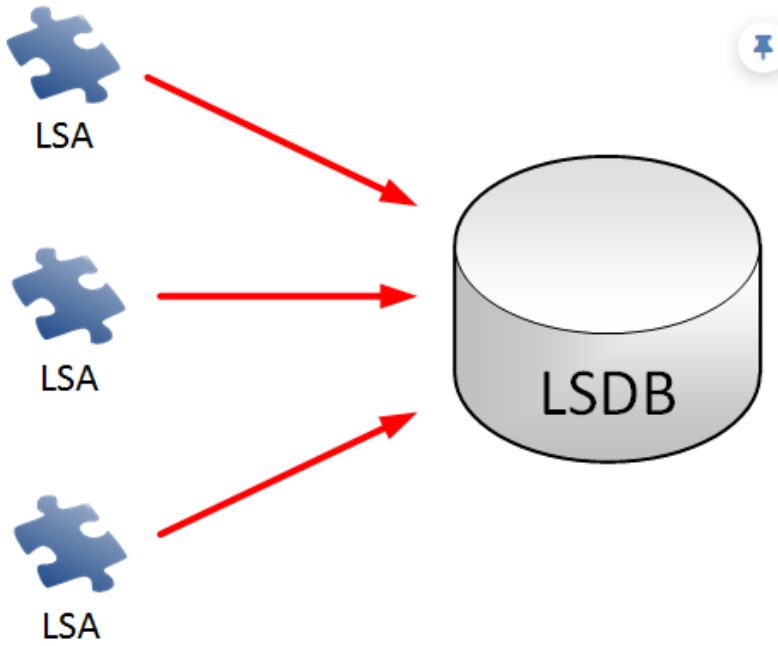
OSPFv3

OSPF link-State როუტინგ პროტოკოლი

Ospf როუტინგ პროტოკოლი მოიცავს მთელი ქსელის რუკას, და ამ რუკის საშუალებით ის არჩევს საუკეთესო გზას დანიშნულების წერტილამდე და გადააქვს ის როუტინგ ცხრილში.

თითოეულ როუტერს აქვს მთლიანი ქსელის ინფორმაცია, თუმცა მასაც აქვს უარყოფითი მხარე რადგან **დიდი ქსელებში როუტერის პროცესორის დატვირთვა** საკმაოდ იზრდება.

OSPF link-State როუტინგ პროტოკოლი



როუტერები ამყარებენ ერთმანეთთან მეზობლობას.

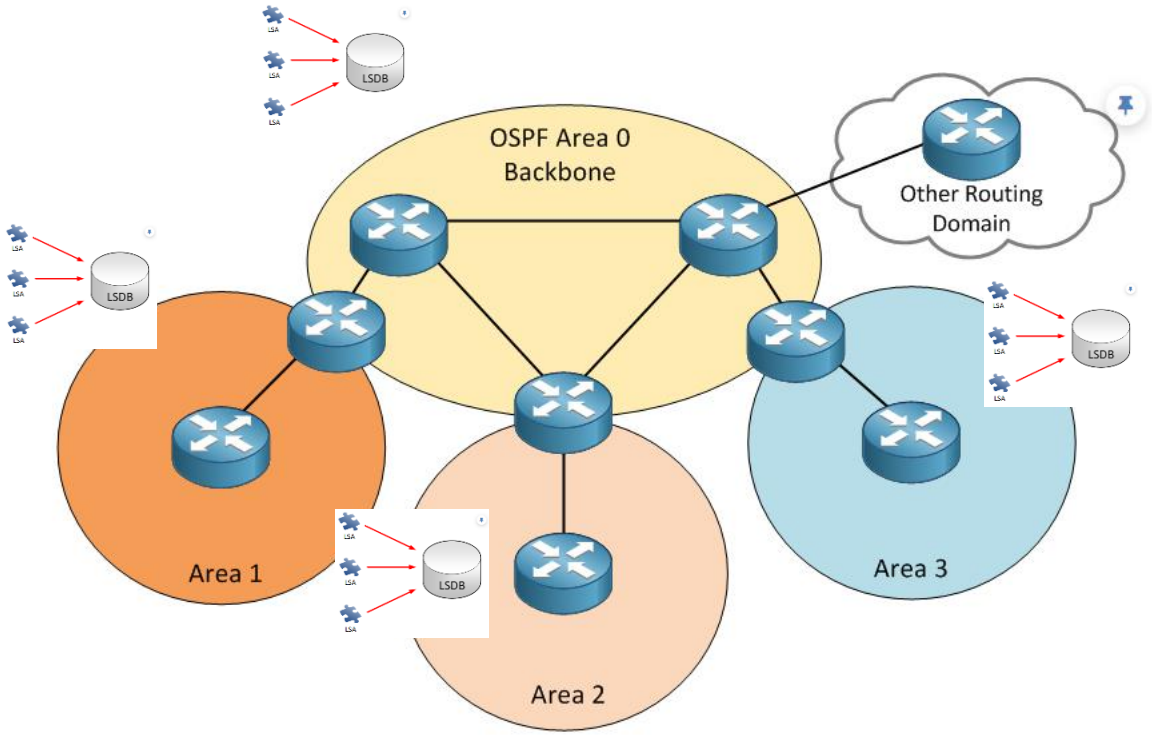
ქსელში ჩართული ყველა *როუტერი* ერთმანეთს უგზავნის **LSA (link state პაკეტებს**

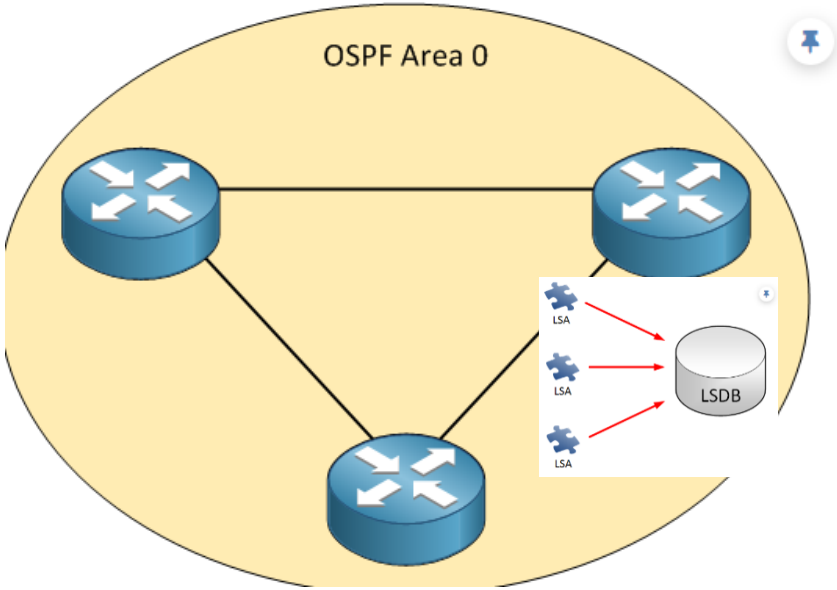
ყველა როუტერი ადგენს **LSDB-ის**

ითვლის ყველა საუკეთესო მარშრუტის სხვადასხვა ქსელამდე.

საუკეთესო გზის ასარჩევად **OSPF** ი იყენებს **SFP** ალგორითმს რომელიც დაფუძნებულია გამტარუნარიანობაზე (bandwidth). ანუ **OSPF** *ისთვის* საუკეთესო გზად ითვლება ის გზა რომელსაც შეუძლია უფრო დიდი ინფორმაციის გატარება.

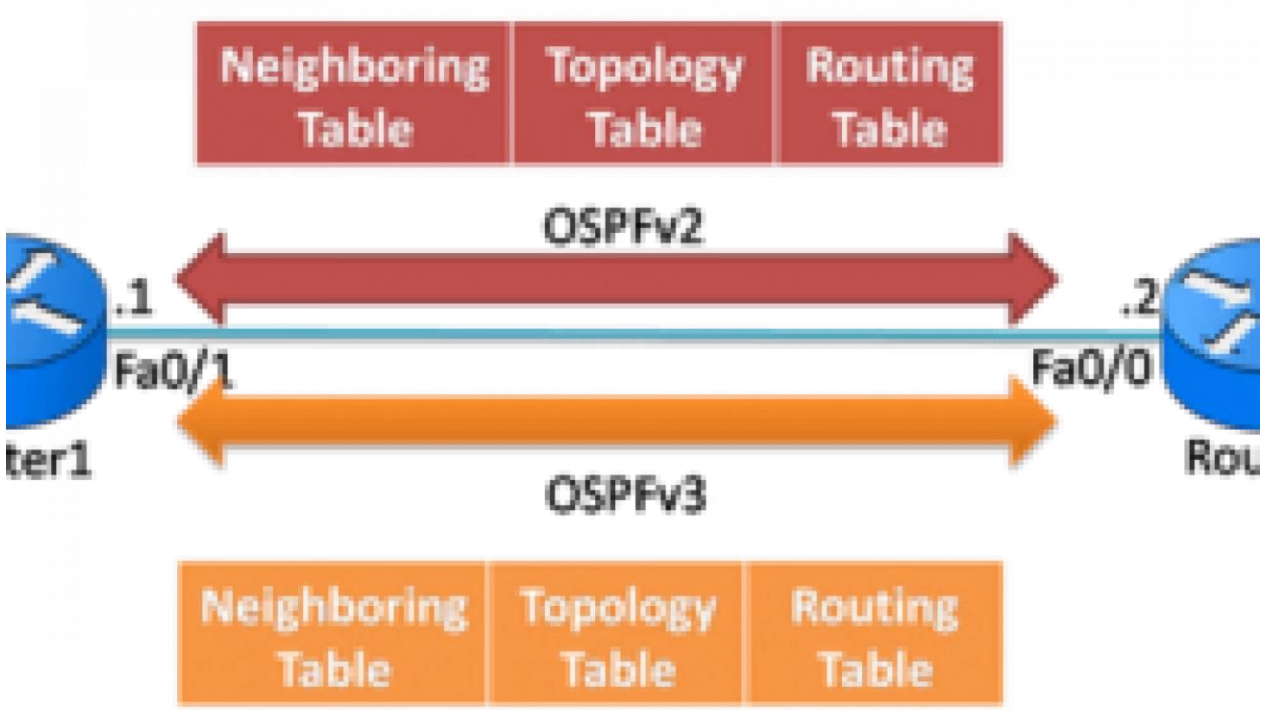
OSPF link-State როუტინგ პროტოკოლი





OSPFv2 vs OSPFv3

OSPFv2 განკუთვნილია IPv4 ქსელებისთვის ხოლო OSPFv3 კი IPv6 ის ქსელებისთვის.
ორივე პროტოკოლს მუშაობს იდენტურად თუმცა მათ შორის არის მცირედი განსხვავება.



განსხვავებას წარმოადგენს შემდეგი:

•**Link-local addresses:** OSPFv3 ის როუტერები ერთმანეთში ინფორმაციის გასაცვლელად იყენებენ link-local IPv6 addresses

•**Links, not networks**: OSPFv3 ის შემთხვევაში გვაქვს ლინკები და არა *networks ები როგორც ეს იყო* OSPFv2ს, შესაბამისად OSPFv3 ის გააქტიურება ხდება ინტერფეისზე..

•**OSPFv3 router ID**: OSPFv3 router ID ის კომფიგურაცია უნდა განხორციელდეს ხელით ის არ აირებს მას ავტომატურად OSPFv2 ის გან განსხვავებით

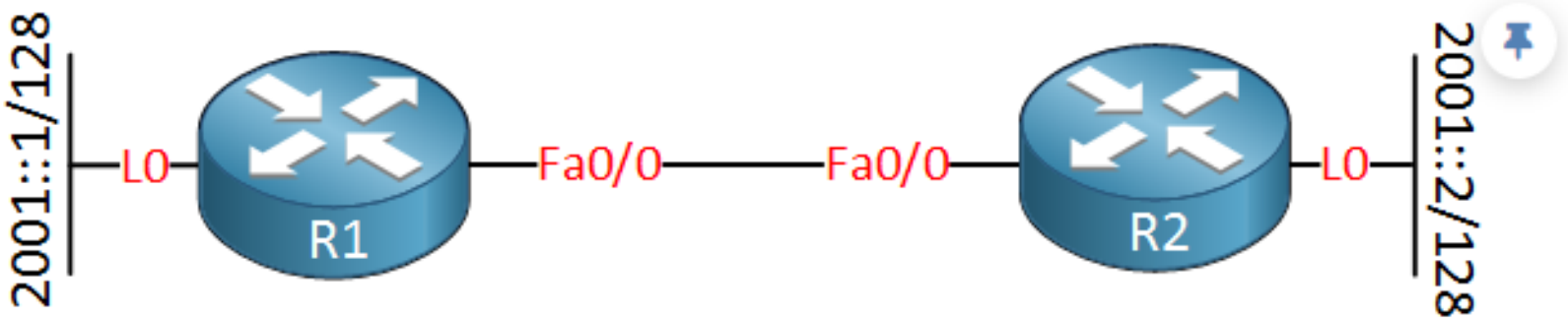
•**Multiple prefixes per interface**: თუ ინტერფეისზე გვაქვს გაწერილი რამდენიმე IPv6 მისამართი OSPFv3 ის გაქტიურებისას დაანონსდება სხვა როუტერებთან ყვალა მათგანი

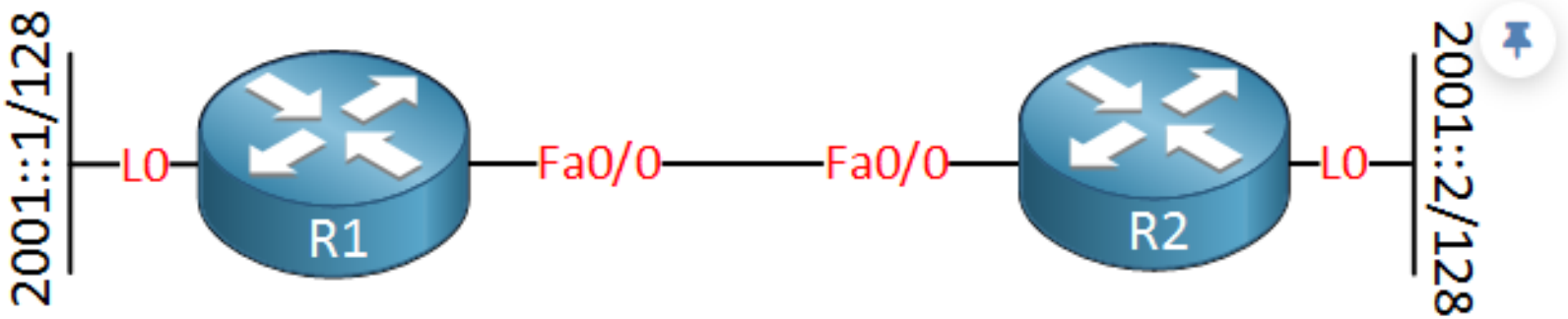
•**Multiple instances per link**: OSPFv3 ში შესაძლებელია რამდეიმე OSPF გაშვება თვითოეულ ლიკზე .

IPv6 OSPFv3

გამვიხილოთ შემდეგი ტოპოლოგია და დავაკომფიგურიროთ OSPFv3.

ნახაზე მოცემულია ორი როუტერი და მათზე არის მიერტებული ორი IPv6 ქსელი.





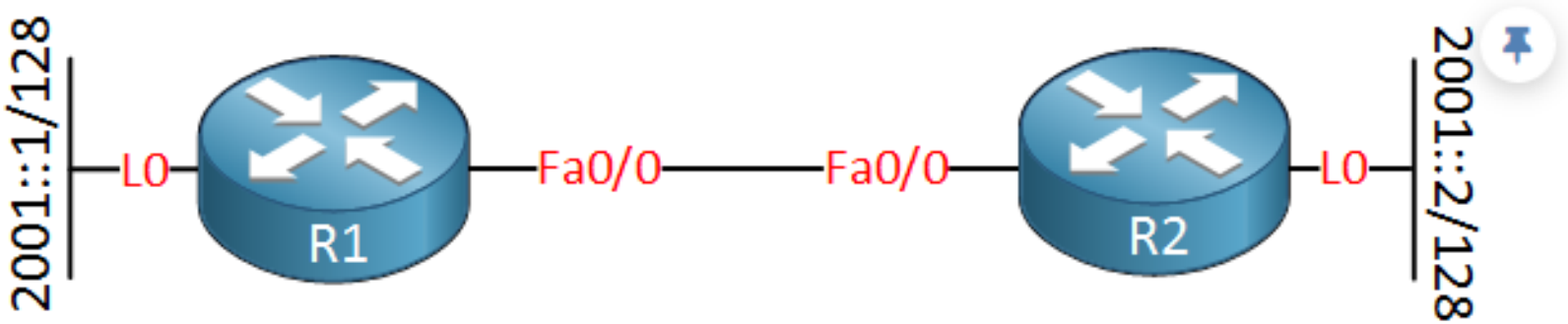

```
R1(config)#ipv6 unicast-routing
R1(config)#interface loopback 0
R1(config-if)#ipv6 address 2001::1/128
```

```
R2(config)#ipv6 unicast-routing
R2(config)#interface loopback 0
R2(config-if)#ipv6 address 2001::2/128
```

პირველ რიგში საჭიროა როუტერებზე გავააქტიუროთ IPv6 დამისამართება და ინტერფეისებს გაუწეროთ ip მისამართები.

IPv6 დამისამართების გააქტიურება ხდება **ipv6 unicast-routing** ბრძანებით.

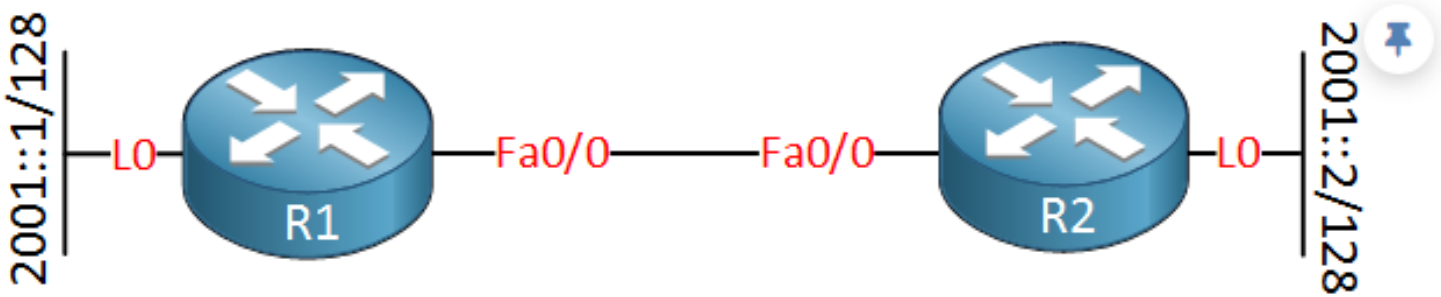
ხოლო ინტერფეისებზე ip მისმართების გაწერა ხდება ბრძანებით **ipv6 address** [**IPv6 მისამართი**]




```
R1#show ipv6 interface brief
FastEthernet0/0            [up/up]
Loopback0                  [up/up]
    FE80::CE09:18FF:FE0E:0
    2001::1
```

```
R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
Loopback0                  [up/up]
    FE80::CE0A:18FF:FE0E:0
    2001::2
```

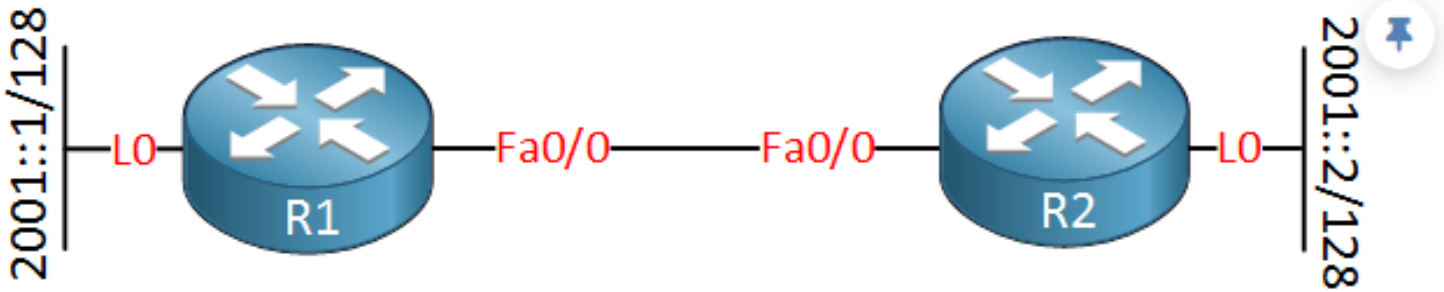
მას შემდეგ რაც გავწერთ ინტერფეისებზე ip მისამართებს შეგვიძლია ისინი შევამოწმოთ შემდეგი ბრძანებით. **show ipv6 interface brief**




```
R1#show ipv6 interface brief
FastEthernet0/0            [up/up]
Loopback0                  [up/up]
    FE80::CE09:18FF:FE0E:0
    2001::1
```

```
R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
Loopback0                  [up/up]
    FE80::CE0A:18FF:FE0E:0
    2001::2
```

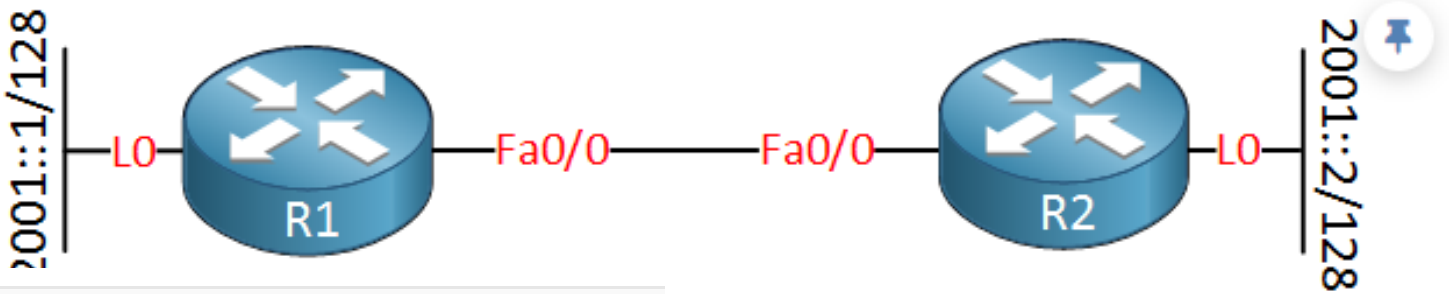
როგორც ვხედავთ fastEthernet 0/0 არ არის link-local IPv6 მისამართი ამიტომ ეს უნდა გამოვასწოროთ და უნდა გავააქტიუროტ ამ ინტერფეისებზე IPv6 დამისამართება.




```
R1(config)#interface fastEthernet 0/0
R1(config-if)#ipv6 enable
```

```
R2(config)#interface fastEthernet 0/0
R2(config-if)#ipv6 enable
```

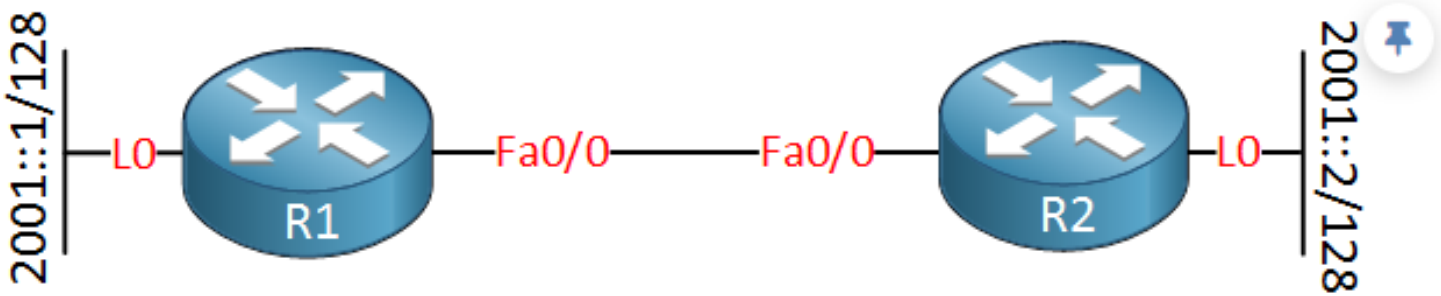
ინტერფეისზე ipv6 ის გააქტიურება ხდება შემდეგი ბრძანებით **ipv6 enable**




```
R1#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::CE09:18FF:FE0E:0
Loopback0                  [up/up]
    FE80::CE09:18FF:FE0E:0
    2001::1
```

```
R2#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::CE0A:18FF:FE0E:0
Loopback0                  [up/up]
    FE80::CE0A:18FF:FE0E:0
    2001::2
```

როგორც კი ინტერფეისზე გავააქტიურებთ ipv6 დამისამართებას ის ავტომატურად დაიგენერირებს link-local IPv6 მისამართს



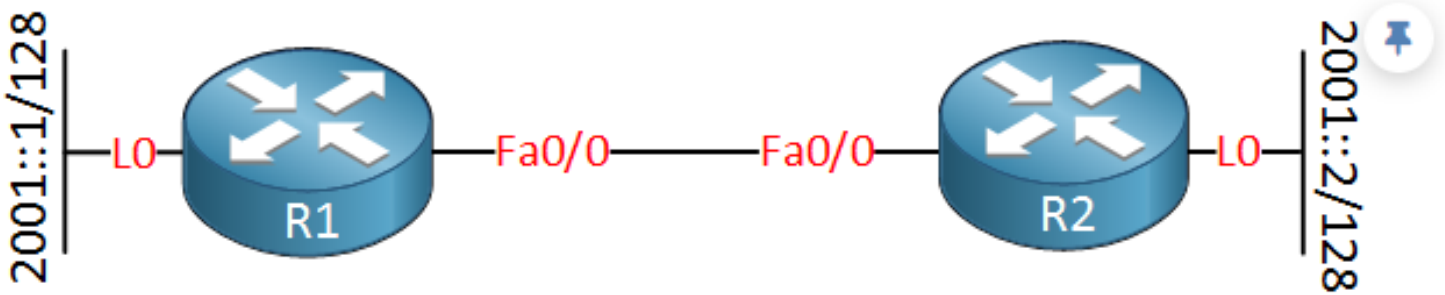
შემდეგ ეტაპს წარმოადგენს ის რომ როუტერებზე გავააქტიუროთ OSPFv3 პროტოკოლი

```
R1(config)#ipv6 router ospf 1
R1(config-rtr)#router-id 1.1.1.1
R1(config-rtr)#exit
```

როუტერზე აქტიურდება ipv6 OSPF პროცესი რომლის ნომერიც არის 1

```
R2(config)#ipv6 router ospf 1
R2(config-rtr)#router-id 2.2.2.2
R2(config-rtr)#exit
```

