

## 13. Статична и централизирана маршрутизация

# Централизирани адаптивни алгоритми

При централизираните адаптивни алгоритми в мрежата се създава един маршрутен управляващ център.

Той изчислява маршрутните таблици на всички възли и им ги изпраща.

За да се адаптират маршрутните таблици към текущата топология и текущия трафик, всички възли трябва да изпращат информация към маршрутния център.

На базата на получените сведения, маршрутният център изчислява теглата на ребрата и след това пресмята оптималният маршрут между всеки два възела.

Добре е да се поддържат алтернативни пътища между възлите.

# Централизирани адаптивни алгоритми

Информацията от по-близките до маршрутния център възли ще пристигне по-бързо отколкото от по-далечните.

Поради това **периодът на обновяване** на маршрутните таблици трябва да е поне два пъти по-голям от времето за преминаване на пакет от маршрутния център до най-отдалечения от него възел.

Преизчислена маршрутна таблица, получена в един възел, не трябва да се използва веднага, тъй като маршрутните таблици пристигат по различно време в различните възли.

# Централизирани адаптивни алгоритми

Ако по някаква причина маршрутният център отпадне, мрежата остава без управление.

За целта може да се дублира маршрутният център, но тогава служебният трафик би се увеличил твърде много.

# ...ОТНОВО централизирана. SDN и OpenFlow

Системите преминаха:

- централизирани - големи машини (mainframes);
- разпределени – PCs;
- пак централизирани – VMs, Cloud.

Същото и с мрежите. Днес **Cloud Networking Services (CNS)**:

- SaaS и/или
- IaaS и/или NaaS.

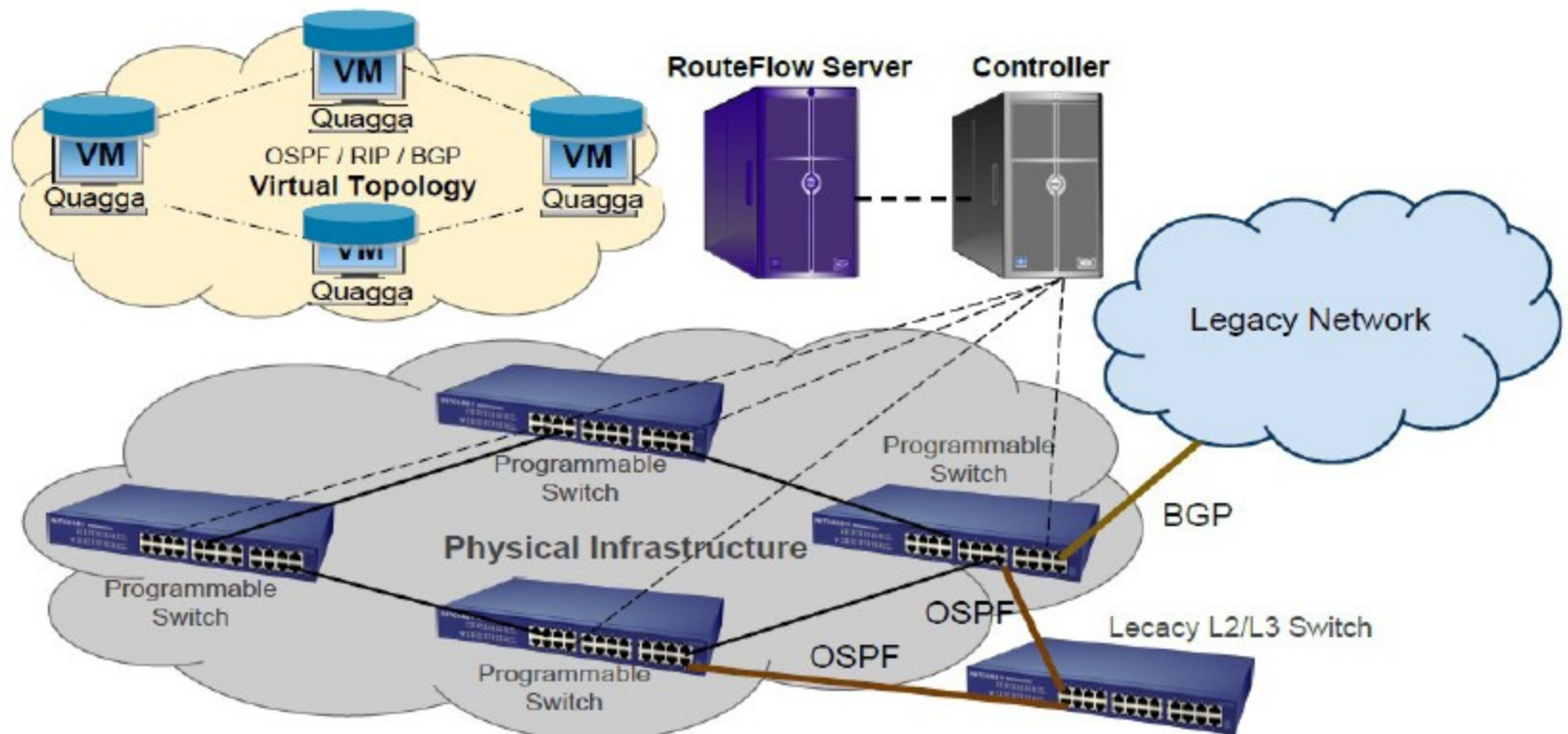
# Проектът OpenFlow ([www.openflow.org](http://www.openflow.org))



