

3

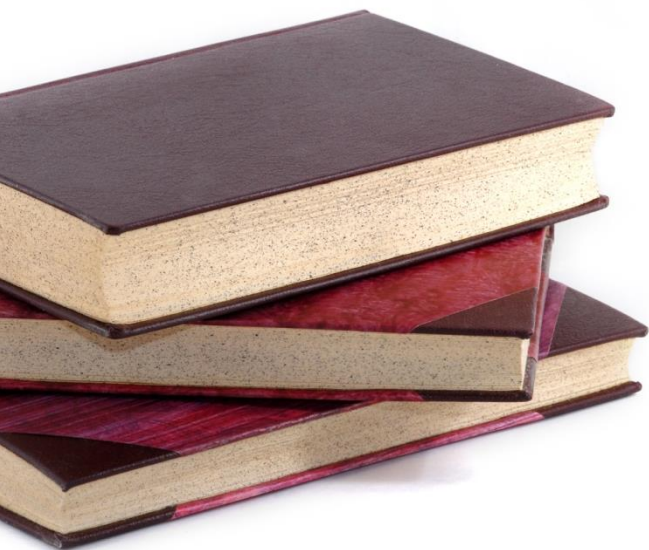
Protocolul IP

20 - 21 octombrie 2015

- Nivelul rețea
- Protocolul IPv4
- ARP

Nivelul rețea

- Necesitatea unei adresări globale
- Funcții
- Protocole



- Problemă: Adresele MAC sunt ineficiente pentru rețele mari:
 - Folosesc o schemă de adresare plată ce nu scalează
- Consecință: adresele MAC sunt folosite doar cu vizibilitate locală (în domeniul local de broadcast)
- Soluție: Este necesară folosirea unui alt set de adrese pentru adresare globală
 - Aceste adrese trebuie să fie organizate ierarhic pentru a putea fi gestionate de echipamentele de rețea

- Codul poștal:

0 6 0042



Regiune poștală:
București



Județ/sector:
Sectorul 6



Stradă + număr:
Splaiul Independenței, Nr. 313

Adresare globală

- Introduce un protocol cu adresare ierarhică numit IP (Internet Protocol)
- Fiecare dispozitiv este identificat în mod unic la nivel global prin adresa sa IP

Comunicație end-to-end fără conexiune

- Protocoalele nivelului rețea sunt de tip best-effort și nu stabilesc conexiuni
- Stabilirea conexiunilor este responsabilitatea protoalelor de nivel superior

Rutare

- Dispozitive intermediare numite rutere iau decizii de dirijare a traficului în funcție de destinație

IPv4

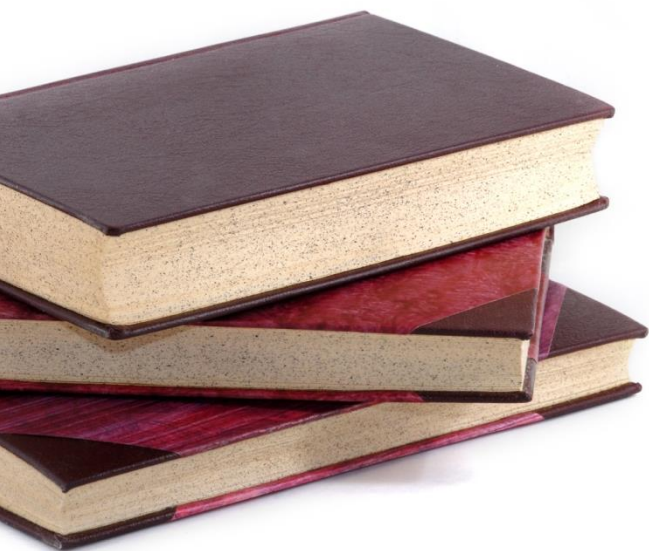
ICMP

IGMP

IPv6

ICMPv6

IPv4



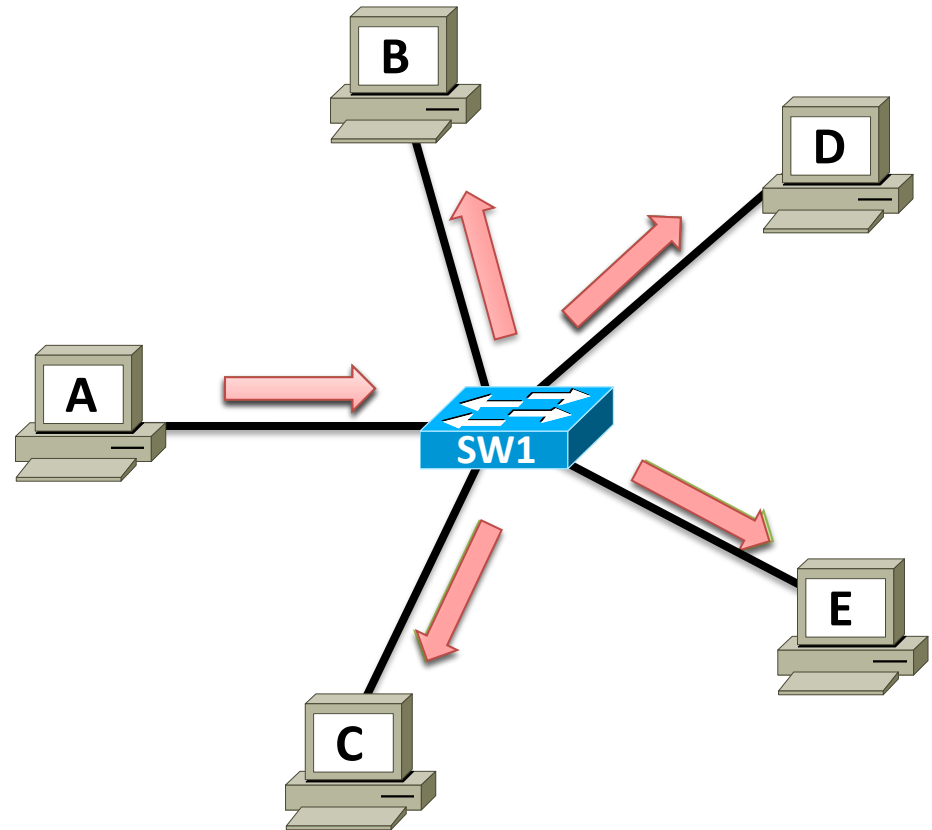
- Funcții
- Format antet
- Adresa IPv4
- Adrese publice și private
- Clase de adrese
- Procesul de subnetare
- VLSM
- Dezavantaje

