Fakulta informačních technologií
Vysoké učení technické v Brně

Počítačové komunikácie a siete – 2. projekt
Variant ZETA: Sniffer paketov

2021 Jakub Bartko (xbartk07)

Obsah

1 Úvod

Táto dokumentácia má za účel poskytnúť základný prehľad o riešenom probléme, implementácií jeho riešenia, využívaných knižniciach a o už existujúcich prostriedkoch s obdobným využitím.

2 Základná problematika

Centrálnym prvkom projektu, ku ktorému táto dokumentácia náleží, je **analyzátor paketov** (angl. aj *packet sniffer*). Takýto program alebo zariadenie skúži na sledovanie, zachytávanie a zaznamenávanie sieťovej prevádzky na určitej digitálnej sieti (preložené z [3]). V tomto projekte je využívaný na nájdenie aktívnych sieťových rozhraní a zachytávanie, filtrovanie a základnú analýzu paketov na konkrétnom rozhraní.

Sieťové rozhranie je prepojenie medzi počítačom a súkromnou alebo verejnou sieťou (preložené z [1]). **Paket** je malý blok dát prenášaný v sieti, ktorý sa skladá z hlavičky a samotných prenášaných dát. Táto hlavička obsahuje informácie, ako napr. zdrojovú a cieľovú adresu paketu, jeho veľkosť a použitý protokol. **Protokol** definuje formát dát paketu, t. j. stanovuje, na ktorých bajtoch sa nachádzajú určité informácie. Každá z vrstiev modelu OSI (abstraktná štruktúra komunikačných a počítačových sieťových protokolov [2]) má vlastný balík protokolov. Paket môže obsahovať viacero hlavičiek protokolov na rôznych vrstvách v súlade s tým, pod ktorú vrstvu spadá. My, na základe zadania, začneme od ethernetového rámca, z ktorého hlavičky zistíme, ktorý protokol sieťovej vrstvy bol použitý, t. j. aký formát ďalšej hlavičky máme očakávať. Z týchto hlavičiek potom zistíme, či na výstupe máme zahrnúť IPv4, IPv6 alebo MAC adresu, prípadne port, rovnako ako to, na ktorých bajtoch tieto informácie nájdeme.

Na analýzu paketov budeme využívať **promiskuitný režim** sieťovej karty, čo znamená, že budeme zachytávať aj sieťovú komunikáciu, ktorá nie je priamo určená našemu zariadeniu [4].

3 Štruktúra kódu a implementácia

Funkcionalitu programu `ipk-sniffer` je v základe možné rozdeliť na spracovanie argumentov príkazového riadka, výpis dostupných sieťových rozhraní a zachytávanie a analýzu paketov na zvolenom rozhraní. Na základné operácie s rozhraniami a paketmi sa využíva knižnica `pcap`[1].

Na základe argumentov príkazového riadka, spracovaných pomocou štandardnej knižnice `getopt`[2], sa zvolí činnosť analyzátora – výpis rozhraní alebo analýza paketov, a zostaví sa filter na ich zachytávanie. Tieto argumenty, spracované funkciou `Options::get_opts`, sú uložené v objekte triedy `Options`. Zoznam názvov dostupných rozhraní sa získa pomocou funkcie `pcap_findalldevs`. Tento zoznam sa následne iteruje, názvy rozhraní sa vypíšu na štandardný výstup a zoznam sa uvoľní.

V prípade analýzy paketov sa najskôr získa „rukoväť“ (ďalej angl. *handle*) v promiskuitnom móde na zachytávanie paketov na špecifikovanom sieťovom rozhraní a vygeneruje, skompiluje a aplikuje sa na ňu filter na základe údajov objektu triedy `Options`. Následne sa zachytí a spracuje zvolený počet paketov.

