Lucrarea de laborator №1

Construirea topologiilor logice de reţea cu Cisco Packet Tracer şi studierea procesului de transmitere a pachetelor de date în reţea

Scopul lucrării constă în formarea unor abilităţi practice de construire a topologiilor logice de reţea cu Cisco Packet Tracer si ilustrarea modului de functionare al switch-urilor, routerelor şi a protocolului ARP

Obiective:

- Construirea unei reţele simple în spaţiul de lucru pentru topologia logică
- Configurarea dispozitivelor din reţea
- Verificarea conexiunii între dispozitivele din reţea, folosind comanda ping
- Stabilirea numărului de hop-uri pe traseul de la sursă la destinaţie, folosind comanda tracert
- Ilustrarea procesului de completare şi aplicare a tabelului MAC la switch-uri
- Ilustrarea procesului de completare şi aplicare a tabelului ARP la host-uri
- Ilustrarea procesului de completare şi aplicare a tabelului ARP la router

1. Instalarea programului Cisco Packet Tracer

Pentru a instala *Cisco Packet Tracer* se vor urma paşii:

1. Creaţi un cont Cisco, accesând link-ul https://identity.cisco.com/ui/tenants/global/v1.0/enrollment-ui
2. Introduceţi datele contului Cisco, completând câmpurile formularului din fereastra încărcată
3. Accesaţi email-ul indicat la formarea contului Cisco pentru a activa contul
4. Accesaţi site-ul https://www.netacad.com şi logaţi-vă (în dreapta sus este Login-ul)
5. În meniul **Resources** alegeţi opţiunea *Download Cisco Packet Tracer*
6. În partea de jos a ferestrei încărcate daţi un click pe link-ul către pachetul de instalare, în dependenţă de sistemul de operare al dispozitivului pe care îl utilizaţi.
7. Instalaţi Cisco Packet Tracer. La deschiderea programului, introduceţi e-mailul şi parola care au fost indicate la crearea contului Cisco.

2. Conectarea a două host-uri în reţeaua locală

Din End Devices selectaţi două calculatoare PC-PT (staţii de lucru) şi plasaţi-le în zona de lucru. Din Network Devices selectaţi un switch (comutator) de model Cisco Catalyst 2950-24 pe 24 de porturi şi plasaţi-l în zona de lucru (a se vedea Figura 1).

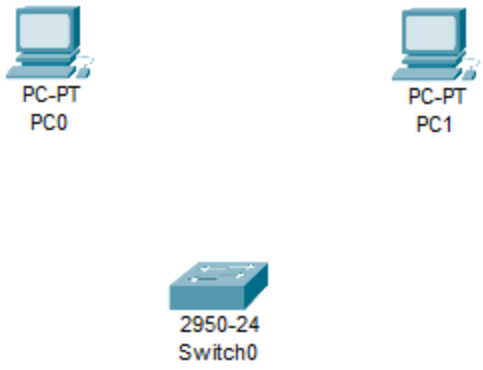


Figura 1

În continuare, conectăm calculatoarele cu switch-ul. Pentru aceasta din Connections (pictograma sub formă de fulger) selectăm linia de comunicaţie automată - de asemenea, pictograma sub formă de fulger (a se vedea Figura 2).

Figura 2

Ţinând apăsată tasta CTRL, faceţi clic pe dispozitivele pe care doriţi să le conectaţi (vezi Figura 3). La final, apăsaţi tasta Esc pentru a ieşi din modul de conectare.

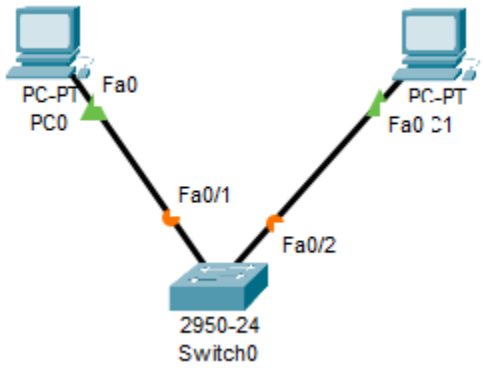


Figura 3

Aşteptăm până când cercurile de culoare portocalie se transformă în triunghiuri verzi - conexiunea este stabilită (vezi Figura 4).

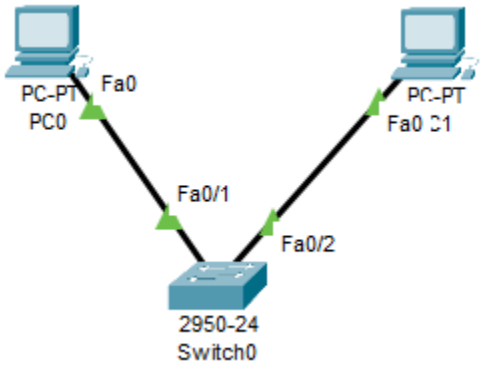


Figura 4

Figura arată numele Fa0 al interfeţelor plăcilor de reţea ale calculatoarelor PC0 şi PC1, precum şi numele Fa0/1 şi Fa0/2 ale porturilor switch-ului. Dacă aceste nume nu sunt vizibile, atunci acestea pot fi activate, bifând caseta de selectare Always show port labels in logical workspace din meniul Opţiuni, submeniul Preferences -> Interface. Deşi am conectat calculatoarele PC0 şi PC1, pentru a putea trimite date între ele, trebuie să setăm adresele IP pe PC0 şi PC1. Pentru a face acest lucru, daţi un click pe pictograma calculatorului PC0 şi în fereastra care se deschide, selectaţi Config şi apoi interfaţa plăcii de reţea FastEthernet0 (vezi Figura 5).

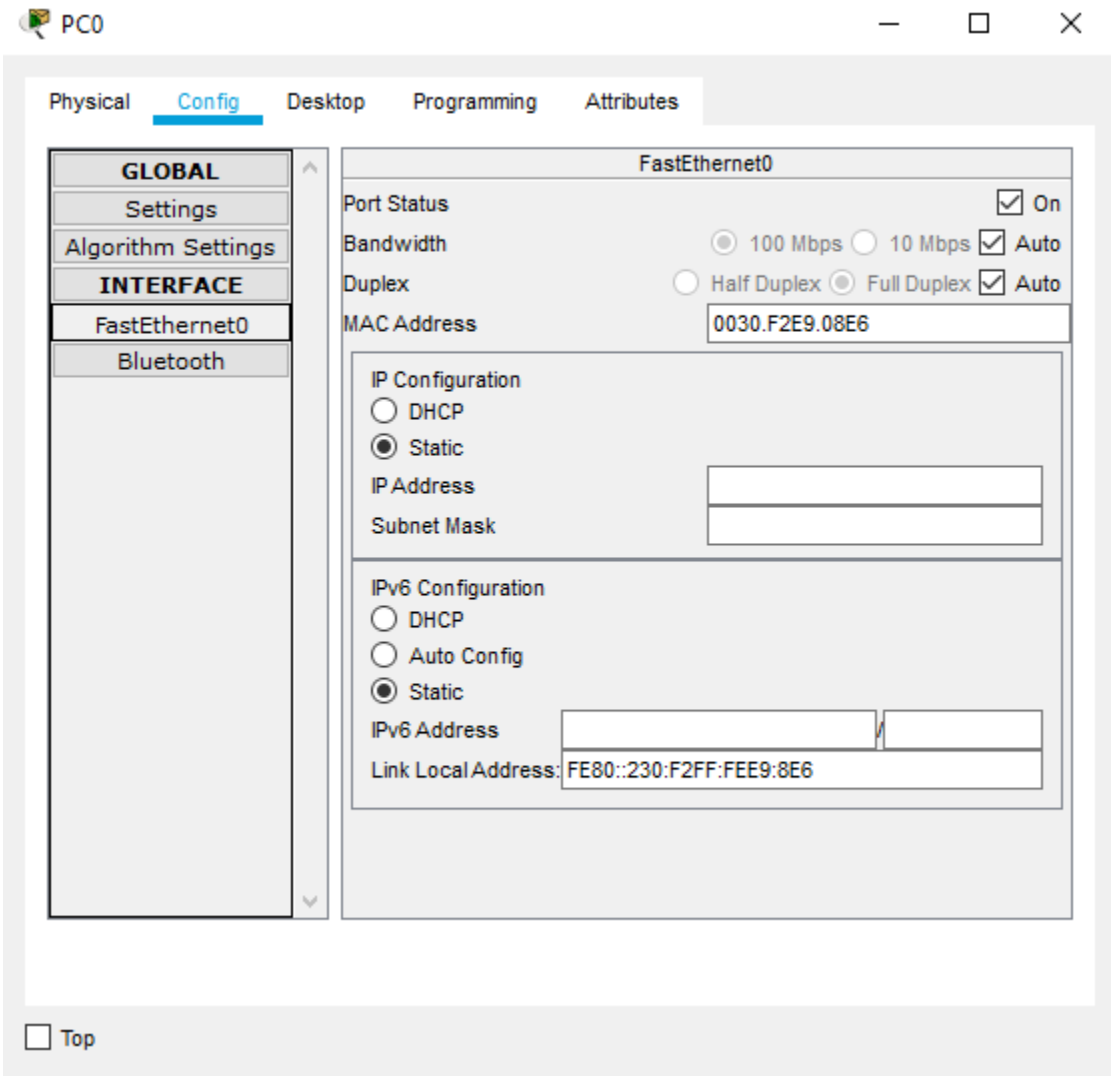


Figura 5

Completăm manual câmpul IP Address cu valoarea 192.168.0.1 şi dăm un click pe câmpul de la Subnet Mask - acolo va fi generată automat masca de subreţea 255.255.255.0. Închidem fereastra.

În acelaşi mod, configurăm al doilea calculator PC1, atribuind acestuia adresa IP 192.168.0.2.

Pentru a verifica dacă există conexiune între PC0 şi PC1, trimitem un ping de la PC0 la PC1. Pentru a face acest lucru, dăm un click pe PC0 -> Desktop -> Command Prompt şi culegem în linia de comandă ping 192.168.0.2.

Apăsând Enter, vom vedea peste ceva timp că au fost trimise şi primite 4 pachete de date, ceea ce dovedeşte că există conexiune cu PC1. În mod similar, verificăm dacă calculatorul PC0 este accesibil de la calculatorul PC1.

Adăugăm încă un dispozitiv în reţeaua noastră - selectăm un Server din End Devices şi îl conectăm la switch (vezi Figura 6). De asemenea, setăm adresa IP 192.168.0.3.

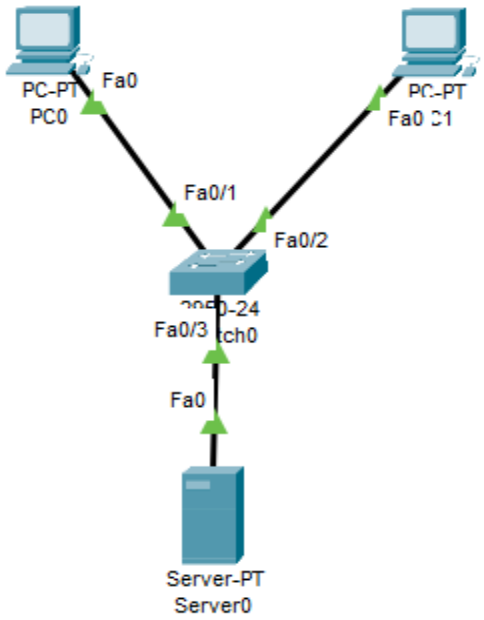


Figura 6

Datele dintre host-uri sunt transmise prin switch, iar topologia reţelei rezultată este o topologie stea. Dacă datele pot fi transmise în ambele direcţii în acelaşi timp, atunci cu ajutorul unui switch se evită coliziunile = semnale suprapuse în canalul de transmisie = datele sunt distruse.

Datele trimise de calculator ajung la un anumit port al switch-ului, mai exact în buffer-ul portului de intrare. Apoi, switch-ul determină portul de ieşire şi transferă datele în buffer-ul acestuia. Dacă canalul corespunzător portului dat este liber, atunci datele sunt trimise la calculatorul corespunzător. Dacă datele de la două calculatoare au ajuns în acelaşi buffer => atunci acestea sunt stocate acolo separat şi sunt trimise pe rând => astfel nu au loc coliziuni. Mai înainte au existat dispozitive numite hub-uri (concentratoare) -> nu aveau memorie buffer -> de aceea aveau loc coliziuni.

Dacă reţeaua implică doar host-uri care sunt conectate prin switch-uri, atunci o astfel de reţea se numeşte domeniu broadcast (sau domeniu de difuzare).

Datele care sunt transmise în limitele unui domeniu broadcast se numesc frame-uri.

Dacă reţeaua este compusă din două sau mai multe domenii broadcast => atunci acestea sunt conectate printr-un router.

3. Protocolul ARP

Adresarea IP funcţionează indirect într-un domeniu broadcast. De bază aici este adresa MAC.

Când transmitem un frame de la PC0 la server => frame-ul este transmis în baza adresei MAC a serverului, şi nu în baza adresei IP acestuia.

Adresa MAC este stocată în memoria echipamentelor de reţea de către producător.

Folosind protocolul ARP => adresa IP a serverului este convertită la o adresă MAC.

Un switch obişnuit nu poate opera cu adrese IP, ci lucrează doar cu adrese MAC.

În baza unui mecanism, switch-ul determină pe care port să transfere frame-ul primit.

Mecanismul asociază fiecărui port al switch-ului o anumită adresă MAC (ce aparţine unui dispozitiv din reţea).

Când un frame vine de la PC0 la switch şi este destinat serverului => switch-ul ştie exact unde să trimită acest frame.

Mecanismul este definit de tabelul MAC al switch-ului

Când switch-ul est conectat în reţea pentru prima dată => tabelul MAC al acestuia nu conţine elemente.

Primul frame primit este direcţionat de switch către toate porturile sale, cu excepţia acelui port din care a venit (procedura este numită unicast flooding).

Imediat ce switch-ul primeşte frame-ul de la PC0, acesta notează în tabelul său MAC că portului Fa0/1 îi corespunde adresa MAC a lui PC0.
Calculatorul către care a fost trimis frame-ul trimite ca răspuns un alt frame.
Când frame-ul răspuns ajunge la switch, acesta scrie adresa MAC a calculatorului care a răspuns în tabelul său MAC şi trimite frame-ul prin portul Fa0/1 către PC0 (a transferat adresa MAC corespunzătoare dintr-un buffer în altul).
Switch-ul a trimis frame-ul de răspuns exact acolo unde era necesar, deoarece în tabelul MAC a găsit că adresa destinatarului se află în spatele acestui port.
Dacă al treilea calculator va trimite un frame, atunci switch-ul va completa tabelul MAC şi va trimite acest frame prin portul corespunzător.
Dimensiunea tabelului de adrese MAC pentru diferite modele variază de la 1000 la 8000 de intrări.
Dacă toate celulele din tabelul MAC sunt deja completate şi switch-ul primeşte un frame de la un dispozitiv pentru care nu are o înregistrare => atunci switch-ul va trimite acest frame la toate calculatoarele şi dacă pentru o perioadă de timp nu există niciun mesaj de răspuns de la vreun calculator, atunci switch-ul va crede că acel calculator la moment nu este funcţional şi va şterge intrarea corespunzătoare din tabelul MAC. Intrarea mai veche este ştearsă, iar cea nouă este scrisă.
Concluzie despre switch-uri
Astfel, switch-ul L2 este utilizat pentru:
1. conectarea host-urilor
2. a preveni coliziunile
3. că foloseşte tabelul de adrese MAC, care permite transferul frame-ului de pe un port pe altul necesar => reduce încărcarea reţelei

În colţul din dreapta jos al Cisco Packet Tracer există un buton de trecere în modul Simulation (Modelare) => dăm un click pe butonul Simulation => am trecut în acest mod
Apăsăm pe pictograma PC0 şi selectăm Command Prompt -> ping 192.168.0.2
S-au format două pachete la PC0 (ICMP şi ARP), dar nu avem dinamică => se aşteaptă apăsarea butonului de trecere dintr-o stare în alta 
În primul rând, este trimis un pachet ARP - astfel, PC0 încearcă să afle adresa MAC a calculatorului cu acest IP: 192.168.0.2.
Să vedem ce se întâmplă cu pachetul de date când facem clic pe butonul Capture/Forward => pachetul trece pe switch.
La următoarea apăsare a butonului Capture/Forward, pachetul este trimis către PC1 şi server, dar serverul vede că pachetul nu este destinat lui (o altă adresă MAC) şi îl respinge (un x pe pachet).
La următoarea apăsare a butonului Capture/Forward, PC1 trimite un răspuns care vine la switch.
Pachetul este direcţionat către PC0 şi nu către server, deoarece switch-ul a învăţat adresa MAC a lui PC0.
În Cisco Packet Tracer, switch-ul poate fi configurat => putem accesa interfaţa sa şi să vedem tabelul MAC al acestuia. Dăm un click pe pictograma switch-ului şi selectăm modul CLI => acest mod este o emulare a unei conexiuni la terminal (fereastra care a fost deschisă este identică cu fereastra de interfaţă dacă ne-am fi conectat printr-un cablu de consolă la un port special al switch-ului) => ne-am conectat la dispozitiv şi îl putem configura
Switch-ul adevărat Cisco Catalyst are un port special pentru consolă:
În comutator - RJ45; în calculator - un port COM sau USB (a se vedea Figura 7)

Figura 7

În realitate, switch-ul este configurat, folosind o interfaţă web pe calculator (cu calculatorul conectat la switch)
Conexiunea se poate realiza nu numai prin consolă, ci şi prin reţea (de la distanţă, folosind Telnet sau SSH)
Toate configurările sunt realizate, folosind linia de comandă
Deci, am încărcat terminalul din linia de comandă. În continuare ne familiarizăm cu interfaţa liniei de comandă
Pentru a obţine o listă de comenzi disponibile, trebuie să tastăm un semn de întrebare:
Obţinem o listă din 12 comenzi: connect, disable, disconnect, enable, exit, logout, ping, resume, show, telnet, terminal, traceroute
Trebuie să trecem la modul de vizualizare completă, numit mod privilegiat
Scriem comanda enable => vom obţine switch# în loc de switch>
Tastăm semnul întrebării => obţinem o listă cu 28 de comenzi disponibile: clear, clock, configure, connect, copy, debug, delete, dir, disable, disconnect, enable, erase, exit, logout, more, no, ping, reload, resume, setup, show, ssh, telnet, terminal, traceroute, undebug, vlan, write
De ce este necesar modul privilegiat? Când dispozitivul este configurat, nu ne va permite din prima să accesăm modul privilegiat, ci va cere să ne autorizăm prin introducerea unei parole (dacă nu ştim parola, nu ne va permite să intrăm)
exit - ieşire din modul privilegiat
putem scrie în formă prescurtată: de exemplu, în loc de comanda enable putem scrie doar en şi să apăsăm tasta Tab => comanda va fi scrisă integral
La fel, putem scrie doar en şi direct să apăsăm Enter
Vrem să vedem tabelul adreselor MAC (se face din modul privilegiat)
Scriem comanda show şi semnul ? => vom vedea comenzile care pot fi utilizate cu comanda show
switch # show mac-address-table => vedem ce adresă MAC este asociată cu fiecare port al switch-ului:

Vlan	MAC address	Type	Ports
1	0001.63ce.679e	Dynamic	Fa0/1
1	0006.2ae8.96be	Dynamic	Fa0/2

În baza datelor din tabel avem:
0001.63ce.679e - adresa MAC a lui PC0
0006.2ae8.96be - adresa MAC a lui PC1
Documentaţia pentru comenzile Cisco poate fi găsită în fişierul
“Команды Cisco, часть I”
Comanda show mac-address-table int fa 0/2 arată doar interfaţa specificată a tabelului MAC

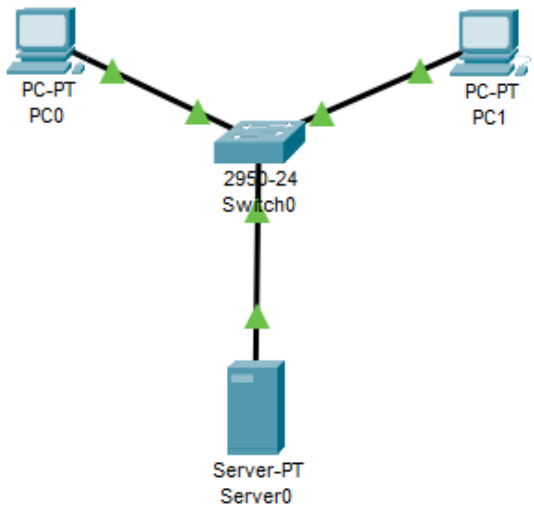


Figura 8

În limitele unui domeniu broadcast, adresarea nu se face prin adrese IP, ci prin adrese MAC
Datele sunt trimise la adrese MAC
Address Resolution Protocol = ARP permite realizarea de corespondenţe între adresele IP şi adresele MAC
Protocolul funcţionează în modul următor:
Când încercăm să accesăm un host după adresa lui IP, se generează un pachet ARP - un mesaj broadcast special, care solicită adresa MAC a host-ului cu adresa IP indicată. Calculatorul cu acest IP răspunde calculatorului iniţial cu adresa sa de MAC. Calculatorul iniţial scrie în tabelul său ARP o intrare cu adresa IP şi adresa MAC corespunzătoare
Tabelul ARP poate fi vizualizat:
În linia de comandă a lui PC0 scriem:

```
arp -a
```

Rezultatul va fi că în acest moment nu există înregistrări în tabel.
Dăm un ping de la PC0 la PC1. Activăm modul Simulation

```
ping 192.168.0.2
```

După ping, se formează două pachete: ICMP şi ARP => dăm un click pe mesajul ARP => o fereastră în care putem vedea că acest mesaj are adresa broadcast: FFFF.FFFF.FFFF ARP Packet
Apăsăm pe Capture/Forward => mesajul ARP este trimis tuturor dispozitivelor din reţea => dar doar un dispozitiv răspunde la acest mesaj - acel cu IP-ul 192.168.0.2.
Şi răspunde exact calculatorului iniţial PC0 (prin switch)
De îndată ce PC0 a primit adresa MAC dorită => îi trimite un pachet de date
Ce s-a întâmplat cu tabelul ARP?

```
arp -a
```

Internet Address	Physical Address	Type
192.168.0.2	0005.5e59.6d45	dynamic

Dacă în continuare vom da ping-uri către calculatorul cu adresa IP 192.168.0.2 => mesajul ARP nu va mai fi generat, deoarece în tabelul ARP al calculatorului ce transmite ping-uri se conţine deja adresa MAC necesară.
În mod similar, dacă se dă de pe PC0 un ping către serverul cu IP-ul 192.168.0.3 => atunci va avea loc un proces analog de determinare a adresei MAC, iar tabelul ARP se va fi completa cu încă o linie (o înregistrare).

Internet Address	Physical Address	Type
192.168.0.2	0005.5e59.6d45	dynamic
192.168.0.3	0080.7049.3a35	dynamic

Dacă dăm comanda *arp -a* pe PC1=>

192.168.0.1	00d0.d394.56eb	dynamic

Atunci când PC0 a trimis o solicitare broadcast de tip ARP către PC1 şi către server, iată atunci calculatorul PC1 a notat în tabelul său ARP adresele IP şi MAC ale lui PC0 (de unde a venit cererea).
În continuare, vom considera o reţea ceva mai complexă, care reprezintă un domeniu broadcast (Figura 1):

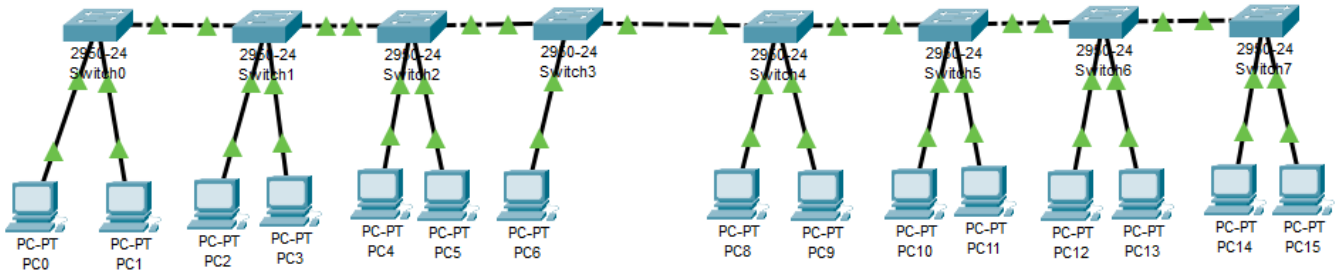


Figura 9

Orice PC poate trimite un frame broadcast şi acesta va ajunge până la fiecare PC din reţea.
Atribuim calculatorului PC0 adresa IP 192.168.0.1, iar lui PC15 - 192.168.0.2.
Pornim modul Simulation
De la calculatorul PC0 dăm un ping către calculatorul PC15: ping 192.168.0.2
Toate calculatoarele primesc mesajul ARP de la PC0 -> rezultă o furtună broadcast;
răspunsul de la PC15 va merge bine
Situaţia va fi foarte neplăcută atunci când în reţea există 1000 de calculatoare şi 500 de switch-uri, ţinând cont că calculatoarele primesc pachete de care nu au nevoie. Pachete broadcast sunt formate şi de multe alte protocoale (nu numai de ARP).

4. Utilizarea routerului

Pentru a limita furtuna broadcast şi dimensiunea domeniului broadcast, există routerele.
În diagrama anterioară vom plasa un router, de exemplu, de modelul 1841:

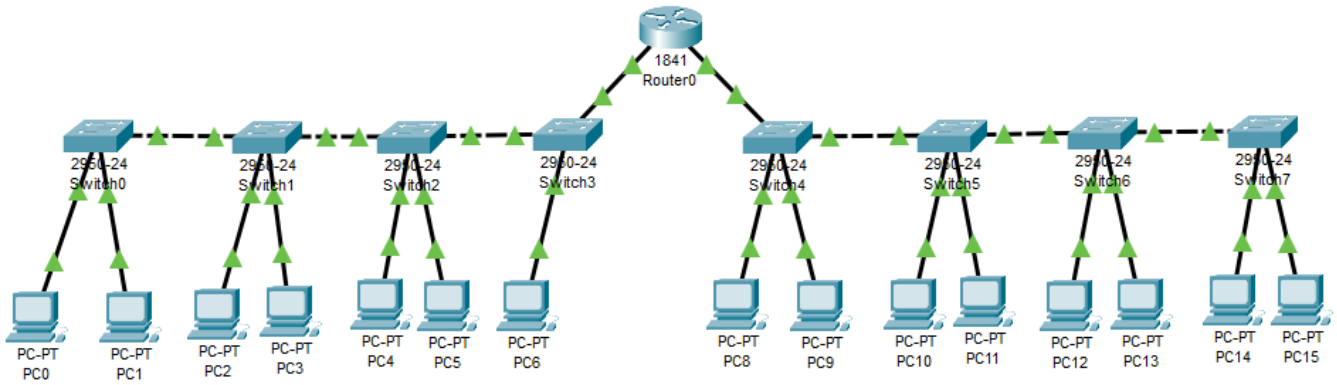


Figura 10

Am obţinut două domenii broadcast. Să arătăm că furtuna broadcast este limitată în limitele unui domeniu broadcast.
Acum PC0 nu poate trimite un mesaj către PC15. Atribuim lui PC0 adresa IP 192.168.0.1, iar lui PC15 - 192.168.0.2. De pe calculatorul PC0 dăm un ping către calculatorul PC15: ping 192.168.0.2. Activăm modul Simulation - pachetul va ajunge la router, dar nu va trece mai departe.
Routerul transferă pachetele de date dintr-un domeniu broadcast în altul => procedura se numeşte rutare de pachete.
Switch-urile (comutatoarele) realizează comutarea de pachete. În switch, în mod implicit, toate porturile sunt conectate. Dimpotrivă, într-un router toate interfeţele sunt dezactivate implicit. Routerul operează cu pachetele, iar switch-ul - cu frame-urile.

Pe interfeţele routerului se vor atribui IP adrese. Vom asigura legătura dintre cele două domenii broadcast prin configurarea interfeţelor routerului. Acest lucru se poate face în două moduri:
Prima metodă: accesăm routerul şi în interfaţa grafică a filei Config dăm un click pe numele FastEthernet0/0 (corespunzător domeniului broadcast din stânga) => setăm adresa IP 192.168.0.1 şi Subnet Mask 255.255.255.0, după care bifăm caseta On de la Port Status (în partea de sus dreapta) -> activăm interfaţa routerului.
Apoi, dăm un click pe numele FastEthernet 0/1 (corespunzător domeniului broadcast din dreapta) => setăm adresa IP 172.20.20.1 şi Subnet Mask 255.255.255.0, după care punem bifa pe On pentru Port Status (în partea de sus dreapta) -> activăm a doua interfaţă a routerului.
A doua metodă:
Accesăm routerul şi trecem în modul CLI (linie de comandă):
În fereastra grafică a apărut o întrebare despre configurarea automată a routerului => selectăm răspunsul No, deoarece vom configura manual routerul
Router> enable => Router #
1. Se atribuie adrese IP pe interfeţele routerului
2. Se activează aceste interfeţe
Comanda conf ter => Router (config) #
Trecem în modul de configurare a interfeţei:

interface fa0/0 => Router(config-if)# ip address 192.168.0.1 255.255.255.0
no shutdown

Configurăm interfaţa fa 0/1 în acelaşi mod. Există două modalităţi de a face acest lucru:
1) se iese cu exit şi se repetă procedura pentru interfaţa fa0/1
2) acolo unde ne aflăm la moment, adică în Router(config-if)#, scriem următoarele:

interface fa0/1 ip address 172.20.20.1 255.255.255.0
no shutdown

Închidem fereastra CLI
Oricărui calculator (de exemplu, PC11) al subreţelei din dreapta îi atribuim un IP, de exemplu, 172.20.20.25 cu masca 255.255.255.0.
Dăm un ping de pe calculatorul PC0 către PC11: ping 172.20.20.25 => nu există conexiune => deoarece trebuie să se specifice adresa routerului implicit (default gateway).
Dacă trebuie să transferăm datele dintr-un domeniu broadcast în altul => se va indica default gateway.
Pe PC0 -> Config -> Gateway: 192.168.0.1 (am indicat IP-ul interfeţei din stânga a routerului nostru)
Pe PC11 -> Config -> Gateway: 172.20.20.1 (am indicat IP-ul interfeţei din dreapta a routerului nostru)
Reamintim că IP-ul de pe PC0 este 192.168.0.2.
Acum, de pe PC0 dăm un ping către PC11: ping 172.20.20.25.
În continuare, vom explica de ce avem nevoie de un default gateway şi ce se întâmplă cu adresele MAC şi solicitările ARP într-o reţea cu mai multe domenii broadcast.
Calculatorul PC0 nu poate ajunge până la PC11 după adresa MAC a acestuia.
Calculatorul PC0 vede că adresa IP a lui PC11 este dintr-o altă subreţea => PC11 se află într-un alt domeniu broadcast => PC0 se adresează către default gateway => pachetul este trimis la router, iar routerul trimite pachetul primit către PC11.
Când scriem ping 172.20.20.25, PC0 trimite o cerere ARP în reţeaua sa (cine are adresa IP 192.168.0.1?) => Ruterul va răspunde lui PC0 cu adresa sa MAC => PC0 va trimite pachetul de date la adresa MAC a interfeţei routerului.
Routerul trebuie să cunoască adresa MAC a lui PC11 => trimite o solicitare ARP în a doua reţea => PC11 îi va răspunde routerului cu adresa sa MAC => routerul va trimite pachetul către PC11 în baza adresei MAC specificate.
Ilustrare: ping 172.20.20.25 de pe PC0 (ignorăm pachetele STP)

Pentru a vedea o informaţie succintă despre interfeţele routerului, dăm comanda:

```
show ip interface brief
```

Pentru a vedea tabelul ARP al routerului, trecem în linia de comandă CLI a routerului şi în modul privilegiat scriem comanda show arp:

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type
Interface
Internet  172.20.20.1             -   0001.6417.4502  ARPA
FastEthernet0/1
Internet  192.168.0.1             -   0001.6417.4501  ARPA
FastEthernet0/0
Router#
```

Figura 11

Dacă traficul de date nu a trecut încă prin router, atunci vom vedea în tabelul ARP - IP-urile interfeţelor routerului şi adresele MAC corespunzătoare acestora (vezi Figura 11).

De fiecare dată când un pachet de date trece prin router, acesta va scrie în tabelul său ARP o linie nouă cu adresele IP şi MAC corespunzătoare ale host-urilor sursă şi destinaţie. De exemplu, dacă dăm un ping către calculatorul PC11 cu adresa IP 172.20.20.25 de pe PC0, atunci în tabelul ARP al routerului obţinem datele prezentate în Figura 12:

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type
Interface
Internet  172.20.20.1             -   0001.6417.4502  ARPA
FastEthernet0/1
Internet  172.20.20.25            0   000B.BE0C.A682  ARPA
FastEthernet0/1
Internet  192.168.0.1             -   0001.6417.4501  ARPA
FastEthernet0/0
Internet  192.168.0.2             0   0030.A3B1.8A2D  ARPA
FastEthernet0/0
Router#
```

Folosind comanda tracert, putem sstabili ruta de la host-ul sursă la host-ul destinaţie (toate routerele intermediare, dacă nu există mai mult de 30). Dacă în Command Prompt pentru PC0 executăm comanda tracert 172.20.20.25, atunci vom obţine rezultatul prezentat în Figura 13.

```
C:\>tracert 172.20.20.25

Tracing route to 172.20.20.25 over a maximum of 30 hops:

  1    0 ms      0 ms      0 ms      192.168.0.1
  2    0 ms      1 ms      0 ms      172.20.20.25

Trace complete.
```

Figura 13

Concluzii despre router:

- Routerul realizează transferul pachetelor IP dintr-un domeniu broadcast în altul.
- Routerele sunt utilizate pentru a limita dimensiunea domeniului broadcast şi pentru a reduce încărcarea în reţea.
- Dacă host-ul sursă nu poate stabili legătură cu host-ul destinaţie prin adresa MAC, atunci host-ul sursă apelează la serviciile routerului implicit (default gateway).

Lucrarea de laborator N2
Reţele locale virtuale VLAN (Virtual LAN)

Scopul principal al acestei lucrări este de a forma abilităţile necesare pentru configurarea reţelelor locale virtuale VLAN.