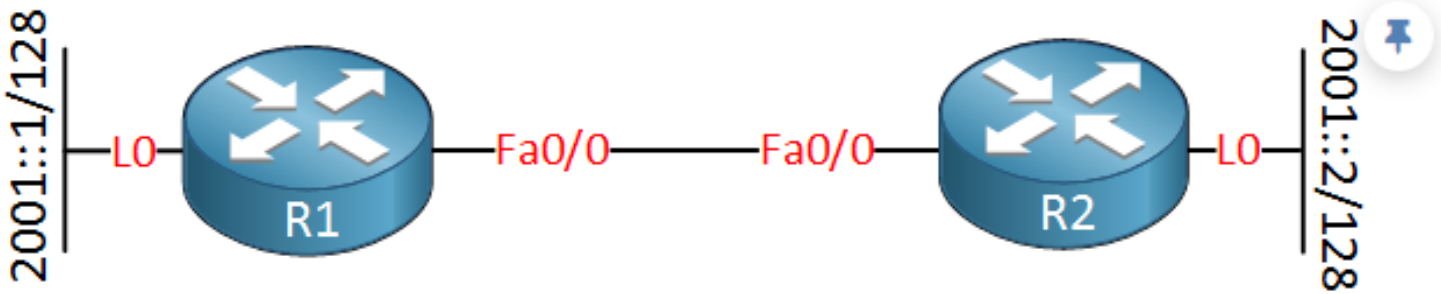
როუტერზე აქტიურდება ipv6 OSPF პროცესი რომლის ნომერიც არის 1




```
R1(config)#interface fastEthernet 0/0
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#exit
R1(config)#interface loopback 0
R1(config-if)#ipv6 ospf 1 area 0
```

```
R2(config)#interface fastEthernet 0/0
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#exit
R2(config)#interface loopback 0
R2(config-if)#ipv6 ospf 1 area 0
```

საბოლოო ეტაპს წარმოადგენს ინტერფეისებზე ipv6 ospf ის პროცესის გააქტიურება **ipv6 ospf 1 area 0**



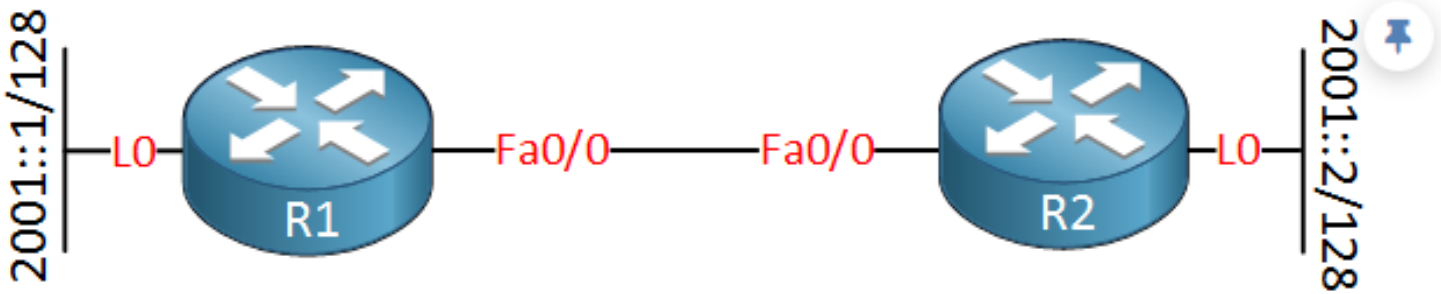
განვიხილოთ შემდეგი ბრძანება

ipv6 ospf 1 area 0

ბრძანების ამ ნაწილით აქტიურდება ინტერფეისზე IPv6 OSPF პროცესი

წარმოადგენს პროცესის ნმერს რომელიც შევქმენით როუტერზე

წარმოაგენს AREA ს ნომერს რომელიშიც უნდა იმყოფებოდეს მოცემული ინტერფეისი



იმისთვის რომ გავარკვიოთ დადგა თუ არა OSPF მეზობლობა როუტერებს შორის შეგვიძლია გამოვიყენოთ შემდეგი ბრძანება **show ipv6 ospf neighbor**
რომელიც გამოიტანს შემდეგ შედეგს :

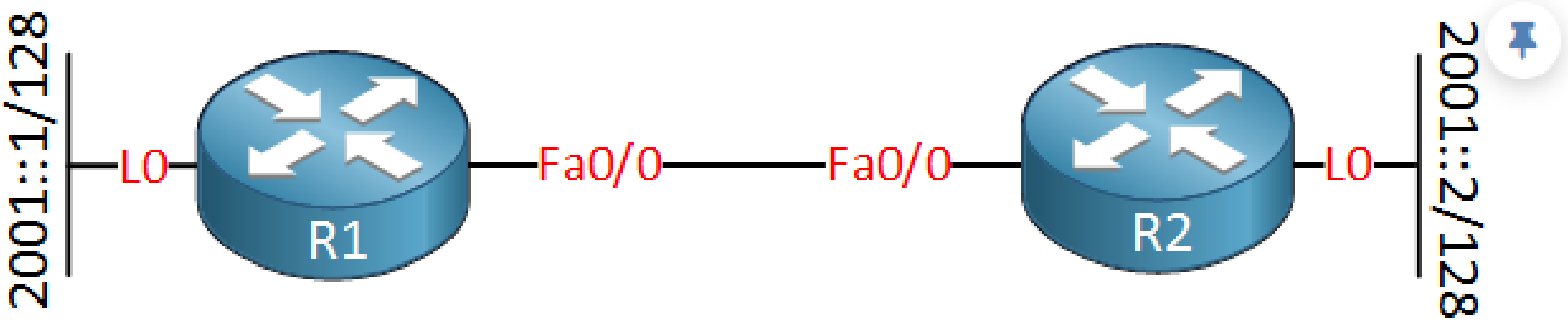
```
R1#show ipv6 ospf neighbor

Neighbor ID  Pri   State           Dead Time   Interface ID    Interface
2.2.2.2        1   FULL/BDR        00:00:30    4               FastEthernet0/0
```

მეზობელი როუტერის ID (იდენთიფიკატორი)

დროის პერიოდი რომლის განმავლობაშიც როუტერებს შორის არის მეზობლობა

ინტერფეისი საიდანაც არის მეზობლობა ამ როუტერტან

FULL ნიშნავს იმას რომ ამ მეზობელტან არის სრულად დამყარებული მეზობლური კავშირი და მათ შეუძლიათ ინფორმაციის გაცვლია



შევამოწმოტ როუტერზე routing table
რისი შემოწმებაც ხდება შემდეგი ბრძანებით **show ipv6 route ospf**

```
R2#show ipv6 route ospf
O   2001::1/128 [110/10]
     via FE80::C00E:1AFF:FEA7:0, FastEthernet0/0
```

განვიზილოთ შემდეგი როუტინგ ინფორმაცია

```
R2#show ipv6 route ospf
O   2001::1/128 [110/10]
     via FE80::C00E:1AFF:FEA7:0, FastEthernet0/0
```

- **O** - ნიშნავს იმას რომ ქსელი არის ნასწავლი OSPF პროტოკოლის მიერ
- **2001::1/128** - არის მეზობელი როუტერის ქსელი
- **FE80::C00F:1AFF:FEA7:0** - ipv6 მისამართი რომელის გავლითაც იცის როუტერმა ეს მისამართი
- **FastEthernet0/0** - ინტერფეისი საიდანაც მიიროო ინფორმაცია როუტერმა ამ ქსელის შესახებ

ლექცია 11

ქსელური მისამართების ტრანსლაცია
network address translation (NAT)
Port Address Translation (PAT)

Network address translation (NAT) წარმოადგენს ტექნოლოგიას, რომელიც შიდა (კერძო) IP მისამართებს გარდაქმნის გარე IP მისამართებად (ინტერნეტი)

გარე IP მისამართები:

- გამოიყენება ინტერნეტში;
- უნდა იყოს უნიკალური;
- მათი გავრცელება ხდება ICANN (სახელების და ნომრების ინტერნეტ კორპორაცია) მიერ;
- IPv4 მისამართი არ არის საკმარისი ყველა ინტერნეტში მოწყობილობისათვის (IPv4 მისამართი დაახლოებით 4 მილიარდი)

შიდა IP მისამართები:

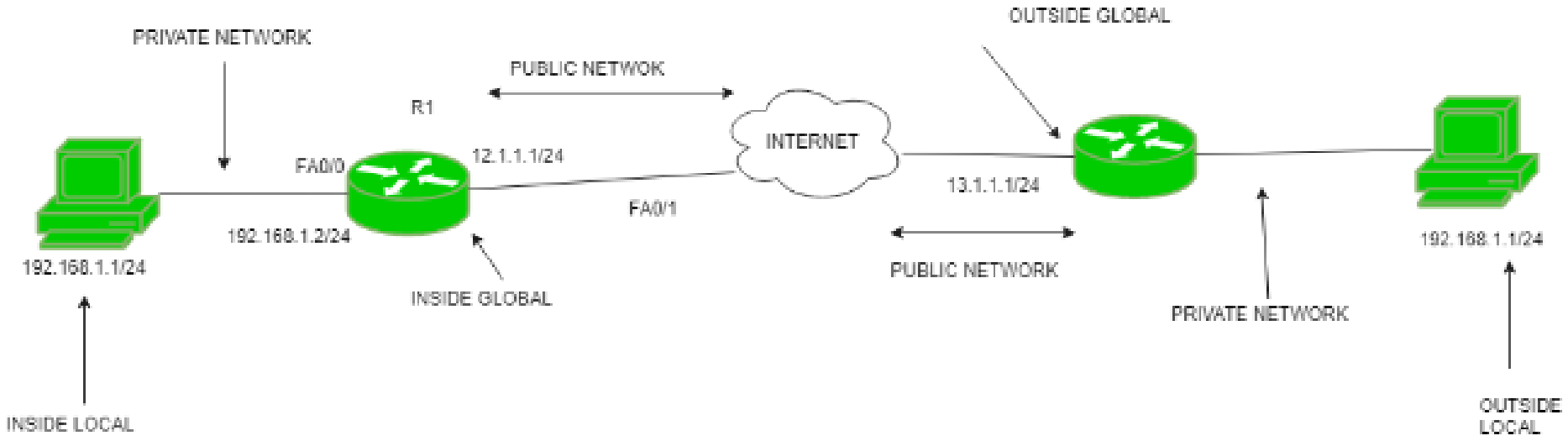
- არ ხდება მათი მარშრუტიზირება ინტერნეტში;
- შესაძლებელია გამოყენებული იქნას ICANN მიმართვის გარეშე;
- დასაშვებია გამოყენებული იქნას ერთიდაიგივე მისამართი სხვადასხვა ქსელში (რადგან არ ჩანან ინტერნეტში)

NAT-ის იდეა არის ის, რომ მრავალ მოწყობილობას დაუშვას ინტერნეტში წვდომა ერთი საჯარო მისამართის მეშვეობით. ამის მისაღწევად საჭიროა ლოკალური (პირადი) IP მისამართის საჯარო IP მისამართად თარგმნა.

ქსელის მისამართის თარგმნა (NAT) არის პროცესი, რომლის დროსაც ერთი ან მეტი ადგილობრივი IP მისამართი ითარგმნება ერთ ან მეტ გლობალურ IP მისამართად და პირიქით, ლოკალური ჰოსტებისთვის ინტერნეტით წვდომის უზრუნველსაყოფად.

Pat აკეთებს პორტის ნომრების თარგმნას, ანუ ნიღბავს ჰოსტის პორტის ნომერს სხვა პორტის ნომრით, იმ პაკეტში, რომელიც გადაიგზავნება დანიშნულების ადგილზე. შემდეგ ის აკეთებს IP მისამართისა და პოrტის ნომრის შესაბამის ჩანაწერებს NAT ცხრილში. NAT ჩვეულებრივ მუშაობს როუტერზე ან firewall-ზე.

ზოგადად, სასაზღვრო როუტერი კონფიგურირებულია NAT-ისთვის, ანუ როუტერისთვის, რომელსაც აქვს **ერთი ინტერფეისი ლოკალურ (შიდა) ქსელში და ერთი ინტერფეისი გლობალურ (გარე) ქსელში**. როდესაც პაკეტი გადის ლოკალური (შიდა) ქსელის გარეთ, NAT გარდაქმნის ამ ლოკალურ (პირად) IP მისამართს გლობალურ (საჯარო) IP მისამართად. როდესაც პაკეტი შედის ლოკალურ ქსელში, გლობალური (საჯარო) IP მისამართი გარდაიქმნება ლოკალურ (პირად) IP მისამართად.



პორტის მისამართის თარგმანი (PAT):

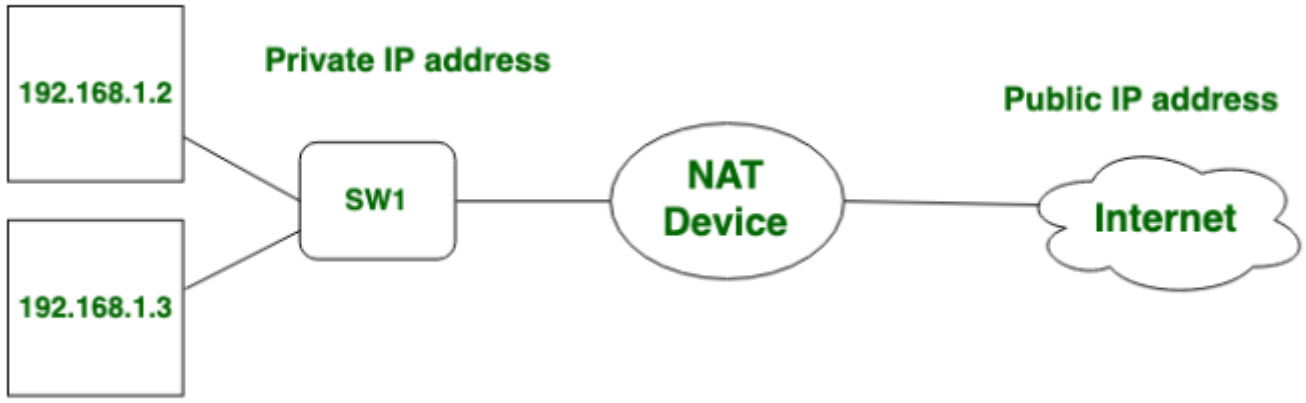
ეს ასევე ცნობილია როგორც NAT overload (გადატვირთვა). ამ შემთხვევაში, ბევრი ადგილობრივი (პირადი) IP მისამართი ითარგმნება ერთ საჯარო IP მისამართზე. ზოგჯერ პირადი მისამართები ითარგმნება ინტერფეისის მისამართში (ერთად). ამ შემთხვევაში, **პორტის ნომრები გამოიყენება ტრაფიკის გასარჩევად, ანუ რომელი ტრაფიკი ეკუთვნის რომელ IP მისამართს**. პროცედურა: პროცედურა თითქმის იგივეა, რაც კეთდება Dynamic NAT-ში, მაგრამ გახსოვდეთ PAT-ში, ერთზე მეტი პირადი IP მისამართი ითარგმნება ერთ საჯარო IP მისამართად.

დავუშვათ, ქსელში ორი ჰოსტი A და B დაკავშირებულია. ახლა ორივე ითხოვს იმავე დანიშნულების ადგილს, იმავე პორტის ნომერზე, ვთქვათ 1000, შიდას მხარეს, ერთსა და იმავე დროს. თუ NAT ახორციელებს მხოლოდ IP მისამართების თარგმნას, მაშინ როდესაც მათი პაკეტები NAT-ში მივა, მათი ორივე IP მისამართი დაიფარება ქსელის გარე IP მისამართით და გაიგზავნება დანიშნულების ადგილზე. დანიშნულების ადგილიდან გამოიგზავნება პასუხები როუტერის საჯარო IP მისამართზე. ამრიგად, პასუხის მიღებისას NAT-ისთვის გაუგებარი იქნება, თუ რომელი პასუხი რომელ ჰოსტს ეკუთვნის (რადგან წყაროს პორტის ნომრები ორივე A და B-სთვის არის იგივე). ამიტომ, ასეთი პრობლემის თავიდან ასაცილებლად, NAT ნიღბავს წყაროს პორტის ნომერსაც და აკეთებს ჩანაწერს NAT ცხრილში.

Network Address Translation (NAT) and Port Address Translation (PAT)

NAT, პირადი IP მისამართs ან ლოკალურ IP მისამართს თარგმნის საჯარო IP მისამართად.

მაგალითი:განვიხილოთ სახლის ქსელი სამი მოწყობილობით: კომპიუტერი, სმარტფონი და ჭკვიანი ტელევიზორი. NAT-ის გარეშე, თითოეულ ამ მოწყობილობას უნდა ჰქონდეს უნიკალური საჯარო IP მისამართი ინტერნეტთან დასაკავშირებლად. თუმცა, **NAT-ით, ყველა ამ მოწყობილობას შეუძლია ერთი საჯარო IP მისამართის გაზიარება და ინტერნეტთან კომუნიკაცია მათი პირადი IP მისამართების გამოყენებით.** როდესაც ერთ-ერთი მოწყობილობა აგზავნის მოთხოვნას ინტერნეტში, NAT თარგმნის მოწყობილობის პირად IP მისამართს ქსელის საჯარო IP მისამართად და აგზავნის მოთხოვნას ინტ



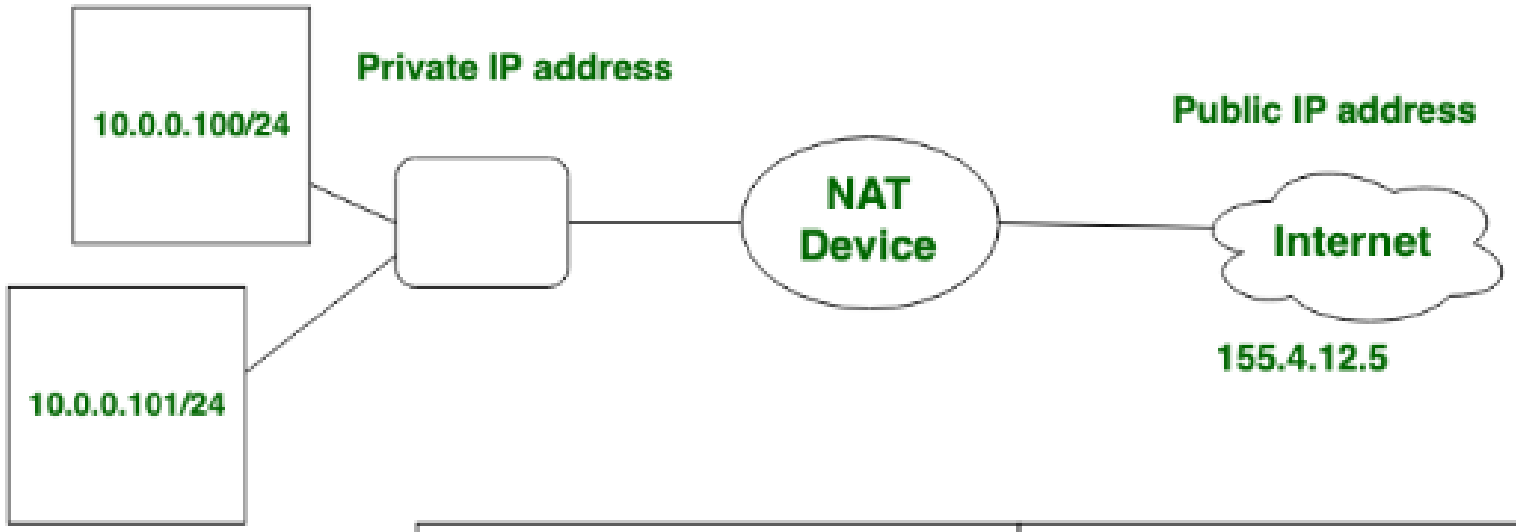
NAT

პორტის მისამართის თარგმანი (PAT):

PAT-ში, შიდა IP მისამართები ითარგმნება გარე IP მისამართში **პორტის ნომრების საშუალებით**. PAT ასევე იყენებს IPv4 მისამართს, მაგრამ პორტის ნომრით. მას აქვს ორი ტიპი:

1. Static
2. verloaded PAT

მაგალითი:განვიხილოთ სახლის ქსელი სამი მოწყობილობით: კომპიუტერი, სმარტფონი და ჭკვიანი ტელევიზორი. **PAT-ის გარეშე, თითოეულ ამ მოწყობილობას უნდა ჰქონდეს უნიკალური საჯარო IP მისამართი ინტერნეტთან დასაკავშირებლად. თუმცა, PAT-ით, ყველა ამ მოწყობილობას შეუძლია ერთი საჯარო IP მისამართის გაზიარება და ინტერნეტთან კომუნიკაცია უნიკალური პორტის ნომრების გამოყენებით**. როდესაც კომპიუტერი აგზავნის მოთხოვნას ინტერნეტში, PAT ანიჭებს მას უნიკალურ პორტის ნომერს და თარგმნის კომპიუტერის პირად IP მისამართს ქსელის საჯარო IP მისამართად. დანიშნულების სერვერი ინტერნეტში იღებს მოთხოვნას და პასუხობს უნიკალურ პორტის ნომერს, რაც საშუალებას აძლევს კომპიუტერს მიიღოს პასუხი.

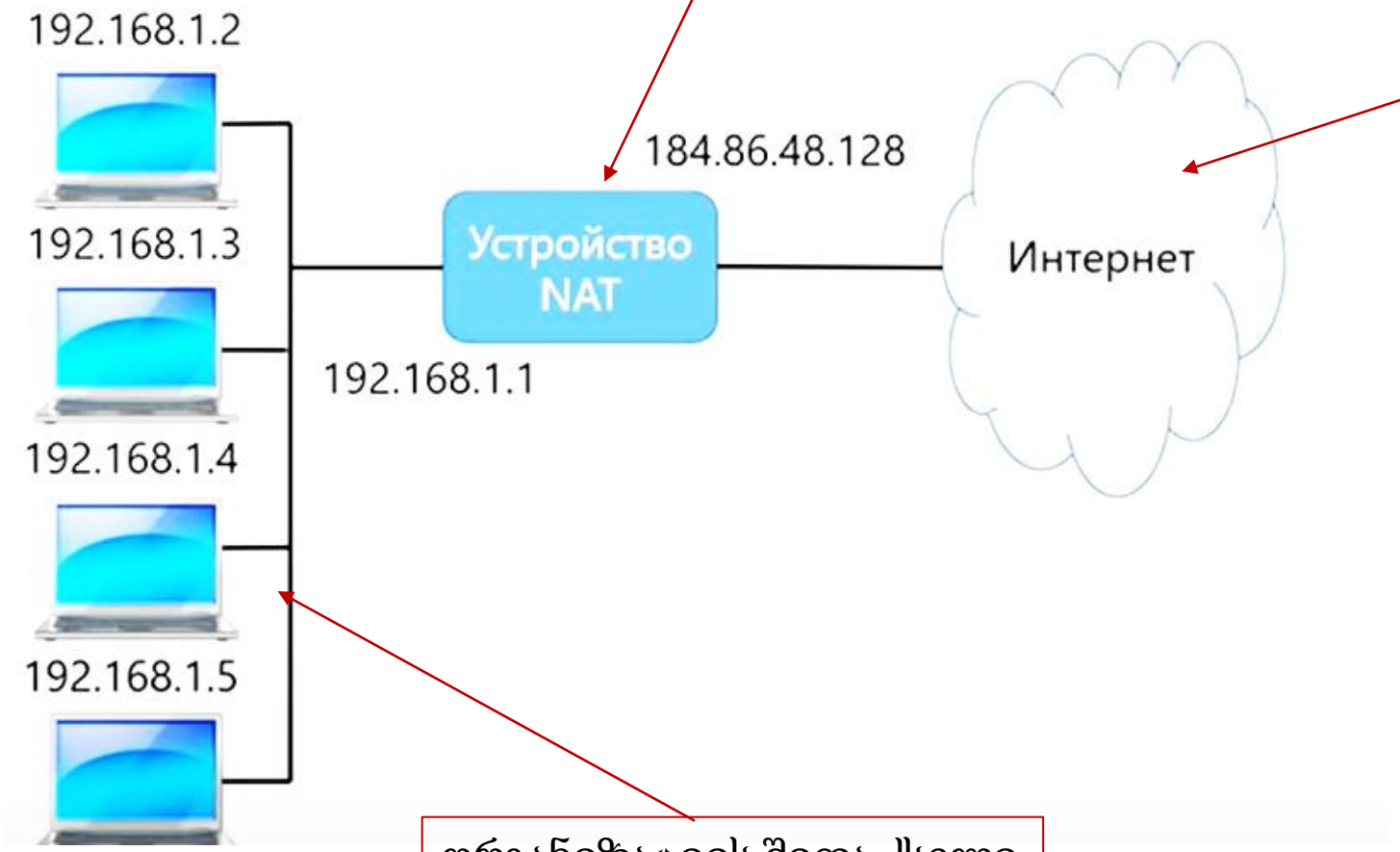


private ip address : Port	public ip address : Port
10.0.0.100:1055	155.4.12.1:1055
10.0.0.101:1056	155.4.12.1:1056

PAT

NAT ტექნოლოგიის მუშაობის პრინციპი

NAT მოწყობილობა, რომლის საშუალებითაც ორგანიზაციის შიდა ქსელის დაკავშირება ხდება ინტერნეტთან. მისი მისამართია 184.86.48.128



ინტერნეტი

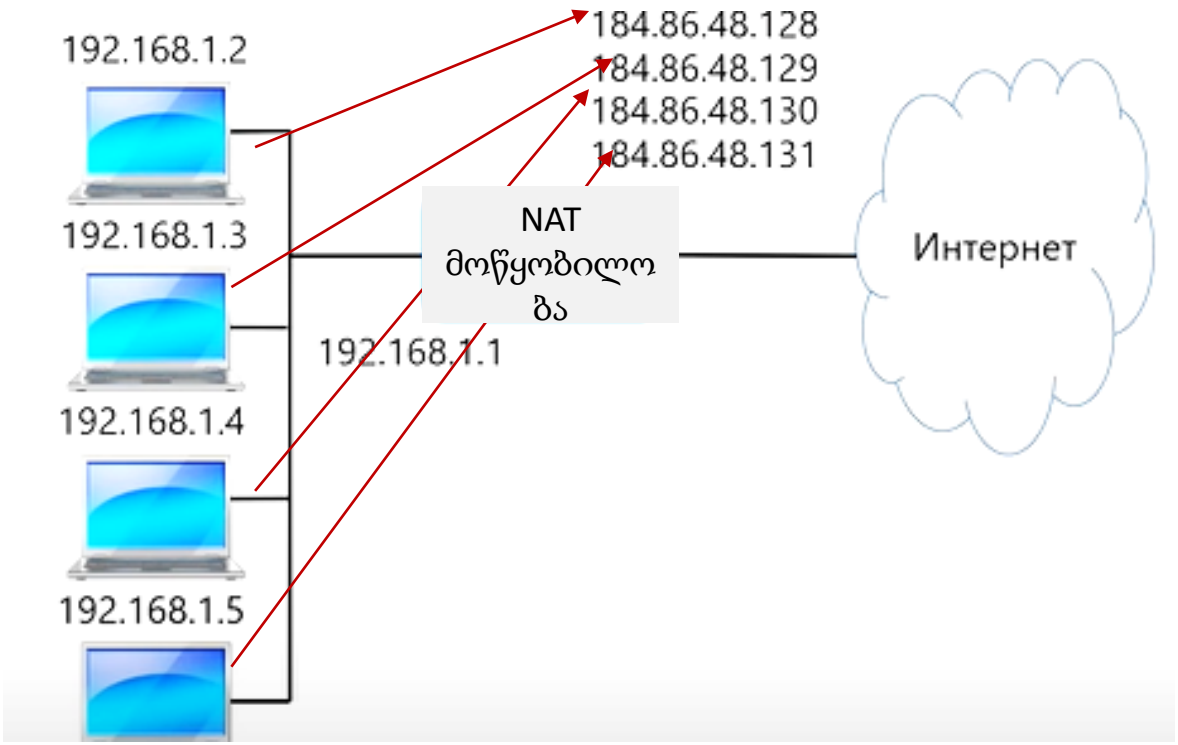
NAT მოწყობილობა შიდა ქსელის მისამართებს გარდაქმნის გარე მისამართად

ორგანიზაციის შიდა ქსელი 192.168.1.0

NAT -ის რამდენიმე ვარიანტი არსებობს:

- სტატიკური

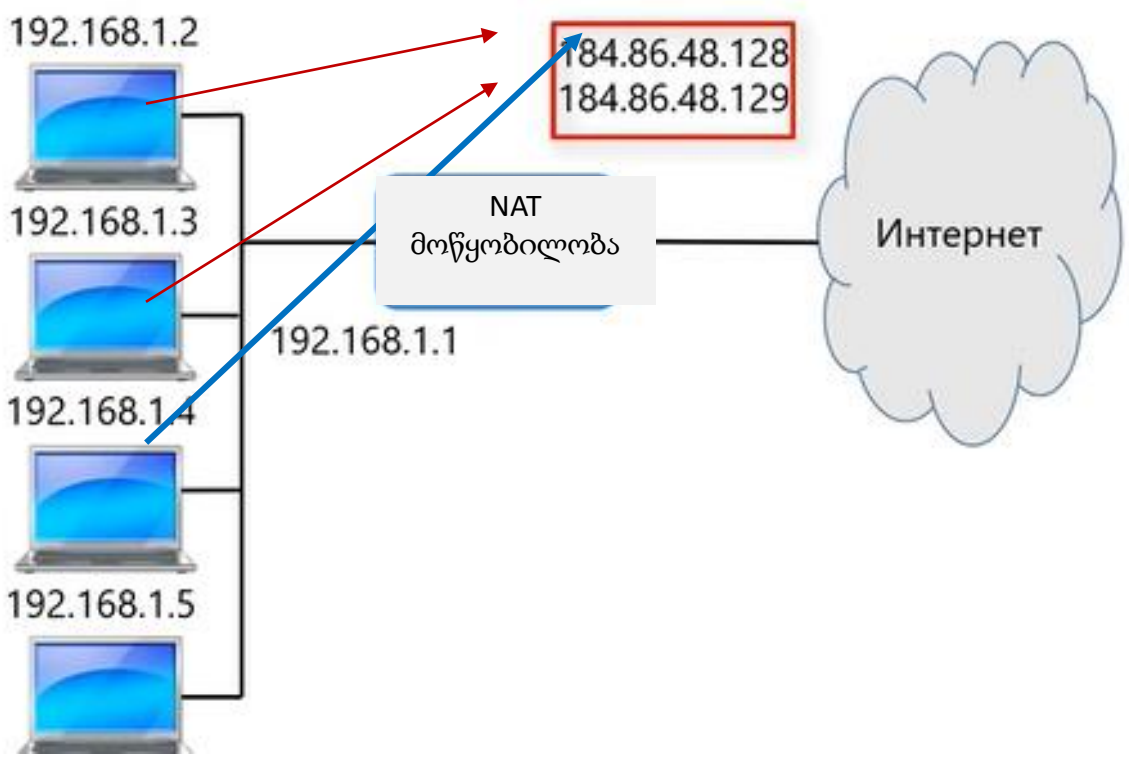
IP მისამართების სტატიკური ასახვა ხდება



ეს ნიშნავს, რომ უნდა არსებობდეს იმდენი IP მისამართი რამდენი კომპიუტერიცაა ქსელში. ეს ვარიანტი გამოიყენება ძალიან იშვიათად

- დინამიური NAT

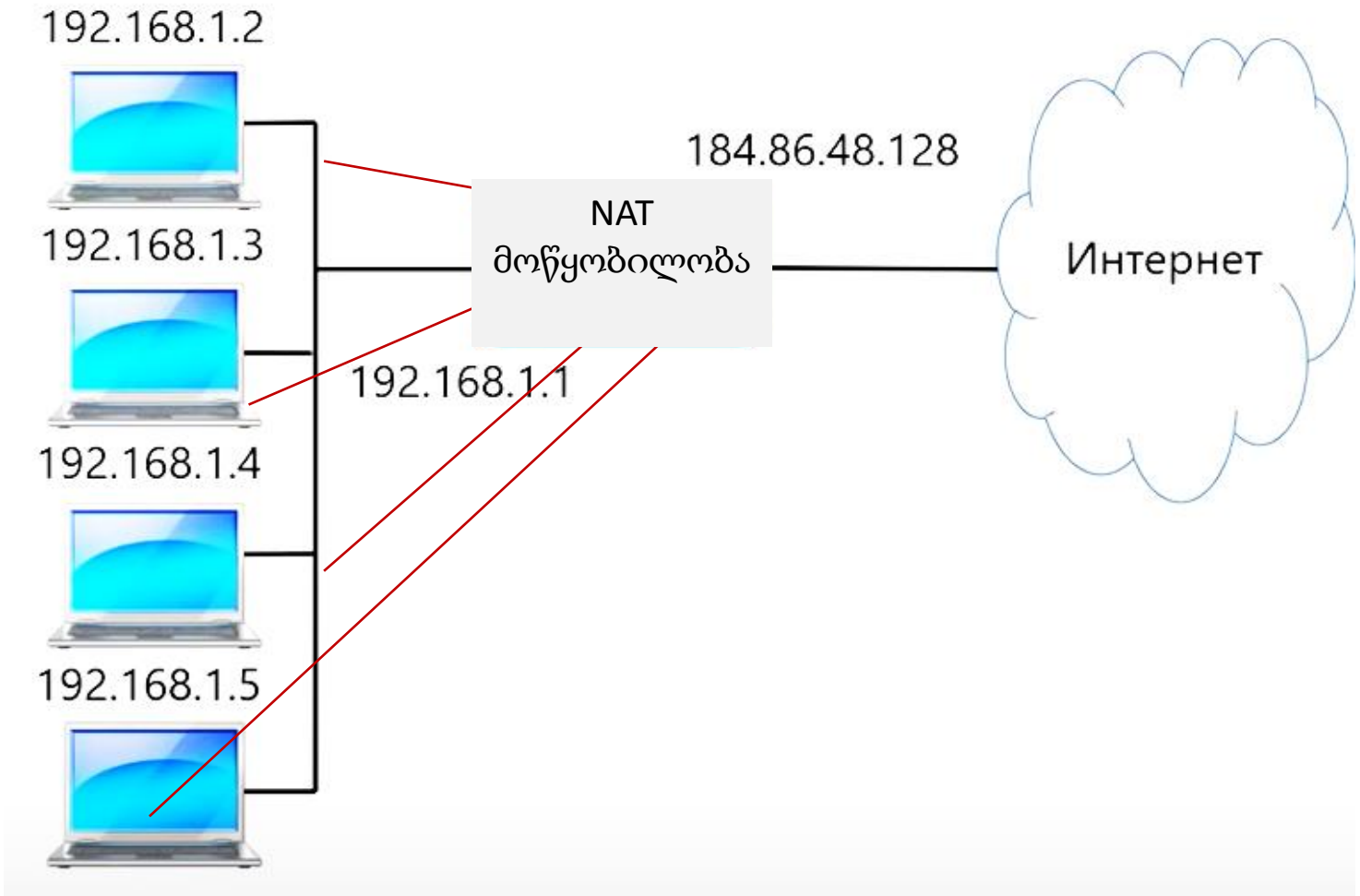
შიდა მისამართების გარდაქმნა ხდება გარე მისამართების ჯგუფებად



მაგალითად პირველი კომპიუტერი იყენებს პირველ მისამართს, მეორე - მეორეს. გარკვეული დროის შემდეგ მესამე კომპიუტერმა შეიძლება გამოიყენოს პირველი მისამართი და ა.შ

- **ერთი მრავალთან**

შიდა მისამართების ასახვა, ხდება ერთ გარე IP მისამართში



განვიხილოთ ერთი მრავალთან, რადგან ის ყველაზე ხშირად გამოიყენება ინტერნეტში ჩასართავად

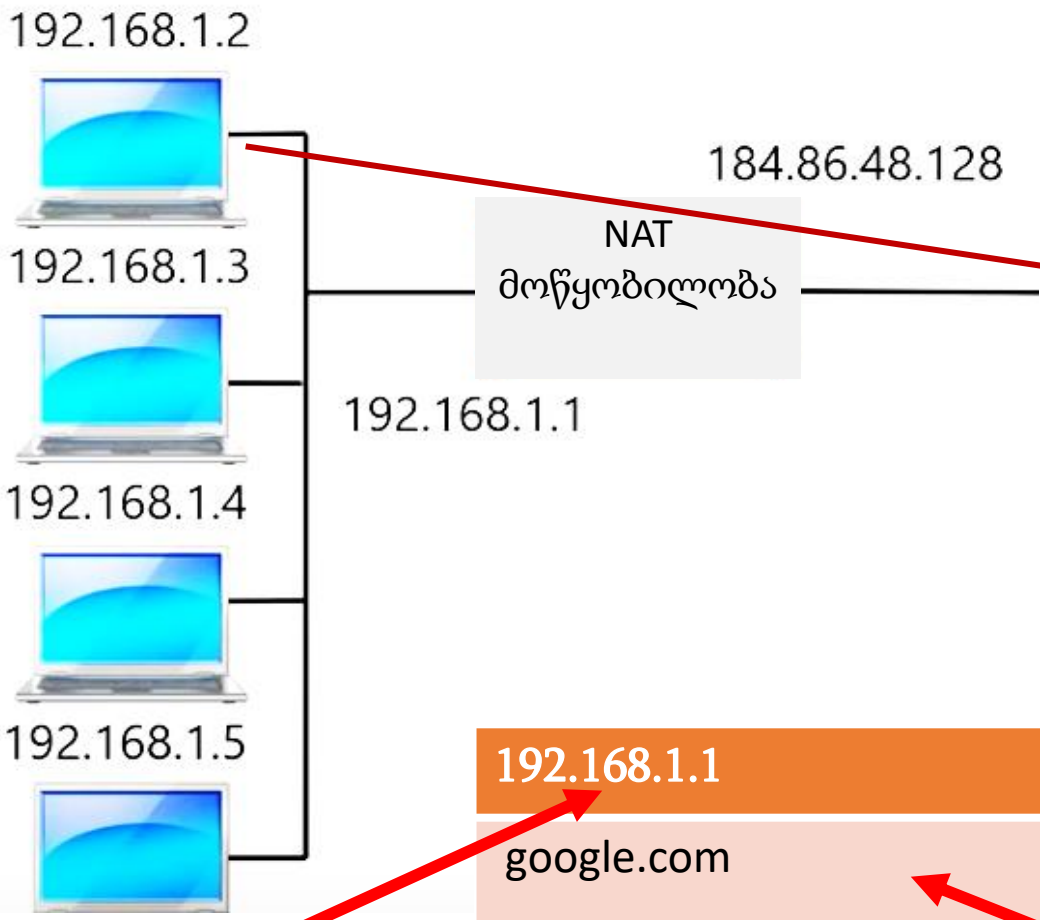
გარდაქმნა ხდება NAT ცხრილის საშუალებით, გამოიყენება კომბინცია IP მისამართი + პორტი.

NAT ცხრილს აქვს სახე

NAT ცხრილი შედგება ოთხი სვეტისგან:

გამგზავნის IP მისამართი	გამგზავნის პორტი	გარე IP მისამართი	დანიშნულების პორტი
192.168.1.2	50300	184.86.48.128	49127
192.168.1.3	52001	184.86.48.128	49128
192.168.1.2	49238	184.86.48.128	49129

დავუშვათ, კომპიუტერი სურს შევიდეს რომელიმე კონრეტულ საიტზე



google.com

ის აგზავნის პაკეტს

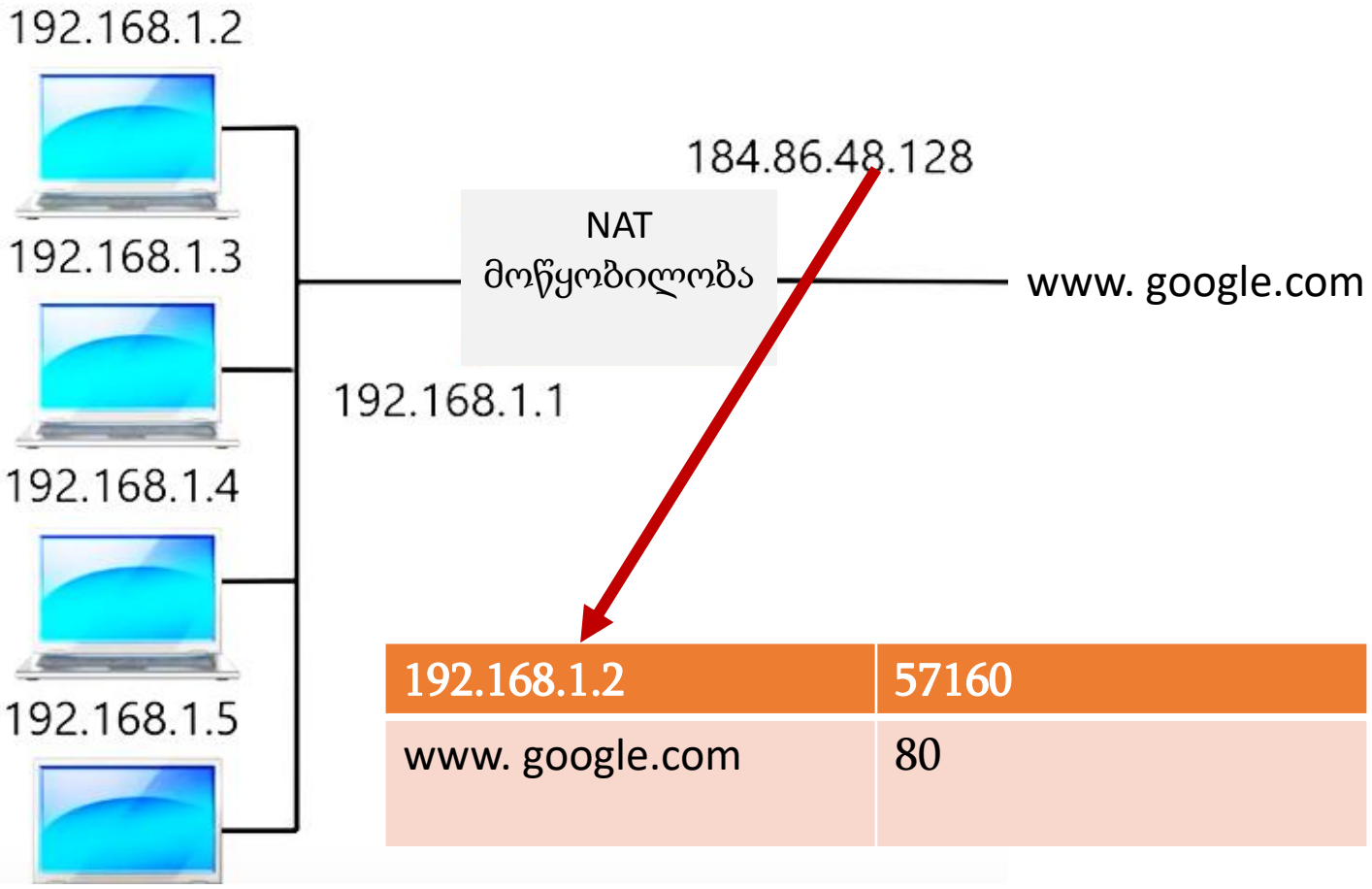
პორტის ველში გენერირდება პორტი,
რომელიც მინიჭერულია ქვს
ოპერაციული სისტემის მიერ

192.168.1.1	57160
google.com	80

გამგზავნი კომპიუტერის შიდა IP მისამართი

პაკეტი უნდა მივიდეს საიტზე და პორტზე

რადგან შიდა IP მისამართი არ შეიძლება გამოყენებული იყოს გარე ქსელში, NAT მოწყობილობამ უნდა შეცვალოს შიდა IP მისამართი გარე IP მისამართით.



192.168.1.2	57160
www. google.com	80

მას შემდეგ რაც NAT მოწყობილობა იღებს პაკეტს, ის წერს შიდა Ip მისამართს და პორტს NAT ცხრილში.

გარე IP მისამართის პორტში წერს 184.86.48.128, გარე პორტს აგენერირებს შემთხვევითათ, მაგალითად 48202

შიდა IP მისამართი	პორტი	გარე IP მისამართი	გარე პორტი
192.168.1.2	57160	184.86.48.128	48202

შემდეგ ეტაპზე ხდება IP ტრანსლაცია, ე.ი ხდება IP მისამართის და პორტის ცვლილება, გამგზავნის Ip მისამართი იქნება 184.86.48.12 პორტის მისამართი იქნება 48202

184.86.48.128	48202
google.com	80

ასეთი სახით ხდება პაკეტის გადაცემა

პასუხის დაბრუნდება შემდეგი სქემის მიხედვით

გამგზავნის IP

გამგზავნის პორტი

google.com	80
184.86.48.128	48202

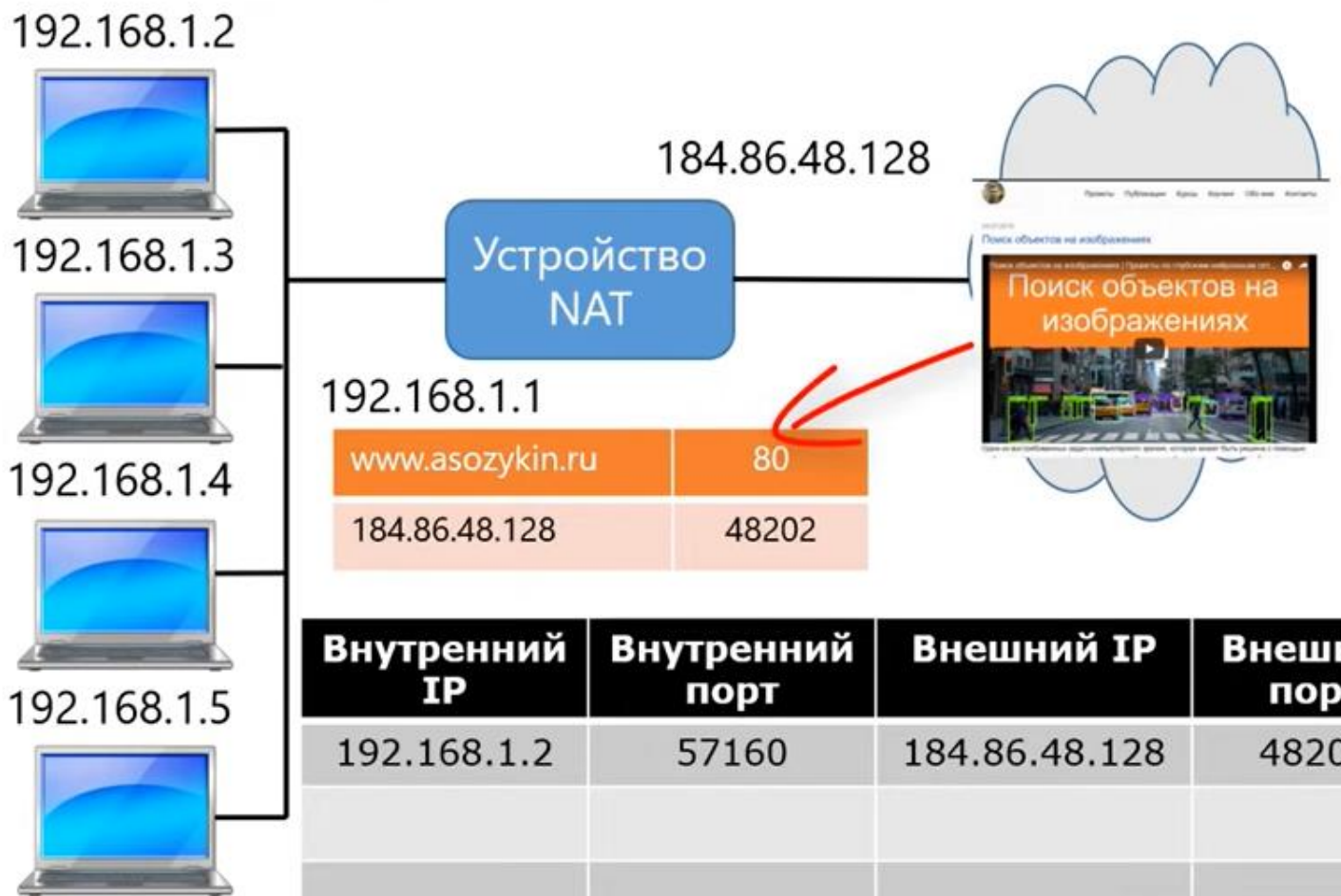
დანიშნუების IP

დანიშნუების პორტი

როუტერს როდესაც შემოუვა გარე პაკეტი რომლის ჩანაწერი გამოიყურება შემდგენაირად
დანიშნულების პორტი 48202 და დანიშნულების ip 184.86.48.128
ის ჩახედავს თავის ნატირების ცხრილს სადაც ნახავს შემდეგ ინფორმაციას ;
თუ პაკეტის დანიშნულების პორტსი არის ჩანაწერი 48202 ის უნდა თარგმნოს შიდა ip მისამართად 192.168.1.2.

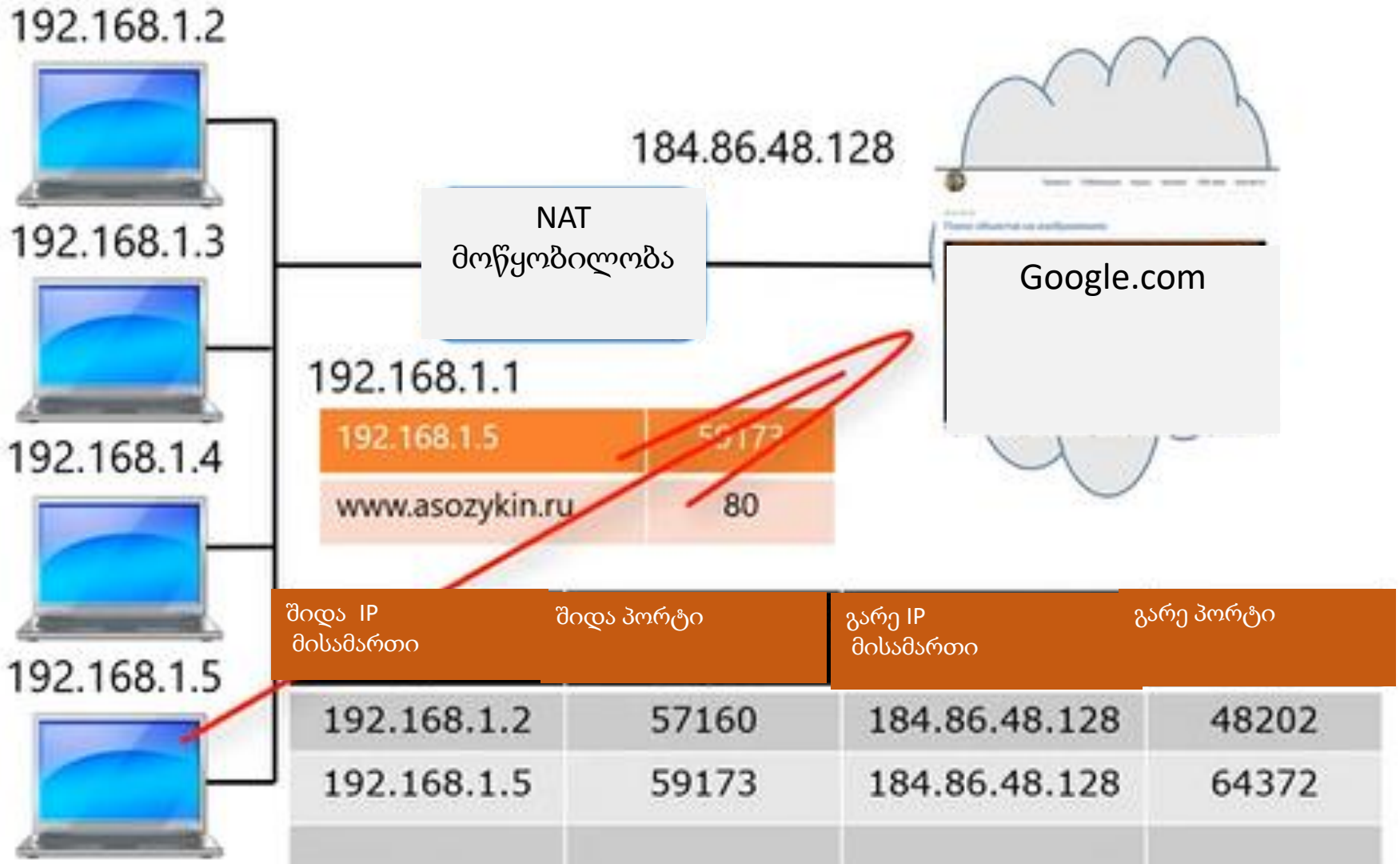
შიდა IP მისამართი	პორტი	გარე IP მისამართი	გარე პორტი
192.168.1.2	57160	184.86.48.128	48202

როდესაც ბრუნდება პასუხი ვებ სერვერიდან, მაშინ

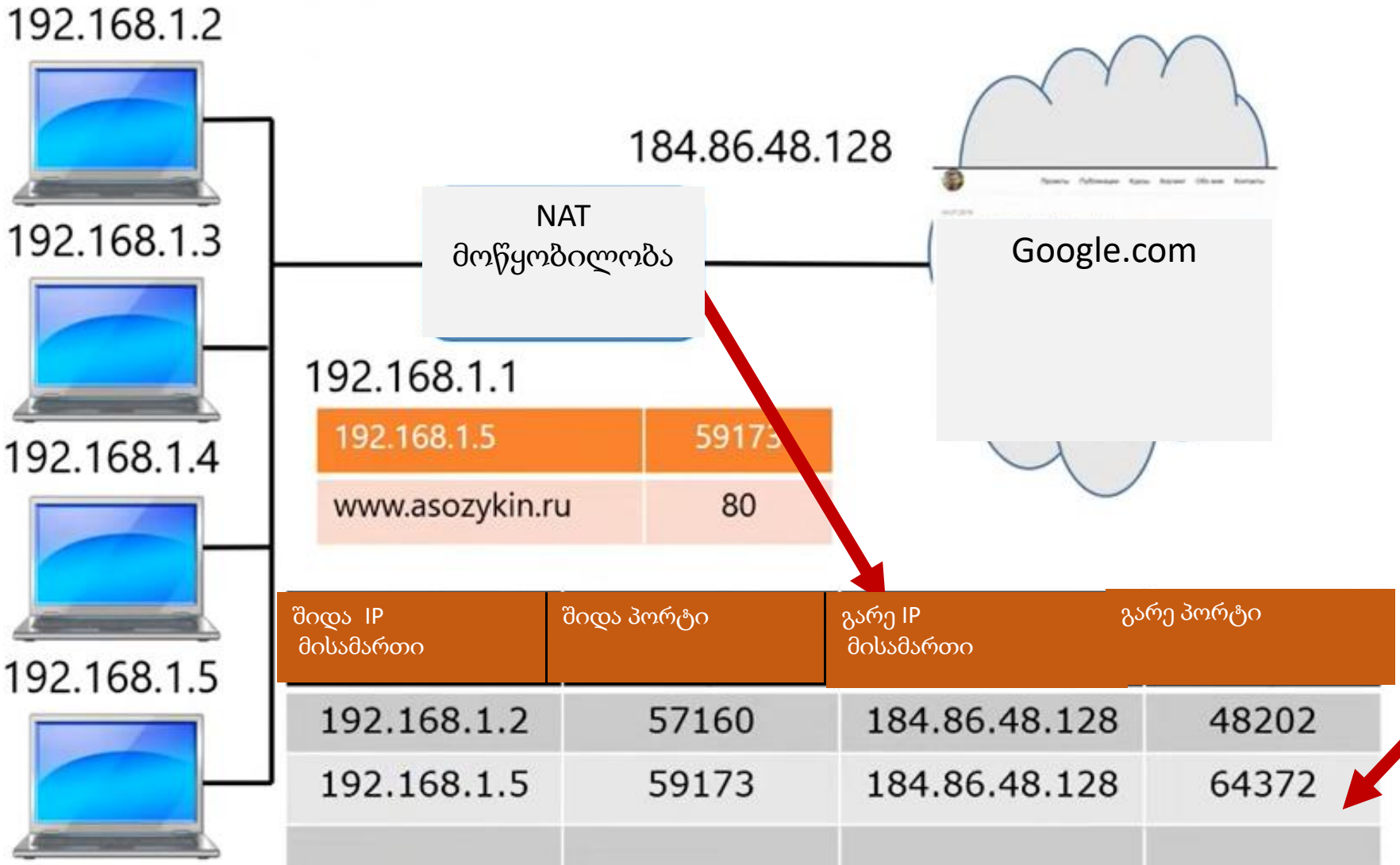


Внутренний IP	Внутренний порт	Внешний IP	Внешний порт
192.168.1.2	57160	184.86.48.128	48202

