SYSTEMS LABORATORY

Cursul 8

NAT și tunelare

RL
crunch it connected

Translatarea adreselor

- Problema epuizării adreselor IPv4
- NAT
- PAT
- Configurare NAT cu iptables
- Dezavantajele translatării

RL
crunch it connected

Problema epuizării adreselor IPv4

- Problemă majoră IPv4
- Au fost introduse mecanisme pentru conservarea spațiului
- S-au alocat trei spații pentru adrese private:
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
- Aceste adrese nu pot fi folosite în Internet
- Pentru ca o stație cu adresă privată să poată accesa Internetul adresa acesteia trebuie translatată

Procesul de translatare

- Atunci când un pachet trece printr-un ruter adresele IP sursă și destinație rămân neschimbate
- Procesul de translatare presupune schimbarea adresei IP sursă sau destinație a unui pachet la trecea printr-un ruter
- Procesul poartă numele de **NAT** (Network Address Translation)
- Pentru conectivitate translatarea trebuie să aibă loc în ambele direcții

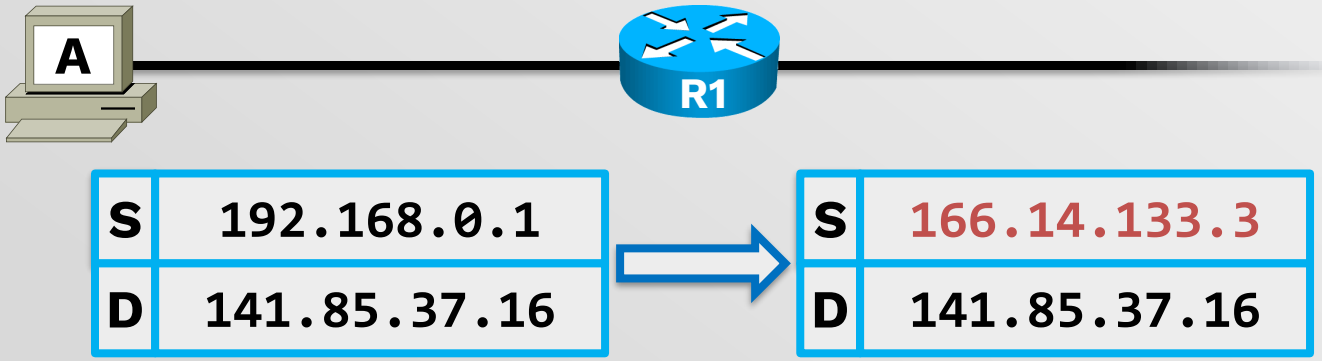
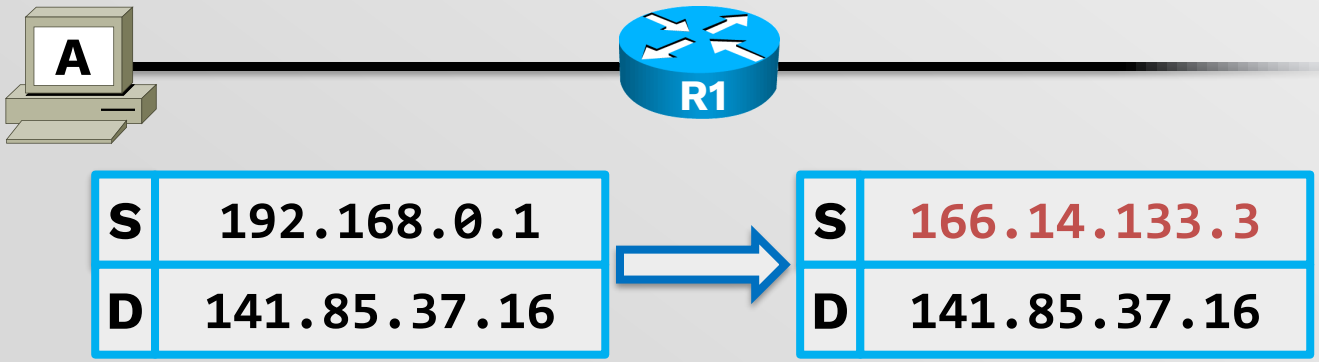


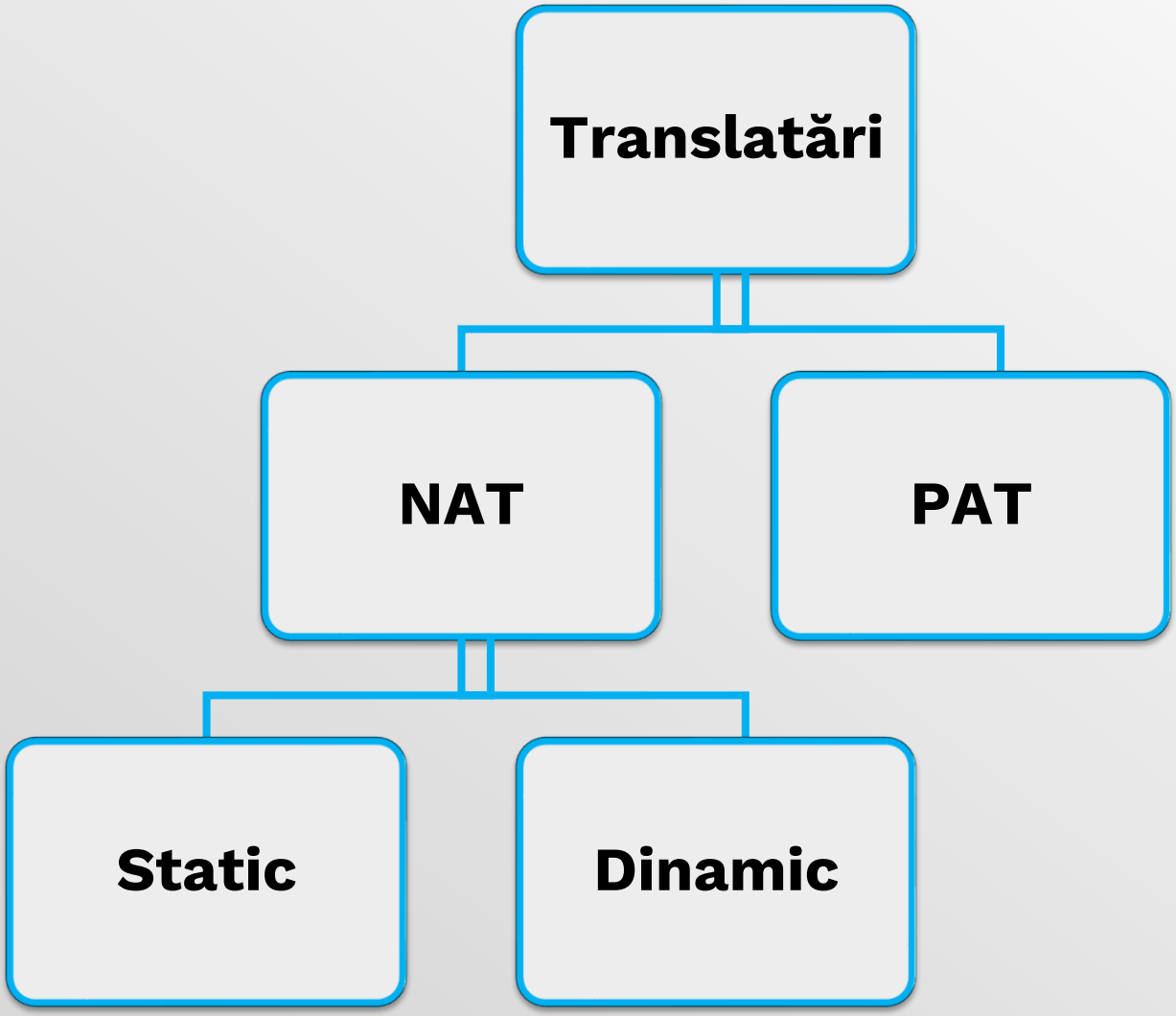
Tabela NAT

- Ruterul ține evidența translatărilor ce trebuie făcute în tabela de NAT
- Tabela NAT:
  - Poate fi construită static (de către administrator) sau dinamic (prin inspectarea traficului ce trece prin ruter)
  - Păstrează o listă de asocieri **adresă internă – adresă externă**