Obiective:
- A explica conceptul de VLAN şi a ilustra serviciile oferite de tehnologia VLAN în Cisco Packet Tracer
- A arăta modul de configurare a VLAN-urilor în reţelele ce constau din host-uri şi switch-uri
- A arăta modul în care este implicat routerul pentru a asigura comunicarea între VLAN-uri şi accesul la Internet
- A ilustra modul în care este utilizat switch-ul de nivel 3 pentru a asigura comunicarea între VLAN-uri

Tehnologia VLAN
I. Iniţial vom considera următoarea configuraţie de reţea formată doar din host-uri şi switch-uri (a se vedea Figura 1):

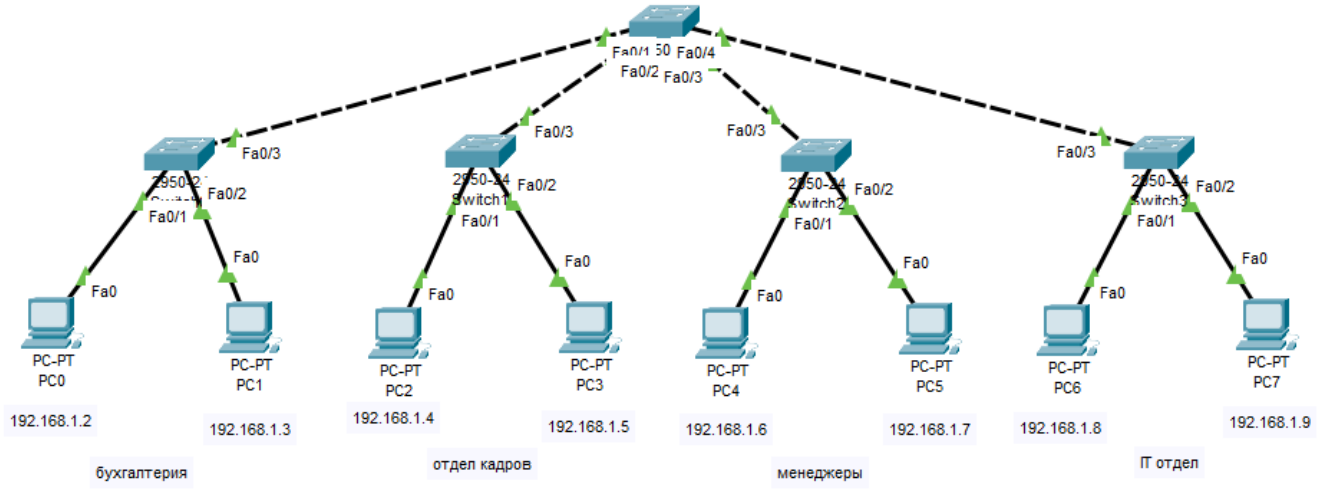


Figura 1

Toate host-urile (din departamente) se află în aceeaşi subreţea 192.168.1.0/24, care formează un domeniu broadcast. Dacă un host din Contabilitate va încerca să contacteze un alt host (de exemplu, host-ul cu adresa IP 192.168.1.9), atunci frame-ul broadcast va fi trimis către toate switch-urile şi către toate host-urile. Dacă trecem în modul de simulare, atunci ne putem convinge că în mare parte switch-urile procesează nu traficul dintre utilizatori (datele utile), ci pachetele broadcast. Numărul mare de pachete broadcast (de la protocoalele ARP, DHCP şi altele) va încărca reţeaua.
Fiecare host generează în mod constant frame-uri broadcast, care ajung la toate host-urile din respectivul domeniu broadcast.
Dacă se vaq întâmpla o furtună broadcast => procesorul switch-ului va procesa frame-urile broadcast prea mult timp şi la un anumit moment reţeaua nu va mai funcţiona.

II. *Schema în care pe fiecare switch avem definit un singur VLAN*
Avem ca scop să formăm grupuri izolate, astfel încât host-urile din Contabilitate (la fel şi cele din departamentul Cadre, departamentul Managerial şi departamentul IT) să schimbe date doar în grupul lor (inclusiv şi pachetele broadcast să rămână în subreţeaua lor), iar host-urile din exterior să nu aibă acces la acestea. Astfel, se urmăreşte ca departamentele la nivelul informaţiei transmise între host-uri să fie izolate între ele.
În acest sens există două probleme:
- În configuraţia actuală, când toate host-urile se află într-o singură reţea, nu există vreun obstacol pentru ca host-urile dintr-un departament să transmită date către host-uri din oricare alt departament (a se vedea Figura 1).

- Din cauza pachetelor broadcast avem canale aglomerate.

=> pentru a depăşi aceste două probleme a fost inventată tehnologia VLAN

a) Prima idee este să plasăm host-urile din departamente în subreţele separate (a se vedea Figura 2):
Contabilitate - subreţeaua 192.168.2.0/24
Resurse umane - subreţeaua 192.168.3.0/24
Administratori - subreţeaua 192.168.4.0/24
Departamentul IT - subreţeaua 192.168.5.0/24

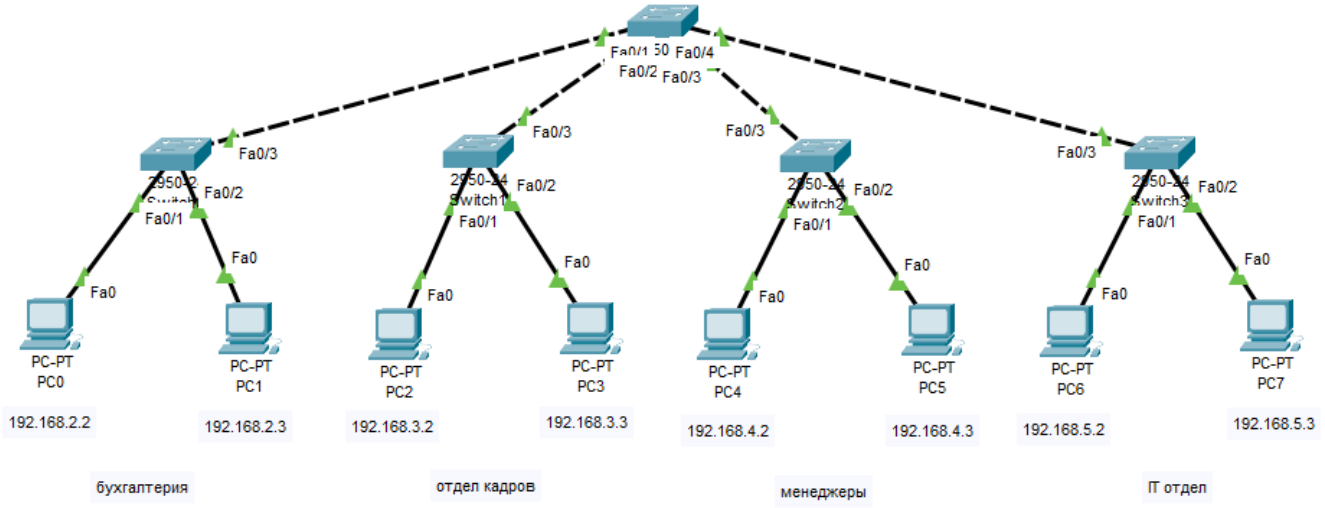


Figura 2

Dacă dăm un ping, de exemplu, de pe host-ul PC0 către PC1, care se află în aceeaşi subreţea, atunci solicitarea ARP broadcast se va răspândi în toate subreţelele. Pachetul broadcast ajunge la Switch 4 (switch-ul central), apoi la Switch1, Switch2 şi Switch3 şi la host-urile care sunt conectate la acestea (treceţi în modul de simulare şi verificaţi acest lucru). Dacă se instalează un router în loc de switch-ul Switch4, atunci se va limita propagarea pachetelor broadcast în limitele subreţelei. Totuşi, utilizarea unui router numai pentru a uni câteva subreţele este iraţională, deoarece în cazul examinat avem nevoie de patru interfeţe pe router, iar modulele pentru interfeţele routerului sunt costisitoare.

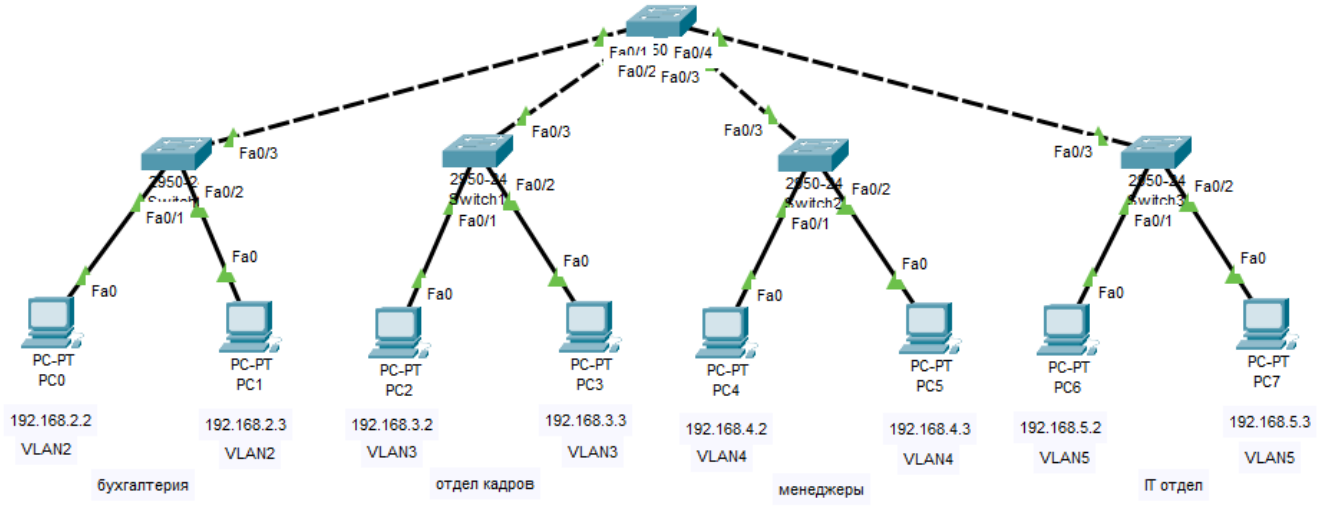


Figura 3

Dacă creăm pentru host-urile din fiecare departament propriul VLAN (a se vedea Figura 3), atunci pachetele broadcast nu vor ieşi din VLAN-ul în care au fost generate. Dacă dăm un ping, de exemplu, de pe PC0 către PC1, atunci solicitarea ARP broadcast, pe lângă PC1, va fi trimisă şi la Switch4, iar de acolo - la Switch1, Switch2 şi Switch3. Acest lucru se datorează faptului că în VLAN 2 pot exista host-uri care nu sunt conectate direct la Switch0, ci prin alte switch-uri. În acest caz, switch-urile Switch1, Switch2 şi Switch3 vor renunţa la pachetele ARP, deoarece nu vor găsi pe porturile lor careva legături la host-urile din VLAN 2.

Ulterior, vom putea înlocui Switch4 cu un router pentru a asigura legătura între VLAN-uri, dar acest

lucru va fi în continuare ineficient, deoarece vor fi utilizate 4 interfeţe de router, al căror cost este semnificativ.
Astfel, aplicarea tehnologiei VLAN asigură:

- Segmentarea logică a reţelei
- Un anumit nivel de securitate
- Limitarea dimensiunii traficului broadcast

Dispozitivele aceluiaşi VLAN aparţin aceleiaşi subreţele => fiecare departament formează un VLAN separat şi o subreţea
Dimensiunea traficului broadcast este în limitele unui VLAN
Traficul broadcast de la primul VLAN nu ajunge în VLAN-urile doi, trei etc. => în acest mod reţeaua este mai puţin încărcată
Host-urile din VLAN2 nu au acces la host-urile din VLAN3, VLAN4 şi VLAN5.
În Figura 3 am examinat schema în care fiecare VLAN este deservit de propriul switch.

Reţeaua virtuală reprezintă un grup de host-uri din reţea al căror trafic (inclusiv pachetele broadcast) de la nivelul legătură de date, este complet izolat de alte host-uri din reţea. Aceasta înseamnă că este imposibilă transmiterea frame-urilor între diferite reţele locale virtuale în baza adresei de la nivelul legătură de date (adresa MAC), indiferent de tipul de adresă - unicast, multicast sau broadcast. În acelaşi timp, în cadrul reţelei virtuale, frame-urile sunt transmise conform tehnologiei de comutare, adică numai către portul asociat adresei MAC a destinaţiei.
Din cele menţionate rezultă că reţeaua virtuală formează un domeniu broadcast.
Scopul tehnologiei VLAN este de a facilita crearea de reţele izolate, care ulterior por fi conectate prin intermediul routerelor, care la nivelul reţea implementează protocolul IP. Acest design de reţea creează bariere serioase în calea traficului broadcast de la o reţea la alta.
Tehnologia reţelelor virtuale creează un fundament pentru construirea unor reţele mari. Astfel, folosind unele aplicaţii software, pe switch-uri sunt configurate VLAN-uri, iar acestea din urmă sunt conectate între ele prin intermediul routerelor.
Aplicarea pe switch-uri a tehnologiei de reţea virtuală VLAN rezolvă problema izolării reciproce a reţelelor, care permite gestionarea drepturilor de acces ale utilizatorilor şi crează bariere în faţa furtunilor broadcast.

III. *Schema în care switch-ul poate gestiona mai multe VLAN-uri* (a se vedea Figura 4)
Pachetul broadcast generat în VLAN2 se va transmite prin switch-uri spre toate host-urile ce se află în VLAN2 (şi doar acolo).

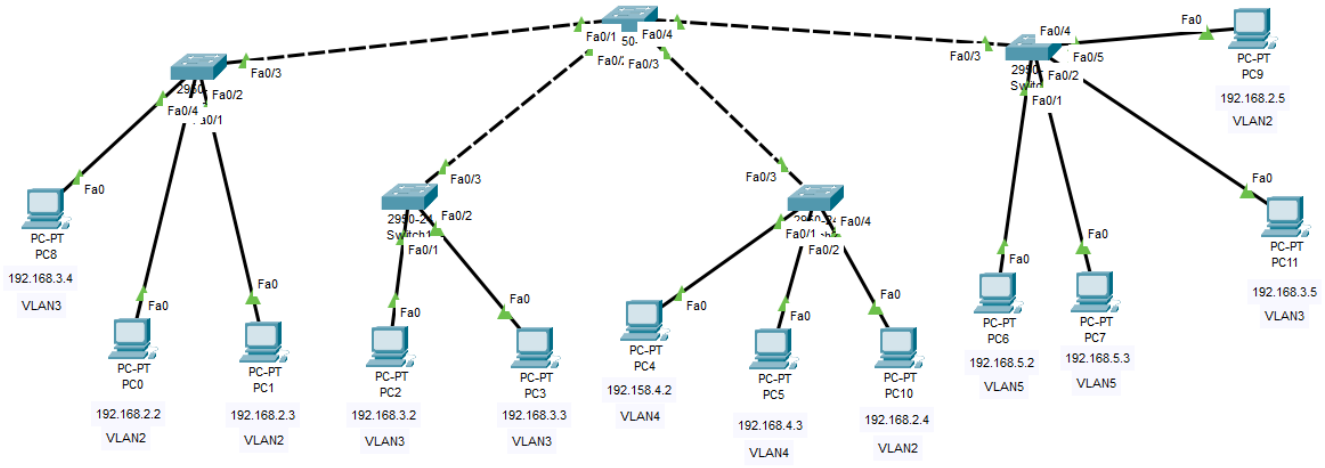


Figura 4

Ceva mai târziu vom reveni şi vom arăta cum sunt create VLAN-uri pentru reţeaua prezentată în Figura 4. Deocamdată, vom analiza unele exemple mai simple.
IV. În primul rând, vom arăta printr-un exemplu de ce sunt necesare VLAN-urile dacă furtuna broadcast poate fi limitată prin divizare în subreţele. Să considerăm configuraţia de reţea din Figura 5.

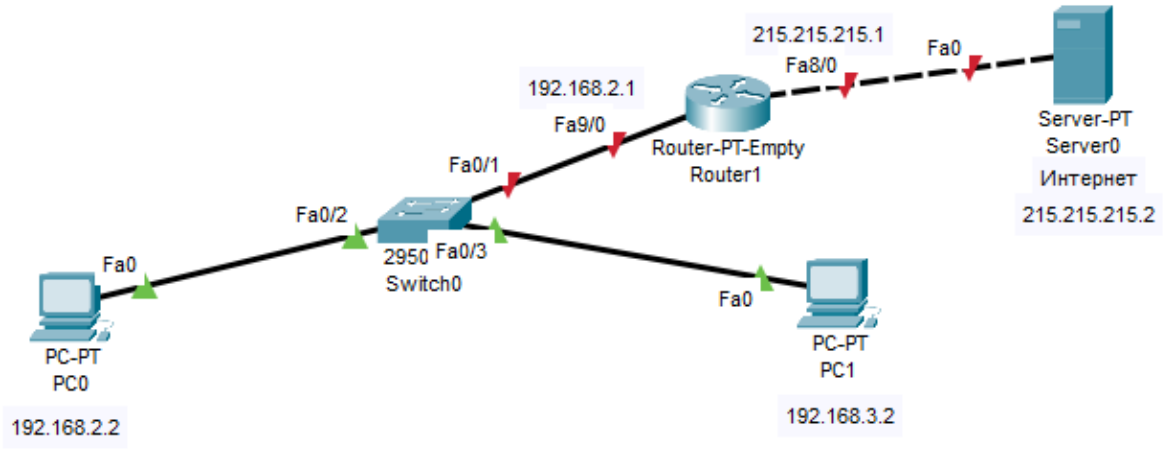


Figura 5

În cadrul rețelei este utilizat un router de model Router-PT-Empty, care presupune adăugarea manuală a modulelor de interfață. Intenționăm să adăugăm 3 module la acest router. Pentru aceasta, dăm un click pe router -> Physical-> Physical Device View și mutăm în dreapta glisorul astfel încât să vedem butonul de oprire/pornire. Dăm un click pe acest buton pentru a stopa funcționarea routerului, după care selectăm, de exemplu, modulul de tip PT-Router-NM-1CFE și îl tragem de trei ori peste sloturile de port. Apăsăm butonul de pornire. După aceasta conectăm deja routerul cu serverul și cu switch-ul.
Pe server setăm IP-ul 215.215.215.2 și adresa de gateway 215.215.215.1.
Pe interfața routerului Fa 8/0 din partea serverului, setăm IP-ul 215.215.215.1, iar pe interfața Fa 9/0 din partea switch-ului - 192.168.2.1 (nu uităm să punem bifa la On - activăm interfața).
Pe host-uri setăm IP-ul 192.168.2.2 și, respectiv, 192.168.3.2.
Apoi, precizăm adresele de gateway pe host-uri. Aici și intervine problema noastră. Pe host-ul cu IP-ul 192.168.2.2 setăm adresa de gateway 192.168.2.1, care este setată și pe interfața corespunzătoare a routerului. Ce gateway ar trebui să instalăm atunci pe PC1? Ar trebui să fie 192.168.3.1, dar pe interfața routerului deja este fixată adresa IP 192.168.2.1 ??????
Există două soluții:
1) se pot utiliza două fire ce conectează două porturi ale switch-ului, corespunzător la două interfețe ale routerului. Pe interfața Fa 9/0 se va seta IP-ul 192.168.2.1, iar pe interfața Fa 7/0 - 192.168.3.1 (a se vedea Figura 6). Pe host-ul PC1 se va seta adresa de gateway - 192.168.3.1.

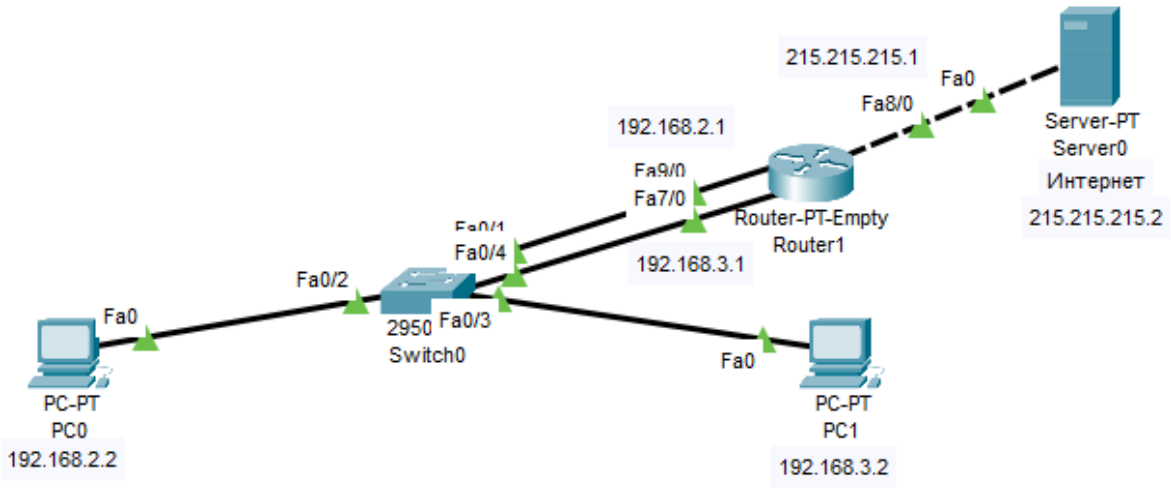


Figura 6

Verificăm conexiunea cu serverul de Internet de pe host-ul PC0 – ping 215.215.215.2 => este conexiune
Verificăm conexiunea cu serverul de Internet de pe host-ul PC1 – ping 215.215.215.2 => este conexiune

Dar totuși există un mare dezavantaj. Dacă switch-ul va conecta, de exemplu, 50 de subrețele => vor fi necesare 50 de cabluri și, cel mai important, 50 de interfețe de router (adică, de fapt, câteva routere). Dacă se ține cont că modulul pentru interfața de router Cisco costă nu mai puțin de 8000 de lei, atunci putem ușor deduce că vom avea cheltuieli semnificative

2) A doua soluție este bazată pe utilizarea VLAN-urilor. Dacă host-ul PC0 este inclus în VLAN2, iar PC1 în VLAN3, atunci în baza unei interfețe fizice de router și a unui singur cablu vom putea accesa Intrenet-ul

V. *Detalii privind configurarea VLAN-urilor*

În continuare vom prezenta modul în care sunt configurate VLAN-urile

În anul 1998 a fost adoptat standardul IEEE 802.1q, care stabilește regulile de bază utilizate în construcția rețelelor locale virtuale VLAN, reguli care nu depind de protocolul de nivel legătură de date utilizat la nivelul switch-ului.

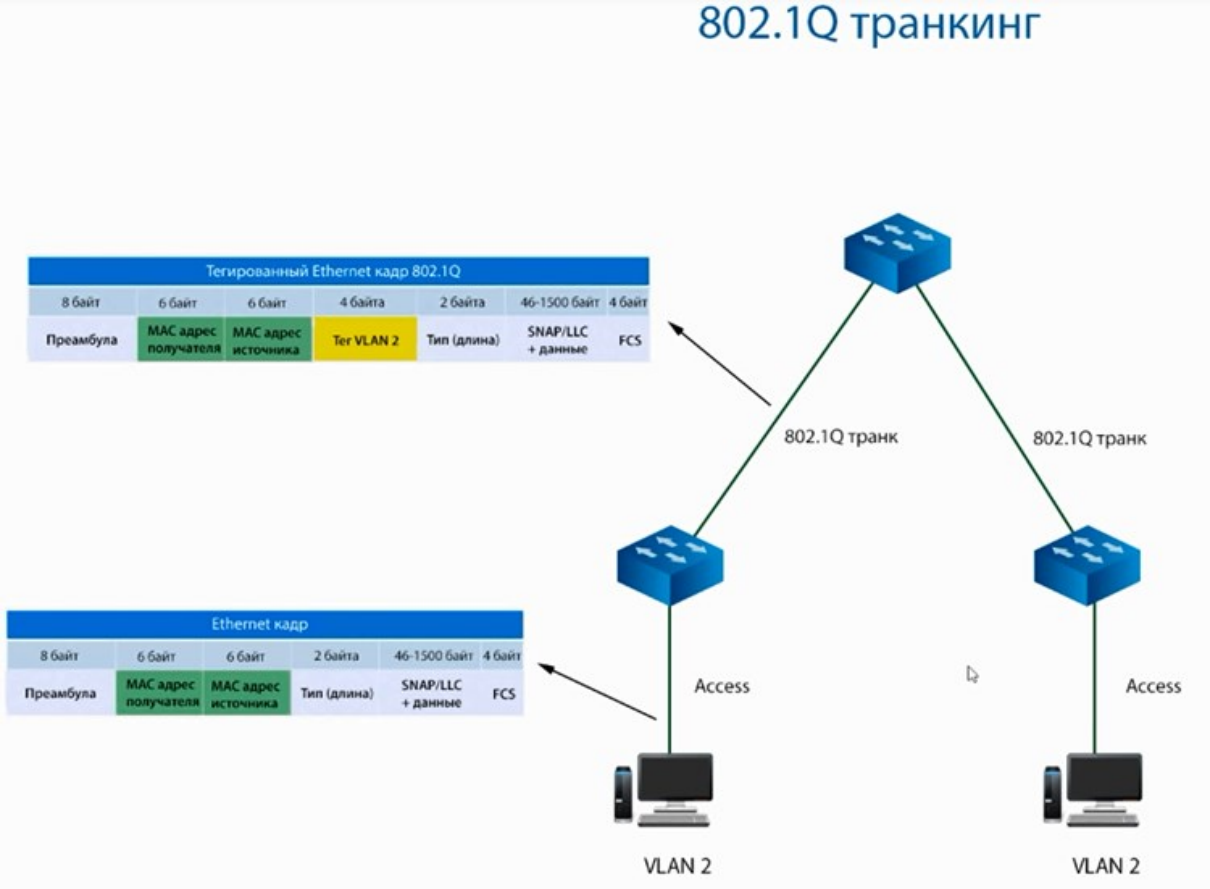


Figura 7

Conexiunea dintre switch și host este numită legătură Access. Conexiunea dintre două switch-uri sau dintre switch și router este numită legătură Trunk. Atunci când un frame trimis de un careva host ajunge pe portul de tip Access al switch-ului, pentru a fi transferat printr-o legătură de tip Trunk, în cadrul frame-ului este inserată o etichetă numită teg. Atunci când frame-ul cu teg ajunge la un switch și urmează a fi transferat printr-o legătură de tip Access, switch-ul elimină teg-ul corespunzător.

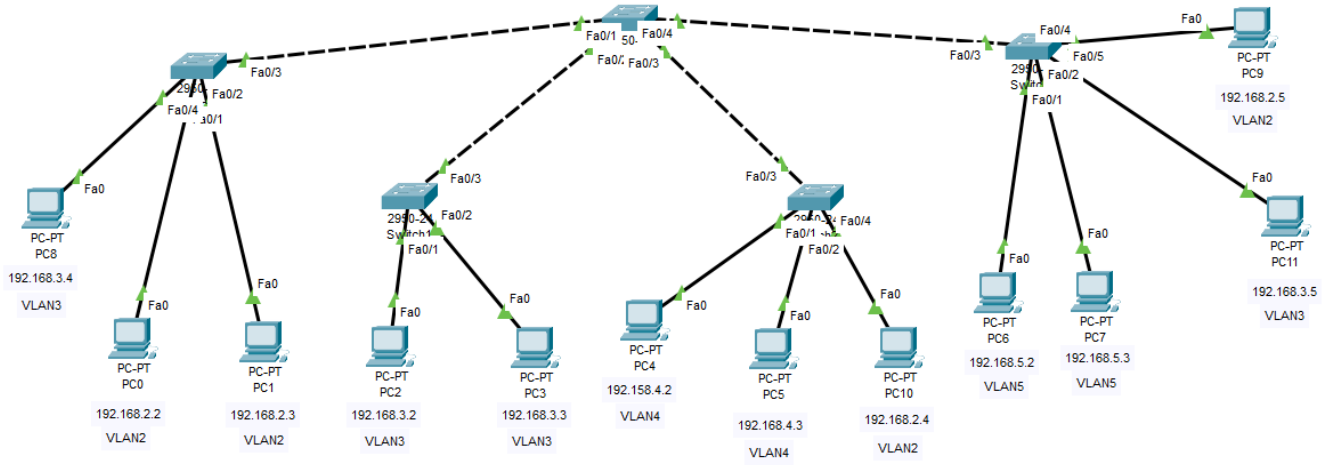
Astfel, prin porturile de tip Trunk se transmit frame-uri cu teg, iar prin porturile de tip Access – frame-uri standard fără teg.

În rețeaua din Figura 7 host-ul din stânga constituie un frame Ethernet standard (host-ul nu știe că se află în careva VLAN) pe care îl transmite switch-ului. Switch-ul stabilește că a recepționat un frame din VLAN2 și formează un frame Ethernet cu teg, pe care îl transmite printr-un port trunk următorului switch, care la rândul său îl transmite fără modificări următorului switch. Următorul switch examinează dacă are dreptul să opereze cu acest frame. Dacă da, atunci switch-ul elimină teg-ul și transmite frame-ul „curățat" către host. Dacă host-ul s-ar fi aflat într-un alt VLAN decât cel indicat în teg, atunci switch-ul ar fi eliminat frame-ul.

În continuare, revenim la rețeaua din Figura 4 și descriem procedura de configurare a VLAN-urilor.

1) Mai întâi setăm pe host-uri IP-urile corespunzătoare.

2) Stabilim legăturile de tip Access între switch-uri și host-uri. Fiecare port al switch-ului îl asociem cu un VLAN concret. Implicit, portul se află în VLAN1 (adică la început toate porturile sunt asociate cu VLAN1). VLAN-ul este creat pe switch și se asociază cu un port.



Accesăm linia de comandă CLI a switch-ului Switch0 şi executăm comenzile următoare:

`Switch>en`	`Trecem în modul privilegiat`
`Switch#conf ter`	`Trecem în modul de configurare globală`
`Switch(config)#int range fa 0/1-2`	`Accesăm interfeţele fa 0/1 şi fa 0/2`
`Switch(config-if-range)#switchport mode access`	`Switch-ul va insera teg-uri în frame-urile recepţionate prin aceste porturi (adică creăm legături de tip Access)`
`Switch(config-if-range)#switchport access vlan 2`	`Se precizează VLAN-ul din care vor face parte host-urile PC0 şi PC1`
`Switch(config-if-range)#exit`	`Ieşim din modul de configurare a interfeţei selectate`
`Switch(config)#int fa 0/4`	`Accesăm modul de configurare a interfeţei fa 0/4 şi executăm aceleaşi comenzi, cu diferenţa că host-ul PC8 va face parte din VLAN3`
`Switch(config-if)#switchport mode access`	
`Switch(config-if)#switchport access vlan 3`	
`Switch(config-if)#exit`	

Pentru a vizualiza legăturile de tip Access stabilite pe switch vom folosi comanda:

```
Switch#show vlan brief
```

În mod analog se configurează legăturile de tip Access pe switch-urile Switch1, Switch2 şi Switch3. În acest mod am stabilit toate legăturile de tip Access.

3) În continuare se stabilesc legăturile de tip trunk dintre switch-ul central (Switch4) şi celelalte switch-uri (Switch0, Switch1, Switch2, Switch3). Pentru aceasta este suficient să se stabilească porturi trunk pe Switch4.

```
Switch>en
Switch#conf ter
Switch(config)#int range fa 0/1-4
Switch(config-if-range)#switchport mode trunk
Switch(config-if-range)#exit
```

Pentru a vizualiza porturile trunk pe un switch se va utiliza comanda

```
Switch# show interface trunk
```

4) Switch-ul central trebuie să ştie identificatorii VLAN-urilor, ai căror frame-uri cu teg acesta urmează să le retransmită. Pentru aceasta creăm pe switch aceste VLAN-uri.

```
Switch(config)#vlan 2
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#exit
Switch(config)#vlan 4
Switch(config-vlan)#exit
Switch(config)#vlan 5
Switch(config-vlan)#exit
Switch(config)#show vlan
```

Pentru a modifica numele VLAN-ului (pentru o identificare mai simplă a acestora) vom folosi comanda *name*:

```
Switch#conf ter
Switch(config)#vlan 2
Switch(config-vlan)#name Buhgalteria
Switch(config-vlan)#exit
Switch(config)#do show vlan
```

Verificaţi dacă există conexiune între PC0 şi PC9:
ping 192.168.2.5 => există conexiune
Examinaţi de sine stătător în modul Simulation cum are loc inserarea teg-ului în frame (urmăriţi conţinutul câmpului TCI în frame-urile ARP, atunci când acestea trec prin link-urile trunk) şi cum are loc eliminarea acestuia

VI. *Asigurarea legăturii dintre VLAN-uri*
Pentru ca să existe posibilitatea de schimb de date între VLAN-uri se va utiliza routerul, iar procedeul este numit Inter VLAN Routing
Deoarece VLAN-urile se află în subreţele diferite => pentru a asigura comunicarea între subreţele este necesar un router
Să examinăm configuraţia de reţea din Figura 8 în care switch-ul central este conectat la un router ce va asigura comunicarea între anumite VLAN-uri

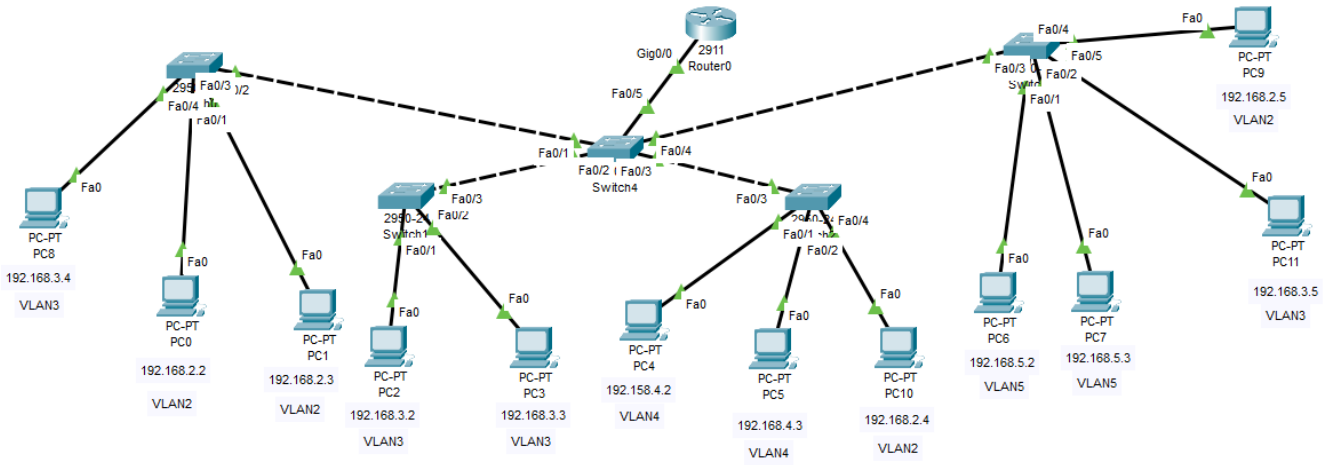


Figura 8

Definim portul Fa 0/5 al switch-ului Switch4 ca un port trunk:

```
Switch>en
Switch#conf ter
Switch(config)#int fa 0/5
Switch(config-if)#switchport mode trunk
Switch(config-if)#exit
```

În continuare, accesăm modul CLI pe router şi configurăm patru subinterfeţe (tot atâtea subinterfeţe câte VLAN-uri avem):

```
Router>en
Router#conf ter
Router(config)#int gig 0/0.2
Router(config-subif)#encapsulation dot1q 2
Router(config-subif)#ip add 192.168.2.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int gig 0/0.3
Router(config-subif)#encapsulation dot1q 3
Router(config-subif)#ip add 192.168.3.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int gig 0/0.4
Router(config-subif)#encapsulation dot1q 4
Router(config-subif)#ip add 192.168.4.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int gig 0/0.5
Router(config-subif)#encapsulation dot1q 5
Router(config-subif)#ip add 192.168.5.1 255.255.255.0
Router(config-subif)#exit
```

Pe interfaţa grafică corespunzătoare lui Gigabit Ethernet 0/0 punem bifa la *On* în drept cu *Port Status*. Lucrul acesta poate fi făcut şi prin următoarele comenzi:

```
Router(config)# interface gigabitEthernet 0/0
Router(config-if)# no shutdown
```

```
Router(config-if)# exit
```
Pe portul Fa 0/5 al switch-ul central se poate preciza lista cu VLAN-urile ce au acces la router (sau de la router), folosind comanda (se scrie în modul de configurare a interfeței Fa 0/5 după *switchport mode trunk*, de exemplu)

switchport trunk allowed vlan 2,3 (se indică VLAN-urile permise => în cazul dat vor trece doar pachetele din VLAN-urile 2 și 3)

Pe fiecare host setăm adresa de router implicit (Default Gateway)
Dăm un ping între două host-uri ce se află în VLAN-uri diferite
Se va examina procesul în modul Simulation (se lasă pentru vizualizare doar pachetele ARP și ICMP)
Dacă se dă ping de la host-ul PC0, care se află în VLAN2, către host-ul PC11, care se află în VLAN3, avem următoarea situație: pachetul ARP ajunge la switch-ul central, care îi permite să treacă prin portul său Fa 0/5 (deoarece vine din VLAN2) la router; routerul îi transmite lui PC0 adresa sa MAC. PC0 îi transmite routerului pachetul de date utile. Routerul formează un pachet ARP în care modifică în 3 eticheta 2 a VLAN-ului (aceasta este scrisă în câmpul TCI – 0x0003) și transmite pachetul în VLAN3 pentru a afla adresa MAC a host-ului destinație. Host-ul destinație transmite routerului adresa sa MAC, iar routerul îi transmite pachetul de date utile.
Dacă se dă ping de la host-ul PC0, care se află în VLAN2, către host-ul PC7, care se află în VLAN5, avem următoarea situație: pachetul ARP ajunge la switch-ul central, care îi permite să treacă prin portul său Fa 0/5 (deoarece vine din VLAN2) la router; routerul îi transmite lui PC0 adresa sa MAC. PC0 îi transmite routerului pachetul de date utile. Routerul formează un pachet ARP în care modifică în 5 eticheta 2 a VLAN-ului (aceasta este scrisă în câmpul TCI – 0x0005) și transmite pachetul în VLAN5 pentru a afla adresa MAC a host-ului destinație. Însă pachetul este stopat de către switch-ul central (în portul Fa 0/5), deoarece este setată permisiune doar pentru pachetele din VLAN-urile 2 și 3.
Astfel, se vede că routerul are capacitatea de a modifica eticheta VLAN-ului (în cazul dat 2 prin 3 (sau prin 5)) și, prin urmare, poate transmite pachetele între VLAN-uri

Lucru individual - Rezolvați problema cu două host-uri din VLAN-e diferite, un switch, un router și un server de internet (a se vedea Figura 6b), astfel încât conexiunea dintre switch și router să se realizeze printr-un singur cablu și o singură interfață de router (trebuie să configurați două subinterfețe pe router).