**OpenFlow** е в основата на software-defined networking (**SDN**). Потребителите дефинират потоците данни и пътищата им независимо от инфраструктурата под тях – маршрутизатори и комутатори.

Проект с **отворен код** – сътрудничество между Stanford University и University of California at Berkeley.

# RouteFlow (QuagFlow)



# RouteFlow (QuagFlow)

Параметри:

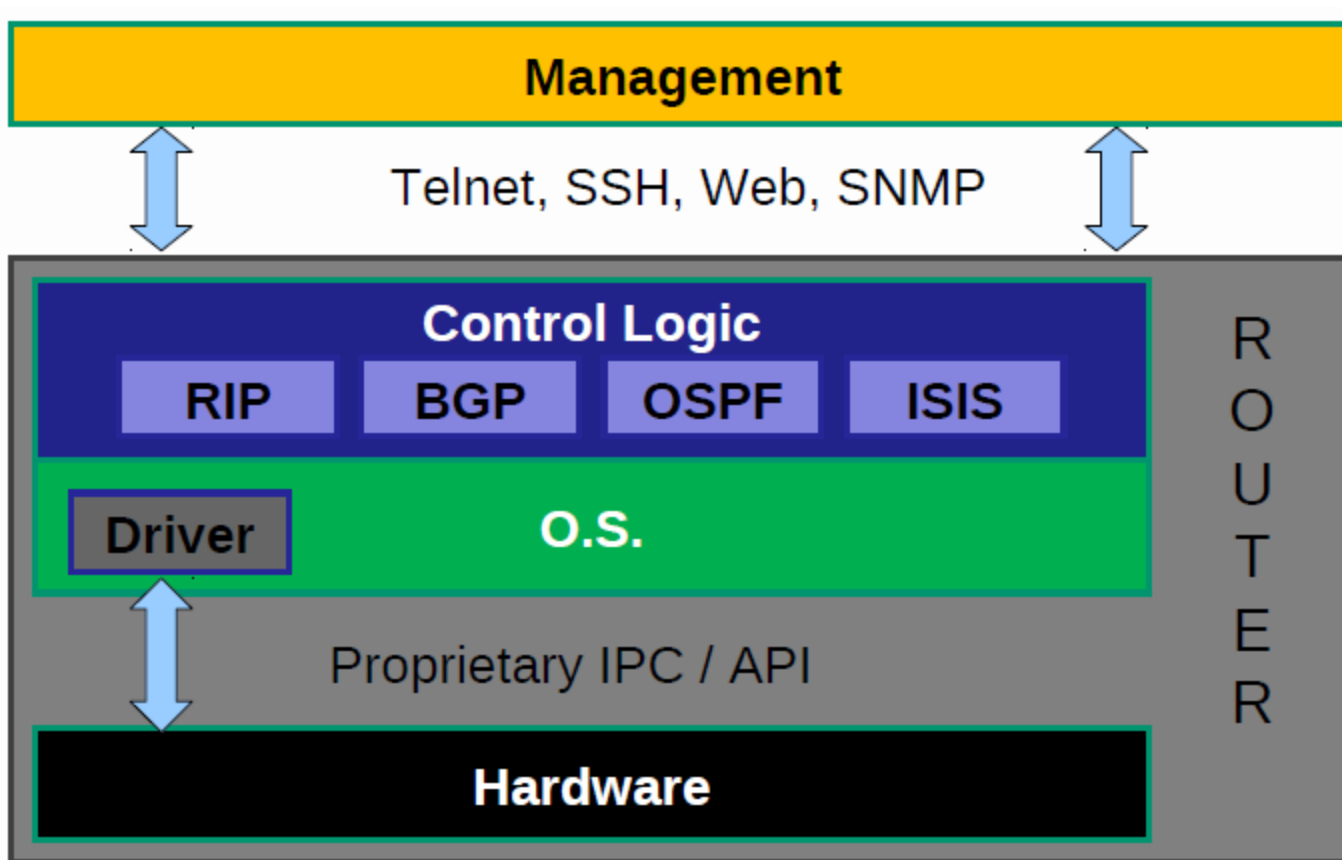
- **open-source** протоколни стекове (напр. **Quagga**)
- комерсиален мрежов хардуер с отворени API-та (напр. OpenFlow)

