سودار

23.10

سودار

Dec 17, 2024

فهرست مطالب

1 امکانات پایه 1
1.1 interface . . . 1
2.1 DHCP . . . 6
3.1 bundle interface . . . 7
4.1 motd . . . 8
5.1 show hardware . . . 8
6.1 ping . . . 9
7.1 traceroute . . . 9
8.1 تنظیمات سیستمی . . . 9

2 پروتکل های مسیریابی 15
1.2 اولویت route ها . . . 15
2.2 BGP . . . 15
3.2 IS-IS . . . 35
4.2 OSPF . . . 37
5.2 Encrypted OSPF . . . 41
6.2 OSPFv3 . . . 43
7.2 RIP . . . 46
8.2 RIPng . . . 48
9.2 Static Route . . . 50
10.2 pim . . . 54
11.2 PBR . . . 55

3 **NAT** 59
1.3 معرفی NAT . . . 59
2.3 Dynamic NAT . . . 59
3.3 بررسی ارتباط NAT . . . 61
4.3 حذف NAT . . . 61
5.3 فعال کردن log های NAT . . . 61
6.3 STATIC NAT . . . 61
7.3 Protocol NAT . . . 62

4 **SLA** 65
1.4 SLA . . . 65

5 **Qos** 73
1.5 مفهوم QOS . . . 73
2.5 آموزش راه اندازی QOS در سودار . . . 73
3.5 ایجاد class-map . . . 74
4.5 ایجاد policy . . . 74
5.5 اعمال policy در اینترفیس . . . 74
6.5 بررسی عملکرد QOS . . . 75
7.5 مشاهده policy ها در روتر . . . 75

8.5 فعال کردن log های qos . . . 75

6 **ACL** 77
1.6 ACL . . . 77
2.6 آشنایی با مفهوم ACL . . . 77
3.6 پیکربندی ACL . . . 78
4.6 اعمال ACL در اینترفیس . . . 78
5.6 بررسی عملکرد ACL . . . 79
6.6 حذف ACL . . . 80
7.6 مثال محدودیت ssh . . . 80
8.6 IPv6 ACl . . . 81
9.6 ACL با نام سرویس . . . 81
10.6 ACl Stateful . . . 82
11.6 فعال کردن log های ACL . . . 83
12.6 resequence کردن . . . 83

7 **VRF** 85
1.7 آشنایی با مفهوم VRF . . . 85
2.7 اضافه کردن VRF جدید . . . 86
3.7 قرار دادن اینترفیس در VRF . . . 86
4.7 vrf trunking . . . 88
5.7 بررسی عملکرد VRF . . . 89
6.7 حذف VRF . . . 89
7.7 فعال کردن log های VRF . . . 90

8 **MPLS** 91
1.8 MPLS . . . 91
2.8 l3vpn-MPLS . . . 95

9 تونل ها 123
1.9 gre . . . 123
2.9 IPSec . . . 126
3.9 wireguard normal . . . 141
4.9 wireguard-routing . . . 152
5.9 vxlan . . . 158
6.9 PPPoE Client . . . 161
7.9 vpls . . . 165

10 ویژگی های **L2** 171
1.10 جدول ARP . . . 171
2.10 VLAN . . . 172
3.10 Bridge . . . 177
4.10 SPAN . . . 179

11 **Scriptable CLI** 181
1.11 Sooshell . . . 181

12 تنظیمات کاربری 207
1.12 تغییر پسورد کاربر admin . . . 207
2.12 تغییر پسورد enable . . . 207
3.12 تغییر پسورد enable config . . . 208
4.12 ذخیره پسورد به صورت رمز شده . . . 208
5.12 تعیین حداقل طول پسورد . . . 208
6.12 تغییر ip پورت SSH . . . 208
7.12 بازیابی رمز کاربر (رمز ssh) . . . 209

13 پشتیبان گیری و بازنشانی تنظیمات 211
1.13 پشتیبان گیری از startup-config 212
2.13 پشتیبان گیری از running-config 213
3.13 مشاهده لیست backup ها 213
4.13 باز نشانی startup-config 214
5.13 مشاهده config هایی که بکاپ گرفته اید 215
6.13 write erase 215
7.13 تنظیم ip پیش فرض در write erase 216

14 بروزرسانی 217
1.14 بروز رسانی نرم افزار روتر 217
2.14 بروز رسانی آنلاین 217
3.14 بروز رسانی آفلاین 219

15 مانیتورینگ 221
1.15 Monitoring 221
2.15 IPFIX 224

16 تنظیمات لاگ 227
1.16 log 227
2.16 Debug 229

17 اتصال از **console** 231
1.17 console 231

18 تنظیمات امنیتی 233
1.18 بلاک کردن ip 233
2.18 urpf 233
3.18 session timeout 234
4.18 show login failure 234
5.18 show user 235
6.18 clear line 235

19 **Tune** 237
1.19 control-palne 237
2.19 data-plane 237

20 **capture** کردن بسته ها 239
1.20 capture کردن بسته ها 239

21 برخی دستورات خاص 243
1.21 find 243
2.21 فیلتر کردن خروجی 244
3.21 نمایش command history 244
4.21 حذف command history 244
5.21 show diagnostic 245

22 ارتباط با **Cisco** 247
1.22 همسایگی RIP بین سودار و cisco 248
2.22 همسایگی OSPF بین سودار و cisco 249
3.22 همسایگی ISIS بین سودار و cisco 252
4.22 همسایگی BGP بین سودار و cisco 255
5.22 تست MPLS با روتر Cisco 258
6.22 تونل GRE بین سودار و cisco 265
7.22 تونل IPSec بین سودار و cisco 267

23 ارتباط با **MikroTik** 271
1.23 همسایگی RIP بین سودار و Mikrotik . . . 271
2.23 همسایگی OSPF بین سودار و Mikrotik . . . 273
3.23 همسایگی BGP بین سودار و Mikrotik . . . 275
4.23 تونل GRE بین سودار و MikroTik . . . 277
5.23 تونل IPSec بین سودار و MikroTik . . . 279
6.23 تونل wireguard بین سودار و میکروتیک . . . 290

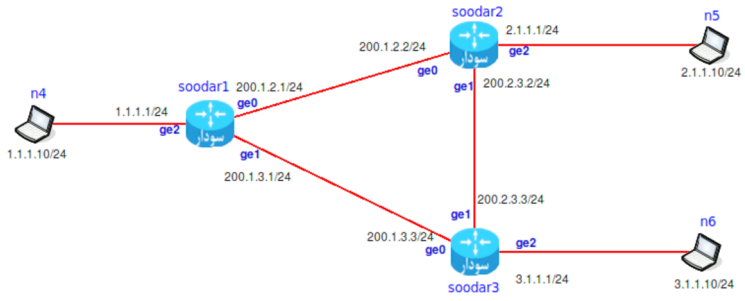
فصل 1

امکانات پایه

interface 1.1

1.1.1 تنظیم ip در اینترفیس

سنا ریوی زیر را در نظر بگیرید . 3 روتر سودار در این سناریو وجود دارد که هر کدام دارای 3 اینترفیس می باشد که با 2 اینترفیس به روتر های دیگر لینک دارد و یک اینترفیس نیز به شبکه محلی متصل می باشد .



تنظیم ip تمامی اینترفیس های این سناریو را در ادامه انجام می دهیم :

soodar1 : ابتدا لازم است که به حالت تنظیم (configure) وارد شویم :

```
soodar1# conf t
soodar1(config) #
```

اینترفیسی که می خواهیم در آن ip تنظیم کنیم انتخاب می کنیم :

```
soodar1(config-if) # interface ge0
```

سپس آدرس دلخواه را در آن تنظیم می کنیم :

```
soodar1(config-if) # ip address  200.1.2.1/24
```

این کار را برای دیگر اینترفیس ها هم تکرار می کنیم :

soodar1

```
soodar1(config) # interface ge1
soodar1(config-if) # ip address   200.1.3.1/24
soodar1(config-if) # q
soodar1(config) # interface ge2
soodar1(config-if) # ip address   1.1.1.1/24
soodar1(config-if) # end
soodar1# write
```

soodar2

```
soodar2# conf t
soodar2(config) #
```

```
soodar2(config-if) # interface ge0
soodar2(config-if) # ip address   200.1.2.2/24
soodar2(config-if) # q
```

```
soodar2(config) # interface ge1
soodar2(config-if) # ip address   200.2.3.2/24
soodar2(config-if) # q
soodar2(config) # interface ge2
soodar2(config-if) # ip address   2.1.1.1/24
soodar2(config-if) # end
soodar2# write
```

soodar3

```
soodar3# conf t
soodar3(config) #
```

```
soodar3(config-if) # interface ge0
soodar3(config-if) # ip address   200.1.3.3/24
```

```
soodar3(config) # interface ge1
soodar3(config-if) # ip address   200.2.3.3/24
soodar3(config-if) # q
soodar3(config) # interface ge2
soodar3(config-if) # ip address   3.1.1.1/24
soodar3(config-if) # end
soodar3# write
```

خاطر نشان

با استفاده از ping بین روتر ها می توانید از صحت ip های تنظیم شده در روتر ها اطمینان حاصل نمایید .

```
soodar3# ping 200.2.3.2
soodar1# ping 1.1.1.10
```

2.1.1 مشاهده اینترفیس ها

برای مشاهده لیست اینترفیس ها به صورت خلاصه از دستور زیر استفاده می شود :

```
soodar1# sh int brief
Interface       Status  VRF            Addresses
---------       ------  ---            ---------
ge0             up      default        200.1.2.1/24
ge1             up      default        200.1.3.1/24
ge2             up      default        1.1.1.1/24

lo              up      default
```

ستون Interface : نام اینترفیس در آن مشخص است
ستون Status : وضعیت up/down بودن اینترفیس را نشان می دهد . اگر اینترفیس توسط ادمین shutdown نشده باشد و کابل هم به اینترفیس وصل باشد اینترفیس up خواهد بود در غیر اینصورت اینترفیس down است .
ستون VRF : نشان می دهد که اینترفیس در کدام vrf قرار دارد
ستون Addresses : آدرس های تنظیم شده در اینترفیس را نشان می دهد

مشاهده جزئیات تنظیمات اینترفیس :

```
soodar1# sh int ge0
Interface ge0 is up, line protocol is up
  Link ups:       1    last: 2023/06/11 10:29:24.13
  Link downs:     1    last: 2023/06/11 10:29:18.32
  vrf: default
  index 6 metric 0 mtu 1500 speed 1000
  flags: <UP,BROADCAST,RUNNING,MULTICAST>
  Type: Ethernet
  HWaddr: 00:ec:ac:cd:e0:9c
  inet 200.1.2.1/24
  Interface Type Other
  Interface Slave Type None
  protodown: off
  Joined group address(es):
    239.255.255.250
    224.0.0.252
    224.0.0.251
    224.0.0.22
    224.0.0.1
    ff02::1
    ff01::1
soodar1#
```

line protocol is up : یعنی کابل به اینترفیس وصل است

Link ups: 1 last: 2023/06/11 10:29:24.13 : تعداد دفعاتی که اینترفیس up شده است و زمان آخرین باری که up شده است

Link downs: 1 last: 2023/06/11 10:29:18.32 تعداد دفعاتی که اینترفیس down شده است و زمان آخرین باری که down شده است

index 6 metric 0 mtu 1500 speed 1000 : . سرعتی که این اینترفیس در آن کار می کند. ممکن است اتصال به سوئیچ سرعت کاری اینترفیس را پایین بیاورد مثلا اینترفیس 1000 باشد اما در سوئیچ 100 که باشد در سرعت 100 کار می کند همچنین مشخص است که mtu برابر 500 بایت است

HWaddr: 00:ec:ac:cd:e0:9c : آدرس MAC

Joined group address(es) : گروه های multicast که اینترفیس در آن قرار دارد

3.1.1 تنظیم **ipv6** در اینترفیس

همانند ipv4 می توان ipv6 نیز در اینترفیس تنظیم کرد :

```
soodar3(config-if) # ipv6 address  2001:1::10/64
```

4.1.1 تنظیم **DHCP** در اینترفیس

برای آشنایی با نحوه تنظیم به بخش DHCP مراجعه نمایید .

5.1.1 تنظیم **bundle-ether**

در واقع با کمک EtherChannel می توانیم پورتهای فیزیکی را گروه بندی منطقی نموده و آنها را در یک گروه قرار دهیم که این امر باعث بالارفتن تحمل خطا در اتصالات و همچنین دسترسی به پهنای باند بیشتر از طریق ترکیب شدن چند لینک ارتباطی می شود. با استفاده از پروتکل LACP یا Link Aggregation Control Protocol یک اینترفیس به نام bundle-ether می سازیم و به طریق زیر آن را تنظیم می کینم :

برای تنظیم اینترفیس Bundle به شکل زیر عمل می کنیم . ابتدا یک bundle-ether با id مورد نظر ایجاد می کنیم :

```
soodar1(config)# bundle-ether 30
soodar1(config-if)# no shutdown
```

در مرحله بعد اینترفیس هایی که قرار است به صورت bundle با هم استفاده شوند به شکل زیر تنظیم می کنیم :

```
soodar1(config)# int ge0
soodar1(config-if)# bundle id 30
```

اگر در اینترفیسی ip تنظیم شده باشد ابتدا باید ip تنظیم شده را از اینترفیس حذف کرده و سپس در آن bundle id مورد نظر را تنظیم کرد .

6.1.1 **Shutdown**

به صورت پیش فرض اینترفیس ها down هستند مگر اینکه در اینترفیس IP تنظیم گردد که به طور خودکار up می شود و یا اینکه به صورت دستی در تنظیمات اینترفیس دستور no shutdown را وارد کنیم . برای down , up کردن اینترفیس ها به شکل زیر عمل می کنیم :

```
soodar1(config)# int ge0
soodar1(config-if)# shutdown
```

```
soodar1(config)# int ge0
soodar1(config-if)# no shutdown
```

توجه

1. اگر در اینترفیسی ip تنظیم شود به طور خودکار up می شود و نیازی به no shutdown نیست .
2. اگر اینترفیسی را به صورت دستی shutdown کنیم اینترفیس غیرفعال می شود و برای استفاده از آن باید اینترفیس را با دستور no shutdown فعال کنیم.
3. اینترفیس های سودار spanning tree را پشتیبانی نمی کنند و دستگاهایی که به سودار به طور مستقیم وصل هستند نباید در اینترفیسی که با سودار در ارتباط هستند **stp** فعال داشته باشد .

7.1.1 نام اینترفیس ها در سودار

اینترفیس ها طبق جدول زیر در سودار نام گذاری می شوند:

نام اینترفیس در سودار	سرعت اینترفیس
ge	1G
te	10G
tw	20G
foe	40G
fie	50G
he	100G

توجه

به صورت پیش فرض ge0 (اولین اینترفیس) به عنوان پورت ssh و با IP برابر با 192.168.1.55/24 تنظیم شده است .

8.1.1 دستور **beacon**

نام اینترفیس ها در سودار با استفاده از pci address کارت شبکه لیست می شود و ممکن است ترتیب آنها در پورت های فیزیکی درست نباشد. با استفاده از دستور beacon ، چراغ اینترفیس روشن می شود و برای مثال می توانید ببینید اینترفیس ge3 کدام پورت سخت افزاری روی دستگاه است :

```
soodar1(config)# int ge0
soodar1(config-if)# beacon ?
  <cr>
  (1000-30000)  Blinking time in milliseconds

soodar1(config-if)# beacon
```

توجه

در صورت نیاز شما می توانید ترتیب اینترفیس را مطابق دلخواه تغییر دهید . این کار با استفاده از Tune انجام می شود

DHCP 2.1

تنظیم DHCP client 1.2.1

برای فعال کردن dhcp در اینترفیس کافی است دستور زیر ر در اینترفیس مربوطه وارد نمایید :

```
soodar(config)# int ge1
soodar(config-if)# ip address dhcp
```

تنظیم DHCP server 2.2.1

اگر می خواهید از روتر سودار به عنوان DHCP server استفاده کنید باید ابتدا یک DHCP pool تعریف کرده و سپس در اینترفیس مورد نظر DHCP server را فعال کنید .

- با دستور **network** مشخص می کنید که در چه رنج شبکه ای باید به کلاینت ها ip اختصاص یابد .
- با **included-address** مشخص می کنید چه رنج ip هایی از شبکه فوق باید به کلاینت ها اختصاص یابد. می توانید چند included-address اضافه کنید .
- **lease** مشخص می کند که تا چه زمانی ip اختصاص داده شده به یک کلاینت را نگه دارد و به شخص دیگری اختصاص ندهد .
- **default-router** آدرس gateway شبکه را مشخص می کند .
- **dns-server** آدرس dns server را تنظیم می کند .
- **ntp-server** آدرس ntp server را تنظیم می کند .

تعریف یک dhcp pool :

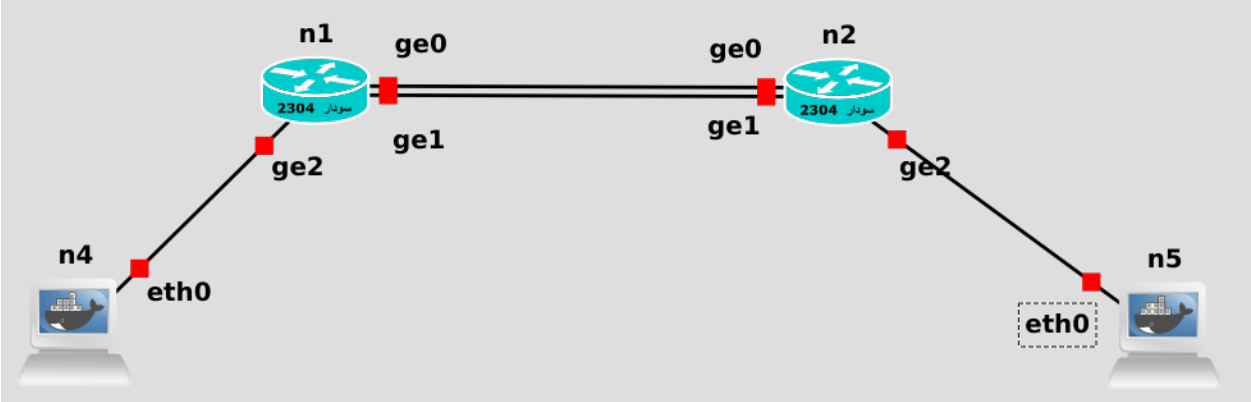
```
soodar(config)# ip dhcp pool POOL1
soodar(dhcp-config)# network 192.168.100.0/24
soodar(dhcp-config)# included-address 192.168.100.100 192.168.100.200
soodar(dhcp-config)# default-router 192.168.100.1
soodar(dhcp-config)# lease 1 0 0
soodar(dhcp-config)# ntp-server 192.168.100.250
soodar(dhcp-config)# dns-server 8.8.8.8
soodar(dhcp-config)# q
```

فعال کردن dhcp server در اینترفیس :

```
soodar(config)# int ge0
soodar(config-if)# ip dhcp server
```

bundle interface 3.1

اینترفیس bundle یک اینترفیس مجازی است که چند اینترفیس فیزیکی را در یک اینترفیس مجازی ترکیب می کند. این اینترفیس با نام های link-aggregation یا port-channeling نیز شانخته می شود . زمانی که چند اینترفیس فیزیکی در یک اینترفیس bundle ترکیب می شوند ، همه آنها به عنوان یک اینترفیس مجازی در شبکه ظاهر می شوند . تمامی تنظیمات اینترفیس ها مانند تنظیم ip و ... می تواند در این اینترفیس انجام شود



1.3.1 تنظیم اینترفیس bundle

ساخت اینترفیس

```
soodar(config)# interface bundle-ether 33
```

تنظیم الگوریتم ها

الگوریتم bonding و همچنین الگوریتم load balancing را می توانید به شکل زیر تنظیم کنید :

الگوریتم های bonding:

- rr :Round-robin mode که در آن بسته ها در بین اینترفیس ها به صورت متوالی توزیع می شوند
- xor :Exclusive-OR mode مانند rr است اما بسته ها به کمک load balance توزیع می شوند.
- active-backup: یکی از اینترفیس ها به عنوان اینترفیس فعال انتخاب می شود و بقیه درحالت آماده بکار قرار می گیرند ، اگر لینک فعال قطع شود یکی دیگر از لینک های آماده بکار فعال می شود
- broadcast: بسته ها به صورت همزمان در همه اینترفیس ها ارسال می شوند
- lacp: از پروتکل LACP استفاده می کند و به صورت دائم برای link aggrigation با دستگاه مقابل مذاکره می کند

الگوریتم های Load balancing :

- l2: جریان بسته ها را طبق مبدا و مقصد MAC آنها محاسبه می کند.
- l23: جریان بسته ها را طبق مبدا و مقصد MAC آنها و همچنین مبدا و مقصد IP آن ها محاسبه می کند.

- l34: جریان بسته ها را طبق مبدا و مقصد IP آنها و همچنین udp/tcp پورت مبدا و مقصد محاسبه می کند اگر بسته tcp یا udp باشد

```
soodar(config-if)# set mode lacp l23
```

اضافه کردن اینترفیس فیزیکی

به شکل زیر اینترفیس هایی که قرار هست با هم bondle شوند را مشخص می کنیم :

```
soodar(config)# interface ge0
soodar(config-if)# no shutdown
soodar(config-if)# bundle id 33
soodar(config-if)# q
soodar(config)# interface ge1
soodar(config-if)# no shutdown
soodar(config-if)# bundle id 33
```

motd 4.1

گاهی اوقات لازم است نکته یا مطلب مهمی را به کلیه ادمین ها در یک روتر اعلام کنید شما می توانید در روتر یک پیام را تنظیم کنید تا هر کس که خواست به روتر وصل شود ابتدا این پیام را ببیند :

```
soodar(config)# banner motd line this is a motd
```

با دستور زیر هم پیام فوق حذف می شود :

```
soodar(config)# no banner motd
```

show hardware 5.1

اطلاعات سخت افزاری و وضعیت جاری memory , cpu و disk را می توانید بررسی کنید :

```
soodar# show hardware cpu
soodar# show hardware memory
soodar# show hardware disk
```

ping 6.1

به شکل زیر می توان از ping استفاده کرد :

```
soodar1# ping 8.8.8.8
soodar1# ping vrf red 200.1.2.2
soodar1(config)# do ping vrf red 200.1.2.2
soodar1# ping 200.1.2.2 count 5 timeout 3
soodar1# ping vrf red 200.1.2.2 count 50 timeout 1
```

traceroute 7.1

به شکل زیر می توان از traceroute استفاده کرد :

```
soodar1# traceroute 8.8.8.8
soodar1# traceroute vrf red 8.8.8.8
```

8.1 تنظیمات سیستمی

1.8.1 تنظیم host name

```
soodar(config)# hostname SOODAR
SOODAR(config)#
```

2.8.1 تنظیم ip host

```
soodar(config)# ip host soodarA 192.168.1.51
```

3.8.1 مشاهده ip host

```
soodar1# sh ip host
iman 192.168.1.30
soodarA 192.168.1.55
n3 192.168.30.253
netem 192.168.30.248
```

4.8.1 تنظیم name-server

```
soodar(config)# ip name-server 8.8.8.8
```

5.8.1 مشاهده name-server

```
soodar(config)# do sh ip name-server
8.8.8.8
45.1.48.76
```

6.8.1 تنظیم ساعت و منطقه زمانی

تنظیم ساعت به دو روش استفاده از ntp و تنظیم دستی انجام می پذیرد . گزینه **NTP service: active** نشان می دهد که سرویس ntp فعال است و ساعت وتاریخ سیستم از طریق ntp تنظیم شده است :

```
soodar1# show clock
               Local time: Thu 2020-09-24 06:44:30 UTC
           Universal time: Thu 2020-09-24 06:44:30 UTC
                 RTC time: Thu 2020-09-24 06:43:47
                Time zone: UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

تنظیم Time Zone

```
n1(config)# clock timezone Asia/Tehran
n1(config)# do sh clock
               Local time: Thu 2020-09-24 10:15:19 +0330
           Universal time: Thu 2020-09-24 06:45:19 UTC
                 RTC time: Thu 2020-09-24 06:44:36
                Time zone: Asia/Tehran (+0330, +0330)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

تنظیم NTP server

```
n1(config)# ntp server ir.pool.ntp.org
n1(config)# do sh ntp sources

  .-- Source mode  '^' = server, '=' = peer, '#' = local clock.
 / .- Source state '*' = current best, '+' = combined, '-' = not combined,
| /             'x' = may be in error, '~' = too variable, '?' = unusable.
||                                               .- xxxx [ yyyy ] +/- zzzz
||      Reachability register (octal) -.         |  xxxx = adjusted offset,
||      Log2(Polling interval) --.      |        |  yyyy = measured offset,
||                                \     |        |  zzzz = estimated error.
||                                 |    |        \
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^* meetbsd.ir                    2   6    37    52    -16us[ +381us] +/-  246ms
```

تنظیم دستی ساعت و تاریخ

دقت کنید اگر سرویس ntp فعال باشد شما نمی توانید به صورت دستی ساعت را تنظیم کنید . ابتدا باید سرویس ntp را غیر فعال کنید :

```
soodar(config)# clock set 10:40:45 2 23 2020
Failed to set time: Automatic time synchronization is enabled
Failed to set time: Automatic time synchronization is enabled
```

غیر فعال کردن ntp

برای تنظیم دستی ساعت باید ابتدا آن را غیر فعال کنید

```
n1(config)# no ntp
```

```
n1(config)# clock set 12:12:12
  (1-12)  Month of the year
n1(config)# clock set 12:12:12 12
  (1-31)  Day of the month
n1(config)# clock set 12:12:12 12 12
  (2000-4192)  Year( no abbreviation)
n1(config)# clock set 12:12:12 12 12 2020
n1(config)# do sh clock
               Local time: Sat 2020-12-12 12:12:14 +0330
           Universal time: Sat 2020-12-12 08:42:14 UTC
                 RTC time: Sat 2020-12-12 08:42:15
                Time zone: Asia/Tehran (+0330, +0330)
System clock synchronized: no
              NTP service: inactive
          RTC in local TZ: no
```

توجه

- به صورت پیش فرض سرویس ntp در سودار فعال است .
- با غیر فعال کردن ntp تمامی ntp server های تنظیم شده حذف می شوند .
- به محض تنظیم هر ntp server سرویس ntp فعال می شود .

7.8.1 سرویس ها

در زیر لیستی از سرویس های فعال در روتر را آورده ایم . هر کدام از سرویس ها بخشی از عملکرد سیستمی سودار را انجام می دهند :

- plane) (datavpp
- (controlplane)frr
- service) ban (ipfail2ban
- services) (monitoring soolog , soomon
- dhcp
- ipsec
- ntp
- ssh
- snmp

مشاهده وضعیت سرویس

برای مشاهده وضعیت سرویس ها دستور زیر را وارد کنید :

```
soodar# sh system service status fail2ban
● fail2ban.service - Fail2Ban Service
     Loaded: loaded (/lib/systemd/system/fail2ban.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-06-01 12:25:43 +0430; 2min 23s ago
       Docs: man:fail2ban(1)
    Process: 380 ExecStartPost=/usr/bin/fail2ban-poststart (code=exited, status=0/SUCCESS)
   Main PID: 378 (f2b/server)
      Tasks: 5 (limit: 3585)
     Memory: 19.1M
     CGroup: /system.slice/fail2ban.service
             `-378 python3 /usr/bin/fail2ban-server -xf --logtarget=sysout start

Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.jail          [378]: INFO    Initiated 'systemd' backend
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.filter        [378]: INFO      maxLines: 1
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.filtersystemd [378]: INFO    [sshd] Added journal match for: '_SYSTEMD_UNIT=sshd.
↪service + _COMM=sshd'
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.filter        [378]: INFO      maxRetry: 3
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.filter        [378]: INFO      findtime: 30
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.actions       [378]: INFO      banTime: 600
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.filter        [378]: INFO      encoding: UTF-8
Jun 01 12:25:43 soodar fail2ban-server[378]: fail2ban.jail          [378]: INFO    Jail 'sshd' started
Jun 01 12:25:43 soodar fail2ban-server[378]: Server ready
Jun 01 12:25:43 soodar systemd[1]: Started Fail2Ban Service.
soodar#
```

Active: active (running) since Thu 2023-06-01 12:25:43 +0430; 2min 23s ago نشان می دهد که سرویس در حال اجرا است و حدود 2 دقیقه از زمان اجرای آن می گذرد و همچنین زمان دقیق استارت شدن سرویس مشخص است در خط های آخر لاگ های سرویس را مشاهده می کنید

ری استارت کردن سرویس

گاهی اوقات ممکن است در شرایط خاص لازم باشد سرویسی را به صورت دستی و از ترمینال ری استارت کنید . دستور مورد نظر به شکل زیر است :

```
soodar(config)# system service restart soolog
```

توجه

- ری استارت کردن سرویس در شرایطی خاص و معمولا به در خواست تیم پشتیبانی سودار باید انجام شود
- ری استارت برای همه سرویس ها امکان پذیر نیست و برای آنها تنها مشاهده وضعیت سرویس مقدور است

8.8.1 خاموش کردن

برای خاموش کردن روتر از دستور زیر استفاده می شود :

```
soodar1# poweroff
All your unsaved changes will be lost. Proceed with reload?[yes/no]
```

9.8.1 ری استارت کردن

برای ری استارت کردن روتر از دستور زیر استفاده می شود :

```
soodar1# reload
All your unsaved changes will be lost. Proceed with reload?[yes/no]
```

در هر دو دستور باید دقت کنید که تنظیماتی که write نکرده اید پس از reload , poweroff از بین می رود .

فصل 2

پروتکل های مسیریابی

1.2 اولویت **route** ها

Administrative distance مشخص می کند که چه route ی باید در روتر نصب شود و در جدول routing اعمال گردد . routeی که کمترین مقدار Distance را داشته باشد انتخاب می شود . در زیر جدول اولویت نصب را طبق پروتکل مشاهده می کنید :

Distance lrsphinxstyletheadfamily Protocol	
0	System
0	Connect
1	Static
1	Wireguard
20	EBGP
90	EIGRP
110	OSPF
115	ISIS
120	RIP
200	IBGP

BGP 2.2

سناریوی زیر را در نظر بگیرید . این سناریو شامل 6 AS است AS65001 , AS40 , AS30 , AS20 , AS10 , AS1 . قصد داریم این سناریو را با BGP تنظیم کنیم . فرض کنید قصد داریم نقش ها و سیاست های زیر را در روتر ها پیاده سازی کنیم :

AS10 :

در این AS روتر های n4 , n3 به عنوان Route Relfector استفاده می شوند و با IBGP با هم در ارتباط هستند و برای دیگر روتر ها به عنوان Route Server استفاده می شوند . سیاستهای این AS :

1. ترافیکی که از AS30 به مقصد AS10 دریافت می شود باید از n2 انتقال یابد . بدین منظور روتر های n2 , n1 باید پارامتر MED را تغییر دهند .
2. remote peer ها می توانند Local Preference را تغییر دهند .

AS20 :

در n13 پروتکل BGP اجرا نیست . سیاستهای این AS :

1. این AS یک transit AS نیست . بنابراین از یک filter list برای جلوگیری از توزیع Route های بدست آمده استفاده می شود . تا Route ها به روتر های دیگر ارسال نشود .
2. ترافیک خروجی از این AS به سمت AS30 باید از AS40 انتقال یابند نه از طریق AS10 . در نتیجه n12 , n11 پارامتر Local Preference را (که به عنوان یک سیاست در سمت ورودی است) تغییر می دهند .

AS30 :

روتر n9 به عنوان Default Route برای AS65001 استفاده می شود . همچنین با تجمیع کردن prefix ها و حذف AS65001 از AS_PATH قبل از توزیع به دیگر روتر ها باعث خلاصه سازی Route ها شده و Route های AS65001 را به عنوان Route های خود و شبکه ها ی خود به دیگر روتر ها تبلیغ و توزیع می کند. سیاست های این AS :

1. ترافیک ورودی به این AS باید از AS10 بیاید و نه از AS40 . بدین منظور روتر n9 چند بار AS خود را به AS-PATH اضافه می کند .

AS40 :

1. ترافیک خروجی به مقصد AS20 باید به n11 ارسال شود .
2. روتر n10 از community به عنوان یک سیاست خروجی برای تاثیر گذاری روی ترافیک ورودی که از AS10 به AS30 می رود تا پس از آن به AS40 برسد ، استفاده می کند .

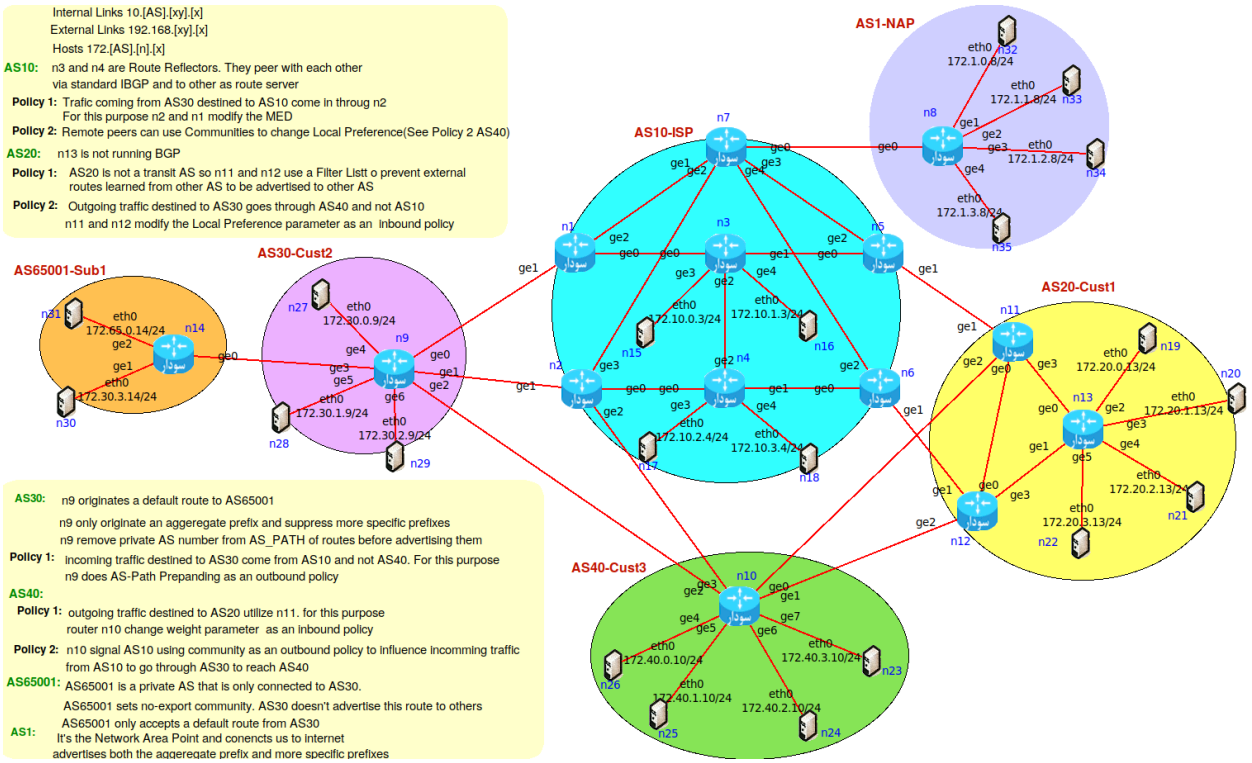
AS65001 :

یک AS خصوصی است که فقط به AS30 متصل است . ویژگی no-export community را تنظیم می کند و AS30 این Route را به دیگران ارسال نمی کند . AS65001 فقط یک default route از AS30 قبول می کند .

AS1 :

این AS یک Network Area Point است و ما را به اینترنت وصل می کند . prefix های تجمیع شده و دیگر prefix ها را تبلیغ می کند .

با توجه به توضیحات بالا حال به تنظیم سناریو می پردازیم :



AS10 1.2.2

n1 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n1
log file
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
ip prefix-list PLIST-AS10 seq 5 permit 172.10.0.0/22 le 32
!
interface loopback0
 ip address 10.10.0.1/32
!
interface ge0
 ip address 10.10.13.1/24
!
interface ge1
 ip address 192.168.19.1/24
!
interface ge2
 ip address 10.10.17.1/24
!
router eigrp 10
 network 10.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 neighbor 192.168.19.9 remote-as 30
 neighbor 192.168.19.9 password as3010
 neighbor 192.168.19.9 ttl-security hops 1
 !
 address-family ipv4 unicast
  neighbor 10.10.0.3 next-hop-self
  neighbor 10.10.0.4 next-hop-self
  neighbor 192.168.19.9 route-map RMAP-SET_LOC_PREF in
  neighbor 192.168.19.9 route-map RMAP-SET_MED out
 exit-address-family
!
bgp community-list standard CLIST-10:110 permit 10:110
bgp community-list standard CLIST-10:90 permit 10:90
!
route-map RMAP-SET_LOC_PREF permit 10
 match community CLIST-10:110
 set local-preference 110
!
```

(continues on next page)

(continued from previous page)

```
route-map RMAP-SET_LOC_PREF permit 20
 match community CLIST-10:90
 set local-preference 90
!
route-map RMAP-SET_LOC_PREF permit 30
!
route-map RMAP-SET_MED permit 10
 match ip address prefix-list PLIST-AS10
 set metric 1000
!
route-map RMAP-SET_MED permit 20
!
line vty
!
```

n2 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n2
log file
debug zebra event
debug if event
debug ospf event
debug vxlan event
debug vlan event
debug isis events
debug mpls ldp event
debug ipsec event
debug vrf event
debug rip events
debug qos event
debug nat44 event
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
ip prefix-list PLIST-AS10 seq 5 permit 172.10.0.0/22 le 32
!
interface loopback0
 ip address 10.10.0.2/32
!
interface ge0
 ip address 10.10.24.2/24
!
interface ge1
 ip address 192.168.29.2/24
!
interface ge2
 ip address 192.168.210.2/24
!
interface ge3
 ip address 10.10.27.2/24
```

(continues on next page)

(continued from previous page)

```
!
router eigrp 10
 network 10.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 neighbor 192.168.29.9 remote-as 30
 neighbor 192.168.29.9 password as1030
 neighbor 192.168.29.9 ttl-security hops 1
 neighbor 192.168.210.10 remote-as 40
 neighbor 192.168.210.10 password as1040
 neighbor 192.168.210.10 ttl-security hops 1
 !
 address-family ipv4 unicast
  neighbor 10.10.0.3 next-hop-self
  neighbor 10.10.0.4 next-hop-self
  neighbor 192.168.29.9 route-map RMAP-SET_LOC_PREF in
  neighbor 192.168.29.9 route-map RMAP-SET_MED out
  neighbor 192.168.210.10 route-map RMAP-SET_LOC_PREF in
 exit-address-family
!
bgp community-list standard CLIST-10:110 permit 10:110
bgp community-list standard CLIST-10:90 permit 10:90
!
route-map RMAP-SET_LOC_PREF permit 10
 match community CLIST-10:110
 set local-preference 110
!
route-map RMAP-SET_LOC_PREF permit 20
 match community CLIST-10:90
 set local-preference 90
!
route-map RMAP-SET_LOC_PREF permit 30
!
route-map RMAP-SET_MED permit 10
 match ip address prefix-list PLIST-AS10
 set metric 500
!
route-map RMAP-SET_MED permit 20
!
line vty
!
```

n3 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n3
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
interface loopback0
 ip address 10.10.0.3/32
!
interface ge0
 ip address 10.10.13.3/24
!
interface ge1
 ip address 10.10.35.3/24
!
interface ge2
 ip address 10.10.34.3/24
!
interface ge3
 ip address 172.10.0.1/24
!
interface ge4
 ip address 172.10.1.1/24
!
router eigrp 10
 network 10.10.0.0/16
 network 172.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 bgp cluster-id 10.10.0.0
 network 172.10.0.0/24
 network 172.10.1.0/24
 neighbor PGROUP-INTERNAL peer-group
 neighbor PGROUP-INTERNAL remote-as 10
 neighbor PGROUP-INTERNAL update-source loopback0
 neighbor 10.10.0.1 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.2 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.5 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.6 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.7 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 !
 address-family ipv4 unicast
  neighbor PGROUP-INTERNAL route-reflector-client
 exit-address-family
!
line vty
!
```

n4 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n4
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
interface loopback0
 ip address 10.10.0.4/32
!
interface ge3
 ip address 172.10.2.1/24
!
interface ge4
 ip address 172.10.3.1/24
!
interface ge0
 ip address 10.10.24.4/24
!
interface ge1
 ip address 10.10.46.4/24
!
interface ge2
 ip address 10.10.34.4/24
!
router eigrp 10
 network 10.10.0.0/16
 network 172.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 bgp cluster-id 10.10.0.0
 network 172.10.2.0/24
 network 172.10.3.0/24
 neighbor PGROUP-INTERNAL peer-group
 neighbor PGROUP-INTERNAL remote-as 10
 neighbor PGROUP-INTERNAL update-source loopback0
 neighbor 10.10.0.1 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.2 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.5 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.6 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.7 peer-group PGROUP-INTERNAL
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 !
 address-family ipv4 unicast
  neighbor PGROUP-INTERNAL route-reflector-client
 exit-address-family
!
line vty
```

n5 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n5
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
interface loopback0
 ip address 10.10.0.5/32
!
interface ge0
 ip address 10.10.35.5/24
!
interface ge1
 ip address 192.168.115.5/24
!
interface ge2
 ip address 10.10.57.5/24
!
router eigrp 10
 network 10.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 neighbor 192.168.115.11 remote-as 20
 neighbor 192.168.115.11 password as1030
 neighbor 192.168.115.11 ttl-security hops 1
 !
 address-family ipv4 unicast
  neighbor 10.10.0.3 next-hop-self
  neighbor 10.10.0.4 next-hop-self
  neighbor 192.168.115.11 route-map RMAP-SET_LOC_PREF in
 exit-address-family
!
bgp community-list standard CLIST-10:110 permit 10:110
bgp community-list standard CLIST-10:90 permit 10:90
!
route-map RMAP-SET_LOC_PREF permit 10
 match community CLIST-10:90
 set local-preference 90
!
route-map RMAP-SET_LOC_PREF permit 20
 match community CLIST-10:110
 set local-preference 110
!
route-map RMAP-SET_LOC_PREF permit 30
!
line vty
!
```

n6 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
hostname n6
interface loopback 0
no ip forwarding
no ipv6 forwarding
!
interface loopback0
 ip address 10.10.0.6/32
!
interface ge0
 ip address 10.10.46.6/24
!
interface ge1
 ip address 192.168.126.6/24
!
interface ge2
 ip address 10.10.67.6/24
!
router eigrp 10
 network 10.10.0.0/16
!
router bgp 10
 bgp log-neighbor-changes
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 neighbor 192.168.126.12 remote-as 20
 neighbor 192.168.126.12 password as1020
 neighbor 192.168.126.12 ttl-security hops 1
 !
 address-family ipv4 unicast
  neighbor 10.10.0.3 next-hop-self
  neighbor 10.10.0.4 next-hop-self
  neighbor 192.168.126.12 route-map RMAP-SET_LOC_PREF in
 exit-address-family
!
bgp community-list standard CLIST-10:110 permit 10:110
bgp community-list standard CLIST-10:90 permit 10:90
!
route-map RMAP-SET_LOC_PREF permit 10
 match community CLIST-10:90
 set local-preference 90
!
route-map RMAP-SET_LOC_PREF permit 20
 match community CLIST-10:110
 set local-preference 110
!
route-map RMAP-SET_LOC_PREF permit 30
!
line vty
!
```

n7 تنظیمات

```
frr version 7.2-0.1
frr defaults traditional
interface loopback 0
no ip forwarding
no ipv6 forwarding
hostname n7
!
interface loopback0
 ip address 10.10.0.7/32
!
interface ge0
 ip address 192.168.78.7/24
!
interface ge1
 ip address 10.10.17.7/24
!
interface ge2
 ip address 10.10.27.7/24
!
interface ge3
 ip address 10.10.57.7/24
!
interface ge4
 ip address 10.10.67.7/24
!
router eigrp 10
 network 10.10.0.0/16
!
router bgp 10
 neighbor 10.10.0.3 remote-as 10
 neighbor 10.10.0.3 update-source loopback0
 neighbor 10.10.0.4 remote-as 10
 neighbor 10.10.0.4 update-source loopback0
 neighbor 192.168.78.8 remote-as 1
 neighbor 192.168.78.8 ttl-security hops 1
 !
 address-family ipv4 unicast
  neighbor 10.10.0.3 next-hop-self
  neighbor 10.10.0.4 next-hop-self
 exit-address-family
!
line vty
!
```

AS20 2.2.2

n11 تنظیمات

```
Building configuration...

Current configuration:
!
hostname n11
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s
!
ip prefix-list PLIST-AS30 seq 5 permit 172.30.0.0/22 le 32




interface loopback0
 no shutdown
 ip address 10.20.0.11/32
!
interface ge0
 no shutdown
 ip address 10.20.112.11/24
!
interface ge1
 no shutdown
 ip address 192.168.115.11/24
!
interface ge2
 no shutdown
 ip address 192.168.111.11/24
!
interface ge3
 no shutdown
 ip address 10.20.113.11/24
!
router eigrp 20
 network 10.20.0.0/16
!
router bgp 20
 bgp log-neighbor-changes
 neighbor 10.20.0.12 remote-as 20
 neighbor 10.20.0.12 update-source loopback0
 neighbor 192.168.111.10 remote-as 40
 neighbor 192.168.115.5 remote-as 10
 neighbor 192.168.115.5 password as1020
 neighbor 192.168.115.5 ttl-security hops 1
```

(continues on next page)

(continued from previous page)

```
!
 address-family ipv4 unicast
  network 172.20.0.0/24
  network 172.20.1.0/24
  network 172.20.2.0/24
  network 172.20.3.0/24
  neighbor 10.20.0.12 next-hop-self
  neighbor 192.168.111.10 route-map RMAP-SET_LOC_PREF_HIGH in
  neighbor 192.168.111.10 filter-list 1 out
  neighbor 192.168.115.5 route-map RMAP-SET_LOC_PREF_LOW in
  neighbor 192.168.115.5 filter-list 1 out
 exit-address-family
!
bgp as-path access-list 1 permit ^$
!
route-map RMAP-SET_LOC_PREF_HIGH permit 10
 match ip address prefix-list PLIST-AS30
 set local-preference 150
!
route-map RMAP-SET_LOC_PREF_HIGH permit 20
!
route-map RMAP-SET_LOC_PREF_LOW permit 10
 match ip address prefix-list PLIST-AS30
 set local-preference 50
!
route-map RMAP-SET_LOC_PREF_LOW permit 20
!
end
```

n12 تنظیمات

```
Building configuration...

Current configuration:
!
hostname n12
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s
!
ip prefix-list PLIST-AS30 seq 5 permit 172.30.0.0/22 le 32




interface loopback0
 no shutdown
 ip address 10.20.0.12/32
```

(continues on next page)

(continued from previous page)

```
!
interface ge0
 no shutdown
 ip address 10.20.112.12/24
!
interface ge1
 no shutdown
 ip address 192.168.126.12/24
!
interface ge2
 no shutdown
 ip address 192.168.112.12/24
!
interface ge3
 no shutdown
 ip address 10.20.123.12/24
!
router eigrp 20
 network 10.20.0.0/16
!
router bgp 20
 bgp log-neighbor-changes
 neighbor 10.20.0.11 remote-as 20
 neighbor 10.20.0.11 update-source loopback0
 neighbor 192.168.112.10 remote-as 40
 neighbor 192.168.126.6 remote-as 10
 neighbor 192.168.126.6 password as1020
 neighbor 192.168.126.6 ttl-security hops 1
 !
 address-family ipv4 unicast
  network 172.20.0.0/24
  network 172.20.1.0/24
  network 172.20.2.0/24
  network 172.20.3.0/24
  neighbor 10.20.0.11 next-hop-self
  neighbor 192.168.112.10 route-map RMAP-SET_LOC_PREF_HIGH in
  neighbor 192.168.112.10 filter-list 1 out
  neighbor 192.168.126.6 route-map RMAP-SET_LOC_PREF_LOW in
  neighbor 192.168.126.6 filter-list 1 out
 exit-address-family
!
bgp as-path access-list 1 permit ^$
!
route-map RMAP-SET_LOC_PREF_HIGH permit 10
 match ip address prefix-list PLIST-AS30
 set local-preference 150
!
route-map RMAP-SET_LOC_PREF_HIGH permit 20
!
route-map RMAP-SET_LOC_PREF_LOW permit 10
 match ip address prefix-list PLIST-AS30
 set local-preference 50
!
route-map RMAP-SET_LOC_PREF_LOW permit 20
!
end
```

(continues on next page)

(continued from previous page)

n13 تنظیمات

```
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s
!
ip route 0.0.0.0/0 10.20.113.11
ip route 0.0.0.0/0 10.20.123.12
!
interface loopback0
 no shutdown
 ip address 10.20.0.13/32
!
interface loopback1
!
interface loopback2
!
interface loopback3
!
interface loopback4
!
interface ge0
 no shutdown
 ip address 10.20.113.13/24
!
interface ge1
 no shutdown
 ip address 10.20.123.13/24
!
interface ge2
 no shutdown
 ip address 172.20.0.1/24
!
interface ge3
 no shutdown
 ip address 172.20.1.1/24
!
interface ge4
 no shutdown
 ip address 172.20.2.1/24
!
interface ge5
 no shutdown
 ip address 172.20.3.1/24
!
router eigrp 20
 network 10.20.0.0/16
 network 172.20.0.0/16
```

(continues on next page)

(continued from previous page)

```
!
end
```

AS30 3.2.2

n9 تنظیمات

```
Building configuration...

Current configuration:
!
hostname n9
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s
!
ip prefix-list PLIST-AS30 seq 5 permit 172.30.0.0/22 le 32
!
interface loopback0
 no shutdown
 ip address 10.30.0.9/32
!
interface loopback1
!
interface loopback2
!
interface loopback3
!
interface ge0
 no shutdown
 ip address 192.168.19.9/24
!
interface ge1
 no shutdown
 ip address 192.168.29.9/24
!
interface ge2
 no shutdown
 ip address 192.168.109.9/24
!
interface ge3
 no shutdown
 ip address 192.168.149.9/24
!
interface ge4
 no shutdown
 ip address 172.30.0.1/24
```

(continues on next page)

(continued from previous page)

```
!
interface ge5
 no shutdown
 ip address 172.30.1.1/24
!
interface ge6
 no shutdown
 ip address 172.30.2.1/24
!
router bgp 30
 bgp log-neighbor-changes
 neighbor 192.168.19.1 remote-as 10
 neighbor 192.168.19.1 password as3010
 neighbor 192.168.19.1 ttl-security hops 1
 neighbor 192.168.29.2 remote-as 10
 neighbor 192.168.29.2 password as1030
 neighbor 192.168.29.2 ttl-security hops 1
 neighbor 192.168.109.10 remote-as 40
 neighbor 192.168.109.10 password as4030
 neighbor 192.168.149.14 remote-as 65001
 neighbor 192.168.149.14 ttl-security hops 1
 !
 address-family ipv4 unicast
  network 172.30.0.0/24
  network 172.30.1.0/24
  network 172.30.2.0/24
  aggregate-address 172.30.0.0/22 summary-only
  neighbor 192.168.19.1 remove-private-AS
  neighbor 192.168.29.2 remove-private-AS
  neighbor 192.168.109.10 remove-private-AS
  neighbor 192.168.109.10 route-map RMAP-SET_PREPEND out
  neighbor 192.168.149.14 default-originate
 exit-address-family
!
route-map RMAP-SET_PREPEND permit 10
 match ip address prefix-list PLIST-AS30
 set as-path prepend 30 30 30 30 30
!
route-map RMAP-SET_PREPEND permit 20
!
end
```

AS40 4.2.2

تنظیمات n10

```
Building configuration...

Current configuration:
```

(continues on next page)

(continued from previous page)

```
!
hostname n10
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s
!
ip prefix-list PLIST-AS20 seq 5 permit 172.20.0.0/22 le 32
ip prefix-list PLIST-AS40 seq 5 permit 172.40.0.0/22 le 32
!
interface loopback0
 no shutdown
 ip address 10.40.0.10/32
!
interface loopback1
!
interface loopback2
!
interface loopback3
!
interface loopback4
!
interface ge0
 no shutdown
 ip address 192.168.111.10/24
!
interface ge1
 no shutdown
 ip address 192.168.112.10/24
!
interface ge2
 no shutdown
 ip address 192.168.109.10/24
!
interface ge3
 no shutdown
 ip address 192.168.210.10/24
!
interface ge4
 no shutdown
 ip address 172.40.0.1/24
!
interface ge5
 no shutdown
 ip address 172.40.1.1/24
!
interface ge6
 no shutdown
 ip address 172.40.2.1/24
!
interface ge7
 no shutdown
 ip address 172.40.3.1/24
!
router bgp 40
```

(continues on next page)

(continued from previous page)

```
 bgp log-neighbor-changes
 neighbor 192.168.109.9 remote-as 30
 neighbor 192.168.109.9 password as4030
 neighbor 192.168.111.11 remote-as 20
 neighbor 192.168.112.12 remote-as 20
 neighbor 192.168.210.2 remote-as 10
 neighbor 192.168.210.2 password as1040
 neighbor 192.168.210.2 ttl-security hops 1
 !
 address-family ipv4 unicast
  network 172.40.0.0/24
  network 172.40.1.0/24
  network 172.40.2.0/24
  network 172.40.3.0/24
  neighbor 192.168.109.9 route-map RMAP-SET_COMMUN out
  neighbor 192.168.111.11 route-map RMAP-SET_WEIGHT_HIGH in
  neighbor 192.168.112.12 route-map RMAP-SET_WEIGHT_LOW in
 exit-address-family
!
route-map RMAP-SET_COMMUN permit 10
 match ip address prefix-list PLIST-AS40
 set community 10:110 additive
!
route-map RMAP-SET_COMMUN permit 20
!
route-map RMAP-SET_WEIGHT_HIGH permit 10
 match ip address prefix-list PLIST-AS20
 set weight 2048
!
route-map RMAP-SET_WEIGHT_HIGH permit 20
!
route-map RMAP-SET_WEIGHT_LOW permit 10
 match ip address prefix-list PLIST-AS20
 set weight 1024
!
route-map RMAP-SET_WEIGHT_LOW permit 20
!
end
```

AS65001 5.2.2

تنظیمات n14

```
Building configuration...

Current configuration:
!
hostname n14
system update server-url https://update.soodar.ir
```

(continues on next page)

(continued from previous page)

```
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s




!
interface loopback0
 no shutdown
 ip address 10.65.0.14/32
!
interface loopback1
!
interface loopback2
!
interface ge0
 no shutdown
 ip address 192.168.149.14/24
!
interface ge1
 no shutdown
 ip address 172.30.3.1/24
!
interface ge2
 no shutdown
 ip address 172.65.0.1/24
!
router bgp 65001
 bgp log-neighbor-changes
 neighbor 192.168.149.9 remote-as 30
 neighbor 192.168.149.9 ttl-security hops 1
 !
 address-family ipv4 unicast
  network 172.30.3.0/24
  network 172.65.0.0/24
  neighbor 192.168.149.9 distribute-list ACL-DEFAULT_ROUTE in
  neighbor 192.168.149.9 route-map RMAP-NO_EXPORT out
 exit-address-family
!
route-map RMAP-NO_EXPORT permit 10
 match ip address ACL-NO_EXPORT
 set community no-export
!
route-map RMAP-NO_EXPORT permit 20
!
end
```

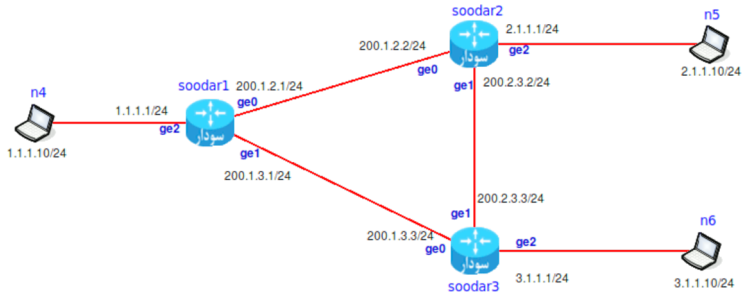
AS1 6.2.2

n8 تنظیمات

```
Building configuration...

Current configuration:
!
hostname n8
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
enable password s

!
interface ge0
 no shutdown
 ip address 192.168.78.8/24
!
interface ge1
 no shutdown
 ip address 172.1.0.1/24
!
interface ge2
 no shutdown
 ip address 172.1.1.1/24
!
interface ge3
 no shutdown
 ip address 172.1.2.1/24
!
interface ge4
 no shutdown
 ip address 172.1.3.1/24
!
router bgp 1
 neighbor 192.168.78.7 remote-as 10
 neighbor 192.168.78.7 ttl-security hops 1
 !
 address-family ipv4 unicast
  network 172.1.0.0/24
  network 172.1.1.0/24
  network 172.1.2.0/24
  network 172.1.3.0/24
  aggregate-address 172.1.0.0/16
 exit-address-family
!
end
```

IS-IS 3.2

در سناریوی زیر قصد داریم مسیریابی پویا را با استفاده از پروتکل IS-IS در روتر ها تنظیم کنیم. فرض شده است که IP در اینترفیس ها در بخش تنظیم ip در اینترفیس تنظیم شده اند :



1.3.2 ایجاد یک instance از isis با نام test :

```
soodar1(config)# router isis test
```

2.3.2 تنظیم net در isis :

```
soodar1(config-router)# net 47.0023.0000.0000.0000.0000.0000.0000.1900.1111.00
soodar1(config-router)# redistribute ipv4 connected level-1
```

net شامل چند بخش می باشد :

47.0023 : area ID معادل است با area در ospf

0000.0000.0000.0000.0000.0000.1900.1111 : id system که باید در کل شبکه منحصر بفرد باشد چون سایز آن با mac برابر است معمولا از mac روتر به جای آن استفاده می شود .

نکته

این بخش میتواند به صورت خلاصه و تنها با استفاده از دو بایت (8 رقم) نیز تنظیم شود .

---

00 : مشخص می کند که device یک روتر است .

3.3.2 بازنشر شبکه های **connected** توسط **isis** :

```
soodar1(config-router)# redistribute ipv4 connected level-1
```

نکته

توجه کنید level-1 مشخص می کند که شبکه های connected فقط در area محلی ارسال توزیع شود و در area های دیگر ارسال نگردد . اگر از level-2 استفاده کنید در area های دیگر نیز توزیع خواهد شد و روتر هایی که area id متفاوت با این روتر را دارند route به شبکه های connected شما را دارند .

4.3.2 تنظیم کردن اینترفیس :

```
soodar1(config)# int ge0
soodar1(config-if)#  ip router isis test
soodar1(config-if)# q
soodar1(config)# int ge1
soodar1(config-if)#  ip router isis test
```

هر اینترفیسی که در روتر isis test قرار داشته باشد در جدول روتینگ isis اضافه می شود و می تواند با دیگر اعضای شبکه ارتباط داشته باشد .

تنظیم soodar3 , soodar2 نیز به شکل زیر است :

```
soodar2(config)# router isis test
soodar2(config-router)# net 47.0023.1900.2222.00
soodar2(config-router)# redistribute ipv4 connected level-1
soodar2(config-router)# q
soodar2(config)# int ge0
soodar2(config-if)#  ip router isis test
soodar2(config-if)# q
soodar2(config)# int ge1
soodar2(config-if)#  ip router isis test
---------------------------------------
soodar3(config)# router isis test
soodar3(config-router)# net 47.0023.1900.3333.00
soodar3(config-router)# redistribute ipv4 connected level-1
soodar3(config-router)# q
soodar3(config)# int ge0
soodar3(config-if)#  ip router isis test
soodar3(config-if)# q
soodar3(config)# int ge1
soodar3(config-if)#  ip router isis tes
```

5.3.2 مشاهده جدول مسیریابی

پس از تنظیم کردن روتر ها جدول مسیریابی را بررسی می کنیم . مشخص است که شبکه های 2.1.1.0 و 3.1.1.0 توسط IS-IS مسیریابی شده اند (حرف I مشخص کننده این است که این مسیر توسط IS-IS در جدول اضافه شده است) :

```
soodar1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.0/24 is directly connected, ge2, 00:01:59
I>* 2.1.1.0/24 [115/20] via 200.1.2.2, ge0, 00:01:37
I>* 3.1.1.0/24 [115/20] via 200.1.3.3, ge1, 00:01:37
C>* 200.1.2.0/24 is directly connected, ge0, 00:01:59
C>* 200.1.3.0/24 is directly connected, ge1, 00:01:59
I>* 200.2.3.0/24 [115/20] via 200.1.2.2, ge0, 00:01:37
```

6.3.2 بررسی عملکرد IS-IS

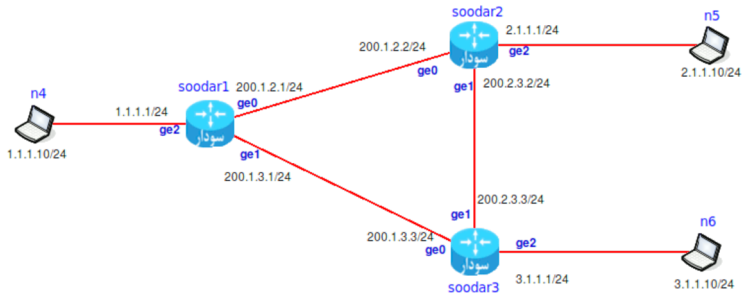
توجه

باید ارتباط شبکه های 1.1.1.0 و 2.1.1.0 و 3.1.1.0 برقرار باشد .

OSPF 4.2

پروتکل ospf هم می تواند در vrf و هم در حالت multi instance استفاده شود شما می توانید ospf را در vrf های جداگانه اجرا کنید یا instance های مختلفی از ospf را استفاده نمایید . در این بخش ما از حالت ساده و پیش فرض برای تنظیم ospf استفاده می کنیم .

در این روتر در اینترفیس های ge1 , ge0 پروتکل OSPF فعال شده است ، همچنین فرض شده است که IP در اینترفیس ها در بخش تنظیم ip در اینترفیس تنظیم شده اند :



1.4.2 تنظیمات در soodar1

تنظیم ospf : با دستور زیر ospf در روتر فعال می شود :

```
soodar1(config)# router ospf
soodar1(config-router)# ospf router-id 222.1.1.1
soodar1(config-router) # end
soodar1#
```

برای اینکه با روتر های همسایه صحبت کند باید اینترفیس هایی که قرار است در آن ها ospf اجرا شود را مشخص کنیم یا اینکه شبکه هایی(network) که قرار است ospf در آن با بقیه صحبت کند مشخص کنیم .

نکته

دقت شود یکی از این دو تنظیم باید انجام شود یا تنظیم اینترفیس یا تنظیم شبکه .

نحوه تنظیم ospf در اینترفیس :

```
soodar1(config)# interface geX
soodar1(config-if)# ip ospf area Y
```

نحوه اضافه کردن network به ospf :

```
soodar1(config)# router ospf
soodar1(config-if)# network x.x.x.x/x area x
```

در اینجا ما قصد داریم ospf با روتر های soodar3 , soodar2 صحبت کند که هم می توانیم اینترفیس های ge1 , ge0 را برای ospf تنظیم کنیم و هم می توانیم شبکه های 200.1.3.0/24 , 200.1.2.0/24 در ospf تنظیم کنیم . ما در این کارگاه در اینترفیس تنظیمات را انجام می دهیم :

```
soodar1(config)# interface ge0
soodar1(config-if)# ip ospf area 0
soodar1(config-if)# q
soodar1(config)# interface ge1
soodar1(config-if)# ip ospf area 0
```

توزیع route ها توسط OSPF : کلیه route هایی که توسط پروتکل های دیگر ,rip,bgp,static,connected)...)در سیستم اضافه شده اند را می توان در ospf تبلیغ یا توزیع کرد که ما در اینجا شبکه های connected را در ospf توزیع می کنیم:

```
soodar1(config)# router ospf
soodar1(config-router)# redistribute connected
```

تنظیمات در soodar2 , soodar3

soodar2

```
soodar2(config)# router ospf
soodar2(config-router)# ospf router-id 222.2.2.2
```

(continues on next page)

(continued from previous page)

```
soodar2(config-router)# redistribute connected
soodar2(config-router) # q
soodar2(config)# int ge0
soodar2(config-if)# ip ospf area 0
soodar2(config)# q
soodar2(config)# int ge1
soodar2(config-if)# ip ospf area 0
```

soodar3

```
soodar3(config)# router ospf
soodar3(config-router)# ospf router-id 222.3.3.3
soodar3(config-router)# redistribute connected
soodar3(config-router) # q
soodar3(config)# int ge0
soodar3(config-if)# ip ospf area 0
soodar3(config)# q
soodar3(config)# int ge1
soodar3(config-if)# ip ospf area 0
```

پس از اعمال شدن تنظیمات کمی صبر می کنیم تا روتر ها با استفاده از پروتکل OSPF جدول Routing را تشکیل دهند . سپس با استفاده از دستور زیر جدول Routing که با استفاده از پروتکل OSPF تشکیل شده است را مشاهده می کنیم :

```
soodar1# sh ip ospf route
```

یا

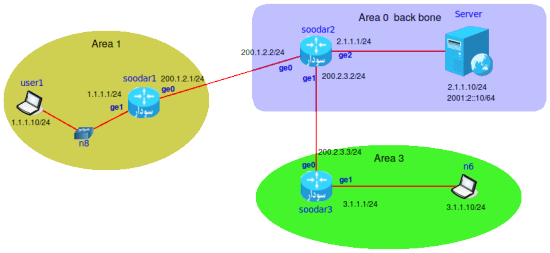
```
soodar1# sh ip fib
```

خاطر نشان

حال باید route بین شبکه های سه روتر برقرار باشد .برای مثال باید نود n4 آدرس 2.1.1.10 را پینگ کند .
با قطع لینک ارتباطی بین soodar1 , soodar2 مسیر ها بروز رسانی شوند و مجددا ارتباط بین دو نود از مسیر جدید برقرار شود .

OSPF-MultiArea

اگر سناریویی دارید که در آن area های مختلفی دارید از روش زیر برای پیکربندی OSPF در حالت چند Area استفاده کنید . در شکل زیر روتر soodar1 در area 1 ، روتر soodar3 در area 3 قرار دارد و روتر soodar2 در area 0 به عنوان backbone در شبکه قرار گرفته است .



توجه

باید ابتدا یک backbone (area 0) تشکیل دهید.
در صورت استفاده از backbone به صورت پیش فرض همه شبکه ها بین Area های مختلف distribute می شوند .

تنظیمات به شکل زیر باید انجام گیرد:

soodar1 تنظیمات

```
interface ge0
 ip address 200.1.2.1/24
!
interface ge1
 ip address 1.1.1.1/24
!
router ospf
 ospf router-id 222.0.0.1
 redistribute connected
 network 1.1.1.0/24 area 1
 network 200.1.2.0/24 area 0
```

soodar2 تنظیمات

```
interface ge0
 ip address 200.1.2.2/24
!
interface ge1
 ip address 200.2.3.2/24
!
router ospf
 ospf router-id 222.0.0.2
 redistribute connected
 network 200.1.2.0/24 area 0
 network 200.2.3.0/24 area 0
!
```

soodar3 تنظیمات

```
interface ge0
 ip address 200.2.3.3/24
!
interface ge1
 ip address 3.1.1.1/24
!
router ospf
 ospf router-id 222.0.0.3
 redistribute connected
 network 3.1.1.0/24 area 3
 network 200.2.3.0/24 area 0
```

خاطر نشان

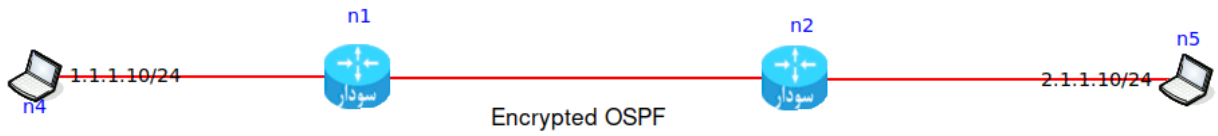
باید ارتباط بین 3.1.1.10 و 1.1.1.10 که در دو Area مختلف قرار دارند برقرار شود .

Encrypted OSPF 5.2

در روتر سودار امکان استفاده از OSPF به صورت رمز شده وجود دارد . دیتای ospf به صورت رمز شده بین روتر های سودار انتقال می یابد . برای این که این قابلیت فعال شود باید در اینترفیس که ospf در آن اجرا است با استفاده از دستورات زیر encryption را فعال کنید :

```
soodar(config) # int ge0
soodar(config-if)# ip ospf encryption
soodar(config-if)# ip ospf encryption-key AAAAAAAAAAAAAAAA
```

در سناریوی زیر در اینترفیس های ge0 هر دو روتر ospf را در حالت رمز شده فعال می کنیم :



1.5.2 تنظیمات در n1

```
interface ge0
 no shutdown
 ip address 200.1.2.1/24
 ip ospf hello-interval 3
 ip ospf encryption
 ip ospf encryption-key 0AAAAAAAAAAAAAAAA00000000000000
exit
```

(continues on next page)

(continued from previous page)

```
!
interface ge1
 no shutdown
 ip address 1.1.1.1/24
exit
!
router ospf
 redistribute connected
 network 200.1.2.0/24 area 0
exit
!
```

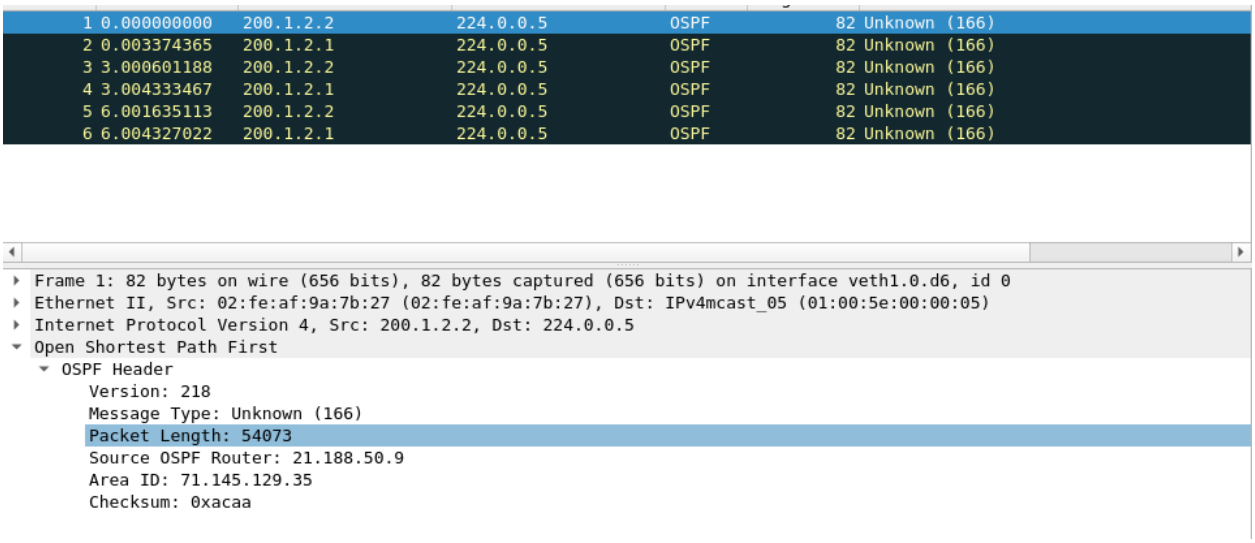
2.5.2 تنظیمات در **n2**

```
interface ge0
 no shutdown
 ip address 200.1.2.2/24
 ip ospf hello-interval 3
 ip ospf encryption
 ip ospf encryption-key 0AAAAAAAAAAAAAAAA00000000000000
exit
!
interface ge1
 no shutdown
 ip address 2.1.1.1/24
exit
!
router ospf
 redistribute connected
 network 200.1.2.0/24 area 0
exit
!
```

نکته

دقت کنید کلید به فرمت hex و 32 کارکتر باشد در صورتی که تعداد حروف رمز کمتر از آن باشد با صفر پر می شود

ما بسته های ospf را که بین روتر ها تبادل می شوند sniff کرده ایم و مشخص است که دیتای آنها رمز شده است و مقادیر معتبر و واقعی در آن دیده نمی شود :



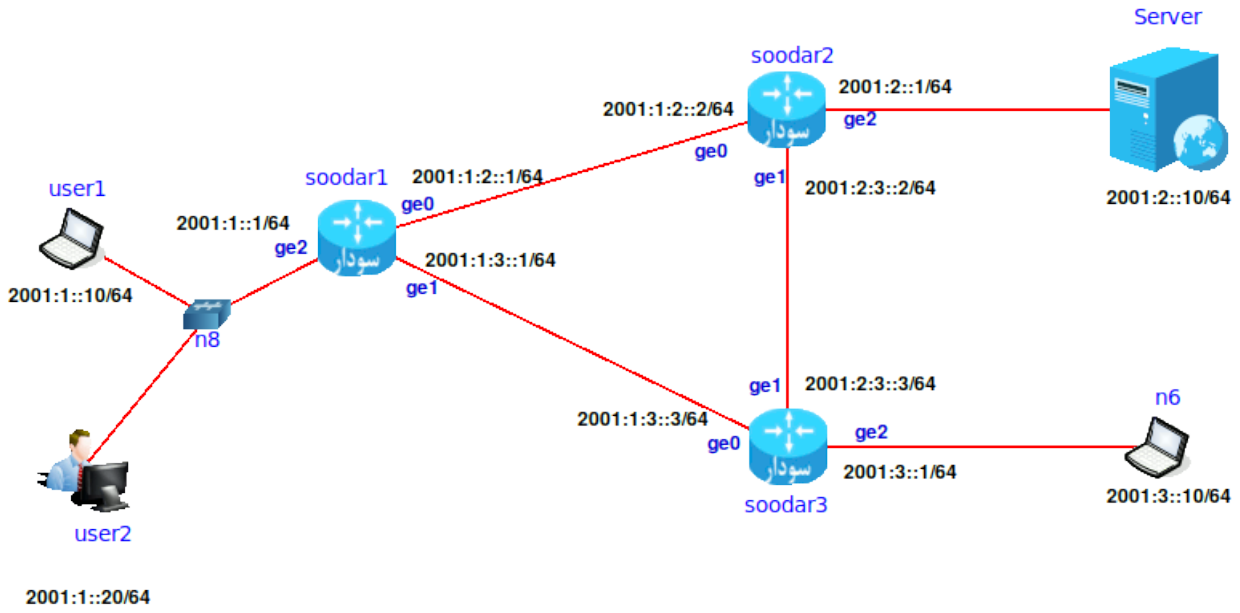
با دستور زیر نیز مشخص می شود که ospf ها با یکدیگر neighbor شده اند :

```
n1# sh ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface                RXmtL RqstL DBsmL
200.1.2.2         1 Full/DR         10.414s 200.1.2.2       ge0:200.1.2.1                1     0     0

n1#
```

OSPFv3 6.2

در سناریوی زیر قصد داریم مسیریابی پویا را با استفاده از پروتکل OSPFv3 در روتر ها تنظیم کنیم.

