Playing with off-Ramp / On-Ramp

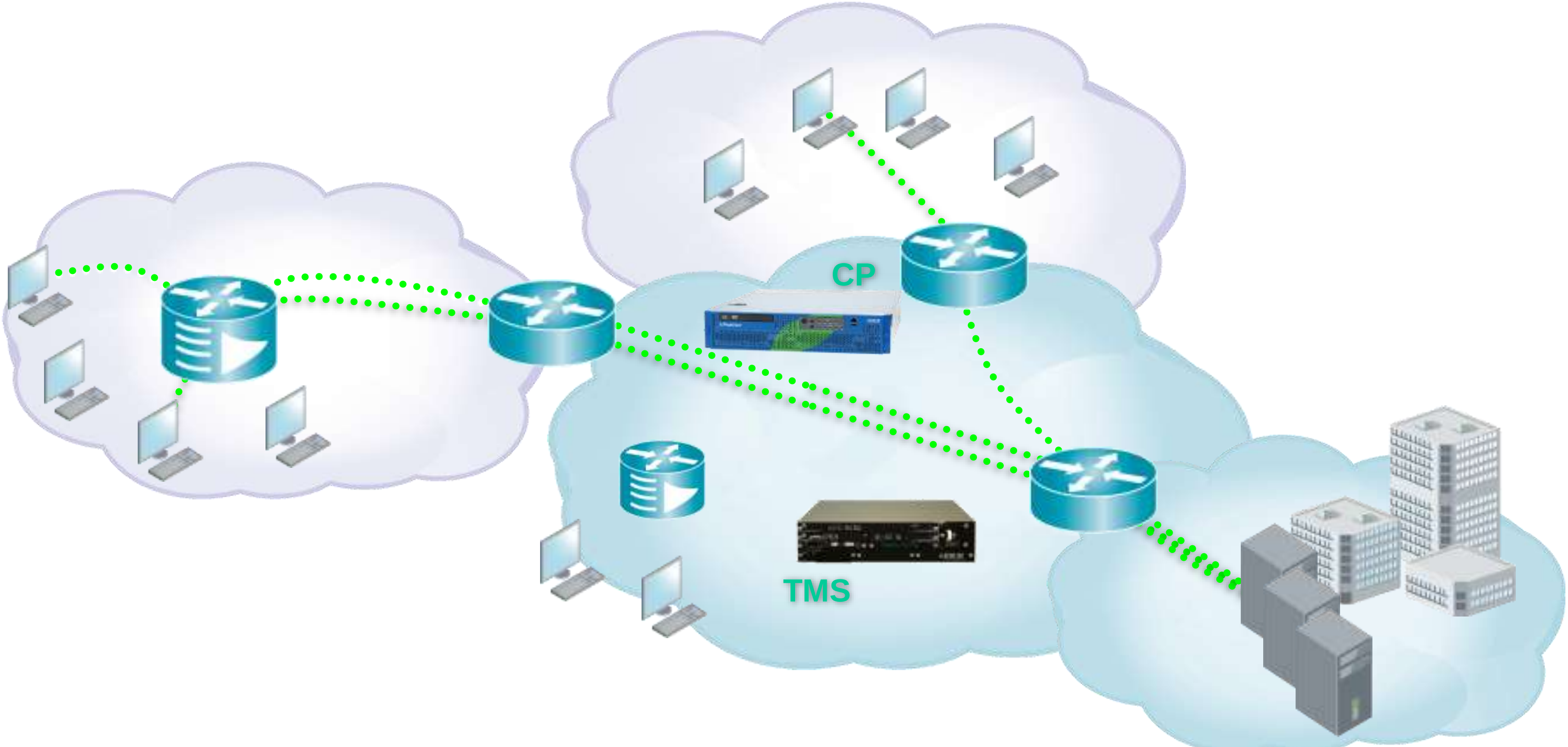
Ferran Orsola
forsola@arbor.net

Agenda

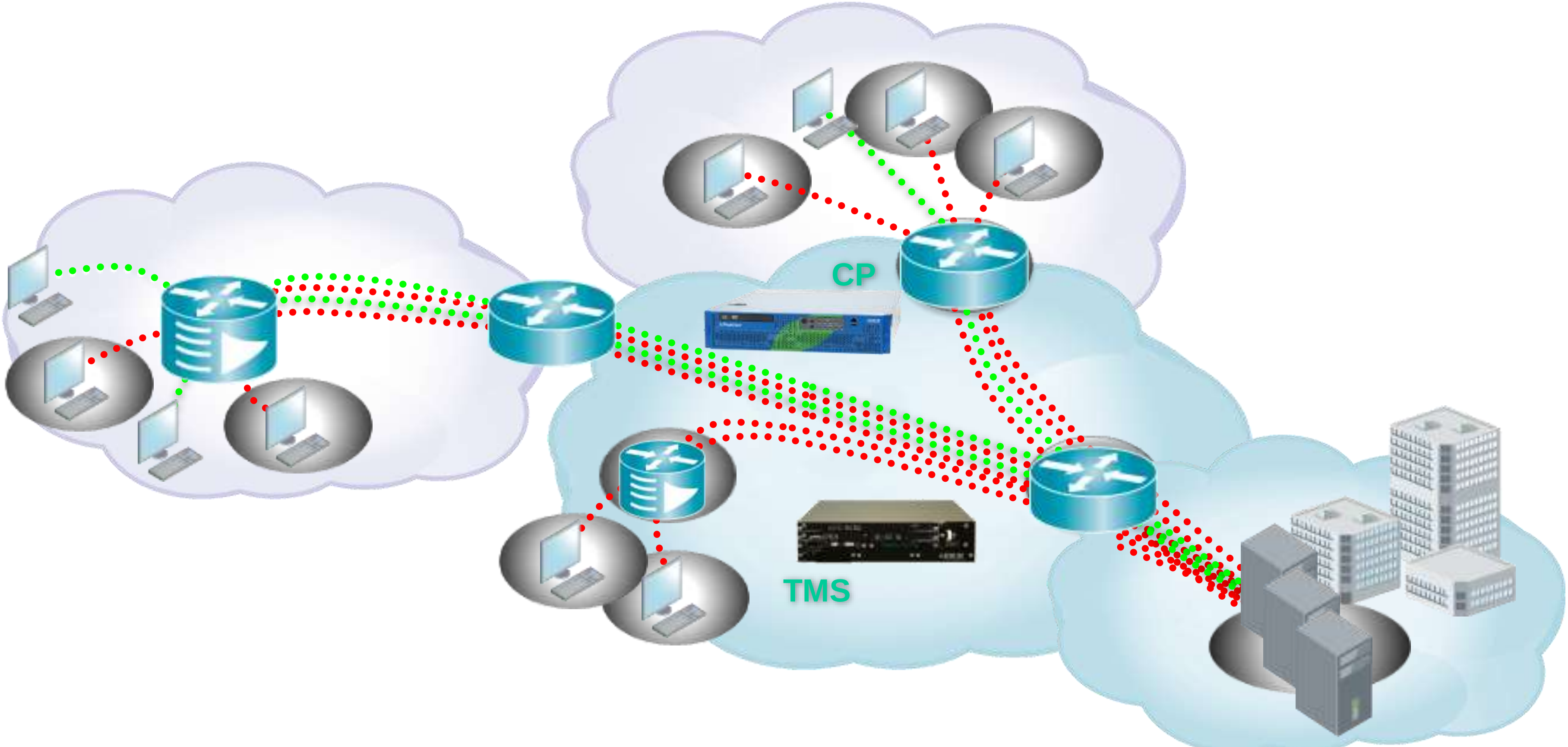
1. DDOS Mitigation in Service Providers
2. General Labs Specifications
3. Cisco – Dirty VRF
4. Cisco – VRF + Static Route Leaking
5. Cisco IOS – Dynamic Route Leaking
6. Cisco IOS-XR – Dynamic Route Leaking
7. Juniper – Rib Groups
8. Flowspec with JunOS