- IPv4: Internet Protocol, versiunea 4
- Definit în **RFC791**, în anul 1981
- IPv4 oferă fiecărui dispozitiv din Internet o adresă unică: **adresa IP**
- IPv4 adaugă informația de adresare prin **încapsulare**
- PDU-ul (Protocol Data Unit) rezultat ca urmare a încapsulării IP poartă numele de **pachet**
- Pe baza informației de adresare conținută în antetul IP se realizează dirijarea traficului în Internet

Unicast

Broadcast

Multicast



11

Version	Header length	Type of Service	Total length		
Identification			Flags	Fragment Offset	
Time to Live		Protocol	Header checksum		
Source IP Address					
Destination IP Address					
Options					
Data					

- Adresa IPv4 este formată din 4 octeți
- Formatul cel mai folosit este zecimal cu punct:

141 . **85** . **241** . **139**

- Utilă pentru calcule mai este reprezentarea adresei în format **binar**:

10001101 . **01010101** . **11110001** . **10001011**

10000000	128
01000000	64
00100000	32
00010000	16
00001000	8
00000100	4
00000010	2
00000001	1

$$141 = 128 + 8 + 4 + 1 \\ = 10001101$$

$$85 = 64 + 16 + 4 + 1 \\ = 01010101$$

$$241 = 128 + 64 + 32 + 16 + 1 \\ = 11110001$$

$$139 = 128 + 8 + 2 + 1 \\ = 10001011$$

- Adresa IPv4 este compusă din două părți:
 - Partea de rețea
 - Partea de host
- Dispozitivele ce au partea de rețea comună sunt situate în aceeași rețea și pot comunica fără să aibă nevoie de un ruter
- Părțile de rețea și de host se determină folosind **masca de rețea** (**Subnet mask**)
- Masca de rețea este o adresă IP specială ce este formată dintr-un șir continuu de 1 urmat de un șir continuu de 0:

11111111.11111111.11111111.00000000 = 255.255.255.0

- Deoarece notația zecimală a unei măști de rețea este dificil de utilizat s-a introdus o notație specială:

$$11111111.11111111.11111111.00000000 \equiv /24$$

- /24 poartă numele de **prefixul rețelei** și reprezintă numărul de 1 din masca rețelei
- O reprezentare completă a unui IP de stație împreună cu rețeaua din care face parte devine:

$$141.85.241.139/24$$

- Prin aplicarea operației de **AND** pe biți între mască și adresa IP se obține **adresa de rețea**:

Partea de rețea				Partea de host			
141	.	85	.	241	.	139	
10001101	.	01010101	.	11110001	.	10001011	AND
11111111	.	11111111	.	11111111	.	00000000	
10001101	.	01010101	.	11110001	.	00000000	
141	.	85	.	241	.	0	

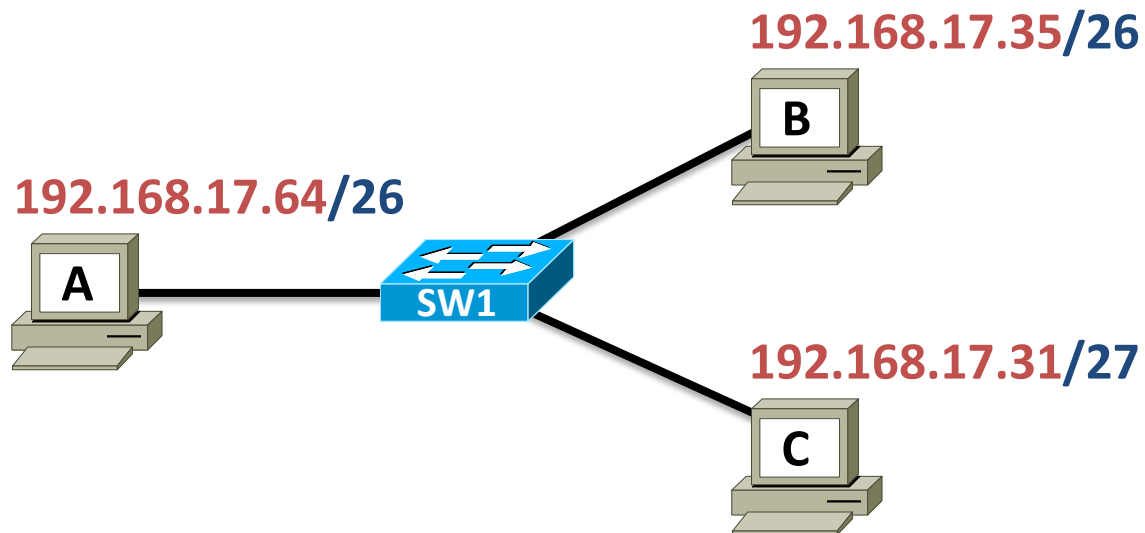
- Adresele de rețea au **toți** biții din partea de host setați pe 0
- Adresa de rețea este folosită de stații pentru a determina dacă să trimită direct destinației sau gateway-ului pachetul

- Prin aplicarea operației de **OR** pe biți între inversa măștii și adresa IP se obține **adresa de broadcast** a rețelei:

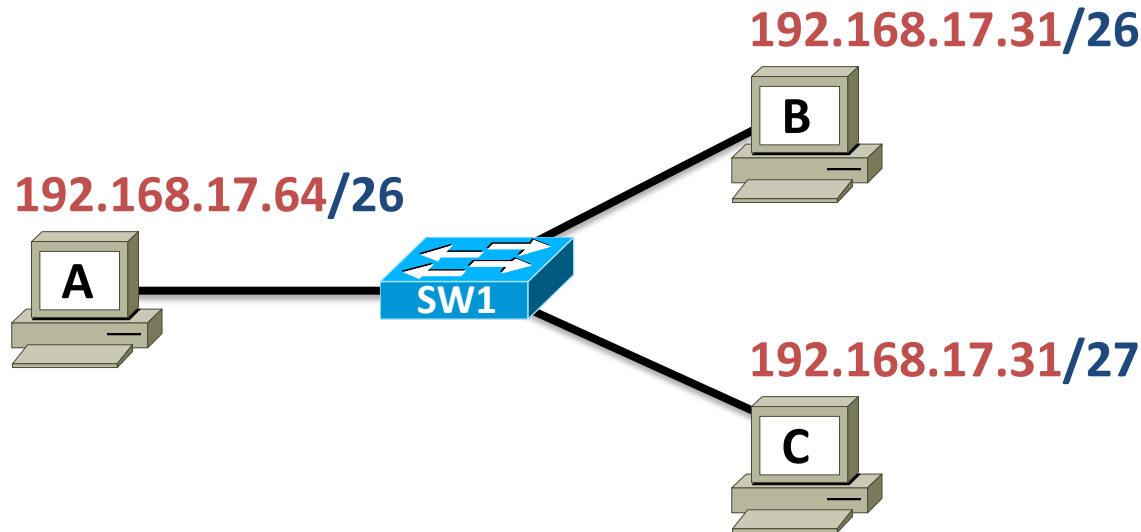
Partea de rețea				Partea de host			
141	.	85	.	241	.	139	
10001101	.	01010101	.	11110001	.	10001011	OR
00000000	.	00000000	.	00000000	.	11111111	
10001101	.	01010101	.	11110001	.	11111111	
141	.	85	.	241	.	255	

- Adresele de broadcast au **toți** biții din partea de host setați pe 1
- Adresa de broadcast este folosită ca adresă destinație în pachete ce vrem să ajungă la toate dispozitivele din respectiva rețea

- O interfață specială a dispozitivelor de rețea este **interfața de loopback**
- Interfața de loopback este virtuală și nu are asociată vreo interfață fizică
- Interfața de loopback este caracterizată prin adresa IP de loopback:
127.0.0.1
- Prin folosirea acestei interfețe se poate testa integritatea stivei de protocoale de pe un sistem

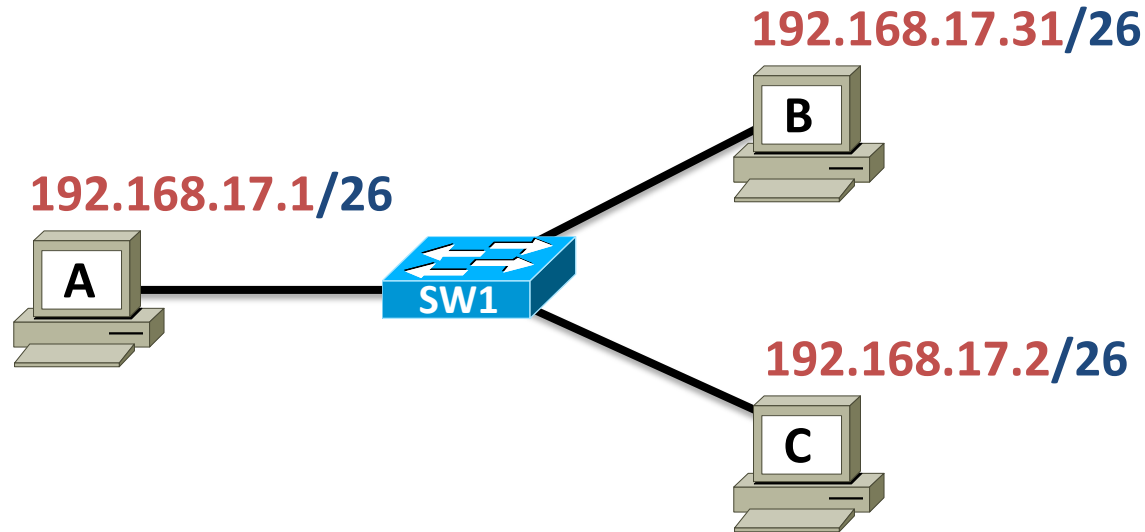


Exercițiul 1: Verificarea configurației



- Stațiile sunt configurate cu IP-urile și măștile din figură. Există vreo problemă cu această configurație?
 - R: Da; A are configurată o adresă de rețea și C are configurată o adresă de broadcast; în plus, C are o mască de rețea diferită de A și B

Exercițiul 2: Broadcast A



- Adresele IP greșite au fost corectate. **A** dă un broadcast. Ce adrese IP sursă și destinație vor fi incluse în antetul IP?
 - R: Sursă: **192.168.17.1**; destinație: **192.168.17.63**
- Care este adresa de rețea a lui **A**?
 - R: **192.168.17.0**

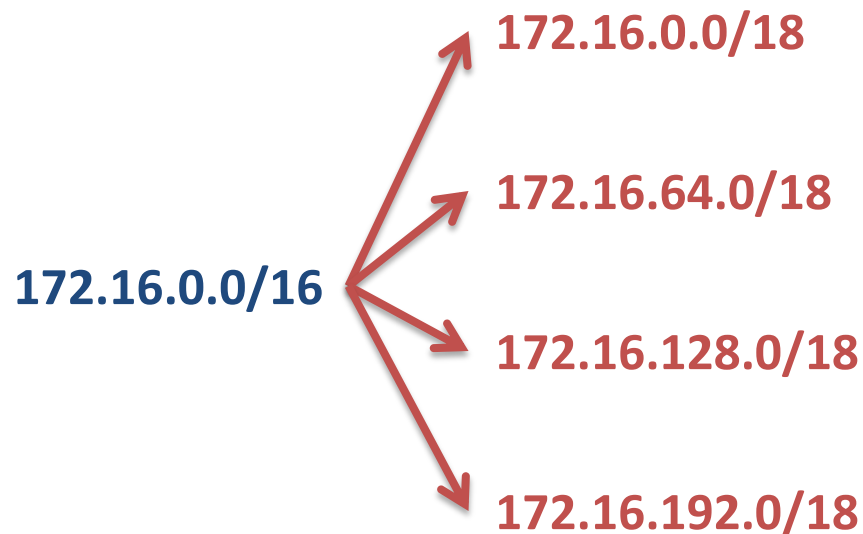
- Adresele IP au fost istoric clasificate în 5 clase de adrese (**A**, **B**, **C**, **D** și **E**), fiecare cu o mască specifică
- Inițial dispozitivele luau în considerare aceste clase pentru a determina masca rețelei
- IANA atribuia unei organizații un întreg bloc classful de adrese, însă cele de clasa **A** erau deseori prea mari și cele de clasa **C** prea mici
- În rețelele moderne clasele de adrese nu mai sunt relevante