თუ იგივე საიტზე უნდა შესვლა სხვა კომპიუტერსაც ხდება NaT ცხრილის საშუალებით



შიდა IP მისამართი	შიდა პორტი	გარე IP მისამართი	გარე პორტი
192.168.1.2	57160	184.86.48.128	48202
192.168.1.5	59173	184.86.48.128	64372



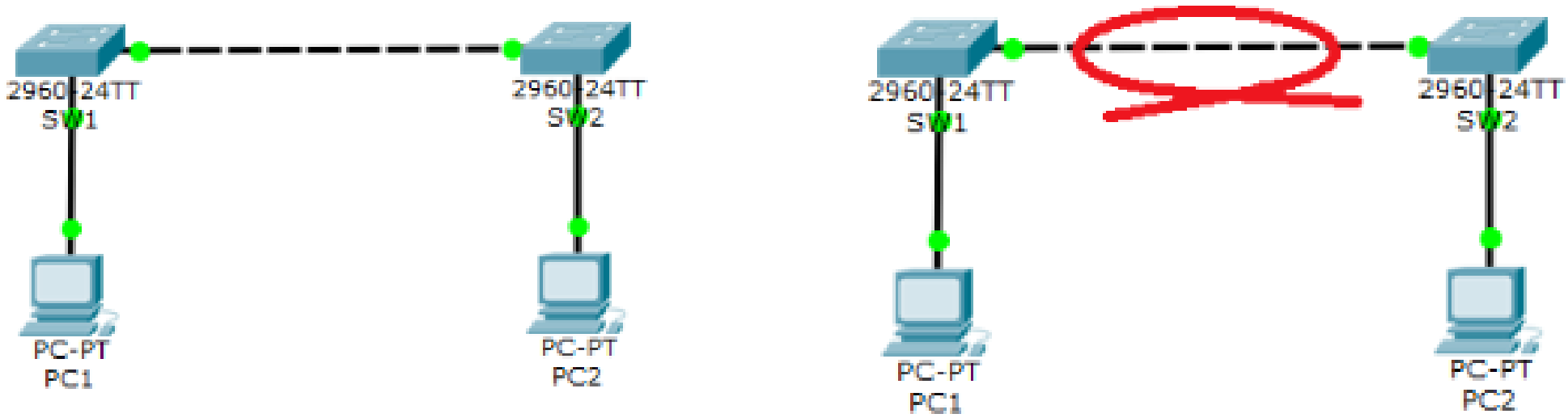
შიდა IP მისამართი	შიდა პორტი	გარე IP მისამართი	გარე პორტი
192.168.1.2	57160	184.86.48.128	48202
192.168.1.5	59173	184.86.48.128	64372

პორტის ნომერი ასევე
გენერირდება შემთხვევითად

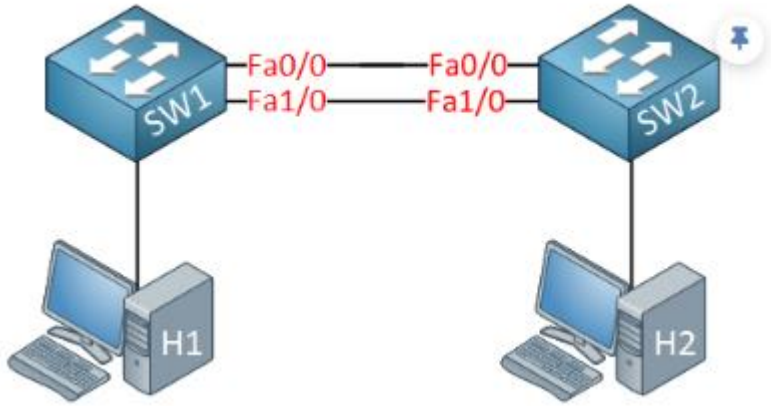
ლექცია 12

Spanning Tree protokol (stp)

დაუშვათ გვაქვს 2 კომპიუტერი და 2 სვიჩი, რომლებიც მიერთებულია ერთმანეთთან. PC1 მისამართია -192.168.1.2, PC2--ს 192.168.1.3, კომპიუტერებს უპრობლემოდ შეუძლიათ ერთმანეთში ინფორმაციის გაცვლა. ასეთ ტოპოლოგიას აქვს ერთი მნიშვნელოვანი მინუსი, არ არსებობს სარეზერვო კავშირო ამ სვიჩებს შორის შესაბამისად თუ ლინკი დაზიანდა სვიჩებს შორის კომპიუტერები ვეღარ შეძლებენ ერთნამეთთან ინფორმაციის გაგზავნას



ამ პრობლემის თავიდან ასაცილებლად, ამატებენ კიდევ ერთ კაბელს:



თუმცა ასეთი ტოპოლოგიას გააჩნია შემდეგი პრობლემა ასეთი ტიპის ქსელში იქმნება (ციკლის) loop-ის საშიშროება.

განვიხილოთ სურათზე მოცემული შემთხვევა:

დავუშვათ H1 დააგენერირა broadcast პაკეტი და გაუგზავნა sw 1- ს.

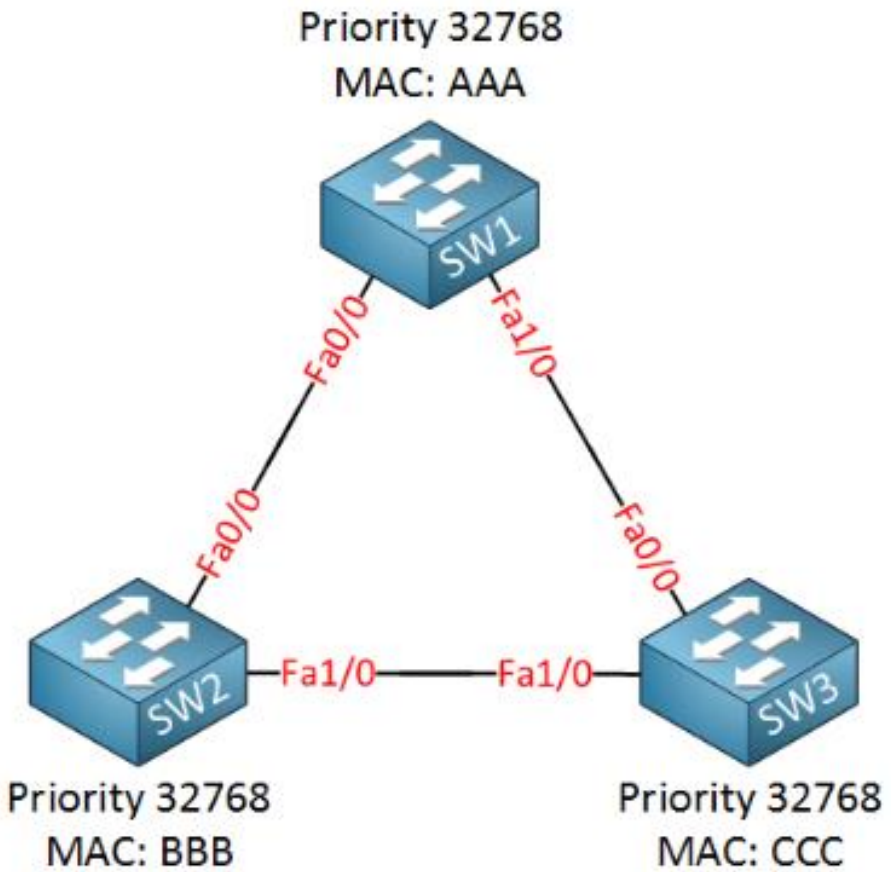
sw 1 მიიღებს ამ პაკეტს და გააგზავნის 1 (Fa0/0 –fa0/0) და 2 (fa0/1 - fa0/2) ლინკზე .

1 ლინკიდან შემოსულ პაკეტს როგორც კი მიიღებს sw 2, გადააგზავნის მას 2 ლინკზე და ასევე

2 ლინკიდან მიღებულ პაკეტს ის გაუშვებს 1 ლინკზე

ამგვარად პაკეტი იმოძრავებს წრეზე, შესაბამისად მოხდება ქსელის Loop-ი

როგორ ხდება loop (მარყუჟის) მოგვარება

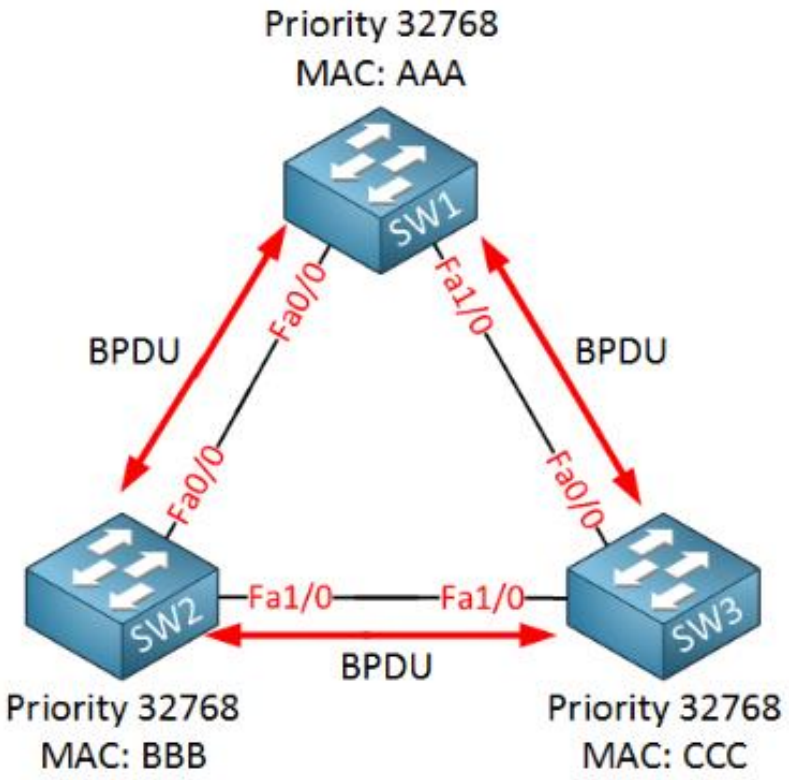


დაუშვათ გვაქვს სვიჩებზე MAC მისამართები (გამარტივებული სახი)

•SW1: MAC AAA
•SW2: MAC BBB
•SW3: MAC CCC

აქაც გვაქვს loop (მარყუჟი) -ს საფრთხე რადგან სვიჩეს შორის იქმნება ლოგიკური წრიული ტოპოლოგია. იმისთვის რომ ქსელში მარყუჟი არ წარმოიქმნას უნდა დაიბლოკოს ერთერთ კავშირის რის შედეგადაც ამ დაბლოკილ კავშირზე არ მიიღება და არც გადაიცემა არანაირი ინფორმაცია.
იმ შემთხვევაში თუ ტოპოლოგიაში რაიმე კავშირი გაწყდა ლინკი ისევ გადავა ფორვარდინგის რეჟიმში და დაიწყებს ინფორმაციის გადცემას და მიღებას.

ამ პრობლემის მოსაგვარებლად გამოიყენებადებს STP პროტოკოლი.



STP პოროტკოლის მუშაობის პრინციპი არის შემდეგი :

პირველ რიგში სვიჩებმა უნდა აირჩიონ ერთერთი სვიჩი როგორც root სვიჩი;

რათა სვიჩებს შორის კავშირი მოხდეს ამ root სვიჩის გავლით.

იმისთვის რომ სვიCებმა აირჩიონ root სვიჩი იყენებენ BPDU მესიჯებს

BPDU მესიჯებში არის მოთავსებული **თითეული სვიჩის ID.**

სვიჩს რომელსაც ექნება ყველაზე დაბალი ID გახდება root სვიჩი

სვიჩის ID შედგება ორი მნიშვნელოვანი კომპონენტისგან:

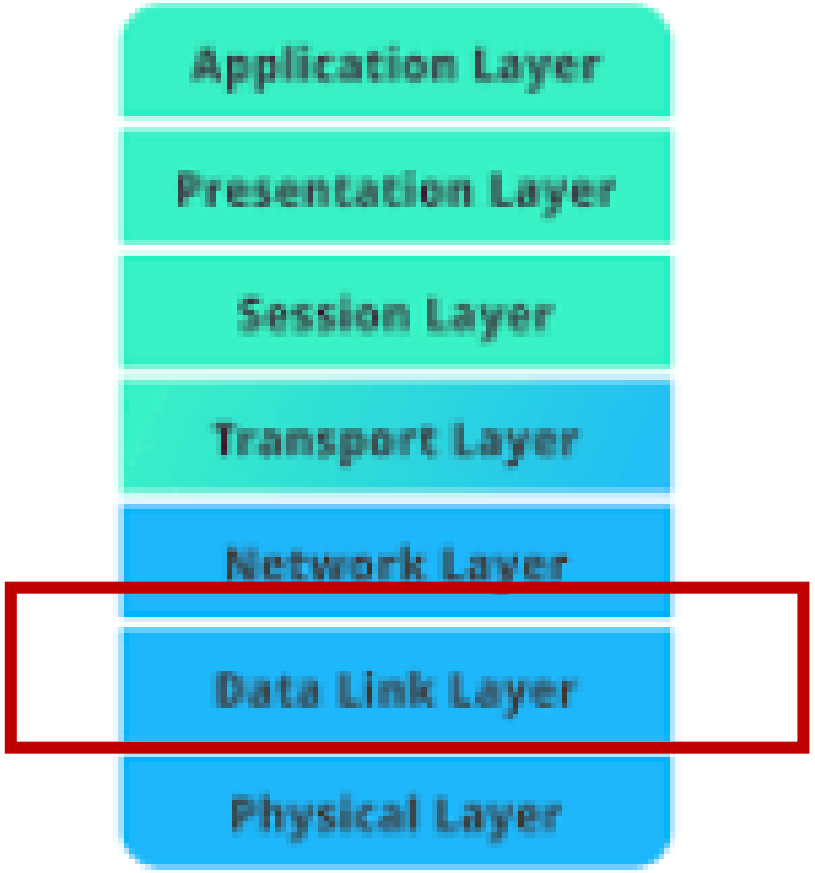
- პრიორიტეტი
- მაკ მისამართი

STP პროტოკოლი

Spanning Tree protocol

STP პროტოკოლი ავტომატურად გამორთავს დუბლირებულ შეერთებას Ethernet-ში უპირატესობა

- სვიჩებს შორის შეერთების საიმედობა;
- კონფიგურაციის შეცდომებისგან დაცვა



STP პროტოკოლი მუშაობს არხის დონეზე

STP პროტოკოლის მუშაობა შედგება სამი ეტაპისაგან:

- ძირითადი ანუ ძირეული **root** სვიჩის არჩევა ;
- უმოკლესი გზის განსაზღვრა ძირითად სვიჩამდე;
- ყველა სხვა შეერთების გათიშვა;

ძირითადი ანუ ძირეული **root switch** არჩევა

STP პროტოკოლის შეტყობინება

სვიჩები კომუნიკაციისთვის იყენებენ **Bridge Protocol Data Units (BPDU)** მესიჯებს

ის შეიცავს ორ მნიშვნელოვან პარამეტრს:

- **MAC** მისამართს
- **Priority**

MAC address და **Priority** ერთად ქმნიან **Bridge ID**

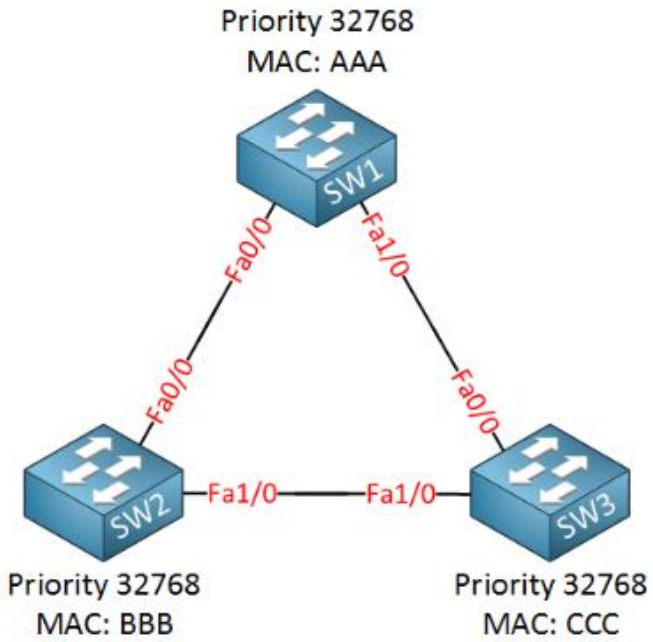
სვიჩის პრიორიტეტი სტანდარტულად არის 32768
ასე რომ აქ მნიშვნელოვან როლს თამაშობს სვიჩის მაკ მისამართი

ძირეული root სვიჩის არჩევა

სვიჩი რომელსაც აქვს უფრო დაბალი თანრიგის მაკ მისამართი ხდება როოტ ბრიჯი მაგალითად თუ ქსელში გვაქვს ვიჩები რომელთა მაკმისამართებია

- SW1: MAC AAA
- SW2: MAC BBB
- SW3: MAC CCC

SW1 გახდება აუცილებლად რუტ ბრიჯი რადგან მას აქვს უფრო დაბალი თანრიგის მაკ მისამართი.



უმოკლესი გზის არჩევა

გამოვთვალოთ უმოკლესი გზა ყველა სვიჩიდან root -მდე. სვიჩებს შორის გზის სიგრძე განისაზღვრება ორი პარამეტრის მიხედვით:

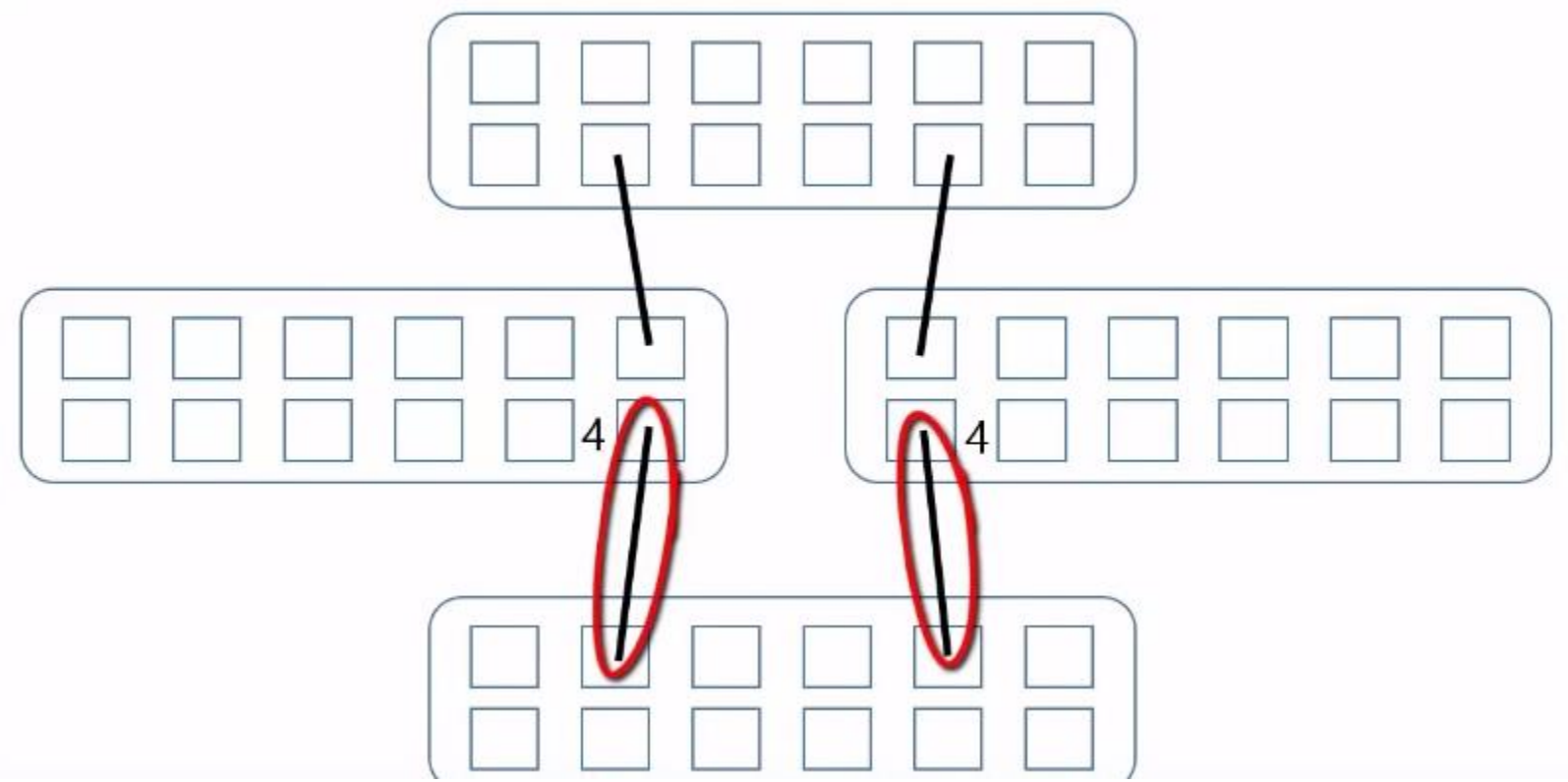
- შუალედური სვიჩების რაოდენობა;
- კავშირის სიჩქარე შუალედურ სვიჩებს შორის.

გადამრთველებს შორის მანძილი განსაზღვრულია IEEE 802.1D სტანდარტით. დავუშვათ, რომ გვაქვს კავშირი 1 გბიტი/წმ სვიჩებთან, ამ შემთხვევაში ჩვენ ვიყენებთ რიცხვს 4, როგორც მანძილის მნიშვნელობა, როგორც ქვემოთ მოცემულ ცხრილში.

შეერთების სიჩქარე	შეერთების ღირებულება IEEE 802.1D
4 Mbit/s	250
10 Mbit/s	100
16 Mbit/s	62
100 Mbit/s	19
1 Gbit/s	4
2 Gbit/s	3
10 Gbit/s	2

კავშირის გათიშვა

შემდეგი ამოცანა არის ერთ-ერთი კავშირის გათiSva ისე, რომ loop არ იყოს. STP პროტოკოლის წესების მიხედვით, თუ ჩვენ გვაქვს ორი გზა root სვიჩამდე, უნდა ავირჩიოთ მინიმალური მანძილის გზა და გამორთოთ სხვა გზა. მაგრამ თუ არის ორი გზა ერთი და იგივე მანძილით . ამ შემთხვევაში გზა უფრო დიდი პორტის მნიშვნელობით გამოირთვება.



Root სვიჩი

პორტების მდგომარეობა STP-ში

Listening - პირველ ეტაპზე როდესაც კაბელი მიუერთდება პორტს, პორტი მუშაობს ამ რეჟიმში, პორტი ამუშავებს მმართველ ინფორმაციას STP, პროტოკოლიდან და არ გადასცემს არანაირ შეტყობინებას.

Learning - მეორეეტაპზე, სახელწოდებით, პორტი იღებს მონაცემებს, მაგრამ არ გადასცემს არსად. მიღებული ფრეიმებიდან, ამოიღება წყაროს მისამართები და გამოიყენება კომუტაციის ცხრილის შესაქმნელად.

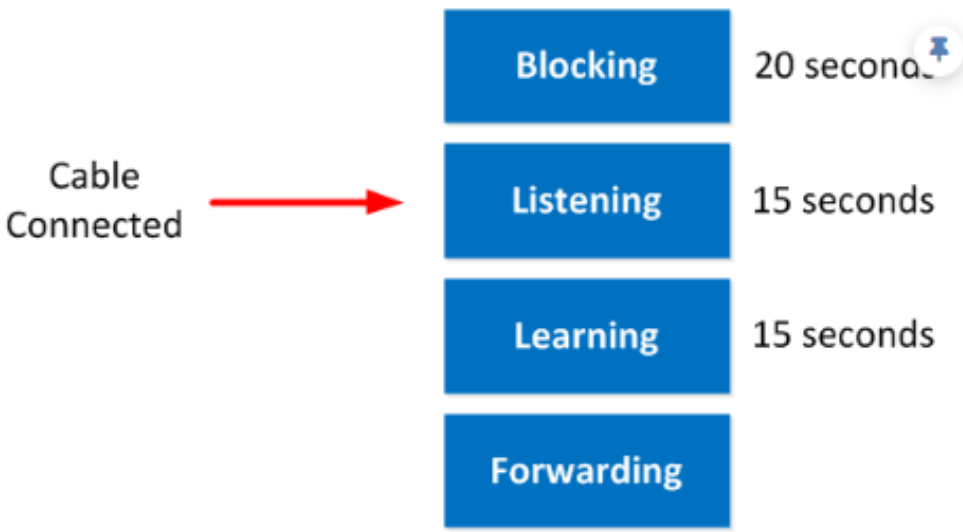
შემდეგ ეტაპზე არსებობს ორი ვარიანტი:

Forwarding თუ აღმოჩნდება, რომ კომპიუტერი ან გადამრთველი უკავშირდება პორტს ციკლის (loop) გარეშე, მაშინ პორტი გადადის გადამისამართების მდგომარეობაში - პორტი იღებს და გადასცემს მონაცემებს, ასევე იღებს და გადასცემს საკონტროლო შეტყობინებას STP პროტოკოლში.

Blocking მაგრამ თუ აღმოჩნდება, რომ გადამრთველი უკავშირდება პორტს და ჩამოყალიბდა loop, მაშინ პორტი გადადის მდგომარეობაში - პორტი იბლოკება პროგრამული უზრუნველყოფის დონეზე ისე, რომ არ იყოს loop.

