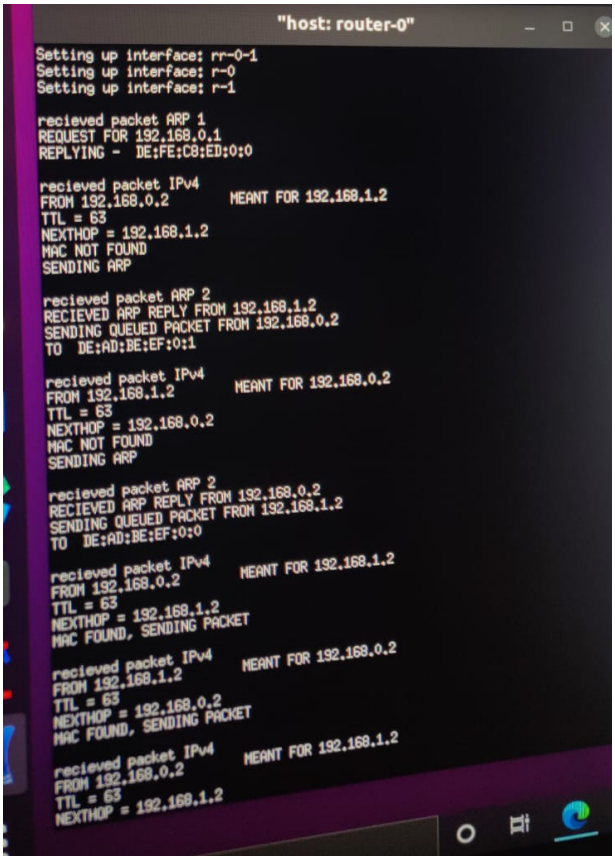
Explicații temă PCOM

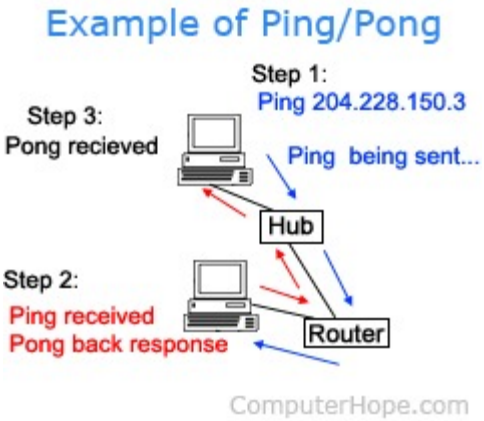
sfaturi de inceput : cititi enuntul de vreo 2 ori

: **puneti foarte foarte multe printf-uri (explicite) de la inceput**…]

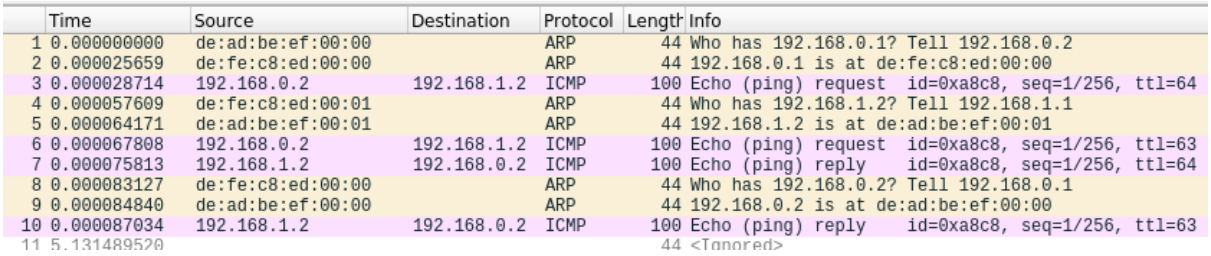
ceva sa arate cam asa



Approach-ul meu ca sa înțeleg / debuguiesc tema a fost să încerc sa fac ping-ul sa functioneze (ping = hostA trimite un semnal la hostB si hostB trimite un semnal inapoi)



