

Dokumentácia k 2. projektu z IPK

Varianta [ZETA]

Sniffer paketov
xmorav41

Contents

Problematika.....	3
TCP	3
UDP	3
Sniffovanie	3
Návrh Aplikácie	3
Popis implementácie.....	4
Funkcia Main():	4
Parsovanie argumentov:	4
Funkcia gotpacket():.....	4
Funkcia print_(udp tcp)_packet():	4
Testovanie	4
Bibliografia	7

Problematika

TCP

Protokol riadenia prenosu (Transmission Control Protocol) je jeden z internetových protokolov vďaka ktorému programy na počítačoch v sieti môžu naviazovať spojenia na posielanie dát. Protokol zaisťuje odoslanie a prijatie dát zo strán v rovnakom poradí a bez chýbajúcich častí.

Rozloženie TCP:

Bity	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	zdrojový port																cieľový port															
32	číslo sekvencie																															
64	potvrdený bajt																															
96	offset dát				rezervované				príznamy								okienko															
128	kontrolný súčet																Urgent Pointer															
160	voľby (voliteľné)																															
192	voľby (pokračovanie)																								výplň (do 32)							
224	dáta																															

UDP

Používateľský datagramový protokol (User Datagram Protocol) je protokol prenášajúci datagramy medzi počítačovými sieťami, ale nezaručuje na rozdiel od TCP, že sa paket nestratí, nezmení ich poradie alebo duplikovanie paketov. Výhoda spočíva v rýchlejšom a efektívnejšom prenose, napríklad na množstvo malých požiadaviek klientov na server.

Pri hlavičke sú polia "zdrojový port" a "kontrolný súčet" voliteľné, v praxi sa kontrolný súčet používa skoro vždy.

Rozloženie UDP:

+	bity 0 - 15	16 - 31
0	zdrojový port	cieľový port
32	dĺžka	kontrolný súčet
64	data	

Sniffovanie

Sniffovanie paketov je činnosť zachytávania niektorých alebo všetkých paketov ktoré prechádzajú cez počítače v lokálnej sieti, používané napr. pri diagnostikovaní siete. Paketom sú myslené prenášané dáta rozdelené na malé časti, ktoré sa u príjemcu poskladajú späť do originálnej podoby.

Návrh Aplikácie

Program bol napísaný v jazyku C ako najviac oboznámený jazyk z možností, a tiež pre jeho jednoduchosť. Celá implementácia sa zakladá na využívaní funkcií z knižnice pcap.h, ktorá začne počúvanie v promiskuitnom móde, aplikuje prípadné

filtre a prejde prijatými paketmi. Pri zachytení paketu rozhodne, či sa jedná o IPv4 alebo IPv6 a následne pozrie protokol paketu (TCP/UDP), ktorého informácie vypíše na štandardný výstup. Výstupné informácie sa skladajú z času obdržania paketu, odosielateľa, zdrojového portu, príjemcu, cieľového portu a nakoniec offset, hexadecimálne a vytlačené dáta paketu.

Popis implementácie

Funkcia Main():

- Spracovanie argumentov
- Inicializácia, filtrovanie a zachytenie paketov na danom rozhraní
- Spracovanie paketov

Parsovanie argumentov:

- Nápoveda: -h alebo --help
- Aplikácia očakáva na prvom mieste argument -i [X] kde X je názov rozhrania.
- Volanie bez rozhrania alebo akýchkoľvek argumentov vypíše zoznam dostupných rozhraní.
- Pri nesprávnom volaní vypíše chybu a nápovedu.

Funkcia gotpacket():

- Napíše čas prijatia paketu
- Zostaví ip header
- Rozhodne o IPv4 alebo IPv6
- Nastaví socket, vytlačí informácie o pakete a zavolá funkciu `print_(udp|tcp)_packet()`

Funkcia print_(udp|tcp)_packet():

- Vytlačí hlavičku a dáta paketu

Testovanie

Výsledky testu boli porovnané s výstupom programu WireShark.

