

Dokumentácia pre program Secret

ISA Projekt, Prenos súboru cez skrytý kanál, 2021/2022

Michal Šlesár (xslesa01)

12.11.2021

1. Obsah

1. Obsah	2
2. Úvod	3
3. Teória	3
4. Implementácia	4
4.1. Základné informácie	4
4.2 Funkcionalita	4
4.2.1 Múd klienta	4
4.2.2 Múd serveru	5
4.3.3 Štruktúra packetu	5
5. Testovanie	6
5.1 Test prenosu textového súboru (IPv4)	6
5.2 Test prenosu textového súboru (IPv6)	7
5.3 Test prenosu binárneho súboru (IPv4)	8
5.4 Test prenosu binárneho súboru (IPv6)	9
5.4 Valgrind	10
6. Zdroje	11

2. Úvod

Cieľom projektu do predmetu ISA bolo implementovať program pre šifrovanú komunikáciu medzi klientom a serverom, pomocou protokolu ICMP. Program operuje v dvoch módoch, mód klienta v ktorom užívateľ zadá súbor a adresu, na ktorú sa odošle v ICMP packetoch zašifrovaná verzia tohto súboru. Druhý mód je mód serveru, ktorý zachytáva zašifrované packety nášho programu, tie následne dešifruje a uloží do požadovaného súboru. Šifrovanie obsahu je implementované pomocou šifry AES.

3. Teória

Protokol ICMP sa využíva na odosielanie a prijímanie obslužných informácií, napríklad chybových stavov. Protokol neobsahuje porty a teda cieľové zariadenie dotazujeme iba pomocou IP adresy. Server nevie len na základe obsahu ICMP hlavičky povedať, ktorý packet mu bol cieľený, teda v implementácii bolo potrebné vyriešiť tento problém pomocou vlastného protokolu. Projekt využíva ICMP echo request a reply packety. ICMP echo reply packet sa generuje automaticky operačným systémom, za predpokladu že echo request má správne vypočítaný checksum.

4. Implementácia

4.1. Základné informácie

Na implementáciu programu som využil primárne jazyk C s podporou zopár špecifických konštrukcií jazyka C++, napríklad `std::string` a `std::map`. Program využíva knižnice **openssl**, kryptografickú knižnicu **crypto** a knižnicu na zachytávanie packetov **pcap**, ktorá sa využíva na strane serveru, z dôvodu jednoduchého zachytávania packetov a s ich následným spracovaním.

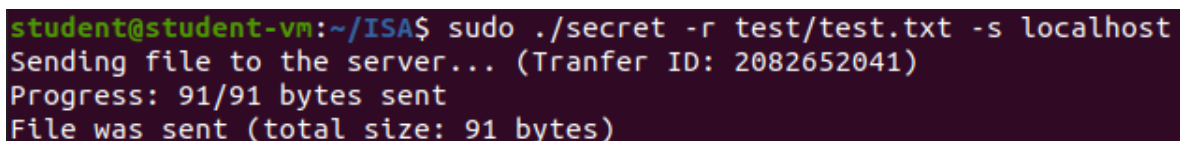
4.2 Funkcionalita

4.2.1 Mód klienta

Ak bol program spustený v móde klienta, užívateľ musí zadať dva parametre, súbor a adresu. Na začiatku programu sa overí, či požadovaný súbor a cieľová adresa existuje. Existujúci súbor je následne šifrovaný AES šifrou, a to postupne po 16 bytoch, ktoré sú zoskupené v packetoch o maximálnej dĺžky 1500 bytov. Pre zjednodušenie komunikácie medzi klientom a serverom disponuje zabezpečený protokol tohto programu vlastnou hlavičkou, v ktorej sa nachádzajú informácie ako napríklad ID prenosu, podľa ktorého vie program následne priradiť obsah jednotlivých packetov, hash programu, podľa ktorého vie server identifikovať, že sa jedná o packet tohto programu, teda server nespracúva cudzie ICMP packety. Maximálna veľkosť obsahu packetu je teda o čosi menšia, nakoľko od čísla 1500 ešte musíme odrátať veľkosti ethernetovej hlavičky, IP hlavičky a hlavičky nášho protokolu.