در این روتر در اینترفیس های ge1 , ge2 پروتکل OSPFv3 فعال شده است :



1.6.2 تنظیم ipv6 در اینترفیس ها

```
soodar1(config)# int ge0
soodar1(config-if)# ipv6 address 2001:1:2::1/64
soodar1(config-if)# q
soodar1(config)# int ge1
soodar1(config-if)# ipv6 address 2001:1:3::1/64
soodar1(config-if)# q
soodar1(config)# int ge2
soodar1(config-if)# ipv6 address 2001:1::1/64
```

```
soodar2(config)# int ge0
soodar2(config-if)# ipv6 address 2001:1:2::2/64
soodar2(config-if)# q
soodar2(config)# int ge1
soodar2(config-if)# ipv6 address 2001:2:3::2/64
soodar2(config-if)# q
soodar2(config)# int ge2
soodar2(config-if)# ipv6 address 2001:2::1/64
```

```
soodar3(config)# int ge0
soodar3(config-if)# ipv6 address 2001:1:3::3/64
soodar3(config-if)# q
soodar3(config)# int ge1
soodar3(config-if)# ipv6 address 2001:2:3::3/64
soodar3(config-if)# q
soodar3(config)# int ge2
soodar3(config-if)# ipv6 address 2001:3::1/64
```

(continues on next page)

(continued from previous page)

```
```

2.6.2 ایجاد یک روتر OSPF6

```
soodar1(config)# router ospf6
soodar1(config-router)# ospf6 router-id 222.1.1.1
```

3.6.2 اضافه کردن اینترفیس ها به تنظیمات OSPF6

```
soodar1(config-router)# interface ge1 area 0.0.0.0
soosar1(config-router)# interface ge2 area 0.0.0.0
soodar1(config-router) # end
soodar1# write
```

4.6.2 توزیع route ها توسط OSPF6

```
soodar1(config-router)# redistribute connected
```

روتر های soodar3 , soodar2 را نیز به همین ترتیب تنظیم می کنیم :

```
soodar2(config)# router ospf6
soodar2(config-router)# ospf6 router-id 222.2.2.2
soodar2(config-router)# interface ge1 area 0.0.0.0
soosar2(config-router)# interface ge2 area 0.0.0.0
soodar2(config-router)# redistribute connected
soodar2(config-if) # end
soodar2# write
-----------------------------------------
soodar3(config)# router ospf6
soodar3(config-router)# ospf6 router-id 222.3.3.3
soodar3(config-router)# interface ge1 area 0.0.0.0
soosar3(config-router)# interface ge2 area 0.0.0.0
soodar3(config-router)# redistribute connected
soodar3(config-if) # end
soodar3# write
```

5.6.2 مشاهده جدول مسیریابی

سپس با استفاده از دستور زیر جدول Routing که با استفاده از پروتکل OSPF6 تشکیل شده است را مشاهده می کنیم :

```
soodar1# sh ip ospf6 route
```

یا

```
soodar1# sh ipv6 fib
```

```
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 2001:1:2::/64 is directly connected, ge0, 00:01:07
C>* 2001:1:3::/64 is directly connected, ge1, 00:01:07
O>* 2001:2:3::/64 [110/20000] via fe80::fe:54ff:fecb:9f70, ge1, 00:00:21
  *                           via fe80::fe:f9ff:fed7:c5d8, ge0, 00:00:21
C>* 2001:1::/64 is directly connected, ge2, 00:01:08
O>* 2001:2::/64 [110/20000] via fe80::fe:f9ff:fed7:c5d8, ge0, 00:00:21
O>* 2001:3::/64 [110/20000] via fe80::fe:54ff:fecb:9f70, ge1, 00:00:21
```

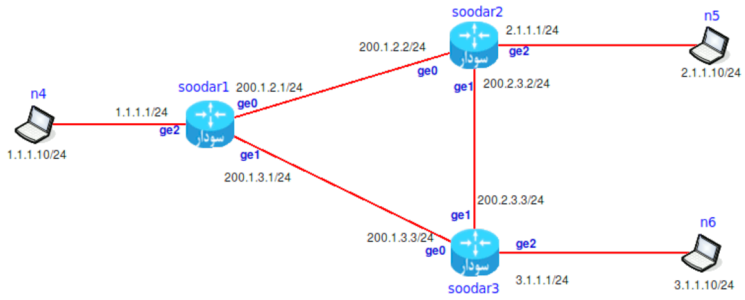
6.6.2 بررسی عملکرد

باید ارتباط شبکه های پشت روتر ها با هم برقرار باشد :

```
soodar1# ping 2001:2::10
64 bytes from 2001:2::10: icmp_seq=3 ttl=63 time=0.386 ms
soodar1# ping 2001:3::10
64 bytes from 2001:3::10: icmp_seq=3 ttl=63 time=0.341 ms
```

RIP 7.2

در سناریوی زیر قصد داریم مسیریابی پویا را با استفاده از RIP در روتر ها تنظیم کنیم. همچنین فرض شده است که IP در اینترفیس ها در بخش تنظیم ip در اینترفیس تنظیم شده اند :



ابتدا به تنظیمات روتر RIP با دستورات زیر وارد می‌شویم :

```
soodar1# conf t
soodar1(config)#
soodar1(config)# router rip
soodar1(config-router)#
```

1.7.2 اضافه کردن شبکه ها

می توان با استفاده از آدرس شبکه و هم چنین نام اینترفیس شبکه هایی که قرار است روی آن ها RIPng اجرا شود و با دیگر روتر ها صحبت کند مشخص کرد :

```
soodar1(config-router)# network 200.1.2.0/24
soodar1(config-router)# network 200.1.3.0/24
```

2.7.2 توزیع مسیرها

```
soodar1(config-router)# redistribute connected
```

تنظیمات را به همین شکل برای soodar2 و soodar3 انجام می دهیم :

```
soodar2# conf t
soodar2(config)# router rip
soodar2(config-router)# network 200.1.2.0/24
soodar2(config-router)# network 200.2.3.0/24
soodar2(config-router)# redistribute connected
soodar2(config-router)#end
soodar2# write
-----------------------------
soodar3# conf t
soodar3(config)# router rip
soodar3(config-router)# network 200.2.3.0/24
soodar3(config-router)# network 200.1.3.0/24
soodar3(config-router)# redistribute connected
soodar3(config-router)#end
soodar3# write
```

3.7.2 مشاهده جدول مسیریابی

با دستور زیر جدول مسیریابی تشکیل شده را مشاهده می کنیم که شبکه های 2.1.1.0 و 3.1.1.0 در روتر soodar1 توسط پروتکل RIP مسیریابی شده اند :

```
soodar1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.0/24 is directly connected, ge2, 00:04:19
R>* 2.1.1.0/24 [120/2] via 200.1.2.2, ge0, 00:04:13
R>* 3.1.1.0/24 [120/2] via 200.1.3.3, ge1, 00:04:15
C>* 200.1.2.0/24 is directly connected, ge0, 00:04:19
C>* 200.1.3.0/24 is directly connected, ge1, 00:04:19
R>* 200.2.3.0/24 [120/2] via 200.1.2.2, ge0, 00:04:13
```

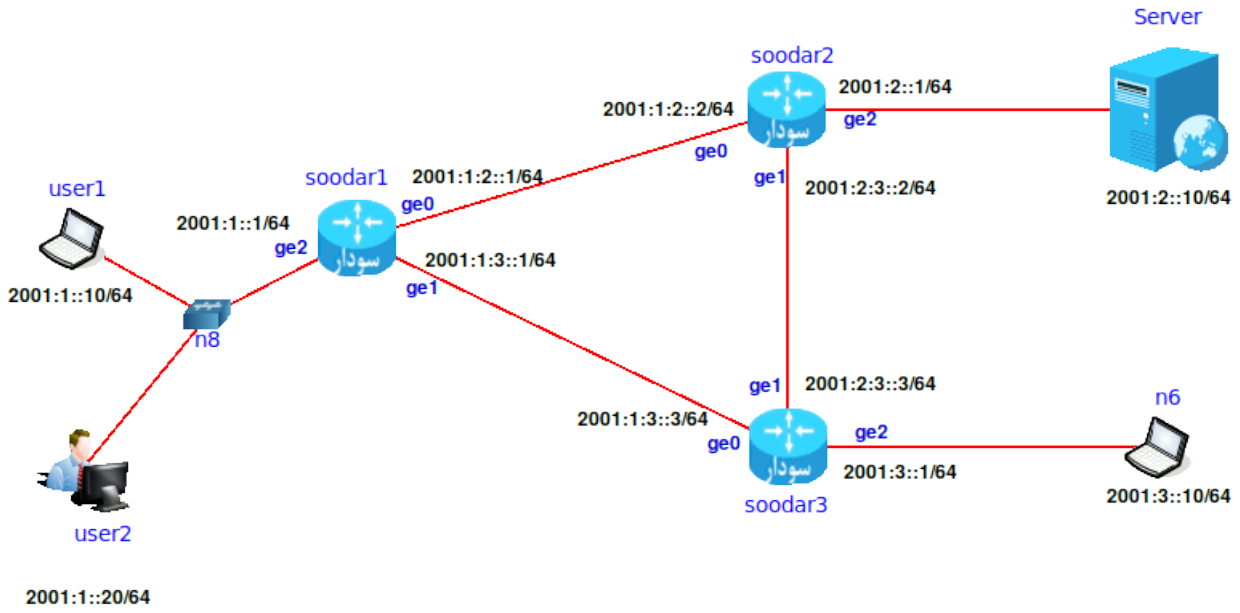
توجه

باید ارتباط شبکه ای 3.1.1.0 , 2.1.1.0 , 1.1.1.0 بوسیله پروتکل RIP برقرار شده باشد .

8.2 RIPng

در سناریوی زیر قصد داریم مسیریابی پویا را با استفاده از پروتکل RIPng در روتر ها تنظیم کنیم.

در این روتر در اینترفیس های ge1 , ge2 پروتکل RIPng فعال شده است :



ابتدا به تنظیمات روتر RIPng با دستورات زیر وارد می‌شویم :

```
soodar1# conf t
soodar1(config)# router ripng
```

1.8.2 اضافه کردن شبکه ها

می توان با استفاده از آدرس شبکه و هم چنین نام اینترفیس شبکه هایی که قرار است روی آن ها RIPng اجرا شود و با دیگر روتر ها صحبت کند مشخص کرد :

```
soodar1(config-router)# network 2001:1:2::0/64
soodar1(config-router)# network ge2
```

2.8.2 توزیع مسیرها

```
soodar1(config-router)# redistribute connected
```

به همین ترتیب تنظیمات برای روتر های soodar3 ، soodar2 را نیز انجام می دهیم :

```
soodar2# conf t
soodar2(config)# router ripng
soodar2(config-router)# network 2001:1:2::0/64
soodar2(config-router)# network 2001:1:3::0/64
soodar2(config-router)# redistribute connected
soodar2(config-router)#end
soodar2# write
-----------------------------------
```

(continues on next page)

(continued from previous page)

```
soodar3# conf t
soodar3(config)# router ripng
soodar3(config-router)# network 2001:1:3::0/64
soodar3(config-router)# network 2001:2:3::0/64
soodar3(config-router)# redistribute connected
soodar3(config-router)#end
soodar3# write
```

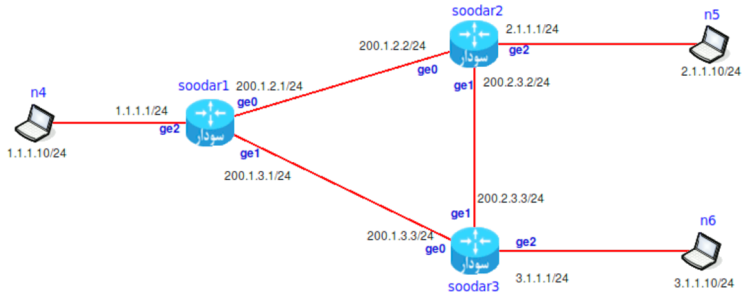
3.8.2 مشاهده جدول مسیریابی

با استفاده از دستور زیر می توانید جدول مسیر یابی را مشاهده کنید :

```
soodar1# sh ipv6 fib
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 2001:1::/64 is directly connected, ge2, 00:08:45
C>* 2001:1:2::/64 is directly connected, ge0, 00:09:14
C>* 2001:1:3::/64 is directly connected, ge1, 00:09:02
R>* 2001:2::/64 [120/2] via fe80::fe:50ff:fefb:819a, ge0, 00:02:38
R>* 2001:2:3::/64 [120/2] via fe80::fe:50ff:fefb:819a, ge0, 00:06:40
R>* 2001:3::/64 [120/2] via fe80::fe:a2ff:fe16:dafd, ge1, 00:02:49
```

Static Route 9.2

پس از تنظیم ip در اینترفیس های روتر ها با اضافه کردن static Route ارتباط بین روتر ها و شبکه های آن ها را با یکدیگر تست می کنیم :



static Route در soodar1 :

```
soodar1# conf t
soodar1(config) #
```

(continues on next page)

(continued from previous page)

```
soodar1(config) # ip route 2.1.1.0/24  200.1.2.2
soodar1(config) # ip route 3.1.1.0/24  200.1.3.3
```

static Route در **soodar2** :

```
soodar2# conf t
soodar2(config) #
soodar2(config) # ip route 1.1.1.0/24  200.1.2.1
soodar2(config) # ip route 3.1.1.0/24  200.2.3.3
```

static Route در **soodar3** :

```
soodar3# conf t
soodar3(config) #
soodar3(config) # ip route 1.1.1.0/24  200.1.3.1
soodar3(config) # ip route 2.1.1.0/24  200.2.3.2
```

با استفاده از اینترفیس هم به شکل زیر می توان static route اضافه کرد :

```
soodar3(config) # ip route 1.1.1.0/24  ge0
soodar3(config) # ip route 2.1.1.0/24  ge1
```

1.9.2 مشاهده جدول **Routing**

حال با استفاده از دستور زیر Route های اضافه شده را در هر روتر مشاهده می کنیم :

```
soodar1# sh ip route
or
soodar1# sh ip fib
or
soodar1# sh ip fib static
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 2.1.1.0/24 [1/0] via 200.1.2.2, soo0, 00:00:18
S>* 3.1.1.0/24 [1/0] via 200.1.3.3, soo1, 00:00:09
```

نکته

1. باید Route های تعریف شده در روتر ها اضافه شده باشند و با حرف S که نشانه static بودن آن هاست مشخص شده باشند.
2. همچنین باید ارتباط شبکه های هر سه روتر با یکدیگر برقرار باشد . این ارتباطات با ping از شبکه های مختلف به یکدیگر تست می شود .
3. با قطع کردن لینک بین 1 و 2 ارتباط بین 1.1.1.10 و 2.1.1.10 قطع شود چون route استاتیک است و update نمی شود.

2.9.2 static route در vrf

برای اضافه کردن route در vrf می توان به شکل زیر عمل کرد :

```
soodar3# conf t
soodar3(config) # vrf red
soodar3(config-vrf) # ip route 12.128.1.0/24  200.1.15.2
```

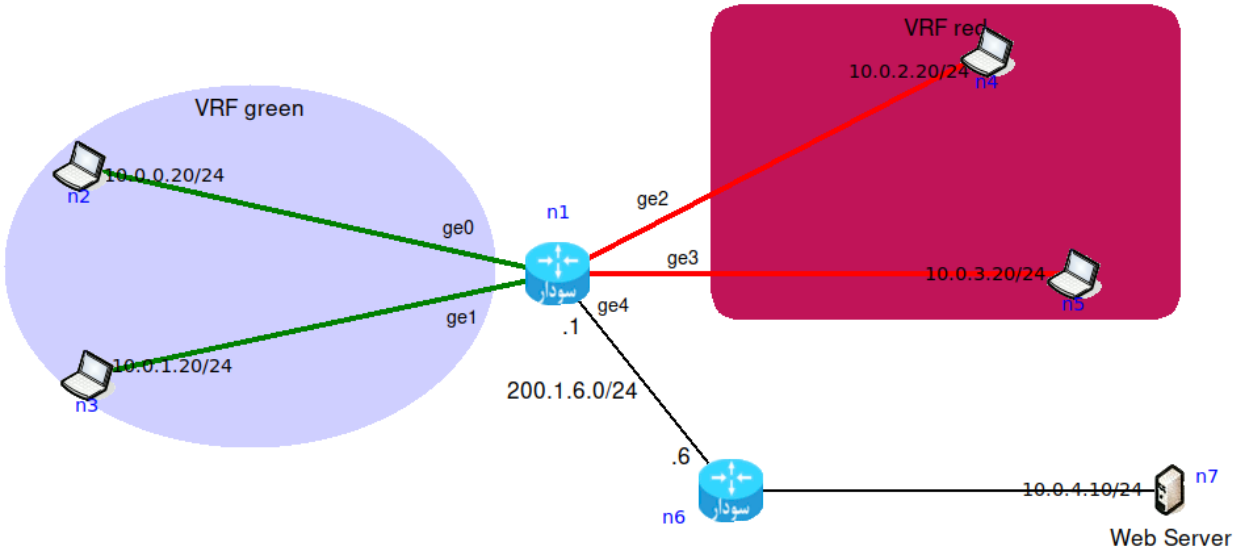
یا به شکل زیر

```
soodar3# conf t
soodar3(config) # ip route 12.128.1.0/24  200.1.15.2 vrf red
```

3.9.2 route leakage

ما با استفاده از vrf شبکه های جدا از هم در یک روتر می سازیم و ارتباطات آنها را از هم جدا می کنیم . گاهی اوقات لازم است در برخی کاربرد ها دسترسی به یک ip یا سرویس خاصی را در vrf دیگر به vrf فعلی بدهیم و این امکان را بدهیم که ارتباطی بین دو vrf جدا از هم برقرار کنیم . در این مواقع route leakage به کمک ما می آید .

فرض کنید در سناریوی زیر کاربران n2 , n3 در vrf green قرار دارند و کاربران n4 , n5 نیز در vrf red قرار دارند . این دو گروه با vrf از هم جدا شده اند و n2 , n3 با هم ارتباط دارند و n4, n5 نیز با هم . حال اگر بخواهیم دسترسی به سرور n7 برای این دو گروه فراهم سازیم باید از route-leakage استفاده کنیم زیرا در حالت معمول چون در vrf های red , green دسترسی به سرور فراهم نیست نمی توانند سرور را ببینند و از آن سرویس بگیرند . برای برقراری ارتباط بین دو vrf نیاز به route leakge داریم .



ابتدا تنظیمات مربوط به اینترفیس ها و vrf ها را به شکل زیر انجام می دهیم :

```
hostname n1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
```

(continues on next page)

(continued from previous page)

```
service password-encryption
no banner motd
!
no ntp
!
interface ge0 vrf green
 ip vrf forwarding green
 no shutdown
 ip address 10.0.0.1/24
exit
!
interface ge1 vrf green
 ip vrf forwarding green
 no shutdown
 ip address 10.0.1.1/24
exit
!
interface green vrf green
 no ip address
 no shutdown
exit
!
interface ge2 vrf red
 ip vrf forwarding red
 no shutdown
 ip address 10.0.2.1/24
exit
!
interface ge3 vrf red
 ip vrf forwarding red
 no shutdown
 ip address 10.0.3.1/24
exit
!
interface red vrf red
 no ip address
 no shutdown
exit
!
interface ge4
 no shutdown
 ip address 200.1.6.1/24
exit
!
end
```

```
hostname n6
service password-encryption
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
no ntp
!
```

(continues on next page)

(continued from previous page)

```
ip route 0.0.0.0/0 200.1.6.1
!
interface ge0
 no shutdown
 ip address 200.1.6.6/24
exit
!
interface ge1
 no shutdown
 ip address 10.0.4.1/24
exit
!
end
```

تا به اینجای کار کاربران n2,n3 با یکدیگر در vrf green ارتباط دارند و کاربران n4,n5 در vrf red با یکدیگر در ارتباط هستند اما هیچ یک از کاربران به web server که در vrf default قرار دارد دسترسی ندارند.

حال به شکل زیر route leakage ها را تعریف می کنیم:

```
n1(config)# vrf green
n1(config-vrf)# ip route 10.0.4.10/32 200.1.6.6 nexthop-vrf default
```

ما در vrf green یک route اضافه کردیم بدین شکل که شبکه 10.0.4.10/32 را gateway 200.1.6.6 که از vrf default قابل دسترسی است route کند

برای vrf red هم همین طور اضافه می کنیم

```
n1(config)#vrf red
n1(config-vrf)# ip route 10.0.4.10/32 200.1.6.6 nexthop-vrf default
```

و در نهایت برای دسترسی به vrf های red و green از vrf default نیز route های زیر در vrf default اضافه می شود :

```
n1(configf)# ip route 10.0.0.20/32 ge0 nexthop-vrf green
n1(configf)# ip route 10.0.1.20/32 ge1 nexthop-vrf green
n1(configf)# ip route 10.0.2.20/32 ge2 nexthop-vrf red
n1(configf)# ip route 10.0.3.20/32 ge3 nexthop-vrf red
```

حال باید همه کاربران n2,n3,n4,n5 به n7 دسترسی داشته باشند و در عین حال n2, n3 نتوانند با n4,n5 ارتباط بگیرند و بالعکس

pim 10.2

1.10.2 تنظیم rp

rp به شکل زیر تنظیم می گردد :

```
soodar(config)# ip pim rp 111.3.3.3 224.0.0.0/4
```

2.10.2 فعال کردن **pim**

در کلیه اینترفیس هایی که برای multicast استفاده می شوند چه اینترفیسی هایی که دیتا می فرستند چه اینترفیس هایی که درخواست ترافیک multicast می دهند، باید pim به شکل زیر فعال گردد :

```
soodar(config)# int ge0
soodar(config-if)# ip pim
```

3.10.2 فعال کردن **igmp**

در اینترفیس هایی که به کلاینت ها متصل هستند باید igmp به شکل زیر فعال شود :

```
soodar(config)# int ge3
soodar(config-if)# ip igmp
```

PBR 11.2

PBR یک ویژگی قدرتمند موجود در روترهای مدرن است که به مدیران شبکه اجازه می دهد تا کنترل دقیقی بر تصمیمات مسیریابی که در شبکه خود گرفته می شود اعمال کنند. برخلاف مسیریابی سنتی، که برای تعیین مسیر بسته‌ها صرفاً به آدرس‌های IP مقصد متکی است، PBR تصمیم‌گیری مسیریابی را بر اساس سیاست‌های تعریف شده توسط مدیران شبکه امکان‌پذیر می‌کند.

Policy-Based Routing بر اساس تعریف سیاست هایی عمل می کند که با معیارهای خاص در بسته ها مطابقت دارند، مانند source ip/port ، destination ip/port , protcol. هنگامی که یک بسته با یک سیاست تعریف شده مطابقت دارد، هدایت می شود تا یک مسیر مسیریابی سفارشی را دنبال کند و جدول مسیریابی معمولی را دور بزند. PBR را می توان در سناریوهای مختلفی استفاده کرد، از جمله:

- load balancer در چندین اتصال ISP.
- اجرای اقدامات امنیتی با هدایت ترافیک به سمت فایروال.
- تغییر مسیر ترافیک برای بهینه سازی performance شبکه.

1.11.2 پیکربندی **PBR**

برای پیکربندی یک PBR، باید سیاست ها را تعریف کنیم. route-map برای تعریف معیارهای مسیریابی مبتنی بر سیاست استفاده می شود. می توانید با دستور route-map و سپس یک نام، یک route-map ایجاد کنید. اجازه دهید در این مثال آن را PBR-Map بنامیم:

```
Router(config)# route-map PBR-Map permit 10
```

- PBR-Map: این نام route-map است. شما می توانید هر نامی را که دوست دارید انتخاب کنید.
- permit 10: این یک sequence number است که ترتیب ارزیابی را تعیین می کند. ابتدا اعداد کوچکتر ارزیابی می شوند.

اکنون در route-map ایجاد شده، معیارهای مطابقت را تعریف می کنیم. معیارهای تطبیق برای سیاست را با استفاده از دستور match در route-map مشخص کنید. این معیار مشخص می کند که سیاست با چه نوع ترافیکی مطابقت دارد.

توجه

توجه داشته باشید که فقط دستور match ip address ACL_NAME می تواند برای PBR در route-map استفاده شود.

```
Router(config-route-map)# match ip address allow_sources
```

- ip address allow_sources: این مشخص می‌کند که route-map باید با بسته‌هایی مطابقت داشته باشد که با لیست کنترل دسترسی (ACL) به نام allowed_sources مطابقت داشته باشد. مرحله بعدی، تعریف یک action است. برای مشخص کردن اقدامی که باید در هنگام match شدن ترافیک انجام شود، از دستور set در route-map استفاده کنید.

توجه

توجه داشته باشید فقط دستور set ip next-hop A.B.C.D می تواند برای PBR در route-map استفاده شود.

```
Router(config-route-map)# set ip next-hop 192.168.1.1
```

- ip next-hop 192.168.1.1: این دستور next-hop را برای بسته های منطبق تنظیم می کند.

توجه

توجه داشته باشید اگر مشخص نشده باشد، hop بعدی از VRF پیش‌فرض جستجو می‌شود. برای تغییر این رفتار، کاربر می تواند از دستور ip next-hop vrf استفاده کند.

در نهایت، شما باید route-map را در interface، subinterface یا VLAN مربوطه که می‌خواهید PBR را پیاده‌سازی کنید اعمال کنید:

ip Policy route-map PBR-Map

- PBR-Map: نام route-map را مشخص می‌کند که حاوی سیاست است. معیارها و اقداماتی که باید در مورد ترافیک اعمال شود.

توجه

توجه داشته باشید که سیاست‌ها برای ترافیک ورودی اعمال می‌شوند، بنابراین route-map باید بر روی رابط(های) داخلی تنظیم شود.

توجه

توجه بسته‌هایی که توسط دستگاه تولید می‌شوند، توسط سیاست مسیریابی نمی‌شوند.

2.11.2 مثال

در این مثال، ما PBR را برای مسیریابی ترافیک HTTP از طریق یک gateway خاص پیکربندی می کنیم در حالی که تمام ترافیک های دیگر از جدول مسیریابی معمولی استفاده می کنند.

فرضیات: روتر دارای دو رابط است: ge0 برای شبکه داخلی و ge1 برای شبکه خارجی. دروازه برای ترافیک HTTP 192.168.1.254 است و تمام ترافیک های دیگر از مسیریابی پیش فرض (192.168.1.1) پیروی می کنند. ما یک ACL تعریف می کنیم که به ترافیک TCP با آدرس IP منبع هر مشتری داخلی و یک پورت مقصد (HTTP) 80 اجازه می دهد:

```
Router(config)# ip access-list http_traffic
Router(config-nacl)# permit tcp any any eq http
```

یک route-map با sequence number 10 ایجاد شده است . که در آن ترافیک خاص با استفاده از ACL با نام http_traffic انتخاب می شود و nexht-hop آن به 192.168.1.254 تغییر می کند . دقت شود مبدا و مقصد بسته تغییری نمی کند و فقط next-hop تغییر می کند

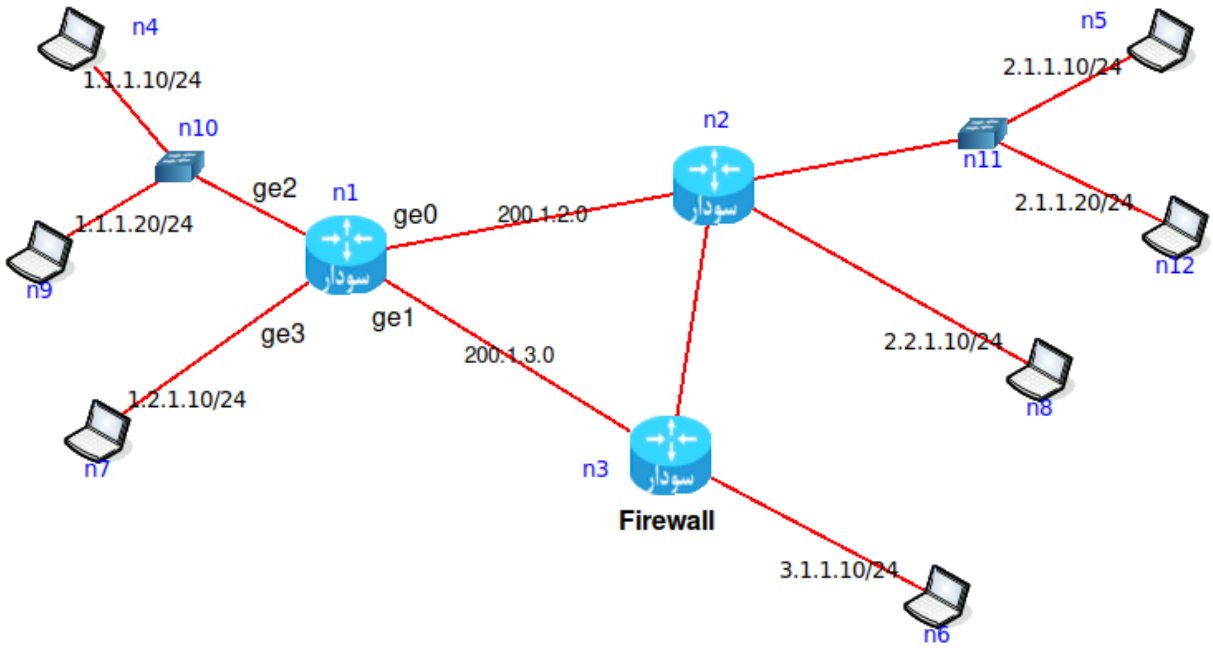
```
Router(config)# route-map redirect_http permit 10
Router(config-route-map)# match ip address http_traffic
Router(config-route-map)# set ip next-hop 192.168.1.254
```

این policy به اینترفیس ge0 اعمال می شود(و فقط در جهت inbond و روی ترافیک ورودی به اینترفیس کار می کند) . بدین معنی که ترافیک http با استفاده از route-map تغییر next-hop انتقال می یابد و بقیه ترافیک ها به صورت عادی طبق جدول routing منتقل می شوند .

```
Router(config)# interface ge0
Router(config-if)# ip policy route-map redirect_http
Router(config)# ip route 0.0.0.0/0 192.168.1.1
```

3.11.2 مثال عملی

فرض کنید شبکه ای مانند شکل زیر داریم . در روتر n1 به صورت پیش فرض تمامی ترافیک از به سمت روتر n2 هدایت می شوند و default gateway ما می باشد . حال ما می خواهیم ترافیکی که از 1.1.1.10 به مقصد 2.1.1.10 انتقال می یابد از طریق فایروال (n3) انتقال پیدا کند و رول های مد نظر روی این ترافیک اعمال شود و مثلا دسترسی به برخی پورت ها را به این کاربر ندهد .



بدین منظور ما ابتدا یک ACL تعریف می کنیم تا ترافیک مد نظر بوسیله آن انتخاب کنیم . در مرحله بعد یک route-map تعریف کرده و مشخص می کنیم اگر ترافیک در ACL تطبیق پیدا کرد next-hop آن باید به 200.1.3.3 تغییر پیدا کند و به سمت فایروال هدایت شود و در نهایت این سیاست را روی اینترفیس ge2 که سر راه 1.1.1.10 هست اعمال می کنیم :

```
n1(config)# ip access-list pbr-acl
n1(config-nacl)# permit 1.1.1.10/32 2.1.1.10/32
```

(continues on next page)

(continued from previous page)

```
n1(config-nacl)# q
n1(config)# route-map PBR_RMAP permit 10
n1(config-route-map)# match ip address pbr-acl
n1(config-route-map)# set ip next-hop 200.1.3.3
n1(config-route-map)# q
n1(config)# int ge2
n1(config-if)# ip policy route-map PBR_RMAP
```

برای حالت استفاده از vrf هم اگر فرض کنیم در n1 اینترفیس ge1 در vrf به نام green قرار دارد تنها در route-map باید vrfی که next-hop ما در آن قرار دارد را مشخص کنیم . و تنظیمات به شکل زیر خواهد بود :

```
n1(config)# ip access-list pbr-acl
n1(config-nacl)# permit 1.1.1.10/32 2.1.1.10/32
n1(config-nacl)# q
n1(config)# route-map PBR_RMAP permit 10
n1(config-route-map)# match ip address pbr-acl
n1(config-route-map)# set ip next-hop vrf green 200.1.3.3
n1(config-route-map)# q
n1(config)# int ge2
n1(config-if)# ip policy route-map PBR_RMAP
```

فصل 3

NAT

1.3 معرفی NAT

هدف اولیه NAT بدین صورت تعریف شده است که کاربران داخلی سازمان با آدرس غیر اینترنتی صرفا با تعداد محدودی آدرس اینترنتی بتوانند با شبکه اینترنت ارتباط برقرار نمایند بدون اینکه مجبور به استفاده از آدرس های Public باشند. بدین صورت که کاربران شبکه از آدرس های محدوده اختصاصی برای ارتباطات داخلی خود استفاده می کنند و در صورتی که بخواهند با اینترنت ارتباط برقرار کنند، تعداد زیادی کاربر صرفا از یک یا چند آدرس محدود اینترنتی به صورت اشتراکی برای اتصال به اینترنت بهره می برند. بدین ترتیب محدودیت تعداد آدرس IPV4 از بین خواهد رفت.

امروزه NAT دیگر به همین یک کاربرد منتهی نمی شود و هدف آن نیز صرفا رفع محدودیت تعداد آدرس نیست. اینکه کاربران اینترنتی بتوانند با سرور سازمان، که آدرس غیر اینترنتی به آن تخصیص داده شده است، ارتباط برقرار نمایند از دیگر کاربردهای NAT است که هدف از قرار دادن آدرس غیر اینترنتی روی سرور ایجاد امنیت بیشتر برای سرور است. اینکه به کاربران و سرورها آدرس اینترنتی اختصاص داده نمی شود، بدین معنی است که از روی اینترنت دیده نخواهند شد و این یک مزیت امنیتی است.

2.3 Dynamic NAT

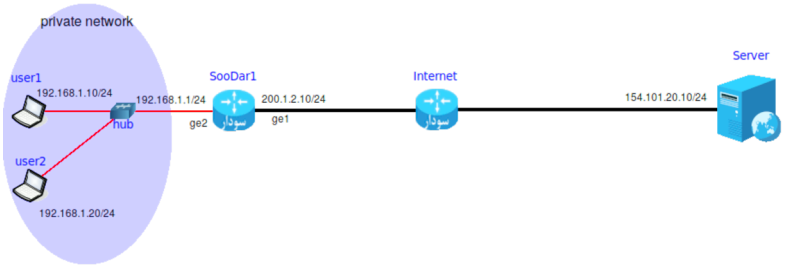
در Dynamic NAT هر بسته ای که از شبکه inside دریافت شود و آدرس مبدا آن در رنج شبکه inside باشد و مقصد آن باعث شود که در اینترفیس outside ارسال شود مبدا بسته به یک ip از pool تعریف شده تغییر می کند . برای هر ترجمه مبدا آدرس یک نشست جدید ایجاد می شود و state آن نگه داشته می شود بنابراین بسته هایی که از شبکه global دریافت می شوند در نشست مربوط به خود قرار گرفته و طبق آن ترجمه می شوند.

```
soodar(config)# ip nat pool PNAT44 A.B.C.D [ A.B.C.D ]
```

می توانید یک ip یا در صورت لزوم با استفاده از ip دوم ، یک رنج ip را به عنوان pool اضافه کنید نام رول اضافه شده هم در این مثال PNAT44 تنظیم شده است .

برای توضیح بهتر در مثال زیر یک نمونه Dynamic NAT را تنظیم می کنیم:

در شکل زیر دو کاربر مختلف با آدرس های غیر اینترنتی 192.168.1.10 و 192.168.10.20 قصد دارند به سروری که در اینترنت قرار دارد وصل شوند . با استفاده از Dynamic NAT در روتر سودار این ارتباط را فراهم می کنیم :



اینترفیس ge1 که در سمت public و اینترنت قرار دارد به عنوان outside و اینترفیس ge2 که در سمت private قرار دارد را به عنوان inside انتخاب می کنیم :

```
soodar1(config)# int ge1
soodar1(config-if)#  ip nat outside
soodar1(config)# int ge2
soodar1(config-if)#  ip nat inside
```

سپس یک نام برای Nat وارد کرده و یک pool که یک رنج ip می باشد را در nat تنظیم می کنیم :

```
soodar1(config)# ip nat pool POOL1 200.1.2.10 200.1.2.10
```

در آخرین مرحله با نعریف یک ACL مشخص می کنیم که چه ترافیک هایی باید NAT شوند (البته در این جا همه ترافیک را NAT می کنیم) :

```
soodar1(config)# ip access-list nat-acl
soodar1(config-acl)# permit any any
soodar1(config-acl)# q
soodar1(config)# ip nat inside source list nat-acl pool POOL1
```

1.2.3 استفاده از interface به جای تعریف pool

در اینجا می توانیم تنظیم را به شکل دیگر نیز انجام دهیم و به جای تعریف pool از interface استفاده کنیم تنظیمات به شکل زیر تغییر خواهد کرد :

```
soodar1(config)# int ge1
soodar1(config-if)#  ip nat outside
soodar1(config)# int ge2
soodar1(config-if)#  ip nat inside
soodar1(config-if)# q
soodar1(config)# ip access-list nat-acl
soodar1(config-acl)# permit any any
soodar1(config-acl)# q
soodar1(config)# ip nat inside source list nat-acl interface ge1
```

توجه

در حالت استفاده از interface به جای pool از اولین **IP** روی اینترفیس برای pool استفاده می شود .

3.3 بررسی ارتباط NAT

اکنون تنظیمات NAT به پایان رسیده است و کاربران user1,user2 می توانند با ip محلی خود به سرور وصل شوند . البته در سمت اینترنت هیچ دسترسی به کاربران وجود ندارد

4.3 حذف NAT

برای حذف NAT می توانید به شکل زیر عمل کنید :

```
soodar1(config)# no ip nat pool nat1
soodar1(config)# int ge1
soodar1(config-if)# no  ip nat outside
soodar1(config)# int ge2
soodar1(config-if)#  no ip nat inside
```

5.3 فعال کردن log های NAT

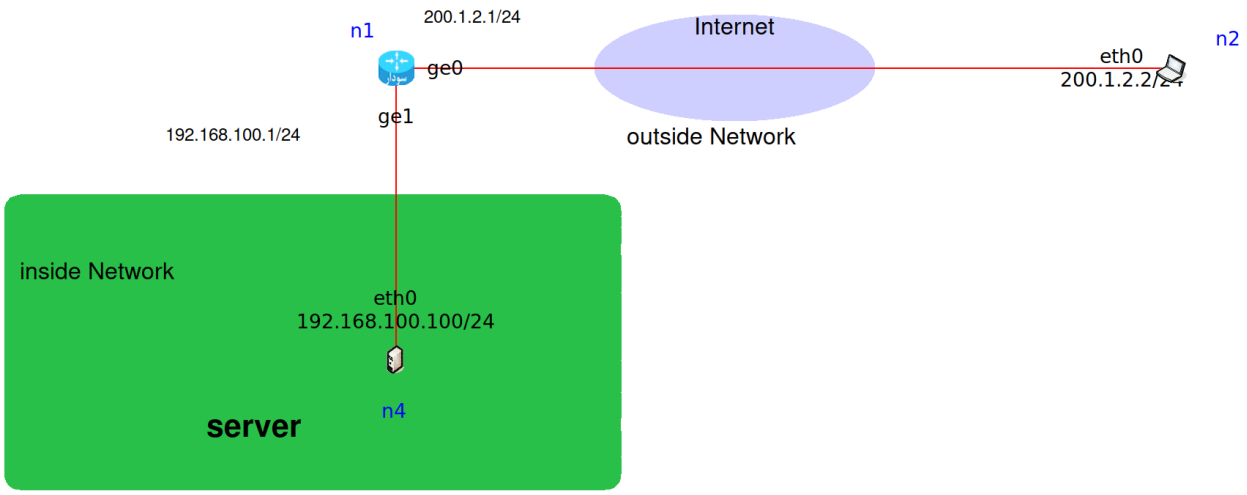
با دستور زیر می توانید Log های مربوط به nat را فعال کنید:

```
soodar1# debug nat44 event
```

6.3 STATIC NAT

در حالت static شما برای تک تک ip هایی که باید NAT شوند باید دستی rule اضافه کنید و دیگر به صورت خودکار ip ها توسط NAT اختصاص و تغییر نمی یابد . به طور مثال فرض کنید در سناریوی زیر قصد داریم تمامی ترافیکی که از سرور n4 به سمت بیرون می رود را تغییر مبدا انجام دهیم و با ip اینترفیس ge0 در روتر n1 یعنی 200.1.2.1 ارسال شوند .

برای اینکار ابتدا همانند حالت داینامیک سمت شبکه اینترنت را به عنوان nat outside و سمت شبکه local را به عنوان nat inside مشخص می کنیم :




```
n1# conf t
n1(config)# int ge0
n1(config-if)# ip nat outside
n1(config-if)# q
n1(config)# int ge1
n1(config-if)# ip nat inside
n1(config-if)# q
n1(config)# ip nat inside source static 192.168.100.100 200.1.2.1
```

با تنظیم بالا NAT به شکل زیر انجام می گیرد:

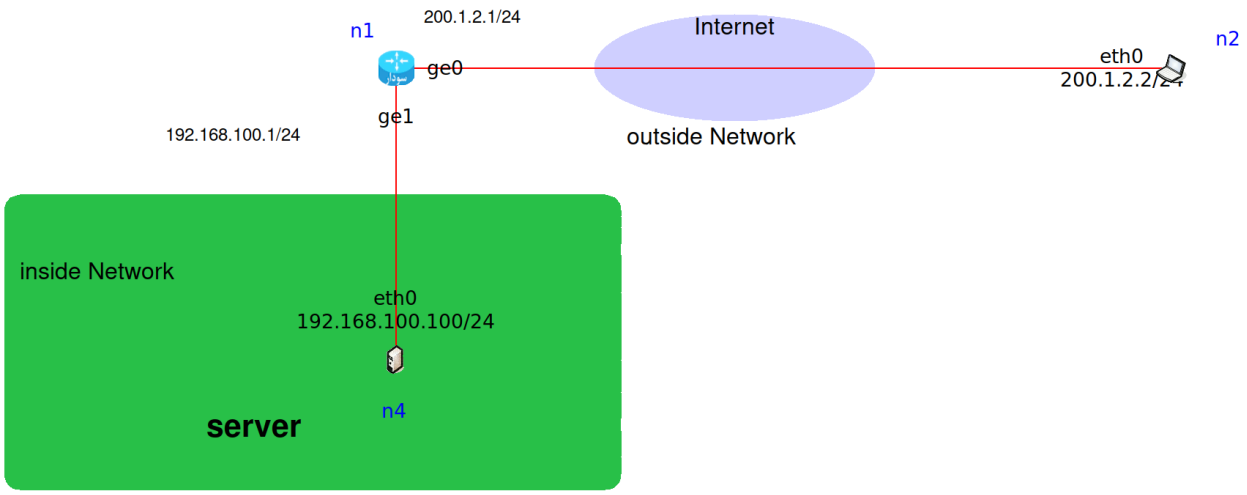
- اگر مبدا بسته با شبکه inside در nat تعریف شده مطابقت داشته باشد و از شبکه inside دریافت شود مبدا بسته با ip جدید (200.1.2.1) تعریف شده جایگزین خواهد شد .
  با این تنظیم مبدا packet خروجی از اینترفیس ge0 که مبدا آن 192.168.100.100 باشد را به ip جدید 200.1.2.1 ترجمه می کند و مبدا بسته را این ip جدید قرار می دهد . (source nat)
- اگر مقصد بسته دریافتی برابر با ip اینترفیس outside باشد (200.1.2.1) و در اینترفیس outside دریافت شود مقصد شبکه به آدرس شبکه inside (192.168.100.100) ترجمه می شود .
  با این تنظیم مقصد packet ورودی به اینترفیس ge0 که destinatoin آن 200.1.2.1 باشد به 192.168.100.100 تغییر خواهد کرد . (destination nat)

Protocol NAT 7.3

در این حالت همان static nat مثال قبل استفاده می شود با این تفاوت که کل ترافیک NAT نمی شود بلکه دقیق مشخص می کنیم چه protocol و چه port ی را باید NAT کنیم .
در این حالت تنظیمات به شکل زیر خواهد بود :

```
ip nat inside source static <tcp|udp> A.B.C.D (1-65535) A.B.C.D (1-65535)
```

این تنظیم یک ورودی به جدول ایستا nat اضافه می کند که اولین ip و پورت مربوط به شکل داخلی (inside) و دومین ip و پورت مربوط به شبکه خارجی (output) می باشد. نحوه تنظیم source nat و destinatio nat همانند بخش static nat می باشد .



مثال 1 :

```
n1# conf t
n1(config)# int ge0
n1(config-if)# ip nat outside
n1(config-if)# q
n1(config)# int ge1
n1(config-if)# ip nat inside
n1(config-if)# q
n1(config)# ip nat inside source static tcp 192.168.100.100 443 200.1.2.1 443
```

بدین ترتیب ترافیک https دریافتی در اینترفیس ge0 با تغییر مقصد به 192.168.100.100 به شبکه inside ارسال می شود و همچنین ترافیک خروجی https از اینترفیس ge0 که مبدا 192.168.100.100 داشته باشد با تغییر مبدا به 200.1.2.1 ارسال خواهد شد .

فصل 4

SLA

SLA 1.4

SLA یک ویژگی است که برای اندازه گیری عملکرد شبکه و تأیید سطوح خدمات شبکه استفاده می شود. این به مدیران شبکه اجازه می دهد تا ترافیک شبکه را شبیه سازی کرده و عملکرد دستگاه ها را اندازه گیری کنند. IP SLA می تواند برای نظارت بر طیف وسیعی از پارامترهای شبکه مانند از دست دادن بسته، تأخیر، jitter و در دسترس بودن استفاده شود. همچنین می‌توان از آن برای راه‌اندازی رویدادهای گزارش‌گیری، مانند failover، زمانی که از آستانه عملکرد فراتر رفت، استفاده کرد. IP SLA می تواند به مدیران شبکه کمک کند تا مشکلات شبکه را شناسایی و عیب یابی کنند و از برآورده شدن سطح خدمات شبکه اطمینان حاصل کنند.

در حال حاضر دو نوع عملکرد SLA داریم :

1. icmp-echo
2. icmp-jitter

icmp echo 1.1.4

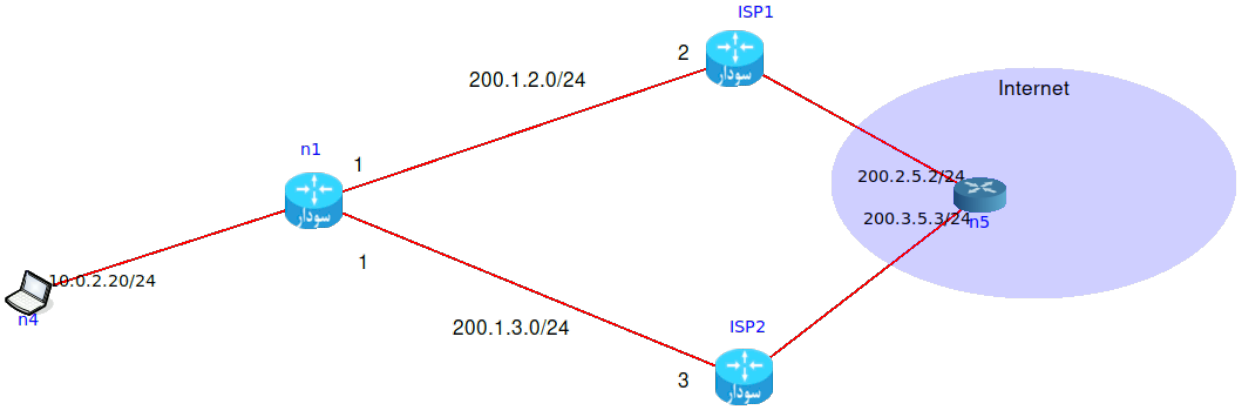
در این حالت بسته های icmp به مقصد مشخص ارسال می شود و rtt بین دستگاه مبدا و مقصد اندازه گیری می شود .

پارامترهای icmp-echo

- **destination** : مقصد بسته های icmp را مشخص می کند
- **source** : مبدا بسته های icmp را مشخص می کند . در واقع مشخص مس کند بسته ها با کدام ip یک از ip های تنظیم شده در اینترفیس ارسال شود
- **frequency** : بازه زمانی (interval) ارسال بسته های icmp را مشخص می کند
- **timeout**: مشخص می کند که حد اکثرچقدر برای دریافت پاسخ بسته icmp صبر کند
- **threshold**: مقدار آستانه بالا را برای rtt مشخص می کند . در مانیتورینگ و بخش reaction استفاده می شود
- **vrf**: مشخص می کند که این sla در چه vrf ی باید اجرا شود
- **request-data-size**: مقدار payload بسته icmp را مشخص می کند

مثال برای icmp echo

فرض کنید در سناریوی زیر در نود n1 قصد داریم sla از نود icmp-echo اجرا کنیم . در این جا ما از دو ISP سرویس اینترنت دریافت می کنیم . به صورت پیش فرض از ISP1 استفاده می کنیم چون سرویس بهتری به ما می دهد . حال اگر مقدار rtt بسته ها از مقدار بیشتر شد از ISP2 برای دسترسی به اینترنت استفاده می کنیم . تنظیمات در روتر n1 به شکل زیر خواهد بود :



اضافه کردن sla

```
n1(config)# ip sla 1
n1(config-ip-sla)# icmp-echo 200.1.2.2
n1(config-ip-sla-echo)# frequency 30  شوند می ارسال بار یک ثانیه 30 هر ها بسته
```

تعریف reaction

با دستور زیر مشخص می کنیم که در صورتی که حتی یک بسته مقدار rttش از 500 ms بیشتر شد لاگ کن و اگر track ی تعریف کرده باشیم که از این reaction استفاده کند وضعیت track با توجه به این reaction تغییر می کند .

```
n1(config)# ip sla reaction-configuration 1 react rtt action-type logOnly threshold-type immediate threshold-value 500 50
```

schedule کردن sla

بعد از اضافه کردن sla باید مشخص کنیم که این sla چه زمان هایی باید اجرا شود . در این مثال ما می خواهیم sla از همین لحظه شروع به کار کند و برای همیشه فعال باشد :

```
n1(config)# ip sla schedule 1 start-time now life forever
informational-ZEBRA: SLA 1 running state changed: Running دهد می نشان sla اجرا شده است
informational-ZEBRA: SLA 1 state changed: Ok دهد می نشان sla up شده است
```

مشاهده تنظیمات اعمال شده

```
n1# sh ip sla configuration 1
Entry number: 1
 Operation timeout (milliseconds): 5000
 Type of operation to perform: icmp-echo
 Target address/Source address: 200.1.2.2/0.0.0.0
 Request size (ARR data portion): 28
 VRF name:
 Schedule:
  Operation frequency (seconds): 30
  Next Scheduled Start Time: Start Time already passed
  Life (seconds): forever
  Recurring (Starting Everyday): FALSE
 Threshold (milliseconds): 5000
n1#
```

مشاهده وضعیت **sla**

```
n1(config)# do sh ip sla statistics 1
IPSLA Operation id: 1
Type of operation: icmp-echo
  Latest RTT: 12 milliseconds
Latest operation start time: Sat Jun 10 14:02:23 2023          است شده اجرا که باری آخرین
Latest successfull operation time: Sat Jun 10 14:02:23 2023  است شده اجرا موفقیت با که باری آخرین
Latest failed operation time: N/A                              است بوده ناموفق که باری آخرین
Latest operation return code: OK                               است شده اجرا که باری آخرین وضعیت
Number of successes: 68                                        است بوده آمیز موفقیت که دفعاتی تعداد
Number of failures: 0                                          است نبوده آمیز موفقیت که دفعاتی تعداد
Operation time to live: Forever                                مانده باقی عمر
n1(config)#
```

اضافه کردن **track**

یک track تعریف می کنیم که برای آن reaction rtt در sla شماره 1 مشخص شده است . یعنی اگر مقدار rtt در sla 1 کمتر از 500 ms باشد track 1 فعال (up) خواهد بود در غیر اینصورت track down خواهد شد .

```
n1(config)# track 1 ip sla 1 reaction rtt
informational-ZEBRA: Track 1 came up
```

سپس با استفاده از track یک route default اضافه می کنیم که این route زمانی در سیستم نصب می شود که track فعال باشد . ما یک route دیگر نیز با مقدار distance برابر با 150 هم اضافه کرده ایم که چون مقدار distance آن بیشتر است نصب نخواهد شد و فقط زمانی که track غیر فعال (down) باشد این route نصب می شود :

```
n1(config)# ip route 0.0.0.0/0 200.1.2.2 track 1
n1(config)# ip route 0.0.0.0/0 200.1.3.3 150
```

(continues on next page)

(continued from previous page)

```
n1(config)# do sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

S>* 0.0.0.0/0 [1/0] via 200.1.2.2, ge0, weight 1, 00:00:14
C>* 10.0.2.0/24 is directly connected, ge2, 01:03:46
C>* 200.1.2.0/24 is directly connected, ge0, 01:04:45
C>* 200.1.3.0/24 is directly connected, ge1, 01:04:02
n1(config)#
```

به شکل زیر نیز می توان وضعیت track را مشاهده کرد:

```
n1# sh track 1
Track 1
  IP SLA 1 reaction
  Reaction is Up
    1 change[s], last change 00:07:59
  Latest operation return code: OK
  Tracked by:
    Static IP Routing
n1#
```

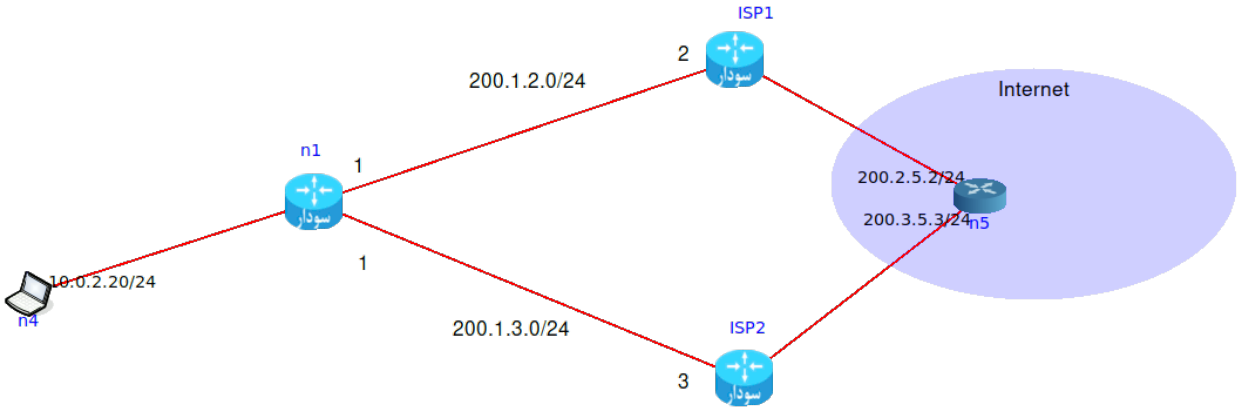
icmp jitter 2.1.4

در این حالت ترافیک icmp به مقصد مشخص ارسال می شود و مقدار jitter , packet loss برای آن محاسبه می گردد.

پارامترهای icmp-jitter

- **destination** : مقصد بسته های icmp را مشخص می کند
- **source** : مبدا بسته های icmp را مشخص می کند . در واقع مشخص مس کند بسته ها با کدام ip یک از ip های تنظیم شده در اینترفیس ارسال شود
- **frequency** : بازه زمانی (interval) ارسال بسته های icmp را مشخص می کند
- **timeout**: مشخص می کند که حد اکثرچقدر برای دریافت پاسخ بسته icmp صبر کند
- **threshold**: مقدار آستانه بالا را برای rtt مشخص می کند . در مانیتورینگ و بخش reaction استفاده می شود
- **vrf**: مشخص می کند که این sla در چه vrf ی باید اجرا شود
- **num-packets**: مشخص می کند در هر بار که sla اجرا می شود چه تعداد بسته ارسال شود
- **interval**: بازه زمانی بین دو بسته icmp متوالی که به سمت مقصد ارسال می شوند را مشخص می کند
- **percentile**: مقدار درصد بسته هایی که rtt کمتری دارد را برای محاسبات آماری مشخص می کند برای مثال وقتی مقدار 95 تنظیم شود ، 5 درصدی که بیشترین مقدار rtt را دارند و به نوعی داده پرت به حساب می آیند در محاسبات لحاظ نشوند و فقط 95 درصد از داده ها جهت تحلیل و تصمیم گیری استفاده شوند

مثال icmp jitter



jitterپیاده سازی کنیم :

```
n1(config)# ip sla 2
n1(config-ip-sla)# icmp-jitter 200.1.2.2
n1(config-ip-sla-echo)# frequency 30
n1(config-ip-sla-icmpjitter)# percentile jitterAvg 90
n1(config-ip-sla-icmpjitter)#
n1(config)# ip sla reaction-configuration 2 react jitterAvg action-type logOnly threshold-type average 5 threshold-value 100 10
n1(config)# ip sla schedule 2 start-time now life 7200
n1(config)# track 1 ip sla 2 reaction jitterAvg
n1(config)# ip route 0.0.0.0/0 200.1.2.2 track 1
n1(config)# ip route 0.0.0.0/0 200.1.3.3 150
```

ما در این مثال یک sla با شماره 2 از نوع jitter به مقصد 200.1.2.2 تعریف کرده ایم که هر 30 ثانیه یک بار اجرا می شود . همچنین 10 درصدی که بیشترین مقدار jitter را دارد از داده ها حذف نمو ده ایم . در ادامه یک reaction اضافه کردیم که از نوع jitterAvg است و اگر میانگین 5 مقدار jitter از 100 بیشتر شود لاگ می کند . پس از آن یک track تعریف کردیم که به این reaction واکنش نشان می دهد و در صورت up بودن آن یک route اضافه می کند

با این تنظیم به صورت پیش فرض اینترنت از ISP1 در دسترس خواهد بود و در صورتی که میانگین jitter در 5 بار اجرا شدن sla بیشتر از 100ms شود اینترنت از ISP2 در دسترس خواهد بود .

```
n1# sh ip sla statistics 2
IPSLA Operation id: 2
Type of operation: icmp-jitter
  Latest RTT: 10 milliseconds
Latest operation start time: Sun Jun 11 09:33:34 2023
Latest successfull operation time: Sun Jun 11 09:33:34 2023
Latest failed operation time: N/A
Latest failed operation error: Timed out
Latest operation return code: OK
RTT Values:
  Number of RTT:10        RTT Min/Avg/Max: 10/12/15 milliseconds
Jitter time:
  Number of Jitter Samples: 9
  Jitter Min/Avg/Max: 0/2/4 milliseconds
Percentile Jitter time:
  Number of Percentile Jitter Samples (90%): 8
  Percentile Jitter Min/Avg/Max: 0/2/4 milliseconds
Over Threshold:
```

(continues on next page)

(continued from previous page)

```
  Number Of RTT Over Threshold: 0
Out of Sequence: 0
Packet Loss: 0
Number of successes: 8
Number of failures: 0
Operation time to live: 01:56:26
```

با دستور زیر می توان جزییات بیشتری را مشاهده کرد :

```
n1(config)# do sh ip sla statistics 2 details
IPSLA Operation id: 2
Type of operation: icmp-jitter
  Latest RTT: 11 milliseconds
Latest operation start time: Sun Jun 11 09:35:04 2023
Latest successfull operation time: Sun Jun 11 09:35:04 2023
Latest failed operation time: N/A
Latest failed operation error: Timed out
Latest operation return code: OK
RTT Values:
  Number of RTT:10        RTT Min/Avg/Max: 7/12/15 milliseconds
Jitter time:
  Number of Jitter Samples: 9
  Jitter Min/Avg/Max: 0/2/4 milliseconds
  Positive Jitter Num/Min/Avg/Max: 6/0/1/4 milliseconds
  Negative Jitter Num/Min/Avg/Max: 3/3/3/4 milliseconds
Percentile Jitter time:
  Number of Percentile Jitter Samples (90%): 8
  Percentile Jitter Min/Avg/Max: 0/2/4 milliseconds
Over Threshold:
  Number Of RTT Over Threshold: 0
Out of Sequence: 0
Packet Loss: 0
Number of successes: 11
Number of failures: 0
Operation time to live: 01:54:56
```

3.1.4 غیر فعال کردن sla

با دستور زیر می توانید schedule را غیر فعال کنید :

```
n1(config)# no ip sla schedule 1
```

4.1.4 فعال کردن لاگ ها

```
n1(config)# debug ip sla event
```

5.1.4 مشاهده لاگ ها

```
n1# show log soosla
```

فصل 5

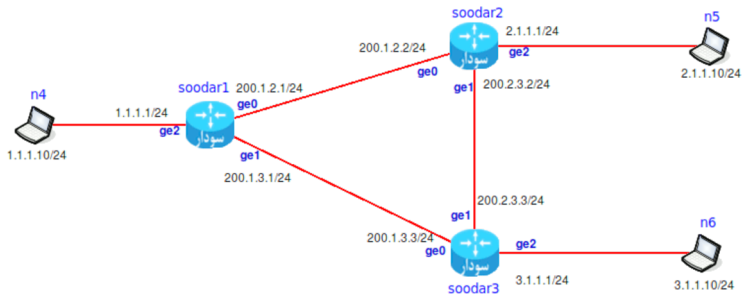
Qos

1.5 مفهوم QOS

QOS یا Quality Of Service ویژگی است که با آن می توان یک سری مدیریت ها و محدودیت هایی روی ترافیک عبوری از روتر اعمال کرد به طوری که اگر ترافیکی در شرایط از قبل مشخص شده تطابق پیدا کرد سیاست تعیین شده مربوط به آن روی آن ترافیک اعمال گردد . این سیاست ها توسط مدیر شبکه و با توجه به نیاز های شبکه تعیین می شود .در روتر سودار شما می توانید با استفاده از QOS محدودیت در پهنای بند استفاده شده برای کاربران و یاشبکه های خاص اعمال کنید .

2.5 آموزش راه اندازی QOS در سودار

مثال : فرض کنید قصد داریم که ترافیک 2.1.1.10 به مقصد 1.1.1.10 را به پهنای باند 2Mb/s محدود کنیم :



ابتدا باید یک class-map ایجاد کنیم که در آن مشخص شود چه ترافیک هایی را می خواهیم محدود کنیم و سپس سیاستی را که روی آن ترافیک قرار است اعمال شود مشخص کرده و در نهایت آن را در اینترفیس مورد نظر اعمال کنیم .

3.5 ایجاد class-map

با class-map ترافیک هدف خود را انتخاب می کنیم . می توانید از destination address یا source address یا access-list و یا ترکیبی از همه این ها استفاده کنید .در اینجا ما از destination address , source address استفاده کرده ایم :

```
soodar2(config)# class-map cmap
soodar2(config-cmap)# match destination-address 1.1.1.10/32
soodar2(config-cmap)# match source-address 2.1.1.10/32
```

نکته

- دقت شود که class-map را می توان در دو حالت match-all و match-any ایجاد کرد . در حالت اول اگر تمامی رول های اضافه شده در class-map به طور همزمان در بسته ورودی تطبیق پیدا کند سیاست مورد نظر در آن اعمال می شود اما در حالت دوم کافی است یکی از رول های تعریف شده در class-map در بسته تطبیق پیدا کند match) شود) تا سیاست مورد نظر در آن اعمال گردد . حالت پیش فرض match-all می باشد .
- اگر در یک policy ما دو یا چند کلاس داشته باشیم که رول های آن ها همپوشانی باشد policer آخرین کلاسی که ترافیک در آن match شود اعمال می شود

4.5 ایجاد policy

در این بخش باید یک policy-map ایجاد کنیم در آن class-map که در مرحله قبل ایجاد شده را ، اضافه کنیم .سپس یک policy ایجاد کنیم که در آن حداکثر پهنای باند را مشخص می کنیم .

```
soodar2(config)# policy-map pmap
soodar2(config-pmap)# class cmap
soodar2(config-pmap-c)# police 2M conform-action transmit exceed-action drop
```

در این دستور مشخص می شود که ترافیک تا 2M بدون مشکل انتقال یابد.اما اجازه نمی دهد که ترافیک بیشتر از 2M شود و ترافیک مازاد drop می شود.

5.5 اعمال policy در اینترفیس

```
soodar2(config-pmap)# int ge2
soodar2(config-if)# service-policy pmap in
```

6.5 بررسی عملکرد QOS

برای نمونه ftpserver را در 2.1.1.10 اجرا کرده و فایل را از 1.1.1.10 دانلود می‌کنیم .

1. باید سرعت دریافت فایل حدود 2M/s باشد .
2. با حذف policy سرعت دانلود فایل به حالت عادی برگردد با دستور :

```
soodar2(config-if)# service-policy pmap in
```

7.5 مشاهده policy ها در روتر

با استفاده از دستور زیر لیست تمامی policy های موجود در روتر را می توانید مشا هده کنید . اگر نام یک policy خاص را در انتهای دستور وارد کنید می توانید تنها همان policy را مشادهد کنید :

```
soodar2# sh policy-map
Policy Map pamp
    Class cmap
        Police CIR 2097152 (bps) CB 524288 (byte) EB 734003 (byte)
        Conform Action : Transmit
        Exceed Action : Drop
```

8.5 فعال کردن log های qos

با دستور زیر می توانید Log های مربوط به qos را فعال کنید:

```
soodar1# debug qos event
```

فصل 6

ACL

ACL 1.6

2.6 آشنایی با مفهوم ACL

به بیانی دیگر ACLها همانند یک صافی عمل کرده و تنها پکت هایی از این صافی ها عبور میکنند که دارای یکسری شرایط خاص باشند.

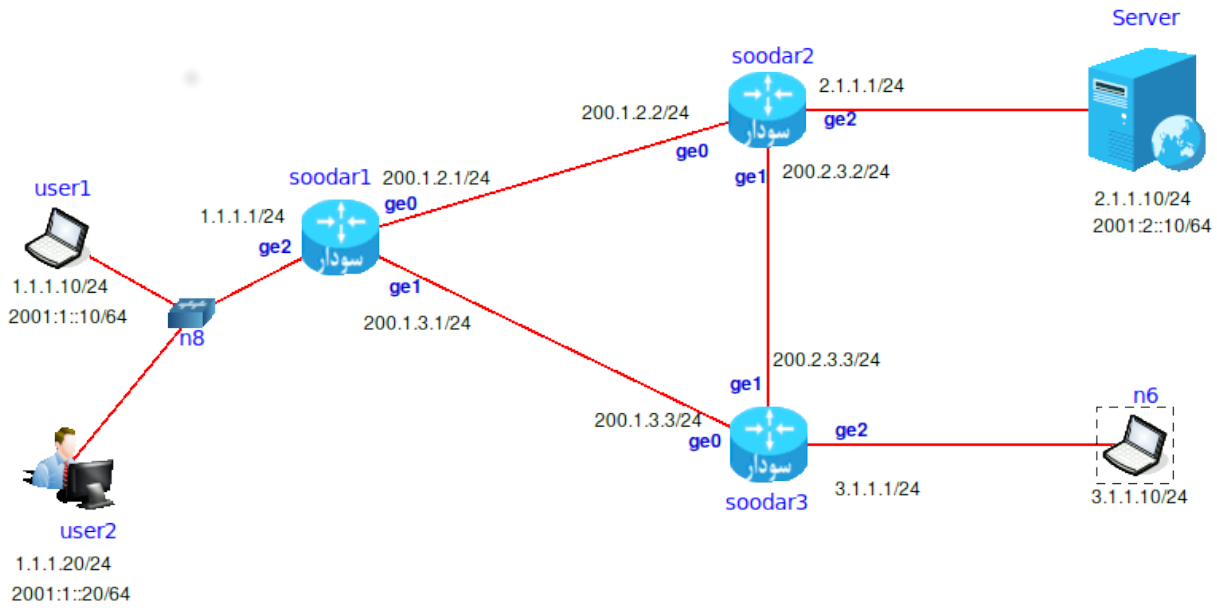
در هنگامی که پکت ها با ACLها مقایسه میشوند، سه موضوع توسط روتر بررسی می شود:

1. پکت ها با تک تک رول ها به صورت ترتیبی مقایسه میشوند.

2. پکت ها تا زمانی با رول ها مقایسه میشوند که با یکی از آنها مطابقت پیدا کند. توجه داشته باشید که به محض برقراری شرط، عملیات مقایسه با سایر رول ها متوقف خواهد شد.

3. در انتهای هر ACL یک دستور Drop به صورت انتزاعی قرار دارد و باعث می شود تا اگر که هیچ یک از رول های ACL با پکت مطابقت پیدا نکند، پکت Drop شود.

3.6 پیکربندی ACL



فرض کنید قصد داریم دسترسی به web را برای user1 مسدود کنیم : به ترتیب زیر ترافیک tcp مربوط به پورت 80 را از مبدا 1.1.1.10 به مقصد web server مسدود می کنیم :

```
soodar1(config)# ip access-list test
soodar1(config-nacl)# deny tcp 1.1.1.10/32 2.1.1.10/24  eq 80
soodar1(config-nacl)# permit any any
```

4.6 اعمال ACL در اینترفیس

هنگامی که یک ACL در روتر تعریف شد، هیچ فیلتری برروی پکتها اعمال نمی شود تا زمانی که ACLها برروی پورت های روتر اعمال شود و اینکه مشخص گردد که ACLها برروی ترافیک های خروجی و یا ورودی اعمال گردند. پس با توجه به این موضوع مسیر جریان های هر اینترفیس، به دو دسته تقسیم می شود که شامل:

ترافیک های لبه ورودی (یا Inbound): در این حالت روتر قبل از اینکه اقدام به مسیریابی پکت ها کند، شرط های ACL را مورد بررسی قرار می دهد و در صورتی که شرایط مشخص شده در ACL برقرار گردید، عملیات مسیریابی پکت ها صورت میپذیرد.

ترافیک های لبه خروجی (یا Outbound): در این حالت روتر بعد از اینکه عملیات مربوط به مسیریابی پکت را انجام داد، قبل از اینکه پکت به سمت خارج هدایت شود، شرط های ACL را مورد بررسی قرار میدهد و در صورتی که شرایط مشخص شده در ACL برقرار گردید، عملیات مسیریابی پکت ها صورت میپذیرد.

حال با اعمال ACl در اینترفیس در جهت inbound محدودیت مورد نظر برای user1 اعمال می شود و کلیه بسته های tcp با پورت 80 را به مقصد server حذف می کند .

```
soodar1(config)# int ge2
soodar1(config-if)# ip access-group test in
```

1.4.6 مشاهده ACL

1. مشاهده یک ACL خاص

با دستور زیر می توان رول های اضافه شده در هر acl را مشاهده کرد :

```
soodar1(config)# show ip access-list test
soodar1# show ip access-list test
IP access list test
    10 deny   icmp 1.1.1.10/32 le 65535 2.1.1.0/24 le 65535
    20 permit any any
```

2. مشاهده تمامی ACL ها

اگر نام ACL را مشخص نکنید تمام ACL های موجود (IPv6 یا IPv4) در روتر را می توانید مشاهده کنید که البته ما در اینجا فقط یک ACL داریم و خروجی هر دو حالت نمایش تفاوتی ندارد:

```
soodar1# show ip access-list
IP access list test
    10 deny   icmp 1.1.1.10/32 le 65535 2.1.1.0/24 le 65535
    20 permit any any
```

5.6 بررسی عملکرد ACL

شرایط زیر پس از اعمال ACL در اینترفیس باید در شبکه برقرار باشد :

1. نباید user1 به web server وصل شود .
2. کاربر user2 باید بتواند به web server وصل شود .
3. کاربر user1 باید بتواند web server را ping کند .

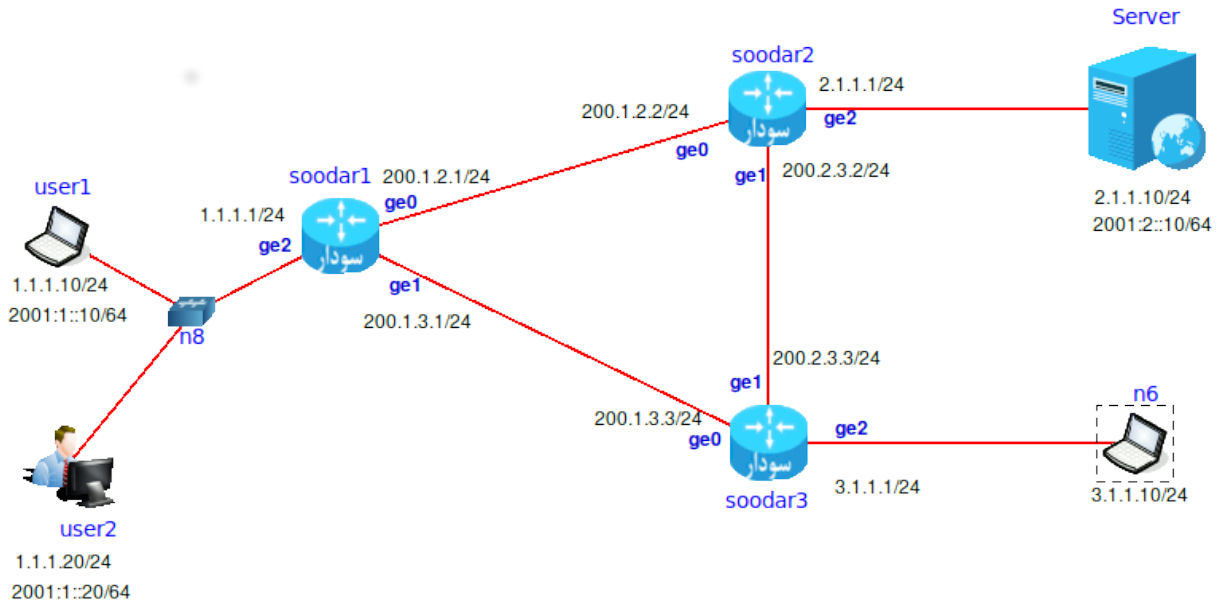
6.6 حذف ACL

حال یک acl دیگر را امتحان می‌کنیم ابتدا acl قبلی را حذف می‌کنیم :

```
soodar1(config)# no ip access-list test
```

7.6 مثال محدودیت ssh

در acl جدید ارتباط ssh به web servre را می بندیم و فقط به user2 اجازه می دهیم که بتواند ssh بزند :



توجه

فرض شده است که ssh روی پورت پیش فرض 22 فعال است

```
soodar1(config)# ip access-list SSH-FILTER
soodar1(config-nacl)# 1 permit tcp 1.1.1.20/32  2.1.1.10/32 eq 22
soodar1(config-nacl)# 2 deny tcp 1.1.1.10/32  2.1.1.10/32 eq 22
soodar1(config-nacl)# permit any any
soodar1(config-nacl)# do sh ip access-list SSH-FILTER
IP access list SSH-FILTER
    seq 1 permit tcp 1.1.1.20/32 2.1.1.10/32 eq 22
    seq 2 deny tcp 1.1.1.10/32 2.1.1.10/32 eq 22
    seq 7 permit any any

soodar1(config)# int ge2
soodar1(config-if)# ip access-group SSH-FILTER
```

IPv6 ACl 8.6

مانند دیگر ویژگی های روتر سودار IPv6 نیز در ACL پشتیبانی می شود و شما می توانید ترافیک شبکه IPv6 خود را نیز با استفاده از ACL ها مدیریت کنید . در مثال زیر نمونه ای از تنظیم یک ACL در IPv6 را مشاهده می کنید :

```
soodar1(config)# ipv6 access-list testipv6
soodar1(config-ipv6-acl)# 1 deny tcp any 2001:1:2::2/64 eq 80
soodar1(config-ipv6-acl)# permit any any
soodar1(config-ipv6-acl)# do sh ipv6  access-list
IPv6  access list testipv6
   seq 1 deny   tcp any 2001:1:2::/64 eq 80
   seq 6 permit any any
```

ACL 9.6 با نام سرویس

در روتر سودار برای راحتی کار ادمین برخی از سرویس های پرکاربرد و به عبارت دیگر پروتکل های پرکاربرد را می توان به راحتی و تنها با استفاده از نام پروتکل در ACL استفاده کرد و ترافیک ها مربوط به آن پروتکل را برای کاربر و یا شبکه خاصی محدود کرد .

برای مثال اگر بخواهیم همان محدودیتی که برای ترافیک tcp با پورت 80 برای user1 در نمونه های قبلی اعمال کردیم را با استفاده از نام سرویس محدود کنیم کافی است پروتکل **http** را برای user1 ببندیم :

```
soodar1(config)# ip access-list denyhttp
soodar1(config-ipv6-acl)# 10 deny http 1.1.1.10/32 2.1.1.10/24
soodar1(config-ipv6-acl)# permit any any
soodar1(config-nacl)# do sh ip access-list
IP access list denyhttp
   seq 10 deny   tcp 1.1.1.10/32 2.1.1.0/24 eq 80
   seq 15 permit any any
```

توجه

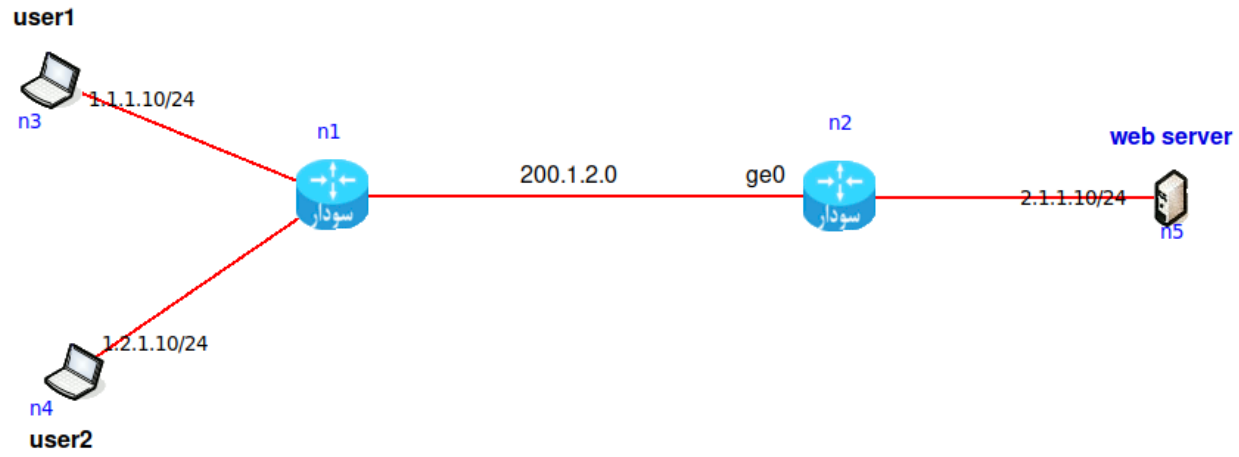
توجه شود در زمان استفاده از نام پروتکل در ACL باید index بعدی رولی که اضافه می کنیم خالی باشد و هیچ رول دیگری وجود نداشته باشد . به طور نمونه در این مثال ما از index شماره 10 استفاده کرده ایم و باید در index شماره 11 هیچ رولی نداشته باشیم . بدین علت این شرایط باید رعایت شود که یک رول برای مسیر کلاینت به سرور و یک رول برای مسیر سرور به کلاینت در ACL اضافه می شود .
همان طور که در sh ip access-list می بینید 2 رول برای http اضافه شده است .

توجه

ACLها را به نحوی تعریف کنید که فیلترهای کلی در ابتدای لیست قرار داشته باشند و فیلترهای جزئی در انتهای لیست.
در صورتی که در انتهای ACL از دستور permit any استفاده نشود و پکت ها با شرط های مشخص شده مطابقت نداشته باشد، پکت دور ریخته **(Drop)** خواهد شد.
ACLها تنها به ترافیک هایی اعمال خواهد شد که از روتر عبور میکنند، پس درنتیجه ترافیک هایی که از روتر تولید و ارسال می شوند با ACLها محدود نخواهند شد.

