Laborator 3
ACL extinse; rutarea dinamică folosind OSPF

Partea 1. Breviar teoretic

1.1 ACL extinse

Reamintim că listele de acces (ACL) sînt de 2 tipuri:

- **Standard ACLs** au numere între 1-99, şi nu permit specificarea decît a *adresei sursă* a pachetului IP. De aceea, ele se plaseaza în reţea cît mai aproape de *destinaţia* afectată de ACL, în ideea că pachetele pot ajunge la acea destinaţie pe mai multe căi.
- **Extended ACLs** au numere între 100-199 sau 2000-2699. Ele permit specificarea *adresei sursă, adresei destinaţie, protocolului* şi *portului* ceea ce le face mult mai versatile. Aşadar ele se pot plasa cît mai aproape de *sursa* pachetelor afectate de ACL, pentru a reduce traficul inutil: din moment ce e cunoscută destinaţia, nu are sens să lăsăm pachetele să circule prin toate nodurile intermediare, ci să le eliminam chiar de la începutul drumului lor.

configurarea (parţială) a unui ACL extins se face astfel:

```
Router(config)# access-list număr deny|permit [remark comentariu]
protocol sursa wildcard destinaţia wildcard [port] [established]
[log]
```

În plus faţă de varianta standard, aici se specifică ambele adrese (sursă şi destinaţie), protocolul (poate fi **ip, icmp, tcp, udp** şi altele; ip le include pe toate), portul însoţit de un modificator: **eq** pentru egal, **neq** pentru non-egal, **gt** pentru "greater than", **lt** pentru "less than" sau un domeniu de porturi specificat cu **range** x y; opţiunea **established** selectează numai acele pachete dintr-o conexiune TCP deja stabilită (nu şi pachetele de stabilire a conexiunii care au flagul SYN), deci configurînd o regula cu **allow** numai pentru pachetele **established** putem să permitem traficul către un host, fără a permite ca altcineva să încerce să se *conecteze* pe acel host.

Exemple:

```
access-list 101 permit ip host 192.168.1.1 host 192.168.7.5
```
permite traficul între adresele specificate

```
access-list 102 deny tcp host 192.168.1.1 any eq 80
```
nu permite traficul de la hostul specificat către nici o destinaţie pe portul 80, implicit interzicînd accesul la web

```
access-list 103 permit tcp 192.168.1.0 0.0.0.255 host 193.1.2.3 eq
23 established
```
permite traficul telnet (tcp, portul 23) de la hosturile din reţeaua 192.168.1.0/24 către hostul specificat, dar numai dacă conexiunea este deja stabilită, adică dacă hostul specificat a iniţiat el conexiunea respectivă (traficul telnet este bidirecţional). Pachetele de stabilire a conexiunii TCP, conţinînd flag-ul TCP *SYN*, nu vor fi permise.

Porturile TCP cele mai uzuale: 20=FTP-DATA, 21=FTP, 22=SSH, 23=Telnet, 25=SMTP (Internet e-mail), 80=HTTP, 110=POP3 iar pentru UDP: 53=DNS, 69=TFTP, 161=SNMP

Ca şi în cazul ACL standard, un ACL extins trebuie plasat pe o interfaţă a ruterului, folosind aceeaşi comandă `ip access-group număr_ACL in|out`

Datorită existenţei a 2 perechi adresă/ *wildcard mask*, faţă de 1 pereche la standard, la ACL extinse regula implicită care există la sfîrşit este `deny ip any any` (în loc de `deny any`), iar pentru ca comportamentul impilicit să fie de a permite tot traficul nespecificat, regula care trebuie scrisă ultima este `permit ip any any`.

1.2 Protocolul OSPF - generalităţi

Protocolul OSPF (*Open Shortest Path First*) este un protocol de rutare dinamică de tip *link-state*, spre deosebire de RIP şi IGRP care erau de tip *distance-vector*. Aceasta înseamnă ca toate ruterele cunosc topologia completă a reţelei şi pot lua decizii de rutare fără riscul de a genera bucle în reţea. Metrica folosită se bazeaza pe un cost care este de obicei *banda disponibilă pe fiecare legătura*, în timp ce la RIP metrica era dată de *numărul de hopuri* (aceasta înseamnă ca, la RIP, este preferata o cale directă între 2 noduri pe o legătura seriala de viteza mică în locul unei cai indirecte, cu un nod intermediar, dar pe o legătura Gigabit Ethernet). Un alt avantaj al OSPF faţă de RIP este numărul mai mare de hopuri suportate (255 faţă de 15) ceea ce permite funcţionarea în reţele de dimensiuni medii/mari.

Ca şi RIP, OSPF este un protocol bazat pe un standard (*open*) astfel încît este posibilă interoperarea între mai multi producători de rutere. Algoritmul pentru determinarea distanţei minime între oricare 2 noduri (*shortest path*) este algoritmul Dijkstra.