Tabela NAT:
192.168.0.1 - 166.14.133.3

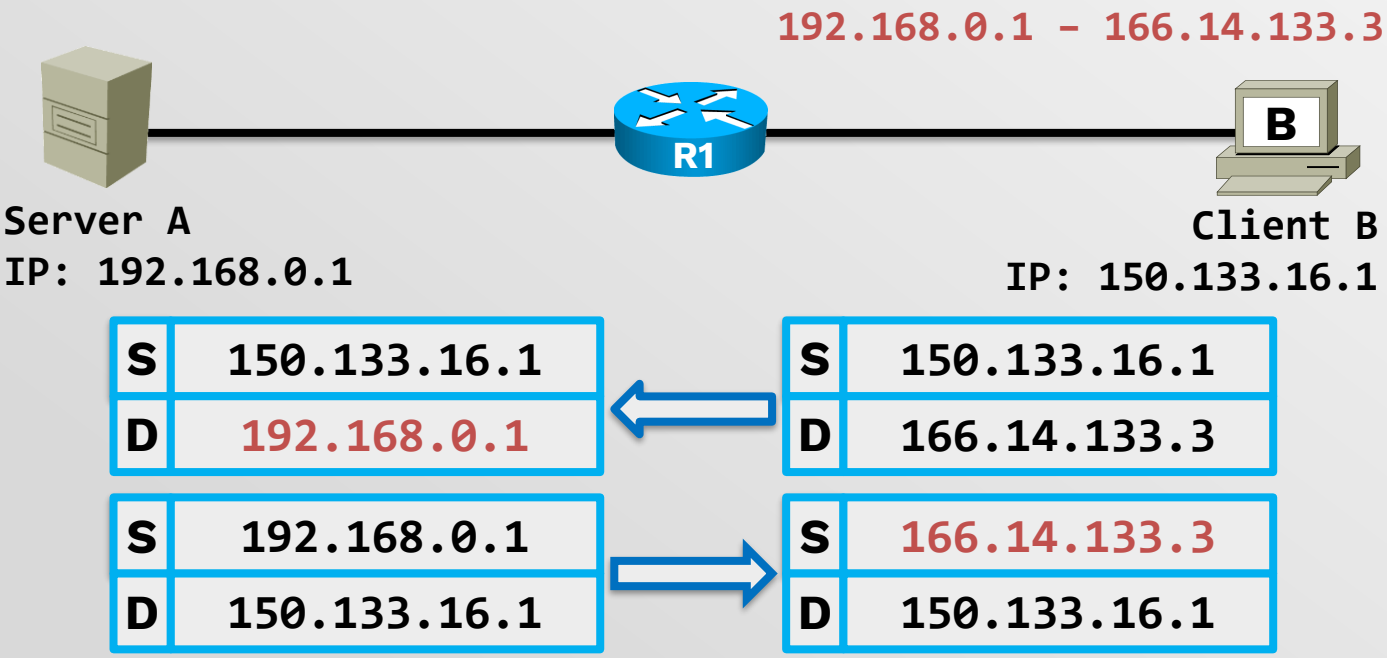


Procesul de translatare



NAT Static

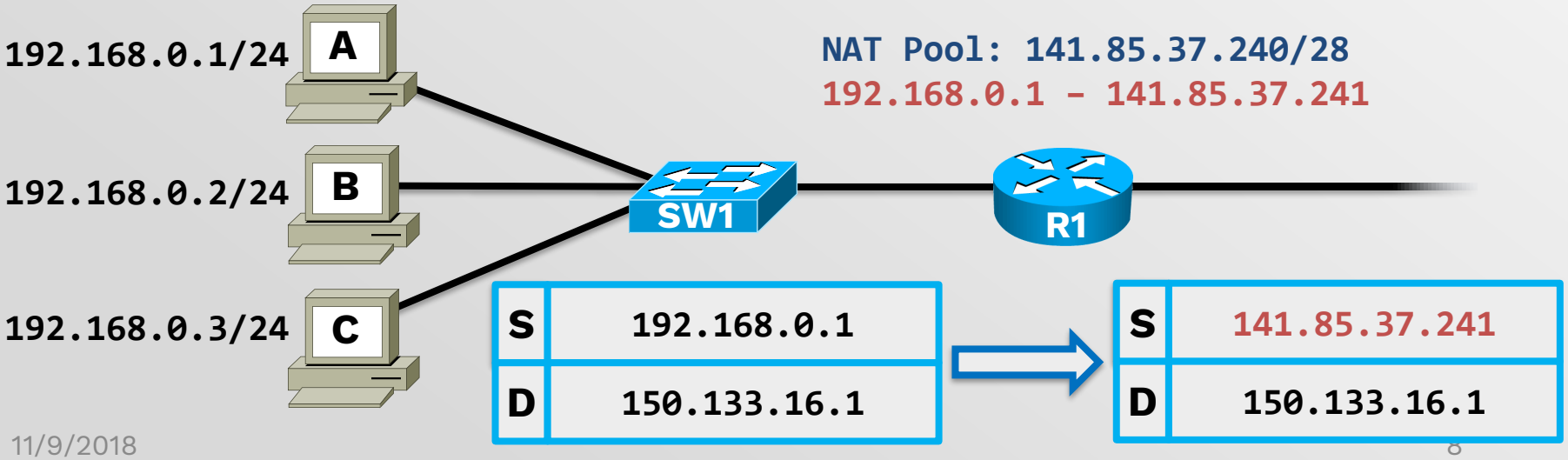
- **Problemă**: **Serverul A** are o adresă privată însă vrem să fie accesibil în exterior printr-o adresă publică unică și constantă
- **Soluție**: NAT Static
  - Adresa internă a serverului este mereu translatată la o adresă publică rezervată



NAT Dinamic

- Problemă: Avem în rețeaua privată 40 de stații dar doar 20 de adrese publice
- Soluție: NAT Dinamic
  - Stațiile care vor să comunice în Internet primesc temporar una din adresele publice disponibile (din NAT Pool), dacă mai există adrese nefolosite

  Ar putea fi o soluție NAT dinamic pentru problema anterioară a serverului?

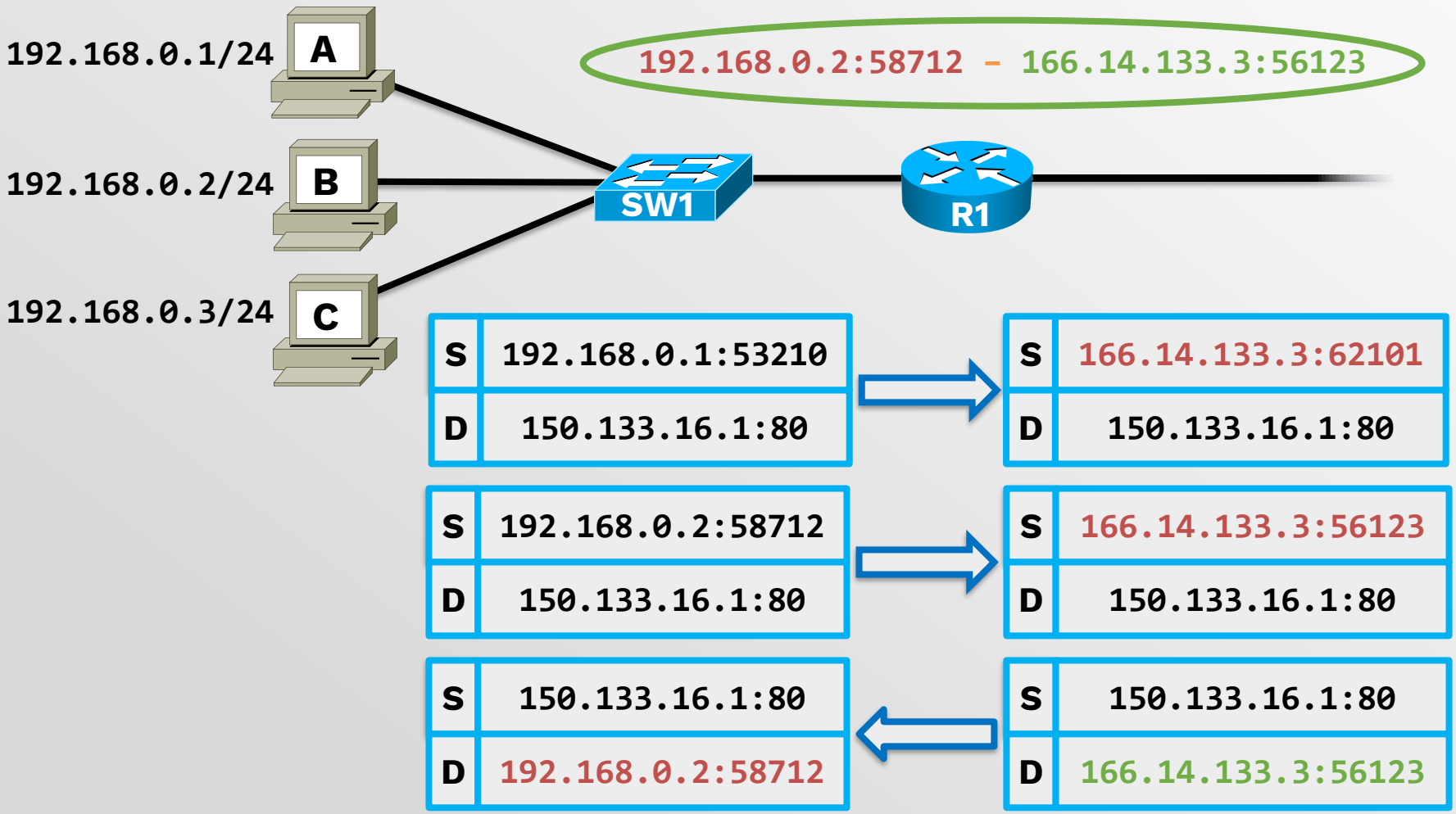


PAT

- Problemă: Avem în rețeaua privată 40 de stații dar o singură adresă publică
- Soluție: PAT (Port Address Translation)
  - Mai poartă și numele de masquerade sau NAT Overload
  - La translatare se asociază fiecărei comunicații și un port (un identificator de nivel transport ce indică programul sursă/destinație) pe ruter
  - Când răspunsul destinatarului ajunge la ruter, acesta citește portul din pachet și consultă tabela NAT pentru a vedea în ce să translateze

Tabela NAT		
192.168.0.1:80	-	166.14.133.3:62101
192.168.0.1:1614	-	166.14.133.3:62102
192.168.0.2:80	-	166.14.133.3:63105
192.168.0.3:1811	-	166.14.133.3:48231

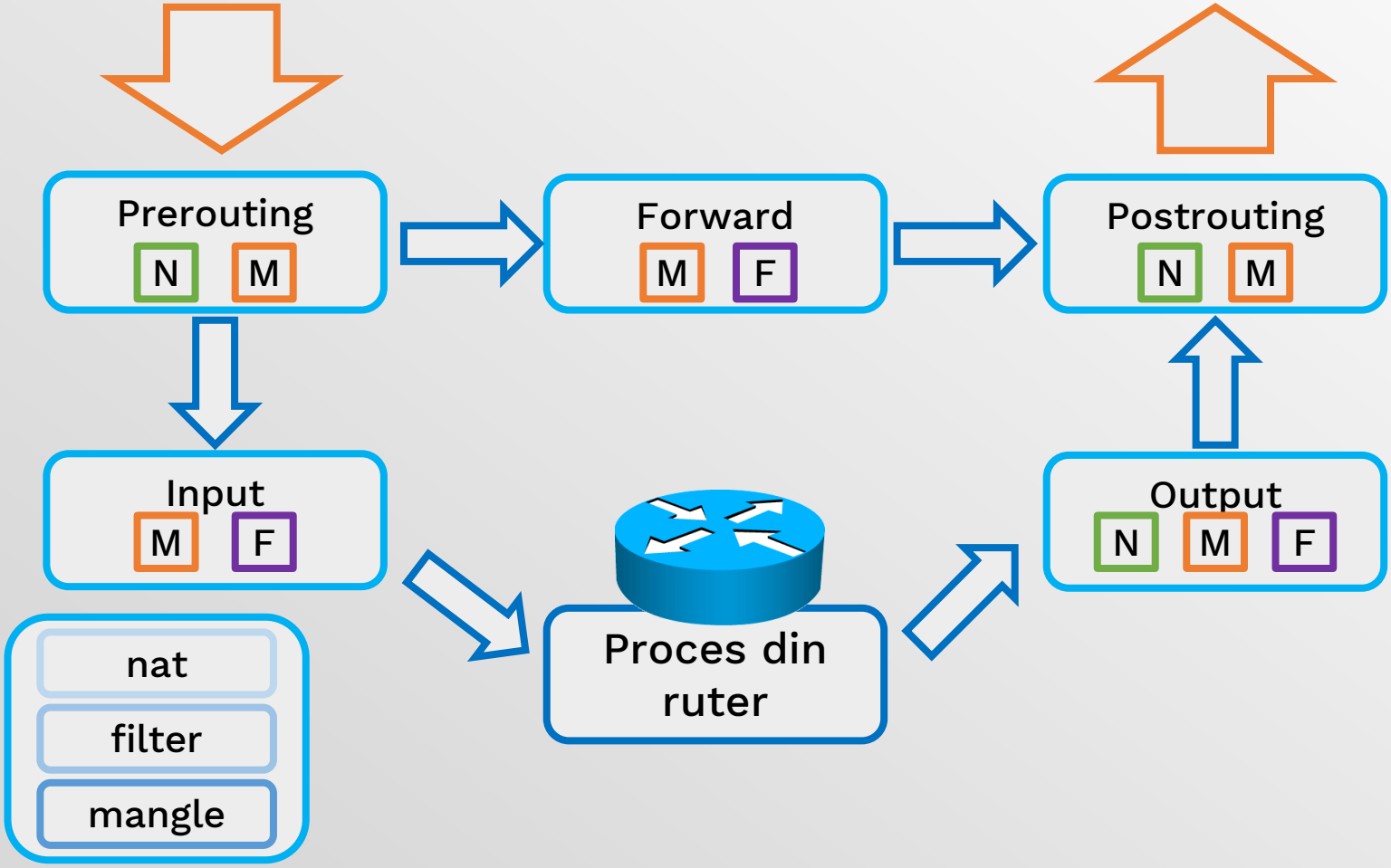
PAT


192.168.0.1:53210 – 166.14.133.3:62101
192.168.0.2:58712 – 166.14.133.3:56123
192.168.0.1/24
A
192.168.0.2/24
B
192.168.0.3/24
C
SW1
R1
S 192.168.0.1:53210
D 150.133.16.1:80
S 166.14.133.3:62101
D 150.133.16.1:80
S 192.168.0.2:58712
D 150.133.16.1:80
S 166.14.133.3:56123
D 150.133.16.1:80
S 150.133.16.1:80
D 192.168.0.2:58712
S 150.133.16.1:80
D 166.14.133.3:56123

NAT în Linux

- Se implementează folosind utilitarul iptables
- Se folosește tabela **nat**
- Lanțurile modificate de comenzile de nat sunt:
  - **PREROUTING** pentru rescrierea destinației
  - **POSTROUTING** pentru rescrierea sursei

