IPK projekt 2
Manuál

21.4.2019 Marián Kapišinský

1. Zadanie projektu

Úlohou projektu bolo vytvoriť jednoduchý sieťový TCP, UDP scanner v C/C++. Program má oskenovať zadanú IP adresu, prípadne doménové meno, a porty a na štandardný výstup vypísať ich stav (otvorený, filtrovaný, uzavretý).

2. Port scanner

2.1. TCP scan

Na skúmanie stavu TCP portu sa na daný port posielajú SYN (synchronize) pakety, nie je potrebné spraviť kompletný 3-way-handshake. Ak príde odpoveď RST (reset) port je označený ako uzavretý. Pokiaľ za daný časový interval (timeout) nepríde žiadna odpoveď port je označený ako filtrovaný. Na základe testovania bol interval nastavený na maximum 30 sekúnd, čo sa ukázalo ako približne najdlhšia doba, po ktorej prišla RST odpoveď. Pokiaľ je na danom porte spustená nejaká služba, je port označený ako otvorený, to znamená, že prišla odpoveď SYN,ACK. Spojenie je potrebné hneď ukončiť (FIN paket), čo ale nie potrebné v implementácii riešiť, pretože ho za nás automaticky odošle kernel.

Program musí správne vyplniť IP a TCP hlavičky:

IP hlavička musí obsahovať všetky polia definované v RFC 791:

Verzia IP protokolu: 4
IHL (Dĺžka IP hlavičky): 20 [1]
Typ služby: 0 (nepotrebné, môže byť 16 pre low delay)
Celková dĺžka: 40 (dĺžka IP a TCP hlavičiek)
ID: náhodne vygenerované 16 bitové číslo
Flags: 0 (nepotrebné)
Fragment offset: 0
TTL (Time to Live): 64 (môže byť aj vyššie)
Číslo nasledujúceho protokolu: 6 (TCP)
Kontrolný súčet IP hlavičky: počítaný CRC algoritmom
Zdrojová IP adresa: získaná zo sieťového rozhrania zariadenia [2]
Cieľová IP adresa: zadaná užívateľom
Voliteľné parametre: žiadne

[1] Minimálna dĺžka IP hlavičky bez žiadnych ďalších voliteľných parametrov
[2] [sekcia 2.3.]

TCP hlavička musí obsahovať všetky polia definované v RFC 793:

Zdrojový port: 43251 [3]
Cieľový port: zadaný užívateľom
Sekvenčné číslo: 1 (pri SYN pakete nezáleží)
Číslo odpovede (ACK):
Data offset: 20 [4]
Reserved: 0
Flags: URG = 0, ACK = 0, PSH = 0, RST = 0, SYN = 1, FIN = 0
Veľkosť okna: 1480 (môže byť aj väčšie, max. 2^{16}, t.j. 65 535)
Kontrolný súčet TCP hlavičky: počítaný CRC algoritmom
Urgent pointer: 0 (nepotrebné)
Voliteľné parametre: žiadne
Data: žiadne

Naplní sa najprv IP hlavička bez kontrolného súčtu (0), potom TCP hlavička, kde kontrolný súčet je vypočítaný pomocou pseudo hlavičky (zdrojová IP adresa, cieľová IP adresa, číslo protokolu, veľkosť hlavičky daného protokolu a dát (veľkosť dát je v našom prípade 0 bajtov)). Pozor, pokiaľ je kontrolný súčet vypočítaný zle, paket sa zahodí. Celková veľkosť datagramu je 40 bajtov (20 bajtov (IP) + 20 bajtov (TCP)).

Následne je takto poskladaný datagram odoslaný na cieľovú adresu pomocou BSD raw socketu. Socketu je potrebné nastaviť flag IP_HDRINCL, ktorý informuje kernel o tom, že IP hlavička je obsiahnutá v nami zaslanom datagrame.