RIP şi OSPF sînt protocoale de rutare *interioare*, adică sînt folosite în interiorul unui domeniu aflat sub o administraţie comună. Un astfel de domeniu se numeşte AS (*Autonomous System*). Rutele, protocoalele, politicile de rutare şi celelate aspecte tehnice şi administrative se definesc în mod unitar şi sînt sub controlul respectivului AS. Prin comparaţie, există şi protocoale de rutare *exterioare* (de exemplu BGP) care sînt folosite *între* AS diferite.

Protocoalele interioare se mai numesc IGP (*internal gateway protocol)* şi cele exterioare EGP.

Din punct de vedere al funcţionării, OSPF este semnificativ mai complex decît RIP. Mesajele de rutare schimbate între rutere se numesc LSA (*Link State Advertisements*). Un ruter trimite în LSA informaţii precum starea legăturilor sale cu celelalte rutere conectate *(link states*). Pe baza acestora fiecare ruter din reţea crează o tabelă (*link state database*). Aceste tabele trebuie să fie identice pentru toate ruterele din reţea (sau, după cum se va vedea, măcar din cadrul unei arii). Aplicînd algoritmul SPF asupra acestei tabele, *fiecare ruter* obţine căile de cost minim către toate destinaţiile din tabelă, ceea ce formează *tabela de rutare (routing table).*

Pentru a reduce numărul de LSA în reţele de mari dimensiuni se foloseşte conceptul de arie (*Area*) care reprezintă o anumită parte a reţelei (Cisco recomandă ca într-o arie să nu fie mai mult de 50 de rutere, dar se pot face şi arii mult mai mici – depinde de viteza legăturilor disponibile, precum şi de puterea de calcul a ruterelor). Împărţirea în arii se face astfel încît să existe o arie numită *Backbone Area* – aria la care se leagă toate celelalte arii, dacă există. Aceasta se numerotează obligatoriu *Area 0*, celalte arii putînd fi numerotate oricum. Informaţiile de *link state* sînt identice numai pentru ruterele din interiorul unei arii. Între două arii nu se transferă informaţii de *link state* referitoare la ruterele din interiorul fiecarei arii, ci doar la ruterele de margine (*Area Border Routers*), ceea ce face OSPF scalabil în AS-uri mari.

1.3 Alegerea DR/BDR

Un ruter îşi crează o tabelă cu toţi vecinii direct conectaţi, numită *Adjacency database*. În cazul unei reţele *point-to-point* cum sînt reţelele seriale, care au fix 2 noduri, este evident că fiecare ruter are un singur vecin pe fiecare astfel de reţea. Dar există şi reţele de tip *broadcast* (în care se

pot transmite mesaje de "difuzare" care pot ajunge la mai mulţi destinatari) cum sînt reţelele Ethernet, care pot conecta în mod direct un număr mare de noduri. Pentru a minimiza numărul de LSA, pe o astfel de reţea au loc "alegeri" şi se desemnează un ruter "şef" numit DR (*Designated Router*) şi o "rezervă" numita BDR (*Backup Designated Router*). Doar DR dintr-o reţea este cel care primeşte/diseminează informaţia de rutare de la/către celelalte rutere din celelate reţele.

Alegerea unui DR/BDR se face pe toate reţelele *broadcast*, chiar dacă la un moment dat într-o astfel de reţea există doar 2 rutere (ca pe o legătură *point-to-point*) şi chiar cînd există unul singur (şi restul nodurilor sînt PC-uri care nu participă la procesul de rutare). Procesul de alegere se bazează pe 2 valori numerice:

- *router ID*, unic la nivel de ruter
- *priority*, la nivel de interfaţă

Dacă nu s-a configurat manual un identificator *router ID*, ruterul alege adresa IP cea mai mare (ca număr de 32 de biţi) dintre cele existente pe interfeţele ruterului. Dacă există şi interfeţe loopback, se alege cea mai mare adresă dintre acestea, chiar dacă este mai mică decît a unei interfeţe non-loopback. Aceasta pentru că interfeţele loopback sînt stabile (nu devin "down" din cauza deconectării unui cablu, de exemplu).

Prioritatea (*priority*) fiecărei interfeţe este implicit 1 şi se poate modifica manual între 0 şi 255. Prioritatea este la nivel de interfaţă pentru că un ruter poate fi conectat la mai multe reţele *broadcast* dacă are mai multe interfeţe; în fiecare reţea în care participă, el poate avea un statut diferit –într-o reţea poate fi DR şi în alta nu.

Procesul de alegere pe o reţea este următorul:

- ruterul cu prioritatea cea mai mare devine DR şi următorul devine BDR. Prioritatea 0 înseamnă că ruterul nu poate ajunge DR.
- în caz de egalitate a priorităţilor, se reia procesul comparînd valorile *router ID*

1.4 *wildcard mask* în OSPF

Chiar dacă funcţionarea OSPF diferă fundamental faţă de RIP, configurarea în interiorul unei singure zone este foarte asemănătoare cu RIP. O diferenţă notabilă este folosirea *wildcard mask* la definirea reţelelor.