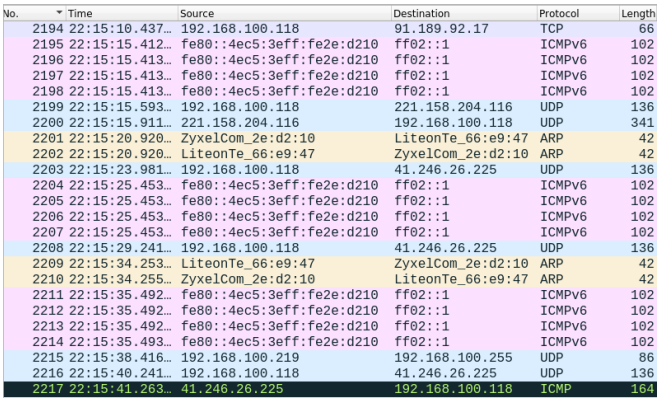
Z hlavičky zachyteného paketu sa získa časová značka a jeho veľkosť v bajtoch. Tento paket sa interpretuje ako ethernetový rámec a na základe hodnoty atribútu `EtherType`[3] sa zistí, či obsahuje IPv4 datagram alebo rámec IPv6 či ARP a získa sa príslušná hlavička. V prípade IPv4 alebo IPv6 sa získa a dekóduje zdrojová a cieľová IP adresa paketu a pri protokoloch TCP a UDP aj príslušná dvojica portov; v prípade ARP sa z hlavičky ethernetového paketu zistí dvojica MAC adries. Údaje popísané v tomto odstavci sa vypíšu na štandardný výstup ako hlavička výpisu zachyteného paketu.

Samotné dáta zachyteného paketu sú po 16 bajtoch na riadok vypísané ako hexadecimálne hodnoty spolu so svojou ASCII reprezentáciou. Prefix každého riadka predstavuje hexadecimálny ofset jeho prvého bajtu.

4 Testovanie funkcionality

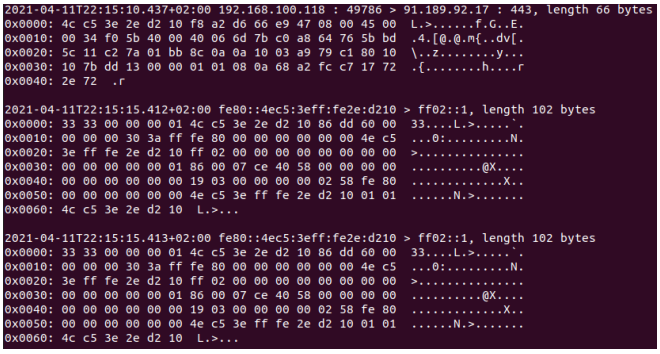
Priebežné testovanie implementovaného programu bolo vykonávané za použitia open-source analyzátora paketov **Wireshark 3.2.3** [5]. Finálne testovanie pozostávalo z niekoľkých testovacích prípadov, ktoré zahŕňali spustenie oboch nástrojov, lokalizáciu zaujímavého úseku paketov, vyexportovanie skúmaných dát a ich porovnanie.

[1] https://www.tcpdump.org/manpages/pcap.3pcap.html
[2] https://man7.org/linux/man-pages/man3/getopt.3.html
[3] https://en.wikipedia.org/wiki/Ethernet_frame#Types



No.	Time	Source	Destination	Protocol	Length
2194	22:15:10.437…	192.168.100.118	91.189.92.17	TCP	66
2195	22:15:15.412…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2196	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2197	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2198	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2199	22:15:15.593…	192.168.100.118	221.158.204.116	UDP	136
2200	22:15:15.911…	221.158.204.116	192.168.100.118	UDP	341
2201	22:15:20.920…	ZyxelCom_2e:d2:10	LiteonTe_66:e9:47	ARP	42
2202	22:15:20.920…	LiteonTe_66:e9:47	ZyxelCom_2e:d2:10	ARP	42
2203	22:15:23.981…	192.168.100.118	41.246.26.225	UDP	136
2204	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2205	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2206	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2207	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2208	22:15:29.241…	192.168.100.118	41.246.26.225	UDP	136
2209	22:15:34.253…	LiteonTe_66:e9:47	ZyxelCom_2e:d2:10	ARP	42
2210	22:15:34.255…	ZyxelCom_2e:d2:10	LiteonTe_66:e9:47	ARP	42
2211	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2212	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2213	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2214	22:15:35.493…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2215	22:15:38.416…	192.168.100.219	192.168.100.255	UDP	86
2216	22:15:40.241…	192.168.100.118	41.246.26.225	UDP	136
2217	22:15:41.263…	41.246.26.225	192.168.100.118	ICMP	164