Odpoveď je odchytená pomocou knižnice *libpcap.* Otvorí sa session na odchytávanie paketov zo sieťových zariadení s filtrom nastaveným na “**tcp src port** <source port number>”, t.zn. odchytávajú sa iba TCP pakety, ktoré majú daný zdrojový port (resp. cieľový port, na ktorý sme posielali SYN paket). Následne ak odpoveď nepríde do vyššie spomínaného 30 sekundového limitu nastaví sa signál alarmu (SIGALRM) a zavolá sa funkcia na prerušenie cyklu čakania na odpoveď a paket sa opätovne pošle. Znova sa po uplynutí 30 sekúnd nastaví signál alarmu (SIGALRM) a cykle sa opäť preruší, **preto overenie, či je port filtrovaný trvá až 1 minútu**. Na základe odpovede je vygenerovaný výstup, podľa vyššie popísaného princípu.

Toto sa opakuje pre všetky zadané TCP porty (argument -pt).

[3] Náhodne vybraný, tak aby nebol v konflikte s portom žiadnej známej služby
[4] Minimálna dĺžka TCP hlavičky bez žiadnych ďalších voliteľných parametrov

2.2. UDP scan

Skúmanie stavu UDP portu sa od TCP líši tým, že UDP port nemusí odoslať žiadnu odpoveď na prijatú správu. Tým pádom pokiaľ do zadaného časového limitu (takisto 30 sekúnd ako pri TCP scanneri [sekcia 2.1.]) nepríde ICMP správa typu 3 kódu 3 (port je nedostupný) je port považovaný za otvorený, pričom ale tiež nie je garantované, že táto správa príde a teda port bude tiež považovaný za otvorený.

Princíp vytvárania UDP paketu je podobný ako pri TCP. V IP hlavičke sa mení Celková dĺžka na 28 bajtov (veľkosť UDP hlavičky je 8 bajtov) a Číslo nasledujúceho protokolu na 17 (UDP).

UDP hlavička musí obsahovať všetky polia definované v RFC 768:

Zdrojový port: 43252 (o 1 vyšší na odlíšenie od TCP)
Cieľový port: zadaný užívateľom
Zeros: 0 (8 bitový nastavených na hodnotu 0)
Dĺžka: 8 (veľkosť UDP hlavičky)
Kontrolný súčet UDP hlavičky: počítaný CRC algoritmom

Ďalší postup je podobný s TCP scanom [sekcia 2.1.]. Na odchytenie paketu sa použije filter “**src net** <source address> **and icmp**”**.**

Celý princíp sa opakuje pre všetky zadané UDP porty (argument -pu).

2.3. Argument -i

Tento *voliteľný* argument špecifikuje sieťové rozhranie, cez ktoré sa budú vytvorené datagramy odosielať. Program pomocou funkcie *ioctl()* s požiadavkou SIOCGIFFLAGS overí, či zadané zariadenie nie je loopback a následne získa jeho adresu požiadavkou SIOCGIFADDR. Pokiaľ scannujeme localhost je loopbackové rozhranie povolené.

Ak ten tento argument nie je zadaný program vyberia prvé neloopbackové rozhranie. Funkcia toto spravujúca využíva kód z https://ubuntuforums.org/showthread.php?t=1396491 , modifikovaný o ignorovanie loopback zariadení.

2.4. Príklady použitia

sudo ./ipk-scan {-i <interface>} -pu <port-ranges> -pt <port-ranges> [<domain-name> | <IP-address>]