Asa arata un ping succesfull daca ne uitam cu wireshark



	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	de:ad:be:ef:00:00		ARP	44	Who has 192.168.0.1? Tell 192.168.0.2
2	0.000025659	de:fe:c8:ed:00:00		ARP	44	192.168.0.1 is at de:fe:c8:ed:00:00
3	0.000028714	192.168.0.2	192.168.1.2	ICMP	100	Echo (ping) request id=0xa8c8, seq=1/256, ttl=64
4	0.000057609	de:fe:c8:ed:00:01		ARP	44	Who has 192.168.1.2? Tell 192.168.1.1
5	0.000064171	de:ad:be:ef:00:01		ARP	44	192.168.1.2 is at de:ad:be:ef:00:01
6	0.000067808	192.168.0.2	192.168.1.2	ICMP	100	Echo (ping) request id=0xa8c8, seq=1/256, ttl=63
7	0.000075813	192.168.1.2	192.168.0.2	ICMP	100	Echo (ping) reply id=0xa8c8, seq=1/256, ttl=64
8	0.000083127	de:fe:c8:ed:00:00		ARP	44	Who has 192.168.0.2? Tell 192.168.0.1
9	0.000084840	de:ad:be:ef:00:00		ARP	44	192.168.0.2 is at de:ad:be:ef:00:00
10	0.000087034	192.168.1.2	192.168.0.2	ICMP	100	Echo (ping) reply id=0xa8c8, seq=1/256, ttl=63
11	5.131489520				44	<Ignored>

am dat ping din h0 la h1 prin router

comenzile sunt:

in router-> “wireshark&”

in router-> ”./router rtable0.txt rr-0-1 r-0 r-1”

in host h-0 -> ”ping h1 -c 1”

(h0 -> deadbeef0000, 192.168.0.2)
(h1 -> deadbeef0001, 192.168.1.2)
(router -> defec8ted0000 si defec8ted0001, 192.168.0.1 respectiv 192.168.1.1)

Observam ca routerul are adrese mac si ip diferite pt fiecare interfata
(o interfata e practic o conexiune cu altcineva)

toate dispozitivele stiu adresele ip ale celorlalte dispozitive
dar nu stiu adresele mac ale acestora.

(De asemenea fiecare are o tabela de rutare din care poate deduce pe unde ar trebuii sa trimita un pachet ca acesta sa ajunga in oricare punct din retea.)

daca analizam un pic ce se intampla in pachetele wireshark:

h0 vrea sa trimita pachet catre h1, stie ca mai intai pachetul trebuie sa treaca prin router
h0 intreaba “care e mac-ul routerului” (ARP REQUEST)
routerul raspunde (ARP REPLY)
h0 trimite routerului un pachet destinat lui h1” (IPV4)
routerul nu stie adresa mac a lui h1, asadar pastreaza pachetul
routerul intreaba “care e mac-ul lui h1” (ARP REQUEST)
h1 raspunde (ARP REPLY)
routerul primeste raspunsul si trimite pachetul de la h0 catre h1 (IPV4)
h1 primeste pachetul si trimite inapoi un “pong” (IPV4)

routerul primeste pachetul dar nu a salvat adresa lui h0 (desi ar fi putut)
intreaba pe h0 care e adresa mac, si apoi trimite pachetul catre acesta

observam ca sunt foarte multe arp-uri pt un singur pachet, insa fiecare dispozitiv are un **cache pt arp unde se vor retine perechi de forma ip - mac** (pentru a nu trebuii sa intrebe de mai multe ori care e adresa mac a cuiva)

bun… ca sa scriem codul usor, am putea sa luam pe rand acesti pasi si sa incercam sa efectuam primele 6 transmisii.

prima data vom rezolva

ARP REQUEST pentru router…

-> atunci cand un host intreaba routerul ce adresa mac are pe o anumita interfata