ACl Stateful 10.6

در acl stateless روتر نشست ها را بررسی نمی کند و شما لازم است برای مسیرهای برگشت خودتان رول های مد نظر را اضافه کنید اما وقتی یک acl در حالت stateful اضافه می شود نشست ها تشخیص داده شده و به محض match شدن یک بسته ، رول برعکس (مسیر برگشت همان بسته) به صورت خودکار اضافه می شود . برای مثال در شکل زیر قصد داریم که user1,user2 به web server دسترسی داشته باشند و user1 امکان ssh به web server را داشته باشد .




```
n2(config-nacl)# permit tcp 1.1.1.10/32 2.1.1.10/32 eq 443 reflect
n2(config-nacl)# permit tcp 1.2.1.10/32 2.1.1.10/32 eq 443 reflect
n2(config-nacl)# permit tcp 1.1.1.10/32 2.1.1.10/32 eq 22 reflect
```

توجه

دقت کنید وقتی در acl از رول های reflect استفاده کرده اید یا به عبارت دیگر acl شما stateful است باید acl را در اینترفیس در هر دو جهت in و out اعمال کنید

با توجه به رول هایی در acl اضافه کرده ایم این acl باید در اینترفیس ge0 اعمال شود حال acl را در اینترفیس ge0 اعمال می کنیم :

```
n2(config)# int ge0
n2(config-if)# ip access-group ACL-web-server in out
n2(config-if)# do sh ip access-list interfaces

Interface ge0
  Egress ACL: ACL-web-server
  Ingress ACL: ACL-web-server

Interface ge1
  Egress ACL:
  Ingress ACL:
n2(config-if)#
```

با این تنظیمات اگر بسته tcp با مبدا 1.1.1.10 (user1) و مقصد 2.1.1.10 (web) (server با پروتکل tcp و destination port برابر با 22 به اینترفیس ge0 در روتر n2 وارد شود ، روتر n2 اجازه خروج بسته پاسخ (بسته با مبدا 2.1.1.10 و مقصد 1.1.1.10 و پروتکل tcp و source port برابر 22) را از اینترفیس ge0 می دهد . به همین شکل برای بسته های https برای user1 و user2 عمل خواهد کرد .

11.6 فعال کردن **log** های **ACL**

با دستور زیر می توانید Log های مربوط به acl را فعال کنید:

```
soodar1# debug acl event
```

12.6 **resequence** کردن

اگر بخواهید شماره تمامی رول های تنظیم شده در یک acl را مجددا تنظیم کنید باید از resequence استفاده کنید :

```
ip access-list resequence ACL4 (1-2147483647) (1-32765)
```

که در آن ACL4 نام acl ، شماره اول شماره شروع و شماره دوم گام های افزایش است .

مثال :

```
soodar1(config-nacl)# do sh ip access-list
IP access list TEST
    10 permit 1.1.1.10/32 2.1.1.10/32
    13 deny   1.1.1.10/32 2.1.1.20/32
    19 deny   1.2.1.10/32 any
    22 deny   1.2.1.2/32 any
soodar1(config-nacl)# q
```

```
soodar1(config)# ip access-list resequence TEST 1000 10
soodar1(config)# do sh ip access-list TEST
IP access list TEST
    1000 permit 1.1.1.10/32 2.1.1.10/32
    1010 deny   1.1.1.10/32 2.1.1.20/32
    1020 deny   1.2.1.10/32 any
    1030 deny   1.2.1.2/32 any
```

فصل 7

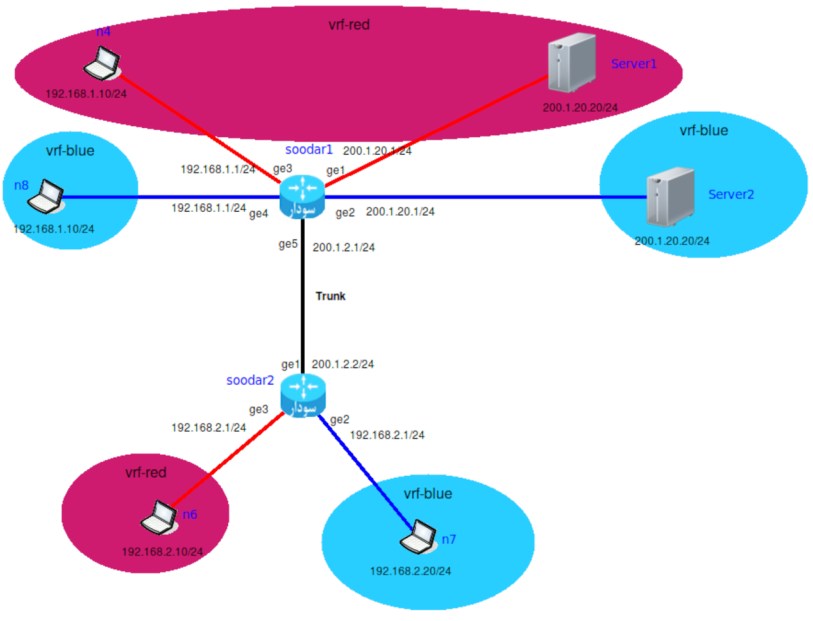
VRF

1.7 آشنایی با مفهوم VRF

تکنولوژی Virtual Routing and Forwarding یا به عبارت دیگر VRF برای پشتیبانی از چندین جدول مسیریابی در یک روتر مورد استفاده قرار میگیرد. به این معنی که یک روتر میتواند چندین routing table داشته باشد که هر کدام از آنها کاملا از هم مستقل هستند.به دلیل همین استقلال، اجازه استفاده از آدرس های یکسان (overlap) را نیز دارید. روتر سودار به صورت پیشفرض از VRF پشتیبانی میکند. همان طوری که VLAN می تواند برودکست دامین های جدید بسازد و از هم تفکیک کند روتر هم با VRF میتواند جدوال مسیر یابی مجزا و تفکیک شده ای را تشکیل دهد .

فرض کنید شما یک روتر دارید که همزمان از روی این روتر به چند مشتری سرویس می دهید ، به صورت پیش فرض و بدون استفاده از VRF همه مسیر های مشتریان شما در جدول روتینگ این روتر ثبت می شوند و به همین دلیل این مشتری های شما می توانند با هم ارتباط داشته باشند ، یا اینکه نیاز دارید از یک رنج IP همزمان به چند اینترفیس روتر IP اختصاص دهید که طبیعتا روتر این اجازه را به شما نخواهد داد . این جاست که vrf به کار می آید .

سناریوی زیر را در نظر بگیرید که در آن دو vrf با نام های vrf-blue , vrf-red وجود دارند . برخی ip ها در هر دو vrf وجود دارند . همچنین از trunking در vrf استفاده شده است .



[VRF Example]

2.7 اضافه کردن VRF جدید

برای تنظیم vrf ابتدا باید vrf را با دستور زیر ایجاد کرد :

soodar1

```
soodar1(config)# vrf vrf-red
soodar1(config)# vrf vrf-blue
```

soodar2

```
soodar2(config)# vrf vrf-red
soodar2(config)# vrf vrf-blue
```

3.7 قرار دادن اینترفیس در VRF

به شکل زیر می توان اینترفیس ها را در vrf مورد نظر قرار داد :

soodar1

```
soodar1(config)# interface ge1
soodar1(config-if)# ip vrf forwarding vrf-red
soodar1(config-if)# ip address 200.1.20.1/24
soodar1(config-if)# exit

soodar1(config)# interface ge3
soodar1(config-if)# ip vrf forwarding vrf-red
soodar1(config-if)# ip address 192.168.1.1/24
soodar1(config-if)# exit

-----

soodar1(config)# interface ge2
soodar1(config-if)# ip vrf forwarding vrf-blue
soodar1(config-if)# ip address 200.1.20.1/24
soodar1(config-if)# exit

soodar1(config)# interface ge4
soodar1(config-if)# ip vrf forwarding vrf-blue
```

(continues on next page)

(continued from previous page)

```
soodar1(config-if)# ip address 192.168.1.1/24
soodar1(config-if)# exit
```

soodar2

```
soodar2(config)# interface ge2
soodar2(config-if)# ip vrf forwarding vrf-blue
soodar2(config-if)# ip address 192.168.2.1/24
soodar2(config-if)# exit
```

```
soodar2(config)# interface ge3
soodar2(config-if)# ip vrf forwarding vrf-red
soodar2(config-if)# ip address 192.168.2.1/24
soodar2(config-if)# exit
```

نکته

دقت شود که با قرار دادن اینترفیس در vrf تمامی ip های آن حذف می شوند .

در زمان استفاده از پروتکل های مسیر یابی نیز به شیوه زیر می توانید از vrf استفاده کنید . در این مثال ما از ospf برای مسیریابی استفاده کرده ایم :

soodar1

```
soodar1(config)# router ospf vrf vrf-red
soodar1(config-router)# network 200.1.20.0/24   area 0
soodar1(config-router)# redistribute connected
soodar1(config-if) # end

-----

soodar1(config)# router ospf vrf vrf-blue
soodar1(config-router)# network 200.1.20.0/24   area 0
soodar1(config-router)# redistribute connected
soodar1(config-if) # end
soodar1# write
```

soodar2

```
soodar2(config)# router ospf vrf vrf-red
soodar2(config-router)# network 200.1.20.0/24   area 0
soodar2(config-router)# redistribute connected
soodar2(config-if) # end

-----

soodar2(config)# router ospf vrf vrf-blue
soodar2(config-router)# network 200.1.20.0/24   area 0
soodar2(config-router)# redistribute connected
soodar2(config-if) # end
soodar2# write
```

vrf trunking 4.7

روتر soodar2 نیز دارای vrf-red , vrf-blue می باشد و برای اینکه بتواند از طریق ospf با روتر soodar1 ارتباط بگیرد باید این دو روتر با 2 اینترفیس جداگانه به هم وصل باشند که در سناریوی ما فقط یک اینترفیس ge5) در soodar1 و ge1 در soodar2) بین دو روتر وجود دارد . برای رفع این مشکل باید از sub interface استفاده کنیم و هر sub interface را در vrf جداگانه ای قرار دهیم :

soodar1

```
soodar1(config)# interface ge5.100
soodar1(config-if)# encapsulation dot1q 100
soodar1(config-if)# ip vrf forwarding vrf-red
soodar1(config-if)# ip address 200.1.2.2/24
soodar1(config-if)# exit
```

```
soodar1(config)# interface ge5.200
soodar1(config-if)# encapsulation dot1q 200
soodar1(config-if)# ip vrf forwarding vrf-blue
soodar1(config-if)# ip address 200.1.2.2/24
soodar1(config-if)# exit
```

soodar2

```
soodar2(config)# interface ge1.100
soodar2(config)# encapsulation dot1q 100
soodar2(config-if)# ip vrf forwarding vrf-red
soodar2(config-if)# ip address 200.1.2.2/24
soodar2(config-if)# exit
```

```
soodar2(config)# interface ge1.200
soodar2(config)# encapsulation dot1q 200
soodar2(config-if)# ip vrf forwarding vrf-blue
soodar2(config-if)# ip address 200.1.2.2/24
soodar2(config-if)# exit
```

بدین ترتیب روتر soodar2 , soodar1 از طریق sub iface های ایجاد شده که هر کدام در vrf جداگانه قرار دارند با هم ارتباط می گیرند و جدول مسیریابی مخصوص به خود را تشکیل می دهند .

پس از اعمال شدن تنظیمات کمی صبر می کنیم تا روتر ها با استفاده از پروتکل ospf جدول Routing را تشکیل دهند . سپس با استفاده از دستور زیر جدول Routing که با استفاده از پروتکل ospf به ازای هر vrf تشکیل شده است را مشاهده می کنیم :

```
soodar1# sh ip ospf vrf vrf-red route
```

یا

```
soodar1# sh ip fib vrf vrf-red
```

5.7 بررسی عملکرد VRF

1. پس تشکیل جدول مسیریابی توسط ospf باید 3 جدول جداگانه وجود داشته باشد یکی برای vrf-blue و یکی برای vrf-red و یکی هم جدول مسیریابی default .

2. ارتباط نود های داخل هر vrf ار بقیه vrf ها جدا باشد . برای مثال نود های n8, n7 به server2 وصل شوند و نتوانند به server1 وصل شوند .

6.7 حذف VRF

برای حذف تنظیمات vrf کافی است با استفاده از کلمه **no** تنظیمات vrf را در هر بخش که اضافه شده اند را حذف کنید . در زیر چند نمونه آورده شده است :

```
soodar1(config)# no ip vrf vrf-red
-------
soodar1(config)# interface ge3
soodar1(config-if)# no ip vrf forwarding vrf-red
-------
soodar1(config)# no router ospf vrf vrf-blue
```

7.7 فعال کردن log های VRF

با دستور زیر می توانید Log های مربوط به vrf را فعال کنید:

```
soodar1# debug vrf event
```

فصل 8

MPLS

MPLS 1.8

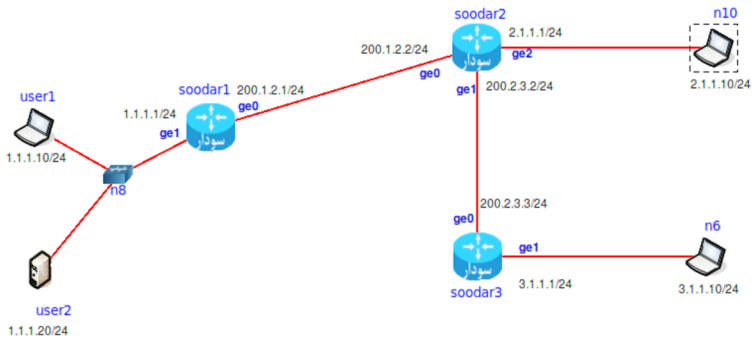
1.1.8 آشنایی با مفهوم MPLS

تنها دستگاهی که از سمت client با ام پی ال اس کار میکند روتر است، پس به عنوان مشترک کار چندانی به ساختار درونی MPLS نداریم و تنها به روتر سمت سرویس دهنده (Provider Edge) - PE) متصل شده و به کمک BGP یا هر روش دیگر؛ از شبکه های دیگر خود متصل به MPLS با خبر شده و اطلاعات خود را از میان ابر MPLS عبور میدهیم.

در نسخه فعلی روتر سودار پروتکل mpls و همچنین تونل vpls را پشتیبانی می کند و شما می توانید از آن در شبکه های خود استفاده نمایید .

آموزش راه اندازی MPLS در سودار

برای تست mpls سناریوی زیر را در نظر بگیرید . فرض می کنیم پروتکل ospf از قبل در این سناریو تنظیم شده است .(نحوه تنظیم ospf) . ما در این بخش فقط نحوه تنظیم پروتکل mpls را در این 3 روتر برای نمونه شرح می دهیم :



2.1.8 ldp router-id (اختیاری)

3.1.8 فعال کردن mpls در اینترفیس ها

1. اختصاص **ip** به اینترفیس **loopback** در ابتدا حتما باید یک اینترفیس loopback داشته باشید و به آن ip اختصاص دهید . اینترفیس loopback0 که یک اینترفیس loopback است به صورت پیش فرض در سودار ایجاد شده است و فقط لازم است یک ip دلخواه به آن اختصاص دهید :

```
soodar(config)# int loopback0
soodar(config-if)# ip address 222.1.1.1/32
```

2. فعال کردن **mpls** در اینترفیس به شکل زیر mpls را در اینترفیس مورد نظر فعال می کنیم :

```
soodar(config)# int ge0
soodar(config-if)# mpls ip
```

4.1.8 تنظیمات ldp (اختیاری)

تنظیم **router-id** می توانید با دستورات ldp router-id دلخواه خود را تنظیم کنید

```
soodar(config)#
soodar(config)# mpls ldp
soodar(config-ldp)# router-id 222.1.1.1
```

تنظیمات دیسکاوری به طور پیش فرض از بزرگترین IP اینترفیس loopback و در صورت نبود اینترفیس Loopback از بزرگترین IP دیگر اینترفیس ها برای discovery transport-address استفاده می شود اما اگر مایل باشید می توانید ip دلخواه خود را برای آن تنظیم کنید (دقت کنید روتر های همسایه برای برقراری ارتباط LDP حتما باید به این آدرس route داشته باشند)

```
soodar(config-ldp)# address-family ipv4
soodar(config-ldp-af)# discovery transport-address 5.10.2.111
```

حال با توجه به توضیحات بالا mpls را در روتر ها اعمال می کنیم :

1. اختصاص **ip** به اینترفیس **loopback0**

```
soodar1(config)# interface loopback0
soodar1(config-if)# ip address 222.1.1.1/32
```

2. فعال کردن **mpls** در اینترفیس ها Mpls را در اینترفیس های ge1 , ge0 در فعال می‌کنیم :

```
soodar1(config)# int ge0
soodar1(config-if)# mpls ip
```

3. تنظیمات پروتکل **LDP**

تنظیم router-id :

```
soodar1(config)#
soodar1(config)# mpls ldp
soodar1(config-ldp)# router-id 222.1.1.1
```

تنظیمات دیسکاوری :

```
soodar1(config-ldp)# address-family ipv4
soodar1(config-ldp-af)#
soodar1(config-ldp-af)# discovery transport-address 222.1.1.1
soodar1(config-ldp-af)# interface ge0
soodar1(config-ldp-af-if)#
soodar1(config-ldp-af-if)# exit
soodar1(config-ldp-af)# interface ge1
soodar1(config-ldp-af) # end
soodar1# write
```

برای دو روتر دیگر نیز به همین شکل عمل می کنیم . تنظیمات soodar2 , soodar3 را نیز در ادامه آورده ایم اما توضیحات مربوط به هر بخش بدلیل تکراری بودن حذف شده است :

soodar2

```
soodar2(config)# interface loopback0
soodar2(config-if)# ip address 222.2.2.2/32
```

```
soodar2(config)# int ge0
soodar2(config-if)# mpls ip
soodar2(config-if)# q
soodar2(config)# int ge1
soodar2(config-if)# mpls ip
```

```
soodar2(config)#
soodar2(config)# mpls ldp
soodar2(config-ldp)# router-id 222.2.2.2
```

```
soodar2(config-ldp)# address-family ipv4
soodar2(config-ldp-af)#
soodar2(config-ldp-af)# discovery transport-address 222.2.2.2
soodar2(config-ldp-af)# interface ge0
soodar2#(config-ldp-af-if)# exit
soodar2(config-ldp-af)# interface ge1
soodar2(config-ldp-af) # end
soodar2# write
```

soodar3

```
soodar3(config)# interface loopback0
soodar3(config-if)# ip address 222.3.3.3/32
```

```
soodar3(config)# int ge0
soodar3(config-if)# mpls ip
```

```
soodar3(config)#
soodar3(config)# mpls ldp
soodar3(config-ldp)# router-id 222.3.3.3
```

```
soodar3(config-ldp)# address-family ipv4
soodar3(config-ldp-af)#
soodar3(config-ldp-af)# discovery transport-address 222.3.3.3
soodar3(config-ldp-af)# interface ge0
soodar3#(config-ldp-af-if)# exit
soodar3(config-ldp-af)# interface ge1
soodar3(config-ldp-af) # end
soodar3# write
```

5.1.8 مشاهده جدول MPLS

با استفاده از دستورات زیر جدول mpls و همسایه های mpls روتر ها را مشاهده می‌کنیم :

```
soodar1# sh mpls ldp neighbor
AF   ID              State       Remote Address    Uptime
ipv4 222.2.2.2       OPERATIONAL 222.2.2.2       00:03:26

soodar1# sh mpls ldp binding
AF   Destination       Nexthop       Local Label Remote Label  In Use
ipv4 1.1.1.0/24        222.2.2.2     imp-null    16            no
ipv4 2.1.1.0/24        222.2.2.2     16          imp-null      yes
ipv4 3.1.1.0/24        222.2.2.2     19          19            yes
ipv4 111.1.1.0/24      222.2.2.2     imp-null    imp-null      no
ipv4 200.1.2.0/24      222.2.2.2     imp-null    imp-null      no
ipv4 200.1.3.0/24      222.2.2.2     imp-null    17            no
ipv4 200.2.3.0/24      222.2.2.2     17          imp-null      yes
ipv4 222.1.1.1/32      222.2.2.2     imp-null    18            no
ipv4 222.2.2.2/32      222.2.2.2     18          imp-null      yes
ipv4 222.3.3.3/32      222.2.2.2     20          20            yes
```

6.1.8 بررسی عملکرد MPLS

حال با ping کردن ارتباط بین شبکه های روتر ها را تست کنید . با چک کردن بسته ای در حال عبور بین روتر ها (با استفاده از پورت span) می توانید label بسته ها را مشاهده کنید و از اعمال شدن پروتکل mpls روی بسته ها اطمینان حاصل کنید .

7.1.8 غیر فعال کردن mpls

برای غیر فعال کردن mpls در روتر کافی است mpls را در اینترفیس هایی که فعال کرده اید غیر فعال کنید . همچنین می توانید پروتکل ldp را کلا در روتر غیر فعال کنید .

8.1.8 غیر فعال کردن **mpls** در اینترفیس

برای غیر فعال کردن mpls در اینترفیس باید دستور زیر را وارد کنید :

```
soodar3(config)# int ge0
soodar3(config-if)# no mpls ip
```

9.1.8 غیر فعال کردن **ldp**

پس از فعال کردن mpls در روتر به طور خودکار پروتکل ldp تنظیم می شود با غیر فعال کردن mpls در اینترفیس ها ldp دیگر روی آن اینترفیس با دیگر روتر ها صحبت نمی کند . اگر مایلید بعد از غیر فعال کردن mpls در همه اینترفیس ها ، تنظیمات ldp را نیز حذف کنید باید از دستور زیر استفاده کنید (دقت شود این دستور را زمانی استفاده کنید که mpls در همه اینترفیس ها غیر فعال شده باشد).

```
soodar3(config)# no mpls ldp
```

10.1.8 **MPLS** در **IPv6**

تنظیمات mpls در IPv6 همانند IPv4 می باشد با این تفاوت که به جای عبارت ip باید از ipv6 استفاده کنید برای مثال :

```
soodar3(config)# int ge0
soodar3(config-if)#  mpls ipv6
--------
soodar3(config-ldp)# address-family ipv6
soodar3(config-ldp-af)#
soodar3(config-ldp-af)# discovery transport-address 222.3.3.3
-------
soodar3(config)# int ge0
soodar3(config-if)#  no mpls ipv6
```

نکته

مسیر هایی که یک hop داشته باشند label نمی خورند و implicit-null هستند . بنابراین باید حداقل 2 hop بین مبدا و مقصد وجود داشته باشد تا label زده شود .

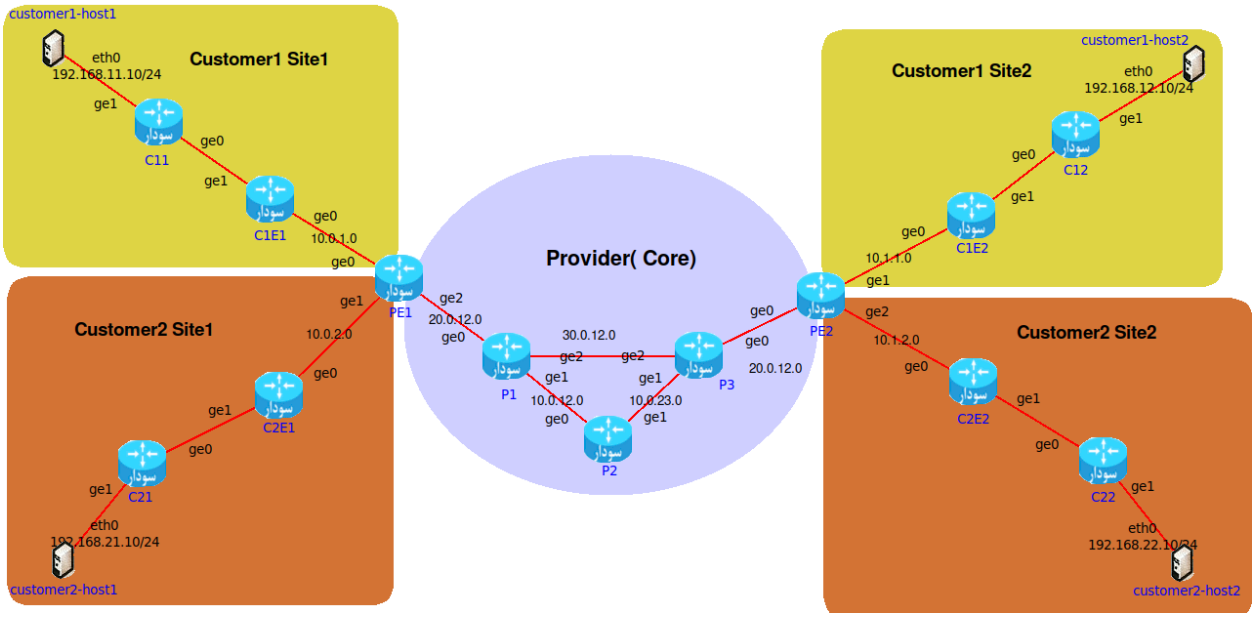
2.8 **l3vpn-MPLS**

1.2.8 آموزش راه اندازی **l3vpn-MPLS** در سودار

توضیح سناریو

در سناریوی زیر provider در core شبکه خود از mpls استفاده می کند و customer ها نیز در شبکه خود از ospf استفاده می کنند . هر customer می خواهد با استفاده از بستر mpls شبکه provider ارتباط بین شبکه (شعب) خود را برقرار کند . provider در روتر های PE خود برای هر customer یک vrf در نظر می گیرد و اینترفیس مربوط به هر customer را در vrf مخصوص به customer قرار می دهد . پس از آن با استفاده از MP-BGP ارتباط vpn بین همه PE ها برقرار می کند . در این vpn بسته های هر customer دارای tag مشخص می باشد که این تگ ها در شبکه core تفسیر نمی شود و فقط به سمت PE ارسال می شود در PE با توجه به tag ، بسته ها به سمت customer مورد نظر forward می شوند . بدین ترتیب چند customer می تواند بصورت جداگانه و بدون ارتباط با دیگر customer ها از شبکه provider استفاده نماید .

در ادامه تنظیمات همه روتر ها را مشاهده می کنید



تنظیمات روتر های هسته مبتنی بر MPLS در سرویس دهنده شبکه :(Provider Core)

P1

```
hostname P1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
service password-encryption
no banner motd
!
no ntp
!

!
interface erspan0
 no ip address
!
interface ge0
 mpls ip
 no shutdown
```

(continues on next page)

(continued from previous page)

```
 ip address 20.0.12.1/24
exit
!
interface ge1
 mpls ip
 no shutdown
 ip address 10.0.12.1/24
exit
!
interface ge2
 mpls ip
 no shutdown
 ip address 30.0.12.1/24
exit
!
interface gre0
 no ip address
!
interface gretap0
 no ip address
!
interface lo
 no ip address
!
interface tunl0
 no ip address
!
interface loopback0
 no shutdown
 ip address 3.3.3.3/32
exit
!
router ospf
 ospf router-id 3.3.3.3
 redistribute connected
 redistribute static
 network 3.3.3.3/32 area 0
 network 10.0.12.0/24 area 0
 network 20.0.12.0/24 area 0
 network 30.0.12.0/24 area 0
exit
!
mpls ldp
 !
 address-family ipv4
  discovery transport-address 3.3.3.3
  label local advertise explicit-null
  !
  interface ge0
  exit
  !
  interface ge1
  exit
  !
  interface ge2
  exit
  !
```

(continues on next page)

(continued from previous page)

```
 exit-address-family
 !
exit
!
end
```

P2

```
hostname P2
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
service password-encryption
no banner motd
!
no ntp
!

!
interface erspan0
 no ip address
!
interface ge0
 mpls ip
 no shutdown
 ip address 10.0.12.2/24
exit
!
interface ge1
 mpls ip
 no shutdown
 ip address 10.0.23.2/24
exit
!
interface ge2
 mpls ip
 ip address 30.0.12.2/24
exit
!
interface gre0
 no ip address
!
interface gretap0
 no ip address
!
interface lo
 no ip address
!
interface tunl0
 no ip address
!
interface loopback0
```

(continues on next page)

(continued from previous page)

```
 no shutdown
 ip address 6.6.6.6/32
exit
!
router ospf
 ospf router-id 6.6.6.6
 redistribute connected
 redistribute static
 network 6.6.6.6/32 area 0
 network 10.0.12.0/24 area 0
 network 10.0.23.0/24 area 0
 network 30.0.12.0/24 area 0
exit
!
mpls ldp
 !
 address-family ipv4
  discovery transport-address 6.6.6.6
  label local advertise explicit-null
  !
  interface ge0
  exit
  !
  interface ge1
  exit
  !
  interface ge2
  exit
  !
 exit-address-family
 !
exit
!
end
```

P3

```
hostname P3
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
service password-encryption
no banner motd
!
no ntp
!

!
interface erspan0
 no ip address
!
```

(continues on next page)

(continued from previous page)

```
interface ge0
 mpls ip
 no shutdown
 ip address 40.0.12.3/24
exit
!
interface ge1
 mpls ip
 no shutdown
 ip address 10.0.23.3/24
exit
!
interface ge2
 no ip address
!
interface gre0
 no ip address
!
interface gretap0
 no ip address
!
interface lo
 no ip address
!
interface tunl0
 no ip address
!
interface loopback0
 no shutdown
 ip address 7.7.7.7/32
exit
!
router ospf
 ospf router-id 7.7.7.7
 redistribute connected
 redistribute static
 network 7.7.7.7/32 area 0
 network 10.0.23.0/24 area 0
 network 40.0.12.0/24 area 0
exit
!
mpls ldp
 !
 address-family ipv4
  discovery transport-address 7.7.7.7
  label local advertise explicit-null
  !
  interface ge0
  exit
  !
  interface ge1
  exit
  !
 exit-address-family
 !
exit
!
```

(continues on next page)

(continued from previous page)

```
end
```

تنظیمات روتر های لبه در سرویس دهنده شبکه provider edge) یا PE) :

PE1

```
hostname PE1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!

!
vrf customer1
!
vrf customer2
!
interface customer1 vrf customer1
 no shutdown
!
interface customer2 vrf customer2
 no shutdown
!
interface loopback10
 no shutdown
 ip address 1.1.1.1/32
!
interface ge0 vrf customer1
 ip vrf forwarding customer1
 no shutdown
 ip address 10.0.1.1/24
!
interface ge1 vrf customer2
 ip vrf forwarding customer2
 no shutdown
 ip address 10.0.2.1/24
!
interface ge2
 mpls ip
 no shutdown
 ip address 20.0.12.3/24
!
router bgp 65000
 no bgp default ipv4-unicast
 neighbor 4.4.4.4 remote-as 65000
 neighbor 4.4.4.4 update-source loopback10
 !
 address-family ipv4 vpn
  neighbor 4.4.4.4 activate
```

(continues on next page)

(continued from previous page)

```
 exit-address-family
!
router bgp 65000 vrf customer1
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute static
  label vpn export auto
  redistribute ospf
  rd vpn export 65000:100
  rt vpn both 65000:100
  export vpn
  import vpn
 exit-address-family
!
router bgp 65000 vrf customer2
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute static
  redistribute ospf
  label vpn export auto
  rd vpn export 65000:200
  rt vpn both 65000:200
  export vpn
  import vpn
 exit-address-family
!
router ospf
 ospf router-id 1.1.1.1
 redistribute connected
 redistribute static
 network 1.1.1.1/32 area 0
 network 20.0.12.0/24 area 0
exit
!
router ospf vrf customer1
 ospf router-id 1.1.1.1
 redistribute bgp
 network 10.0.1.1/24 area 0
exit
!
router ospf vrf customer2
 ospf router-id 1.1.1.1
 redistribute bgp
 network 10.0.2.1/24 area 0
exit
!
mpls ldp
 !
 address-family ipv4
  discovery transport-address 1.1.1.1
  label local advertise explicit-null
  !
  interface ge2
  !
 exit-address-family
```

(continues on next page)

(continued from previous page)

```
 !
!
line vty
!
```

PE2

```
hostname PE2
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!

!
vrf customer1
!
vrf customer2
!
interface customer1 vrf customer1
 no shutdown
!
interface customer2 vrf customer2
 no shutdown
!
interface loopback10
 no shutdown
 ip address 4.4.4.4/32
!
interface ge1 vrf customer1
 ip vrf forwarding customer1
 no shutdown
 ip address 10.1.1.1/24
!
interface ge2 vrf customer2
 ip vrf forwarding customer2
 no shutdown
 ip address 10.1.2.1/24
!
interface ge0
 mpls ip
 no shutdown
 ip address 40.0.12.4/24
!
router bgp 65000
 no bgp default ipv4-unicast
 neighbor 1.1.1.1 remote-as 65000
 neighbor 1.1.1.1 update-source loopback10
 !
 address-family ipv4 vpn
```

(continues on next page)

(continued from previous page)

```
  neighbor 1.1.1.1 activate
 exit-address-family
!
router bgp 65000 vrf customer1
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute static
  redistribute ospf
  label vpn export auto
  rd vpn export 65000:100
  rt vpn both 65000:100
  export vpn
  import vpn
 exit-address-family
!
router bgp 65000 vrf customer2
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute static
  redistribute ospf
  label vpn export auto
  rd vpn export 65000:200
  rt vpn both 65000:200
  export vpn
  import vpn
 exit-address-family
!
router ospf
 ospf router-id 4.4.4.4
 redistribute connected
 redistribute static
 network 4.4.4.4/32 area 0
 network 40.0.12.0/24 area 0
exit
!
router ospf vrf customer1
 ospf router-id 4.4.4.4
 redistribute bgp
 network 10.1.1.1/24 area 0
exit
!
router ospf vrf customer2
 ospf router-id 4.4.4.4
 redistribute bgp
 network 10.1.2.1/24 area 0
exit
!
mpls ldp
 !
 address-family ipv4
  discovery transport-address 4.4.4.4
  label local advertise explicit-null
  !
  interface ge0
  !
```

(continues on next page)

(continued from previous page)

```
 exit-address-family
 !
!
line vty
!
```

تنظیمات روتر های لبه Customer :

C1E1

```
hostname C1E1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!

!
interface loopback0
 no shutdown
 ip address 8.8.8.8/32
!
interface ge0
 no shutdown
 ip address 10.0.1.2/24
!
interface ge1
 no shutdown
 ip address 10.0.8.1/24
!
router ospf
 ospf router-id 8.8.8.8
 redistribute connected
 redistribute static
 network 8.8.8.8/32 area 0
 network 10.0.1.2/24 area 0
 network 10.0.8.1/24 area 0
exit
!
```

C1E2

```
hostname C1E2
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!

!
interface loopback0
 no shutdown
 ip address 10.10.10.10/32
!
interface ge0
 no shutdown
 ip address 10.1.1.2/24
!
interface ge1
 no shutdown
 ip address 10.0.10.1/24
!
router ospf
 ospf router-id 10.10.10.10
 redistribute connected
 redistribute static
 network 10.10.10.10/32 area 0
 network 10.1.1.2/24 area 0
 network 10.0.10.1/24 area 0
exit
!
```

C2E1

```
hostname C2E1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!

!
interface loopback0
 no shutdown
 ip address 9.9.9.9/32
!
interface ge0
 no shutdown
```

(continues on next page)

(continued from previous page)

```
 ip address 10.0.2.2/24
!
interface ge1
 no shutdown
 ip address 10.0.9.1/24
!
router ospf
 ospf router-id 9.9.9.9
 redistribute connected
 redistribute static
 network 9.9.9.9/32 area 0
 network 10.0.2.2/24 area 0
 network 10.0.9.1/24 area 0
exit
!
```

C2E2

```
hostname C2E2
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
interface loopback0
 no shutdown
 ip address 11.11.11.11/32
!
interface ge0
 no shutdown
 ip address 10.1.2.2/24
!
interface ge1
 no shutdown
 ip address 10.0.11.1/24
!
router ospf
 ospf router-id 11.11.11.11
 redistribute connected
 redistribute static
 network 11.11.11.11/32 area 0
 network 10.1.2.2/24 area 0
 network 10.0.11.1/24 area 0
exit
!
```

تنظیمات روتر های داخلی شبکه Customer:

C11

```
hostname C11
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
!
!
interface loopback0
 no shutdown
 ip address 12.12.12.12/32
!
interface ge0
 no shutdown
 ip address 10.0.8.2/24
!
interface ge1
 no shutdown
 ip address 192.168.11.1/24
!
router ospf
 ospf router-id 12.12.12.12
 redistribute connected
 redistribute static
 network 12.12.12.12/32 area 0
 network 10.0.8.2/24 area 0
exit
!
```

C12

```
hostname C12
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
interface loopback0
 no shutdown
 ip address 14.14.14.14/32
!
interface ge0
 no shutdown
 ip address 10.0.10.2/24
!
interface ge1
 no shutdown
 ip address 192.168.12.1/24
```

(continues on next page)

(continued from previous page)

```
!
router ospf
 ospf router-id 14.14.14.14
 redistribute connected
 redistribute static
 network 14.14.14.14/32 area 0
 network 10.0.10.2/24 area 0
exit
!
```

C21

```
hostname C21
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
interface loopback0
 no shutdown
 ip address 13.13.13.13/32
!
interface ge0
 no shutdown
 ip address 10.0.9.2/24
!
interface ge1
 no shutdown
 ip address 192.168.21.1/24
!
router ospf
 ospf router-id 13.13.13.13
 redistribute connected
 redistribute static
 network 13.13.13.13/32 area 0
 network 10.0.9.2/24 area 0
exit
!
```

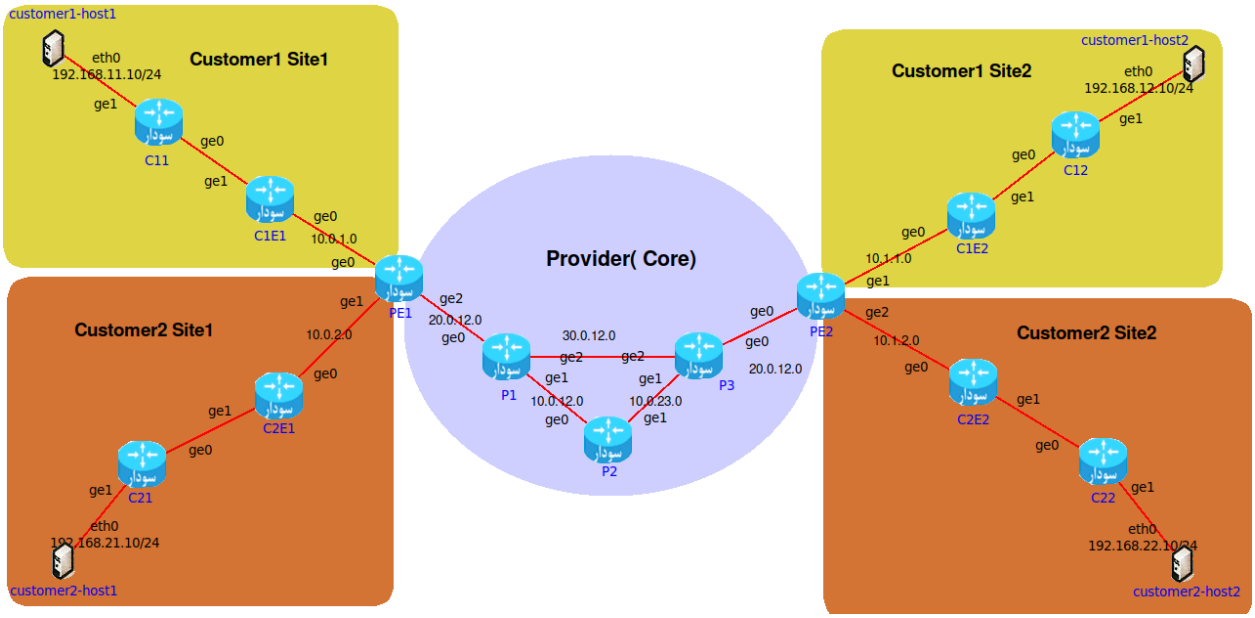
C22

```
hostname C22
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
```

(continues on next page)

(continued from previous page)

```
interface loopback0
 no shutdown
 ip address 15.15.15.15/32
!
interface ge0
 no shutdown
 ip address 10.0.11.2/24
!
interface ge1
 no shutdown
 ip address 192.168.22.1/24
!
router ospf
 ospf router-id 15.15.15.15
 redistribute connected
 redistribute static
 network 15.15.15.15/32 area 0
 network 10.0.11.2/24 area 0
exit
!
```



مشاهده جدول MPLS و bgp vpn

با استفاده از دستورات زیر جدول mpls و همسایه های mpls روتر ها را مشاهده می‌کنیم :

PE1

```
PE1# sh mpls ldp binding
AF   Destination          Nexthop        Local Label Remote Label  In Use
ipv4 1.1.1.1/32           3.3.3.3        exp-null    21            no
ipv4 3.3.3.3/32           3.3.3.3        66          exp-null      yes
ipv4 4.4.4.4/32           3.3.3.3        67          16            yes
ipv4 6.6.6.6/32           3.3.3.3        68          17            yes
ipv4 7.7.7.7/32           3.3.3.3        69          18            yes
ipv4 10.0.12.0/24         3.3.3.3        70          exp-null      yes
ipv4 10.0.23.0/24         3.3.3.3        71          19            yes
ipv4 20.0.12.0/24         3.3.3.3        exp-null    exp-null      no
ipv4 30.0.12.0/24         3.3.3.3        72          exp-null      yes
ipv4 40.0.12.0/24         3.3.3.3        73          20            yes

PE1#

PE1# sh ip bgp ipv4 vpn
BGP table version is 11, local router ID is 1.1.1.1, vrf id 0
Default local pref 100, local AS 65000
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 65000:100
*> 8.8.8.8/32       10.0.1.2@6<              10         32768 ?
   UN=10.0.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*> 10.0.1.0/24      0.0.0.0@6<               0          32768 ?
   UN=0.0.0.0 EC{65000:100} label=16 type=bgp, subtype=5
*> 10.0.8.0/24      10.0.1.2@6<              20         32768 ?
   UN=10.0.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*>i10.0.10.0/24     4.4.4.4                  20     100      0 ?
   UN=4.4.4.4 EC{65000:100} label=16 type=bgp, subtype=0
*>i10.1.1.0/24      4.4.4.4                  0      100      0 ?
   UN=4.4.4.4 EC{65000:100} label=16 type=bgp, subtype=0
*>i10.10.10.10/32   4.4.4.4                  10     100      0 ?
   UN=4.4.4.4 EC{65000:100} label=16 type=bgp, subtype=0
*> 12.12.12.12/32   10.0.1.2@6<              20         32768 ?
   UN=10.0.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*>i14.14.14.14/32   4.4.4.4                  20     100      0 ?
   UN=4.4.4.4 EC{65000:100} label=16 type=bgp, subtype=0
*> 192.168.11.0/24  10.0.1.2@6<              20         32768 ?
   UN=10.0.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*>i192.168.12.0/24  4.4.4.4                  20     100      0 ?
   UN=4.4.4.4 EC{65000:100} label=16 type=bgp, subtype=0
Route Distinguisher: 65000:200
```

(continues on next page)

(continued from previous page)

```
*> 9.9.9.9/32       10.0.2.2@7<             10         32768 ?
   UN=10.0.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*> 10.0.2.0/24      0.0.0.0@7<              0          32768 ?
   UN=0.0.0.0 EC{65000:200} label=17 type=bgp, subtype=5
*> 10.0.9.0/24      10.0.2.2@7<             20         32768 ?
   UN=10.0.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*>i10.0.11.0/24     4.4.4.4                 20    100      0 ?
   UN=4.4.4.4 EC{65000:200} label=17 type=bgp, subtype=0
*>i10.1.2.0/24      4.4.4.4                 0     100      0 ?
   UN=4.4.4.4 EC{65000:200} label=17 type=bgp, subtype=0
*>i11.11.11.11/32   4.4.4.4                 10    100      0 ?
   UN=4.4.4.4 EC{65000:200} label=17 type=bgp, subtype=0
*> 13.13.13.13/32   10.0.2.2@7<             20         32768 ?
   UN=10.0.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*>i15.15.15.15/32   4.4.4.4                 20    100      0 ?
   UN=4.4.4.4 EC{65000:200} label=17 type=bgp, subtype=0
*> 192.168.21.0/24  10.0.2.2@7<             20         32768 ?
   UN=10.0.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*>i192.168.22.0/24  4.4.4.4                 20    100      0 ?
   UN=4.4.4.4 EC{65000:200} label=17 type=bgp, subtype=0

Displayed  20 routes and 20 total paths
```

PE2

```
PE2# sh mpls ldp binding
AF   Destination          Nexthop         Local Label Remote Label  In Use
ipv4 1.1.1.1/32           7.7.7.7         73          22            yes
ipv4 3.3.3.3/32           7.7.7.7         68          17            yes
ipv4 4.4.4.4/32           7.7.7.7         exp-null    16              no
ipv4 6.6.6.6/32           7.7.7.7         69          18            yes
ipv4 7.7.7.7/32           7.7.7.7         66          exp-null      yes
ipv4 10.0.12.0/24         7.7.7.7         70          19            yes
ipv4 10.0.23.0/24         7.7.7.7         67          exp-null      yes
ipv4 20.0.12.0/24         7.7.7.7         71          20            yes
ipv4 30.0.12.0/24         7.7.7.7         72          21            yes
ipv4 40.0.12.0/24         7.7.7.7         exp-null    exp-null        no


PE2# sh ip bgp ipv4  vpn
BGP table version is 11, local router ID is 4.4.4.4, vrf id 0
Default local pref 100, local AS 65000
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 65000:100
*>i8.8.8.8/32       1.1.1.1                 10    100      0 ?
   UN=1.1.1.1 EC{65000:100} label=16 type=bgp, subtype=0
```

(continues on next page)

(continued from previous page)

```
*>i10.0.1.0/24      1.1.1.1                  0    100      0 ?
   UN=1.1.1.1 EC{65000:100} label=16 type=bgp, subtype=0
*>i10.0.8.0/24      1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:100} label=16 type=bgp, subtype=0
*> 10.0.10.0/24     10.1.1.2@5<             20         32768 ?
   UN=10.1.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*> 10.1.1.0/24      0.0.0.0@5<               0         32768 ?
   UN=0.0.0.0 EC{65000:100} label=16 type=bgp, subtype=5
*> 10.10.10.10/32   10.1.1.2@5<             10         32768 ?
   UN=10.1.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*>i12.12.12.12/32   1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:100} label=16 type=bgp, subtype=0
*> 14.14.14.14/32   10.1.1.2@5<             20         32768 ?
   UN=10.1.1.2 EC{65000:100} label=16 type=bgp, subtype=5
*>i192.168.11.0/24  1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:100} label=16 type=bgp, subtype=0
*> 192.168.12.0/24  10.1.1.2@5<             20         32768 ?
   UN=10.1.1.2 EC{65000:100} label=16 type=bgp, subtype=5
Route Distinguisher: 65000:200
*>i9.9.9.9/32       1.1.1.1                 10    100      0 ?
   UN=1.1.1.1 EC{65000:200} label=17 type=bgp, subtype=0
*>i10.0.2.0/24      1.1.1.1                  0    100      0 ?
   UN=1.1.1.1 EC{65000:200} label=17 type=bgp, subtype=0
*>i10.0.9.0/24      1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:200} label=17 type=bgp, subtype=0
*> 10.0.11.0/24     10.1.2.2@6<             20         32768 ?
   UN=10.1.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*> 10.1.2.0/24      0.0.0.0@6<               0         32768 ?
   UN=0.0.0.0 EC{65000:200} label=17 type=bgp, subtype=5
*> 11.11.11.11/32   10.1.2.2@6<             10         32768 ?
   UN=10.1.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*>i13.13.13.13/32   1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:200} label=17 type=bgp, subtype=0
*> 15.15.15.15/32   10.1.2.2@6<             20         32768 ?
   UN=10.1.2.2 EC{65000:200} label=17 type=bgp, subtype=5
*>i192.168.21.0/24  1.1.1.1                 20    100      0 ?
   UN=1.1.1.1 EC{65000:200} label=17 type=bgp, subtype=0
*> 192.168.22.0/24  10.1.2.2@6<             20         32768 ?
   UN=10.1.2.2 EC{65000:200} label=17 type=bgp, subtype=5

Displayed  20 routes and 20 total paths
PE2#
```

P1

```
P1# sh mpls ldp binding
AF   Destination        Nexthop         Local Label Remote Label  In Use
ipv4 1.1.1.1/32         1.1.1.1         21          exp-null      yes
ipv4 1.1.1.1/32         6.6.6.6         21          22            no
ipv4 3.3.3.3/32         1.1.1.1         exp-null    66            no
ipv4 3.3.3.3/32         6.6.6.6         exp-null    16            no
ipv4 4.4.4.4/32         1.1.1.1         16          67            no
```

(continues on next page)

(continued from previous page)

```
ipv4 4.4.4.4/32        6.6.6.6        16        17             yes
ipv4 6.6.6.6/32        1.1.1.1        17        68             no
ipv4 6.6.6.6/32        6.6.6.6        17        exp-null       yes
ipv4 7.7.7.7/32        1.1.1.1        18        69             no
ipv4 7.7.7.7/32        6.6.6.6        18        18             yes
ipv4 10.0.12.0/24      1.1.1.1        exp-null  70             no
ipv4 10.0.12.0/24      6.6.6.6        exp-null  exp-null       no
ipv4 10.0.23.0/24      1.1.1.1        19        71             no
ipv4 10.0.23.0/24      6.6.6.6        19        exp-null       yes
ipv4 20.0.12.0/24      1.1.1.1        exp-null  exp-null       no
ipv4 20.0.12.0/24      6.6.6.6        exp-null  19             no
ipv4 30.0.12.0/24      1.1.1.1        exp-null  72             no
ipv4 30.0.12.0/24      6.6.6.6        exp-null  20             no
ipv4 40.0.12.0/24      1.1.1.1        20        73             no
ipv4 40.0.12.0/24      6.6.6.6        20        21             yes
```

P2

```
P2# sh mpls ldp binding
AF   Destination        Nexthop        Local Label Remote Label  In Use
ipv4 1.1.1.1/32         3.3.3.3        22        21             yes
ipv4 1.1.1.1/32         7.7.7.7        22        22             no
ipv4 3.3.3.3/32         3.3.3.3        16        exp-null       yes
ipv4 3.3.3.3/32         7.7.7.7        16        17             no
ipv4 4.4.4.4/32         3.3.3.3        17        16             no
ipv4 4.4.4.4/32         7.7.7.7        17        16             yes
ipv4 6.6.6.6/32         3.3.3.3        exp-null  17             no
ipv4 6.6.6.6/32         7.7.7.7        exp-null  18             no
ipv4 7.7.7.7/32         3.3.3.3        18        18             no
ipv4 7.7.7.7/32         7.7.7.7        18        exp-null       yes
ipv4 10.0.12.0/24       3.3.3.3        exp-null  exp-null       no
ipv4 10.0.12.0/24       7.7.7.7        exp-null  19             no
ipv4 10.0.23.0/24       3.3.3.3        exp-null  19             no
ipv4 10.0.23.0/24       7.7.7.7        exp-null  exp-null       no
ipv4 20.0.12.0/24       3.3.3.3        19        exp-null       yes
ipv4 20.0.12.0/24       7.7.7.7        19        20             no
ipv4 30.0.12.0/24       3.3.3.3        20        exp-null       yes
ipv4 30.0.12.0/24       7.7.7.7        20        21             no
ipv4 40.0.12.0/24       3.3.3.3        21        20             no
ipv4 40.0.12.0/24       7.7.7.7        21        exp-null       yes

P2#
```

P3

```
P3# sh mpls ldp binding
AF   Destination          Nexthop         Local Label Remote Label  In Use
ipv4 1.1.1.1/32           4.4.4.4         22          73             no
ipv4 1.1.1.1/32           6.6.6.6         22          22             yes
ipv4 3.3.3.3/32           4.4.4.4         17          68             no
ipv4 3.3.3.3/32           6.6.6.6         17          16             yes
ipv4 4.4.4.4/32           4.4.4.4         16          exp-null       yes
ipv4 4.4.4.4/32           6.6.6.6         16          17             no
ipv4 6.6.6.6/32           4.4.4.4         18          69             no
ipv4 6.6.6.6/32           6.6.6.6         18          exp-null       yes
ipv4 7.7.7.7/32           4.4.4.4         exp-null    66             no
ipv4 7.7.7.7/32           6.6.6.6         exp-null    18             no
ipv4 10.0.12.0/24         4.4.4.4         19          70             no
ipv4 10.0.12.0/24         6.6.6.6         19          exp-null       yes
ipv4 10.0.23.0/24         4.4.4.4         exp-null    67             no
ipv4 10.0.23.0/24         6.6.6.6         exp-null    exp-null       no
ipv4 20.0.12.0/24         4.4.4.4         20          71             no
ipv4 20.0.12.0/24         6.6.6.6         20          19             yes
ipv4 30.0.12.0/24         4.4.4.4         21          72             no
ipv4 30.0.12.0/24         6.6.6.6         21          20             yes
ipv4 40.0.12.0/24         4.4.4.4         exp-null    exp-null       no
ipv4 40.0.12.0/24         6.6.6.6         exp-null    21             no

P3#
```

مشاهده جدول مسیریابی

C1E1

```
C1E1# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF default:
C>* 8.8.8.8/32 is directly connected, loopback0, 00:43:07
C>* 10.0.1.0/24 is directly connected, ge0, 00:43:10
C>* 10.0.8.0/24 is directly connected, ge1, 00:43:10
O>* 10.0.10.0/24 [110/20] via 10.0.1.1, ge0, weight 1, 00:42:08
O>* 10.1.1.0/24 [110/20] via 10.0.1.1, ge0, weight 1, 00:42:08
O>* 10.10.10.10/32 [110/20] via 10.0.1.1, ge0, weight 1, 00:42:08
O>* 12.12.12.12/32 [110/10] via 10.0.8.2, ge1, weight 1, 00:42:18
O>* 14.14.14.14/32 [110/20] via 10.0.1.1, ge0, weight 1, 00:42:08
O>* 192.168.11.0/24 [110/20] via 10.0.8.2, ge1, weight 1, 00:42:17
```

(continues on next page)

(continued from previous page)

```
O>* 192.168.12.0/24 [110/20] via 10.0.1.1, ge0, weight 1, 00:42:08
C1E1#
```

C1E2

```
C1E2# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF default:
O>* 8.8.8.8/32 [110/20] via 10.1.1.1, ge0, weight 1, 00:49:06
O>* 10.0.1.0/24 [110/20] via 10.1.1.1, ge0, weight 1, 00:49:06
O>* 10.0.8.0/24 [110/20] via 10.1.1.1, ge0, weight 1, 00:49:06
C>* 10.0.10.0/24 is directly connected, ge1, 00:50:07
C>* 10.1.1.0/24 is directly connected, ge0, 00:50:07
C>* 10.10.10.10/32 is directly connected, loopback0, 00:50:05
O>* 12.12.12.12/32 [110/20] via 10.1.1.1, ge0, weight 1, 00:49:06
O>* 14.14.14.14/32 [110/10] via 10.0.10.2, ge1, weight 1, 00:49:16
O>* 192.168.11.0/24 [110/20] via 10.1.1.1, ge0, weight 1, 00:49:06
O>* 192.168.12.0/24 [110/20] via 10.0.10.2, ge1, weight 1, 00:49:15
C1E2#
```

C2E1

```
C2E1# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF default:
C>* 9.9.9.9/32 is directly connected, loopback0, 00:50:31
C>* 10.0.2.0/24 is directly connected, ge0, 00:50:33
C>* 10.0.9.0/24 is directly connected, ge1, 00:50:33
O>* 10.0.11.0/24 [110/20] via 10.0.2.1, ge0, weight 1, 00:49:32
O>* 10.1.2.0/24 [110/20] via 10.0.2.1, ge0, weight 1, 00:49:32
O>* 11.11.11.11/32 [110/20] via 10.0.2.1, ge0, weight 1, 00:49:32
O>* 13.13.13.13/32 [110/10] via 10.0.9.2, ge1, weight 1, 00:49:37
O>* 15.15.15.15/32 [110/20] via 10.0.2.1, ge0, weight 1, 00:49:32
O>* 192.168.21.0/24 [110/20] via 10.0.9.2, ge1, weight 1, 00:49:36
O>* 192.168.22.0/24 [110/20] via 10.0.2.1, ge0, weight 1, 00:49:32
```

(continues on next page)

(continued from previous page)

```
C2E1#
```

C2E2

```
C2E2# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF default:
O>* 9.9.9.9/32 [110/20] via 10.1.2.1, ge0, weight 1, 00:49:55
O>* 10.0.2.0/24 [110/20] via 10.1.2.1, ge0, weight 1, 00:49:55
O>* 10.0.9.0/24 [110/20] via 10.1.2.1, ge0, weight 1, 00:49:55
C>* 10.0.11.0/24 is directly connected, ge1, 00:50:56
C>* 10.1.2.0/24 is directly connected, ge0, 00:50:56
C>* 11.11.11.11/32 is directly connected, loopback0, 00:50:54
O>* 13.13.13.13/32 [110/20] via 10.1.2.1, ge0, weight 1, 00:49:55
O>* 15.15.15.15/32 [110/10] via 10.0.11.2, ge1, weight 1, 00:50:00
O>* 192.168.21.0/24 [110/20] via 10.1.2.1, ge0, weight 1, 00:49:55
O>* 192.168.22.0/24 [110/20] via 10.0.11.2, ge1, weight 1, 00:49:59
C2E2#
```

PE1

```
PE1# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF customer1:
O>* 8.8.8.8/32 [110/10] via 10.0.1.2, ge0, weight 1, 00:41:38
C>* 10.0.1.0/24 is directly connected, ge0, 00:42:34
O>* 10.0.8.0/24 [110/20] via 10.0.1.2, ge0, weight 1, 00:41:38
B>  10.0.10.0/24 [20/20] via 4.4.4.4 (vrf default) (recursive), label 16, weight 1, 00:41:
32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/16, weight 1, 00:41:
32
B>  10.1.1.0/24 [20/0] via 4.4.4.4 (vrf default) (recursive), label 16, weight 1, 00:41:32
  *                   via 20.0.12.1, ge2 (vrf default), label 16/16, weight 1, 00:41:32
B>  10.10.10.10/32 [20/10] via 4.4.4.4 (vrf default) (recursive), label 16, weight 1, 00:4
1:32
```

(continues on next page)

(continued from previous page)

```
  *                    via 20.0.12.1, ge2 (vrf default), label 16/16, weight 1, 00:4
1:32
O>* 12.12.12.12/32 [110/20] via 10.0.1.2, ge0, weight 1, 00:41:38
B>  14.14.14.14/32 [20/20] via 4.4.4.4 (vrf default) (recursive), label 16, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/16, weight 1, 00:41:32
O>* 192.168.11.0/24 [110/20] via 10.0.1.2, ge0, weight 1, 00:41:37
B>  192.168.12.0/24 [20/20] via 4.4.4.4 (vrf default) (recursive), label 16, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/16, weight 1, 00:41:32

VRF customer2:
O>* 9.9.9.9/32 [110/10] via 10.0.2.2, ge1, weight 1, 00:41:37
C>* 10.0.2.0/24 is directly connected, ge1, 00:42:33
O>* 10.0.9.0/24 [110/20] via 10.0.2.2, ge1, weight 1, 00:41:37
B>  10.0.11.0/24 [20/20] via 4.4.4.4 (vrf default) (recursive), label 17, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/17, weight 1, 00:41:32
B>  10.1.2.0/24 [20/0] via 4.4.4.4 (vrf default) (recursive), label 17, weight 1, 00:41:32
  *                  via 20.0.12.1, ge2 (vrf default), label 16/17, weight 1, 00:41:32
B>  11.11.11.11/32 [20/10] via 4.4.4.4 (vrf default) (recursive), label 17, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/17, weight 1, 00:41:32
O>* 13.13.13.13/32 [110/20] via 10.0.2.2, ge1, weight 1, 00:41:37
B>  15.15.15.15/32 [20/20] via 4.4.4.4 (vrf default) (recursive), label 17, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/17, weight 1, 00:41:32
O>* 192.168.21.0/24 [110/20] via 10.0.2.2, ge1, weight 1, 00:41:36
B>  192.168.22.0/24 [20/20] via 4.4.4.4 (vrf default) (recursive), label 17, weight 1, 00:41:32
  *                    via 20.0.12.1, ge2 (vrf default), label 16/17, weight 1, 00:41:32

VRF default:
C>* 1.1.1.1/32 is directly connected, loopback10, 00:42:31
O>* 3.3.3.3/32 [110/10] via 20.0.12.1, ge2, label IPv4 Explicit Null, weight 1, 00:41:37
O>* 4.4.4.4/32 [110/40] via 20.0.12.1, ge2, label 16, weight 1, 00:41:37
O>* 6.6.6.6/32 [110/20] via 20.0.12.1, ge2, label 17, weight 1, 00:41:37
O>* 7.7.7.7/32 [110/30] via 20.0.12.1, ge2, label 18, weight 1, 00:41:37
O>* 10.0.12.0/24 [110/20] via 20.0.12.1, ge2, label IPv4 Explicit Null, weight 1, 00:41:37
O>* 10.0.23.0/24 [110/30] via 20.0.12.1, ge2, label 19, weight 1, 00:41:37
C>* 20.0.12.0/24 is directly connected, ge2, 00:42:33
O>* 30.0.12.0/24 [110/20] via 20.0.12.1, ge2, label IPv4 Explicit Null, weight 1, 00:41:37
O>* 40.0.12.0/24 [110/40] via 20.0.12.1, ge2, label 20, weight 1, 00:41:37
PE1#
```

PE2

```
PE2# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF customer1:
B>  8.8.8.8/32 [20/10] via 1.1.1.1 (vrf default) (recursive), label 16, weight 1, 00:48:28
  *                    via 40.0.12.3, ge0 (vrf default), label 22/16, weight 1, 00:48:28
```

(continues on next page)

(continued from previous page)

```
B>  10.0.1.0/24 [20/0] via 1.1.1.1 (vrf default) (recursive), label 16, weight 1, 00:48:28
  *                    via 40.0.12.3, ge0 (vrf default), label 22/16, weight 1, 00:48:28
B>  10.0.8.0/24 [20/20] via 1.1.1.1 (vrf default) (recursive), label 16, weight 1, 00:48:28
  *                     via 40.0.12.3, ge0 (vrf default), label 22/16, weight 1, 00:48:28
O>* 10.0.10.0/24 [110/20] via 10.1.1.2, ge1, weight 1, 00:48:33
C>* 10.1.1.0/24 is directly connected, ge1, 00:49:28
O>* 10.10.10.10/32 [110/10] via 10.1.1.2, ge1, weight 1, 00:48:33
B>  12.12.12.12/32 [20/20] via 1.1.1.1 (vrf default) (recursive), label 16, weight 1, 00:48:28
  *                        via 40.0.12.3, ge0 (vrf default), label 22/16, weight 1, 00:48:28
O>* 14.14.14.14/32 [110/20] via 10.1.1.2, ge1, weight 1, 00:48:33
B>  192.168.11.0/24 [20/20] via 1.1.1.1 (vrf default) (recursive), label 16, weight 1, 00:48:28
  *                         via 40.0.12.3, ge0 (vrf default), label 22/16, weight 1, 00:48:28
O>* 192.168.12.0/24 [110/20] via 10.1.1.2, ge1, weight 1, 00:48:32

VRF customer2:
B>  9.9.9.9/32 [20/10] via 1.1.1.1 (vrf default) (recursive), label 17, weight 1, 00:48:28
  *                    via 40.0.12.3, ge0 (vrf default), label 22/17, weight 1, 00:48:28
B>  10.0.2.0/24 [20/0] via 1.1.1.1 (vrf default) (recursive), label 17, weight 1, 00:48:28
  *                    via 40.0.12.3, ge0 (vrf default), label 22/17, weight 1, 00:48:28
B>  10.0.9.0/24 [20/20] via 1.1.1.1 (vrf default) (recursive), label 17, weight 1, 00:48:28
  *                     via 40.0.12.3, ge0 (vrf default), label 22/17, weight 1, 00:48:28
O>* 10.0.11.0/24 [110/20] via 10.1.2.2, ge2, weight 1, 00:48:33
C>* 10.1.2.0/24 is directly connected, ge2, 00:49:28
O>* 11.11.11.11/32 [110/10] via 10.1.2.2, ge2, weight 1, 00:48:33
B>  13.13.13.13/32 [20/20] via 1.1.1.1 (vrf default) (recursive), label 17, weight 1, 00:48:28
  *                        via 40.0.12.3, ge0 (vrf default), label 22/17, weight 1, 00:48:28
O>* 15.15.15.15/32 [110/20] via 10.1.2.2, ge2, weight 1, 00:48:33
B>  192.168.21.0/24 [20/20] via 1.1.1.1 (vrf default) (recursive), label 17, weight 1, 00:48:28
  *                         via 40.0.12.3, ge0 (vrf default), label 22/17, weight 1, 00:48:28
O>* 192.168.22.0/24 [110/20] via 10.1.2.2, ge2, weight 1, 00:48:32

VRF default:
O>* 1.1.1.1/32 [110/40] via 40.0.12.3, ge0, label 22, weight 1, 00:48:33
O>* 3.3.3.3/32 [110/30] via 40.0.12.3, ge0, label 17, weight 1, 00:48:34
C>* 4.4.4.4/32 is directly connected, loopback10, 00:49:27
O>* 6.6.6.6/32 [110/20] via 40.0.12.3, ge0, label 18, weight 1, 00:48:34
O>* 7.7.7.7/32 [110/10] via 40.0.12.3, ge0, label IPv4 Explicit Null, weight 1, 00:48:34
O>* 10.0.12.0/24 [110/30] via 40.0.12.3, ge0, label 19, weight 1, 00:48:34
O>* 10.0.23.0/24 [110/20] via 40.0.12.3, ge0, label IPv4 Explicit Null, weight 1, 00:48:34
O>* 20.0.12.0/24 [110/40] via 40.0.12.3, ge0, label 20, weight 1, 00:48:34
O>* 30.0.12.0/24 [110/40] via 40.0.12.3, ge0, label 21, weight 1, 00:48:34
C>* 40.0.12.0/24 is directly connected, ge0, 00:49:28
PE2#
```

P1

```
P1# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
```

(continues on next page)

(continued from previous page)

```
      t - trapped, o - offload failure

VRF default:
O>* 1.1.1.1/32 [110/10] via 20.0.12.3, ge0, label IPv4 Explicit Null, weight 1, 00:46:23
C>* 3.3.3.3/32 is directly connected, loopback0, 00:47:17
O>* 4.4.4.4/32 [110/30] via 10.0.12.2, ge1, label 17, weight 1, 00:46:24
O>* 6.6.6.6/32 [110/10] via 10.0.12.2, ge1, label IPv4 Explicit Null, weight 1, 00:46:24
O>* 7.7.7.7/32 [110/20] via 10.0.12.2, ge1, label 18, weight 1, 00:46:24
C>* 10.0.12.0/24 is directly connected, ge1, 00:47:19
O>* 10.0.23.0/24 [110/20] via 10.0.12.2, ge1, label IPv4 Explicit Null, weight 1, 00:46:24
C>* 20.0.12.0/24 is directly connected, ge0, 00:47:19
C>* 30.0.12.0/24 is directly connected, ge2, 00:47:18
O>* 40.0.12.0/24 [110/30] via 10.0.12.2, ge1, label 21, weight 1, 00:46:24
P1#
```

P2

```
P2# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF default:
O>* 1.1.1.1/32 [110/20] via 10.0.12.1, ge0, label 21, weight 1, 00:47:15
O>* 3.3.3.3/32 [110/10] via 10.0.12.1, ge0, label IPv4 Explicit Null, weight 1, 00:47:16
O>* 4.4.4.4/32 [110/20] via 10.0.23.3, ge1, label 16, weight 1, 00:47:16
C>* 6.6.6.6/32 is directly connected, loopback0, 00:48:09
O>* 7.7.7.7/32 [110/10] via 10.0.23.3, ge1, label IPv4 Explicit Null, weight 1, 00:47:16
C>* 10.0.12.0/24 is directly connected, ge0, 00:48:12
C>* 10.0.23.0/24 is directly connected, ge1, 00:48:12
O>* 20.0.12.0/24 [110/20] via 10.0.12.1, ge0, label IPv4 Explicit Null, weight 1, 00:47:16
O>* 30.0.12.0/24 [110/20] via 10.0.12.1, ge0, label IPv4 Explicit Null, weight 1, 00:47:16
O>* 40.0.12.0/24 [110/20] via 10.0.23.3, ge1, label IPv4 Explicit Null, weight 1, 00:47:16
P2#
```

P3

```
P3# sh ip fib vrf all
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure
```

(continues on next page)

(continued from previous page)

```
VRF default:
O>* 1.1.1.1/32 [110/30] via 10.0.23.2, ge1, label 22, weight 1, 00:47:33
O>* 3.3.3.3/32 [110/20] via 10.0.23.2, ge1, label 16, weight 1, 00:47:34
O>* 4.4.4.4/32 [110/10] via 40.0.12.4, ge0, label IPv4 Explicit Null, weight 1, 00:47:34
O>* 6.6.6.6/32 [110/10] via 10.0.23.2, ge1, label IPv4 Explicit Null, weight 1, 00:47:34
C>* 7.7.7.7/32 is directly connected, loopback0, 00:48:27
O>* 10.0.12.0/24 [110/20] via 10.0.23.2, ge1, label IPv4 Explicit Null, weight 1, 00:47:34
C>* 10.0.23.0/24 is directly connected, ge1, 00:48:29
O>* 20.0.12.0/24 [110/30] via 10.0.23.2, ge1, label 19, weight 1, 00:47:34
O>* 30.0.12.0/24 [110/30] via 10.0.23.2, ge1, label 20, weight 1, 00:47:34
C>* 40.0.12.0/24 is directly connected, ge0, 00:48:30
P3#
```

نکته

در جدول Route در روتر های PE مشاهده می شود که برای customer2 , customer1 جدول های route جداگانه وجود دارد و مسیریابی این دو از یکدیگر جدا است . این مسئله در جدول route روترهای CE (C1E1,C1E2,C2E1,C2E2) نیز قابل مشاهده است که در جدول هر کدام مسیرهای مربوط به شبکه خود را دارند .

فصل 9

تونل ها

gre 1.9

معرفی GRE 1.1.9

ما در این بخش ترافیک IP را روی بستر اشتراکی IP با روش GRE تونل می زنیم تا بتوانیم ارتباط سایت ها را روی بسترهای عمومی و اشتراکی مانند اینترنت به یکدیگر متصل کنیم.

دقیق تر اینکه اگر دو سایت داریم که محدوده آدرس آنها اختصاصی است و روی بستر اینترنت به یکدیگر متصل هستند، ارتباط کاربران این دو سایت به صورت کاملا Transparent بدون GRE Tunnel دشوار است. به این دلیل که آدرس های محدوده اختصاصی روی اینترنت مسیریابی نمی شوند و برای ارتباط هر دو کاربری که قصد ارتباط روی اینترنت را دارند باید از روش هایی مانند NAT استفاده شود که Transparent نیست. GRE Tunnel ترافیک بین دو سایت را (که آدرس مبدا و مقصد آن اختصاصی است) مجددا روی ترافیک جدیدی از نوع IP بسته بندی می کند که آدرس مبدا و مقصد آن آدرس بیرونی و اینترنتی روترهای مرزی دو سایت است و بدین ترتیب ارتباط بین دو سایت امکان پذیر می شود.

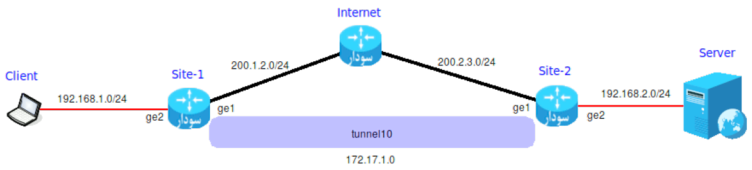
آموزش راه اندازی GRE در روتر سودار 2.1.9

اگر به شکل زیر توجه کنید دو سایت با آدرس های 192.168.1.0/24 و 192.168.2.0/24 از طریق اینترنت به یکدیگر متصل هستند. کاربران این دو سایت نمی توانند به صورت transparent و همانند LAN با یکدیگر ارتباط برقرار کنند زیرا آدرس های محدوده private در اینترنت مسیریابی نمی شوند. اینترفیس بیرونی روترهای مرزی این دو سایت آدرس اینترنتی IP1 و IP2 دارند که از طریق اینترنت قابل دیدن هستند و بنابراین روترهای مرزی این دو سایت می توانند کاملا به صورت transparent با هم در ارتباط باشند.

در چنین سناریویی با ایجاد Tunnel می توان ارتباط بین کاربران دو سایت را نیز به صورت کاملا transparent برقرار نمود. بدین صورت که ترافیک بین دو سایت که آدرس مبدا و مقصد آن 192.168.1.x و 192.168.2.x است، وقتی در روتر مرزی سایت مبدا ،که همان ابتدای Tunnel است، دریافت می شود، روی ترافیک دیگری با آدرس مبدا و مقصد 200.1.2.1/24 و 200.2.3.3/24 که آدرس ابتدا و انتهای tunnel هستند، سوار می شود. ترافیک جدید روی اینترنت مسیریابی شده و در روتر مرزی سایت مقصد که همان انتهای Tunnel است، دریافت می شود. بسته اصلی از سربار ایجاد شده جدا می شود و تحویل سایت مقصد خواهد شد.

GRE Tunnel روشی از tunneling است که قادر است هر نوع ترافیکی را روی IP سوار و منتقل کند. ضمنا قابلیت انتقال ترافیک های multicast و broadcast را نیز دارد. این بدان معنی است که اگر از سایت مبدا ترافیک multicast و یا broadcast به سایت مقصد ارسال شود، قابل انتقال توسط GRE خواهد بود. این ویژگی اجرای پروتکل های مسیریابی را روی GRE امکان پذیر می کند و یکی از مزایای مهم GRE محسوب می شود

در ذیل شیوه پیاده سازی تونل GRE بین دو سایت روی بستر اینترنت نشان داده شده است :



برای پیاده سازی تونل GRE مراحل زیر انجام می پذیرد:

برای تنظیم تونل gre باید ابتدا یک اینترفیس tunnel ایجاد کنید :

```
Site-1(config)# interface tunnel 10
```

سپس مبدا و مقصد تونل را مشخص می کنیم و به اینترفیس تونل ip می دهیم :

```
Site-1(config-if)# tunnel source 200.1.2.1
Site-1(config-if)# tunnel destination 200.2.3.3
Site-1(config-if)# ip address 172.17.1.10/32
```

در Site-2 هم به همین ترتیب عمل می کنیم :

```
Site-2(config)# interface tunnel 10
Site-2(config-if)# tunnel source 200.1.2.3
Site-2(config-if)# tunnel destination 200.1.2.1
Site-2(config-if)# ip address 172.17.1.20/32
```

اضافه کردن **Route** از طریق تونل

اکنون می توانیم با استفاده از static route ارتباط بین این دو شبکه را از طریق تونل برقرار کنیم :

```
Site-1(config)# ip route 192.168.2.0/24 tunnel10
Site-2(config)# ip route 192.168.1.0/24 tunnel10
```

تنظیم **mtu** در اینترفیس تونل

```
Site-1(config)# interface tunnel10
Site-1(config-if)# ip mtu 1464
Site-2(config)# interface tunnel10
Site-2(config-if)# ip mtu 1464
```

بررسی ارتباط تونل gre

پس از تنظیم تونل gre باید ارتباط کلاینت با سرور از طریق تونل gre برقرار باشد .

تنظیم vrf در تونل

به شکل زیر می توان جدول vrf که باید برای تونل gre استفاده شود را مشخص کرد :

```
soodar(config-if)# tunnel vrf green
```

با تنظیم فوق مشخص می شود که برای برقراری تونل gre و اصطلاحا lookup کردن ip مقصد از vrf green باید استفاده شود .

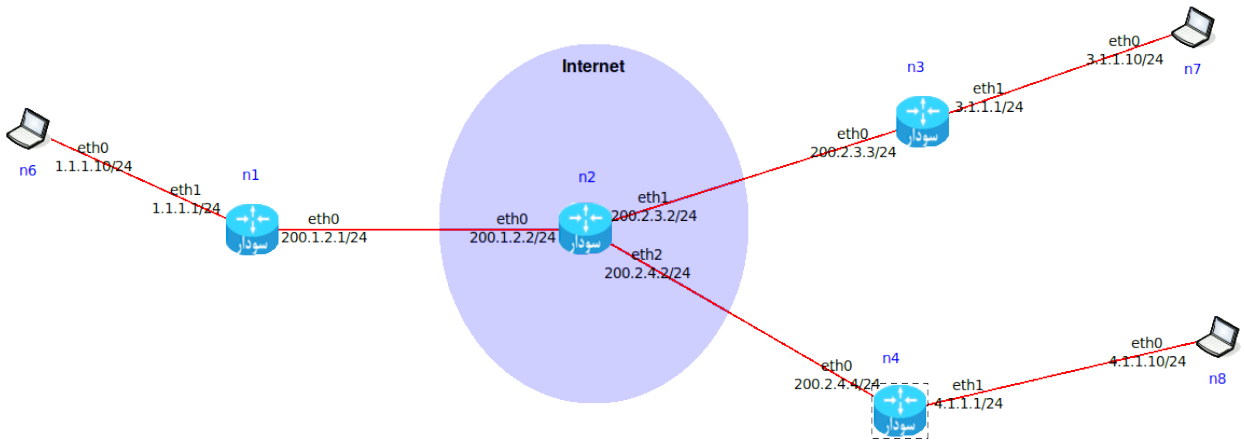
فعال کردن log های GRE

با دستور زیر می توانید Log های مربوط به gre را فعال کنید:

```
soodar1# debug gre event
```

mgre 3.1.9

در حال حاضر فقط نگاشت استاتیک NHRP در تونل mgre پشتیبانی می شود . سناریوی زیر را در نظر بگیرید . در این سناریو روتر n1 به روتر های n3 , n4 , تونل gre از نوع gre-mp دارد ارتباط این سه روتر از طریق اینترنت (نود n2) برقرار می شود .