sudo ./ipk-scan -pt 21,22,143 -pu 53,67 localhost

sudo ./ipk-scan -i ens33 -pt 21,22-25,443 -pu 53,67 localhost

sudo ./ipk-scan -i lo -pt 21 127.0.0.1 alebo localhost

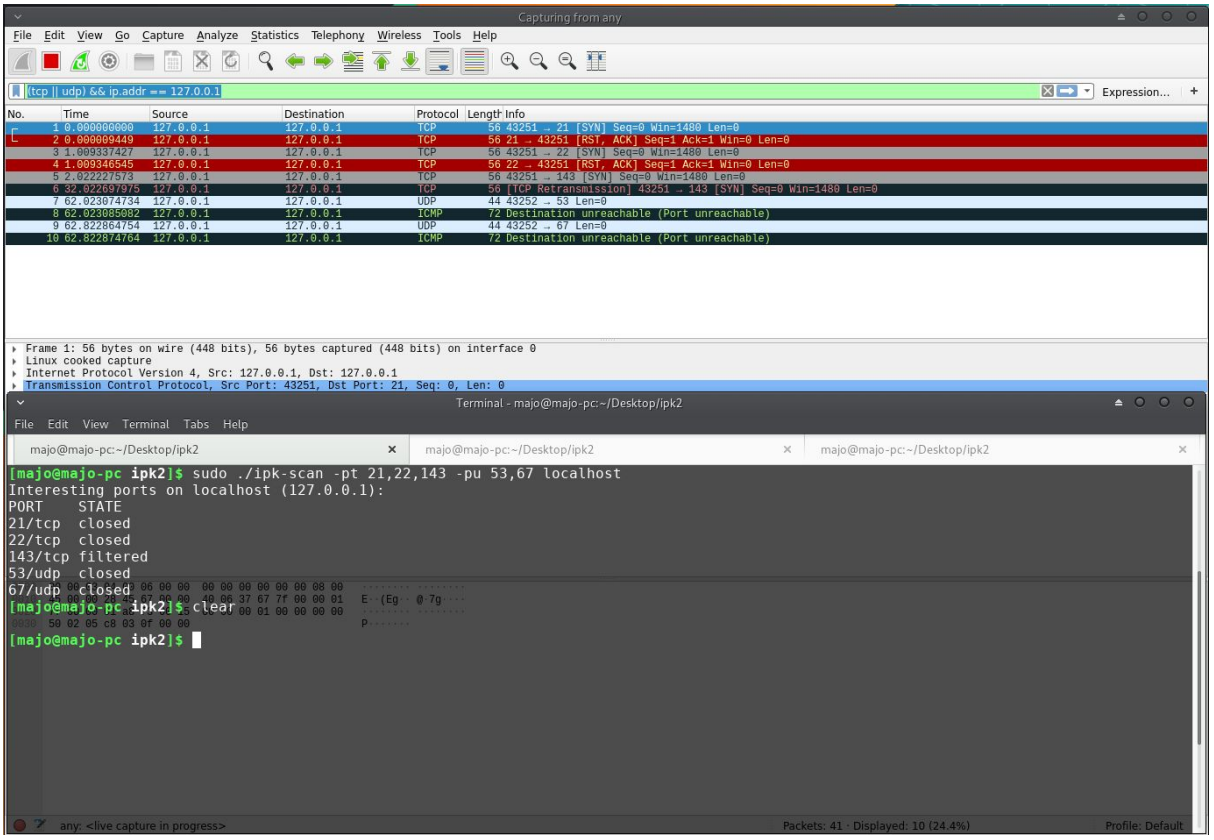
sudo ./ipk-scan -pt 21,22,143 -pu 53,67 merlin.fit.vutbr.cz