Recapitulare: iptables

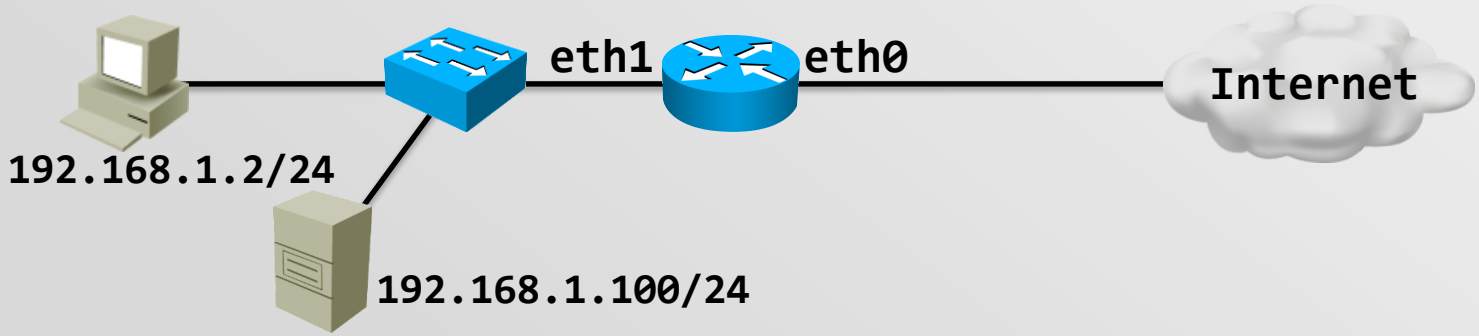


NAT static cu iptables

- Regulile sunt adăugate în tabela **nat** – lanțul **POSTROUTING**
- Este folosit target-ul **SNAT**:
  - Specifică în ce să fie rescrise IP-ul și portul sursă
  - Procesarea lanțului se încheie
- Pentru NAT static trebuie specificată sursa (-s)

```
linux# iptables -t nat -A POSTROUTING -s 192.168.1.100 -j SNAT --to-source 141.85.200.1
```

- Atenție: **SNAT** vine de la Source NAT (nu de la static NAT)



NAT static cu iptables

- Dacă este inițiată din exterior conexiunea, aceasta nu va ajunge la server
- Trebuie creată și regula inversă, care rescrie adresa destinație la trecerea prin ruter
- Rescrierea destinației se face cu target-ul **DNAT** (Destination NAT)
  - Se folosește lanțul de **PREROUTING** în acest caz
    - De ce?

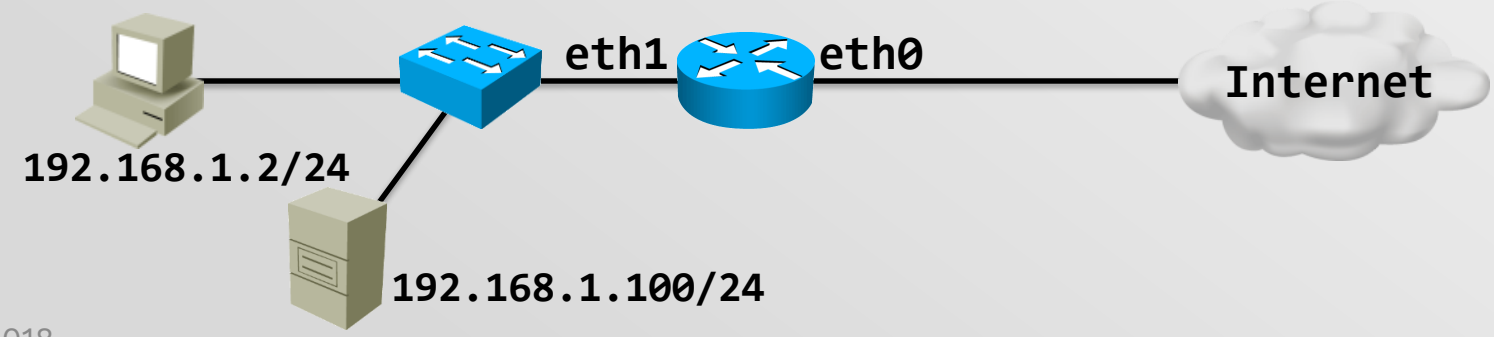
```
linux# iptables -t nat -A PREROUTING -d 141.85.200.1 -j DNAT --to-destination 192.168.1.100
```

NAT dinamic/PAT cu iptables

- Regulile sunt adăugate în tabela **nat** – lanțul **POSTROUTING**
- Tot target-ul **SNAT** este folosit:
  - Pentru NAT dinamic se poate specifica un range de adrese IP
  - Ruterul nu mapează adrese unu la unu (se folosește de fapt o combinație de NAT dinamic cu PAT)

```
linux# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -j SNAT --to-source 141.85.200.2-141.85.200.6
```

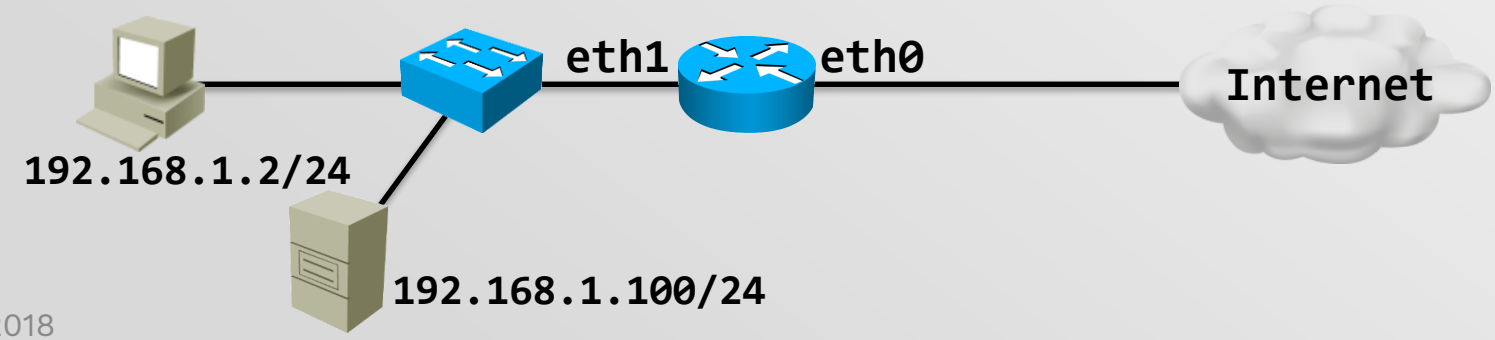
- Vor putea fi inițiate conexiuni din exterior?



NAT cu iptables

- Este vreo problemă cu setul de reguli de mai jos?
  - **R:** Da. Niciodată nu se va face match pe a doua regulă de NAT deoarece sursa 192.168.1.100 va face match pe prima regula

```
linux# iptables -t nat -F
linux# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -j SNAT --to-source 141.85.200.2-141.85.200.6
linux# iptables -t nat -A POSTROUTING -s 192.168.1.100 -j SNAT --to-source 141.85.200.1
linux# iptables -t nat -A PREROUTING -d 141.85.200.1 -j DNAT --to-destination 192.168.1.100
```



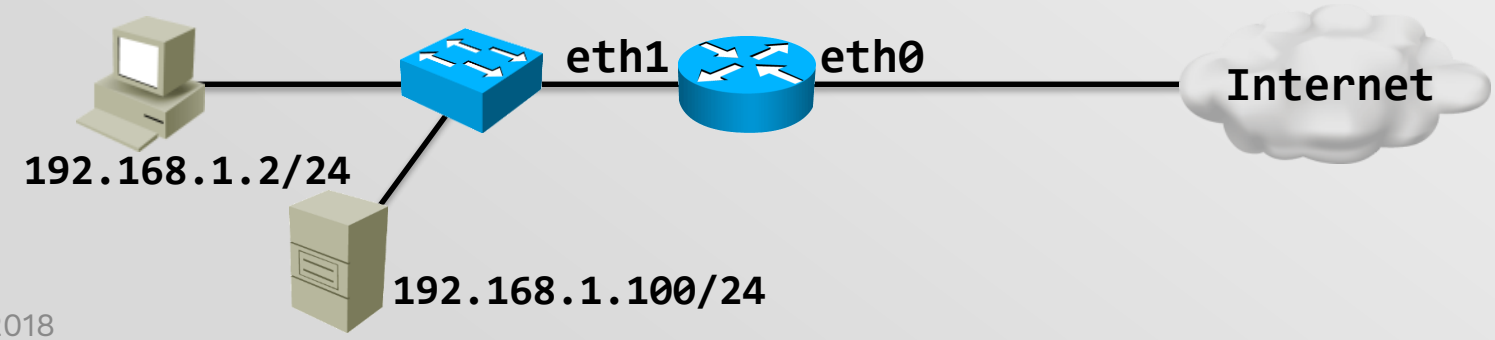
NAT dinamic/PAT cu iptables

- Target-ul **MASQUERADE** specifică faptul că se va folosi IP-ul interfeței de ieșire în translatare
- Utilă când interfața către Internet ia prin DHCP adresa
  - **MASQUERADE** face flush la mapări când interfața e repornită

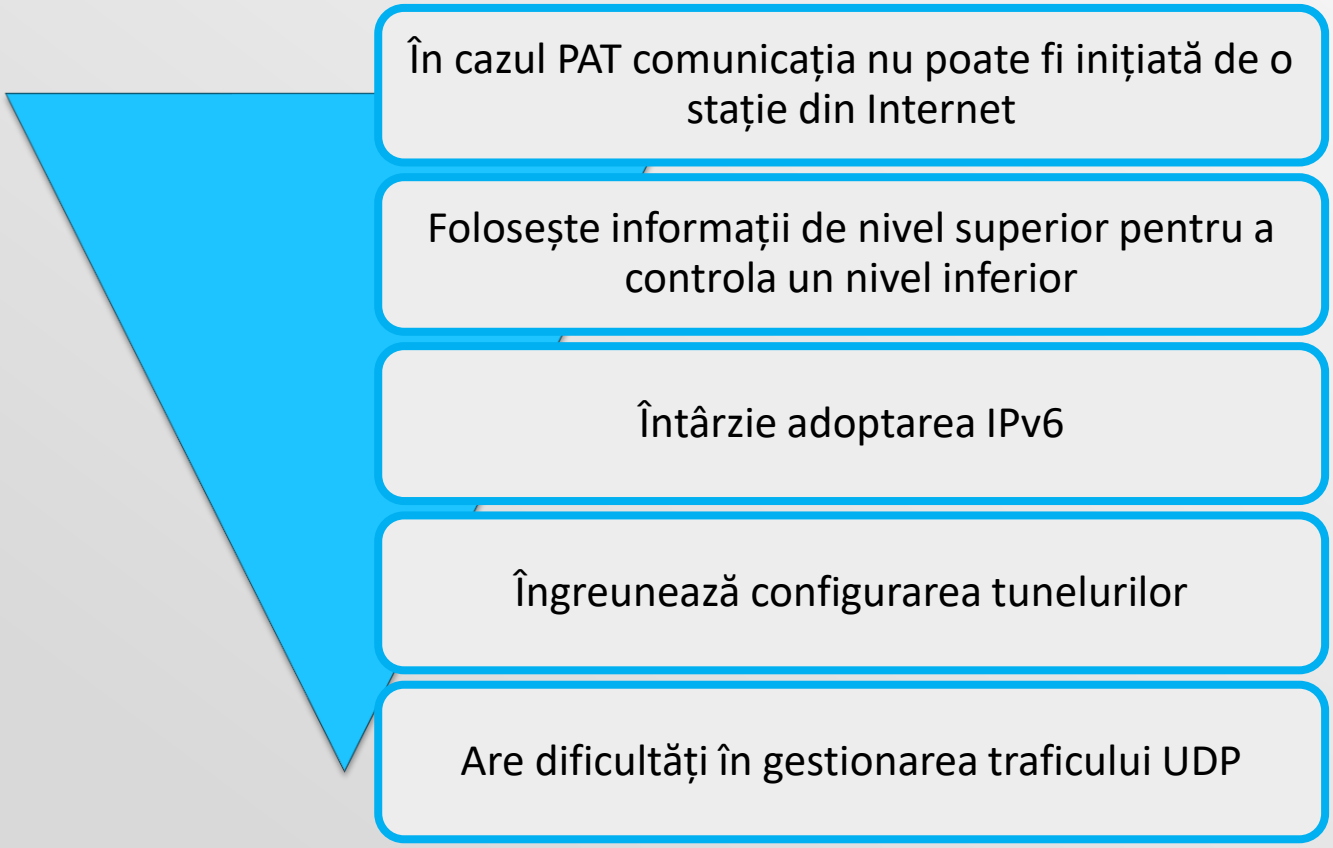
```
linux# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

- Se poate folosi pentru PAT doar un subset de porturi cu --to-ports
  - Trebuie specificat tipul de trafic (UDP sau TCP):