Постига се:

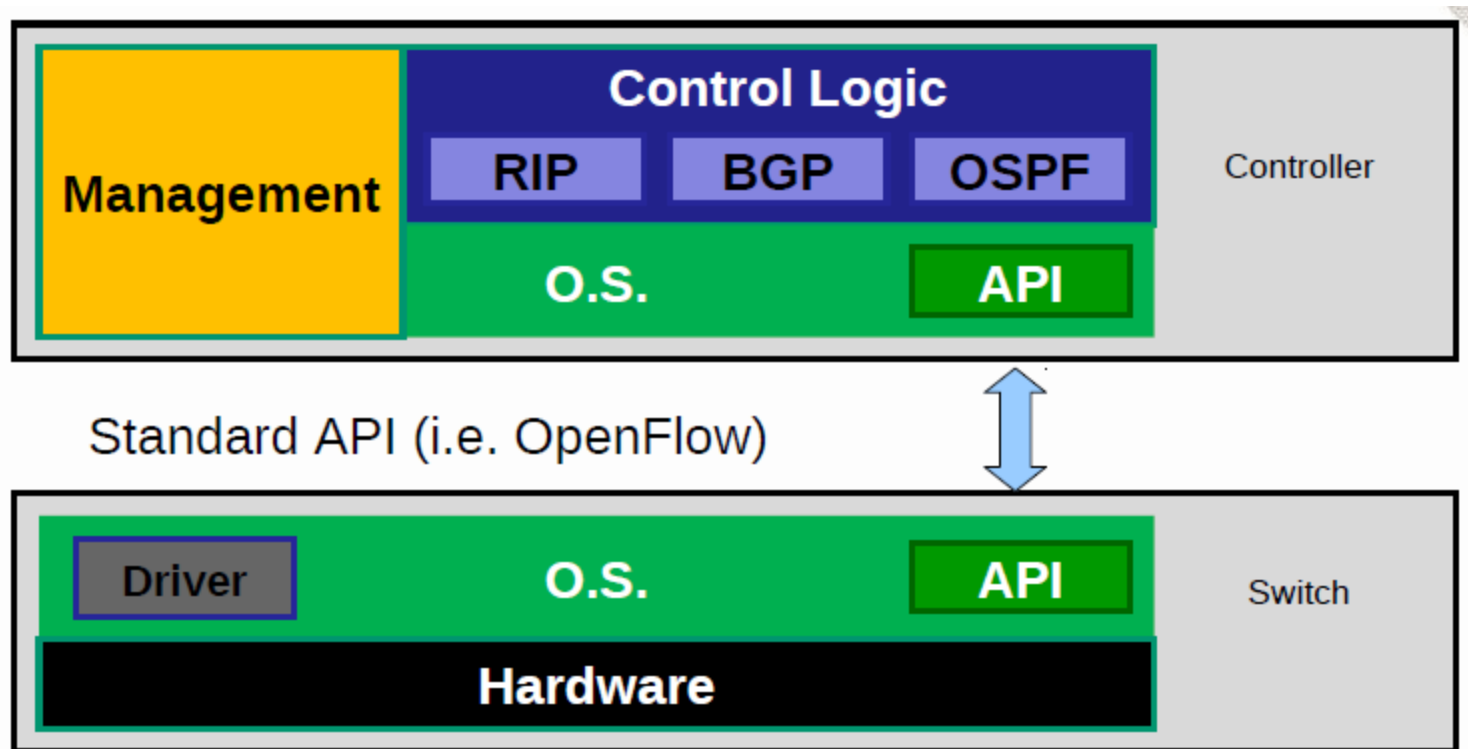
- производителност, съизмерима със скоростта на линията;
- оптимално съотношение цена/производителност;
- гъвкавост.



