Proiectul Netdiag(A)

Băncescu Rareș-Emil, Grupa 2A6

Facultatea de Informatică UAIC

1 Introducere:

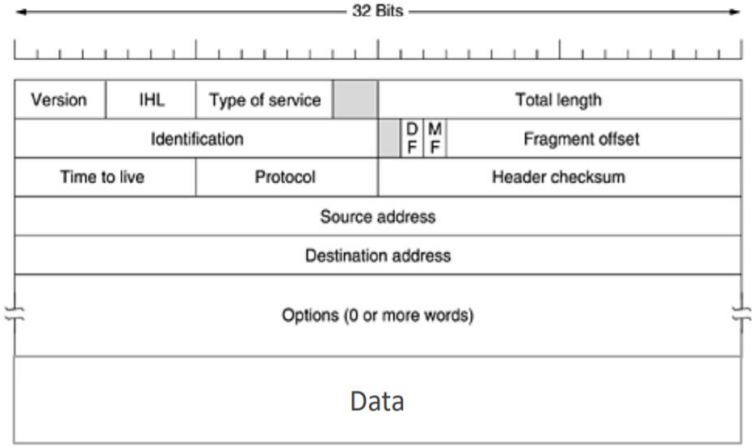
Proiectul NetDiag presupune implementarea unei aplicații care afișează toate adresele IP ale routerelor intermediare (hop-uri) și timpul de răspuns(actualizat periodic) prin care trec pachetele până la o anumită adresă destinație, dacă aceasta este accesibilă. Această aplicatie va funcționa în mod similar comenzii mtr, comanda folosită pentru Network Diagnostic și Network Troubleshooting. Comanda afișează pentru fiecare hop numele (DNS name) sau adresa IP, procentajul de distrugere a pachetelor trimise(Loss) și o statistică a timpului necesari ajungerii pachetelor până la acel hop. Aplicația presupune două componente, clientul si serverul, clientul avand rolul de a programa serverul cu diverși parametrii, iar serverul de a executa Traceroute-ul.

Am ales acest proiect deoarece descoperirea problemelor din retea reprezintă o problemă foarte importantă in rețelistică, iar implementarea acestui proiect presupune utilizarea multor protocoale și concepte teoretice ce sunt esențiale pentru înțelegerea funcționalității rețelelor de calculatoare.

2 Tehnologii utilizate:

2.1 Protocolul IP:

Protocolul IP este un protocol de la nivelul rețea care ajută la trimiterea pachetelor de la o gazdă la alta. O datagramă IP(un pachet), ce este trimis către destinație constă dintr-o parte de antet, unde sunt prezente diferite informații despre pachet(TTL, protocol, checksum...) ce are dimensiunea de 20 de octeți și o parte opțională de lungime variabilă.

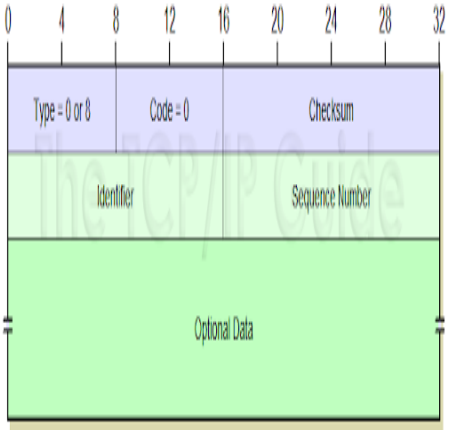


Version: Versiunea IP care este folosită (de obicei 4).
IHL: Lungimea antetului IP.
Type of service: Permite gazdei să comunice ce tip de serviciu dorește (8 biți, primii 3 precedență, iar următorii specifică delay, throughput, reliability, monetary cost, ultimul bit fiind rezervat, mereu setat 0).
Total Length: Lungimea totală a pachetului trimis.
Identification: Identidificarea pachetului(la o datagramă).
Flagurile DF/MF: Indică dacă o datagramă este fragmentată sau nu.
Fragment Offset: Locul fragmentului în datagramă (dacă aceasta a fost fragmentată).
TTL(time to live): Specifică durata de viață a pachetului.
Protocol: Specifică protocolul de nivel superior prin care este trimis pachetul.
Header Checksum: Detectare de erori.

Optiuni: Acest câmp a fost depreciat în ultimul timp. **Source Adress și Destination adress:** Sursă și destinație.