DDOS Mitigation in Service Providers

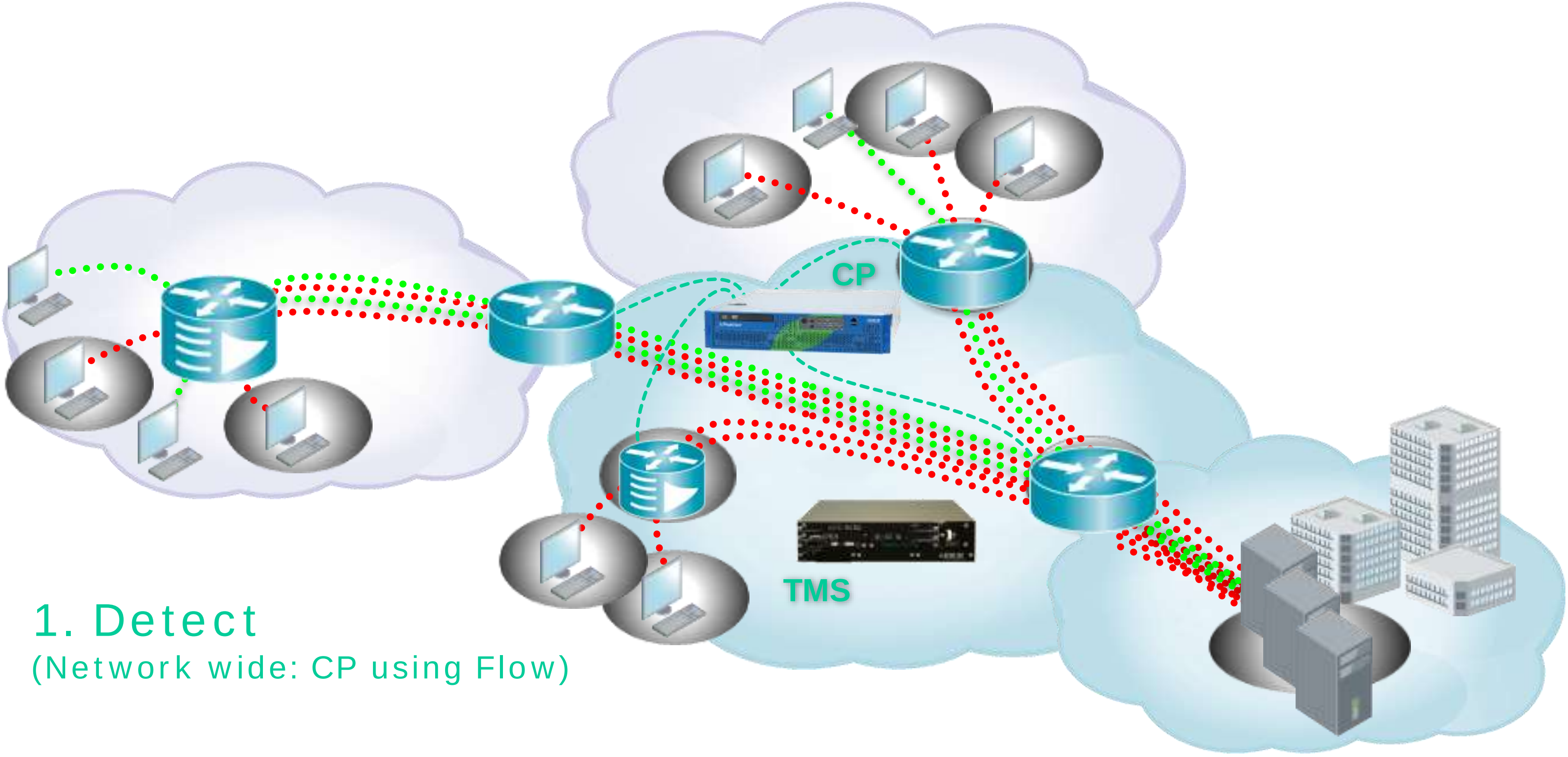
DDoS - Mitigation



DDoS - Mitigation



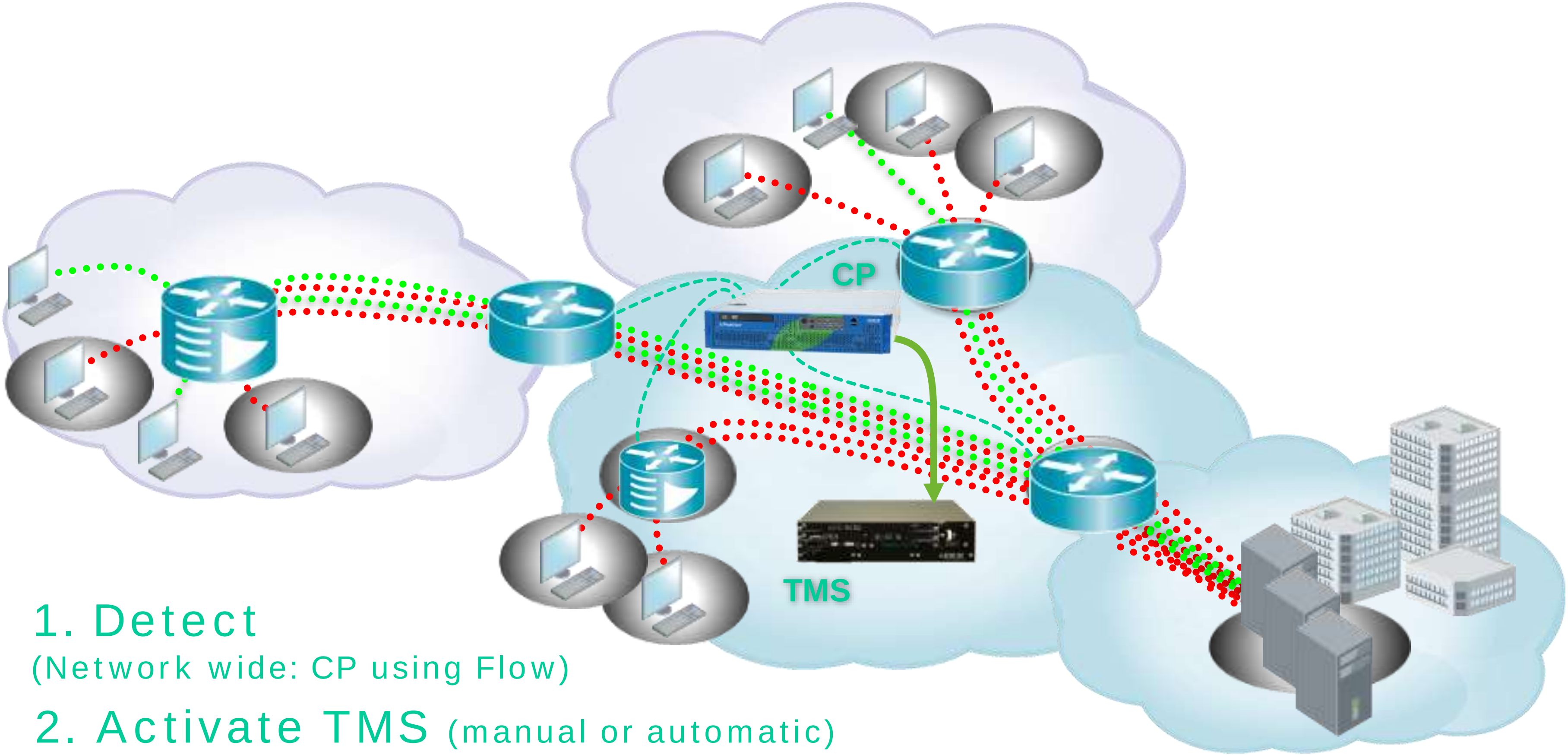
DDoS - Mitigation



1. Detect

(Network wide: CP using Flow)

