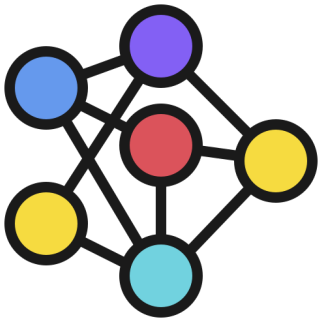
Final Paper
Computer Networks Design and Management



Szymon Budziak
Raffaele Carillo
Federica Giampetraglia
Valerio Maietta
Miglė Babickaitė
Francesco Riccardi

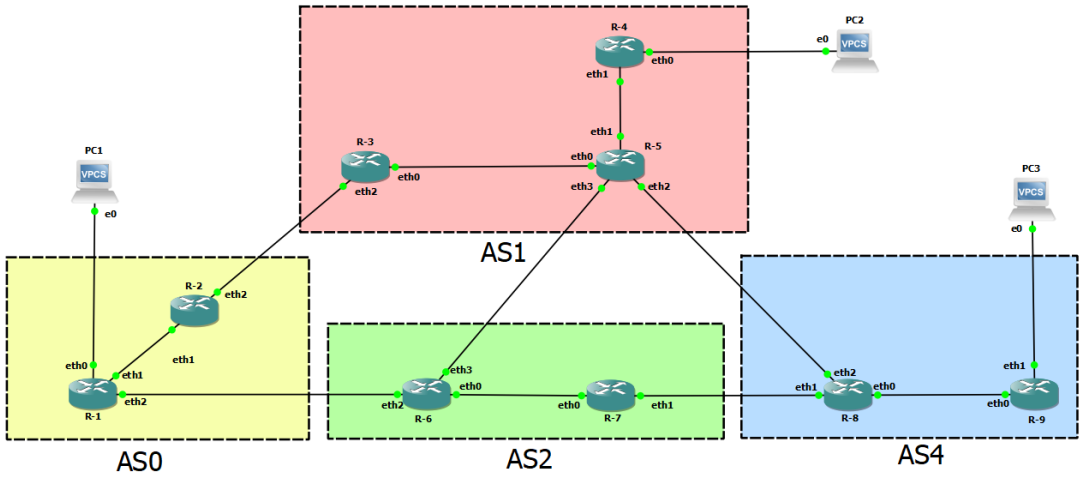
December 2022

Contents

Chapter 1

Topology and Configuration

First we build the topology using routers, end-devices (VPCS) and linking them as shown below.



1.1 IP-addresses

Device	**Interface**	IP-**address**
PC1	eth0	192.172.1.2/24
PC2	eth0	10.10.1.2/28
PC3	eth0	42.42.42.2/28
R1	eth0	192.172.1.1/24
R1	eth1	192.172.0.1/30
R1	eth2	13.10.0.1/30
R2	eth1	192.172.0.2/30
R2	eth2	13.10.1.2/30
R3	eth0	10.10.0.16/24
R3	eth2	13.10.1.1/30
R4	eth0	10.10.1.1/28
R4	eth1	10.10.1.17/24
R5	eth0	10.10.0.17/24
R5	eth1	10.10.1.16/24
R5	eth2	13.10.3.1/30
R5	eth3	13.10.2.1/30
R6	eth0	20.20.0.1/30
R6	eth2	13.10.0.2/30
R6	eth3	13.10.2.2/30
R7	eth0	20.20.0.2/30
R7	eth1	13.10.4.1/30
R8	eth0	42.42.42.16/24
R8	eth1	13.10.4.2/30
R8	eth2	13.10.3.2/30
R9	eth0	42.42.42.17/24
R9	eth1	42.42.42.1/28

1.2 Implementation using GNS3

After creating the topology and linking the devices, we proceed giving IP-addresses first to the end devices and then to the routers.

1.2.1 End-devices

First we turn on the device (ex: PC1) then we open the console:

– Code to assign the IP-address and the gateway

```
> ip 192.172.1.2/24 192.172.1.1/24
> save
```

In this case we'll have 192.172.1.2/24 as IP address and 192.172.1.1/24 as gateway (R-1). We proceed assigning the IP to the end-devices.

1.2.2 Routers

First we turn on the device and then open the auxiliary console then we assign the IP addresses to the interfaces (es: R-1).

```
> interface eth0
> ip address 192.172.1.1/24
> exit

> interface eth1
> ip address 192.172.0.1/30
> exit

> interface eth2
> ip address 13.10.0.1/30
> exit

> write
```

We proceed in the same way for all the other routers. After doing it we have to reboot the routers and check the configuration using the following command:

```
> sh run
```

1.2.3 Enabling OSPF and BGP

After setting all the topology we continue with OSPF and BGP configuration. In order to do that first we have to enable the daemons in all the routers:

```
> vi etc/frr/daemons
- press "i" to enter in the wrting mode in vim
- use the arrows to find ospfd bgpd and write "yes" where is written "no"
- press "esc"
> :wq
(this command save the chenges and quite)
- reboot the router
> reboot
```

Chapter 2

OSPF: Open Shortest Path First

2.1 Router configuration

Every autonomous system will have an Area 0. The internal links choose as Area 0 were:
AS0: R1-R2
AS1: R3-R5
AS2: R6-R7
AS4: R8-R9