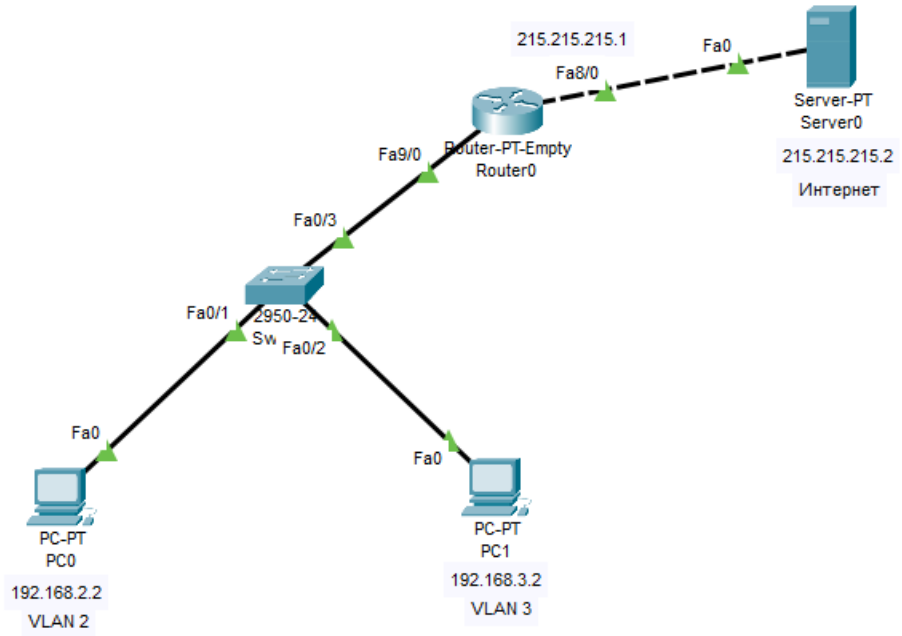


Figura 6b

VII. *Asigurarea legăturii dintre VLAN-uri prin intermediul switch-ului de nivel 3*
Pentru a conecta rețelele virtuale într-o rețea comună este necesar să se implice protocolul IP de la nivelul rețea. Acesta din urmă poate fi implementat printr-un router sau printr-un switch de nivel 3.
Să considerăm configurația de rețea din Figura 9 și să configurăm switch-ul L3.

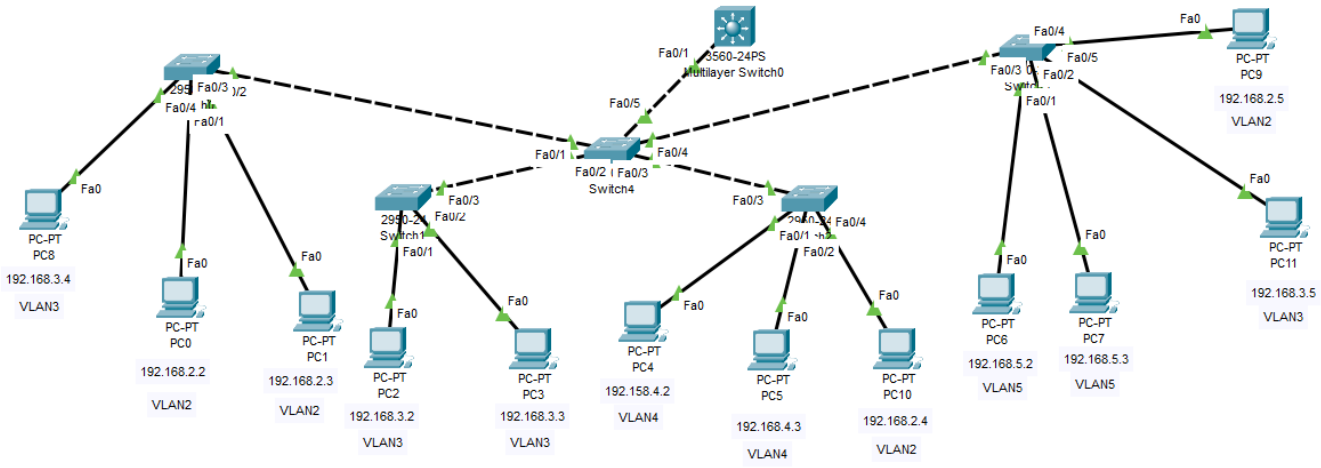


Figura 9

Trecem în modul CLI al switch-ului L3 şi executăm următoarele comenzi:

```
Switch>en
Switch#conf ter
Switch(config)#int fa 0/1
% nu va trece direct switchport mode trunk, deoarece în acest caz există şi ISL
Switch(config-if)#switchport trunk encapsulation dot1q
Switch(config-if)#switchport mode trunk
Switch(config-if)#exit
% делаем VLAN-ы:
Switch(config)#vlan 2
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#exit
Switch(config)#vlan 4
Switch(config-vlan)#exit
Switch(config)#vlan 5
Switch(config-vlan)#exit
% Pentru a atribui adrese IP, în loc de subinterfeţe se generează interfeţe virtuale:
Switch(config)#int vlan 2  % am generat interfaţa virtuală ce va deservi VLAN-ul 2
Switch(config-if)#ip add 192.168.2.1 255.255.255.0
Switch(config-if)#exit
Switch(config)#int vlan 3
Switch(config-if)#ip add 192.168.3.1 255.255.255.0
Switch(config-if)#exit
Switch(config)#int vlan 4
Switch(config-if)#ip add 192.168.4.1 255.255.255.0
Switch(config-if)#exit
Switch(config)#int vlan 5
Switch(config-if)#ip add 192.168.5.1 255.255.255.0
Switch(config-if)#exit
Switch(config)#ip routing
```

De pe host-ul PC0 (care se află în VLAN2) dăm un ping către host-ul PC11 (din VLAN3). În modul Simulation vom vedea că solicitarea ARP este transmisă tuturor host-urilor din VLAN2, inclusiv şi switch-ului L3. Switch-ul L3 îi răspunde lui PC0 cu adresa sa MAC. PC0 îi transmite frame-ul cu date utile switch-ului L3. În continuare, switch-ul L3 transmite o solicitare ARP tuturor host-urilor din VLAN3. Host-ul PC11 îi răspunde switch-ului cu adresa sa MAC. Switch-ul L3 transmite pachetul cu date utile către PC11.

Cum se poate modifica VLAN-ul implicit Native VLAN (VLAN1)? De ce este necesar să fie modificat? – deoarece în pachetele ARP din VLAN1 nu se indică eticheta VLAN-ului, iar acest fapt a fost exploatat prin construcţia unor atacuri asupra Native VLAN, care combat securitatea reţelei

Accesăm portul switch-ului şi dăm comanda:

```
Switch(config)# int fa 0/3
Switch(config-if)# switchport trunk native vlan 50
(de exemplu, 50 => datele din VLAN-ul 50 nu vor fi marcate cu teg)
```

Acelaşi lucru se face şi pe celelalte porturi trunk

Lucrarea de laborator N3
Protocolul STP. Tehnologia Etherchannel

Scopul principal al acestei lucrări este de a ilustra modul de funcţionare al protocolului STP şi al tehnologiei Etherchannel.

Obiective:

- A explica conceptele de agregare, fiabilitate, balansare a traficului
- A arăta modul de funcţionare a protocolului STP şi în special a versiunii actuale PVST+ a acestuia
- A ilustra posibilităţile tehnologiei Etherchannel, care permite folosirea simultană a mai multor link-uri între două dispozitive pentru transmiterea simultană a traficului

Protocolul STP – Spanning Tree Protocol

Vom analiza cum se rezolvă problema formării ciclurilor în reţea, atunci când se construiesc topologii Ethernet fiabile, care să funcţioneze la ieşirea din funcţiune a unor componente vitale ale reţelei, prin inserarea unor linii de comunicaţii suplimentare (redundante).

I. *Pentru ce este necesar STP?*

Cisco recomandă utilizarea următorului model ierarhic pe 3 nivele (a se vedea Figura 1) pentru construcţia reţelelor de dimensiuni mari şi medii (modelul este aplicat chiar dacă reţeaua este de ordinul a câteva sute de host-uri)

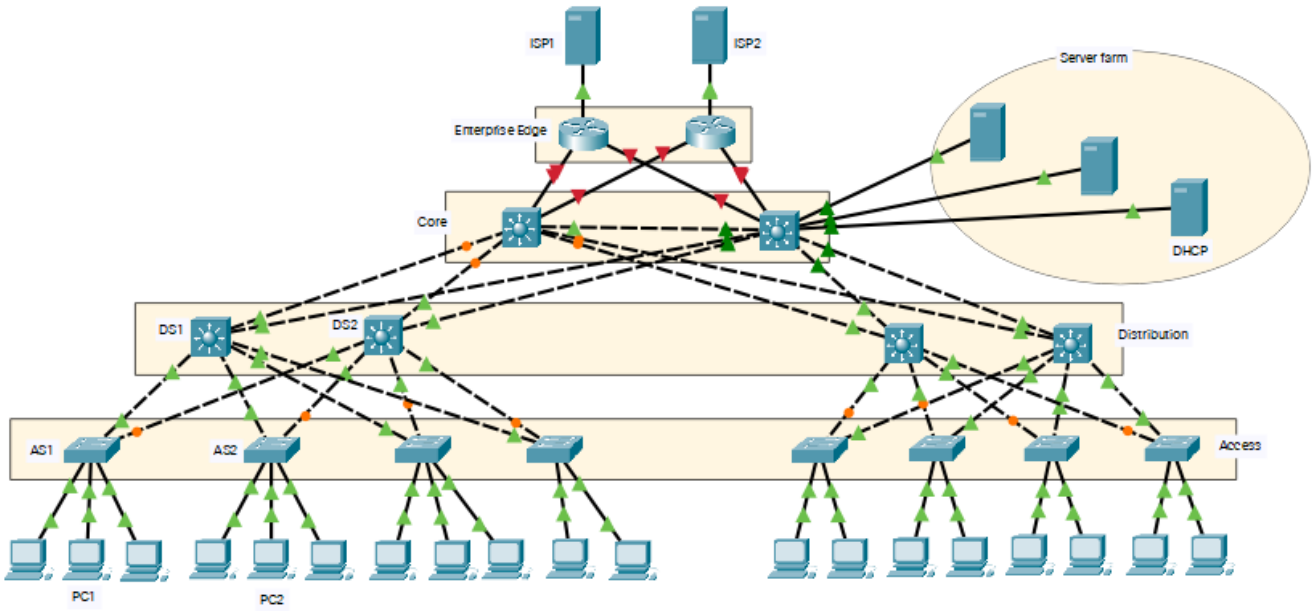


Figura 1

Din figură se vede că fiecare switch de la nivelul de Access este conectat la două switch-uri de la nivelul Distribution.

Dacă un switch de la nivelul Distribution iese din funcţiune => legătura nu se va pierde.

Dar pentru a asigura conexiune, în caz de avarie, au fost folosite linii suplimentare (redundante).

Cablurile încrucişate sunt suplimentare – dar fără de acestea nu este posibilă realizarea unei astfel de scheme.

Aşadar, există problema: cablurile încrucişate formează un ciclu în reţea

Dacă PC1 vrea să afle adresa MAC a lui PC2 => se trimite o solicitare ARP spre AS1 => acest pachet broadcast ajunge la DS1 şi DS2 => de la DS1 la AS2 => de la AS2 la DS2 => de la DS2 la AS1 => astfel frame-ul s-a întors la AS1 => procesul se repetă => se formează un ciclu (frame-ul broadcast parcurge până la infinit acest ciclu).

În reţele este absolut necesar de implementat o schemă ce asigură stabilitate în funcţionare

Switch-ul de la nivelul Distribution poate deservi un număr mare de switch-uri de la nivelul Access => iar în fiecare switch de nivel Access avem câte 48 de porturi => astfel, switch-ul de nivel Distribution

deserveşte un număr mare de host-uri => dacă switch-ul de nivel Distribution iese din funcţiune, atunci se va crea un mare disconfort pentru utilizatori
Însă dacă este prezent încă un switch de nivel Distribution, conectat la switch-ul de nivel Access, atunci acesta preia traficul => problema este rezolvată la acest moment
În Figura 2 se vede că dacă dispare legătura dintre switch-uri => dispare legătura dintre PC1 şi PC2

Figura 2

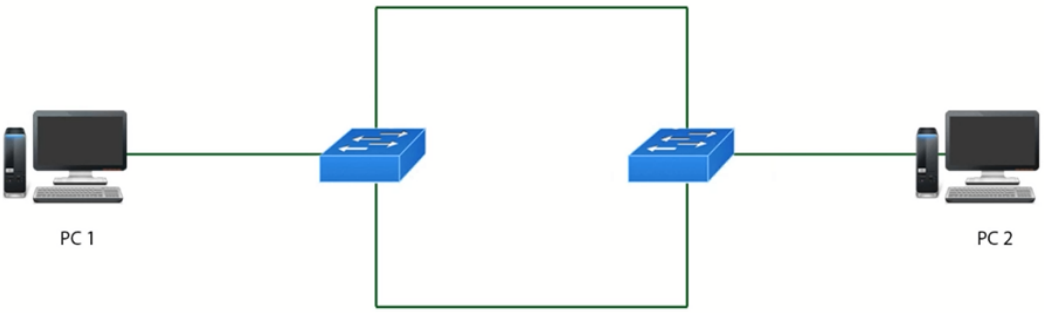


Figura 3

Pot fi utilizate două cabluri (a se vedea Figura 3) => apare o ciclare în comutarea pachetelor => frame-ul va parcurge până la infinit acest ciclu, până nu vom reîncărca switch-ul
Frame-ul broadcast de la PC1 => Sw1 => Sw1 trimite două copii ale frame-ului la Sw2 => PC2 primeşte două copii ale frame-ului
Două copii ale frame-ului vor trece permanent de la Sw1 la Sw2 şi înapoi
PC1 şi PC2 permanent formează frame-uri broadcast => acestea rămân în ciclul corespunzător pe care îl parcurg până la infinit => în rezultat, în ciclu se acumulează frame-uri => peste ceva timp canalul se va umple la 100 % => va avea loc furtuna broadcast
În canale se acumulează frame-uri şi creşte încărcarea asupra dispozitivelor de reţea => reţeaua nu va funcţiona
Dacă pe switch toate indicatoarele clipesc simultan => rezultă că a avut loc furtuna broadcast
O altă problemă – tabelul MAC al switch-ului este completat incorect
De exemplu, pentru switch-ul Sw1: mai întâi adresa MAC a lui PC1 se atribuie lui Fa 0/1, după care lui Fa 0/2, Fa 0/3 ş.a.m.d. => tabelul de adrese MAC permanent se modifică
La fel şi cu Sw2
=>
1) refuz în deservirea reţelei din cauza încărcării acesteia => furtuna broadcast (canalele sunt înfundate cu pachete broadcast)
2) instabilitatea tabelului cu adrese MAC

Cum pot fi soluţionate aceste probleme, dar astfel încât să rămână două cabluri, adică să fie asigurată funcţionarea reţelei în caz de avarie? => Protocolul *Spanning Tree Protocol* (STP)
STP – mecanismul ce permite eliminarea ciclurilor în reţea
Dacă pe switch-urile din Figura 3 va rula protocolul STP, atunci acesta va elimina ciclul prin blocarea unui port de switch (blocarea canalului). Dacă o conexiune se va întrerupe, atunci protocolul STP va debloca a doua conexiune
La fel în Figura 1 link-ul dintre AS1 şi DS2 va fi deconectat
Dacă DS1 va înceta să funcţioneze => protocolul STP va debloca link-ul blocat anterior => AS1 va putea funcţiona în continuare
Dacă vom conecta trei switch-uri ca în Figura 4, atunci reţeaua nu va funcţiona imediat (avem cercuri de culoare orange), ci peste 15-30 de secunde => din cauza că rulează protocolul STP, care verifică dacă în reţea există cicluri => va vedea că nu există cicluri şi va debloca porturile
Dacă vom conecta Sw1 cu Sw2 (a se vedea Figura 5) => reţeaua iarăşi va înceta să funcţioneze (culoare

orange) => protocolul STP verifică dacă există cicluri => pe Sw1 şi Sw2 activează ambele porturi, iar pe Sw0 – activează doar unul din cele două porturi
Se blochează nu tot link-ul, ci doar portul

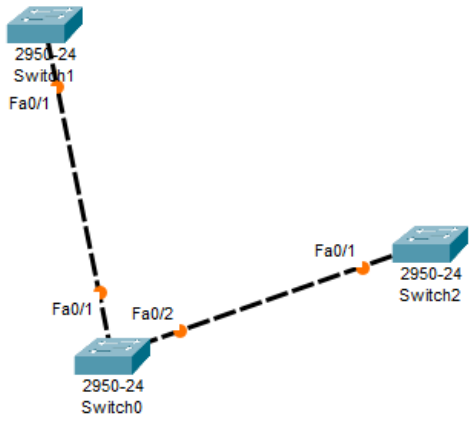


Figura 4

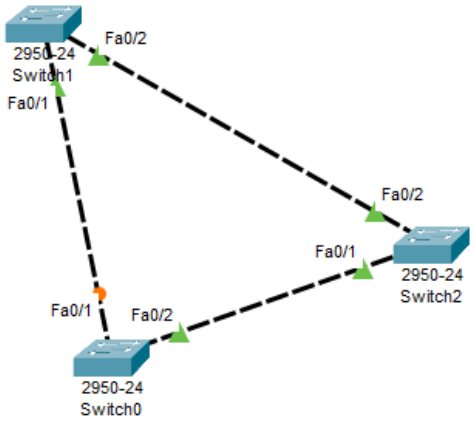


Figura 5

Întrebare: de ce a fost blocat anume acel port şi nu altul?
Este necesar să se ştie cum poate fi dirijat protocolul STP, astfel încât să fie blocat link-ul cel mai puţin eficient
II. *Cum funcţionează protocolul STP?*
Prima etapă: Selectarea switch-ului rădăcină (numit root). Celelalte switch-uri stabilesc rute până la switch-ul rădăcină
Etapa a doua: Selectarea porturilor rădăcină (root) = porturi care „privesc direct" spre switch-ul rădăcină (a se vedea Figura 6)

Figura 6

Figura 7

Figura 8

Etapa trei:
a) Selectarea porturilor desemnate (designated) = porturi care nu „privesc" spre switch-ul rădăcină, dar prin care pot fi transmise pachete de date (a se vedea Figura 7)
b) Selectarea porturilor alternative (alternate) = porturile ce sunt blocate (a se vedea Figura 8)
O descriere mai detaliată:
1. Selectarea switch-ului rădăcină

Toate switch-urile construiesc arborele de legătură până la switch-ul rădăcină şi formează şi configuraţia iniţială de reţea o topologie stea (care garantat nu include cicluri)
Obiectivul de bază al fiecărui switch este să construiască o rută până la switch-ul rădăcină
Selectarea switch-ului rădăcină este bazată pe comparaţia între unele valori, precum *prioritatea* (priority) şi *adresa MAC a switch-ului* (a se vedea Figura 9)
Switch-urile ce menţin STP => au adrese MAC
Primul criteriu de selectare a switch-ului rădăcină – ca switch rădăcină este selectat acel switch care are cea mai mică valoare a *priorităţii*
Valoarea implicită a priorităţii este – 32768
Dacă prioritățile sunt egale => atunci se compară adresele MAC, octeţii corespunzători de la stânga la dreapta => switch-ul ce are cea mai mică valoare a adresei MAC este numit ca switch rădăcină

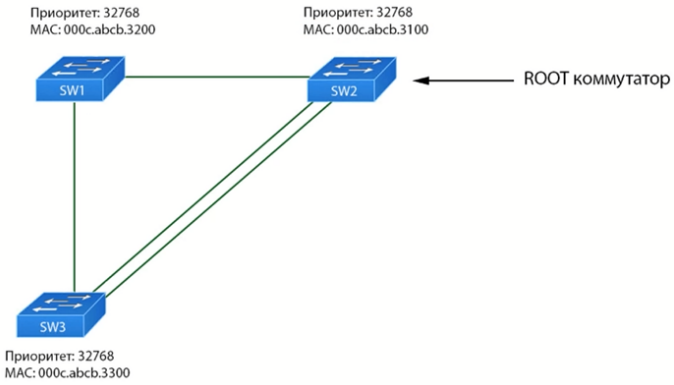


Figura 9

Valorile priorităţii şi a adresei MAC se transmit împreună => acestea se conţin în câmpul *Bridge ID* (Figura 10)

Figura 10

Switch-ul cu valoarea cea mai mică pentru Bridge ID este switch-ul rădăcină
Dacă accesăm linia de comandă CLI a switch-ului şi dăm comanda
show spanning-tree => vom vedea separat valorile pentru prioritate şi pentru adresa MAC
Între switch-uri se transmit mesaje BPDU (Bridge Protocol Data Units)
În Figura 11 este arătată structura unui frame BPDU

Структура BPDU кадра											
2 байта	1 байт	1 байт	1 байт	8 байт	4 байта	8 байт	2 байта	2 байта	2 байта	2 байта	2 байта
Protocol ID	Version	Message Type	Flags	Root Bridge ID	Root Path Cost	Sender Bridge ID	Port ID	Message Age	Maximum Age	Hellow Time	Forward Delay

Figura 11

Într-un câmp pe 8 octeţi, numit Root Bridge ID
Dacă în reţeaua prezentată prin Figura 5 trecem în modul Simulation, apăsăm butonul Show all/none, după care pe butonul Edit filters -> Misc şi lăsăm bifafte doar pachetele STP, vom vedea cum se propagă frame-urile STP (acolo se poate vedea că acest tip de pachet este de nivelul doi al stivei de protocoale TCP/IP) – frame-urile STP (în realitate BPDU) trec chiar şi prin porturile blocate
La prima conectare fiecare switch crede că el este switch-ul rădăcină şi transmite celorlalţi vecini că este switch-ul rădăcină
De exemplu, switch-ul Sw3 a obţinut două BPDU (de la Sw1 şi Sw2) şi compară valoarea proprie Bridge ID cu Bridge ID-urile lui Sw1 şi Sw2. Dacă va vedea că valoarea Bridge ID a unui alt switch este mai mică (de exemplu, la Sw2) => Sw3 va considera de acum că Sw2 este switch-ul rădăcină
În continuare, la transmiterea BPDU către vecini, Sw3 va preciza nu valoarea sa Bridge ID, ci a noului switch rădăcină (Sw2) => Sw3 arată lui Sw1 că switch-ul rădăcină este Sw2
Procedura se repetă până atunci când toate switch-urile vor stabili acelaşi switch rădăcină

II. Selectarea porturilor rădăcină
Se face în baza costului interfeței
Costul asociat switch-ului rădăcină este 0
Costul se transmite prin frame-ul BPDU în câmpul *Root Path Cost*

Accesăm linia de comandă CLI de pe Switch0 (a se vedea Figura 5):
În modul privilegiat scriem:
Switch# show spanning-tree (se poate scrie numai *span*)
Vom obține informație despre Root ID, Bridge ID, Interface (se vede rolul porturilor):

```
Switch>en
Switch#show span
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    32769
              Address     0006.2AB1.A3EA
              Cost        19
              Port        2(FastEthernet0/2)
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     00D0.97B3.6E78
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- --------
--------------------------------
Fa0/1            Altn BLK 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p
```

Cum putem impune un switch ca el să devină switch rădăcină (din careva considerente)?
Aceasta se poate face în modul următor (în linia de comandă CLI de pe Sw2, de exemplu):

```
Switch>en
Switch#conf ter
Switch(config)#spanning-tree vlan 1 root primary
```

Prioritatea lui Sw2 va deveni cu 8192 (iar la repetarea procedeului cu 4096) unități mai mică ca prioritatea lui Sw1 (care este switch-ul rădăcină) => are loc recalculul topologiei => aproximativ peste 30 de secunde legăturile se vor schimba
Pentru stabilirea manuală a priorității:

```
Switch(config)#spanning-tree vlan 1 priority 4096
```

PVST+ și Rapid PVST+ => versiuni actuale ale protocolului STP (dar reprezintă un soft proprietar Cisco)

Atunci când în rețea avem VLAN-uri => fizic există un ciclu, dar din punctul de vedere logic al VLAN-urilor – nu există ciclu
Uneori punctele ce corespund porturilor deconectate ale switch-ului, oricum rămân de culoare verde
Aceasta este deoarece pentru un VLAN se deconectează un port, iar pentru celălalt – uneori altul
Cu ajutorul comenzii *show spanning-tree* se poate vizualiza informația corespunzătoare fiecărui VLAN
Să examinăm configurația din Figura 12

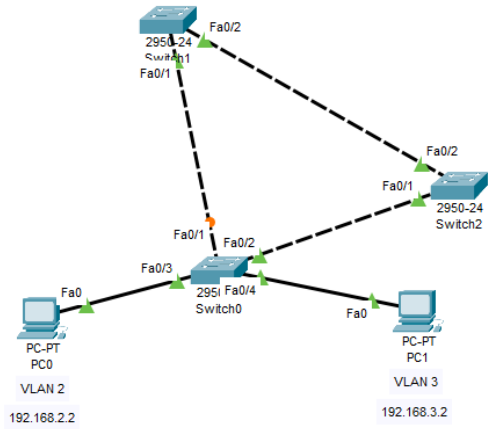


Figura 12

Protocolul PVST+ lucrează cu VLAN-urile și construiește un arbore separat în raport cu fiecare VLAN
Se poate de făcut astfel încât să avem un switch rădăcină pentru VLAN 2 și un alt switch rădăcină pentru

VLAN3

Mai întâi creăm VLAN-urile 2 şi 3 (conform schemei ilustrate anterior când am examinat tema dedicată VLAN-urilor)

```
Pe Switch0:
Switch>en
Switch#conf ter
Switch(config)#int fa 0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#exit
Switch(config)#int fa 0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#exit
Switch(config)#int fa 0/2
Switch(config-if)#switchport mode trunk
Switch(config-if)#exit
Switch(config)#int fa 0/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#exit
```

```
Pe Switch1:
Switch>en
Switch#conf ter
Switch(config)#int fa 0/2
Switch(config-if)#switchport mode trunk
Switch(config-if)#exit
Switch(config)#vlan 2
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#exit
```

```
Pe Switch2:
Switch>en
Switch#conf ter
Switch(config)#int range fa 0/1-2
Switch(config-if-range)#switchport mode trunk
Switch(config-if-range)#exit
Switch(config)#vlan 2
Switch(config-vlan)#exit
Switch(config)#vlan 3
Switch(config-vlan)#exit
```

Accesăm linia de comandă CLI de pe Switch0:

```
Switch#show span
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    24577
             Address     0006.2AB1.A3EA
             Cost        19
             Port        2(FastEthernet0/2)
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     00D0.97B3.6E78
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Altn BLK 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p

VLAN0002
  Spanning tree enabled protocol ieee
  Root ID    Priority    32770
```

```
             Address     0006.2AB1.A3EA
             Cost        19
             Port        2(FastEthernet0/2)
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32770  (priority 32768 sys-id-ext 2)
             Address     00D0.97B3.6E78
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Altn BLK 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p
Fa0/3            Desg FWD 19        128.3    P2p

VLAN0003
  Spanning tree enabled protocol ieee
  Root ID    Priority    32771
             Address     0006.2AB1.A3EA
             Cost        19
             Port        2(FastEthernet0/2)
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32771  (priority 32768 sys-id-ext 3)
             Address     00D0.97B3.6E78
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Altn BLK 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p
Fa0/4            Desg FWD 19        128.4    P2p
```

Am obţinut informaţia pentru VLAN1, VLAN2 şi VLAN3; vedem care port-uri sunt blocate în raport cu fiecare VLAN

Accesăm linia de comandă CLI pe Switch2 şi dăm comanda *show span*:

```
Switch#show span
VLAN0001
Spanning tree enabled protocol ieee
Root ID Priority 24577
Address 0006.2AB1.A3EA
This bridge is the root
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 24577 (priority 24576 sys-id-ext 1)
Address 0006.2AB1.A3EA
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 20

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1 Desg FWD 19 128.1 P2p
Fa0/2 Desg FWD 19 128.2 P2p

VLAN0002
Spanning tree enabled protocol ieee
Root ID Priority 32770
Address 0006.2AB1.A3EA
This bridge is the root
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32770 (priority 32768 sys-id-ext 2)
Address 0006.2AB1.A3EA
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
```

```
Aging Time 20

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1 Desg FWD 19 128.1 P2p
Fa0/2 Desg FWD 19 128.2 P2p

VLAN0003
Spanning tree enabled protocol ieee
Root ID Priority 32771
Address 0006.2AB1.A3EA
This bridge is the root
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32771 (priority 32768 sys-id-ext 3)
Address 0006.2AB1.A3EA
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 20

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1 Desg FWD 19 128.1 P2p
Fa0/2 Desg FWD 19 128.2 P2p
```

Vedem că Switch2 reprezintă switch-ul rădăcină pentru VLAN 2 şi VLAN 3

Vom face ca switch rădăcină pentru VLAN-ul 3 să devină switch-ul Switch1:
Accesăm Switch1 şi în linia de comandă executăm:

```
Switch#conf ter
Switch(config)#spanning-tree vlan 3 root primary
```

Dacă vom intra în linia de comandă a lui Switch0 şi vom scrie comanda *show span*, atunci vom vedea că porturile au roluri diferite în raport cu VLAN 2 şi VLAN 3:

```
VLAN0002
Spanning tree enabled protocol ieee
Root ID Priority 32770
Address 0006.2AB1.A3EA
Cost 19
Port 2(FastEthernet0/2)
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32770 (priority 32768 sys-id-ext 2)
Address 00D0.97B3.6E78
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 20

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1 Altn BLK 19 128.1 P2p
Fa0/2 Root FWD 19 128.2 P2p
Fa0/3 Desg FWD 19 128.3 P2p

VLAN0003
Spanning tree enabled protocol ieee
Root ID Priority 24579
Address 00D0.9772.8013
Cost 19
Port 1(FastEthernet0/1)
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

Bridge ID Priority 32771 (priority 32768 sys-id-ext 3)
Address 00D0.97B3.6E78
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 20

Interface Role Sts Cost Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
```

```
Fa0/1 Root FWD 19 128.1 P2p
Fa0/2 Altn BLK 19 128.2 P2p
Fa0/4 Desg FWD 19 128.4 P2p
```

Pentru VLAN 3 Switch1 a devenit switch rădăcină; pentru VLAN 2 Switch2 a devenit switch rădăcină
În continuare vom arăta cum se face ieşirea în Internet prin router (a se vedea Figura 13)

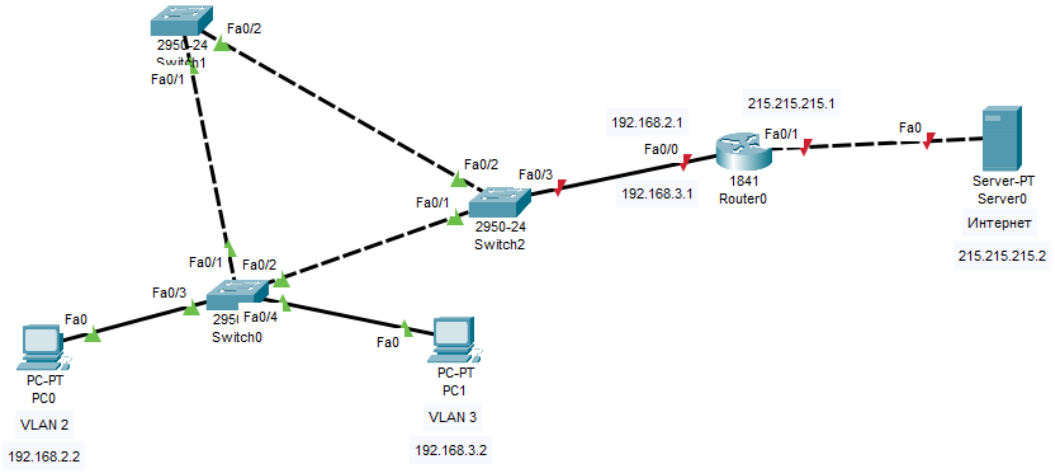


Figura 13

Pentru aceasta vom executa următoarele:

1. Pe host-urile PC0 şi PC1 configurăm adresele IP ale routerului implicit 192.168.2.1 şi, corespunzător 192.168.3.1
2. Facem portul Fa 0/3 al switch-ului Switch2 să fie de tip trunk
3. Configurăm două subinterfeţe (Fa 0/0.2 şi Fa 0/0.3) pe portul Fa 0/0 al routerului

```
Router>en
Router#conf ter
Router(config)#int fa 0/0.2
Router(config-subif)#encapsulation dot1Q 2
Router(config-subif)#ip add 192.168.2.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int fa 0/0.3
Router(config-subif)#encapsulation dot1Q 3
Router(config-subif)#ip add 192.168.3.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int fa 0/0
Router(config-if)#no shut    % включаем физический интерфейс
```

4. Atribuim IP-ul 215.215.215.1 pe interfaţa Fa 0/1

```
Router(config)#int fa 0/1
Router(config-if)#ip add 215.215.215.1 255.255.255.0
Router(config-if)#no shut
```

5. Pe server configurăm IP-ul 215.215.215.2 şi adresa routerului implicit 215.215.215.1

Dacă dăm un ping către serverul de Internet 215.215.215.2 de pe host-urile PC0 şi PC1 => vedem că este conexiune
În modul Simulation vom urmări traseul pachetelor de date (lăsăm să fie vizualizate doar pachetele ICMP):
Pe host-ul PC0:
În VLAN2 şi în VLAN3 avem trasee diferite pe care se deplasează pachetele
Pe host-ul PC1:
În VLAN2 şi în VLAN3 avem trasee diferite pe care se deplasează pachetele

Lucrarea de laborator №4

IP-adresarea şi divizarea reţelelor în subreţele

Scopul lucrării constă în formarea abilităţilor practice de planificare şi utilizare a IP-adreselor şi a măştilor de reţea, de divizare în subreţele (subnetare) şi implementare a schemei de adresare VLSM

Obiective:

- Analiza conceptelor de adresă IPv4, mască de reţea implicită, mască de reţea extinsă (cu prefix extins), identificator de reţea, identificator de host în reţea, adresă de broadcast
- Elaborarea unei scheme de adrese IPv4 pentru subreţelele reţelei, folosind aceeaşi mască extinsă pentru fiecare subreţea sau măşti extinse posibil diferite pentru subreţele (procedeul VLSM - Variable Length Subnet Mask)
- Configurarea adreselor IP pe dispozitivele de reţea (routere şi host-uri)

Repere teoretice şi exemple de soluţionare a sarcinilor

În cazul unei adrese IPv4 precizăm următoarele:

- **adresa IPv4** - 4 grupuri a câte 8 biţi. Exemplu: 192.168.100.200
- **masca de reţea** (subnet mask) - 4 grupuri a câte 8 biţi, cu proprietatea că se începe cu bitul 1, iar toţi biţii de 1 sunt consecutivi, alternanţa repetată de biţi 0 şi 1 fiind interzisă. De exemplu 11111111.00000000.00000000.00000000 este o mască de reţea validă, iar 11000001.00000000.00000000.0000000 este o mască nevalidă. Pentru a uşura citirea măştii aceasta se scrie în zecimal, similar IP adresei: 11111111.00000000.00000000.00000000 = 255.0.0.0. Datorită proprietăţii speciale în care biţii de 1 sunt consecutivi, o altă formă în care poate fi specificată masca de reţea este forma cu prefix (numită şi notaţie CIDR): /X, unde X reprezintă numărul de biţi de 1: 11111111.00000000.00000000.00000000 = 255.0.0.0 = /8.