V prvom odoslanom packete (`file_packet`) sa nachádzajú úvodné informácie o prenose, ktoré obsahujú názov súboru, veľkosť súboru a ID prenosu.

V každom ďalšom packete (`data_packet`) sa nachádza v hlavičke už iba ID prenosu a obsah súboru. O stave prenosu je užívateľ informovaný počtom doposiaľ odoslaných bytov. Program končí odoslaním celého súboru.



```
student@student-vm:~/ISA$ sudo ./secret -r test/test.txt -s localhost
Sending file to the server... (Transfer ID: 2082652041)
Progress: 91/91 bytes sent
File was sent (total size: 91 bytes)
```

Obrázok 1 -Odoslanie súboru od klienta

4.2.2 Mód serveru

Ak bol program spustený v móde serveru, parametre súboru a adresy sú ignorované. Na začiatku si program nastaví prijímacie zariadenie pomocou knižnice **pcap**. Server odpočúva na zariadení “any”, aby bolo možné prijímať aj localhost správy. Celá funkcionálnosť prebieha v pcap handleri prijatého packetu. Ako prvé server zistí či sa jedná o IPV4 alebo IPV6 ICMP správu. Na základe toho sa vypočíta offset našich dát.

V prípade že server prijal file_packet, server vytvorí cieľový súbor a pridá informácie o prenose do zoznamu aktívnych prenosov.

V prípade že server prijal data_packet, server na základe ID prenosu pridá tieto dáta danému aktívnemu prenosu, obsah packetu odšifruje a zapíše do cieľového súboru. Prenos súboru končí úspešným zápisom plnej veľkosti súboru.

V prípade že server neobdržal už nejaký čas dáta k niektorému z aktívnych prenosov, je tento prenos ukončený.

```
student@student-vm:~/ISA$ sudo ./secret -l  
(TID: 2082652041) Started receiving file: test.txt, size: 91 bytes  
(TID: 2082652041) File received successfully
```

Obrázok 2 - Prijatie súboru na serveri

4.3.3 Štruktúra packetu - Protokol

Štruktúra packetu teda vypadá nasledovne:

- **Ethernet hlavička** - 16 bytov - (nie 14 nakoľko pcap používa vlastnú “cooked hlavičku” vrátane akejkoľvek rozhrania)
- **IP hlavička** - 20 alebo 40 bytov - podľa toho či sa jedná o IPv4 alebo IPv6 komunikáciu
- **ICMP hlavička** - 8 bytov
- Vlastná secret hlavička
 - **protocol hash** - 6 bytov - identifikácia secret packetu
 - **protocol type** - 4 byty - informácia o tom či sa jedná o file alebo data packet
 - **transfer id** - 4 byty - unikátny identifikátor prenosu
 - **File packet hlavička** - prvý packet prenosu
 - **transfer size** - 4 byty - celková veľkosť súboru
 - **transfer name** - premenná veľkosť - názov súboru
 - **Data packet hlavička** - zvyšné packety
 - **transfer data** - premenná veľkosť - zašifrovaný obsah súboru

5. Testovanie

Program som vyvíjal a priebežne testoval na svojom stroji s operačným systémom Windows, pomocou nástroja WSL. Testy do tejto dokumentácie som ale simuloval lokálne na referenčnom virtuálnom stroji s operačným systémom Ubuntu, pomocou nástroja **VMWare Workstation**.

Postupne som program otestoval na textovom a binárnom súbore osobitne pre IPv4 a IPv6.