DDoS - Mitigation



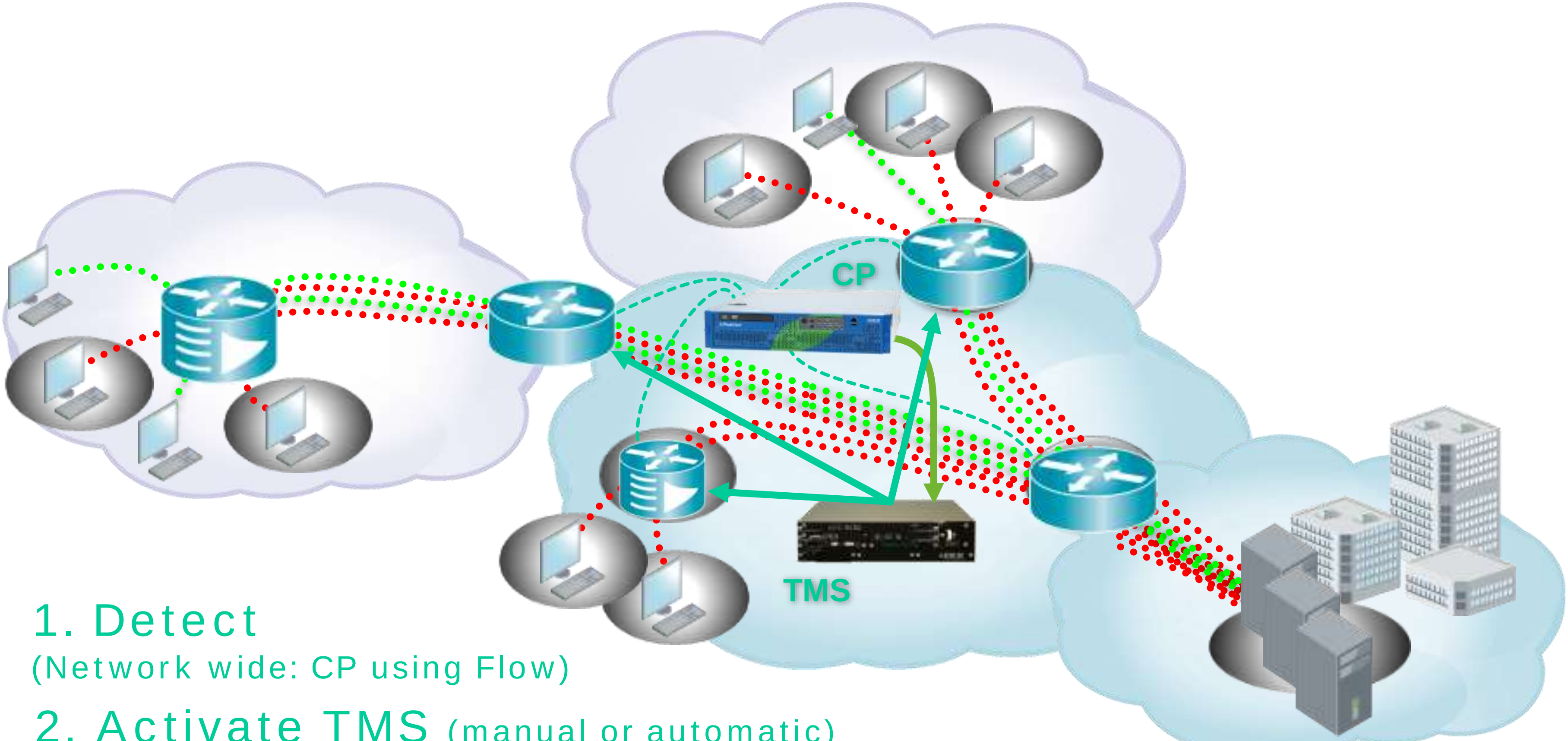
1. Detect

(Network wide: CP using Flow)