- Clasele sunt identificate după primii biți ai primului octet

Clasă	Primul octet	Gama de adrese	Mască	Scop
A	0...	0.0.0.0 – 127.255.255.255	/8	
B	10...	128.0.0.0 – 191.255.255.255	/16	
C	110...	192.0.0.0 – 223.255.255.255	/24	
D	1110...	224.0.0.0 – 239.255.255.255		Multicast
E	1111...	240.0.0.0 – 255.255.255.255		Experimental

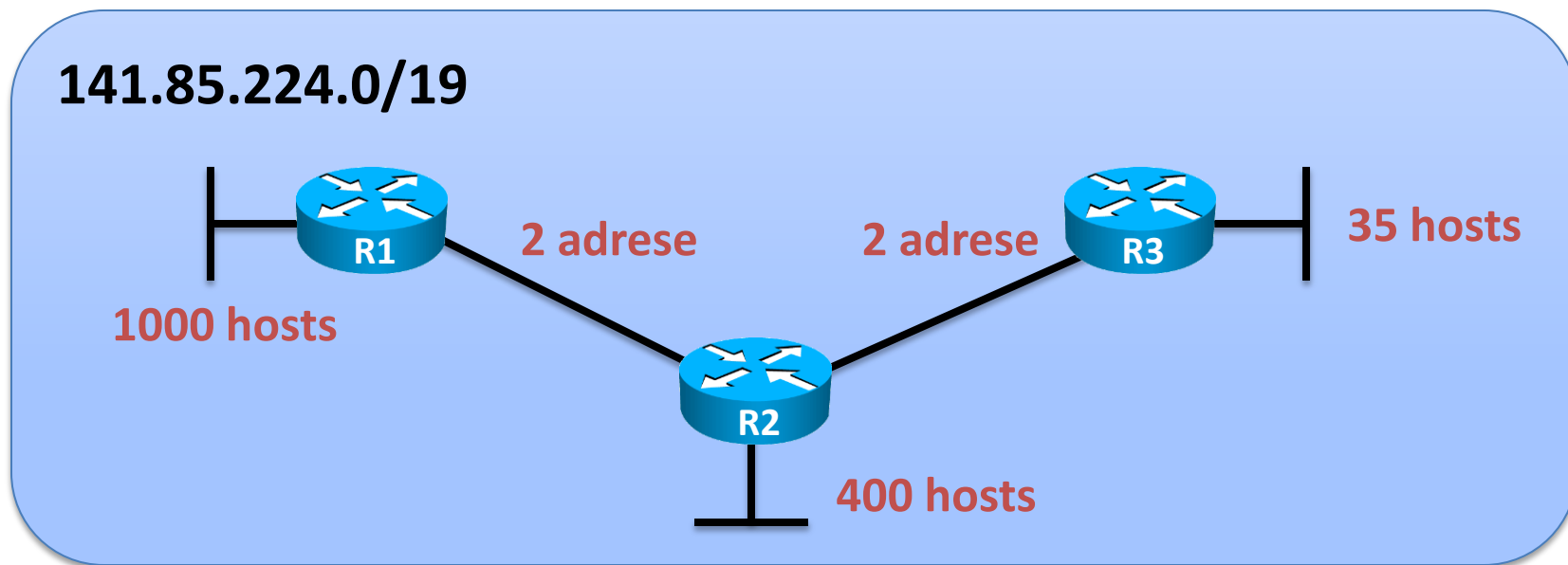
- Pentru a economisi adrese, RFC1918 a alocat trei spații de adrese pentru rețele private:
 - 10.0.0.0/8 – 10.255.255.255/8
 - 172.16.0.0/12 – 172.31.255.255/12
 - 192.168.0.0/16 – 192.168.255.255/16
- Adresele private nu pot fi atribuite unei organizații și nu pot fi folosite în Internet
- Pentru a conecta o stație cu adresă privată la Internet aceasta trebuie translatată la o adresă publică, proces numit **NAT** (Network Address Translation)

- Istoric, un **subnet** reprezenta o rețea obținută prin deplasarea la dreapta a unei măști de rețea classful:



- Rețelele actuale au abandonat ideea de rețele classful și folosesc **VLSM** (Variable Length Subnet Mask); în acestea un subnet nu este cu nimic diferit de o rețea

- O definiție actuală pentru subnet ar putea fi orice rețea ce face parte din spațiul de adresă a unei rețele mai mari
- Procesul de subnetare (**subnetting**) constă în a împărți o rețea mai mare în mai multe rețele ce respectă un set de cerințe



- Înțelegerea procesului de subnetare ne ajută să răspundem la întrebările:
 - Este blocul de adrese cumpărat suficient pentru cerințele organizației?
 - Putem organiza rețelele astfel încât să fim pregătiți pentru extinderea numărului de stații?
 - Este necesară o atribuire optimă a spațiilor de adresă sau este suficientă împărțirea egală între departamente?
 - Putem optimiza tabelele de rutare dacă avem o rețea mare?
- Există două tipuri de subnetare:
 - În subnet-uri egale
 - Optimă (cu pierdere minimă de adrese)

- Exemplu: Să se subneteze spațiul de adrese **192.168.10.0/24** pentru a acomoda trei rețele având **60**, **30** respectiv **15** stații. Subrețelele obținute să fie egale ca dimensiune.
 - Avem nevoie de 3 subrețele deci trebuie împrumutați **2 biți** pentru partea de subrețea a adresei IP