```
linux# iptables -t nat -A POSTROUTING -o eth0 -p tcp -j
MASQUERADE --to-ports 50000-55000
```



Dezavantaje NAT



Tunelare

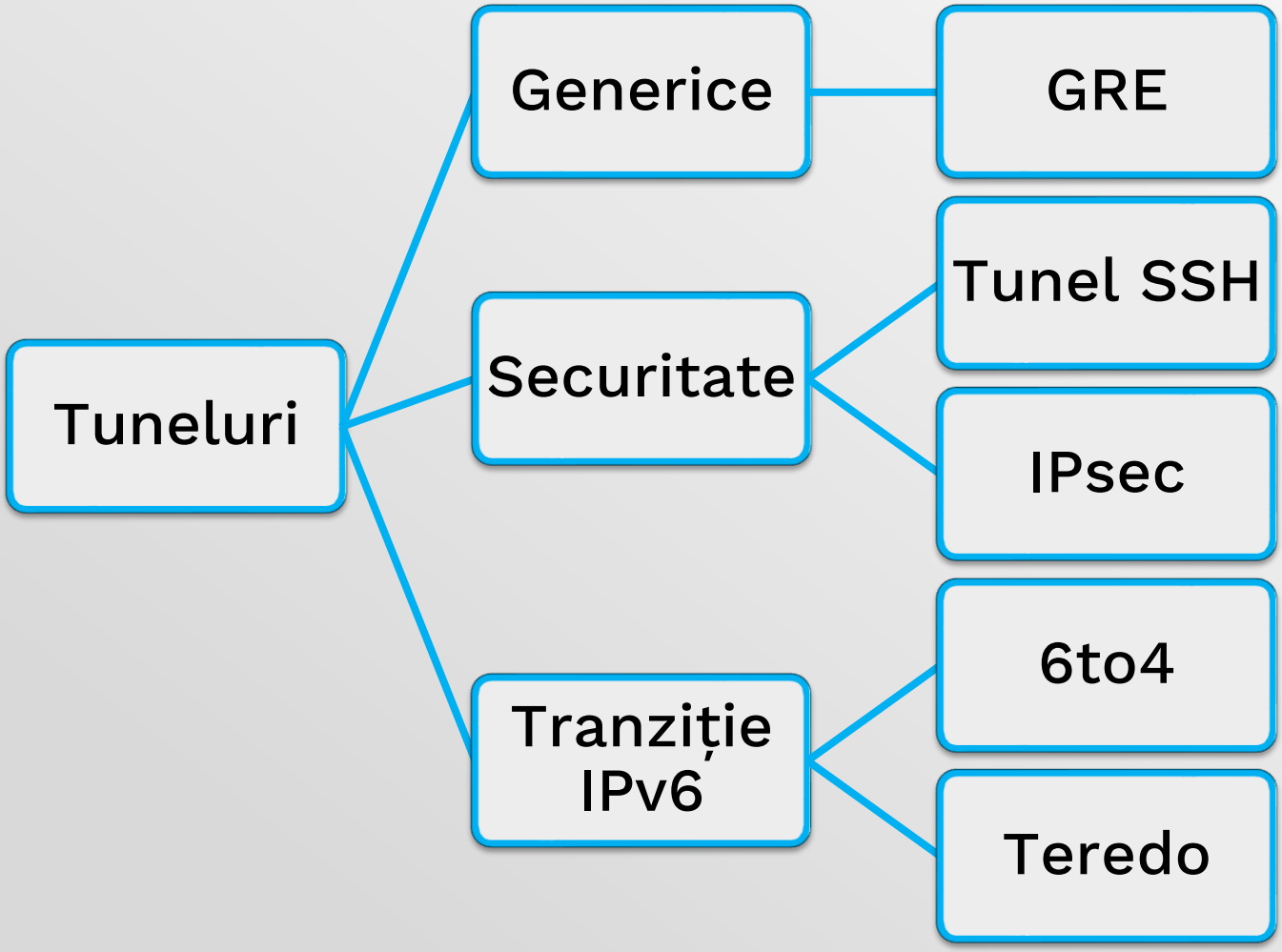
- Conceptul de tunelare
- GRE
- SSH
- 6to4

RL
crunch it connected

Conceptul de tunelare

- Procesul de tunelare constă în încapsularea datelor unui protocol (**payload protocol**) într-un alt protocol (**delivery protocol**)
- **Observație**: Deși IP încapsulează datele TCP și Ethernet încapsulează datele IP, acestea nu sunt considerate exemple de tunelare

Exemple de tuneluri

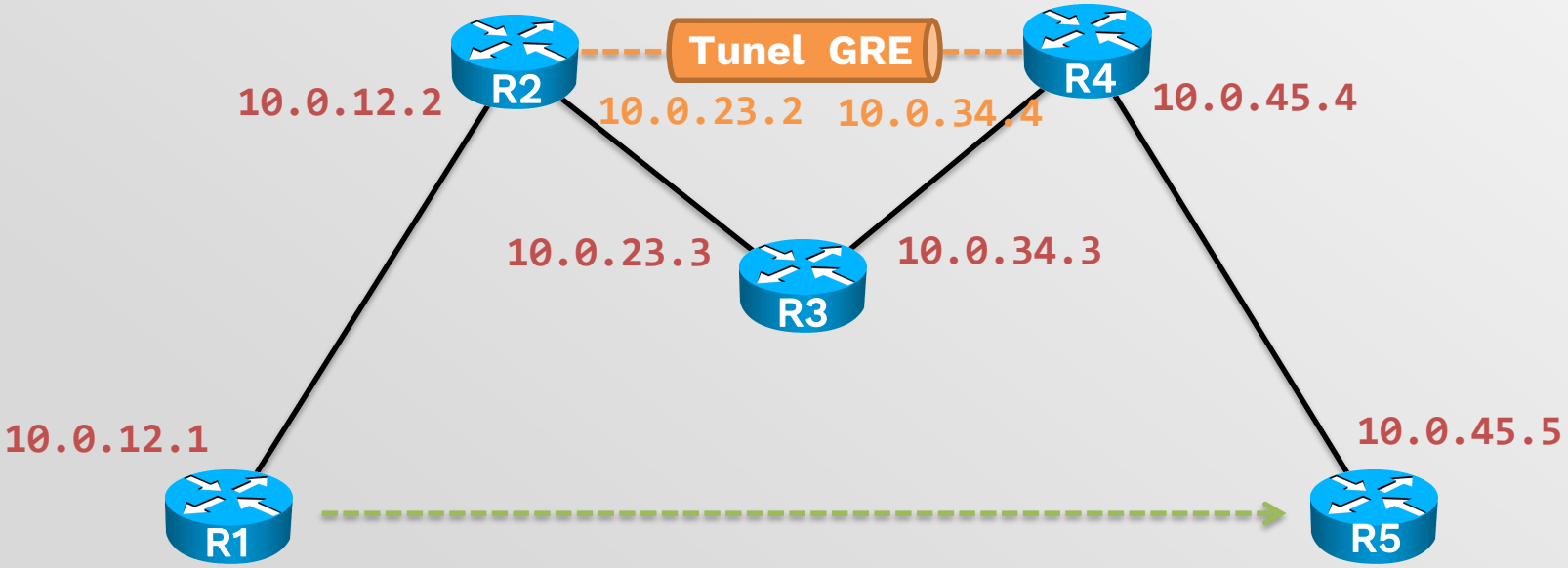


Tunel GRE (Generic Routing Encapsulation)

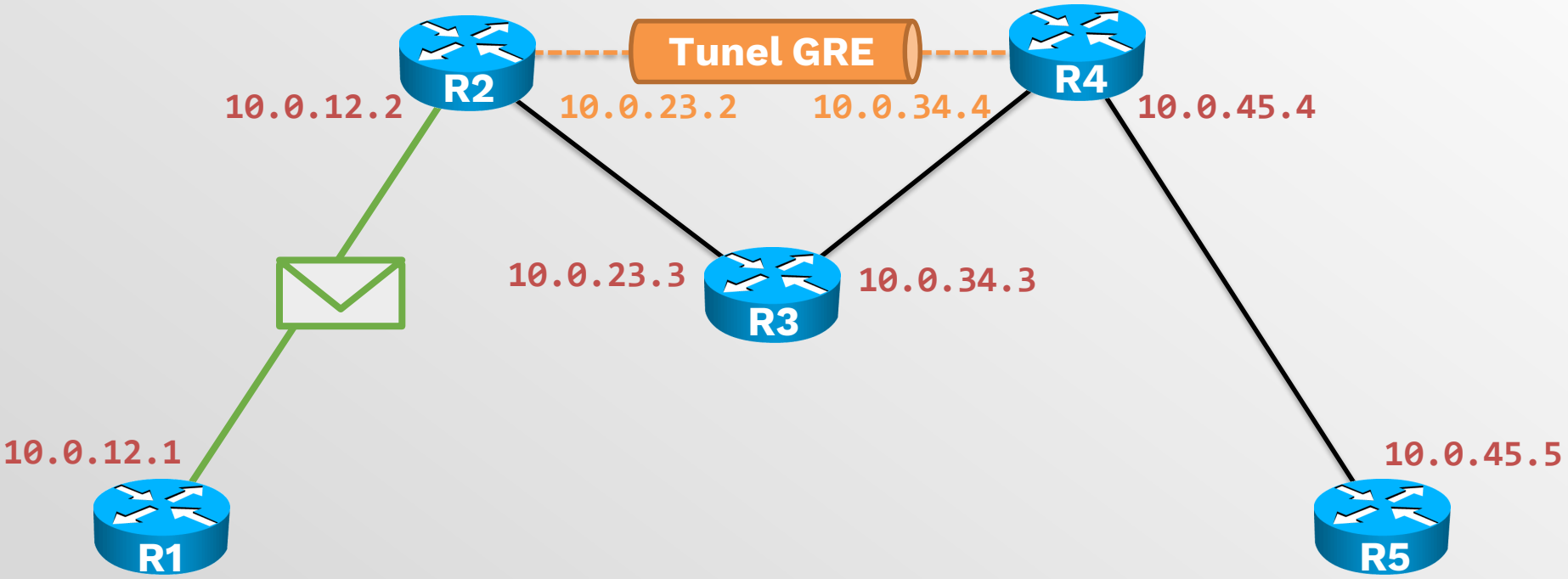
Tunel GRE	
Delivery protocol:	IPv4, IPv6
Payload protocol:	Protocoale de nivel 3
Nivel OSI:	3
Funcție:	Folosit pentru transport de pachete IP fără a fi procesate de ruterele intermediare

Tunel GRE (Generic Routing Encapsulation)

- R1 trimite un pachet către R5
- Între R2 și R4 este configurat un tunel GRE (nu este o legătură fizică)
  - Capetele tunelului sunt reprezentate de IP-urile 10.0.23.2 și 10.0.34.4 de pe interfețele fizice

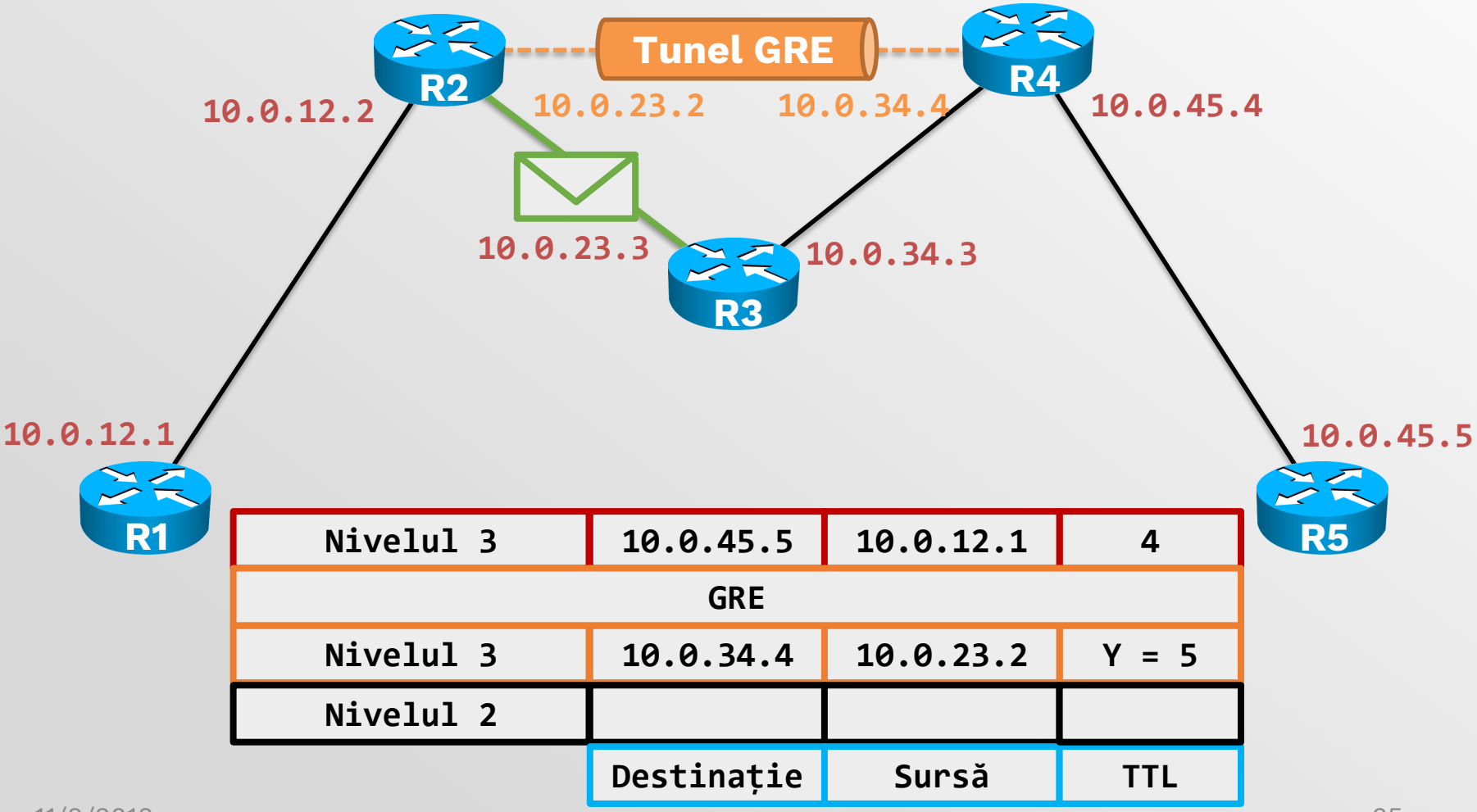


Tunel GRE (Generic Routing Encapsulation)

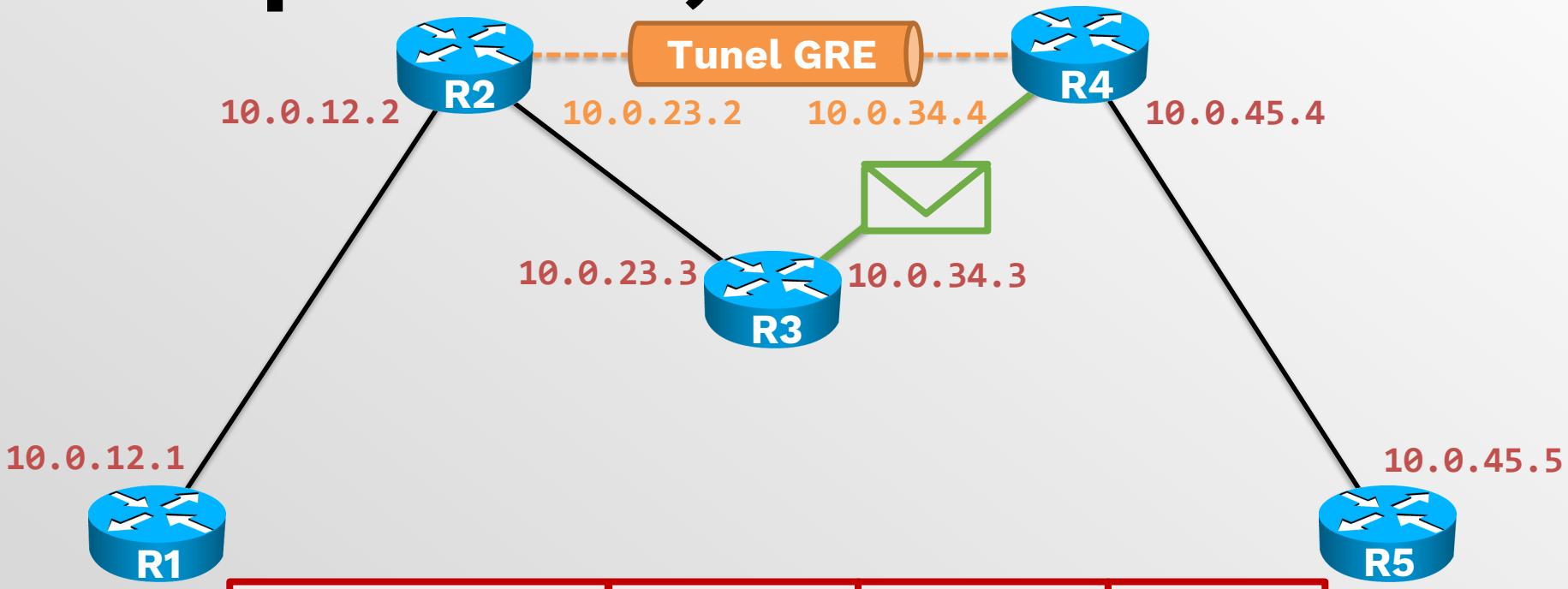


