Laboratorul 05 - ICMP

Înainte de laborator

Laboratorul nu va funcționa decât pe Linux. Nu WSL sau macOS.

La adresa https://ocw.cs.pub.ro/courses/pc/res/mv puteți pasi o masina virtuala cu Ubuntu 18.04.

Introducere

ICMP este un protocol folosit în Internet cu rol de diagnostic: semnalare de erori, informații operaționale etc. Deși pachetele ICMP sunt încapsulate în pachete IP, protocolul ICMP este considerat tot un protocol de nivel 3 și reprezintă o parte integrală a funcționalității IP-ului.

Probabil cea mai populară aplicație a protocolului, este unealta `ping` (disponibilă pe mai toate sistemele de operare), pe care ați mai folosit-o în trecut pentru a testa conectivitatea cu un host anume:

```
$ ping google.com
PING google.com (172.217.22.78) 56(84) bytes of data.
64 bytes from fra15s17-in-f14.1e100.net (172.217.22.78): icmp_seq=1 ttl=52 time=26.4 ms
64 bytes from fra15s17-in-f14.1e100.net (172.217.22.78): icmp_seq=2 ttl=52 time=26.3 ms
64 bytes from fra15s17-in-f14.1e100.net (172.217.22.78): icmp_seq=3 ttl=52 time=26.4 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 26.350/26.401/26.437/0.137 ms
```

Deasemenea, ICMP mai este folosit și de routere pentru a semnifica, de exemplu, lipsa unei rute către destinație, sau renunțarea la un pachet din cauza expirării câmpului Time-To-Live.

Antetul ICMP

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Rest of Header                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Type** - tipul mesajului ICMP (e.g. Echo request, destination unreachable); împreună cu câmpul **code** identifică exact mesajul ICMP
- **Code** - împreună cu câmpul **type** identifică exact mesajul ICMP
- **Checksum** - suma de control pentru antetul de ICMP; se calculează identic ca cea pentru IPv4 (1's complement of words).
- **Rest of Header** - semnificația acestor 4 octeți depinde de combinația de **type** și **code**.

Ping

`ping` este un utilitar care se folosește de mesaje ICMP tip ECHO_REQUEST pentru a solicita de la un alt host un mesaj de tip ECHO_REPLY, putând fi astfel folosit pentru a vedea dacă un host e accesibil. Deasemenea, `ping` oferă și alte informații, e.g. rtt

Exemplu

```
$ ping -c 4 google.com
PING google.com (172.217.22.78) 56(84) bytes of data.
64 bytes from fra15s17-in-f78.1e100.net (172.217.22.78): icmp_seq=1 ttl=52 time=26.4 ms
64 bytes from fra15s17-in-f78.1e100.net (172.217.22.78): icmp_seq=2 ttl=52 time=26.9 ms
64 bytes from fra15s17-in-f78.1e100.net (172.217.22.78): icmp_seq=3 ttl=52 time=26.3 ms
64 bytes from fra15s17-in-f78.1e100.net (172.217.22.78): icmp_seq=4 ttl=52 time=26.4 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 26.388/26.575/26.975/0.307 ms
```

Traceroute

`traceroute` este un utilitar folosit pentru a inspecta traseul până la un host.

Exemplu

```
$ traceroute google.com
traceroute to google.com (172.217.22.78), 30 hops max, 60 byte packets
 1  _gateway (172.19.2.254)  0.247 ms  0.231 ms  0.223 ms
 2  141.85.233.1 (141.85.233.1)  0.639 ms  0.634 ms  0.698 ms
 3  172.31.255.85 (172.31.255.85)  1.228 ms  1.373 ms  1.533 ms
 4  172.31.255.1 (172.31.255.1)  0.922 ms  0.915 ms  0.907 ms
 5  po-23.accl.buc.roedu.net (37.128.225.225)  1.138 ms  1.130 ms  1.122 ms
 6  bu-13.core2.buc.roedu.net (37.128.232.177)  1.752 ms  3.117 ms  3.100 ms
 7  hu-0-0-0-0.core3.nat.roedu.net (37.128.239.101)  1.610 ms  4.478 ms  4.500 ms
 8  te-0-6-0-1.peer1.nat.roedu.net (37.128.239.42)  1.928 ms  1.918 ms  1.924 ms
 9  google.interlan.ro (80.184.125.129)  29.255 ms  29.241 ms  27.005 ms
10  108.170.252.1 (108.170.252.1)  27.537 ms  27.608 ms 108.170.251.129 (108.170.251.129)  28.521 ms
11  72.14.232.33 (72.14.232.33)  26.544 ms  26.567 ms  26.463 ms
12  fra15s17-in-f78.1e100.net (172.217.22.78)  26.285 ms  26.279 ms  26.372 ms
```