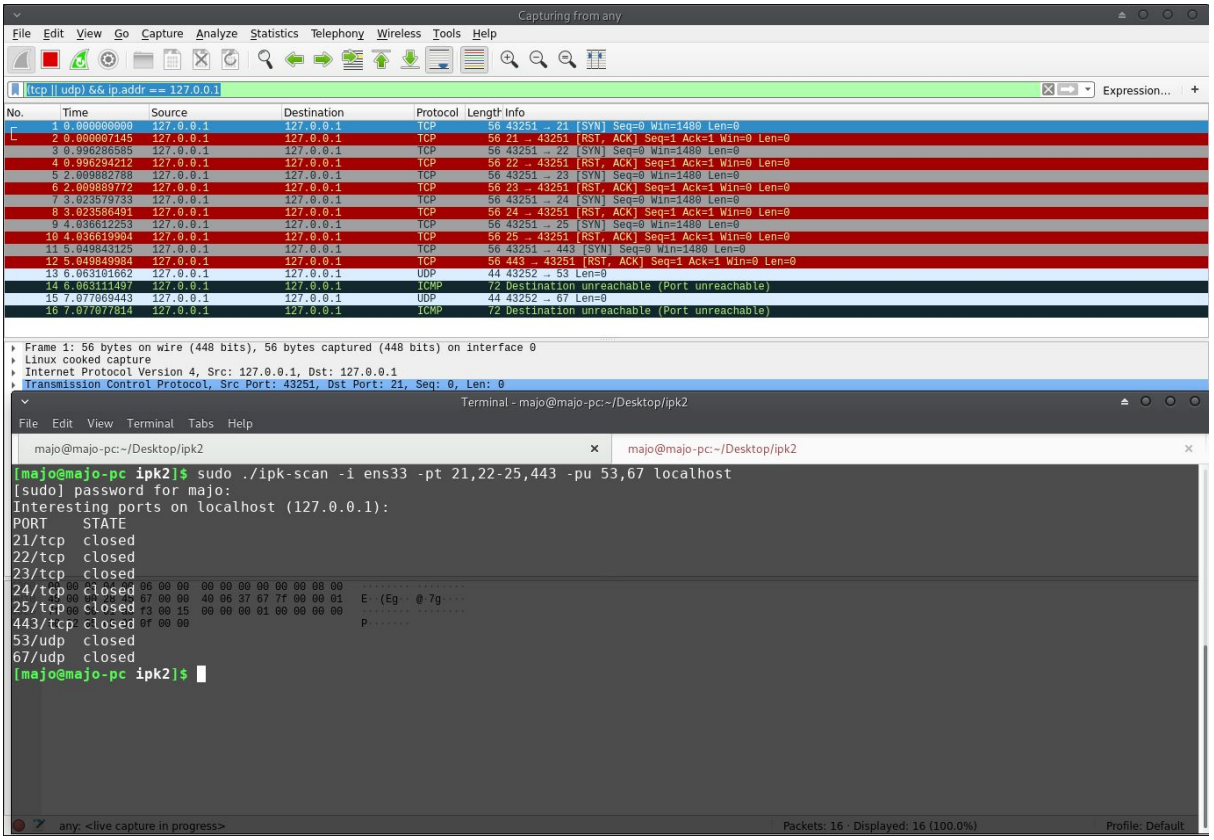
Program by mal byť ošetrený voči neplatným IP adresám, doménovým menám, neplatným číslam a intervalom portov.