… inainte de toate, sa vedem ce contine un “pachet”, pachetele sunt primite cu get_packet(&pachet), ele au o interfata pe care au fost primite, o lungime, si un payload (atat)

toate antetele noastre vor fi puse in payload.
de asmeneea, toate mesajele noastre vor contine un header de Ethernet, iar acesta are urmatoarele campuri:
.ether_dhost //mac-ul destinatie
.ether_shost //mac-ul sursa
.ether_type //tipul pachetului

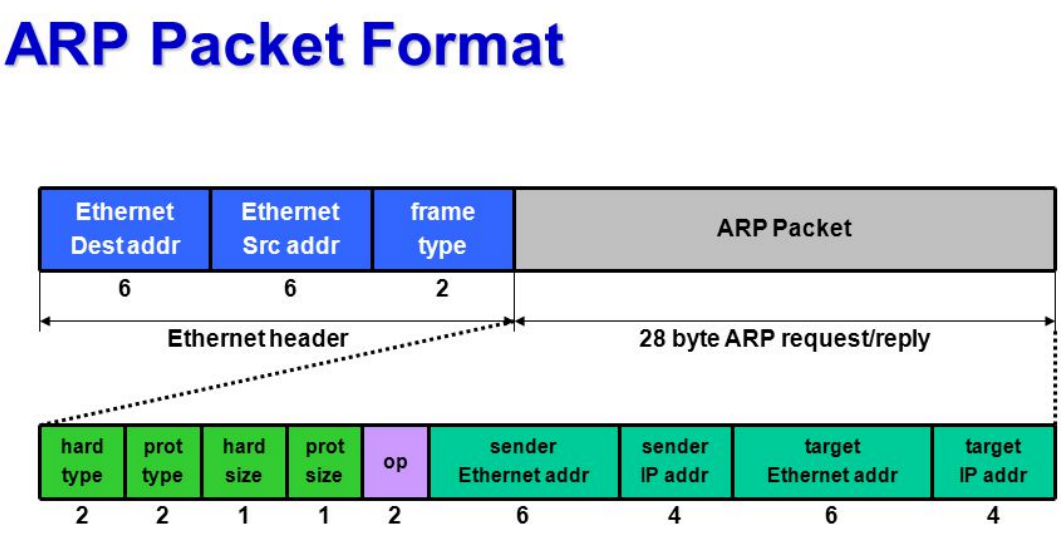
prima data trebuie sa ne asiguram ca pachetele sunt pentru noi (mac-ul destinatie coincide cu mac-ul interfetei pe care am primit pachetul, sau daca mac-ul destinatie este de broadcast (pentru toata lumea) (acel FFFFFFFFF))

/!\ asta ar fi primul pas teoretic, in practica checkerul e facut prost si anumite teste pica fiindca hosturile trimit mac-ul destinatie gresit, deci vom ignora pasul asta
(o sa dea update la checker probabil)

sa incepem cu arp-ul
-> daca ethertype are valoarea pentru ARP (0x0806)

> /!\ aici intervine si problema cu little / high endian…
> reguliile dupa care ma ghidez eu in general sunt urmatoarele
>
> -> daca avem nevoie sa comparam valori din medii diferite, ca aici, unde numarul 0x0806 provine din codul nostru, iar ethertype-ul este un short int provenit din retea…
> va fi nevoie sa transformam unul din numere in celalalt mediu (htons sau ntohs)
>
> -> daca aplicam operatii “aritmetice” asupra valorilor provenite din retea, trebuie mai intai sa le trecem in little endian (cu ntohl sau ntohs), pentru ca sa facem bine calculele (atentie, prin operatii aritmetice ma refer inclusiv la > < etc)
>
> -> daca vrem doar sa verificam egalitatea intre chestii din acelasi mediu nu e necesara nici o prelucrare

putem extrage headerul arp acum… (cam la fel cum am extrage un header ip, este scris codul in enunt pt ip)...



