

Rutare dinamică

O rută statică este exactă, previzibilă și nu se schimbă niciodată - inclusiv atunci când legătura pe care o descrie a căzut acum zece minute. Într-o rețea cu trei rutere se poate trăi așa. Într-una cu douăzeci, fiecare cu câte trei ieșiri, numărul de linii de configurat depășește repede ce poate ține un om în minte, iar fiecare cablu scos cere intervenție manuală. Cursul acesta prezintă protocoalele care întrețin singure tabela de rutare: cum își descoperă ruterele vecinii, cum își construiesc o hartă comună a rețelei și cum calculează, fiecare pentru sine, drumurile cele mai scurte. Accentul cade pe familia link-state și pe reprezentantul ei standard, OSPF.

Din capitolul precedent.

- O **rută** este tripletul destinație, direcție, distanță.
- **Distanța administrativă** spune cât de multă încredere are ruterul în **sursa** rutei.
- **Metrica** spune cât de bună este ruta, în interiorul aceleiași surse.
- Decizia se ia prin **longest prefix match**: prefixul mai lung câștigă, indiferent de AD.
- O rută statică are AD **1**; una direct conectată, **0**.

OBIECTIVELE CAPITOLULUI

- Să explicați de ce rutele statice nu scalează și ce anume rezolvă un protocol dinamic
- Să distingeți între distance-vector, link-state și path-vector
- Să descrieți problema buclilor la distance-vector și mecanismele care o atenuează
- Să enumerați cei patru pași ai unui protocol link-state
- Să aplicați Dijkstra pe o topologie mică și să obțineți arborele drumurilor minime
- Să calculați costul OSPF al unei legături și să știți când formula dă rezultate greșite
- Să explicați de ce se alege DR și BDR și cum decurge alegerea
- Să configurați OSPF single-area și să depanați o adiacență care nu se formează

Astăzi umplem tabela altfel. Sursele noi de rute - RIP, OSPF, EIGRP - au fiecare propria distanță administrativă și propria metrică, iar ruterul trebuie să le compare pe toate. Ordinea comparării este exact cea din cursul 6, și e bine să fie proaspătă în minte.

10.1 De ce nu ne ajung rutele statice

Aritmetica administratorului. Într-o rețea cu n rutere, fiecare trebuie să știe drumul spre rețelele celorlalți. Dacă fiecare ruter are câte două rețele locale, sunt aproximativ $2n$ destinații, iar fiecare dintre cele n rutere are nevoie de o linie pentru fiecare destinație care nu e a lui.

Pentru 5 rutere: circa 40 de linii. Pentru 20 de rutere: circa **760**. Scrise de mână, în ambele sensuri, fără greșeli.

Iar la fiecare rețea nouă adăugată undeva, trebuie modificate **toate** ruterele.



Figura 10.1 Ce se întâmplă când cade o legătură.

Tabelul 10.1 De ce nu ne ajung rutele statice.

	Rutare statică	Rutare dinamică
Efort de configurare	crește cu pătratul dimensiunii	câteva linii per ruter, indiferent de mărime
Reacție la defecte	manuală	automată, în secunde
Consum de resurse	zero	procesor, memorie, bandă
Previzibilitate	totală	depinde de starea rețelei
Securitate	nimic de falsificat	anunțurile pot fi falsificate; e nevoie de autentificare
Unde se folosește	rețele mici, rute implicite, legături de rezervă	orice rețea de dimensiune medie în sus

Cele două nu se exclud. Chiar și într-o rețea complet OSPF, ruta implicită spre furnizor este aproape întotdeauna statică - pentru că nu are cu cine să fie negociată.

10.2 Familiile de protocoale

Prima împărțire ține de **unde** se folosesc: în interiorul unei organizații, sau între organizații.

Tabelul 10.2 Familiile de protocoale.

Categorie	Unde	Protocoale
IGP - Interior Gateway Protocol	în interiorul unui sistem autonom	RIP, EIGRP, OSPF, IS-IS
EGP - Exterior Gateway Protocol	între sisteme autonome	BGP

DEFINIȚIA 10.1 (SISTEM AUTONOM (AS)). Un grup de rețele aflate sub o administrare unică, cu o politică de rutare comună. Fiecare AS are un număr unic la nivel mondial. Internetul este, la scară mare, o rețea de aproximativ 75 000 de sisteme autonome care își anunță reciproc rutele prin BGP.

