jeden bajt a každá IP adresa potřebuje 4 bajty, celkem tedy 9 bajtů (1 + 4 + 4).

(12) Jaká je maximální velikost přidaných dat u pingu na vutbr.cz, která lze nastavit při vypnuté fragmentaci a zároveň nevypíše chybu v příkazové řádce?

O: Maximální velikost se pohybuje kolem 1452 až 1472 bajtů, viz kapitola o TTL, kde je vysvětleno co vše se přidává k datům a výsledná velikost ICMP paketu se pohybuje kolem 1500 bajtů, což je běžná hodnota MTU.

(13) Jaká chyba se vypíše v příkazové řádce v případě, že chceme bez fragmentace odeslat větší množství dat než je hodnota MTU? Hodnotu MTU zjistěte přes již dříve uvedený příkaz netsh interface ipv4 show interface.

O: Vypíše se chyba - Packet needs to be fragmented but DF set.

(14) Kolik ICMP paketů bylo zachyceno Wiresharkem v případě odesílání 1600 bajtů přidaných dat u pingu na vutbr.cz s parametrem -f?

O: Nezachytí se žádné pakety, protože se paket ani neodešle ze síťové karty našeho počítače, protože je překročena hodnota MTU.

C.6 Další prvky ICMP záhlaví

- (15) Jak Wireshark respektive příkazová řádka poznají, které IPv4 fragmenty bez ICMP záhlaví a ICMP pakety patří k sobě? (Nápověda: hledejte v IPv4 záhlaví.)
 O: ICMP pakety jsou se svými fragmenty spojeny pomocí identifikátoru, který se nachází v IPv4 záhlaví obou paketů. Toto číslo je stejné u obou a funguje tedy podobně jako sekvenční číslo u echo reply a echo request zpráv u paketů ICMP.
- (16) Jaká data vkládá ICMP protokol do přidaných dat, které se odesílají spolu s pingem. Hledejte v dolní části programu Wireshark, kde se přidaná data paketu zobrazují v ASCII znacích.

O: Pro vyplnění přidaných dat, které se připojují k pingu (ve výchozím nastavení 32 bajtů) se vkládají písmena abecedy v pořadí za sebou a až w. V případě že se odesílá vice než 23 bajtů přidaných dat, tak se písmena opakují opět od a po w.

C.7 DNS resolver

 $(17)\,$ Jaké doménové j
méno se skrývá za hledanou IP adresou 217.31.205.50?

O: Na této adrese lze najít stránky domény nic.cz.

- (18) Jaké hlavní činnosti má na starosti sdružení, které se nachází na dané adrese? O: Hlavními činnostmi sdružení (CZ.NIC) jsou provozování registru jmen domén registrovaných pod doménou CZ, zabezpečování provozu domény nejvyšší úrovně .CZ a osvěta v oblasti jmen domén. V současné době se sdružení intenzivně věnuje rozšiřování technologie DNSSEC a služby mojeID.
- (19) Pomocí kolika IPv4 a IPv6 adres lze ve webovém prohlížeči Chrome zobrazit stránku sport.cz?

O: Jedná se o 4 adresy. Dvě z toho jsou IPv4 (77.75.76.63 a 77.75.78.63) a dvě jsou IPv6 (2a02:598:2::63 a 2a02:598:a::78:63).

D Řešení druhého simulačního scénáře

D.1 Popis prostředí programu Linphone

D.2 Linphone registrace a nastavení

(1) Kontaktujte některého ze svých spolužáků a ověřte funkčnost spojení.

O: V případě, že se nedaří uskutečnit hovor je nutné překontrolovat správné vyplněné údaje v nastavení "SIP account". Případně zkontrolovat, že jsou vybrané správné audio kodeky PCMU a PCMA v "Audio settings" nebo to zda student potvrdil registraci Linphone účtu přes zadaný email.

D.3 Rekonstrukce hovoru a jeho základní zabezpečení

- (2) Projděte si výpis paketů zachyceného hovoru jak pro sestavení spojení (SIP), tak pro samotný přenos dat (RTP).
- (3) Jaký port je využíván při RTP komunikaci na našem zařízení?
 O: Jsou využívány porty číslo 7078 a 7076. Odpoveď nalezneme v zachycených paketech VoIP hovoru vybereme RTP paket, kde zdrojová adresa je naše IP adresa podrobnosti o paketu User Datagram protocol Source port: 7078 případně 7076.
- (4) Zobrazte si bajty jednotlivých RTP paketů v dolní části programu. Je možné z takovýchto dat zpětně rekonstruovat hovor?