Un *wildcard mask* (similar cu cel folosit la ACL) este un număr de 32 biţi care permite "mascarea" sau mai precis "selectarea" anumitor biţi dintr-o adresă IP ca specificînd reţeaua dorită. Regula este urmatoarea: pentru ca un bit din adresa sa fie selectat, pe poziţia respectivă din masca trebuie sa fie "0". Asadar "0" are semnificatia "*Check*" şi "1" are semnificatia *"Ignore"*.

În cazul OSPF se observă că ***wildcard mask* este inversul *netmask*-ului**, întrucît convenţia la *netmask* era ca biţii din partea de reţea sa fie marcaţi cu "1".

De exemplu, o reţea de clasă C 192.168.1.0/24 cu *netmask* 255.255.255.0 are *wildcard mask* 0.0.0.255. Similar, o reţea 192.168.100.128/26, cu *netmask* 255.255.255.192 are *wildcard mask* 0.0.0.63 (ultimul octet este 11000000 la *netmask* şi 00111111 la *wildcard mask*).

1.5 Configurarea OSPF cu o singură arie (*single-area* sau *Area 0*)

Configurarea se face astfel: pentru fiecare ruter, se porneşte procesul de rutare OSPF cu un *process id* (un număr la alegere) şi se specifică fiecare reţea direct conectată, *wildcard mask*-ul acesteia şi aria (*area*) din care face parte:

```
Router(config)# router ospf 1
Router(config-router)#network reţea wildcard_mask area number
Router(config-router)#network reţea wildcard_mask area number
Router(config-router)# … alte reţele …
Router(config-router)# exit
```

Process Id-ul este neimportant în acest caz, poate fi util dacă se pornesc mai multe procese de rutare pe acelasi ruter.

De exemplu, fie o topologie ca în figura 1:

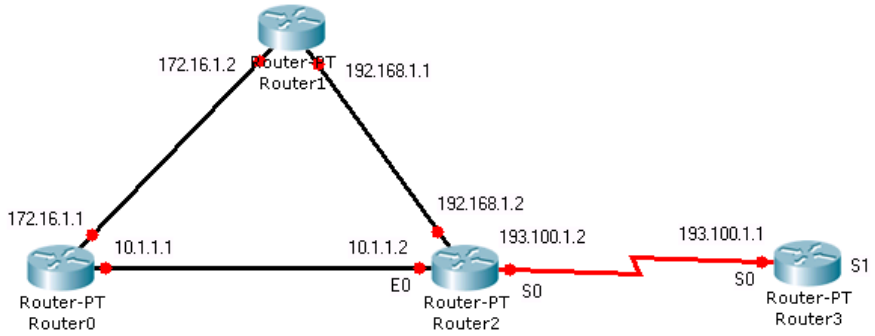

Fig. 1 Exemplu de topologie

OSPF, ca şi RIP v2, suportă VLSM (*variable length subnet mask*, adică măştile de reţea pot fi de lungimi diferite, şi nu se respectă obligatoriu măştile implicite date de clasele A,B,C). Presupunem că toate reţelele din figură au măştile implicite: 10.0.0.0/8, 172.16.0.0/16, etc. Atunci, pe ruterul 0:

```
Router0(config)# router ospf 1
Router0(config-router)# network 172.16.0.0 0.0.255.255 area 0
Router0(config-router)# network 10.0.0.0 0.255.255.255 area 0
```

Cîteva comenzi care permit studiul tabelelor de rutare şi a altor aspecte asociate rutării:

comanda	*descriere*
`show ip route`	arată tabela de rutare; rutele sînt clasificate după tip: cele direct conectate, cele introduse manual (static) şi cele învăţate prin intermediul protocoalelor de rutare
`show ip route` *`destination_network`*	arată informaţii despre ruta către un net
`show ip protocols`	oferă informaţii despre protocoalele de rutare care ruleaza
`show ip ospf`	informaţii despre OSPF, rutere, arii, statistici
`show ip ospf database`	informaţii despre tabela de rutare OSPF
`show ip ospf interface`	informaţii specifice OSPF despre o interfaţă
`show ip ospf neighbor`	informaţii despre adiacente (vecinii direct conectati)

1.6 Configurarea OSPF cu mai multe arii *(multi-area)*

Avantajele împărţirii în mai multe arii:
- tabele de rutare mai mici;
- informaţia de semnalizare mai puţină (se evită inundarea reţelei);
- capacitatea necesară de procesare a ruterelor mai scăzută;
- se reduce frecvenţa de calcul a arborelui SPF (Shortest Path First).