Rețeaua de subnetat: **192** . **168** . **10** . **0** /24

Primul subnet: **11000000.10101000.00001010.00000000/26**

Al doilea subnet: **11000000.10101000.00001010.01000000/26**

Al treilea subnet: **11000000.10101000.00001010.10000000/26**

Rețeaua de subnetat: 192 . 168 . 10 . 0 /24

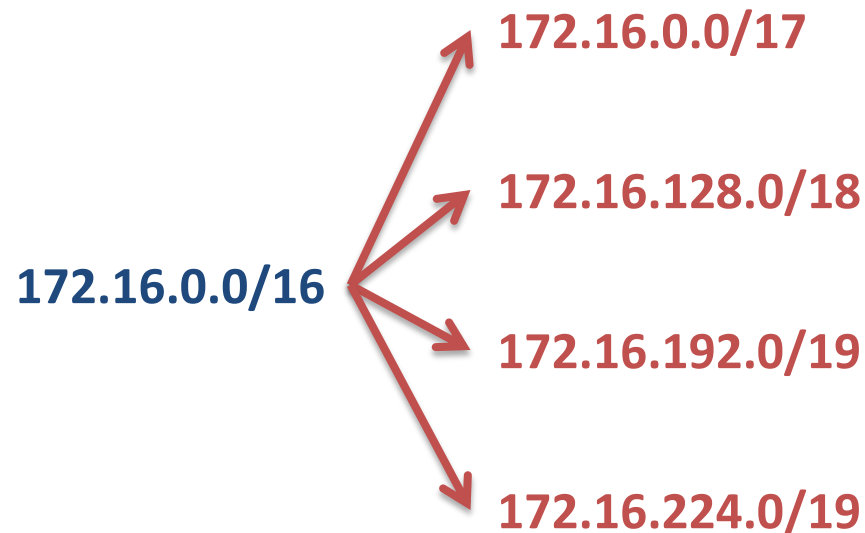
Primul subnet: 11000000.10101000.00001010.00000000/26

Al doilea subnet: 11000000.10101000.00001010.01000000/26

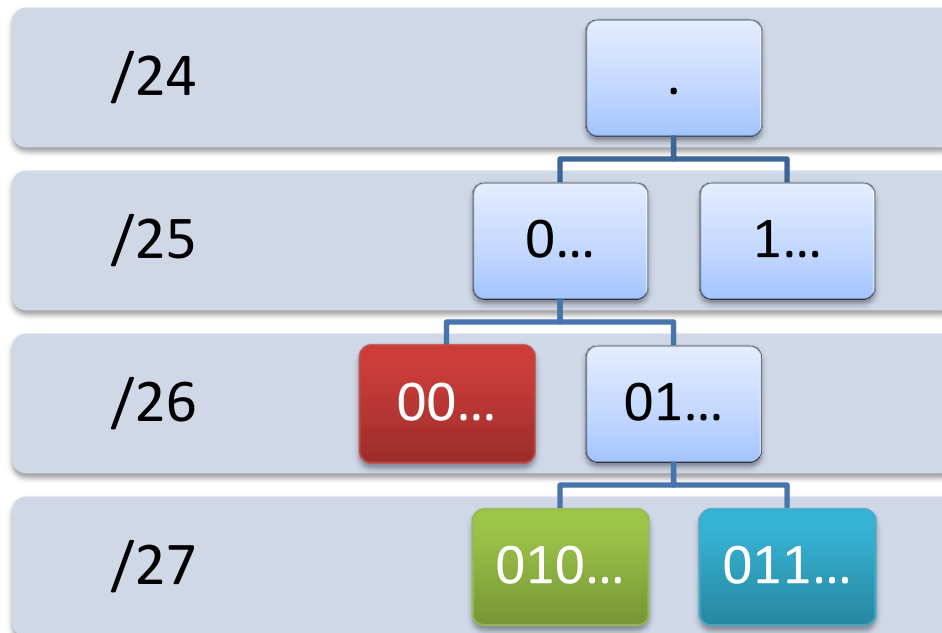
Al treilea subnet: 11000000.10101000.00001010.10000000/26

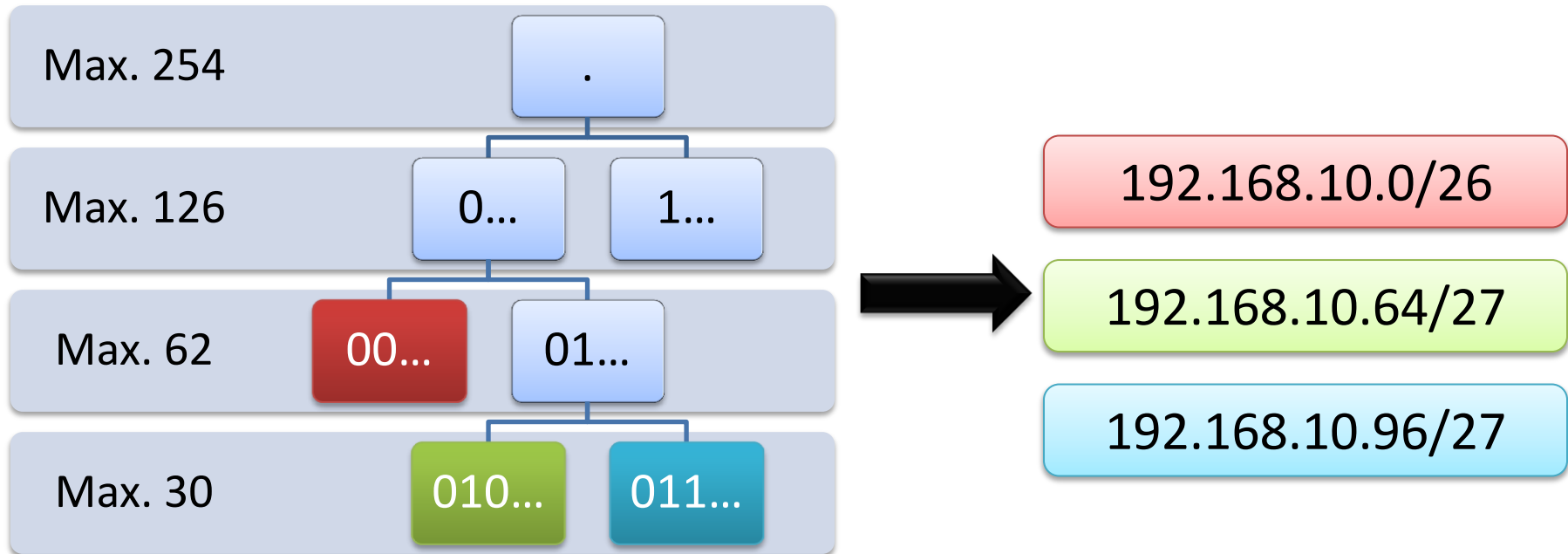
- Cerințele de subrețele erau de 60, 30 și 15 stații. Sunt suficient de mari subrețelele obținute?
 - R: **Da**. Necesarul este de 6, 5, respectiv 5 biți de stație. De ce sunt 5 biți necesari pentru ultima subrețea?
- Cât de multe adrese IP de stații au fost risipite?
 - R: $62 - 60 = 2$; $62 - 30 = 32$; $62 - 15 = 47$; Total: 81