Obr. 1: Skúmaný úsek paketov – *Wireshark*



```
2021-04-11T22:15:10.437+02:00 192.168.100.118 : 49786 > 91.189.92.17 : 443, length 66 bytes
0x0000: 4c c5 3e 2e d2 10 f8 a2 d6 66 e9 47 08 00 45 00  L.>......f.G..E.
0x0010: 00 34 f0 5b 40 00 40 06 6d 7b c0 a8 64 76 5b bd  .4.[@.@.m{..dv[.
0x0020: 5c 11 c2 7a 01 bb 8c 0a 0a 10 03 a9 79 c1 80 10  \..z........y...
0x0030: 10 7b dd 13 00 00 01 01 08 0a 68 a2 fc c7 17 72  .{........h....r
0x0040: 2e 72                                            .r

2021-04-11T22:15:15.412+02:00 fe80::4ec5:3eff:fe2e:d210 > ff02::1, length 102 bytes
0x0000: 33 33 00 00 00 01 4c c5 3e 2e d2 10 86 dd 60 00  33....L.>.....`.
0x0010: 00 00 00 30 3a ff fe 80 00 00 00 00 00 00 4e c5  ...0:.........N.
0x0020: 3e ff fe 2e d2 10 ff 02 00 00 00 00 00 00 00 00  >...............
0x0030: 00 00 00 00 00 01 86 00 07 ce 40 58 00 00 00 00  ..........@X....
0x0040: 00 00 00 00 00 00 19 03 00 00 00 00 02 58 fe 80  .............X..
0x0050: 00 00 00 00 00 00 4e c5 3e ff fe 2e d2 10 01 01  ......N.>.......
0x0060: 4c c5 3e 2e d2 10  L.>...

2021-04-11T22:15:15.413+02:00 fe80::4ec5:3eff:fe2e:d210 > ff02::1, length 102 bytes
0x0000: 33 33 00 00 00 01 4c c5 3e 2e d2 10 86 dd 60 00  33....L.>.....`.
0x0010: 00 00 00 30 3a ff fe 80 00 00 00 00 00 00 4e c5  ...0:.........N.
0x0020: 3e ff fe 2e d2 10 ff 02 00 00 00 00 00 00 00 00  >...............
0x0030: 00 00 00 00 00 01 86 00 07 ce 40 58 00 00 00 00  ..........@X....
0x0040: 00 00 00 00 00 00 19 03 00 00 00 00 02 58 fe 80  .............X..
0x0050: 00 00 00 00 00 00 4e c5 3e ff fe 2e d2 10 01 01  ......N.>.......
0x0060: 4c c5 3e 2e d2 10  L.>...
```

Obr. 2: Časť skúmaného úseku paketov – *ipk-sniffer*

4.1 Správnosť zachytávania paketov

Prvý testovací prípad skúma, či implementovaný analyzátor zachytáva rovnaké pakety. Zvolíme si prvé 4 pakety skúmaného úseku a porovnáme ich časovú značku a veľkosť. *WS* – Wireshark, *IPK* – ipk-sniffer.

WS čas	IPK čas	WS veľkosť	IPK veľkosť
22:15:10.437	22:15:15.437+02:00	66	66
22:15:15.412	22:15:15.412+02:00	102	102
22:15:15.413	22:15:15.413+02:00	102	102
22:15:15.413	22:15:15.413+02:00	102	102

Tabuľka 1: Porovnanie zachytených paketov

Zdá sa, že obidva analyzátory zachytávajú rovnaké pakety. Ďalej teda budeme testovať ich obsah.