sudo ./ipk-sniffer -i wlp2s0 -p 80 -t -n 10

```
kringe@Kringe-Aspire:~/Documents/IPK/Projekt2/testovane$ sudo ./ipk-sniffer -i wlp2s0 -p
22:40:01.000202 Kringe-Aspire : 47286 > 157.140.2.32 : 80
0x0000: 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E.
0x0010: 00 3c ab c9 40 00 40 06 2e 34 c0 a8 00 6a 9d 8c .<...@...4...j..
0x0020: 02 20 b8 b6 00 50 31 f6 14 d0 00 00 00 00 a0 02 . ...P1.....
0x0030: fa f0 60 ed 00 00 02 04 05 b4 04 02 08 0a a5 fa ..`.....
0x0040: e5 1d 00 00 00 00 01 03 03 07 .....

22:40:01.000233 157.140.2.32 : 80 > Kringe-Aspire : 47286
0x0000: 3c 95 09 96 0a f7 10 fe ed a5 01 56 08 00 45 00 <.....V..E.
0x0010: 00 3c 00 00 40 00 2a 06 ef fd 9d 8c 02 20 c0 a8 .<...@.*.....
0x0020: 00 6a 00 50 b8 b6 8c bf b2 b4 31 f6 14 d1 a0 12 .j.P.....1....
0x0030: 71 20 e5 d1 00 00 02 04 05 64 04 02 08 0a 74 11 q .....d....t.
0x0040: 52 23 a5 fa e5 1d 01 03 03 07 R#.....

22:40:01.000253 Kringe-Aspire : 47286 > 157.140.2.32 : 80
0x0000: 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E.
0x0010: 00 34 ab ca 40 00 40 06 2e 3b c0 a8 00 6a 9d 8c .4...@...;...j..
0x0020: 02 20 b8 b6 00 50 31 f6 14 d1 8c bf b2 b5 80 10 . ...P1.....
0x0030: 01 f6 60 e5 00 00 01 01 08 0a a5 fa e5 55 74 11 ..`.....Ut.
0x0040: 52 23 .....R#

4 0.307564009 192.168.0.1 239.255.255.250 SSDP 370 NOTIFY * HTTP/1.1
5 0.409386003 192.168.0.1 239.255.255.250 SSDP 315 NOTIFY * HTTP/1.1
6 0.489393461 192.168.0.106 157.140.2.32 TCP 74 47286 -> 80 [SYN] Seq=0 Win=64240 Len=
7 0.512099300 192.168.0.1 239.255.255.250 SSDP 354 NOTIFY * HTTP/1.1
8 0.536349828 157.140.2.32 192.168.0.106 TCP 74 80 -> 47286 [SYN, ACK] Seq=0 Ack=1 Win=
9 0.536404998 192.168.0.106 157.140.2.32 TCP 66 47286 -> 80 [ACK] Seq=1 Ack=1 Win=64254

me 6: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
ernet II, Src: LiteonTe_96:0a:f7 (3c:95:09:96:0a:f7), Dst: Tp-LinkT_a5:01:56 (10:fe:ed:a5:01:56)
ernet Protocol Version 4, Src: 192.168.0.106, Dst: 157.140.2.32
mission Control Protocol, Src Port: 47286, Dst Port: 80, Seq: 0, Len: 0
```

```
10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V< .....E.
00 3c ab c9 40 00 40 06 2e 34 c0 a8 00 6a 9d 8c .<...@...4...j..
02 20 b8 b6 00 50 31 f6 14 d0 00 00 00 00 a0 02 . ...P1.....
fa f0 60 ed 00 00 02 04 05 b4 04 02 08 0a a5 fa ..`.....
e5 1d 00 00 00 00 01 03 03 07 .....
00 00 00 00
```