Server bol spustený pri každom teste pomocou `sudo ./secret -l`

5.1 Test prenosu textového súboru (IPv4)

```
student@student-vm:~/ISA$ sudo ./secret -r test/lorem.txt -s localhost
Sending file to the server... (Transfer ID: 524121226)
Progress: 301856/301856 bytes sent
File was sent (total size: 301856 bytes)
```

Obrázok 3 - Odoslanie textového súboru

```
student@student-vm:~/ISA$ sudo ./secret -l
(TID: 524121226) Started receiving file: lorem.txt, size: 301856 bytes
(TID: 524121226) File received successfully
```

Obrázok 4 - Prijatie tohto textového súboru

```
student@student-vm:~/ISA$ md5sum lorem.txt
bdfa47a4d11bc74a2a290bc6b5847144  lorem.txt
student@student-vm:~/ISA$ md5sum test/lorem.txt
bdfa47a4d11bc74a2a290bc6b5847144  test/lorem.txt
```

Obrázok 5 - Porovnanie checksumu oboch súborov - sedí

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0x7558, seq=0/0, ttl=64 (reply in 2)
2	0.000010162	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0x7558, seq=0/0, ttl=64 (request in 1)
3	0.002239012	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0xd6e7, seq=256/1, ttl=64 (reply in 4)
4	0.002244183	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0xd6e7, seq=256/1, ttl=64 (request in 3)
5	0.008023595	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0xfd79, seq=512/2, ttl=64 (reply in 6)
6	0.008032636	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0xfd79, seq=512/2, ttl=64 (request in 5)
7	0.008797376	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0x6fc0, seq=768/3, ttl=64 (reply in 8)
8	0.008801567	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0x6fc0, seq=768/3, ttl=64 (request in 7)
9	0.009372259	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0x2973, seq=1024/4, ttl=64 (reply in ...)

▶ Frame 3: 1466 bytes on wire (11728 bits), 1466 bytes captured (11728 bits) on interface lo, id 0
 ▶ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 ▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 ▶ Internet Control Message Protocol

0000	00 00 00 00 00 00 00 00	00 00 00 00 08 00 45 00E..
0010	05 ac 79 87 40 00 40 01	bd c7 7f 00 00 01 7f 00	..y.@@.....
0020	00 01 08 00 26 6a d6 e7	01 00 49 53 41 5f 33 00	...&j...ISA_3
0030	00 00 02 00 00 00 8a 74	3d 1f 3a 14 dd 8d d1 bct=.....
0040	40 02 b9 01 1f 83 4f cb	5c 5c f2 c4 ee 5b 88 a30.\\.....[
0050	e8 c1 1d 5e 50 e6 f5 87	06 bb 60 ab 04 29 81 b7^P.....)
0060	c8 a1 24 8b 2f 3c 33 c1	5e 9b b4 17 dc 3e b2 40	...\$/<3^.....@
0070	15 7f 06 16 4f ce 59 3e	f7 28 45 ca 0a 79 83 0dO.Y>.(E.y..
0080	4d ed f3 b6 c9 81 44 76	c0 83 7f e5 c7 81 c8 b7	M.....Dv.....
0090	05 eb 60 62 58 a1 94 e8	59 0f 50 4d 64 68 b6 50	..bX...Y.PMdh.P
00a0	96 a4 69 9f 3c 51 ad 19	12 68 83 7c d5 ac 40 01	..i.<Q...h.l..@.
00b0	ca 4c 47 49 00 73 42 3d	b0 0a 10 62 ff 54 d1 55	...LGI-sB=...b.T.U
00c0	2d 04 43 4a 09 da 65 74	67 47 02 b2 52 1a 96 b8	...CJ...et gG..R...
00d0	9a 5a 10 b0 02 87 8c 24	04 a6 9b d7 c2 70 fb e2	..Z.....\$.....p...
00e0	16 60 bd 5c c7 0d dc 63	f9 7b ce 87 90 5f d7 89	...\\...c...{.....
00f0	a9 1c 20 34 d5 76 0f df	e5 60 6e eb bc e5 f3 df	...4.v...n.....