Debugging

Pe lângă tehnicile de debugging aplicabile programării în C (gdb, printf etc.), v-ar ajuta și abilitatea de a observa ce pachete vin și pleacă de pe mașina voastră.

Pentru asta, recomandăm două unelte, pe care probabil le veți găsi folositoare atât la acest laborator, cât și la cele viitoare și la teme.

tcpdump

`tcpdump` este un utilitar din linia de comandă cu care puteți monitoriza traficul de pe mașina voastră. Cel mai simplu exemplu de rulare este:

Va trebui să rulați `tcpdump` ca root!

```
$ sudo tcpdump
09:56:18.293641 IP rhel75.localdomain.ssh > 192.168.64.1.56322: Flags [P.], seq 3770820720:3770820916
09:56:18.293794 IP 192.168.64.1.56322 > rhel75.localdomain.ssh: Flags [.], ack 196, win 391, options
09:56:18.295058 IP rhel75.59883 > gateway.domain: 24864+ PTR? 1.64.168.192.in-addr.arpa. (43)
09:56:18.310225 IP gateway.domain > rhel75.59883: 24864 NXDomain* 0/1/0 (102)
09:56:18.312482 IP rhel75.49685 > gateway.domain: 34242+ PTR? 28.64.168.192.in-addr.arpa. (44)
09:56:18.322425 IP gateway.domain > rhel75.49685: 34242 NXDomain* 0/1/0 (103)
09:56:18.323164 IP rhel75.56631 > gateway.domain: 29984+ PTR? 1.122.168.192.in-addr.arpa. (44)
09:56:18.323342 IP rhel75.localdomain.ssh > 192.168.64.1.56322: Flags [P.], seq 196:584, ack 1, win 3
09:56:18.323563 IP 192.168.64.1.56322 > rhel75.localdomain.ssh: Flags [.], ack 584, win 411, options
09:56:18.335569 IP gateway.domain > rhel75.56631: 29984 NXDomain* 0/1/0 (103)
09:56:18.336429 IP rhel75.44007 > gateway.domain: 61677+ PTR? 98.122.168.192.in-addr.arpa. (45)
09:56:18.336655 IP gateway.domain > rhel75.44007: 61677* 1/0/0 PTR rhel75. (65)
09:56:18.337177 IP rhel75.localdomain.ssh > 192.168.64.1.56322: Flags [P.], seq 584:1644, ack 1, win
^C
```

Atfel, `tcpdump` monitorizează continuu traficul de pe toate interfețele și vă afișează, pentru fiecare pachet, un timestamp și rezumatul pachetului.

Pentru depanarea programelor în care creați de la zero pachete, vă recomandăm să rulați:

```
$ sudo tcpdump -vv -XX
21:58:17.465707 IP (tos 0x0, ttl 64, id 57672, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.0.73 > 8.8.8.8: ICMP echo request, id 5, seq 1, length 64
        0x0000:  ac22 0554 df5e 1863 c883 8db1 0800 4500  ."..T.^.c.....E.
        0x0010:  0054 e148 4000 4001 b85f c0a8 0049 0808  .T.H@.@.._...I..
        0x0020:  0808 0800 d424 0005 0001 d988 6e5e 0000  .....$......n^..
        0x0030:  0000 1b1b 0700 0000 0000 1011 1213 1415  ................
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
        0x0060:  3637                                     67
21:58:17.525308 IP (tos 0x0, ttl 55, id 0, offset 0, flags [none], proto ICMP (1), length 84)
    8.8.8.8 > 192.168.0.73: ICMP echo reply, id 5, seq 1, length 64
        0x0000:  1863 c883 8db1 ac22 0554 df5e 0800 4500  .c.....".T.^..E.
        0x0010:  0054 0000 0000 3701 b2a8 0808 0808 c0a8  .T....7.........
        0x0020:  0049 0000 dc24 0005 0001 d988 6e5e 0000  .I...$......n^..
        0x0030:  0000 1b1b 0700 0000 0000 1011 1213 1415  ................
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425  ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()*+,-./012345
        0x0060:  3637                                     67
^C
```