نحوه تنظیم تونل mgre به شکل زیر است :

```
n1(config)# interface tunnel100
n1(config-if)# tunnel source 200.1.2.1
n1(config-if)# tunnel mode gre multipoint
n1(config-if)# ip address 192.168.1.1/32
```

(continues on next page)

(continued from previous page)

```
n1(config-if)# ip nhrp map 192.168.1.3 200.2.3.3
n1(config-if)# ip nhrp map 192.168.1.4 200.2.4.4
```

آدرس های 192.168.1.3 و 192.168.1.4 به ترتیب مربوط به ip اینترفیس تونل در روترهای n3 و n4 می باشد .

حال staticRoute را برای برقراری ارتباط شبکه های 4.1.1.0 , 3.1.1.0 , 1.1.1.0 بدین شکل در n1 اضافه می کنیم

```
n1(config)# ip route 3.1.1.0/24 192.168.1.3
n1(config)# ip route 4.1.1.0/24 192.168.1.4
```

تنظیمات تونل در n3 :

```
n3(config)# interface tunnel30
n3(config-if)# tunnel source 200.2.3.3
n3(config-if)# tunnel destination 200.1.2.1
n3(config-if)# ip address 192.168.1.3/32
n3(config)# ip route 1.1.1.0/24 192.168.1.3
n3(config)# ip route 4.1.1.0/24 192.168.1.3
```

تنظیمات تونل در n4 :

```
n4(config)# interface tunnel40
n4(config-if)# tunnel source 200.2.4.4
n4(config-if)# tunnel destination 200.1.2.1
n4(config-if)# ip address 192.168.1.4/32
n4(config)# ip route 1.1.1.0/24 192.168.1.4
n4(config)# ip route 3.1.1.0/24 192.168.1.4
```

بدین ترتیب یک تونل multi-point در روتر n1 ایجاد شده و ارتباط شبکه های 4.1.1.0/24 , 3.1.1.0/24 , 1.1.1.0/24 از طریق تونل gre برقرار می گردد .

2.9 IPSec

1.2.9 معرفی IPSec

IPSec یا Internet Protocol security عبارت است از مجموعه‌ای از چندین پروتکل که برای ایمن‌سازی پروتکل اینترنت در ارتباطات بوسیله احراز هویت و رمزگذاری در هربسته(packet) در یک سیر داده به کار می‌رود. IPSec بطور گسترده در تکنولوژی VPN جهت احراز هویت، محرمانگی، یکپارچگی و مدیریت کلید در شبکه های مبتنی بر IP، مورد استفاده قرار می گیرد. IPsec امنیت ارتباطات را در بطن شبکه با کمک سرویس های امن رمزنگاری برقرار می کند. برای عملکرد صحیح و کامل IPSec، هر دو طرف فرستنده و گیرنده باید یک کلید عمومی را به اشتراک بگذارند که بواسطه استفاده از پروتکل "مدیریت کلید" عملی می شود. این پروتکل به گیرنده این اجازه را می دهد تا یک کلید عمومی را بدست آورده و فرستنده را بر اساس امضای دیجیتال احراز هویت نماید. همچنین میتوان با استفاده از pre-shared-key یا همان کلید های از پیش به اشتراک گذاشته شده استفاده کرد . مزایایی که IPSec برای یک ارتباط به ارمغان می آورد، شامل موارد زیر می باشد:

- محافظت از حمله replay
- محرمانگی اطلاعات (رمزنگاری)
- یکپارچکی اطلاعات
- احراز منبع و منشاء اطلاعات
- احراز هویت در لایه Network

2.2.9 نحوه پیکربندی

3.2.9 تنظیمات IKE (phase 1)

IKEv2(Internet Key Exchange version 2) پروتکلی است برای راه اندازی و تشکیل SA که جهت استفاده در تونل ipsec به کار گرفته می شود

IKEv2 dpd

dead peer detection . زمانی که بعد از آن طرف مقابل حذف شده و فرآیند ike بین دو طرف مجددا راه اندازی می شود

```
crypto ikev2 dpd (1-3600) [(1-100)]
```

عدد اول طول عمر را مشخص می کند و پارامتر دوم حداکثر تعداد تلاش ها برای حذف طرف مقابل را مشخص می کند

IKEv2 proposal

proposal مجموعه ای از الگوریتم های رمزنگاری برای تشکیل SA است

ساخت یک پروپوزال ike با نام IKEPOSAL

```
soodar(config)#crypto ikev2 proposal IKEPOSAL
```

مشخص کردن الگوریتم های رمز نگاری در پروپوزال

```
soodar(config-ikev2-proposal)# encryption ALGORITHM

soodar(config-ikev2-proposal)# integrity ALGORITHM
```

مشخص کردن یک گروه به عنوان گروه Diffie-Hellman

```
soodar(config-ikev2-proposal)# group GROUP
```

مثال :

```
soodar(config)# crypto ikev2 proposal proposal-exemplary
soodar(config-ikev2-proposal)# encryption aes-192
soodar(config-ikev2-proposal)# integrity sha1-96
soodar(config-ikev2-proposal)# group 14
```

IKEv2 keyring

همانطور که از این نام پیداست، مجموعه‌ای از PSKها(Pre-Shared Key) اینجا قرار داده میشود. این PSKها با بر اساس هویت(Identity) در صورت تنظیم شدن IKE برای استفاده از PSK مورد استفاده قرار میگیرد.

ساخت یک keyring جدید با نام IKEKEYRING

```
soodar(config)# crypto ikev2 keyring IKEKEYRING
```

ساخت peer جدید در keyring به نام PEER

```
soodar(config-ikev2-keyring)# peer PEER
```

تنظیم آدرس شبکه peer

```
soodar(config-ikev2-keyring-peer)# address <A.B.C.D|X:X::X:X>
```

تنظیم psk برای احراز هویت peer

```
soodar(config-ikev2-keyring-peer)# pre-shared-key LINE
```

استفاده از یک آدرس به عنوان peer id . این id در زمان برقراری نشست ike بین دو طرف استفاده می شود.

```
soodar(config-ikev2-keyring-peer)# identity address <A.B.C.D|X:X::X:X>
```

استفاده از یک FQDN به عنوان peer id . این id در زمان برقراری نشست ike بین دو طرف استفاده می شود.

```
soodar(config-ikev2-keyring-peer)# identity fqdn FQDN
```

استفاده از یک EMAIL آدرس به عنوان peer id . این id در زمان برقراری نشست ike بین دو طرف استفاده می شود.

```
soodar(config-ikev2-keyring-peer)# identity email MAIL
```

مثال:

```
soodar(config)# crypto ikev2 keyring keyring-1
soodar(config-ikev2-keyring)# peer PC-1
soodar(config-ikev2-keyring-peer)# address 192.168.1.100
soodar(config-ikev2-keyring-peer)# identity email soodar2@soodar.ir
soodar(config-ikev2-keyring-peer)# pre-shared-key 123@321
soodar(config-ikev2-keyring)# peer PC-2
soodar(config-ikev2-keyring-peer)# address 192.168.1.20
soodar(config-ikev2-keyring-peer)# identity address 1.1.1.1
soodar(config-ikev2-keyring-peer)# pre-shared-key ITSAHARDPASSWD!!
```

IKEv2 profile

profile بخش اصلی ike می باشد . تعریف یک آدرس به عنوان id محلی . در نشست ike توسط این id به طرف های مقابل شناسانده می شود

```
soodar(config-ikev2-profile)# identity local address <A.B.C.D|X:X::X:X>
```

تعریف یک FQDN به عنوان id محلی . در نشست ike توسط این id به طرف های مقابل شناسانده می شود

```
soodar(config-ikev2-profile)# identity local fqdn FQDN
```

تعریف یک EMAIL به عنوان id محلی . در نشست ike توسط این id به طرف های مقابل شناسانده می شود

```
soodar(config-ikev2-profile)# identity local email MAIL
```

id طرف مقابل را به شکل address وارد کنید بقیه اطلاعات از keyring استخراج خواهد شد .

```
soodar(config-ikev2-profile)# match identity remote address <A.B.C.D|X:X::X:X>
```

id طرف مقابل را به شکل FQDN وارد کنید بقیه اطلاعات از keyring استخراج خواهد شد .

```
soodar(config-ikev2-profile)# match identity remote fqdn FQDN
```

id طرف مقابل را به شکل EMAIL وارد کنید بقیه اطلاعات از keyring استخراج خواهد شد .

```
soodar(config-ikev2-profile)# match identity remote email EMAIL
```

مشخص کردن keyring برای پیداکردن peer مربوطه برای برقراری نشست ike

```
soodar(config-ikev2-profile)# keyring local IKEKEYRING
```

استفاده از IKEPOSAL برای تشکیل IKEv2 SA

```
soodar(config-ikev2-profile)# proposal IKEPOSAL
```

مثال :

```
soodar(config)# crypto ikev2 profile VPN
soodar(config-ikev2-profile)# identity local address 192.168.1.1
soodar(config-ikev2-profile)# match identity remote soodar1@soodar.ir
soodar(config-ikev2-profile)# keyring local keyring-1
soodar(config-ikev2-profile)# proposal proposal-exemplary
```

پس از تنظیم ike می توان آن را در پروفایل ipsec استفاده کرد : مثال:

```
soodar(config)# crypto ipsec profile ipsec-Site2-profile
soodar(ipsec-profile)#  set transform-set ipsec-Site2-TS
soodar(ipsec-profile)#  set ikev2-profile VPN
```

4.2.9 تنظیمات IPSec (phase 2)

تنظیمات ipsec شامل دو قسمت است . یکی تنظیم Transform Set و بخش دوم تنظیم IPSec profile :

IPSec Transform-set

یک TS، شامل تنظیمات پایه‌ای برای برقراری ارتباط بین دو سر IPSec میباشد که در واقع شامل تنظیمات الگوریتم‌های رمزنگاری می باشد.

نکته

اگر transform-set تنظیم نشود ، از transform-set پیش فرض استفاده می شود . که یک سری الگوریتم های رمزنگاری را به طرف مقابل پیشنهاد می کند و در صورت توافق طرفین تونل وصل می شود .

IPSec Profile

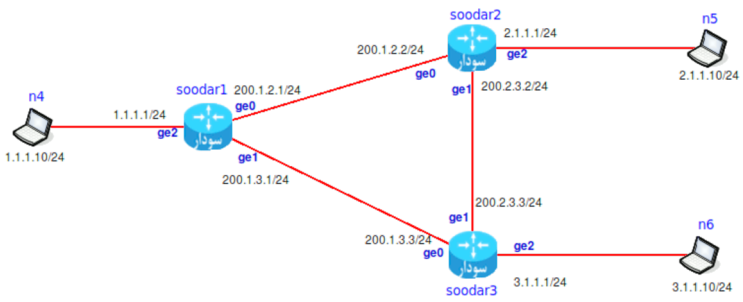
در این پروفایل TS تنظیم شده برای تونل انتخاب می شود و همچنین ike profile که قبلا تنظیم شده برای برقرای ارتباط ike انتخاب می شود . پس از آن این پروفایل به تونل اعمال می شود و تونل با استفاده از ipsec محافظت شده و ترافیک آن رمز شده انتقال می باید .

نکته

در روتر سودار حالت تونل برای رمز شدن ترافیک پشتیبانی می شود و حالت استفاده از crypto-map که ترافیک خاصی برای رمز شدن انتخاب می شود فعلا پشتیبانی نمی شود .

5.2.9 مثال عملی با استفاده از PSK

حال فرض کنید قصد داریم تونل GRE که در بخش کارگاه gre اضافه کردیم را با IPsec حفاظت کرده و رمز کنیم . ابتدا به تنظیم روتر Soodar1 میپردازیم:



تنظیمات ike (phase 1)

soodar1 ike proposal

```
soodar1(config)# crypto ikev2 proposal IKE-PP
soodar1(config-ikev2-proposal)# encryption aes-192
soodar1(config-ikev2-proposal)# integrity sha1-96
soodar1(config-ikev2-proposal)# group 14
```

soodar1 ike keyring

```
soodar1(config)# crypto ikev2 keyring KEYRING-1
soodar1(config-ikev2-keyring)# peer SOODAR2
soodar1(config-ikev2-keyring-peer)# address 200.1.2.2
soodar1(config-ikev2-keyring-peer)# identity address 222.2.2.2
soodar1(config-ikev2-keyring-peer)# pre-shared-key 123@321
```

soodar1 ike profile

```
soodar1(config)# crypto ikev2 profile VPN
soodar1(config-ikev2-profile)# identity local address 222.1.1.1
soodar1(config-ikev2-profile)# match identity remote address 222.2.2.2
soodar1(config-ikev2-profile)# keyring local KEYRING-1
soodar1(config-ikev2-profile)# proposal IKE-PP
```

تنظیمات ipsec (phase 2)

soodar1 Transform-set

```
soodar1(config)# crypto ipsec transform-set IPSEC-SITE1-TS esp hmac sha-256 cipher aes-256
soodar1(cfg-crypto-trans)# mode transport
```

soodar1 ipsec profile

```
soodar1(config)# crypto ipsec profile IPSEC-SITE1-PROFILE
soodar1(ipsec-profile)# set transform-set IPSEC-SITE1-TS
soodar1(ipsec-profile)# set ikev2-profile VPN
```

ساخت تونل در soodar1

```
soodar1(config)# interface tunnel 12
soodar1(config-if)# tunnel source 200.1.2.1
soodar1(config-if)# tunnel destination 200.1.2.2
soodar1(config-if)# tunnel protection ipsec profile IPSEC-SITE1-PROFILE
soodar1(config-if)# ip address 10.0.0.12/32
```

اضافه کردن route از تونل

```
soodar1(config)# ip route 2.1.1.0/24 tunnel12
```

تنظیمات در **soodar1** به پایان رسید . تنظیمات در soodar2 نیز به همین شکل است .

soodar2 ike proposal

```
soodar2(config)# crypto ikev2 proposal IKE-PP-2
soodar2(config-ikev2-proposal)# encryption aes-192
soodar2(config-ikev2-proposal)# integrity sha1-96
soodar2(config-ikev2-proposal)# group 14
```

soodar2 ike keyring

```
soodar2(config)# crypto ikev2 keyring KEYRING-2
soodar2(config-ikev2-keyring)# peer SOODAR1
soodar2(config-ikev2-keyring-peer)# address 200.1.2.1
soodar2(config-ikev2-keyring-peer)# identity address 222.1.1.1
soodar2(config-ikev2-keyring-peer)# pre-shared-key 123@321
```

soodar2 ike keyring

```
soodar2(config)# crypto ikev2 profile VPN-2
soodar2(config-ikev2-profile)# identity local address 222.2.2.2
soodar2(config-ikev2-profile)# match identity remote address 222.1.1.1
soodar2(config-ikev2-profile)# keyring local KEYRING-2
soodar2(config-ikev2-profile)# proposal IKE-PP-2
```

soodar2 Transform-set

```
soodar2(config)# crypto ipsec transform-set IPSEC-SITE2-TS esp hmac sha-256 cipher aes-256
soodar2(cfg-crypto-trans)# mode transport
```

soodar2 ipsec profile

```
soodar2(config)# crypto ipsec profile IPSEC-SITE2-PROFILE
soodar2(ipsec-profile)# set transform-set IPSEC-SITE2-TS
soodar2(ipsec-profile)# set ikev2-profile VPN-2
```

ساخت تونل در **soodar2**

```
soodar2(config)# interface tunnel 21
soodar2(config-if)# tunnel source 200.1.2.2
soodar2(config-if)# tunnel destination 200.1.2.1
soodar2(config-if)# tunnel protection ipsec profile IPSEC-SIT2-PROFILE
soodar2(config-if)# ip address 10.0.0.21/32
```

اضافه کردن **route** از تونل

```
soodar2(config)# ip route 1.1.1.0/24 tunnel21
```

6.2.9 مشاهده SA

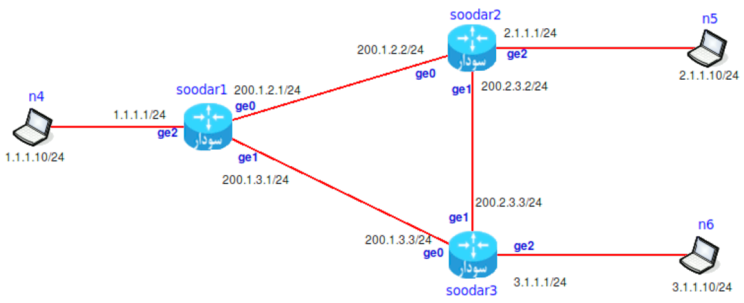
```
soodar# show crypto ikev2 sa
soodar# show crypto ipsec sa
```

7.2.9 بررسی عملکرد تونل

اولا باید ارتباط تونل gre برقرار باشد و نود های 1.1.1.10 و 2.1.1.10 باید یکدیگر را ببینند در ثانی با بررسی ترافیک خروجی روی تونل gre باید مشخص شود که ترافیک به صورت رمز شده و با استفاده از ipsec در حال انتقال است .

8.2.9 مثال عملی با استفاده از RSA

همان سناریو قبل را مجددا در نظر بگیرید :



در ابتدا لازم است که PKI در هر دو روتر پیکربندی شود. با فرض وجود یک سرور خارجی امن که ادمین به آن دسترسی دارد و کلید CA در آن قرار دارد و میتواند با آن گواهی صادر کند، در روتر soodar1 یک کلید RSA به عنوان کلید خصوصی خود و trustpoint ایجاد می‌نماییم:

```
soodar1(config)# crypto key generate rsa label soodar1-key
soodar1(config)# crypto pki trustpoint soodar1-cert
```

در تنظیمات Trustpoint، اطلاعات مربوط به گواهینامه روتر soodar1 را تنظیم میکنیم:

```
soodar1(ca-trustpoint)# rsakeypair soodar1-key
soodar1(ca-trustpoint)# subject-name C=IR, O=Soodar, CN=s1.local.net
soodar1(ca-trustpoint)# subject-alt-name soodar1.local.net
soodar1(ca-trustpoint)# subject-alt-name soodar1.afta.net
```

سپس این TP را Authenticate کرده و CA Certificate سرور خارجی را به روتر می‌افزاییم:

```
soodar1(config)# crypto pki authenticate soodar1-cert
```

پس از اینکه CA به TP افزوده شد، میتوانیم یک CSR ایجاد نماییم:

```
soodar1(config)# crypto pki enroll soodar1-cert
```

خروجی دستور را کپی کرده و توسط سرور خارجی CA گواهینامه را امضا و صادر نمایید(خروجی به فرمت PKCS#10 میباشد). گواهینامه امضا شده را به دستگاه وارد نمایید:

```
soodar1(config)# crypto pki import soodar1-cert certificate
```

نکته: اگر گواهینامه‌ی هر دو روتر soodar1 و soodar2 توسط یک CA امضا میشوند، نیازی به وارد کردن CA دیگری نیست. اما اگر روتر soodar2 از CA دیگری استفاده میکند، تنها کافیست این CA را به روتر به روش زیر اضافه کنیم:

```
soodar1(config)# crypto pki trustpoint soodar2-ca
soodar1(config)# crypto pki authenticate soodar2-ca
```

مراحل بالا را نیز برای روتر soodar2 انجام میدهیم:

```
soodar2(config)# crypto key generate rsa label soodar2-key
soodar2(config)# crypto pki trustpoint soodar2-cert
soodar2(ca-trustpoint)# rsakeypair soodar2-key
soodar2(ca-trustpoint)# subject-name C=IR, O=Soodar, CN=s2.local.net
soodar2(ca-trustpoint)# subject-alt-name soodar2.local.net
soodar2(ca-trustpoint)# subject-alt-name soodar2.afta.net
soodar2(config)# crypto pki enroll soodar2-cert
soodar2(config)# crypto pki import soodar2-cert certificate
```

و در صورت لازم، CA مربوط به گواهینامه soodar1 را می افزاییم:

```
soodar2(config)# crypto pki trustpoint soodar1-ca
soodar2(config)# crypto pki authenticate soodar1-ca
```

حال در روتر soodar1 تنظیمات IKEv2 را اعمال میکنیم. در ابتدا یک پروپوزال تعریف میکنیم:

```
soodar1(config)# crypto ikev2 proposal ikeProposal
soodar1(config-ikev2-proposal)# integrity sha-384
soodar1(config-ikev2-proposal)# encryption aes
soodar1(config-ikev2-proposal)# group 28
```

سپس پروفایل IKEv2 را میسازیم:

```
soodar1(config)# crypto ikev2 profile ikeProfile
soodar1(config-ikev2-profile)# authentication local rsa-sig
soodar1(config-ikev2-profile)# authentication remote rsa-sig
soodar1(config-ikev2-profile)# proposal ikeProposal
```

با توجه به اینکه در گواهینامه‌ی کاربردی‌ای که برای روتر soodar1 صادر شده است، در بخش SAN آدرس‌های soodar1.local.net و soodar1.afta.net ذکر شده است، نیاز است که حتما از یکی از این آدرس‌ها به عنوان Local Identity استفاده گردد. و همین قاعده برای Remote identity نیز برقرار است:

```
soodar1(config-ikev2-profile)# identity local fqdn soodar1.local.net
soodar1(config-ikev2-profile)# match identity remote fqdn soodar2.local.net
```

اکنون Transform Set و IPSec Profile را تشکیل میدهیم:

```
soodar1(config)# crypto ipsec transform-set ts esp hmac sha-96 cipher aes
soodar1(cfg-crypto-trans)# mode transport

soodar1(config)# crypto ipsec profile ipsecProfile
soodar1(ipsec-profile)# set transform-set ts
soodar1(ipsec-profile)# set ikev2-profile ikeProfile
```

در آخرین مرحله نیاز است یک تونل بین دو دستگاه ساخته شود و توسط این پروفایل IPSec محافظت گردد. سپس ترافیک مورد نظر را از طریق این تونل Route کرده تا مطمئن شویم ارتباط امن صورت میپذیرد:

```
soodar1(config)# interface tunnel 10
soodar1(config-if)# tunnel source 200.1.2.1
soodar1(config-if)# tunnel destination 200.1.2.2
soodar1(config-if)# tunnel protection ipsec profile ipsecProfile
soodar1(config-if)# ip address 10.1.1.1/32
soodar1(config)# ip route 2.1.1.0/24 tunnel10
```

متناظر این دستورات در روتر دیگر نیز انجام میگیرد:

```
soodar2(config)# crypto ikev2 proposal ikeProposal
soodar2(config-ikev2-proposal)# integrity sha-384
soodar2(config-ikev2-proposal)# encryption aes
soodar2(config-ikev2-proposal)# group 28
soodar2(config)# crypto ikev2 profile ikeProfile
soodar2(config-ikev2-profile)# authentication local rsa-sig
soodar2(config-ikev2-profile)# authentication remote rsa-sig
soodar2(config-ikev2-profile)# proposal ikeProposal
soodar2(config-ikev2-profile)# identity local fqdn soodar2.local.net
soodar2(config-ikev2-profile)# match identity remote fqdn soodar1.local.net
soodar2(config)# crypto ipsec transform-set ts esp hmac sha-96 cipher aes
soodar2(cfg-crypto-trans)# mode transport
soodar2(config)# crypto ipsec profile ipsecProfile
soodar2(ipsec-profile)# set transform-set ts
soodar2(ipsec-profile)# set ikev2-profile ikeProfile
soodar2(config)# interface tunnel 10
soodar2(config-if)# tunnel source 200.1.2.2
soodar2(config-if)# tunnel destination 200.1.2.1
soodar2(config-if)# tunnel protection ipsec profile ipsecProfile
soodar2(config-if)# ip address 10.1.1.2/32
soodar2(config)# ip route 1.1.1.0/24 tunnel10
```

جال باید ارتباط بین دو نود 1.1.1.10 و 2.1.1.10 برقرار باشد و ترافیک بین دو روتر از طریق تونل و به صورت رمز شده انتقال یابد

9.2.9 فعال کردن log های IPSec

با دستور زیر می توانید Log های مربوط به ipsec را فعال کنید:

```
soodar# debug ipsec event
```

10.2.9 PKI

در ادامه توضیحاتی در مورد کلید های pki در روتر ارایه می دهیم :

ایجاد کلید خصوصی: کلید خصوصی برای ایجاد « درخواست امضای گواهی» (Certificate Signing Request) و امضای دیجیتال به کار میرود. دستور زیر کلیدی با نام NAME را میسازد:

```
soodar(config)# crypto key generate rsa label <NAME> modulus (2048-4096)
```

در صورتی که طول کلید مشخص نگردیده باشد، به طور پیش فرض طول 2048 برای کلید در نظر گرفته میشود. وارد کردن یک CA برای وارد کردن CA ما ابتدا نیاز به ساختن یک trustpoint داریم. یک trustpoint همان CA هست که میتواند یک Certificate امضا شده توسط همان CA را نیز شامل شود(مانند مفهوم Trustpoint در روترهای سیسکو). این CA میتواند یک Self-signed CA باشد.

نکته: تمامی ورودی‌ها/خروجی‌ها(اعم از CA، Certificate، CSR و ...) به فرمت PEM می‌باشد.

برای ساختن یک trustpoint با نام TP میتوان از دستور زیر استفاده نمود:

```
soodar(config)# crypto pki trustpoint TP
```

پس از ساختن یک trustpoint در ابتدا نیاز است این trustpoint را authenticate کرد و یک CA صحیح را به آن نسبت داد(وارد کرد). اگر یک trustpoint به یک CA صحیح منسوب نباشد و authenticate نشده باشد، نمیتوان با آن گواهی ای را وارد کرد.

با دستور زیر میتوان یک trustpoint را authenticate کرد:

```
soodar(config)# crypto pki trustpoint TP
```

بطور مثال:

```
soodar(config)# crypto pki trustpoint root-ca
soodar(config)# crypto pki authenticate root-ca
Enter the base 64 encoded CA certificate
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
  MIIDSzCCAjOgAwIBAgIIQMT8Qv03sXYwDQYJKoZIhvcNAQELBQAwNTELMAkGA1UE
  BhMCSVIxEzARBgNVBAoTClRlbXAgQ29ycC4xETAPBgNVBAMTCHRlbXAuY29tMB4X
  DTIxMDEyMDExNDIzNFoXDTI0MDEyMDExNDIzNFowNTELMAkGA1UEBhMCSVIxEzAR
  BgNVBAoTClRlbXAgQ29ycC4xETAPBgNVBAMTCHRlbXAuY29tMIIBIjANBgkqhkiG
  9w0BAQEFAAOCAQ8AMIIBCgKCAQEAy1KPgdCS6BB7PCdeggnsf6NjW4KBxeG6H18R
  lOHYoTBMlR3QrvCrpgoZv3DtGR8T6Ch0/HdL1GdFJ7RcJqZPbaxdepqI08SZG4VD
  CcZbIOdCNgKWD+jaO0vgyfcK2cXKY70bdyUuJLwNvSvPEPhzH1UNx7kfBdvGn2Vg
  s/XyYhsn3xc6ioODT+HUAAd2WvBIOzd+RUo0yANJRKbPnLPgpNEiE1wG6Bj6orjR
  ajnC8SYt5XGqD0DX7JGi7bELHw0JGdDk1acr9GQyjwVobDYCKDTuW4ELDsS+2GIK
  E76rmlAGrJGy3po2itVbmMprhbTl3EOpxPzl78qkG/r0i4lUXQIDAQABo18wXTAP
  BgNVHRMBAf8EBTADAQH/MA4GA1UdDwEB/wQEAwIBBjAdBgNVHQ4EFgQU7CsuL8vJ
  o0kfANvQjVQkaR4K/WQwGwYDVR0RBBQwEoIQb3RoZXIuZG9tYWluLmNvbTANBgkq
  hkiG9w0BAQsFAAOCAQEAHe8iOUjW8+BNBCfyyfcQOokd7UuK/0DE40wEXVRpMzyv
  4IoLNnz5SmWBZo5WdtkIUfGMc9ll8uRsBpIcqHOR8ZSRkjswtOFn+C5KxNXum1pQ
  cLmNpxn2ecsr2K2qW6IRfig8cQwzpFe3c59zFf13gKdr6g0B+lpx/hMBdhyaUn6A
  9uXtvgeCzAqdJehpo12IKNnYeL+GrfHcFe7R7BRLD2XzoAgjFR48w24h3FbrxM8I
  1jqEwbvnGT7FECGZbyKGBEM/dY1gbVD19GTJlaZ8z3HrHdaRFvCYgAqFLTVtU8Q+
  lq+EWiCSMRlPPx1OiLDddbxRw2JIjdF7XIsU3WGhtw==
-----END CERTIFICATE-----
Updating certificates in /etc/ssl/certs...
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

```
soodar# show crypto pki certificate root-ca
Trustpoint: root-ca
CA
  subject:  "C=IR, O=Temp Corp., CN=temp.com
  issuer:   "C=IR, O=Temp Corp., CN=temp.com
  validity:  not before Jan 20 15:12:34 2021, not valid yet (valid in 58 seconds)
             not after  Jan 20 15:12:34 2024, ok (expires in 1095 days)
  serial:    40:c4:fc:42:fd:37:b1:76
  altNames:  other.domain.com
  flags:     CA CRLSign self-signed
  subjkeyId: ec:2b:2e:2f:cb:c9:a3:49:1f:00:db:d0:8d:54:24:69:1e:0a:fd:64
   pubkey:    RSA 2048 bits
   keyid:     cf:d8:04:82:62:b9:f1:a9:84:75:56:e7:1b:5b:ac:4a:c8:ba:ae:21
```

(continues on next page)

(continued from previous page)

```
subjkey:   ec:2b:2e:2f:cb:c9:a3:49:1f:00:db:d0:8d:54:24:69:1e:0a:fd:64
Fingerprint: 954E9105EEE221C7BCDF351BBA0184E950F82C75
```

ایجاد یک Certificate و CSR: کاربر میتواند یک « درخواست امضای گواهی» را ایجاد کرده و گواهی امضا شده را به دستگاه وارد کند. برای این کار، trustpoint در تنظیمات خود باید موارد زیر را داشته باشد: یک SN برای گواهی(و به شکل دلبخواه، تعدادی SAN) یک جفت کلید RSA برای امضای CSR یک روش enrollment. در حال حاضر تنها روش enrollment از طریق ترمینال و به شیوه کپی/پیست میباشد و میتوان تنظیم نکرد.

پس از تنظیم کردن trustpoint یک CSR میتواند ایجاد شود. CSR ایجاد شده به فرمت PKCS#10 میباشد و بر روی ترمینال(با توجه به روش enrollment که در حال حاضر فقط از طریق ترمینال میباشد) چاپ میشود. کاربر میتواند CSR چاپ شده را کپی کرده و با کلید خصوصی CA امضا کرده و Certificate امضا شده را به دستگاه وارد کند.

نکته: اگر trustpoint از قبل authenticate نشده باشد، نمیتوان Certificate امضا شده را به دستگاه وارد کرد. چرا که نیاز است حتما این گواهی با CA مرتبط با trustpoint امضا شده باشد.

دستورات زیر برای تنظیم trustpoint میباشد:

```
soodar(ca-trustpoint)# subject-name <SN LINE>
soodar(ca-trustpoint)# subject-alt-name <SAN>
soodar(ca-trustpoint)# rsakeypair <KEY>
```

برای ایجاد یک CSR باید trustpoint را enroll کنیم:

```
soodar(config)# crypto pki enroll <TP>
```

دستور بالا CSR را ایجاد میکند و بر روی صفحه نمایش میدهد.

پس از امضا شدن CSR توسط CA و ایجاد گواهی معتبر، میتوان این گواهی را به دستگاه افزود:

```
soodar(config)# crypto pki import <TP> certificate
```

بطور مثال:

```
n1(config)# crypto key generate rsa label mycert-key modulus 2048
n1# show crypto key mycert-key
Keypair Label: mycert-key
  Algorithm:   RSA
  Modulus:     2048 bits
  Subject key: fcc893035eda7e736d0a612bad1d000612c87724
  Key ID:      E5611192FEAD3FDFA877A0BAC5F336480A8C2D97

n1(config)# crypto pki trustpoint mycert
n1(ca-trustpoint)# subject-name C=IR, O=My Org, CN=my.org
n1(ca-trustpoint)# subject-alt-name other.my.org
n1(ca-trustpoint)# subject-alt-name other2.my.com
n1(ca-trustpoint)# rsakeypair mycert-key
```

```
n1(config)# crypto pki authenticate mycert
Enter the base 64 encoded CA certificate
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIIM7DVFqEvgxgwDQYJKoZIhvcNAQELBQAwOjELMAkGA1UE
BhMCQ0gxEzARBgNVBAoTCnN0cm9uZ1N3YW4xFjAUBgNVBAMTDXN0cm9uZ1N3YW4g
Q0EwHhcNMjAxMTExMDk1OTEzWhcNMjMxMTExMDk1OTEzWjA6MQswCQYDVQQGEwJD
SDETMBEGA1UEChMKc3Ryb25nU3dhbjEWMBQGA1UEAxMNc3Ryb25nU3dhbiBDQTCC
ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANjhd9ZFsCS4O3TcnXWFy/cr
wXnVCxev6g5XecHG0A+jaOS6MyJowjJU/CY5S8/LWKIBlKFhdswDT0LaPodnKw8e
```

(continues on next page)

(continued from previous page)

```
RVGwAfQSYb8OymUeHByzxxfhqcCjYu0qWdb2Tf9yVadkt//qW5n2F78j3prFlZ4o
pbG1sLhACY+729iJxB7dg5DKXxECBzSiMo2dScZpQKuADiev4g7TmEH0u3MUa9zU
CzIhoqjzEJ1wF4YC7Y6BZxQU4c04RZGctaOmKRUT0NfVGbqseJHsJVZSCDFud/ls
48tDmQ08GULFNFlFAeGWcUnLle2sorsB+zjfQrJQJBtE/RuoKZ3ODK+ZwGH8wHEC
AwEAAaNCMEAwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMCAQYwHQYDVR0O
BBYEFNET3aeJu4082kUYI8TpeBK4w61sMA0GCSqGSIb3DQEBCwUAA4IBAQCB2ciJ
D197+CIwL/DveAJf7Bt0cMD2lPwY4hsHUyHridX2B/t6EMOOujWPouSeBYjLBz7s
akHwh3G9Yx4wlS+k+du5AbkQHMnYigeO4rul+tCg7FzouxFtKEcD6T707DnSEkP+
iA9mLeKxCK3P4vGY2H9x6McqZ1aM55xmdEbvD3QhUMLePBk4aMVKyOr4yWRQgUPB
oBqRVSEvthOyXEWtPkqxY72O/5IQmHDSncBP/D+wiC2wQsYQZhmDoN6d74OqkcBr
HMWDCUM1b8RfVBTeIKvkvQ14BgwPveO99E+P6rrNhdxRA8BwmnNyMvrd81Z1FDU/
J+XkIuPRfz33vO00
-----END CERTIFICATE-----
Updating certificates in /etc/ssl/certs...
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.

n1(ca-trustpoint)# enrollment terminal pem
```

```
n1(config)# crypto pki enroll mycert
-----BEGIN CERTIFICATE REQUEST-----
MIICrTCCAZUCAQAwLzELMAkGA1UEBhMCSVIxDzANBgNVBAoTBk15IE9yZzEPMA0G
A1UEAxMGbXkub3JnMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtvWm
Xi+CtKrJndPw09hPOnTO8DSDIJqi3GdcNDVRcdKb/FB+/C++Vyb2vOLNICxCmRJH
RnoZKPnWqRHWyHeVCNr+Da+bFYHXd4LyaZtCzEoUrmULMyBWGmbUfUlfFpOCa4yq
28qV1BjYXEm93X56XIaT/WpqXELihJC2nnBPxhkLHA80fLmQPZdOzytrjeJt1Rvn
I/PpI+OzEN9/pUvGLv29wfzUN2T9WGdIY/SJuyafQ2972juRA2OTTSsMSOxM4fuj
Mk116RixYvHCd454gehPKOqMUHbXKZ7tQXPaDFtiQIgNqBMz4AlT40Wn3GsODV8Y
AtJ9UOvhmMW1iTHC2wIDAQABoDkwNwYJKoZIhvcNAQkOMSowKDAmBgNVHREEHzAd
ggxvdGhlci5teS5vcmeCDW90aGVyMi5teS5jb20wDQYJKoZIhvcNAQELBQADggEB
AKwvB+bPTMpU2t3HE6CA0mLA9ufc9EqWx2YCTyddTJ8Qp7xhdXywzB64R5Um/mqy
x7lMEyS69pZzTMivm28piIEplSdjKSiHmRpVZsXGWvhpz1alqA6h5IaWlm9s3Bga
YKBmaC0uEsuhXnAxFBPtbwWSaGN0uD5kKTkwZXMxKv4gVkTbrdZfZ2uJR2CiZu1q
yb7u47MeZF4xfcnvFZCuUjlLmpFXMLXjYuNywJP6U/i1DpSG07mDYcnEfS9Ku/o/
gdNBahSspRtBVOx4QtnN4bGZ0MDEn5cEBuWcN4dNbE30dn70NKaNe1DhdKQ/lUxQ
qyIP+5tc2i8GoJsL9wyWJIo=
-----END CERTIFICATE REQUEST-----
```

```
n1(config)# crypto pki import mycert certificate
Enter the base 64 encoded CA certificate
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDMTCCAhmgAwIBAgIIVmyRIVfPsKowDQYJKoZIhvcNAQELBQAwNTELMAkGA1UE
BhMCSVIxEzARBgNVBAoTClRlbXAgQ29ycC4xETAPBgNVBAMTCHRlbXAuY29tMB4X
DTIxMDEyMDExNDgzNloXDTI0MDEyMDExNDgzNlowLzELMAkGA1UEBhMCSVIxDzAN
BgNVBAoTBk15IE9yZzEPMA0GA1UEAxMGbXkub3JnMIIBIjANBgkqhkiG9w0BAQEF
AAOCAQ8AMIIBCgKCAQEAtvWmXi+CtKrJndPw09hPOnTO8DSDIJqi3GdcNDVRcdKb
/FB+/C++Vyb2vOLNICxCmRJHRnoZKPnWqRHWyHeVCNr+Da+bFYHXd4LyaZtCzEoU
rmULMyBWGmbUfUlfFpOCa4yq28qV1BjYXEm93X56XIaT/WpqXELihJC2nnBPxhkL
HA80fLmQPZdOzytrjeJt1RvnI/PpI+OzEN9/pUvGLv29wfzUN2T9WGdIY/SJuyaf
Q2972juRA2OTTSsMSOxM4fujMk116RixYvHCd454gehPKOqMUHbXKZ7tQXPaDFti
QIgNqBMz4AlT40Wn3GsODV8YAtJ9UOvhmMW1iTHC2wIDAQABo0swSTAfBgNVHSME
GDAWgBTsKy4vy8mjSR8A29CNVCRpHgr9ZDAmBgNVHREEHzAdggxvdGhlci5teS5v
cmeCDW90aGVyMi5teS5jb20wDQYJKoZIhvcNAQELBQADggEBAGbt3R0FyA48FWUh
```

(continues on next page)

(continued from previous page)

```
eoud1zh6ujrg0PgFjOhAMnWaln8nXdhMjJJvOI/MZtcyl7fghXr1Asr2M9I3KMxh
BbBefCci5+94g+QucP/R0v5/fzFpiV8gRYXD8o7UWyYanQG5SUyTCdpR5vXxVbEW
FXp3Yk1HBYXDe09AK9AGwRVFHTkaaPze8U5FyJpbrjDZuD/cbkN4lFn+lw49JahO
cVqYXyY84rHjvbq98081NsittSa4QUqBNo8nUXYj+yLuNiV39Zh1pWzl/kugy0yR
mvrqC3irZGXeJbSLDaAT1LdJhiu2Axc7EjwKxcNK+GiXyN/B/7JJrWLL0u6xaA9L
ezbvqQw=
-----END CERTIFICATE-----
Installed successfully
```

```
n1# show crypto pki certificate mycert
Trustpoint: n1Cert
CA:
  subject:  "C=IR, O=Temp Corp., CN=temp.com"
  issuer:   "C=IR, O=Temp Corp., CN=temp.com"
  validity:  not before Jan 20 15:12:34 2021, ok
             not after  Jan 20 15:12:34 2024, ok (expires in 1094 days)
  serial:    40:c4:fc:42:fd:37:b1:76
  altNames:  other.domain.com
  flags:     CA CRLSign self-signed
  subjkeyId: ec:2b:2e:2f:cb:c9:a3:49:1f:00:db:d0:8d:54:24:69:1e:0a:fd:64
  pubkey:    RSA 2048 bits
  keyid:     cf:d8:04:82:62:b9:f1:a9:84:75:56:e7:1b:5b:ac:4a:c8:ba:ae:21
  subjkey:   ec:2b:2e:2f:cb:c9:a3:49:1f:00:db:d0:8d:54:24:69:1e:0a:fd:64
  Fingerprint: 954E9105EEE221C7BCDF351BBA0184E950F82C75

General Purpose Certificate:
  subject:  "C=IR, O=My Org, CN=my.org"
  issuer:   "C=IR, O=Temp Corp., CN=temp.com"
  validity:  not before Jan 20 15:18:36 2021, ok
             not after  Jan 20 15:18:36 2024, ok (expires in 1094 days)
  serial:    56:6c:91:21:57:cf:b0:aa
  altNames:  other.my.org, other2.my.com
  flags:
  authkeyId: ec:2b:2e:2f:cb:c9:a3:49:1f:00:db:d0:8d:54:24:69:1e:0a:fd:64
  subjkeyId: fc:c8:93:03:5e:da:7e:73:6d:0a:61:2b:ad:1d:00:06:12:c8:77:24
  pubkey:    RSA 2048 bits
  keyid:     e5:61:11:92:fe:ad:3f:df:a8:77:a0:ba:c5:f3:36:48:0a:8c:2d:97
  subjkey:   fc:c8:93:03:5e:da:7e:73:6d:0a:61:2b:ad:1d:00:06:12:c8:77:24
  Keypair:   mycert-key
  Fingerprint: D51636591648DBDE21FEEFA4C6DF4B38A96502B5
```

حذف کلید خصوصی: کاربر میتواند کلیدهای بدون استفاده را حذف کند. حذف کلیدها به شکل امن ابتدا با صفر کردن محتوای کلید در حافظه و سپس حذف فایل انجام میگردد.

```
soodar(config)# crypto key zeroize <RSAKEY>
```

```
soodar(config)# no crypto pki trustpoint <TP>
```

توجه

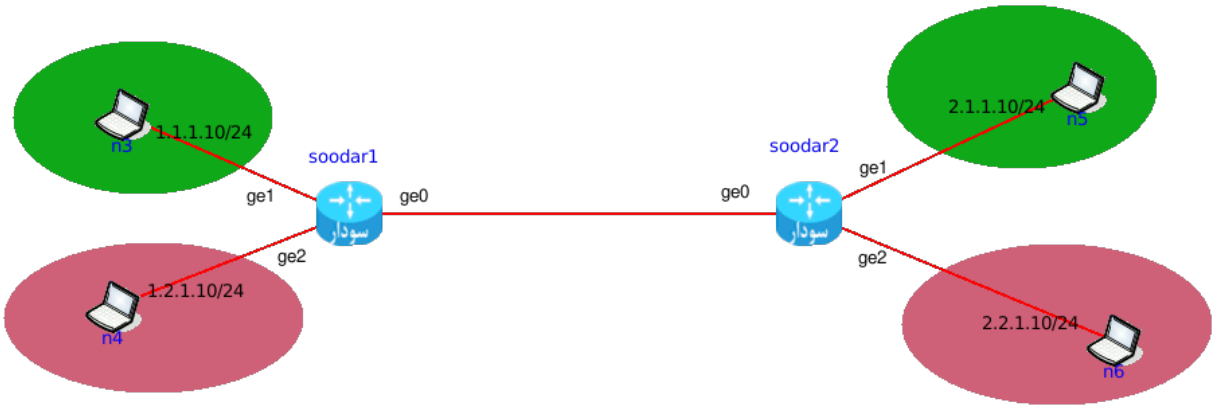
1. حذف کردن یک کلید خصوصی ممکن است باعث از کار افتادن Certificate هایی شود. وظیفه کاربر است که از بی خطر بودن این حذف اطمینان حاصل کند.
2. کاربر میتواند یک Trustpoint را حذف کند. حذف یک Trustpoint باعث حذف شدن CA آن و گواهی همه منظوره متناظرش از سیستم میشود.

---

wireguard normal 3.9

1.3.9 آموزش راه اندازی تونل wireguard در سودار

فرض کنید در سناریو زیر قصد داریم دو تونل wireguard بزنیم که دو شبکه پشتی هر روتر در vrf جداگانه قرار داشته باشند و هر کدام از تونل مربوط به خود برای ارتباط استفاده کنند و هیچ ارتباطی به شبکه دیگر نتواند برقرار کند :



ابتدا در هر روتر 3 vrf جداگانه می سازیم . یک vrf برای شبکه بین روتر ها (اینترنت یا public) و برای هر شبکه پشت روتر هم یک vrf جداگانه می سازیم :

soodar1

```
soodar1(config)# vrf pub
soodar1(config)# vrf green
soodar1(config)# vrf red
```

soodar2

```
soodar2(config)# vrf pub
soodar2(config)# vrf green
soodar2(config)# vrf red
```

اینترفیس ها را در vrf مربوطه قرار می دهیم (در اینجا pub برای شبکه public و دو vrf دیگر برای شبکه private استفاده می شوند):

soodar1

```
soodar1(config)# interface ge0
soodar1(config-if)# ip vrf forwarding pub
soodar1(config-if)# ip address 200.1.2.1/24
soodar1(config-if)# q
soodar1(config)# interface ge1
soodar1(config-if)# ip vrf forwarding green
soodar1(config-if)# ip address 1.1.1.1/24
soodar1(config-if)# q
soodar1(config)# interface ge2
soodar1(config-if)# ip vrf forwarding red
soodar1(config-if)# ip address 1.2.1.1/24
soodar1(config-if)# q
```

soodar2

```
soodar2(config)# interface ge0
soodar2(config-if)# ip vrf forwarding pub
soodar2(config-if)# ip address 200.1.2.2/24
soodar2(config-if)# q
soodar2(config)# interface ge1
soodar2(config-if)# ip vrf forwarding green
soodar2(config-if)# ip address 2.1.1.1/24
soodar2(config-if)# q
soodar2(config)# interface ge2
soodar2(config-if)# ip vrf forwarding red
soodar2(config-if)# ip address 2.2.1.1/24
soodar2(config-if)# q
```

2.3.9 ساخت تونل **wireguard**

در این سناریو ما :

- نیاز به دو تونل wg داریم تا دو شبکه متفاوت در vrf جداگانه را به هم وصل کند .
- هر اینترفیس تونل باید در یک vrf قرار بگیرد .
- باید route هایی که از طریق هر تونل عبور کنند را مشخص کنیم
- به صورت پیش فرض از الگوریتم استاندارد خود wireguard برای رمزنگاری استفاده می شود اما در صورت نیاز می توان از الگوریتم بومی نیز استفاده کرد .

3.3.9 ساخت کلید wg

در ابتدا برای هر تونل یک کلید wg تولید می کنیم .

نکته

دقت شود که کلید حتما باید از نوع x25519 انتخاب گردد.

---

soodar1

```
soodar1(config)# crypto key generate x25519 label wg10
informational-ZEBRA: new X25519 key wg10 generated

soodar1(config)# crypto key generate x25519 label wg11
informational-ZEBRA: new X25519 key wg11 generated

soodar1(config-if)# do sh crypto key wg10
Keypair Label: wg10
  Algorithm:  X25519
  Public key: A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76

soodar1(config-if)# do sh crypto key wg11
Keypair Label: wg11
  Algorithm:  X25519
  Public key: 4161E811D7BEE4AA61E2079FAE1A76ED03E5F21CEC38B1A70D9C2781247D7D5A

soodar1(config-if)#
```

soodar2

```
soodar2(config)# crypto key generate x25519 label wg20
informational-ZEBRA: new X25519 key wg20 generated

soodar2(config)# crypto key generate x25519 label wg21
informational-ZEBRA: new X25519 key wg21 generated

soodar2(config)# do sh crypto key wg20
Keypair Label: wg20
  Algorithm:  X25519
  Public key: 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831

soodar2(config)# do sh crypto key wg21
Keypair Label: wg21
  Algorithm:  X25519
  Public key: 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
```

4.3.9 ساخت اینترفیس **wireguard**

- اینترفیس تونل باید در vrf ی قرار بگیرد که قرار است ترافیک شبکه private مربوط به همان vrf را از خود عبور دهد (ge1 , 1.1.1.1/24)
- کلیدی که قرار است برای این تونل استفاده شود را مشخص می کنیم . **public key** این کلید در تنظیمات تونل طرف مقابل استفاده می شود
- پورتی که روی آن گوش می کنیم و بسته ها را با آن پورت مقصد قبول می کنیم را هم مشخص می کنیم این پورت در طرف مقابل تونل در بخش **peer** ها باید مد نظر گرفته شود
- یک ip هم به اینترفیس اختصاص می دهیم

soodar1

```
soodar1(config)# interface wireguard10
soodar1(config-if)# ip vrf forwarding green
soodar1(config-if)# wireguard source 200.1.2.1
soodar1(config-if)# wireguard private-key wg10
soodar1(config-if)# wireguard port 1212
soodar1(config-if)# ip address 10.0.12.12/32
soodar1(config-if)# end
soodar1#
```

soodar2

```
soodar2(config)# interface wireguard20
soodar2(config-if)# ip vrf forwarding green
soodar2(config-if)# wireguard source 200.1.2.2
soodar2(config-if)# wireguard private-key wg20
soodar2(config-if)# wireguard port 2121
soodar2(config-if)# ip address 10.0.12.21/32
soodar2(config-if)# q
soodar2(config)#
```

5.3.9 افزودن **peer**

اطلاعات مربوط به طرف مقابل تونل را را در این بخش مشخص می کنیم .

- ابتدا یک نام برای peer مشخص می کنیم .
- public-key مربوط به peer را که قبلا در peer اضافه شده است را تنظیم می کنیم (با دستور sh crypto key KEYNAME می توان public key کلید را مشاهده کرد) .
- آدرس شبکه هایی که قرار است از این تونل route شود و همچنین از این تونل به مقصد ما پذیرفته شود را در **allowed ip** اضافه می کنیم .

نکته

دقت شود اگر بسته ای از طریق تونل به هر نحوی به ما برسد اما آن آدرس جزو allowed ip های ما نباشد، drop خواهد شد .

soodar1

```
soodar1(config)# interface wireguard10
soodar1(config-if)# wireguard peer soodar2
soodar1(config-wg-peer)# public-key 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
soodar1(config-wg-peer)# endpoint 200.1.2.2 port 2121
soodar1(config-wg-peer)# allowed-ip 2.1.1.0/24
soodar1(config-wg-peer)# end
soodar1#
```

soodar2

```
soodar2(config)# interface wireguard20
soodar2(config-if)# wireguard peer soodar1
soodar2(config-wg-peer)# public-key A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
soodar2(config-wg-peer)# endpoint 200.1.2.1 port 1212
soodar2(config-wg-peer)# allowed-ip 1.1.1.0/24
soodar2(config-wg-peer)# end
soodar2#
```

استفاده از **preshared-key**

ما در این سناریو از psk استفاده نکرده ایم اما در صورت نیاز می توانید از preshared-key هم در تنظیمات peer استفاده کنید . کلید باید از نوع HEX و طول آن دقیقا 32 بایت باشد .

```
soodar(config-wg-peer)# preshared-key ABCDABCDABCDABCDABCDABCDABCDABCDABCDABCDABCDABCDABCDABCDABCD
```

شما می توانید در هر اینترفیس وایرگارد که می سازید **peer**های متعددی اضافه نمایید یعنی با ساخت یک اینترفیس تونل و peer های مختلف چندین تونل ایجاد نمایید . البته می توان برای هر peer نیز اینترفیس جداگانه ای ساخت .
مطلب قابل توجه بعدی در این بخش این است که مقصد تونل (endpoint) فقط لازم است در یک سمت تونل تنظیم شود و در سمت دیگر (سرور) که peer های زیادی می تواند داشته باشد تعریف نمی شود و ip طرف مقابل و port مهم نیست و فقط public key مر بوط به peer تعیین کننده است .
در این سناریو در soodar1 که در این جا به عنوان سرور در نظر گرفته شده ، endpoint ی تنظیم نشده است . اما در soodar2 این تنظیم انجام شده است و آدرس ip و port ی که soodar1 منتظر دریافت بسته برای تشکیل تونل است را وارد کرده ایم .

تنظیم **vrf** برای ارتباط با **peer**
اگر route مربوط به peer در vrf خاصی وجود دارد (در اینجا در pub قرار دارد) باید در تنظیمات peer مشخص کنیم که از کدام vrf برای ارتباط با peer استفاده کند که در این سناریو چون اینترفیس های ge0 هر دو روتر در vrf pub قرار دارند در نتیجه یکدیگر را از vrf pub مسیریابی می کنند و این موضوع در تنظیمات peer با دستور **vrf pub** مشخص شده است .

soodar1

```
soodar1(config-wg-peer)# vrf pub
soodar1(config-wg-peer)# end
soodar1#
```

soodar2

```
soodar2(config-wg-peer)# vrf pub
soodar2(config-wg-peer)# end
soodar2#
```

6.3.9 مشاهده وضعیت تونل

برای مشاهده وضعیت تونل های wirguard می توان از دستور زیر استفاده کرد. در نظر داشته باشید گاهی اوقات ممکن است بروز شدن وضعیت تونل کمی با تاخیر انجام شود و به طور مثال تونل وصل شده باشد اما این دستور تونل را قطع نشان دهد .

```
soodar1# sh wireguard
Wireguard 10
  Source: 200.1.2.1
  Key: wg10
  Public key: A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
  Port: 1212

  Peer soodar2:
    Public key: 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
    Persistent keepalive: 10
    VRF: pub
    Connected: True
    Allowed IPs:
     - 2.1.1.0/24

soodar1#
```

توجه

Connected: True مشخص می کند که تونل وصل شده است .

در نمایش وضعیت اینترفیس هم اینترفیس wireguard در vrf مربوطه قرار دارد و up شده است .

```
soodar1(config)# show int vrf all brief
Interface       Status  VRF             Addresses
---------       ------  ---             ---------
```

(continues on next page)

(continued from previous page)

```
pimreg          up      default

Interface       Status  VRF             Addresses
---------       ------  ---             ---------
ge1             up      green           1.1.1.1/24
green           up      green
pimreg300       up      green
wireguard10     up      green           10.0.0.1/32

Interface       Status  VRF             Addresses
---------       ------  ---             ---------
ge0             up      pub             200.1.2.1/24
pub             up      pub

Interface       Status  VRF             Addresses
---------       ------  ---             ---------
ge2             up      red             1.2.1.1/24
red             up      red

soodar1#
```

7.3.9 افزودن تونل دوم

تنظیم تونل دوم هم مانند تونل اول اضافه می شود توجه شود که در تونل جدید باید از کلید های جدید استفاده شود :

soodar1

```
soodar1(config)# interface wireguard11
soodar1(config-if)# ip vrf forwarding red
soodar1(config-if)# wireguard source 200.1.2.1
soodar1(config-if)# wireguard private-key wg11
soodar1(config-if)# wireguard port 4040
soodar1(config-if)# ip address 10.0.120.120/32
soodar1(config-if)# wireguard peer soodar2
soodar1(config-wg-peer)# public-key 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
soodar1(config-wg-peer)# allowed-ip 2.2.1.0/24
soodar1(config-wg-peer)# vrf pub
soodar1(config-if)# end
soodar1#
```

soodar2

```
soodar2(config)# interface wireguard21
soodar2(config-if)# ip vrf forwarding red
soodar2(config-if)# wireguard source 200.1.2.2
soodar2(config-if)# wireguard private-key wg21
soodar2(config-if)# wireguard port 6060
soodar2(config-if)# ip address 10.0.120.210/32
soodar2(config-if)# wireguard peer soodar1
soodar2(config-wg-peer)# public-key 4161E811D7BEE4AA61E2079FAE1A76ED03E5F21CEC38B1A70D9C2781247D7D5A
soodar2(config-wg-peer)# allowed-ip 1.2.1.0/24
soodar2(config-wg-peer)# endpoint 200.1.2.1 port 4040
soodar2(config-wg-peer)# vrf pub
soodar2(config-wg-peer)# end
soodar2#
```

توجه

تنظیم endpoint در دو طرف تونل الزامی نیست و می توان تنها در یک طرف تونل آن را تنظیم کرد .

8.3.9 نمایش تنظیمات روتر ها

soodar1

```
soodar1# show running-config
hostname soodar1
no zebra nexthop kernel enable
security passwords min-length 8
log syslog
log monitor
no banner motd
!
!
vrf pub
!
vrf green
!
vrf red
!
interface ge0 vrf pub
 ip vrf forwarding pub
 no shutdown
 ip address 200.1.2.1/24
!
interface pub vrf pub
 no ip address
 no shutdown
!
```

(continues on next page)

(continued from previous page)

```
interface ge1 vrf green
 ip vrf forwarding green
 no shutdown
 ip address 1.1.1.1/24
!
interface green vrf green
 no ip address
 no shutdown
!
interface wireguard10 vrf green
 ip vrf forwarding green
 wireguard source 200.1.2.1
 wireguard private-key wg10
 wireguard port 1212
 wireguard peer soodar2
  public-key 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
  vrf pub
  allowed-ip 2.1.1.0/24
 no shutdown
 ip address 10.0.12.12/32
!
interface ge2 vrf red
 ip vrf forwarding red
 no shutdown
 ip address 1.2.1.1/24
!
interface red vrf red
 no ip address
 no shutdown
!
interface wireguard11 vrf red
 ip vrf forwarding red
 wireguard source 200.1.2.1
 wireguard private-key wg11
 wireguard port 4040
 wireguard peer soodar2
  public-key 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
  vrf pub
  allowed-ip 2.2.1.0/24
 no shutdown
 ip address 10.0.120.120/32
!
line vty
!
end


soodar1#
```

soodar2

```
soodar2# show running-config
hostname soodar2
no zebra nexthop kernel enable
security passwords min-length 8
log syslog
log monitor
no banner motd
!
!
vrf pub
!
vrf green
!
vrf red
!
interface ge0 vrf pub
 ip vrf forwarding pub
 no shutdown
 ip address 200.1.2.2/24
!
interface pub vrf pub
 no ip address
 no shutdown
!
interface ge1 vrf green
 ip vrf forwarding green
 no shutdown
 ip address 2.1.1.1/24
!
interface green vrf green
 no ip address
 no shutdown
!
interface wireguard20 vrf green
 ip vrf forwarding green
 wireguard source 200.1.2.2
 wireguard private-key wg20
 wireguard port 2121
 wireguard peer soodar1
  public-key A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
  endpoint 200.1.2.1 port 1212
  vrf pub
  allowed-ip 1.1.1.0/24
 no shutdown
 ip address 10.0.12.21/32
!
interface ge2 vrf red
 ip vrf forwarding red
 no shutdown
 ip address 2.2.1.1/24
!
interface red vrf red
 no ip address
 no shutdown
```

(continues on next page)

(continued from previous page)

```
!
interface wireguard21 vrf red
 ip vrf forwarding red
 wireguard source 200.1.2.2
 wireguard private-key wg21
 wireguard port 6060
 wireguard peer soodar1
  public-key 4161E811D7BEE4AA61E2079FAE1A76ED03E5F21CEC38B1A70D9C2781247D7D5A
  endpoint 200.1.2.1 port 4040
  vrf pub
  allowed-ip 1.2.1.0/24
 no shutdown
 ip address 10.0.120.210/32
!

line vty
!
end
soodar2#
```

9.3.9 حذف تونل wireguard

برای حذف تونل wireguard دستور زیر را وارد کنید :

```
soodar1(config)# int wireguard10
soodar1(config-if)# shutdown
informational-ZEBRA: Interface wireguard10 has gone DOWN
soodar1(config-if)# q
soodar1(config)# no int wireguard10
informational-ZEBRA: vrf-change for wireguard10 vrf_id 6 -> 0
soodar1(config)#
```

10.3.9 فعال کردن log های wireguard

با دستور زیر می توانید Log های مربوط به wireguard را فعال کنید:

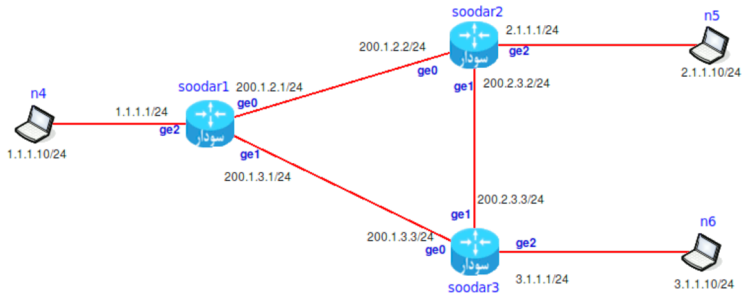
```
soodar1# debug wireguard event
```

wireguard-routing 4.9

1.4.9 آموزش راه اندازی تونل wireguard-routing در سودار

پروتکل wireguard فقط static route را پشتیبانی می کند و شما با اضافه کردن allowed-ip ها در تونل route های استاتیک از طریق تونل وایرگارد ایجاد می کنید . در روتر سودار امکان اجرای ospf در وایرگارد و در واقع Dynamic routing در تونل وایرگارد فراهم شده است .

فرض کنید در سناریو زیر قصد داریم بین روتر ها تونل وایرگارد در حالت routing ایجاد کنیم به نحوی که ارتباط شبکه های پشتی روتر از طریق تونل برقرار شود :



در این سناریو ما :

- در هر روتر یک تونل wireguard می سازیم و در آن peer 2 اضافه می کنیم . برای مثال در soodar1 روتر های soodar3 , soodar2 را به عنوان peer اضافه می کنیم .
- سپس در تونل ospf را اجرا می کنیم
- بررسی می کنیم که تونل ها وصل شده باشند
- همسایگی ospf و جدول routing را بررسی کرده و تست ارتباط از درون تونل را انجام می دهیم

ساخت کلید wg

در ابتدا برای هر تونل یک کلید wg تولید می کنیم . دقت شود که کلید حتما باید از نوع x25519 انتخاب گردد :

نکته

دقت شود که کلید حتما باید از نوع x25519 انتخاب گردد.

---

soodar1

```
soodar1(config)# crypto key generate x25519 label wg10
informational-ZEBRA: new X25519 key wg10 generated

soodar1(config-if)# do sh crypto key wg10
Keypair Label: wg10
  Algorithm:   X25519
  Public key:  A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76

soodar1(config-if)#
```

soodar2

```
soodar2(config)# crypto key generate x25519 label wg20
informational-ZEBRA: new X25519 key wg20 generated

soodar2(config)# do sh crypto key wg20
Keypair Label: wg20
  Algorithm:   X25519
  Public key:  045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831

soodar2(config-if)#
```

soodar3

```
soodar3(config)# crypto key generate x25519 label wg30
informational-ZEBRA: new X25519 key wg30 generated

soodar3(config-if)# do sh crypto key wg30
Keypair Label: wg30
  Algorithm:   X25519
  Public key:  10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300

soodar3(config-if)#
```

ساخت اینترفیس **wireguard**

- قبل ساخت اینترفیس وایرگارد، ospf را فعال می کنیم تا بتوانیم بعدا از آن در اینترفیس وایرگارد استفاده کنیم.
- کلیدی که قرار است برای این تونل استفاده شود را مشخص می کنیم . **public key** این کلید در تنظیمات تونل طرف مقابل استفاده می شود
- پورتی که روی آن گوش می کنیم و بسته ها را با آن پورت مقصد قبول می کنیم را هم مشخص می کنیم این پورت در طرف مقابل تونل در بخش **peer** ها باید مد نظر گرفته شود
- یک ip هم به اینترفیس اختصاص می دهیم . netmask این ip حتما باید 32 باشد .

soodar1

```
soodar1(config)# router ospf
soodar1(config-router)# ospf router-id 222.1.1.1
soodar1(config-router)# redistribute connected
soodar1(config-router)# q
soodar1(config)# interface wireguard10
soodar1(config-if)# wireguard mode routing
soodar1(config-if)# wireguard source 0.0.0.0
soodar1(config-if)# wireguard private-key wg10
soodar1(config-if)# wireguard port 1100
soodar1(config-if)# ip address 10.0.0.10/32
soodar1(config-if)# ip ospf network point-to-multipoint
soodar1(config-if)# ip ospf area 0
soodar1(config-if)# end
soodar1#
```

soodar2

```
soodar2(config)# router ospf
soodar2(config-router)# ospf router-id 222.2.2.2
soodar2(config-router)# redistribute connected
soodar2(config-router)# q
soodar2(config)# interface wireguard20
soodar2(config-if)# wireguard mode routing
soodar2(config-if)# wireguard source 0.0.0.0
soodar2(config-if)# wireguard private-key wg20
soodar2(config-if)# wireguard port 1200
soodar2(config-if)# ip address 10.0.0.20/32
soodar2(config-if)# ip ospf network point-to-multipoint
soodar2(config-if)# ip ospf area 0
soodar2(config-if)# q
soodar2(config)#
```

soodar3

```
soodar3(config)# router ospf
soodar3(config-router)# ospf router-id 222.3.3.3
soodar3(config-router)# redistribute connected
soodar3(config-router)# q
soodar3(config)# interface wireguard30
soodar3(config-if)# wireguard mode routing
soodar3(config-if)# wireguard source 0.0.0.0
soodar3(config-if)# wireguard private-key wg30
soodar3(config-if)# wireguard port 1300
soodar3(config-if)# ip address 10.0.0.30/32
soodar3(config-if)# ip ospf network point-to-multipoint
```

(continues on next page)

(continued from previous page)

```
soodar3(config-if)# ip ospf area 0
soodar3(config)#
```

توجه

1. تنظیم **wireguard source 0.0.0.0** بدین معنی است که از هر اینترفیسی peer را پیدا کند تونل وصل می شود و به اینترفیس خاصی bind نمی شود. برای مثال soodar1 به soodar2 از طریق اینترفیس ge0 و به soodar3 از طریق اینترفیس ge1 وصل می شود .
2. دقت شود چون دو نوع تونل وایرگارد در سودار وجود دارد (هم استاتیک و هم داینامیک) حتما **wireguard mode routing** را در اینترفیس وارد کنید .

افزودن **peer**

اطلاعات مربوط به طرف مقابل تونل را را در این بخش مشخص می کنیم .

- ابتدا یک نام برای peer مشخص می کنیم .
- public-key مربوط به peer را که قبلا در peer اضافه شده است را تنظیم می کنیم (با دستور sh crypto key KEYNAME می توان public key کلید را مشاهده کرد) .
- آدرس ip مروبط به اینترفیس وایرگارد peer را در بخش allowed-ip اضافه می کنیم .

soodar1

```
soodar1(config)# interface wireguard10
soodar1(config-if)# wireguard peer soodar2
soodar1(config-wg-peer)# public-key 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
soodar1(config-wg-peer)# endpoint 200.1.2.2 port 1200
soodar1(config-wg-peer)# allowed-ip 10.0.0.20/32
soodar1(config-wg-peer)# vrf default
soodar1(config-wg-peer)# q


soodar1(config-if)# wireguard peer soodar3
soodar1(config-wg-peer)# public-key 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
soodar1(config-wg-peer)# allowed-ip 10.0.0.30/32
soodar1(config-wg-peer)# endpoint 200.1.3.3 port 1300
soodar1(config-wg-peer)# end
soodar1#
```

soodar2

```
soodar2(config)# interface wireguard20
soodar2(config-if)# wireguard peer soodar1
soodar2(config-wg-peer)# public-key A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
soodar2(config-wg-peer)# allowed-ip 10.0.0.10/32
soodar2(config-wg-peer)# endpoint 200.1.2.1 port 1100
soodar2(config-wg-peer)# q
soodar2(config-if)# wireguard peer soodar3
soodar2(config-wg-peer)# public-key 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
soodar2(config-wg-peer)# allowed-ip 10.0.0.30/32
soodar2(config-wg-peer)# endpoint 200.2.3.3 port 3100
soodar2#
```

soodar3

```
soodar3(config)# interface wireguard30
soodar3(config-if)# wireguard peer soodar1
soodar3(config-wg-peer)# public-key A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
soodar3(config-wg-peer)# allowed-ip 10.0.0.10/32
soodar3(config-wg-peer)# endpoint 200.1.2.1 port 1100
soodar3(config-wg-peer)# q
soodar3(config-if)# wireguard peer soodar2
soodar3(config-wg-peer)# public-key 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
soodar3(config-wg-peer)# allowed-ip 10.0.0.20/32
soodar3(config-wg-peer)# endpoint 200.2.3.2 port 2100
soodar3#
```

شما می توانید در هر اینترفیس وایرگاردی که می سازید **peer**های متعددی اضافه نمایید یعنی با ساخت یک اینترفیس تونل و peer های مختلف چندین تونل ایجاد نمایید . البته می توان برای هر peer نیز اینترفیس جداگانه ای ساخت .

تنظیم **vrf** برای ارتباط با **peer** اگر route مربوط به peer در vrf خاصی وجود دارد (در اینجا در default قرار دارد) باید در تنظیمات peer مشخص کنیم که از کدام vrf برای ارتباط با peer استفاده کند. این موضوع در تنظیمات peer با دستور **vrf default** مشخص شده است .

2.4.9 مشاهده وضعیت تونل

برای مشاهده وضعیت تونل های wirguard می توان از دستور زیر استفاده کرد.

```
soodar1# sh wireguard
Wireguard 10
  Mode: Routing
  Source: 200.1.2.1
  Key: wg10
  Public key: A1D71640B325F3275C4C92431EF7A41DF5F25EF010FA7FAEB5F1214A34F80D76
  Port: 1100

  Peer soodar2:
```

(continues on next page)

(continued from previous page)

```
  Public key: 045CB4AD197FD8C60B8AA0C966B625B3209570DFC42039CD9EC8983B6E818831
  Endpoint: 200.1.2.2
  Current Endpoint: 200.1.2.2
  Current Source: 200.1.2.1
  Persistent keepalive: 10
  Port: 1200
  VRF: default
  Connected: True
  Allowed IPs:
   - 10.0.0.20/32

Peer soodar3:
  Public key: 10295CB9E3CE6AF124630E8351B5C46E0F54C107346F679BCEE9FAA970F16300
  Endpoint: 200.1.3.3
  Current Endpoint: 200.1.3.3
  Current Source: 200.1.3.1
  Persistent keepalive: 10
  Port: 1300
  VRF: default
  Connected: True
  Allowed IPs:
   - 10.0.0.30/32
```

توجه

Connected: True مشخص می کند که تونل وصل شده است .

در نمایش وضعیت اینترفیس هم اینترفیس wireguard در vrf مربوطه قرار دارد و up شده است .

```
soodar1(config)# show int vrf all brief
Interface       Status  VRF             Addresses
---------       ------  ---             ---------
pimreg          up      default
ge0             up      default         200.1.2.1/24
ge1             up      default         200.1.3.1/24
ge2             up      default         1.1.1.1/24
wireguard10     up      default         10.0.0.10/32

soodar1#
```

3.4.9 حذف تونل **wireguard**

برای حذف تونل wireguard دستور زیر را وارد کنید :

```
soodar1(config)# int wireguard10
soodar1(config-if)# shutdown
informational-ZEBRA: Interface wireguard10 has gone DOWN
soodar1(config-if)# q
soodar1(config)# no int wireguard10
```

(continues on next page)

(continued from previous page)

```
informational-ZEBRA: vrf-change for wireguard10 vrf_id 6 -> 0
soodar1(config)#
```

4.4.9 فعال کردن log های wireguard

با دستور زیر می توانید Log های مربوط به wireguard را فعال کنید:

```
soodar1# debug wireguard event
```

5.9 vxlan

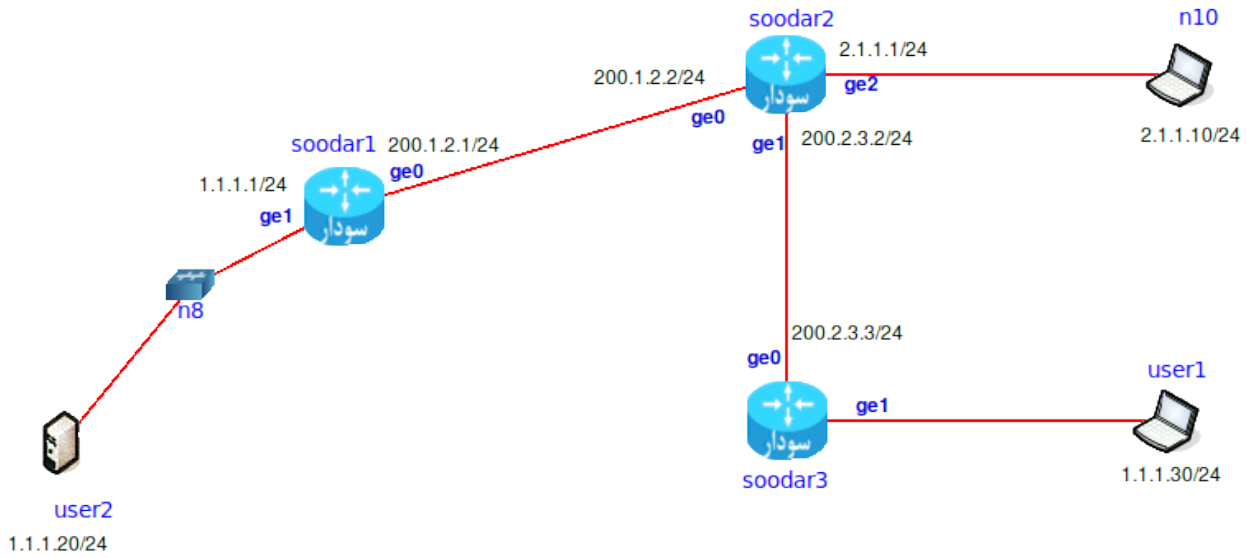
1.5.9 آشنایی با مفهوم VXLAN

در مقایسه با VLAN تکنولوژی VXLAN مزایای زیر را خواهد داشت:

- قابلیت انتقال و جابجایی VLAN بر روی سگمنت مشترک و در نتیجه جابجایی شبکه های لایه 2ای مشتریان از یک شبکه به شبکه دیگر
- گستردگی آدرس دهی بسیار بیشتر از VLAN زیرا VLAN از یک آدرس 12 بیتی استفاده میکند که امکان تولید 4096 سگمنت برای ما فراهم می کند در حالی که VXLAN از یک آدرس 24 بیتی استفاده کرده و امکان آدرس دهی 16 میلیون سگمنت را برای ما فراهم میکند.
- کارایی و بازدهی بیشتر نسبت به VLAN داشته زیرا در مدل VLAN ما درگیر پروتکل STP بوده و بسیاری از مسیرهای ما به خاطر وجودLOOP بلاک شده و تقریباً نیمی از ظرفیت شبکه ما استفاده نمیگردد و این درحالیست که با معماری VXVLAN ارسال ترافیک لایه 2 ای ما بر روی بستر IP بوده و به دلیل قابلیت های Load Balancing بر رویIP و استفاده همزمان از تمامی لینک ها، کارایی شبکه بسیار بیشتر می باشد.

2.5.9 آموزش راه اندازی VXLAN در سودار

فرض کنید در شکل زیر قصد داریم بین دو روتر soodar3 , soodar1 تونل vxlan بزنیم و user2 , user1 بتوانند از طریق این تونل با یکدیگر ارتباط داشته باشند :



ایجاد اینترفیس nve

ایندکس اینترفیس های vxlan می تواند از 0 تا 1023 باشد . در واقع در هر روتر حداکثر می توان 1024 اینترفیس vxlan داشت . با دستور زیر ابتدا یک اینترفیس vxlan ایجاد کنید :

```
soodar1(config)# interface nve 10
```

تعیین مبدا تونل

در این بخش آدرس IP مبدا تونل که همان ip اینترفیس روتر است که قرار است مبدا تونل باشد(ge0 در soodar1) :

```
soodar1(config-if)# source-ip 200.1.2.1
```

تعیین مقصد تونل

در این بخش آدرس IP مقصد تونل که همان ip اینترفیس روتر است که قرار است به آن تونل بزنیم (ge0 در soodar3) :

```
soodar1(config-if)# ingress-replication 200.2.3.3
```

مشخص کردن شماره vni

شماره vni به عنوان شناسه تونل vxlan استفاده می شود و با این شناسه در هر شبکه می توان 16 میلیون segment مختلف ایجاد کرد . شماره vni از 1 تا 16777214 می تواند مقدار دهی شود .