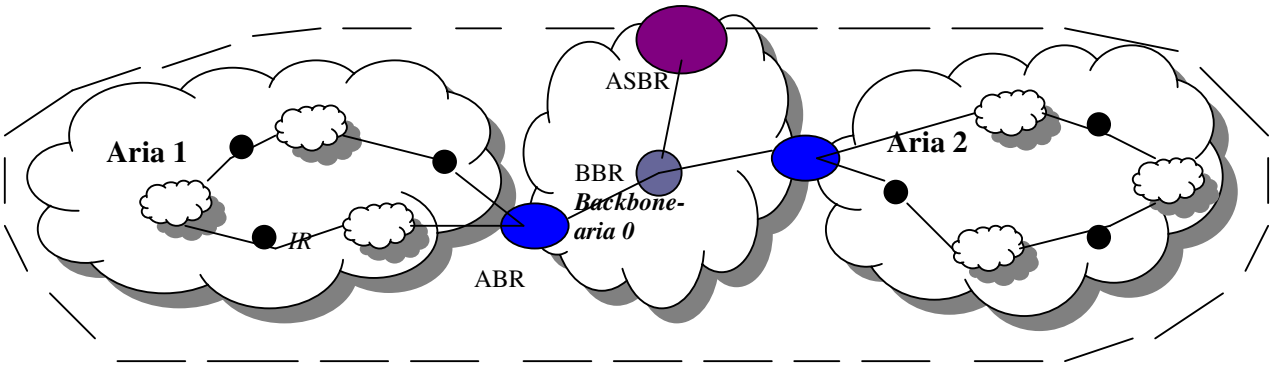

Fig. 2 Exemplu de împărţire a unui AS în trei arii
(Aria 0 –Backbone, Aria 1 şi Aria 2)

Terminologie folosită:
IR – Interior Router
BBR – Backbone Router
BR – Border Router
ABR – Area Border Router
ASBR – Autonomous System Boundary Router

Ruterul interior **IR** va rula o instanţă interioară a algoritmului; el cunoaşte complet doar aria sa.

Ruterul de la frontiera unei arii (care e conectat la backbone – **ABR**) va rula o instanţă de proces pentru aria sa şi încă o instanţă de proces pentru Backbone.

Ruterul de legătura cu alte sisteme autonome **ASBR** rulează o instanţă de proces pentru Backbone dar şi un protocol exterior de rutare – EGP (informaţiile despre rute exterioare sînt distribuite de ASBR în interiorul domeniului).

Observaţie 1: Fiecare arie are:
- propriul algoritm OSPF;
- propria baza de date;
- topologia ariei nu este vizibila în exterior (OSPF imprumută şi de la protocoale bazate *distance-vector*).

Observaţie 2: pentru a reduce şi mai mult numărul de mesaje LSA schimbate între arii, acele arii care se conectează printr-un singur punct la restul AS (au un singur ABR, de exemplu aria 2 pe fig. 2) şi nu comunică cu exteriorul AS (deci nu conţin ASBR) se pot declara *arii stub* (*stub* = ciot). Ariile *stub* folosesc rute *default* (0.0.0.0) pentru a comunica destinaţii din afara ariei respective. Aria *N* se declară manual *stub* folosind comanda `area N stub` pe ABR corespunzător, şi aceasta are ca efect ca în aria *N* nu se vor mai primi LSA despre rute externe AS (şi rute sumarizate din alte arii ale aceluiaşi AS, în versiunea numită de Cisco *totally stub*).

Configurări:

A) Configurarea ruterului din interiorul ariei se face conform subpunctului 1.5

B) Configurarea ruterului care leagă două sau mai multe arii între ele (arii aparţinînd aceluiaşi AS) se face configurînd interfeţele să opereze în ariile corespunzătoare.

De exemplu, fie o topologie ca în figura de mai jos (în care Router1 este în aria 0, Router2 este în aria 1, iar Router0 face legătura între aria 0 – serial1, respectiv aria 1 – serial0):

Ruterul 0 este deci un ABR, conectînd între ele ariile 0 şi 1. Putem avea două procese OSPF sau unul singur, specificînd însă aria corespunzătoare pentru care rulează:

```
Router0(config)#router ospf 40
Router0(config-router)#network 172.16.0.0 0.0.255.255 area 1
Router0(config-router)#network 192.168.14.0 0.0.0.255 area 0
```

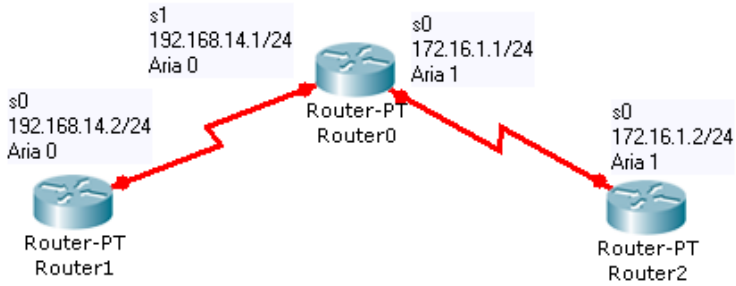

Fig. 3 Exemplu de topologie (Router0 este ABR)

C) Configurarea ruterului care leagă două sau mai multe AS între ele se face împărţind/ redistribuind rutele între AS-urile la care este conectat ruterul. De exemplu, pe aceeaşi topologie din Fig. 4, Router0 leagă între ele AS1 şi AS2; în AS1 rulează RIP, iar în AS 2 ruleaza OSPF (am folosit pentru simplitate RIP, care este tot un protocol de tip IGP)

Rolul ruterului Router0 (care este un ASBR) este de a **redistribui rutele** RIP în AS pe care ruleaza OSPF, şi, respectiv, de a redistribui rutele OSPF în AS pe care rulează RIP.