Iniţial, pentru a putea adresa reţele de diferite dimensiuni, adresele IPv4 au fost împărţite în 5 clase: A, B, C, D şi E. Clasele D şi E au o utilizare specială.

Clasa A are primul bit ‚0' şi adresa de reţea din 8 biţi. Masca de subreţea este 255.0.0.0. Sunt $128=2^7$ de astfel de reţele. Utilizare: reţele foarte mari.

Clasa B are primii biţi '10' şi adresa de reţea de 16 biţi. Masca de subreţea este 255.255.0.0. Sunt maximum 16384 de astfel de reţele. Utilizare: reţele mari.

Clasa C are primii biţi '110' şi adresa de reţea de 24 biţi. Masca de subreţea este 255.255.255.0. Sunt maximum 2097152 de astfel de reţele. Utilizare: reţele mici.

Clasa D are primii biţi '1110' şi adresa de reţea de 32 biţi. Masca de subreţea este 255.255.255.255. Sunt maximum 268435456 de astfel de reţele. Utilizare: adresa de grup multicasting, nu are host-uri.

Clasa E rezervata pentru scopuri experimentale.

Clasa adresei IP	Prefixul	Primii biţi ai adresei IP	Intervalul de adrese	Numărul de subreţele definite de adresa de clasa dată	Numărul de host-uri definite de adresa de clasa dată
A	/8	0xxx	0.0.0.0 - 127.255.255.255	128	16777214
B	/16	10xx	128.0.0.0 - 191.255.255.255	16384	65534
C	/24	110x	192.0.0.0 - 223.255.255.255	2097152	254
D	/32	1110	224.0.0.0 - 239.255.255.255	268435456	0
E	nedefinit	1111	240.0.0.0 - 255.255.255.255	nedefinit	nedefinit

Prin intermediul măştii de reţea se face divizarea între partea de reţea şi partea de host. Numărul de biţi pentru partea de host indică numărul maximal de IP adrese care pot fi atribuite host-urilor. Două dintre aceste adrese, însă, nu sunt atribuite host-urilor. Este vorba de prima adresă IP şi ultima adresă IP.

Prima adresă IP a intervalului de adrese conţine doar biţi de 0 pentru partea de host şi coincide cu **adresa de reţea**. Ultima adresă IP a spaţiului de adrese conţine doar biţi de 1 pentru partea de host

și coincide cu **adresa de broadcast**.
Dacă avem masca de rețea /24, adică 8 biți (32-24) pentru partea de host, vom avea un spațiul posibil de 2^8 = 256 adrese. Din aceste adrese 2 nu sunt utilizate (adresa de rețea și adresa de broadcast) și vom avea, așadar 254 de adrese ce pot fi atribuite host-urilor.
În general, dacă avem N biți alocați pentru partea de host, vom avea 2^N-2 adrese ce pot fi utilizate.

Clasa	Retele/Clasa	Hosturi/Clasa
A	2^{8-1} = 128	2^{24}-2 = 16.777.214
B	2^{16-2} = 16.384	2^{16}-2 = 65.534
C	2^{24-3} = 2.097.152	2^8-2 = 254
D	2^{32-4} = 268.435.456	0

Pornind de la adresa IP și masca de rețea, putem identifica adresa de rețea și adresa de broadcast (pentru exemplificare vom folosi adresa IP 192.168.100.200 cu masca 255.255.255.0:

- **adresa de rețea** - se obține, calculând AND-ul logic între biții adresei IP și biții măștii de rețea
  192.168.100.200 & 255.255.255.0 = **192.168.100.0**
- **adresa de broadcast** - se obține calculând OR-ul logic între biții adresei IP și biții complementului măștii de rețea (complementul se obține, inversând valoarea biților de pe fiecare poziție)
  192.168.100.200 | 0.0.0.255 = **192.168.100.255**

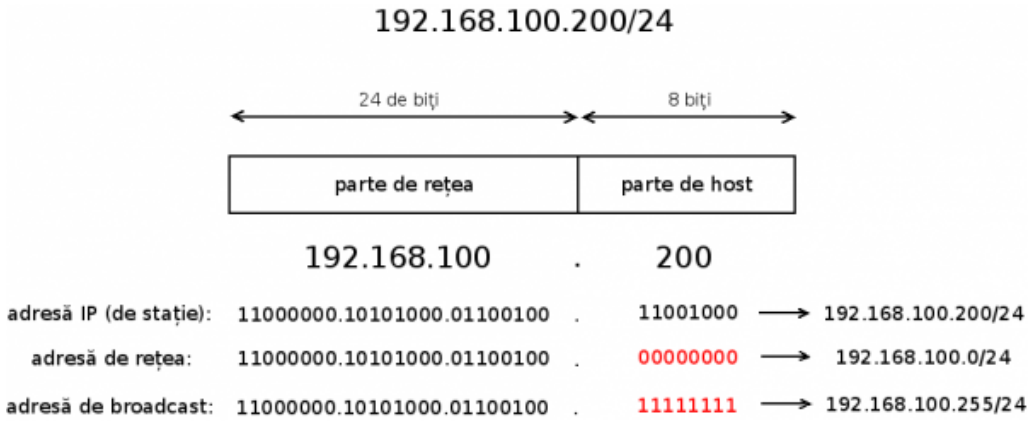
AND	1	0
1	1	0
0	0	0

OR	1	0
1	1	1
0	1	0

Atunci când cunoaștem adresa IP și masca de rețea și vrem să obținem adresa de rețea și adresa de broadcast, este util să folosim masca de rețea pentru a diviza adresa IP în două:

- O **parte de subrețea**, care se întinde pe câți biți de 1 are masca de rețea. E vorba de 24 de biți, pentru o mască /24 (sau 255.255.255.0) sau 16 biți pentru o mască /16 (sau 255.255.0.0) sau 20 de biți pentru o mască /20 (sau 255.255.240.0).
- O **parte de host,** care se întinde pe restul spațiului (32 - numărul de biți de 1 ai măștii de rețea). E vorba de 8 biți pentru o mască /24 (32-24 = 8) sau de 16 biți pentru o mască /16 (32-16 = 16) sau de 12 biți pentru o mască /20 (32-20 = 12).

De exemplu, dacă vom diviza aceeași adresă IP 192.168.100.200/24, vom obține aceleași valori precum cele calculate mai sus, lucru reflectat și în figura de mai jos.



Să determinăm adresa de rețea și adresa de broadcast corespunzătoare adresei 172.16.200.100/20. Scriem în formă hibridă adresa: 172.16.1100|1000.xxxxxxxx. Am folosit operatorul | (pipe) pentru a separa **partea de rețea** (primii 20 de biți conform măștii de rețea) de **partea de host** (ceilalți 32-20 = 12 biți). Deoarece pentru stabilirea obiectivului propus nu sunt relevanți biții ultimului octet, am folosit simbolul xxxxxxxx în locul acestuia.
Adresa de rețea are **toți biții părții de host egali cu 0**, adică avem 172.16.1100|0000.00000000. Rezultă adresa de rețea 172.16.192.0/20.
Adresa de broadcast are **toți biții părții de host egali cu 1**, adică avem

172.16.1100|1111.11111111. Rezultă adresa de broadcast 172.16.207.255/20.

Subnetarea

Pentru a putea considera subintervale de adrese din clasa A, B sau C, a fost propusă procedura de **subnetare**. În acest mod, există posibilitatea să se atribuie câte un *identificator de subreţea* pentru fiecare subreţea internă, fără a mai solicita noi adrese.

+----------adresa IP----------+
|Identificator de reţea|----------Identificator de host----------|
|Identificator de reţea|Identificator de subreţea|Identificator de host|
+----------Identificatorul de reţea extins----------+

Lungimea *Identificatorului de reţea extins* este egală cu numărul de biţi consecutivi egali cu 1 în masca de subreţea.

Exemplu

192.129.213.54 255.255.255.0 Identificatorul de reţea extins	11000000.10000001.11010101.00110110 11111111.11111111.11111111.00000000 +----------24 biţi----------+

Un alt mod de a exprima această adresă IP şi masca de subreţea corespunzătoare este 192.129.213.54/24

Exemplu

adresa IP masca de subreţea	89.102.131.21 255.255.248.0

89.102.131.21 255.255.248.0	89.102.10000011.00010101 255.255.11111000.00000000 +--21 de biţi-- +

adresa de reţea: 89.102.10000000.00000000 (89.102.128.0)
prima adresă ce poate fi atribuită host-urilor: 89.102.10000000.00000001 (89.102.128.1)
ultima adresă ce poate fi atribuită host-urilor: 89.102.10000111.11111110 (89.102.135.254)
adresa de broadcast în reţea: 89.102.10000111.11111111 (89.102.135.255)
Adresa de reţea poate fi obţinută, aplicând operaţia and logic pe biţi, între adresa IP şi masca de subreţea, exprimate în binar.

Adrese private sunt:

10.0.0.0 - 10.255.255.255 (Adrese private de clasa A)
172.16.0.0 - 172.31.255.255 (Adrese private de clasa B)
192.168.0.0 - 192.168.255.255 (Adrese private de clasa C)

Definirea subreţelelor

Pentru a defini în mod eficient un plan de adresare IP într-o reţea, trebuie să ştim numărul de subreţele necesar şi numărul maxim de host-uri din fiecare subreţea. Pe lângă aceste date, trebuie să ţinem cont şi de evoluţiile ulterioare ale organizaţiei. Vom prezenta un exemplu în care se defineşte o schemă de adresare IP.

Exemplul 1. O companie are adresa de reţea 192.129.213.0/24 şi intenţionează să creeze 5 subreţele a câte 20 de host-uri fiecare.

Pentru a elabora o schemă de adrese IP pentru fiecare din cele 5 subreţele, vom împrumuta 3 biţi (2^3=8 variante de subreţele) de la identificatorul de host, adică definim masca extinsă /27. Nu putem împrumuta doar doi biţi, deoarece atunci vom avea adrese pentru 2^2=4 subreţele. Din cele 8 subreţele, 3 subreţele (8-5=3) sunt rezervate pentru dezvoltări ulterioare.

Vom utiliza reprezentările binare pentru IP adresă şi masca de reţea:

	Reprezentare în zecimal cu punct	*Reprezentare în binar*
IP adresa	192.129.213.0	11000000.10000001.11010101.00000000
Masca de reţea iniţială	255.255.255.0	11111111.11111111.11111111.00000000
Masca de subreţea extinsă	255.255.255.224	11111111.11111111.11111111.11100000

Cele 8 subreţele menţionate sunt descrise prin următoarele blocuri de adrese IP:

11000000.10000001.11010101.**000**00000=192.129.213.0/27
11000000.10000001.11010101.**001**00000=192.129.213.32/27
11000000.10000001.11010101.**010**00000=192.129.213.64/27
11000000.10000001.11010101.**011**00000=192.129.213.96/27
11000000.10000001.11010101.**100**00000=192.129.213.128/27
11000000.10000001.11010101.**101**00000=192.129.213.160/27
11000000.10000001.11010101.**110**00000=192.129.213.192/27
11000000.10000001.11010101.**111**00000=192.129.213.224/27

Cei 3 biţi împrumutaţi pentru definirea a 8 subreţele constituie următoarele combinaţii: $(000)_2$, $(001)_2$, $(010)_2$, $(011)_2$, $(100)_2$, $(101)_2$, $(110)_2$, $(111)_2$.

Din ultimul octet al adresei IP am folosit 3 biţi pentru formarea subreţelelor, deci mai ramân disponibili 5 biţi pentru atribuirea de adrese host-urilor în subreţea. În baza celor 5 biţi se pot atribui adrese la 30 (=2^5-2) de host-uri (ceea ce constituie un număr mai mare ca cele 20 cerute iniţial!).

Din cele 30 de adrese IP de host, două au fost eliminate, deoarece varianta cu $(00000)_2$ pe partea de host coincide cu adresa subreţelei, iar varianta cu $(11111)_2$ pe partea de host – cu adresa de broadcast a subreţelei.

Astfel am obţinut:

Masca de subreţea: 255.255.255.224 (aceeaşi pentru fiecare subreţea)
Număr de subreţele: 8 (iniţial solicitate 5)
Număr de host-uri în fiecare subreţea: 30 (iniţial solicitate 20).

Pentru exemplificare, vom arăta cum se obţin adresele host-urilor din subreţeaua 6 (101) sunt:

adresa subreţelei 6 – 11000000.10000001.11010101.10100000=192.129.213.160/27
adresa host 1 - 11000000.10000001.11010101.101**00001**=192.129.213.161/27
adresa host 2 - 11000000.10000001.11010101.101**00010**=192.129.213.162/27
adresa host 3 - 11000000.10000001.11010101.101**00011**=192.129.213.163/27
……
adresa host 28 - 11000000.10000001.11010101.101**11100**=192.129.213.188/27
adresa host 29 - 11000000.10000001.11010101.101**11101**=192.129.213.189/27
adresa host 30 - 11000000.10000001.11010101.101**11110**=192.129.213.190/27
adresa de broadcast în subreţea - 11000000.10000001.11010101.101**11111**=192.129.213.191/27

În reţelele mari, se porneşte de la un spaţiu de adrese (bloc de adrese IP) oferit de o autoritate, din care se separă blocuri de adrese pentru subreţelele reţelei. Subreţelele vor acoperi zone/departamente diferite. Procesul de divizare în subreţele şi atribuire de adrese IP în acestea este numit *subnetare*. În varianta simplă se face divizarea în subreţele cu acelaşi număr de adrese de host-uri (deoarece la baza fiecărei subreţele stă aceeaşi mască de subreţea).

Utilizarea notaţiei CIDR, adică a măştii extinse, elimină ideea de clase de IP adrese (clasa A, B şi C pentru dispozitivele de reţea) şi conduce la alocarea mai eficientă a spaţiului de IP adrese. Procedeul VLSM, care admite utilizarea măştilor distincte în procesul de subnetare, divizează mai econom în blocuri spaţiul de IP adrese alocate reţelei.

Subnetarea VLSM (Variable Length Subnet Mask)

Procedeul de subnetare examinat mai sus, diviza reţeaua iniţială într-un careva număr de subreţele astfel încât la elaborarea unui interval de adrese IP pentru fiecare subreţea să folosim aceeaşi mască. Acest fapt conduce la acelaşi număr de adrese IP în fiecare subreţea.

Conceptul de VLSM permite o utilizare mai eficientă a spaţiului de adrese IP, datorită faptului că pentru divizarea în subreţele de dimensiuni diferite pot fi aplicate diferite măşti de reţea. Măştile distincte permit divizarea chiar a unei subreţele în mai multe sub-subreţele. În acest mod, dacă este necesar spaţiul de adrese IP disponibil (de exemplu, al unei companii) poate fi divizat succesiv în subintervale.

Exemplul 2. Vom realiza procedura de subnetare VLSM în reţeaua unei companii mari cu următoarea structură (a se vedea Figura 1).

Pentru implementarea a 14 subreţele, pornind de la blocul de adrese IP 183.103.0.0./16 de clasă B, vom împrumuta 4 biţi (2^4=16 subreţele) de la identificatorul de host.

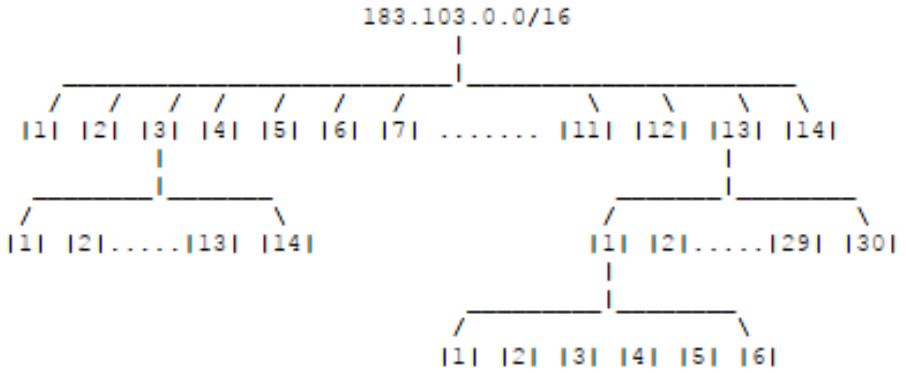


Figura 1

	Reprezentare în zecimal cu punct	*Reprezentare în binar*
IP adresa	183.103.0.0	10110111.01100111.00000000.00000000
Masca de reţea iniţială	255.255.0.0	11111111.11111111.00000000.00000000
Masca de subreţea (extinsă)	255.255.240.0	11111111.11111111.11110000.00000000

În rezultat, obţinem următoarele 16 adrese de subreţea:

subreţea 0 - 10110111.01100111.**0000**0000.00000000=183.103.0.0/20
subreţea 1 - 10110111.01100111.**0001**0000.00000000=183.103.16.0/20
subreţea 2 - 10110111.01100111.**0010**0000.00000000=183.103.32.0/20
subreţea 3 - 10110111.01100111.**0011**0000.00000000=183.103.48.0/20
subreţea 4 - 10110111.01100111.**0100**0000.00000000=183.103.64.0/20
subreţea 5 - 10110111.01100111.**0101**0000.00000000=183.103.80.0/20
.....
subreţea 12 - 10110111.01100111.**1100**0000.00000000=183.103.192.0/20
subreţea 13 - 10110111.01100111.**1101**0000.00000000=183.103.208.0/20
subreţea 14 - 10110111.01100111.**1110**0000.00000000=183.103.224.0/20
subreţea 15 - 10110111.01100111.**1111**0000.00000000=183.103.240.0/20

Putem folosi oricare 14 subreţele din cele 16. Pentru exemplificare, vom folosi în continuare, subreţelele 1-14. Adresa fiecărei din cele 16 subreţele stabilite are 12 biţi disponibili pentru atribuirea de adrese IP host-urilor. Astfel pot fi atribuite adrese IP la 4094 de host-uri (2^12-2=4096-2=4094) în fiecare subreţea.

Conform cerinţelor, subreţeaua 3 o divizăm la rândul ei în 16 subreţele (conform cerinţelor – 14 subreţele dintre acestea), împrumutând 4 biţi (2^4=16 subreţele) de la identificatorul de host a adresei 183.103.48.0/20 = 10110111.01100111.0011**0000**.00000000, adică extinzând masca de subreţea până la 24 de biţi – prefixul /24.

În rezultat, obţinem următoarele adrese de subreţea:

subreţea 0 - 10110111.01100111.0011**0000**.00000000=183.103.48.0/24
subreţea 1 - 10110111.01100111.0011**0001**.00000000=183.103.49.0/24
subreţea 2 - 10110111.01100111.0011**0010**.00000000=183.103.50.0/24
subreţea 3 - 10110111.01100111.0011**0011**.00000000=183.103.51.0/24
subreţea 4 - 10110111.01100111.0011**0100**.00000000=183.103.52.0/24
.....
subreţea 12 - 10110111.01100111.0011**1100**.00000000=183.103.60.0/24
subreţea 13 - 10110111.01100111.0011**1101**.00000000=183.103.61.0/24
subreţea 14 - 10110111.01100111.0011**1110**.00000000=183.103.62.0/24
subreţea 15 - 10110111.01100111.0011**1111**.00000000=183.103.63.0/24

Pot fi utilizate oricare 14 din cele 16 subreţele stabilite. Fiecare subreţea are 8 biţi pentru adresarea host-urilor. Astfel pot fi adresate 254 de host-uri (2^8-2=256-2=254) în fiecare subreţea a subreţelei 3 iniţiale.

Subreţeaua 13 iniţială, adică 183.103.208.0/20=10110111.01100111.11010000.00000000, o divizăm la rândul ei în 32 de subreţele (în realitate, sunt necesare 30!), împrumutând 5 biţi (2^5=32>30) ai identificatorului de host corespunzător:

subreţea 0 - 10110111.01100111.1101**0000**.**0**0000000=183.103.208.0/25
subreţea 1 - 10110111.01100111.1101**0000**.**1**0000000=183.103.208.128/25
subreţea 2 - 10110111.01100111.1101**0001**.**0**0000000=183.103.209.0/25
subreţea 3 - 10110111.01100111.1101**0001**.**1**0000000=183.103.209.128/25
subreţea 4 - 10110111.01100111.1101**0010**.**0**0000000=183.103.210.0/25
.....
subreţea 28 - 10110111.01100111.1101**1110**.**0**0000000=183.103.222.0/25
subreţea 29 - 10110111.01100111.1101**1110**.**1**0000000=183.103.222.128/25
subreţea 30 - 10110111.01100111.1101**1111**.**0**0000000=183.103.223.0/25
subreţea 31 - 10110111.01100111.1101**1111**.**1**0000000=183.103.223.128/25

Fiecare din cele 32 de adrese de subretele au disponibili 7 biţi pentru adresarea host-urilor. Astfel pot fi atribuite 126 (2^7-2=128-2=126) de adrese de host-uri în fiecare din aceste subreţele. Vom folosi subreţelele notate cu 1, 2, ..., 30.

Subreţeaua 1 (183.103.208.128/25=10110111.01100111.11010000.10000000) a reţelei 183.103.208.0/20 o divizăm la rândul ei în 8 subreţele (în realitate avem nevoie doar de 6 la moment), împrumutând 3 biţi (2^3=8>6).

subreţea 0 - 10110111.01100111.11010000.1**000**0000=183.103.208.128/28
subreţea 1 - 10110111.01100111.11010000.1**001**0000=183.103.208.144/28
subreţea 2 - 10110111.01100111.11010000.1**010**0000=183.103.208.160/28
subreţea 3 - 10110111.01100111.11010000.1**011**0000=183.103.208.176/28
subreţea 4 - 10110111.01100111.11010000.1**100**0000=183.103.208.192/28
subreţea 5 - 10110111.01100111.11010000.1**101**0000=183.103.208.208/28
subreţea 6 - 10110111.01100111.11010000.1**110**0000=183.103.208.224/28
subreţea 7 - 10110111.01100111.11010000.1**111**0000=183.103.208.240/28

Cele 8 subreţele au 4 biţi pentru adresarea host-urilor. Astfel pot fi atribuite adrese IP la 14 host-uri (2^4-2=16-2=14) în fiecare din aceste subreţele.

În continuare, vom expune procedura de soluţionare a două probleme de subnetare a reţelei, care va servi ca model pentru studenţi la îndeplinirea lucrării de laborator.

Exemplul 3. Subnetarea IPv4 cu aceeaşi mască de reţea pentru toate subreţelele create

Este dată topologia logică de reţea din Figura 2.

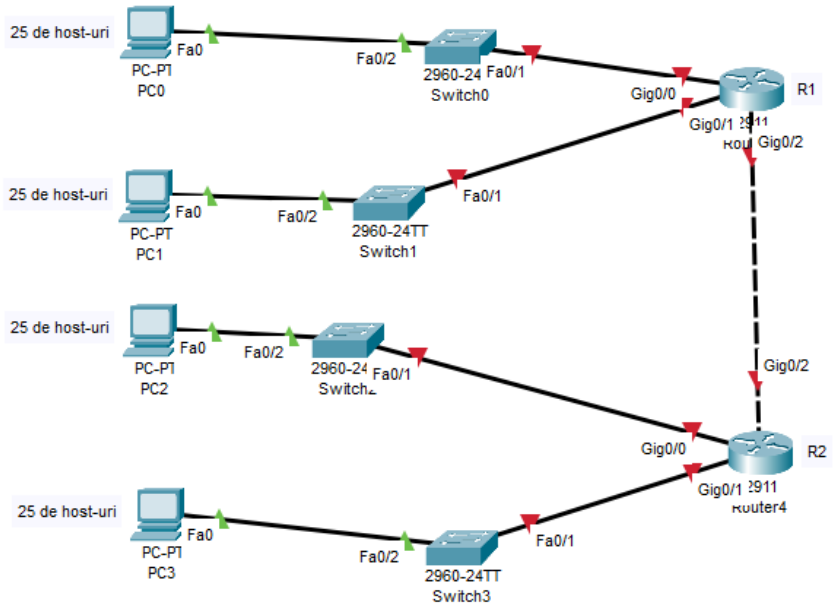


Figura 2

Dispozitivele utilizate:

Două routere Cisco 2911, 4 switch-uri Cisco 2960, un număr de PC-uri precizat în varianta de laborator corespunzătoare, cabluri Ethernet.

Dispozitiv	Interfaţă	IP adresă	Mască de subreţea	Adresa implicită a routerului
R1	G0/0			
	G0/1			
	G0/2			
R2	G0/0			
	G0/1			
	G0/2			
PC0	NIC			
PC1	NIC			
PC2	NIC			
PC3	NIC			

Tabelul 3

Obiective

Partea 1: Elaborarea unei scheme de subnetare IPv4 astfel încât subreţelele să aibă aceeaşi mască de subreţea

Partea 2: Atribuirea IP adreselor stabilite dispozitivelor din reţea şi verificarea conexiunii dintre acestea

Scenariul conform căruia este soluţionată problema

Ţinând cont de configuraţia de reţea prezentată în Figura 2, vom elabora o schemă de subnetare, pornind de la *adresa de reţea 192.168.100.0/24*, astfel încât să fie asigurate cu numărul necesar de adrese host-urile din cele patru subreţele, dar şi interfeţele corespunzătoare ale celor două routere. Fiecare subreţea necesită cel puţin 25 de adrese pentru host-uri. Conexiunea dintre routerele R1 şi R2 va necesita o IP adresă pentru fiecare capăt al link-ului dintre acestea. De fapt, pe link-ul ce le conectează routerele R1 şi R2 la fel formează o subreţea – a cincea subreţea în configuraţia examinată.

Instrucţiuni

Partea 1: Elaborarea unei scheme de IP adrese pentru subnetare

Pasul 1: Se subnetează reţeaua 192.168.100.0/24 în numărul corespunzător de subreţele

a. Reieşind din configuraţia prezentată în Figura 2, câte subreţele sunt necesare de realizat?
   5 – patru pentru LAN-uri şi una pentru link-ul dintre routere
b. Câţi biţi urmează a fi împrumutaţi pentru a suporta numărul de subreţele preconizat?
   3
c. În acest caz, câte subreţele se vor crea?
   2^3=8
d. Câte adrese IP pot fi atribuite host-urilor în fiecare subreţea?
   30 (2^5-2=32-2)
e. Determinaţi reprezentările binare pentru primele cinci subreţele:

Subreţea	*Adresa de reţea*	*Biţii ultimului octet*							
0	192.168.100.0	**0**	**0**	**0**	0	0	0	0	0
1	192.168.100.32	**0**	**0**	**1**	0	0	0	0	0
2	192.168.100.64	**0**	**1**	**0**	0	0	0	0	0
3	192.168.100.96	**0**	**1**	**1**	0	0	0	0	0
4	192.168.100.128	**1**	**0**	**0**	0	0	0	0	0

f. Se determină reprezentarea binară şi zecimală cu punct pentru masca de subreţea extinsă:

Primul octet	*Octetul doi*	*Octetul trei*	*Octetul patru al măştii extinse*							
11111111	11111111	11111111	1	1	1	0	0	0	0	0
Primul octet în zecimal	***Octetul doi în zecimal***	***Octetul trei în zecimal***	***Octetul patru în zecimal***							
255	255	255	224							

g. Completaţi tabelul de subreţea (a se vedea Tabelul 4) cu valorile zecimale cu punct ale

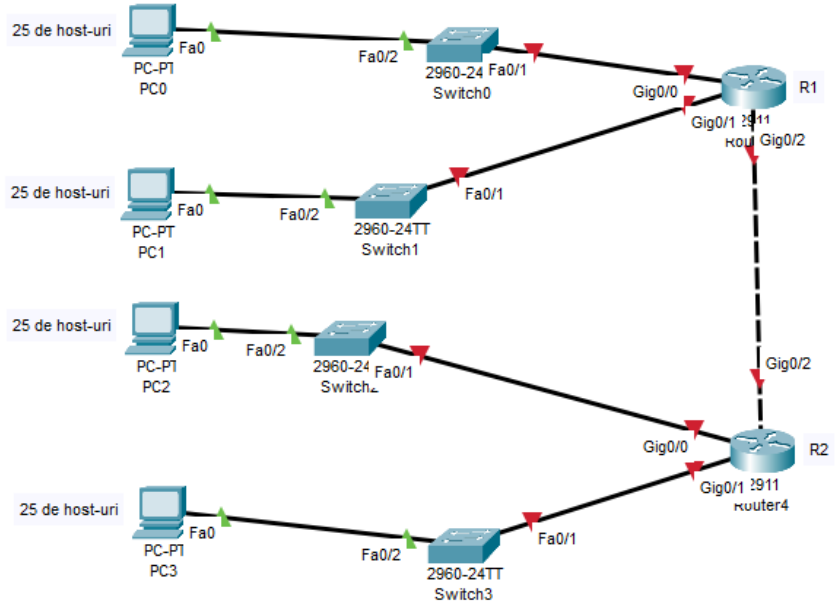
subreţelelor disponibile, prima şi ultima adresă IP ce poate fi atribuită host-urilor şi adresa de broacast în subreţea.

Numărul subreţelei	*Adresa subreţelei*	*Prima adresă de host utilizabilă*	*Ultima adresă de host utilizabilă*	*Adresa de broadcast în subreţea*
0	192.168.100.0/27	192.168.100.1	192.168.100.30	192.168.100.31
1	192.168.100.32/27	192.168.100.33	192.168.100.62	192.168.100.63
2	192.168.100.64/27	192.168.100.65	192.168.100.94	192.168.100.95
3	192.168.100.96/27	192.168.100.97	192.168.100.126	192.168.100.127
4	192.168.100.128/27	192.168.100.129	192.168.100.158	192.168.100.159
5	192.168.100.160/27	192.168.100.161	192.168.100.190	192.168.100.191
6	192.168.100.192/27	192.168.100.193	192.168.100.222	192.168.100.223
7	192.168.100.224/27	192.168.100.225	192.168.100.254	192.168.100.255

Tabelul 4

Pasul 2. În configuraţia de reţea prezentată în Figura 2 se atribuie corespunzător adresele de subreţele determinate

a. Atribuim adresa subreţelei 0 reţelei LAN conectate la interfaţa GigabitEthernet 0/0 a routerului R1: 192.168.100.0 /27
b. Atribuim adresa subreţelei 1 reţelei LAN conectate la interfaţa GigabitEthernet 0/1 a routerului R1: 192.168.100.32 /27
c. Atribuim adresa subreţelei 2 reţelei LAN conectate la interfaţa GigabitEthernet 0/0 a routerului R2: 192.168.100.64 /27
d. Atribuim adresa subreţelei 3 reţelei LAN conectate la interfaţa GigabitEthernet 0/1 a routerului R2: 192.168.100.96 /27
e. Atribuim adresa subreţelei 4 link-ului WAN dintre R1 şi R2: 192.168.100.128 /27



Pasul 3: Documentăm schema de adresare

Un exemplu de completare a datelor din tabelul de adresare (a se vedea Tabelul 3) este dat în continuare:

a. Atribuim prima IP adresă utilizabilă din subreţelele 0 şi 1, corespunzătoare celor două interfeţe G0/0 şi G0/1 ale routerului R1, iar interfeţei lui R1 din partea link-ului WAN (G0/2) îi atribuim prima adresă utilizabilă a subreţelei 4 (de exemplu!).
b. Atribuim prima IP adresă utilizabilă din subreţelele 2 şi 3 celor două interfeţe G0/0 şi G0/1 (pentru link-urile LAN) ale routerului R2, iar ultima IP adresă utilizabilă din subreţeaua 4 se va atribui interfeţei lui R2 din partea link-ului WAN (G0/2).
c. Ultimele IP adrese utilizabile din subreţelele 0,1,2 şi 3 sunt atribuite host-urilor din fiecare subreţea.

Dispozitiv	Interfaţă	IP adresă	Mască de subreţea	Adresa implicită a routerului
R1	G0/0	192.168.100.1	255.255.255.224	N/A
	G0/1	192.168.100.33	255.255.255.224	N/A
	G0/2	192.168.100.129	255.255.255.224	N/A
R2	G0/0	192.168.100.65	255.255.255.224	N/A
	G0/1	192.168.100.97	255.255.255.224	N/A
	G0/2	192.168.100.158	255.255.255.224	N/A
PC0	NIC	192.168.100.30	255.255.255.224	192.168.100.1
PC1	NIC	192.168.100.62	255.255.255.224	192.168.100.33
PC2	NIC	192.168.100.94	255.255.255.224	192.168.100.65
PC3	NIC	192.168.100.126	255.255.255.224	192.168.100.97

Partea 2: Atribuim adrese IP dispozitivelor din reţea şi verificăm conexiunea între ele

Pentru a completa configurarea reţelei, pe fiecare din routerele R1 şi R2 este configurată o rută statică implicită.

Un router foloseşte un tabel de rutare pentru a determina unde să trimită pachetele. Tabelul de rutare conţine informaţie ce precizează interfaţa routerului prin care se va transmite pachetul de date cu o adresă IP de destinaţie specificată.

Iniţial, tabelul de rutare conţine doar informaţia despre reţelele conectate direct la interfeţele routerului. Pentru a comunica cu reţele aflate la distanţă, routerele schimbă între ele informaţia pe care o deţin în tabelele lor de rutare.

Pentru a configura (manual) în tabelul de routare al routerului o rută la o anumită reţea aflată la distanţă, ştiind interfaţa de ieşire a routerului sau adresa IP a următorului hop (router sau host), putem defini pe acesta o rută statică implicită. O rută implicită este un tip de rută statică care specifică ce gateway (interfaţa routerului implicit) să se utilizeze atunci când tabelul de rutare nu conţine o cale pentru reţeaua de destinaţie. O rută statică implicită este o rută statică cu 0.0.0.0 ca adresă IP de destinaţie şi mască de subreţea. Acest tip de rută mai este numită rută "*quad zero*".