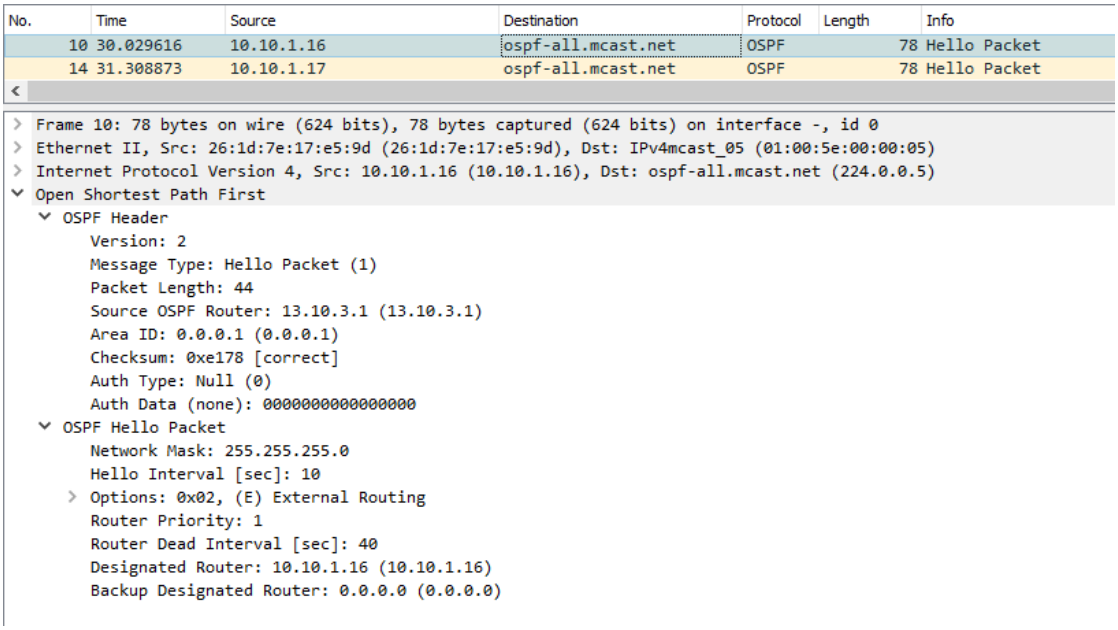
While for Area 1 we choose:
AS0: PC1-R1
AS1: PC2-R4-R5

PC3 is not linked using OSPF because it's connected using the sub-netting. For the configuration we proceed as it follows (for every router: start/open auxiliary console/vtysh/conf..)

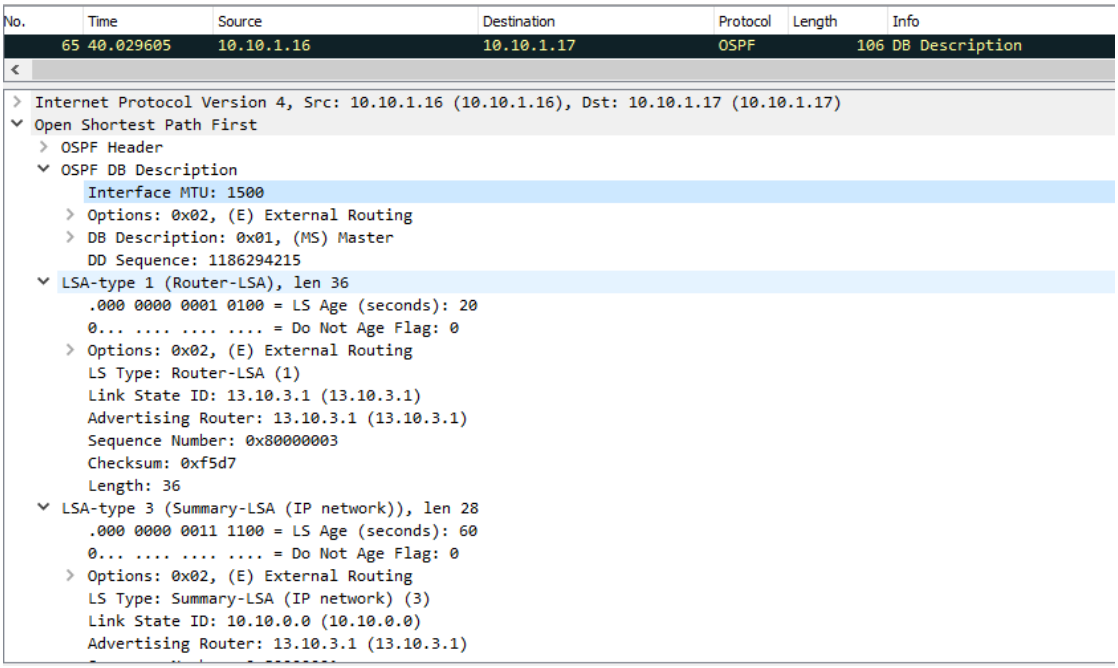
```
ROUTER1
  router ospf
  network 192.172.0.0/30 area 0
  network 192.172.1.0/24 area 1
  exit

ROUTER2
  router ospf
  network 192.172.0.0/30 area 0
  exit

ROUTER3
  router ospf
  network 10.10.0.0/24 area 0
  exit

ROUTER4
  router ospf
  network 10.10.1.0/24 area 1
  exit

ROUTER5
  router ospf
  network 10.10.0.0/24 area 0
  network 10.10.1.0/24 area 1
  exit

ROUTER6
  router ospf
  network 20.20.0.0/30 area 0
  exit

ROUTER7
  router ospf
  network 20.20.0.0/30 area 0
  exit

ROUTER8
  router ospf
  network 42.42.42.0/24 area 0
  exit

ROUTER9
  router ospf
  network 42.42.42.0/24 area 0
  exit
```

2.2 Packet analysis via Wireshark

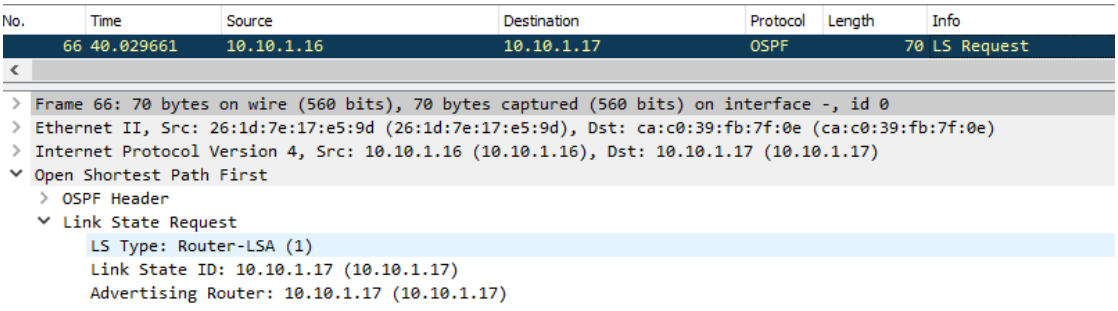
OSPF produces messages that are encapsulated in IP datagrams with port number 89. We captured the packet flow between the routers R4 and R5. The first message we have is the OSPF Hello Packet:



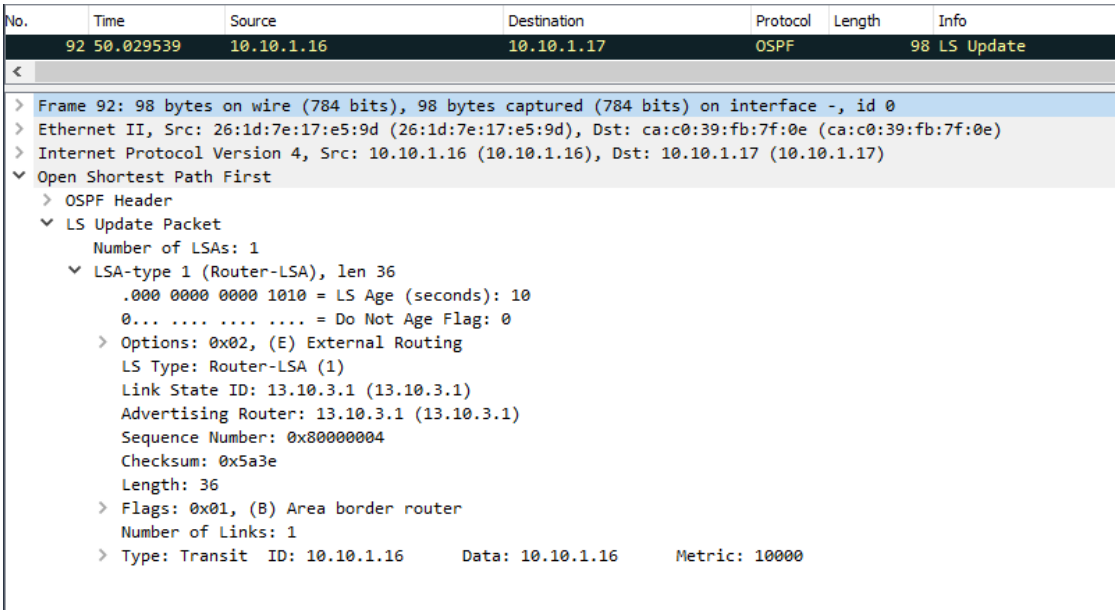
We can read the header's fields, for example the Message Type etc.. and also the OSPF Hello Packet itself.
OSPF Database Description (OSPF DD) packets are used to synchronize LSDB between routers in an adjacency relationship. DD packets provide an overview of the LSDB. The DD packet includes a list of LSA headers as an overview of the LSDB. When DD packets are exchanged between routers in an adjacency, they will know which LSAs they do not have with each other.



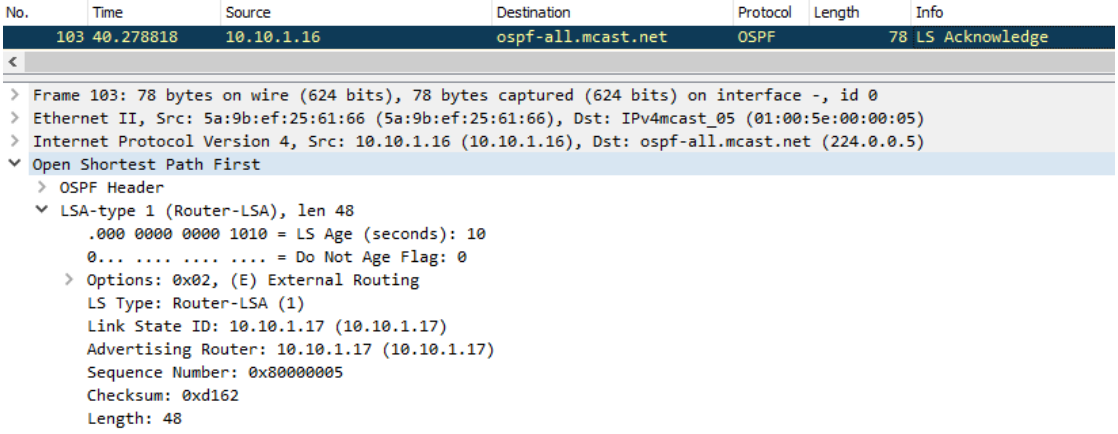
After exchanging Database Description packets with a neighboring router, a router may find that parts of its topological database are out of date. The Link State Request packet is used to request the pieces of the neighbor's database that are most up to date.



The Link-State Update (LSU) is a packet that contains fully detailed LSAs, typically sent in response to an LSR message. In our case, the R5 advertise R4 about the "link state ID: 13.10.3.1" :



Finally we have Link-State Acknowledgment (LSAck) sent to confirm receipt of an LSU message.



Chapter 3

BGP: Border Gateway Protocol

3.1 Router configuration

For our example we shall refer to R5.

```
> router bgp 101
> bgp router-id 10.10.0.1
> no bgp suppress-duplicates
```

We enable BGP protocol with specified AS number, after that we can input any BGP command. Then we specify the router ID and we disable duplicates suppression because we do not want to suppress duplicate updates if the route actually not changed.

```
> bgp bestpath as-path multipath-relax
> bgp bestpath compare-routerid
```

With these commands BGP considers paths of equal AS_PATH length for multipath computation. Also when compared routes are equal on all the metrics like AS_PATH length or IGP cost the tie is broken based on router ID.

3.1.1 Setting up IBGP

```
> neighbor AS1 peer-group
> neighbor AS1 remote-as internal
> neighbor AS1 capability extended-nexthop
> neighbor eth0 interface peer-group AS1
> neighbor eth1 interface peer-group AS1
```

We define a new peer-group called AS1 and by setting up the remote-as as internal we enable iBGP for this group. Then we allow BGP to negotiate the extended-nexthop capability with it's peers. Finally we add the interfaces eth0

and eth1 to the AS1 peer-group.