```
soodar1(config-if)# member vni 30
```

اضافه کردن اینترفیس vxlan به bridge

در نهایت باید یک bridge بین اینترفیس تونل vxlan و اینترفیسی که قرار است از تونل استفاده کند برقرار کرد ge1) در soodar1) :

```
soodar1(config)# interface ge1
soodar1(config-if)# bridge-group 200
soodar1(config-if)# no shutdown
soodar1(config)# interface nve10
soodar1(config-if)# bridge-group 200
soodar1(config-if)# do show bridge 200
 |-----------------------------------------------|
 | Domain | Interface | Split-Horizon Group | BVI |
 |--------+-----------+---------------------+-----|
 | 200    | ge1       | 0                   | -   |
 |--------+-----------+---------------------+-----|
 |        | nve10     | 0                   | -   |
 |--------+-----------+---------------------+-----|
```

تنظیمات مربوط به vxlan در soodar1 به پایان رسید . به همین شکل تنظیمات را در روتر soodar3 نیز انجام می دهیم .

```
soodar3(config)# interface nve 10
soodar3(config-if)# source-ip 200.1.2.1
soodar3(config-if)# ingress-replication 200.2.3.3
soodar3(config-if)# member vni 30
--------
soodar3(config)# interface ge1
soodar3(config-if)# bridge-group 200
soodar3(config-if)# no shutdown
soodar3(config)# interface nve10
soodar3(config-if)# bridge-group 200
```

حال ارتباط بین user2 , user1 به صورت لایه 2 توسط تونل vxlan برقرار می باشد.

تنظیم vrf در vxlan

اگر می خواهید برای برقراری تونل vxlan و رسیدن به مقصد تونل از vrf خاصی استفاده کنید به شکل زیر باید در تمظیمات vrf مربوطه مقدار vni را تنظیم کنید تا تونل vxlan برای lookup کردن آدرس مقصد تونل از این vrf استفاده کند :

```
soodar(config)# vrf green
soodar(config-vrf)# vni 30
soodar(config-vrf)#
```

این تنظیم مشخص می کند که vxlan با vni شماره 30 باید از vrf green برای برقراری تونل استفاده کند .

3.5.9 فعال کردن **log** های **vxlan**

با دستور زیر می توانید Log های مربوط به vxlan را فعال کنید:

```
soodar1# debug vxlan event
```

PPPoE Client 6.9

پروتکل PPPoE یک پروتکل شبکه است که به طور معمول برای برقراری اتصال مستقیم بین دستگاه مشتری و ارائه دهنده خدمات اینترنت (ISP) با استفاده از بستر Ethernet به عنوان رسانه‌ای پایه استفاده می‌شود. این امکان را به کاربران می‌دهد که از طریق یک مودم DSL به اینترنت متصل شوند و روشی امن و کارآمد برای انتقال داده‌ها در شبکه ISP فراهم می‌کند.

در یک تنظیم PPPoE، دستگاه مشتری با بسته‌بندی فریم های PPP داخل فریم های اترنت، یک نشست PPP را با شبکه ISP آغاز می‌کند. این بسته‌بندی اجازه می‌دهد تا ترافیک PPP بر روی زیرساخت شبکه مبتنی بر اترنت ISP منتقل شود و به طور موثر یک اتصال نقطه به نقطه ایجاد کند. فرآیند PPPoE دارای دو مؤلفه اصلی است:

کلاینت **PPPoE**: این در دستگاه کاربر، مانند رایانه یا روتر، قرار دارد و مسئول آغاز و مدیریت نشست PPP است. وقتی کاربر می‌خواهد به اینترنت متصل شود، کلاینت PPPoE یک بسته کشف PPPoE را برای یافتن و ارتباط با سرور PPPoE ISP ارسال می‌کند.

سرور **PPPoE**: این در شبکه ISP قرار دارد و اتصالات ورودی PPPoE را اداره می‌کند. سرور بسته کشف PPPoE را دریافت کرده، تأیید هویت مشتری را انجام می‌دهد و شناسه نشست یکتا (معروف به شناسه نشست یا SID) را تعیین می‌کند تا یک پیوند اختصاصی برای آن اتصال خاص برقرار شود.

فرآیند اتصال PPPoE معمولاً شامل سه مرحله است:

کشف: کلاینت PPPoE یک بسته کشف را برای یافتن سرور PPPoE ISP ارسال می‌کند. این بسته شامل فریم اترنت ویژه با پیام فعال‌سازی کشف PPPoE (PADI) است. سرور با پاسخ به پیام فعال‌سازی کشف PPPoE (PADO)، پارامترهای اتصال و گزینه‌های نشست را فراهم می‌کند.

آغاز نشست: پس از دریافت پیام PADO، کلاینت گزینه‌های مناسب نشست را انتخاب کرده و پیام درخواست فعال‌سازی کشف PPPoE (PADR) را به سرور ارسال می‌کند. پس از تأیید موفق کلاینت، سرور پیام تأیید نشست فعال‌سازی PPPoE (PADS) را ارسال کرده و نشست PPP برقرار می‌کند.

انتقال داده: پس از برقراری نشست PPP، داده‌ها می‌توانند به صورت دوطرفه بین مشتری و سرور منتقل شوند. این بسته‌بندی داده اطمینان می‌دهد که فریم های PPP در داخل فریم های اترنت بسته‌بندی می‌شوند و به آنها اجازه می‌دهد تا در شبکه اترنت ISP منتقل شوند.

در روزگار حاضر، PPPoE همچنان در برخی حالت‌ها به دلایل خاص استفاده می‌شود: احراز هویت و امنیت: PPPoE از طریق پروتکل‌های PAP، CHAP یا MS-CHAP یک مکانیزم احراز هویت قابل اعتماد فراهم می‌کند. این اطمینان را می‌دهد که تنها کاربران مجاز می‌توانند به خدمات اینترنت دسترسی پیدا کنند. چون امنیت یک نگرانی مهم در شبکه‌های مدرن است، قابلیت‌های احراز هویت PPPoE همچنان مورد توجه و ارزش است.

مدیریت منابع : PPPoE به ISPها کمک می‌کند تا منابع شبکه خود را به صورت کارآمد مدیریت کنند. این به ISPها اجازه می‌دهد تا پهنای باند را کنترل و تخصیص دهند، سیاست‌های کیفیت خدمات (QoS) را اجرا کنند و اتصالات کاربران فردی را نظارت کنند. این سطح کنترل به بهینه سازی عملکرد شبکه و اطمینان از استفاده منصفانه بین مشترکین کمک می‌کند.

سازگاری NAT (ترجمه آدرس شبکهPPPoE :(با NAT سازگار است که به طور معمول برای حل محدودیت دسترسی محدود آدرس‌های IPv4 استفاده می‌شود. ISPها اغلب PPPoE را به همراه NAT استفاده می‌کنند، که به چندین دستگاه در شبکه مشتری اجازه می‌دهد تا یک آدرس IP عمومی تک را به اشتراک بگذارند.

توجه

در حال حاضر سودار فقط از PPPoE-client پشتیبانی می کند.

ابتدا یک اینترفیس dialer می سازیم . این اینترفیس ها معمولا برای ارتباطات point-to-point(PPP) و PPP over Ehternet (PPPoE) استفاده می شوند . یک شماره بین 1 تا 255 به آن اختصاص می یابد . به عبارت دیگر شما می توانید 255 اینترفیس dialer در روتر سودار اضافه کنید .

```
n1(config)# interface dialer1
n1(config-if)# encapsulation ppp
n1(config-if)# dialer pool 1
n1(config-if)# ppp chap hostname test
n1(config-if)# ppp chap password 123
n1(config-if)# ppp pap sent-username test password 123
n1(config)# interface ge0
n1(config-if)# no ip address
n1(config-if)# pppoe-client dial-pool-number 1
n1(config-if)# no shutdown
```

encapsulation ppp : این دستور برای پیکربندی پروتکل PPP (Point-to-Point Protocol) روی یک اینترفیس dialer استفاده می‌شود. دستور "encapsulation ppp" امکان ارسال فریم‌های PPP را از طریق اینترفیس dialer مشخص شده فراهم می‌کند، اجازه می‌دهد تا فریم‌های PPP را از طریق لینک ارسال کند.

dialer pool 1: این دستور برای ارتباط دادن یک اینترفیس dialer با یک dialer pool استفاده می‌شود.dialer pool ها در پیکربندی‌های پروتکل نقطه به نقطه (PPP) برای مدیریت چندین اینترفیس فیزیکی مانند اینترفیس های اترنت استفاده می‌شوند که به شبکه‌های از راه دور یا ارائه‌دهنده‌های خدمات اینترنت (ISP) متصل می‌شوند. دستور "dialer pool" به اینترفیس dialer اجازه می‌دهد تا از یکی از اینترفیس های فیزیکی موجود در pool برای برقراری اتصال PPP استفاده کند.

توجه

در حال حاضر فقط یک اینترفیس فیزیکی می تواند در یک dialer pool استفاده شود .

1.6.9 پیکربندی PPPoE

برای ایجاد یک اتصال PPPoE، ابتدا باید یک اینترفیس dialer ایجاد کنیم. اینترفیس dialer یک اینترفیس مجازی است که پیکربندی‌های PPPoE را ذخیره می‌کند.

interface dialer (1-255)

این دستور برای پیکربندی یک اینترفیس dialer بر روی یک دستگاه استفاده می‌شود. اینترفیس dialer معمولاً در پروتکل Point-to-Point (PPP) و PPP over Ethernet (PPPoE) استفاده می‌شوند.

- (1-255): شماره نمونه اینترفیس dialer را مشخص می‌کند. مقادیر معتبر اعداد صحیح از 1 تا 255 هستند.

encapsulation ppp

این دستور برای پیکربندی کپسوله‌سازی Point-to-Point Protocol (PPP) بر روی یک اینترفیس dialer استفاده می‌شود. دستور "encapsulation ppp" کپسوله‌سازی PPP را بر روی اینترفیس dialer مشخص شده پشتیبانی می‌کند، که اجازه می‌دهد تا فریم‌های PPP را بر روی اتصال انتقال دهد.

توجه

این کپسوله‌سازی پیش‌فرض برای اینترفیس های dialer است.

---

dialer pool (1-255)

این دستور برای ارتباط دادن یک اینترفیس dialer مجازی با یک dialer pool استفاده می‌شود. dialer pool ها در پروتکل Point-to-Point (PPP) برای مدیریت چندین interface فیزیکی، مانند interfaceهای اترنت که به شبکه‌های از راه دور یا ارائه دهنده‌های خدمات اینترنت (ISPs)

متصل می‌شوند، استفاده می‌شوند. دستور "dialer pool" به اینترفیس dialer مجازی اجازه می‌دهد تا یکی از interfaceهای فیزیکی موجود در pool را برای برقراری اتصال PPP استفاده کند.

- (1-255): شماره dialer pool را برای ارتباط با اینترفیس dialer مشخص می‌کند. مقادیر معتبر اعداد صحیح از 1 تا 255 هستند.

توجه

در حال حاضر، تنها یک interface فیزیکی ممکن است در یک dialer pool باشد.

---

pppoe-client dial-pool-number (1-255)

این دستور برای پیکربندی یک نشست PPPoE client بر روی یک اینترفیس و ارتباط آن با یک شماره dialer pool خاص استفاده می‌شود. دستور "pppoe-client dial-pool-number" قابلیت PPPoE client را فعال می‌کند و آن را به یک dialer pool خاص متصل می‌کند که شامل اینترفیس های فیزیکی برای نشست‌های PPPoE است.- :(1-255) شماره dialer pool را برای ارتباط با PPPoE client مشخص می‌کند. مقادیر معتبر اعداد صحیح از 1 تا 255 هستند.

در زیر مثالی از پیکربندی PPPoE client بر روی یک interface اترنت و ارتباط آن با یک dialer pool آمده است:

```
n1 (config)# interface ge0
n1 (config-if)# pppoe-client dial-pool-number 1
n1 (config)# interface dialer 1
n1 (config-if)# dialer pool 1
```

در این مثال، ge0 اینترفیس اترنت است که در آن PPPoE client پیکربندی شده است. دستور pppoe-client dial-pool-number 1 این PPPoE client را با شماره dialer pool 1 ارتباط می‌دهد. این dialer pool توسط اینترفیس dialer مجازی dialer1 برای برقراری نشست‌های PPP استفاده می‌شود. بسته‌های PPP به interface ge0 منتقل می‌شوند و در آنجا، آن‌ها بر روی اترنت (PPPoE) انتقال داده می‌شوند.

ppp pap sent-username USER password PASS

این دستور برای پیکربندی اعتبار PAP (Password Authentication Protocol) برای احراز هویت در پروتکل Point-to-Point (PPP) استفاده می‌شود. PAP یک روش احراز هویت ساده است که برای تأیید هویت PPP client هنگام برقراری اتصال PPP استفاده می‌شود. این دستور نام کاربری و رمز عبور را که روتر محلی (به عنوان یک PPP client) آن‌ها را به روتر راه دور هنگام فرآیند احراز هویت ارسال می‌کند، مشخص می‌کند.

- USER: نام کاربری را که هنگام احراز هویت PAP ارسال شود، مشخص می‌کند.
- PASS: رمز عبور را که هنگام احراز هویت PAP ارسال شود، مشخص می‌کند.

ppp chap hostname HOSTNAME

این دستور برای پیکربندی hostname ارسال شده در فرآیند چالش دستورالعمل احراز هویت (CHAP) در پروتکل Point-to-Point (PPP) استفاده می‌شود. CHAP یک روش احراز هویت امن‌تر نسبت به Password Authentication Protocol (PAP) است و به طور معمول برای تأیید هویت PPP client هنگام برقراری اتصال PPP استفاده می‌شود. دستور "ppp chap hostname" hostname استفاده شده توسط روتر محلی (به عنوان یک PPP client) هنگام پاسخ به چالش‌های CHAP از روتر از راه دور را مشخص می‌کند.

- HOSTNAME: نام hostname را که هنگام احراز هویت CHAP استفاده شود، مشخص می‌کند.

ppp chap password PASSWORD

این دستور برای پیکربندی رمز عبور client برای چالش دستورالعمل احراز هویت (CHAP) در پروتکل Point-to-Point (PPP) استفاده می‌شود. هم روتر محلی (به عنوان یک PPP client) و هم روتر از راه دور (به عنوان سرور احراز هویت) دارای همان رمز عبور هستند. دستور "ppp chap password" رمز عبوری را که توسط روتر محلی در فرآیند احراز هویت CHAP استفاده شود، مشخص می‌کند.

- PASSWORD: رمز عبور مشترکی را که هنگام احراز هویت CHAP استفاده شود، مشخص می‌کند.

ppp timeout idle (30-15552000)

این دستور برای پیکربندی مقدار زمان خاموشی برای نشست‌های Point-to-Point (PPP) بر روی یک interface شبکه استفاده می‌شود. زمان خاموشی حداکثر مدت زمان غیرفعال بودن در اتصال PPP قبل از قطع خودکار آن است. وقتی هیچ داده‌ای از طریق اتصال PPP منتقل یا دریافت نشود، روتر جلسه PPP را به منظور صرفه‌جویی در منابع شبکه متوقف می‌کند.

- : (30-15552000) مدت زمان timeout بیکاری به ثانیه مشخص شده است. مقادیر معتبر در بازه 30 تا 15،552،000 ثانیه است (تقریباً 6 ماه).

توجه

توجه مقدار timeout به صورت مضربی از 10 گرد می‌شود.

توجه

توجه استفاده از # در نام کاربری و رمز عبور PAP یا CHAP ممنوع است.

show pppoe session

دستور برای نمایش اطلاعات درباره جلسات فعال پروتکل PPPoE (Point-to-Point Protocol over Ethernet) در روتر استفاده می‌شود. دستور "show pppoe session" خلاصه‌ای از جلسات فعال PPPoE را ارائه می‌دهد که شامل شناسه جلسه، آدرس MAC محلی و از راه دور و وضعیت فعلی جلسه است.

مثال

فرض کنید سناریوی شبیه شکل زیر داریم که یک PPPoE-server داریم و یک روتر سودار که برای دسترسی به اینترنت (شبکه پشتی pppoe-server) باید یک اتصال pppoe برقرار کنیم و همچنین با استفاده از NAT ارتباط شبکه local و اینترنت را جدا کنیم :

برای برقراری یک نشست PPPoE از طریق ge0 interface با استفاده از اینترفیس Dialer به نام dialer1 و با هر دو اعتبارسنجی PAP و CHAP و با نام کاربری "test" و رمز عبور "123" تنظیمات به شکل زیر خواهدبود:

```
n1 (config)# interface dialer1
n1 (config-if)# encapsulation ppp
n1 (config-if)# dialer pool 1
n1 (config-if)# ppp chap hostname test
n1 (config-if)# ppp chap password 123
n1 (config-if)# ppp pap sent-username test password 123
n1 (config)# interface ge0
n1 (config-if)# no ip address
```

(continues on next page)

(continued from previous page)

```
n1 (config-if)# pppoe-client dial-pool-number 1
n1 (config-if)# no shutdown
```

1. این dialer1 یک dialer interface مجازی است که با interface فیزیکی از طریق Dialer pool مرتبط خواهد شد. ما کپسوله‌سازی PPP را پیکربندی و Dialer1 را با استفاده از dialer pool 1 مرتبط می‌کنیم.
2. ما اعتبارسنجی CHAP را بر روی dialer1 پیکربندی کرده‌ایم، با تعیین)"test"hostname (و رمز عبور)"123") استفاده شده برای اعتبارسنجی CHAP با روتر از راه دور.
3. همچنین ما اعتبارسنجی PAP را بر روی dialer1 پیکربندی کرده‌ایم، با تعیین نام کاربری)"test") و رمز عبور)"123") که روتر محلی در طول فرآیند اعتبارسنجی PAP ارسال خواهد کرد.
4. ما آدرس های ip روی ge0 حذف نموده و آن را به dialer pool 1 برای نشست‌های PPPoE اختصاص می‌دهیم.

با این پیکربندی، روتر اکنون آماده برقراری یک نشست PPPoE از طریق ge0 با استفاده از dialer1 به عنوان اینترفیس مجازی است. نشست PPPoE از هر دو روش اعتبارسنجی PAP و CHAP با استفاده از نام کاربری "test" و رمز عبور "123" در طول فرآیند اعتبارسنجی استفاده خواهد کرد.

```
n1# show pppoe session
|-------------------------------------------------------------------|
| SID | Remote MAC        | Local MAC         | Interface | Port | State |
|-----+-------------------+-------------------+-----------+------+-------|
| 3   | 08:19:21:ff:00:00 | 02:fe:53:5e:11:4a | dialer1   | ge0  | Up    |
|-------------------------------------------------------------------|
```

حال اگر بخواهیم ترافیک خروجی از اینترفیس dialer1 را NAT کنیم باید ابتدا یک ACL تعریف کرده و ترافیک مد نظر برای NAT شدن را توسط ACL انتخاب کنیم و همچنین اینترفیس dialer1 به عنوان nat outside و اینترفیس ge1 که شبکه داخلی ماست را به عنوان nat inside مشخص کنیم . تنظیمات به شکل زیر خواهد بود:

```
n1(config)# ip access-list nat-acl
n1(config-nacl)# permit 1.1.1.0/24 any
n1(config-nacl)# q
n1(config)# ip nat inside source list nat-acl interface dialer1
n1(config)# int dialer1
n1(config-if)# ip nat outside
n1(config-if)# q
n1(config)# int ge1
n1(config-if)# ip nat inside
```

توجه

در تعریف NAT در اینجا هیچ pool ی اضافه نکردیم و اینترفیس dialer1 را به عنوان pool به NAT اختصاص دادیم و هر ip که به اینترفیس dialer1 توسط PPPoE-server داده شود به عنوان pool استفاده شده و مبدا بسته های خروجی از روتر سودار همان IP خواهد بود.

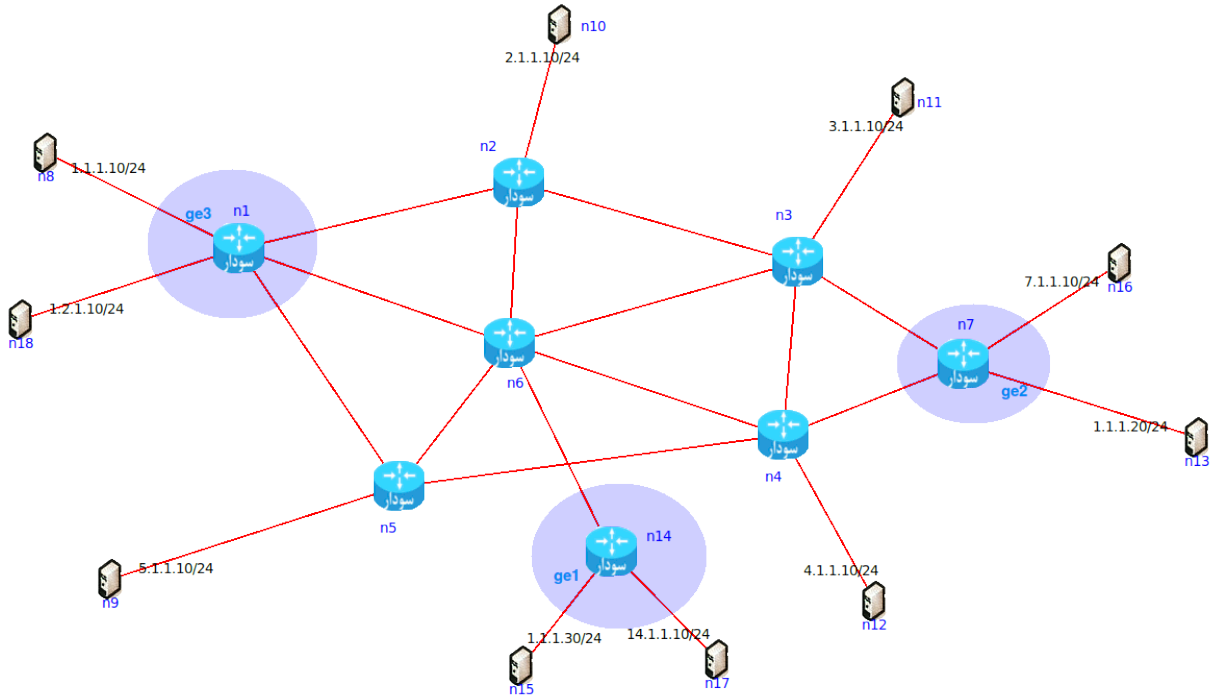
---

vpls 7.9

1.7.9 آشنایی با مفهوم VPLS

2.7.9 آموزش راه اندازی VPLS در سودار

فرض کنید در سناریو زیر LDP در کل شبکه اجرا شده است و تمام مسیر MPLS میباشد. میخواهیم بین روترهای n1، n7 و n14 یک تونل VPLS برقرار کنیم.



برای ساخت یک سناریو VPLS باید به دو نکته توجه داشت:

1. ارتباطات به صورت Full-mesh میباشد.
2. نام‌گذاری اینترفیس تونل‌ها به صورت mpls-tunnelX میباشد و لازم است که X از صفر شروع شده و به ترتیب افزایش پیدا کند.

حال به Config روترها میپردازیم:

n1

ابتدا اینترفیس ge3 را در یک دامنه bridge اضافه میکنیم

```
n1(config)# interface ge3
n1(config-if)# bridge-group 200
n1(config-if)# no shutdown
```

حال دو اینترفیس mpls-tunnel(یکی برای اتصال به n7 و دیگری برای اتصال به n14) میسازیم و به همان دامنه bridge می‌افزاییم:

```
n1(config)# interface mpls-tunnel0
n1(config-if)# bridge-group 200 split-horizon group 100
n1(config-if)# no shutdown
```

```
n1(config)# interface mpls-tunnel1
n1(config-if)# bridge-group 200 split-horizon group 100
n1(config-if)# no shutdown
```

توجه داشته باشید برای جلوگیری از ایجاد loop در bridge، این دو اینترفیس باید در یک split-horizon group قرار داشته باشند.

```
n1(config)# mpls ldp
n1(config-ldp)# router-id 222.1.1.1
```

برای ساخت VPLS لازم است که ldp lsr-id روترهای هدف(در این سناریو n7 و n14) را داشته باشیم. حال به ساختن VPLS میپردازیم:

```
n1(config)# l2vpn VPLS_TUNNEL type vpls
n1(config-l2vpn)# member pseudowire mpls-tunnel0
n1(config-l2vpn-pw)# neighbor lsr-id 222.7.7.7
n1(config-l2vpn-pw)# pw-id 170
n1(config-l2vpn)# member pseudowire mpls-tunnel1
n1(config-l2vpn-pw)# neighbor lsr-id 222.14.14.14
n1(config-l2vpn-pw)# pw-id 1140
```

هر تونل با pw-id آن شناخته میشود و در دو سر تونل باید این مقادیر یکسان باشد.
به شیوه مشابه روترهای n7 و n14 را نیز تنظیم مینماییم:

n7

```
n7(config)# interface ge2
n7(config-if)# bridge-group 200
n7(config-if)# no shutdown
n7(config)# interface mpls-tunnel0
n7(config-if)# bridge-group 200 split-horizon group 100
n7(config-if)# no shutdown
n7(config)# interface mpls-tunnel1
n7(config-if)# bridge-group 200 split-horizon group 100
n7(config-if)# no shutdown
n7(config)# l2vpn VPLS_TUNNEL type vpls
n7(config-l2vpn)# member pseudowire mpls-tunnel0
n7(config-l2vpn-pw)# neighbor lsr-id 222.1.1.1
n7(config-l2vpn-pw)# pw-id 170
n7(config-l2vpn)# member pseudowire mpls-tunnel1
n7(config-l2vpn-pw)# neighbor lsr-id 222.14.14.14
n7(config-l2vpn-pw)# pw-id 7140
```

n14

```
n14(config)# interface ge1
n14(config-if)# bridge-group 200
n14(config-if)# no shutdown
n14(config)# interface mpls-tunnel0
n14(config-if)# bridge-group 200 split-horizon group 100
n14(config-if)# no shutdown
n14(config)# interface mpls-tunnel1
n14(config-if)# bridge-group 200 split-horizon group 100
n14(config-if)# no shutdown
n14(config)# l2vpn VPLS_TUNNEL type vpls
n14(config-l2vpn)# member pseudowire mpls-tunnel0
n14(config-l2vpn-pw)# neighbor lsr-id 222.1.1.1
n14(config-l2vpn-pw)# pw-id 1140
n14(config-l2vpn)# member pseudowire mpls-tunnel1
n14(config-l2vpn-pw)# neighbor lsr-id 222.14.14.14
n14(config-l2vpn-pw)# pw-id 7140
```

3.7.9 مشاهده تونل های vpls

با زدن دستور show mpls pseudowires میتوان از وضعیت تونل‌های خود با خبر شد.

```
n1# sh mpls pseudowires
Interface       Neighbor            Labels      Protocol Status
mpls-tunnel0    222.7.7.7           16/16       ldp      UP
mpls-tunnel1    222.14.14.14        17/16       ldp      UP
```

بخشی از تنظیمات اعمال شده در n1 را مشاهده می کنید :

```
!
interface loopback0
 no shutdown
 ip address 222.1.1.1/32
!
interface mpls-tunnel0
 bridge-group 200 split-horizon group 100
 no shutdown
!
interface mpls-tunnel1
 bridge-group 200 split-horizon group 100
 no shutdown
!
interface ge0
 mpls ip
 no shutdown
 ip address 200.1.2.1/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
 ip ospf priority 254
!
interface ge1
 mpls ip
 no shutdown
 ip address 200.1.6.1/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
!
interface ge2
 mpls ip
 no shutdown
 ip address 200.1.5.1/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
!
interface ge3
 bridge-group 200 split-horizon group 0
 no shutdown
!
interface ge4
 no shutdown
 ip address 1.2.1.1/24
!
router ospf
 ospf router-id 222.1.1.1
```

(continues on next page)

(continued from previous page)

```
 redistribute connected
 redistribute static
 passive-interface ge3
 network 1.2.1.0/24 area 0
 network 200.1.2.0/24 area 0
 network 200.1.5.0/24 area 0
 network 200.1.6.0/24 area 0
 network 222.1.1.1/32 area 0
!
mpls ldp
 router-id 222.1.1.1
 dual-stack transport-connection prefer ipv4
 dual-stack cisco-interop
 !
 address-family ipv4
  discovery transport-address 222.1.1.1
  label local advertise explicit-null
  !
  interface ge0
  !
  interface ge1
  !
  interface ge2
  !
 exit-address-family
 !
!
l2vpn VPLS_TUNNEL type vpls
 !
 member pseudowire mpls-tunnel0
  neighbor lsr-id 222.7.7.7
  pw-id 100
 !
 member pseudowire mpls-tunnel1
  neighbor lsr-id 222.14.14.14
  pw-id 200
 !
!
end
```

4.7.9 بررسی عملکرد **VPLS**

بعد از برقراری تونل ها باید ارتباط n8 , n13, n15 از طریق تونل vpls برقرار باشد .

5.7.9 حذف تونل VPLS

برای حذف تونل vpls دستور زیر را وارد کنید. پس از آن اگر مجددا لیست تونل های vpls را مشاهده کنید مشخص است که تونل ها حذف شده اند .

```
n1(config)# no l2vpn VPLS_TUNNEL type vpls
---------
n1# sh mpls pseudowires
Interface       Neighbor              Labels      Protocol Status
```

6.7.9 فعال کردن log های vpls

با دستور زیر می توانید Log های مربوط به vpls را فعال کنید:

```
soodar1# debug vpls event
```

فصل 10

ویژگی های L2

1.10 جدول ARP

1.1.10 Dynamic ARP

برای مشاهده جدول ARP از دستور زیر استفاده می کنیم . این دستور جدول را برای همه اینترفیس ها نشان می دهد . می توانید جدول را برای اینترفیس خاص نیز مشاهده کرد :

```
soodar# show ip arp

soodar# show ip arp ge1
```

هر سطر در جدول ARP 4 ستون دارد :

Interface, L2 Address) MAC Address), L3 Address) IP Address) و State.

مقادیر state :

```
Permanent: ندارد مجدد تایید به نیازی و شود نمی منقضی زمان هیچ

Noarp: ندارد تایید به نیاز اما شود می منقضی

Reachable: شد. خواهد منقضی و است شده تایید اکنون

Stale: است نشده تایید هنوز اما است استفاده قابل

Delay: است شده اسکجول ARP درخواست

Probe: است ارسال حال در ARP درخواست

Incomplete: است. شده ارسال ARP درخواست اولین
```

برای خالی کردن جدول ARP نیز از دستور زیر استفاده می کینم :

1. پاک کردن کل جدول ARP :

```
soodar# clear ip arp
```

2. پاک کردن سطرهایی که از طریق اینترفیس ge0 آموخته شده اند :

```
soodar# clear ip arp ge0
```

3. پاک کردن سطر مربوط به اینترفیس و آدرس خاص :

```
soodar# clear ip arp ge0 192.168.1.1
```

Static ARP 2.1.10

به صورت استاتیک نیز می توان ورودی در جدول ARP اضافه کرد یا یک ورودی استاتیک را از آن حذف کرد .

```
soodar(config)# arp 1.1.1.1 00:12:13:ff:0a:4c
```

```
soodar(config)# no arp 1.1.1.1 00:12:13:ff:0a:4c
```

یا

```
soodar(config)# no arp 1.1.1.1
```

VLAN 2.10

1.2.10 آشنایی با مفهوم VLAN

VLAN یا همان Virtual LAN یا به تعبیر فارسی "شبکه های مجازی" قابلیتی است که در روتر سودار این امکان را فراهم میکند که بتوان چندین کامپیوتر را که بصورت فیزیکی به یک روتر متصل است را بصورت منطقی از هم جدا کرد و در گروههای خاص و مدنظر قرار داد.به این ترتیب هر گروه (VLAN) تبدیل به یگ گروه Broadcast جدا میشود که ترافیک آن از سایر گروه ها مجزا میشود.یکی از بارزترین مزایای استفاده از VLAN جدانمودن ترافیک بخش های مختلف از یکدیگر است.

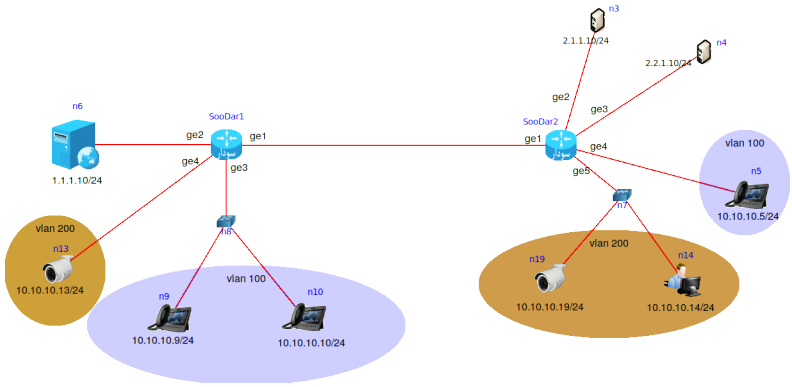
بعنوان مثال در یک شرکت ، ترافیک کامپیوترهای واحد مالی میتواند از ترافیک کامپیوترهای واحد فروش یا بازرگانی و یا هر بخش دیگری مجزا باشد.برای رسیدن به این هدف باید کامپیوترهای واحد مالی را در یک VLAN و کامپیوترهای واحد بازرگانی را در یک VLAN دیگر قرار داد.در این صورت ارتباطی بین کامپیوترهای واحد مالی و کامپیوترهای واحد بازرگانی وجود نخواهد داشت.

چنانچه در VALN ای که کامپیوترهای واحد مالی در آن قرار دارند عمل Broadcast صورت بگیرد این Broadcast فقط در بین کامپیوترهای مالی صورت گرفته و بقیه کامپیوترها از جمله کامپیوترهای بازرگانی ترافیک Broadcast مربوطه را دریافت نخواهند کرد.

2.2.10 پیکربندی VLAN

حال یک شبکه را برای نمونه vlan بندی می کنیم . در شکل زیر فرض کنید قصد داریم شبکه دوربین ها و ip Phone ها را با استفاده از vlan از یکدیگر جدا کنیم :

vlan شماره 100 را به IP Phone ها و vlan شماره 200 را به شبکه دوربین ها اختصاص می دهیم :



به صورت کلی در سناریوهای vlan اینترفیس ها در یکی از این سه حالت قرار دارند :

- اینترفیس در حالت trunk قرار دارد و در واقع یک sub interface است . به عبارت درست تر همه sub interface ها در یک bridge قرار گرفته و یک trunk را تشکیل می دهند.
- اینترفیس در حالت access قرار دارد و بسته ها بدون tag از سمت access وارد اینترفیس می شوند ودر اینترفیس باید با دستور rewrite tag push به بسته ها tag زده شود.
- بسته ها به صورت tag دار از سمت access وارد اینترفیس می شوند . در این حالت اینترفیس هیچ کاری(نه tag و نه untag) روی بسته ها انجام نمی هد و فقط به bridge مربوطه تحویل می دهد.

در سناریوی فوق اینترفیس های ge1 در هر دو روتر اینترفیس trunk هستند و بقیه اینترفیس ها هم access هستند و در اینترفیس های access با توجه به vlan ی که در آن قرار دارند باید tag مربوطه زده شود .

ساخت trunk

به ازای هر vlan باید یک sub interface بسازیم و همه این subinterface ها را در یک bridge قرار دهیم :

```
SooDar1(config)# int ge1.100
SooDar1(config-if)# encapsulation dot1q 100
SooDar1(config-if)# bridge-group 1 split-horizon group 1
SooDar1(config-if)# no shutdown
SooDar1(config-if)# q
SooDar1(config)# int ge1.200
SooDar1(config-if)# encapsulation dot1q 200
SooDar1(config-if)# bridge-group 1 split-horizon group 1
SooDar1(config-if)# no shutdown
```

```
SooDar2(config)# int ge1.100
SooDar2(config-if)# encapsulation dot1q 100
SooDar2(config-if)# bridge-group 1 split-horizon group 1
SooDar2(config-if)# no shutdown
```

(continues on next page)

(continued from previous page)

```
SooDar2(config-if)# q
SooDar2(config)# int ge1.200
SooDar2(config-if)# encapsulation dot1q 200
SooDar2(config-if)# bridge-group 1 split-horizon group 1
SooDar2(config-if)# no shutdown
```

نکته

دقت کنید جهت جلوگیری از loop حتما باید همه subinterface هایی که در یک bridge قرار دارند به همراه اینترفیس اصلی که از روی آن بقیه sub interface ها ساخته شده اند در split-horizon جداگانه قرار بگیرند تا بسته های دریافتی از یک اینترفیس مجددا در دیگر sub interface ها ارسال نشود. اینترفیس هایی که split-horizon یکسان و غیر صفر دارند بسته های دریافتی از horizon خود را باز نشر نمی کنند.

tag زدن در access

با دو دستور rewrite tag push و rewrite tag pop می توان در اینترفیس tag یا untag انجام داد . ما در اینجا در اینترفیس های سمت access باید tag بزنیم:

```
SooDar1(config)# int ge3
SooDar1(config-if)# bridge-group 1
SooDar1(config-if)# rewrite tag push 1 dot1q 100
SooDar1(config-if)# no shutdown
SooDar1(config-if)# q
SooDar1(config)# int ge4
SooDar1(config-if)# bridge-group 1
SooDar1(config-if)# rewrite tag push 1 dot1q 200
SooDar1(config-if)# no shutdown
```

```
SooDar2(config)# int ge4
SooDar2(config-if)# bridge-group 1
SooDar2(config-if)# rewrite tag push 1 dot1q 100
SooDar2(config-if)# no shutdown
SooDar2(config-if)# q
SooDar2(config)# int ge5
SooDar2(config-if)# bridge-group 1
SooDar2(config-if)# rewrite tag push 1 dot1q 200
SooDar2(config-if)# no shutdown
```

حال IP Phone ها در Vlan شماره 100 قرار می گیرند و می توانند با هم تماس داشته باشند و دوربین ها و کاربر Admin دوربین ها در Vlan شماره 200 قرار می گیرند و ارتباطشان برقرار می شود .

Inter Vlan Routing 3.2.10

اگر بخواهیم ارتباط شبکه های Vlan را با شبکه های بدون VLAN وصل کنیم در واقع routing در vlan را اضافه کنیم باید از inter vlan routing استفاده کنیم . برای routing در vlan باید یک اینترفیس bvi ایجاد کنید و آن را به bridge مربوط به vlan اضافه کنید این اینترفیس به عنوان gateway شبکه vlan استفاده می شود .

1. اینترفیس bvi

همه اینترفیس های loopback از نوع bvi هستند و برای intervlan routing باید یک اینترفیس loopback ایجاد کرده و آن را به bridge مربوطه اضافه کنید .

نکته

در زمان استفاده از اینترفیس bvi لازم است به نکاتی توجه فرمایید :

1. نمی توان همزمان دو اینترفیس looback را در یک bridge قرار داد چون در هر bridge فقط یک bvi می تواند وجود داشته باشد .
2. در اینترفیس bvi نباید بسته tag داشته باشد . زیرا بسته ها از طریق این اینترفیس به شبکه های دیگر route می شوند و باید بسته معمولی بدون تگ وارد اینترفیس شود. در نتیجه اگر بخواهیم inter vlan routing داشته باشیم باید در همه اینترفیس هایی که در bridge قرار دارند با دستور rewrite tag pop عملیات untag را انجام دهیم .

در این مثال ما اینترفیس loopback10 را به عنوان gateway و در soodar1 قرار داده ایم . به آن ip می دهیم و در host ها نیز این ip را به عنوان gateway تنظیم می کنیم . دقت شود که این ip به عنوان gateway برای هر دو روتر در نظر گرفته می شود و لازم نیست در soodar2 تنظیم دیگری برای bvi انجام دهید .

```
SooDar1(config)# int loopback 10
SooDar1(config-if)#  bridge-group 1
SooDar1(config-if)# ip address 10.10.10.1/24
SooDar1(config-if)#  no shutdown
```

2. تنظیمات routing 4.2.10

ما در اینجا از ospf برای مسیریابی استفاده کرده ایم بنابراین برای inter vlan routing تنظیمات به شکل زیر خواهد بود. توجه شود که شبکه 10.10.10.0/24 جز شبکه ای connected در SooDar1 می باشد و توسط ospf در شبکه distribute خواهد شد .

SooDar1

```
SooDar1(config)# int ge1
SooDar1(config-if)# ip address 200.1.2.1/24
SooDar1(config-if)# q
SooDar1(config)# int ge2
SooDar1(config-if)# ip address 1.1.1.1/24
SooDar1(config-if)# q
SooDar1(config)# router ospf
SooDar1(config-router)# network 200.1.2.0/24
SooDar1(config-router)# redistribute connected
SooDar1(config-if)# q
```

SooDar2

```
SooDar2(config)# int ge1
SooDar2(config-if)# ip address 200.1.2.2/24
SooDar2(config-if)# q
SooDar2(config)# int ge2
SooDar2(config-if)# ip address 2.1.1.1/24
SooDar2(config-if)# q
SooDar2(config)# int ge3
SooDar2(config-if)# ip address 2.2.1.1/24
SooDar2(config-if)# q
SooDar2(config)# router ospf
SooDar2(config-router)# network 200.1.2.0/24
SooDar2(config-router)# redistribute connected
SooDar2(config-if)# q
```

5.2.10 بررسی ارتباط Vlan

نکته

پس از تنظیمات فوق باید ارتباطات به شکل زیر برقرار باشد :
1. باید ارتباط بین نود های n9,n10 , n5 از طریق vlan شماره 100 برقرار باشد.
2. همچنین نود (Admin) n14 به دوربین ها n13,n19 دسترسی داشته باشد .
3. دوربین ها و IPPhone ها باید با استفاده از ویژگی inter valn routing به (server n6 (1.1.1.10 دسترسی داشته باشند .
4. شبکه دوربین ها و تلفن های ip باید از یکدیگر جدا باشد و دسترسی به یکدیگر نداشته باشند .

6.2.10 vlan in vlan (QinQ)

شما می توانید درون یک vlan نیز با استفاده از tag دوم vlan جدیدی تعریف کنید . روش تنظیم vlan با دو تگ به شکل زیر است :

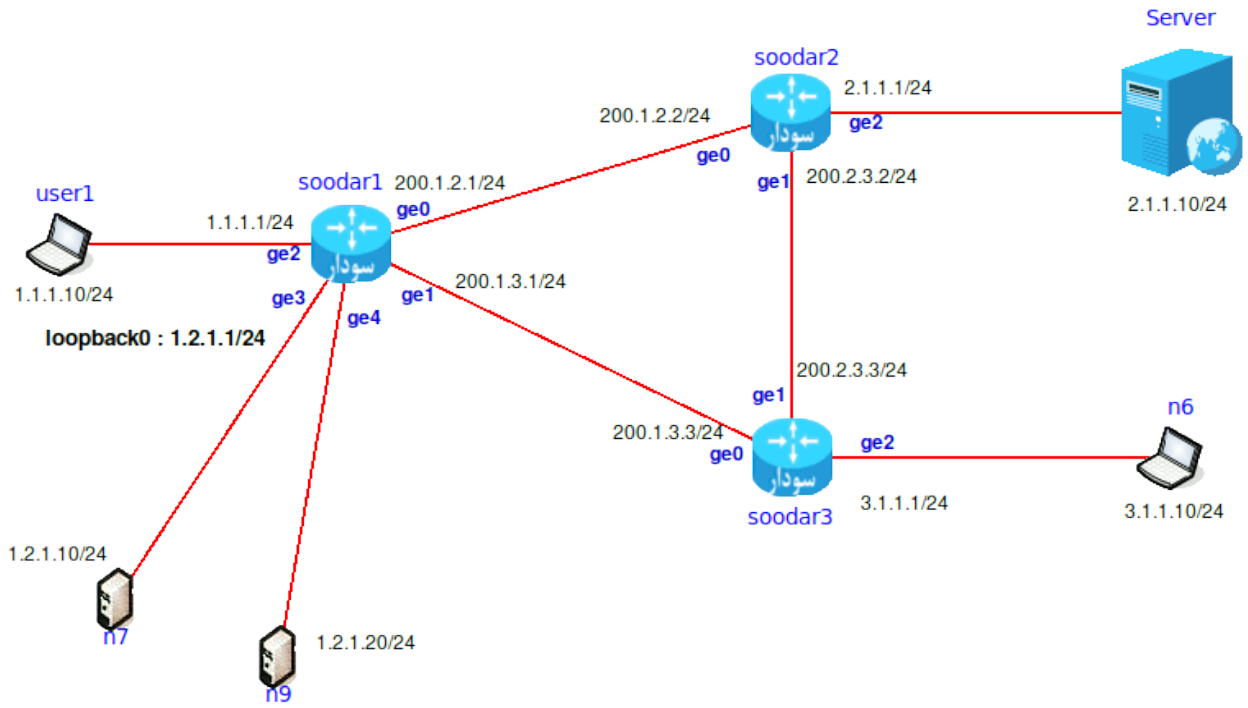
```
SooDar(config-if)# encapsulation dot1q 100 second-dot1q 110
```

7.2.10 فعال کردن log های VLAN

با دستور زیر می توانید Log های مربوط به vlan را فعال کنید:

```
soodar1# debug vlan event
```

Bridge 3.10



1.3.10 اضافه کردن اینترفیس در bridge

برای اضافه کردن اینترفیس ها در bridge به صورت زیر عمل می کنیم:

```
soodar1(config)#  int ge3
soodar1(config-if)#  bridge-group 150
soodar1(config-if)# no shutdown
soodar1(config)#  int ge4
soodar1(config-if)#  bridge-group 150
soodar1(config-if)# no shutdown
```

2.3.10 مشاهده bridge

برای مشاهده bridge می توان از دستور زیر استفاده کرد :

```
soodar1# show bridge 150
|-----------------------------------------------|
| Domain | Interface | Split-Horizon Group | BVI |
|--------+-----------+---------------------+-----|
```

(continues on next page)

(continued from previous page)

```
| 150   | ge3      | 0                  | -  |
|-------+----------+--------------------+----|
|       | ge4      | 0                  | -  |
|-------+----------+--------------------+----|
```

بدین ترتیب این دو اینترفیس به همدیگر bridge می شوند و ping بین دو نود 1.2.1.20 , 1.2.1.10 برقرار می شود .

3.3.10 اضافه کردن اینترفیس bvi

برای برقراری ارتباط بین این دو نود و شبکه های دیگر لازم است از bvi استفاده کنیم . یک اینترفیس loopback ایجاد می کنیم (loopback) و آن را به bridge که دو اینترفیس ge4 , ge3 قرار دارند اضافه می کینم . اینترفیس loopback به طور خودکار از نوع bvi می شود و با اضافه کردن ip می تواند به عنوان gateway استفاده شود . در این مثال loopback0 به عنوان gateway شبکه 1.2.1.0/24 می تواند استفاده شود .

```
soodar1(config)# interface loopback 0
soodar1(config-if)#  bridge-group 150
soodar1(config-if)#  ip address 1.2.1.1/24
soodar1(config)# do sh bridge 150
```

4.3.10 حذف اینترفیس از bridge

```
soodar1(config)#  int ge1
soodar1(config-if)# no bridge-group 150
soodar1(config)#  int ge2
soodar1(config-if)# no bridge-group 150
```

5.3.10 split-horizon

برای جلوگیری از بروز loop در زمان استفاده از bridge در لایه 2 می توان از split-horizon استفاده کرد . بدین ترتیب که اینترفیس هایی که در یک split-horizon قرار می گیرند پیام های یکدیگر را دریافت نمی کنند و loop رخ نمی دهد . به طور مثال اگر سه اینترفیس در یک bridge قرار بگیرند و مقدار split-horizon یکسان و غیر صفر داشته باشند بسته ای که از یک اینترفیس دریافت می شود به اینترفیس های دیگری که در آن split-horizon هستند ارسال نمی شود.

بصورت پیش فرض همه اینترفیس هایی که به bridge اضافه می شوند در horizon 0 قرار می گیرند یعنی این که اینترفیس در هیچ horizon قرار ندارد و محدودیتی برای ارسال بسته بین اینترفیس ها وجود ندارد و مثلا بسته ها بردوکست به همه اینترفیس های این sh ارسال می شود.

```
soodar1(config)#  int ge1
soodar1(config-if)#  bridge-group 150 split-horizon group 25
soodar1(config)#  int ge2
soodar1(config-if)# no bridge-group 150 split-horizon group 25


soodar1# show bridge 150
|-------------------------------------------|
| Domain | Interface | Split-Horizon Group | BVI |
```

(continues on next page)

(continued from previous page)

```
|--------+----------+----------------------+-----|
| 150    | ge1      | 25                   | -   |
|--------+----------+----------------------+-----|
|        | ge2      | 25                   | -   |
|--------+----------+----------------------+-----|
```

6.3.10 فعال کردن log های Bridge

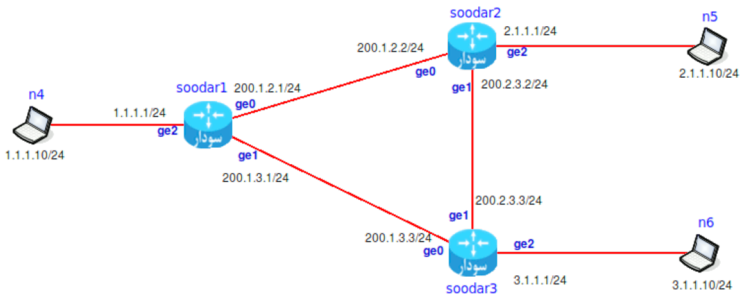
با دستور زیر می توانید Log های مربوط به bridge را فعال کنید:

```
soodar1# debug bridge event
```

SPAN 4.10

1.4.10 آشنایی با مفهوم SPAN

2.4.10 آموزش راه اندازی SPAN Port در سودار



مثال : ترافیک ge0 روی ge2 با دو دستور زیر قابل مشاهده است .

```
soodar2(config)# monitor session 12 source interface ge0
soodar2(config)# monitor session 12 destination interface ge2
soodar2(config)# interface ge2
soodar2(config-if)# no shutdown
```

3.4.10 مشاهده ترافیک ورودی یا خروجی

فقط ترافیک ورودی به اینترفیس ge0 به span port ارسال شود :

```
soodar2(config)# monitor session 12 source interface ge0  rx
```

فقط ترافیک خروجی از اینترفیس ge0 به span port ارسال شود :

```
soodar2(config)# monitor session 12 source interface ge0  tx
```

هم ترافیک ورودی و هم ترافیک خروجی در اینترفیس ge0 به span port ارسال شود :

```
soodar2(config)# monitor session 12 source interface ge0  both
```

4.4.10 حذف پورت **span**

```
soodar2(config)# no monitor session 12
```

5.4.10 **debug** کردن **span**

با دستور زیر می توانید Log های مربوط به span را فعال کنید:

```
soodar2# debug span event
```

فصل 11

Scriptable CLI

Sooshell 1.11

پوسته کاربری sooshell با بهره گیری از قابلیت های zsh امکانات جدیدی را در اختیار کاربران قرار میدهد. در sooshell شما در آن واحد هم به دستورات پیکربندی روتر سودار دسترسی دارید، و هم از ساختار های نحوی شل های POSIX میتوانید استفاده کنید. sooshell چه در زمینه کاربرد تعاملی و چه در زمینه script نویسی قابلیت های جدیدی را به کاربران سودار و مدیران شبکه میدهد که از طریق آن میتوانید کارایی خود را چند برابر کنید.

sooshell ویژگی های منحصر به فردی برای استفاده تعاملی کاربران ارائه کرده که کار کردن با CLI را آسان تر، کارآمد تر و لذت بخش تر میکند.

1.1.11 نمایش نام در Prompt

پرامت sooshell وضعیت دقیق نشست کاربری را از لحاظ مسیری پیکربندی که در آن هستیم مشخص میکند:

```
router1/c/interface/ge0#
```

به عنوان نمونه پرامت بالا نشان میدهید که ما وارد مسیر

enable

<

config

< interface : ge0 شده ایم.

```
router1/c/i/g/link-params#
```

یا در اینجا وارد

enable

<

config

<

interface : ge0

< link-params شده ایم.

این ویژگی باعث می شود که admin همیشه بداند که در حال تنظیم کردن کدام بخش هست . اگر در حال تنظیم اینترفیس هست نام اینترفیس در prompt وجود دارد یا اگر دارد acl تنظیم می کند نام acl در prompt مشخص است و باعث می شود که خطای admin کاهش یابد و مثلا اینترفیسی را اشتباهی shutdown نکند یا رولی را در acl ی حذف و اضافه نکند . این به اصطلاح context در همه مسیرهای کانفیگ وجود دارد . در class-map,policy-map,route-map,access-list,interface,ike-config,ipsec-config.... .

نکته

وقتی prompt طولانی باشد تنها حرف اول هر بخش در prompt نوشته می شود در مثال بالا c/i/g که به ترتیب نشانگر config,interface و ge0 می باشد . هر جا که نیاز داشتید که prompt را به شکل کامل ببینید می توانید از دستور pwd استفاده کنید :

```
n1/c/i/g/link-params# pwd
/enable/config/interface/ge0/link-params/
n1/c/i/g/link-params#
```

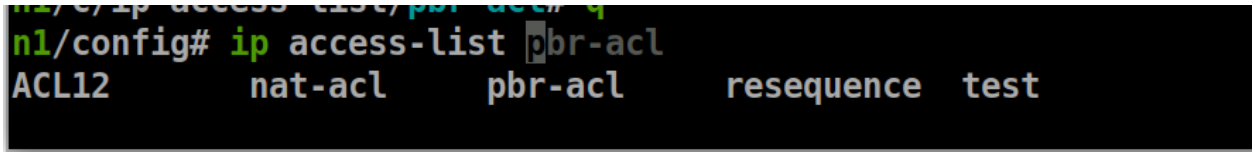
2.1.11 تکمیل خودکار

با استفاده از قابلیت تکمیل خودکار میتوان از نوشتن کامل دستورات خود داری و در زمان صرفه جویی کرد. برای کامل کردن یک کلمه کافیست کلید TAB را فشار دهید تا sooshell کلمات ممکن برای تکمیل دستورات را به شما پیشنهاد دهد:

```
router1/config# p<TAB>

password    pbr    pbr-map    pseudowire
```

میتوان با فشردن کلیدهای جهت دار □□□□ یا همان کلید TAB یکی از گزینه ها را انتخاب نمود. همچنین اگر sooshell فقط یک حالت ممکن برای تکمیل کردن دستور پیدا کند، بدون نیاز به فشردن مجدد کلید دیگری آن را بطور خودکار کامل میکند.

```
router1/config# ps<TAB>
router1/config# pseudowire
```



3.1.11 تاریخچه

sooshell لیستی از دستوراتی که قبلا وارد کرده اید را ذخیره میکند. تاریخچه دستورات با بستن سشن از بین نمیرد و در اجرای بعدی sooshell نیز در دسترس خواهند بود. با فشردن کلید های جهت دار بالا و پایین □□ میتوان تاریخچه را مشاهده کرد و دستوراتی که قبلا وارد شده اند را مجددا اجرا کنید.

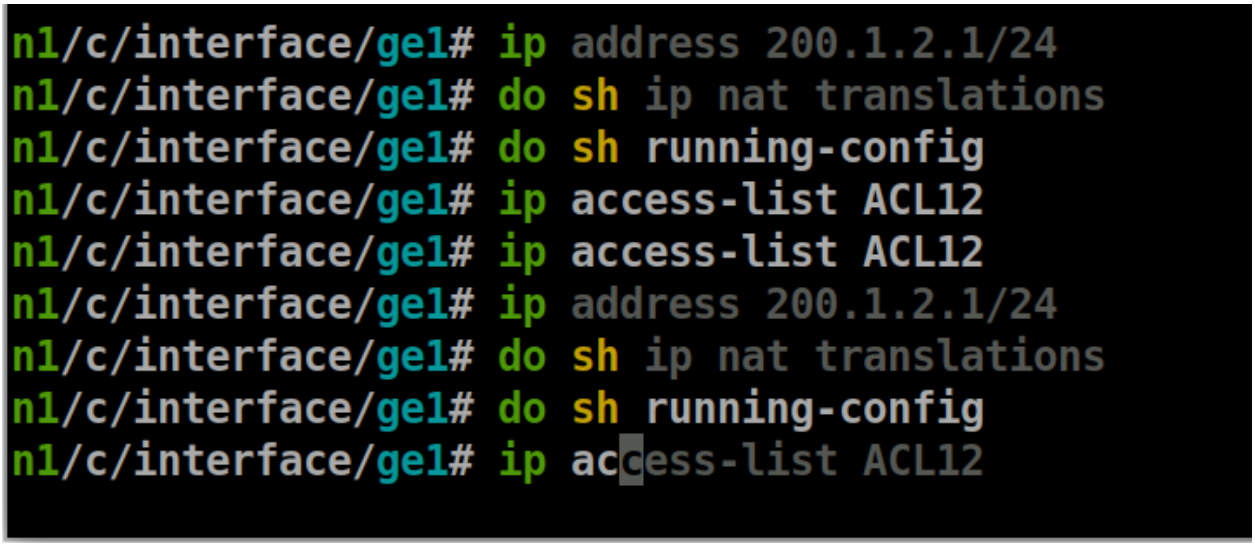
4.1.11 هایلایت

sooshell با رنگی کردن کلمه ابتدایی دستورات نشان میدهد که آیا دستور وارد شده صحیح است یا خیر:

- رنگ قرمز: چنین دستوری وجود ندارد
- رنگ سبز: دستور وجود دارد
- رنگ زرد: دستور تا اینجایی که تایپ شده مبهم است و لازم است مقدار بیشتری از آن تایپ شود
- رنگ آبی: دستور وجود دارد، اما در مسیر پیکربندی قبلی بوده و اجرای آن باعث میشود تا از مسیر فعلی خارج شویم.

5.1.11 پیشنهاد خودکار

sooshell همزمان با تایپ کردن کاربر بر اساس تاریخچه دستوراتی که کاربر وارد کرده و ورودی که تا آن لحظه تایپ کرده، یک دستور را به صورت کمرنگ در جلوی ورودی کاربر پیشنهاد میدهد. کاربر میتواند با فشردن کلید جهت دار سمت راست آن پیشنهاد را انتخاب کند و یا در غیر این صورت به تایپ کردن خود ادامه دهد.



6.1.11 نویسی script

قابلیت script نویسی sooshell موجب میشود تا بتوانیم بسیاری از کارهای دستی خسته کننده را به صورت اتوماتیک انجام دهیم. سینتکس script های sooshell همان سینتکس شل های POSIX در سیستم عامل های یونیکسی نظیر Bash و Zsh میباشد. قابل برنامه نویسی بودن sooshell این امکان را به مدیران شبکه میدهد تا برای اتوماسیون پیکربندی و هر گونه روال دلخواهی بتوانند script مورد نیاز خود را بنویسند و اجرا کنند.

script ها را به شکل تعاملی نیز میتوان اجرا کرد. کافی است کد اسکریپتی که از قبل نوشته اید را در ترمینال paste کنید تا اجرا شود. اما روش کارآمد تر اجرای script ها از طریق ذخیره کردن و فراخوانی آنها با نام آنها میباشد که در ادامه به آن میپردازیم.

ذخیره و فراخوانی اسکریپت

TODO

در اینجا با مثال های متعدد نشان خواهیم داد که چگونه حداکثر استفاده را از قابلیتهای script نویسی sooshell ببریم:

تعریف متغیر

تعریف متغیر ها در متن دستوراتی که اجرا میکنم، این امکان را به ما میدهد تا بتوانیم صرفا با تغییر دادن مقدار آن متغیر، دستورات خود را به ازای مقادیر مختلفی اجرا کنیم، بدون آنکه نیاز باشد خود دستور را تغییر دهیم.

```
# set variable IFNAME to ge0
IFNAME=ge0
# following command evaluates to 'interface ge0'
interface $IFNAME
```

پاک کردن متغیر:

```
# delete IFNAME
unset IFNAME
```

لازم به ذکر است که خط هایی که با # شروع میشوند کامنت هستند.

سینتکس اجرای دستورات

میتوان دستورات فوق را در یک خط نوشت:

```
IFNAME=ge0; interface $IFNAME && { no shut; quit; }
```

- دستورات توسط کاراکتر ؛ از یکدیگر جدا شده اند.
- سری && نیز مانند ؛ عمل میکند، با این تفاوت که دستور دوم تنها در صورتی اجرا میشود که دستور اول با موفقیت اجرا شده باشد.
- کاراکتر های {} یک یا چند دستور را در یک بلوک قرار میدهند.

در مثال بالا اگر مقداری که در متغیر IFNAME قرار داده باشیم نام یک اینترفیس موجود نباشد، دستور interface $IFNAME شکست میخورد و دستورات قرار داده شده در بلوک اجرا نمیشوند.

پردازش خروجی

دستور زیر را در نظر بگیرید:

```
router1# show interface brief
Interface       Status  VRF             Addresses
---------       ------  ---             ---------
lo              up      default
ge0             up      default         192.168.1.1/24
```

(continues on next page)

(continued from previous page)

```
ge1          up     default       192.168.2.1/24
wg0          down   default       10.10.49.1/16
```

میخواهیم فقط اطلاعات مربوط به ge1را دریافت کنیم . sooshell این امکان را به ما میدهد تا خروجی یک دستور را به عنوان ورودی دستور دیگری قرار دهیم. دستور grep میتواند در ورودی خود به دنبال یک الگو بگردد. کافی است نام اینترفیس مورد نظرمان را همراه با ورودی دستور قبلی به grep بدهیم:

```
router1# show interface brief | grep ge1
ge1          up     default       192.168.2.1/24
```

دستور دیگری که میتوان از آن استفاده کرد دستور awk است. این دستور قابلیت جست و جوی پیشرفته تری را به کاربر ارائه میدهد. مثلا میخواهیم آدرس آی پی اینترفیس ge1 را پیدا کنیم:

```
router1# show interface brief | awk 'NR > 1 && $1 == "ge1" { print $4 }'
192.168.2.1/24
```

این احتمال نیز وجود دارد که بخواهیم خروجی یک دستور را در یک متغیر ذخیره کنیم تا بعدا از آن استفاده کنیم که این کار با با کمک سینتکس VAR=$(COMMAND) انجام میدهیم:

```
ip_ge1=$(show interface brief | awk '$1 == "ge1" { print $4 }')
```

عملیات شرطی

گاهی نیاز داریم برخی دستورات فقط در شرایط خاصی اجرا شوند. در sooshell هر شرط خود در واقع یک دستور است که موفقیت آمیز بودن اجرای دستور به منزله برقرار بودن شرط میباشد. فرض کنید میخواهیم همه اینترفیس ها را در صورتی که تعدادشان از 5 تا کمتر باشد نشان دهیم:

```
# get all rows except the first 2 heading rows
interfaces=$(show interface brief | tail +3)
# count the number of rows
count=$(wc -l <<< $interfaces)
# check if we have less than 5 rows
if [[ $count -lt 5 ]]
then
    # print all rows
    echo $interfaces
fi
```

- در خط اول با استفاده از دستور tail فقط از خط سوم به بعد را ذخیره میکنیم.
- تعداد ردیف ها با استفاده از دستور wc -l بدست می آوریم.
- در خط بعدی از ساختار if استفاده میکنیم تا شرط ذکر شده را بررسی کنیم
- اگر شرط برقرار بود همه ردیف ها را چاپ میکنیم

با سینتکس variable$ <<< command arg1 arg2 میتوان مقدار یک متغییر را به عنوان ورودی به یک دستور داد.

سینتکس کلی ساختار کنترلی شرطی به این صورت است:

```
if command1
then
    command2
    command3
fi
```

که در صورت موفقیت آمیز بودن دستور اول، هر دو دستور بعدی اجرا میشوند. در مثالی که ذکر کردیم دستور اول همان [[$count -lt 5]] میباشد. رشته [[در واقع یک دستور است که با آرگومان های $count,-lt,5 فراخوانی شده. آرگومان دوم آن یعنی -lt به معنی کوچکتر میباشد و دستور در صورتی که آرگومان اول از سومی کوچکتر باشد موفقیت آمیز بوده و شرط برقرار میشود.

اکنون فرض کنید بخواهیم اگر تعداد اینترفیس ها بین 5 الی 10 دستور بود فقط اسم اینترفیس ها و اگر بیشتر بود فقط تعداد آنها چاپ شود:

```
# get all rows except the first 2 heading rows
interfaces=$(show interface brief | tail +3)
# count the number of rows
count=$(wc -l <<< $interfaces);
# check if we have less than 5 rows
if [[ $count -lt 5 ]]
then
    # print all rows
    echo $interfaces
elif [[ $count -lt 10 ]]
    # print first column of all rows
    echo $interfaces | awk '{print $1}'
else
    echo "number of interfaces: $count"
fi
```

سینتکس کامل ساختار کنترلی شرطی به این صورت است:

```
if cond1
then
    command1-1
    command1-2
    ...
elif cond2
then
    command2-1
    command2-2
    ...
elif...
fi
```

دیگر عملگرهای شرطی:

- آیا طول رشته a بیشتر از صفر است؟ [[-n a]]
- آیا طول رشته a برابر با صفر است؟ [[-z a]]
- آیا رشته های a و b با یکدیگر برابر اند؟ [[a == b]]
- آیا رشته های a و b با یکدیگر برابر نیستند؟ [[a != b]]
- آیا رشته a با رگکس b مچ میشود؟ [[a =~ b]]
  [[b lrtextasciitilde]]
- آیا عدد a کوچکتر از عدد b است؟ [[a -lt b]]
- آیا عدد a بزرگتر از عدد b است؟ [[a -gt b]]
- آیا عدد a کوچکتر-مساوی عدد b است؟ [[a -le b]]
- آیا عدد a بزرگتر-مساوی عدد b است؟ [[a -ge b]]
- آیا عدد a مساوی با عدد b است؟ [[a -eq b]]
- آیا عدد a نامساوی با عدد b است؟ [[a -ne b]]

7.1.11 دستورات کمکی

هنگامی که در حال تعامل با sooshell یا اجرای script هستید، همواره در یک مسیر پیکربندی قرار دارید.

```
router1/c/interface/ge0# node
interface
```

با این دستور میتوانید نام نودی که در آن هستید را بدست بیاورید. اگر آن نود اطلاعات اضافه ای داشته باشد نیز میتوان آن را بدست آورد:

```
router1/c/interface/ge0# node -x
ge0
```

ضمنا میتوان از دستور شرطی زیر نیز برای بررسی اینکه در نود خاصی هستیم یا خیر استفاده کنیم:

```
# check if we're in node
[[ -in config ]]
```

مثال:

```
if [[ ! -in config ]]
then
    echo can\'t run from $(node)
    echo must be in config
    return 1
fi
# or simply
[[ ! -in config ]] && return 1
```

8.1.11 حلقه for

در sooshell script برای انجام کارهای تکراری میتوان از حلقه تکرار استفاده نمود. حلقه for در sooshell سینتکس ساده ای دارد و با استفاده از آن میتوان به ازای بازه ای از اعداد یک کار را انجام داد. مثلا در اینجا میخواهیم دستور مربوط به اضافه کردن نت استاتیکی به ازای اعداد 1200 تا 1400 تکرار شود (به ازای خود 1200 و 1400 هم اجرا میشود):

```
for i in {1200..1400}; do
    ip nat inside static source tcp "1.1.1.10" $i "200.1.2.2" $i
done
```

در این حلقه متغیر i همان اندیس حلقه است که در دستورات درون حلقه میتوان از آن استفاده کرد.

گاهی میخواهیم به ازای هر عنصر در یک آرایه دستوراتی اجرا شوند. در آن صورت مانند مثال زیر عمل میکنیم:

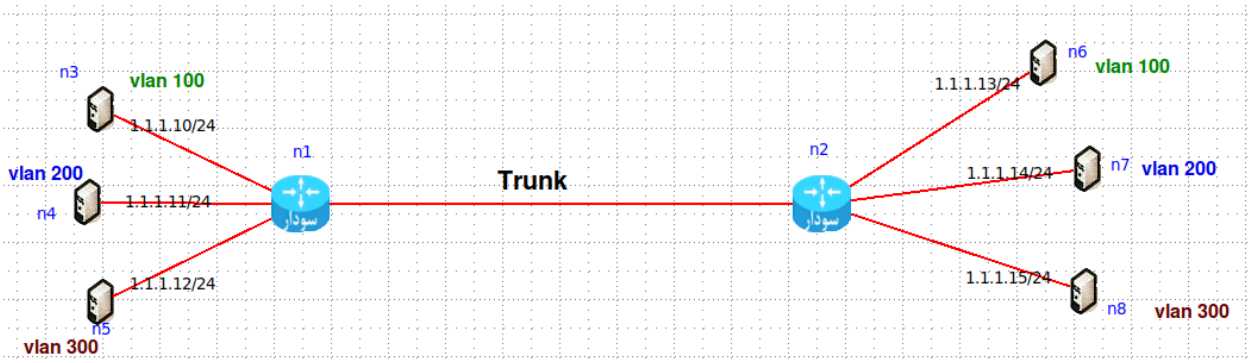
```
# Define variables for VLAN configuration
vlan_ids=(100 200 300)
vlan_ifname="ge0"

# Loop through VLAN configuration
for idx in $vlan_ids; do
    vlan_name="$vlan_ifname.$idx"
    echo "Configuring VLAN $vlan_name ..."
    interface "$vlan_name"
    encapsulation dot1q $idx
```

(continues on next page)

(continued from previous page)

```
    bridge-group $idx
    no shutdown
    exit
    # tag rewrite in access side
    ifidx=$((idx / 100))
    interface ge$ifidx
    no ip address
    rewrite tag push 1 dot1q $idx
    bridge-group $idx
done

echo "VLAN configuration completed."
```

برای مثال با استفاده از script بالا می توان تنظیم vlan را در روتر های n1,n2 انجام داد . در این مثال اینترفیس ge0 سمت Trunk و اینترفیس های ge1,ge2و ge3 به ترتیب مربوط به vlan های 100 و200 و 300 هستند :




```
n1/config# vlan_ids=(100 200 300)
n1/config# vlan_ifname="ge0"
n1/config#
n1/config# # Loop through VLAN configuration
n1/config# for idx in $vlan_ids; do
for>     vlan_name="$vlan_ifname.$idx"
for>     echo "Configuring VLAN $vlan_name ..."
for>     interface "$vlan_name"
for>     encapsulation dot1q $idx
for>     bridge-group $idx
for>     no shutdown
for>     exit
for>     # tag rewrite in access side
for>     ifidx=$((idx / 100))
for>     interface ge$ifidx
for>     no ip address
for>     rewrite tag push 1 dot1q $idx
for>     bridge-group $idx
for> done
Configuring VLAN ge0.100 ...
Configuring VLAN ge0.200 ...
Configuring VLAN ge0.300 ...
n1/c/interface/ge3#
```

همین script را هم می توان در n2 اجرا کرد و ارتباط بین vlan ها را برقرار کرد .

در این مثال دستورات درون حلقه 3 بار اجرا میشوند، که یکبار idx برابر با 100، یکبار 200 و یکبار هم 300 میباشد.

مثال حذف تمامی sub interface ها

برای حذف sub interface ها ابتدا باید لیست آنها را بدست آورده و سپس shutdown کنیم و در آخر آن ها را حذف می کنیم

```
subifs=($(do show int brief | grep -E ".+\..+" | awk '{print $1}'))

for if in $subifs; do
  interface $if
    shutdown
    q
  no int $if
done
```

9.1.11 حلقه **while**

نوعی دیگر از حلقه تکرار حلقه های while هستند که تا زمانی که شرط آنها برقرار باشد دستورات درون آنها اجرا میشوند. در مثال زیر اسکریپتی نوشته شده که نام یک پروتکل را از ورودی میخواند و تا مادامی که ورودی خالی نباشد، آن پروتکا را deny میکند و پروتکل بعدی را از ورودی میگیرد:

```
# read protocol names (one per line) and deny them in current ACL
[[ -in ip-access-list ]] || {
    echo Error: Must be in ACL node 1>&2
    return 1
}
while read PROTOCOL && [[ -n $PROTOCOL ]]
do
    deny $PROTOCOL any any
    echo denied protocol $PROTOCOL in ACL $(node -x)
done
permit any
```

دستور read یک خط از ورودی میخواند و آن را در متغیری که نامش را به عنوان آرگومان دریافت کرده (در اینجا Protocol) قرار میدهد. اگر ورودی خالی دریافت کند، دستور شکست میخورد و شرط حلقه برقرار نمیشود.

10.1.11 جست و جو در کانفیگ

گاهی اوقات پیش می آید که که برای یک ip خاص یا یک سری از ip ها تنظیماتی در ACL ها و NAT و بخش های دیگر کانفیگ انجام داده ایم و این تنظیمات بزرگ و قدیمی است و دقیقا یادمان نیست چطور کانفیگ کرده ایم . حال اگر بخواهیم مثلا کلیه تنظیماتی که برای 192.168.1.24 انجام داده ایم برای 192.168.1.33 هم انجام دهیم کار کمی سخت خواهد بود .

با استفاده از ویژگی جست و جوی پیشرفته در کانفیگ می توانیم جاهایی که تنظیمی برای 192.168.1.24 انجام شده را فیلتر کنیم و دقیقا همان تنظیم برای 192.168.1.33 انجام دهیم . در این فیلتر کردن کانفیگ می توانیم مشخص چند خط بعد و قبل از عبارت مورد نظر را هم برای ما نشان دهد . همچنین می توانیم عبارت مورد نظر را با regex مشخص کنیم :

```
sh running-config | grep 192.168.1.24 -B 10 -A 5
```

```
 wireguard port 1403
 wireguard peer tehran-site
  public-key DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDI
  endpoint 0.0.0.0 port 0
  allowed-ip 10.101.10.0/24
  allowed-ip 10.0.0.98/32
 ip address 10.0.0.99/32
exit
!
ip access-list nat-acl
 remark used for nat
 1 deny 10.10.110.0/24 192.168.1.24/32
 4 deny 192.168.30.4/32 94.182.18.201/32
 10 deny 172.140.105.152/29 any
--
 remark tehran-site
 3 permit 10.10.100.40/32 10.10.170.102/32 reflect
 4 permit 10.10.100.0/24 10.0.0.198/32 reflect
 5 permit 10.10.100.0/24 10.10.170.0/24 reflect
 10 permit icmp 10.0.0.199/32 10.0.0.198/32 reflect
 15 permit tcp 10.10.170.0/24 10.10.100.222/32 eq 443 reflect
 20 permit tcp 10.0.0.199/32 10.0.0.198/32 eq 22 reflect
 21 permit tcp 10.10.100.0/24 10.0.0.198/32 eq 22 reflect
 22 permit tcp 10.20.100.0/24 10.0.0.198/32 eq 22 reflect
 25 permit 10.0.0.198/32 10.10.160.10/32 reflect
 26 permit 10.0.0.198/32 10.10.100.39/32 reflect
 30 permit 192.168.1.24/32 172.140.105.154/32
 40 permit 192.168.1.24/32 10.10.110.0/24
 41 permit 172.22.15.154/32 192.168.1.24/32
 42 permit 10.10.110.0/24 192.168.1.24/32
 43 permit 10.10.140.45/32 10.10.170.0/24
 44 permit 10.10.170.0/24 10.10.140.45/32
 45 deny any any
```

11.1.11 جست و جو در لاگ ها

زمانی که حجم لاگ ها زیاد باشد بررسی log ها و پیدا کردن مشکل مورد نظر سخت خواهد بود شما با استفاده از modifier ها می توانید بخش های خاصی از لاگ ها را بررسی کند و به جای بررسی خط به خط لاگ ها که هم زمانبر خواهد بود و هم خسته کننده ، شما می تواند=ید با استفاده از regex ها الگوی مورد نظر را در لاگ ها جست و جو کنید . برای مثال اگر بخواهیم لاگ مربوط به قطع و وصل شدن تونل های وایرگارد را مشاهده کنیم کافی است لاگ ها را به شکل زیر با استفاده از regex فیلتر کنیم :

```
sh log all | grep -i -E "wireguard.+peer.+ (connected|disconnected)"
```

```
Jun 27 21:14:21 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jun 28 10:34:18 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jun 28 21:26:48 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jun 29 07:37:11 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jun 29 17:18:26 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jun 30 08:09:43 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jun 30 17:56:43 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jul 01 08:14:20 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jul 01 20:22:34 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jul 02 08:10:33 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jul 02 20:52:22 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jul 03 06:31:55 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
Jul 03 20:05:22 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 disconnected
Jul 04 06:38:25 Soodar-router zebra[675]: Wireguard peer MSHD@wireguard403 connected
```

5 خط قبل و 3 خط بعد از این لاگ ها هم چاپ شود تا لاگ های قبل و بعد از این الگو را هم داشته باشیم : به شکل زیر میتوان تعیین کرد که

```
sh log all | grep -i -E "wireguard.+peer.+ (connected|disconnected)" -B 5 -A 3
```