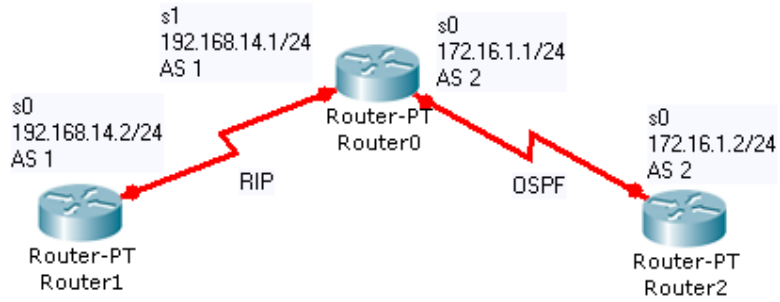

Fig. 4 Exemplu de topologie (Router0 este ASBR)

```
Router0(config)#router ospf 1
Router0(config-router)#network 172.16.0.0 0.0.255.255 area 1
Router0(config-router)#redistribute rip

Router0(config)#router rip
Router0(config-router)#network 192.168.14.0
Router0(config-router)#redistribute ospf 1
```

Cîteva comenzi pentru studiul tabelelor de rutare pentru rutere ABR, respectiv ASBR:

comanda	*descriere*
`show ip ospf border-router`	arată tabela de rutare a unui ruter ABR
`show ip ospf` *`process-id`*	arată informaţii despre toate ariile la care un ruter este conectat şi indica dacă un ruter este ABR, ASBR sau ambele. Numărul de identificare al procesului este un parametru de identificare definit de utilizator la

	activarea rutării OSPF.
`show ip ospf database summary`	oferă informaţie despre starea legăturilor unui ruter ABR
`show ip ospf database asbr-summary`	oferă informaţie despre starea legăturilor unui ruter ASBR
`show ip ospf database external`	arată informaţii despre starea legăturilor externe unui sistem autonom
`show ip ospf database database-summary`	arată informaţii despre baza de date la modul general

1.7 Sumarizare

Reamintim că prin *sumarizare* se înţelege concentrarea mai multor rute/destinaţii într-una singură (*n* rute din interiorul unei anumite zone sînt percepute în afara acelei zone ca o singura rută). Acest lucru presupune o adresare continguă în acea zonă – ca şi în cazul RIPv2, dacă subneturile aceluiaşi net sînt distribuite aleator pe mai multe rutere, nu se poate face sumarizarea.

Tipuri de sumarizare OSPF:

- ***sumarizare externă:***

Sistemele autonome AS (din motive fie ele administrative, economice sau de securitate) nu transmit în totalitate informaţiile de rutare interne către alte sisteme autonome la care sînt conectate prin intermediul unui ruter ASBR, ci doar un "rezumat" (*summary*) al acestora, suficient pentru a informa asupra fiecărei reţele componente ale AS. De fiecare dată cînd e posibil, subreţelele componente unei reţele se combină într-o singură reţea.

Ruterului ASBR îi revine deci sarcina de a realiza sumarizarea rutelor către subreţelele din interiorul unui AS la care e conectat. Se foloseşte comanda:

```
ASBR(config)# router ospf 1
ASBR(config-router)#summary-address address mask
```

unde `address` şi `mask` sînt adresa, respectiv masca super-reţelei

Dacă se combină toate reţelele 10.10.1.0/24, 10.10.2.0/24, …, 10.10.255.0/24 în reţeaua 10.10.0.0/16, comanda devine:

```
ASBR(config-router)#summary-address 10.10.0.0 255.255.0.0
```

- ***sumarizare inter-arie:***

Analog cazului anterior, şi între ariile aceluiaşi sistem autonom se poate utiliza sumarizarea, realizată de data aceasta de un ruter ABR. Comanda este:

```
ABR(config)# router ospf 1
ABR(config-router)#area id range address mask
```

unde `address` şi `mask` sînt adresa, respectiv masca super-reţelei, iar `id` este identificatorul ariei pentru care se realizează sumarizarea. De exemplu, pentru un ABR aflat la graniţa dintre ariile 1 şi 0, în ideea că reţelele din exemplul 2 se află în aria 1, supernetul obţinut prin sumarizare este injectat în aria 0 specificînd că el este un *range* (domeniu de adrese) al ariei 1:

```
ABR(config-router)#area 1 range 10.10.0.0 255.255.0.0
```