3.1.2 Setting up EBGP

```
> neighbor fabric peer-group
> neighbor fabric remote-as external
> neighbor fabric capability extended-nexthop
> neighbor eth2 interface peer-group fabric
> neighbor eth3 interface peer-group fabric
```

We do the same for eBGP but we setup the fabric peer-group as external since its interfaces are going to be connected to other AS with eBGP.

3.1.3 Advertising the networks

```
> network 10.10.0.0/24
> network 10.10.1.0/24
```

We declare the IPv4 networks that we want to advertise. In this case we are going to advertise the 10.10.0.0/24 and the 10.10.1.0/24 networks connected to R5.

```
> route-map ALLOW-ALL permit 100
> exit
> ip nht resolve-via-default
```

We configure the route map permission policy and we also allow IPv4 nexthop tracking to resolve via the default route.
The configuration for other routers is generally very similar. Parameters that are mutable are:

- BGP number while enabling this protocol.
- BGP router ID.
- Peer group definitions.
- Interfaces connections with internal and external peers.
- Networks which are declared for advertisement.

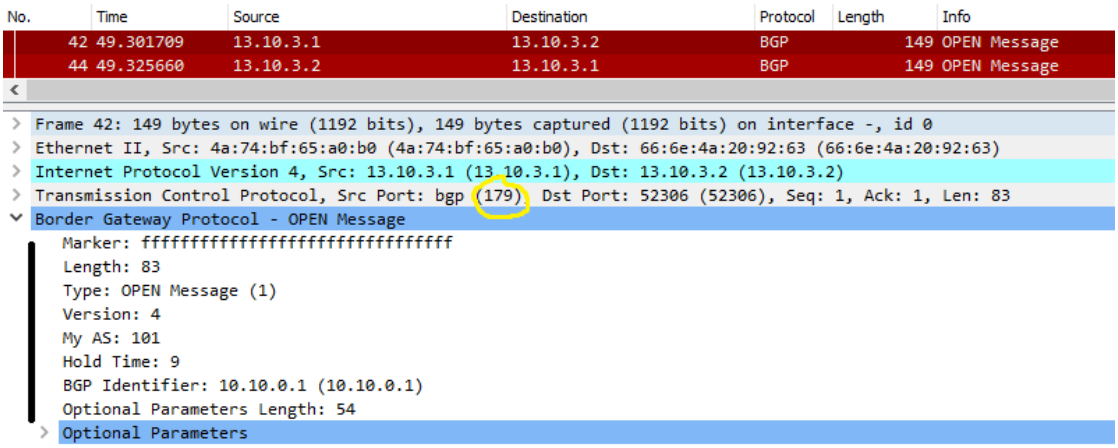
3.1.4 Route Reflector and next-hop-self

```
> neighbor AS1 route-reflector-client
> neighbor AS1 next-hop-self
```

One of the issues addressed during this project was how to advertise routes to R4. By default iBGP advertisements happen only to routers directly connected so with a maximum distance of just one hop. In order to obtain routes to AS0 from R3 a route reflector was setup on R5 so any iBGP path received by it will be advertised (reflected) to all of its peers, R4 included. Another issue was an improper annoucement of the next-hop field in eBGP routes to other AS learned with iBGP. With the next-hop-self setting we specify an announced route's nexthop as being equivalent to the address of the announcing BGP router if it is learned via eBGP

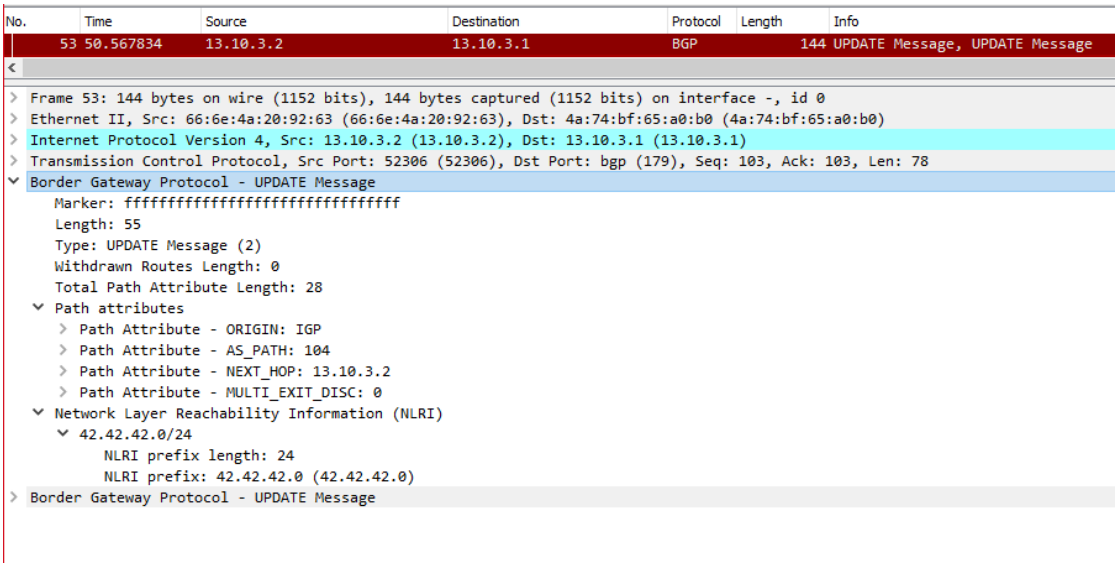
3.2 Packet analysis via Wireshark

In this section, we analyze the BGP packets between R8 and R5. First of all, we have the OPEN message, that is used to enstablish a BGP session.