```
n1# sh log all | grep -i -E "wireguard.+peer.+ (connected|disconnected)" -B 5 -A 3
2024/04/09 13:22:10 ZEBRA: Wireguard peer node2@wireguard10 changed. src=200.1.2.1, dst=0.0.0.0:6060, state=0
2024/04/09 13:22:10 ZEBRA: vici_reconnect: failure connecting VICI socket: No such file or directory
2024/04/09 13:22:12 ZEBRA: VICI: Connected
2024/04/09 13:22:19 ZEBRA: root@CONSOLE> en
2024/04/09 13:22:29 ZEBRA: root@CONSOLE# sh int brief
2024/04/09 13:22:34 ZEBRA: Wireguard peer node2@wireguard10 connected
2024/04/09 13:22:34 ZEBRA: Wireguard peer node2@wireguard10 changed. src=200.1.2.1, dst=200.1.2.2:6060, state=1
2024/04/09 13:23:44 ZEBRA: root@CONSOLE> en
2024/04/09 13:23:46 ZEBRA: root@CONSOLE# sh ip arp
--
2024/04/27 14:38:45 ZEBRA: VICI: Connected
2024/04/27 14:38:57 ZEBRA: root@CONSOLE> en
2024/04/27 14:39:05 ZEBRA: root@CONSOLE# sh wireguard
2024/04/27 14:39:07 ZEBRA: root@CONSOLE# sh wireguard
2024/04/27 14:39:07 ZEBRA: root@CONSOLE# sh wireguard
2024/04/27 14:39:09 ZEBRA: Wireguard peer node2@wireguard10 connected
2024/04/27 14:39:09 ZEBRA: Wireguard peer node2@wireguard10 changed. src=200.1.2.1, dst=200.1.2.2:6060, state=1
2024/04/27 14:39:14 ZEBRA: root@CONSOLE# sh wireguard
2024/04/27 14:39:34 ZEBRA: root@CONSOLE# sh wireguard
--
2024/07/07 10:11:39 ZEBRA: Wireguard peer node2@wireguard10 changed. src=200.1.2.1, dst=0.0.0.0:6060, state=0
2024/07/07 10:11:39 ZEBRA: vici_reconnect: failure connecting VICI socket: No such file or directory
2024/07/07 10:11:41 ZEBRA: VICI: Connected
2024/07/07 10:11:47 ZEBRA: vtysh> en

2024/07/07 10:12:03 ZEBRA: Wireguard peer node2@wireguard10 connected
2024/07/07 10:12:03 ZEBRA: Wireguard peer node2@wireguard10 changed. src=200.1.2.1, dst=200.1.2.2:6060, state=1
2024/07/07 10:12:09 ZEBRA: root@CONSOLE# c

n1#
```

یا در مثال زیر لاگهای اتصالات موفق و ناموفق ssh به روتر را بدین شکل از بین لاگ ها استخراج می کنیم :

```
PPPoE-Client# sh log ssh | grep -i -E "(Accepted|Failed) password"
Jun 02 11:59:46 router sshd[2977]: Accepted password for sadmin from 10.10.100.39 port 38988 ssh2
Jun 02 12:36:48 router sshd[647]: Accepted password for admin from 10.10.100.39 port 42576 ssh2
Jun 02 12:45:49 newroterpppor sshd[5671]: Accepted password for admin from 10.10.100.39 port 39866 ssh2
Jun 02 12:47:03 newroterpppor sshd[6406]: Accepted password for admin from 10.10.100.39 port 54300 ssh2
Jun 02 12:47:04 newroterpppor sshd[6539]: Accepted password for sadmin from 10.10.100.39 port 54308 ssh2
Jun 02 12:47:06 newroterpppor sshd[6624]: Accepted password for sadmin from 10.10.100.39 port 54312 ssh2
Jun 02 12:47:07 newroterpppor sshd[6702]: Accepted password for sadmin from 10.10.100.39 port 54320 ssh2
Jun 02 12:47:07 newroterpppor sshd[6806]: Accepted password for sadmin from 10.10.100.39 port 54326 ssh2
Jun 02 12:47:07 newroterpppor sshd[6884]: Accepted password for sadmin from 10.10.100.39 port 54340 ssh2
Jun 02 12:47:08 newroterpppor sshd[6962]: Accepted password for sadmin from 10.10.100.39 port 54350 ssh2
Jun 02 12:47:46 router sshd[631]: Accepted password for admin from 10.10.100.39 port 38116 ssh2
Jun 02 13:08:10 PPPoE-Client sshd[4763]: Accepted password for admin from 10.10.100.39 port 33710 ssh2
Jun 02 14:09:59 PPPoE-Client sshd[33806]: Accepted password for admin from 10.10.100.39 port 57160 ssh2
Jun 02 14:11:29 PPPoE-Client sshd[35517]: Accepted password for admin from 10.10.100.30 port 54236 ssh2
Jun 05 08:29:04 PPPoE-Client sshd[29437]: Accepted password for admin from 192.168.30.4 port 57876 ssh2
Jun 05 08:39:19 PPPoE-Client sshd[36285]: Accepted password for admin from 192.168.30.4 port 56698 ssh2
Jun 05 08:39:33 PPPoE-Client sshd[36474]: Accepted password for admin from 192.168.30.4 port 53174 ssh2
Jul 08 13:43:06 PPPoE-Client sshd[386496]: Failed password for admin from 10.10.100.39 port 58172 ssh2
PPPoE-Client#
```

در مثالی دیگر up/down شدن اینترفیس ها را در لاگها فیلتر می کنیم :

```
n1# sh log all | grep -iE "interface.+ (up|down)"
2024/04/29 10:17:20 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:17:39 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:18:07 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:18:44 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:19:10 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:19:27 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:33:25 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:33:44 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:34:17 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:34:54 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:35:01 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:35:18 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:37:10 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:37:29 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:38:02 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:41:47 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:42:05 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:42:39 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:44:14 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:44:32 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:45:05 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:45:42 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:45:49 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:46:05 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/30 08:35:53 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:36:12 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/30 08:36:41 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:38:28 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:38:47 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/30 08:39:23 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:40:00 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/30 08:40:07 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:40:25 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/30 08:40:25 ZEBRA: Interface tunnel10 has came UP
2024/04/30 08:40:25 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:33:52 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:34:11 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:34:44 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:35:21 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:35:28 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:35:44 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:37:21 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:37:40 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:38:15 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:38:51 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:38:59 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:39:16 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 10:59:27 ZEBRA: Interface tunnel10 has came UP
2024/05/05 10:59:46 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:00:20 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:00:56 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:01:03 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:01:20 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:37:36 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:37:55 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:38:29 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:39:05 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:39:14 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:39:31 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:57:16 ZEBRA: Interface tunnel10 has came UP
2024/05/05 11:57:34 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 11:58:09 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:00:26 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:00:45 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 12:01:20 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:01:57 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 12:02:23 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:03:24 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:03:42 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 12:04:16 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:04:53 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/05 12:05:01 ZEBRA: Interface tunnel10 has came UP
2024/05/05 12:05:18 ZEBRA: Interface tunnel10 has gone DOWN
2024/05/07 15:06:19 ZEBRA: Interface tunnel10 has came UP
2024/05/12 10:12:25 ZEBRA: Interface green has came UP
2024/05/12 10:12:26 ZEBRA: Interface red has came UP
2024/05/12 10:12:28 ZEBRA: Interface blue has came UP
2024/05/12 10:12:32 ZEBRA: Interface ge3 has gone DOWN
2024/05/12 10:12:32 ZEBRA: Interface ge3 has came UP
2024/05/12 10:12:36 ZEBRA: Interface ge1 has gone DOWN
2024/05/12 10:12:37 ZEBRA: Interface ge1 has came UP
2024/05/12 10:12:37 ZEBRA: Interface ge1 has gone DOWN
2024/05/12 10:12:40 ZEBRA: Interface ge1 has came UP
2024/05/12 10:12:46 ZEBRA: Interface loopback100 has gone DOWN
2024/05/12 10:12:48 ZEBRA: Interface loopback100 has gone DOWN
2024/05/12 10:12:49 ZEBRA: Interface loopback100 has came UP
2024/06/11 14:34:00 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:34:19 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:36:00 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:36:18 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:36:50 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:37:28 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:37:55 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:38:11 ZEBRA: Interface tunnel10 has gone DOWN
n1#
```

اگر باز هم میزان خروجی لاگ ها بعد از فیلتر زیاد باشد می توانیم مشخص کنیم چند خط پایانی یا ابتدایی را برای ما نشان دهد :

```
n1# sh log all | grep -iE "interface.+ (up|down)" | tail -n 20
2024/05/12 10:12:25 ZEBRA: Interface green has came UP
2024/05/12 10:12:26 ZEBRA: Interface red has came UP
2024/05/12 10:12:28 ZEBRA: Interface blue has came UP
2024/05/12 10:12:32 ZEBRA: Interface ge3 has gone DOWN
2024/05/12 10:12:32 ZEBRA: Interface ge3 has came UP
2024/05/12 10:12:36 ZEBRA: Interface ge1 has gone DOWN
2024/05/12 10:12:37 ZEBRA: Interface ge1 has came UP
2024/05/12 10:12:37 ZEBRA: Interface ge1 has gone DOWN
2024/05/12 10:12:40 ZEBRA: Interface ge1 has came UP
2024/05/12 10:12:46 ZEBRA: Interface loopback100 has gone DOWN
2024/05/12 10:12:48 ZEBRA: Interface loopback100 has gone DOWN
2024/05/12 10:12:49 ZEBRA: Interface loopback100 has came UP
2024/06/11 14:34:00 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:34:19 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:36:00 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:36:18 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:36:50 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:37:28 ZEBRA: Interface tunnel10 has gone DOWN
2024/06/11 14:37:55 ZEBRA: Interface tunnel10 has came UP
2024/06/11 14:38:11 ZEBRA: Interface tunnel10 has gone DOWN
n1# sh log all | grep -iE "interface.+ (up|down)" | head -n 20
2024/04/29 10:17:20 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:17:39 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:18:07 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:18:44 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:19:10 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:19:27 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:33:25 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:33:44 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:34:17 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:34:54 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:35:01 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:35:18 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:37:10 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:37:29 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:38:02 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:41:47 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:42:05 ZEBRA: Interface tunnel10 has gone DOWN
2024/04/29 10:42:39 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:44:14 ZEBRA: Interface tunnel10 has came UP
2024/04/29 10:44:32 ZEBRA: Interface tunnel10 has gone DOWN
n1#
```

12.1.11 توابع

sooshell امکان تعریف تابع و فراخوانی آنها را جهت اسفاده مجدد از کدها به کاربر میدهد. توابع هنگام فراخوانی میتوانند آرگومان دریافت کنند و حتی یک کد وضعیت نیز بازگردانند. کد وضعیت تابع مانند کد وضعیت بقیه دستورات عمل میکند، کد صفر به معنای موفقیت و سایر کدها به معنای شکست میباشد. توابع همچنین میتوانند مانند دستورات پردازش خروجی عمل کنند، یعنی بعد از کاراکتر | فراخوانی شوند تا خروجی دستور قبلی را به عنوان ورودی بگیرند.

در اینجا یک تابع ساده تعریف میکنیم که اگر همه آرگومان های آن با هم برابر نبودند، شکست میخورد:

```
check-equality()
{
   if [[ $# -eq 0 ]]; then
      return 2
   fi
   for arg in $@; do
      if [[ $arg -neq $1 ]]; then
         return 1
      fi
   done
   return 0
}
```

برای تعریف تابع باید از سینتکس name() { ... } استفاده کنیم. تعریف تابع در بین دو آکولاد قرار میگیرد. در ابتدای تابع بررسی میکنیم آیا اصلا آرگومانی به تابع داده شده یا خیر. عبارت $# بیانگر تعداد آرگومانها میباشد. میخواهیم اگر آرگومانی داده نشد تابع شکست بخورد و مقدار 2 باز گردانده شود. در ادامه یک حلقه میگذاریم که تک تک آرگومان ها را با استفاده از عبارت $@ پیمایش میکند و آنها را با اولین آرگومان یعنی $1 مقایسه میکند (بطور کلی $n یعنی آرگومان n ام.) و اگر برابر نبودند مقدار 1 را به معنی خطا بازمیگرداند.

اکنون میخواهیم تابعی که نوشته ایم را فراخوانی کنیم. فراخوانی تابع به صورت check-equality arg1 arg2 arg3 ... میباشد. در مثال پایین در خط اول با عملگر && بررسی میکنیم که آیا اجرا موفقیت آمیز بوده یا خیر، چرا که اگر شکست خورده باشد دستور بعدی نباید اجرا شود. در خط بعدی نیز از شکست خوردن تابع اطمینان حاصل میکنیم، زیرا عملگر || با معنی "یا" باعث میشود تا اگر دستور اول موفقیت آمیز بود، sooshell از اجرای دستور بعدی صرف نظر کند.

```
check-equality a a a && echo '1st' case passed
check-equality a a b || echo '2nd' case failed
```

خروجی اجرا:

```
1st case passed
2nd case passed
```

تابعی برای دریافت **IP ,MAC** اینترفیسها

```
show_int(){
sh int json | jq -r  ' ["Interface          ", "IP Address       ", "MAC Address       "] ,
(to_entries[] | [  .key,  (.value.ipAddresses | if length > 0 then .[0].address else "N/A" end),  .value.hardwareAddress] | map(if length < 20 then . + " "
↪ * (20 - length) else . end)) | @tsv'
}
```

```
amnesh-router# show_int
Interface                IP Address              MAC Address
ge0                      192.168.111.4/24        00:ec:ac:ce:74:65
ge0.100                  10.10.100.1/24          00:ec:ac:ce:74:65
ge0.110                  10.10.110.1/24          00:ec:ac:ce:74:65
ge0.120                  10.10.120.1/24          00:ec:ac:ce:74:65
ge0.130                  10.10.130.1/24          00:ec:ac:ce:74:65
ge0.140                  10.10.140.1/24          00:ec:ac:ce:74:65
ge0.150                  10.10.150.1/24          00:ec:ac:ce:74:65
ge0.160                  10.10.160.1/24          00:ec:ac:ce:74:65
ge0.190                  10.10.190.1/24          00:ec:ac:ce:74:65
ge0.200                  10.10.200.3/24          00:ec:ac:ce:74:65
ge0.210                  10.10.210.1/24          00:ec:ac:ce:74:65
ge0.245                  10.10.245.1/24          00:ec:ac:ce:74:65
ge1                      192.168.1.4/24          00:ec:ac:ce:74:66
ge2                      192.168.30.4/24         00:ec:ac:ce:74:67
ge3                      192.168.10.4/24         00:ec:ac:ce:74:68
lo                       N/A
amnesh-router#
```

تابعی برای **up** کردن همه اینترفیس های **vlan**

```
vlanup()
{
    if [[ ! -in config ]]; then
        echo this command must be run in config node, not $(node) node 1>&2
        return 1
    fi

    for ifname in $(interfaces); do
        if [[ $ifname =~ .*\\..* ]]; then
            interface $ifname
            no shut
            quit
        fi
    done
}
# usage
valnup
```

تنظیم IP در اینترفیس ها

تابعی که جفتهای اینترفیس/آیپی را دریافت میکند و آیپی ها را روی اینترفیس تنظیم میکند.

```
addips()
{
    local retcode=0

    if [[ ! -in config ]]; then
        echo this command must be run in config node, not $(node) node 1>&2
        return 1
    fi

    for pair in "$@"
    do
        ifname=($(cut -d '=' -f '1' <<< $pair))
        ipaddr=($(cut -d '=' -f '2' <<< $pair))
        if do sh int json | jq -e -r ".$ifname" &> /dev/null
        then
            echo "On $ifname set IP $ipaddr"
            interface $ifname
            ip address $ipaddr
            no shut
            quit
        else
            echo "interface $ifname does not exist" 1>&2
            retcode=1
        fi
    done
    return $retcode
}
# usage
addips wg0="192.168.1.1/24" wg1="192.168.2.1/24" wg2="192.168.3.1/24"
```

تابعی برای ساختن تونل IPSec

تابعی که چهار ورودی از کاربر می گیرد و تمامی تنظیمات تونل IPSec را که بسیار زمانبر است انجام میدهد : شما لازم است , source_ip , remote_node_id , local_node_id , destination_ip . را به عنوان ورودی به تابع بدهید node_id ها می تواند عددی باشد که شما به هر روتر در سناریوی خود اختصاص می دهید . در ادامه با ذکر مثال نحوه تعریف و استفاده از این تابع را با یک کثال توضیح میدهیم .

```
add-ipsec()
{
if [ "$#" -ne 4 ]; then
    echo "need argumenst : $0 <source_ip> <destination_ip> <my_id> <peer_id>"
    return 1
fi

local_id="222.$3.$3.$3"
remote_id="222.$4.$4.$4"
tunnel_id=$(($3+$4))
tunnel_ip="10.0.$tunnel_id.$3/24"
local_ip=$1
```

(continues on next page)

(continued from previous page)

```
remote_ip=$2

echo local_ip=$local_ip
echo remote_ip=$remote_ip
echo local_id=$local_id
echo remote_id=$remote_id
echo tunnel_id=$tunnel_id
echo tunnel_ip=$tunnel_ip
psk="salam!-$tunnel_id!"

crypto ikev2 proposal IKE_PROPOSAL
 integrity sha-96
 encryption aes-256
 group 20
!
crypto ikev2 profile IKE_PROFILE_$3_$4
 keyring local IKE_KEYRING
 identity local address $local_id
 match identity remote address $remote_id
 authentication local pre-share
 authentication remote pre-share
 proposal IKE_PROPOSAL
!
crypto ikev2 keyring IKE_KEYRING
 peer $remote_ip
  address $remote_ip
  pre-shared-key $psk
  identity address $remote_id
!
crypto ipsec transform-set IPSec_TS esp hmac sha-256 cipher aes-256
 mode transport
!
crypto ipsec profile IPSec_PROFILE_$3_$4
 set transform-set IPSec_TS
 set ikev2-profile IKE_PROFILE_$3_$4
 set security-association lifetime seconds 3600
!
!
interface tunnel$tunnel_id
 tunnel source $local_ip
 tunnel destination $remote_ip
 tunnel protection ipsec profile IPSec_PROFILE_$3_$4
 no shutdown
 ip address $tunnel_ip
!

}
```

ابتدا تابع را در روتر 1 به شکل زیر تعریف می کنیم و در واقع تابع را کپی کرده و در ترمینال paste می کنیم :

```
n1/config# add-ipsec()
function> {
function> if [ "$#" -ne 4 ]; then
function then>     echo "need argumenst : $0 <source_ip> <destination_ip> <my_id> <peer_id>"
function then>     return 1
function then> fi
function>
```

(continues on next page)

(continued from previous page)

```
function> local_id="222.$3.$3.$3"
function> remote_id="222.$4.$4.$4"
function> tunnel_id=$(($3+$4))
function> tunnel_ip="10.0.$tunnel_id.$3/24"
function> local_ip=$1
function> remote_ip=$2
function>
function> echo local_ip=$local_ip
function> echo remote_ip=$remote_ip
function> echo local_id=$local_id
function> echo remote_id=$remote_id
function> echo tunnel_id=$tunnel_id
function> echo tunnel_ip=$tunnel_ip
function> psk="salam!-$tunnel_id!"
function>
function> crypto ikev2 proposal IKE_PROPOSAL
function>  integrity sha-96
function>  encryption aes-256
function>  group 20
function> !
function> crypto ikev2 profile IKE_PROFILE_$3_$4
function>  keyring local IKE_KEYRING
function>  identity local address $local_id
function>  match identity remote address $remote_id
function>  authentication local pre-share
function>  authentication remote pre-share
function>  proposal IKE_PROPOSAL
function> !
function> crypto ikev2 keyring IKE_KEYRING
function>  peer $remote_ip
function>   address $remote_ip
function>   pre-shared-key $psk
function>   identity address $remote_id
function> !
function> crypto ipsec transform-set IPSec_TS esp hmac sha-256 cipher aes-256
function>  mode transport
function> !
function> crypto ipsec profile IPSec_PROFILE_$3_$4
function>  set transform-set IPSec_TS
function>  set ikev2-profile IKE_PROFILE_$3_$4
function>  set security-association lifetime seconds 3600
function> !
function> !
function> interface tunnel$tunnel_id
function>  tunnel source $local_ip
function>  tunnel destination $remote_ip
function>  tunnel protection ipsec profile IPSec_PROFILE_$3_$4
function>  no shutdown
function>  ip address $tunnel_ip
function> !
function>
function> }
n1/config#
n1/config#
```

و سپس به شکل زیر تابع را با ورودی های مورد نظر فراخوانی می کنیم .

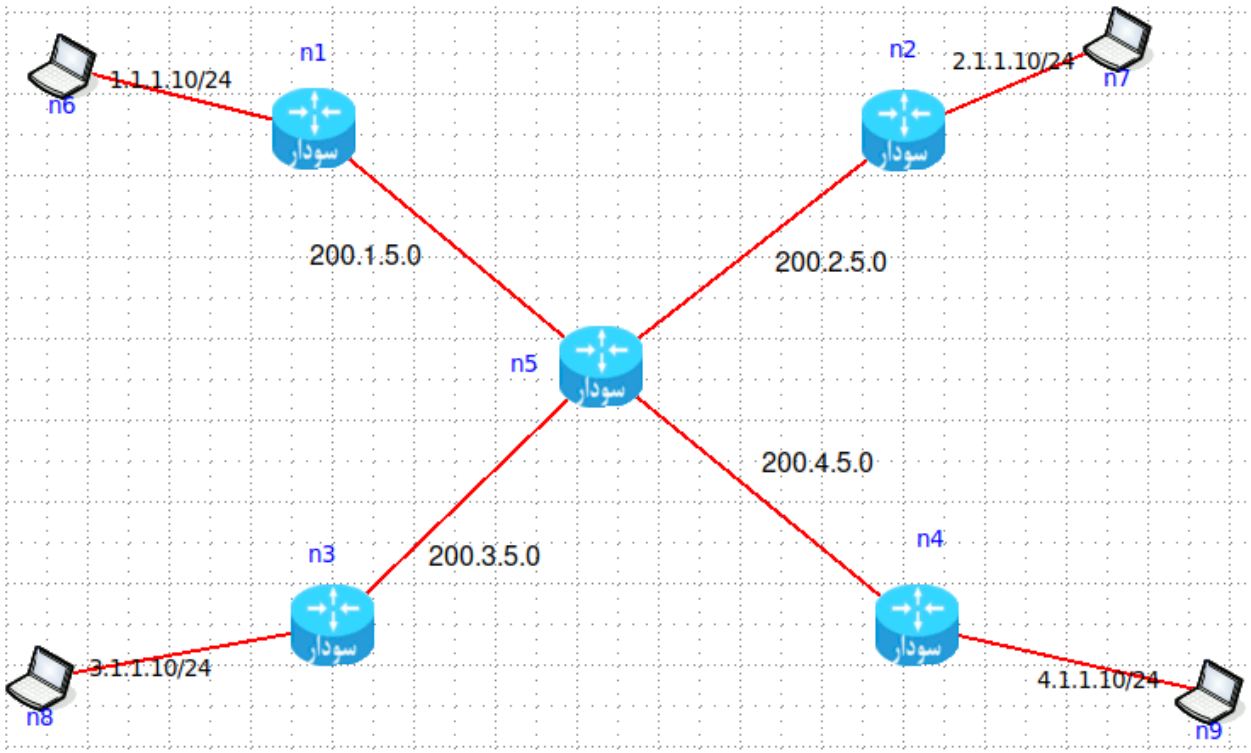
source_ip: 200.1.5.1

destination_ip: 200.2.5.2
local_id : 1
remote_id : 2
بدین شکل تونل IPSec از روتر 1 به روتر 2 تنظیم می شود

```
n1/config#
n1/config# add-ipsec 200.1.5.1 200.2.5.2  1  2
local_ip=200.1.5.1
remote_ip=200.2.5.2
local_id=222.1.1.1
remote_id=222.2.2.2
tunnel_id=3
tunnel_ip=10.0.3.1/24
n1/c/interface/tunnel3#
```

تابعی برای ساخت تونل **wireguard**

حال فرض کنید یک سناریو به شکل زیر داریم و قصد داریم بین نود های 1 و 2 و 3 و4 تونل های ipsec به صورت fullmesh ایجاد کنیم :



تونل به روتر 2 که تنظیم شد ،می توان به راحتی هر چه تمام تر تونل به روتر های 3 و 4 را هم ساخت :

```
n1/config# add-ipsec 200.1.5.1 200.3.5.3  1  3
```

```
n1/config# add-ipsec 200.1.5.1 200.4.5.4  1  4
```

برای تنظیم تونل در بقیه روتر ها هم کافی است این تابع را در آن ها تعریف کرده و از آن استفاده کنیم :

```
n2/config# add-ipsec 200.2.5.2 200.1.5.1  2  1
n2/config# add-ipsec 200.2.5.2 200.3.5.3  2  3
n2/config# add-ipsec 200.2.5.2 200.4.5.4  2  4
```

```
n3/config# add-ipsec 200.3.5.3 200.1.5.1  3  1
n3/config# add-ipsec 200.3.5.3 200.2.5.2  3  2
n3/config# add-ipsec 200.3.5.3 200.4.5.4  3  4
```

```
n4/config# add-ipsec 200.4.5.4 200.1.5.1  4  1
n4/config# add-ipsec 200.4.5.4 200.3.5.3  4  3
n4/config# add-ipsec 200.4.5.4 200.2.5.2  4  2
```

بدین ترتیب با استفاده از این تابع توانستیم در هر نود 3 تونل IPSec و در مجموع 12 تونل را فقط با 12 خط دستور بسازیم .

برای نمونه تنظیمات به شکل زیر در n1 انجام شده است :

```
hostname n1
service password-encryption
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
no ntp
!
ip route 0.0.0.0/0 200.1.5.5
!
crypto ikev2 proposal IKE_PROPOSAL
 integrity sha-96
 encryption aes-256
 group 20
exit
!
crypto ikev2 profile IKE_PROFILE_1_2
 keyring local IKE_KEYRING
 proposal IKE_PROPOSAL
 match identity remote address 222.2.2.2
 identity local address 222.1.1.1
 authentication local pre-share
 authentication remote pre-share
exit
!
crypto ikev2 profile IKE_PROFILE_1_3
 keyring local IKE_KEYRING
 proposal IKE_PROPOSAL
 match identity remote address 222.3.3.3
 identity local address 222.1.1.1
 authentication local pre-share
 authentication remote pre-share
exit
!
crypto ikev2 profile IKE_PROFILE_1_4
 keyring local IKE_KEYRING
 proposal IKE_PROPOSAL
 match identity remote address 222.4.4.4
 identity local address 222.1.1.1
 authentication local pre-share
```

(continues on next page)

(continued from previous page)

```
 authentication remote pre-share
exit
!
crypto ikev2 keyring IKE_KEYRING
 peer 200.2.5.2
  address 200.2.5.2
  pre-shared-key salam!-3!
  identity address 222.2.2.2
 peer 200.3.5.3
  address 200.3.5.3
  pre-shared-key salam!-4!
  identity address 222.3.3.3
 peer 200.4.5.4
  address 200.4.5.4
  pre-shared-key salam!-5!
  identity address 222.4.4.4
exit
!
crypto ipsec transform-set IPSec_TS esp hmac sha-256 cipher aes-256
 mode transport
exit
!
crypto ipsec profile IPSec_PROFILE_1_2
 set transform-set IPSec_TS
 set ikev2-profile IKE_PROFILE_1_2
 set security-association lifetime seconds 3600
exit
!
crypto ipsec profile IPSec_PROFILE_1_3
 set transform-set IPSec_TS
 set ikev2-profile IKE_PROFILE_1_3
 set security-association lifetime seconds 3600
exit
!
crypto ipsec profile IPSec_PROFILE_1_4
 set transform-set IPSec_TS
 set ikev2-profile IKE_PROFILE_1_4
 set security-association lifetime seconds 3600
exit
!

interface tunnel3
 no shutdown
 tunnel source 200.1.5.1
 tunnel destination 200.2.5.2
 tunnel protection ipsec profile IPSec_PROFILE_1_2
 ip address 10.0.3.1/24
exit
!
interface tunnel4
 no shutdown
 tunnel source 200.1.5.1
 tunnel destination 200.3.5.3
 tunnel protection ipsec profile IPSec_PROFILE_1_3
 ip address 10.0.4.1/24
exit
!
```

(continues on next page)

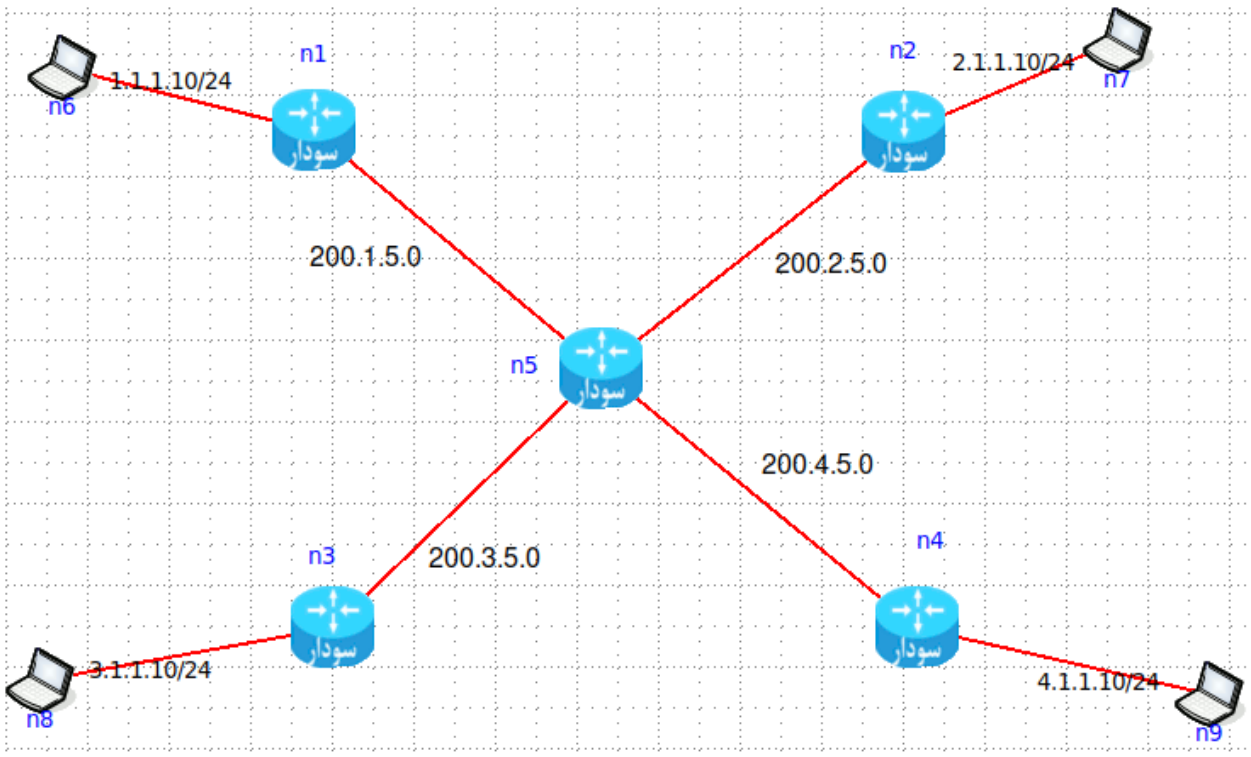
(continued from previous page)

```
interface tunnel5
 no shutdown
 tunnel source 200.1.5.1
 tunnel destination 200.4.5.4
 tunnel protection ipsec profile IPSec_PROFILE_1_4
 ip address 10.0.5.1/24
exit
```

تابعی برای ساختن تونل wireguard

```
add-wg()
{
if [ "$#" -ne 4 ]; then
    echo "need argumenst : $0 <destination_ip> <my_id> <peer_id> <peer-pub-key>"
    return 1
fi

echo $1 $2 $3 $4



tunnel_id=$(($2+$3))
tunnel_ip="10.10.$tunnel_id.$3/32"
port=$((1000+$tunnel_id))

echo tunnel_id=$tunnel_id
echo tunnel_ip=$tunnel_ip

prvkey="wg$2-$3"

crypto key gen x25519 label $prvkey
my_pubkey=$(do sh crypto key $prvkey json | jq -r 'to_entries [] | .value[0].public_key')

interface wireguard$tunnel_id
 wireguard source 0.0.0.0
 wireguard private-key $prvkey
 wireguard port $port
 wireguard peer $1
  endpoint $1 port $port
  allowed-ip 0.0.0.0/0
  public-key $4
 no shutdown
 ip address $tunnel_ip
 wireguard peer $1
!
echo my public_key:  $my_pubkey
}
```

اگر مجدد سناریوی که تونل ipsec ایجاد کردیم را در نظر بگیریم می توانیم با تابع فوق تونل های wireguard بصورت fullmesh بین روتر ها ایجاد نماییم . برای این کار کافی است در هر روتر ابتدا یک کلید wireguard بسازیم تا pub-key آن را به عنوان ورودی به تابع به همراه دیگر آرگومان ها اختصاص دهیم :




```
n1/config# crypto key generate x25519 label wg1
n1/config# do sh crypto key wg1
Keypair Label: wg1
  Algorithm:  X25519
  Public key: 9213E1463CBEA9A28A9E91A2659E32C1EDFDEDC2AF956D79D22F4482A6FBD769
  Public key base64: khPhRjy+qaKKnpGiZZ4ywe397cKvlW150i9Egqb712k=
------------------------------------
n2/config# crypto key generate x25519 label wg2
n2/config# do sh crypto key wg2
Keypair Label: wg2
  Algorithm:  X25519
  Public key: 0980BC9E8F7E8E45FE2DC4F6B845B161F8B4BB3F21EFB01B5E3502867C669F10
  Public key base64: CYC8no9+jkX+LcT2uEWxYfi0uz8h77AbXjUChnxmnxA=
------------------------------------
n3/config# crypto key generate x25519 label wg3
n3/config# do sh crypto key wg3
Keypair Label: wg3
  Algorithm:  X25519
  Public key: CAA953CF7B569365DC7F6728308E59B2743FF4C2C97435C6DA7CA8D7FC9F5036
  Public key base64: yqlTz3tWk2Xcf2coMI5ZsnQ/9MLJdDXG2nyo1/yfUDY=
------------------------------------
n4/config# crypto key generate x25519 label wg4
n4/config# do sh crypto key wg4
Keypair Label: wg4
  Algorithm:  X25519
  Public key: 256AD17ED4ACB5F879A649F4ACA2AD07EE5E7B37982169F26536FB6655E3B257
  Public key base64: JWrRftSstfh5pkn0rKKtB+5eezeYIWnyZTb7ZlXjslc=

n4/config#
```

```
n1/config# add-wg 200.2.5.2 1 2 0980BC9E8F7E8E45FE2DC4F6B845B161F8B4BB3F21EFB01B5E3502867C669F10
n1/config# add-wg 200.4.5.4 1 4 256AD17ED4ACB5F879A649F4ACA2AD07EE5E7B37982169F26536FB6655E3B257
n1/config# add-wg 200.3.5.3 1 3 CAA953CF7B569365DC7F6728308E59B2743FF4C2C97435C6DA7CA8D7FC9F5036
```

```
n2/config# add-wg 200.1.5.1 2 1 9213E1463CBEA9A28A9E91A2659E32C1EDFDEDC2AF956D79D22F4482A6FBD769
n2/config# add-wg 200.4.5.4 2 4 256AD17ED4ACB5F879A649F4ACA2AD07EE5E7B37982169F26536FB6655E3B257
n2/config# add-wg 200.3.5.3 2 3 CAA953CF7B569365DC7F6728308E59B2743FF4C2C97435C6DA7CA8D7FC9F5036
```

```
n3/config# add-wg 200.1.5.1 3 1 9213E1463CBEA9A28A9E91A2659E32C1EDFDEDC2AF956D79D22F4482A6FBD769
n3/config# add-wg 200.4.5.4 3 4 256AD17ED4ACB5F879A649F4ACA2AD07EE5E7B37982169F26536FB6655E3B257
n3/config# add-wg 200.2.5.2 3 2 0980BC9E8F7E8E45FE2DC4F6B845B161F8B4BB3F21EFB01B5E3502867C669F10
```

```
n4/config# add-wg 200.1.5.1 4 1 9213E1463CBEA9A28A9E91A2659E32C1EDFDEDC2AF956D79D22F4482A6FBD769
n4/config# add-wg 200.3.5.3 4 3 CAA953CF7B569365DC7F6728308E59B2743FF4C2C97435C6DA7CA8D7FC9F5036
n4/config# add-wg 200.2.5.2 4 2 0980BC9E8F7E8E45FE2DC4F6B845B161F8B4BB3F21EFB01B5E3502867C669F10
```

بدین شکل 12 تونل wireguard بین روتر ها تنظیم می شود .

فصل 12

تنظیمات کاربری

در حال حاضر فقط یک کاربر به نام admin بر روی روتر های سودار فعال می باشد . این همان نامی است که برای ارتباط ssh استفاده می شود

1.12 تغییر پسورد کاربر admin

برای تغییر پسورد کاربر admin از دستور زیر استفاده می‌کنیم :

```
soodar(config)# password
Changing password for admin.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
```

2.12 تغییر پسورد enable

برای تغییر پسورد Enable از دستور زیر استفاده می‌کنیم :

```
soodar(config)# enable password
Password:
soodar(config)#
```

3.12 تغییر پسورد enable config

برای تغییر پسورد enable config از دستور زیر استفاده می‌کنیم :

```
soodar(config)# enable config password
Password:
soodar(config)#
```

4.12 ذخیره پسورد به صورت رمز شده

کافی است دستور زیر را وارد نمایید تا پسورد ها در تنظیمات به صورت رمز شده ذخیره و نمایش داده شود :

```
soodar(config)# service password-encryption
```

نکته

اگر این دستور را no کنید و بخواهید رمز شدن پسورد ها را غیر فعال کنید تمامی پسورد هایی که تنظیم کرده اید حذف می شود و باید مجدد آن ها را تنظیم کنید .

5.12 تعیین حداقل طول پسورد

به صورت پیش فرض رمز های enable , config باید حداقل 8 کارکتر باشد . می توانید طول پسورد را به عددی بزرگتر تغییر دهید :

```
soodar(config)# security passwords min-length 12
```

و همچنین می توان این تنظیم را با دستور زیر غیر فعال کرد :

```
soodar(config)# no security passwords min-length 8
```

6.12 تغییر ip پورت SSH

به صورت پیش فرض پورت ge0 (اولین پورت) با آدرس 192.168.1.55 و نت مسک 24 برای ssh در سودار تنظیم شده است و شما در صورت نیاز می توانید آن را تغییر دهید .

7.12 بازیابی رمز کاربر (رمز **ssh**)

اگر رمز عبور روتر را فراموش کردید برای تنظیم مجدد آن باید از ایمیج ریکاوری سودار استفاده کنید . این ایمیج باید روی یک فلش به صورت bootable نوشته شود و پس از آن فلش را به روتر متصل کنید و از روی آن بوت شوید . بعد از اینکه روتر از روی فلش بوت شد شما گزینه هایی برای ریست کردن می بینید و می توانید رمز عبور و همچنین تنظیمات روتر را ری ست کنید .

نکته

✓ دقت شود با ری ست کردن رمز عبور ، رمز عبور شما به admin تغییر می کند .
✓ ریست کردن تنظیمات فقط تنظیمات روتری را به حالت پیش فرض بر می گرداند و تنظیمات سیستمی مانند رمز عبور کاربر و ساعت سیستم و کلیدها و ... تغییر نمی کند .
✓ با ریست کردن تنظیمات در اینترفیس ge0 شما (اولین اینترفیس روتر) ip به آدرس 55/24.1.168.192 تنظیم می شود .

فصل 13

پشتیبان گیری و بازنشانی تنظیمات

فایل ها و تنظیماتی که از آن ها می توانید پشتیبان گیری نمایید یا آنها را بازنشانی کنید :

- running-config : می توانید از running-config بکاپ بگیرید و بعدا آن را در startup-config بازنشانی کنید
- startup-config : می توانید از startup-config بکاپ بگیرید یا آن را باز نشانی کنید

پشتیبان گیری و بازنشانی را می توانید در روتر (system) و یا در دستگاهی دیگر با sftp انجام دهید

- sftp : پشیبان گیری و بازنشانی در دستگاه دیگر
- system : پشیبان گیری و بازنشانی در خود روتر
- no-pki : به صورت پیش فرض در زمان backup گرفتن از تمامی کلید های pki و wireguard هم backup گرفته میشود تا در زمان restore کردن کانفیگ کامل باشد و نیاز به تغییر در شبکه نباشد . اما admin می تواند با no-pki در آخر دستور مشخص کند که از کلید ها backup گرفته نشود و فقط config را backup بگیرد
  توجه کنید که pki شامل تمامی کلید های certificates , wireguard,ssh , rsa، ... می شود .
- Repository path : مسیری که فایل های backup باید در سرور ذخیره گردد . اگر این مسیر مقدار دهی نشود در home ذخیره می گردد
- Repository name : اسم فولدری که باید فایل های backup در آن قرار گیرد
- Repository password : رمز عبور مربوط به repository که شما قصد دارید در آن backup بگیرید و فایل ها در آن ذخیره گردد

با استفاده از repository کاربر یا کاربران مختلف می تواند repo های مختلف داشته باشند که از یکدیگر جدا بوده و هر دسترسی آن با پسورد خواهد بود . لازم به ذکر است اولین باری که یک repo ساخته میشود رمز آن توسط کاربر تعیین می گردد

توجه

پشتیبان گیری از کلید ها به لحاظ امنیتی درست نیست چون private-key شما هم انتقال پیدا می کند و به صورت کلی این عملیات روی pki توصیه نمی شود . اما در صورت نیاز admin می تواند با عنایت به حساسیت مسئله از این امکان استفاده نماید

در ادامه مثال هایی را برای نمونه در حالت های مختلف بیان کم کنیم :

1.13 پشتیبان گیری از startup-config

1.1.13 loacl

با این دستور تنظیمات startup-config در روتر با تگ bkp1 ذخیره می شود

```
router# copy startup-config system:
  <cr>
  no-pki  Do not backup keys/certificates
router# copy startup-config system:
Repository name [router-backup]? repo1
Repository password []?
Destination tag [router-config]? bkp1
Backup successful!
router#
```

2.1.13 sftp

با این دستور تنظیمات startup-config با ssh در 192.168.230.130 در home کاربر iman ذخیره خواهد شد .

توجه

قبل از اجرای این دستور باید قبلا یک بار به ssh 192.168.230.130 زده باشید و کلید آن در روتر ذخیره شده باشد و گر نه دستور زیر با خطا (Connection reset by peer) مواجه خواهد شد .

```
soodar# copy startup-config sftp:iman@192.168.230.130
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?
Destination tag [router-config]? bkp2
Backup successful!
soodar#
```

مثال backup گیری فقط از config و بدون pki :

```
soodar# copy startup-config sftp:iman@192.168.230.130 no-pki
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?
```

(continues on next page)

(continued from previous page)

```
Destination tag [router-config]? bkp3
Backup successful!
soodar#
```

2.13 پشتیبان گیری از **running-config**

برای running-config هم مثل startup-config عمل می کنیم با این تفاوت که در اینجا running-config پشتیبان گیری می کنیم و اما عملیات بازنشانی برای running-config وجود ندارد . به عبارت دیگر عمل restore فقط روی startup-config انجام می شود و نمی توان backup را روی running-config نشاند .

3.13 مشاهده لیست **backup** ها

```
soodar# sh archive snapshots system:

    Tag         Host            Time            Type
=========================================================

wireguard-keys   soodar   Sat Jun  3 10:05:44 2023   PKI
---------------------------------------------------------
bkp1             soodar   Sat Jun  3 10:12:02 2023   Config
soodar#
```

```
soodar# sh archive snapshots sftp:iman@192.168.230.130
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?


      Tag           Host              Time            Type
=================================================================
-----------------------------------------------------------------
n3-config          n3          Thu May 11 13:33:40 2023   Config
-----------------------------------------------------------------
bkp2               soodar      Sat Jun  3 09:55:19 2023   Config
-----------------------------------------------------------------
soodar#
```

4.13 باز نشانی startup-config

local 1.4.13

```
soodar# copy system:bkp1 startup-config
Repository name [router-backup]? repo1
Repository password []?
Tag to restore [bkp1]? bkp1
Restore successful! Restart your device to load new startup config
soodar#
```

sftp 2.4.13

```
soodar# copy sftp:iman@192.168.230.130 startup-config
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?
Tag to restore [router-config]? bkp2
Restore successful! Restart your device to load new startup config
soodar#
```

توجه

دقت شود در زمان backup گرفتن از license دستگاه هم backup گرفته می شود . در زمان restore کردن با no-license مشخص کنید که license به دستگاه کپی نشود . این مورد وقتی کاربرد دارد که شما backup گرفته شده از یه دستگاه دیگر را بخواهید در دستگاه دیگری restore کنید چون license دستگاه هم در backup وجود دارد اگر به دستگاه جدید کپی شود لایسنس مربوط به این روتر نیست و روتر با لایسنس پیش فرض بالا می آید .

مثال restore کردن backup بدون license :

```
soodar# copy sftp:iman@192.168.230.130 startup-config no-license
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?
Tag to restore [router-config]? bkp2
Restore successful! Restart your device to load new startup config
soodar#
```

5.13 مشاهده **config** هایی که بکاپ گرفته اید

می توانید config هایی که بک آپ گرفته اید را قبل از restore کردن مشاهده کنید :

```
soodar# sh archive config system:bkp1
Repository name [router-backup]? repo1
Repository password []?
Destination tag [bkp1]?

bkp1
====
hostname soodar

interface ge0
    ip address 192.168.1.55/24



soodar#
```

```
soodar# sh archive config sftp:iman@192.168.230.130
Address or name of remote host [192.168.230.130]?
Remote host user [iman]?
Remote host password [admin]?
Repository path []? backup
Repository name [router-backup]? site1
Repository password []?
Destination tag [router-config]? bkp2

bkp2
====
hostname soodar

interface ge0
    ip address 192.168.1.55/24



soodar#
```

write erase 6.13

برای reset کردن تنظیمات از دستور زیر استفاده می شود :

```
soodar1# write erase
```

توجه

دقت شود پس از write erase کلیه تنظیمات روتری حذف می شود اما تنظیماتی چون رمز عبور کاربر و ساعت سیستم و کلید های , ssh , pki wireguard و پروفایل های Tune باقی می ماند و اگر ip پیش فرض ge0 را مشخص نکنید اینترفیس ge0 دارای آدرس 192.168.1.55/24 خواهد شد .

7.13 تنظیم **ip** پیش فرض در **write erase**

از آنجا کلیه تنظیمات پس از این دستور به حالت پیش فرض بر می گردد و ip اینترفیس ge0 به 192.168.1.55/24 تغییر می کند ، در صورت نیاز شما می توانید تعیین کنید که چه ip در اینترفیس ge0 تنظیم شود و gateway روتر چه ip ی باشد . با این حالت شما می توانید روتر را از راه دور ری ست کنید و ارتباط شبکه ای شما هم قطع نشود و نیازی به دسترسی فیزیکی به روتر نداشته باشید :

```
soodar1# write erase  85.15.233.19/24  85.15.233.1
```

که در آن 85.15.233.19/24 مقدار ip اینترفیس ge0 و 85.15.233.1 مقدار gateway روتر خواهد بود . برای اعمال شدن تغییرات کانفیگ باید روتر reload شود :

```
soodar1# reload
```

فصل 14

بروزرسانی

1.14 بروز رسانی نرم افزار روتر

سودار از mender به عنوان سیستم بروز رسانی استفاده می کند . بروز رسانی به دو صورت online و offline انجام می شود و در صورت بروز مشکل در بروز رسانی به نسخه قبلی soodaros که از آن استفاده می کردید بر می گردید .

2.14 بروز رسانی آنلاین

بروز رسانی آنلاین از سروری که بدین منظور فراهم می شود صورت می پذیرد و به صورت پیش فرض غیر فعال است . زمانی که بروز رسانی فعال شود روتر به طور خودکار چک می کند که در سرور بروز رسانی وجود دارد و اگر بروز رسانی در سرور وجود داشته باشد عملیات بروز رسانی در روتر به طور خودکار انجام می شود .

1.2.14 تنظیم **dns server**

برای دسترسی به update server از طریق domain name باید ابتدا آدرس dns server را در روتر به شکل زیر تنظیم کنید :

```
soodar1(config)# ip name-server 8.8.8.8
```

2.2.14 فعال کردن بروز رسانی

الف) تنظیم url سرور بروز رسانی

```
soodar1(config)# system update  server-url  https://update.soodar.ir
```

ب) برای تنظیم بازه زمانی که روتر باید چک کند که آیا بروز رسانی جدید داریم :

```
soodar1(config)# system update update-poll-interval  60
```

با دستور فوق هر 60 ثانیه چک می شود که اگر بروز رسانی جدیدی داریم روتر خودش را بروز کند

ج) با دستور زیر هر 120 ثانیه اطلاعات اینترفیس ها و اطلاعات سیستمی روتر نظیر mac و برخی اطلاعات سخت افزاری برای سرور بروز رسانی ارسال می شود :

```
soodar1(config)#system update inventory-poll-interval 120
```

د) با دستور زیر بروز رسانی آنلاین فعال می شود .

```
soodar1(config)# system update enable
```

برای غیر فعال کردن بروز رسانی هم باید دستور زیر را وارد کنید :

```
soodar1(config)# no system update enable
```

3.2.14 مشاهده تنظیمات بروز رسانی

به دستور زیر می توانید تنظیمات فعلی بروز رسانی را مشاهده نمائید :

```
soodar1(config)# do sh system update
Online update details:
======================
  ❄ Update status: Enabled
    ❄ Server URL: https://sooman.net.ir
    ❄ Poll inventory updates each 120 seconds
    ❄ Poll updates each 60 seconds
```

نکته

دقت داشته باشید پس از انجام تغییر در بروزرسانی آنلاین توسط دستورات بالا حتما تنظیمات را write کنید .

3.14 بروز رسانی آفلاین

باید از یک حافظه خارجی متصل به سیستم (usb flash) برای بروز رسانی استفاده نمایید مراحل کار به شکل زیر است :

1. فلش را به روتر متصل کرده و دستور زیر را وارد نمایید و لیست فایل های بروز رسانی که در مسیر root فلش قرار دارد را مشاهده کنید :

```
soodar1(config)# system update offline list
```

مثال:

```
n1(config)# system update offline list
1 rls-20
2 rls-21
3 rls-21.1
```

2. سپس دستور زیر را برای بروز رسانی وارد نمایید :

```
n1(config)# system update offline install ARTIFACT
```

که پس از آن بروز رسانی از روی فلش آغاز خواهد شد . ARTIFACT آدرس فایل بروز رسانی بدون پسوند (.mender) است که در فلش قرار دارد .

3. پس از پایان یافتن مرحله قبل ، ابتدا فلش را از سیستم جدا کرده و سپس روتر را ری بوت نمایید .

4. پس از بالا آمدن روتر اگر مشکلی در بروز رسانی رخ نداده بود و تمایل داشتید این نسخه در روتر اعمال شود می توانید با دستور زیر بروز رسانی را تایید نمایید در غیر این صورت روتر را ری بوت نمایید (تا به نسخه قبلی روتر برگردید) :

```
soodar1(config)# system update offline commit
```

نکته

در زمان بروز رسانی آفلاین به نکات زیر دقت کنید :

1. بروز رسانی آنلاین را غیر فعال کنید (no system update enable)
2. در زمان ری بوت شدن نود هیچ حافظه خارجی و فلشی به روتر متصل نباشد

فصل 15

مانیتورینگ

Monitoring 1.15

snmp 1.1.15

روتر سودار از پروتکل snmp پشتیبانی می کند . شما می توانید با تنظیم snmp-server روتر سودار را مانیتور کرده و پارامتر های مربوط به اینترفیس ها و دیگر بخش های روتر را مانیتور نمایید

```
soodar1(config)# snmp-server user USER auth <md5|sha> PASSWORD [priv des56 PRIV]
```

logs , metrics 2.1.15

با دستور زیر می توانید لاگ های روتر را به سرور مانیتورینگ ارسال نمایید و همچنین کلیه متریک های سیستمی از قبیل cpu , ram , disk , network traffic ، ... و همچنین متریک های مربوط به dataplane سودار در دسترس قرار دارد و می توانید آن ها را در سرور مانیتورینگ تان دریافت نموده و در محیطی مانند grafana که از data source prometheus پشتیبانی می کند به صورت نمودار های گرافیکی در دشبورد نمایش دهید و تحلیل بسیار خوبی از وضعیت روتر داشته باشید .

```
soodar1(config)# log syslog <server address> vector port 9000
```

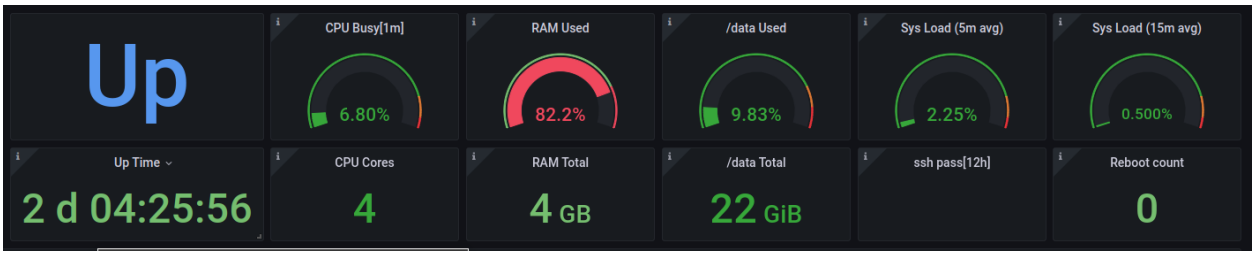
توجه

دقت شود metrics ها توسط prometheus از روتر دریافت می شود و با log ها تجمیع شده و توسط vector به سرور مانیتورینگ ارسال می گردد .

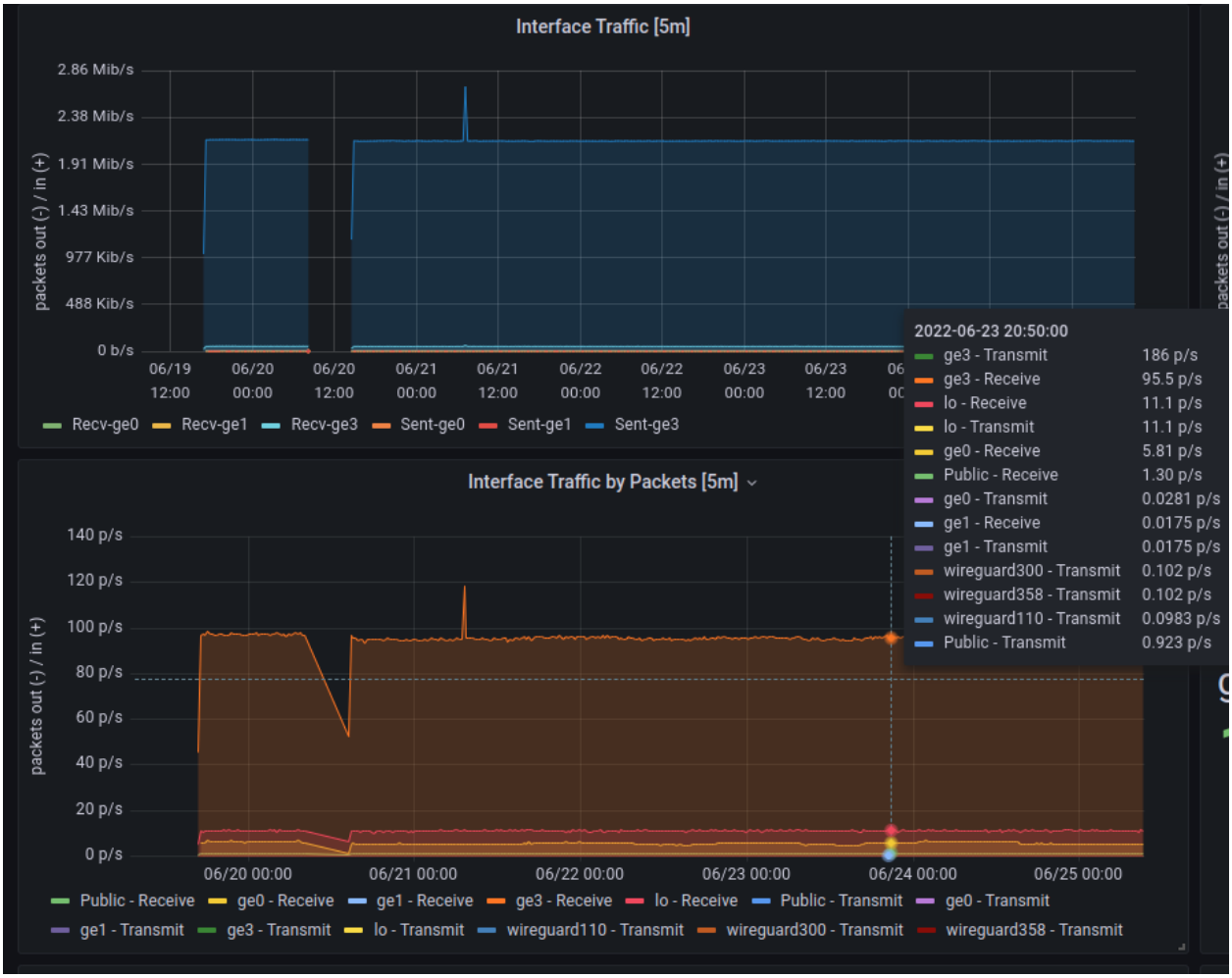
اگر بخواهید از rsyslog برای دریافت لا گها استفاده کنید دستور به شکل زیر خواهد بود :

```
soodar1(config)# log syslog <server address> tcp|udp port PORT
```

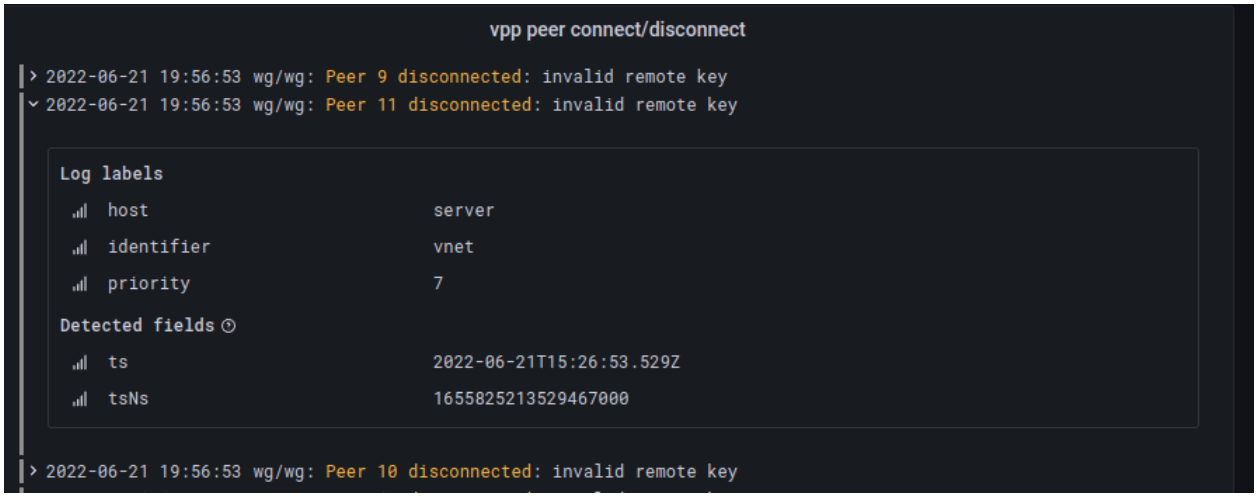
در ادامه چند پنل از مانیتورینگ را مشاهده می کنید :



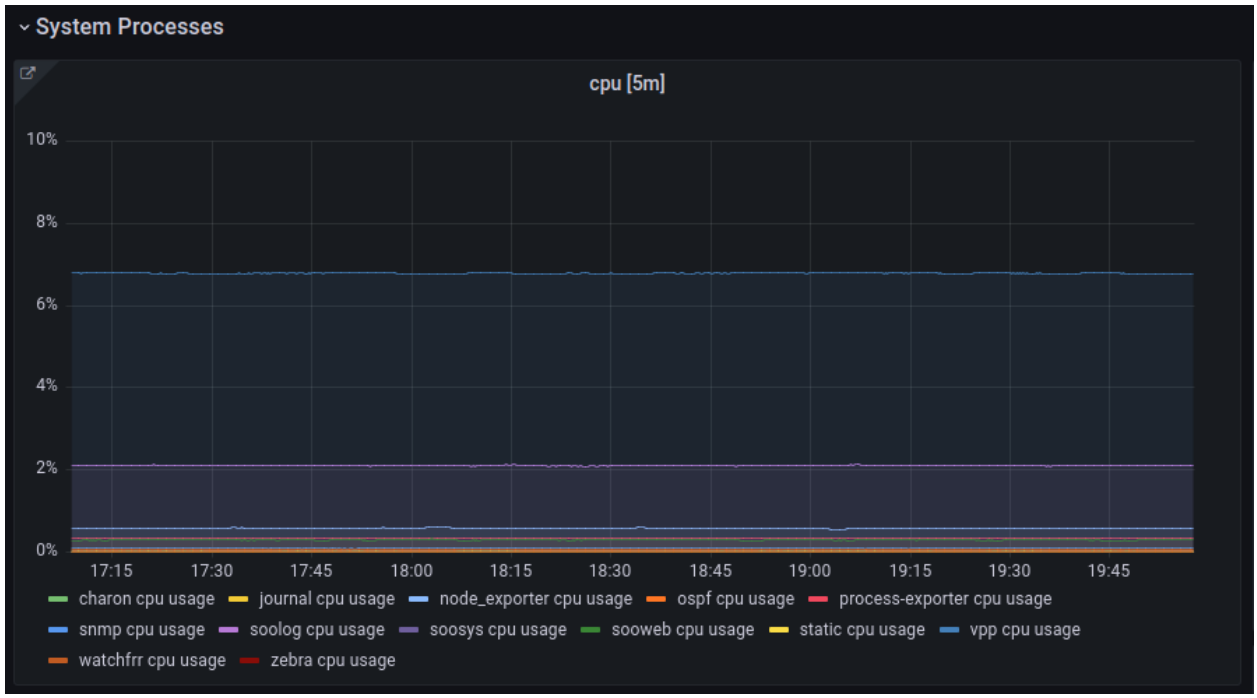
اطلاعات منابع سیستم



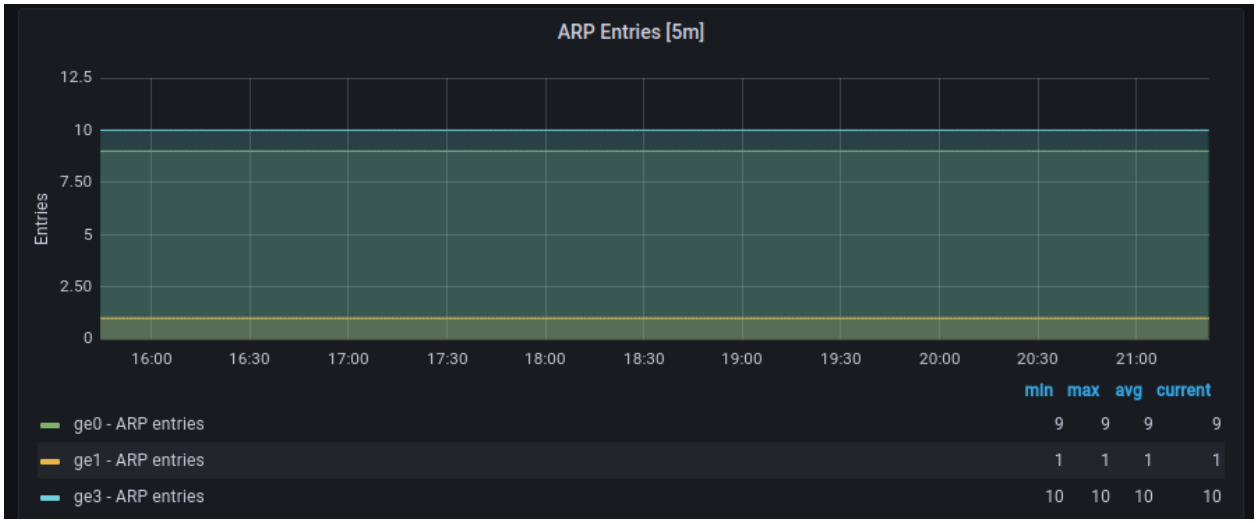
دار ترافیک اینترفیس ها



لع و وصلی تونل وایرگارد



ی پروسس های مهم روتر



دول ARP اینترفیس ها

نمودار io مربوط به disk

IPFIX 2.15

پروتکل IPFIX(Internet Protocol Flow Information Export) یک پروتکل IETF و نام گروه کاری IETF است که پروتکل را تعریف می‌کند. این بر اساس نیاز به یک استاندارد مشترک و جهانی تبدیل شده که اطلاعات فلوهای IP از روترها، پروب‌ها و دیگر دستگاه‌هایی که توسط سیستم‌های واسط، سیستم‌های حسابداری/صورتحسابی و سیستم‌های مدیریت شبکه را برای تسهیل خدماتی مانند اندازه‌گیری، حسابداری و ارائه می دهد. استاندارد IPFIX نحوه فرمت‌بندی و انتقال اطلاعات جریان IP را از یک فرستنده به یک گیرنده تعریف می‌کند.

IPFIX flow exporter 1.2.15

یک صادرکننده جریان IPFIX یک مؤلفه است که اطلاعات جریان شبکه را از دستگاه به یک مجموعه دهنده یا تجزیه و تحلیل کننده برای تحلیل، تصویرسازی و گزارش دهی بیشتر جمع‌آوری و ارسال می‌کند.

اطلاعات جریان شامل اطلاعاتی درباره ترافیک شبکه نظیر آدرس‌های IP مبدأ و مقصد، پورت‌ها، انواع پروتکل، تعداد بسته و بایت، timestamp و داده‌های فراداده مربوط دیگر می‌شود. صادرکنندگان جریان IPFIX می‌توانند با استفاده از گزینه‌های فیلترکردن و نمونه‌برداری مختلف تنظیم شوند تا مقدار داده‌هایی که صادر می‌شود را کاهش دهند و بهره‌وری و مقیاس‌پذیری را افزایش دهند.

دستورات

flow exporter

دستور flow exporter برای تعیین پارامترهای است که برای ارسال به سمت گیرنده اطلاعات IPFIX لازم داریم .

```
destination A.B.C.D
```

مقصدی که اطلاعات به آن ارسال میشود

```
source A.B.C.D
```

مبدا بسته های IPFIX . بدیهی است این IP باید در روتر وجود داشته باشد

```
transport udp (1-65535)
```

شماره پورت UDP که بسته ها به آن پورت ارسال می شوند .

توجه

پورت پیش 4739 است .

IPFIX flow monitor 2.2.15

دستورات

flow monitor

به تنظیمات flow monitor وارد می شوید .

```
cache timeout active (1-604800)
```

حداکثر زمانی که یک flow ی فعال می تواند در حالت active cache باقی بماند و پس از آن ارسال شود را مشخص می کند

توجه

مقدار پیش فرض 120 است.

```
cache timeout inactive (1-604800)
```

حداکثر زمانی که یک flow ی غیر فعال می تواند در حالت inactive cache باقی بماند و پس از آن حذف شود را مشخص می کند

توجه

مقدار پیش فرض 20 است .

```
record netflow <ipv4|ipv6> prefix-port
```

شروع به ضبط flow ها طبق الگوی 5 تایی (source address, destination address, protocol, source port و destination port) می کند .

3.2.15 اعمال در اینترفیس

```
ip flow monitor {output|input}
```

flow moitor را در اینترفیس اعمال می کند .

input: اطلاعات flow های ورودی به اینترفیس را جمع آوری می کند

output: اطلاعات flow های خروجی از اینترفیس را جمع آوری می کند

در مثال زیر یک flow exporter تعریف کرده ایم که مبدا و مقصد بسته های IPFIX را در آن مشخص کرده ایم بین ترتیب که بسته های IPFIX که حاوی اطلاعات flow ها هستند از طریق اینترفیس ge3 با مبدا 192.168.1.10 و مقصد 192.168.1.20 و udp port مقصد 15200 ارسال می شوند .

در بخش flow monitor تایمر های مد نظر را تنظیم کرده ایم . flow های فعال تا 120 ثانیه cache می شوند و پس از آن ارسال می شوند و flow های غیر فعال پس از گذشت 45 ثانیه حذف می شوند .

و در نهایت flow monitor را در اینترفیس ge0 و برای مانیتور کردن بسته های خروجی از اینترفیس اعمال کرده ایم.

```
soodar(config)# interface ge3
soodar(config-if)# ip address 192.168.1.10/24
soodar(config-if)# flow exporter
soodar(config-flow-exporter)# destination 192.168.1.20
soodar(config-flow-exporter)# source 192.168.1.10
soodar(config-flow-exporter)# transport udp 15200
-----------------------------------------
soodar(config-flow-exporter)# flow monitor
soodar(config-flow-monitor)# cache timeout active 120
soodar(config-flow-monitor)# cache timeout inactive 45
soodar(config-flow-monitor)# record netflow ipv4 prefix-port
-----------------------------------------
soodar(config-flow-monitor)# interface ge0
soodar(config-if)# ip flow monitor output
```

فصل 16

تنظیمات لاگ

1.16 log

1.1.16 فعال کردن log

دقت کنید قبل از debug کردن باید با دستور زیر logging را در روتر فعال کنید . دستور زیر log را در سطح errors فعال می کند :

```
soodar(config)# log syslog errors
```

2.1.16 تنظیم log server

تنظیم سرور لاگ می تواند به شکل زیر با پروتکل های udp , tcp و loki انجام شود :

```
soodar(config)# log syslog http://192.168.1.55 loki
soodar(config)# log syslog http://192.168.1.55 port 3000 tcp
soodar(config)# log syslog http://192.168.1.55 port 3100 udp
```

3.1.16 مشاهده وضعیت log

دستور زیر وضعیت logging را نشان می دهد :

```
soodar# sh logging
Syslog logging: level errors, facility daemon, ident zebra
File logging: disabled
Monitor logging: enabled
Record priority: disabled
Timestamp precision: 0
```

4.1.16 مشاهده لاگ سرویس ها

لاگ ها به توجه به سرویس هایی که آن لاگ را تولید می کنند قابل دسته بندی هستند و شما می توانید تمامی لاگها و یا لاگ های سرویس خاصی را مشاهده نمایید :

```
soodar# show log all
soodar# show log frr
soodar# show log mender
```

نکته

✓ ویرایشگر متن vim برای مشاهده لاگها در سودار استفاده شده است .
✓ با استفاده از q نیز می توانید از بخش مشاهده فایل لاگ خارج شوید .
✓ برای جست و جو در لاگها از / و عبارت مورد نظر استفاده نمایید .

5.1.16 مشاهده آنلاین لاگ

برای مشاهده log های روتر دستور زیر را وارد نمایید :

```
soodar# show log all follow
soodar# show log frr follow
```

نکته

با استفاده از Ctrl + c نیز می توانید از بخش مشاهده آنلاین لاگ خارج شوید .

6.1.16 log rotation

تنظیمات log rotation (شامل حداکثر سایز فایل ، حداکثر عمر فایل و ...) به شکل زیر قابل انجام است :

```
soodar(config)# log rotate
  max-file-life  Log file's life before rotation
  max-file-size  Single log file maximum size
  max-files      Total number of log files. when exceeds, older logs get removed
  max-retention  Log's rentetion
  max-use        Enforce size limits on the log files stored
```

نکته

حداقل سایز فایل لاگ 512K است.

log monitor 7.1.16

با فعال کردن log monitor برخی لاگ ها و پیام ها در زمان تنظیم کردن روتر برای شما نمایش داده می شود . پیام هایی مثل up/down شدن اینترفیس ها و تونل ها و ... . با no کردن هم غیر فعال می شود .

```
soodar(config)# log monitor

soodar(config)# no log monitor
```

clear log 8.1.16

این دستور همه لاگ های روتر را پاک می کند :

```
soodar# clear log syslog
```

Debug 2.16

1.2.16 فعال کردن debug

در بخش زیر نمونه هایی از فعال کردن debug را مشاهده می کنید :

```
soodar#  debug ospf packet hello send
soodar#  debug mpls ldp event
soodar#  debug vxlan event
soodar#  debug if event
soodar#  debug gre event
soodar#  debug vpls event
soodar#  debug vlan event
soodar#  debug ipsec event
soodar#  debug dplane ikev2
soodar#  debug vrf event
soodar#  debug vrf dplane event
.
.
.
```

2.2.16 مشاهده وضعیت debug

برای مشاهده debug های فعال از دستور زیر استفاده کنید :

```
soodar# sh debugging
Zebra debugging status:
  GRE events debugging is on
  IPSec events debugging is on
  VLAN events debugging is on
  VPLS events debugging is on
  VXLAN events debugging is on

OSPF debugging status:
  OSPF packet Hello send debugging is on


LDP debugging status:
  LDP events debugging is on



Staticd debugging status

Soosys debugging status:
```

فصل 17

اتصال از console

console 1.17

برای اتصال فیزیکی به روتر سودار دو راه وجود دارد :

1. اتصال از طریق پورت سریال

```
baudrate = 115200
type = 8n1
```

2. اتصال با استفاده از مانیتور و کی بورد

در این حالت کافی است یک مانیتور و کی بورد به روتر وصل کنید و روتر را تنظیم کنید .

فصل 18

تنظیمات امنیتی

1.18 بلاک کردن ip

با استفاده از دستورات زیر می توان تنظیم کرد اگر کاربری رمز عبور برای اتصال ssh اشتباه وارد کرد ip آن کاربر بلاک شود و نتواند ssh بزند

در مثال زیر اگر کاربر 5 بار در مدت 60 ثانیه رمز عبور ssh را اشتباه وارد کند به مدت 120 ثانیه بلاک خواهد شد .

```
soodar(config)# login block-for 120 attempts 5 within 60
```

توجه

اگر این تنظیم انجام نشود بصورت پیش اگر در مدت 60 ثانیه 5 بار لاگین ناموفق رخ دهد ip به مدت 600 ثانیه بلاک خواهد شد . در ضمن لاگین های موفق counter را صفر نمی کند یعنی اگر در بازه 60 ثانیه 4 بار لاگین ناموفق داشته باشد و سپس 1 لاگین موفق داشته باشد و سپس 1 لاگین ناموفق ، باز هم IP بلاک خواهد شد .

به کمک دستور زیر می توان ip های پلاک شده را unblock کرد.
می توانید با وارد کردن آدرس ip مورد نظر آن را unblock کنید یا با استفاده از گزینه all همه ip های بلاک شده را باز کنید .