2.2 Protocolul ICMP:

Protocolul ICMP este un protocol de la nivelul rețea care se folosește la semnalizarea și diagnosticarea problemelor din rețea. Mesajele ICMP sunt incapsulate în interiorul pachetelor IP(câmpurile Protocol și Data). Principalele programe care se bazează pe acest protocol sunt Ping si Traceroute.Putem folosi acest protocol pentru a transmite informație, deși nu este un protocol de la nivelul transport deoarece mesajele ICMP sunt procesate de software-ul IP, nu de procesele utilizatorului. Mesajele trimise cu ajutorul acestui protocol sunt de tipul Echo Request, iar cele primite sunt de tipul Echo Reply. Cu ajutorul acestui protocol putem afla adresele IP ale hop-urilor intermediare. Utilizând acest protocol împreună cu IP putem trimite pachete către o sursă fară sa stabilim o conexiune înainte(PING).



Type: Tipul pachetului trimis:
0: Echo reply.
1,2: Rezervat.
3: Destinație de neatins.
8: Echo Request
11: Time exceeded (TTL devine 0).
30: Traceroute.
Code: Subtipul pachetului ICMP în funcție cu tipul selectat anterior.
0(In cazul Echo reply): Răspuns de tip ecou.
9(In cazul 3(Destinație de neatins): Acces rețea interzis.
Toate Posibilitățile
Checksum: Modalitatea de a verifica integritatea pachetului(Opțională)
Identifier: Un câmp de identificare utilizat pentru a utiliza mesaje de tip echo.
Sequence Number: Pentru mesaje de tip echo.

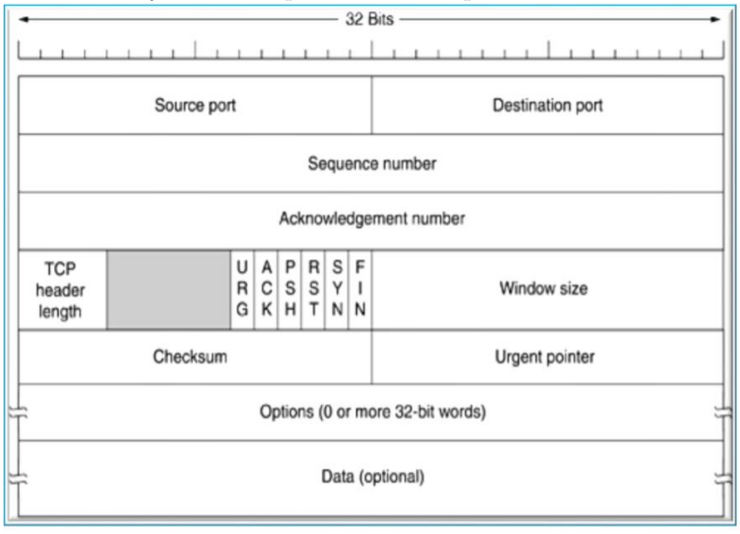
2.3 Protocolul UDP:

Protocolul UDP este un protocol de comunicație ce aparține nivelului transport. Este un protocol ce constituie modul de comunicare fără conexiune, astfel nu există o legatură efectivă între expeditor si destinatar, neoferind siguranța sosirii datelor și nici mecanisme de verificare a ordinii sosirii datagramelor. Totuși acest protocol dispune de suma de control, ce poate fi folosită pentru verificarea integrității datelor. Vom folosi acest protocol trimite pachete spre routere(fără a realiza conexiune), pentru a obține reply cu IP-ul router-ului.

Port Sursa și Port Destinație: End Point-urile de pe mașina sursă și destinație
Lungime: Lungimea datagramei
Suma de control:(Optional) Folosită pentru verificare

2.4 Protocolul TCP:

TCP este un protocol ce apare la nivelul transport, este un protocol ce constituie modul de comunicare orientat conexiune, între două puncte terminale. Utilizându-se acest protocol se efectuează o conexiune virtuală full duplex (fiecare punct este definit de către o adresă IP și de un port TCP). TCP, spre deosebire de UDP este folosit fără a folosi protocolul IP. Utilizând acest protocol pot fi trimise multe date pe Internet, fara a exista posibilitatea pierderii datelor, ca in cazul protocolului IP, in care datele sunt fragmentate in mai multe pachete. Datagramele beneficiază de modalități de retransmitere,confirmare,verificare de pierdere de date, fiind foarte folositor pentru conexiuni sigure. Un dezavantaj al acestui protocol este faptul că trimiterea datelor necesită mult timp.