Nivelul 3	10.0.45.5	10.0.12.1	X = 5
Nivelul 2			
	Destinație	Sursă	TTL

Tunel GRE (Generic Routing Encapsulation)

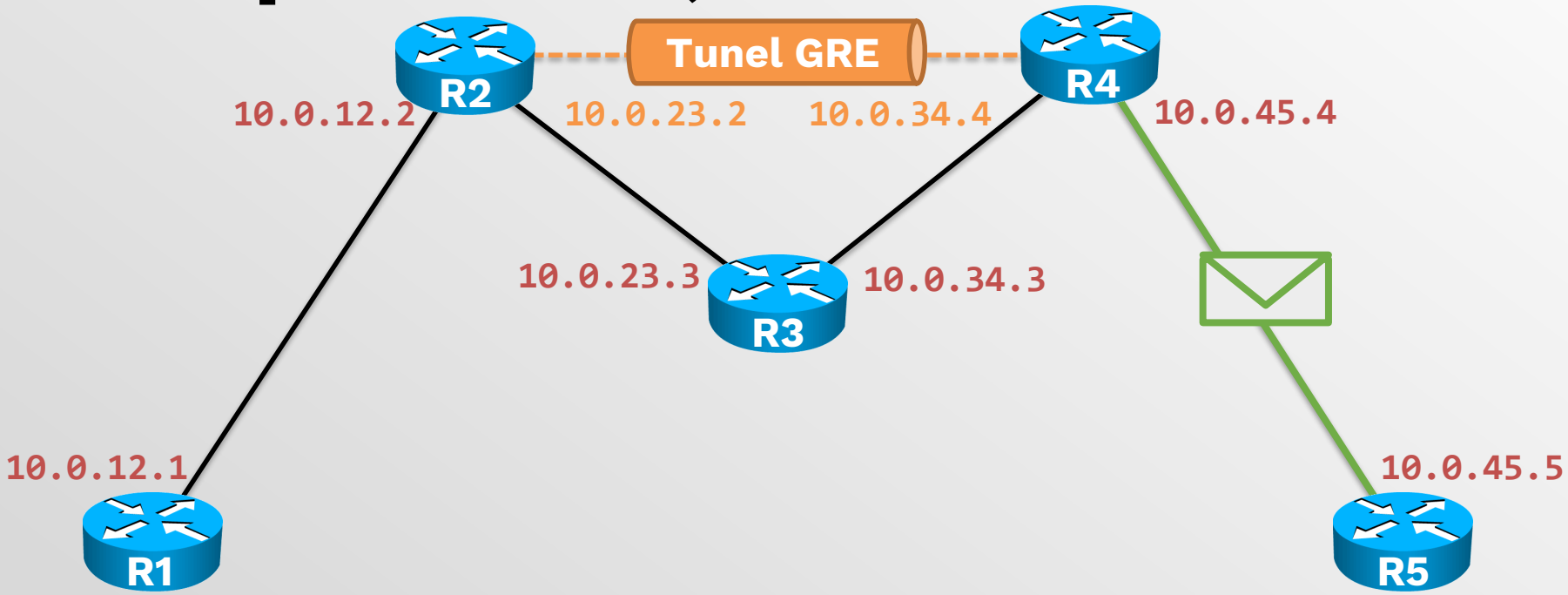


Tunel GRE (Generic Routing Encapsulation)



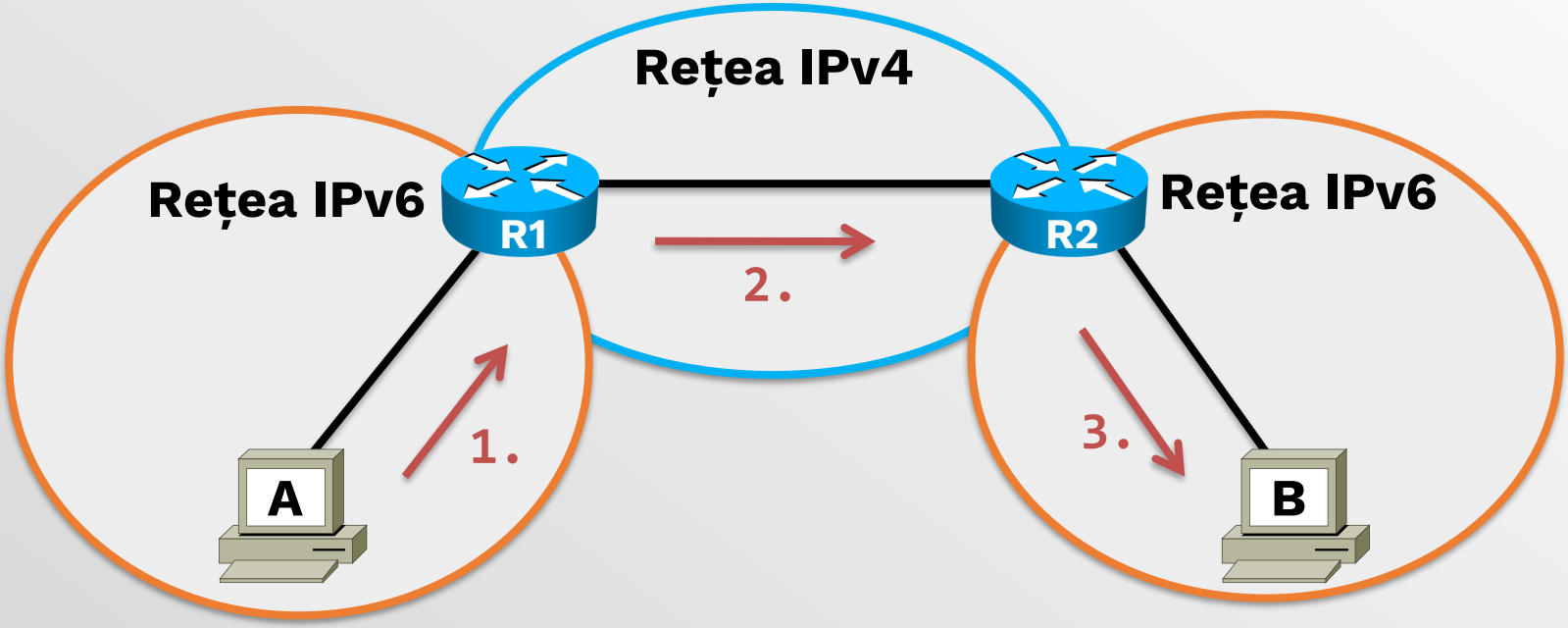
Nivelul 3	10.0.45.5	10.0.12.1	4
GRE			
Nivelul 3	10.0.34.4	10.0.23.2	Y - 1 = 4
Nivelul 2			
	Destinație	Sursă	TTL

Tunel GRE (Generic Routing Encapsulation)



Nivelul 3	10.0.45.5	10.0.12.1	4
Nivelul 2			
	Destinație	Sursă	TTL

Aplicație GRE: IPv6 peste IPv4



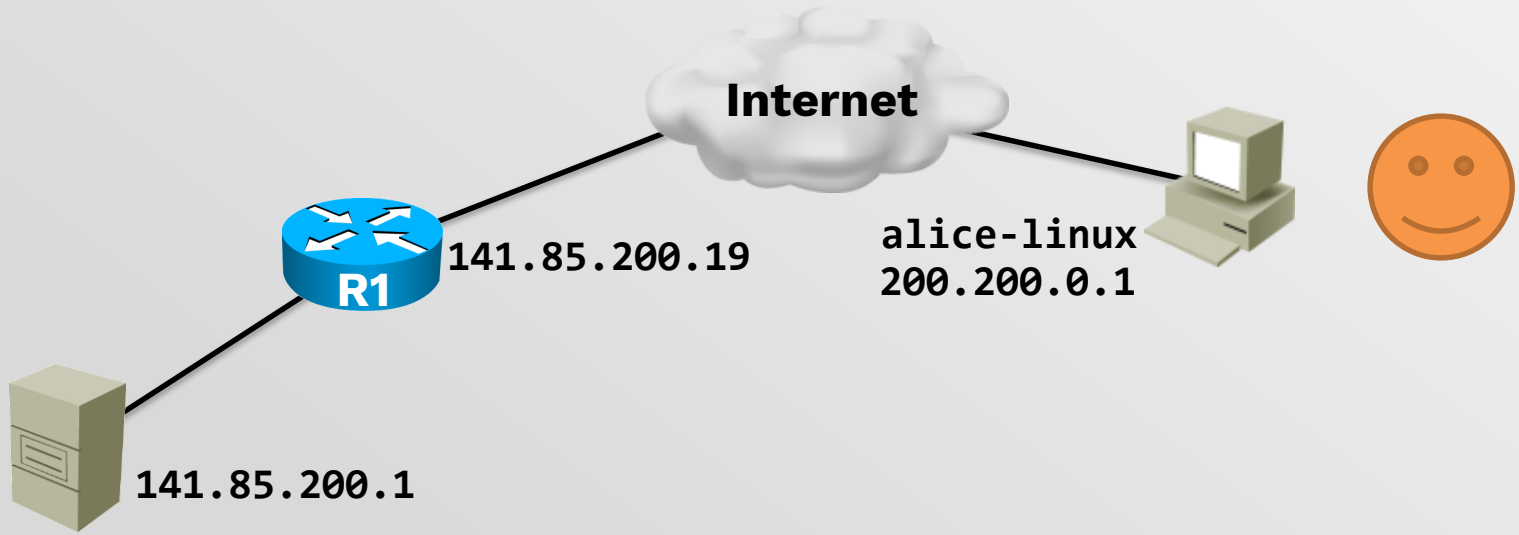
1.		Payload protocol: IPv6
2.	Delivery protocol: IPv4	Payload protocol: IPv6
3.		Payload protocol: IPv6

Tunel SSH

Tunel SSH	
Delivery protocol:	SSH
Payload protocol:	Protocoale de nivel 4
Nivel OSI:	7
Funcție:	Folosit pentru transportul securizat al traficului (integritate, autentificare, confienționalitate)

Tunel SSH

- Utilizatorul Alice are un cont pe ruterul R1
- R1 este de fapt o mașină Linux ce are SSH instalat
- Serverul este vechi și nu permite instalarea de SSH
- Alice vrea ca traficul său să fie criptat peste Internet, dar totuși să poată controla prin Telnet serverul



Tunel SSH

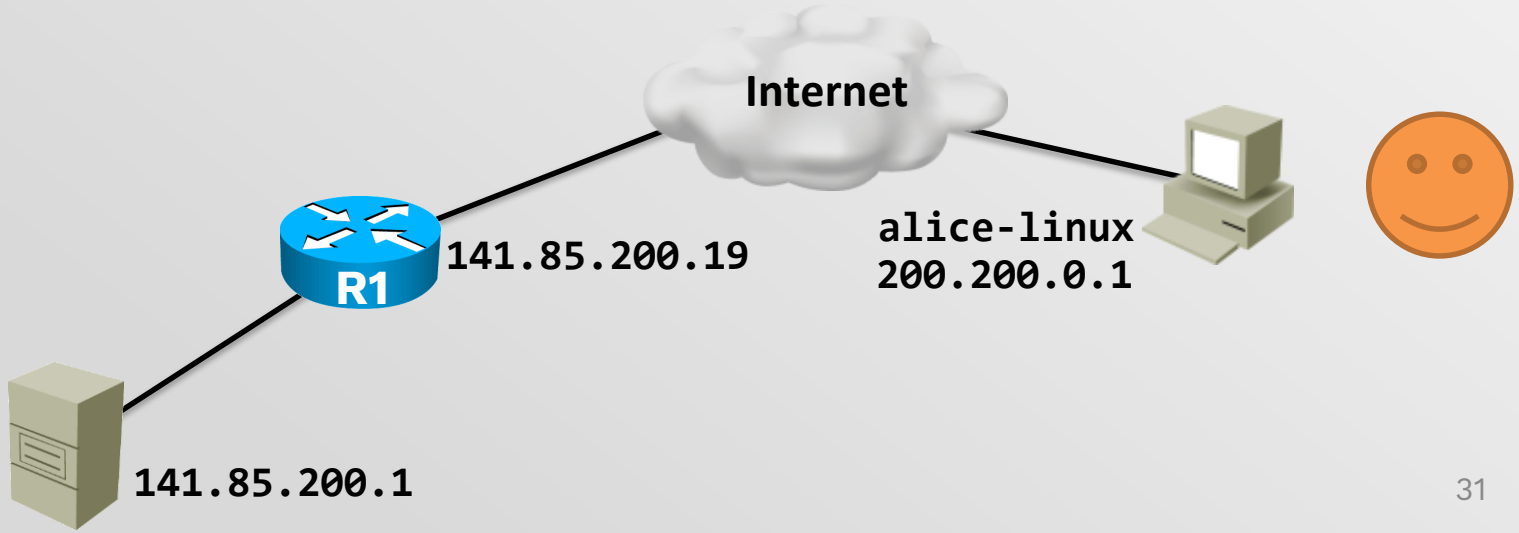
- Soluția este crearea unui tunel SSH