```
soodar(config)# login unblock
  A.B.C.D   Unban a blocked conenction
  X:X::X:X  IPv4 address of banned connection
  all       IPv6 address of banned connection
```

2.18 urpf

کلمه uRPF تشکیل شده از کلمات Unicast Reverse Path Forwarding می‌باشد. مکانیزمی است به منظور جلوگیری از حملات جعل IP و یا همان IP Spoofing .

نحوه تنظیم urpf در سودار :

```
soodar(config)# int ge0
soodar(config-if)# ip verify unicast source reachable-via
  any  Source is reachable via any interface
  rx   Source is reachable via interface on which packet was received
soodar(config-if)# ip verify unicast source reachable-via rx
```

حالت any
در حالت any کافی است source بسته در جدول مسیریابی روتر وجود داشته باشد و مهم نیست که دقیقا از همان اینترفیسی که روتر به آن route دارد بسته دریافت شود .

حالت rx
در این حالت باید مبدا(source) بسته در جدول مسیریابی روتر موجود باشد و دقیقا در همان اینترفیسی که ما از طریق آن به آن مبدا route داریم دریافت شود و گرنه بسته drop خواهد شد .

session timeout 3.18

می توان برای session هایی که به shell روتر ایجاد می شود timeout مشخص کرد و پس از کارنکردن با vtysh در بازه زمانی تعیین شدن session به طور خودکار بسته خواهد شد .
در مثال زیر بعد از غیرفعال بودن نشست به مدت 1 دقیقه نشست بطور خودکار بسته خواهد شد

```
soodar/config# line vty
soodar/c/vty# exec-timeout
  (0-35791)  Timeout in minutes
soodar/c/vty# exec-timeout 1
  <cr>
  (0-2147483)  Timeout in seconds
soodar/c/vty# exec-timeout 1 0
soodar/c/vty#
```

show login failure 4.18

می توانید تلاش های ناموفقی که برای لاگین کردن به روتر انجام شده است را مشاهده کنید . چه از طریق ssh و چه از طریق (پورت سریال. console(این اطلاعات فقط برای یک ماه اخیر نگهداری می شود البته در لاگ های ssh وجود دارد

```
soodar# show login failures
admin      ssh:notty      192.168.1.30     Sun Jun 11 10:34
admin      ssh:notty      192.168.1.30     Sun Jun 11 10:34
admin      ssh:notty      192.168.1.30     Sun Jun 11 10:34
admin      ssh:notty      192.168.1.20     Wed Jun 7 10:43
admin      ssh:notty      192.168.1.30     Wed Jun 7 10:30
admin      ssh:notty      192.168.1.20     Tue Jun 6 15:14
iman       ssh:notty      192.168.1.20     Tue Jun 6 15:13
admin      ssh:notty      192.168.1.30     Thu Jun 1 08:17
admin      ssh:notty      192.168.1.30     Thu Jun 1 08:17
admin      ssh:notty      192.168.1.30     Thu Jun 1 08:17
admin      ssh:notty      192.168.1.30     Thu Jun 1 08:11
soodar#
```

show user 5.18

کاربرانی در حال حاضر به روتر لاگین کرده اند را نمایش می دهد

```
soodar# sh user

    Number  Line     User    Peer Address        Session
------------------------------------------------------------
  * 0       SSH      admin   192.168.1.3:34718   c5
    1       Console  admin   ---                 c6
```

clear line 6.18

می توانید یک کاربر را با شماره tty line آن که در دستور قبل مشخص است را از cli روتر بیرون کنید و ارتباط او با روتر قطع شود

```
soodar# clear line 1
[Ok]
informational-SOOSYS: session c6(0.0.0.0:0) terminated by user admin(192.168.1.3:34718)
soodar#
```

فصل 19

Tune

برخی تنظیمات برای عملکرد بهتر روتر از لحاظ performance و کارایی می تواند در اختصاص cpu memory انجام گیرد تا روتر با توجه به کاربردی که دارد و منابع سخت افزاری که در اختیار دارد در بهترین حالت ممکن قرار داشته باشد .

control-palne 1.19

می توانید core های cpu را مدیریت کنید و مثلا core 2 را به engine سودار اختصاص دهید و دیگر پروسس ها از آن استفاده نکند

```
soodar(config)# system tune profile CONTROL_PLANE_TUNE
soodar(tune-profile)# control-plane
soodar(tune-cp-cfg)# cpu exclusive 2
soodar(tune-cp-cfg)# memory max 1G
```

data-plane 2.19

در بخش data-plane هم تنظیماتی برای cpu و memory وجود دارد : برای مثال می توانید core3 از cpu را به data-palne اختصاص دهید :

```
soodar(config)# system tune profile DATA_PLANE_TUNE
soodar(tune-profile)# data-plane
soodar(tune-dp-cfg)# cpu main exclusive 3
```

یا حداکثر memory که data-plane می تواند به خود اختصاص دهد را محدود کنید :

```
soodar(tune-dp-cfg)# memory max 8G
```

interface mapping 1.2.19

همچنین تیون دیگری نیز برای map کردن نام اینترفیس ها وجود دارد . سیستم عامل سودار اینترفیس ها را طبق id کارت شبکه نام گذاری می کند . گاهی اوقات ممکن است چینش کارت های شبکه به گونه ای باشد که ترتیب اینترفیس ها درست نباشد (در واقع ID کارت های شبکه به ترتیب نباشد) اینجا شما می توانید با تیون کردن دستگاه ترتیب ها را اصلاح نمایید . در زیر یک نمونه تیون اینترفیس را مشاهده می کنید :

```
soodar(config)# system tune profile interface-mapping
soodar(tune-profile)# data-plane
soodar(tune-dp-cfg)# map interface ge2 0
interface ge2 changed to interface ge0
soodar(tune-dp-cfg)#
```

فرض کنید به صورت پیش فرض اطلاعات کارت شبکه به شکل زیر باشد :

Pci Address lrsphinxstyletheadfamily order lrsphinxstyletheadfamily Interface Name lrsphinxstyletheadfamily Mac Address			
00:04.0	0	ge0	0c:61:0c:83:00:00
00:05.0	1	ge1	0c:61:0c:83:00:01
00:06.0	2	ge2	0c:61:0c:83:00:02
00:07.0	3	ge3	0c:61:0c:83:00:03

ادمین بعد از بررسی متوجه می شود که اولین اینترفیس ، ge2 شناخته شده است و باید اینترفیس

با اعمال تیون تغییر به شکل زیر اعمال خواهد شد :

Pci Address lrsphinxstyletheadfamily order lrsphinxstyletheadfamily Interface Name lrsphinxstyletheadfamily Mac Address			
00:06.0	0	ge0	0c:61:0c:83:00:02
00:05.0	1	ge1	0c:61:0c:83:00:01
00:04.0	2	ge2	0c:61:0c:83:00:00
00:07.0	3	ge3	0c:61:0c:83:00:03

توجه

با استفاده از دستور beacon در هر اینترفیس می توانید چراغ اینترفیس را روشن کنید و نام و جایگیری فیزیکی اینترفیس ها را تشخیص دهید.

فصل 20

capture کردن بسته ها

1.20 capture کردن بسته ها

گاهی اوقات لازم است بسته های در حال عبور روتر را رصد کنیم تا بتوانیم علت رفتار نامطلوب روتر را بیابیم . در روتر سودار شما می توانید به دو شکل این کار را انجام دهید . در capture شما بسته های رد و بدل شده در اینترفیس ها را می توانید capture کنید و سپس با export کردن در سیستم دیگر ، آن ها را مشاهده و تحلیل نمایید . در حالت dispatch trace اطلاعات بیشتری در مورد بسته ها و مسیرهایی که بسته در dataplane طی می کند ارائه می دهد که این اطلاعات برای گزارش به تیم پشتیبانی سودار جهت رفع مشکل احتمالی مفید است و ممکن است برای ادمین شبکه خیلی مفید و قابل توجه نباشد . بنابراین ادمین شبکه بیشتر از capture استفاده می کند و بسته ها را پش از انتقال به دستگه دیگر با نرم افزارهای تحلیل شبمه مانند wiresharsk می تواند بسته های ورودی و خروجی به روتر را مشاهده کرده و برای رفع مشکل احتمالی تصمیم بگیرد .

1.1.20 capute

```
soodar# monitor capture  ?
direction  Packets direction to capture
export     Export Buffer
interface  Interface
limit      Limit number of packets to capture
start      Enable Capture
stop       Disable Capture
```

direction: می توان بسته های ورودی یا خروجی یا هردو که به اینترفیس خاص یا همه اینترفیس ها وارد/خارج می شوند را ضبط کرد

export: انتقال فایل ذخیره شده به کامپیوتر دیگر با این گزینه انجام می شود .

interface: اینترفیسی که قصد گوش دادن در آن را دارید

limit: تعدا بسته هایی که باید ضبط شوند را می توانید محدود کنید

start: شروع به گوش دادن می کند

stop : به گوش دادن خاتمه می دهد و فایل را ذخیره می کند

مثال : در این نمونه capture در اینترفیس ge1 و بسته های ورودی و خروجی (both) و با محدودیت حداکثر 1000 بسته start شده است :

```
soodar# monitor capture interface ge1 direction both limit 1000 start
```

بدین شکل نیز captrure کردن را متوقف می کنیم و

```
soodar# monitor capture  stop
Write 274 packets to /tmp/capture.pcap, and stop capture...

soodar#
```

و سپس برای بررسی بسته ها در سیستم دیگر آن را export می کنیم :

توجه

قبل از انتقال فایل به کامپیوتر دیگر باید به آن یکبارssh زده باشید و کلید آن در روتر وجود داشته باشد

```
soodar# monitor  capture export scp:ubuntu@192.168.111.78:
Address or name of remote host [192.168.111.78]?
Remote host user [ubuntu]?
Remote host password [admin]?
soodar#
```

2.1.20 dispatch trace

در dispatch اینترفیس نداریم و dispatch در کل dataplane انجام می شود :

```
soodar# monitor dispatch-trace limit 200000 start
```

```
soodar# monitor dispatch-trace stop
Write 728 packets to /tmp/dispatch_capture.pcap, and stop capture...

soodar#
```

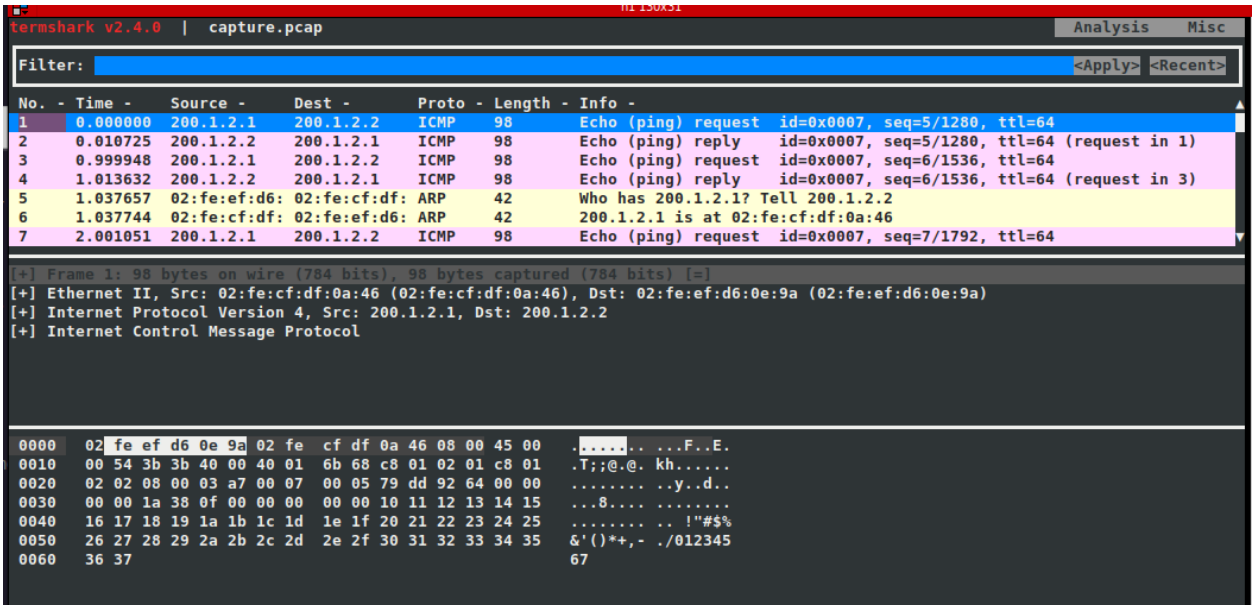
```
soodar# monitor dispatch-trace  export scp:ubuntu@192.168.111.78:
Address or name of remote host [192.168.111.78]?
Remote host user [ubuntu]?
Remote host password [admin]?
soodar#
```

3.1.20 مشاهده فایل capture شده در خود روتر

با دستور زیر می توانید فایل های dispatch-trace , capture که با دستورات قبل در روتر ذخیره شده اند را مشاهده کنید و آن را تجزیه تحلیل نمایید :

```
soodar# sh monitor capture
soodar# sh monitor dispatch
```

در تصویر زیر یک نمونه فایل capture شده را می بینید که در cli روتر باز شده است :


termshark v2.4.0 | capture.pcap
Analysis Misc
Filter:
<Apply> <Recent>
No. - Time - Source - Dest - Proto - Length - Info -
1 0.000000 200.1.2.1 200.1.2.2 ICMP 98 Echo (ping) request id=0x0007, seq=5/1280, ttl=64
2 0.010725 200.1.2.2 200.1.2.1 ICMP 98 Echo (ping) reply id=0x0007, seq=5/1280, ttl=64 (request in 1)
3 0.999948 200.1.2.1 200.1.2.2 ICMP 98 Echo (ping) request id=0x0007, seq=6/1536, ttl=64
4 1.013632 200.1.2.2 200.1.2.1 ICMP 98 Echo (ping) reply id=0x0007, seq=6/1536, ttl=64 (request in 3)
5 1.037657 02:fe:ef:d6: 02:fe:cf:df: ARP 42 Who has 200.1.2.1? Tell 200.1.2.2
6 1.037744 02:fe:cf:df: 02:fe:ef:d6: ARP 42 200.1.2.1 is at 02:fe:cf:df:0a:46
7 2.001051 200.1.2.1 200.1.2.2 ICMP 98 Echo (ping) request id=0x0007, seq=7/1792, ttl=64
[+] Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) [=]
[+] Ethernet II, Src: 02:fe:cf:df:0a:46 (02:fe:cf:df:0a:46), Dst: 02:fe:ef:d6:0e:9a (02:fe:ef:d6:0e:9a)
[+] Internet Protocol Version 4, Src: 200.1.2.1, Dst: 200.1.2.2
[+] Internet Control Message Protocol
0000 02 fe ef d6 0e 9a 02 fe cf df 0a 46 08 00 45 00 .....F..E.
0010 00 54 3b 3b 40 00 40 01 6b 68 c8 01 02 01 c8 01 .T;;@.@. kh......
0020 02 02 08 00 03 a7 00 07 00 05 79 dd 92 64 00 00 ..y..d..
0030 00 00 1a 38 0f 00 00 00 00 00 10 11 12 13 14 15 ...8....
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 !"#$%
0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 &'()*+,- ./012345
0060 36 37 67

فصل 21

برخی دستورات خاص

find 1.21

برای پیدا کردن دستوری که دقیق نمی دانیم چطور باید از آن استفاده کنیم کافی است با استفاده از دستور find آن را پیدا کنیم .بدین شکل که ابتدا دستور find را وارد کرده سپس عبارت مورد نظر را وارد کنید . این عبارت می تواند به شکل regex باشد . در زیر نمونه هایی را مشاهده می کنید :

```
soodar# find ike
  (view)  show crypto ikev2 sa
  (enable)  [no] debug dplane ikev2
  (enable)  show crypto ikev2 sa
  (config)  crypto ikev2 proposal IKEPOSAL
  (config)  crypto ikev2 profile IKEPROF
  (config)  crypto ikev2 keyring IKEKEYRING
  (config)  [no] debug dplane ikev2
  (config)  crypto ikev2 dpd (1-3600) [(1-100)]
  (config)  no crypto ikev2 keyring IKEKEYRING
  (config)  no crypto ikev2 profile IKEPROF
  (config)  no crypto ikev2 proposal IKEPOSAL
  (ipsec profile)  set ikev2-profile IKEPROF
```

config , enable , view نشان می دهد که هر دستور کجا قابل استفاده است

```
soodar# find syslog*
  (config)  log syslog [<emergencies|alerts|critical|errors|warnings|notifications|informational|debugging>]
  (config)  no log syslog [<emergencies|alerts|critical|errors|warnings|notifications|informational|debugging>]
  (config)  log facility <kern|user|mail|daemon|auth|syslog|lpr|news|uucp|cron|local0|local1|local2|local3|local4|local5|local6|local7>
  (config)  no log facility [<kern|user|mail|daemon|auth|syslog|lpr|news|uucp|cron|local0|local1|local2|local3|local4|local5|local6|local7>]
  (config)  log syslog <A.B.C.D|X:X::X:X|HOST> [port (100-65535)] [<udp|tcp>]
  (config)  log syslog <A.B.C.D|X:X::X:X|HOST> [port (100-65535)] loki
```

2.21 فیلتر کردن خروجی

با دستور زیر می توانید بخش خاصی از تنظیمات را ببینید . برای مثال می توانید تنظیمات بخش router ospf را به شکل زیر مشاهده کنید :

```
soodar# sh running-config  | section router ospf
router ospf
 ospf router-id 222.1.1.1
 redistribute connected
 redistribute static
```

یا به شکل زیر هر خطی که در آن ge0 وجود دارد را در خروجی چاپ کند :

```
soodar# sh running-config  | include ge0
interface ge0
soodar#
```

و یا به شکل زیر می توان مشخص کرد که همه تنظیمات به جز بخش router ospf در خروجی نمایش داده شود

```
soodar# sh running-config  | section-exclude router ospf
```

3.21 نمایش command history

مشاهده تاریخچه دستوراتی که در روتر وارد شده است توسط دستور زیر میسر می شود . این تاریخچه شامل تمامی دستوراتی است که کاربر admin وارد کرده است :

```
soodar#show command history

service password-encryption
hostname SOODAR
reload
en
show clock
clock timezone UTC
```

4.21 حذف command history

با دستور زیر می توانید تمامی دستورات را حذف کنید یا مشخص کنید چند دستور آخر را حفظ کرده و بقیه را حذف کنید :

حذف تمامی دستورات :

```
soodar(config)# clear command history
```

حذف تمامی تاریخچه دستورات بجز 20 دستور آخر :

```
soodar(config)# clear command history 20
```

show diagnostic 5.21

با استفاده از این دستور می توانید یک گزارش از بخش های مختلف سیستم داشته باشید . شامل : وضعیت اینترفیس ها ، وضعیت سرویس ها ، کرش های رخ داده و خطاهای احتمالی در دیتا پلن

```
soodar# sh diagnostic
```

Interface status:

Interface	Admin state	Link state	Speed(Mb/s)	Linkups	Linkdowns	Last linkup	Last linkdown
ge1	up(up)	up(up)	100/(100)	1	1	21 days, 19:03:06 ago	21 days, 19:03:11 ago
ge2	up(up)	up(up)	1000/(1000)	1	1	21 days, 19:03:06 ago	21 days, 19:03:11 ago
ge3	up(up)	up(up)	1000/(1000)	1	1	21 days, 19:03:06 ago	21 days, 19:03:11 ago
ge0	up(up)	up(up)	100/(100)	1	1	21 days, 19:03:06 ago	21 days, 19:03:11 ago

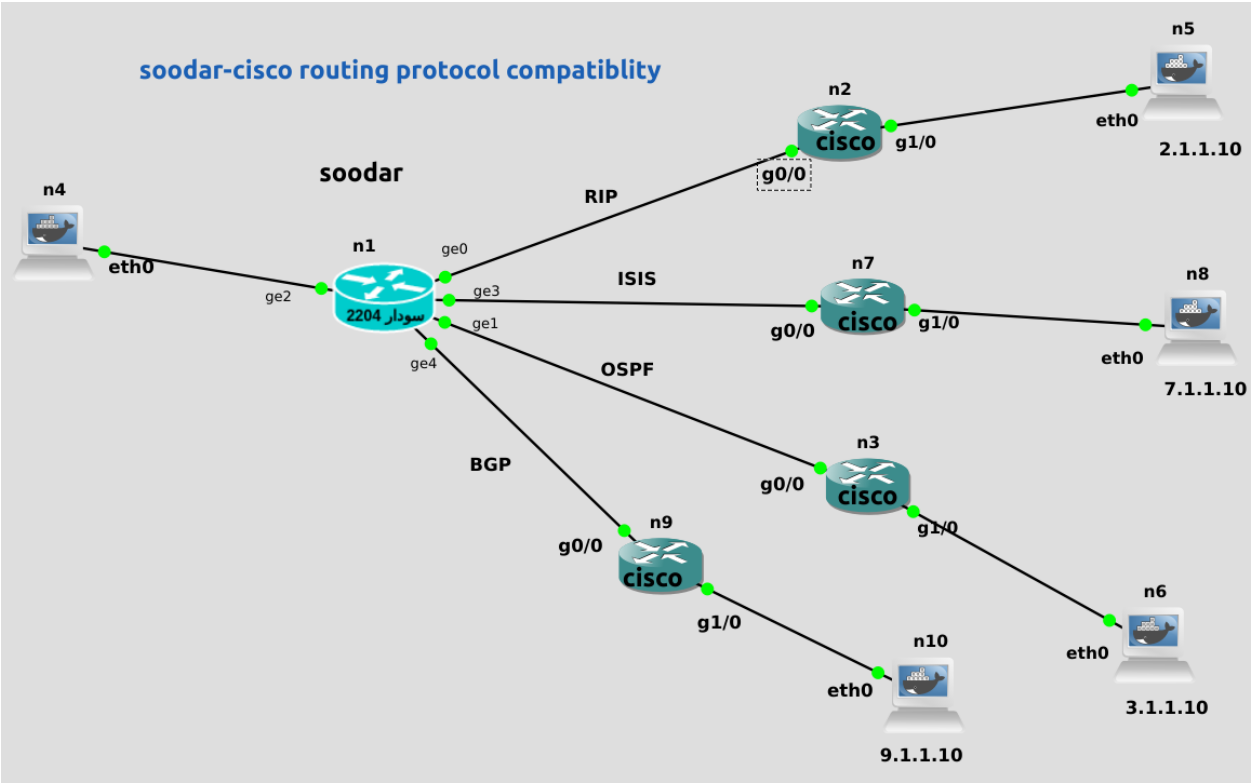
Services status:

Service	Active state	Sub state	Automatic restarts	Last activatied	Last inactivated
frr	active	running	0	21 days, 19:03:10 ago	-
vpp	active	running	0	21 days, 19:03:21 ago	-
soosys	active	running	0	21 days, 19:03:21 ago	-
strongswan	active	running	0	21 days, 19:03:18 ago	-
soosla	active	running	0	21 days, 19:03:10 ago	-
fail2ban	active	running	0	21 days, 19:03:18 ago	-
soolog	failed	failed	0	21 days, 19:03:07 ago	0:00:10 ago
kea-dhcp4	active	running	0	21 days, 19:03:12 ago	-
mender-client	inactive	dead	0	-	-
mender-connect	inactive	dead	0	-	-
soomon-vpp	active	running	0	21 days, 19:03:07 ago	21 days, 19:03:09 ago
soomon-node	active	running	0	21 days, 19:03:07 ago	21 days, 19:03:07 ago
soomon-process	active	running	0	21 days, 19:03:09 ago	21 days, 19:03:09 ago
soomon-ipsec	active	running	0	21 days, 19:03:09 ago	21 days, 19:03:09 ago
chronyd	active	running	0	4 days, 17:14:42 ago	4 days, 17:14:42 ago
snmpd	active	running	0	21 days, 19:03:20 ago	-

فصل 22

ارتباط با Cisco

در این بخش قصد داریم ارتباط روتر سودار با روتر cisco با پروتکل های مسیریابی RIP,OSPF,ISIS و BGP بررسی کنیم . فرض کنید سناریوی زیر را داریم که روتر n1 سودار و بقیه روتر ها cisco هستند سودار با هر کدام از روترهای cisco با یک پروتکل مسیریابی کار می کند :



1.22 همسایگی RIP بین سودار و cisco

1.1.22 تنظیمات RIP در روتر سودار

```
router rip
 network 200.1.2.0/24
 redistribute connected
```

2.1.22 تنظیمات RIP در روتر cisco

```
router rip
 redistribute connected
 network 200.1.2.0
```

بررسی همسایگی در سودار و در سیسکو

سودار :

```
n1# sh ip rip status
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 28 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected
  Default version control: send version 2, receive any version
    Interface        Send  Recv   Key-chain
    ge0              2     1 2
  Routing for Networks:
    200.1.2.0/24
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    200.1.2.2             1         0       120    00:00:12
  Distance: (default is 120)
n1#
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 01:23:01
```

(continues on next page)

(continued from previous page)

```
R>* 2.0.0.0/8 [120/2] via 200.1.2.2, ge0, weight 1, 00:10:04
C>* 200.1.2.0/24 is directly connected, ge0, 01:23:01
C>* 200.1.3.0/24 is directly connected, ge1, 01:23:01
C>* 200.1.7.0/24 is directly connected, ge3, 01:23:01
C>* 200.1.9.0/24 is directly connected, ge4, 01:23:01
C>* 222.1.1.1/32 is directly connected, loopback100, 01:23:07
n1#
```

:cisco

```
n2#sh ip rip database
1.0.0.0/8    auto-summary
1.1.1.0/24
    [1] via 200.1.2.1, 00:00:03, GigabitEthernet0/0
2.0.0.0/8    auto-summary
2.1.1.0/24    redistributed
    [1] via 0.0.0.0,
200.1.2.0/24    auto-summary
200.1.2.0/24    directly connected, GigabitEthernet0/0
```

```
n2#sh ip route rip
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is not set

     1.0.0.0/24 is subnetted, 1 subnets
R       1.1.1.0 [120/1] via 200.1.2.1, 00:00:19, GigabitEthernet0/0
R    200.1.3.0/24 [120/1] via 200.1.2.1, 00:00:19, GigabitEthernet0/0
R    200.1.7.0/24 [120/1] via 200.1.2.1, 00:00:19, GigabitEthernet0/0
R    200.1.9.0/24 [120/1] via 200.1.2.1, 00:00:19, GigabitEthernet0/0
     222.1.1.0/32 is subnetted, 1 subnets
R       222.1.1.1 [120/1] via 200.1.2.1, 00:00:19, GigabitEthernet0/0
n2#
```

2.22 همسایگی OSPF بین سودار و cisco

1.2.22 تنظیمات OSPF در روتر سودار

```
interface ge1
 no shutdown
 ip address 200.1.3.1/24
```

(continues on next page)

(continued from previous page)

```
 ip ospf area 0
exit
!
interface ge2
 no shutdown
 ip address 1.1.1.1/24
 ip ospf area 0
 ip ospf passive
exit
!


router ospf
 ospf router-id 222.1.1.1
 auto-cost reference-bandwidth 100
!
```

2.2.22 تنظیمات OSPF در روتر cisco

```
interface GigabitEthernet0/0
 ip address 200.1.3.3 255.255.255.0
 ip ospf 1 area 0
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 !
!
interface GigabitEthernet1/0
 ip address 3.1.1.1 255.255.255.0
 ip ospf 1 area 0
 negotiation auto
 !
!
router ospf 1
 router-id 222.3.3.3
 log-adjacency-changes
 passive-interface GigabitEthernet1/0
!
```

بررسی همسایگی در سودار و در سیسکو

سودار :

```
n1# sh ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
            RXmtL RqstL DBsmL
222.3.3.3         1 Full/DR         39.188s 200.1.3.3       ge1:200.1.3.1
                0    0    0
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 01:43:44
R>* 2.0.0.0/8 [120/2] via 200.1.2.2, ge0, weight 1, 00:30:47
O>* 3.1.1.0/24 [110/101] via 200.1.3.3, ge1, weight 1, 00:03:15
C>* 200.1.2.0/24 is directly connected, ge0, 01:43:44
C>* 200.1.3.0/24 is directly connected, ge1, 01:43:44
C>* 200.1.7.0/24 is directly connected, ge3, 01:43:44
C>* 200.1.9.0/24 is directly connected, ge4, 01:43:44
C>* 222.1.1.1/32 is directly connected, loopback100, 01:43:50
```

سیسکو:

```
n3#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
222.1.1.1         1   FULL/BDR        00:00:34    200.1.3.1       GigabitEthernet0/0
```

```
n3#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is not set

     1.0.0.0/24 is subnetted, 1 subnets
O       1.1.1.0 [110/101] via 200.1.3.1, 00:04:45, GigabitEthernet0/0
O E2  2.0.0.0/8 [110/20] via 200.1.3.1, 00:04:45, GigabitEthernet0/0
     3.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       3.1.1.0/24 is directly connected, GigabitEthernet1/0
L       3.1.1.1/32 is directly connected, GigabitEthernet1/0
O E2  200.1.2.0/24 [110/20] via 200.1.3.1, 00:04:45, GigabitEthernet0/0
     200.1.3.0/24 is variably subnetted, 2 subnets, 2 masks
C       200.1.3.0/24 is directly connected, GigabitEthernet0/0
L       200.1.3.3/32 is directly connected, GigabitEthernet0/0
O E2  200.1.7.0/24 [110/20] via 200.1.3.1, 00:04:45, GigabitEthernet0/0
O E2  200.1.9.0/24 [110/20] via 200.1.3.1, 00:04:45, GigabitEthernet0/0
     222.1.1.0/32 is subnetted, 1 subnets
O E2    222.1.1.1 [110/20] via 200.1.3.1, 00:04:47, GigabitEthernet0/0
n3#
```

3.22 همسایگی ISIS بین سودار و cisco

1.3.22 تنظیمات ISIS در روتر سودار

```
!
debug isis events
!
interface lo
 no ip address
exit
!
interface ge0
 no shutdown
 ip address 200.1.2.1/24
exit
!
interface ge1
 no shutdown
 ip address 200.1.3.1/24
exit
!
interface ge2
 no shutdown
 ip address 1.1.1.1/24
exit
!
interface ge3
 no shutdown
 ip address 200.1.7.1/24
 ip router isis test
exit
!
interface loopback100
 no shutdown
 ip address 222.1.1.1/32
exit
!
router isis test
 net 47.0023.1111.1111.00
 metric-style transition
 redistribute ipv4 connected level-1
 redistribute ipv4 connected level-2
 redistribute ipv4 ospf level-1
 redistribute ipv4 ospf level-2
 redistribute ipv4 rip level-1
 redistribute ipv4 rip level-2
exit
!
end
```

2.3.22 تنظیمات ISIS در روتر cisco

```
interface Loopback100
 ip address 222.7.7.7 255.255.255.255
 !
!
interface Ethernet0/0
 no ip address
 shutdown
 duplex auto
 !
!
interface GigabitEthernet0/0
 ip address 200.1.7.7 255.255.255.0
 ip router isis test
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 !
!
interface GigabitEthernet1/0
 ip address 7.1.1.1 255.255.255.0
 negotiation auto
 !
!
!
!
router isis test
 net 47.0023.7777.7777.00
 lsp-mtu 1000
 redistribute connected
!
```

نکته

در تنظیم isis یک گزینه به نام metric-style وجود دارد که برای محاسبه متریک لینک ها استفاده می شود که سه حالت برای آن وجود دارد narrow , transition و wide . در soodar به صورت پیش فرض در حالت wide قرار دارد اما در سیسکو در حالت narrow . باید این تنظیم در دو طرف یکسان باشد یا در یک طرف transition باشد تا route ها تبادل گردد .

بررسی همسایگی در سودار و در سیسکو

سودار :

```
n1# sh isis neighbor
Area test:
  System Id           Interface   L  State        Holdtime SNPA
 n7                 ge3         1  Up           8        ca03.c2db.0008
 n7                 ge3         2  Up           7        ca03.c2db.0008
n1#
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
```

(continues on next page)

(continued from previous page)

```
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 01:55:21
R>* 2.0.0.0/8 [120/2] via 200.1.2.2, ge0, weight 1, 00:42:24
O>* 3.1.1.0/24 [110/101] via 200.1.3.3, ge1, weight 1, 00:14:52
I>* 7.1.1.0/24 [115/10] via 200.1.7.7, ge3, weight 1, 01:54:56
B>* 9.1.1.0/24 [20/0] via 200.1.9.9, ge4, weight 1, 01:28:21
C>* 200.1.2.0/24 is directly connected, ge0, 01:55:21
C>* 200.1.3.0/24 is directly connected, ge1, 01:55:21
C>* 200.1.7.0/24 is directly connected, ge3, 01:55:21
C>* 200.1.9.0/24 is directly connected, ge4, 01:55:21
C>* 222.1.1.1/32 is directly connected, loopback100, 01:55:27
I>* 222.7.7.7/32 [115/10] via 200.1.7.7, ge3, weight 1, 01:54:56
B>* 222.9.9.9/32 [20/0] via 200.1.9.9, ge4, weight 1, 01:28:21
n1#
```

سیسکو:

```
n7#sh isis neighbors

System Id      Type Interface   IP Address      State Holdtime Circuit Id
n1             L1   Gi0/0       200.1.7.1       UP    29       n7.01
n1             L2   Gi0/0       200.1.7.1       UP    28       n7.01
```

```
n7#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is 200.1.7.1 to network 0.0.0.0

i*L1  0.0.0.0/0 [115/10] via 200.1.7.1, GigabitEthernet0/0
      1.0.0.0/24 is subnetted, 1 subnets
i L1     1.1.1.0 [115/10] via 200.1.7.1, GigabitEthernet0/0
i L1  2.0.0.0/8 [115/10] via 200.1.7.1, GigabitEthernet0/0
      3.0.0.0/24 is subnetted, 1 subnets
i L1     3.1.1.0 [115/10] via 200.1.7.1, GigabitEthernet0/0
      7.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        7.1.1.0/24 is directly connected, GigabitEthernet1/0
L        7.1.1.1/32 is directly connected, GigabitEthernet1/0
i L1  200.1.2.0/24 [115/10] via 200.1.7.1, GigabitEthernet0/0
i L1  200.1.3.0/24 [115/10] via 200.1.7.1, GigabitEthernet0/0
      200.1.7.0/24 is variably subnetted, 2 subnets, 2 masks
C        200.1.7.0/24 is directly connected, GigabitEthernet0/0
L        200.1.7.7/32 is directly connected, GigabitEthernet0/0
i L1  200.1.9.0/24 [115/10] via 200.1.7.1, GigabitEthernet0/0
      222.1.1.0/32 is subnetted, 1 subnets
```

(continues on next page)

(continued from previous page)

```
i L1     222.1.1.1 [115/10] via 200.1.7.1, GigabitEthernet0/0
      222.7.7.0/32 is subnetted, 1 subnets
C        222.7.7.7 is directly connected, Loopback100
n7#
```

4.22 همسایگی BGP بین سودار و cisco

1.4.22 تنظیمات BGP در روتر سودار

```
interface ge4
 no shutdown
 ip address 200.1.9.1/24
exit
!
router bgp 65001
 no bgp ebgp-requires-policy
 neighbor 200.1.9.9 remote-as 65002
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute rip
  redistribute ospf
  redistribute isis
 exit-address-family
exit
!
```

2.4.22 تنظیمات BGP در روتر cisco

```
interface GigabitEthernet0/0
 ip address 200.1.9.9 255.255.255.0
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 !
!
interface GigabitEthernet1/0
 ip address 9.1.1.1 255.255.255.0
 negotiation auto
 !
!
!
router bgp 65002
 bgp log-neighbor-changes
```

(continues on next page)

(continued from previous page)

```
 neighbor 200.1.9.1 remote-as 65001
 !
 address-family ipv4
  no synchronization
  redistribute connected
  neighbor 200.1.9.1 activate
  no auto-summary
 exit-address-family
!
```

بررسی همسایگی در سودار و در سیسکو

سودار:

```
n1# sh ip bgp summary

IPv4 Unicast Summary (VRF default):
BGP router identifier 222.1.1.1, local AS number 65001 vrf-id 0
BGP table version 37
RIB entries 23, using 4232 bytes of memory
Peers 1, using 723 KiB of memory

Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down Sta
te/PfxRcd   PfxSnt Desc
200.1.9.9       4      65002       155       171        0    0    0 00:05:54
        3       12 N/A

Total number of neighbors 1
n1#
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 02:09:13
R>* 2.0.0.0/8 [120/2] via 200.1.2.2, ge0, weight 1, 00:56:16
O>* 3.1.1.0/24 [110/101] via 200.1.3.3, ge1, weight 1, 00:28:44
I>* 7.1.1.0/24 [115/10] via 200.1.7.7, ge3, weight 1, 02:08:48
B>* 9.1.1.0/24 [20/0] via 200.1.9.9, ge4, weight 1, 00:07:15
C>* 200.1.2.0/24 is directly connected, ge0, 02:09:13
C>* 200.1.3.0/24 is directly connected, ge1, 02:09:13
C>* 200.1.7.0/24 is directly connected, ge3, 02:09:13
C>* 200.1.9.0/24 is directly connected, ge4, 02:09:13
C>* 222.1.1.1/32 is directly connected, loopback100, 02:09:19
I>* 222.7.7.7/32 [115/10] via 200.1.7.7, ge3, weight 1, 02:08:48
B>* 222.9.9.9/32 [20/0] via 200.1.9.9, ge4, weight 1, 00:07:15
n1#
```

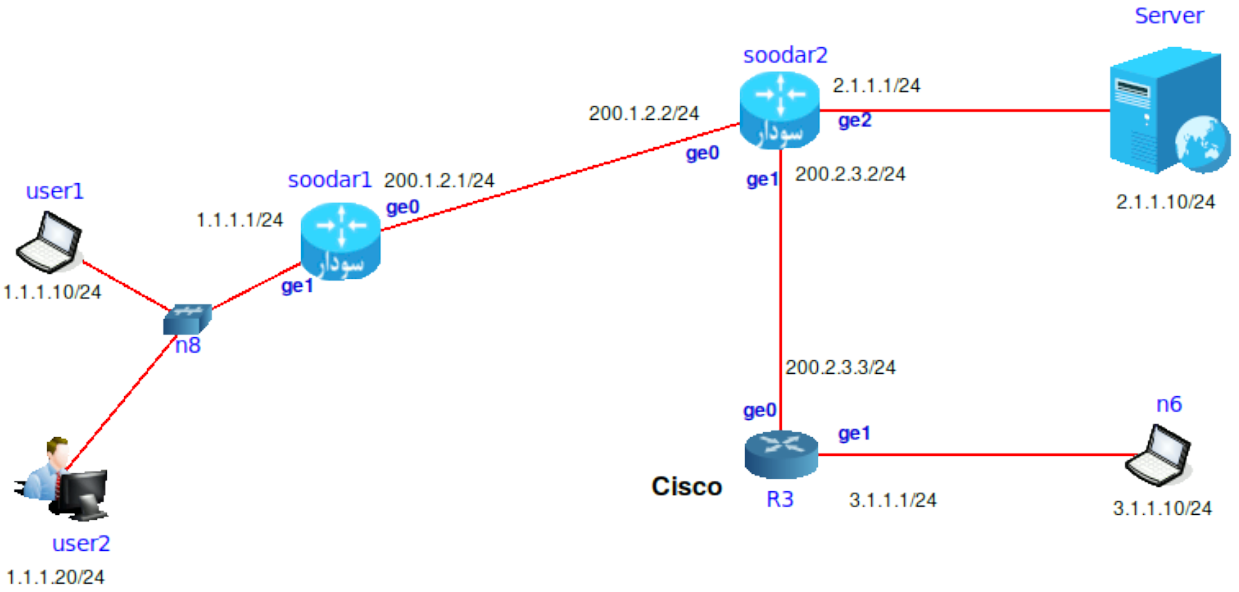
سیسکو:

```
n9# sh ip bgp summary
BGP router identifier 222.9.9.9, local AS number 65002
BGP table version is 59, main routing table version 59
12 network entries using 1440 bytes of memory
13 path entries using 676 bytes of memory
5/5 BGP path/bestpath attribute entries using 620 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 2760 total bytes of memory
BGP activity 30/18 prefixes, 37/24 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
200.1.9.1       4        65001      18      11       59    0    0 00:06:16        10
```

```
n9#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is not set

     1.0.0.0/24 is subnetted, 1 subnets
B       1.1.1.0 [20/0] via 200.1.9.1, 00:07:49
B    2.0.0.0/8 [20/2] via 200.1.9.1, 00:07:49
     3.0.0.0/24 is subnetted, 1 subnets
B       3.1.1.0 [20/101] via 200.1.9.1, 00:07:49
     7.0.0.0/24 is subnetted, 1 subnets
B       7.1.1.0 [20/10] via 200.1.9.1, 00:07:49
     9.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       9.1.1.0/24 is directly connected, GigabitEthernet1/0
L       9.1.1.1/32 is directly connected, GigabitEthernet1/0
B    200.1.2.0/24 [20/0] via 200.1.9.1, 00:07:49
B    200.1.3.0/24 [20/0] via 200.1.9.1, 00:07:49
B    200.1.7.0/24 [20/0] via 200.1.9.1, 00:07:49
     200.1.9.0/24 is variably subnetted, 2 subnets, 2 masks
C       200.1.9.0/24 is directly connected, GigabitEthernet0/0
L       200.1.9.9/32 is directly connected, GigabitEthernet0/0
     222.1.1.0/32 is subnetted, 1 subnets
B       222.1.1.1 [20/0] via 200.1.9.1, 00:07:51
     222.7.7.0/32 is subnetted, 1 subnets
B       222.7.7.7 [20/10] via 200.1.9.1, 00:07:51
     222.9.9.0/32 is subnetted, 1 subnets
C       222.9.9.9 is directly connected, Loopback100
n9#
```

5.22 تست MPLS با روتر Cisco

برای تست سازگار بودن روتر با روتر های سیسکو سناریویی شامل 3 روتر که یک روتر سیسکو با دو روتر سودار در یک شبکه قرار گرفته اند . پروتکل OSPF برای مسیریابی انتخاب شده است همچنین پروتکل MPLS نیز در شبکه فعال شده است .



تنظیمات اعمال شده در روتر ها را در ادامه می بینیم (نحوه تنظیم MPLS , OSPF را می توانید در اینجا مشاهده کنید تنظیم OSPF ، تنظیم MPLS) . سپس در بخش نهایی جداول مسیریابی و جدول mpls در روتر ها را مشاهده کرده و ارتباطات شبکه ها را بررسی می کنیم :

تنظیمات soodar1

```
soodar1# sh run

Building configuration...

Current configuration:
log file
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
hostname soodar1
enable password s
!
interface loopback0
 no shutdown
 ip address 222.1.1.1/32
!
interface ge0
 mpls ip
```

(continues on next page)

(continued from previous page)

```
 no shutdown
 ip address 200.1.2.1/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
!
interface ge1
 mpls ip
 no shutdown
 ip address 1.1.1.1/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
 !
router ospf
 ospf router-id 222.1.1.1
 redistribute kernel
 redistribute connected
 redistribute static
 network 1.1.1.0/24 area 0
 network 200.1.2.0/24 area 0
 network 222.1.1.1/32 area 0
 !
mpls ldp
 router-id 222.1.1.1
 dual-stack transport-connection prefer ipv4
 dual-stack cisco-interop
 neighbor 222.2.2.2 password testmpls
 neighbor 222.3.3.3 password testmpls
 !
 address-family ipv4
  discovery transport-address 222.1.1.1
  !
  interface ge0
  !
  interface ge1
  !
 exit-address-family
 !
end
```

تنظیمات soodar2

```
soodar2# sh run



Building configuration...

Current configuration:
log file
system update server-url https://update.soodar.ir
system update update-poll-interval 10
system update inventory-poll-interval 15
no ip forwarding
no ipv6 forwarding
```

(continues on next page)

(continued from previous page)

```
hostname soodar2
enable password s
!
interface loopback0
 no shutdown
 ip address 222.2.2.2/32
!
interface ge0
 mpls ip
 no shutdown
 ip address 200.1.2.2/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
!
interface ge1
 mpls ip
 no shutdown
 ip address 200.2.3.2/24
 ip ospf hello-interval 3
 ip ospf dead-interval 10
!
interface ge2
 no shutdown
 ip address 2.1.1.1/24
!
router ospf
 ospf router-id 222.2.2.2
 redistribute kernel
 redistribute connected
 redistribute static
 network 2.1.1.0/24 area 0
 network 200.1.2.0/24 area 0
 network 200.2.3.0/24 area 0
 network 222.2.2.2/32 area 0
!
mpls ldp
 router-id 222.2.2.2
 dual-stack transport-connection prefer ipv4
 dual-stack cisco-interop
 neighbor 222.1.1.1 password testmpls
 neighbor 222.3.3.3 password testmpls
 !
 address-family ipv4
  discovery transport-address 222.2.2.2
  !
  interface ge0
  !
  interface ge1
  !
 exit-address-family
 !
end
```

تنظیمات روتر cisco

```
R3#sh run


Building configuration...

Current configuration : 1449 bytes
upgrade fpd auto
version 15.0
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
hostname R3
boot-start-marker
boot-end-marker
no aaa new-model

ip source-route
no ip icmp rate-limit unreachable
ip cef

no ip domain lookup
no ipv6 cef
multilink bundle-name authenticated
mpls ldp neighbor 222.2.2.2 password testmpls
redundancy
ip tcp synwait-time 5
interface Loopback0
 ip address 222.3.3.3 255.255.255.0
 !
interface Ethernet0/0
 no ip address
 shutdown
 duplex auto
 !
interface GigabitEthernet0/0
 ip address 200.2.3.3 255.255.255.0
 ip ospf hello-interval 3
 ip ospf dead-interval 10
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 mpls ip
 !
interface GigabitEthernet1/0
 ip address 3.1.1.1 255.255.255.0
 negotiation auto
 !
router ospf 1
 router-id 222.3.3.3
 log-adjacency-changes
 network 3.1.1.0 0.0.0.255 area 0
 network 200.2.3.0 0.0.0.255 area 0
 network 222.3.3.0 0.0.0.255 area 0
```

(continues on next page)

(continued from previous page)

```
ip forward-protocol nd
no ip http server
no ip http secure-server
no cdp log mismatch duplex
control-plane
 !
mgcp fax t38 ecm
mgcp behavior g729-variants static-pt
gatekeeper
 shutdown
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
end
```

1.5.22 مشاهده جداول **mpls , ospf** در روتر ها

soodar1 2.5.22

```
soodar1# sh ip fib

Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.0/24 is directly connected, soo1, 00:29:19
O>* 2.1.1.0/24 [110/20000] via 200.1.2.2, soo0, label implicit-null, 00:44:18
O>* 3.1.1.0/24 [110/20001] via 200.1.2.2, soo0, label 18, 00:20:15
C>* 200.1.2.0/24 is directly connected, soo0, 00:44:33
O>* 200.2.3.0/24 [110/20000] via 200.1.2.2, soo0, label implicit-null, 00:44:18
C>* 222.1.1.1/32 is directly connected, loopback0, 00:44:33
O>* 222.2.2.2/32 [110/20000] via 200.1.2.2, soo0, label implicit-null, 00:44:18
O>* 222.3.3.3/32 [110/20001] via 200.1.2.2, soo0, label 20, 00:20:15
```

```
soodar1# sh mpls ldp binding

------------------------------------------------
AF   Destination          Nexthop       Local Label Remote Label  In Use
ipv4 1.1.1.0/24           222.2.2.2     imp-null    19            no
```

(continues on next page)

(continued from previous page)

```
ipv4 2.1.1.0/24          222.2.2.2      16          imp-null        yes
ipv4 3.1.1.0/24          222.2.2.2      19          18              yes
ipv4 111.1.1.0/24        222.2.2.2      imp-null    imp-null        no
ipv4 200.1.2.0/24        222.2.2.2      imp-null    imp-null        no
ipv4 200.2.3.0/24        222.2.2.2      17          imp-null        yes
ipv4 222.1.1.1/32        222.2.2.2      imp-null    17              no
ipv4 222.2.2.2/32        222.2.2.2      18          imp-null        yes
ipv4 222.3.3.3/32        222.2.2.2      20          20              yes
```

soodar2

```
soodar2# sh ip fib

Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O>* 1.1.1.0/24 [110/20000] via 200.1.2.1, soo0, label implicit-null, 00:30:40
C>* 2.1.1.0/24 is directly connected, soo2, 00:45:54
O>* 3.1.1.0/24 [110/10001] via 200.2.3.3, soo1, label implicit-null, 00:21:36
C>* 111.1.1.0/24 is directly connected, ctrl0, 00:45:55
C>* 200.1.2.0/24 is directly connected, soo0, 00:45:54
C>* 200.2.3.0/24 is directly connected, soo1, 00:45:54
O>* 222.1.1.1/32 [110/20000] via 200.1.2.1, soo0, label implicit-null, 00:45:31
C>* 222.2.2.2/32 is directly connected, loopback0, 00:45:54
O>* 222.3.3.3/32 [110/10001] via 200.2.3.3, soo1, 00:21:36
```

```
soodar2# sh mpls ldp binding

AF   Destination        Nexthop       Local Label Remote Label  In Use
ipv4 1.1.1.0/24         222.1.1.1     19          imp-null        yes
ipv4 1.1.1.0/24         222.3.3.3     19          19              yes
ipv4 2.1.1.0/24         222.1.1.1     imp-null    16              no
ipv4 2.1.1.0/24         222.3.3.3     imp-null    18              no
ipv4 3.1.1.0/24         222.1.1.1     18          19              yes
ipv4 3.1.1.0/24         222.3.3.3     18          imp-null        yes
ipv4 111.1.1.0/24       222.1.1.1     imp-null    imp-null        no
ipv4 111.1.1.0/24       222.3.3.3     imp-null    21              no
ipv4 200.1.2.0/24       222.1.1.1     imp-null    imp-null        no
ipv4 200.1.2.0/24       222.3.3.3     imp-null    20              no
ipv4 200.2.3.0/24       222.1.1.1     imp-null    17              no
ipv4 200.2.3.0/24       222.3.3.3     imp-null    imp-null        no
ipv4 222.1.1.1/32       222.1.1.1     17          imp-null        yes
ipv4 222.1.1.1/32       222.3.3.3     17          17              yes
ipv4 222.2.2.2/32       222.1.1.1     imp-null    18              no
ipv4 222.2.2.2/32       222.3.3.3     imp-null    16              no
ipv4 222.3.3.0/24       222.3.3.3     -           imp-null        yes
ipv4 222.3.3.3/32       222.1.1.1     20          20              yes
```

R3

```
R3#sh ip route
```

(continues on next page)

(continued from previous page)

```
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is not set

      1.0.0.0/24 is subnetted, 1 subnets
O        1.1.1.0 [110/20001] via 200.2.3.2, 00:19:07, GigabitEthernet0/0
      2.0.0.0/24 is subnetted, 1 subnets
O        2.1.1.0 [110/10001] via 200.2.3.2, 00:19:07, GigabitEthernet0/0
      3.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        3.1.1.0/24 is directly connected, GigabitEthernet1/0
L        3.1.1.1/32 is directly connected, GigabitEthernet1/0
      111.0.0.0/24 is subnetted, 1 subnets
O E2     111.1.1.0 [110/20] via 200.2.3.2, 00:19:07, GigabitEthernet0/0
O     200.1.2.0/24 [110/10001] via 200.2.3.2, 00:19:07, GigabitEthernet0/0
      200.2.3.0/24 is variably subnetted, 2 subnets, 2 masks
C        200.2.3.0/24 is directly connected, GigabitEthernet0/0
L        200.2.3.3/32 is directly connected, GigabitEthernet0/0
      222.1.1.0/32 is subnetted, 1 subnets
O        222.1.1.1 [110/20001] via 200.2.3.2, 00:19:09, GigabitEthernet0/0
      222.2.2.0/32 is subnetted, 1 subnets
O        222.2.2.2 [110/10001] via 200.2.3.2, 00:19:09, GigabitEthernet0/0
      222.3.3.0/24 is variably subnetted, 2 subnets, 2 masks
C        222.3.3.0/24 is directly connected, Loopback0
L        222.3.3.3/32 is directly connected, Loopback0
```

```
R3#sh mpls ldp bindings

  lib entry: 1.1.1.0/24, rev 14
        local binding:  label: 19
        remote binding: lsr: 222.2.2.2:0, label: 19
  lib entry: 2.1.1.0/24, rev 12
        local binding:  label: 18
        remote binding: lsr: 222.2.2.2:0, label: imp-null
  lib entry: 3.1.1.0/24, rev 6
        local binding:  label: imp-null
        remote binding: lsr: 222.2.2.2:0, label: 18
  lib entry: 111.1.1.0/24, rev 18
        local binding:  label: 21
        remote binding: lsr: 222.2.2.2:0, label: imp-null
  lib entry: 200.1.2.0/24, rev 16
        local binding:  label: 20
        remote binding: lsr: 222.2.2.2:0, label: imp-null
  lib entry: 200.2.3.0/24, rev 4
        local binding:  label: imp-null
        remote binding: lsr: 222.2.2.2:0, label: imp-null
  lib entry: 222.1.1.1/32, rev 10
        local binding:  label: 17
        remote binding: lsr: 222.2.2.2:0, label: 17
  lib entry: 222.2.2.2/32, rev 8
        local binding:  label: 16
```

(continues on next page)

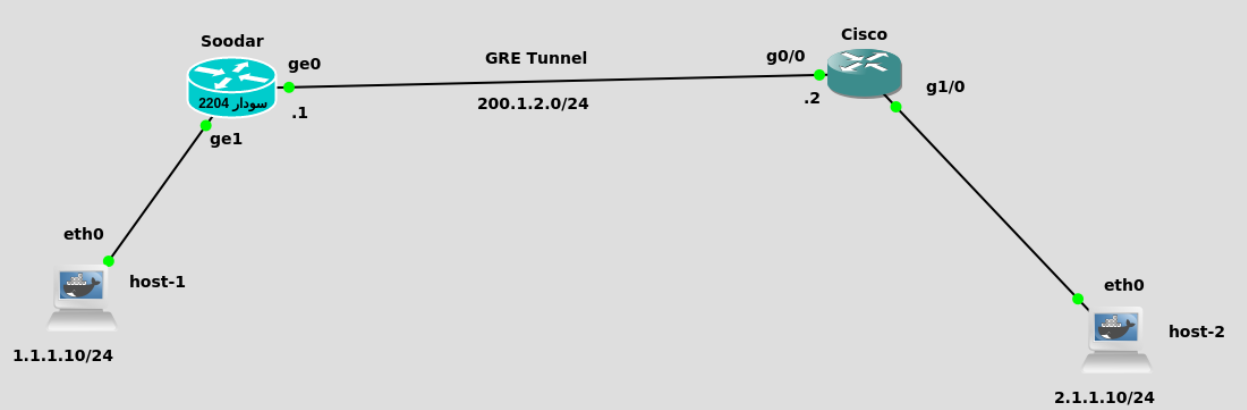
(continued from previous page)

```
    remote binding: lsr: 222.2.2.2:0, label: imp-null
lib entry: 222.3.3.0/24, rev 2
    local binding:  label: imp-null
lib entry: 222.3.3.3/32, rev 19
    remote binding: lsr: 222.2.2.2:0, label: 20
```

بررسی عملکرد

1. ارتباط شبکه های 3.1.1.0 , 2.1.1.0 , 1.1.1.0 برقرار شده است و روتر سودار با روتر سیسکو با پروتکل OSPF ارتباط گرفته است . جدول مسیریابی روتر ها به درستی ایجاد شده است .
2. ارتباط با روتر سیسکو با استفاده از پروتکل mpls نیز میسر می باشد و بسته ها در مسیر هایی که می باید با label و پروتکل mpls انتقال می یابند . جداول mpls نیز به درستی تشکیل شده اند .

6.22 تونل GRE بین سودار و cisco



1.6.22 تنظیمات در سودار

```
hostname soodar
no ipv6 forwarding
no zebra nexthop kernel enable
security passwords min-length 8
log syslog errors
log monitor
no banner motd
!
no ntp
!
ip route 2.1.1.0/24 tunnel10
```

(continues on next page)

(continued from previous page)

```
!
interface lo
 no ip address
!
interface tunnel10
 tunnel source 200.1.2.1
 tunnel destination 200.1.2.2
 no shutdown
 ip address 10.1.2.1/24
exit
!
interface ge0
 no shutdown
 ip address 200.1.2.1/24
exit
!
interface ge1
 no shutdown
 ip address 1.1.1.1/24
exit
!
end
```

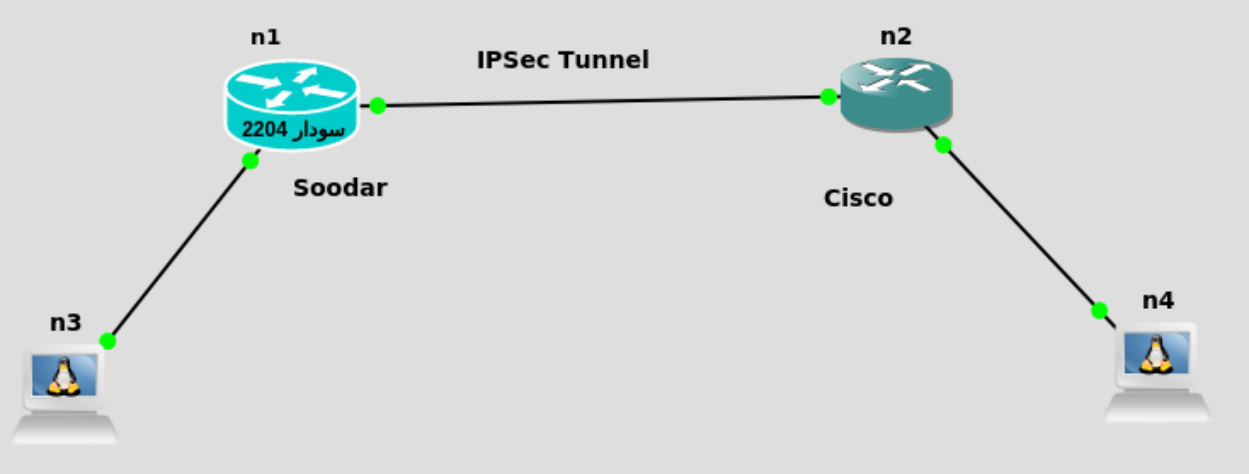
2.6.22 تنظیمات در روتر cisco

```
interface Tunnel200
 ip address 10.1.2.2 255.255.255.0
 tunnel source 200.1.2.2
 tunnel destination 200.1.2.1
 !
!
interface Ethernet0/0
 no ip address
 shutdown
 duplex auto
 !
!
interface GigabitEthernet0/0
 ip address 200.1.2.2 255.255.255.0
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 !
!
interface GigabitEthernet1/0
 ip address 2.1.1.1 255.255.255.0
 negotiation auto
 !
!
ip route 1.1.1.0 255.255.255.0 Tunnel200
```

7.22 تونل IPSec بین سودار و cisco

1.7.22 نحوه تنظیم تونل ipsec در cisco و سودار

در این مقاله قصد داریم نحوه تنظیم تونل ipsec بین روتر سیسکو و روتر سودار را شرح دهیم . تونل ipsec دارای دو فاز است که فاز اول تنظیم **IKE** و فاز دوم تنظیم **IPSec** می باشد .در ادامه نحوه پیکربندی روتر سیسکو و سودار را در این دو فاز به صورت گام به گام ارائه می دهیم .



2.7.22 1.نحوه تنظیم ike (فاز اول)

IKE پروتکلی است برای راه اندازی و تشکیل SA که جهت استفاده در تونل ipsec به کار گرفته می شود .
در سیسکو تنظیم profile برای بخش ike استفاده می شود . این تنظیم در سودار در proposal مربوط به ike انجام می پذیرد :

تنظیمات **IKE** در سودار

```
crypto ikev2 proposal PH1-PROPOSAL
 integrity sha-256
 encryption aes-256
 group 20
!
crypto ikev2 profile PH1-PROFILE
 keyring local PH1-KEYRING
 identity local address 222.1.1.1
 match identity remote address 222.2.2.2
 authentication local pre-share
 authentication remote pre-share
 proposal
!
crypto ikev2 keyring PH1-KEYRING
 peer CISCO
  address 200.1.2.2
  pre-shared-key salam
```

(continues on next page)

(continued from previous page)

```
  identity address 222.2.2.2
!
```

تنظیمات **IKE** در **cisco**

```
crypto ikev2 proposal PP
 encryption aes-cbc-256
 integrity sha256
 group 20
!
no crypto ikev2 policy default
crypto ikev2 policy pL
 proposal PP
!
crypto ikev2 keyring KEYRING
 peer SOODAR
  address 200.1.2.1
  identity address 222.1.1.1
  pre-shared-key salam
 !
crypto ikev2 profile PF
 match identity remote address 222.1.1.1 255.255.255.255
 identity local address 222.2.2.2
 authentication remote pre-share
 authentication local pre-share
 keyring local KEYRING
```

3.7.22 2.نحوه تنظیم **ipsec** (فاز دوم)

تنظیمات **ipsec** در سودار

```
crypto ipsec transform-set TS esp hmac sha-256 cipher aes-256
 mode transport
!
crypto ipsec profile PH2-PROFILE
 set transform-set TS
 set ikev2-profile PH1-PROFILE
!
```

تنظیمات **ipsec** در **Cisco**

```
crypto ipsec transform-set TS esp-aes 256 esp-sha256-hmac
 mode transport
!
crypto ipsec profile IPPF
 set transform-set TS
 set ikev2-profile PF
!
```

4.7.22 3.ساخت تونل **gre** و محافظت آن با **ipsec**

دقت شود که مقدار mtu در اینترفیس gre باید از مقدار پیش فرض به 1380 تغییر کند تا سربار افزایشی ipsec روی بسته ها هم در نظر گرفته شود :

تونل **gre** محافظت شده در سودار

```
interface tunnel10
 tunnel source 200.1.2.1
 tunnel destination 200.1.2.2
 tunnel protection ipsec profile PH2-PROFILE
 no shutdown
 ip address 10.0.12.1/24
```

تونل **gre** محافظت شده در **Cisco**

```
interface Tunnel21
 ip address 10.0.12.21 255.255.255.0
 tunnel source 200.1.2.2
 tunnel destination 200.1.2.1
 tunnel protection ipsec profile IPPF
```

5.7.22 4.اضافه کردن **static route**

در آخرین مرحله باید یک route اضافه کنیم تا دسترسی بین pc2 , pc1 از طریق تونل ipsec برقرار شود . برای این کار باید در سیسکو route شبکه 1.1.1.0/24 و در روتر سودار route شبکه 2.1.1.0/24 را از طریق تونلی که ایجاد کرده ایم تعریف نماییم :

static route در سودار

```
ip route 2.1.1.0/24 tunnel10
```

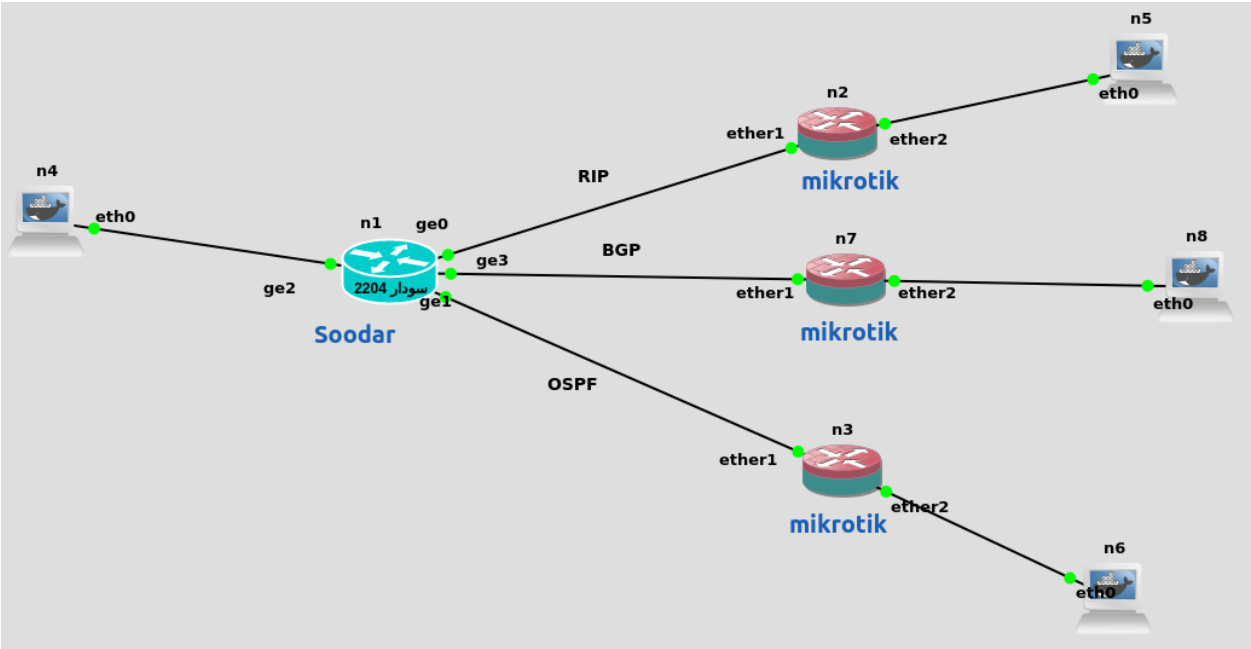
static route در **Cisco**

```
ip route 1.1.1.0 255.255.255.0 Tunnel21
```

فصل 23

ارتباط با MikroTik

در این بخش قصد داریم ارتباط روتر سودار با روتر Mikrotik با پروتکل های مسیریابی RIP,OSPF و BGP بررسی کنیم . فرض کنید سناریوی زیر را داریم که روتر n1 سودار و بقیه روتر ها Mikrotik هستند سودار با هر کدام از روترهای Mikrotik با یک پروتکل مسیریابی کار می کند :



1.23 همسایگی RIP بین سودار و Mikrotik

1.1.23 تنظیمات RIP در روتر سودار

```
router rip
 network 200.1.2.0/24
```

(continues on next page)

(continued from previous page)

```
redistribute connected
```

2.1.23 تنظیمات RIP در روتر Mikrotik

```
[admin@MikroTik] > /ip address/print
Columns: ADDRESS, NETWORK, INTERFACE
# ADDRESS       NETWORK    INTERFACE
0 200.1.2.2/24  200.1.2.0  ether1
1 2.1.1.1/24    2.1.1.0    ether2
```

```
[admin@MikroTik] > /routing/rip/interface/print
 0 instance=1 address=200.1.2.2/ether1

 1 instance=1 address=2.1.1.1/ether2
```

```
[admin@MikroTik] > /routing/rip/instance/print
Flags: X - disabled
 0   name="1" redistribute=connected
```

```
[admin@MikroTik] > /routing/rip/export
# jun/17/2023 05:22:10 by RouterOS 7.1rc7
# software id =
#
/routing rip instance
add name=1 redistribute=connected
/routing rip interface-template
add instance=1
```

بررسی همسایگی در سودار و در میکروتیک

سودار :

```
n1# sh ip rip status
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 28 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected
  Default version control: send version 2, receive any version
    Interface        Send  Recv   Key-chain
    ge0              2     1 2
  Routing for Networks:
    200.1.2.0/24
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    200.1.2.2             1         0       120    00:00:12
  Distance: (default is 120)
n1#
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 01:23:01
R>* 2.0.0.0/8 [120/2] via 200.1.2.2, ge0, weight 1, 00:10:04
C>* 200.1.2.0/24 is directly connected, ge0, 01:23:01
C>* 200.1.3.0/24 is directly connected, ge1, 01:23:01
C>* 200.1.7.0/24 is directly connected, ge3, 01:23:01
C>* 200.1.9.0/24 is directly connected, ge4, 01:23:01
C>* 222.1.1.1/32 is directly connected, loopback100, 01:23:07
n1#
```

:Mikrotik

```
[admin@MikroTik] > /routing/rip/neighbor/print
Flags: D - dynamic
 0 D instance=1 address=200.1.2.1%ether1 routes=5 packets-total=34
    packets-bad=0 entries-bad=0 last-update=6s
```

```
[admin@MikroTik] > /ip route/print
Flags: D - DYNAMIC; A - ACTIVE; c, r, y - COPY
Columns: DST-ADDRESS, GATEWAY, DISTANCE
   DST-ADDRESS   GATEWAY    DISTANCE
DAr 1.1.1.0/24    200.1.2.1      120
DAc 2.1.1.0/24    ether2           0
DAc 200.1.2.0/24  ether1           0
DAr 200.1.3.0/24  200.1.2.1      120
DAr 200.1.7.0/24  200.1.2.1      120
[admin@MikroTik] >
```

2.23 همسایگی **OSPF** بین سودار و **Mikrotik**

1.2.23 تنظیمات **OSPF** در روتر سودار

```
router ospf
 ospf router-id 222.1.1.1
 redistribute connected
 redistribute rip
 network 200.1.3.0/24 area 0
!
```

2.2.23 تنظیمات OSPF در روتر Mikrotik

```
interface GigabitEthernet0/0
 ip address 200.1.3.3 255.255.255.0
 ip ospf 1 area 0
 duplex full
 speed 1000
 media-type gbic
 negotiation auto
 !
!
interface GigabitEthernet1/0
 ip address 3.1.1.1 255.255.255.0
 ip ospf 1 area 0
 negotiation auto
 !
!
router ospf 1
 router-id 222.3.3.3
 log-adjacency-changes
 passive-interface GigabitEthernet1/0
!
```

بررسی همسایگی در سودار و در میکروتیک

سودار :

```
n1# sh ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface
            RXmtL RqstL DBsmL
200.1.3.3         128 Full/DR         38.398s 200.1.3.3       ge1:200.1.3.1
                0     0     0
```

```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 00:25:58
R>* 2.1.1.0/24 [120/2] via 200.1.2.2, ge0, weight 1, 00:25:57
O>* 3.1.1.0/24 [110/101] via 200.1.3.3, ge1, weight 1, 00:25:42
C>* 200.1.2.0/24 is directly connected, ge0, 00:25:58
C>* 200.1.3.0/24 is directly connected, ge1, 00:25:58
C>* 200.1.7.0/24 is directly connected, ge3, 00:25:58
```

میکروتیک:

```
[admin@MikroTik] > /routing/ospf/instance/print
Flags: X - disabled, I - inactive
```

(continues on next page)

(continued from previous page)

```
 0   name="ospf-instance-1" version=2 vrf=main router-id=main
     redistribute=connected
```

```
[admin@MikroTik] > /routing/ospf/interface/print
Flags: D - dynamic
 0 D address=200.1.3.3/ether1 area=0.0.0.0 state=dr network-type=broadcast
     bdr=200.1.3.1 cost=1 priority=128 retransmit-interval=5s
     transmit-delay=1s hello-interval=10s dead-interval=40s
```

```
[admin@MikroTik] > /routing/ospf/neighbor/print
Flags: V - virtual; D - dynamic
 0  D instance=ospf-instance-1 area=0.0.0.0 address=200.1.3.1 priority=1
      router-id=222.1.1.1 dr=200.1.3.3 bdr=200.1.3.1 state="Full"
      state-changes=6 adjacency=19m48s timeout=32s
[admin@MikroTik] >
```

```
[admin@MikroTik] > /ip route/print
Flags: D - DYNAMIC; A - ACTIVE; c, o, y - COPY
Columns: DST-ADDRESS, GATEWAY, DISTANCE
    DST-ADDRESS   GATEWAY          DISTANCE
DAo 1.1.1.0/24    200.1.3.1/ether1      110
DAo 2.1.1.0/24    200.1.3.1/ether1      110
DAc 3.1.1.0/24    ether2                  0
DAo 200.1.2.0/24  200.1.3.1/ether1      110
DAc 200.1.3.0/24  ether1                  0
DAo 200.1.7.0/24  200.1.3.1/ether1      110
```

3.23 همسایگی BGP بین سودار و Mikrotik

1.3.23 تنظیمات BGP در روتر سودار

```
interface ge3
 no shutdown
 ip address 200.1.7.1/24
exit
!
router bgp 1717
 no bgp ebgp-requires-policy
 neighbor 200.1.7.7 remote-as 7171
 !
 address-family ipv4 unicast
  redistribute connected
  redistribute rip
  redistribute ospf
 exit-address-family
exit
```

2.3.23 تنظیمات BGP در روتر Mikrotik

```
[admin@MikroTik] > /ip address/print
Columns: ADDRESS, NETWORK, INTERFACE
# ADDRESS      NETWORK   INTERFACE
0 7.1.1.1/24   7.1.1.0   ether2
1 200.1.7.7/24 200.1.7.0 ether1
```

```
[admin@MikroTik] > /routing/bgp/export
# jun/17/2023 05:41:43 by RouterOS 7.1rc7
# software id =
#
/routing bgp connection
add address-families=ip as=7171 disabled=no local.role=ebgp name=bgp1 \
    output.redistribute=connected remote.address=200.1.7.1 .as=1717 \
    router-id=222.7.7.7 routing-table=main
[admin@MikroTik] >
```

بررسی همسایگی در سودار و در میکروتیک

سودار:

```
n1# sh ip bgp summary

IPv4 Unicast Summary (VRF default):
BGP router identifier 200.1.7.1, local AS number 1717 vrf-id 0
BGP table version 13
RIB entries 13, using 2392 bytes of memory
Peers 1, using 723 KiB of memory

Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down Sta
te/PfxRcd   PfxSnt Desc
200.1.7.7       4       7171        44        49        0    0    0 00:37:01
        2        7 N/A

Total number of neighbors 1
```

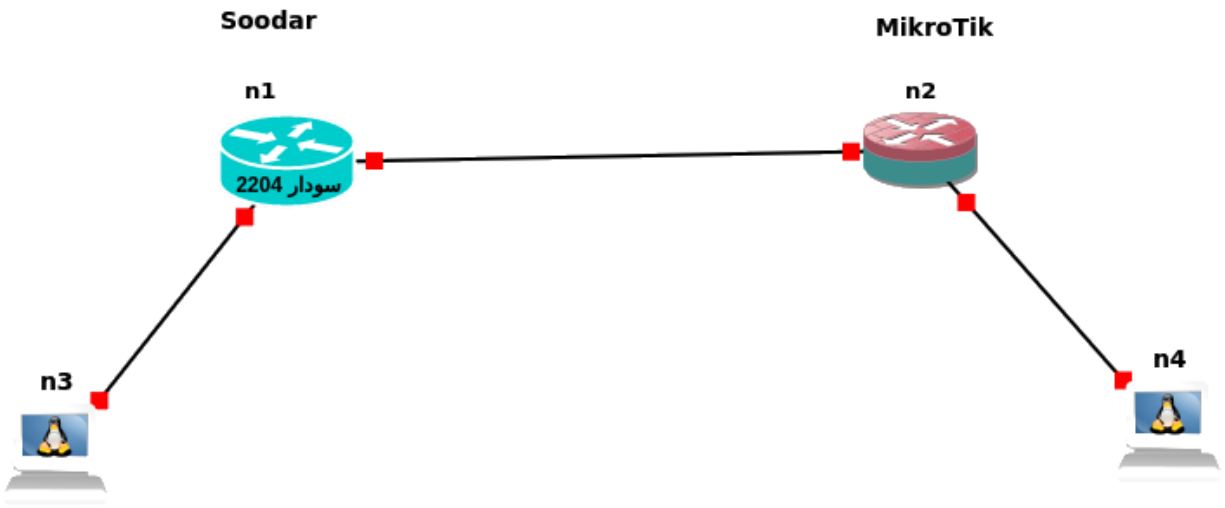
```
n1# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge2, 00:37:07
R>* 2.1.1.0/24 [120/2] via 200.1.2.2, ge0, weight 1, 00:37:06
O>* 3.1.1.0/24 [110/101] via 200.1.3.3, ge1, weight 1, 00:36:51
B>* 7.1.1.0/24 [20/0] via 200.1.7.7, ge3, weight 1, 00:37:06
C>* 200.1.2.0/24 is directly connected, ge0, 00:37:07
C>* 200.1.3.0/24 is directly connected, ge1, 00:37:07
C>* 200.1.7.0/24 is directly connected, ge3, 00:37:07
n1#
```

میکروتیک:

```
[admin@MikroTik] > /routing/bgp/session/print
Flags: E - established
 0 E remote.address=200.1.7.1 .as=1717 .id=200.1.7.1 .refused-cap-opt=no
     .capabilities=mp,rr,em,gr,as4,ap,err,fqdn .messages=47 .bytes=1146
     .gr-time=120 .eor=ip
     local.address=200.1.7.7 .as=7171 .id=222.7.7.7
     .capabilities=mp,rr,gr,as4 .messages=41 .bytes=812 .eor=""
     output.procid=20
     input.procid=20 ebgp
     hold-time=3m keepalive-time=1m
```

```
[admin@MikroTik] > /ip route/print
Flags: D - DYNAMIC; A - ACTIVE; c, b, y - COPY
Columns: DST-ADDRESS, GATEWAY, DISTANCE
   DST-ADDRESS   GATEWAY    DISTANCE
DAb 1.1.1.0/24    200.1.7.1       20
DAb 2.1.1.0/24    200.1.7.1       20
DAb 3.1.1.0/24    200.1.7.1       20
DAc 7.1.1.0/24    ether2           0
DAb 200.1.2.0/24  200.1.7.1       20
DAb 200.1.3.0/24  200.1.7.1       20
D b 200.1.7.0/24  200.1.7.1       20
DAc 200.1.7.0/24  ether1           0
```

4.23 تونل GRE بین سودار و MikroTik



1.4.23 تنظیمات در سودار

```
ip route 2.1.1.0/24 tunnel10
!
interface ge2
 no ip address
!
interface ge3
 no ip address
!
interface lo
 no ip address
!
interface tunnel10
 tunnel source 200.1.2.1
 tunnel destination 200.1.2.2
 no shutdown
 ip address 10.0.12.1/24
exit
!
interface ge0
 no shutdown
 ip address 200.1.2.1/24
exit
!
interface ge1
 no shutdown
 ip address 1.1.1.1/24
exit
!
end
```

2.4.23 تنظیمات در روتر **mikrorik**

```
/interface gre
add local-address=200.1.2.2 name=tunnel21 remote-address=200.1.2.1

/ip address
add address=200.1.2.2/24 interface=ether1 network=200.1.2.0
add address=2.1.1.1/24 interface=ether2 network=2.1.1.0
add address=10.0.12.2/24 interface=tunnel21 network=10.0.12.0