A doua împărțire, cea care ne interesează astăzi, ține de **ce știe** fiecare ruter:

Tabelul 10.3 Familiile de protocoale.

Familie	Ce știe un ruter	Protocoale
Distance-vector	doar direcția și distanța, aflate de la vecini	RIP, IGRP, EIGRP (hibrid)
Link-state	graful complet al topologiei	OSPF, IS-IS
Path-vector	lista completă de sisteme autonome traversate	BGP

Analogie Un ruter **distance-vector** este ca un turist care întreabă trecătorii: „cât mai e până la gară?”. Primește un număr și o direcție, are încredere și pleacă. Nu vede orașul.

Un ruter **link-state** are harta orașului în buzunar. Fiecare stradă închisă îi este comunicată, el o taie de pe hartă și își recalculează singur traseul.

Un ruter **path-vector** primește itinerarul complet: „prin Sibiu, apoi Cluj, apoi Oradea”. Poate refuza un drum pentru că trece printr-un oraș care nu îi place - și asta e chiar ideea BGP-ului. **Distance-vector**

- Cunosce doar vectorul spre destinație
- Trimite **toată** tabela de rutare, periodic
- Converge lent

- Consumă puține resurse
- Ușor de configurat
- Vulnerabile la bucle de rutare

Link-state.

- Cunosce graful întregii topologii
- Trimite doar **ce s-a schimbat**, când s-a schimbat
- Converge rapid
- Consumă memorie și procesor
- Mai greu de configurat și de proiectat
- Imune la buclele clasice, prin construcție



Figura 10.2 Protocol și caracteristică.

10.3 Distance-vector și problema buclelor

Un protocol distance-vector, precum RIP, funcționează pe un principiu simplu: fiecare ruter trimite periodic vecinilor **întreaga sa tabelă de rutare**, iar aceștia adaugă costul legăturii și decid. Nimeni nu vede topologia; fiecare crede ce i se spune. De aici și poezia care i s-a lipit definitiv: **routing by rumour** - rutare din zvonuri.

Tabelul 10.4 Distance-vector și problema buclelor.

Limitare	Ce înseamnă practic
Scalabilitate	RIP consideră inaccesibilă orice destinație aflată la peste 15 salturi - o limită care într-o rețea mare devine reală
Convergență lentă	o schimbare se propagă din vecin în vecin, la fiecare interval de actualizare; pentru un diametru de 3 salturi și interval K, propagarea durează 3K
Vedere locală	ruterul nu poate detecta o buclă pe care nu o vede
Metrică simplistă	numărul de salturi ignoră complet lățimea de bandă: o cale prin trei legături de fibră pare mai proastă decât una printr-un singur modem

Numărătoarea spre infinit



Figura 10.3 Numărătoarea spre infinit.

10.4 Cum lucrează un protocol link-state

Ideea centrală este că fiecare ruter nu află **rutele** de la vecini, ci află **starea legăturilor** din întreaga rețea, își construiește harta completă și calculează singur drumurile. Patru elemente fac asta posibil.

Adiacențele și protocolul Hello

Doi vecini direct conectați formează o **adiacență**. Relația se stabilește și se întreține prin mesaje Hello periodice, care au dublu rol: **keep-alive** - dacă mesajele încetează, vecinul e considerat căzut - și sincronizarea parametrilor. Vecinii cunoscute se păstrează într-o **tabelă de vecini**.

```
ID_R2 - Fa0/1 - 00:13 - FULL
ID_R3 - Fa0/2 - 00:00 - LOADING
!      interfata timp de la ultimul hello starea adiacentei
```

Listingul 10.1 tabela de vecini a lui R1

De ce Hello-ul rezolvă problema din secțiunea 2. O rută statică nu are cum să afle că ruterul de la celălalt capăt a murit, atâta timp cât cablul e în priză și interfața e **up**.

Un protocol dinamic nu se uită la cablu, ci la **conversație**. Dacă timp de 40 de secunde (temporizatorul *dead*) nu sosește niciun Hello, vecinul e declarat pierdut - indiferent ce spune starea interfeței.

Este diferența dintre „firul e conectat” și „mai e ceva acolo”.