iar acest header arp are rumatoarele informatii:
.op -> tipul operatiei (REQUEST cu val 1 / REPLY cu val 2)

//astea sunt relevante doar cand facem noi un pachet
.htype -> tipul adresei hardware (MAC / Ethernet -> valoarea 1)
.hlen -> lungimea adresei hardware (6)
.ptype -> tipul adresei de protocol (IP -> val 0x0800)
.plen -> lungimea adresei de protocl (4)

.sha -> adresa mac a sursei (cine a trimis pachetul, reply-ul va fi trimis aici)
.tha -> adresa mac a “tintei” (cui ii este destinat pachetul sau gol in caz de request)
.spa -> adresa ip a sursei (ca sha)
.tpa -> adresa ip a “tintei” (adresa destinatarului / adresa cui incercam sa ii aflam mac-ul)

pentru un arp request destinat routerului, cel care a trimis pachetul stie toate informatiile mai putin adresa de mac a routerului, deci noi va trebuii sa punem in **sha** adresa mac corespunzatoare interfetei pe care a fost primit request-ul.

ca sa trimitem inapoi un reply, nu este nevoie sa generam un pachet de la 0 (asta e o tema recurenta , in general doar vom schimba niste lucruri si vom trimite pachetul mai departe)

pe langa actualizarea sha-ului (singurul camp necunoscut), trebuie sa mai facem niste schimbari pentru a transforma requestul intr-un reply

-> op devine codul pentru reply
-> tpa, tsa vor fi adresele destinatarului (entitatea care a dat requestul)
-> spa devine adresa sursei (adresa ip a interefetei de la router)

de asmeenea nu trebuie sa uitam sa modificam si in headerul de ethernet sursa si destinatarul (sursa fiind routerul acum, destinatarul fiind calculatorul)

(practic am inversat mai toate campurile pentru a trimite un reply, inapoi in punctul din care a venit requestul).

Odata implementat codul pt primirea unui arp request, putem testa in wireshark, iar h0 ar trebuii sa inceapa sa ne trimita si pachete ip.

FORWARD Pasul 1

urmatoarea etapa din ping era sa dam forward la ping-ul de la h0 spre h1…
(acesta este un pachet de tip IP)

chestia asta e descrisa bine in enunt, si este implementata si in laborator.

// legat de tabela de rutare, aceasta poate fi citita luand argumentul 1 al functiei main si folosind functia din schelet pt parsare… apoi putem cauta in ea o intrare cu prefixul bun si cea mai lunga masca (initial recomand sa faceti asta liniar pentru a fi mai usor)

singura problema este ca , noi acum avem next-hop-ul , dar nu avem de unde sa stim adresa mac a acestuia… asadar **va trebuii noi sa facem un ARP REQUEST**, pentru a afla mac-ul next-hop-ului.

inainte de orice este necesar sa salvam pachetul nostru IP, pentru a-l putea trimite mai tarziu, cand eventual vom primii reply-ul cu mac-ul cerut.

pentru a trimite un arp request,

facem un pachet nou
ii punem headerul de ethernet (cu destinatia de broadcast, ca toata lumea sa auda “întrebarea”)
ii punem headerul de arp si umplem toate campurile cu ce trebuie (target-ul va fi next-hop-ul, sursa va fi interfata ruterului)
la final ii dam send, si putem trece la bucla urmatoare in while pentru a astepta un reply

intre timp noi mai putem primii si alte pachete ip, alte request-uri, orice altceva practic… asta e rolul acelei cozi, sa putem pastra pachetele care pentru care nu am primit un reply necesar.