Obrázok 6 - Detaily prenosu v programe Wireshark

5.2 Test prenosu textového súboru (IPv6)

```
student@student-vm:~/ISA$ sudo ./secret -r test/lorem.txt -s ::1
Sending file to the server... (Transfer ID: 732619164)
Progress: 301856/301856 bytes sent
File was sent (total size: 301856 bytes)
```

Obrázok 7 - Odoslanie textového súboru

```
student@student-vm:~/ISA$ sudo ./secret -l
(TID: 732619164) Started receiving file: lorem.txt, size: 301856 bytes
(TID: 732619164) File received successfully
```

Obrázok 8 - Prijatie tohto textového súboru

```
student@student-vm:~/ISA$ md5sum lorem.txt
bdfa47a4d11bc74a2a290bc6b5847144  lorem.txt
student@student-vm:~/ISA$ md5sum test/lorem.txt
bdfa47a4d11bc74a2a290bc6b5847144  test/lorem.txt
```

Obrázok 9 - Porovnanie checksumu oboch súborov - sedí

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	::1	::1	ICMPv6	1486	Echo (ping) request id=0x88ec, seq=0, hop limit=64 (reply in ...)
2	0.000000000	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x88ec, seq=0, hop limit=64 (request in ...)
3	0.001728660	::1	::1	ICMPv6	1486	Echo (ping) request id=0x8673, seq=256, hop limit=64 (reply in ...)
4	0.001733030	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x8673, seq=256, hop limit=64 (request in ...)
5	0.003251242	::1	::1	ICMPv6	1486	Echo (ping) request id=0x1b87, seq=512, hop limit=64 (reply in ...)
6	0.003255792	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x1b87, seq=512, hop limit=64 (request in ...)
7	0.004803333	::1	::1	ICMPv6	1486	Echo (ping) request id=0x0fd9, seq=768, hop limit=64 (reply in ...)
8	0.004807933	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x0fd9, seq=768, hop limit=64 (request in ...)
9	0.006386165	::1	::1	ICMPv6	1486	Echo (ping) request id=0xbe04, seq=1024, hop limit=64 (reply in ...)

▶ Frame 3: 1486 bytes on wire (11888 bits), 1486 bytes captured (11888 bits) on interface lo, id 0

▶ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)

▶ Internet Protocol Version 6, Src: ::1, Dst: ::1

▶ Internet Control Message Protocol v6

0030	00 00 00 00 00 01 80 00	79 90 86 73 01 00	40 53y..s..IS
0040	41 5f 33 00 00 02 00	00 00 9c e1 aa 2b 3a 14	A.3.....+..	
0050	dd 8d d1 bc 40 02 b9 01	1f 83 4f cb 5c 5c f2 c4	...@.....0\\...	
0060	ee 5b 88 a3 e8 c1 1d 5e	50 e6 f5 87 06 bb 60 ab	...^P.....	
0070	04 29 81 b7 c8 a1 24 8b	2f 3c 33 c1 5e 9b b4 17)...\$.<3^...	
0080	dc 3e b2 40 15 7f 06 16	4f ce 59 3e f7 28 45 ca	->@.....0Y>.(E	
0090	0a 79 83 0d 4d ed f3 b6	c9 81 44 76 c0 83 7f e5	y..M.....Dy...	
00a0	c7 81 c8 b7 05 eb 60 62	58 a1 94 e8 59 9f 50 4dbX...Y.PM	
00b0	64 68 b6 50 9e a4 69 9f	3c 51 ad 19 12 68 83 7c	dh.P..i.<Q...h..	
00c0	d5 ac 40 01 ca 4c 47 49	00 73 42 3d b0 0a 10 62	..@..LGI..sB=...b	
00d0	ff 54 d1 55 2d 04 43 4a	09 da 65 74 67 47 02 b2	.T.U..CJ..etg6...	
00e0	52 1a 96 b8 9a 5a 10 b0	02 87 8c 24 04 a6 9b d7	R...Z...\$....	
00f0	c2 70 fb e2 16 60 bd 5c	c7 0d dc 63 f9 7b ce 87	p...\\...c{...	
0100	90 5f d7 89 a9 1c 20 34	d5 76 0f df e5 60 6e eb4.v...n.	