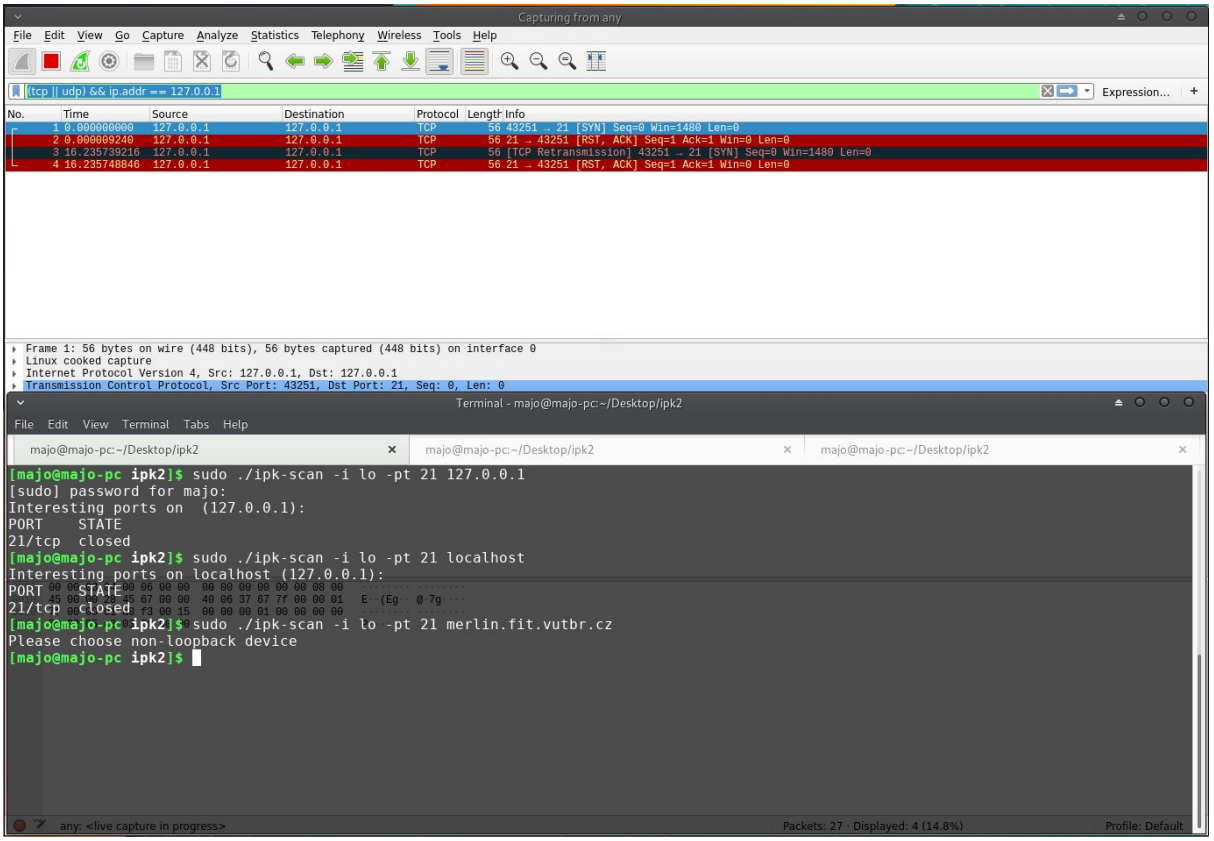
3. Testovanie

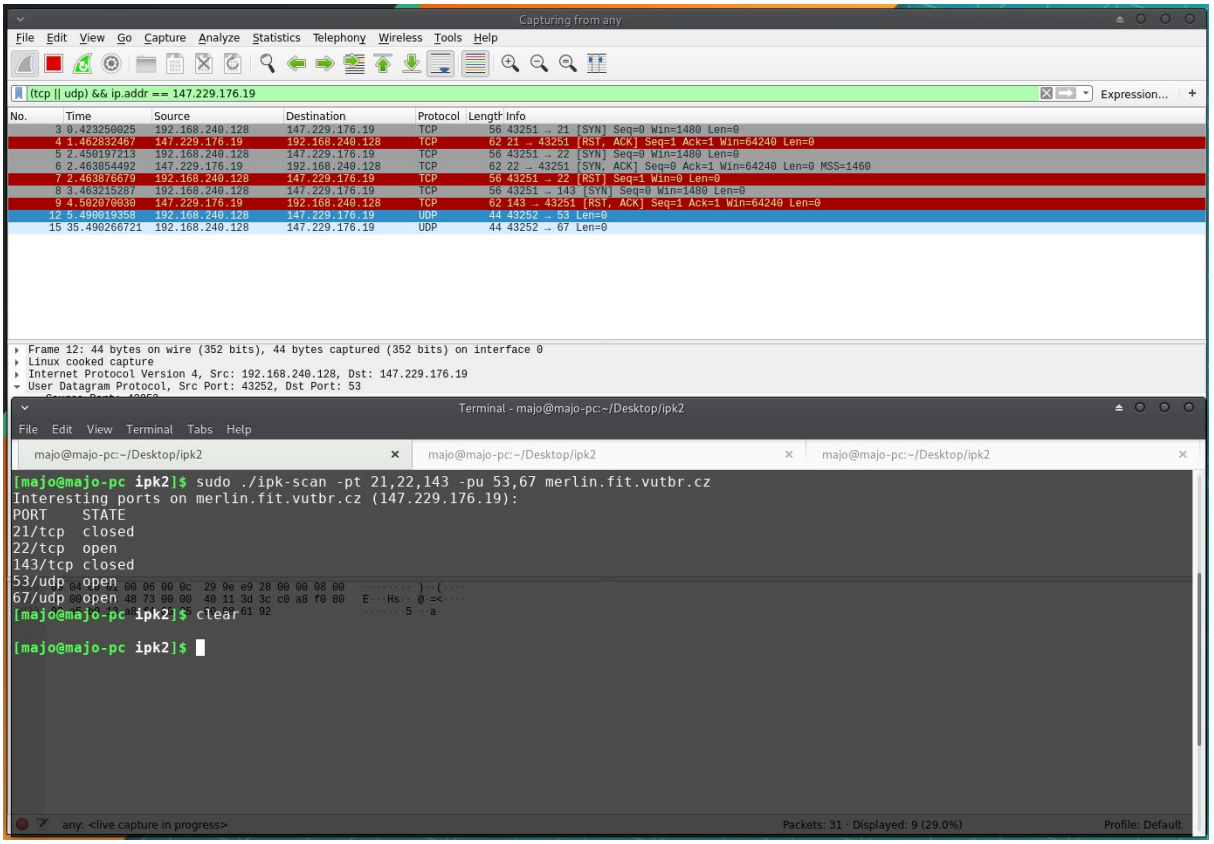
Testovanie projektu prebehlo na 2 virtuálnych strojoch (Manjaro Deepin a Ubuntu (referenčný stroj)). Aplikácia bola testovaná so vstupmi uvedenými v sekcií 2.4. a mnohými ďalšími na localhoste, merlinovi a iných doménach, napr. google.com. Testovala sa hlavná funkcionalita programu a rôzne kombinácie argumentov.

Screenshoty z terminálu a wiresharku:


Capturing from any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(tcp || udp) && ip.addr == 127.0.0.1
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 TCP 56 43251 → 21 [SYN] Seq=0 Win=1480 Len=0
2 0.000009449 127.0.0.1 127.0.0.1 TCP 56 21 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3 1.009337427 127.0.0.1 127.0.0.1 TCP 56 43251 → 22 [SYN] Seq=0 Win=1480 Len=0
4 1.009346545 127.0.0.1 127.0.0.1 TCP 56 22 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
5 2.022227573 127.0.0.1 127.0.0.1 TCP 56 43251 → 143 [SYN] Seq=0 Win=1480 Len=0