Source port, Destination port: Emițător, Receptor.
Sequence Number: Număr de ordine (Depinde în funcție de flagul SYN)
Acknowledgement Number: Funcționează dacă flagul ACK este setat și are valoarea numărului de ordine al următorului octet care trebuie recepționat.
Window Size: Numărul de octeți pe care receptorul dorește să îl recepționeze (controlul fluxului).
TCP header length: Lungimea pachetului.
Flaguri: URG-urgent, **ACK**-confirmare pachet, **RST**-erori conexiune, **PSH**-date trimise imediat, **SYN**-stabilirea conexiunii, **FIN**-încheiere conexiune
Window size: Numărul de octeți pe care receptorul dorește să îl recepționeze(controlul fluxului)
Checksum: Suma de control.
Urgent pointer: Functionează dacă URG este setat și indică deplasamentul față de numărul curent de ordine la care se găsește informația urgentă

2.5 SQLite:

SQLite este o librărie din limbajul C care implementează o bază de date de dimensiune mică,rapidă,cu fiabilitate înaltă, completă. Acest tip de bază de date este simplu de administrat și simplu de folosit, fiind o modalitate eficientă de a stoca datele de care vom avea nevoie in acest proiect. Pentru a folosi SQLite vom folosi librăria $< sqlite3.h >$, iar pentru a comunica cu baza de date vom folosi : sqlite3_open(...), sqlite3_close(...), sqlite3_free(...) etc.

2.6 Socketuri RAW:

Socketurile RAW sunt o altă opțiune de socket față de $SOCK_STREAM$ și $SOCK_DGRAM$. Acest tip de socket ofera mai multă libertate către nivelurile de jos ale modelului TCP/IP față de restul tipurilor de socket, ale căror antete sunt definite de sistemul de operare. Utilizand RAW socket putem accesa nivelul rețea și nivelul transport, modificând antetul protocolului IP(și ICMP) si antetele protocolului UDP. Dezavantajul socketului RAW este faptul că nu putem folosi protocului TCP în mod direct, din motive de securitate. La acest tip de socket avem două moduri în care ne putem folosi de nivelul transport, unul dintre ele fiind crearea socket-ului utilizând socket(AF_INET, SOCK_RAW,IPPROTO_ICMP)(utilizăm ICMP). Nu putem utiliza socket(AF_INET, SOCK_RAW,IPPROTO_UDP). Pentru a crea un socket care folosește UDP, vom folosi socket(AF_INET, SOCK_RAW,IPPROTO_RAW) și vom implementa noi un protocol asemănător.

3 Arhitectura Aplicației:

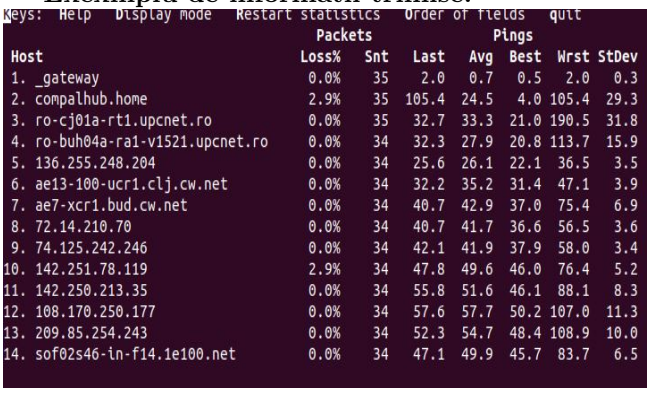
Aplicația va conține două componente principale, Client și Server ce vor comunica utilizand modelul client/server concurent TCP. Acest model este cel mai potrivit pentru comunicarea între cele două componente, deoarece acestea schimbă informație ce nu trebuie pierdută, avem nevoie de o conexiune sigură, orientată.

3.1 Componentele:

Client: Are rolul de a programa serverul cu diverși parametrii, similar executării comenzii mtr în terminal. Acesta va cere afișarea unei rute (ex:de la client la un site web), tipul de pachet cu care se dorește efectuarea traceroute(UDP, ICMP), modalitatea de afișare a numelui HOP-ului (hostname sau/și IP), reafișarea rutei la un anumit timp, a timpilor, TTL. Acesta poate cere afișarea informațiilor în diferite moduri (constant, la un anumit timp).

Server: Are rolul de a verifica informațiile primite de la clienți.După verificare acesta va efectua traceroute(afișarea adreselor IP ale routerelor până la destinație, corespunzător cerințelor clientului și va stoca informațiile în baza de date. La cerere acesta va trimite clienților informațiile cerute la cerere.