2. Activate TMS (manual or automatic)

ARBOR NETWORKS

DDoS - Mitigation

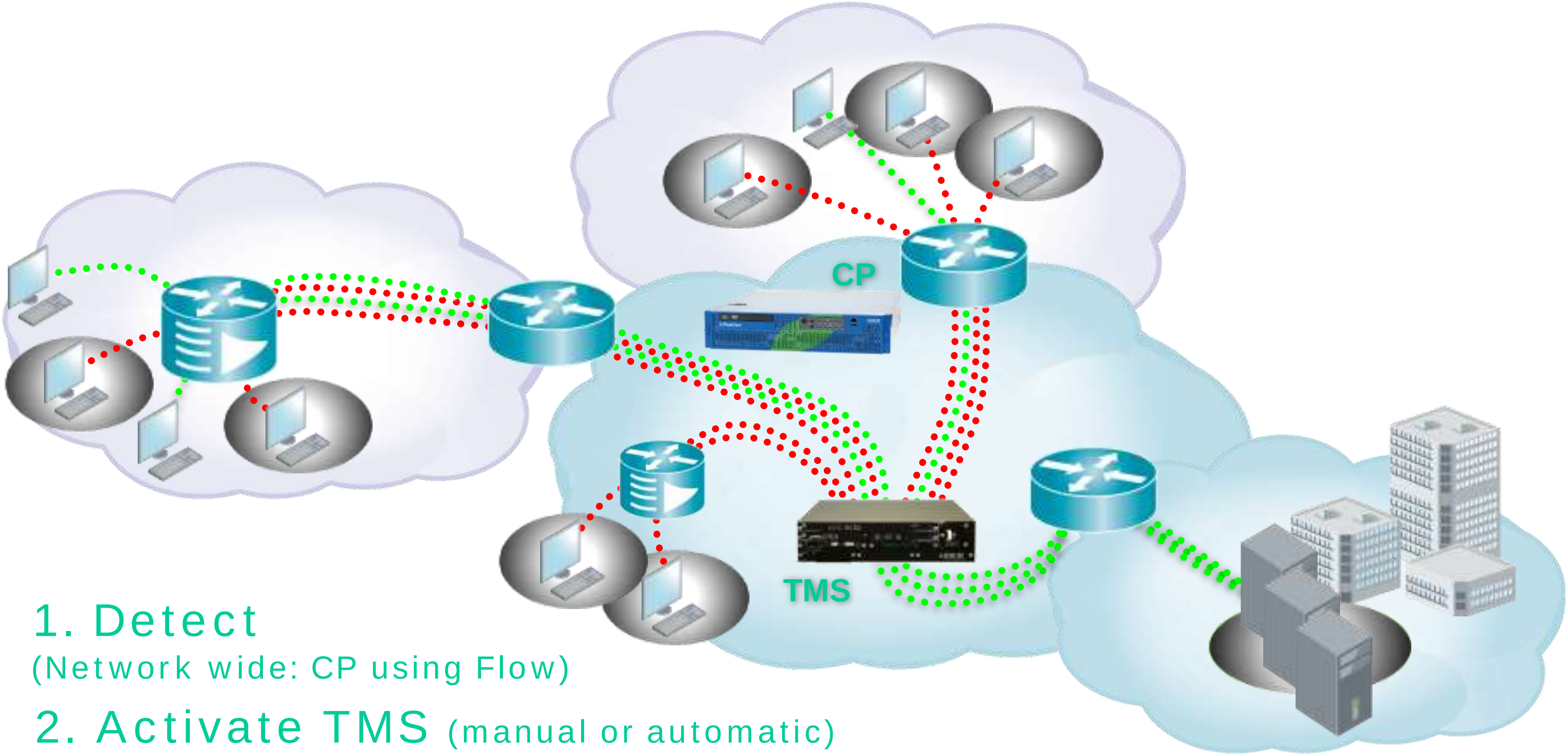


1. Detect
(Network wide: CP using Flow)

2. Activate TMS (manual or automatic)

3. Divert Traffic (Network wide: BGP OFF-Ramp announcement)

DDoS - Mitigation



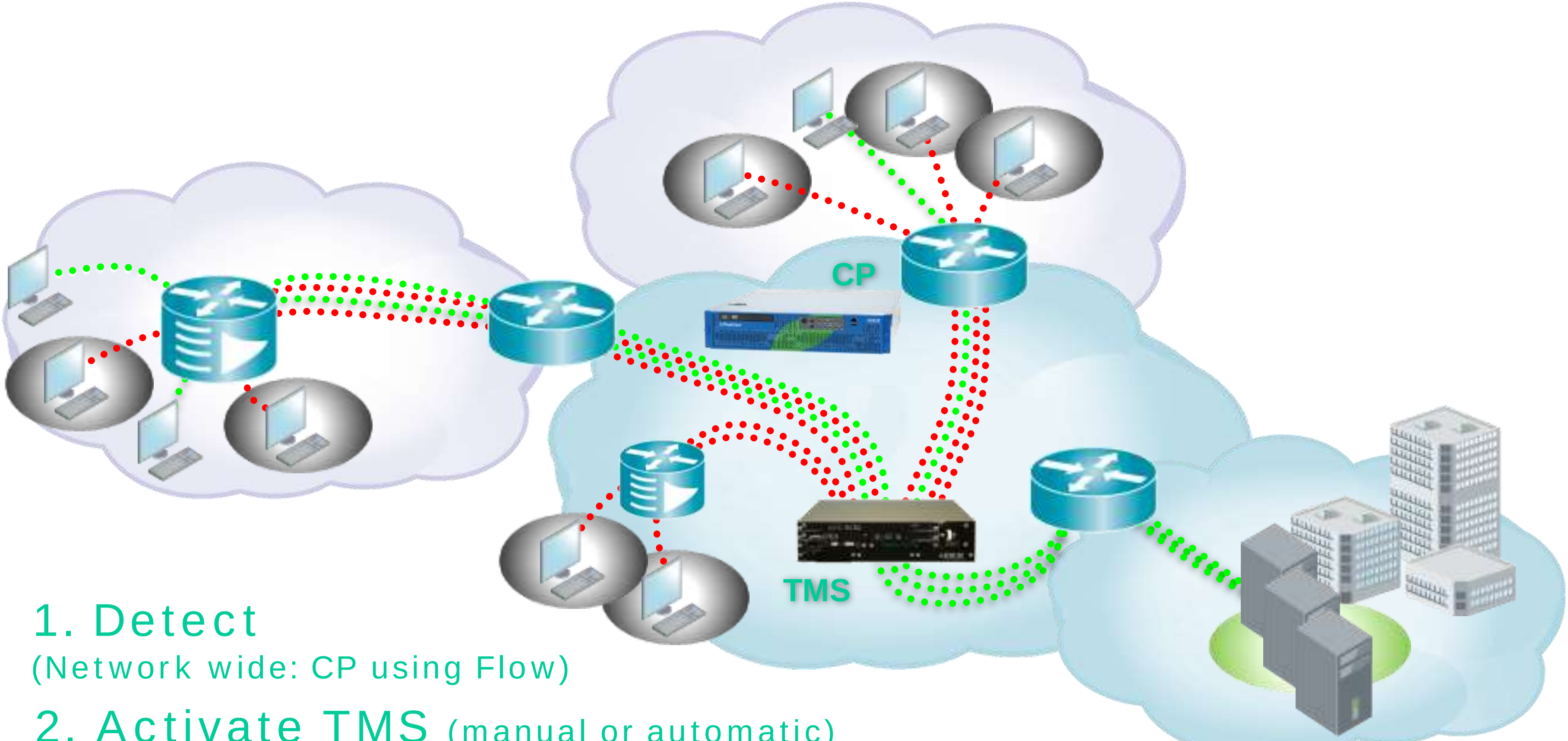
1. Detect
(Network wide: CP using Flow)

2. Activate TMS (manual or automatic)

3. Divert Traffic (Network wide: BGP OFF-Ramp announcement)

4. Clean the Traffic and forward the legitimate
(Network wide: using ON-**Ramp Technique [e.g. MPLS, GRE, VLAN, …])**

DDoS - Mitigation



1. Detect
(Network wide: CP using Flow)

2. Activate TMS (manual or automatic)

3. Divert Traffic (Network wide: BGP OFF-Ramp announcement)

4. Clean the Traffic and forward the legitimate
(Network wide: using ON-**Ramp Technique [e.g. MPLS, GRE, VLAN, …])**

5. Protected

General Labs Specifications

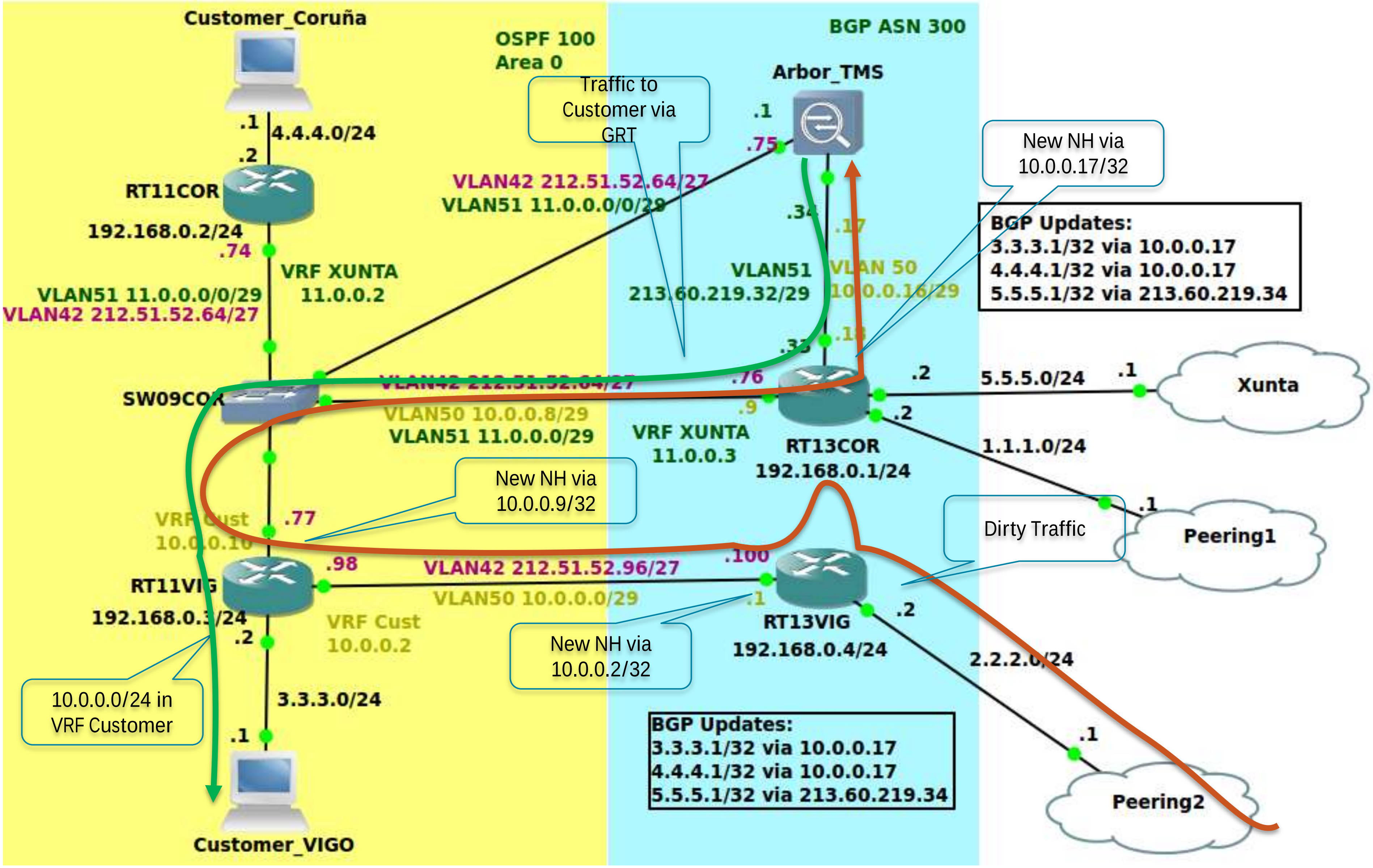
How to use the labs ?