O: Ano, je to možné. Rekonstrukce hovoru bude provedena v následujících podkapitolách.

- (5) Vyberte typ šifrování SRTP a opakujte předešlé kroky.
 O: Typ šifrování se zvolí v Linphone nastavení Advanced settings Media encryption SRTP.
- (6) Lze hovor zachytit ve Wiresharku? Lze jej přehrát?
 O: Hovor je zachycen, ale v šifrované podobě. LZE PŘEHRÁT, ale místo hovoru je zachyceno silné šumění.
- (7) Ve které části RTP paketu jsou uložena šifrovaná data? Kolik bajtů obsahují?
 O: Data se v SRTP paketu nachází v podrobnostech paketu Real
 Time transport protocol SRTP encrpted payload. Šifrovaná data jsou zasílána po částech o velikosti 160 bajtů. Stejná velikost je použita i u nešifrované varianty RTP.

(8) Změňte protokol pro navázání hovoru na TLS a opakujte předešlé kroky.

O: Změna se provede v nastavení programu Linphone - SIP accounts -Transport - TLS. Následně je potřeba nastavení uložit tlačítkem dole vpravo.

(9) Lze hovor zachytit ve Wiresharku? Lze jej přehrát?

O: Hovor nelze zachytit a tím pádem ani přehrát, protože Wireshark nedokáže ze zašifrovaných TLS paketů rozpoznat VoIP hovor.

D.4 Představení funkce analýzy pomocí grafů

(10) Vytvořte grafy pro oba transportní protokoly a zvolte vhodné barvy pro odlišení.

O: Vytvoří se dvě křivky s filtrem tcp a udp, stejně jako na obr. D.1.

(11) Pomocí filtru IP adresy vytvořte další 2 křivky v grafu, kdy každá bude zobrazovat buď příchozí nebo odchozí množství UDP paketů.

O: Vytvoří se dvě křivky s filtrem jako na obr. D.2.

- (12) Přepněte zobrazení osy Y z paketů na bity a interval změňte na 1s.
 O: Přepnutí se provádí ve stejném okně, kde se vytváří grafy, viz obr D.2.
- (13) Odpovídají hodnoty bitů v grafu hodnotám z teoretického úvodu o PCMU kodeku?

O: Ano odpovídají, viz obr. D.2. PCMU kodek je nastaven na bitrate 80 kb/s. My máme jeden stream pro přenos našeho hlasu k volanému a druhý stream, který přenáší hlas volaného k nám. Tzn. v grafu by měly být vidět 2 křivky s hodnotami kolem 80 kbps.



Obr. D.1: Vytvořené grafy pro protokoly TCP a UDP ze zachyceného VoIP hovoru.



Obr. D.2: Zobrazení toku protokolu UDP pro příchozí a odchozí hovory, kde počet bitů přenesených za 1 sekundu odpovídá udávanému bitratu kodeku PCMU 80 kbit/s.

D.5 Spojení zachycených souborů

(14) Vytvořte ze spojených souborů graf, který zobrazí počet použitých paketů TCP a UDP pro zobrazení webové stránky.

O: Otevřete jeden ze souborů a druhý soubor pak otevřeme přes File - Merge. Tím jsme spojily soubory dohromady a graf se vytvoří pomocí filtru tcp nebo udp stejně jako u úkolu D.4. Na vytvořeném grafu vidíme, že soubory byly vytvořeny v delším časovém rozestupu od sebe (osa x) a tudíž nám graf nezobrazuje mnoho užitečných informací. Proto je následně ve scénáři vysvětlena funkce nastavení referenčního času u později vytvořeného souboru.

D.6 Srovnání protokolů TCP a UDP u DNS resolveru

(15) Vytvořte popsané grafy s použitím filtru IP adresy DNS serveru a použitého transportního protokolu.

O: Použije se stejný filtr jako na obr. D.3.

(16) Který z transportních protokolů potřebuje více paketů pro přenos těchto dat? Proč tomu tak je.