Disabled – პორტი გამორთულია ადმინისტრატორის მიერ

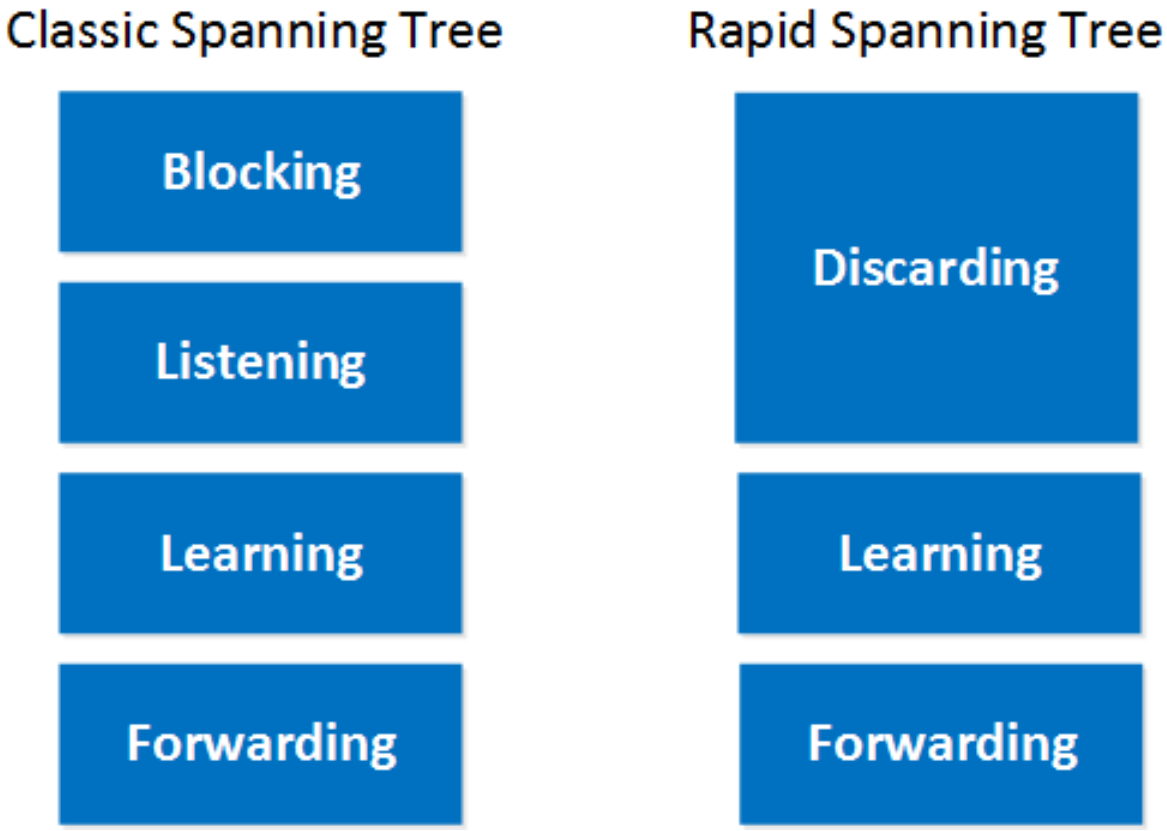
STP პროტოკოლში იმსითვის სვიჩის პორტი კაბელის მიერთების შემდეგ გადავიდეს ინფორმაციის გადაცემის რეჟიმში სჭირდება 50 წამი რაც საკმაოდ დიდი დროა .



ამ დროის შესამცირებლად კი შეიქმნა ახალი პორტოკოლი RSTP რომეიც უფრო სწრაფად ახერხებს პორტის დაბლოკვის რეჟიიდან ინფორმაციის რეჟიმში გადართვას.

მუშაობის ალგორითმი RSTP პროტოკოლის შემთხვევაშიც იდენტურია თუმცა არის განსხვავება პირველ რისი შემცირებულია პორტების რეჟიმები და ის გამოიყურება შემდეგნაირად.

RSTP პროტოკოლი



RSTP პროტოკოლის შემთხვევაში გვაქსვ შემდეგი პორტის სტატუსები

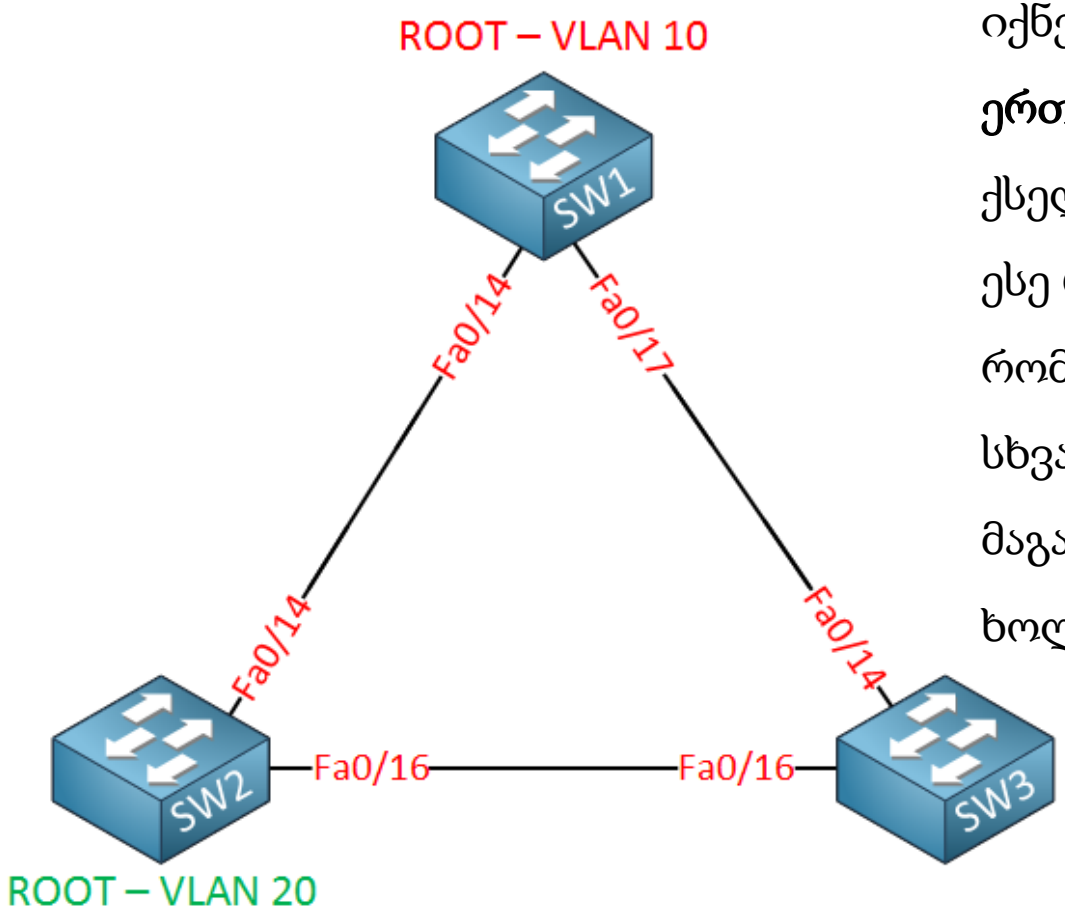
DISCARDIng

LEARNING

FORWARDING

ამ შემთხევაში დისქარდინგ რეჟიმს შეთავსებული აქვს ბლოქინგის და ლისენინგ რეჟიმის ფუნქციები და შესაბამისად პორტის ფორვარდინგში გადასვლა დროის კუთხით უფრო სწრაფად ხდება.

Per VLAN Spanning Tree



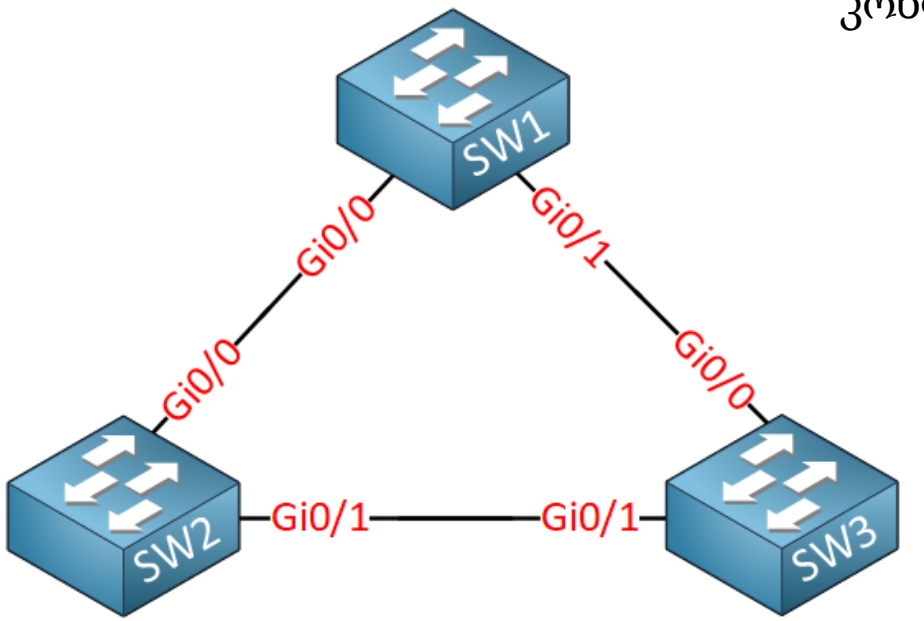
იმ შემთხვევაში თუ ქსელში გვაქვს დიდი რაოდენობით ვიალნი STP პროტოკოლის მიერ არჩეული რუთ სვიჩი ყველა ვილანისთვის იქნება რუთი შესაბამისად მოხდება

ერთი სვიჩის სრული დატვირთვა რადგან მას მოუწევს მთლიანი ქსელის ტრაფიკის დამუშავება.

ესე რომ არ მოხდეს ცისკომ შეიმუშავა PVSTP პროტოკოლი რომლის საშუალებითაც შესაძლებელია

სხვადასხვა ვილანისვის ავირჩიოთ სხვადასხვა რუთ სვიჩი მაგალითად, რუთ სვიჩი ვილან ათისთვის იქნება სვიჩ ერთი ხოლო ვილან 20 ისთვის იქნება სვიჩ 2

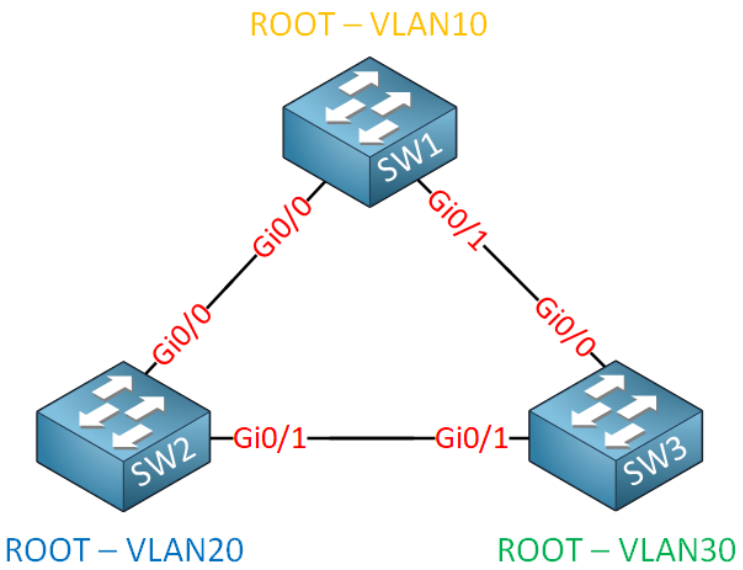
კონფიგურაცია



ნახაზზე გვაქვს სამი სვიჩი და თვითოეულ სვიჩე არის სამი ვლიანი შექმნილი 10,20 და 30

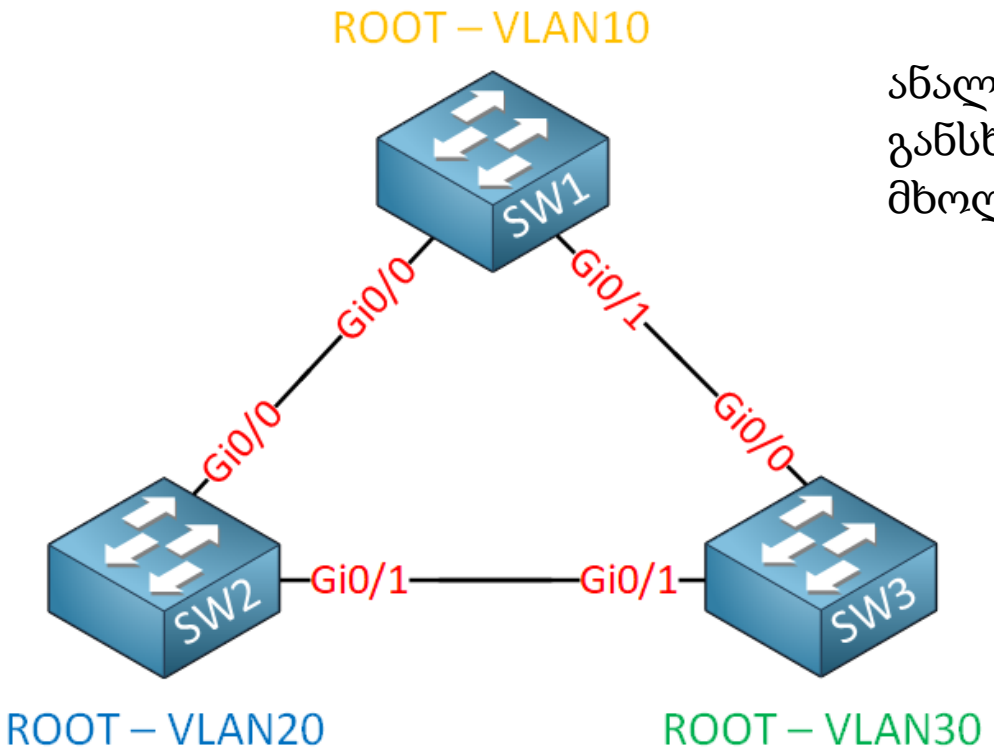
ჩვენი ამოცანა რომ სვიჩ 1 რუთ სვიცი იყოს ვილან 10 ისთვის სვიჩ 2 რუთ სვიჩი იყოს ვილან 20 ისთვის და სვიჩ 3 რუთ ბრიჯი უნდა იყოს ვილან 30 ისთვის.
ამის გასაკეთებლად პრიველ რიგში უნდა შევქმნათ ვილანები სამივე სვიჩზე.

SW1, SW2 & SW3
(config)#vlan 10
(config)#vlan 20
(config)#vlan 30



შემდეგი ეტაპზე უნდა შევიდეთ სვიჩ 1-ზე და „ვუთხრათ“ რომ ის გახდეს ვილან ათისთვის რუთ სვიჩი
ამას ვაკეთებთ შემდეგი ბრძანებით

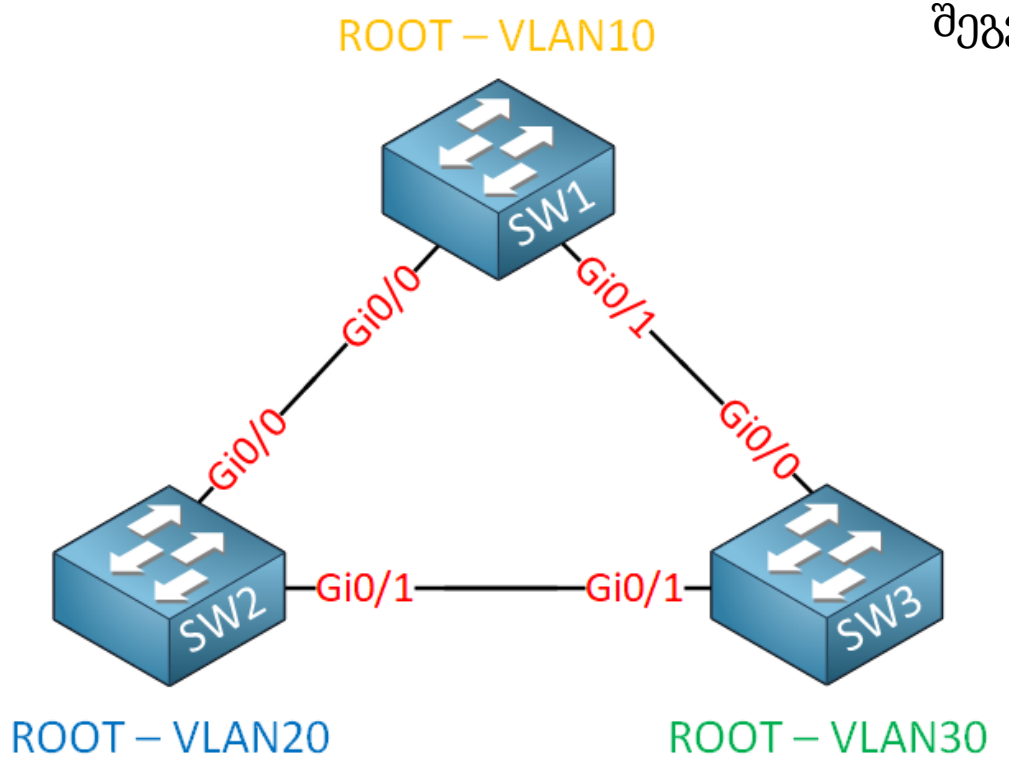
SW1(config)#**spanning-tree vlan 10** root primary



ანალოგიურად უნდა მოვიქცეთ სხვა სვიჩებისთვისაც იმ განსხვავებით რომ სვიჩ 2-ს უნდა უთხრათ იგივე ბრძანება მხოლოდ ვილან 20 -ისთვის ხოლო სვიჩ 3-ს კი ვილან 30 ისთვის

SW2(config)#**spanning-tree vlan 20** root primary

SW3(config)#**spanning-tree vlan 30** root primary



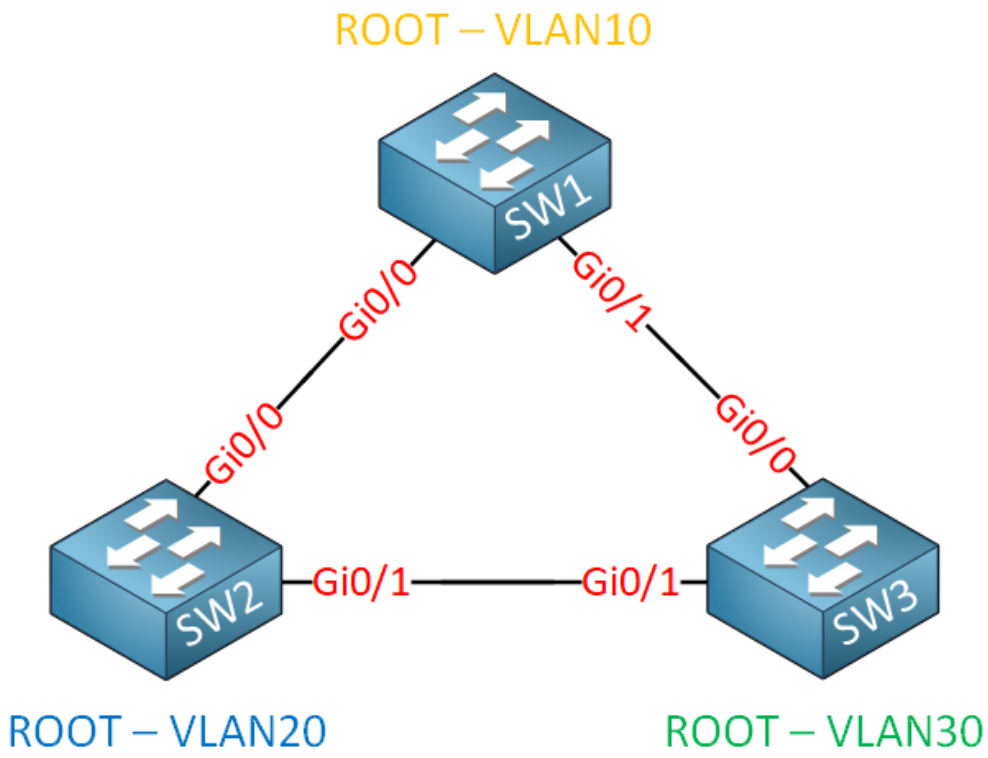
იმისთვის რომ შევამოწმოთ კმფიგურაცია თვითოეულ ვიჩზე შეგვიძლია გამოვიყენოთ შემდეგი ბრძანება **show spanning-tree**

SW1#show spanning-tree

SW1#show spanning-tree

VLAN0010
Spanning tree enabled protocol ieee
Root ID Priority 24586
Address 5254.001a.935a
This bridge is the root
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec

ეს ბრძანება გამოგვიტანს ინფორაციას მთლაიან სპანინგ თრის შესახებ და მათ შორის გვეტყვის რომ sw1 რუთ ბრიჯი ვილან 10 ისთვის

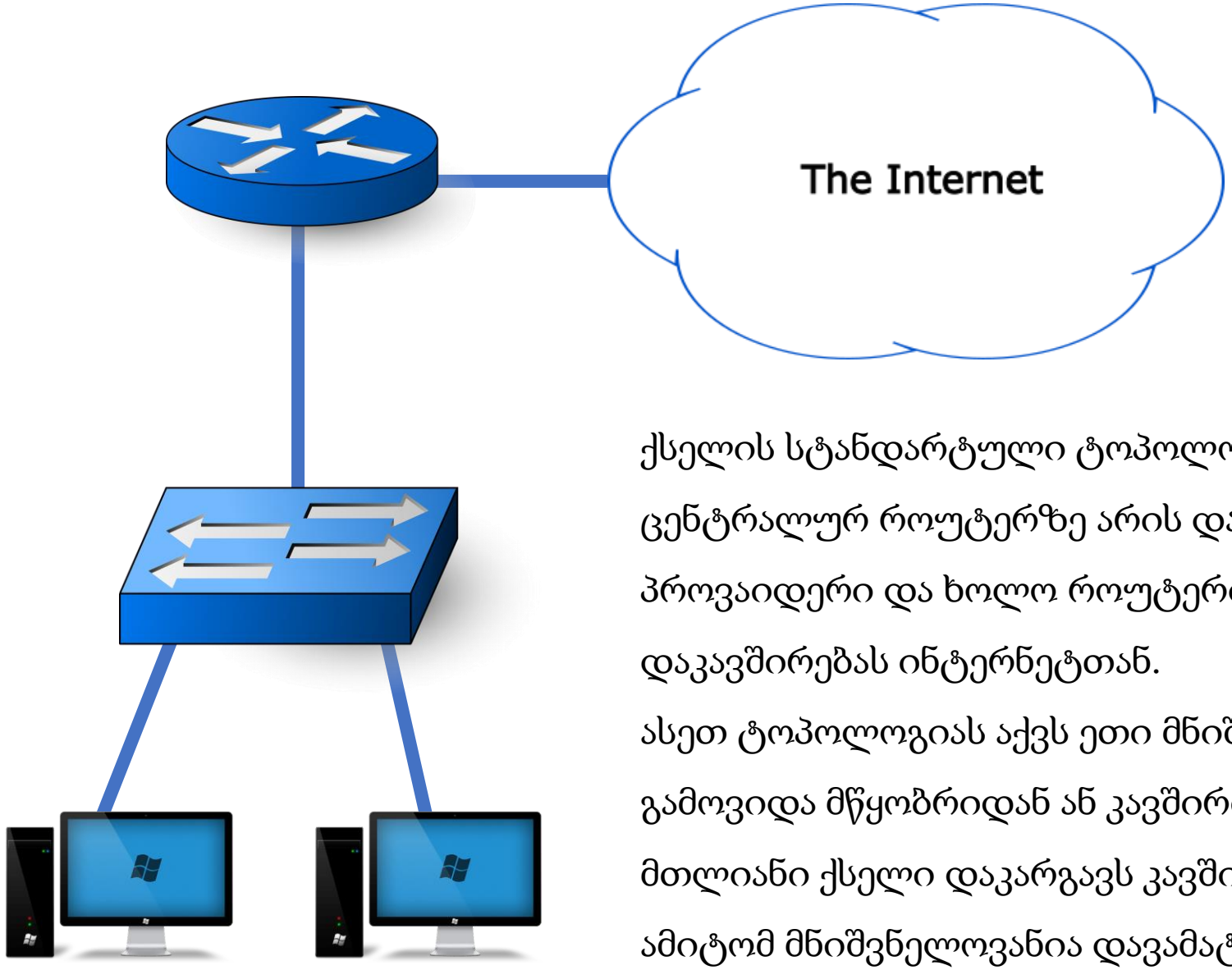


ქსელში შესაძლოა ასევ გვქონდეს რუთ სვიჩის შემცვლელი სვიჩი, იმ შემთხვევისთვის თუ რუთ სვიჩი გამოვა მწყობრიდან ხელახლა რომ არ ჩატარდეს რუთ ის არჩევნები.
ჩვენ შეგვიძლია ერთერთ სვიჩს ვუთხრათ რომ ის გახდეს რუთ სვიჩი რომელიმე კონკრეტული ვილანისთვის მაშინვე როგორც კი მთავარი რუთ სვიჩი გამოვა მწყობრიდან.
განვიხილოთ ისევ ზედა მაგალითი და სვიჩ ორს დავაკონფიგურიროთ ისე რომ ის გახდეს რთ ბრიჯი ვილან 10 ისთვის მაშინ როგორც კი სვიჩ 1 გაითიშება ან რაიმე პრობლემა დაფიქსირდება.
ამისთვის უნდა შევიდეთ სვიჩ 2 ზე და „ვუთხრათ“ შემდეგი ნრძანება

SW2(config)#spanning-tree vlan 10 root secondary

ამ ბრძანების შემდეგ სვიჩ ერთი დარჩება რუთ სვიჩად თუმცა თუ რაიმე პრობლემა დაფიქსირდა სვიჩ 1 მიმარტებაში სვიჩ 2 ავტომატურად შეითავსებს ვილან 10 ის რუთ სვიჩობას.
შესაბამისად სვიჩ 2 გახდება რუთ სვიჩი როგორც 20 ასევე 10 ვილანისთვის.

Gateway Redundancy
სარეზერვო Gateway

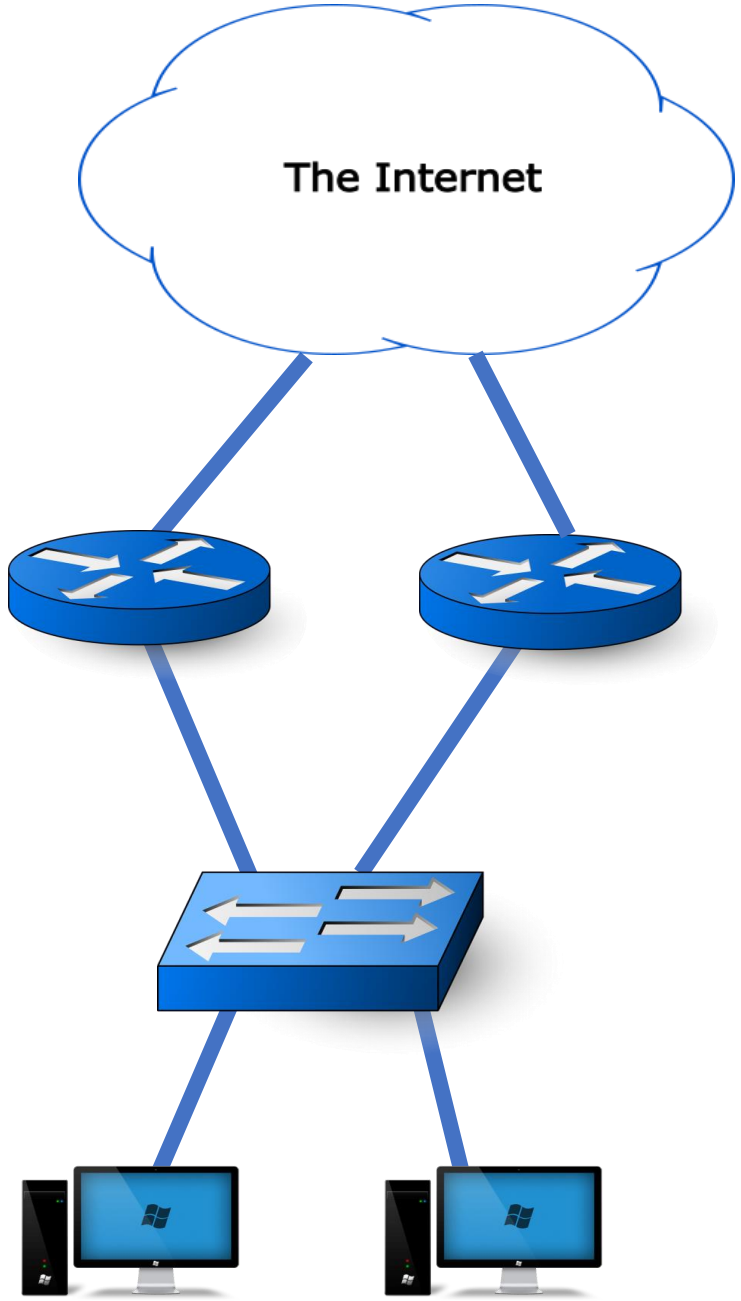


ქსელის სტანდარტული ტოპოლოგია მსგავსად გამოიყურება.