sudo ./ipk-sniffer -i wlp2s0 -p 22 -n 10

```
kringe@Kringe-Aspire: ~/Documents/IPK/Proj...
File Edit View Search Terminal Help

kringe@Kringe-Aspire:~/Documents/IPK/Projekt2/testovanie$ sudo ./ipk-sniffer -i wlp2s0 -p 22 -n 10
22:47:13.000586 Kringe-Aspire : 40765 > merlin.fit.vutbr.cz : 22
0x0000: 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E.
0x0010: 00 3c c3 da 40 00 40 06 71 d6 c0 a8 00 6a 93 e5 .<..@.q....j..
0x0020: b0 13 9f 3d 00 16 77 0d 03 af 00 00 00 00 a0 02 ...=.W.....
0x0030: fa f0 05 3a 00 00 02 04 05 b4 04 02 08 0a 42 b7 ...:.....B.
0x0040: dd 2c 00 00 00 00 01 03 03 07 .....

Atom 3.000659 merlin.fit.vutbr.cz : 22 > Kringe-Aspire : 40765
0x0000: 3c 95 09 96 0a f7 10 fe ed a5 01 56 08 00 45 00 <.....V..E.
0x0010: 00 3c 00 00 40 00 35 06 40 b1 93 e5 b0 13 c0 a8 .<..@.5.@.....
0x0020: 00 6a 00 16 9f 3d a7 f0 80 d9 77 0d 03 b0 a0 12 .j...=.W.....
0x0030: 7c 70 69 a2 00 00 02 04 05 b4 04 02 08 0a 25 58 |pl.....%X
0x0040: d4 b8 42 b7 dd 2c 01 03 03 09 ..B.,....

22:47:13.000700 Kringe-Aspire : 40765 > merlin.fit.vutbr.cz : 22
0x0000: 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E.
0x0010: 00 34 c3 db 40 00 40 06 71 dd c0 a8 00 6a 93 e5 .4..@.q....j..
0x0020: b0 13 9f 3d 00 16 77 0d 03 b0 a7 f0 80 da 80 10 ...=.W.....
0x0030: 01 f6 05 32 00 00 01 01 08 0a 42 b7 dd 46 25 58 ...2.....B..F%X
0x0040: d4 b8 .....

22:47:13.000715 Kringe-Aspire : 40765 > merlin.fit.vutbr.cz : 22
0x0000: 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E.
0x0010: 00 50 c3 dc 40 00 40 06 71 c0 c0 a8 00 6a 93 e5 .P..@.q....j..
0x0020: b0 13 9f 3d 00 16 77 0d 03 b0 a7 f0 80 da 80 18 ...=.W.....
0x0030: 01 f6 05 4e 00 00 01 01 08 0a 42 b7 dd 46 25 58 ...N.....B..F%X
0x0040: d4 b8 53 53 48 2d 32 2e 30 2d 50 75 54 54 59 5f ..SSH-2.0-PuTTY_
0x0050: 52 65 6c 65 61 73 65 5f 30 2e 37 30 0d 0a Release 0.70..

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port == 22

Packet list Narrow & Wide Case sensitive String 40765

No. Time Source Destination Proto Length Info
28 2.357484258 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
30 2.767455703 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
34 3.628551272 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
36 4.199756418 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
38 5.736794182 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
40 8.295898948 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
41 9.242377255 192.168.0.106 92.123.36.106 TCP 66 50400 -> 443 [ACK] Seq=
43 9.249593911 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
44 9.275499528 92.123.36.106 192.168.0.106 TCP 66 [TCP ACKed unseen seq]
46 9.729562291 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
50 10.654954196 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
55 10.839751131 192.168.0.106 147.229.176.19 TCP 74 40765 -> 22 [SYN] Seq=
56 10.885872563 147.229.176.19 192.168.0.106 TCP 74 22 -> 40765 [SYN, ACK]
57 10.865903791 192.168.0.106 147.229.176.19 TCP 66 40765 -> 22 [ACK] Seq=
59 10.893878411 147.229.176.19 192.168.0.106 TCP 66 22 -> 40765 [ACK] Seq=
61 10.949605896 192.168.0.106 162.159.135.234 TCP 54 39820 -> 443 [ACK] Seq=
103 12.134407329 192.168.0.106 172.217.23.238 TCP 66 53662 -> 443 [ACK] Seq=
25 1.129038184 162.159.135.234 192.168.0.106 TLSv1.2 127 Application Data
27 2.357439390 162.159.135.234 192.168.0.106 TLSv1.2 294 Application Data
29 2.767433174 162.159.135.234 192.168.0.106 TLSv1.2 191 Application Data
35 5.600400000 162.159.135.234 192.168.0.106 TLSv1.2 400 Application Data

Frame 55: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: LiteonTe 96:0a:f7 (3c:95:09:96:0a:f7), Dst: Tp-LinkT a5:01:56 (18:fe:ed:a5:01:56)
Internet Protocol Version 4, Src: 192.168.0.106, Dst: 147.229.176.19
Transmission Control Protocol, Src Port: 40765, Dst Port: 22, Seq: 0, Len: 0

0000 10 fe ed a5 01 56 3c 95 09 96 0a f7 08 00 45 00 .....V<.....E-
0010 00 3c c3 da 40 00 40 06 71 d6 c0 a8 00 6a 93 e5 .<..@.q....j..
0020 b0 13 9f 3d 00 16 77 0d 03 af 00 00 00 00 a0 02 ...=.W.....
0030 fa f0 05 3a 00 00 02 04 05 b4 04 02 08 0a 42 b7 ...:.....B.
0040 dd 2c 00 00 00 00 01 03 03 07 .....

```

Bibliografia

Informácie som čerpal vo veľkej prevahe zo stránky <https://www.tcpdump.org/pcap.html>, pričom som bol inšpirovaný uvedeným zdrojovým kódom <https://www.tcpdump.org/sniffex.c>.
Z tohto dokumentu priamo citujem 2 funkcie:

```
void print_payload(const u_char *payload, int len);  
void print_hex_ascii_line(const u_char *payload, int len, int offset);
```

využité na vytlačenie dát a hexadecimálnych hodnôt.

Priama citácia:

- Autor: Tim Carstens
- Dátum: 03-05-2020
- Názov: sniffex.c
- Verzia 0.1.1 (2005-07-05)
- Typ: Zdrojový kód
- Webová adresa: <http://www.tcpdump.org/sniffex.c>