primire ARP REPLY

daca detectam un arp reply, vom adauga perechea spa / sha in cache (sha, spa sunt adresele celui care a dat reply-ul)

si apoi vom parcurge coada de pachete “netrimise” pana cand gasim pachetul care trebuia sa ajunga la spa (atentie, ca next-hop, nu neaparat ca destinatie)

odata gasit (atentie pot fi mai multe pachete care asateptau sa fie trimise la aceiasi destinatie, sau chiar 0 pachete), vom adauga in ether_dhost-ul pachetului netrimis adresa mac primita prin arp, iar apoi vom trimite pachetul…

cam astia sunt toti pasi pentru a efectua un ping cu succes (si sa vedem pe wireshark totul frumos), de asemenea ar trebuii sa dea cel putin o parte din teste

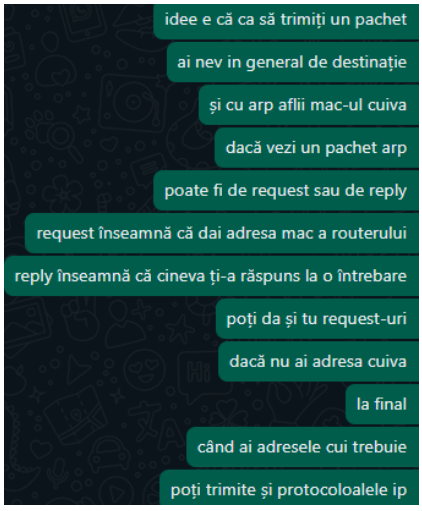
/// completez mai tarziu informatiile pentru ICMP

FAQ

-> in ce ordine ar trebuii abortata tema

routerul sa poata da arp reply (la arp request primit pentru router),
ip recieve,
cautare next-hop,
arp request,
sa primeasca arp reply si sa trimita pachetul mai departe

-> pe scurt, ce presupune ARP

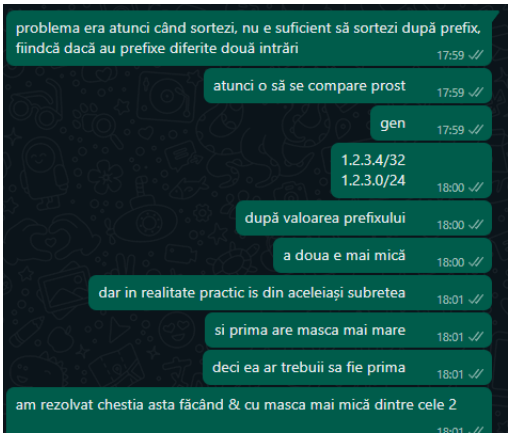


+ recomand video-ul de la ben eater

-> La ARP ruterul trimite Request-uri și primește Reply-uri sau se poate sa primească Request-uri la rândul lui?

poate sa primeasca si requesturi pentru adresa lui mac, se intampla asta initial (asa cum am descris)

-> nu merge cautarea binara desi am sortat bine dupa prefix / masca



-> nu merg testele forward20……

pica fiindca checkerul trimite prost niste destinatii ethernet… nu treceti la bucla urmatoare daca aveti un pachet cu daddr gresit.

posibil sa pice daca cautarea e facuta liniar

-> routerul 1 primeste unele mesaje de la el insusi

nu stiu care e cauza, dar pot fi evitate problemele daca va asigurati ca mac-ul sursa si destinatie nu coincid

-> icmp pare bine dar tot pica

nu uitati sa actualizati tot_len din ipheader si lungimea pachetului

asigurati-va ca cei 64 de octeti sunt copiati bine

ipchecksum se face de la icmp header pana la finalul payload

nu uitati sa setati protocolul din ipheader la codul pt icmp

-> Toate rezolvarile se fac in router.c in cadrul instructiunii while care primeste pachete?

Da, cu exceptia anumitor operatii care se efectueaza o singura data (ex sortarea tabelei de rutare, initializari structuri de date etc)

-> Cum compar doua adrese de ethernet?

for de la 0 la <=5 si testăm octet cu octet dacă e vreunul diferit.