# Архитектура на маршрутизатор (сега)



# OpenFlow модел



# Към IP Network-as-a-Service

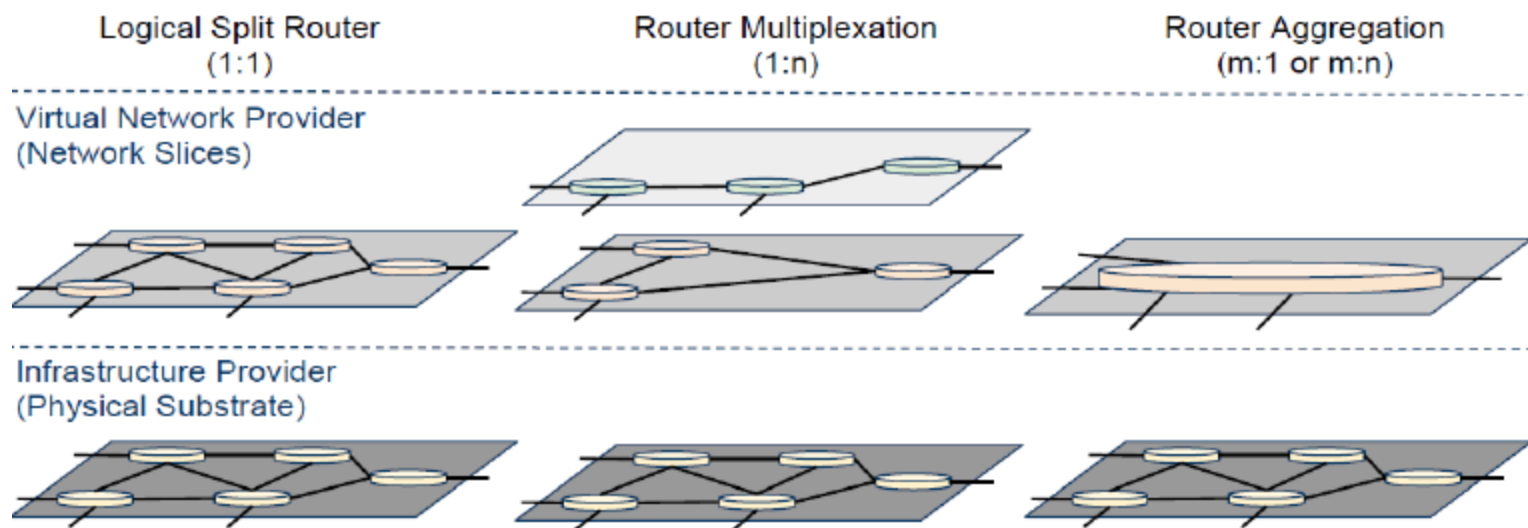
Постепенен преход от днешните IP мрежи към SDN:

- Хибридни модели: традиционни IP мрежи съвместно с SDN
- Опростяване на управлението и оптимизиране на IP мрежите