Obrázok 10 - Detaily prenosu v programe Wireshark

5.3 Test prenosu binárneho súboru (IPv4)

```
student@student-vm:~/ISA$ sudo ./secret -r test/docker.exe -s 127.0.0.1
Sending file to the server... (Transfer ID: 1679246095)
Progress: 535429608/535429608 bytes sent
File was sent (total size: 535429608 bytes)
```

Obrázok 11 - Odoslanie textového súboru

```
student@student-vm:~/ISA$ sudo ./secret -l
(TID: 1679246095) Started receiving file: docker.exe, size: 535429608 bytes
(TID: 1679246095) File received successfully
```

Obrázok 12 - Prijatie tohto textového súboru

```
student@student-vm:~/ISA$ md5sum test/docker.exe
d7781c41ca4da8b7449dc54cc136876d test/docker.exe
student@student-vm:~/ISA$ md5sum docker.exe
d7781c41ca4da8b7449dc54cc136876d docker.exe
```

Obrázok 13 - Porovnanie checksumu oboch súborov - sedí

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0x2eb1, seq=0/0, ttl=64 (reply in 2)
2	0.000009201	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0x2eb1, seq=0/0, ttl=64 (request in 1)
3	0.002016923	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0x8ef6, seq=256/1, ttl=64 (reply in 4)
4	0.002020894	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0x8ef6, seq=256/1, ttl=64 (request in 3)
5	0.004141578	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0xe40e, seq=512/2, ttl=64 (reply in 6)
6	0.004146559	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0xe40e, seq=512/2, ttl=64 (request in 5)
7	0.005942420	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0xe659, seq=768/3, ttl=64 (reply in 8)
8	0.005947301	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) reply id=0xe659, seq=768/3, ttl=64 (request in 7)
9	0.007487936	127.0.0.1	127.0.0.1	ICMP	1466	Echo (ping) request id=0xa836, seq=1024/4, ttl=64 (reply in 9)

▶ Frame 3: 1466 bytes on wire (11728 bits), 1466 bytes captured (11728 bits) on interface lo, id 0
 ▶ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 ▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 ▶ Internet Control Message Protocol

0020	00 01 08 00 0a 2f 8e f6	01 00 49 53 41 5f 33 00ISA.3
0030	00 00 02 00 00 0f 43	17 64 d6 b3 7f b9 2b 2bC.d....++
0040	e1 e1 1f b5 e2 0b 96 0f	49 e0 00 42 06 5b 28 27I..B.(['
0050	0a 62 e3 da 7d 5d 51 0f	62 4a 4e 2a 6e b4 9b a8	jb...}]Q..bJN'n...
0060	f1 be df 77 f6 d1 85 d3	cb 3e 09 4c 37 b7 ce 93	...w.....->[7...
0070	40 da 53 3a 2b 9c 8a f8	ee fa 43 63 32 f8 50 05	@.S:.....Cc2.P...
0080	9e d0 19 98 75 9e 35 b6	89 76 26 7b 26 a2 4f fe	...u-5...v8{&.0...
0090	8b 91 cf 3f c5 d8 97 2d	51 06 a2 56 3a 9b e7 59	...?.....Q..V:..Y
00a0	89 e4 62 0e 87 8f ca a6	57 d7 14 db d8 c1 e8 08	..b.....W.....
00b0	16 7a 4a 74 65 41 39 1e	1e 37 6e 89 55 44 4a f1	..zDteA9...7n-UDJ...
00c0	8c 24 24 a2 08 3e 2f ad	bc 08 9e 2e d5 d1 90 36	..\$\$..>/.....0...
00d0	1a 8d 1f f7 c8 40 c8 72	b9 af f9 9a 35 1b 0f 34@.r....5..d...
00e0	5e 6b 5c 11 54 6c e2 a3	61 b0 2e 60 c1 ca 9f bb	AK\..T1...a.....
00f0	25 11 49 03 a4 36 e6 db	f3 78 9b f9 a5 be e0 a9	%I..6...x.....
0100	fa d4 42 5b f2 2a 3e fe	af fd c0 07 fd 69 b4 4d	..B[.*>.....i.M...
0110	0d 7e 79 0a 66 00 06 32	fa c7 48 d4 d8 b2 50 43	..y-f'.2...H...PC...
0120	61 a1 6e bc 70 a8 55 0a	6f 33 48 d4 d8 b2 50 43	a-n-p-U..o3H...PC...
0130	61 a1 6e bc 70 a8 55 0a	6f 33 09 37 45 12 ca 91	a-n-p-U..o37E...