O: Více paketů využívá v této situaci protokol TCP, protože kromě paketů s daty se přenáší také potvrzovací pakety s příznaky ACK a také se pro každý DNS dotaz provádí sestavení a ukončení TCP spojení pomocí příznaků SYN a FIN.



Obr. D.3: Porovnání počtu použitých paketů TCP a UDP pro přenos deseti DNS dotazů.

D.7 Nastavení prohlížeče a Wiresharku pro analýzu protokolu QUIC

D.8 Srovnání protokolů při načítání webových stránek

(17) Sestavte graf zobrazující počet celkově přenesených bajtů protokolu UDP (QUIC) a TCP (TCP+TLS).

O: Použijí se filtry uvedené ve scénáři. Použitý filtr lze vidět také na obr. D.4.

(18) Pomocí metody, která je popsána v následujícím odstavci, určete celkovou dobu přenosu u obou protokolů.



Obr. D.4: Graf srovnávající množství potřebných bajtů protokolu TCP+TLS a QUIC pro načtení webové stránky.

D.9 Změna parametrů sítě a analýza změn

(19) Vytvořte nový sloupec v hlavním okně programu Wireshark, který nám u každého QUIC paketu vypíše použitý typ záhlaví. Využijte řetězec quic.header_form.

O: V hlavním okně klikneme pravým tlačítkem myši na záhlaví okna se zachycenými pakety. Zvolíme "Column preferences" - dole tlačítko "+" - do sloupce Fields vložíme výše uvedený řetězec a pojmenujeme nově vytvořený sloupec ve sloupci Title. Potvrdíme tlačítkem OK.

(20) Kolik bajtů dat je potřeba pro přenos informace o používané verzi protokolu QUIC u delší verze záhlaví?

O: Verzi protokolu QUIC identifikují v záhlaví paketu 4 bajty. Odpověď nalezneme zvolením QUIC paketu s dlouhým záhlavím podrobnosti o paketu - QUIC IETF - QUIC Connection Information - Version. Při kliknutí na verzi se na dolní části okna ukáže velikost 4 bajty.

- (21) Jak se změnil procentuální poměr výskytu obou protokolů při vyšší latenci?
 O: Vyšší latence se projevila vyšším počtem paketů TCP protkolu. Množství QUIC paketů zůstalo přibližně stejné. Z toho nám vychází že TCP paketů bylo přibližně 75% a procenta u QUICu se zmenšili přibližně na 25%. Oproti původním hodnotám je to výrazná změna, před změnou oba protokoly využily zhruba stejné množství paketů.
- (22) Jak se vložení umělého zpoždění projevilo na celkové době přenosu u jednotlivých protokolů?

O: S ohledem na vyšší zpoždění se také zvýšila doba potřebná pro načtení stránky. Doba načítání se zvýšila na 2-3 sekundy. Hodnota se samozřejmě může lišit a záleží na aktuálních podmínkách.

(23) Okomentujte změnu počtu využitých paketů a také změnu počtu bajtů, které využili oba protokoly dle hodnot zapsaných v tabulce.

O: Tabulka zachycených hodnot při testování scénáře je zachycena na obr. D.5. Hodnoty se mohou lišit dle aktuální situace. Obecně lze říci, že největší vliv na počet paketů i bajtů měla zvolená hodnota omezující šířku pásma. (24) Který z testovaných parametrů měl největší vliv na dobu, která byla potřeba pro stažení a zobrazení webové stránky?

O: Největší vliv na dobu potřebnou ke stažení a zobrazení webové stránky měla nastavená hodnota maximální šířky pásma.

	A	В	С	D	E	F	G	Н	I.
1		bandwidth [kbps]	packet loss [%]	latency [ms]	počet bajtů	počet paketů	čas - Wireshark	čas - Chrome	
2	TCP+TLS		0	0	1,94 <mark>6</mark>	1612	0,65	1,32	TCP+TLS
3	QUIC	-	0	Ū	1,2	1122	0,729	1,15	QUIC
4	TCP+TLS	-	0	200	5,043	5765	7,46	3,86	TCP+TLS
5	QUIC				1,942	1692	3,16	2,39	QUIC
6	TCP+TLS		2	0	2,694	3188	2,699	1,94	TCP+TLS
7	QUIC	-	2	U	1,962	1978	1,343	1,34	QUIC
8	TCP+TLS	4000	0	0	8,633	10743	16,767	14,98	TCP+TLS
9	QUIC	4000		U	1,908	1699	6,369	5,28	QUIC

Obr. D.5: Tabulka zobrazující zachycené hodnoty při změne parametrů síťového adaptéru.