- Putem reduce pierderea de adrese folosind subnetare bazată pe **VLSM**
- VLSM permite creare de subnet-uri ce nu mai au măști de aceeași lungime



- Reluăm exemplul anterior: Să se subneteze spațiul de adrese **192.168.10.0/24** pentru a acomoda trei rețele având **60**, **30** respectiv **15** stații. Subnetarea să risipească un număr minim de adrese.
 - Se observă că pentru cele trei rețele avem nevoie de **6**, **5** respectiv **5** biți de host
 - Putem reprezenta arborescent divizarea ierarhică a ultimului octet:



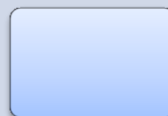


- Cât de multe adrese IP de stații au fost risipite?
 - R: $62 - 60 = 2$; $30 - 30 = 0$; $30 - 15 = 15$; Total: 17

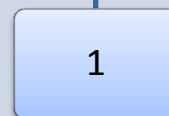
- Să se subneteze optim spațiul de adrese **172.18.240.0/23** astfel încât să fie acomodate cerințele:
 - O rețea cu **200** de host-uri
 - O rețea cu **90** de host-uri
 - Două rețele cu **20** de host-uri
 - O rețea cu **6** host-uri
 - Trei rețele cu **4** host-uri

- Cerințe: 200; 90; 20; 20; 6; 4; 4; 4

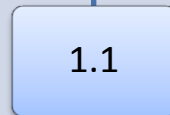
/23



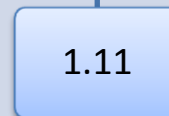
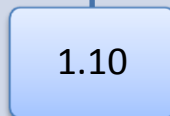
/24



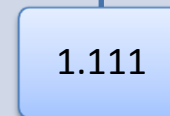
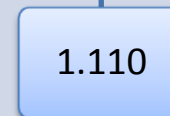
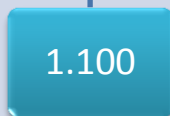
/25



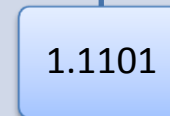
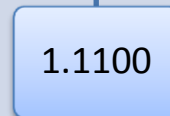
/26



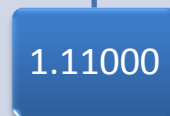
/27



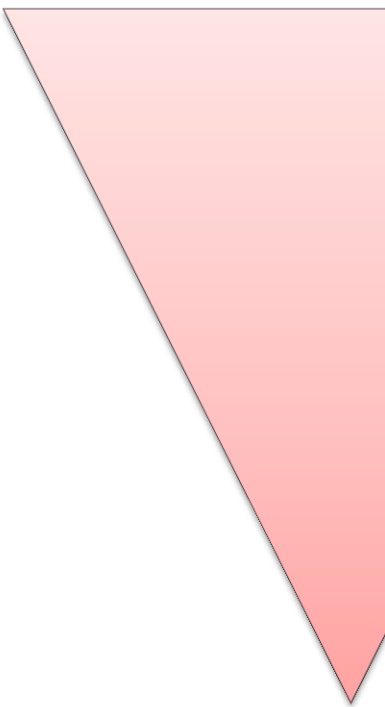
/28



/29



- R:
 - 172.18.240.0/24
 - 172.18.241.0/25
 - 172.18.241.128/27
 - 172.18.241.160/27
 - 172.18.241.192/29
 - 172.18.241.200/29
 - 172.18.241.208/29
 - 172.18.241.216/29



Adrese insuficiente pentru a face față creșterii numărului de dispozitive cu acces la Internet

Antet complicat

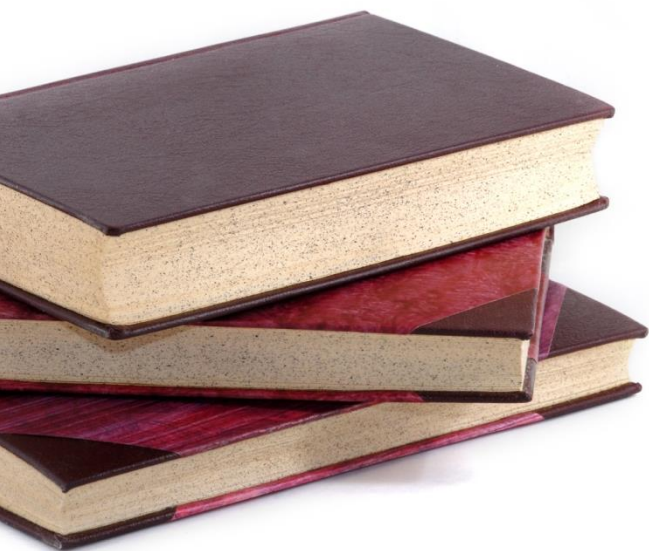
Nu suportă pachete de dimensiuni foarte mari