ცენტრალურ როუტერზე არის დაერთებული ინტერნეტ სერვის პროვაიდერი და ხოლო როუტერი უზრუნველყოფს შიდა ქსელის დაკავშირებას ინტერნეტთან.

ასეთ ტოპოლოგიას აქვს ეთი მნიშვნელოვანი პრობლემა თუ როუტერი გამოვიდა მწყობრიდან ან კავშირი სვიჩსა და როუტერს სორის დაიკარგა მთლიანი ქსელი დაკარგავს კავშირს გარე ქსელთან.

ამიტომ მნიშვნელოვანია დავამატოთ კიდევ ერთი როუტერი ჩვენს ქსელს და ტოპოლოგიას მივცეთ შემდეგნაირი სახე.



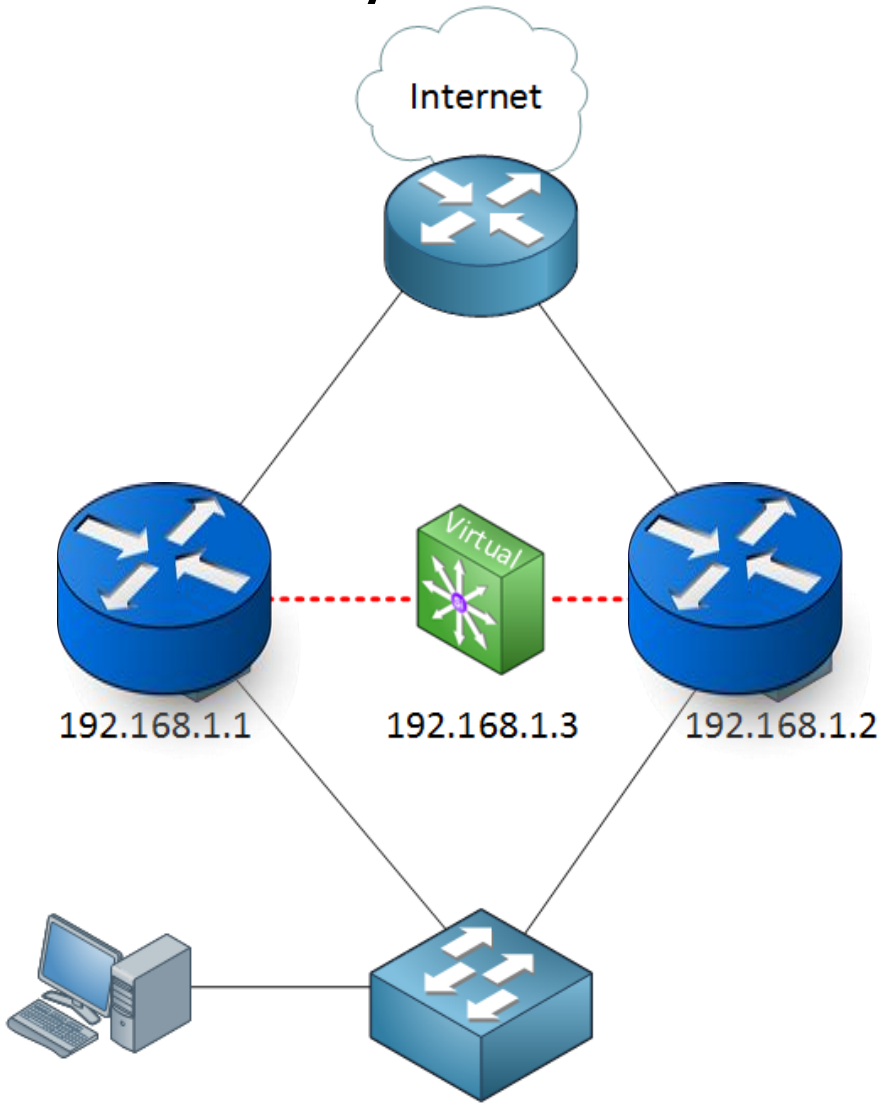
ქსელში კიდევ ერთი როუტერის დამატებით თუ პირველი როუტერი გამოვა მწყობრიდან მეორე როუტერი შეძლებს სვიჩებზე მიერთებულ მოწყობილობებს მიაწოდოს ქსელი.

თუმცა აქ რჩება შემდეგი პრობლემა კომპიუტერს შეუძლია აიღოს მხოლოდ ერთი დეფაულტ გეითვეი ამითომ ჩვენ შეგვიძლია მიუთითოთ ან ერთი როუტერის ip მისამართი როგორც გეითვეი ან მეორე როუტერის გეითვეი.

ასეთ შემთხვევაში თუ ერთერთი როუტერი იქნება მოწყობილობებზე გაწერილი როგორც დეფაულტ Gateway მისი მწყობრიდან გამოსვლის შემთხვევაში საჭიროა ყველა მოწყობილობას შუცვალოთ დეფაულტ გეითვეის ip მისამართი რაც გამოიწვევს ქსელის მუშაობის შეფერხებას.

სორედ ამ პრობლემის მოსაგვარებლად გამოიყენება Gateway Redundancy პროტოკოლები

Gateway Redundancy



Gateway Redundancy პროტოკოლი მუშაობს შემდეგნაირად იქმნება ერთ ვირტუალური როუტერი რომელსაც ენიჭება ip მისამართი და ამ ვირტუალური როუტერის ip მისამართი იქნება დეფაულტ Gateway ქსელში აღნიშნული მოწყობილობებისთვის. ფიზიკური როუტერებიდა ერთერთი იქნება აქტიურ მდგომარეობაში ხოლო მეორე გადავა სთენდბაი (მოლოდინის რეჟიმში) აქტიურ როუტერს მიემაგრება ვირტუალური როუტერი.

თუ აქტიური როუტერი გამოვა მწყობრიდან ამ შემთხვევაში მოლოდინის რეჟიმში მყოფი როუტერი ავტომატურად გახდება აქტიური და ეს ვირტუალური როუტერი მიმაგრდება ამ აქტიურ როუტერზე შესაბამისად ქსელი გააგრელებს მუშაობას ჩვეულ რეჟიმში და საბოლო მოწყობილობები არ იგრძნობენ ქსელის შეფერხებას

Gateway Redundancy

არსებობს ორი ძირითადი პროტოკოლი რომელიც უზრუნველყოფს Gateway Redundancy ქსელის ტოპოლოგიის შექმნას.

- **HSRP (Hot Standby Routing Protocol)**
- **VRRP (Virtual Router Redundancy Protocol)**

ამ ორივე პროტოკოლის მუშაობის პრინციპი ერთნაირია. სხვაობა ის რომ HSRP არის ცისკოს მიერ შემოუშავებული პროტოკოლი ხოლო VRRP არის საერთაშორისო სტანდარტი და მუშაობს ნებისმიერ მოწყობილობაზე.

HSRP

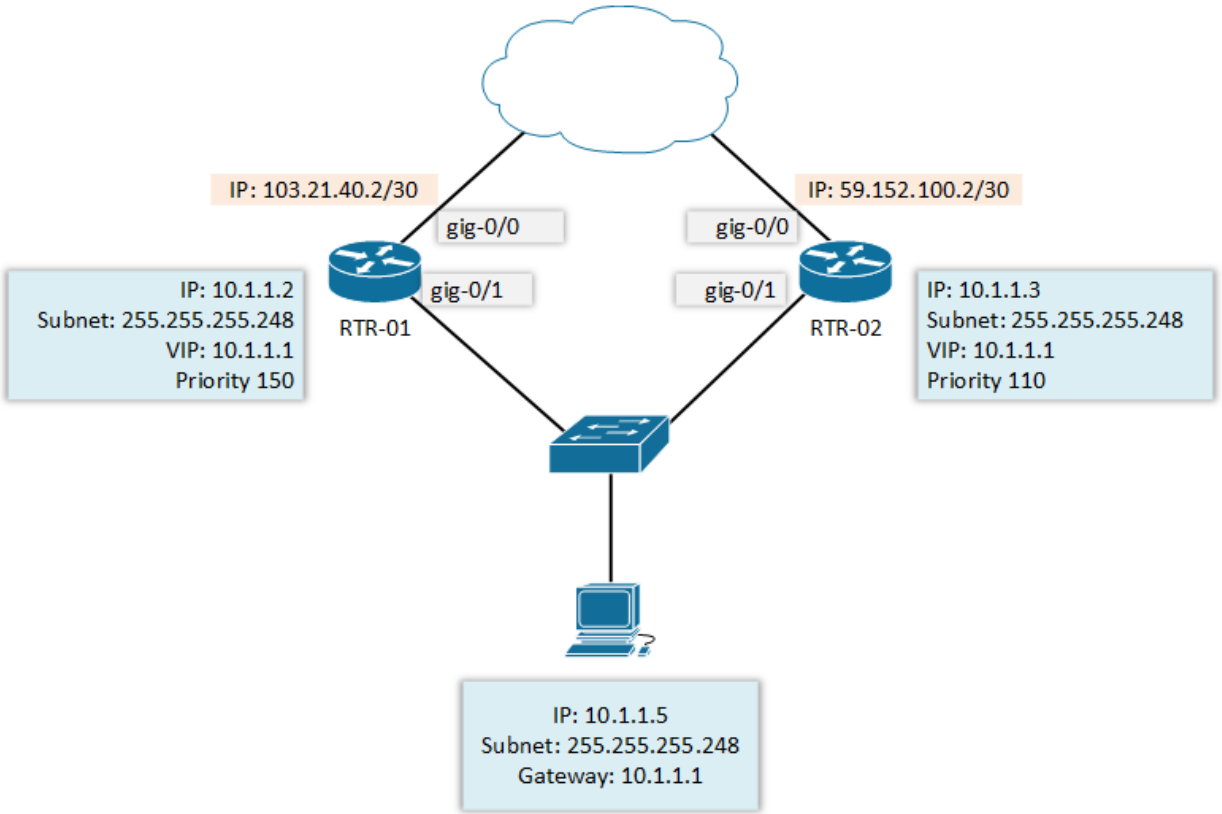
როგორ მუშაობს HSRP? ის აერთიანებს რამდენიმე როუტერს და ქმნის ვირტუალურ როუტერს რომელიც მიებმება აქტიურ როუტერს.

ქსეთ ჯგუფში არის მხოლოდ ერთი აქტიური როუტერი ხოლო დანარჩენები არიან სთენდბაი როუტერები. რომელი როუტერი იქნება აქტიური და რომელი როუტერი იქნება პასიური ამის არჩევა ხდება როუტერის პრიორიტეტის მიხედვით .

აქტიური როუტერის მთავარი ფუნქცია მიმაგროს ვირტუალური როუტერი და დაამუშაოს მთლიანი ქსელის ტრაფიკი ის ასევე სთენდბაი როუტერებს უგზავნის ჰელოუ პაკეტებს პერიოდულად იმისთვის რომ მოახდინოს მათთან კომუნიკაცია იმის შესახებ, რომ ყველაფერი არის წესრიგში.

სთენდბაი როუტერებს გააჩნიათ თავიანთი მოლოდინის დრო რა დროის განმავლობაშიაც ელოდებიან აქტიური როუტერისგან ჰელოუ პაკეტებს თუ ამ დროის განმავლობაში როუტერისგან არ მიიღეს ჰელოუ პაკეტები მაშინ სთენდბაი რეჟიმში მყოფი როუტერი გადავა აქტიურ როუტერად და მიბავს ვირტუალურ როუტერს.

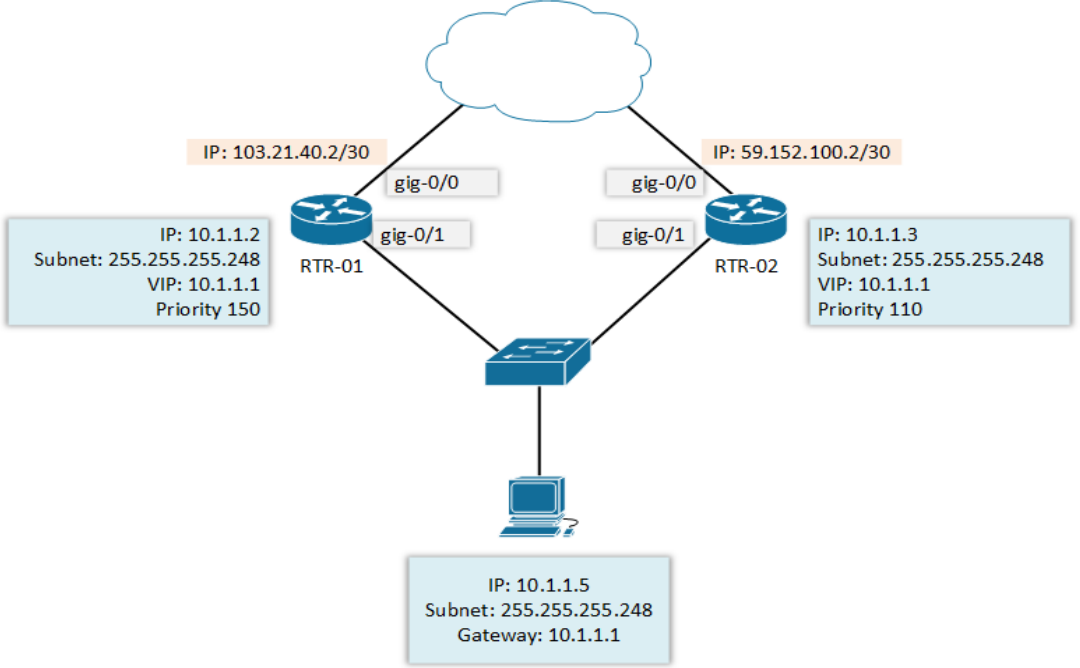
ჰელოუ პაკეტები იგზავნება 224.0.0.2 მულთიქასთ ip მისამართზე




```
RTR-01#configure terminal
RTR-01(config)#
RTR-01(config)#interface gigabitEthernet 0/0
RTR-01(config-if)#ip address 103.21.40.2 255.255.255.252
RTR-01(config-if)#no shutdown
RTR-01(config-if)#exit
RTR-01(config)#
RTR-02#configure terminal
RTR-02(config)#
RTR-02(config)#interface gigabitEthernet 0/0
RTR-02(config-if)#ip add 59.152.100.2 255.255.255.252
RTR-02(config-if)#no shutdown
RTR-02(config-if)#exit
RTR-02(config)#
```

```
RTR-01(config)#interface gigabitEthernet 0/1
RTR-01(config-if)#ip address 10.1.1.2 255.255.255.248
RTR-01(config-if)#no shutdown
RTR-02(config)#interface gigabitEthernet 0/1
RTR-02(config-if)#ip add 10.1.1.3 255.255.255.248
RTR-02(config-if)#no shutdown
```

პირველ რიგში აუცილებელია როუტერებზე გავწეროტ ip მისამართები აუცილებელია როუტერების შიდა ინტერფეისები ყვნენ ერთიდაიმავე ქსელში წინაღმდეგ შემთხვევაში არ დადგება მათ შორის HSPR კავშირი




```
RTR-01(config)#interface gigabitEthernet 0/1
RTR-01(config-if)#standby 1 ip 10.1.1.1
RTR-01(config-if)#standby 1 priority 150
RTR-01(config-if)#
```

```
RTR-02(config)#interface gigabitEthernet 0/1
RTR-02(config-if)#standby 1 ip 10.1.1.1
RTR-02(config-if)#standby 1 priority 110
RTR-02(config-if)#
```

შემდეგი ეტაპზე როუტერებზე უნდა შევქმნათ ვირტუალური ჯგუფები და ასევე შევქმნათ ვირტუალური რომელზეაც გავწერთ იპ მისამართს ასევე როუტერებს უნდა მიუთითოთ პრიორიტეტები რაა მოხდეს აქტიური როუტერის და სთენდბაი როუტერის არჩევა.
ამისთვის უნდა შევიდეთ როუტერების შიდა ინტერფეისებზე და შეგვაქვს standby ბრძანება, რომლის შემდეგაც ეთითება ჯგუფის ნომერი, მაგალითად 1 და შემდეგ ip ბრძანების შემდეგ ვწერთ ვირტუალური როუტერის p მისამართს ამ შემთხვევაში 10.1.1.1
შემდეგ ისევ სევიტან standby ბრძანებას ვწერთ ისევე ჯგუფის ნომერს და ვკრიპავთ priority ბრძანებას სადაც ვუთითებთ პრიორიტეტს, მაგალითად 150 ს. როუტერის რომელსაც ექება მაღალი პრიორიტეტი იქნება აქტიური როუტერი.
შემდეგ როუტერზეც ანალოგიურად უნდა მოხდეს კონფიგურაციის გაწერა გასათვალისწინებელია ის რომ
Standby ბრძანების შემდეგ მითითებული ჯგუფის ნომერი ორივე როუტერზე უნდა იყოს იდენტური ასევე იდენტური უნდა იყოს ვირტუალური გეუთვეის IP მისამართი. პრიორიტეტI ორივე როუტერზე უნდა იყოს სხვადასხვა.
ვირტუალური **IP** მისამართი იქნება დეფაულტ გეითვეი ქსელში ჩართული მოწყობილობებისთვის

HSRP Timers

- **HSRP** პროტოკოლიყოველ სამ წამში აგზავნის ჰელოუ პაკეტებს სთენდბაი როუტერებთან
- ხოლო მოლოდინის რეჟიმი არის 10 წამი

კონფიგურაციის შემოწმება

show standby ბრძანებით როუტერი გამოიტანს შემდეგ მონაცემებს

Group 1 ჯგუფის ნომერი

State is Standby

3 state changes, last state change 00:03:33

Virtual IP address is 10.1.1.1 ვირტუალური როუტერის მისამართი

Active virtual MAC address is 0000.0c07.ac01 აქტიური ვირტუალური როუტერის მაკ მისამართი

Local virtual MAC address is 0000.0c07.ac01 ლოკალური როუტერის მაკ მისამართი

Hello time 3 sec, hold time 10 sec ჰელოუ პაკეტების გაგზავნის და მიღების დრო

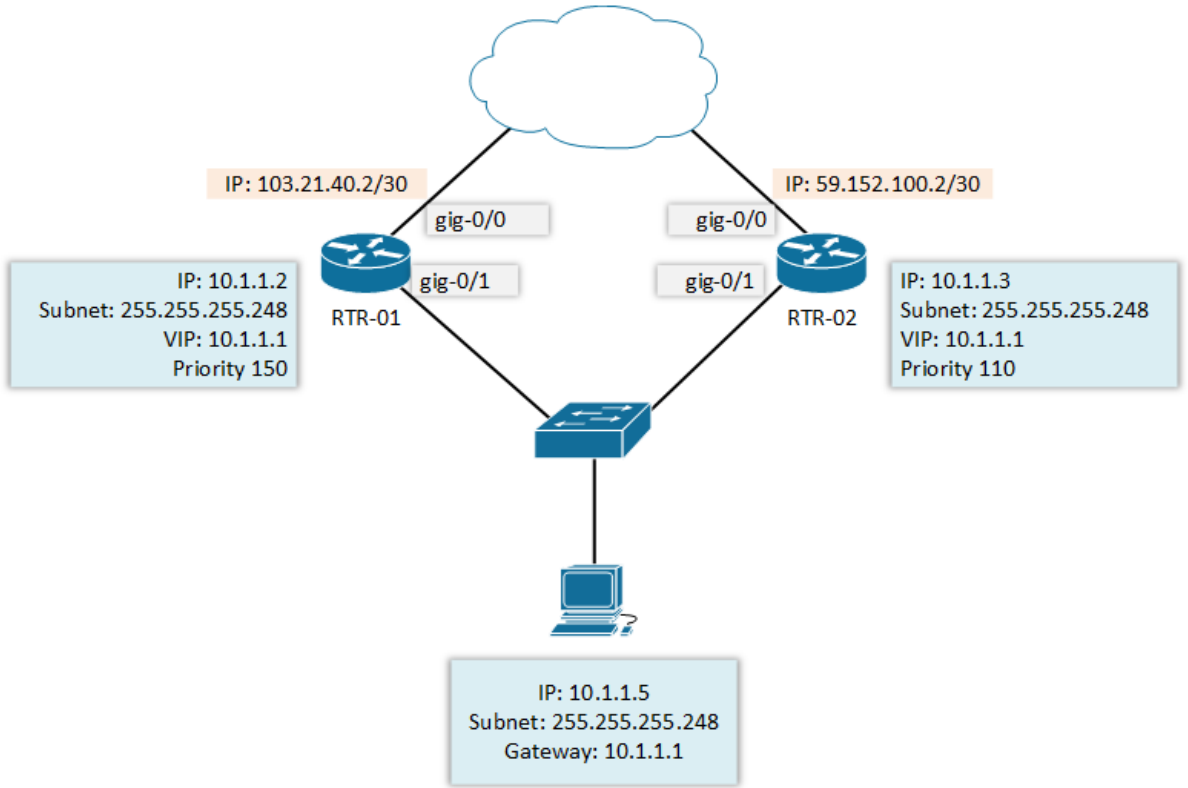
Next hello sent in 0.144 secs რამდენ ხანში გაიგზავნება შემდეგი ჰელოუ პაკეტი

Active router is 192.168.1.2, priority 150 აქტიური როუტერის იპ მისამართი და პრიორიტეტი

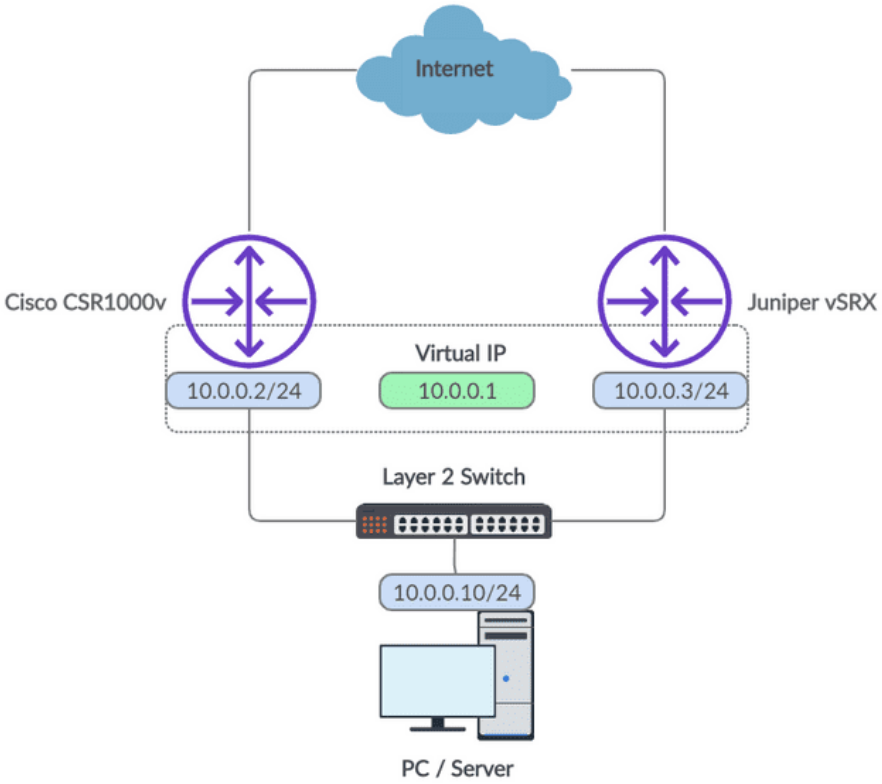
- უფრო შემოკლებული ინფორმაციის მისაღებად იმისთვის რომ გავიგროთ რომელი როუტერია აქტიური და რომელი სთენდბაი გამოვიყენებთ შემდეგ ბრძანებას **show standby brief**

show standby brief

Interface	Grp	Pri P	State	Active	Standby	Virtual IP
Gig0/1	1	150	Active	local	10.1.1.2	10.1.1.1



VRRP-ის კონფიგურაცია



VRRP ის კონფიგრაციაც არის ანალოგიური როგროც HSRP ერთი მარტივი განსხვავებით VRRP ის ჯგუფის შესაქმნელად ვიყენებთ ბრძანებას vrrp

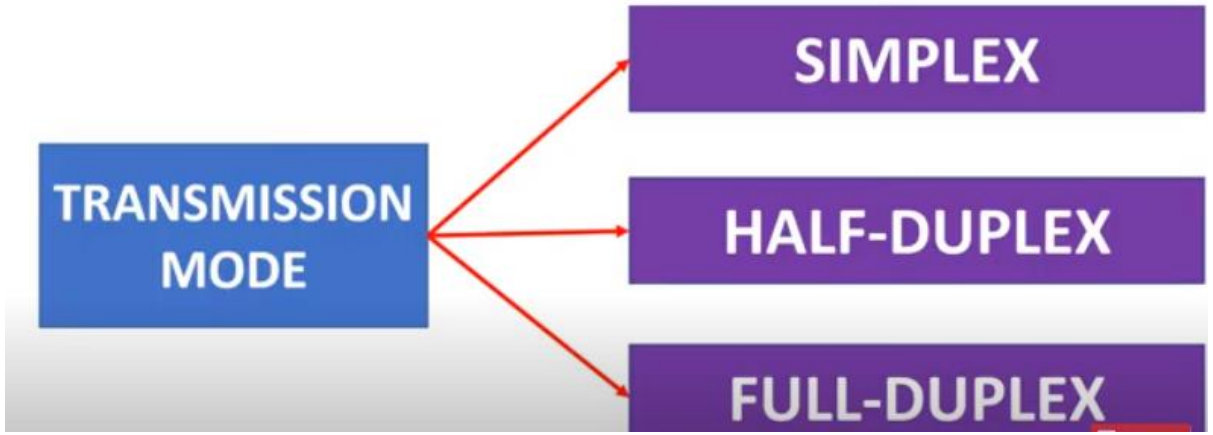
```
interface GigabitEthernet1
 ip address 10.0.0.2 255.255.255.0
 vrrp 1 ip 10.0.0.1
 vrrp 1 priority 90
 no shut
```

interface GigabitEthernet1
ip address 10.0.0.2 255.255.255.0 ინტერფეისზე იპ მისამართის მინიჭება
vrrp 1 ip 10.0.0.1 ამ ბრძანებით იქმნება ვირტუალური როუტერი და ენიჭება იპ მისამართი
vrrp 1 priority 90 ხდება როუტერის პრიორიტეტის განსაზღვრა

ლექცია 13

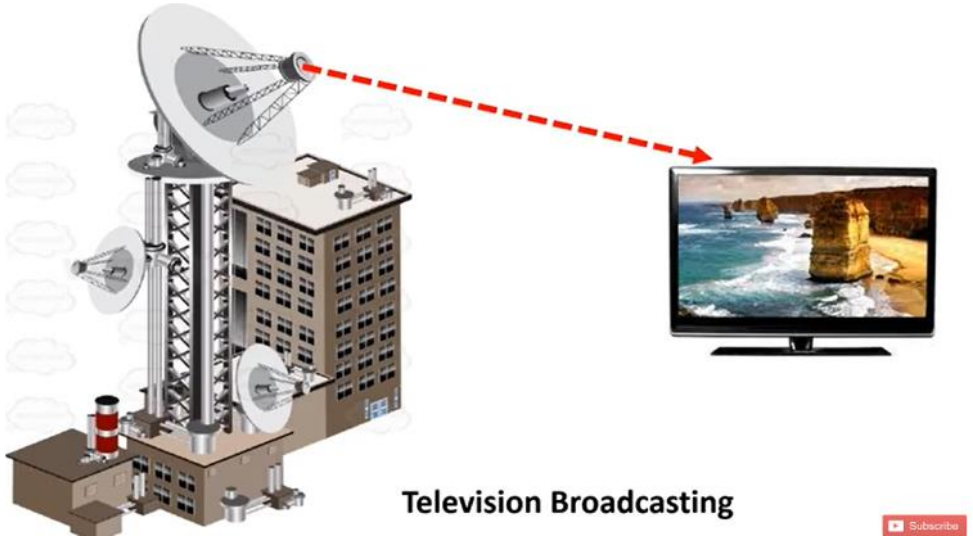
ხშირად იდენტიფიცირებული პრობლემები; პრობლემების გადაჭრის ნაბიჯები; ქსელურ მოდელში პრობლემების იდენტიფიცირებისა და აღმოფხვრის პრაქტიკული მაგალითები

გადაცემის რეჟიმი



სიმპლექსის რეჟიმი, რომელშიც სიგნალები ერთდროულად მხოლოდ ერთი მიმართულებით გადეცემა. ერთ ბოლოზე არის გადამცემი (გადამცემი) და მეორეზე მიმღები (მიმღები) და უკუ მიმართულებით გადაცემა არ ხდება.