Obrázok 14 - Detaily prenosu v programe Wireshark

5.4 Test prenosu binárneho súboru (IPv6)

```
student@student-vm:~/ISA$ sudo ./secret -r test/docker.exe -s ::1
Sending file to the server... (Transfer ID: 1660392032)
Progress: 535429608/535429608 bytes sent
File was sent (total size: 535429608 bytes)
```

Obrázok 15 - Odoslanie textového súboru

```
student@student-vm:~/ISA$ sudo ./secret -l
(TID: 1660392032) Started receiving file: docker.exe, size: 535429608 bytes
(TID: 1660392032) File received successfully
```

Obrázok 16 - Prijatie tohto textového súboru

```
student@student-vm:~/ISA$ md5sum test/docker.exe
d7781c41ca4da8b7449dc54cc136876d test/docker.exe
student@student-vm:~/ISA$ md5sum docker.exe
d7781c41ca4da8b7449dc54cc136876d docker.exe
```

Obrázok 17 - Porovnanie checksumu oboch súborov - sedí

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	::1	::1	ICMPv6	1486	Echo (ping) request id=0x9b1e, seq=0, hop limit=64 (reply in ...)
2	0.000000081	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x9b1e, seq=0, hop limit=64 (request in ...)
3	0.004708393	::1	::1	ICMPv6	1486	Echo (ping) request id=0x58b8, seq=256, hop limit=64 (reply in ...)
4	0.004712814	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x58b8, seq=256, hop limit=64 (request in ...)
5	0.006232261	::1	::1	ICMPv6	1486	Echo (ping) request id=0x4eaa, seq=512, hop limit=64 (reply in ...)
6	0.006236372	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x4eaa, seq=512, hop limit=64 (request in ...)
7	0.007789664	::1	::1	ICMPv6	1486	Echo (ping) request id=0x1042, seq=768, hop limit=64 (reply in ...)
8	0.007793934	::1	::1	ICMPv6	1486	Echo (ping) reply id=0x1042, seq=768, hop limit=64 (request in ...)
9	0.008516218	::1	::1	ICMPv6	1486	Echo (ping) request id=0xbd6a, seq=1024, hop limit=64 (reply in ...)

▶ Frame 3: 1486 bytes on wire (11888 bits), 1486 bytes captured (11888 bits) on interface lo, id 0
 ▶ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 ▶ Internet Protocol Version 6, Src: ::1, Dst: ::1
 ▶ Internet Control Message Protocol v6