D.10 Analýza QUICu v prostředí prohlížeče Chrome

(25) Ve Wiresharku vytvořte nový sloupec zobrazující hodnoty "Destination Connection ID" a "Source Connection ID". Využijte řetězec quic.dcid a quic.scid.

O: V hlavním okně klikneme pravým tlačítkem myši na záhlaví okna se zachycenými pakety. Zvolíme "Column preferences" - dole tlačítko "+" - do sloupce Fields vložíme filtr (quic.dcid or quic.scid) a pojmenujeme nově vytvořený sloupec ve sloupci Title. Potvrdíme tlačítkem OK.

(26) Porovnejte Connection ID zachycená v prohlížeči s těmi ve Wiresharku. Shodují se?

O: Ano, hodnoty by se měly shodovat. Je možné že ve Wiresharku budou seřazeny v jiném pořadí než v utilitě Netlog Viewer ve webovém prohlížeči.

(27) Zobrazte si QUIC pakety zachycené ve Wiresharku. Můžeme zde také zobrazi
t $\rm HTML~k\acuteod?$

O: HTML kód zobrazit nemůžeme, protože Wireshark zachytil zašifrovanou podobu dat.

E Obsah poskytnutého odkazu se soubory

Všechny soubory potřebné k vypracování nebo pozdější editaci jednotlivých scénářů a vytvořených vektorových obrázků budou nahrány na toto vzdálené úložiště¹ dle struktury, která je uvedena níže. Bude zde také nahrán obraz virtualizovaného operačního systému. Soubory byly předány také vedoucímu práce. Stejnou strukturu bude mít i příloha odevzdaná do informačního systému s tím rozdílem, že do informačního systému nebude z důvodu velikosti souborů nahrán obraz virtualizovaného operačního systému (11 GB).

/kořenový adresář poskytnutého odkazu se soubory
První scénář Všechny soubory k prvnímu scénáři
Latex soubory Vše potřebné k editaci prvního scénáře
Laboratorní úloha
sablona-prace.tcpLatex projekt pro editaci prvního scénáře
sablona-prace.pdf Výsledný pdf soubor prvního scénáře
Otázky a odpovědi
otazky-prvni.tex Latex soubor pro editaci otázek a odpovědí
parametry.tcp Latex projekt pro editaci otázek a odpovědí
parametry.pdfVýsledný pdf soubor s otázkami a odpověďmi
Předpřipravené soubory (Wireshark)
První-scénář.pdf Vypracovaný první scénář
První-scénář-řešení.pdf Otázky a odpovědi k prvnímu scénáři
Druhý scénářVšechny soubory k druhému scénáři
Latex souboryVše potřebné k editaci druhého scénáře
Laboratorní úloha
sablona-prace.tcpLatex projekt pro editaci druhého scénáře
sablona-prace.pdfVýsledný pdf soubor druhého scénáře
Otázky a odpovědi
otazky-druhy.tex Latex soubor pro editaci otázek a odpovědí
parametry.tcp Latex projekt pro editaci otázek a odpovědí
parametry.pdfVýsledný pdf soubor s otázkami a odpověďmi
Předpřipravené soubory (Wireshark)
Druhý-scénář.pdfVypracovaný druhý scénář
Druhý-scénář-řešení.pdfOtázky a odpovědi k druhému scénáři
Bakalářská práceVšechny soubory k editaci BP
textSložka s tex soubory pro editaci bakalářské práce
sablona-prace.tcpLatex projekt pro editaci bakalářské práce
L sabiona-prace.pdf Výsledný pdf soubor bakalářské práce
Virtualizovaný operační systémSoubory pro import virtualizovaného OS
Editovatelne obrázky (vektory)Složka s editovatelnými obrázky
L Bakalářská-práce-Síma.pdf Kompletní bakalářská práce

 $^{{}^{1} \}tt{https://owncloud.cesnet.cz/index.php/s/HatHDvasMOBmGER}$