/ip route
add dst-address=1.1.1.0/24 gateway=tunnel21
```

5.23 تونل **IPSec** بین سودار و **MikroTik**

1.5.23 نحوه تنظیم تونل **ipsec** در میکروتیک و سودار

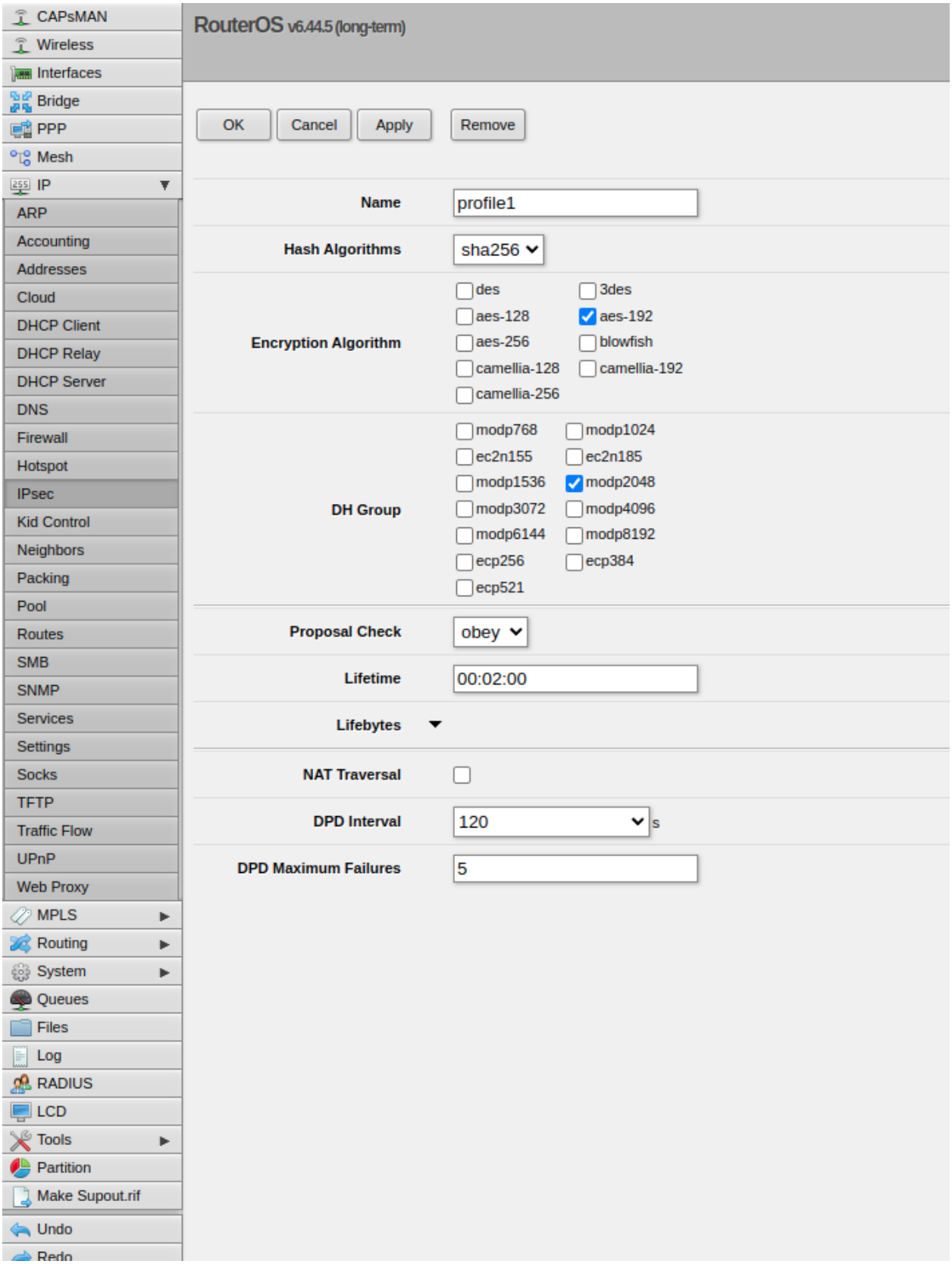
در این مقاله قصد داریم نحوه تنظیم تونل ipsec بین روتر میکروتیک و روتر سودار را شرح دهیم . تونل ipsec دارای دو فاز است که فاز اول تنظیم **IKE** و فاز دوم تنظیم **IPSec** می باشد .در ادامه نحوه پیکربندی روتر میکروتیک و سودار را در این دو فاز به صورت گام به گام ارائه می دهیم .

2.5.23 1.نحوه تنظیم **ike** (فاز اول)

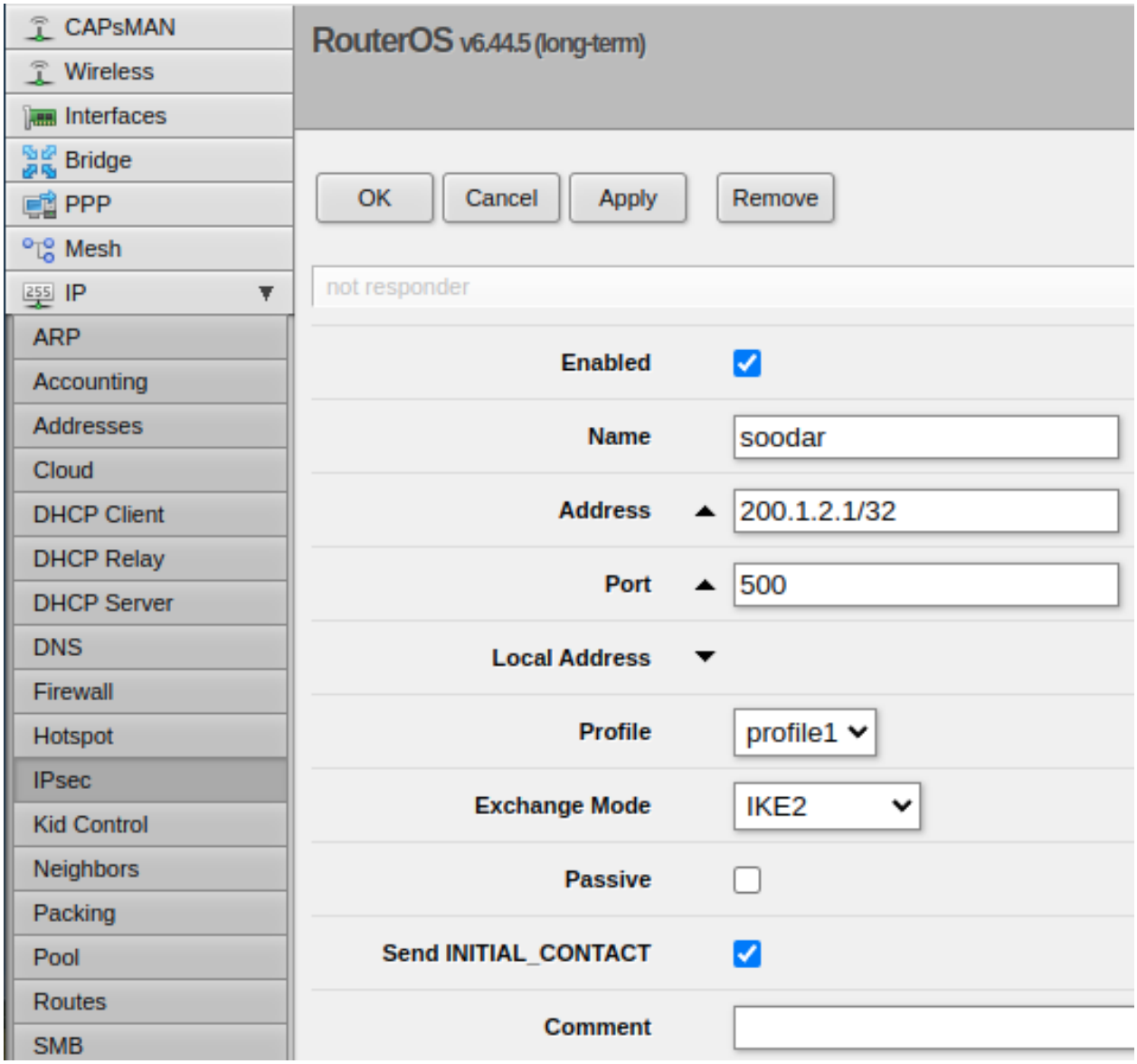
IKE پروتکلی است برای راه اندازی و تشکیل SA که جهت استفاده در تونل ipsec به کار گرفته می شود .
در میکروتیک تنظیم profile برای بخش ike استفاده می شود . این تنظیم در سودار در proposal مربوط به ike انجام می پذیرد :

1.1 تنظیم میکروتیک فاز 1

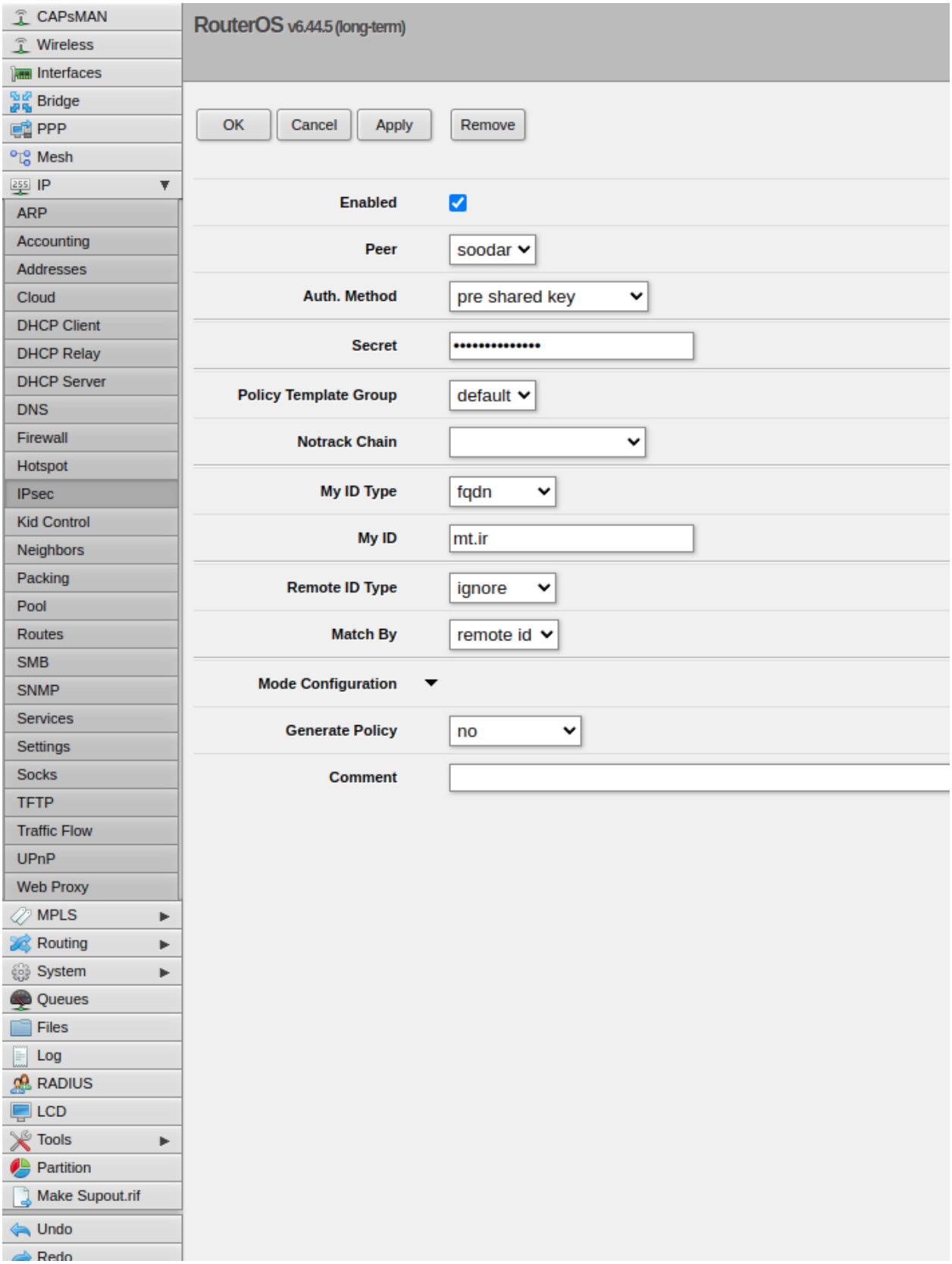
تنظیمات در منوی ip ipsec profile به شکل زیر است :



تنظیمات در بخش peer به شکل زیر انجام می گیرد :



و آخرین مرحله در فاز 1 در روتر میکروتیک، تنظیم identity است :



1.2 تنظیم سودار فاز 1

تنظیمات ipsec در سودار از لحاظ شکل دستورات شبیه cisco است .
باید تنظیمات ike به شکل زیر در سودار انجام گیرد و keyring , proposal و profile مربوط به ike اضافه شود :

```
crypto ikev2 proposal IKE_PROPOSAL
 integrity sha-256
 encryption aes-192
 group 14
```

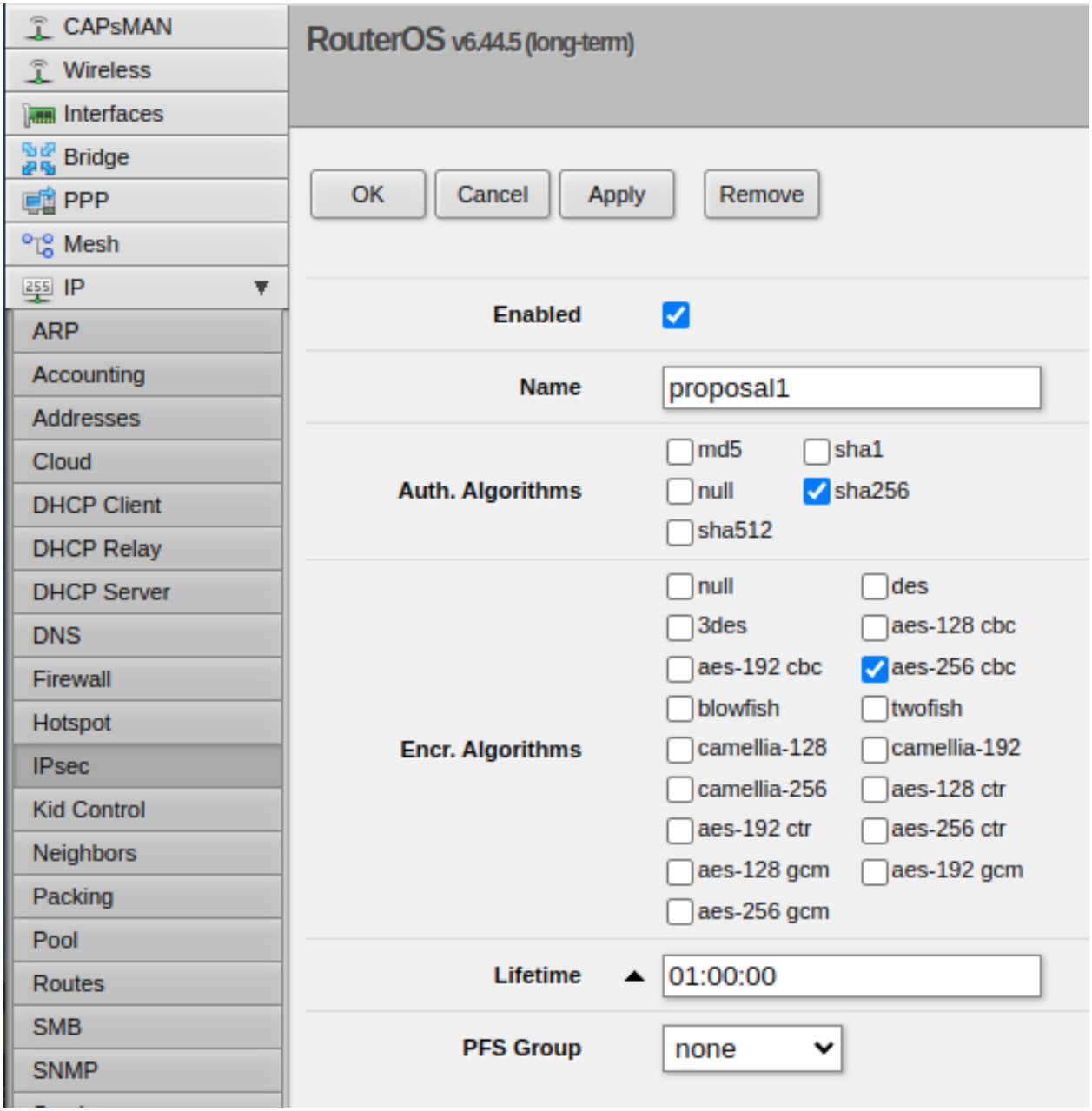
```
crypto ikev2 profile IKE_PROFILE
 keyring local IKE_KEYRING
 identity local fqdn soodar.ir
 lifetime 120
 match identity remote fqdn mt.ir
 authentication local pre-share
 authentication remote pre-share
 proposal IKE_PROPOSAL
```

```
crypto ikev2 keyring IKE_KEYRING
 peer mt
  address 200.1.2.2
  pre-shared-key salam@@!!SALAM
  identity fqdn mt.ir
```

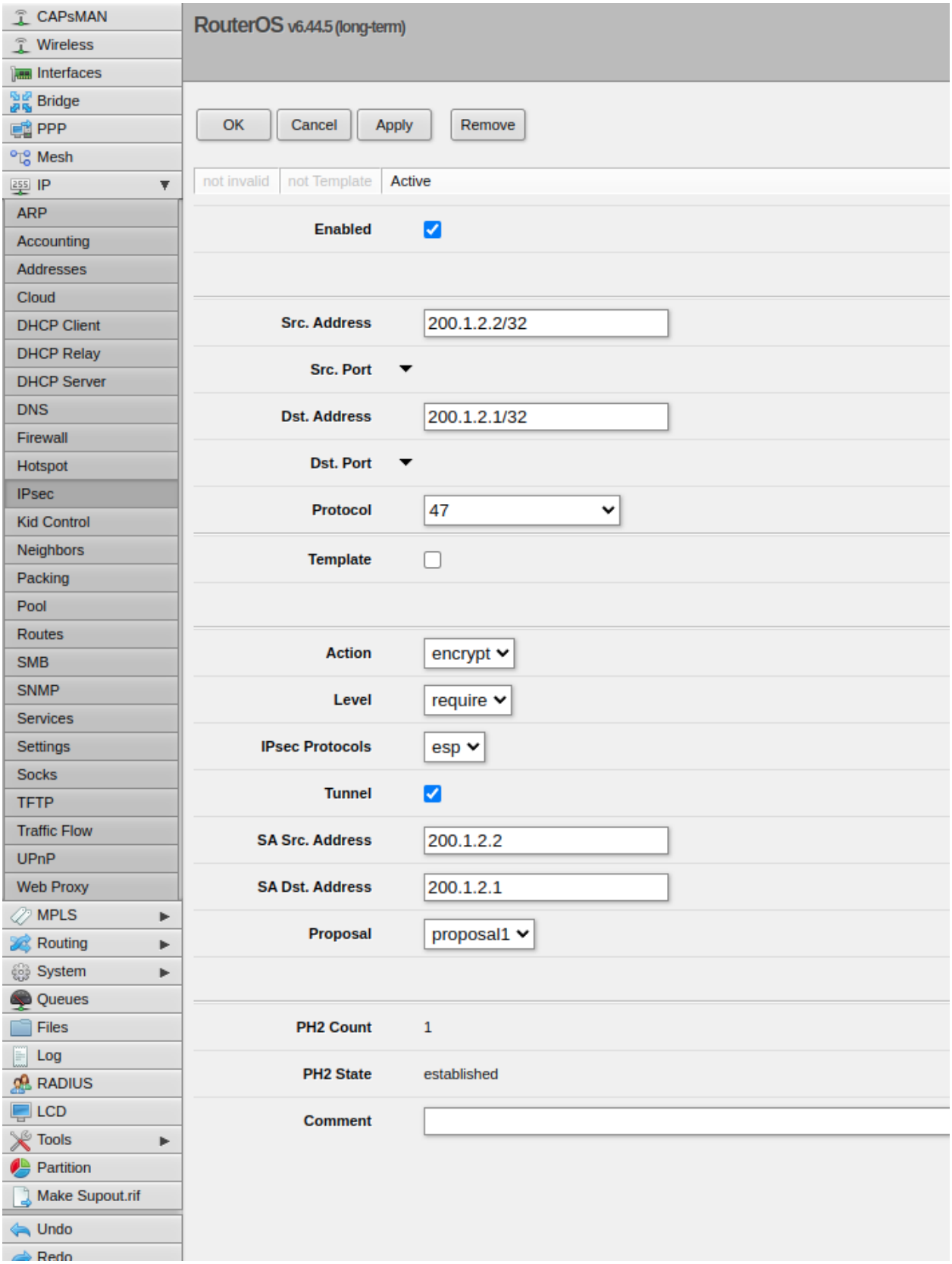
3.5.23 2.نحوه تنظیم **ipsec** (فاز دوم)

2.1 تنظیم میکروتیک فاز 2

در این مرحله باید یک proposal تعریف کنیم :



و همچنین یک policy به شکل زیر اضافه می گردد :



2.2 تنظیم سودار فاز 2

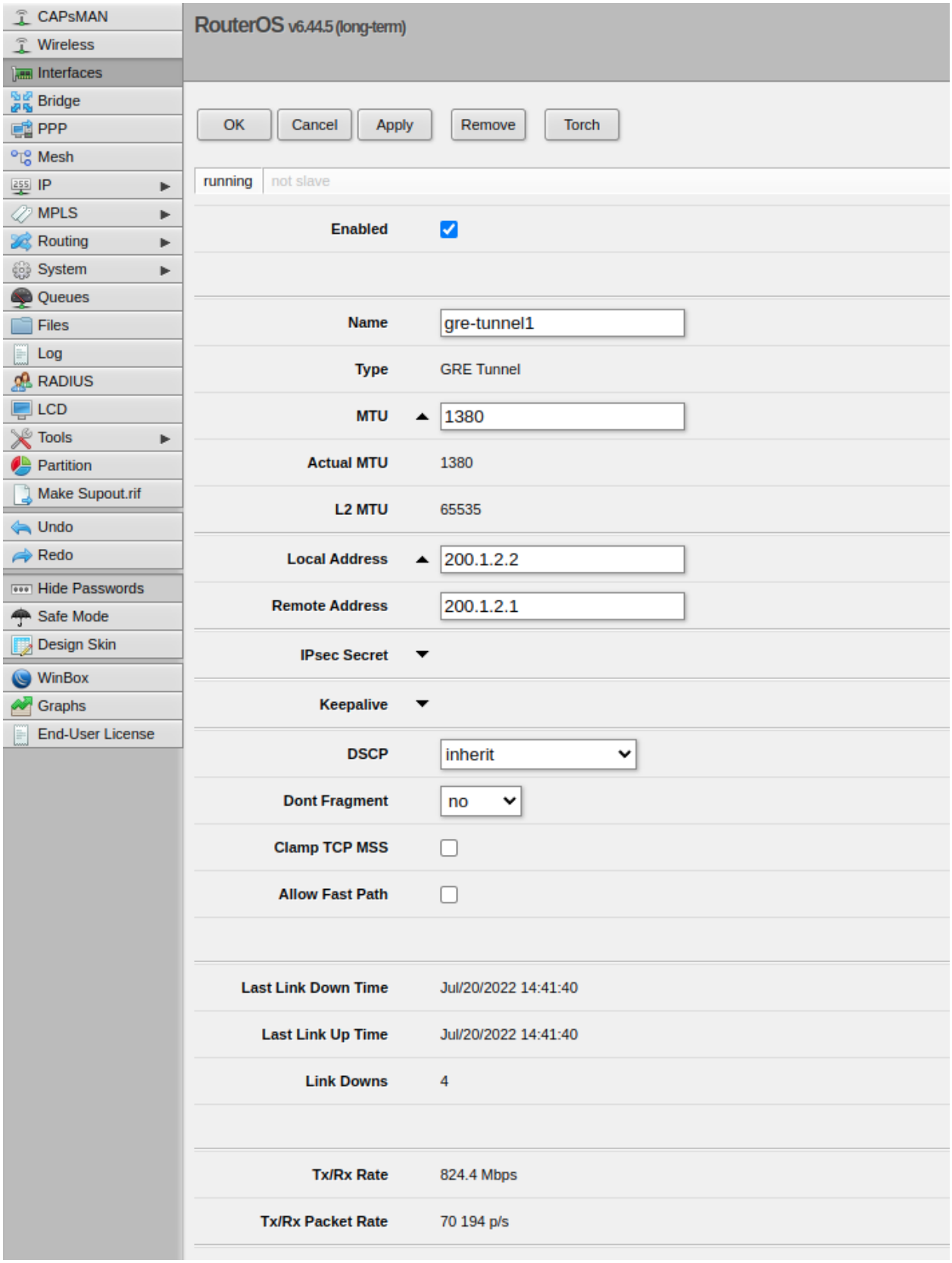
در فاز 2 باید تنظیمات ipsec انجام شود . ابتدا یک transform set برای ipsec اضافه می کنیم (بدیهی است الگوریتم های انتخاب شده باید با الگوریتم های انتخاب شده در فاز 2 میکروتیک یعنی بخش ip ipsec proposal یکسان باشد). سپس یک ipsec profile تعریف کرده و به شکل زیر در آن ike ipsec ts , profile و همچنین lifetime را در آن مشخص می کنیم :

```
crypto ipsec transform-set IPSEC_TS esp hmac sha-256 cipher aes-256
 mode transport
```

```
crypto ipsec profile IPSEC_PROFILE
 set transform-set IPSEC_TS
 set ikev2-profile IKE_PROFILE
 set security-association lifetime seconds 3600
```

4.5.23 3.ساخت تونل **gre** و محافظت آن با **ipsec**

3.1 ساخت تونل **gre** در میکروتیک
دقت شود که مقدار mtu در اینترفیس gre باید از مقدار پیش فرض به 1380 تغییر کند تا سربار افزایشی ipsec روی بسته ها هم در نظر گرفته شود :



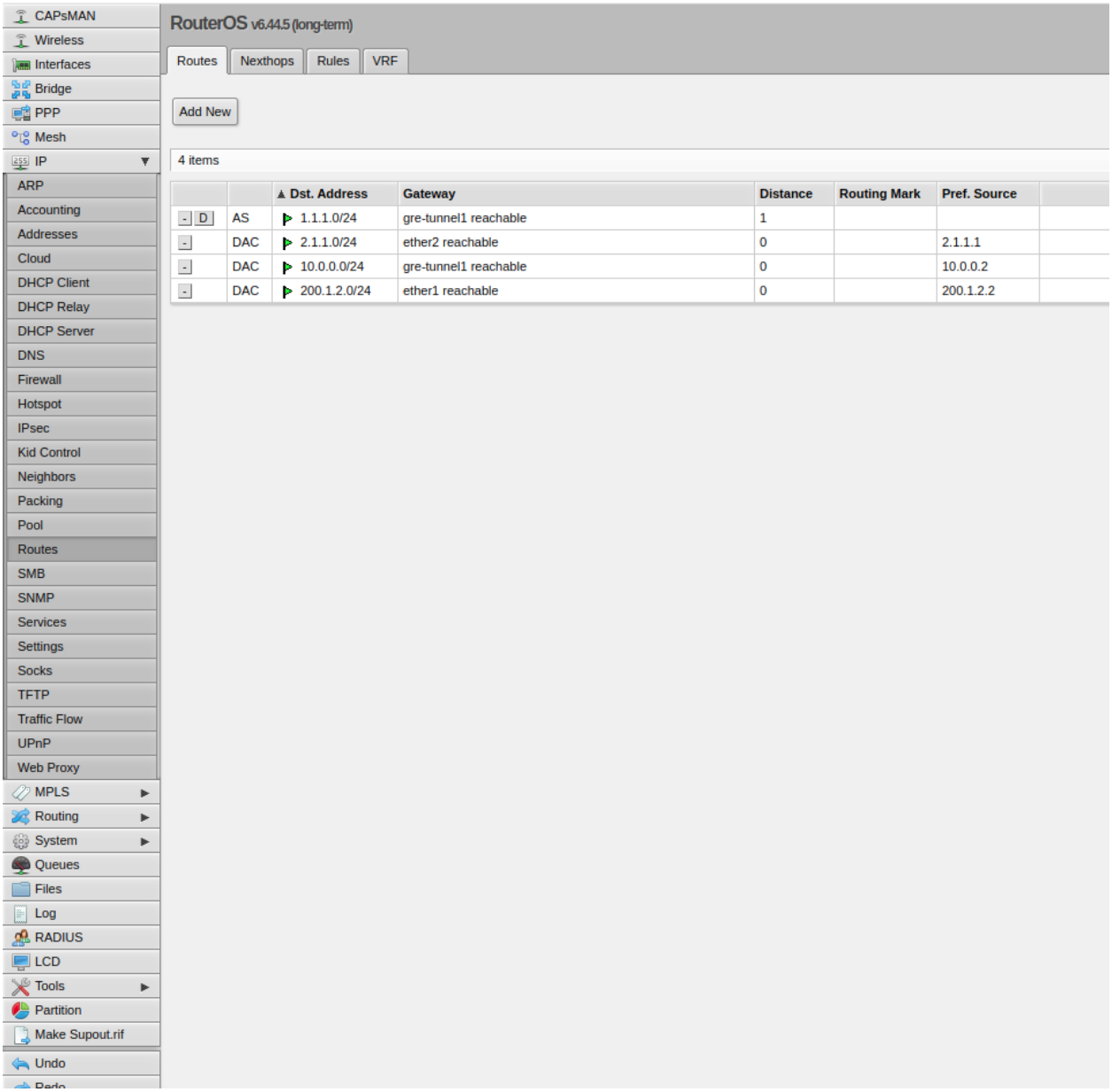
3.2 ساخت تونل **gre** در سودار

```
interface tunnel10
 ip mtu 1380
 tunnel source 200.1.2.1
 tunnel destination 200.1.2.2
 tunnel protection ipsec profile IPSEC_PROFILE
 ip address 10.0.0.1/24
```

5.5.23.4 اضافه کردن **static route**

در آخرین مرحله باید یک route اضافه کنیم تا دسترسی بین pc2 , pc1 از طریق تونل ipsec برقرار شود . برای این کار باید در میکروتیک route شبکه 1.1.1.0/24 و در روتر سودار route شبکه 2.1.1.0/24 را از طریق تونلی که ایجاد کرده ایم تعریف نماییم :

1.4 **static route** در میکروتیک



2.4 **static route** در سودار

```
ip route 2.1.1.0/24 tunnel10
```

6.23 تونل **wireguard** بین سودار و میکروتیک

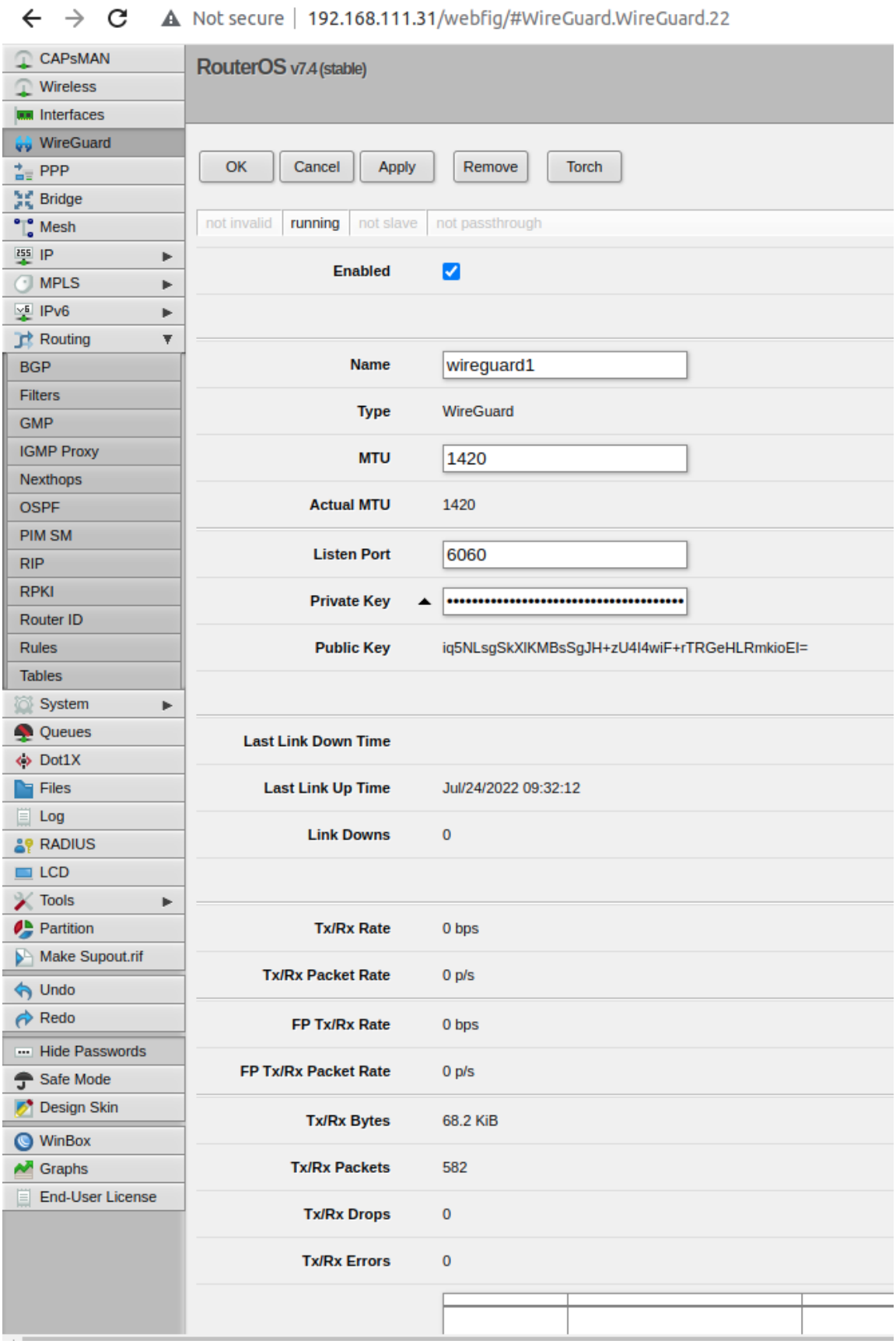
1.6.23 نحوه تنظیم تونل **wireguard** در میکروتیک و سودار

تنظیم تونل wireguard شامل دو بخش است . که ابتدا یک اینترفیس wireguard ساخته و تنظیماتی مثل listen port و کلیدی که برای تونل استفاده می شود را انجام می دهیم و سپس اطلاعات peer هایی که باید به آن ها وصل شویم را اضافه می کنیم که این اطلاعات شامل آدرس ip و شماره پورتی که طرف مقابل در آن گوش می کند و همچنین public key طرف مقابل می شود .
در ادامه تنظیمات را مطابق شکل فوق در Mikrotik-1 انجام می دهیم و دو تونل wireguard و دو peer (soodar و MikroTik-2) را در آن اضافه می کنیم . سپس در انتها نحوه تنظیم تونل wireguard در روتر سودار را شرح می دهیم .
فرض کنید سناریوی طبق شکل فوق داریم که در آن یک روتر سودار و دو روتر میکروتیک داریم که قرار است یک شبکه fullmesh از تونل های wireguard بین روتر ها داشته باشیم یعنی همه روتر ها به یکدیگر تونل داشته باشند :

2.6.23 1.نحوه تنظیم تونل **wireguard** در میکروتیک

1.1 اضافه کردن اینترفیس **wireguard**

ابتدا یک اینترفیس wireguard اضافه می کنیم و در آن تنظیمات مربوطه را انجام می دهیم ما در اینجا فقط listen port را مشخص می کنیم و بقیه تنظیمات پیش فرض استفاده می شود :

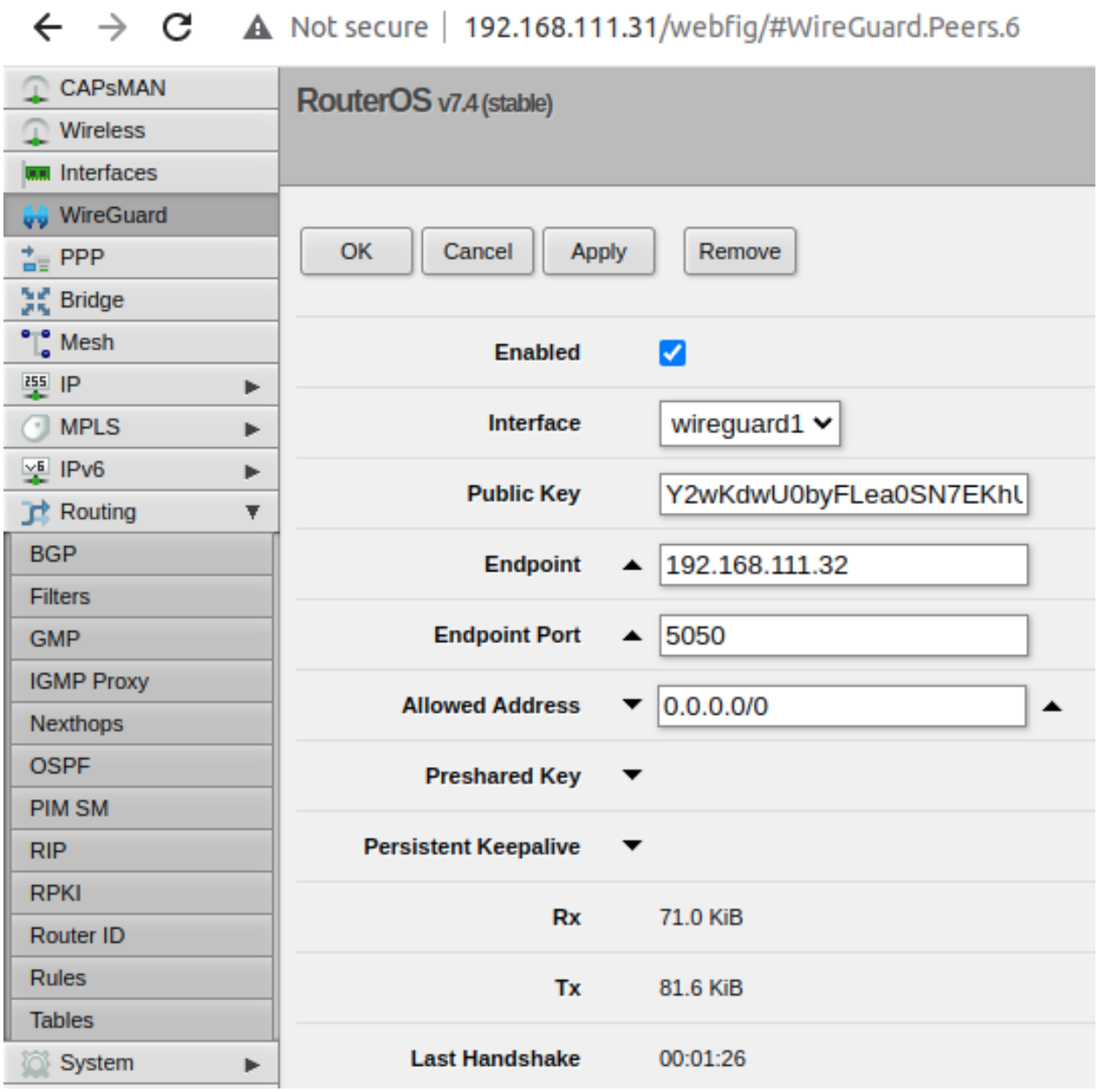

Not secure | 192.168.111.31/webfig/#WireGuard.WireGuard.22
RouterOS v7.4 (stable)
CAPsMAN
Wireless
Interfaces
WireGuard
PPP
Bridge
Mesh
IP
MPLS
IPv6
Routing
BGP
Filters
GMP
IGMP Proxy
Nexthops
OSPF
PIM SM
RIP
RPKI
Router ID
Rules
Tables
System
Queues
Dot1X
Files
Log
RADIUS
LCD
Tools
Partition
Make Supout.rif
Undo
Redo
Hide Passwords
Safe Mode
Design Skin
WinBox
Graphs
End-User License
OK
Cancel
Apply
Remove
Torch
not invalid
running
not slave
not passthrough
Enabled
Name
wireguard1
Type
WireGuard
MTU
1420
Actual MTU
1420
Listen Port
6060
Private Key
Public Key
iq5NLsgSkXlKMBsSgJH+zU4l4wiF+rTRGeHLRmkioEI=
Last Link Down Time
Last Link Up Time
Jul/24/2022 09:32:12
Link Downs
0
Tx/Rx Rate
0 bps
Tx/Rx Packet Rate
0 p/s
FP Tx/Rx Rate
0 bps
FP Tx/Rx Packet Rate
0 p/s
Tx/Rx Bytes
68.2 KiB
Tx/Rx Packets
582
Tx/Rx Drops
0
Tx/Rx Errors
0

1.2 اضافه کردن **peer**

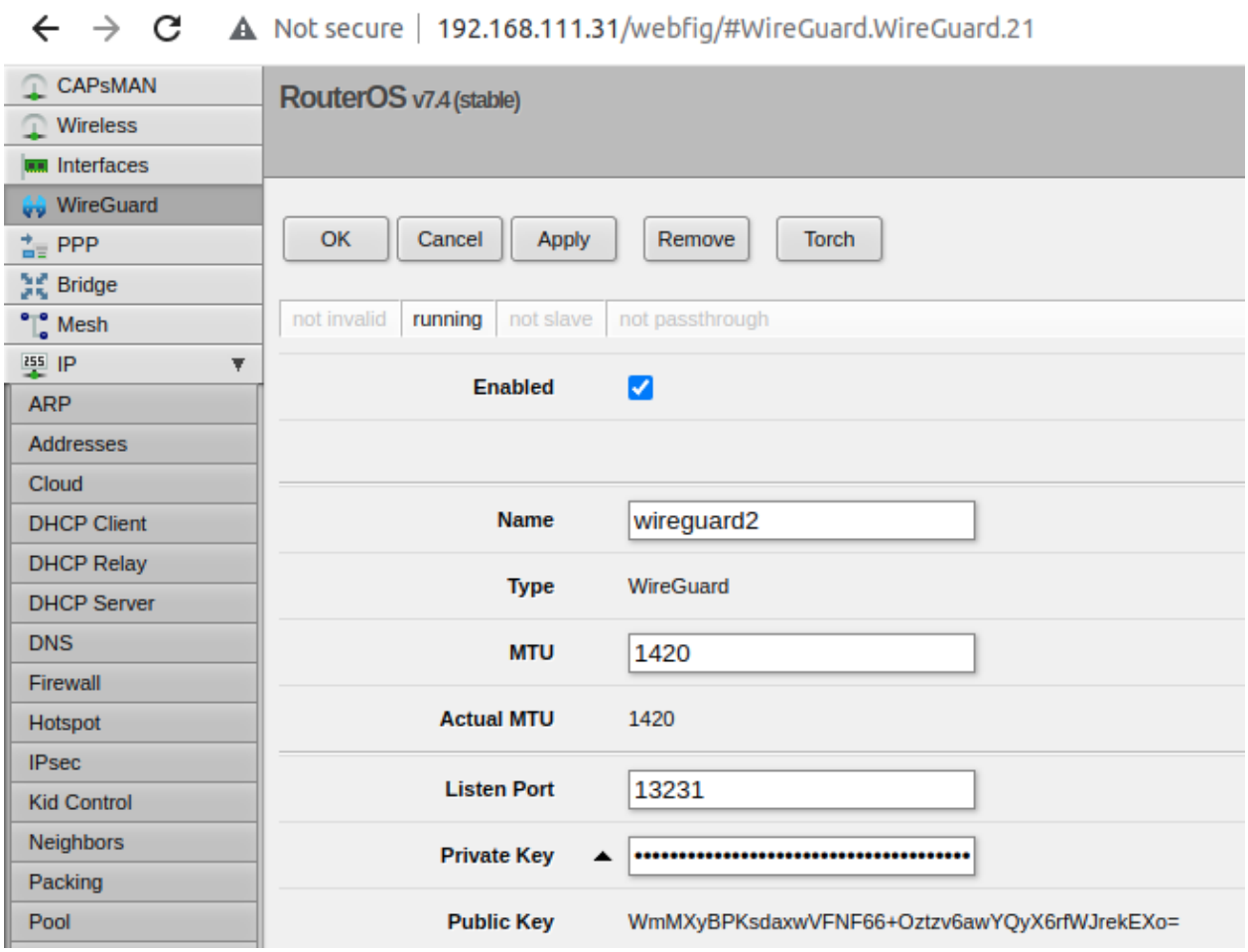
حال peer ها را اضافه می کنیم و اطلاعات مربوط به آنها را وارد می کنیم :

1. مشخص می کنیم که این peer مربوط به کدام اینترفیس است
2. public key طرف مقابل را وارد می کنیم
3. در بخش endpoint آدرس ip سمت public(شبکه عمومی) یا اینترنت را وارد می کنیم.
4. در endpoint port شماره portی که طرف مقابل در آن گوش می کند را وارد می کنیم .
5. allowed ip را هم 0.0.0.0/0 تعیین می کنیم تا اجازه عبور تمامی شبکه ها را داشته باشد .

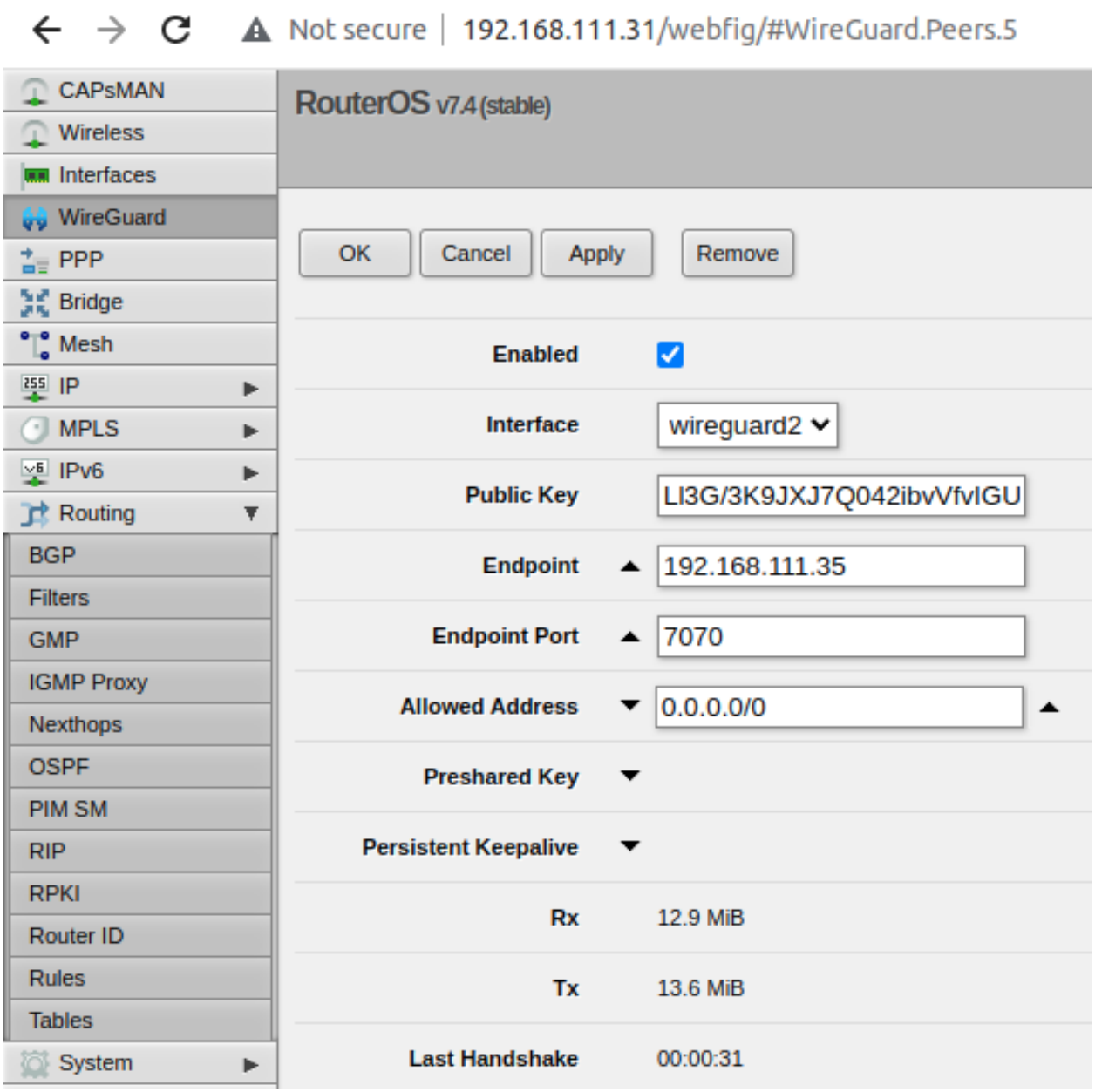
تنظیمات برای **peer MikroTik2** به شکل زیر خواهد بود:



به همین شکل اینترفیس wireguard و peer بعدی را هم اضافه می کنیم :

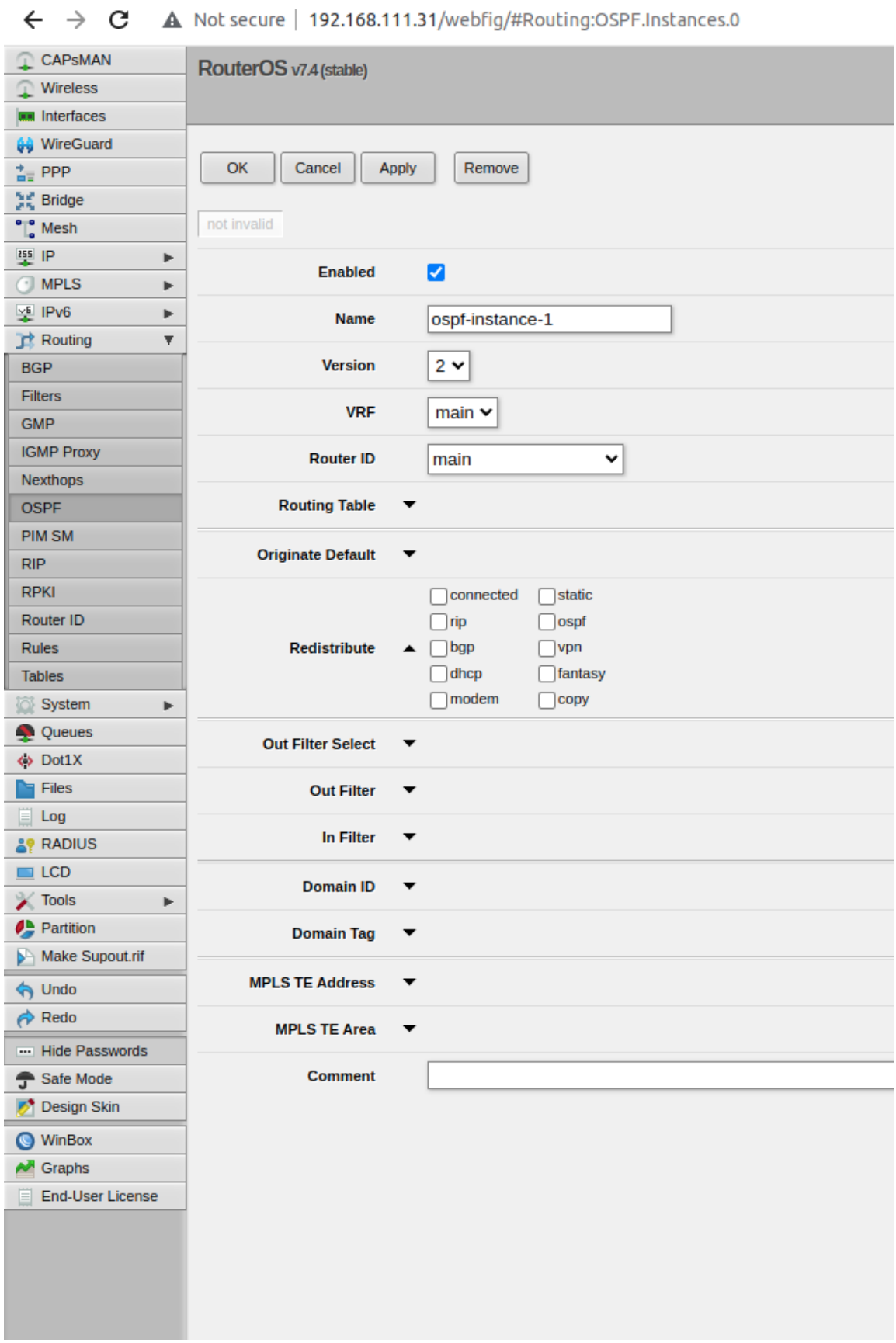


تنظیمات برای **peer soodar** به شکل زیر خواهد بود:

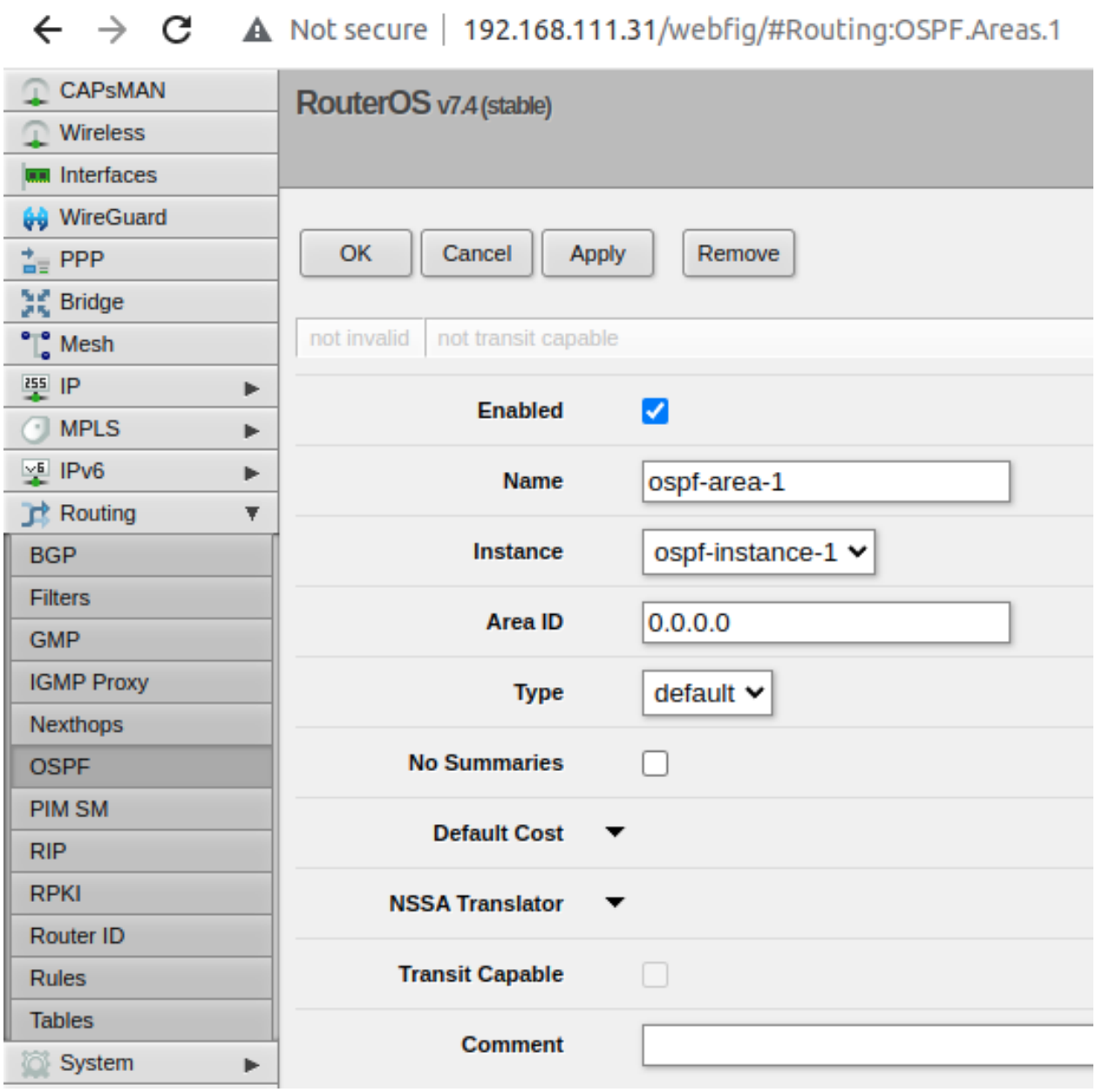


1.3 فعال کردن **ospf** در تونل **wireguard**

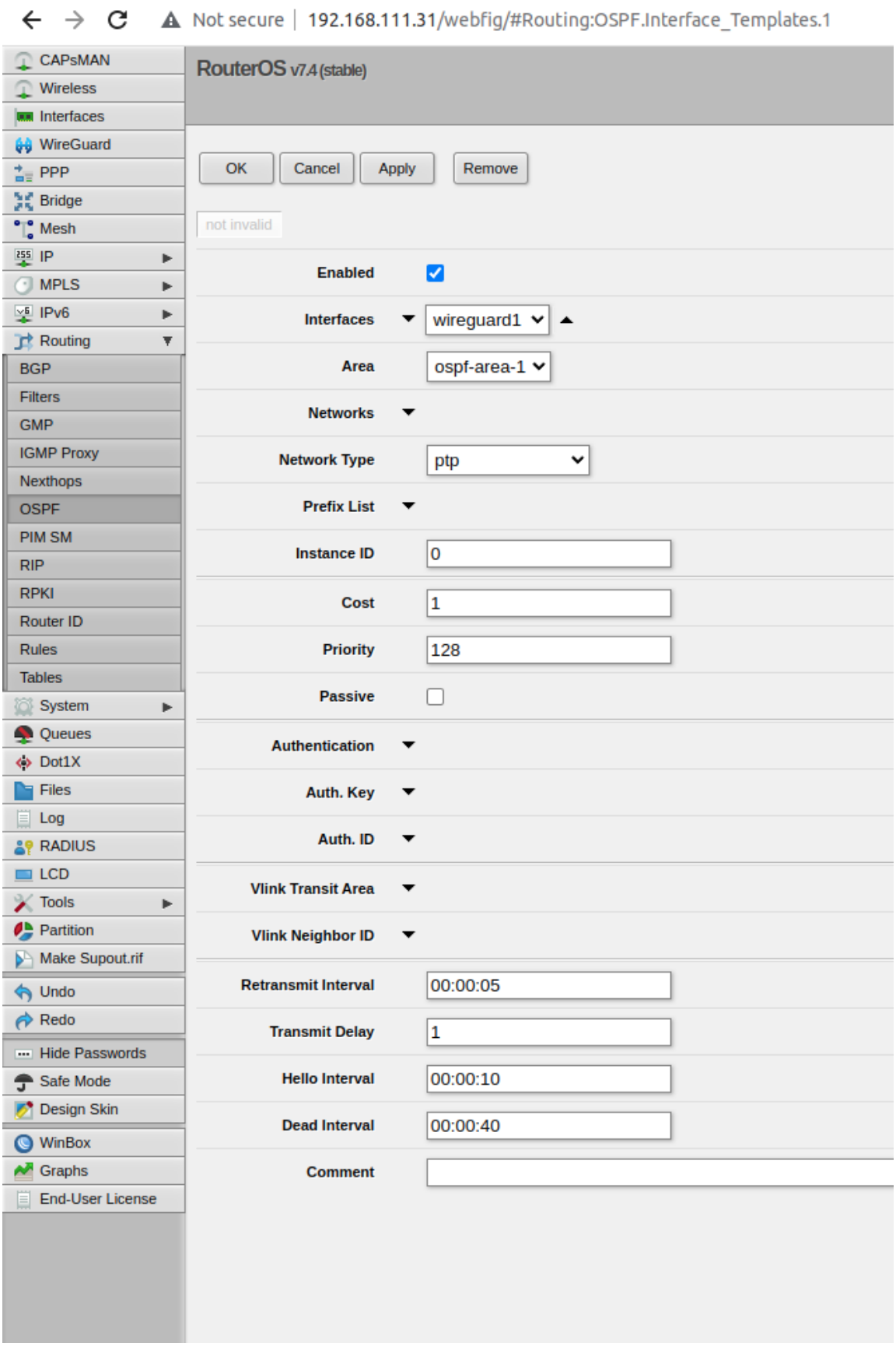
برای راه اندازی ospf باید یک instance از ospf بسازیم :


Not secure | 192.168.111.31/webfig/#Routing:OSPF.Instances.0
CAPsMAN
Wireless
Interfaces
WireGuard
PPP
Bridge
Mesh
IP
MPLS
IPv6
Routing
BGP
Filters
GMP
IGMP Proxy
Nexthops
OSPF
PIM SM
RIP
RPKI
Router ID
Rules
Tables
System
Queues
Dot1X
Files
Log
RADIUS
LCD
Tools
Partition
Make Supout.rif
Undo
Redo
Hide Passwords
Safe Mode
Design Skin
WinBox
Graphs
End-User License
RouterOS v7.4 (stable)
OK
Cancel
Apply
Remove
not invalid
Enabled
Name
ospf-instance-1
Version
2
VRF
main
Router ID
main
Routing Table
Originate Default
Redistribute
connected
static
rip
ospf
bgp
vpn
dhcp
fantasy
modem
copy
Out Filter Select
Out Filter
In Filter
Domain ID
Domain Tag
MPLS TE Address
MPLS TE Area
Comment

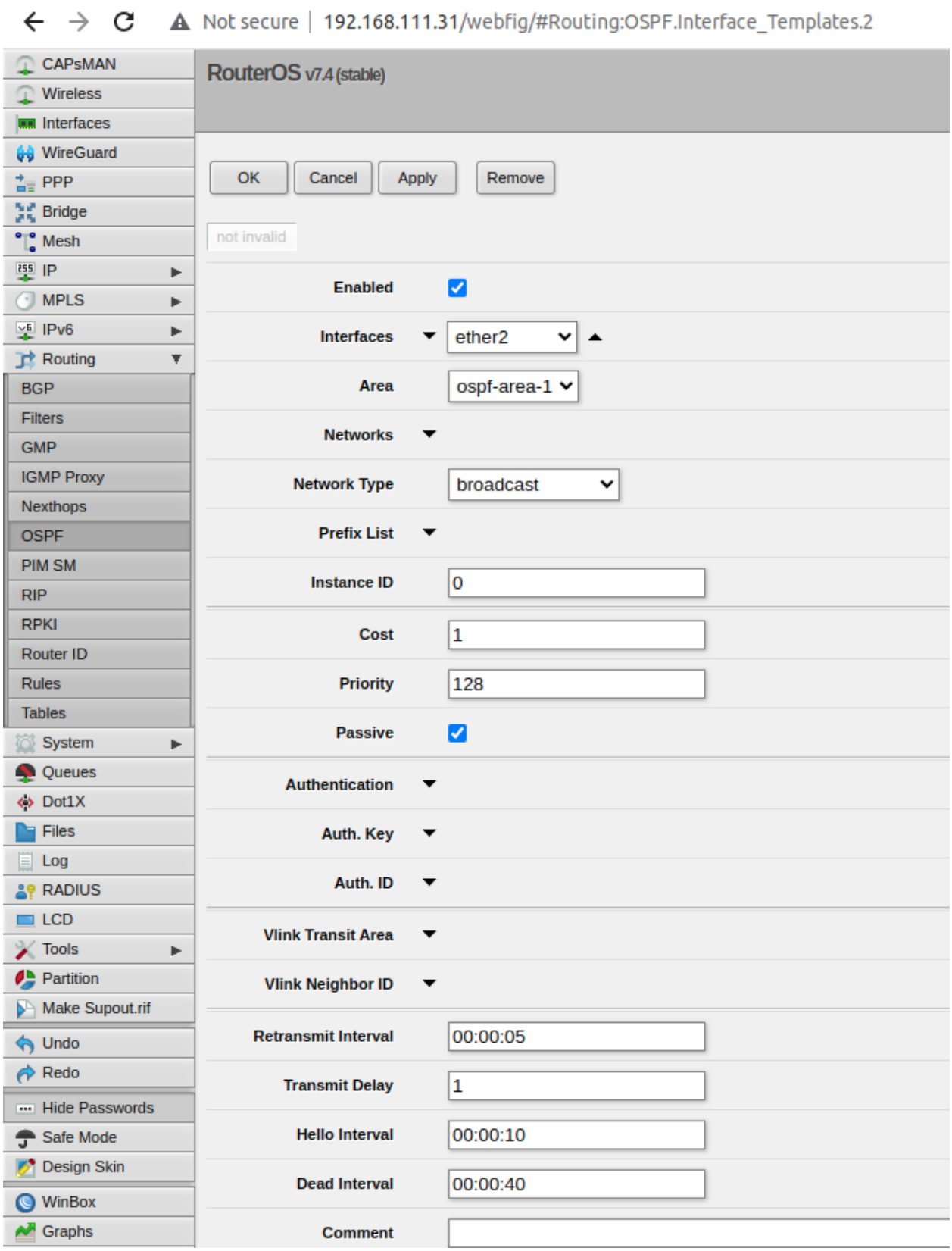
یک area هم اضافه می کنیم :



سپس اینترفیس هایی که باید در آن ospf فعال شود (اینترفیس wireguard) را در بخش interface template اضافه می کنیم :


Not secure | 192.168.111.31/webfig/#Routing:OSPF.Interface_Templates.1
CAPsMAN
Wireless
Interfaces
WireGuard
PPP
Bridge
Mesh
IP
MPLS
IPv6
Routing
BGP
Filters
GMP
IGMP Proxy
Nexthops
OSPF
PIM SM
RIP
RPKI
Router ID
Rules
Tables
System
Queues
Dot1X
Files
Log
RADIUS
LCD
Tools
Partition
Make Supout.rif
Undo
Redo
Hide Passwords
Safe Mode
Design Skin
WinBox
Graphs
End-User License
RouterOS v7.4 (stable)
OK
Cancel
Apply
Remove
not invalid
Enabled
Interfaces
wireguard1
Area
ospf-area-1
Networks
Network Type
ptp
Prefix List
Instance ID
0
Cost
1
Priority
128
Passive
Authentication
Auth. Key
Auth. ID
Vlink Transit Area
Vlink Neighbor ID
Retransmit Interval
00:00:05
Transmit Delay
1
Hello Interval
00:00:10
Dead Interval
00:00:40
Comment

همچنین اینترفیس هایی که باید آدرس آنها توسط ospf توزیع (distribute) شود را هم اضافه می کنیم با این تفاوت که گزینه passive را در آن ها فعال می کنیم تا بسته های ospf در آن ارسال نشود و فقط آدرس آن توسط ospf تبلیغ شود :


Not secure | 192.168.111.31/webfig/#Routing:OSPF.Interface_Templates.2
CAPsMAN
Wireless
Interfaces
WireGuard
PPP
Bridge
Mesh
IP
MPLS
IPv6
Routing
BGP
Filters
GMP
IGMP Proxy
Nexthops
OSPF
PIM SM
RIP
RPKI
Router ID
Rules
Tables
System
Queues
Dot1X
Files
Log
RADIUS
LCD
Tools
Partition
Make Supout.rif
Undo
Redo
Hide Passwords
Safe Mode
Design Skin
WinBox
Graphs
RouterOS v7.4 (stable)
OK
Cancel
Apply
Remove
not invalid
Enabled
Interfaces
ether2
Area
ospf-area-1
Networks
Network Type
broadcast
Prefix List
Instance ID
0
Cost
1
Priority
128
Passive
Authentication
Auth. Key
Auth. ID
Vlink Transit Area
Vlink Neighbor ID
Retransmit Interval
00:00:05
Transmit Delay
1
Hello Interval
00:00:10
Dead Interval
00:00:40
Comment

3.6.23 2.نحوه تنظیم **wireguard** در روتر سودار

در شکل زیر تنظیمات انجام شده برای روتر سودار را مشاهده می کنید که در آن یک اینترفیس wireguard10 اضافه شده است و سپس peer های مربوطه هم تنظیم شده اند :

ابتدا باید یک کلید برای تونل wireguard با دستور زیر در روتر بسازیم :

```
soodar(config)# crypto key generate x25519 label wgkey
```

سپس تنظیمات را با در نظر گرفتن توضیحات زیر انجام می دهیم :

1. مقدار mtu باید **1420** تنظیم شود
2. دقت شود wireguard mode در حالت **routing** انتخاب شود
3. تونل source و همچنین کلید تونل را مشخص می کنیم
4. portی که این اینترفیس در آن گوش می کند مشخص می کنیم که طبق سناریو 7070 تنظیم می شود
5. در ادامه peer ها تعریف می کنیم و در هر یک public-key,endpoint و allowed-ip را مشخص می کنیم. چون در سودار **allowed-ip** در جدول routing اضافه میشوند در هر peer باید فقط تک ip طرف مقابل با **prefix /32** به عنوان allowed-ip اضافه گردد .
6. دقت شود که تنظیم ospf در اینترفیس وایرگارد به شکل **point-to-multipoint** انجام شود
7. با دستور ip ospf area 0 ما ospf را در اینترفیس فعال می کنیم
8. در نهایت هم یک ip به اینترفیس اختصاص می دهیم

```
interface wireguard10
  ip mtu 1420
  wireguard mode routing
  wireguard source 192.168.111.35
  wireguard private-key wgkey
  wireguard port 7070
  wireguard peer mikrotik
    public-key 5A6317C813CAB1D6B1C1514D17AEBE3B3B73BFA6B0610C97EAB7D626B7A4117A
    endpoint 192.168.111.31 port 13231
    allowed-ip 10.0.0.22/32      # ip of wg tunnel in mikrotik
  wireguard peer mikrotik2
    public-key 9C786196656518AC0876ACB749C86FAD5A7CACF635E2B212D2A9B4434450BB3C
    endpoint 192.168.111.32 port 13231
    allowed-ip 10.0.0.33/32      # ip of wg tunnel in mikrotik2
  no shutdown
  ip address 10.0.0.1/32  # ip of local wg tunnel(wireguard10)
  ip ospf network point-to-multipoint
  ip ospf area 0
exit
!
interface ge0
 no shutdown
 ip address 192.168.111.35/24
exit
!
interface ge1
 no shutdown
 ip address 1.1.1.1/24
 ip ospf area 0
 ip ospf passive
exit
!
router ospf
exit
!
```

چون در سودار فرمت پیش فرض کلید ها hex می باشد در running-config فقط مقدار hex نمایش داده می شود . البته شما در نمایش کلید (show crypto key <wgkey>) و همچنین در نمایش وضعیت تونل های وایرگارد (show wireguard) می توانید فرمت base64 آن را هم مشاهده نمایید.

```
soodar# sh crypto key wgkey
Keypair Label: wgkey
  Algorithm:  X25519
  Public key:  2E5DC6FF72BD25727B434E3689BBD57EF20652E1817E120543CFE92DF536C519
  Public key base64:  Ll3G/3K9JXJ7Q042ibvVfvIGUuGBfhIFQ8/pLfU2xRk=

soodar#
```

```
soodar# sh wireguard
Wireguard 10
  Mode: Routing
  Source: 192.168.111.35
  Key: wgkey
  Public key: 2E5DC6FF72BD25727B434E3689BBD57EF20652E1817E120543CFE92DF536C519
  Public-key Base64: Ll3G/3K9JXJ7Q042ibvVfvIGUuGBfhIFQ8/pLfU2xRk=
  Port: 7070

  Peer mikrotik:
    Public key: 5A6317C813CAB1D6B1C1514D17AEBE3B3B73BFA6B0610C97EAB7D626B7A4117A
    Public-key Base64: WmMXyBPKsdaxwVFNF66+Oztzv6awYQyX6rfWJrekEXo=
    Endpoint: 192.168.111.31
    Current Endpoint: 192.168.111.31
    Current Source: 192.168.111.35
    Persistent keepalive: 10
    Port: 13231
    VRF: default
    Connected: True
    Allowed IPs:
     - 10.0.0.22/32

  Peer mikrotik2:
    Public key: 9C786196656518AC0876ACB749C86FAD5A7CACF635E2B212D2A9B4434450BB3C
    Public-key Base64: nHhhlmVlGKwIdqy3SchvrVp8rPY14rIS0qm0Q0RQuzw=
    Endpoint: 192.168.111.32
    Current Endpoint: 192.168.111.32
    Current Source: 192.168.111.35
    Persistent keepalive: 10
    Port: 13231
    VRF: default
    Connected: True
    Allowed IPs:
     - 10.0.0.33/32

soodar#
```

Connected: True نشان می دهد که تونل به peer مربوطه وصل است. که در اینجا هر دو تونل به میکروتیک ها وصل هستند

بررسی **ospf neighbor** در soodar :

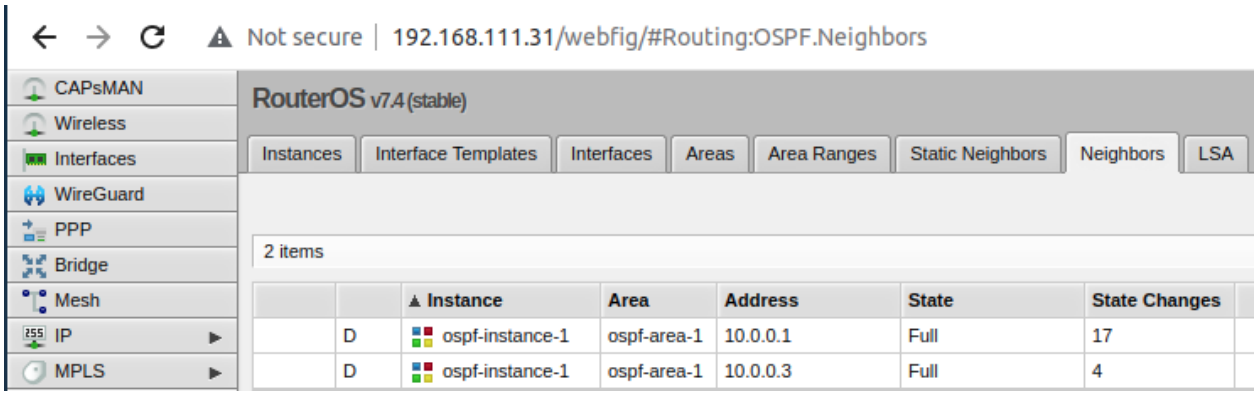
```
soodar# show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface                        RXmtL RqstL DBsmL
192.168.111.31  128 Full/DROther      33.639s 10.0.0.22       wireguard10:10.0.0.1                 0     0     0
192.168.111.32  128 Full/DROther      35.943s 10.0.0.33       wireguard10:10.0.0.1                 0     0     0
```

(continues on next page)

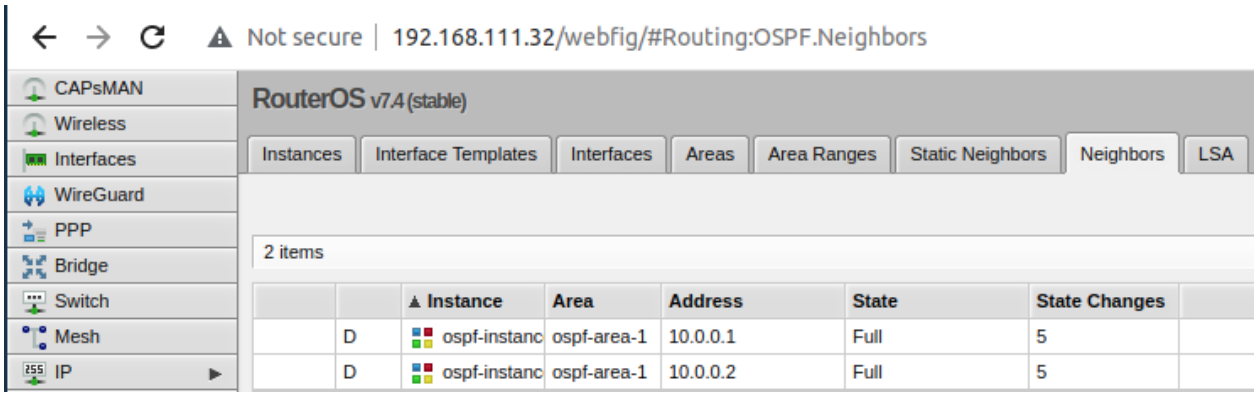
(continued from previous page)

```
soodar#
```

در MikroTik1 :



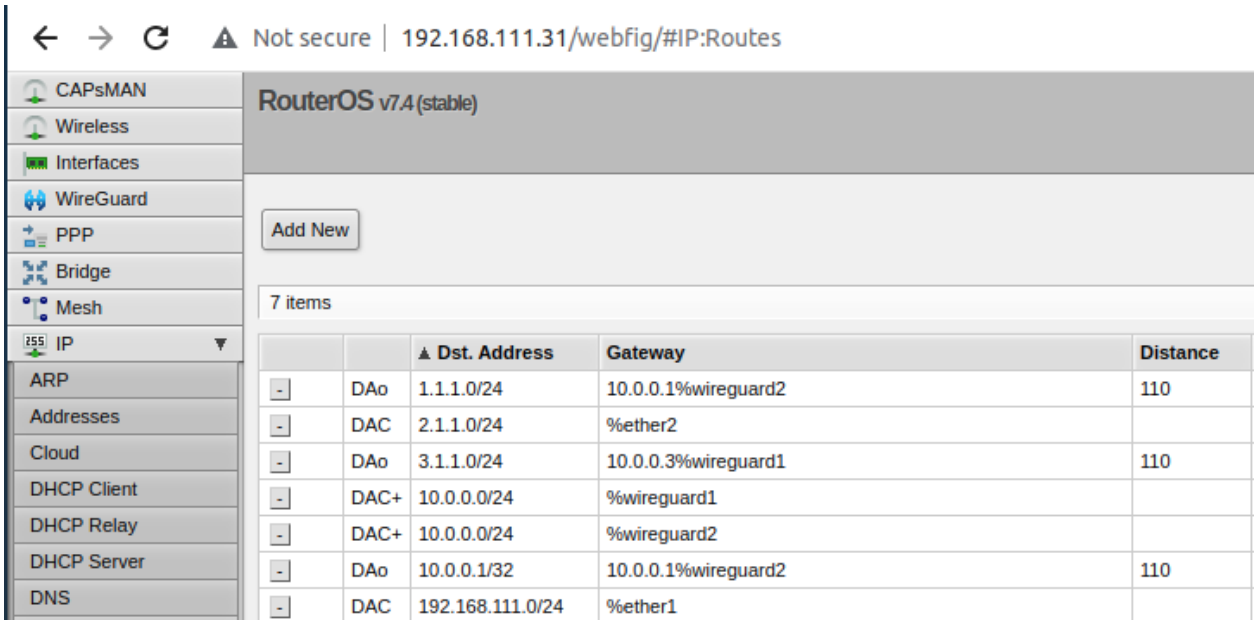
در MikroTik2 :



بررسی جدول **route** در soodar :

```
soodar# sh ip fib
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric, W - WG,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

C>* 1.1.1.0/24 is directly connected, ge1, 05:04:14
O>* 2.1.1.0/24 [110/11] via 10.0.0.22, wireguard10 onlink, weight 1, 03:09:00
O>* 3.1.1.0/24 [110/11] via 10.0.0.33, wireguard10 onlink, weight 1, 04:04:47
O>* 10.0.0.0/24 [110/11] via 10.0.0.22, wireguard10 onlink, weight 1, 03:09:00
  *                     via 10.0.0.33, wireguard10 onlink, weight 1, 03:09:00
C>* 10.0.0.1/32 is directly connected, wireguard10, 04:34:55
W>* 10.0.0.22/32 [1/0] is directly connected, wireguard10, weight 1, 05:04:14
W>* 10.0.0.33/32 [1/0] is directly connected, wireguard10, weight 1, 05:04:14
C>* 192.168.30.0/24 is directly connected, ge0, 05:04:14fnrj
C>* 192.168.111.0/24 is directly connected, ge0, 05:04:14
soodar#
```

در MikroTik1 :



در MikroTik2 :