```
alice-linux# ssh -N -L 0.0.0.0:5000:141.85.200.1:23 alice@141.85.200.19
```

Utilizatorul și adresa pentru conectare

Local - Tunelul va avea punctul de intrare pe mașina locală

Nu se deschide un shell pe mașina intermediară



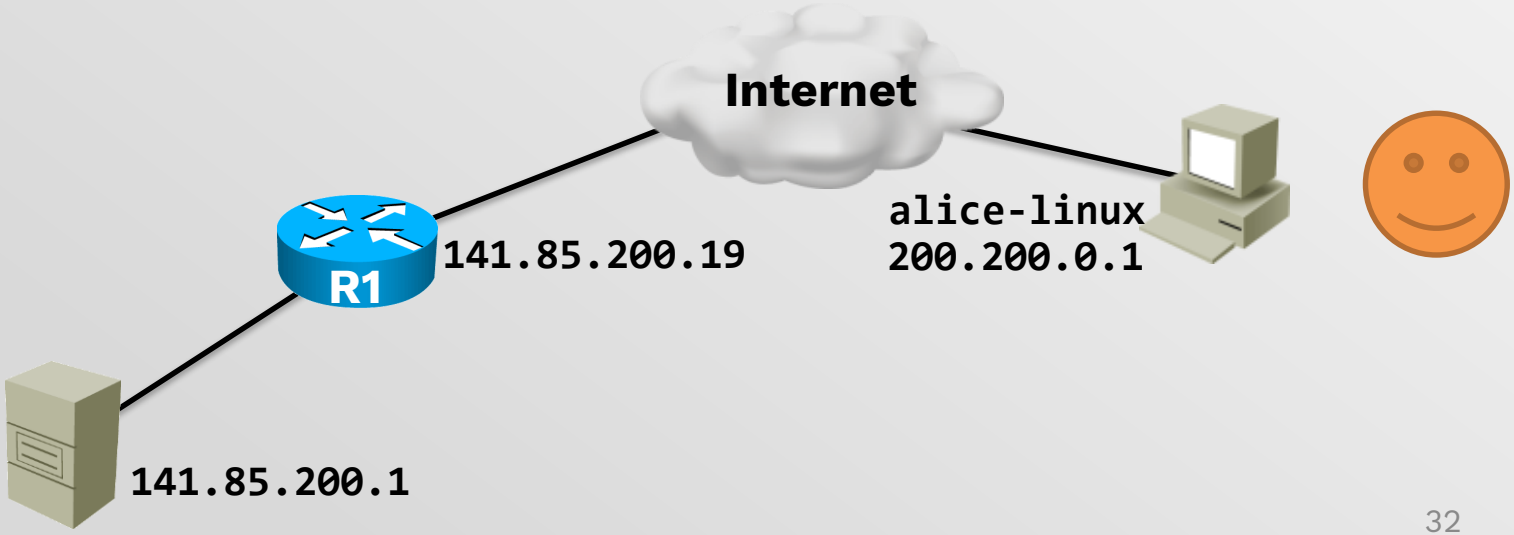
Tunel SSH