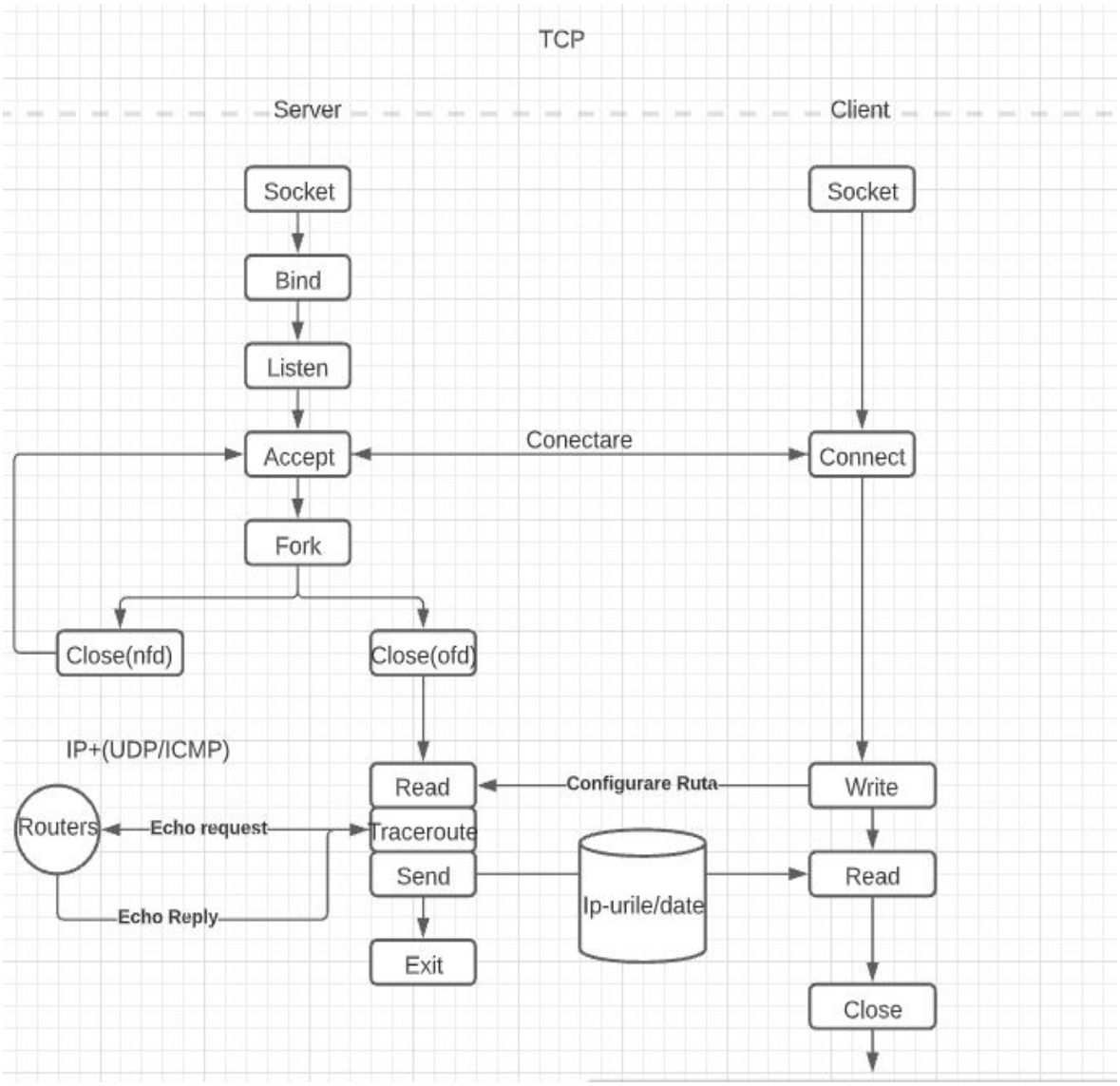
Exexmplu de informatii trimise:



Keys: Help Display mode Restart statistics Order of fields quit

	Packets		Pings				
Host	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. _gateway	0.0%	35	2.0	0.7	0.5	2.0	0.3
2. compalhub.home	2.9%	35	105.4	24.5	4.0	105.4	29.3
3. ro-cj01a-rt1.upcnet.ro	0.0%	35	32.7	33.3	21.0	190.5	31.8
4. ro-buh04a-ra1-v1521.upcnet.ro	0.0%	34	32.3	27.9	20.8	113.7	15.9
5. 136.255.248.204	0.0%	34	25.6	26.1	22.1	36.5	3.5
6. ae13-100-ucr1.clj.cw.net	0.0%	34	32.2	35.2	31.4	47.1	3.9
7. ae7-xcr1.bud.cw.net	0.0%	34	40.7	42.9	37.0	75.4	6.9
8. 72.14.210.70	0.0%	34	40.7	41.7	36.6	56.5	3.6
9. 74.125.242.246	0.0%	34	42.1	41.9	37.9	58.0	3.4
10. 142.251.78.119	2.9%	34	47.8	49.6	46.0	76.4	5.2
11. 142.250.213.35	0.0%	34	55.8	51.6	46.1	88.1	8.3
12. 108.170.250.177	0.0%	34	57.6	57.7	50.2	107.0	11.3
13. 209.85.254.243	0.0%	34	52.3	54.7	48.4	108.9	10.0
14. sof02s46-in-f14.1e100.net	0.0%	34	47.1	49.9	45.7	83.7	6.5

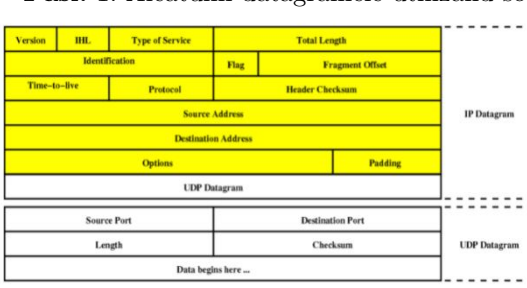
3.2 Diagrama:

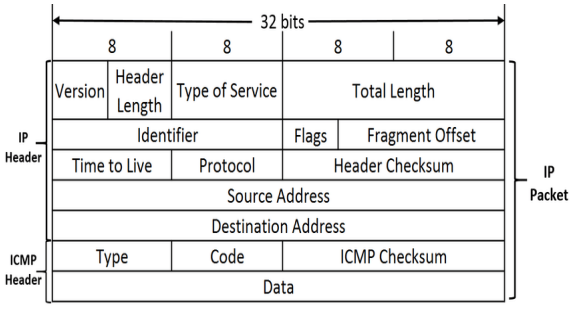


4 Detalii de implementare:

Pentru a obține adresele IP ale routerelor intermediare până la o anumită adresă destinație, vom folosi socket-uri raw ce ne oferă două posibilități de comunicare.
Prima posibilitate de comunicare se bazează pe modelul UDP, neorientat conexiune, nesigur, minimal. Nu avem nevoie de confirmarea primirii pachetelor,de confirmarea transmiterii pachetelor, avem nevoie doar de primirea unui pachet de tip ICMP(Echo Reply), după transmiterea unui pachet (creat utilizând socket-uri raw) pentru a verifica codul protocolului ICMP, pentru a determina problemele din rețea.
A doua posibilitate de comunicare se bazează pe protocolul ICMP(echo request). Utilizand socket-uri RAW vom completa antetul pachetelor IP și al protocolului ICMP, având astfel un pachet pe care îl vom trimite spre destinația sursă, hop cu hop, generând astfel pe parcurs primiri de mesaje tip ICMP(echo reply).

Pasi: 1. Alcătuim datagramele utilizand socket-uri RAW:






```
ip_h->ip_hl = 0;
ip_h->ip_v = 0;
ip_h->ip_tos = 0;
ip_h->ip_len = 0;
ip_h->ip_id = 0;
ip_h->ip_off = 0;
ip_h->ip_ttl = "Numarul care este incrementat periodic";
ip_h->ip_p = IPPROTO_ICMP;
inet_pton (AF_INET, "Adresa de unde se trimite", &(ip_h->ip_src));
inet_pton (AF_INET, "Sursa destinatie", &(ip_h->ip_dst));
ip_hdr->ip_sum = 0
```

```
icmp_h->type = 8;
icmp_h->code = 0;
icmp_h->checksum = 0;
icmp_h->un.echo.id = 0;
icmp_h->un.echo.sequence = "numarul hopului" + 1;
icmp_h->checksum = "suma de control";
```

2. Incrementăm TTL(incepem cu 1) înainte de trimiterea pachetului(echo request) pentru a primi echo reply de la noduri aflate la o distanță din ce în ce mai mare.
3. Astfel vom analiza ruta până la destinația sursă, hop cu hop. În funcție de TTL putem ajunge la destinație sau nu putem ajunge, acest lucru fiind semnalat de câmpurile antetului ICMP din mesajul primit.
4. Vom adăuga datele necesare în baza de date(prezentate de rularea comenzii mtr) (timpul nu se află în acest mesaj, dar poate fi calculat), iar la cerere din partea clientului vor fi trimise.