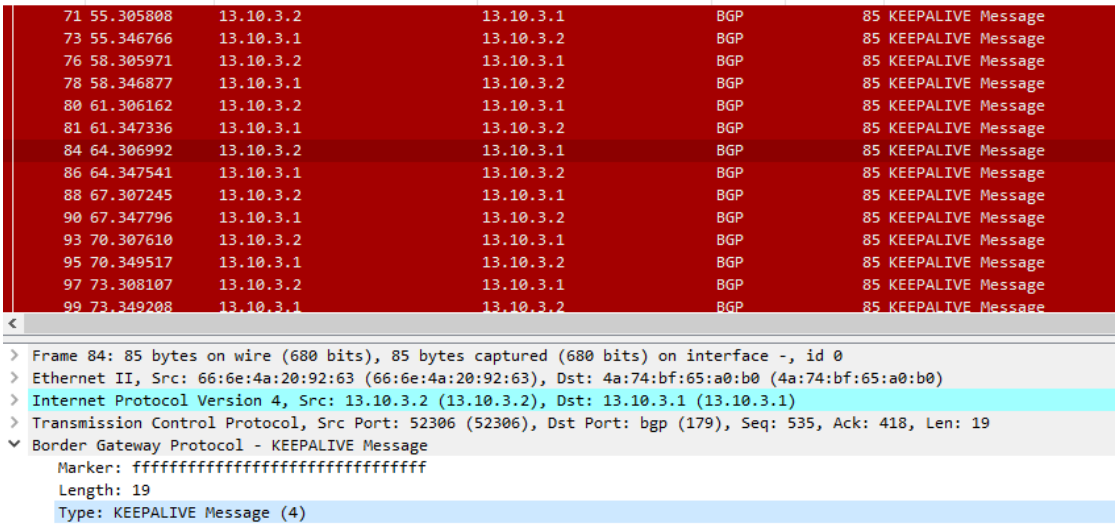


First of all, the BGP is using the TCP port number 179, and the source is R5 (13.10.3.1), is sending the OPEN message to R8 (13.10.3.2). In the packet, we read the information about the AS of R5, the BGP identifier, and so on. An important parameter is the hold time, that is the maximum delay between successive KEEPALIVE, and/or UPDATE messages.

Let's analyze the UPDATE message. It is used to carry both IPv4 route announcements and route withdrawals. The packet sent from R8 to R5 is the following:



We can recognize two important things. The first one is "AS_PATH", that is a list of AS numbers that a route has traversed to reach a destination. When a BGP router advertises a prefix, it will include its own AS number to the left of the AS path attribute. The second one is the "NEXT_HOP", which identify the IP address used to send the packet inter AS. Obviously, if we use the iBGP to propagate the route the Next hop will not be modified by the routers. Also, we have the Network Layer Reachability Information (NLRI), that lists the announced networks.
Then we have the KEEPALIVE messages. These BGP Messages contain only the default header. Every HoldTime/3 seconds, a KEEPALIVE message is sent if no recent BGP message was sent.


71 55.305808 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
73 55.346766 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
76 58.305971 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
78 58.346877 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
80 61.306162 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
81 61.347336 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
84 64.306992 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
86 64.347541 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
88 67.307245 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
90 67.347796 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
93 70.307610 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
95 70.349517 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
97 73.308107 13.10.3.2 13.10.3.1 BGP 85 KEEPALIVE Message
99 73.349208 13.10.3.1 13.10.3.2 BGP 85 KEEPALIVE Message
Frame 84: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface -, id 0
Ethernet II, Src: 66:6e:4a:20:92:63 (66:6e:4a:20:92:63), Dst: 4a:74:bf:65:a0:b0 (4a:74:bf:65:a0:b0)
Internet Protocol Version 4, Src: 13.10.3.2 (13.10.3.2), Dst: 13.10.3.1 (13.10.3.1)
Transmission Control Protocol, Src Port: 52306 (52306), Dst Port: bgp (179), Seq: 535, Ack: 418, Len: 19
Border Gateway Protocol - KEEPALIVE Message
Marker: ffffffffffffffffffffffffffffffff
Length: 19
Type: KEEPALIVE Message (4)

Chapter 4

Latency, rerouting delay and network's improvements

We ran ping between PC3 and PC2 for 5 minutes and checked which path is used to reach PC2. After two minutes, we suspended one of the links along this path so that another path is preferred. At this point, we evaluated the mean and the variance of the rerouting times on 10 experiments.

First path:

```
PC2> trace 42.42.42.2 -P 1
trace to 42.42.42.2, 8 hops max (ICMP), press Ctrl+C to stop
1   10.10.1.1   0.148 ms  0.083 ms  0.255 ms
2   10.10.1.16   0.260 ms  0.194 ms  0.125 ms
3     *  *  *\\
4   42.42.42.17   0.368 ms  0.211 ms  0.282 ms
5   42.42.42.2   0.524 ms  0.813 ms  0.335 ms
```

Second path (after suspending the link between R8 and R5):

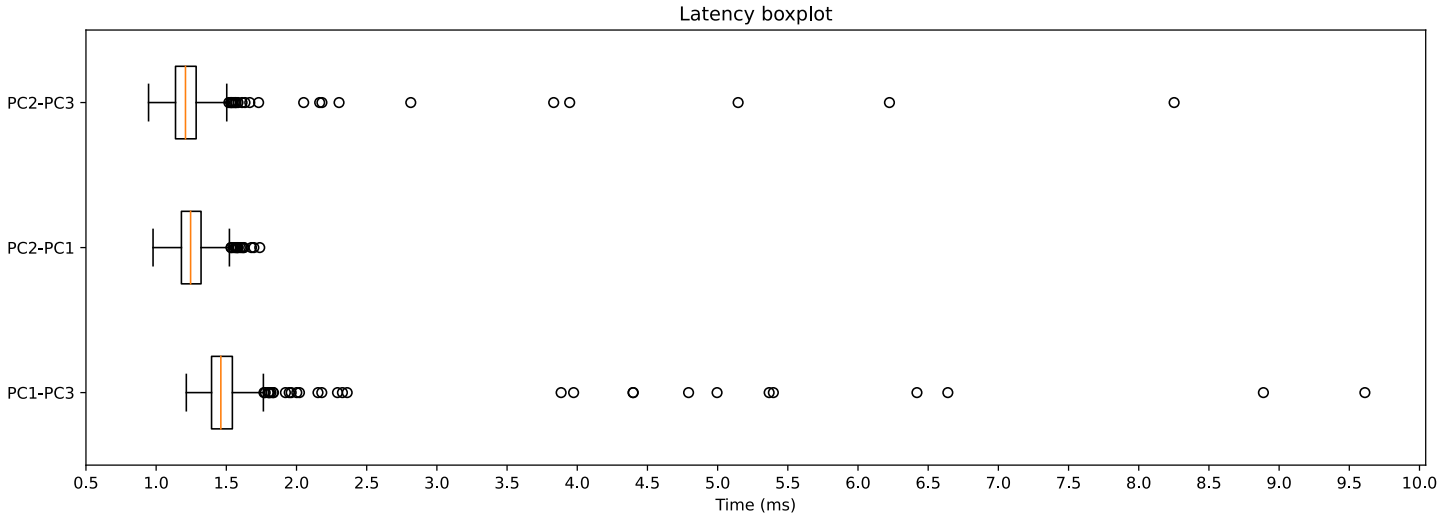
```
PC2> trace 42.42.42.2 -P 1
trace to 42.42.42.2, 8 hops max (ICMP), press Ctrl+C to stop
1   10.10.1.1   0.208 ms  0.172 ms  0.086 ms
2   10.10.1.16   0.210 ms  0.133 ms  0.166 ms
3     *  *  *
4   20.20.0.2   0.520 ms  0.351 ms  0.438 ms
5     *  *  *
6   42.42.42.17   0.620 ms  0.634 ms  0.335 ms
7   42.42.42.2   0.430 ms  0.545 ms  0.590 ms
```