-vv: output verbose, vă oferă mai multe informații despre pachete (e.g. dacă au checksum greșit)

-XX: hexdump în întreg conținutul pachetului

Cel mai probabil, veți observa că în orice moment de timp, pe mașina voastră există foarte mult trafic, care va polua outputul de `tcpdump` și va face foarte dificil să monitorizați doar pachetele de care sunteți interesați. Pentru asta, există filtre. De exemplu, pentru a vedea doar pachetele de tip ICMP:

```
$ sudo tcpdump -vv -XX icmp
```

Pentru a vedea doar pachetele către și de la `8.8.8.8`:

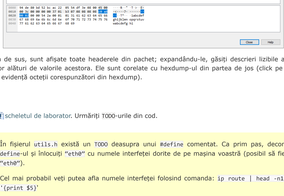
```
$ sudo tcpdump -vv -XX host 8.8.8.8
```

Pentru mai multe opțiuni și filtre, vă recomandăm pagina de manual.

Wireshark

Wireshark este alt tool pentru monitorizarea traficului, care dispune de o interfață grafică. Este o unealtă educativă foarte bună, deoarece vă sustrage de nevoia de a cunoaște offseturi și valori speciale, atunci când doriți să inspectați hexdump-ul unui pachet.

Observați outputul pentru un singur pachet:



În partea de sus, sunt afișate toate headerele din pachet; expandându-le, găsiți descrieri lizibile ale tuturor câmpurilor alături de valorile acestora. Ele sunt corelate cu hexdump-ul din partea de jos (click pe un câmp, scoate în evidență octeții corespunzători din hexdump).

Taskuri

Descărcați scheletul de laborator. Urmăriți TODO-urile din cod.

În fișierul `utils.h` există un `TODO` deasupra unui `#define` comentat. Ca prim pas, decomentați `define`-ul și înlocuiți `"eth0"` cu numele interfeței dorite de pe mașina voastră (posibil să fie totuși `"eth0"`).

Cel mai probabil veți putea afla numele interfeței folosind comanda: `ip route | head -n1 | awk '{print $5}'`

Dacă comanda de mai sus nu ajută, pentru a afla numele interfeței, rulați `ifconfig`. Dintre interfețele care apar, probabil veți dori una care începe cu `"eth"` sau `"en"` (dacă aveți conexiune pe fir) sau cu `"wl"` pentru conexiune wireless.

Ne dorim să implementăm funcționalitățile de bază ale `ping` și `traceroute`. Programul va putea fi rulat în următoarele moduri:

```
$ sudo ./icmp ping google.com       # ping către un URL
$ sudo ./icmp ping 8.8.8.8          # ping către o adresă IP
$ sudo ./icmp ping google.com 6     # 6 pinguri către un host
$ sudo ./icmp traceroute google.com # traceroute către un host
```

Parsarea argumentelor din linia de comandă și convertirea unui URL la o adresă IP este deja realizată în scheletul de cod!

1. Implementați funcționalitatea de `ping`.
2. Implementați funcționalitatea de `traceroute`; orientați-vă după modul de lucru folosit la `ping`.

FAQ

- Dacă la rulare primiți o eroare legată de apelul către `socket`, cel mai probabil nu rulați ca root; folosiți `sudo`.

```
(icmp.c:185): socket
Operation not permitted
```

- Dacă la rulare primiți o eroare legată de apelul către `send`, cel mai probabil ați greșit valoarea câmpului `size` al pachetului.

```
(icmp.c:34): send
Invalid argument
```

- Dacă observați că unele câmpuri au valori aiurea, în special mult prea mari sau mult prea mici, probabil ați uitat să convertiți valorile din host-order în network-order (sau invers):

```
#include <arpa/inet.h>
uint32_t htonl(uint32_t hostlong);
uint16_t htons(uint16_t hostshort);
uint32_t ntohl(uint32_t netlong);
uint16_t ntohs(uint16_t netshort)
```

- Cum arata antetul IP? https://tools.ietf.org/html/rfc791#page-11