LSP: unitatea de informație

Un **LSP** (Link State Packet) descrie legăturile unui ruter: pentru fiecare interfață, rețeaua atașată, tipul legăturii, costul, starea ei și vecinul de la celălalt capăt. LSP-urile se trimit:

- la **inițializarea** procesului de rutare;
- imediat ce apare o **schimbare** în topologie - mecanismul de *triggered update*, sursa convergenței rapide;
- **periodic**, la interval mare (în OSPF, o dată la 30 de minute), ca reîmprospătare.

Imediat ce un ruter primește un LSP nou, îl retransmite tuturor celorlalți vecini. Procesul se numește **flooding** și asigură că, în câteva zecimi de secundă, toată rețeaua are aceeași informație.

Interfețe care oscilează. O interfață care se activează și se dezactivează repetat (*flapping*) ar declanșa recalculări continue ale topologiei, consumând tot procesorul tuturor rutelor din arie. De aceea orice protocol link-state așteaptă un scurt interval de stabilizare înainte de a recalcula - și de aceea o interfață instabilă e mai periculoasă decât una căzută de tot.

Cei patru pași

1. **Adiacențe și rețele conectate.** Ruterul își descoperă vecinii și rețelele direct conectate.
2. **Flooding de LSP-uri.** Fiecare ruter anunță ce știe și retransmite ce primește.
3. **Popularea tabelii de topologie.** Toate rețelele primite se păstrează, cu costurile lor - nu doar cele mai bune. Aceasta este harta.
4. **Dijkstra.** Se rulează algoritmul drumurilor minime, cu ruterul curent în vârf. Doar rezultatele ajung în tabela de rutare.



Figura 10.4 Puneți în ordine cei șapte pași.

Trei tabele, nu una. Un ruter link-state ține **tabela de vecini** (cine e lângă mine și în ce stare), **tabela de topologie** (harta întregii arii, identică pe toate ruterele) și **tabela de rutare** (rezultatul lui Dijkstra, diferit pe fiecare ruter).

Dacă tabelele de topologie a două rutere din aceeași arie **diferă**, ceva e stricat - de regulă o adiacență care nu a ajuns niciodată la starea Full.

10.5 Algoritmul lui Dijkstra

Ideea, în trei propoziții. Pornim din nodul propriu, cu distanța 0. Toate celelalte au, provizoriu, distanța infinit.

La fiecare pas alegem nodul **nevizitat cu distanța cea mai mică** și îl declarăm definitiv - de acolo nu se mai poate ajunge mai ieftin. Apoi, prin el, încercăm să îmbunătățim distanțele vecinilor lui.

Repetăm până se termină nodurile. Rezultatul este un **arbore** al drumurilor minime, cu noi în rădăcină.



Figura 10.5 Algoritmul lui Dijkstra pe o topologie de șapte rutere.

Alegeți ruterul care calculează și observați că arborele obținut este **diferit pentru fiecare**: fiecare ruter se pune pe sine în vârf. Harta e aceeași; arborele nu.

Apoi modificați costul unei legături din arbore și urmăriți cum traficul se mută pe altă cale - exact ce se întâmplă într-o rețea reală când un administrator ajustează banda declarată a unei interfețe. Este principalul instrument de inginerie a traficului în OSPF: nu se schimbă cablurile, se schimbă costurile.

Exercițiu rezolvat 10.1. Un ruter are două căi spre aceeași rețea: una prin trei legături de cost 1 fiecare, alta printr-o singură legătură de cost 4. Care câștigă și de ce nu numărul de salturi?

Rezolvare. Prima cale: $1 + 1 + 1 = 3$. A doua cale: 4. Câștigă prima, deși are de trei ori mai multe salturi.

Costul OSPF vine din **lățimea de bandă**. Cost 1 înseamnă 100 Mbps sau mai mult; cost 4 înseamnă 25 Mbps. Deci: trei legături rapide bat una lentă.

Un protocol distance-vector clasic, care numără salturile, ar fi ales exact invers - și ar fi trimis tot traficul pe legătura de 25 Mbps. Este, în două rânduri, motivul pentru care metrica bazată pe bandă a înlocuit numărătoarea de salturi.

10.6 OSPF: caracteristici de bază

OSPF (Open Shortest Path First) a fost dezvoltat de IETF începând cu 1988, ca alternativă **deschisă** la protocoalele proprietate ale vremii. OSPFv2 (RFC 2328) este varianta pentru IPv4, OSPFv3 (RFC 5340) cea pentru IPv6.

Tabelul 10.5 OSPF: caracteristici de bază.