Într-o rută implicită, poate fi specificată fie adresa IP a următorului hop, fie interfaţa de ieşire. Pentru a configura o rută statică implicită, folosiţi următoarea comandă:

```
Router(config)# ip route 0.0.0.0  0.0.0.0 {ip-address or exit-intf}
```

Pasul 1: Configurăm interfeţele LAN ale routerelor R1 şi R2

a. Se configurează ambele interfeţe LAN cu adresele indicate în tabelul de adresare (192.168.100.1/27 şi 192.168.100.33/27 pentru R1 şi 192.168.100.65/27 şi 192.168.100.97/27 pentru R2)
b. Se configurează interfeţele astfel încât host-urile din cadrul LAN-urilor să aibă conexiune la routerul implicit

Pasul 2: Configurăm host-urile PC0-PC3

Setăm pe fiecare host adresa IP corespunzătoare din tabel şi adresa routerului implicit

Pasul 3: Verificăm conexiunea

Se va testa prin comanda *ping* dacă este conexiune între oricare două componente ale reţelei create

Exemplu de configurări ale dispozitivelor

R1:

```
enable
configure terminal
interface GigabitEthernet0/0
ip address 192.168.100.1 255.255.255.224
no shutdown
interface GigabitEthernet0/1
ip address 192.168.100.33 255.255.255.224
no shutdown
end
```

PC3:

```
IP address: 192.168.100.126
Subnet Mask: 255.255.255.224
Default gateway: 192.168.100.97
```

Exemplul 4. Subnetarea IPv4 folosind VLSM

Fiind date o adresă de reţea şi numărul de host-uri din fiecare subreţea, vom elabora o schemă VLSM de adresare IPv4, după care, în baza tabelului de adresare vom atribui adrese şi vom configura routerele şi host-urile din cadrul configuraţiilor de reţea examinate.

Dispozitivele utilizate:

Două routere Cisco 2911, 4 switch-uri Cisco 2960, un număr de PC-uri precizat în varianta de laborator corespunzătoare, cabluri Ethernet.

Urmăriţi exemplele rezolvate în continuare, apoi analizaţi în mod analog problema ce v-a fost repartizată.

Se vor examina trei topologii:

Necesităţile de adrese de host-uri pot fi descrise prin intermediul tabelului următor:

LAN	Numărul de adrese necesare
S1Name	NumHostS1
S2Name	NumHostS2
S3Name	NumHostS3
S4Name	NumHostS4

Topologia 1

Este examinat blocul de adrese Net_1=192.168.72.0/24, iar topologia logică de reţea este dată în Figura 3.

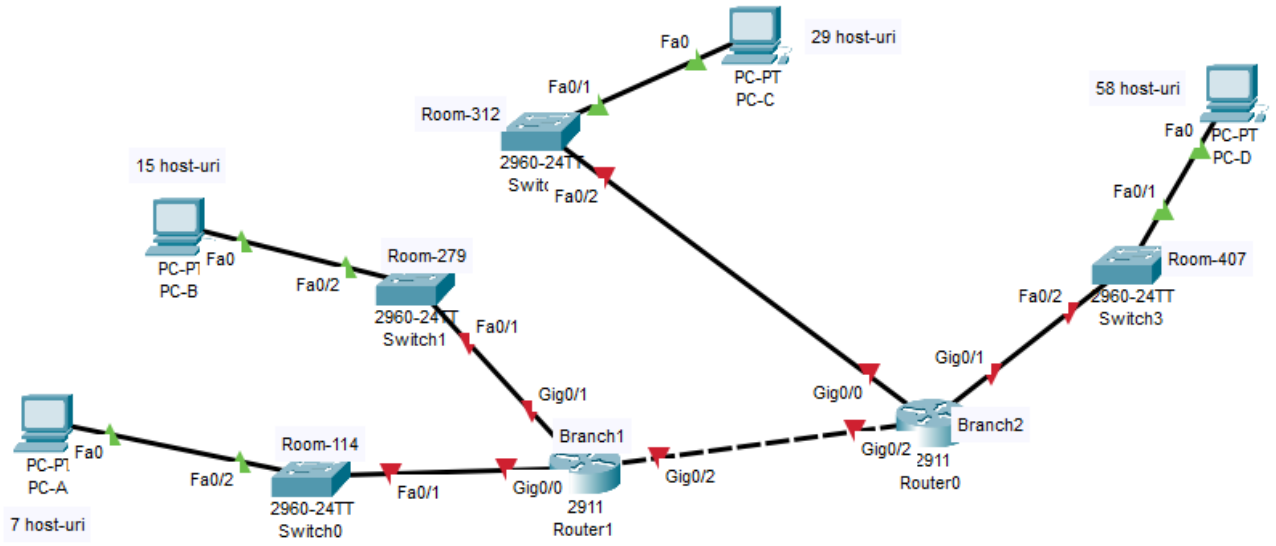


Figura 3

Nume subreţea definită de switch:

S1Name$_1$= **Room-114**, S2Name$_1$= **Room-279**, S3Name$_1$= **Room-312**, S4Name$_1$= **Room-407**,

Nume router din cadrul topologiei:

R1Name$_1$=**Branch1**, R2Name$_2$=**Branch2**

Topologia 2

Este examinat blocul de adrese Net_2=10.11.48.0/24, iar topologia logică de reţea este dată în Figura 4.

Nume reţea definită de switch:

S1Name$_1$= **ASW-1**, S2Name$_1$= **ASW-2**, S3Name$_1$= **ASW-3**, S4Name$_1$= **ASW-4**,

Nume router din cadrul topologiei:

R1Name$_1$=**Building1**, R2Name$_2$=**Building2**

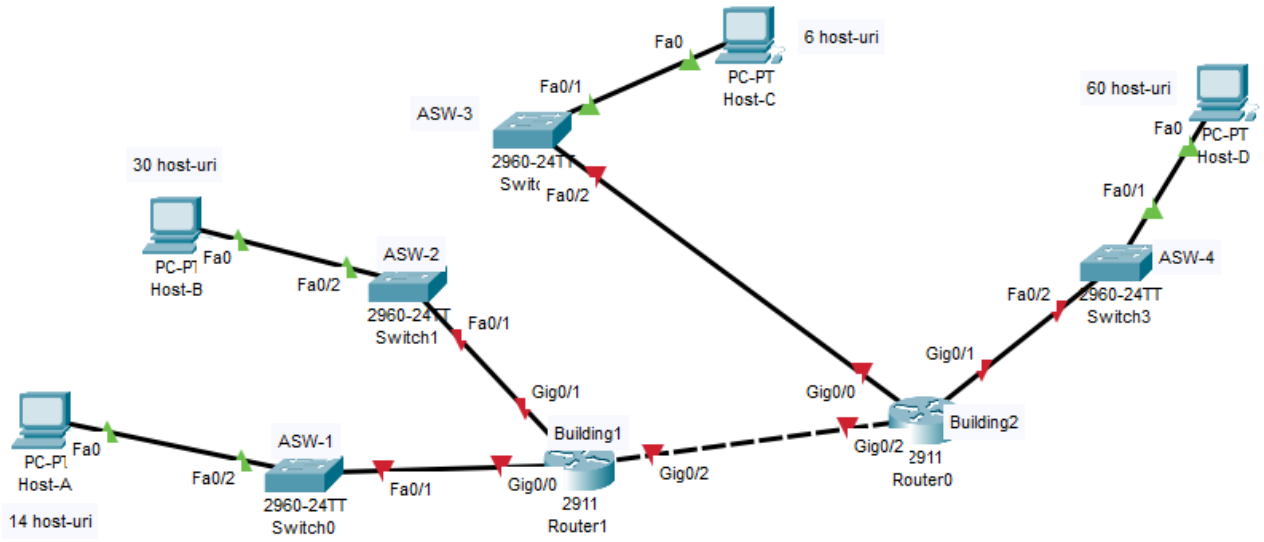


Figura 4

Topologia 3

Este examinat blocul de adrese Net_3=172.31.103.0/24, iar topologia logică de reţea este dată în Figura 5.

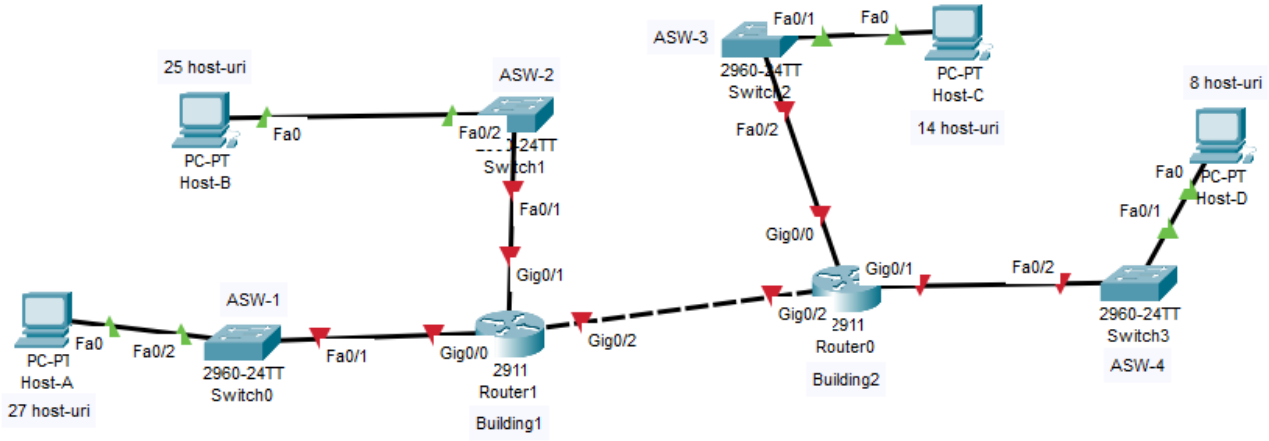


Figura 5

Nume reţea definită de switch:

$S1Name_1$= **ASW-1**, $S2Name_1$= **ASW-2**, $S3Name_1$= **ASW-3**, $S4Name_1$= **ASW-4**,

Nume router din cadrul topologiei:

$R1Name_1$=**Building1**, $R2Name_2$=**Building2**

La fel, avem:

Numărul de adrese IP pentru host-urile din subreţeaua $S1Name_i$ a topologiei $i \in \{1,2,3\}$:

$NumHostS1_1$=7, $NumHostS1_2$=14, $NumHostS1_3$=27

Numărul de adrese IP pentru host-urile din subreţeaua $S2Name_i$ a topologiei $i \in \{1,2,3\}$:

$NumHostS2_1$=15, $NumHostS2_2$=30, $NumHostS2_3$=25

Numărul de adrese IP pentru host-urile din subreţeaua $S3Name_i$ a topologiei $i \in \{1,2,3\}$:

$NumHostS3_1$=29, $NumHostS3_2$=6, $NumHostS3_3$=14

Numărul de adrese IP pentru host-urile din subreţeaua $S4Name_i$ a topologiei $i \in \{1,2,3\}$:

$NumHostS4_1$=58, $NumHostS4_2$=60, $NumHostS4_3$=8

Obiective

Partea 1: Examinarea condiţiilor impuse asupra reţelei

Partea 2: Elaborarea schemei de subnetare (adresare) VLSM

Partea 3: Atribuirea de adrese IP dispozitivelor şi verificarea conexiunii

Scenariu

În această activitate, vi se oferă o adresă de reţea cu prefixul /24, pe care să o utilizaţi pentru a proiecta o schemă de adresare IPv4, folosind procedeul VLSM. În baza unui set de cerinţe, veţi atribui adrese IPv4 subreţelelor şi veţi configura dispozitivele reţelei, verificând conexiunea dintre acestea.

Instrucţiuni

Partea 1: Examinarea condiţiilor impuse asupra reţelei

Pasul 1: Se determină numărul necesar de subreţele

Se va subneta adresa de reţea Net_i, i∈{1,2,3}. Reţeaua impune următorele condiţii:
Subreţeaua $S1Name_i$ necesită $NumHostS1_i$ IP adrese pentru host-uri
Subreţeaua $S2Name_i$ necesită $NumHostS2_i$ IP adrese pentru host-uri
Subreţeaua $S3Name_i$ necesită $NumHostS3_i$ IP adrese pentru host-uri
Subreţeaua $S4Name_i$ necesită $NumHostS4_i$ IP adrese pentru host-uri
Câte subreţele sunt necesare în topologia de reţea? 5 (4+1 subreţea pe link-ul dintre routere)

Pasul 2: Pentru fiecare subreţea se determină masca de subreţea

a Ce mască de subreţea va asigura numărul de IP adrese necesare pentru $S1Name_i$, i∈{1,2,3}? Câte adrese utilizabile de host-uri va suporta această subreţea?

b. Ce mască de subreţea va asigura numărul de IP adrese necesare pentru $S2Name_i$, i∈{1,2,3}? Câte adrese utilizabile de host-uri va suporta această subreţea?

c Ce mască de subreţea va asigura numărul de IP adrese necesare pentru $S3Name_i$, i∈{1,2,3}? Câte adrese utilizabile de host-uri va suporta această subreţea?

d. Ce mască de subreţea va asigura numărul de IP adrese necesare pentru $S4Name_i$, i∈{1,2,3}? Câte adrese utilizabile de host-uri va suporta această subreţea?

e. Ce mască de subreţea va asigura numărul de IP adrese necesare pentru conexiunea dintre routerele $R1Name_i$ şi $R2Name_i$, i∈{1,2,3}?

Partea 2: Elaborarea schemei de subnetare (adresare) VLSM

Pasul 1: Se divide reţeaua Net_i, i∈{1,2,3} în baza numărului de host-uri per subreţea

a) Folosiţi prima subreţea pentru a asigura numărul necesar de adrese în LAN-ul cu cele mai multe host-uri

I. Pentru adresa de reţea 10.11.48.0/24 (00001010.00001011.00110000.00000000 în binar) avem:

Subreţeaua cu cele mai multe host-uri necesită 60 de host-uri. Pentru a asigura atâtea adrese de host-uri este necesar ca identificatorul de host-uri al IP adresei să fie pe 6 biţi (2^6-2=62), iar atunci masca de reţea extinsă va fi pe 26 de biţi – 255.255.255.192. Avem posibilitate de a genera 4 subreţele, variind cu biţii 25 şi 26 ai IP adresei de reţea (masca a fost extinsă de la 24 la 26):

Subreţeaua 1: 00001010.00001011.00110000.**00**000000=10.11.48.0/26
Subreţeaua 2: 00001010.00001011.00110000.**01**000000=10.11.48.64/26
Subreţeaua 3: 00001010.00001011.00110000.**10**000000=10.11.48.128/26
Subreţeaua 4: 00001010.00001011.00110000.**11**000000=10.11.48.192/26

Astfel, putem atribui primei subreţele din 60 de host-uri adresa de subreţea 10.11.48.0/26.

II. Pentru adresa de reţea 172.31.103.0/24 (10101100.00011111.01100111.00000000 în binar) avem:

Subreţeaua cu cele mai multe host-uri necesită 27 de host-uri. Pentru a asigura atâtea adrese de host este necesar ca identificatorul de host al IP adresei să fie pe 5 biţi (2^5-2=30), iar atunci masca de reţea extinsă va fi pe 32-5=27 de biţi – 255.255.255.224. Avem posibilitate de a genera 2^3=8 subreţele, variind cu biţii 25, 26 şi 27 ai IP adresei de reţea (masca a fost extinsă de la 24 la 27):

Subreţeaua 1: 10101100.00011111.01100111.**000**00000 =172.31.103.0/27
Subreţeaua 2: 10101100.00011111.01100111.**001**00000 =172.31.103.32/27
Subreţeaua 3: 10101100.00011111.01100111.**010**00000 =172.31.103.64/27
Subreţeaua 4: 10101100.00011111.01100111.**011**00000 =172.31.103.96/27
Subreţeaua 5: 10101100.00011111.01100111.**100**00000 =172.31.103.128/27
Subreţeaua 6: 10101100.00011111.01100111.**101**00000 =172.31.103.160/27
Subreţeaua 7: 10101100.00011111.01100111.**110**00000 =172.31.103.192/27
Subreţeaua 8: 10101100.00011111.01100111.**111**00000 =172.31.103.224/27

Astfel, putem atribui primei subreţele din 27 de host-uri adresa de subreţea 172.31.103.0/27.
De menţionat că fiecăreia din cele 8 subreţele formate mai sus îi corespunde aceeaşi mască de subreţea - 255.255.255.224 (/27)

III. Pentru adresa de reţea 192.168.72.0/24 (11000000.10101000.01001000.00000000 în binar) avem:

Subreţeaua cu cele mai multe host-uri, PC-D LAN, necesită 58 de host-uri. Pentru a asigura atâtea adrese de host este necesar ca identificatorul de host al IP adresei să fie pe 6 biţi (2^6-2=62), iar atunci masca de reţea extinsă va fi pe 32-6=26 de biţi – 255.255.255.192. Avem posibilitate de a genera 2^2=4 subreţele, variind cu biţii 25 şi 26 ai IP adresei de reţea (masca a fost extinsă de la 24 la 26):

Subreţeaua 1: 11000000.10101000.01001000.**00**000000 =192.168.72.0/26
Subreţeaua 2: 11000000.10101000.01001000.**01**000000 =192.168.72.64/26
Subreţeaua 3: 11000000.10101000.01001000.**10**000000 =192.168.72.128/26
Subreţeaua 4: 11000000.10101000.01001000.**11**000000 =192.168.72.192/26

Astfel, putem atribui primei subreţele din 58 de host-uri adresa de subreţea 192.168.72.0/26.
De menţionat că fiecăreia din cele 4 subreţele formate mai sus îi corespunde aceeaşi mască de subreţea - 255.255.255.192 (/26)

b) Folosiţi a doua subreţea pentru a asigura numărul necesar de adrese în LAN-ul care se află pe locul doi după numărul de host-uri

I. Pentru adresa de reţea 10.11.48.0/24 (00001010.00001011.00110000.00000000 în binar) avem:

Folosim a doua subreţea 10.11.48.64/26 de la punctul a). Deoarece în subreţeaua cu 30 de host-uri sunt necesari 5 biţi (2^5-2=30) pentru a asigura cu IP adrese aceste host-uri, vom aplica masca extinsă /27. Deci vom avea două subreţele

00001010.00001011.00110000.01**0**00000=10.11.48.64/27
00001010.00001011.00110000.01**1**00000=10.11.48.96/27

dintre care prima 10.11.48.64/27 o asociem cu Host-B LAN, iar a doua 10.11.48.96/27 o vom utiliza în continuare.
Pentru a asigura numărul necesar de adrese în LAN-ul care se află pe locul trei după numărul de host-uri (14 host-uri în Host-C LAN) avem nevoie de 4 biţi (2^4-2=14) ai identificatorului de reţea din adresa 10.11.48.96/27. Astfel, vom avea masca extinsă pe 28 de biţi, adică în baza bitului 28 putem genera IP adrese pentru încă două subreţele:

00001010.00001011.00110000.011**0**0000=10.11.48.96/28
00001010.00001011.00110000.011**1**0000=10.11.48.112/28

Prima IP adresă de subreţea o atribuim lui Host-C LAN, iar a doua 10.11.48.112/28 o vom utiliza în continuare.
Pentru a asigura numărul necesar de adrese în LAN-ul care se află pe locul patru după numărul de host-uri (6 host-uri în Host-D LAN) avem nevoie de 3 biţi (2^3-2=6) ai identificatorului de reţea din adresa 10.11.48.112/28. Astfel, vom avea masca extinsă pe 29 de biţi, adică în baza bitului 29 putem genera IP adrese pentru încă două subreţele:

00001010.00001011.00110000.0110**0**000=10.11.48.112/29
00001010.00001011.00110000.0111**1**000=10.11.48.120/29

Prima IP adresă de subreţea o atribuim lui Host-D LAN, iar a doua 10.11.48.120/29 o vom utiliza în continuare.
Pentru a asigura numărul necesar de adrese în LAN-ul care asigură conexiunea dintre routerele $R1Name_i$ şi $R2Name_i$ (2 IP adrese la interfeţele corespunzătoare ale routerelor) sunt necesari 2 biţi (2^2-2=2) ai identificatorului de reţea din adresa 10.11.48.120/29. Astfel, vom avea masca extinsă pe 30 de biţi, adică în baza bitului 30 putem genera IP adrese pentru încă două subreţele:

00001010.00001011.00110000.01100**0**00=10.11.48.112/30
00001010.00001011.00110000.01111**1**00=10.11.48.124/30

Prima IP adresă de subreţea 10.11.48.112/30 o atribuim LAN-ului dintre routere.

II. Pentru adresa de reţea 172.31.103.0/24 (10101100.00011111.01100111.00000000 în binar) avem:

Folosim a doua subreţea 172.31.103.32/27 de la punctul a). Deoarece în subreţeaua cu 25 de host-uri sunt necesari 5 biţi (2^5-2=30) pentru a asigura cu IP adrese aceste host-uri, vom aplica masca extinsă /27. Astfel pentru Host-B LAN putem folosi în calitate de adresă de subreţea a doua adresă de la punctul a) – 172.31.103.32/27.
Deoarece a treia după numărul de host-uri reţea include 14 host-uri, sunt necesari 4 biţi (2^4-2=14) pentru a asigura cu IP adrese aceste host-uri. Astfel, vom aplica masca extinsă /28 la a treia subreţea de la punctul a) - 172.31.103.64/27. Vom obţine două subreţele:

10101100.00011111.01100111.010**0**0000=172.31.103.64/28
10101100.00011111.01100111.010**1**0000=172.31.103.80/28

dintre care prima 172.31.103.64/28 o asociem cu Host-C LAN, iar a doua 172.31.103.80/28 o vom utiliza în continuare.
Pentru a asigura numărul necesar de adrese în LAN-ul Host-D cu 8 host-uri avem nevoie de 4 biţi (2^4-2=14) pentru identificatorul de host, adică o mască extinsă pe 32-4=28 de biţi. De aceea, putem utiliza IP adresa 172.31.103.80/28 a celei de-a doua subreţele de la cazul anterior.
Pentru a asigura numărul necesar de adrese în LAN-ul care asigură conexiunea dintre routerele R1Name$_i$ şi R2Name$_i$ (2 IP adrese la interfeţele corespunzătoare ale routerelor) sunt necesari 2 biţi (2^2-2=2) pentru identificatorul de host. Astfel, vom avea masca extinsă pe 30 de biţi. Implicăm IP adresa subreţelei a patra de la punctul a) 172.31.103.96/27:

10101100.00011111.01100111.011**000**00/30=172.31.103.96/30
10101100.00011111.01100111.011**001**00/30=172.31.103.100/30

……
Prima IP adresă de subreţea 172.31.103.96/30 o atribuim LAN-ului dintre routere.

III. Pentru adresa de reţea 192.168.72.0/24 avem:
Folosim a doua subreţea 192.168.72.64/26 de la punctul a). Deoarece în subreţeaua cu 29 de host-uri sunt necesari 5 biţi (2^5-2=30) pentru a asigura cu IP adrese aceste host-uri, vom aplica masca extinsă /27 (32-5=27). Folosind a doua adresă de subreţea de la punctul a) – 192.168.72.64/26 şi masca extinsă /27, putem forma două subreţele:

11000000.10101000.01001000.01**0**00000=192.168.72.64/27
11000000.10101000.01001000.01**1**00000=192.168.72.96/27

Astfel pentru PC-C LAN putem folosi în calitate de adresă de subreţea, de exemplu, adresa 192.168.72.64/27
Deoarece a treia după numărul de host-uri subreţea, PC-B LAN include 15 host-uri, sunt necesari 5 biţi (2^5-2=30) pentru a asigura cu IP adrese aceste host-uri. Astfel, vom aplica o mască extinsă /27. Deoarece avem deja stabilită o adresă de subreţea /27 - 192.168.72.96/27 - o vom atribui subreţelei PC-B LAN.
Deoarece a patra după numărul de host-uri subreţea, PC-A LAN include 7 host-uri, sunt necesari 4 biţi (2^4-2=14) pentru a asigura cu IP adrese aceste host-uri. Astfel, vom aplica o mască extinsă /28. Vom implica a treia subreţea de la punctul a) – 192.168.72.128/26. Vom obţine 4 subreţele (se variază cu biţii 27 şi 28):

11000000.10101000.01001000.10**00**0000=192.168.72.128/28
11000000.10101000.01001000.10**01**0000=192.168.72.144/28
11000000.10101000.01001000.10**10**0000=192.168.72.160/28
11000000.10101000.01001000.10**11**0000=192.168.72.176/28

dintre care prima 192.168.72.128/28 o asociem cu PC-A LAN, iar a doua 192.168.72.144/28 o vom utiliza în continuare.
Pentru a asigura numărul necesar de adrese în LAN-ul care asigură conexiunea dintre routerele R1Name$_i$ şi R2Name$_i$ (2 IP adrese la interfeţele corespunzătoare ale routerelor) sunt necesari 2 biţi (2^2-2=2) pentru identificatorul de host. Astfel, vom avea masca extinsă /30. În baza IP adresei 192.168.72.144/28 obţinem:

11000000.10101000.01001000.1001**00**00/30=192.168.72.144/30
11000000.10101000.01001000.1001**01**00/30=192.168.72.148/30

……

Prima IP adresă de subreţea 192.168.72.144/30 o atribuim LAN-ului dintre routere.

Pasul 2: Documentaţi subreţelele VLSM

Completaţi tabelul pentru subreţele (a se vedea Tabelul 5), care conţine descrierea fiecărei subreţele (de exemplu, S1Name$_i$ LAN), numărul necesar de host-uri, adresa subreţelei, prima şi ultima adresă de host utilizabilă şi adresa de broadcast.

Descrierea subreţelei	*Numărul necesar de host-uri*	*Adresa reţelei/CIDR*	*Prima adresă de host utilizabilă*	*Ultima adresă de host utilizabilă*	*Adresa de broadcast*

Tabelul 5

De exemplu, pentru reţeaua **10.11.48.0/24** avem următorul tabel pentru subreţele:

Descrierea subreţelei	*Numărul necesar de host-uri*	*Adresa reţelei/CIDR*	*Prima adresă de host utilizabilă*	*Ultima adresă de host utilizabilă*	*Adresa de broadcast*
Host-D LAN	*60*	*10.11.48.0/26*	*10.11.48.1*	*10.11.48.62*	*10.11.48.63*
Host-B LAN	*30*	*10.11.48.64/27*	*10.11.48.65*	*10.11.48.94*	*10.11.48.95*
Host-A LAN	*14*	*10.11.48.96/28*	*10.11.48.97*	*10.11.48.110*	*10.11.48.111*
Host-C LAN	*6*	*10.11.48.112/29*	*10.11.48.113*	*10.11.48.118*	*10.11.48.119*
Link-ul WAN	*2*	*10.11.48.120/30*	*10.11.48.121*	*10.11.48.122*	*10.11.48.123*

Pentru reţeaua **172.31.103.0/24** avem următorul tabel pentru subreţele:

Descrierea subreţelei	*Numărul necesar de host-uri*	*Adresa reţelei/CIDR*	*Prima adresă de host utilizabilă*	*Ultima adresă de host utilizabilă*	*Adresa de broadcast*
Host-A LAN	*27*	*172.31.103.0/27*	*172.31.103.1*	*172.31.103.30*	*172.31.103.31*
Host-B LAN	*25*	*172.31.103.32/27*	*172.31.103.33*	*172.31.103.62*	*172.31.103.63*
Host-C LAN	*14*	*172.31.103.64/28*	*172.31.103.65*	*172.31.103.78*	*172.31.103.79*
Host-D LAN	*8*	*172.31.103.80/28*	*172.31.103.81*	*172.31.103.94*	*172.31.103.95*
Link-ul WAN	*2*	*172.31.103.96/30*	*172.31.103.97*	*172.31.103.98*	*172.31.103.99*

Pentru reţeaua **192.168.72.0/24** avem următorul tabel pentru subreţele:

Descrierea subreţelei	*Numărul necesar de host-uri*	*Adresa reţelei/CIDR*	*Prima adresă de host utilizabilă*	*Ultima adresă de host utilizabilă*	*Adresa de broadcast*
PC-A LAN	*7*	*192.168.72.128/28*	*192.168.72.129*	*192.168.72.142*	*192.168.72.143*
PC-B LAN	*15*	*192.168.72.96/27*	*192.168.72.97*	*192.168.72.126*	*192.168.72.127*
PC-C LAN	*29*	*192.168.72.64/27*	*192.168.72.65*	*192.168.72.94*	*192.168.72.95*
PC-D LAN	*58*	*192.168.72.0/26*	*192.168.72.1*	*192.168.72.62*	*192.168.72.63*
Link-ul WAN	*2*	*192.168.72.144/30*	*192.168.72.145*	*192.168.72.146*	*192.168.72.147*

Pasul 3: Documentaţi schema de adresare

a. Atribuiţi primele IP adrese utilizabile din subreţelele corespunzătoare interfeţelor G0/1 şi G0/2 ale routerului **R1Name**$_i$. Atribuiţi interfeţei G0/2 a routerului **R1Name**$_i$ prima IP adresă utilizabilă stabilită pe link-ul WAN

b. Atribuiţi primele IP adrese utilizabile din subreţelele corespunzătoare interfeţelor G0/1 şi G0/2 ale routerului **R2Name**$_i$ ce conectează cele două link-uri LAN. Atribuiţi interfeţei G0/2 a routerului **R2Name**$_i$ ultima IP adresă utilizabilă stabilită pe link-ul WAN

c. Atribuiţi host-urilor ultimele IP adrese utilizabile

Dacă schema de adrese şi implementarea acesteia sunt corecte, atunci trebuie să existe conexiune între toate host-urile şi dispozitivele din reţea

Partea 3: Atribuiţi dispozitivelor IP adrese şi verificaţi conexiunea dintre dispozitive

Efectuaţi următorii paşi pentru a completa configurarea de adrese:

Pasul 1: Folosind tabelul cu adrese elaborat, atribuiţi câte o IP adresă şi o mască de subreţea fiecărei interfeţe a routerelor R1Name$_i$ şi R2Name$_i$. Activaţi interfeţele. Pe fiecare router trebuie să existe un protocol de rutare pentru ca alte dispozitive să fie „conştiente" de subreţele celuilalt router.

Pasul 2: Configuraţi IP adrese pe toate host-urile, inclusiv adresa routerului implicit

Pasul 3: Verificaţi conexiunea

Pentru topologia 10.11.48.0/24:

Building 1

```
en
conf ter
int g0/0
ip add 10.11.48.97  255.255.255.240
no shut
int g0/1
ip add 10.11.48.65  255.255.255.224
no shut
int g0/2
ip add 10.11.48.121  255.255.255.252
no shut
```

Pentru topologia 192.168.72.0/24:

Branch1

```
en
conf ter
int g0/0
ip add 192.168.72.129  255.255.255.240
no shut
int g0/1
ip add 192.168.72.97  255.255.255.224
no shut
int g0/2
ip add 192.168.72.145  255.255.255.252
no shut
```

Pentru topologia 172.31.103.0/24:

Building 1

```
en
conf ter
int g0/0
ip add 172.31.103.1  255.255.255.224
no shut
int g0/1
ip add 172.31.103.33  255.255.255.224
no shut
int g0/2
ip add 172.31.103.97  255.255.255.252
no shut
```

Folosind comanda *ping* se va testa conexiunea dintre dispozitive.

Lucrarea de laborator №5

Rutare statică şi protocoale de rutare dinamică

Scopul lucrării constă în formarea abilităţilor practice de configurare statică şi dinamică în Cisco Packet

Tracer a tabelelor de rutare ale routerelor din componenţa reţelelor

Obiective:

- Prezentarea conceptului de rutare şi de tabel de rutare pe router
- Realizarea rutării statice pe routerele unei reţele
- Configurarea protocoalelor de rutare dinamică RIP, EIGRP şi OSPF
- Configurarea protocolului OSPF pe mai multe domenii
- Prezentarea conceptului de redistribuire a rutelor între sistemele autonome
- Configurarea protocolului de rutare dinamică BGP între sistemele autonome

Indicaţii metodice privind realizarea lucrării

Prin rutare se înţelege procesul prin care routerul determină automat în baza tabelului său de rutare traseul pe care trebuie transmis un pachet de date recepţionat pentru a ajunge la destinaţie.

Tabelul de rutare include anumite rute până la subreţelele reţelei iniţiale. Fiecare rută (pe lângă IP-ul şi masca reţelei destinaţie) defineşte interfaţa routerului prin care se intră în reţeaua destinaţie.

Iniţial, la conectarea routerului în reţea, în tabelul de rutare al acestuia sunt scrise adresele subreţelelor direct conectate la acesta. Pentru a avea posibilitate de transmis date către celelalte reţele, în tabelul de rutare este necesar să se definească rute corespunzătoare până la acestea. Definirea rutelor se poate face manual sau în mod automatizat. Procedeul prin care se definesc manual rutele pe fiecare router al reţelei este numit rutare statică, iar procedeul automatizat este realizat printr-un protocol de rutare dinamică.

Rutarea statică

Vom ilustra mai întâi cum se configurează manual o rută statică (numită recursivă) şi rutele statice implicite (care specifică IP-ul interfeţei routerului către care se va transmite pachetul de date, atunci când tabelul de rutare al routerului curent nu include un traseu până la reţeaua destinaţie).

Sintaxa aplicată la definirea unei rute statice recursive este următoarea:

```
Router(config)# ip route network-address subnet-mask ip-address
```

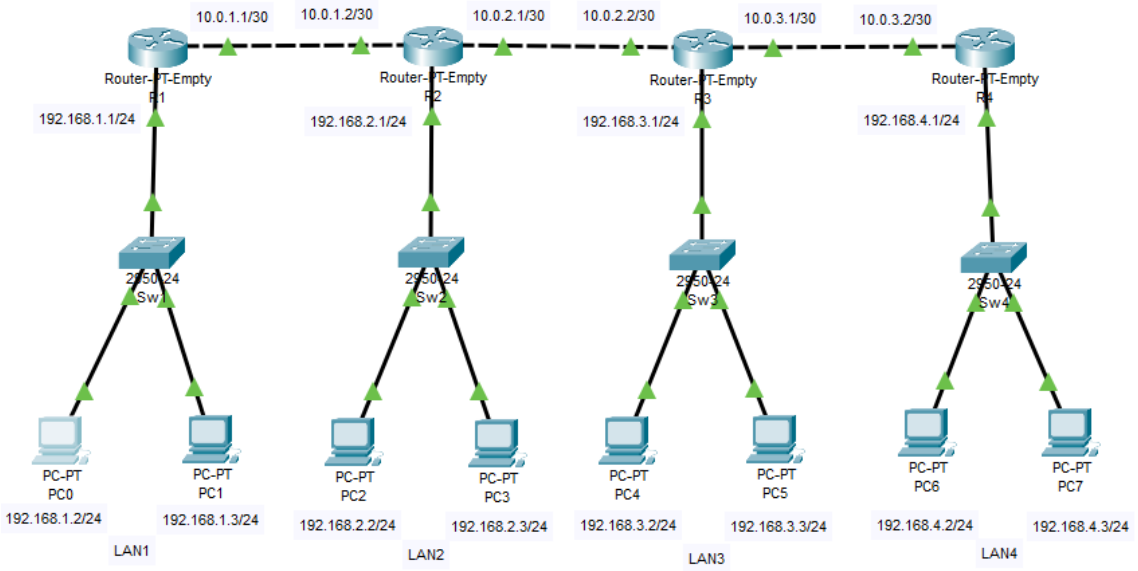
unde network-address şi subnet-mask sunt adresa şi masca subreţelei destinaţie, iar ip-address este IP adresa interfeţei de intrare în următorul router în calea către subreţeaua destinaţie.

Sintaxa aplicată la definirea unei rute statice implicite este următoarea:

```
Router(config)# ip route 0.0.0.0 0.0.0.0 {ip-address}
```

unde 0.0.0.0 0.0.0.0 este IP-ul şi masca destinaţie, iar ip-address – adresa IP a interfeţei de intrare în următorul router în calea către subreţeaua destinaţie.