0000	00 00 00 00 00 00 00 00	00 00 00 00 86 dd 60 0f
0010	bb 1c 05 98 3a 40 00 00	00 00 00 00 00 00 00 00:@.....
0020	00 00 00 00 00 01 00 00	00 00 00 00 00 00 00 000.....
0030	00 00 00 00 00 01 00 00	91 4a 58 b8 01 00 49 53JX...IS
0040	41 5f 33 00 00 00 02 00	00 00 60 92 f7 62 d6 b3	A_3.....b...
0050	7f b9 2b 2b e1 e1 1f b5	e2 8b 96 9f 49 e0 0d 42	..+.....I..B
0060	06 5b 28 27 6a 62 e3 da	7d 5d 51 0f 62 4a 4e 2a	..[('jb..}]Q.bJN*
0070	6e b4 9b a8 f1 be df 77	f6 d1 85 d3 cb 3e 09 4c	n.....w.....>.L
0080	37 b7 ce 93 40 da 53 3a	2b 9c 8a f8 ee fa 43 63	7...@.S: +.....Cc
0090	32 f8 50 05 9e d0 19 08	75 9e 35 b6 89 76 26 7b	2.P.....u.S.v&
00a0	26 a2 4f fe 8b 01 cf 3f	c5 d8 97 2d 51 06 a2 56	&.O...?....Q.V
00b0	3a 9b e7 59 89 e4 62 0e	87 8f ca a6 57 d7 14 db	..Y..b.....W...
00c0	d8 c1 e9 08 16 7a 44 74	65 41 39 1e 1e 37 6e 89zDt eA9..7n..
00d0	55 44 a4 f1 8c 24 24 a2	08 3e 2f ad bc 08 9e 2e	UDJ..\$\$.>/.....
00e0	d5 d1 90 30 1a 8d 1f f7	c8 40 c8 72 b9 af f9 9a	...0....@r.....
00f0	35 1b 0f 34 5e 6b 5c 11	54 6c e2 a3 61 b0 2e 60	5..4%k..Tl..a..
0100	c1 ca 9f bb 25 11 49 03	a4 36 e6 db f3 78 9b f9%I..6...x..
0110	a5 be e9 a9 fa 44 42 5b	f2 2a 3e fe af fd c0 07B[...>....
0120	fd 69 b4 4d 0d 7e 79 0a	66 60 06 32 fa c7 48 d4	..1.M..y f..2..H
0130	d8 b2 50 43 61 a1 6e bc	70 a8 55 0a 6f 33 48 d4	..PCa.n p.U.o3H
0140	d8 b2 50 43 61 a1 6e bc	70 a8 55 0a 6f 33 09 37	..PCa.n p.U.o37
0150	45 12 ca 91 4d d1 8d 8f	64 bf ee 4d 86 e0 3e ab	E...MA..d..M..>
0160	d6 ba 8e 85 e4 9c a7 54	2f 9a 43 64 cf 28 f6 97T /..Cd..>

Obrázok 18 - Detaily prenosu v programe Wireshark

5.4 Valgrind

```
student@student-vm:~/ISA$ sudo valgrind ./secret -r test/lorem.txt -s ::1
==7634== Memcheck, a memory error detector
==7634== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==7634== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==7634== Command: ./secret -r test/lorem.txt -s ::1
==7634==
Sending file to the server... (Transfer ID: 1734988567)
Progress: 301856/301856 bytes sent
File was sent (total size: 301856 bytes)
==7634==
==7634== HEAP SUMMARY:
==7634==   in use at exit: 0 bytes in 0 blocks
==7634==   total heap usage: 5 allocs, 5 frees, 78,372 bytes allocated
==7634==
==7634== All heap blocks were freed -- no leaks are possible
==7634==
==7634== For lists of detected and suppressed errors, rerun with: -s
==7634== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

6. Zdroje

Pri práci na tomto projekte som čerpal z nasledovných zdrojov:

1. Wireshark User's Guide. *Wireshark* [online]. [cit. 2021-11-12]. Dostupné z: https://www.wireshark.org/docs/wsug_html_chunked
2. Ping, ping6 - send ICMP ECHO_REQUEST to network hosts. Linux man page [online]. [cit. 2021-11-12]. Dostupné z: <https://linux.die.net/man/8/ping>
3. GitHub - the-tcpdump-group/libpcap. GitHub [online]. [cit. 2021-11-12]. Dostupné z: <https://github.com/the-tcpdump-group/libpcap>
4. Network interface. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-04-21]. Dostupné z: https://en.wikipedia.org/wiki/Network_interface
5. Internet Control Message Protocol. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-04-21]. Dostupné z: https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol
6. Ping in C - GeeksforGeeks. GeeksforGeeks | A computer science portal for geeks [online]. [cit. 2021-11-12]. Dostupné z: <https://www.geeksforgeeks.org/ping-in-c/>
7. Beej's Guide to Network Programming. Beej's Web Page [online]. Copyright © November 20, 2020 [cit. 2021-11-12]. Dostupné z: <https://beej.us/guide/bgnet/html/#client-server-background>
8. Using libpcap in C | DevDungeon. DevDungeon | Virtual Hackerspace [online]. [cit. 2021-11-12]. Dostupné z: <https://www.devdungeon.com/content/using-libpcap-c>