Caracteristică	Valoare
Transport	direct peste IP, protocol 89 - nu folosește TCP sau UDP
Siguranța transmisiei	mecanism propriu de confirmare (LSAck)
Distanță administrativă	110
Adrese multicast	224.0.0.5 (toate ruterele OSPF), 224.0.0.6 (DR și BDR)
Metrică	$\text{cost} = 10^8 / \text{lățimea de bandă în bps}$
Timere hello / dead	10 s / 40 s pe rețele multiacces și punct-la-punct; 30 s / 120 s pe NBMA
Autentificare	opțională, cu parolă simplă sau MD5 / SHA



Figura 10.6 Costul OSPF al unei legături.

Problema formulei implicite. Cu referința implicită de 10^8 (adică 100 Mbps), orice legătură de 100 Mbps **sau mai rapidă** primește costul 1, pentru că minimul este 1.

O legătură Fast Ethernet și una de 10 Gbps devin astfel **echivalente**, iar OSPF nu mai poate distinge între ele. Selectați în calculatorul de mai sus Gigabit cu referință implicită și veți vedea: cost 1. Selectați apoi referința de 10000 și diferența reapare.

Într-o rețea modernă, referința se ajustează cu `auto-cost reference-bandwidth 10000` - **aceeași valoare pe toate ruterele din domeniu**, altfel costurile devin incoerente și rutarea imprevizibilă.

Tabelul 10.6 OSPF: caracteristici de bază.

Mediu	Cost cu referința implicită
Serial 56 kbps	1785
T1 (1,544 Mbps)	64
Ethernet 10 Mbps	10
Fast Ethernet și mai rapid	1

10.7 Mesajele și stările adiacenței

Tabelul 10.7 Mesajele și stările adiacenței.

Mesaj	Rol
Hello	descoperă vecinii și menține adiacențele
DBD - Database Description	listă abreviată cu ce conține baza de date link-state - un fel de cuprins
LSR - Link-State Request	cere detalii despre o intrare văzută în DBD și lipsă local
LSU - Link-State Update	răspunsul la LSR; conține unul sau mai multe LSA-uri
LSAck	confirmă primirea unui LSU

De ce nu se trimite direct toată baza de date. Ar fi risipă: cele două rutere au, de regulă, aproape aceleași informații.

De aceea schimbul e în doi timpi: întâi **cuprinsul** (DBD), apoi fiecare cere doar **capitolele care îi lipsesc** (LSR) și le primește (LSU). Este exact modelul unei sincronizări eficiente de fișiere.

Tabelul 10.8 Mesajele și stările adiacenței.

Stare	Ce s-a întâmplat
Down	nu s-a primit niciun Hello
Init	ruterul a primit un Hello, dar nu se regăsește în el
Two-Way	a primit un Hello în care se regăsește propriul Router ID - vecinul îl vede. Pe rețele multiacces, aici se aleg DR și BDR
Ex-Start	se stabilește cine este master și cine slave pentru schimbul de baze de date
Exchange	se schimbă DBD-urile
Loading	se cer și se primesc detaliile lipsă (LSR, LSU)
Full	stare finală: bazele de date link-state sunt identice

**Figura 10.7** Stările unei adiacențe, în ordine.

Depanarea după starea în care s-a blocat. Adiacența se formează doar dacă toți parametrii critici coincid: numărul **ariei**, intervalele **hello** și **dead**, **MTU**-ul interfeței, bitul de stub și autentificarea - iar Router ID-urile trebuie să fie **diferite**.

Blocată în **Init** → Hello-urile nu ajung în ambele sensuri: ACL, VLAN greșit, sau `passive-interface` pus din greșeală.

Blocată în **Two-Way** între două DROTHER-e → **normal**, nu e o problemă: pe un segment multiacces, două rutere obișnuite nu formează adiacență completă.

Blocată în **Exchange** sau **Ex-Start** → aproape sigur **MTU diferit** între capete.

Oscilează între Full și Down → legătură instabilă sau temporizatoare nepotrivite.

10.8 DR și BDR pe rețele multiacces

Pe o legătură punct-la-punct există o singură adiacență posibilă și nicio problemă. Pe un segment Ethernet cu n rutere, dacă fiecare ar forma adiacență cu fiecare, ar rezulta $n(n-1)/2$ relații - 45 pentru zece rutere, fiecare cu propriile mesaje Hello, propriile sincronizări și propriile retransmiteri.