Partea 2. Desfăşurare; exerciţii

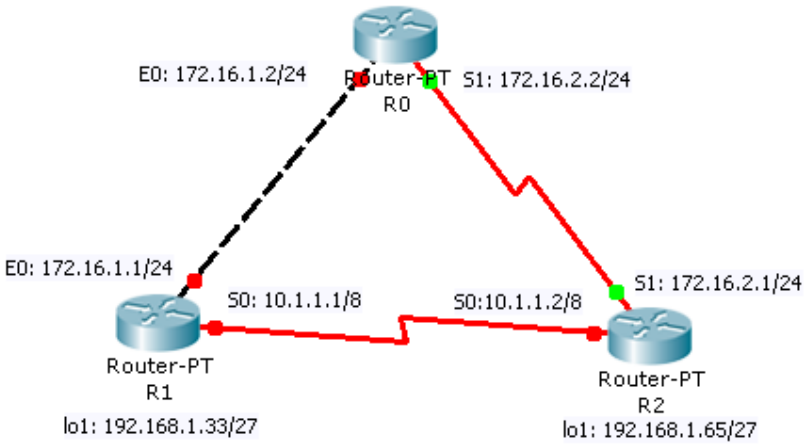

Fig. 5 Topologia de test

1) Se va lucra pe echipe. Se va realiza topologia de test din figură, configurînd pe fiecare ruter:
- *hostname*-ul corespunzator
- adresele IP ale interfeţelor (inclusiv cele 2 loopbacks)
- clock-ul în cazul interfeţelor seriale DCE va fi 64000
- `no shutdown` pe interfeţe
- nu se leagă PC-uri pe ethernet la topologie

Q1. Desenaţi topologia corespunzatoare echipei, notînd la fiecare ruter numele de pe etichetă, numele interfeţelor folosite, precum şi capătul DCE la seriale.

Faza 1: Configurarea OSPF

2) După ce configuraţia e completă, se testează fiecare link cu ping şi, în caz de succes, se salvează configuraţia curentă în NVRAM folosind `copy run start`. Pentru a întrerupe un ping care nu funcţionează se foloseşte secvenţa CTRL-SHIFT-6.

3) Se porneşte un proces de rutare OSPF cu identificatorul 1 pe fiecare ruter (toate ruterele vor fi în *Area 0*) astfel încît sa fie posibil ca orice ruter să poată da ping la interfaţa oricărui alt ruter.

Q2. Scrieti pe desenul de la Q1 reţelele configurate şi *wildcard mask*-urile corespunzătoare.

4) Se verifică pe R1 folosind `sh ip ospf neighbor` starea legăturii lui R1 cu celelate 2 rutere; dacă aceasta nu este FULL, nu s-au format adiacenţe între rutere şi deci procesul OSPF nu a inclus legătura respectivă în tabela de rutare (ceea ce indică o problemă).

Q3. Care este starea celor 2 legături (*state*) ?

Se verifică tabela de rutare cu `sh ip route`, observînd cu ce literă sînt marcate rutele învăţate via OSPF.

Q4. Cît este distanţa administrativă a rutelor OSPF (prima cifră din paranteza dreaptă)? reamintim că la RIP este 120. Dacă exista o rută RIP şi una OSPF, care va fi preferată?
Q5. OSPF face sumarizarea rutelor la limita de clasă, asemenea lui RIP?

5) Se verifică *router id* (identificatorul fiecarui ruter, din punct de vedere al OSPF) folosind `sh ip protocols` şi apoi `sh ip ospf`.

Q6. Care este *router id* pentru fiecare ruter? pe ce criteriu s-a ales?

Se verifică rezultatul alegerilor pe reţele de tip *broadcast* folosind cele 2 comenzi show precedente, precum şi `show ip ospf neighbor` şi `show ip ospf interface` *`interfaţa`*. Aceasta se face pentru că rezultatul alegerilor este diferit pe fiecare reţea în care participă ruterele, şi doar unele din reţele sînt de tip *broadcast*.

Q7. Care dintre reţelele prezente este de tip *broadcast*?
Q8. Cine este cîştigătorul alegerilor (DR) ? pe ce criteriu s-a ales?

Faza 2: Modificarea costului rutelor

6) Reamintim că metrica la OSPF este dată de bandă, în timp ce la RIP este dată de numărul de hopuri. Metrica este asociată conceptului de *cost*, care este cel mai mic pentru ruta preferată. Exprimînd banda în *bps*, OSPF calculează costul cu formula:

$$cost = 10^8 / banda$$

Astfel costul e mai mic pentru legăturile de viteză mai mare, ceea ce e intuitiv.

Q9. Aflaţi banda legăturilor seriale folosind `show interface` *`serialN`*, precum şi banda legăturii ethernet/fast ethernet disponibilă.
Q10. Conform formulei, cît este costul legăturilor seriale, *Fast Ethernet*, *Gigabit Ethernet* şi *10 Gigabit Ethernet*?
Q11. Cît este costul fiecarei rute din tabela de rutare a lui R1? explicaţi valoarea indicată (a doua cifră din paranteza dreaptă), ştiind că costul este cumulativ.

7) Vom scădea banda legăturii dintre R0 şi R1 pentru a fi inferioară benzii unei interfeţe seriale. Banda unei interfeţe se configurează cu comanda `bandwidth` *`banda_în_kbps`* şi trebuie configurată la ambele capete ale legăturii. Dacă de exemplu banda unei interfeţe seriale era 1544kbps, vom scădea banda pe Ethernet la o valoare de minim 4 ori mai mică decît aceasta, de exemplu (banda se specifică în kbps):

```
R1(config)# int E0
R1(config-if)# bandwidth 386
```

După verificarea benzii cu `sh int …`, vizualizaţi noua tabelă de rutare. Verificaţi cu `traceroute` pe ce cale se ajunge de la R1 la R0/S1.