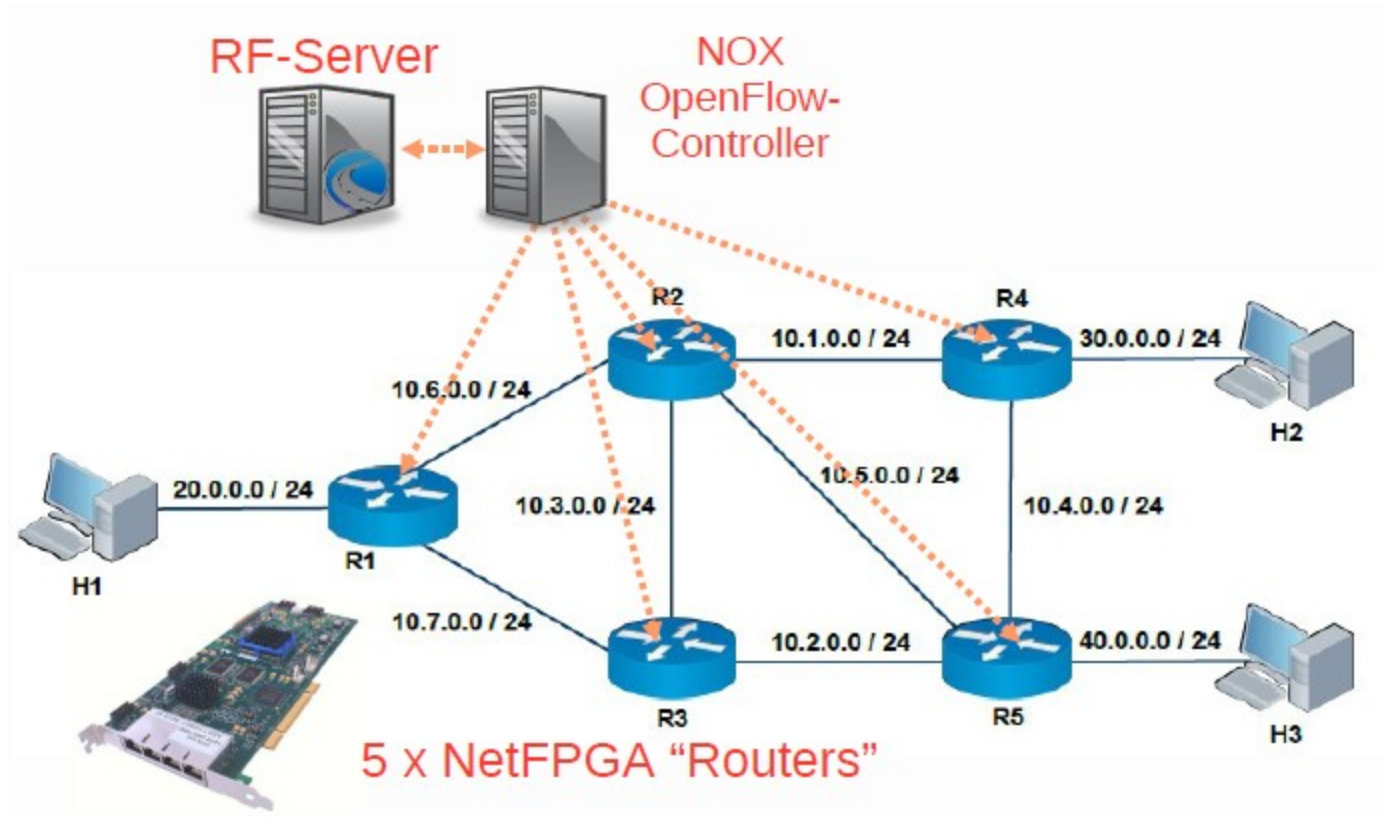
Напред към IP Network Virtualization:

- От Virtual Routers към IP Network-as-a-Service (NaaS)

# Виртуализация

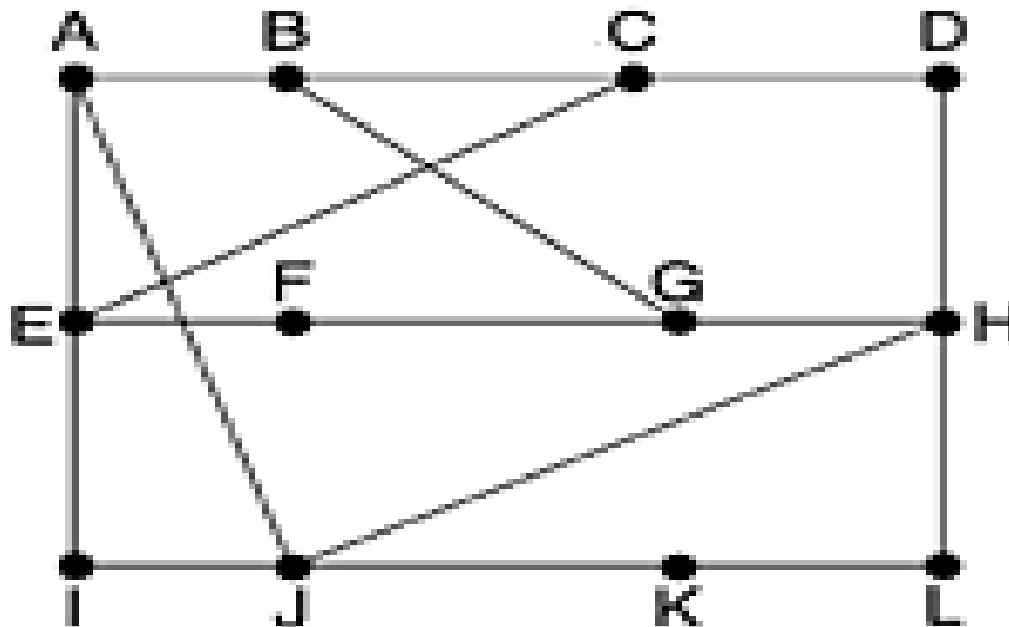


# Тестова постановка (OSPF мрежа)



# Пример на статична маршрутизация

Нека е дадена следната мрежа.