Soluția este centralizarea: se alege un **DR** (Designated Router), care primește și redistribuie actualizările, și un **BDR** (Backup DR), care preia dacă DR cade. Toate celelalte rutere (**DROTHER**) formează adiacență doar cu DR și BDR. Numărul de adiacențe scade la $2(n-1)$.

Tabelul 10.9 DR și BDR pe rețele multiacces.

Rutare pe segment	Fără DR	Cu DR și BDR	Economie
5	10	8	20%
10	45	18	60%
20	190	38	80%
50	1225	98	92%

Criteriile de alegere

1. **Prioritatea cea mai mare** de pe interfață (implicit 1). Valoarea **0** exclude ruterul din alegere.
2. La egalitate, **Router ID-ul cel mai mare**.

Router ID-ul se determină, la rândul lui, în trei pași, în ordine strictă:

1. valoarea configurată manual cu `router-id`;
2. cea mai mare adresă IP de pe o interfață de **loopback**;
3. cea mai mare adresă IP de pe o interfață fizică **activă**.

Alegerea nu este preemptivă. Un ruter nou, chiar cu prioritate mai mare, **nu** preia rolul de DR. Dacă DR cade, BDR îi ia locul și se alege un nou BDR dintre DROTHER-e.

Consecință practică: dacă vrei ca un anumit ruter să fie DR, porniți-l primul sau forțați o realegere cu `clear ip ospf process`.

Consecință și mai practică: folosiți **interfețe de loopback** pentru Router ID. Ele nu cad niciodată, deci identitatea ruterului nu se schimbă când pică un cablu.

```
! ruterul care trebuie sa fie DR
R1(config)# interface gigabitEthernet 0/0
R1(config-if)# ip ospf priority 100

! ruterul care nu trebuie sa fie niciodata DR sau BDR
R3(config)# interface gigabitEthernet 0/0
R3(config-if)# ip ospf priority 0

! Router ID stabil, printr-o interfata care nu cade
R1(config)# interface loopback 0
R1(config-if)# ip address 1.1.1.1 255.255.255.255
R1(config)# router ospf 1
R1(config-router)# router-id 1.1.1.1

! alegerea nu e preemptiva - trebuie fortata
R1# clear ip ospf process
```

Listingul 10.2 influențarea alegerii copiază

10.9 OSPF pe mai multe arii

Dijkstra rulează cu atât mai lent cu cât graful e mai mare, iar orice schimbare într-un colț al rețelei obligă **toate** ruterele să recalculeze totul. Soluția de scalare este împărțirea domeniului OSPF în **arii**: fiecare arie rulează Dijkstra pe propriul graf, iar pentru destinațiile din afara ei doar adună distanța până la ruterul de frontieră.

Regula de aur a ariilor. Toate ariile trebuie să aibă conectivitate la **aria 0** (backbone). Traficul dintre două arii non-zero trece **obligatoriu** prin backbone.

Când o arie nu poate fi conectată fizic la aria 0, se configurează o legătură virtuală - sau, mai simplu și mai onest, un tunel GRE, ca în cursul trecut.

Tabelul 10.10 OSPF pe mai multe arii.

Tip de ruter	Poziție
Internal router	toate interfețele într-o singură arie
Backbone router	ruter cu cel puțin o interfață în aria 0
ABR - Area Border Router	face legătura între două arii; are sincronizate bazele de date ale ambelor
ASBR - Autonomous System Border Router	introduce în OSPF rute externe, de exemplu prin redistribuire din alt protocol sau o rută implicită

Tipurile de LSA

Tabelul 10.11 Tipurile de LSA.

Tip	Denumire	Generat de	Ce descrie
1	Router LSA	fiecare ruter	starea propriilor legături, în interiorul ariei
2	Network LSA	DR	ruterele cu care DR are adiacență pe acel segment
3	Summary LSA	ABR	rutele dintr-o arie, anunțate celorlalte arii
4	ASBR Summary LSA	ABR	cum se ajunge la un ASBR
5	External LSA	ASBR	rute externe domeniului OSPF
7	NSSA External	ASBR dintr-o arie NSSA	rute externe într-o arie care nu acceptă tipul 5

Codurile din tabela de rutare

Tabelul 10.12 Codurile din tabela de rutare.