To get the measurements we have opened PC2 console and used the following commands:

```
ping 42.42.42.2 -i 10 -w 10 -c 30000
```

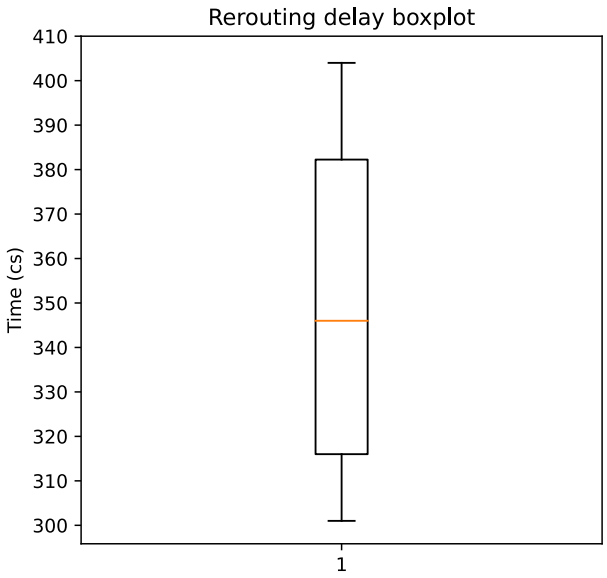
-i: time interval between every sent ICMP packet.
-w: wait time for the reply.
-c: number of pings. Since that we are going to do 30.000 ping, the experiment will last 5 minutes.
During the ping-process we wait for $icmp\_seq \geq 12000$ and then suspend the link between R8 and R5.

4.1 Latency and rerouting delay



	PC2 - PC3	**PC2 - PC1**	**PC1 - PC3**
Mean (ms):	1.26	1.26	1.56
Std.dev (ms):	0.44	0.11	0.66

The latency is calculated as the sum of the delays of the individual connections. The delay of each link is equal to the sum of the processing, queuing, propagation and transmission delays.



The rerouting delay is the time needed to reroute the packets. It's calculated as the number of failed pings during the rerouting times the interval between each packet.

	Rerouting delay (ms)
Mean	3487
Std. dev.	390

The graphs that we've shown in this chapters are called "box plot". A box plot shows the distribution of data for a continuous variable. A box plot allows us to visualize the center and distribution of data. Furthermore, it can be used as a visual tool for checking normality or identifying possible outliers. For examples in the Latency box plot we have a lot of outliers. In the Rerouting time box plot there aren't outliers.

The center line in the box represents the median of the data. Half of the data is above this value, the other half below. If the data is symmetric, the median is in the center of the box. However, if the data is skewed, the median will be closer to the top or bottom of the box.

4.2 Improvements

One further implementation could be the addition of a next-hop for backup. This would lead to a rerouting time improvement. The BGP protocol allows us to do it with the following command:

```
apply fast-reroute backup-nexthop ipv4-address
```

If we would have a multi-hop path we could improve the rerouting time using the following command:

```
bgp-fast-convergence
```

Whenever BGP peer address becomes unreachable we must bring down the BGP session immediately. Currently only single-hop EBGP sessions are brought down immediately.IBGP and multi-hop EBGP sessions wait for hold-timer expire to bring down the sessions.

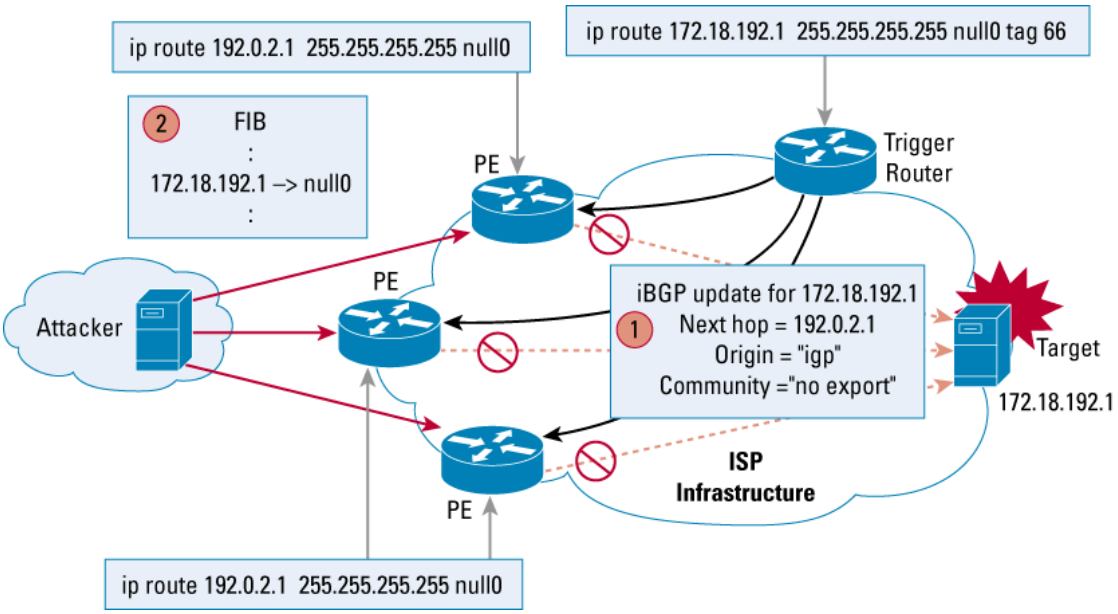
This new configuration option helps user to teardown BGP sessions immediately whenever a peer becomes unreachable.
This configuration is available at the BGP level. When enabled, the setting is applied to all the neighbors configured in that BGP instance.