```
alice-linux# ssh -N -L 0.0.0.0:5000:141.85.200.1:23 alice@141.85.200.19
```

Adresa și portul destinației

Portul pe care se ascultă stația locală

Bind address – adresele locale pe care se va asculta

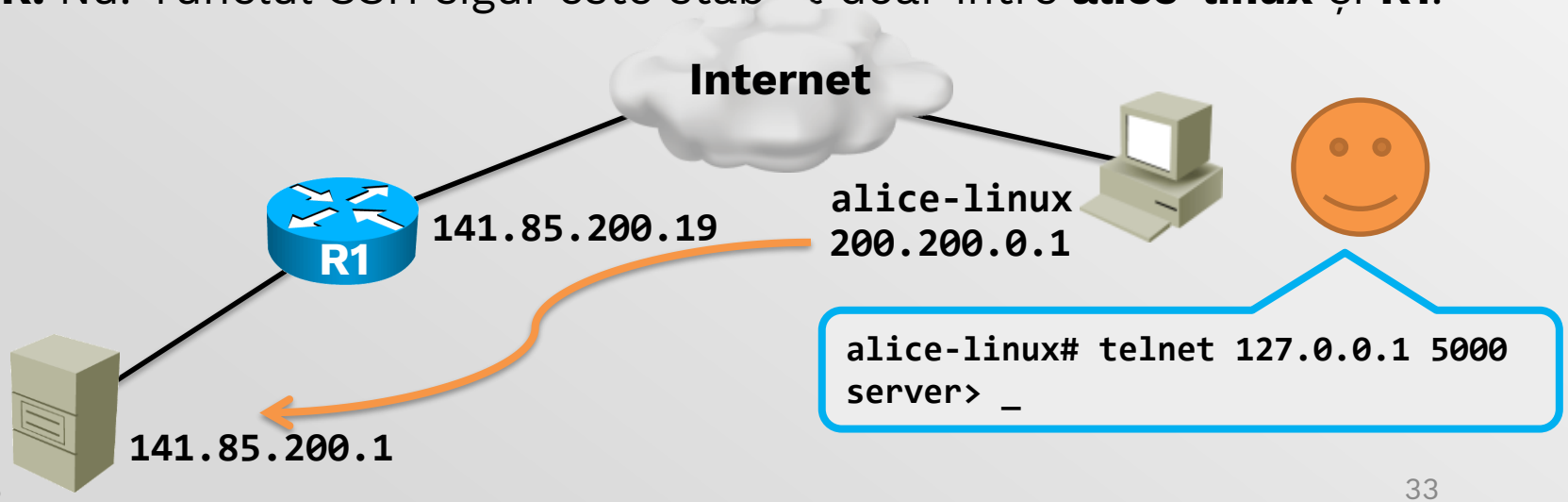


Tunel SSH

- În urma comenzii pe **alice-linux** se deschide portul 5000