6 32.922697975 127.0.0.1 127.0.0.1 TCP 56 [TCP Retransmission] 43251 → 143 [SYN] Seq=0 Win=1480 Len=0
7 62.023074734 127.0.0.1 127.0.0.1 UDP 44 43252 → 53 Len=0
8 62.023085082 127.0.0.1 127.0.0.1 ICMP 72 Destination unreachable (Port unreachable)
9 62.822864764 127.0.0.1 127.0.0.1 UDP 44 43252 → 67 Len=0
10 62.822874764 127.0.0.1 127.0.0.1 ICMP 72 Destination unreachable (Port unreachable)
Frame 1: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 43251, Dst Port: 21, Seq: 0, Len: 0
Terminal - majo@majo-pc:~/Desktop/ipk2
File Edit View Terminal Tabs Help
majo@majo-pc:~/Desktop/ipk2
majo@majo-pc:~/Desktop/ipk2
majo@majo-pc:~/Desktop/ipk2
[majo@majo-pc ipk2]$ sudo ./ipk-scan -pt 21,22,143 -pu 53,67 localhost
Interesting ports on localhost (127.0.0.1):
PORT STATE
21/tcp closed
22/tcp closed
143/tcp filtered
53/udp closed
67/udp closed
[majo@majo-pc ipk2]$ clear
[majo@majo-pc ipk2]$
any: <live capture in progress>
Packets: 41 · Displayed: 10 (24.4%)
Profile: Default