# Пример на статична маршрутизация

Маршрутната таблица на възела J би могла да има по следния вид:

|   |    |      |    |      |    |      |
|---|----|------|----|------|----|------|
| A | A  | 0.63 | I  | 0.21 | H  | 0.16 |
| B | A  | 0.46 | H  | 0.31 | I  | 0.23 |
| C | A  | 0.34 | I  | 0.33 | H  | 0.33 |
| D | H  | 0.50 | A  | 0.25 | I  | 0.25 |
| E | A  | 0.40 | I  | 0.40 | H  | 0.20 |
| F | A  | 0.34 | H  | 0.33 | I  | 0.33 |
| G | H  | 0.46 | A  | 0.31 | K  | 0.23 |
| H | H  | 0.63 | K  | 0.21 | A  | 0.16 |
| I | I  | 0.65 | A  | 0.22 | H  | 0.13 |
| J | -- | --   | -- | --   | -- | --   |
| K | K  | 0.67 | H  | 0.22 | A  | 0.11 |
| L | K  | 0.42 | H  | 0.42 | A  | 0.16 |

# Пример на статична маршрутизация

Първата колона показва дестинацията на пакета. Нека, например, в J пристигне пакет за A. Тогава J генерира случайно число между 0 и 1.

Ако числото е  $\leq 0.63$ , пакетът се изпраща към A, ако числото е  $> 0.63$  и  $\leq 0.84$  пакетът се изпраща към I,

ако числото е  $> 0.84$ , пакетът се изпраща към H.



# Статична маршрутизация. По-подробно.

Ръчното добавяне на маршрути в конфигурацията на маршрутизатор се нарича **статична маршрутизация (static routing)**. Подходящо е за малки мрежи. Маршрутите се описват чрез фиксирани пътища (**статични маршрути**), които се въвеждат в маршрутизатора от мрежовия администратор.

# Статична маршрутизация. По-подробно.

Така чрез статични маршрути (**static routes**) може да се опише цялата мрежа.

Този подход не е отказоустойчив. В случай на промени или прекъсвания на връзки и/или устройства не се получава автоматично пренасочване на трафика.

# Статична маршрутизация

В някои случаи статичните маршрути са даже за предпочитане, даже влияят положително на производителността.

Това са мрежите с един единствен изход (**stub networks**) и маршрутите по подразбиране (**default routes**).