Cod	Provine din	Semnificație
O	LSA 1 și 2	rută din aceeași arie
O IA	LSA 3	rută inter-arie
O E1 / O E2	LSA 5	rute externe; E1 acumulează costul intern, E2 păstrează cost fix (implicit 20)
O N1 / O N2	LSA 7	rute externe într-o arie NSSA

Când există atât o rută E1, cât și una E2 spre aceeași destinație, se preferă **E1**: fiind cumulativă, reflectă mai fidel costul real al drumului. E2 este implicit tocmai pentru că e mai simplu - dar e și mai puțin exact.

10.10 Configurare și verificare

```
R1(config)# router ospf 1
R1(config-router)# router-id 1.1.1.1

! ce interfete participa - masca este WILDCARD, ca la ACL-uri
R1(config-router)# network 192.168.10.0 0.0.0.255 area 0
R1(config-router)# network 10.0.0.0 0.0.0.3 area 0

! interfata spre statii: anunta reteaua, dar nu trimite Hello
R1(config-router)# passive-interface gigabitEthernet 0/0

! ajustarea referintei - ACEEASI valoare pe toate ruterele
R1(config-router)# auto-cost reference-bandwidth 10000

! ruta implicita spre furnizor, anuntata in tot domeniul OSPF
R1(config)# ip route 0.0.0.0 0.0.0.0 203.0.113.1
R1(config-router)# default-information originate
```

Listingul 10.3 OSPFv2 într-o singură arie copiază

Numărul de proces este local. router ospf 1 pe R1 și router ospf 55 pe R2 formează perfect adiacență. Numărul identifică procesul **pe acel ruter**, nu domeniul de rutare. Ce trebuie să coincidă este numărul **ariei**.

passive-interface se pune pe orice interfață pe care nu există alt ruter: rețeaua rămâne anunțată în OSPF, dar nu se mai trimit mesaje Hello inutile spre stații - ceea ce economisește bandă și închide o ușă de atac.



Figura 10.8 Terminal: verificarea OSPF.

Priviți prima comandă: a treia linie arată 2WAY/DROTHER cu prioritate 0. Nu este o eroare - este exact ce trebuie să se întâmple între două rutere obișnuite pe un segment multiacces. Doar adiacențele cu DR și BDR ajung la FULL.

Iar în tabela de rutare se citesc, dintr-o privire, cele trei origini: ○ din aceeași arie cu cost 2, ○ IA din altă arie cu cost 12, și ○ E2 externă, cu costul fix 20 care nu se schimbă oricât ar fi drumul intern.

10.11 Greșeli frecvente

Greșeli frecvente.

„Adiacența e blocată în Exchange”

Cele două capete au MTU diferit și refuză să sincronizeze bazele de date.

show ip ospf interface pe ambele, comparați MTU-ul. Sau ip ospf mtu-ignore , ca soluție de avarie.

„Vecinii sunt în Two-Way și nu trec mai departe”

Dacă sunt două DROTHER-e pe un segment multiacces, este comportamentul corect .

Verificați cine e DR. Adiacențele complete există doar cu DR și BDR.

„Am pus reference-bandwidth doar pe un ruter”

Costurile devin incoerente: același drum are valori diferite văzut din capete diferite.

Aceeași valoare pe toate ruterele din domeniu. Fără excepție.

„Am folosit masca de rețea în comanda network”

OSPF cere wildcard , exact ca ACL-urile.

network 10.0.0.0 0.0.0.3 area 0 , nu 255.255.255.252 .

„Am uitat passive-interface pe segmentele cu stații”

Ruterul trimite Hello-uri spre utilizatori: bandă irosită și o ușă de atac deschisă - oricine poate porni un proces OSPF fals.

passive-interface default , apoi no passive-interface doar unde chiar există vecini.

„Ruterul nou nu devine DR, deși are prioritate 200”

Alegerea nu este preemptivă. Nu e o defecțiune.

clear ip ospf process , dacă chiar aveți nevoie de realegere.

„Am configurat OSPF, dar ruta implicită nu ajunge la ceilalți”

O rută statică nu intră singură în OSPF.

default-information originate pe ruterul de margine.

10.12 Rezumat și glosar

Ce trebuie să rămână.