- I used GNS3 for all the labs, they are available in https://arbor.box.com/Mitigation-Labs
- Cisco IOS works emulated real firmware (dynamips)
- Virtual Images for IOS-XR and JunOS works running Vmware (vmdk files) with qemu).
- Clients are emulated with VPCS
- All Router configuration are saved in config folder in each of the labs.
- Routers are available in https://arbor.box.com/s/27q4932mbh4lgtp2do38

If you want access to the labs please ASK!!!!!

Cisco IOS – Dirty VRF Design

Cisco IOS – Dirty VRF to Customers



Traffic to Customer via GRT

New NH via 10.0.0.17/32

BGP Updates:
3.3.3.1/32 via 10.0.0.17
4.4.4.1/32 via 10.0.0.17
5.5.5.1/32 via 213.60.219.34

New NH via 10.0.0.9/32

Dirty Traffic

New NH via 10.0.0.2/32

10.0.0.0/24 in VRF Customer

BGP Updates:
3.3.3.1/32 via 10.0.0.17
4.4.4.1/32 via 10.0.0.17
5.5.5.1/32 via 213.60.219.34

Cisco IOS – VRF for Dirty Traffic: Customers

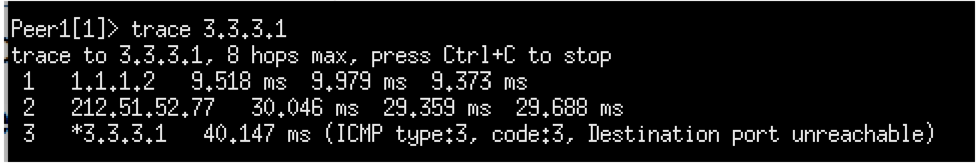
In order to Simulate the route poison from Arbor SP add:

1.- ip route 3.3.3.1 255.255.255.255 10.0.0.17 (all peering routers)

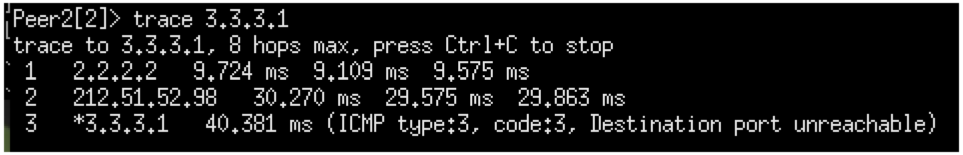
2.- ping 3.3.3.1 from any peering IP (1.1.1.1 or 2.2.2.2) using VPCS

Test: traceroute from any Peer to any Customer:

- Before Poisoning the route:

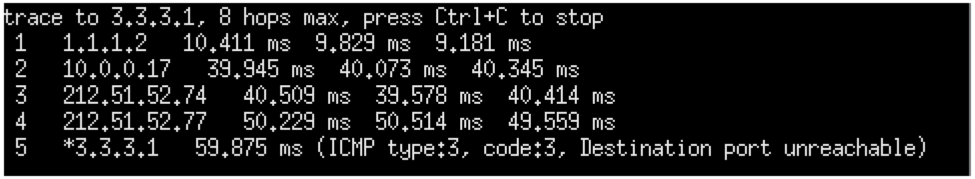


```
Peer1[1]> trace 3.3.3.1
trace to 3.3.3.1, 8 hops max, press Ctrl+C to stop
 1   1.1.1.2   9.518 ms  9.979 ms  9.373 ms
 2   212.51.52.77   30.046 ms  29.359 ms  29.688 ms
 3   *3.3.3.1   40.147 ms (ICMP type:3, code:3, Destination port unreachable)
```

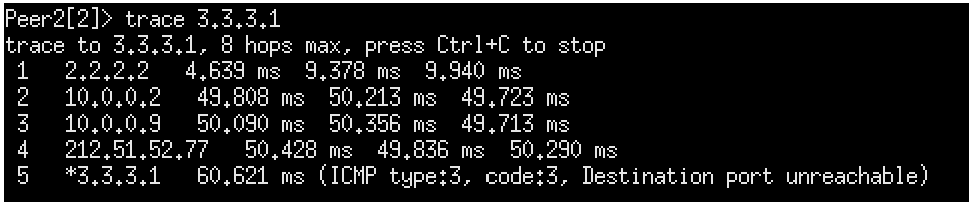


```
Peer2[2]> trace 3.3.3.1
trace to 3.3.3.1, 8 hops max, press Ctrl+C to stop
 1   2.2.2.2   9.724 ms  9.109 ms  9.575 ms
 2   212.51.52.98   30.270 ms  29.575 ms  29.863 ms
 3   *3.3.3.1   40.381 ms (ICMP type:3, code:3, Destination port unreachable)
```

- After TMS update:



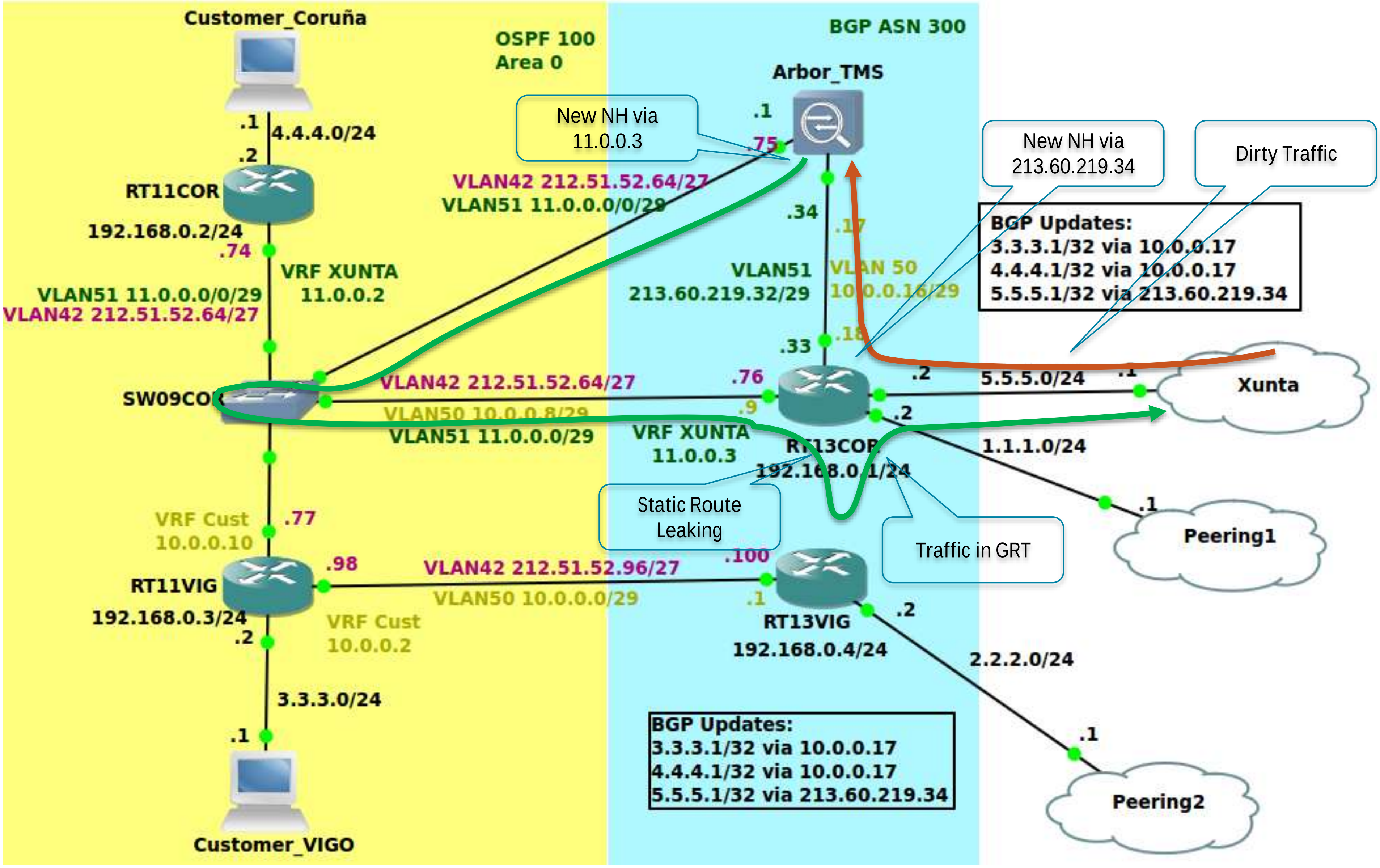
```
trace to 3.3.3.1, 8 hops max, press Ctrl+C to stop
 1   1.1.1.2   10.411 ms  9.829 ms  9.181 ms
 2   10.0.0.17   39.945 ms  40.073 ms  40.345 ms
 3   212.51.52.74   40.509 ms  39.578 ms  40.414 ms
 4   212.51.52.77   50.229 ms  50.514 ms  49.559 ms
 5   *3.3.3.1   59.875 ms (ICMP type:3, code:3, Destination port unreachable)
```



```
Peer2[2]> trace 3.3.3.1
trace to 3.3.3.1, 8 hops max, press Ctrl+C to stop
 1   2.2.2.2   4.639 ms  9.378 ms  9.940 ms
 2   10.0.0.2   49.808 ms  50.213 ms  49.723 ms
 3   10.0.0.9   50.090 ms  50.356 ms  49.713 ms
 4   212.51.52.77   50.428 ms  49.836 ms  50.290 ms
 5   *3.3.3.1   60.621 ms (ICMP type:3, code:3, Destination port unreachable)
```

Cisco IOS – Static Routing Leaking

Cisco IOS – VRF + Static Route Leaking



New NH via 11.0.0.3

New NH via 213.60.219.34

Dirty Traffic

BGP Updates:
3.3.3.1/32 via 10.0.0.17
4.4.4.1/32 via 10.0.0.17
5.5.5.1/32 via 213.60.219.34

Static Route Leaking

Traffic in GRT

BGP Updates:
3.3.3.1/32 via 10.0.0.17
4.4.4.1/32 via 10.0.0.17
5.5.5.1/32 via 213.60.219.34

Cisco IOS – Static Route Leaking

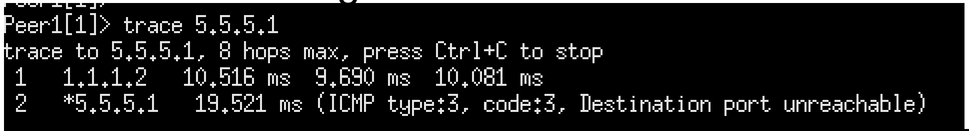
In order to Simulate the route poison from Arbor SP add:

1.- ip route 5.5.5.1 255.255.255.255 213.60.219.34

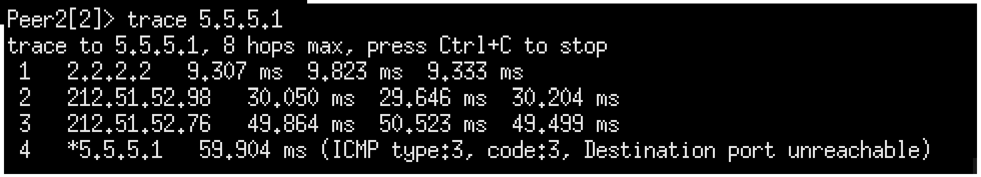
2.- ping 5.5.51 from any peering IP (1.1.1.1 or 2.2.2.2) using VPCS

Test: traceroute from any Peer to Xunta:

- Before Poisoning the route:

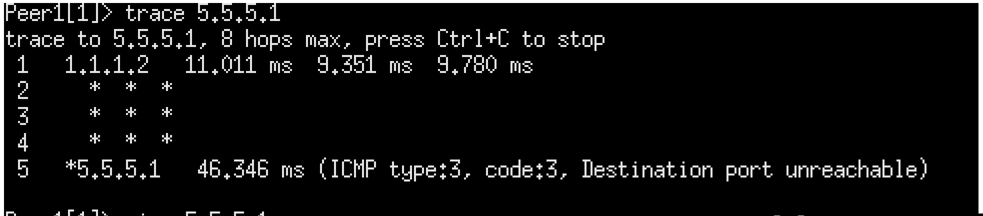


```
Peer1[1]> trace 5.5.5.1
trace to 5.5.5.1, 8 hops max, press Ctrl+C to stop
 1   1.1.1.2   10.516 ms  9.690 ms  10.081 ms
 2   *5.5.5.1   19.521 ms (ICMP type:3, code:3, Destination port unreachable)
```