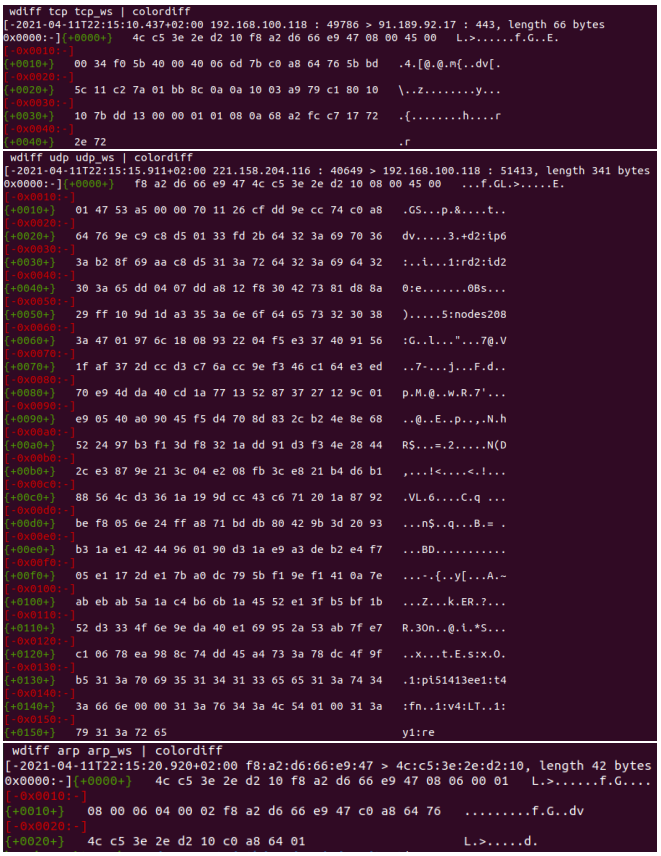
4.2 Správnosť obsahu paketov

V rámci tohoto testovacieho prípadu si zvolíme niekoľko paketov zo skúmaného úseku s rôznymi protokolmi a porovnáme ich obsahy v hexadecimálnej aj ASCII reprezentácii.

No.	Time	Source	Destination	Protocol	Length
2194	22:15:10.437…	192.168.100.118	91.189.92.17	TCP	66
2195	22:15:15.412…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2196	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2197	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2198	22:15:15.413…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2199	22:15:15.593…	192.168.100.118	221.158.204.116	UDP	136
2200	22:15:15.911…	221.158.204.116	192.168.100.118	UDP	341
2201	22:15:20.920…	ZyxelCom_2e:d2:10	LiteonTe_66:e9:47	ARP	42
2202	22:15:20.920…	LiteonTe_66:e9:47	ZyxelCom_2e:d2:10	ARP	42
2203	22:15:23.981…	192.168.100.118	41.246.26.225	UDP	136
2204	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2205	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2206	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2207	22:15:25.453…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2208	22:15:29.241…	192.168.100.118	41.246.26.225	UDP	136
2209	22:15:34.253…	LiteonTe_66:e9:47	ZyxelCom_2e:d2:10	ARP	42
2210	22:15:34.255…	ZyxelCom_2e:d2:10	LiteonTe_66:e9:47	ARP	42
2211	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2212	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2213	22:15:35.492…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2214	22:15:35.493…	fe80::4ec5:3eff:fe2e:d210	ff02::1	ICMPv6	102
2215	22:15:38.416…	192.168.100.219	192.168.100.255	UDP	86
2216	22:15:40.241…	192.168.100.118	41.246.26.225	UDP	136
2217	22:15:41.263…	41.246.26.225	192.168.100.118	ICMP	164

Obr. 3: Zvolené pakety (tmavo modré) – *Wireshark*

Porovania boli vykonané príkazom[4] `wdiff {protocol} {protocol}_ws | colordiff`. Z výsledkov tohoto testovacieho prípadu na obr. 4 a 5 vidíme, že výstup programu *ipk-sniffer* sa od dát zachytených programom *Wireshark* líši jedine vo výpise hlavičky a vo formáte ofsetu na začiatku riadku.