Suport redus pentru Multicast și IPsec

NAT introduce multe probleme

ARP

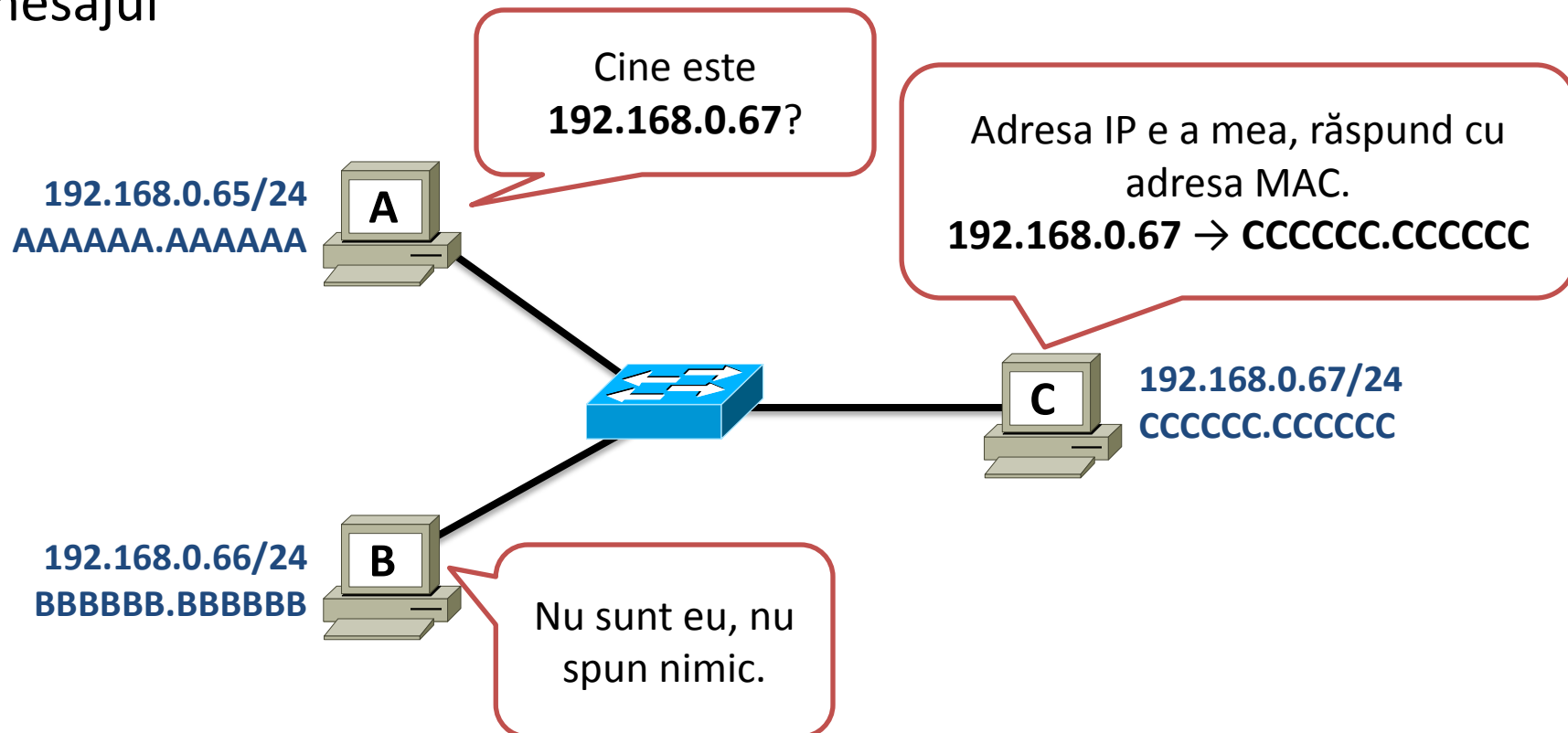
- Descriere
- Format antet
- Exemplu
- Proxy ARP



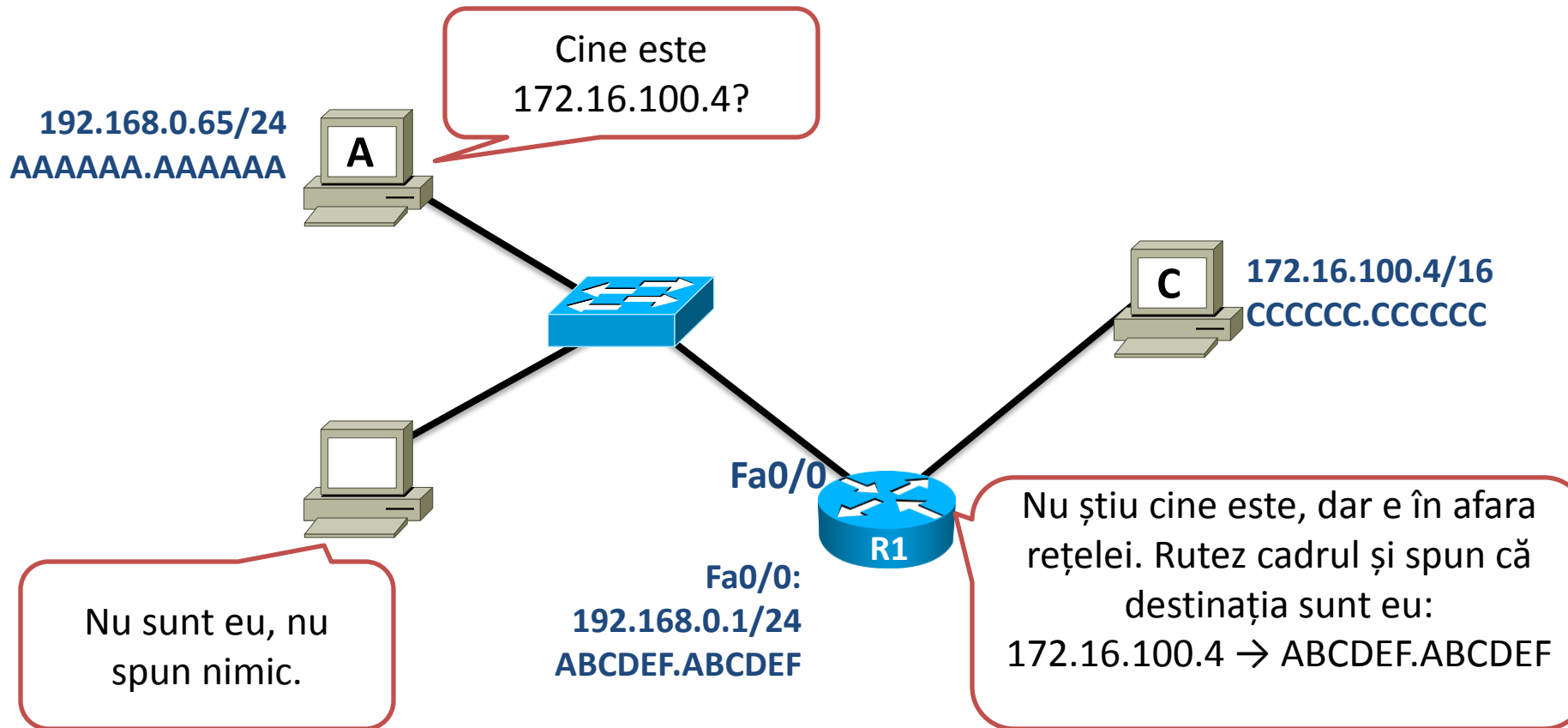
- Când o stație vrea să trimită un pachet într-o rețea Ethernet, aceasta dispune de adresa IP dar nu și de adresa MAC
- Pentru a putea transmite cadrul și a fi acceptat la destinație este necesară determinarea acestei adrese
- Protocolul care determină adresa MAC pornind de la adresa IP poartă numele de ARP (Address Resolution Protocol)