Chapter 5

DOS attack

A Denial of Service attack (DoS attack) is a cyber-attack in which the attacker tries to make a service or a resource unavailable temporarily or indefinitely. Denial of service is typically accomplished by flooding the targeted machine or resource with a lot of requests in order to overload victim systems. As a defensive strategy against DoS attacks we decided to make use of the blackholing technique.
Black holes, from a network security perspective, are placed in the network where traffic is forwarded and dropped. Once an attack has been detected, black holing can be used to drop all attack traffic at the edge of an Internet service provide (ISP) network, based on either destination or source IP addresses.

5.1 Remote Triggered Black-Holing



RTBH filtering is a technique that uses routing protocol updates to manipulate route tables at the network edge or anywhere else in the network to specifically drop undesirable traffic before it enters the service provider network.
RTBH filtering provides a method for quickly dropping undesirable traffic at the edge of the network, based on either source addresses or destination addresses by forwarding it to a null0 interface. Null0 is a pseudointerface that is always up and can never forward or receive traffic. Forwarding packets to null0 is a common way to filter packets to a specific destination.
A typical deployment scenario for RTBH filtering would require running iBGP at the access and aggregation points and configuring a separate device to act as a trigger. The triggering device sends iBGP updates to the edge, that causes undesirable traffic to be forwarded to a null0 interface and dropped.
Usually destination based blackholing is preferred since the source IP address is spoofed most of the times. With a simple black hole all the traffic to the victim destination is dropped at the edge of the network. If the source of the attack is known it's possible to limit the application of the black holes to a specific region. This would allow uninterrupted traffic from other regions while mitigating the attack.

5.1.1 End devices setup

First thing first we have to configure the two docker images. On the d-itg-1 terminal we ran the following commands:

```
ifconfig eth0 192.172.1.2 netmask 255.255.255.0
route add default gw 192.172.1.1 eth0
```

On the d-itg-2 terminal we ran the following commands:

```
ifconfig eth0 10.10.1.2 netmask 255.255.255.240
route add default gw 10.10.1.1 eth0
```

To generate traffic among the two d-itg hosts, on d-itg-2 terminal we ran:

```
ITGRecv
```

And on the d-itg-1 terminal we ran

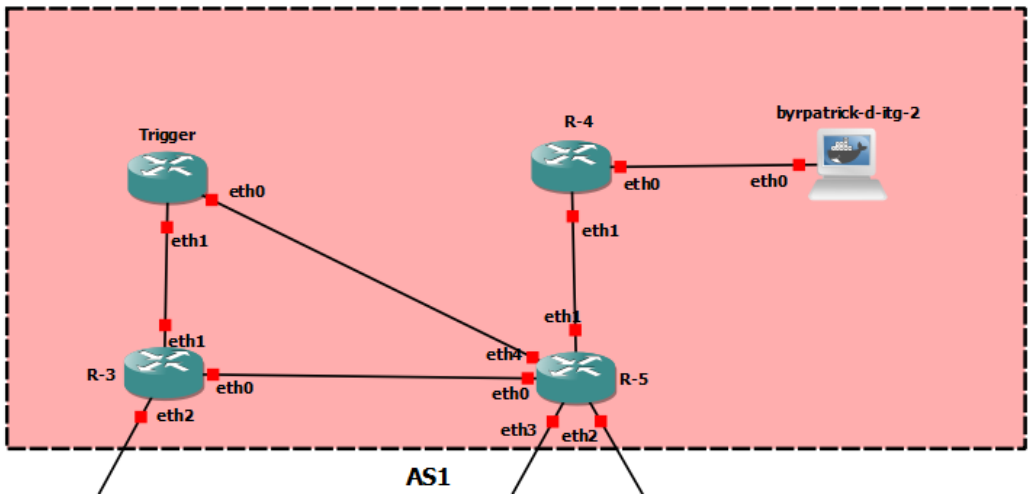
```
ITGSend -T UDP -a 10.10.1.2 -C 800 -t 250000 -l sender.log -x receiver.log
```

To show a summary report of the transmission we can run the following command on d-itg-2

```
ITGDec receiver.log
```

By sending a high enough number of packets it's possible to flood d-itg-2 with malicious traffic and cause any service it's offering to time out.

5.1.2 Topology changes



Device	Interface	IP-Address
R5	eth4	10.10.2.2/30
R3	eth1	10.10.2.6/30
Trigger	eth0	10.10.2.1/30
Trigger	eth1	10.10.2.5/30

To implement the RTBH a Trigger router was added to the AS1 and connected on eth0 and eth1 to R-5 and R-3 respectively. For those new links couple of new subnets (10.10.2.0 and 10.10.2.4) were created. The Trigger router was added to the peer-group AS1 for iBGP communication.
At first a more complex topology was tested. We put a Route Reflector between R5 and R4 and connected it to R3, R4, R5 and the Trigger in order to reflect the blackholing routes sent by the Trigger. This topology was scrapped because of how FRR changes the next-hop using the connected interface instead of the IP address. Additionally such a complex solution was overkill in a simple network like this.