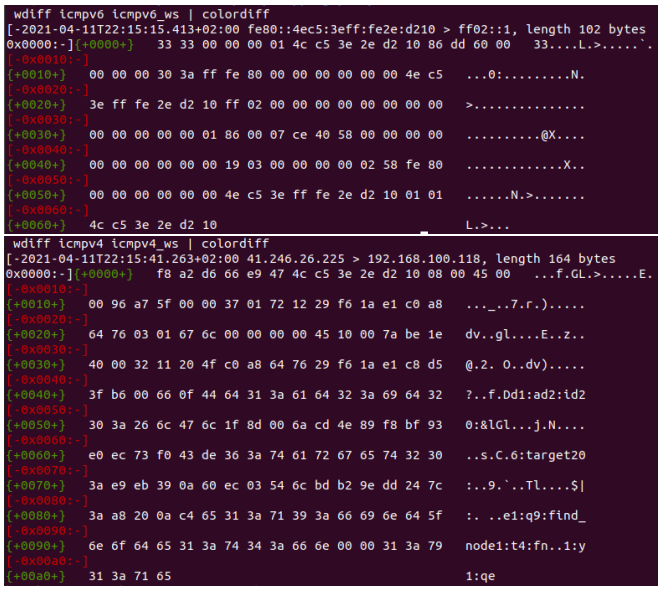
```
wdiff tcp tcp_ws | colordiff
[-2021-04-11T22:15:10.437+02:00 192.168.100.118 : 49786 > 91.189.92.17 : 443, length 66 bytes
0x0000:-][+0000+]   4c c5 3e 2e d2 10 f8 a2 d6 66 e9 47 08 00 45 00   L.>......f.G..E.
[-0x0010:-]
{+0010+}   00 34 f0 5b 40 00 40 06 6d 7b c0 a8 64 76 5b bd   .4.[@.@.m{..dv[.
[-0x0020:-]
{+0020+}   5c 11 c2 7a 01 bb 8c 0a 0a 10 03 a9 79 c1 80 10   \..z........y...
[-0x0030:-]
{+0030+}   10 7b dd 13 00 00 01 01 08 0a 68 a2 fc c7 17 72   .{........h....r
[-0x0040:-]
{+0040+}   2e 72                                             .r

wdiff udp udp_ws | colordiff
[-2021-04-11T22:15:15.911+02:00 221.158.204.116 : 40649 > 192.168.100.118 : 51413, length 341 bytes
0x0000:-][+0000+]   f8 a2 d6 66 e9 47 4c c5 3e 2e d2 10 08 00 45 00   ...f.GL.>.....E.
[-0x0010:-]
{+0010+}   01 47 53 a5 00 00 70 11 26 cf dd 9e cc 74 c0 a8   .GS...p.&....t..
[-0x0020:-]
{+0020+}   64 76 9e c9 c8 d5 01 33 fd 2b 64 32 3a 69 70 36   dv.....3.+d2:ip6
[-0x0030:-]
{+0030+}   3a b2 8f 69 aa c8 d5 31 3a 72 64 32 3a 69 64 32   :..i...1:rd2:id2
[-0x0040:-]
{+0040+}   30 3a 65 dd 04 07 dd a8 12 f8 30 42 73 81 d8 8a   0:e.......0Bs...
[-0x0050:-]
{+0050+}   29 ff 10 9d 1d a3 35 3a 6e 6f 64 65 73 32 30 38   ).....5:nodes208
[-0x0060:-]
{+0060+}   3a 47 01 97 6c 18 08 93 22 04 f5 e3 37 40 91 56   :G..l..."...7@.V
[-0x0070:-]
{+0070+}   1f af 37 2d cc d3 c7 6a cc 9e f3 46 c1 64 e3 ed   ..7-...j...F.d..
[-0x0080:-]
{+0080+}   70 e9 4d da 40 cd 1a 77 13 52 87 37 27 12 9c 01   p.M.@..w.R.7'...
[-0x0090:-]
{+0090+}   e9 05 40 a0 90 45 f5 d4 70 8d 83 2c b2 4e 8e 68   ..@..E..p..,.N.h
[-0x00a0:-]
{+00a0+}   52 24 97 b3 f1 3d f8 32 1a dd 91 d3 f3 4e 28 44   R$...=.2.....N(D
[-0x00b0:-]
{+00b0+}   2c e3 87 9e 21 3c 04 e2 08 fb 3c e8 21 b4 d6 b1   ,...!<....<.!...
[-0x00c0:-]
{+00c0+}   88 56 4c d3 36 1a 19 9d cc 43 c6 71 20 1a 87 92   .VL.6....C.q ...
[-0x00d0:-]