- Rutele statice nu **scalează** și nu **reacționează**. Ambele probleme le rezolvă un protocol dinamic.
- **Distance-vector** = zvonuri de la vecini; **link-state** = hartă proprie.
- Numărătoarea spre infinit e problema fundamentală a DV; split horizon și poison reverse o atenuează.
- Cei patru pași link-state: adiacențe → flooding → topologie → Dijkstra.
- Un ruter link-state ține **trei** tabele: vecini, topologie, rutare.
- OSPF: protocol 89, AD **110**, cost = 10^8 / bandă, multicast 224.0.0.5.
- Adiacența trece prin Init → Two-Way → Ex-Start → Exchange → Loading → **Full**.
- **DR** și **BDR** reduc adiacențele de la $n(n-1)/2$ la $2(n-1)$; alegerea **nu** e preemptivă.
- Toate ariile se conectează la **aria 0**.
- În comanda `network`, masca este **wildcard**.

Exerciții

1. Ce pastreaza un ruter link-state in tabela de topologie?
 - a. Doar cea mai buna ruta spre fiecare destinatie
 - b. Toate rutele primite prin LSP-uri, cu costurile lor
 - c. Doar rutele statice
 - d. Adresele MAC ale vecinilor
2. De ce limita RIP este de 15 salturi?
 - a. Pentru ca antetul are doar 4 biti pentru metrica
 - b. Pentru ca 16 inseamna infinit, iar limita opreste numaratoarea spre infinit intr-un timp rezonabil
 - c. Pentru ca ruterele nu pot stoca mai mult
 - d. Este o alegere arbitrara

3. Ce face split horizon?
 - a. Imparte rețeaua în două jumătăți
 - b. Nu anunță o rută înapoi pe interfața de la care a fost aflată
 - c. Criptează actualizările de rutare
 - d. Reduce numărul de adiacente
4. De ce se alege un DR pe un segment Ethernet cu mai multe rutări OSPF?
 - a. Pentru că doar el poate ruta
 - b. Ca să reducă numărul de adiacente de la $n(n-1)/2$ la $2(n-1)$
 - c. Pentru că celelalte rutări nu au Router ID
 - d. Ca să cripteze mesajele
5. Un rutier nou, cu prioritate 200, este adăugat pe un segment unde există deja un DR cu prioritate 1. Ce se întâmplă?
 - a. Devine imediat DR
 - b. Nu se întâmplă nimic - alegerea DR nu este preemptivă
 - c. Devine BDR automat
 - d. Adiacentele se rup
6. Două rutări OSPF rămân blocate în starea Exchange. Cauza cea mai probabilă?
 - a. Router ID-uri identice
 - b. MTU diferit pe cele două interfețe
 - c. Priorități diferite
 - d. Numere de proces diferite
7. Cu referința implicită de 10^8 , ce cost OSPF au o legătură de 100 Mbps și una de 10 Gbps?
 - a. 1 și respectiv 0,01
 - b. Ambele 1 - OSPF nu le poate distinge
 - c. 10 și respectiv 1
 - d. 64 și respectiv 1
8. Prin ce arie trece obligatoriu traficul dintre aria 10 și aria 20?
 - a. Direct, fără arie intermediară
 - b. Prin aria 0, backbone-ul
 - c. Prin aria cu numărul cel mai mic
 - d. Depinde de protocolul de redistribuire

9. Ce masca se folosește în comanda network a lui OSPF?
- Masca de rețea obișnuită
 - Masca wildcard, ca la listele de acces
 - Nu se folosește masca
 - Prefixul CIDR

Note bibliografice

Cursul următor părăsește firul și trece în aer: rețele wireless, cu problemele lor specifice de acces la mediu - nu se poate asculta și transmite simultan, deci CSMA/CD nu funcționează - și cu o problemă de securitate pe care rețeaua cablată nu o are: mediul este public prin definiție.

Laboratorul 5 configurează OSPF single-area pe exact aceeași topologie pe care ați scris, mai devreme, rute statice. Comparația dintre cele două configurații, ca număr de linii și ca reacție la un cablu scos, este scopul lucrării.

- RFC 2328 - OSPF versiunea 2
- RFC 5340 - OSPF pentru IPv6
- RFC 3630 - extensii de inginerie a traficului pentru OSPF
- RFC 2453 - RIP versiunea 2, pentru comparație
- John Moy, *OSPF: Anatomy of an Internet Routing Protocol*

Extras din volumul „Rețele Locale - Note de curs și lucrări de laborator”. Varianta online a acestui capitol conține figurile în formă interactivă.