Q12. Cît este noul cost al rutelor din tabela lui R1? Justificaţi.
Q13. Care este ruta aleasă de OSPF pentru a ajunge de pe R1 pe R0/S1 ? Cît este numărul de hopuri pînă la această destinaţie?

Observaţie: nu alegeţi să testaţi o destinaţie direct conectată, de exemplu de la R1 la R0/E0; pentru destinaţiile direct conectate, se foloseşte calea directă şi OSPF nu intră în discuţie, deci modificările de bandă şi cost nu au efect.

Q14. Care ar fi fost ruta aleasă de RIP pentru a ajunge de pe R1 pe R0/S1, ştiind că acesta nu ţine seama de bandă ?

8) Costul se poate modifica şi manual, în cazurile în care preferăm o anumită cale din anumite motive, sau cînd folosim rutere de la fabricanţi diferiţi, care nu calculează costul cu aceeasi formulă ca Cisco, sau cînd avem legături *Gigabit Ethernet* sau *10 Gigabit Ethernet*.

Pentru modificarea costului la nivel de interfaţă, comanda este:

```
R1(config)# int interfaţa
R1(config-if)# ip ospf cost NNN
```

Modificaţi ambele capete ale linkului R1-R2 astfel încît costul sa fie 10. Vizualizaţi noua tabelă de rutare.

Q15. Care este noua rută aleasă de OSPF pentru a ajunge de pe R1 pe R0/S1? s-a modificat faţă de punctul precedent?

Faza 3: Configurarea OSPF cu mai multe arii

9) Pe fiecare ruter, ştergeţi configuraţia protocolului OSPF folosind `no router ospf N`. Opriţi legătura serială dintre R1 şi R2, dînd `shutdown` celor 2 interfeţe seriale (cablul nu se scoate).

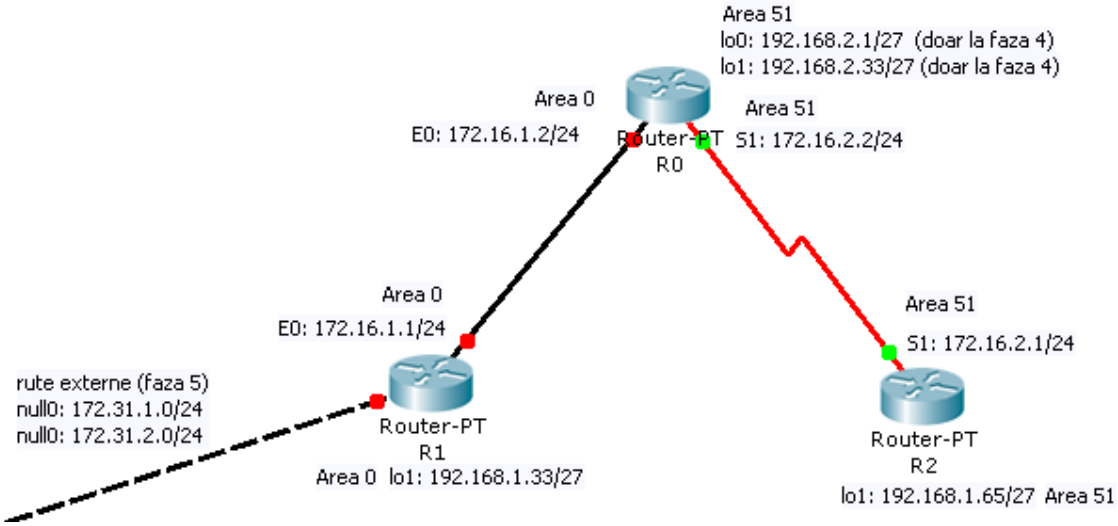

Fig. 6 Modificarea topologiei de test

Configuraţi din nou protocolul OSPF (fig. 6) pentru:

- reţeaua 172.16.1.0/24, ruterele R0 şi R1, precum şi lo0 de pe R1, aria 0;
- reţeaua 172.16.2.0/24, ruterele R0 şi R2, precum şi lo0 de pe R2, aria 51;

Se va folosi ***acelaşi*** identificator de proces pe R0 pentru ambele arii.
Vizualizaţi OSPF pe fiecare ruter folosind `sh ip ospf`

Q16. Ce fel de rutere sînt R0, R1 respectiv R2, conform clasificării din fig. 2 ?

Q17. Pe cele trei rutere vizualizaţi tabela de rutare; apare ceva nou faţă de cazul anterior (OSPF cu o singură arie)? Ce prefixe (litere) au reţelele învăţate prin rutare dinamică?

10) Observaţi în tabela de pe R0 masca rutelor către cele 2 interfeţe loopback.

Q18. Cît este această mască ? ce adresă apare în cazul fiecărei interfeţe?

Se observă că OSPF comunică rutele către interfeţele loopback ca "host route" şi nu "network route". Masca unui host este cea observată.