```
alice-linux# ssh -N -L 0.0.0.0:5000:141.85.200.1:23 alice@141.85.200.19
```

- Tot traficul primit pe portul 5000 este redirectat către serverul de SSH de pe **R1**
- **R1** redirectează traficul către destinație (**Server**)
- Este traficul între **R1** și **Server** criptat?
  - **R:** Nu. Tunelul SSH sigur este stabilit doar între **alice-linux** și **R1**.



Tunel 6to4

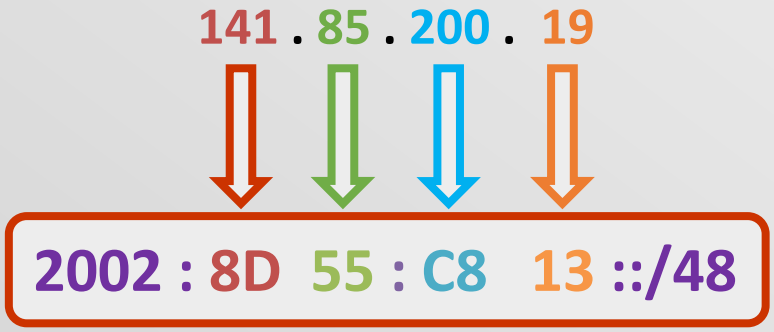
Tunel 6to4	
Delivery protocol:	IP
Payload protocol:	IPv6
Nivel OSI:	3
Funcție:	Folosit pentru migrarea către IPv6

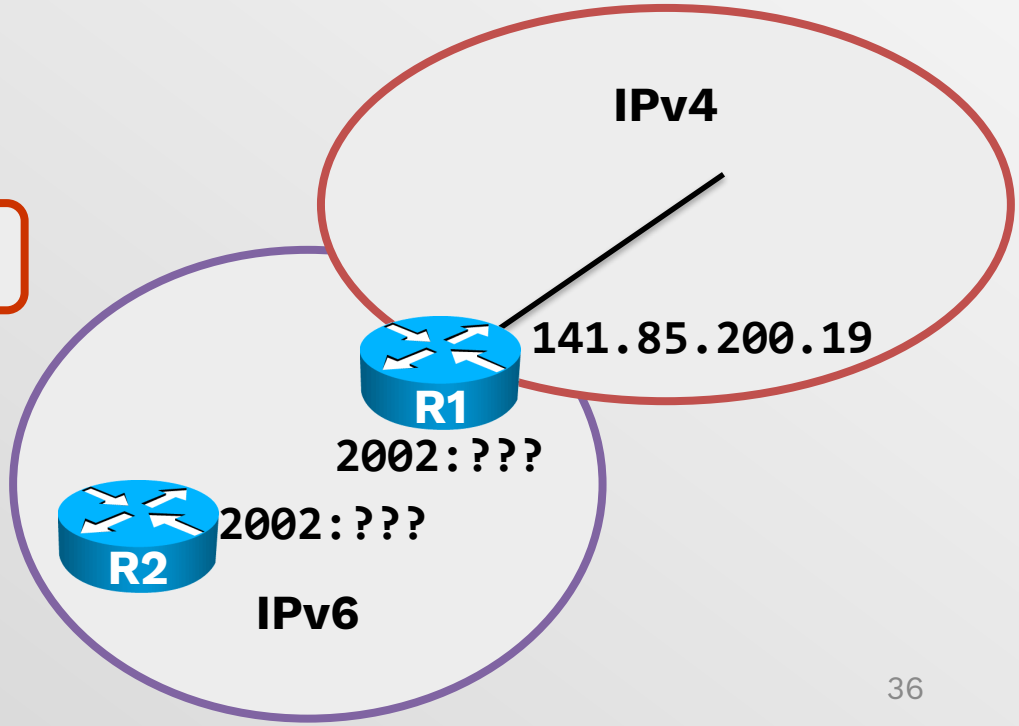
Tunel 6to4

- Migrarea de la IPv4 la IPv6 are loc treptat
  - Insule IPv6
  - Backbone IPv4
- Pentru comunicare este necesară tunelarea traficului IPv6
- Două soluții:
  - Tunele statice
    - Dezavantaje: greu de administrat, trebuie configurate, pot fi introduse erori
  - Tunele automate
    - Ușor de administrat
    - Se construiesc automat când sunt necesare

Tunel 6to4

- Adresele IPv6 trebuie să fie din rețeaua **2002::/16**
- Următorii 32 de biți sunt luați din adresa IPv4 de la ieșirea insulei IPv6





Tunel 6to4

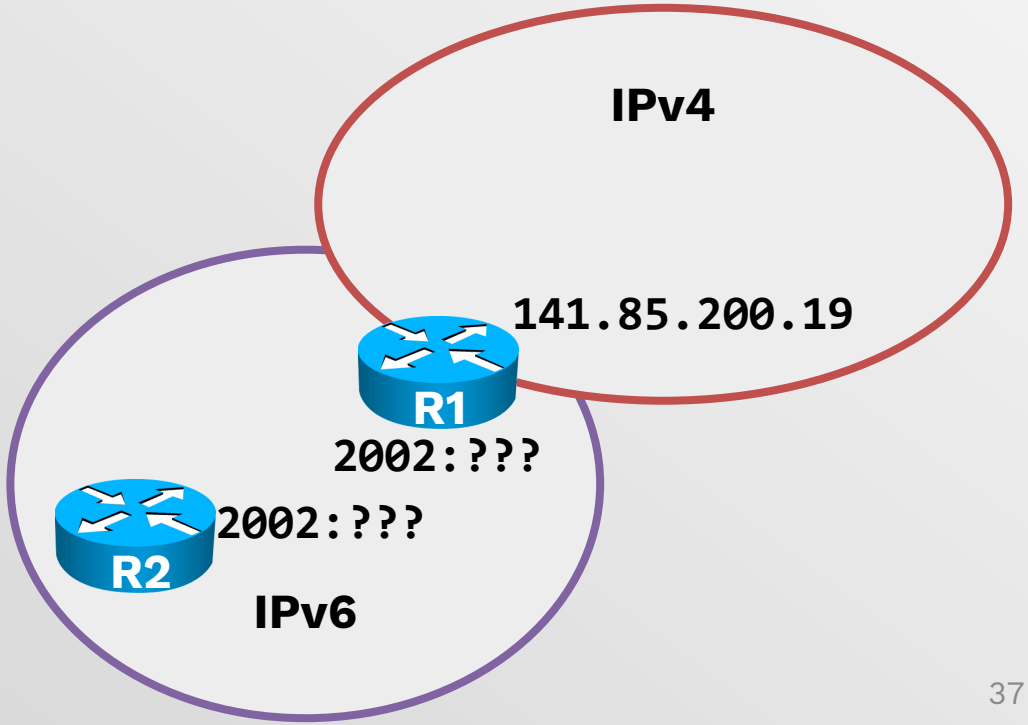
R1: 2002:8D55:C813::/48

R2: 2002:8D55:C813::/48

- Ultimii 16 biți din partea de rețea → subnetting

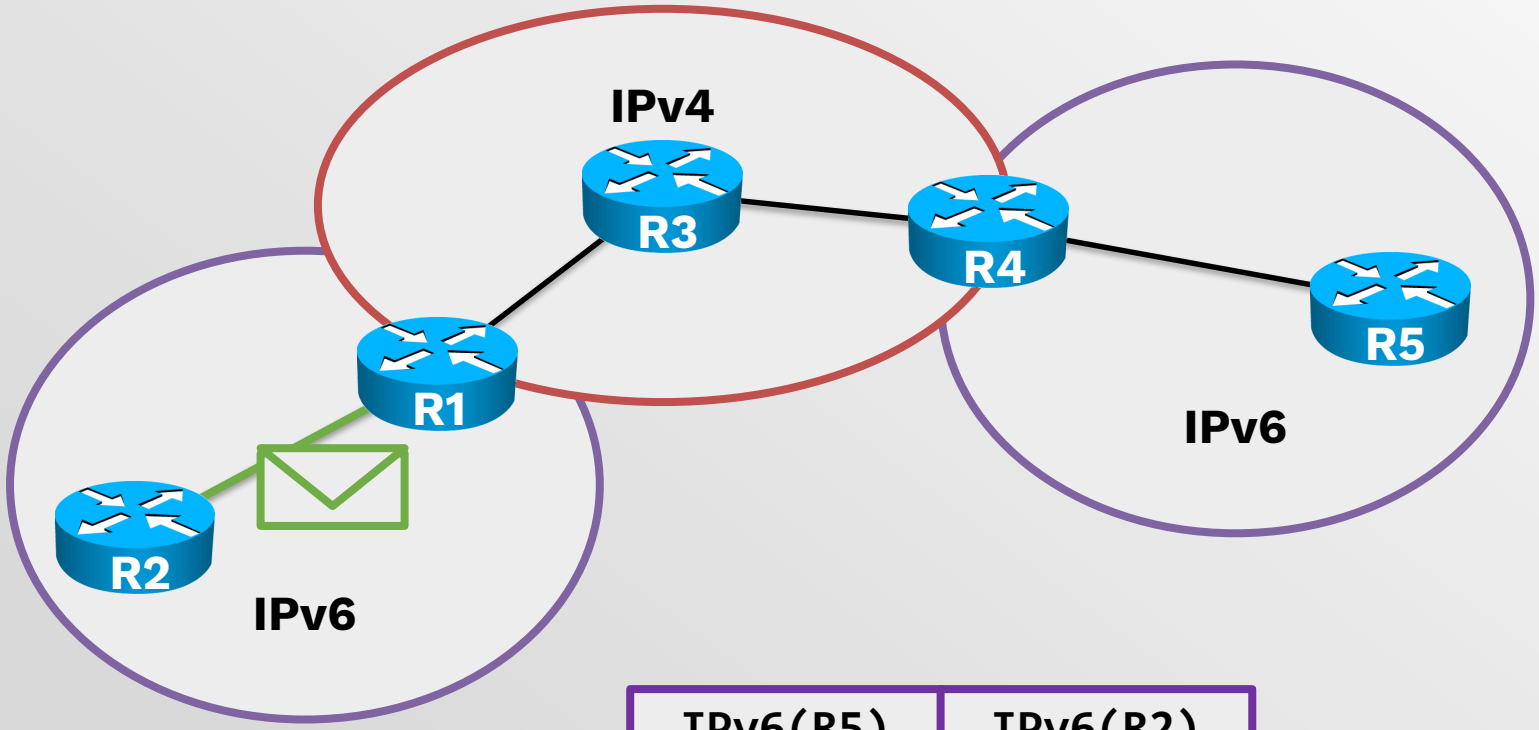
R1: 2002:8D55:C813::1/64

R2: 2002:8D55:C813::2/64



Tunel 6to4

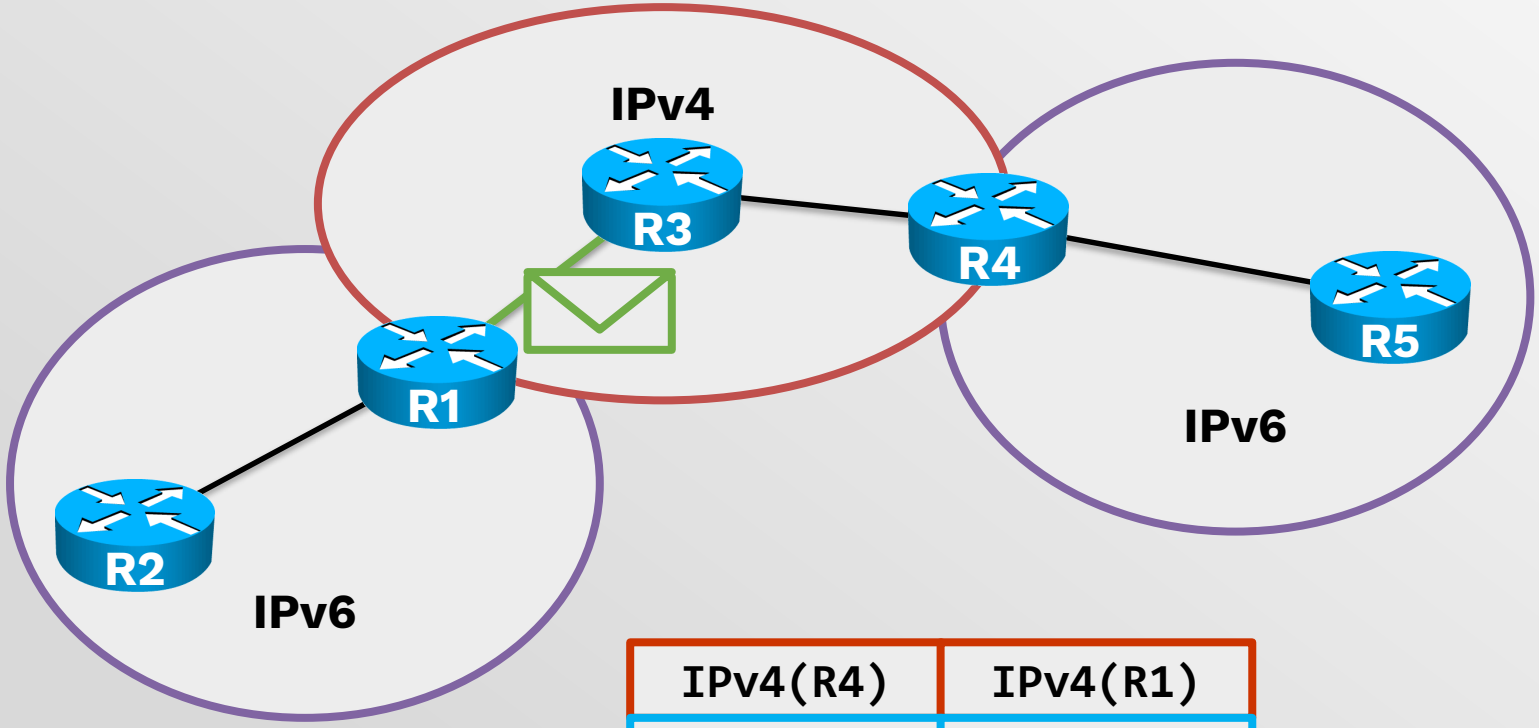
- **R2** vrea să comunice cu **R5**



IPv6(R5)	IPv6(R2)
Destinație	Sursă

Tunel 6to4

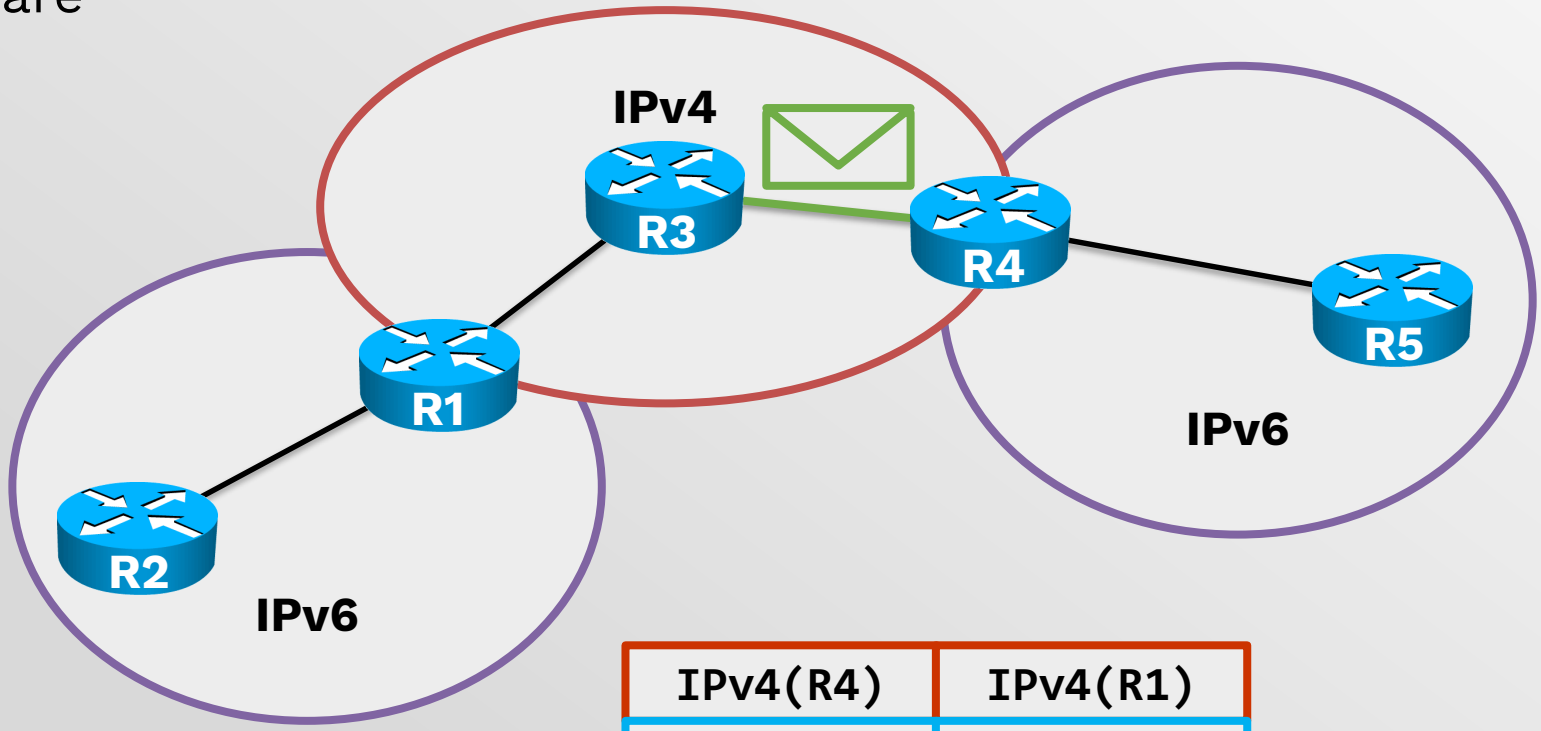
- **R1** primește pachetul și îl încapsulează într-un pachet IPv4
- Adresele IPv4 sunt obținute din biții 17-48 din adresele IPv6



IPv4(R4)	IPv4(R1)
Destinație	Sursă

Tunel 6to4

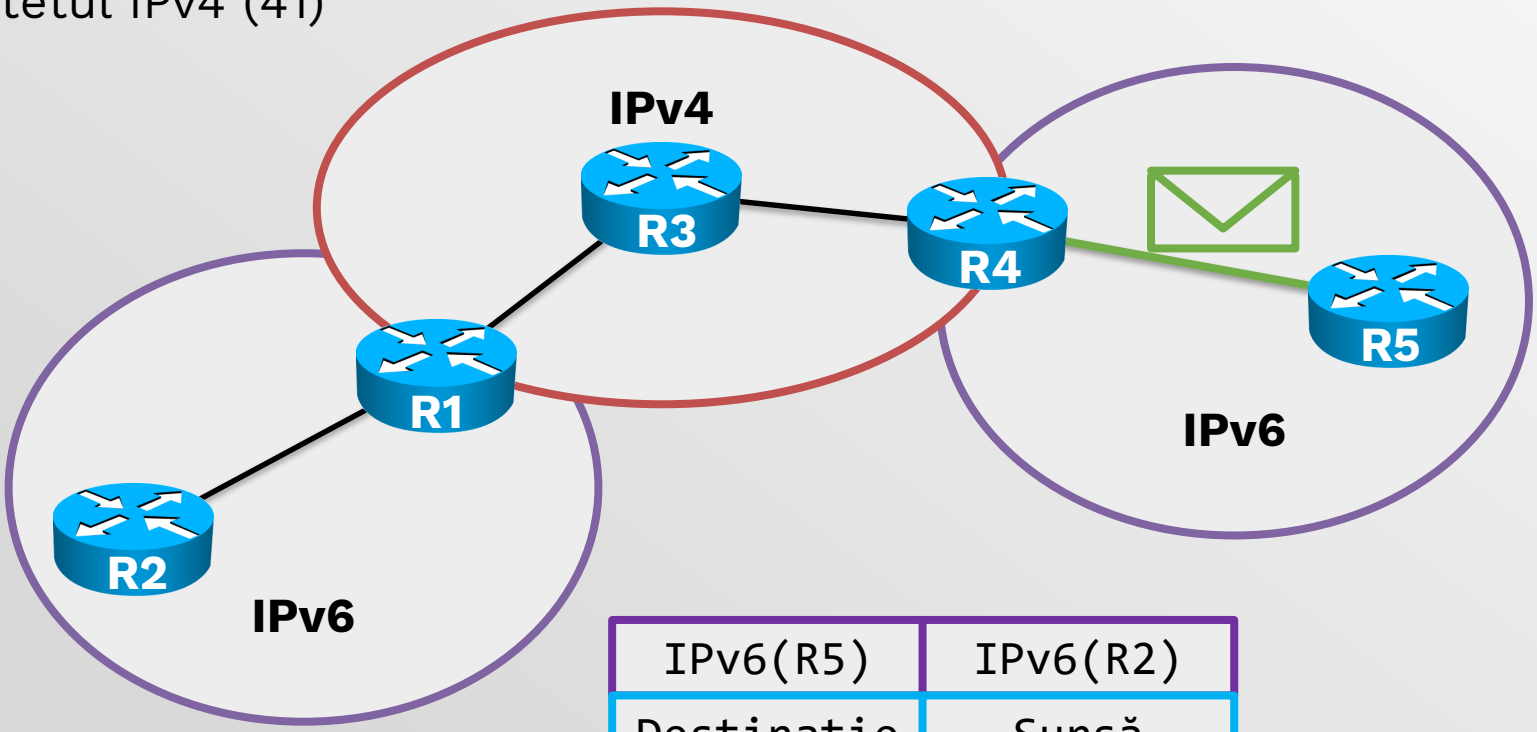
- **R3** nu cunoaște nimic despre rețelele IPv6
- Întrucât destinația e IPv4 se efectuează un proces normal de rutare



IPv4(R4)	IPv4(R1)
Destinație	Sursă

Tunel 6to4

- **R4** este capăt de tunel și decapsulează antetul IPv6
- **R4** știe că pachetul este destinat IPv6 din câmpul de protocol din antetul IPv4 (41)



IPv6(R5)	IPv6(R2)
Destinație	Sursă

Tunel L2TP

Tunel L2TP	
Delivery protocol:	UDP
Payload protocol:	PPP, ATM, Frame Relay
Nivel OSI:	2
Funcție:	Folosit pentru transportul peste infrastructuri IP al conexiunilor PPP

Tunel Teredo

Tunel Teredo	
Delivery protocol:	UDP
Payload protocol:	IPv6
Nivel OSI:	3
Funcție:	Folosit pentru transportul peste infrastructuri IP al traficului IPv6

Cuvinte cheie