Hardware Type	
Protocol Type	
Hardware Address Length	Protocol Address Length
Operation (1 = request; 2 = reply)	
Sender Hardware Address (48 bits)	
Sender Protocol Address (32 bits)	
Target Hardware Address (48 bits)	
Target Protocol Address (32 bits)	

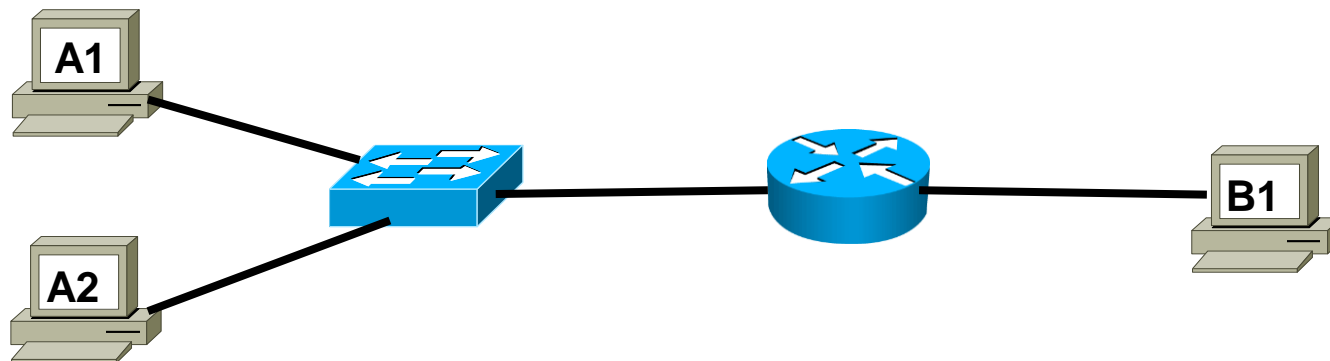
- Inițial emițătorul dă un mesaj la adresa MAC **FFFFFF.FFFFFF** și adresa IP a destinației în care cere adresa MAC
- Doar stația cu IP-ul respectiv va răspunde, restul vor ignora mesajul



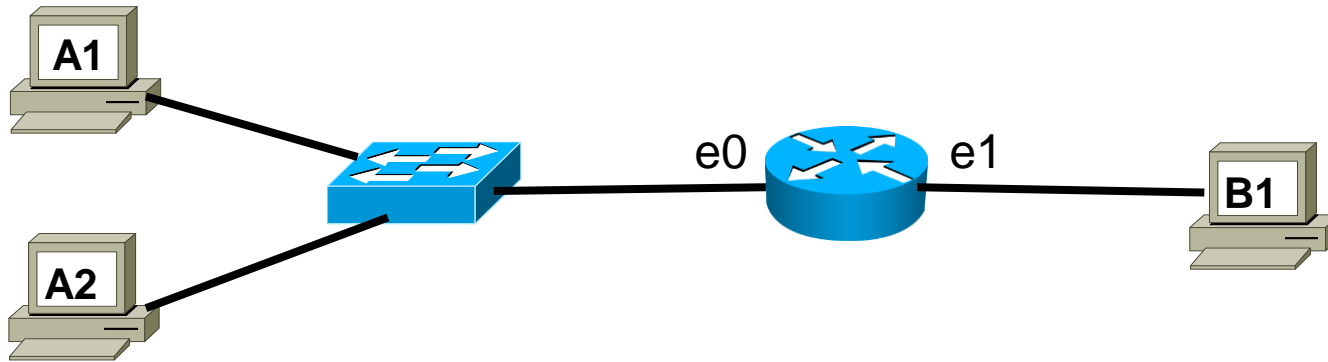
- Tehnică de ARP
- Ruterul răspunde cu propria sa adresă MAC pentru o adresă IP aflată în afara rețelei emițătorului



- Stația sursă verifică dacă destinația se află în aceeași rețea
- Dacă da, cererea ARP va conține adresa IP destinație
- Dacă nu, cererea ARP va conține adresa IP a default gateway-ului
- Ce se întâmplă în cazul în care sursa nu știe adresa default gateway-ului?
 - R: Va trimite un cadru ARP de broadcast la care îi va răspunde ruter-ul de la ieșirea din rețea doar dacă are serviciul de proxy ARP activat.



- Un ruter va avea câte o tabelă ARP pe fiecare interfață multiacces activă.
- Câte tabele ARP are un switch? De ce?
 - R: 0.



A1 → A2

FF...	MAC A1	IP A2	IP A1	ARP rq
MAC A1	MAC A2	IP A1	IP A2	ARP rs
MAC A2	MAC A1	IP A2	IP A1	Date

A1 → B1 (default gateway)

FF...	MAC A1	IP e0	IP A1	ARP rq
MAC A1	MAC e0	IP A1	IP e0	ARP rs
MAC e0	MAC A1	IP B1	IP A1	Date

A1 → B1 (proxy ARP)

FF...	MAC A1	IP B1	IP A1	ARP rq
MAC A1	MAC e0	IP A1	IP B1	ARP rs
MAC e0	MAC A1	IP B1	IP A1	Date