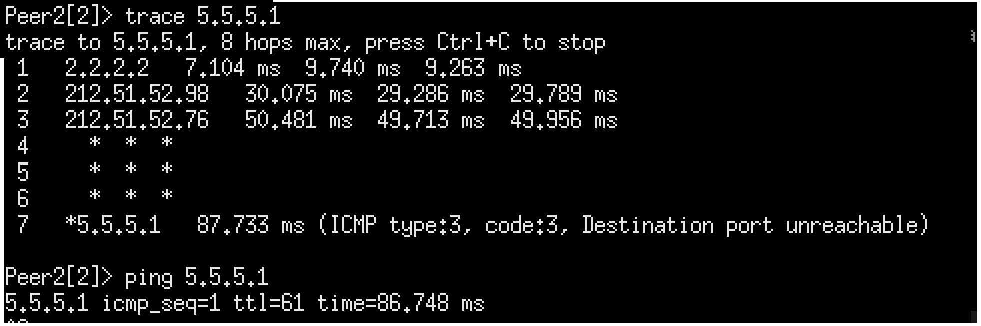


```
Peer2[2]> trace 5.5.5.1
trace to 5.5.5.1, 8 hops max, press Ctrl+C to stop
 1   2.2.2.2   9.307 ms  9.823 ms  9.333 ms
 2   212.51.52.98   30.050 ms  29.646 ms  30.204 ms
 3   212.51.52.76   49.864 ms  50.523 ms  49.499 ms
 4   *5.5.5.1   59.904 ms (ICMP type:3, code:3, Destination port unreachable)
```

- After TMS update:



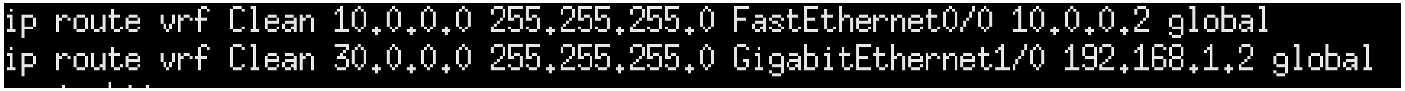
```
Peer1[1]> trace 5.5.5.1
trace to 5.5.5.1, 8 hops max, press Ctrl+C to stop
 1   1.1.1.2   11.011 ms  9.351 ms  9.780 ms
 2     *  *  *
 3     *  *  *
 4     *  *  *
 5   *5.5.5.1   46.346 ms (ICMP type:3, code:3, Destination port unreachable)

Peer1[1]> ping 5.5.5.1
5.5.5.1 icmp_seq=1 ttl=63 time=50.185 ms
```



```
Peer2[2]> trace 5.5.5.1
trace to 5.5.5.1, 8 hops max, press Ctrl+C to stop
 1   2.2.2.2   7.104 ms  9.740 ms  9.263 ms
 2   212.51.52.98   30.075 ms  29.286 ms  29.789 ms
 3   212.51.52.76   50.481 ms  49.713 ms  49.956 ms
 4     *  *  *
 5     *  *  *
 6     *  *  *
 7   *5.5.5.1   87.733 ms (ICMP type:3, code:3, Destination port unreachable)

Peer2[2]> ping 5.5.5.1
5.5.5.1 icmp_seq=1 ttl=61 time=86.748 ms
```

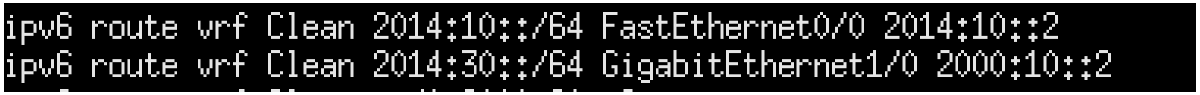
Cisco IOS – Static Route Leaking

- Configuration for Cisco IOS:
  - IPv4:



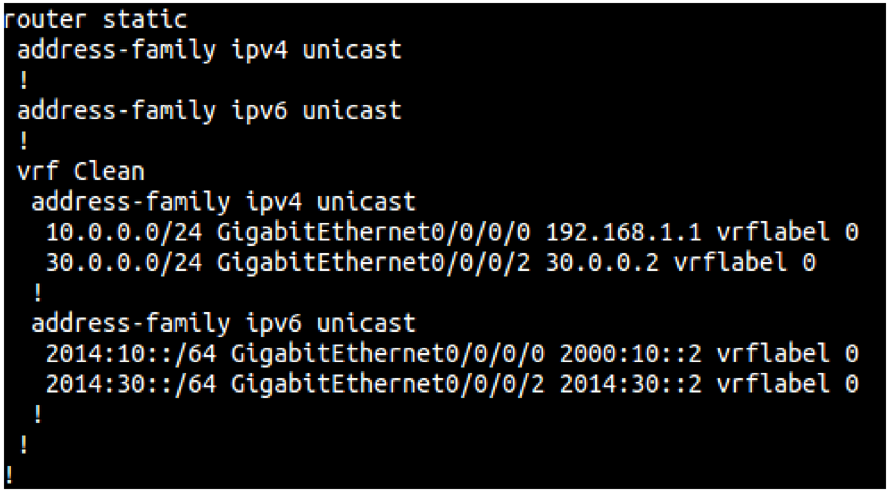
```
ip route vrf Clean 10.0.0.0 255.255.255.0 FastEthernet0/0 10.0.0.2 global
ip route vrf Clean 30.0.0.0 255.255.255.0 GigabitEthernet1/0 192.168.1.2 global
```

  - IPv6:



```
ipv6 route vrf Clean 2014:10::/64 FastEthernet0/0 2014:10::2
ipv6 route vrf Clean 2014:30::/64 GigabitEthernet1/0 2000:10::2
```

- Configuration for IOS-XR (IPv4 and IPv6):

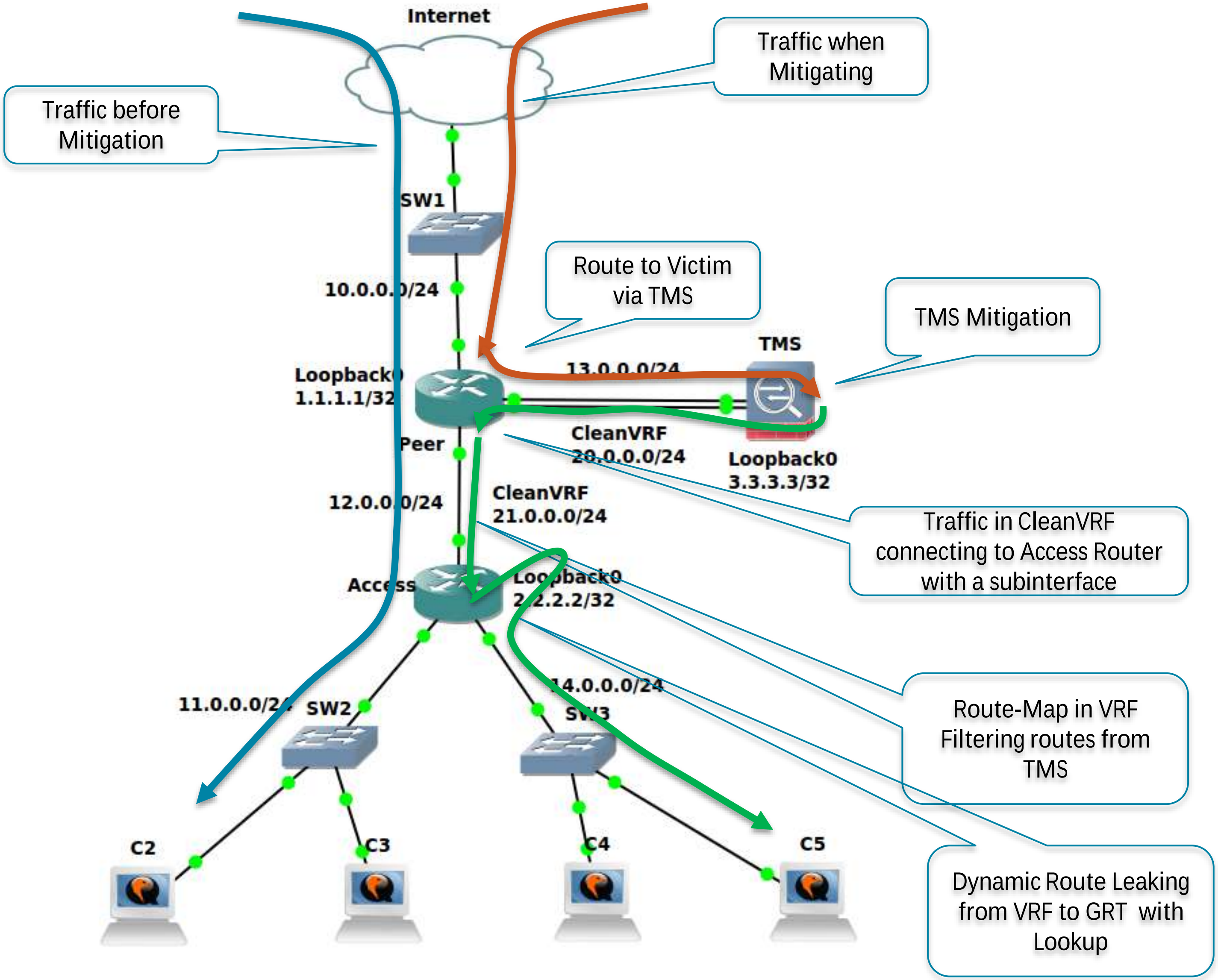


```
router static
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 vrf Clean
  address-family ipv4 unicast
   10.0.0.0/24 GigabitEthernet0/0/0/0 192.168.1.1 vrflabel 0
   30.0.0.0/24 GigabitEthernet0/0/0/2 30.0.0.2 vrflabel 0
  !
  address-family ipv6 unicast
   2014:10::/64 GigabitEthernet0/0/0/0 2000:10::2 vrflabel 0
   2014:30::/64 GigabitEthernet0/0/0/2 2014:30::2 vrflabel 0
  !
 !
!
```