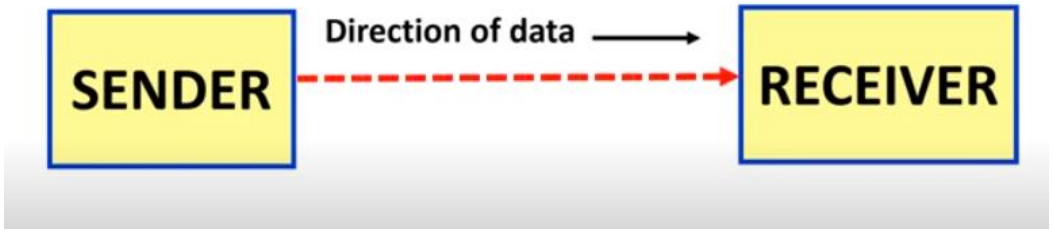
მაგალითები:
სატელევიზიო გადაცემები, Broadcasting

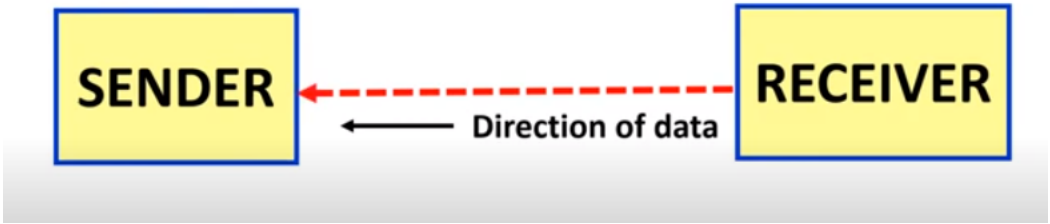


Keyboard sending data to the monitor

HALF-DUPLEX

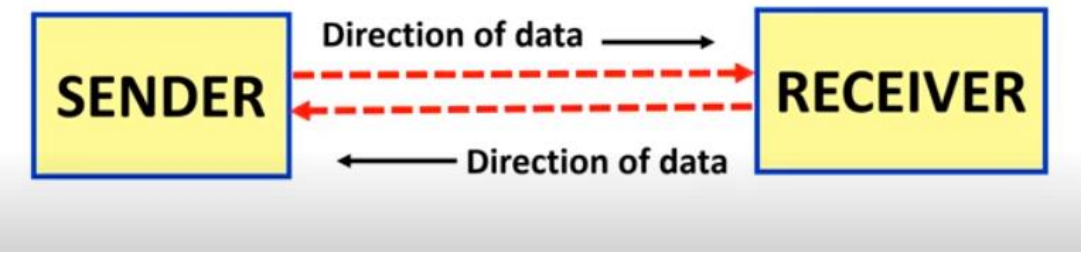
მაგალითი `
რაცია





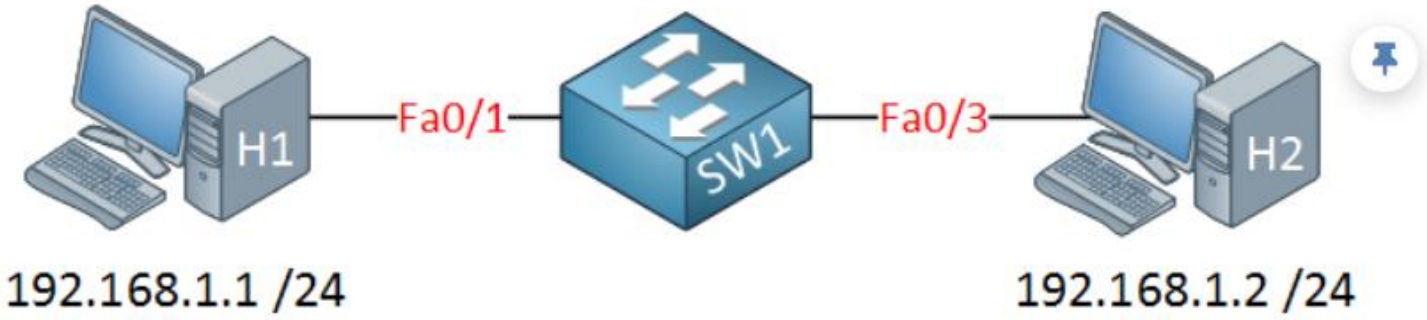
ნახევრად დუპლექსური კავშირები არის კავშირები,
რომლებშიც მონაცემები გადადის ორივე მიმართულებით,
მაგრამ არა ორივე მიმართულებით ერთდროულად

FULL-DUPLEX



Full duplex არის გადაცემა ორივე მიმართულებით ერთდროულად. მუშაობის ეს მეთოდი ყველაზე ეფექტურია. ამის მაგალითია სატელეფონო კომუნიკაცი

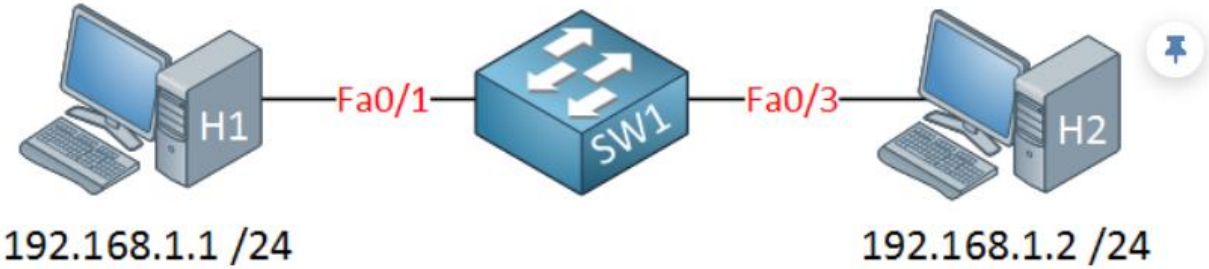
მაგალითი
სატელეფონო ქსელები



განვიხილოთ ტოპოლოგია ერთ სვიჩზე გვაქვს მიერთებული ორი მოწყობილობა H1 და H2 რომლებსაც მითითებული აქვთ თავისი ip მისამართები.
შევამოწმოთ მათ შორის კავშირი ping ის საშუალებით

```
C:>Documents and Settings>H1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

აღნიშნულ შემთხვევაში მოწყიობილობებს შორის კავშირი არ არის, დავიწყოთ აღნიშნული ქსელის პრობლემის მოგვარებაზე მუშაობა ასეთ შემთხვევაში ქსელის შემოწმება იყოფა რამდენიმე ეტაპად:
ესა არის ფიზიკური ქსელის შემოწმება. **უნდა შემოწმდეს ფიზიკური მოწყობილობები** : კაბელი, ჯეკი, ქსელის კარტა და ასე შემდეგ ანუ ფიზიკური ნაწილი თუ ფიზიკური ნაწილი სრულ წესრიგშია გადავდივართ უკვე შემდეგ ნაწილზე რაც გულისხმობს იმას რომ შევამოწმოთ **კონფიგურაცია**, ხომ არ გვაქვს რაიმე ტიპის შეცდომა



ამისთვის, შევიდეთ სვიჩზე და გაუშვათ ბრძანება :
show interfaces fa0/1 რომელიც გამოიტანს სრულ ინფორმაციას აღნიშნულ ინტერფეისზე.
ამ ბრძანების შედეგად გამოჩნდება შემდეგი ინფორმაცია ჩვენს ეკრანზე.

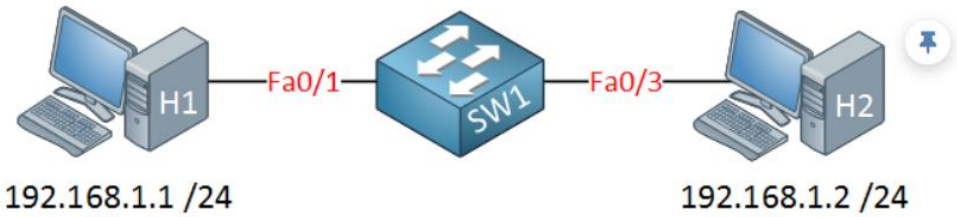
```
SW1#show interfaces fa0/1
FastEthernet0/1 is down, line protocol is down (notconnect)
  Hardware is Fast Ethernet, address is 0011.bb0b.3603 (bia 0011.bb0b.3603)
  MTU 1900 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Half-duplex, Auto-speed, media type is 10/100BaseTX
  input flow-control is off, output flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:26:47, output 00:19:17, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
```

თუ კარგად დავაკვირდებით აქ მკაფიოდ ჩანს რომ პორტი არის გამორთულია
FastEthernet0/1 is down, line protocol is down (notconnect).

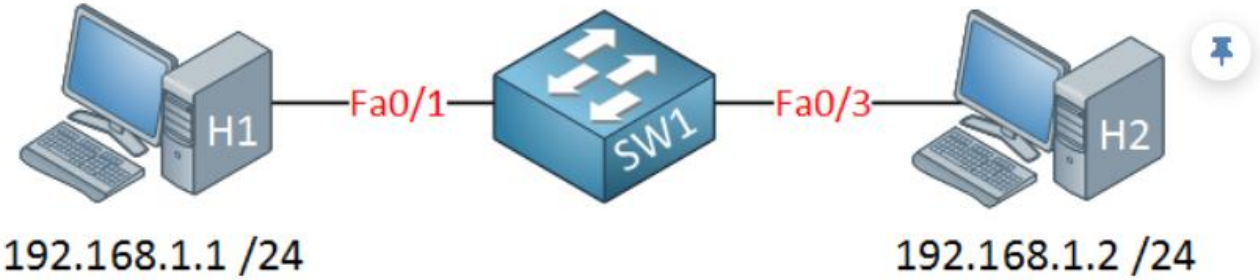
ხოლო მიზეზი მითითებულია შემდეგ სტრიქონში
Half-duplex, Auto-speed, media type is 10/100BaseTX

Half-duplex, Auto-speed, media type is 10/100BaseTX

გავარჩიოთ ეს ჩანაწერი:
Half-duplex: ნიშნავს რომ პორტი მუშაობს ამ ეტაპზე **Half-duplex** ტექნოლოგიაზე თუმცა აუცილებლად გასათვალისწინებელია ის რომ სვიჩი როუტერი და ასევე დღევანდელი ქსელური მოწყობილობების უმრავლესობა მუშაობს **FULL-duplex** ტექნოლოგიაზე.
Auto-speed: გულისხმობს იმას რომ პორტი იმუშავებს იმ სიჩქარეზე რასაც უკარნახებს მასზე მიერთებული მოწყობილობა ამ შემთხვევაში რადგან პორტი არის 100 მეგაბიტი გამტარობის თუ ჩვენ მასზე მივაერთებთ მოწყობილობას რომელიც იმუშავებს 10 მეგაბიტ გამტარობაზე პორტს არ შეექმნება პრობლემა და თავისუფლად დაამუშავებს მის მიერ გამოგზავნილ ტრაფიკს უბრალოდ მათ შორის გამტარობა იქნება მაქსიმუმ 10 მეგაბიტი
10/100BaseTX : წარმოადგენს სტანდარტს კაბელის



ჩვენს შემთხვევაში პრობლემა იდენთიფიცირებულია პორტი მუშაობს Half-duplex ტექნოლოგიაზე როდესაც მასზე მიერთებული კომპიუტერი მუშაობს FULL-duplex ტექნოლოგიაზე



იმისთვის რომ შევცვალოთ დუპლექსის ტიპი უნდა შევიდეთ ჩვენთვის სასურველ პორტზე და მიუთითოთ შემდეგი ბრძანება

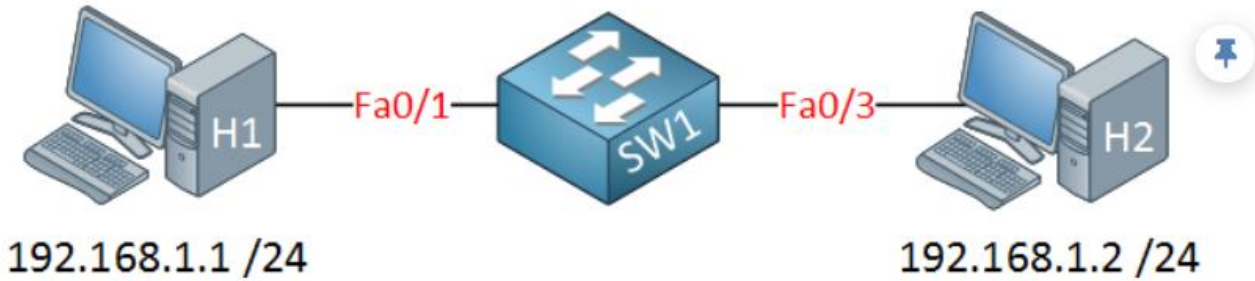
```
SW1(config)#interface fa0/1
SW1(config-if)   duplex full
```

duplex full ბრძანებით პორტი გადაგვყავს **FULL-duplex** გადაცემის ტექნოლოგიაზე

როგორც კი შევიყვანთ ამ ბრძანებსას ეკრანზე გამოჩნდება შემდეგი ჩანაწერი

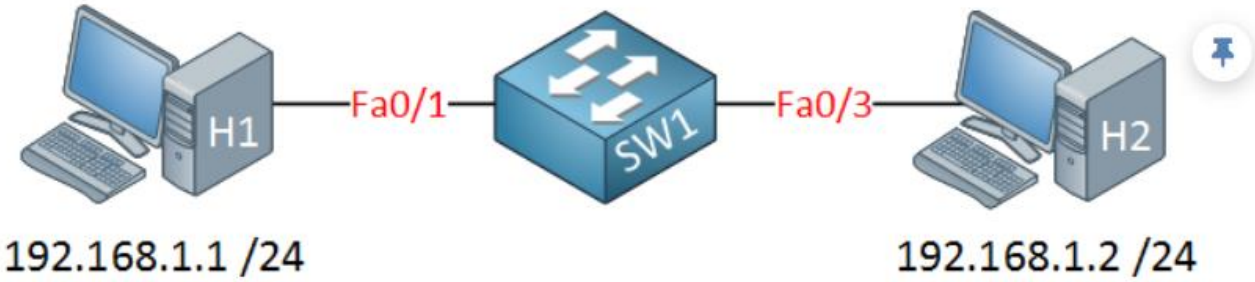
```
SW1#
%LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up
```

რაც ნიშნავს იმას რომ პორტი ამ ეტაპზე ჩაირთო ამუშავდა




```
C:Documents and SettingsH1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

ახლა კიდევ ერთხელ გავტესტოთ კავშირი მოწყობილობებს შორის როგორც ვხედავთ მოწყობილობებს შორის კავშირი დადგა.



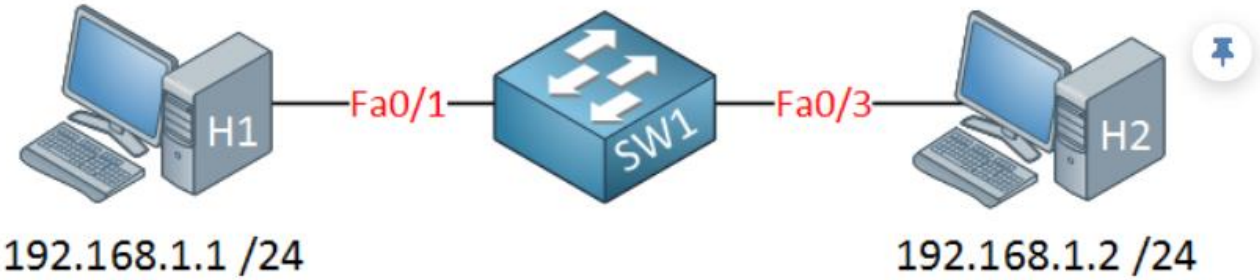
განვიხილოთ იგივე ტოპოლოგია და და შევამოწმოთ არის თუ არა კავშირი მოწყობილობებს შორი:

```
C:>Documents and Settings>H1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

კავშირი ამ შემთხვევაშიც არ გვაქვს .
შევამოწმოთ რომელიმე ინტერფეისი ხომ არ არის გამორტული რაიმე მიზეზის გამო:

```
SW1#show ip int brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/1        unassigned      YES unset  up                    up
FastEthernet0/3        unassigned      YES unset  up                    up
```

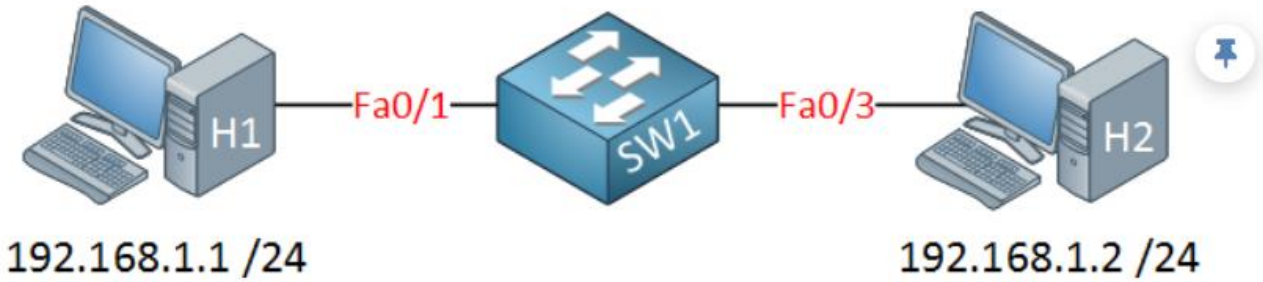
show ip int brief ბრძანება გვიჩვენებს ინტერფეისის მდგომარეობას ამ ბრძნაებიდან გამომდინარე ჩვენ ვხედავთ რომ ორივე ინტერფეისი არის up მდგომარეობაში



შემდეგ ეტაპზე შევამოწმოთ არიან თუარა აღნიშნული მოწყობილობები ერთი და იმავე ვილანში ამ ყველაფერის ნახვა ჩვენ შეგვიძლია **show vlan** ბრძანებით
show vlan ბძანებას გამოაქვს ინფორმაცია ყველა აღნიშნული Vlan-ზე რაც არის სვიჩზე და ასევე გამოაქვს ინფორმაცია რომელ Vlan ში რომელი პორტია გაწევრიანებული

```
SW1#show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/4, Fa0/5
                                                Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/11, Fa0/12,Fa0/13
                                                Fa0/14, Fa0/15, Fa0/16,Fa0/17
                                                Fa0/18, Fa0/19, Fa0/20,Fa0/21
                                                Fa0/22, Fa0/23, Fa0/24, Gi0/1
                                                Gi0/2
2    VLAN0002                         active    Fa0/3
```

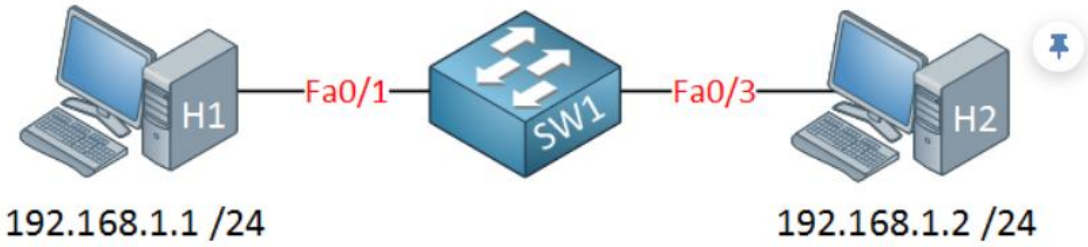
ამ ბრძანების შედეგად ვხედავთ რომ Fa0/1 იმყოფება VLAN 1 ში რომეიც არის დეფაულტ Vlan ხოლო Fa0/3 არის Vlan 2-ში
რადგან სურათზე მოცემული სვიჩი არის L2 დონის მას არ შეუძლია Vlan ებს შორის ინფორმაციის გაცვლა. იმისთვის რომ Vlanebs შორის ინფორმაცია გაიცვალოს გვესაჭიროება L3 დონის მოწყობილობა მაგალითად როუტერი. აღნიშნულ პრობლემას რაც ჩვენ ქსელში ფიქსირდდება მოაგვარებს ის რომ ორივე პორტი უნდა განვათავსორთ ერთ ვლანში



ამ შემთხვევაში ავიღოთ Fa0/3 და გადავსვათ Vlan1 ში და გავტესტოთ კავშირი მოწყობილობებს შორის

```
SW1(config)#interface fa0/3
SW1(config-if)#switchport access vlan 1
```

```
C:Documents and SettingsH1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```



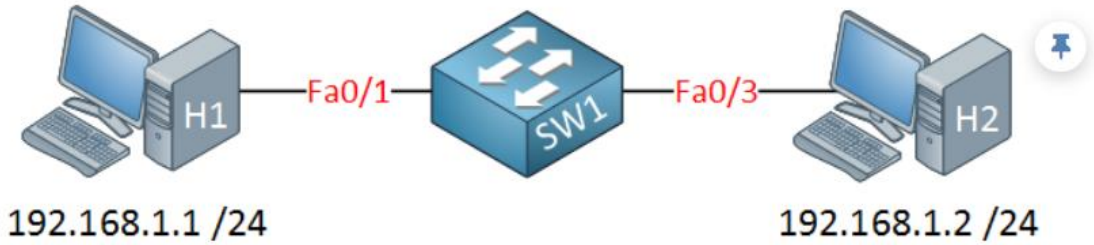
განვიხილოთ მსგავსი ტოლოლოგია და ისევ შევამოწმოთ კავშირი ჰოსტებს შორის:

```
C:>Documents and Settings>H1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

როგორც ვხედავთ კავშირი ჰოსტებს შორის არ არის, შევამოწმოტ ორივე პორტი არის თუ არა ერთიდაიმავე vlan -ში Show vlan ბრძანებით

```
SW1#show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/2, Fa0/4, Fa0/5, Fa0/6
                                                Fa0/7, Fa0/8, Fa0/9, Fa0/10
                                                Fa0/11, Fa0/12, Fa0/13,Fa0/14
                                                Fa0/15, Fa0/16, Fa0/17,Fa0/18
                                                Fa0/19, Fa0/20, Fa0/21,Fa0/22
                                                Fa0/23, Fa0/24, Gi0/1, Gi0/2
10   VLAN0010                         active    Fa0/1
```

ჩვენ ვხედავთ რომ fa0/1 პორტი ზის vlan 10 ში თუმცა fa0/3 არ არის არცერთ ვილანში განთავსებული



ეს იმის მანიშნებელია რომ პორტს შეიძლება ჰქონდეს რაიმე პრობლემა ამიტომ მოდი შევამოწმოტ ინტერფეისების სტატუსი **show ip interface brief**

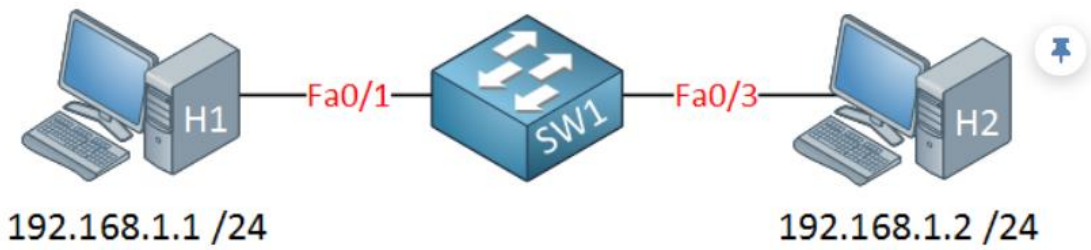
```
SW1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/1        unassigned      YES unset  up                    up
FastEthernet0/3        unassigned      YES unset  up                    up
```

როგორც ჩანს ორივე პორტი არის აქტიურ მდგომარეობაში მოდით გამოვიტანოთ უფრო დეტალური ინფორმაცია f10/3 პორტზე

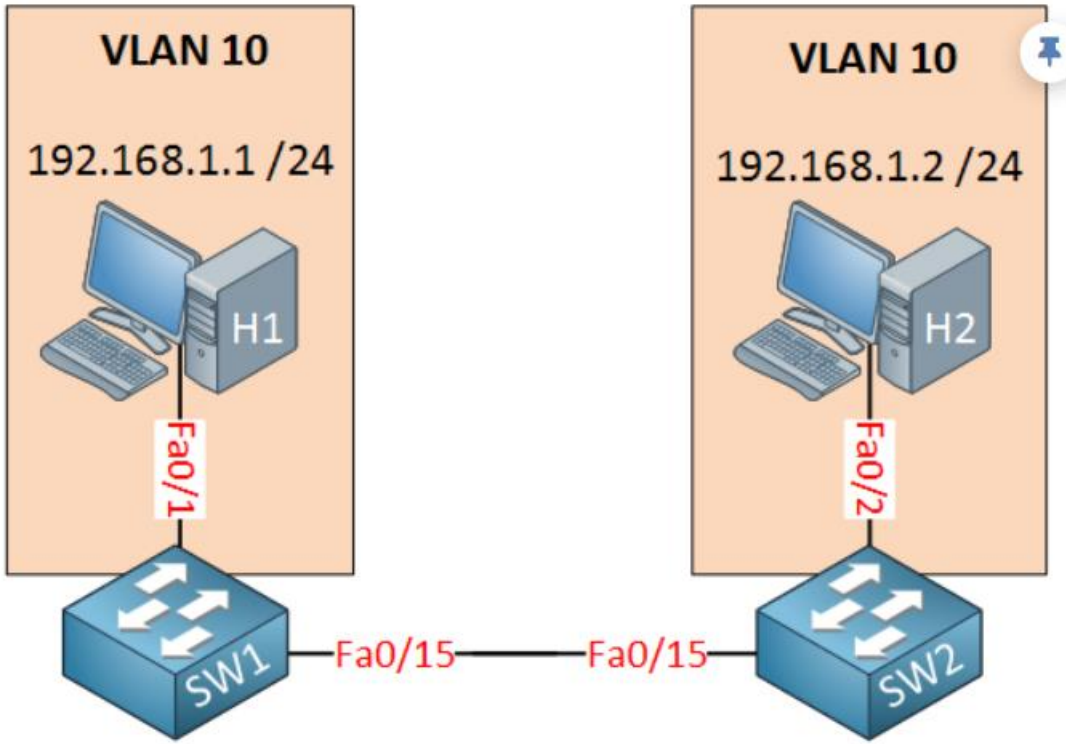
იმისთვის რომ პორტზე მივიღოთ უფრო დეტალური ინფორმაცია ჩვენ შეგვიძლია გამოვიყენოთ შემდეგი ბრძანება **show interfaces fa0/3 switchport**