{+00d0+}   be f8 05 6e 24 ff a8 71 bd db 80 42 9b 3d 20 93   ...n$..q...B.= .
[-0x00e0:-]
{+00e0+}   b3 1a e1 42 44 96 01 90 d3 1a e9 a3 de b2 e4 f7   ...BD...........
[-0x00f0:-]
{+00f0+}   05 e1 17 2d e1 7b a0 dc 79 5b f1 9e f1 41 0a 7e   ...-.{..y[...A.~
[-0x0100:-]
{+0100+}   ab eb ab 5a 1a c4 b6 6b 1a 45 52 e1 3f b5 bf 1b   ...Z...k.ER.?...
[-0x0110:-]
{+0110+}   52 d3 33 4f 6e 9e da 40 e1 69 95 2a 53 ab 7f e7   R.3On..@.i.*S...
[-0x0120:-]
{+0120+}   c1 06 78 ea 98 8c 74 dd 45 a4 73 3a 78 dc 4f 9f   ..x...t.E.s:x.O.
[-0x0130:-]
{+0130+}   b5 31 3a 70 69 35 31 34 31 33 65 65 31 3a 74 34   .1:pi51413ee1:t4
[-0x0140:-]
{+0140+}   3a 66 6e 00 00 31 3a 76 34 3a 4c 54 01 00 31 3a   :fn..1:v4:LT..1:
[-0x0150:-]
{+0150+}   79 31 3a 72 65                                    y1:re

wdiff arp arp_ws | colordiff
[-2021-04-11T22:15:20.920+02:00 f8:a2:d6:66:e9:47 > 4c:c5:3e:2e:d2:10, length 42 bytes
0x0000:-][+0000+]   4c c5 3e 2e d2 10 f8 a2 d6 66 e9 47 08 06 00 01   L.>......f.G....
[-0x0010:-]
{+0010+}   08 00 06 04 00 02 f8 a2 d6 66 e9 47 c0 a8 64 76   .........f.G..dv
[-0x0020:-]
{+0020+}   4c c5 3e 2e d2 10 c0 a8 64 01                     L.>.....d.
```

Obr. 4: Výsledky porovnávania dát paketov – 1. časť

[4] `wdiff` – GNU nástroj na porovnávanie súborov po slovách; `colordiff` – https://linux.die.net/man/1/colordiff



Obr. 5: Výsledky porovnávania dát paketov – 2. časť

Literatúra

[1] What Is a Network Interface? – The Java™ Tutorials. [online], 2021. URL `https://docs.oracle.com/javase/tutorial/networking/nifs/definition.html`.

[2] Model OSI – Wikipedia. [online], 2021. URL `https://sk.wikipedia.org/wiki/Model_OSI`.

[3] What is a Packet Analyzer? – Definition from Techopedia. [online], 2021. URL `https://www.techopedia.com/definition/25323/packet-analyzer`.

[4] Promiskuitní režim – Wikipedia. [online], 2021. URL `https://cs.wikipedia.org/wiki/Promiskuitn%C3%AD_re%C5%BEim`.

[5] Wireshark. [online], 2021. URL `https://www.wireshark.org/`.