Să examinăm reţeaua compusă din 4 subreţele (în raport cu switch-urile)



Dacă dăm un ping de pe PC0 către PC6, routerul R1 ne va anunţa că „*Destination host unreachable*”

```
C:\>ping 192.168.4.2

Pinging 192.168.4.2 with 32 bytes of data:

Reply from 192.168.1.1: Destination host unreachable.
Reply from 192.168.1.1: Destination host unreachable.
Reply from 192.168.1.1: Destination host unreachable.
Reply from 192.168.1.1: Destination host unreachable.

Ping statistics for 192.168.4.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Routerul R0 nu ştie cum să transmită pachetul până la subreţeaua LAN4 => trebuie să-i spunem acestui router cum să ajungă la subreţeaua LAN4, adică să-i indicăm ruta!
În acest sens, pentru a accede în subreţeaua LAN4 urmează să se transmită pachetul către routerul R2, iar lui R2 îi „vom spune" că pentru a ajunge în LAN4 urmează să transmită pachetul lui R3.
La fel lui R3 îi „vom spune" că pentru a ajunge în LAN4 urmează să transmită pachetul lui R4, iar acesta din urmă deja „ştie ce trebuie să facă" în subreţeaua (domeniu broadcast) ce este conectată direct la el.
În Cisco Packet Tracer vom utiliza comanda *ip route*

Pentru routerul R1:
Router>en Router#conf ter Router(config)#ip route 192.168.4.0 255.255.255.0 10.0.1.2
Pentru routerul R2:
Router>en Router#conf ter Router(config)#ip route 192.168.4.0 255.255.255.0 10.0.2.2
Pentru routerul R3:
Router>en Router#conf ter Router(config)#ip route 192.168.4.0 255.255.255.0 10.0.3.2

Totuşi, dacă dăm din nou un ping de pe PC0 către PC6, vom fi refuzaţi. De ce?
În modul de simulare vedem că pachetul ajunge până la destinaţie, dar când se întoarce – se blochează la routerul R4. Aceasta înseamnă că trebuie de indicat şi calea înapoi, de la destinaţie la sursă:

Pentru routerul R4:
Router>en Router#conf ter Router(config)#ip route 192.168.1.0 255.255.255.0 10.0.3.1
Pentru routerul R3:
Router#conf ter Router(config)#ip route 192.168.1.0 255.255.255.0 10.0.2.1
Pentru routerul R2:
Router>en Router#conf ter Router(config)#ip route 192.168.1.0 255.255.255.0 10.0.1.1

În acest moment, ping-ul către PC6 trece cu succes:

```
C:\>ping 192.168.4.2

Pinging 192.168.4.2 with 32 bytes of data:

Reply from 192.168.4.2: bytes=32 time<1ms TTL=124
Reply from 192.168.4.2: bytes=32 time<1ms TTL=124
Reply from 192.168.4.2: bytes=32 time=6ms TTL=124
Reply from 192.168.4.2: bytes=32 time=1ms TTL=124

Ping statistics for 192.168.4.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 6ms, Average = 1ms
```

Am scris ruta de la prima la a patra subreţea
Dar analog trebuie de scris rutele între oricare două subreţele
Mai trebuie de completat rutele dintre subreţelele: de la 1 la 2, de la 1 la 3, de la 2 la 3, de la 3 la 2, de la 4 la 3, de la 4 la 2
Astfel, pentru a asigura accesul la aceste subreţele, am fost nevoiţi să scriem rutele indicate
Dacă se închide accesul la istoricul comenzilor introduse în linia de comandă a routerului, atunci pot fi

scrise următoarele două comenzi pentru a evita o astfel de situaţie:

```
Router(config)#line console 0
Router(config-line)#exec-timeout ?
  <0-35791>  Timeout in minutes
Router(config-line)#exec-timeout 0 0
```

Cum putem vizualiza rutele pe care le-am scris? În tabelul de rutare! Aceasta din urmă poate fi vizualizată pe fiecare router (tabelul de rutare al routerului respectiv!), folosind comanda *show ip route* din modul privilegiat sau comanda *do show ip route* din modul de configurare globală:
De exemplu, pe routerul R2 avem:

```
Router>en
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route

Gateway of last resort is not set

10.0.0.0/30 is subnetted, 2 subnets
C 10.0.1.0 is directly connected, FastEthernet9/0
C 10.0.2.0 is directly connected, FastEthernet8/0
S 192.168.1.0/24 [1/0] via 10.0.1.1
C 192.168.2.0/24 is directly connected, FastEthernet7/0
S 192.168.4.0/24 [1/0] via 10.0.2.2
```

```
Router#conf ter
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#do show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/30 is subnetted, 2 subnets
C       10.0.1.0 is directly connected, FastEthernet9/0
C       10.0.2.0 is directly connected, FastEthernet8/0
S    192.168.1.0/24 [1/0] via 10.0.1.1
C    192.168.2.0/24 is directly connected, FastEthernet7/0
S    192.168.4.0/24 [1/0] via 10.0.2.2
```

Rutele notate cu S sunt cele statice, adică cele setate manual pe routerul respectiv
Rutele notate cu C sunt cele conectate direct la routerul curent
Analog pot fi vizualizate rutele şi pe celelalte routere
Ceea ce am realizat acum se numeşte rutare statică!
Dar în reţelele organizaţiilor cu mai mult de 5 routere se foloseşte rutarea dinamică!

Redistribuirea rutelor între protocoalele de rutare dinamică

Deseori în diferite sisteme autonome (domenii de rutare) rulează diferite protocoale de rutare dinamică. Chiar şi în cadrul unei reţele compuse pe anumite segmente ar putea să fie configurate diferite protocoale de rutare.

În acest caz, pentru a asigura transmiterea pachetelor de date între oricare două dispozitive din reţea, se aplică procedeul de redistribuire a rutelor, care permite ca rutele generate de un protocol să fie transmise spre utilizare unui alt protocol de rutare (se asigură comunicare între protocoalele de rutare). Protocolul de rutare care a recepţionat rutele redistribuite, le etichetează pe acestea din urmă ca fiind rute externe.

Între două domenii de rutare trebuie să existe cel puţin un punct de redistribuire - un router pe care vor rula ambele protocoale de rutare din cele două domenii.

Este posibil de realizat redistribuirea rutelor unui protocol în acelaşi protocol ce rulează în alt domeniu de rutare. La fel, pot fi redistribuite rutele statice într-un protocol de rutare dinamică.

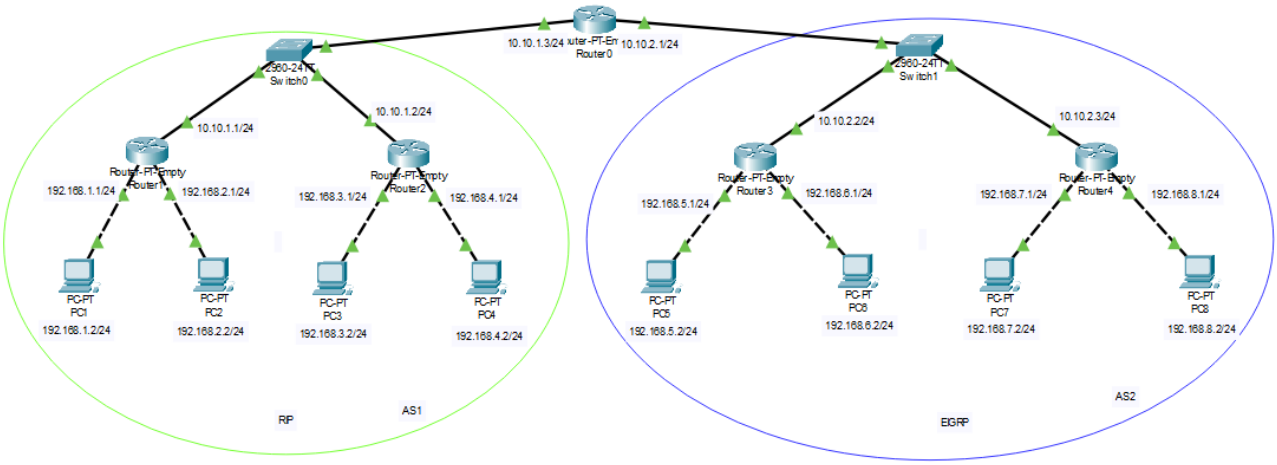
Pot fi redistribuite acele rute care se conţin în tabelul de rutare al routerului.

Metrica de rutare reprezintă componenta de bază în redistribuirea rutelor. Cu excepţia lui EIGRP, celelalte protocoale utilzează o metrică unică. Rutelor redistribuite trebuie să li se asocieze manual o metrică pe care o „înţelege" protocolul ce recepţionează aceste rute.

Redistribuirea rutelor între RIP şi EIGRP

Amintim că RIP şi EIGRP sunt protocoale bazat pe vectori distanţă, care utilzează în calitate de metrică numărul de hop-uri (routere) intermediare până la reţeaua destinaţie.

Să considerăm reţeaua următoare:



În cele 5 subreţele din AS1 este configurat protocolul de rutare dinamică RIP, iar în AS2 – protocolul EIGRP. Vom redistribui rutele între AS1 şi AS2 prin intermediul routerului Router0. În acest sens, mai întâi vom configura pe Router0 atât protocolul RIP, cât şi EIGRP, declarând pentru RIP reţeaua 10.10.1.0/24, iar pentru EIGRP – reţeaua 10.10.2.0/24:

```
Router(config)#router rip
Router(config-router)#version 2
Router(config-router)#network 10.10.1.0
Router(config-router)#no auto-summary
Router(config-router)#exit
Router(config)#router eigrp 1
Router(config-router)#network 10.10.2.0 0.0.0.255
Router(config-router)#no auto-summary
Router(config-router)#do wr
```

Pe Router0 se redistribuie rutele RIP în EIGRP şi reciproc:

```
Router(config)#router eigrp 1
Router(config-router)#redistribute rip metric 10000 100 255 1 1500
Router(config-router)#exit
```

```
Router(config)#router rip
Router(config-router)#redistribute eigrp 1 metric 1
Router(config-router)#exit
Router(config)#do wr
```

EIGRP implică cinci valori atunci când redistribuie rutele din alte protocoale (în loc de *redistribute rip* poate fi *redistribute ospf*, *redistribute static* ş.a.): lăţimea de bandă (bandwidth), întârzierea (delay), fiabilitatea (reliability), încărcarea (load) şi, respectiv, MTU.
În rezultatul execuţiei comenzilor menţionate mai sus, tabelele de rutare pentru Router1 şi Router2 vor include toate rutele din configuraţia de reţea:

```
     10.0.0.0/24 is subnetted, 2 subnets
C       10.10.1.0 is directly connected, GigabitEthernet7/0
R       10.10.2.0 [120/1] via 10.10.1.3, 00:00:08, GigabitEthernet7/0
C    192.168.1.0/24 is directly connected, GigabitEthernet9/0
C    192.168.2.0/24 is directly connected, GigabitEthernet8/0
R    192.168.3.0/24 [120/1] via 10.10.1.2, 00:00:14,
GigabitEthernet7/0
R    192.168.4.0/24 [120/1] via 10.10.1.2, 00:00:14,
GigabitEthernet7/0
R    192.168.5.0/24 [120/1] via 10.10.1.3, 00:00:08,
GigabitEthernet7/0
R    192.168.6.0/24 [120/1] via 10.10.1.3, 00:00:08,
GigabitEthernet7/0
R    192.168.7.0/24 [120/1] via 10.10.1.3, 00:00:08,
GigabitEthernet7/0
R    192.168.8.0/24 [120/1] via 10.10.1.3, 00:00:08,
GigabitEthernet7/0
```

La fel, şi tabelele de rutare pentru Router3 şi Router 4:

```
     10.0.0.0/24 is subnetted, 2 subnets
D EX    10.10.1.0 [170/258560] via 10.10.2.1, 00:08:15,
GigabitEthernet9/0
C       10.10.2.0 is directly connected, GigabitEthernet9/0
D EX 192.168.1.0/24 [170/258560] via 10.10.2.1, 00:08:14,
GigabitEthernet9/0
D EX 192.168.2.0/24 [170/258560] via 10.10.2.1, 00:08:14,
GigabitEthernet9/0
D EX 192.168.3.0/24 [170/258560] via 10.10.2.1, 00:08:14,
GigabitEthernet9/0
D EX 192.168.4.0/24 [170/258560] via 10.10.2.1, 00:08:14,
GigabitEthernet9/0
C    192.168.5.0/24 is directly connected, GigabitEthernet7/0
C    192.168.6.0/24 is directly connected, GigabitEthernet8/0
D    192.168.7.0/24 [90/7680] via 10.10.2.3, 00:08:14,
GigabitEthernet9/0
D    192.168.8.0/24 [90/7680] via 10.10.2.3, 00:08:14,
GigabitEthernet9/0
```

Dacă dăm un ping între oricare două host-uri din reţea, vom vedea că există conexiune între acestea.
Redistribuirea EIGRP într-un alt proces EIGRP nu necesită careva conversie de metrică, deci nu este necesar să se definească vreo metrică (de exemplu, cea predefinită) pe parcursul redistribuirii.
Pentru redistribuirea în EIGRP din alte protocoale se va utiliza una din variantele indicate mai jos:

```
router eigrp 1
redistribute static
redistribute ospf 1
redistribute rip
default-metric 10000 100 255 1 1500
```

Pentru redistribuirea în RIP din alte protocoale se va utiliza una din variantele indicate mai jos:

```
router rip
redistribute static
redistribute eigrp 1
redistribute ospf 1
default-metric 1
```

Redistribuirea între EIGRP şi OSPF
În cele 5 subreţele din AS1 este configurat protocolul de rutare dinamică EIGRP, iar în AS2 – protocolul OSPF.
Vom redistribui rutele între AS1 şi AS2 prin intermediul routerului Router0. În acest sens, mai întâi vom configura pe Router0 atât protocolul EIGRP, cât şi OSPF, declarând pentru EIGRP reţeaua 10.10.1.0/24, iar pentru OSPF – reţeaua 10.10.2.0/24:

```
Router(config)#router eigrp 1
Router(config-router)#network 10.10.1.0 0.0.0.255
Router(config-router)#no auto-summary
Router(config-router)#exit
Router(config)#do wr
Router(config)#router ospf 1
Router(config-router)#network 10.10.2.0 0.0.0.255 area 0
Router(config-router)#exit
Router(config)#do wr
```

Pe Router0 se redistribuie rutele EIGRP în OSPF şi reciproc:

```
Router(config)#router eigrp 1
Router(config-router)#redistribute ospf 1 metric 10000 100 255 1 1500
Router(config-router)#exit
```

```
Router(config)#router ospf 1
Router(config-router)#redistribute eigrp 1 metric 100 subnets
Router(config-router)#exit
Router(config)#do wr
```

În rezultatul execuţiei comenzilor menţionate mai sus, tabelele de rutare pentru Router1 şi Router2 vor include toate rutele din configuraţia de reţea:

```
     10.0.0.0/24 is subnetted, 2 subnets
C       10.10.1.0 is directly connected, GigabitEthernet7/0
D EX    10.10.2.0 [170/284160] via 10.10.1.3, 05:28:14,
GigabitEthernet7/0
C    192.168.1.0/24 is directly connected, GigabitEthernet9/0
C    192.168.2.0/24 is directly connected, GigabitEthernet8/0
D    192.168.3.0/24 [90/7680] via 10.10.1.2, 06:05:02,
GigabitEthernet7/0
D    192.168.4.0/24 [90/7680] via 10.10.1.2, 06:05:02,
GigabitEthernet7/0
D EX 192.168.5.0/24 [170/284160] via 10.10.1.3, 05:28:14,
GigabitEthernet7/0
D EX 192.168.6.0/24 [170/284160] via 10.10.1.3, 05:28:14,
GigabitEthernet7/0
D EX 192.168.7.0/24 [170/284160] via 10.10.1.3, 05:28:14,
GigabitEthernet7/0
D EX 192.168.8.0/24 [170/284160] via 10.10.1.3, 05:28:14,
GigabitEthernet7/0
```

La fel, şi tabelele de rutare pentru Router3 şi Router 4:

```
     10.0.0.0/24 is subnetted, 2 subnets
O E2    10.10.1.0 [110/100] via 10.10.2.1, 05:27:19,
GigabitEthernet9/0
C       10.10.2.0 is directly connected, GigabitEthernet9/0
O E2 192.168.1.0/24 [110/100] via 10.10.2.1, 05:27:19,
GigabitEthernet9/0
O E2 192.168.2.0/24 [110/100] via 10.10.2.1, 05:27:19,
GigabitEthernet9/0
O E2 192.168.3.0/24 [110/100] via 10.10.2.1, 05:27:19,
GigabitEthernet9/0
O E2 192.168.4.0/24 [110/100] via 10.10.2.1, 05:27:19,
GigabitEthernet9/0
O    192.168.5.0/24 [110/2] via 10.10.2.2, 05:55:46,
GigabitEthernet9/0
O    192.168.6.0/24 [110/2] via 10.10.2.2, 05:55:46,
GigabitEthernet9/0
C    192.168.7.0/24 is directly connected, GigabitEthernet7/0
C    192.168.8.0/24 is directly connected, GigabitEthernet8/0
```

Dacă dăm un ping între oricare două host-uri din reţea, vom vedea că există conexiune între acestea.

Pentru a redistribui rute statice, RIP sau EIGRP pe un router pe care este configurat protocolul OSPF se vor utiliza comenzile

```
router ospf 1
redistribute static metric 200 subnets
redistribute rip metric 200 subnets
redistribute eigrp 1 metric 100 subnets
```

Metrica OSPF este costul definit ca 10^8 / lățimea de bandă a link-ului în biți/sec. De exemplu, costul OSPF al Ethernet este 10: 10^8/10^7 = 10

Dacă nu este specificată o metrică, OSPF atribuie o valoare implicită de 20 la redistribuirea rutelor din toate protocoalele, cu excepția rutelor din protocolul BGP (Border Gateway Protocol), care obțin metrica 1.

Nu trebuie să definim o metrică sau să utilizăm metrica implicită atunci când redistribuim un proces OSPF într-un alt proces OSPF (de exemplu, *redistribute ospf 2 subnets*).

Redistribuirea rutelor statice printre rutele protocoalelor de rutare dinamică

În cele 5 subrețele din AS1 este realizată rutarea statică, iar în AS2 – protocolul OSPF.

Vom redistribui rutele între AS1 și AS2 prin intermediul routerului Router0.

Nu putem redistribui rutele unui protocol de rutare dinamică în cele statice. Pentru aceasta, pe fiecare router din AS1 (în care se configurează static rutele!) vom defini ca statice rutele din AS2, toate cu IP-ul interfeței de intrare 10.10.1.3 al Router0 (din partea AS1!).

Pe Router0 configurăm atât rutele statice, cât și cele OSPF:

```
Router(config)#ip route 192.168.1.0 255.255.255.0 10.10.1.1
Router(config)#ip route 192.168.2.0 255.255.255.0 10.10.1.1
Router(config)#ip route 192.168.3.0 255.255.255.0 10.10.1.2
Router(config)#ip route 192.168.4.0 255.255.255.0 10.10.1.2
Router(config)#router ospf 1
Router(config-router)#network 10.10.2.0 0.0.0.255 area 0
```

Pe Router0 se redistribuie rutele statice în OSPF:

```
Router(config)#router ospf 1
Router(config-router)#redistribute static metric 200 subnets
Router(config-router)#exit
```

În rezultatul execuției comenzilor menționate mai sus, tabelele de rutare pentru Router1 și Router2 vor include toate rutele din configurația de rețea:

```
     10.0.0.0/24 is subnetted, 2 subnets
C       10.10.1.0 is directly connected, GigabitEthernet7/0
S       10.10.2.0 [1/0] via 10.10.1.3
C    192.168.1.0/24 is directly connected, GigabitEthernet9/0
C    192.168.2.0/24 is directly connected, GigabitEthernet8/0
S    192.168.3.0/24 [1/0] via 10.10.1.2
S    192.168.4.0/24 [1/0] via 10.10.1.2
S    192.168.5.0/24 [1/0] via 10.10.1.3
S    192.168.6.0/24 [1/0] via 10.10.1.3
S    192.168.7.0/24 [1/0] via 10.10.1.3
S    192.168.8.0/24 [1/0] via 10.10.1.3
```

La fel, și tabelele de rutare pentru Router3 și Router 4:

```
     10.0.0.0/24 is subnetted, 1 subnets
C       10.10.2.0 is directly connected, GigabitEthernet9/0
O E2 192.168.1.0/24 [110/200] via 10.10.2.1, 00:13:54,
GigabitEthernet9/0
O E2 192.168.2.0/24 [110/200] via 10.10.2.1, 00:13:54,
GigabitEthernet9/0
O E2 192.168.3.0/24 [110/200] via 10.10.2.1, 00:13:54,
GigabitEthernet9/0
O E2 192.168.4.0/24 [110/200] via 10.10.2.1, 00:13:54,
GigabitEthernet9/0
C    192.168.5.0/24 is directly connected, GigabitEthernet7/0
C    192.168.6.0/24 is directly connected, GigabitEthernet8/0
O    192.168.7.0/24 [110/2] via 10.10.2.3, 00:48:17,
GigabitEthernet9/0
O    192.168.8.0/24 [110/2] via 10.10.2.3, 00:48:04,
GigabitEthernet9/0
```

Dacă dăm un ping între oricare două host-uri din rețea, vom vedea că există conexiune între acestea.

Redistribuirea rutelor între două sisteme autonome cu același protocol de rutare

Vom ilustra printr-un exemplu modul în care are loc redistribuirea rutelor dinamice între două procese OSPF (din două sisteme autonome).

Vom redistribui rutele între AS1 și AS2 prin intermediul routerului Router0. În acest sens, mai întâi vom configura pe Router0 două procese OSPF (1 și 2):

```
Router(config)#router ospf 1
```

```
Router(config-router)#network 10.10.1.0 0.0.0.255 area 0
Router(config-router)#exit
Router(config)#router ospf 2
Router(config-router)#network 10.10.2.0 0.0.0.255 area 0
Router(config-router)#exit
```

Pe Router0 se redistribuie rutele OSPF 1 în OSPF 2 şi reciproc:

```
Router(config)#router ospf 1
Router(config-router)#redistribute ospf 2 subnets
Router(config-router)#exit
```

```
Router(config)#router ospf 2
Router(config-router)#redistribute ospf 1 subnets
Router(config-router)#exit
```

În rezultatul execuţiei comenzilor menţionate mai sus, tabelele de rutare pentru Router1 şi Router2 vor include toate rutele din configuraţia de reţea:

```
     10.0.0.0/24 is subnetted, 2 subnets
C       10.10.1.0 is directly connected, GigabitEthernet7/0
O E2    10.10.2.0 [110/20] via 10.10.1.3, 00:04:27,
GigabitEthernet7/0
C    192.168.1.0/24 is directly connected, GigabitEthernet9/0
C    192.168.2.0/24 is directly connected, GigabitEthernet8/0
O    192.168.3.0/24 [110/2] via 10.10.1.2, 00:15:17,
GigabitEthernet7/0
O    192.168.4.0/24 [110/2] via 10.10.1.2, 00:15:17,
GigabitEthernet7/0
O E2 192.168.5.0/24 [110/20] via 10.10.1.3, 00:04:16,
GigabitEthernet7/0
O E2 192.168.6.0/24 [110/20] via 10.10.1.3, 00:04:16,
GigabitEthernet7/0
O E2 192.168.7.0/24 [110/20] via 10.10.1.3, 00:04:16,
GigabitEthernet7/0
O E2 192.168.8.0/24 [110/20] via 10.10.1.3, 00:04:16,
GigabitEthernet7/0
```

La fel, şi tabelele de rutare pentru Router3 şi Router 4:

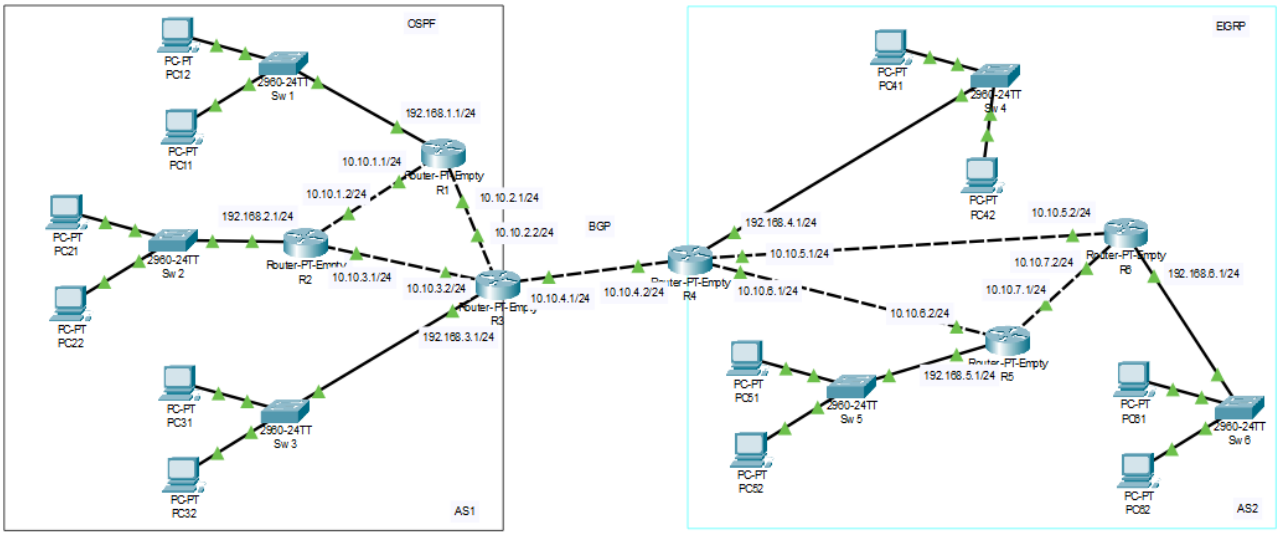
```
     10.0.0.0/24 is subnetted, 2 subnets
O E2    10.10.1.0 [110/20] via 10.10.2.1, 00:02:46,
GigabitEthernet9/0
C       10.10.2.0 is directly connected, GigabitEthernet9/0
O E2 192.168.1.0/24 [110/20] via 10.10.2.1, 00:02:46,
GigabitEthernet9/0
O E2 192.168.2.0/24 [110/20] via 10.10.2.1, 00:02:46,
GigabitEthernet9/0
O E2 192.168.3.0/24 [110/20] via 10.10.2.1, 00:02:46,
GigabitEthernet9/0
O E2 192.168.4.0/24 [110/20] via 10.10.2.1, 00:02:46,
GigabitEthernet9/0
C    192.168.5.0/24 is directly connected, GigabitEthernet7/0
C    192.168.6.0/24 is directly connected, GigabitEthernet8/0
O    192.168.7.0/24 [110/2] via 10.10.2.3, 00:10:02,
GigabitEthernet9/0
O    192.168.8.0/24 [110/2] via 10.10.2.3, 00:09:52,
GigabitEthernet9/0
```

Protocolul de rutare dinamică BGP

BGP (Border Gateway Protocol) este un protocol de rutare între sistemele autonome, care este conceput pentru a oferi rutare fără bucle între organizaţii (a se vedea Figura de mai jos). În continuare este explicată implementarea Cisco IOS a configurării protocolului BGP.

De regulă, BGP este utilizat pentru a conecta o reţea locală la o reţea externă cu scopul de a obţine acces la Internet sau pentru a realiza conectarea la alte organizaţii. Când este realizată conexiunea la o organizaţie externă, sunt create sesiuni de peering extern BGP (eBGP). Deşi BGP este denumit protocol de gateway exterior (EGP), multe reţele din cadrul organizaţiilor au devenit atât de complexe, încât BGP poate fi utilizat pentru a simplifica reţeaua internă utilizată în cadrul organizaţiei. În cadrul aceleiaşi organizaţii se schimbă informaţii de rutare prin sesiuni interne de peering BGP (iBGP).

Vom ilustra cum se configurează protocolul BGP pe exemplul următoarei reţele:



În AS1 pe routerele R1, R2 şi R3 este configurat protocolul OSPF. Pe routerul R3 la configurarea protocolului OSPF se indică doar reţelele ce se referă la AS1 (reţeaua 10.10.4.0 asigură legătura dintre AS1 şi AS2!):

```
router ospf 1
 network 10.10.3.0 0.0.0.255 area 0
 network 10.10.2.0 0.0.0.255 area 0
 network 192.168.3.0 0.0.0.255 area 0
exit
```

Pe routerul R3 este configurat protocolul BGP:

```
router bgp 1   (routerul R3 se află în AS1)
 neighbor 10.10.4.2 remote-as 2  (routerul de legătură R4 se află în AS2)
(În continuare se indică adresele şi măştile reţelelor din AS1:)
 network 192.168.1.0 mask 255.255.255.0
 network 192.168.2.0 mask 255.255.255.0
 network 192.168.3.0 mask 255.255.255.0
 network 10.10.1.0 mask 255.255.255.0
 network 10.10.2.0 mask 255.255.255.0
 network 10.10.3.0 mask 255.255.255.0
exit
```

În AS2 pe routerele R4, R5 şi R6 este configurat protocolul EIGRP. Pe routerul R4 la configurarea protocolului EIGRP se indică doar reţelele ce se referă la AS2 (reţeaua 10.10.4.0 asigură legătura dintre AS1 şi AS2!):

```
router eigrp 1
 network 192.168.4.0
 network 10.10.5.0 0.0.0.255
 network 10.10.6.0 0.0.0.255
 no auto-summary
exit
```

Pe routerul R4 este configurat protocolul BGP:

```
router bgp 2   (routerul R4 se află în AS2)
 neighbor 10.10.4.1 remote-as 1   (routerul de legătură R3 se află în AS1)
(În continuare se indică adresele şi măştile reţelelor din AS2:)
 network 192.168.4.0 mask 255.255.255.0
 network 192.168.5.0 mask 255.255.255.0
 network 192.168.6.0 mask 255.255.255.0
 network 10.10.5.0 mask 255.255.255.0
 network 10.10.6.0 mask 255.255.255.0
 network 10.10.7.0 mask 255.255.255.0
exit
```

La moment, în tabelul de rutare al routerului R3, datorită protocolului BGP, sunt prezente toate rutele din reţea:

```
R3(config)#do show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
```

```
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 7 subnets
O       10.10.1.0 [110/2] via 10.10.3.1, 01:50:23, GigabitEthernet7/0
                  [110/2] via 10.10.2.1, 01:50:23, GigabitEthernet8/0
C       10.10.2.0 is directly connected, GigabitEthernet8/0
C       10.10.3.0 is directly connected, GigabitEthernet7/0
C       10.10.4.0 is directly connected, GigabitEthernet6/0
B       10.10.5.0 [20/0] via 10.10.4.2, 00:00:00
B       10.10.6.0 [20/0] via 10.10.4.2, 00:00:00
B       10.10.7.0 [20/0] via 10.10.4.2, 00:00:00
O    192.168.1.0/24 [110/2] via 10.10.2.1, 01:50:23, GigabitEthernet8/0
O    192.168.2.0/24 [110/2] via 10.10.3.1, 01:50:23, GigabitEthernet7/0
C    192.168.3.0/24 is directly connected, GigabitEthernet9/0
B    192.168.4.0/24 [20/0] via 10.10.4.2, 00:00:00
B    192.168.5.0/24 [20/0] via 10.10.4.2, 00:00:00
B    192.168.6.0/24 [20/0] via 10.10.4.2, 00:00:00
```

La fel, în tabelul de rutare al routerului R4, datorită protocolului BGP, sunt prezente toate rutele din rețea:

```
R4(config)#do show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 7 subnets
B       10.10.1.0 [20/0] via 10.10.4.1, 00:00:00
B       10.10.2.0 [20/0] via 10.10.4.1, 00:00:00
B       10.10.3.0 [20/0] via 10.10.4.1, 00:00:00
C       10.10.4.0 is directly connected, GigabitEthernet6/0
C       10.10.5.0 is directly connected, GigabitEthernet7/0
C       10.10.6.0 is directly connected, GigabitEthernet8/0
D       10.10.7.0 [90/3072] via 10.10.5.2, 01:55:17, GigabitEthernet7/0
                  [90/3072] via 10.10.6.2, 01:55:17, GigabitEthernet8/0
B    192.168.1.0/24 [20/0] via 10.10.4.1, 00:00:00
B    192.168.2.0/24 [20/0] via 10.10.4.1, 00:00:00
B    192.168.3.0/24 [20/0] via 10.10.4.1, 00:00:00
C    192.168.4.0/24 is directly connected, GigabitEthernet9/0
D    192.168.5.0/24 [90/5376] via 10.10.6.2, 01:55:17, GigabitEthernet8/0
D    192.168.6.0/24 [90/5376] via 10.10.5.2, 01:55:17, GigabitEthernet7/0
```

Pe routerul R3 sunt redistribuite rutele BGP1 în OSPF:

```
router ospf 1
 redistribute bgp 1 subnets
exit
```

Pe routerul R4 sunt redistribuite rutele BGP2 în EIGRP:

```
router eigrp 1
 redistribute bgp 2 metric 10000 100 255 1 1500
exit
```

În rezultat, toate rutele din rețea sunt disponibile pe fiecare router. De exemplu, pe routerul R2 avem:

```
R2(config)#do show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
```

```
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 6 subnets
C       10.10.1.0 is directly connected, GigabitEthernet8/0
O       10.10.2.0 [110/2] via 10.10.1.1, 02:29:49, GigabitEthernet8/0
                  [110/2] via 10.10.3.2, 02:29:49, GigabitEthernet7/0
C       10.10.3.0 is directly connected, GigabitEthernet7/0
O E2    10.10.5.0 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
O E2    10.10.6.0 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
O E2    10.10.7.0 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
O    192.168.1.0/24 [110/2] via 10.10.1.1, 02:29:49, GigabitEthernet8/0
C    192.168.2.0/24 is directly connected, GigabitEthernet9/0
O    192.168.3.0/24 [110/2] via 10.10.3.2, 02:29:49, GigabitEthernet7/0
O E2 192.168.4.0/24 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
O E2 192.168.5.0/24 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
O E2 192.168.6.0/24 [110/20] via 10.10.3.2, 01:27:33, GigabitEthernet7/0
```

Se poate verifica cu ping că există conexiune între oricare două host-uri din rețea. La fel, se poate vedea traseul de la sursă la destinație, folosind traceroute, de exemplu, de pe PC21 obținem:

```
C:\>tracert 192.168.6.2

Tracing route to 192.168.6.2 over a maximum of 30 hops:

  1    0 ms      0 ms      3 ms      192.168.2.1
  2    0 ms      0 ms      0 ms      10.10.3.2
  3    0 ms      0 ms      0 ms      10.10.4.2
  4    0 ms      0 ms      0 ms      10.10.5.2
  5    0 ms      0 ms      0 ms      192.168.6.2