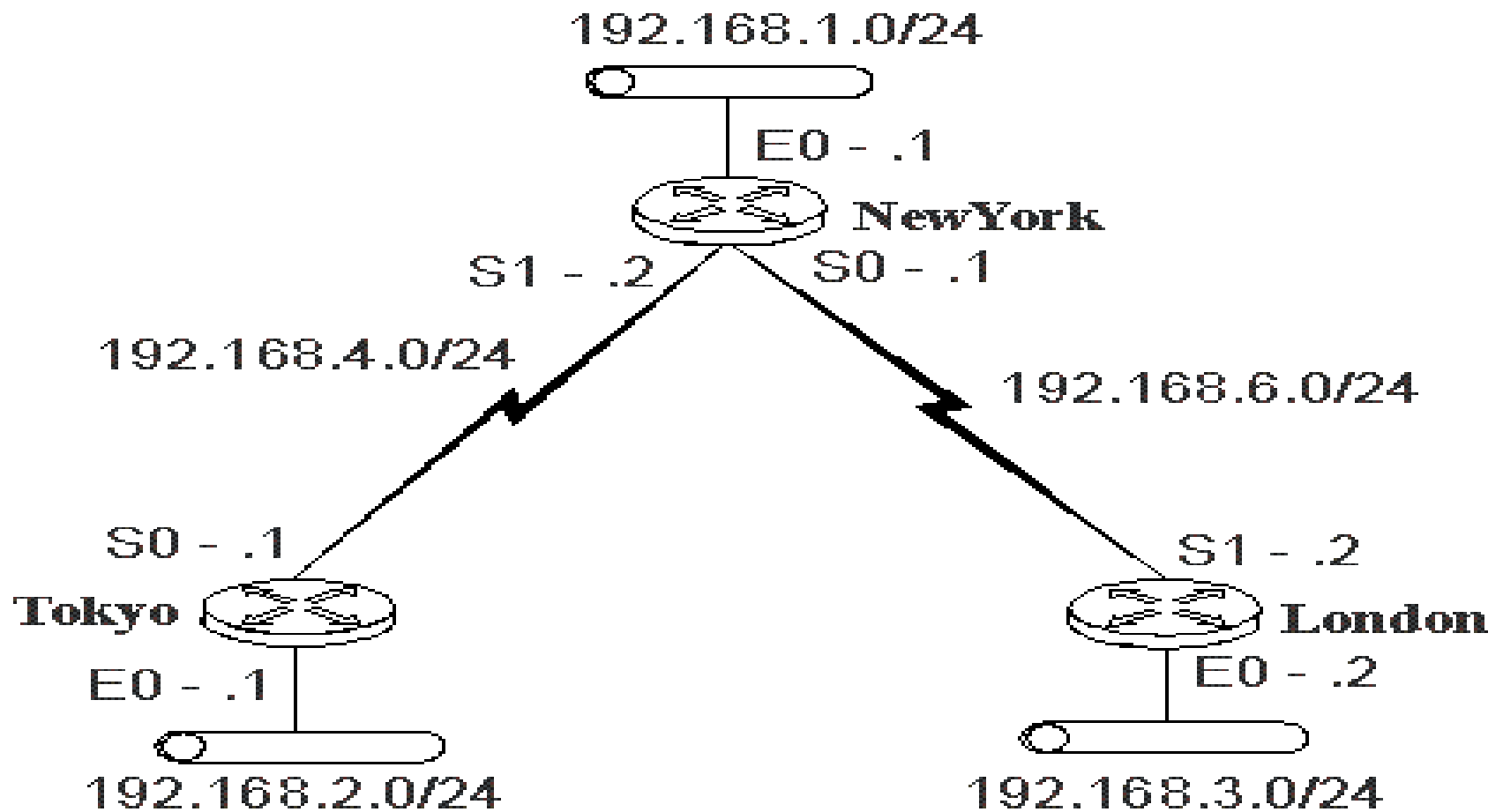
# Stub Networks

Статичните маршрути най-добре се вписват в **Stub** мрежите.

**stub network** е като “задънена улица” – само с един изход към външния свят.

На фигурата по-долу двете клонови локални мрежи - Tokyo и London са с по един изход към главната квартира.

# Stub Networks



# Stub Networks

Не е необходимо да пускаме динамичен протокол за маршрутизация по WAN линиите между NewYork и клоновете.

Статични маршрути отвеждат трафика до “клоновете” LAN-ве.

Докато клоновите маршрутизатори е необходимо да “знаят” само дали пакета е насочен към мрежа извън техните локални.

Т.е да го “докарат” до NewYork, който има по-богата маршрутна таблица.

# Конфигурации

По-долу са дадени статичните маршрути в NewYork и Tokyo.

Редове 10 и 11 са статичните маршрути в NewYork към съответните “клонови LAN-ве”.

Ред 19 е по подразбиране (default route) в Tokyo (маршрут до мрежа 0.0.0.0), който сочи обратно пътя към NewYork.

Конфигурацията на London е подобна на Tokyo.

# Конфигурации: static routes

## NewYork Configuration

...

10) ip route 192.168.2.0 255.255.255.0  
192.168.4.1

11) ip route 192.168.3.0 255.255.255.0  
192.168.6.2

!!! В конфигурацията освен мрежата и маската е зададен и IP адреса следващия възел по пътя (**Next Hop**)



# Конфигурации: default route

Tokyo Configuration

...

```
19) ip route 0.0.0.0 0.0.0.0 192.168.4.2
```

# Частен случай: интерфейсни маршрути

Статичен маршрут, на който е зададен само **изходящия интерфейс** на рутера, се нарича още **интерфейсен**.

В този случай все едно, че мрежата на крайната точка е директно закачена за интерфейс на рутер.

# Интерфейсни маршрути

```
NewYork#conf t
```

```
NewYork(config)#no ip route 192.168.2.0  
255.255.255.0 192.168.4.1
```

```
NewYork(config)#no ip route 192.168.3.0  
255.255.255.0 192.168.6.2
```

```
NewYork(config)#ip route 192.168.2.0  
255.255.255.0 s 1
```

```
NewYork(config)#ip route 192.168.3.0  
255.255.255.0 s 0
```

# Интерфейсни маршрути

NewYork#sh ip route

...

C 192.168.4.0/24 is directly connected, Serial1

C 192.168.6.0/24 is directly connected, Serial0

C 192.168.1.0/24 is directly connected, Ethernet0

S 192.168.2.0/24 is directly connected, Serial1

S 192.168.3.0/24 is directly connected, Serial0

# Default Route

Маршрут по подрабиране (**default route**) се използва, когато в таблицата няма друг маршрут до целта.