Capturing from any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(tcp || udp) && ip.addr == 127.0.0.1
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 TCP 56 43251 → 21 [SYN] Seq=0 Win=1480 Len=0
2 0.000007145 127.0.0.1 127.0.0.1 TCP 56 21 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3 0.996286335 127.0.0.1 127.0.0.1 TCP 56 43251 → 22 [SYN] Seq=0 Win=1480 Len=0
4 0.996294212 127.0.0.1 127.0.0.1 TCP 56 22 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
5 2.009882788 127.0.0.1 127.0.0.1 TCP 56 43251 → 23 [SYN] Seq=0 Win=1480 Len=0
6 2.009889772 127.0.0.1 127.0.0.1 TCP 56 23 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
7 3.023579733 127.0.0.1 127.0.0.1 TCP 56 43251 → 24 [SYN] Seq=0 Win=1480 Len=0
8 3.023586491 127.0.0.1 127.0.0.1 TCP 56 24 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9 4.036612253 127.0.0.1 127.0.0.1 TCP 56 43251 → 25 [SYN] Seq=0 Win=1480 Len=0
10 4.036619904 127.0.0.1 127.0.0.1 TCP 56 25 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
11 5.049843125 127.0.0.1 127.0.0.1 TCP 56 43251 → 443 [SYN] Seq=0 Win=1480 Len=0
12 5.049849984 127.0.0.1 127.0.0.1 TCP 56 443 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
13 6.063101662 127.0.0.1 127.0.0.1 UDP 44 43252 → 53 Len=0
14 6.063111497 127.0.0.1 127.0.0.1 ICMP 72 Destination unreachable (Port unreachable)
15 7.077069443 127.0.0.1 127.0.0.1 UDP 44 43252 → 67 Len=0
16 7.077077814 127.0.0.1 127.0.0.1 ICMP 72 Destination unreachable (Port unreachable)
Frame 1: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 43251, Dst Port: 21, Seq: 0, Len: 0
Terminal - majo@majo-pc:~/Desktop/ipk2
File Edit View Terminal Tabs Help
majo@majo-pc:~/Desktop/ipk2
majo@majo-pc:~/Desktop/ipk2
[majo@majo-pc ipk2]$ sudo ./ipk-scan -i ens33 -pt 21,22-25,443 -pu 53,67 localhost
[sudo] password for majo:
Interesting ports on localhost (127.0.0.1):
PORT STATE
21/tcp closed
22/tcp closed
23/tcp closed
24/tcp closed
25/tcp closed
443/tcp closed
53/udp closed
67/udp closed
[majo@majo-pc ipk2]$
any: <live capture in progress>
Packets: 16 · Displayed: 16 (100.0%)
Profile: Default