Porniţi *debugging*-ul folosind `debug ip ospf events` pe R1.

Q19. De la ce rutere (şi din ce arii) vedeţi mesaje de informare? explicaţi.

Opriţi *debugging*-ul folosind `undebug all`

Faza 4: Sumarizarea inter-arie

11) Pe ruterul R0 configuraţi 2 interfeţe loopback lo0 şi lo1 aparţinînd reţelelor 192.168.2.0/27 şi 192.168.2.32/27. Introduceţi aceste reţele în procesul ospf de pe R0, în aria 51.

Verificaţi cu ping de pe R1 şi R2 că sînt accesibile aceste 2 noi loopback-uri.

Q20. Verificaţi existenţa celor 2 reţele în tabela lui R1 şi scrieţi cum apar.

Se doreşte ca R0 să realizeze sumarizarea inter-arie (să distribuie un rezumat al celor 2 reţele din aria 51 în aria 0). Se folosesc comenzile:

```
R0(config)# router ospf 1
R0(config-router)#area 51 range address mask
```

Atenţie! în poziţia *mask* se foloseşte masca obişnuită (netmask) şi nu wildcard mask !

Q21. Ce adresă şi mască aţi folosit pentru a cuprinde exact cele 2 reţele ale loopback-urilor ?
Q22. Verificaţi tabelele de rutare pe R0 (cel care trimite) şi R1 (cel care primeşte); notaţi diferenţele faţă de pînă acum.

Observaţie: pe R0 ruta de tip *summary* apare conectată la o interfaţă numită **Null0**, care nu există. De fapt, ruterul crează această interfaţă "virtuală" pentru a rezolva problema inexistenţei unei interfeţe propriu-zise care să "cuprindă" împreună interfeţele lo0 şi lo1.

Faza 5: Redistribuirea şi sumarizarea externă a rutelor

12) Mai întîi, presupunem că pe R1 (vezi figura 6) există o rută către reţeaua externă 172.31.1.0. Cum nu avem o interfaţă căreia să i-o ataşăm, vom folosi o interfaţă virtuală Null0 şi vom defini o rută statică care "iese" prin acea interfaţă:

```
R1(config)# ip route 172.31.1.0 255.255.255.0 Null0
```

Q23. Verificaţi în tabela de rutare pe R1 cu ce literă (prefix) apare ruta respectivă.
În continuare, vom importa ruta în OSPF:

```
R1(config)# router ospf 1
R1(config-router)# redistribute static subnets
```

Observaţie: fără cuvîntul cheie *subnets* ruterul dă un mesaj de genul:

```
R1(config-router)# redistribute static
% Only classful networks will be redistributed
```

ceea ce se întîmplă pentru că ruta noastră nu corespunde clasei.

Q24. Verificaţi cu `show ip ospf` pe R1 de ce tip apare acum ruterul, după ce aţi făcut redistribuirea (conform clasificării din fig. 2)
Q25. Verificaţi în tabelele de rutare de pe R0 şi R2 cu ce literă (prefix) apare ruta respectivă.

13) Presupunem acum că se mai adaugă şi alte rute externe pe R1. Mai definim o rută prin aceeaşi interfaţă:

```
R1(config)# ip route 172.31.2.0 255.255.255.0 Null0
```

Q26. Au apărut ambele rute în tabelele de rutare de pe R0 şi R2 ? cu ce literă (prefix)?

Întrucît prin continuarea acestui proces vom ajunge să creştem inutil tabela de rutare "exportată" de către OSPF, vom face o sumarizare externă pe ASBR:

```
R1(config)# router ospf 1
R1(config-router)# summary-address adresă mască
```

Q27. Ce adresă şi mască aţi folosit ?
Q28. Ce s-a modificat în tabelele de rutare de pe R0 şi R2 ? cîte rute apar de data aceasta ?

Faza 6: Instalarea şi testarea ACL extinse

14) Scrieţi şi testaţi ACL-uri extinse care interzic următoarele tipuri de trafic, şi specificaţi pe ce ruter şi interfaţă le plasaţi. *Atenţie!* tot restul traficului trebuie să fie permis!
Restricţiile se aplică şi se testează ***pe rînd.*** Pentru testare poate fi necesar *extended ping* şi/sau specificarea adresei sursă la telnet. Înaintea testării următorului ACL, se şterge (sau se suprascrie) ACL-ul curent. Pentru a nu complica inutil exerciţiul, se va alege o singură adresă destinaţie atunci cînd se interzice accesul pe un ruter (care poate avea 2-3 interfeţe).

Q29. interziceţi ping de pe R1/lo1 pe R0
Q30. interziceti telnet de pe R2 pe R1
Q31. interziceti telnet de pe R1/lo1 pe toate ruterele din topologie
Q32. interziceti telnet de pe R1 pe R2, fără a interzice şi invers. Cum faceţi această separare ?
Q33. interziceti ping de pe adrese *pare* din reţeaua 172.16.1.0/24 pe R2

15) Folosind `erase startup-config` se şterge configuraţia fiecărui ruter.