Trace complete.
```

Lucrarea de laborator №6

Configurarea serverelor NAT, DHCP, DNS şi Email

Scopul lucrării constă în formarea abilităţilor practice de configurare în Cisco Packet Tracer a serverelor NAT, DHCP, DNS şi Email

Obiective:

- A explica conceptul de NAT şi a ilustra în Cisco Packet Tracer modul de funcţionare al acestuia
- A arăta cum se configurează NAT pe routerele de frontieră din reţelele locale
- A arăta cum se configurează pe server şi pe router protocolul DHCP pentru distribuirea automatizată a datelor IP host-urilor din reţeaua locală
- A arăta cum se configurează serverul DNS pentru a transla numele de domeniu în adresa IP corespunzătoare
- A arăta cum se configurează serverul de email prin intermediul căruia utilizatorii reţelei îşi pot transmite mesaje de e-mail

Indicaţii metodice privind realizarea lucrării

Configurarea routerului NAT

NAT (Network Address Translation) – translarea adreselor de reţea

NAT -> are ca obiectiv să ascundă adresele din interiorul reţelei locale de dispozitivele din exterior. Tehnologia este bazată pe înlocuirea adresei private a host-ului sursă, la ieşirea din reţeaua locală, cu o adresă publică. Soft-ul NAT este configurat pe routerul de frontieră, care separă reţeaua locală de reţeaua din exteriorul acesteia.

Vom explica modul de funcţionare al tehnologiei NAT pe exemplul următoarei configuraţii de reţea (a se vedea Figura 1):

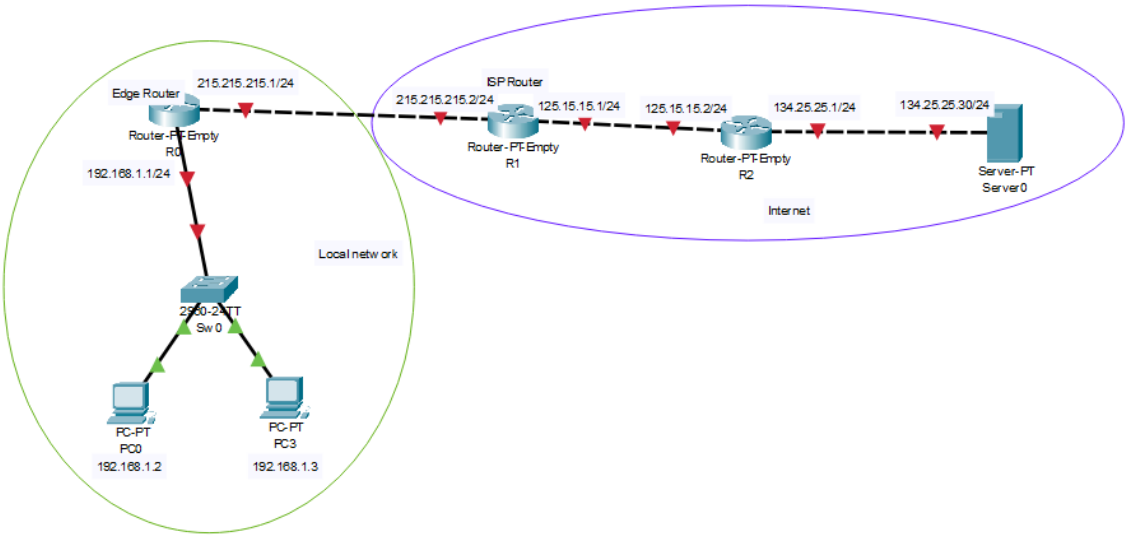


Figura 1

1) Setăm adrese IP pe interfeţele routerelor şi pe host-uri:

`Router>en`
`Router#conf ter`
`Router(config)#host R0`
`R0(config)#int fa 9/0`
`R0(config-if)#ip add 192.168.1.1 255.255.255.0`
`R0(config-if)#no sh`
`R0(config)#int fa 8/0`
`R0(config-if)#ip add 215.215.215.1 255.255.255.0`
`R0(config-if)#no sh`
`R0(config-if)#exit`
	`Router>en`
	`Router#conf ter`
	`Router(config)#host R1`
	`R1(config)#int fa 8/0`
	`R1(config-if)#ip add 215.215.215.2 255.255.255.0`
	`R1(config-if)#no sh`
	`R1(config-if)#exit`
	`R1(config)#int fa 9/0`
	`R1(config-if)#ip add 125.15.15.1 255.255.255.0`
	`R1(config-if)#no sh`
	`R1(config-if)#exit`
	`R1(config)#do wr`

`Router>en`
`Router#conf ter`
`Router(config)#host R2`
	`Server0:`
	`IP Address: 134.25.25.30`
	`Subnet Mask: 255.255.255.0`

```
R2(config)#int fa 8/0
R2(config-if)#ip add 125.15.15.2 255.255.255.0
R2(config-if)#no sh
R2(config-if)#exit
R2(config)#int fa 9/0
R2(config-if)#ip add 134.25.25.1 255.255.255.0
R2(config-if)#no sh
R2(config-if)#exit
R2(config)#do wr
```

```
Default Gateway: 134.25.25.1

PC0:
IP Address: 192.168.1.2
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.1.1

PC3:
IP Address: 192.168.1.3
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.1.1
```

2) Configurăm rutarea:

```
R0(config)#ip route 0.0.0.0 0.0.0.0 215.215.215.2   // rută statică implicită
R0(config)#do wr

R1(config)#ip route 134.25.25.0 255.255.255.0 125.15.15.2  // rută statică
R1(config)#do wr

R2(config)#ip route 215.215.215.0 255.255.255.0 125.15.15.1 // rută statică
R2(config)#do wr
```

3) Despre adresele IP private şi publice:

Adresele IP private – nu sunt rutate în Internet

Trei intervale de adrese IP private:

10.0.0.0/8 (adică adresele cu primul octet 10)

172.16.0.0/12 (adică adresele de la 172.16.0.0 până la 172.31.255.255 inclusiv)

192.168.0.0/16 (adică adresele de la 192.168.0.0 până la 192.168.255.255 inclusiv)

Exemple de adrese IP publice – care pot fi rutate în Internet

178.217.16.246	217.69.139.200	78.110.50.125	81.192.76.172

În reţelele locale (organizaţii) host-urilor li se atribuie adrese IP private. Dar atunci când este necesar ca un host cu o adresă privată să acceseze Internet – adresa sa privată este înlocuită cu o adresă publică prin mecanismul numit NAT.

În diferite organizaţii, de regulă sunt reţele cu aceleaşi adrese private, de exemplu, 192.168.x.x. Iar dacă este necesar de ieşit în exterior – adresa privată este translată într-o adresă publică, care va putea fi rutată.

În loc să se repartizeze adrese publice către host-uri – PC0, PC1, ..., se repartizează o singură adresă publică, care este setată pe interfaţa de ieşire a routerului de frontieră (de exemplu, 215.215.215.1). Serverul cu adresa 134.25.25.30 îi va răspunde pe această adresă publică routerului de frontieră, iar NAT-ul ce funcţionează pe router va decide cărui host anume din interiorul reţelei îi este destinat pachetul recepţionat.

Nu am setat pe routerele R1 şi R2 cum se poate ajunge până la reţeaua 192.168.1.0, deoarece aceasta este o reţea locală, iar în tabelul de rutare al routerelor R1 şi R2 nu sunt adrese IP private (amintim că reţeaua 192.168.1.0 există în reţelele locale ale unui număr enorm de organizaţii => nu are sens să se stocheze adrese private în tabelul de rutare)

În modul Simulation de pe host-ul PC0 dăm un ping (vom lăsa să fie văzute doar pachetele ICMP): ping 134.25.25.30 => pachetul ajunge până la Server0, dar nu poate ajunge înapoi, deoarece în tabelul de rutare al lui R2 nu există IP adresa 192.168.1.2

În acest moment se dovedeşte a fi util conceptul de NAT:

NAT-ul static (Static NAT) – translarea unei adrese IP private într-o adresă adresă IP publică. Fiecărui host cu adresa privată îi corespunde o adresă publică pentru ieşire în exterior

NAT-ul dinamic (Dynamic NAT) – translarea mai multor adrese IP private în mai multe adrese IP publice. Fiecărui dispozitiv din reţeaua locală i se atribuie o adresă publică (din setul disponibil) la ieşire în exterior

NAT cu supraîncărcare (numit şi PAT, NAPT, NAT Overload) – translarea mai multor adrese IP private în aceeaşi adresă IP publică, dar cu implicarea numerelor de port (sau a numerelor de secvenţă)

Static NAT:

R0(config)#	int fa 9/0	Este accesată interfaţa fast ethernet 9/0
R0(config-if)#	ip nat inside	Se stabileşte interfaţa fa9/0 ca de interior (reţeaua locală) din punct

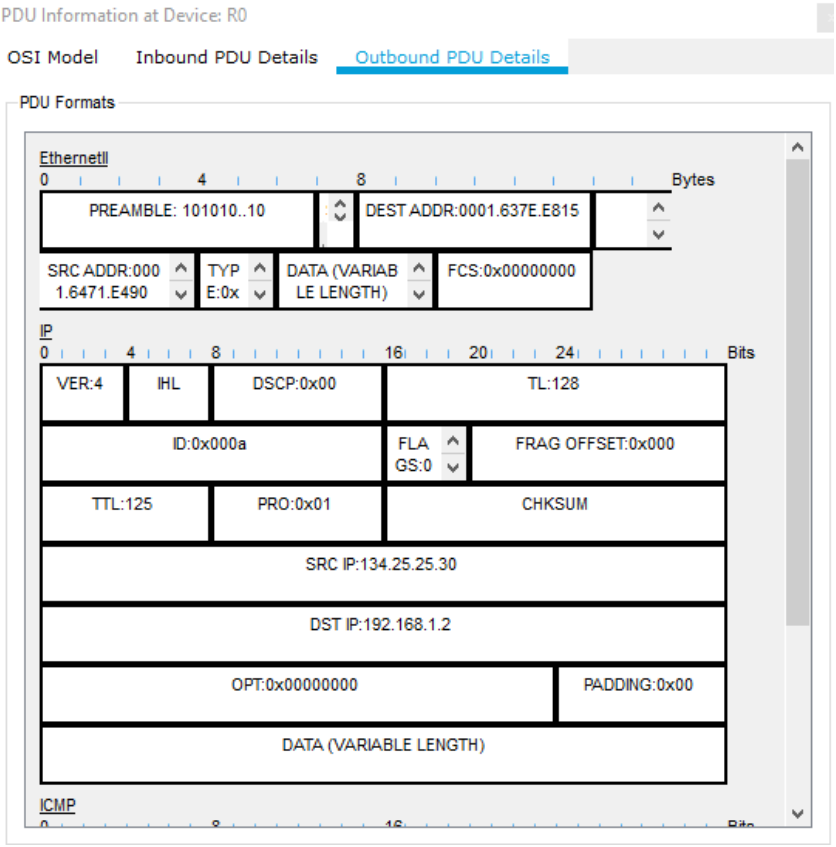
		de vedere NAT
R0(config)#	int fa 8/0	Este accesată interfața fast ethernet 8/0
R0(config-if)#	ip nat outside	Se stabilește interfața fa8/0 ca de exterior (Internet)
R0(config)#	ip nat inside source static 192.168.1.2 215.215.215.20 (se consideră că provider-ul a repartizat acest IP, dar se poate de folosit adresa IP 215.215.215.1 a interfeței)	Se activează NAT-ul static. IP adresa 192.168.1.2 se va translata în 215.215.215.20

De pe host-ul PC0: ping 134.25.25.30 – este conexiune
De ce este conexiune? Trecem în modul Simulation şi lăsăm să treacă doar pachetele ICMP. Dăm în execuţie:

ping 134.25.25.30

Examinăm conţinutul pachetului care a ajuns la switch-ul Sw0. Vedem acolo că:
SRC IP (sursa): 192.168.1.2
DST IP (destinaţia): 134.25.25.30

La ieşirea din routerul R0, pe care este activat NAT avem SRC IP: 215.215.215.20 (adresa 192.168.1.2 este înlocuită cu una publică)
Când pachetul trece în sens invers:
Mai întâi DST IP: 215.215.215.20
Însă pe routerul R0 (NAT): DST IP: 192.168.1.2



Pe router se poate vedea procesul de translatare prin comanda: *show ip nat translations*

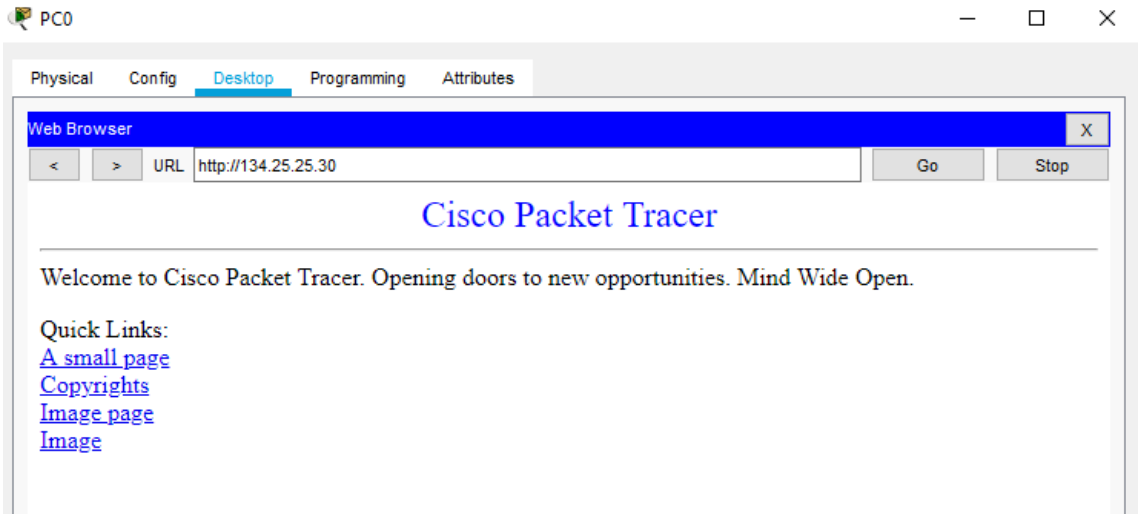
```
R0(config)#do show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
icmp 215.215.215.20:13 192.168.1.2:13     134.25.25.30:13    134.25.25.30:13
icmp 215.215.215.20:14 192.168.1.2:14     134.25.25.30:14    134.25.25.30:14
icmp 215.215.215.20:15 192.168.1.2:15     134.25.25.30:15    134.25.25.30:15
icmp 215.215.215.20:16 192.168.1.2:16     134.25.25.30:16    134.25.25.30:16
---  215.215.215.20    192.168.1.2        ---                ---
```

Inside local => IP adresa din reţeaua locală, adresă ce este înlocuită
Inside global => IP adresă prin care se înlocuieşte
Outside local & Outside global => coincid => IP adresa destinaţiei

Înregistrarea cu procesul de translare este stocată o perioadă limitată de timp. Dacă peste ceva timp vom da din nou comanda *show ip nat translations*, atunci nu vom mai vedea întregul tabel ca mai sus (ci doar ceea ce am definit prin intermediul comenzilor)

```
R0#show ip nat tran
Pro  Inside global     Inside local       Outside local      Outside global
---  215.215.215.20    192.168.1.2        ---                ---
```

Dacă din nou dăm un ping, vor apărea date noi despre procesul de translatare
Atunci când dăm în execuţie comanda *ping*, tabelul de translatare arată pachetele ICMP corespunzătoare
Atunci când pe host-ul PC0 accesăm Web Server (pe PC0 în Desktop alegem Web Browser şi scriem adresa 134.25.25.30) => arată pachete *tcp*:




```
R0#show ip nat tran
Pro  Inside global     Inside local       Outside local      Outside global
---  215.215.215.20    192.168.1.2        ---                ---
tcp 215.215.215.20:1025192.168.1.2:1025   134.25.25.30:80    134.25.25.30:80
```

Pentru a elibera (a şterge toate datele) tabelul de translatare dăm comanda: *clear ip nat translation **
Dacă dăm de pe host-ul PC3 comanda ping 134.25.25.30 => vom vedea că nu trece
Asta deoarece anterior am definit doar regula de translatare: 192.168.1.2 este înlocuit prin 215.215.215.20
În cazul dat avem 192.168.1.3! => este necesar de definit o nouă regulă de translatare, dar pentru aceasta se va utiliza o altă adresă publică (dacă se aplică NAT-ul static!). De exemplu, dacă provider-ul a oferit o a doua adresă publică 215.215.215.21 pentru ieşire în Internet
Pe routerul R0 dăm comanda: *ip nat inside source static 192.168.1.3 215.215.215.21*
De pe PC3 dăm comanda: *ping 134.25.25.30* => deja trece!
Adresa IP privată se va înlocui prin adresa publică 215.215.215.21
În modul Simulation cum se deplasează pachetele (de la PC0 şi PC3)
În sens invers, dacă serverul va transmite un mesaj de răspuns către PC0, atunci acesta (serverul) va utiliza adresa 215.215.215.20
Dacă serverul va transmite un răspuns către PC3, atunci va utiliza adresa publică 215.215.215.21
Fiecare adresă privată este translatată într-o adresă IP publică
Mecanismul NAT în forma examinată nu este aplicat, deoarece acesta nu face economie de adrese publice utilizate. Se ştie că numărul de adrese IPv4 publice este aproape epuizat.
Dynamic NAT:
Translarea mai multor adrese IP private în mai multe adrese IP publice
Mai întâi, pe routerul R0 se va dezactiva Static NAT pe care l-am activat anterior (vom folosi aceeaşi configuraţie):

```
R0(config)# no ip nat inside source static 192.168.1.2 215.215.215.20
R0(config)# no ip nat inside source static 192.168.1.3 215.215.215.21
```

Vom considera că provider-ul a distribuit reţelei un set (un stoc) de adrese IP publice 215.215.215.20 – 215.215.215.30. În continuare, vom configura NAT-ul dinamic:

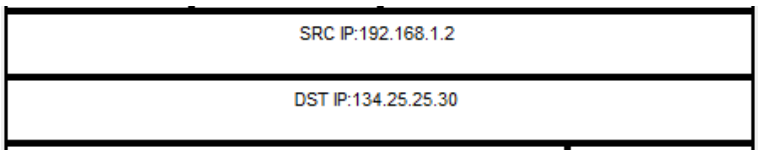
`R0(config)#`	`int fa9/0`	`Accesăm interfaţa fast ethernet 9/0`

R0(config-if)#	ip nat inside	Stabilim interfaţa fa9/0 ca de interior (reţeaua locală)
R0(config)#	int fa8/0	Accesăm interfaţa fast ethernet 8/0
R0(config-if)#	ip nat outside	Stabilim interfaţa fa8/0 ca de exterior (Internet)
R0(config)#	access-list 1 permit 192.168.1.0 0.0.0.255	Generăm o listă de acces standard ACL, care va arăta intervalul de adrese private cărora li se va permite să fie translatate în adrese publice pentru ieşire în exterior. (Atenţie! Adresele private care nu au nimerit în listă – nu vor fi translatate!)
R0(config)#	ip nat pool TRANS 215.215.215.20 215.215.215.30 netmask 255.255.255.0	Creăm set-ul de adrese IP cu numele TRANS. Aceste adrese IP vor înlocui pe cele private din lista de acces 1. În acest caz, adresele IP private vor fi translatate în adrese publice, începând cu 215.215.215.20 până la 215.215.215.30
R0(config)#	ip nat inside source list 1 pool TRANS	Activăm NAT-ul dinamic, unde list 1 – intervalul de adrese private, pool TRANS – intervalul de adrese IP publice

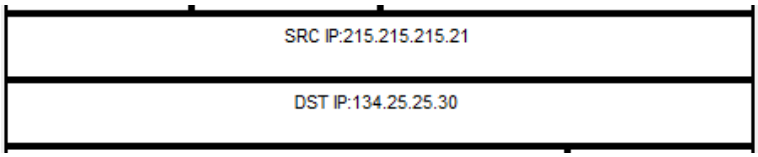
Dacă este necesar să se definească mai multe reţele cărora li se va permite ieşirea în exteriorul reţelei locale, atunci la definirea listei de acces vom aplica o construcţie de genul:

```
Router(config)#ip access-list standard L1
Router(config-std-nacl)#permit 192.168.2.0 0.0.0.255
Router(config-std-nacl)#permit 192.168.3.0 0.0.0.255
Router(config-std-nacl)#permit 192.168.4.0 0.0.0.255
Router(config-std-nacl)#exit
```

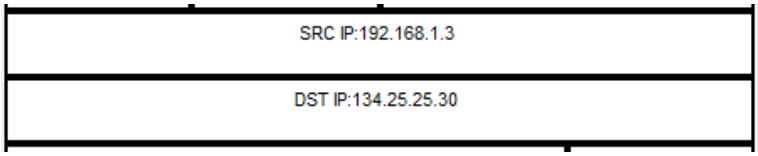
iar comanda finală va fi: *ip nat inside source list L1 pool TRANS*
De pe PC0: ping 134.25.25.30 – este conexiune
De pe PC3: ping 134.25.25.30 – este conexiune
Trecem în modul Simulation:
De pe PC0: ping 134.25.25.30 => pe routerul R0 vedem:
La intrare:



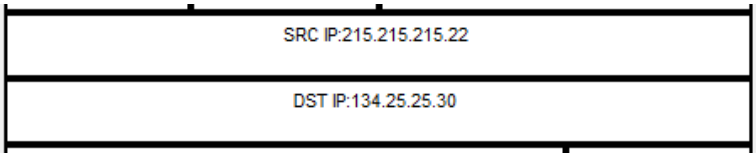
La ieşire:



De pe PC3: ping 134.25.25.30 => pe routerul R0 vedem:
La intrare:



La ieşire:



Dacă PC0 nu se va adresa ceva timp în Internet, atunci IP-ul 215.215.215.21 se va elibera.
Comanda
show ip nat translations – permite să vedem tabelul de translatare

```
R0(config)#do show ip nat tr
Pro  Inside global     Inside local       Outside local      Outside global
icmp 215.215.215.21:10 192.168.1.2:10     134.25.25.30:10    134.25.25.30:10
icmp 215.215.215.21:11 192.168.1.2:11     134.25.25.30:11    134.25.25.30:11
icmp 215.215.215.21:12 192.168.1.2:12     134.25.25.30:12    134.25.25.30:12
icmp 215.215.215.21:9  192.168.1.2:9      134.25.25.30:9     134.25.25.30:9
icmp 215.215.215.22:5  192.168.1.3:5      134.25.25.30:5     134.25.25.30:5
```

show ip nat statistics – permite să vedem o statistică a procesului de translatare

```
R0(config)#do show ip nat stat
Total translations: 5 (0 static, 5 dynamic, 5 extended)
Outside Interfaces: FastEthernet8/0
Inside Interfaces: FastEthernet9/0
Hits: 11  Misses: 17
Expired translations: 12
Dynamic mappings:
-- Inside Source
access-list 1 pool TRANS refCount 5
 pool TRANS: netmask 255.255.255.0
       start 215.215.215.20 end 215.215.215.30
       type generic, total addresses 11 , allocated 2 (18%), misses 0
```

Static NAT – am indicat manual pentru fiecare adresă IP privată o adresă IP publică concretă
Dynamic NAT – la fel, dar în mod dinamic – se indică un set de adrese IP private şi un set de adrese Ip publice, iar routerul NAT în mod automatizat decide care va fi corespondenţa între adrese
Host-ul ce nu mai are nevoie de Internet – eliberează acea IP adresă, care este atribuită ulterior unui alt host
Oricum, în acest mod nu se contribuie nicidecum la minimizarea numărului de IP adrese publice utilizate pentru ieşire în Internet
În forma descrisă Static NAT şi Dynamic NAT nu sunt aplicate

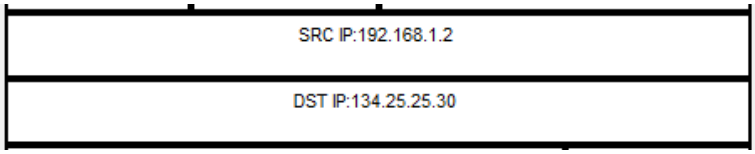
NAT-ul cu supraîncărcare (PAT, NAPT, NAT Overload)
NAT-ul overload – translatarea mai multor adrese IP private într-o adresă IP publică => procedeul este aplicat actualmente pe routere în reţele
Pentru a ieşi în Internet, toate host-urile din cadrul organizaţiei folosesc aceeaşi adresă IP publică, dar pentru a determina sursa căreia i se răspunde din exterior, se aplică mecanismul numerelor de port (sau a numerelor de secvenţă – sequence number)
Fie că provider-ul a distribuit pentru translatarea adreselor private din reţeaua locală a organizaţiei adresa publică 215.215.215.20
Trecem la configurarea PAT (Port Address Translation):
Mai întâi anulăm configurările precedente ale NAT-ului dinamic:

```
R0(config)# no ip nat inside source list 1 pool TRANS
R0(config)# no ip nat pool TRANS 215.215.215.20 215.215.215.30 netmask 255.255.255.0
R0(config)# no access-list 1 permit 192.168.1.0 0.0.0.255
```

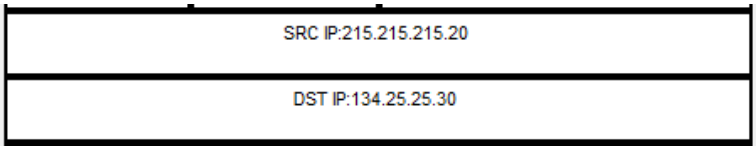
R0(config)#	int fa9/0	Accesăm interfaţa fast ethernet 9/0
R0(config-if)#	ip nat inside	Stabilim interfaţa fa9/0 ca de interior (reţeaua locală)
R0(config)#	int fa8/0	Accesăm interfaţa fast ethernet 8/0
R0(config-if)#	ip nat outside	Stabilim interfaţa fa8/0 ca de exterior (Internet)
R0(config)#	access-list 1 permit 192.168.1.0 0.0.0.255	Se indică set-ul de adrese private, care vor fi translatate (Atenţie! Adresele private ce nu vor fi incluse în listă – nu vor fi translatate!)
R0(config)#	ip nat pool TRANS 215.215.215.20 215.215.215.20 netmask 255.255.255.0	Se indică adresa IP publică în care vor fi translatate adresele private din reţeaua locală

R0(config)#	ip nat inside source list 1 pool TRANS overload	Se activează PAT, unde list 1 – intervalul de adrese IP private, pool TRANS – intervalul de adrese IP publice

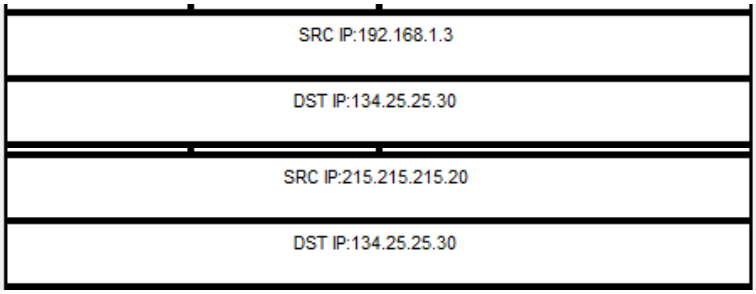
De pe PC0: ping 134.25.25.30 => este conexiune
De pe PC3: ping 134.25.25.30 => este conexiune
În modul Simulation lăsăm să treacă doar pachetele ICMP (apăsăm *Show All/None*, după care pe *Edit Filters* şi punem bifa la *ICMP*). Dăm un ping de pe PC0 şi de pe PC3 (simultan) către serverul cu adresa IP 134.25.25.30:
Atunci când pachetele pe rând vor ajunge până la routerul R0, vom vedea că R0 a translatat adresele IP ale lui PC0 şi PC3 în aceeaşi adresă IP publică 215.215.215.20:
La intrarea în R0 avem



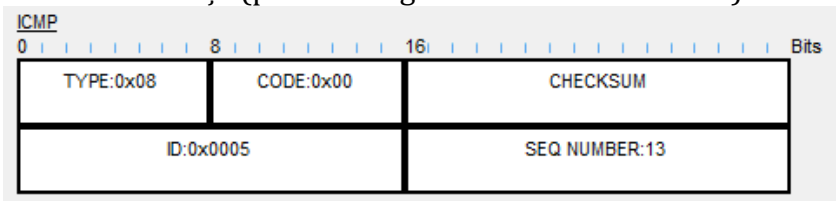
iar la ieşire



La fel



Întrebare: atunci când serverul transmite mesajul de răspuns către sursă, cum determină routerul NAT cu ce adresă privată trebuie să înlocuiască adresa publică indicată?
Server0 transmite toate pachetele la o singură adresă - 215.215.215.20. Nu e clar care pachet este destinat către 192.168.1.2 şi care către 192.168.1.3? În acest caz, routerul NAT stabileşte destinaţia pachetului după numărul de secvenţă (pe care l-a generat host-ul din start):

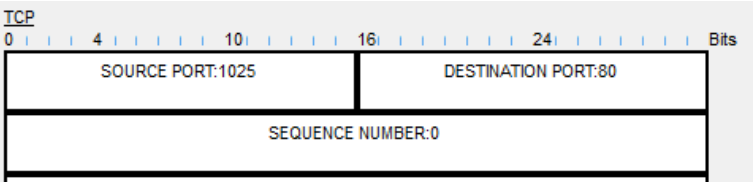


Host-ul PC0 a generat numărul 13. Dacă pachetele ce au fost transmise anterior nu au utilizat acest număr, atunci pachetul este transmis mai departe fără modificări şi ajunge la server. Serverul transmite un mesaj de răspuns, incluzând acest număr în acel mesaj. Pachetul de răspuns de la server ajunge la routerul NAT R0, care în baza acestui număr determină în tabelul său de translatare linia corespunzătoare (a se vedea Figura 2), unde găseşte adresa IP a host-ului din reţeaua locală, căruia îi şi transmite pachetul.

```
R0(config)#do show ip nat tr
Pro  Inside global     Inside local       Outside local      Outside global
icmp 215.215.215.20:10 192.168.1.2:10     134.25.25.30:10    134.25.25.30:10
icmp 215.215.215.20:11 192.168.1.2:11     134.25.25.30:11    134.25.25.30:11
icmp 215.215.215.20:12 192.168.1.2:12     134.25.25.30:12    134.25.25.30:12
icmp 215.215.215.20:13 192.168.1.2:13     134.25.25.30:13    134.25.25.30:13
icmp 215.215.215.20:5  192.168.1.2:5      134.25.25.30:5     134.25.25.30:5
icmp 215.215.215.20:6  192.168.1.2:6      134.25.25.30:6     134.25.25.30:6
icmp 215.215.215.20:7  192.168.1.2:7      134.25.25.30:7     134.25.25.30:7
icmp 215.215.215.20:8  192.168.1.2:8      134.25.25.30:8     134.25.25.30:8
icmp 215.215.215.20:9  192.168.1.2:9      134.25.25.30:9     134.25.25.30:9
```

Figura 2

Dacă ne adresăm la server de pe host-ul PC0 ca la web browser:
Apăsăm pe PC0, apoi pe Web Browser şi culegem adresa 134.25.25.30, apoi trecem în modul Simulation, lăsând să treacă doar pachetele TCP. În Web Browser apăsăm butonul *Go* şi analizăm cum trece pachetul. Se poate vedea că translatarea adresei are loc în baza numărului de port (Source Port), pe care host-ul îl generează în mod aleator.



Atunci când pachetele de la PC0 şi PC3 trec prin R0 către server, atunci routerul R0 înregistrează care adresă IP privată şi de sub care port se adresează la serverul web.
Serverul răspunde cu un pachet ce conţine acel număr de port ca destinaţie (a se vedea Figura 3)

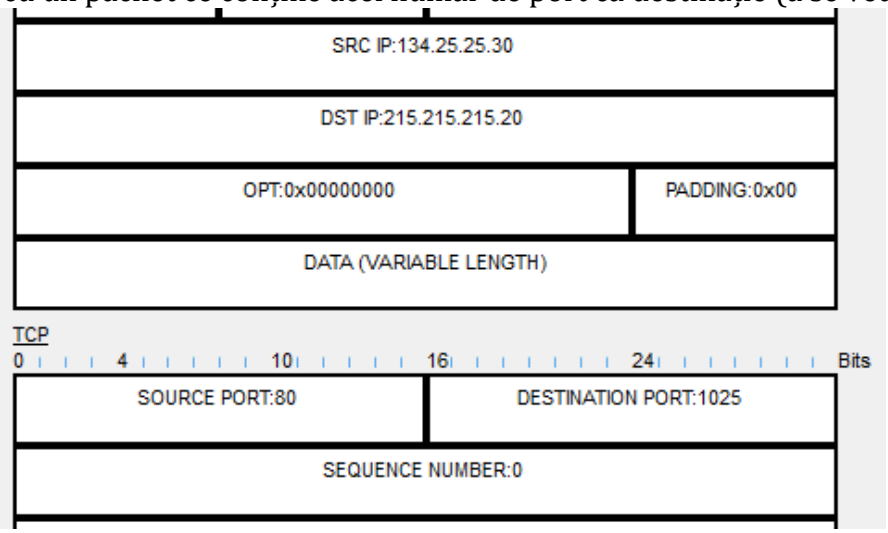


Figura 3

Routerul R0 în baza numărului de port destinaţie recepţionat găseşte în tabelul său de translatare IP-ul host-ului corespunzător (PC0 sau PC3).
Ştergem conţinutul tabelului de translatare dinamică prin comanda: *clear ip nat tr ** (configurările nu sunt şterse)
Ce se întâmplă dacă PC0 şi PC3 se vor adresa la server cu acelaşi număr de port? De exemplu, PC0 şi PC3 au generat acelaşi număr de port 1025. Putem modela o astfel de situaţie cu ajutorul instrumentului Traffic Generator (Generatorul de trafic).
Trecem în modul Simulation. Lăsăm să treacă doar pachetele TCP şi UDP
Pe PC0 apăsăm pe butonul Traffic Generator şi completăm cu următoarele date:
Select Applications: alegem OTHER
Destination IP Address: 134.25.25.30
Source IP Address: 192.168.1.2
Starting Source Port: 1025
Destination Port: 80
Jos apăsăm pe butonul Send
Pe PC3 apăsăm pe Traffic Generator şi completăm cu următoarele date:
Select Applications: alegem OTHER
Destination IP Address: 134.25.25.30
Source IP Address: 192.168.1.3
Starting Source Port: 1025
Destination Port: 80
Jos apăsăm pe butonul Send
Urmărim cum trec pachetele. Când pe routerul R0 a ajuns primul pachet (de la PC0), în linia de comandă a lui R0 scriem *do show ip nat tr*:

```
R0(config)#do show ip nat tr
Pro  Inside global     Inside local       Outside local      Outside global
udp 215.215.215.20:1025192.168.1.2:1025   134.25.25.30:80    134.25.25.30:80
```

Atunci când la routerul R0 ajunge al doilea pachet (de la PC3), în linia de comandă a lui R0 scriem *do show ip nat tr*:

```
R0(config)#do show ip nat tr
Pro  Inside global     Inside local       Outside local      Outside global
udp 215.215.215.20:1024192.168.1.3:1025   134.25.25.30:80    134.25.25.30:80
udp 215.215.215.20:1025192.168.1.2:1025   134.25.25.30:80    134.25.25.30:80
```

Vedem că routerul a înlocuit nu doar adresa IP a sursei, dar şi numărul de port corespunzător (deoarece acesta coincide cu precedentul). În Internet pachetele pleacă cu numere de port diferite. Atunci când pachetul se întoarce de la server înapoi la routerul R0, atunci R0 va şti că dacă numărul de port este 1024, atunci adresa IP se va înlocui cu 192.168.1.3 şi portul 1025.

Dacă două host-uri se adresează la aceeaşi adresă de IP, în acelaşi timp, cu numere de port identice, atunci se înlocuieşte nu doar IP-ul iniţial, dar şi portul iniţial.

Configurarea mecanismului NAT pe routerul Cisco care gestionează câteva VLAN-uri

Prezintă interes situaţia când în reţea avem un router pe care avem configurate mai multe subinterfeţe, câte una pentru fiecare VLAN din reţea, şi tot pe acest router urmează să configurăm mecanismul NAT. Să considerăm următoarea topologie logică de reţea:

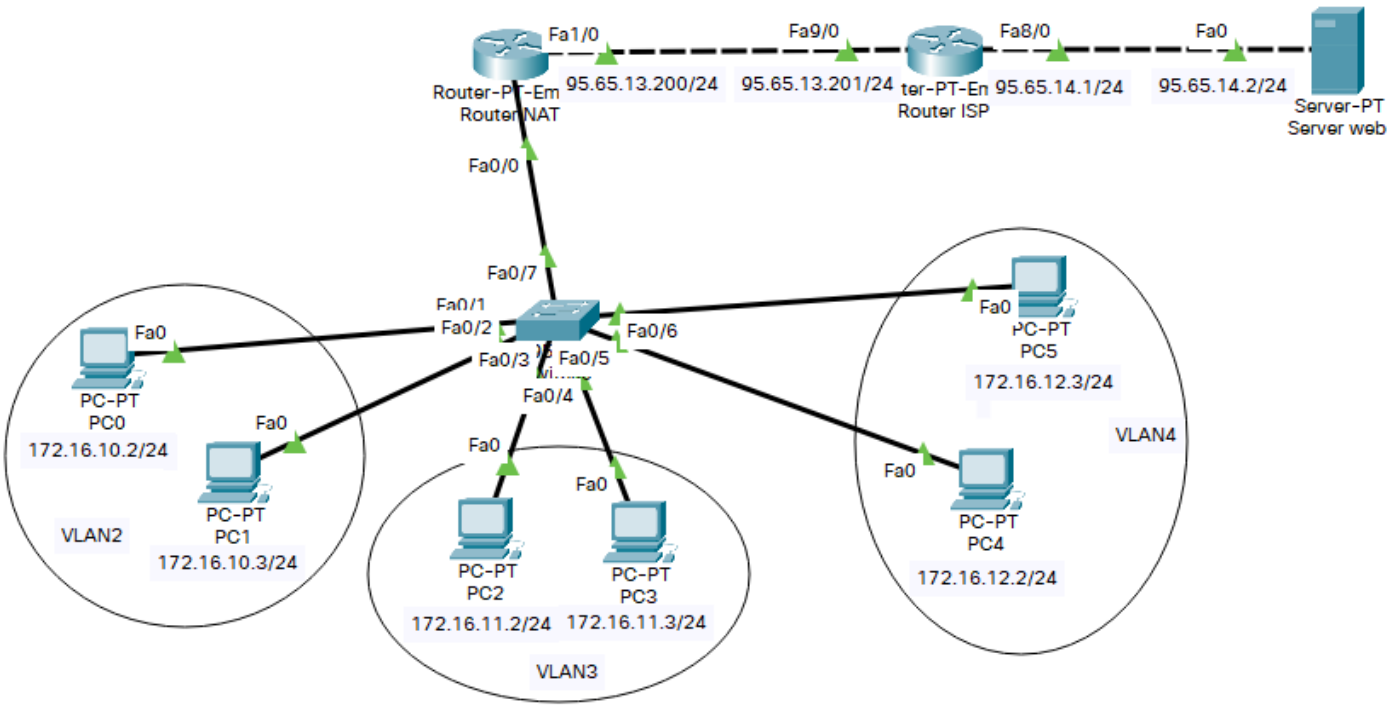


Figura 3b

Vom configura mecanismul NAT pentru cele 3 VLAN-uri din reţeaua reprezentată în Figura 3b. Este posibil de asigurat translatarea adreselor private din interior într-o singură adresă publică sau în adrese publice separate. Vom ilustra varianta cu o singură adresă publică.

Pe switch am configurat legăturile de tip acces şi de tip trunk, iar pe routerul NAT – 3 subinterfeţe, câte una pentru fiecare VLAN.

Pe routerul NAT pentru fiecare VLAN vom crea câte o listă de acces:

```
Router(config)#access-list 2 permit 172.16.10.0 0.0.0.255
Router(config)#access-list 3 permit 172.16.11.0 0.0.0.255
Router(config)#access-list 4 permit 172.16.12.0 0.0.0.255
```

În continuare, creăm mulţimea pool pentru VLAN-uri, aceeaşi pentru fiecare VLAN:

```
ip nat pool trans 95.65.13.200 95.65.13.200 netmask 255.255.255.0
```

Configurăm tipul de NAT cu overload, corespunzător fiecărui VLAN:

```
Router(config)#ip nat inside source list 2 pool trans overload
Router(config)#ip nat inside source list 3 pool trans overload
Router(config)#ip nat inside source list 4 pool trans overload
```

În partea finală definim interfeţele NAT de interior şi de exterior. Atenţie, aici se utilizează subinterfeţele

create pentru VLAN-uri şi nu interfaţa logică!