Това е маршрут до мрежата 0.0.0.0. Изходът от командата **show ip route** показва на първия ред на маршрутната таблица:

**Gateway of last resort is 192.168.4.1 to network 0.0.0.0**

Пакет, който не намери съвпадение в маршрутната таблица, поема към “**gateway of last resort**”.

Това е рутер с по-подробна информация за маршрутите.

Ако няма default route и адреса на получателя не бъде открит в таблицата, пакетът се изхвърля и на IP адреса на източника се въща ICMP съобщение: ‘**Destination or Network Unreachable**’.

# Терминът Gateway

‘Gateway’ е по-стария термин за маршрутизатор (рутер). През него можеше да се изпратят пакети към мрежа с различна преносна среда и канални протоколи.

В днешно време показва IP адреса на устройството, от което се излиза от локалната мрежа към “вЪНШНИЯ СВЯТ”. И обикновено е с два интерфейса – вътрешен и външен.

Докато маршрутизаторът има повече от два интерфейса и изпълнява по-сложни функции по маршрутизацията.

# Default Route

Маршрутът по подразбиране е много полезен в “stub” мрежите.

Благодарение на него не се налага да се използва динамичен протокол за маршрутизация, което не е и необходимо в такива мрежи.

Така се спестява пропускателна способност на линиите, “цикли” на CPU на рутера и памет.

# Default Route. Конфигуриране.

```
Tokyo(config)#ip route 0.0.0.0 0.0.0.0 192.168.4.2
```

...

```
Tokyo#sh ip route
```

...

```
Gateway of last resort is 192.168.4.2 to network  
0.0.0.0
```

```
C 192.168.4.0/24 is directly connected, Serial0
```

```
C 192.168.2.0/24 is directly connected, Ethernet0
```

```
S* 0.0.0.0/0 [1/0] via 192.168.4.2
```



# ip route

```
[root@shuttle ~]# ip route
```

```
62.44.109.0/26 dev eth0 proto
```

```
kernel scope link src
```

```
62.44.109.11
```

```
169.254.0.0/16 dev eth0 scope
```

```
link
```

```
default via 62.44.109.5 dev eth0
```

# ip -6 route

```
[root@shuttle ~]# ip -6 route
```

```
...
```

```
fe80::/64 dev eth0 metric 256  
expires 21207257sec mtu 1500  
advms 1440 hoplimit 4294967295  
default via 2001:67c:20d0:10::5  
dev eth0 metric 1 expires  
21207261sec mtu 1500 advms 1440  
hoplimit 4294967295
```

# telnet localhost zebra

```
zebra@rec-gw> en
```

```
zebra@rec-gw# sh run
```

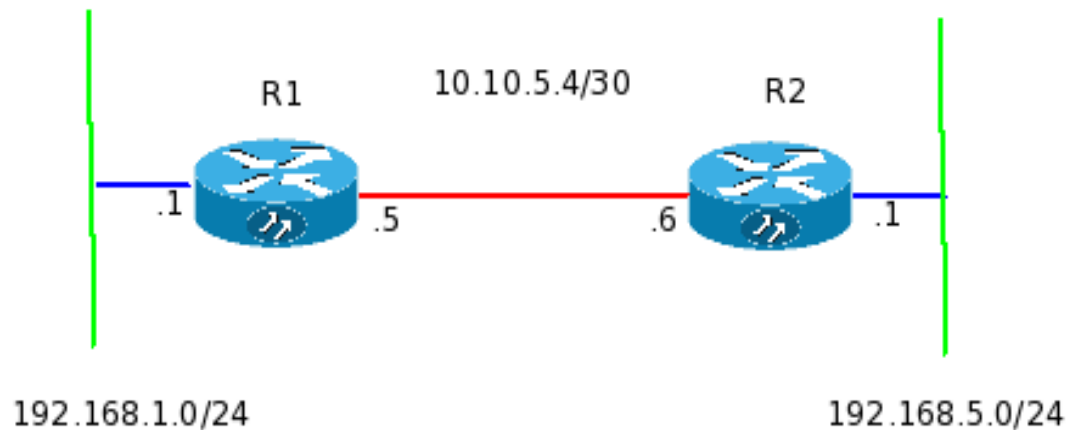
```
...
```

```
ip route 62.44.116.0/24 62.44.110.14
```

```
ip route 62.44.117.0/25 62.44.110.9
```

```
ipv6 route 2001:67c:20d1:116::/64  
2001:67c:20d1:110::14
```

# Задача



- На R1 създайте статичен маршрут до 192.168.5.0/24.
- Защо `ping 192.168.5.1` работи, а
- `ping -I 192.168.1.1 192.168.5.1` не работи?