Capturing from any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(tcp || udp) && ip.addr == 127.0.0.1
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 TCP 56 43251 → 21 [SYN] Seq=0 Win=1480 Len=0
2 0.000009240 127.0.0.1 127.0.0.1 TCP 56 21 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3 16.235739216 127.0.0.1 127.0.0.1 TCP 56 [TCP Retransmission] 43251 → 21 [SYN] Seq=0 Win=1480 Len=0
4 16.235748846 127.0.0.1 127.0.0.1 TCP 56 21 → 43251 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
Frame 1: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 43251, Dst Port: 21, Seq: 0, Len: 0
Terminal - majo@majo-pc:~/Desktop/ipk2
File Edit View Terminal Tabs Help
majo@majo-pc:~/Desktop/ipk2
[majo@majo-pc ipk2]$ sudo ./ipk-scan -i lo -pt 21 127.0.0.1
[sudo] password for majo:
Interesting ports on (127.0.0.1):
PORT STATE
21/tcp closed
[majo@majo-pc ipk2]$ sudo ./ipk-scan -i lo -pt 21 localhost
Interesting ports on localhost (127.0.0.1):
PORT STATE
21/tcp closed
[majo@majo-pc ipk2]$ sudo ./ipk-scan -i lo -pt 21 merlin.fit.vutbr.cz
Please choose non-loopback device
[majo@majo-pc ipk2]$
any: <live capture in progress>
Packets: 27 · Displayed: 4 (14.8%)
Profile: Default


Capturing from any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(tcp || udp) && ip.addr == 147.229.176.19
Expression...
No. Time Source Destination Protocol Length Info
3 0.423250025 192.168.240.128 147.229.176.19 TCP 56 43251 → 21 [SYN] Seq=0 Win=1480 Len=0
4 1.462832467 147.229.176.19 192.168.240.128 TCP 62 21 → 43251 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
5 2.450197213 192.168.240.128 147.229.176.19 TCP 56 43251 → 22 [SYN] Seq=0 Win=1480 Len=0
6 2.463854492 147.229.176.19 192.168.240.128 TCP 62 22 → 43251 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
7 2.463876679 192.168.240.128 147.229.176.19 TCP 56 43251 → 22 [RST] Seq=1 Win=0 Len=0
8 3.403215287 192.168.240.128 147.229.176.19 TCP 56 43251 → 143 [SYN] Seq=0 Win=1480 Len=0
9 4.59207[illegible] 147.229.176.19 192.168.240.128 TCP 62 143 → 43251 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
12 5.490019358 192.168.240.128 147.229.176.19 UDP 44 43252 → 53 Len=0
15 35.490266721 192.168.240.128 147.229.176.19 UDP 44 43252 → 67 Len=0
Frame 12: 44 bytes on wire (352 bits), 44 bytes captured (352 bits) on interface 0
Linux cooked capture
Internet Protocol Version 4, Src: 192.168.240.128, Dst: 147.229.176.19
User Datagram Protocol, Src Port: 43252, Dst Port: 53
Terminal - majo@majo-pc:~/Desktop/ipk2
File Edit View Terminal Tabs Help
majo@majo-pc:~/Desktop/ipk2
[majo@majo-pc ipk2]$ sudo ./ipk-scan -pt 21,22,143 -pu 53,67 merlin.fit.vutbr.cz
Interesting ports on merlin.fit.vutbr.cz (147.229.176.19):
PORT STATE
21/tcp closed
22/tcp open
143/tcp closed
53/udp open
67/udp open
[majo@majo-pc ipk2]$ clear
[majo@majo-pc ipk2]$
any: <live capture in progress>
Packets: 31 · Displayed: 9 (29.0%)
Profile: Default

4. Obmedzenia

Aplikácia nepodporuje IPv6.

Zdroje:

- RFC 793 - Transmission Control Protocol
- RFC 791 - Internet Protocol
- RFC 768 - User Datagram Protocol
- RFC 792 - Internet Control Message Protocol
- HE RAW SOCKET PROGRAM EXAMPLES - https://www.tenouk.com/Module43a.html
- Port scanner - http://en.wikipedia.org/wiki/Port_scanner
- TCP/UDP scan - http://nmap.org/nmap_doc.html
- https://ubuntuforums.org/showthread.php?t=1396491
- https://github.com/praveenkmurthy/Raw-Sockets/blob/master/tcp_handler.c - použitá funckia csum

- https://www.tcpdump.org/pcap.html
- https://www.tcpdump.org/manpages/pcap.3pcap.html