```
Router(config)#int fa 1/0
Router(config-if)#ip nat outside
Router(config-if)#int fa 0/0.2
Router(config-subif)#ip nat inside
Router(config-subif)#int fa 0/0.3
Router(config-subif)#ip nat inside
Router(config-subif)#int fa 0/0.4
Router(config-subif)#ip nat inside
Router(config-subif)#exit
```

Dacă se dă un ping de la oricare dintre staţiile din VLAN-uri, se vede că procedura decurge corect!

Protocolul DHCP (Dynamic Host Configuration Protocol) – protocolul de configurare dinamică a host-ului => permite host-urilor din reţea să obţină şi să seteze în mod automatizat o IP adresă
Dacă se vor atribui în mod manual IP adrese host-urilor, atunci administratorul reţelei va trebui să formeze manual şi să întreţină o bază de date cu aceste adrese, ceea ce poate fi un lucru migălos
De ce DHCP nu este numit protocol de stabilire automatizată a IP adresei? Deoarece DHCP distribuie (şi atribuie) nu doar IP adresa, ci şi masca de subreţea, adresa routerului implicit, adresa serverului DNS şi alte date.
Protocolul DHCP poate fi configurat atât pe un server dedicat, cât şi pe un router din reţea
În continuare, examinăm cum se efectuează configurarea serverului DHCP pe dispozitive Cisco (server dedicat sau router), folosind Cisco Packet Tracer

1. Cazul în care clientul DHCP se află în acelaşi domeniu broadcast ca şi serverul DHCP

Să considerăm o configuraţie de reţea formată din 2 host-uri, 1 switch şi 1 server, care va îndeplini rolul de DHCP server (a se vedea Figura 4)

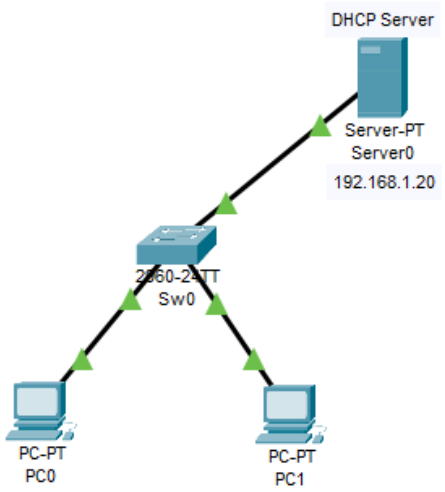


Figura 4

Deoarece un server DHCP nu poate să-şi atribuie IP adresă, o vom atribui manual acestuia şi anume IP adresa 192.168.1.20 şi masca de subreţea 255.255.255.0.
Dăm un click pe serverul DHCP, apoi pe tab-ul Services şi pe butonul DHCP (amplasat pe stânga), după care bifăm pe *On* pe lângă *Service* (a se vedea Figura 5).

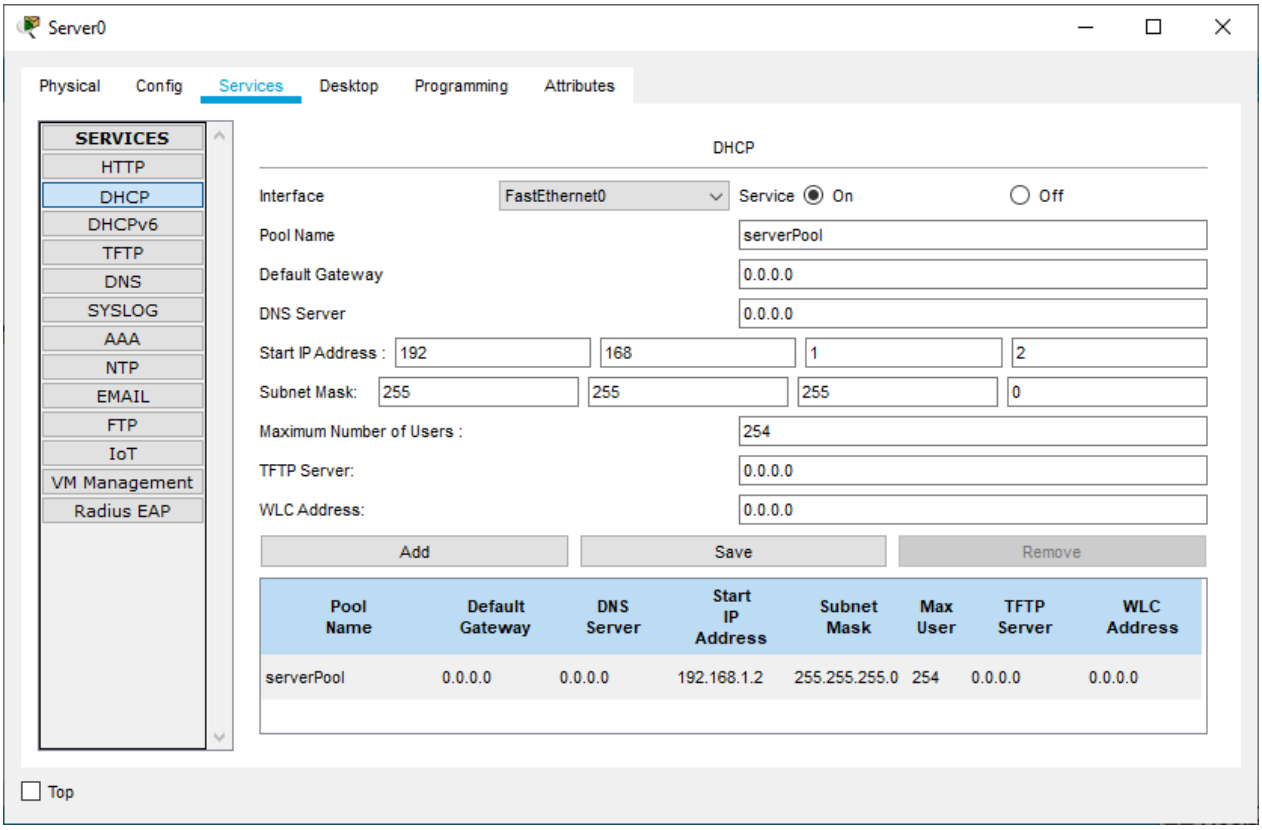


Figura 5

Indicăm intervalul din care se vor distribui IP adrese:

Start IP Address: 192.168.1.2

Subnet Mask: 255.255.255.0

La moment, nu vom indica valori pentru *Default Gateway* şi *DNS Server*

Apăsăm butonul *Save*

Mai jos în Figura 5 se vede că host-uri cu masca indicată pot fi maxim 254

În lucrările precedente pe host-uri completam manual IP Address, Subnet Mask şi Default Gateway. Acum pe *PC0 -> Desktop -> IP Configuration* punem bifa la DHCP. Peste un interval relative mic de timp host-ul PC0 va obţine IP adresa 192.168.1.2 şi masca de subreţea 255.255.255.0. Nu au fost atribuite valori pentru Default Gateway şi DNS deoarece acestea nu au fost configurate pe serverul DHCP. Să setăm pe serverul DHCP (Server0) valoarea Default Gateway la 192.168.1.1 (DNS îl vom atribui mai târziu) => Save. După aceasta, încă o dată bifăm la DHCP pe hostul PC0 => în mod automatizat vom obţine şi Default Gateway 192.168.1.1.

În mod analog se obţin valori pentru IP Address, Subnet Mask şi Default Gateway pe host-ul PC1.

Vom explica cum are loc stabilirea datelor IP prin protocolul DHCP:

Host-ul care solicită o IP adresă, încearcă să afle locaţia serverului DHCP. Pentru aceasta, host-ul formează un mesaj DHCP Discover (Identificare DHCP), care este transmis către toate dispozitivele din limitele domeniului broadcast. După ce serverul DHCP a recepţionat acest mesaj, serverul caută în baza sa cu adrese IP dacă sunt disponibile adrese IP. Dacă da, serverul DHCP selectează o adresă IP şi formează mesajul DHCP Offer (Oferta DHCP), pe care îl transmite clientului. Clientul examinează oferta şi dacă aceasta îl satisface, atunci formează mesajul DHCP Request (Cerere de IP către DHCP). Prin acest mesaj clientul confirmă că este de acord să utilizeze adresa IP propusă şi transmite acest mesaj sub formă broadcast, pentru ca celelalte servere DHCP (dacă există) din reţea să vadă că această IP adresă este atribuită acestui client. Serverul după recepţionarea mesajului DHCP Request, dacă încă nu a atribuit IP adresa curentă unui alt client, atunci formează şi transmite clientului mesajul DHCP Ack (confirmare DHCP), prin care îl anunţă pe client că contractul de închiriere a IP adresei este încheiat.

Dacă în domeniul broadcast există mai multe servere DHCP, atunci mesajul DHCP Discover este recepţionat de către toate aceste servere şi, în acest caz, acestea formează mesajul DHCP Offer pe care îl transmit clientului. Clientul alege o IP adresă şi transmite mesajul broadcast DHCP Request, deşi interacţiunea are loc doar cu serverul selectat.

Să ilustrăm acest proces în modul Simulation. Scoatem adresa IP de la PC0 şi bifăm din nou pe DHCP. Pe host-ul PC0 se formează mesajul broadcast DHCP Discover. Acest mesaj ajunge până la server, iar serverul găseşte un IP în baza sa de IP-uri, dar înainte ca să îl propună clientului, verifică dacă acest IP nu este utilizat de către vreun host în reţea (poate a fost atribuit de către un alt server!), printr-o solicitare ARP (pachet broadcast) prin care întreabă dacă în reţea este utilizată adresa IP curentă. Dacă la solicitarea ARP nu va fi un răspuns => rezultă că IP adresa este liberă şi poate fi atribuită clientului. Dar dacă va veni un răspuns la solicitarea ARP, atunci serverul va selecta o altă adresă IP pe care la fel o va verifica.
Peste ceva timp serverul va forma mesajul DHCP Offer pe care îl va transmite clientului. Dacă clientul este de acord cu oferta, atunci formează mesajul DHCP Request pe care îl transmite serverului, iar serverul transmite clientului mesajul de confirmare.
În partea finală a procesului, în mod analog, printr-o solicitare ARP clientul verifică IP adresa primită de la server.
Dacă în modul Simulation vom examina conţinutul pachetului DHCP, atunci ne vom convinge că DHCP este un protocol de nivel aplicaţie (ale modelelor OSI şi TCP/IP). Mesajul DHCP este încapsulat în câmpul de date al user datagramei UDP (la nivelul transport), care, la rândul său, este încapsulată într-o datagramă IP la nivelul reţea.
Prin definiţie, adresa IP este atribuită clientului pentru perioada de 24 ore (lease time = timpul de arendare a IP adresei). La expirarea a jumătate din acest interval de timp alocat, clientul PC0 iniţiază procedura de prelungire a arendei, prin transmiterea unui mesaj DHCP Request către server. Dacă de la serverul DHCP se va recepţiona un mesaj de confirmare DHCP Ack, aceasta va însemna că perioada *Lease time* a fost prelungită cu încă o zi. Dacă după o jumătate din timp serverul nu a trimis vreun răspuns, clientul în continuare transmite mesaje DHCP Request. Dar dacă şi în a doua jumătate de zi clientul nu reuşeşte să prelungească termenul de arendare a IP adresei, atunci clientul nu va mai putea să utilizeze IP adresa curentă şi va fi nevoit să solicite o altă adresă IP.
Pentru ce este necesar mecanismul de arendare a adresei IP? Dacă un careva host nu va funcţiona o perioadă de timp (24 de ore, de exemplu), atunci DHCP serverul trebuie să elimine înregistrarea conform căreia IP adresa este atribuită acestui client (pentru a avea posibilitate să o atribuie unui alt client, care solicită o IP adresă). Este necesar ca baza de date de pe server să fie permanent (dinamic) actualizată.
Timpul de arendare a adresei IP poate fi modificat. De exemplu, la o cafenea clientul a stat vreo 20 de minute în Internet, după care a plecat. În acest caz, nu are sens ca timpul de arendare să fie o zi, deoarece aceasta poate să conducă la epuizarea adreselor IP disponibile. Clientul demult a plecat, dar adresa IP este atribuită dispozitivului acestuia şi nu se repartizează unui alt client.
Serverul DHCP setat pe router
Nu numaidecât ca serverul DHCP să fie un dispozitiv dedicat. Acesta poate fi configurat pe un router din reţea. Să considerăm schema din Figura 6.

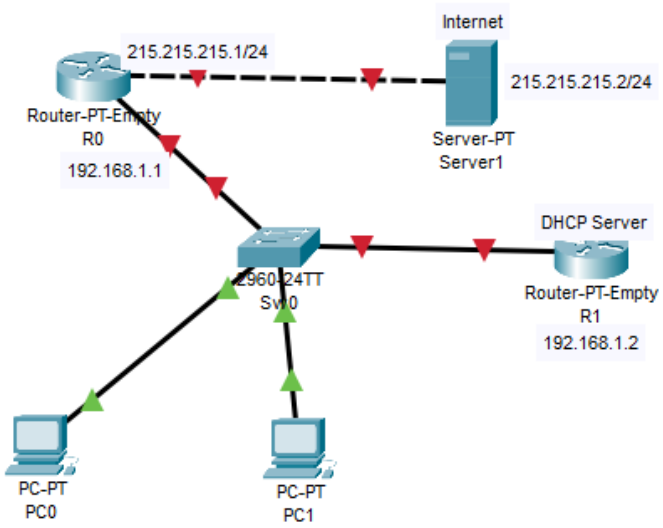


Figura 6

Inițial configurăm adresele IP pe routerele R0 și R1, dar și pe serverul Internet.
După aceea, pe routerul R1 definim mulțimea de adrese IP pe care le va putea atribui serverul DHCP:

```
Router(config)#ip dhcp pool NOVLANPOOL  // numele setului
Router(dhcp-config)#network 192.168.1.0 255.255.255.0 // se destribuie adrese de la 192.168.1.1 până la 192.168.1.254
Router(dhcp-config)#default-router 192.168.1.1    // adresa routerului implicit, care nu va fi atribuită host-urilor
Router(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.2 // se exclude posibilitatea atribuirii acestor adrese
```

Accesăm host-ul PC0 => activăm DHCP => a fost atribuită host-ului adresa IP 192.168.1.3
De pe PC0 dăm un ping 215.215.215.2 (pentru ieșire în Internet) => este conexiune
Accesăm host-ul PC1 => activăm DHCP => a fost atribuită adresa IP 192.168.1.4
De pe PC1 dăm un ping 215.215.215.2 (pentru ieșire în Internet) => este conexiune
Comanda *show ip dhcp binding* arată adresele IP (la fel și adresele MAC) care au fost atribuite clienților

```
Router(config)#do show ip dhcp binding
IP address      Client-ID/              Lease expiration        Type
                Hardware address
192.168.1.3     00E0.F774.743A          --                      Automatic
192.168.1.4     000C.8554.0260          --                      Automatic
```

Dacă accesăm PC0 și schimbăm bifa de la DHCP la Static, atunci PC0 va trimite către serverul DHCP un mesaj prin care îl anunță că nu mai are nevoie de acel IP
Trecem în modul Simulation. Pe host-ul PC0 schimbăm bifa pe Static => vedem că de la PC0 se transmite pachetul DHCP Release (eliberare). Dăm încă o dată comanda *show ip dhcp binding* => vedem că adresa IP 192.168.1.3 a fost eliberată și poate fi repartizată unui alt client
Acum pe PC0 schimbăm bifa de la Static la DHCP => dăm comanda *show ip dhcp binding* => vedem că adresa IP 192.168.1.3 din nou a fost repartizată lui PC0.
Totuși, dacă vom întrerupe conexiunea dintre PC0 și Sw0 => vom vedea că adresa IP 192.168.1.3 nu a fost eliberată. Iată aici și își găsește aplicație perioada Lease time – dacă peste o zi PC0 nu va transmite mesajul cu cererea de actualizare => adresa IP 192.168.1.3 va fi eliberată (înregistrarea din tabel va fi eliminată)
Perioada de arendare a adresei IP poate fi modificată, folosind comanda

```
lease {days [hours] [minutes]| infinite}
```

Valoarea *infinite* specifică posibilitatea de arendare nelimitată a adresei IP. Un exemplu de comandă este

```
Router(dhcp-config)# lease 30
```

Schimbăm pe host-urile PC0 și PC1 bifele de pe DHCP pe Static => tabelul cu adresele IP atribuite nu mai conține date
Pe PC1 definim adresa statică:
IP Address: 192.168.1.3
Subnet Mask: 255.255.255.0
Atunci când punem bifa la DHCP pe host-ul PC0, serverul DHCP va încerca să atribuie adresa IP 192.168.1.3
Să trecem în modul Simulation pentru a vedea ce se întâmplă:
Serverul încearcă să afle dacă este deja atrbuită adresa IP 192.168.1.3 și, într-adevăr, vede că aceasta este deja utilizată de altcineva
Apoi verifică adresa 192.168.1.4 și vede că este liberă
Comanda *show ip dhcp conflict* arată adresele IP pe care serverul DHCP a încercat să le repartizeze și a stabilit că acestea sunt deja ocupate:

```
Router(config)#do show ip dhcp conflict
IP address      Detection method   Detection time          VRF
192.168.1.3     Ping               mar. 1 1993 01:10 a.m.
```

Comanda *clear ip dhcp conflict ** elimină tot conținutul tabelului cu IP-urile ce nu au putut fi distribuite

Configurarea serverului DHCP dacă în rețea sunt VLAN-uri

Cum se configurează un server DHCP în cazul în care în rețea avem mai multe VLAN-uri? Cum se poate indica serverului DHCP că dacă host-urile sunt din același VLAN – atunci urmează ca serverul să distribuie adrese IP din aceeași subrețea, iar dacă host-urile sunt din VLAN-uri diferite – atunci lor li se vor atribui adrese IP din diferite subrețele? Să considerăm configurația din Figura 7.

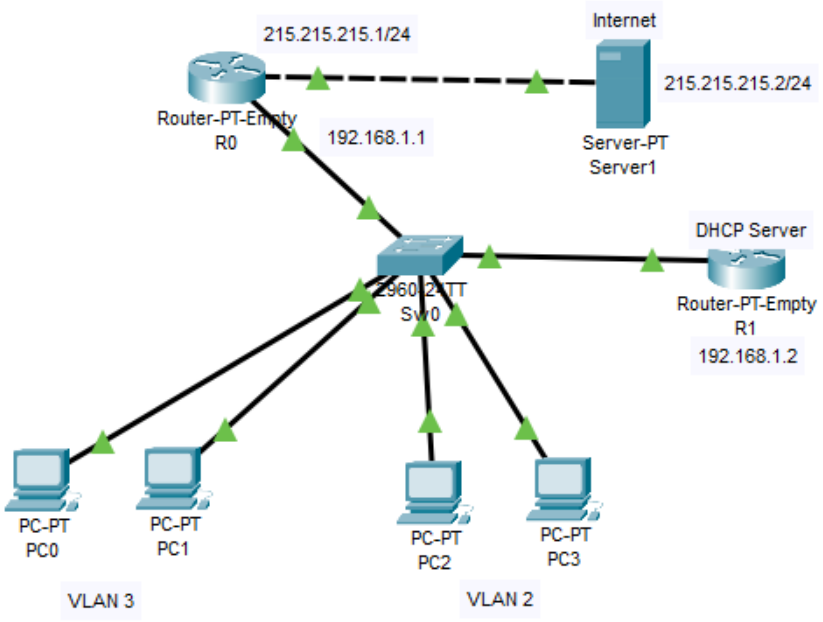


Figura 7

Mai întâi se configurează switch-ul Sw0 astfel încât să funcţioneze cu VLAN-uri:

```
Switch(config)#int fa 0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access VLAN 3
Switch(config-if)#exit
Switch(config)#int fa 0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access VLAN 3
Switch(config-if)#exit
Switch(config)#int fa 0/5
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access VLAN 2
Switch(config-if)#exit
Switch(config)#int fa 0/6
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access VLAN 2
Switch(config-if)#exit
Switch(config)#do wr
```

Mai întâi pe toate host-urile se pune bifa la Static. Pe routerul cu rol de server DHCP se vor crea două subinterfeţe logice (deoarece avem două VLAN-uri în reţea):

```
R1(config)#int fa 9/0
R1(config-if)#no ip add
R1(config-if)#exit
R1(config)#no ip dhcp pool NOVLANPOOL
R1(config)#no ip dhcp excluded-address 192.168.1.1 192.168.1.2
R1(config)#int fa 9/0.2
R1(config-subif)#
R1(config-subif)#encapsulation dot1Q 2
R1(config-subif)#ip add 192.168.2.2 255.255.255.0
R1(config-subif)#no shut
R1(config-subif)#exit
R1(config)#int fa 9/0.3
R1(config-subif)#encapsulation dot1Q 3
R1(config-subif)#ip add 192.168.3.2 255.255.255.0
R1(config-subif)#no shut
R1(config-subif)#exit
```

Mai departe, pe routerul R1 se va crea set-ul de IP-uri pentru VLAN-ul 2 şi set-ul de IP-uri pentru VLAN-ul 3:

```
R1(config)#ip dhcp pool vlan2
R1(dhcp-config)#network 192.168.2.0 255.255.255.0
```

```
R1(dhcp-config)#default-router 192.168.2.1 // va trebui să completăm şi pe R0
R1(dhcp-config)#exit
R1(config)#ip dhcp pool vlan3
R1(dhcp-config)#network 192.168.3.0 255.255.255.0
R1(dhcp-config)#default-router 192.168.3.1
R1(dhcp-config)#exit
```

În fine, conexiunea dintre Sw0 şi R1 trebuie făcută de tip trunk. Serverul DHCP va lucra cu frame-uri Ethernet etichetate (cu tag-uri VLAN). Pe Sw0 scriem:

```
Switch(config)#int fa 0/4
Switch(config-if)#sw mode trunk
Switch(config-if)#exit
```

Acum intrăm pe host-uri şi punem bifa la DHCP. Vedem că acestora li se atribuie IP adrese şi se ţine cont de VLAN-ul din care face parte host-ul.
Adresele IP 192.168.2.1, 192.168.2.2, 192.168.3.1 şi 192.168.3.2 trebuie să fie rezervate:

```
R1(config)#ip dhcp excluded-address 192.168.2.1 192.168.2.2
R1(config)#ip dhcp excluded-address 192.168.3.1 192.168.3.2
```

În ce mod serverul DHCP înţelege cărui host urmează să îi repartizeze un IP din subreţeaua doi şi cărui host – un IP din subreţeaua trei? Răspuns - în baza interfeţei de la care a recepţionat această interogare. Pentru a asigura ieşirea în Internet pe interfaţa Fa9/0 a routerului R0 este necesar de creat subinterfeţele Fa 9/0.2 şi Fa 9/0.3, cu adresa IP 192.168.2.1 şi, corespunzător, 192.168.3.1:

```
Router(config)#int fa 9/0.2
Router(config-subif)#encapsulation dot1Q 2
Router(config-subif)#ip add 192.168.2.1 255.255.255.0
Router(config-subif)#int fa 9/0.3
Router(config-subif)#encapsulation dot1Q 3
Router(config-subif)#ip add 192.168.3.1 255.255.255.0
Router(config-subif)#int fa 9/0
Router(config-if)#no ip add
Router(config-if)#exit
```

Pe legătura dintre R0 şi Sw0 se va crea un port trunk:

```
Switch>en
Switch#conf ter
Switch(config)#int fa 0/3
Switch(config-if)#switchport mode trunk
```

Pe fiecare host punem bifa la DHCP
De pe host-urile PC2 şi PC3: ping 215.215.215.2 => este conexiune
De pe host-urile PC0 şi PC1: ping 215.215.215.2 => este conexiune

2. Cazul în care clientul se află într-un alt domeniu broadcast în raport cu serverul DHCP

Ce e de făcut dacă serverul DHCP nu se află în acelaşi domeniu broadcast ca şi host-ul ce solicită date IP? Pentru soluţionarea acestei probleme este aplicată schema DHCP Relay (retranslaţie), care utilizează în calitate de retranslator un switch de nivelul 3 (L3 Switch). Retranslatorul va „prinde" frame-urile broadcast ce conţin mesaje DHCP şi le va muta într-un alt domeniu broadcast. Pentru aceasta pe switch-ul de nivel 3 se activează funcţia de retranslare corespunzătoare.
Să examinăm următoarea configuraţie de reţea (a se vedea Figura 8):

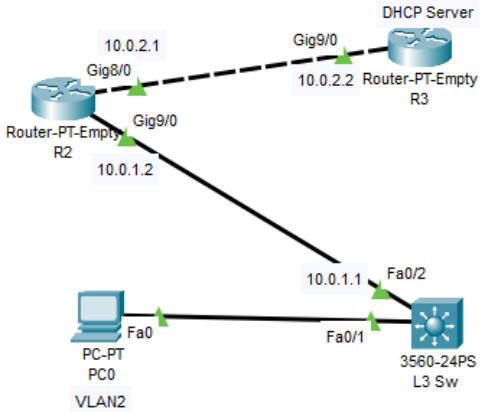


Figura 8

Host-ul PC0 se află la distanţă de 3 domenii broadcast de la serverul DHCP R3

```
Pe R2:
Router(config)#int gig 8/0
Router(config-if)#ip add 10.0.2.1 255.255.255.0
Router(config-if)#no sh
Router(config-if)#exit
Router(config)#int gig 9/0
Router(config-if)#ip add 10.0.1.2 255.255.255.0
Router(config-if)#no sh
Router(config-if)#exit

Pe R3:
Router(config)#int gig 9/0
Router(config-if)#ip add 10.0.2.2 255.255.255.0
Router(config-if)#no sh
Router(config-if)#exit
```

La fel, pe routerul R3 creăm set-ul de IP adrese pe care va putea să le repartizeze serverul DHCP:

```
Router(config)#ip dhcp pool VLAN2POOL
Router(dhcp-config)#network 192.168.2.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.2.1
```

Configurările de pe switch-ul de nivel 3, L3 Sw:

```
Switch(config)#int fa 0/2
Switch(config-if)#no switchport
Switch(config-if)#ip add 10.0.1.1 255.255.255.0
Switch(config-if)#exit
Switch(config)#ip routing
Switch(config)#int fa 0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#exit
Switch(config)#spanning-tree mode rapid-pvst
```

Dacă la acest moment vom încerca să obţinem o adresă IP prin DHCP (punem bifa la DHCP!), vom vedea că serverul DHCP nu funcţionează corect. Este necesar de făcut ca L3 Sw să transforme pachetul broadcast în unul unicast şi să îl retransleze anume către serverul DHCP. Pentru aceasta pe L3 Sw dăm comenzile:

```
Switch(config)#int vlan 2
Switch(config-if)#ip add 192.168.2.1 255.255.255.0
Switch(config-if)#ip helper-address 10.0.2.2   // IP-ul serverului DHCP
```

Adică L3 Sw pe interfaţa virtuală vlan 2 va „prinde” frame-urile broadcast şi dacă va vedea că sunt destinate pentru DHCP server, le va transmite ca unicast către server pe adresa IP 10.0.2.2. Dar până când L3 Sw nu ştie cum să ajungă până la reţeaua 10.0.2.0 (unde se află serverul DHCP). Deci este necesar pe L3 Sw de scris ruta:

```
Switch(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.2
```

Iar pe serverul DHCP:

```
R3(config)#ip route 192.168.2.0 255.255.255.0 10.0.2.1
```

Iar pe routerul R2:

```
R2(config)#ip route 192.168.2.0 255.255.255.0 10.0.1.1
```

Acum dacă schimbăm pe host-ul PC0 bifa de la Static la DHCP => se atribuie corect adresa IP

Varianta când avem mai multe VLAN-uri

Să examinăm configuraţia de reţea din Figura 9:

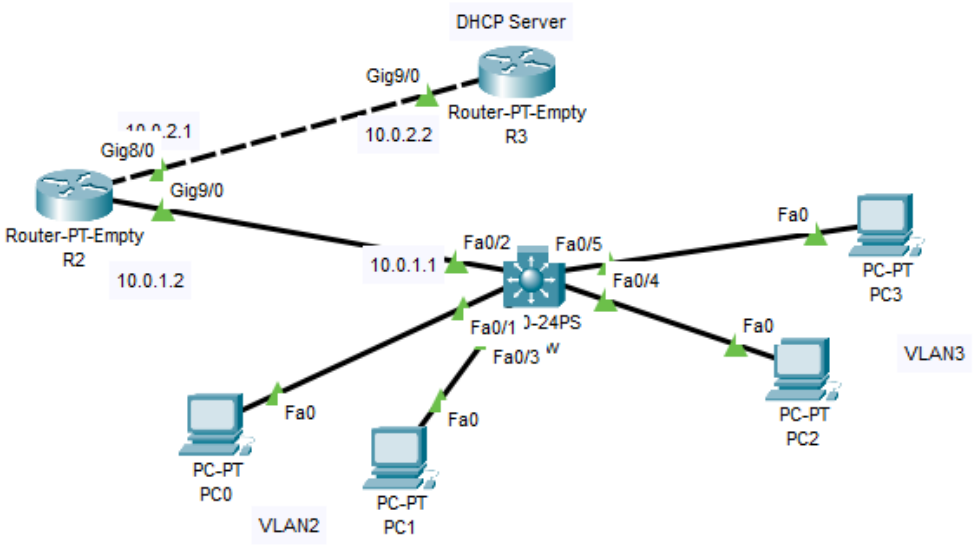


Figura 9

Salvăm configurările efectuate mai sus.
Dar adăugăm următoarele:
pe L3 Sw:

```
Switch(config)#int fa 0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#exit
Switch(config)#int fa 0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#exit
Switch(config)#int fa 0/5
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#exit
```

```
Pe R3:
R3(config)#ip route 192.168.3.0 255.255.255.0 10.0.2.1
Pe R2:
R2(config)#ip route 192.168.3.0 255.255.255.0 10.0.1.1
```

Pe serverul DHCP este necesar de creat set-ul de adrese IP pentru VLAN 3:

```
R3(config)#ip dhcp pool VLAN3POOL
R3(dhcp-config)#network 192.168.3.0 255.255.255.0
R3(dhcp-config)#default-router 192.168.3.1
```

Pe routerul R3 interzicem atribuirea adreselor IP 192.168.2.1, 192.168.2.2, 192.168.3.1 şi 192.168.3.2:

```
R3(config)#ip dhcp excluded-address 192.168.2.1 192.168.2.2
R3(config)#ip dhcp excluded-address 192.168.3.1 192.168.3.2
```

Pe L3 Sw se va crea interfaţa virtuală:

```
Switch(config)#int vlan 3
Switch(config-if)#ip add 192.168.3.1 255.255.255.0
Switch(config-if)#ip helper-address 10.0.2.2
Switch(config-if)#exit
```

Acum intrăm pe host-urile PC0, PC1, PC2, PC3 şi bifăm pe DHCP => se atribuie corect IP adrese în fiecare VLAN.

Etichetarea VLAN funcţionează doar în limitele unui domeniu broadcast, adică la ieşirea din L3 Sw din frame se va şterge eticheta. Când mesajul DHCP va ajunge la L3 Sw, pe care este activată funcţia de retranslator, L3 Sw va plasa în câmpul Agent IP Address adresa IP de pe interfaţa lui virtuală: dacă este VLAN 2, atunci va lucra a doua interfaţă virtuală, iar în mesajul DHCP în câmpul Relay Agent Address se va scrie această adresă IP.
Să vedem în modul Simulation – pachetul DHCP de la PC0 (bifăm la DHCP) ajunge la L3 Sw => deschidem

Outbound PDU pe L3 Sw – în pachetul DHCP avem câmpul RELAY AGENT ADDRESS inițializat cu valoarea 192.168.2.1 => această adresă IP este stocată. Când pachetul ajunge la serverul DHCP, acela vede acest IP și astfel află care set de IP-uri să utilizeze pentru a repartiza un IP către client (dacă IP-ul este din subrețeaua 2 => se va repartiza un IP din subrețeaua 2).