4.1 Cod:

```
int hostname_to_ip(char *nume_host, char* ip_host) /
{
  struct hostent *x;
  struct in_addr **lista_addr;
  int i;

  if ((x=gethostbyname(nume_host))==NULL)
  {
    //printf("Nu exista acest nume de host");
    return 0;
  }

  lista_addr=(struct in_addr **) x->h_addr_list;

  for (i=0;lista_addr[i]!=NULL;i++)   // pentru un ho
  {
  strcpy(ip_host,inet_ntoa(*lista_addr[i]));
  return 1;
  }
  return 1;
}
```

```
if (interogare==3) //toate
{
  strcpy(comanda,"SELECT hopnumber,numehost,adresaip,timp,mesajhop,packet
  coloane=6;
}
strcat(comanda,numar);
strcat(comanda," ");
strcat(comanda,"and trim(adresaceruta) like '");
strcat(comanda,destinatie);
strcat(comanda,"';");

sqlite3_stmt *stat=NULL;
int r=sqlite3_prepare_v2(database,comanda,-1,&stat,NULL);
//verificare
if (r!=SQLITE_OK)
{
  printf("Eroare la trimiterea mesajelor din baza de date");
}


r=sqlite3_step(stat); //parcurgem pas cu pas interogarea
int tip_coloana;
int randuri=0;
```

```
 header_icmp->checksum=0;
  header_icmp->un.echo.id=0;
  header_icmp->un.echo.sequence=hop+1;
   header_icmp->checksum=check_sum((unsigned short *) (mesaj+20),4); // ver

  //trimitem pachetul
  clock_t timp;
  timp=clock();
  sendto(soc,mesaj,sizeof(struct ip)+ sizeof(struct icmphdr),0, (struct soc

  //primim pachetul
  char mesaj_primit[4096]={0};
  struct sockaddr_in adresa2;
  int lungime= sizeof(struct sockaddr_in);
  return_status=recvfrom(soc,mesaj_primit,sizeof(mesaj_primit),0, (struct s
  fflush(stdout);


  // vom folosi pentru analiza retelei
  struct icmphdr *verificare = (struct icmphdr *) (mesaj_primit+20);
```

5 Scenarii de utilizare:

Aplicatia poate fi utilizata pentru controlul retelei si verificarea problemelor acesteia. Pentru a verifica corectitudinea programului putem compara ruta obtinuta in urma rularii cu o rulare a comenzii mtr

asupra aceeasi destinatii. Timpul de raspuns al celor 2 metode poate varia in functie de tipul de pachet trimis, de momentul in care vom executa, insa ruta va fi aceeasi daca executam relativ in acelasi timp.

6 Concluzii:

Proiectul este un prim pas în diagnosticarea problemelor unei rețele și în arhitectura unei rețele. Traceroute este folosit pe scară largă, iar înțelegerea și implementarea conceptelor oferă o bună înțelegere asupra Internetului. O viitoare imbunătățire a proiectului ar fi găsirea celei mai scurte rute între sursă și destinație, utilizând protocolul OSPF.

7 Bibliografie:

https://profs.info.uaic.ro/~computernetworks/cursullaboratorul.php
https://profs.info.uaic.ro/~georgiana.calancea/rc-home.html
https://www.opensourceforu.com/2015/03/a-guide-to-using-raw-sockets/
https://www.codeproject.com/Articles/18807/Traceroute-Using-RAW-Socket-UDP/
https://www.cnic.ro/telecom/user_datagram_protocol__udp.html/
http://www.tcpipguide.com/free/t_ICMPv4EchoRequestandEchoReplyMessages-2.html/
https://ro.wikipedia.org/wiki/Traceroute/
https://ro.wikipedia.org/wiki/Internet_Protocol/
https://ro.wikipedia.org/wiki/User_Datagram_Protocol/
https://ro.wikipedia.org/wiki/Transmission_Control_Protocol/
https://ro.wikipedia.org/wiki/Internet_Control_Message_Protocol/
https://www.metaswitch.com/knowledge-center/reference/what-is-open-shortest-path-first-ospf