```
SW1#show interfaces fa0/3 switchport
Name: Fa0/3
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
```

გამოსული ინფორმაციიდან ჩანს რომ fa0/3 არის TRUNK მდგომარეობაში და შესაბამისად ვირებთ შემდეგ მოცემულობას.
Fa0/1 არის VLAN 10 ის წევრი ხოლო Fa0/3 aატის ტრანკ პორტი ამიტომ ვერ ხერხდება მათ შორის კომუნიკაცია. თუ ჩვენ ჩავსვავთ Fa0/3 ს VALn10 Si მაშინ პრობლემა მოგვარდება და მოწყობილობები შეძლებენ ერთმანეთთან კომუნიკაციას



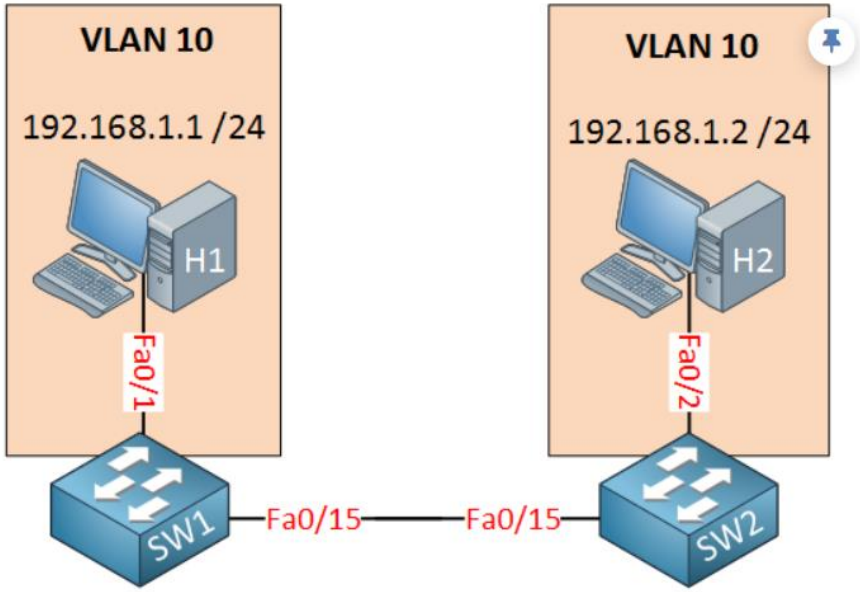

```
SW1(config)#interface fa0/3
SW1(config-if)#switchport mode access
SW1(config-if)#switchport access vlan 10
```



ახლა განვიხილოთ შემდეგი ტოპოლოგია ნახაზზე მოცემულია ორი სვიჩი ორივეზე არის მიერთებული მოწყობილობები და ეს მოწყობილობები არიან განთავსებული VLAN 10 ში სვიჩებს შორის პორტი არის Trunk პორტი და შესაბამისად ამ მოწყობილობესბ უნდა შეეძლოთ ერთმანეთთან კომუნიკაცია.
გავატაროთ პინგი ერთი მოწყობილობიდან მეორეზე

```
C:>Documents and Settings>H1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

როგორც ჩანს მოწყობილობებს შორის პინგი არ აგდის ეს იმას ნიშნავს რომ მათ არ შეუძლიათ ერთმანეთში ინფორმაციის გაცვლა

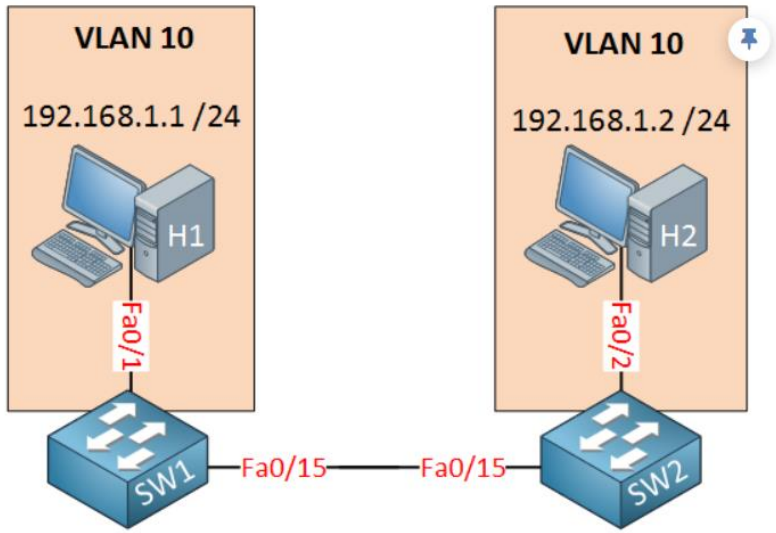


შევამოწმოთ სვიჩებს შორის პორტი Fa0/15 არის თუ არა ტრუნკ კავშირი ამისთვის გამოვიყენოთ **show interfaces fa0/15 switchport** ბრძანება.

```
SW1#show interfaces fa0/15 switchport
Name: Fa0/15
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
```

```
SW2#show interfaces fa0/15 switchport
Name: Fa0/15
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
```

გამოტანილი ინფორმაციიდან ჩანს რომ ინტერფეისი Fa0/15 არის ტრუნკ პორტი ორივე სვიჩზე

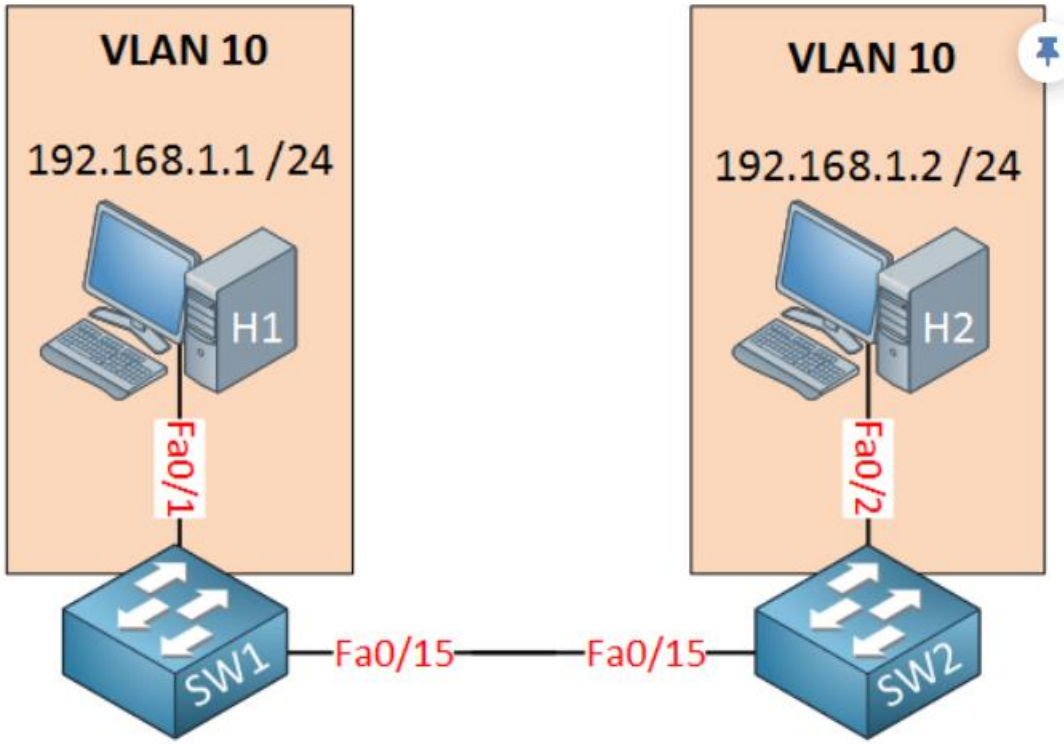


შემდეგი ეტაპი არის ის რომ შევამოწმოთ ორივე კომპიუტერი ნამდვილათ ზის თუ არა ერთიდა იმავე VLAN ში Show vlan ბრძანებთ.
თუ ორივე fa0/1 და fa0/2 არის vlan 10 სი ესეგი ამ მხრივაც პრობელემა არ არის.
გამოვიტანოთ უფრო დეტალური ინფორმაცია ტრანკ პორტის შესახებ შემდეგი ბრძანებით
show interfaces fa0/15 trunk

```
SW1#show interfaces fa0/15 trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/15      on               802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/15      20
```


```
SW1#show interfaces fa0/15 trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/15      on               802.1q         trunking      1

Port        Vlans allowed on trunk
Fa0/15      20
```


პრობლემაც სახეზეა როგორც გამოტანილი ინფორმაციიდან ჩანს ტრანკ პორტი ატარებს მხოლოდ vlan 20 ის ტრაფიკს რაც არ აძლევს vlan 10-ს იმის საშუალებას გაიაროს ტრუნკ ინტერფეისში. ამ პრობლემის გასასწორებლად უნდა დავუშვათ vlan 10 ის ტრაკნ პორტზე



ამისთვის უნდა შევიდეთ ტრანკ ინტერფეისზე და ვუთხრათ შემდეგი ბრძანება

switchport trunk allowed vlan all

ამ ბრძანებიტ ჩვენ ვეუბნებიტ ტრანკ პორტს რომ გაატაროს ყველანაირი ვილანი

შევამოწმოთ კავშირი ჰოსტებს შორის.

```
SW1(config)#interface fa0/15
SW1(config-if)#switchport trunk allowed vlan all
```

```
SW2(config)#interface fa0/15
SW2(config-if)#switchport trunk allowed vlan all
```

```
C:Documents and SettingsH1>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
```

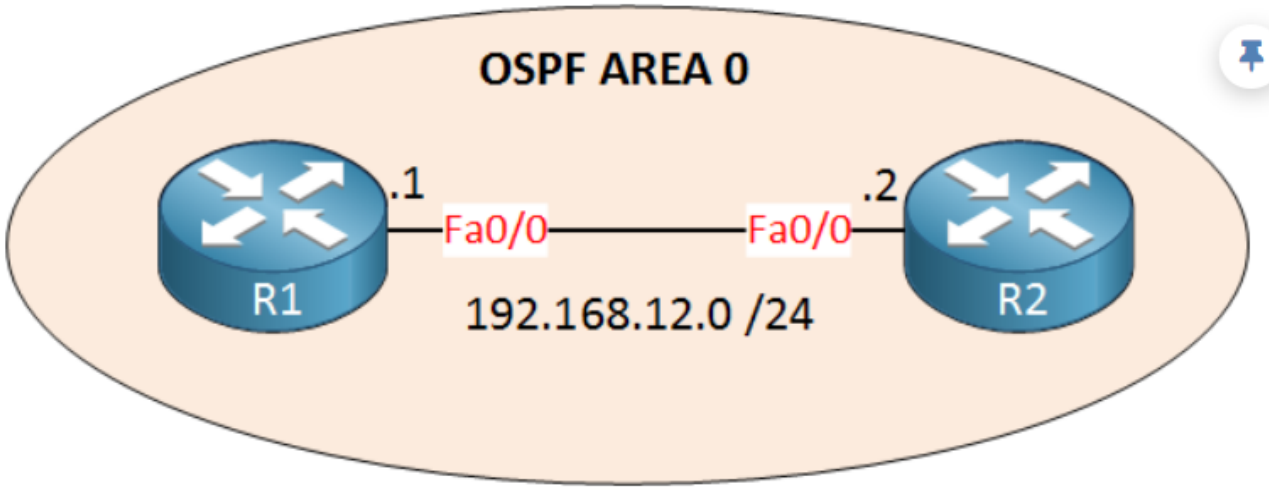
OSPF პროტოკოლი

OSPF ი როგორც ვიცით არის დინამიური მარშუტიზაციის პროტოკოლი რაც ნიშნავს იმას რომ ამ პროტოკოლის გამოყენებიტ როუტერები ერთმანეც უცვლიან მარშუტიზაციისთვის საჭირო ინფორმაციას.

თუმცა იმისთვის რომ როუტერებმა გაცვალონ ერთმანეთში ინფორმაცია საჭიროა მათ შორის დამყარდეს მეზობლობა.

ორ ospf როუტერს შორის მეზობლომა მყარდება იმ შემთხვევაში თუ

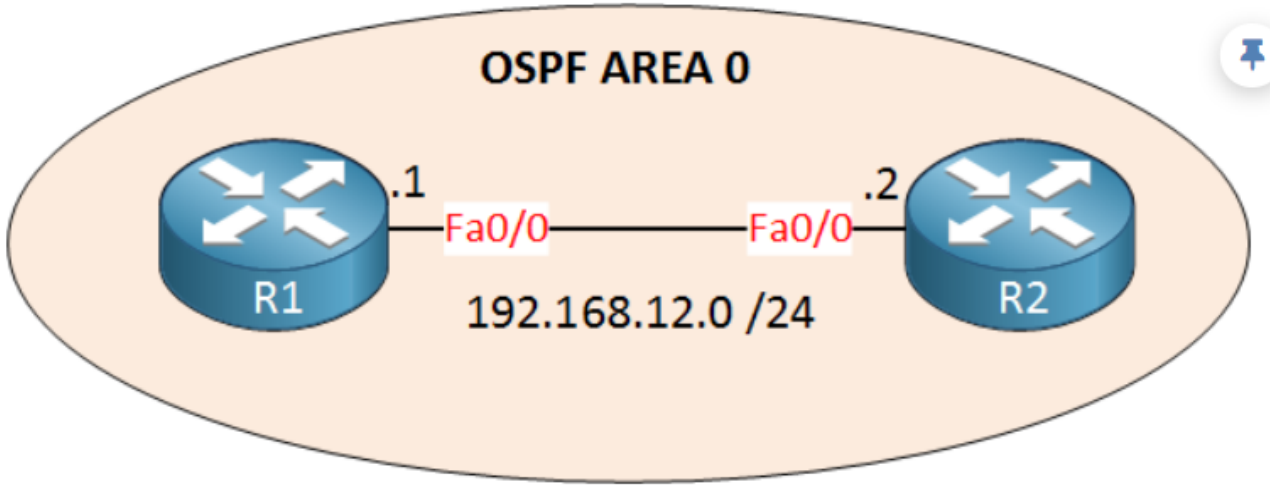
- ორივე როუტერის დამაკავშირებელი ინტერფეისი არის ერთი და იმავე ქსელში;
- ორივე როუტერი აროს გაერთიანებული ერთიდა იმავე არეაში;
- და მათ კომუნიკაციას არ ბლოკავს აქსეს ლისტი



განვიხილოტ შემდეგი მაგალითი ნახაზზე მოცემულია ორი როუტერი და მათ შორის არის აწყობილი OSPF პროტოკოლი შევამოწმოთ დგას თუ არა მათ შორის მეზობლობა შემდეგი ბრძანებით **show ip ospf neighbor**

```
R1#show ip ospf neighbor
```

ამ ბრძანების შედეგად როუტერს არ გამოუტანია არანაირი ინფორმაცია შესაბამისად ეს ნიშნავს იმას რომ როუტერებს შორის არ დგას OSPF მეზობლობა.



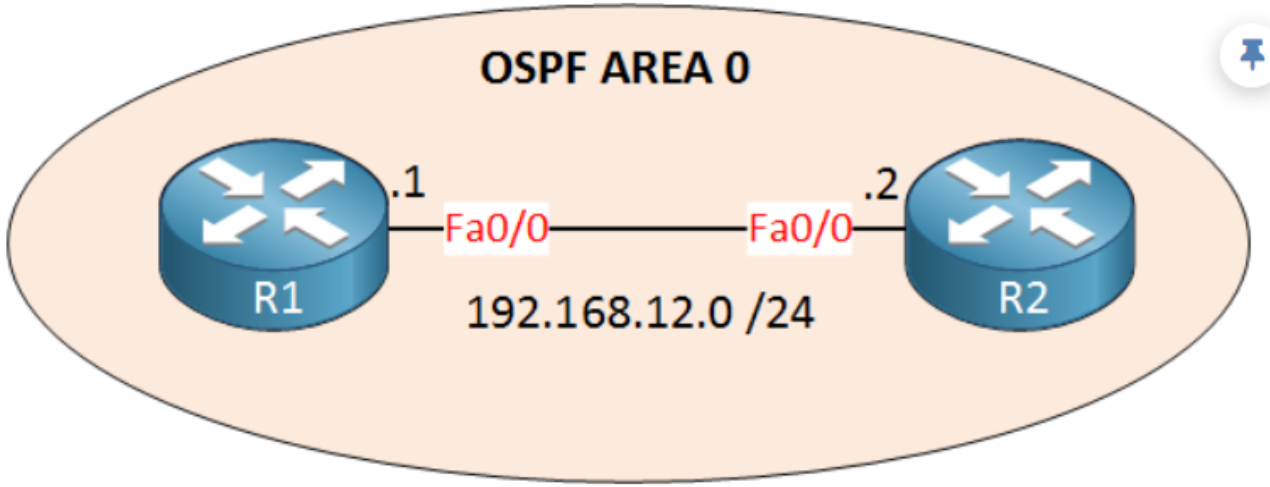
შევამოწმოთ ამ ინტერფეისებზე ორივე როუტერზე არის თუ არა ჩართული OSPF პროტოკოლი შემდეგი ბრძანებით

show ip ospf interface fastEthernet 0/0

```
R1#show ip ospf interface fastEthernet 0/0
%OSPF: OSPF not enabled on FastEthernet0/0
```

```
R2#show ip ospf interface fastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Internet Address 192.168.12.2/24, Area 0
  Process ID 1, Router ID 192.168.12.2, Network Type BROADCAST, Cost: 1
```

ამ ბრძანების შედეგად ჩანს რომ პირველ როუტერზე fa 0/0 -ზე არ არის გაშვებული OSPF პროცესი შესაბამისათ როუტერებს შორის ვერ დადგება მეზობლობა.

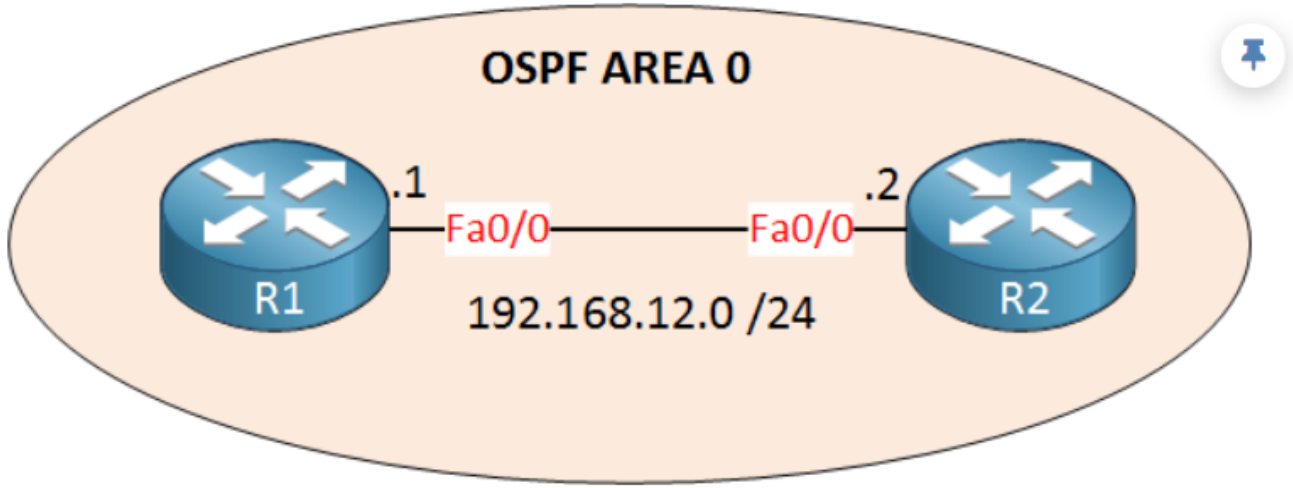


მოდით ჩაუღრმავდეთ ამ დეტალს და გამოვიტანოთ R1 ზე გაწერილი OSPF ის კონფიგურაციია შემდეგი ბრძანებით

show run | section ospf

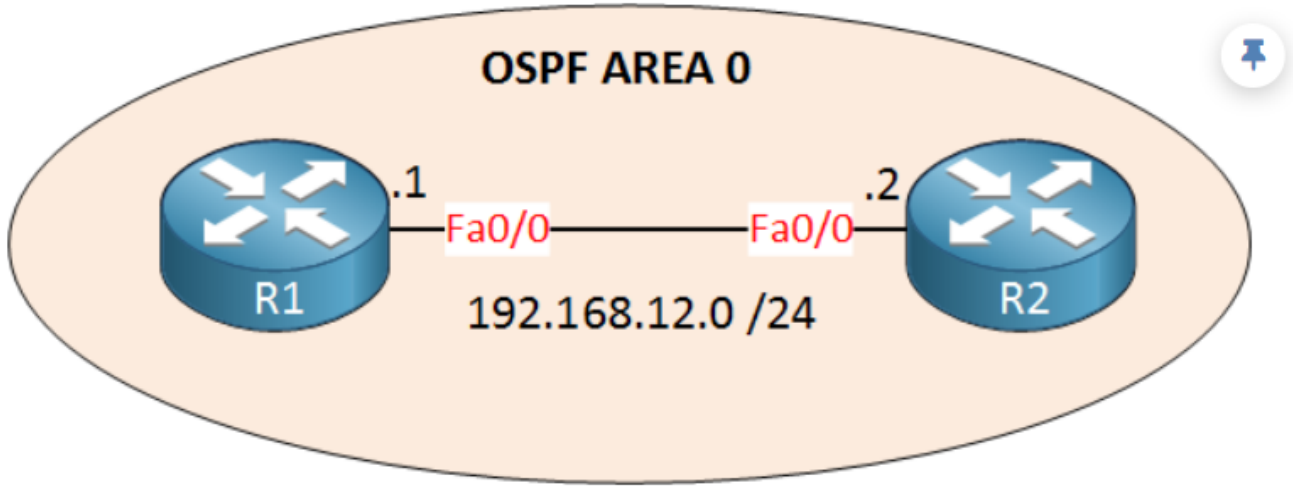
```
R1#show run | section ospf
router ospf 1
 log-adjacency-changes
 network 192.168.21.0 0.0.0.255 area 0
```

გამოტანილი ინფორმაციდან ჩვენ ვხედავთ რომ OPSF პროტოკოლში 192.168.12.0 ქსელის მაგივრად მითთებულია 192.168.21.0 ქსელი რაც იწვევს იმას რომ როუტერებს შორის არ დგება მეზობლობა.



იმისთვის რომ გამოვასწოროთ ეს პრობლემა უნდა მოვიქცეთ შემდეგნაირად უნდა შევიდეთ R1 ის OSPF ის კონფიგურაციაში და 192.168.21.0 ჩავანაცვლოთ 192.168.12.0 ქსელთ

```
R1(config)#router ospf 1
R1(config-router)#no network 192.168.21.0 0.0.0.255 area 0
R1(config-router)#network 192.168.12.0 0.0.0.255 area 0
```



შევამოწმოთ დადგა თუ არა მეზობლობა როუტერებს შორის.
show ip ospf neighbor

```
R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
192.168.12.2      1   FULL/DR         00:00:31    192.168.12.2    FastEthernet0/0
```

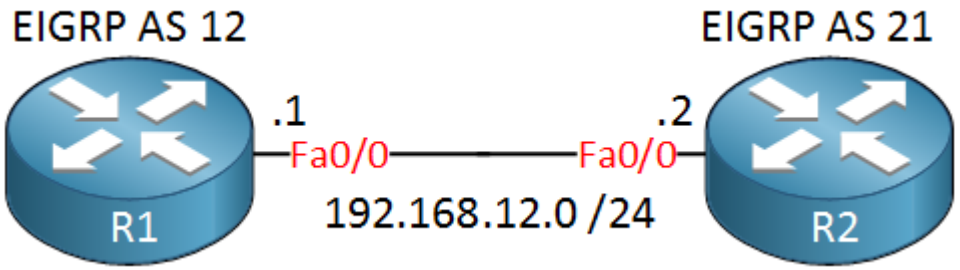
როგორც ხედავთ მეზობლობა როუტერებს შორის დადგა.
განვიხილოთ ეს ჩანაწერი უფრო დეტალურად.

```
R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
192.168.12.2      1   FULL/DR         00:00:31    192.168.12.2    FastEthernet0/0
```

- **Neighbor ID** მითითებულია მეზობელი როუტერის ID [იდენთიფიკატორი]
- **State** მითიტებულია მეზობლობის მდგომარეობა ამ შემთხვევაში არის FULL რაც ნიშნავს იმას რომ როუტერებს შორის სრულად არის მეზობლობა დამდგარი და არ არსებობს არანაირი პრობლემა
- **Dead Time** რა დრო არის გასული იმიშ შემდეგ რაც როუტერებს შორის მეზობლობა დადგა
- **Address** მეზობელი როუტერის მისამართი
- **Interface** რომელ ინტერფეისზე არის მიერტებული ეს მეზობელი

EIGRP პროტოკოლი

EIGRP იც OSPF ის მზგავსად არის დინამიური მარშუტიზაციის პროტოკოლი იმ განსხვავებით რომ ეს პროტოკოლი არის ცისკოს მიერ დაპროექტებული და შექმნიპი პროტოკოლი.
თუმცა აქაც OSPF ის მზგავსად საჭიროა როუტერებს შორის დადგეს მეზობლობა.
განვიხილოთ შემთხვევა რატომაც შესაძლებელია ეს პროცესი არ შესრულდეს.

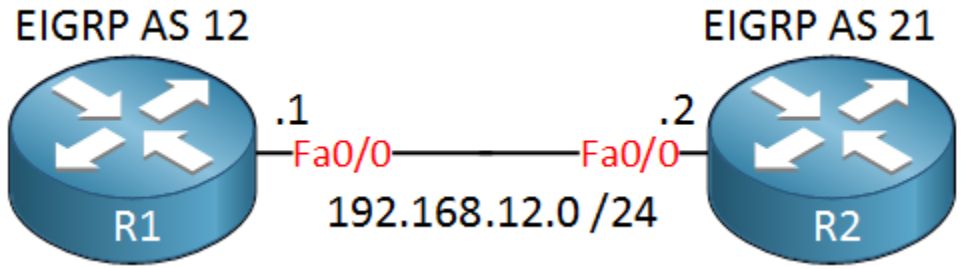



```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 12
```

```
R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 21
```

ნახაზზე მოცემულია როი როუტერი რომლებზეც აწყობილია EIGRP პროტოკოლი შევამოწმოთ მათ შორის დგას თუ არა მეზობლობა. **show ip eigrp neighbors** ბრძანებთ.

აქ ჩვენ ვხედავთ რომ R1 ზე გაშვებულია EIGRP პროცესის ნომრით 12 ხოლო R2 ზე EIGRP პროცესის ნომრით 21 Sesabamisad maT soris mezobloba ver damyardeb



იმისთვის რომ ეს პრობლემა მოგვარდეს საჭიროა შევიდეთ ერთერთ როუტერზე და შევცვალოთ პორცესის ნომერი.

```
R2(config)#no router eigrp 21


router eigrp 12
 network 192.168.12.0
```

ამისთვის შვდივართ როუტერზე მაგალითად R2 ზე
და ვაუქმებთ ძველ EIGRP პროცესს შემდეგი ბრძანებით
no router eigrp 21
ამის შემდეგ კი ვქმნით ახალპროცესის ნომერს **router eigrp 12**
და ვაკონფიგურირებთ EIGRP ი პროტოკოლს

როგორც კი პროცესი დამთავრდება ჩვენ დავინახავთ შემდეგ მესიჯს როუტერებზე რაც ნიშნავს იმას რომ მათ შორის როუტინგ მეზობლობა დადგა

```
R1# %DUAL-5-NBRCHANGE: IP-EIGRP(0) 12: Neighbor 192.168.12.2 (FastEthernet0/0) is up: new adjacency
```

```
R2# %DUAL-5-NBRCHANGE: IP-EIGRP(0) 12: Neighbor 192.168.12.1 (FastEthernet0/0) is up: new adjacency
```