5.1.3 Trigger setup

The purphose of Trigger is to inject with iBGP a static route to a dummy IP address which will serve as next-hop address for the routing on the Edge Routers. To do so three route maps were defined:

```
 route-map black-hole-trigger permit 10
     match tag 66
     set community no-export
     set ip next-hop 192.0.2.1
     set local-preference 200
     set origin igp
exit
!
route-map black-hole-trigger permit 15
     match tag 76
     set community no-export
     set ip next-hop 192.0.2.2
     set local-preference 200
     set origin igp
exit
!
route-map black-hole-trigger permit 20
     match tag 86
     set community no-export
     set ip next-hop 192.0.2.3
     set local-preference 200
     set origin igp
exit
!
route-map black-hole-trigger deny 25
exit
```

Each route map was defined as follow:

- A tag based matching condition for which the route map is going to be applied.
- The BGP community is set to no-export which means that the route is going to be advertised just to internal BGP nodes and not to other AS.
- The next-hop is set to a dummy IP address. In particular addresses from the 192.0.2.0/24 block were used, this block works nicely for dummy addresses since it's reserved for documentation and examples. The next-hop parameter could be used also to create a sinkhole that is used to send all the suspicious trafic for analysys.
- In order to prefer this route when the RTBH is enabled an higher (200) local-pref was set.
- The route origin was set to IGP.

In order to advertise those routes to the neighbours BGP was set to redistribute all the static routes. Also the next-hop was set to not be changed during the route advertisement.

```
router bgp 101
 address-family ipv4 unicast
  redistribute static route-map black-hole-trigger
  neighbor AS1 next-hop-self
  neighbor AS1 attribute-unchanged next-hop
 exit-address-family
exit
```

5.1.4 Edge routers setup

For the edge routers all is needed is to setup a static route from the dummy addresses towards the Null0 interface.

For R-3:

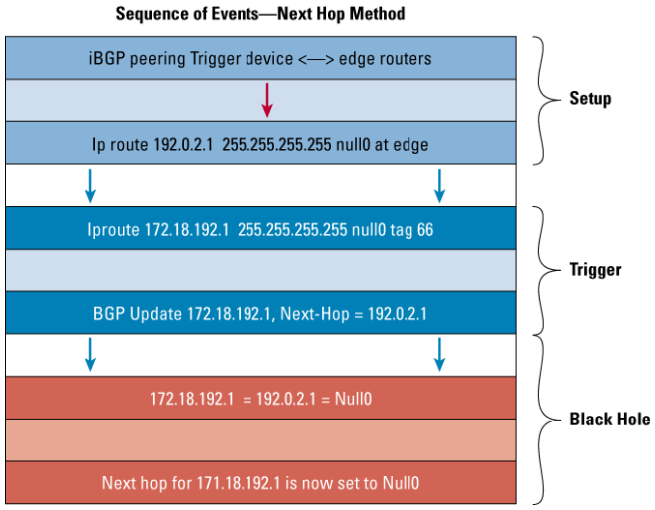
```
ip route 192.0.2.1/32 Null0
ip route 192.0.2.3/32 Null0
```

For R-5:

```
ip route 192.0.2.2/32 Null0
ip route 192.0.2.3/32 Null0
```

With this configuration depending on the tag setted on Trigger a different region will be blocked. For tag 66 the region connected to R-3 will be blackholed, for tag 76 this will happen for the region connected to R-5, while for tag 86 blackholing will occur to both.

5.1.5 Triggering the black hole



Once a DoS attack is detected a network administrator can use the Trigger router to inject a static route to the edge routers toward the dummy IP addresses. For example by issuing the command

```
ip route 10.10.1.2 255.255.255.255 null0 tag 66
```

we setup a static route towards the victim (PC2) with destination Null0 and tag 66. The tag is matched in the route maps and so the next-hop 192.0.2.1 with $local - pref = 200$ is advertised to the trigger peers. The edge routers will receive this new next-hop and they are going to add it to their BGP routing paths.
For those devices which don't define a static route for the dummy IP address the path is invalid and is not used.

```
R-5# sh ip bgp
BGP table version is 7, local router ID is 10.10.0.1, vrf id 0
Default local pref 100, local AS 101
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
*> 10.10.0.0/24     0.0.0.0(R-5)             0         32768 i
*> 10.10.1.0/24     0.0.0.0(R-5)             0         32768 i
  i10.10.1.2/32     192.0.2.1(Trigger)
                                             0    200      0 i
*> 20.20.0.0/30     13.10.2.2(R-6)           0             0 102 i
*                   13.10.3.2(R-8)                         0 104 102 i
*  42.42.42.0/28    13.10.2.2(R-6)                         0 102 104 i
*>                  13.10.3.2(R-8)                         0 104 i
*  192.172.1.0/24   13.10.2.2(R-6)                         0 102 100 i
*>i                 eth0                     0    100      0 100 i
```

For those who have defined a route towards Null0 for the dummy address the path is valid and it's going to be preferred to the normal path because of the higher local-pref.

```
R-3# sh ip bgp
BGP table version is 8, local router ID is 10.10.0.2, vrf id 0
Default local pref 100, local AS 101
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
*>i10.10.0.0/24     eth0                     0    100      0 i
*>i10.10.1.0/24     eth0                     0    100      0 i
*>i10.10.1.2/32     192.0.2.1(Trigger)
                                             0    200      0 i
*  20.20.0.0/30     13.10.1.2(R-2)                         0 100 102 i
*>i                 eth0                     0    100      0 102 i
*>i42.42.42.0/28    eth0                          100      0 104 i
*> 192.172.1.0/24   13.10.1.2(R-2)           0             0 100 i
```

As shown in the IP routes of R-3 the path towards PC2 from this region is now blackholed.

```
 R-3# sh ip route
Codes: S - static, B - BGP
       > - selected route, * - FIB route,

B>  10.10.1.2/32 [200/0] via 192.0.2.1 (recursive), weight 1
  *                        unreachable (blackhole), weight 1
S>* 192.0.2.1/32 [1/0] unreachable (blackhole), weight 1
S>* 192.0.2.3/32 [1/0] unreachable (blackhole), weight 1
```

With this all the traffic towards PC2 from a specific region will be dropped at the edge routers while PC2 can still talk with hosts and networks from different regions.