- Full Lab in https://arbor.box.com/Mitigation-Labs/StaticRouterLeaking.tar.gz
- Additional Lab https://arbor.box.com/Mitigation-Labs/RouteLeakingAll.tar.gz (IOS vs IOS-XR and IPv4 vs IPV6)

Cisco IOS – Dynamic Route Leaking

Cisco IOS – Dynamic Route Leaking



Traffic before Mitigation

Traffic when Mitigating

Route to Victim via TMS

TMS Mitigation

Traffic in CleanVRF connecting to Access Router with a subinterface

Route-Map in VRF Filtering routes from TMS

Dynamic Route Leaking from VRF to GRT with Lookup

Cisco IOS – Dynamic Route Leaking

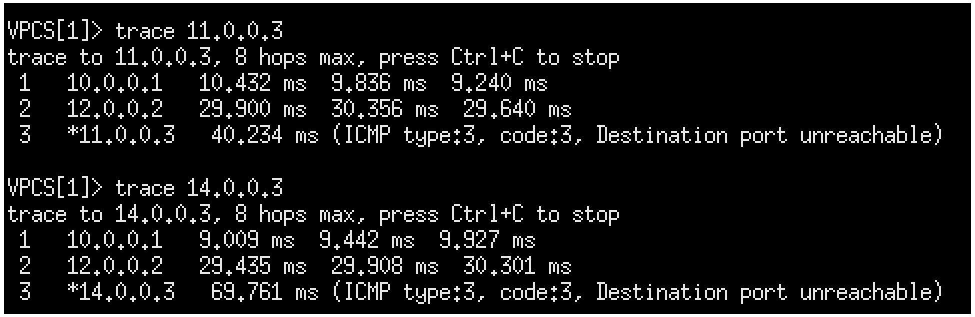
In order to Simulate the route poison from Arbor SP add:

1.- Router TMS has static route to 11.0.0.2 but not for 11.0.0.3

2.- Router TMS has static route to 14.0.0.2 but not for 14.0.0.3

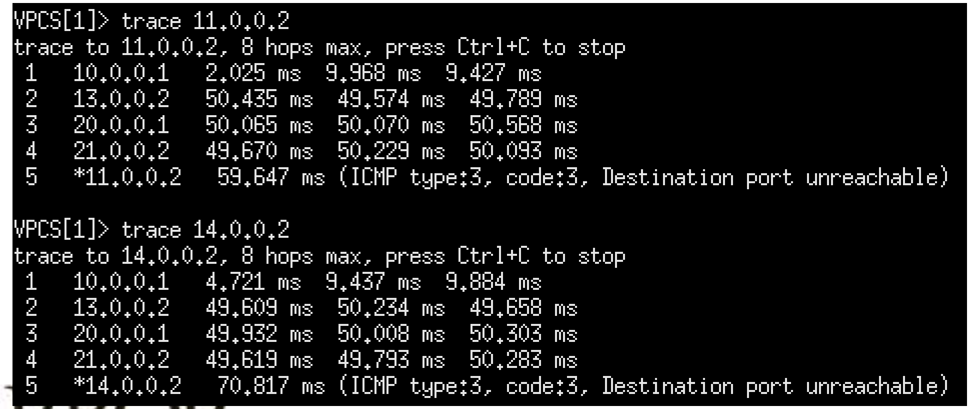
Test: traceroute from any Peer to Host:

- Results for non-poisoned Host:



```
VPCS[1]> trace 11.0.0.3
trace to 11.0.0.3, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   10.432 ms  9.836 ms  9.240 ms
 2   12.0.0.2   29.900 ms  30.356 ms  29.640 ms
 3   *11.0.0.3   40.234 ms (ICMP type:3, code:3, Destination port unreachable)

VPCS[1]> trace 14.0.0.3
trace to 14.0.0.3, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   9.009 ms  9.442 ms  9.927 ms
 2   12.0.0.2   29.435 ms  29.908 ms  30.301 ms
 3   *14.0.0.3   69.761 ms (ICMP type:3, code:3, Destination port unreachable)
```

- Results for poisoned Hosts:

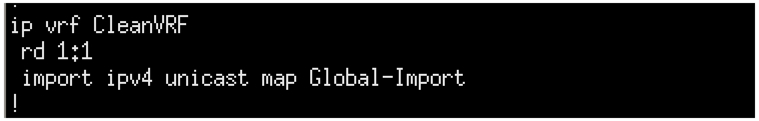


```
VPCS[1]> trace 11.0.0.2
trace to 11.0.0.2, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   2.025 ms  9.968 ms  9.427 ms
 2   13.0.0.2   50.435 ms  49.574 ms  49.789 ms
 3   20.0.0.1   50.065 ms  50.070 ms  50.568 ms
 4   21.0.0.2   49.670 ms  50.229 ms  50.093 ms
 5   *11.0.0.2   59.647 ms (ICMP type:3, code:3, Destination port unreachable)

VPCS[1]> trace 14.0.0.2
trace to 14.0.0.2, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   4.721 ms  9.437 ms  9.884 ms
 2   13.0.0.2   49.609 ms  50.234 ms  49.658 ms
 3   20.0.0.1   49.932 ms  50.008 ms  50.303 ms
 4   21.0.0.2   49.619 ms  49.793 ms  50.283 ms
 5   *14.0.0.2   70.817 ms (ICMP type:3, code:3, Destination port unreachable)
```

Cisco IOS – Dynamic Route Leaking

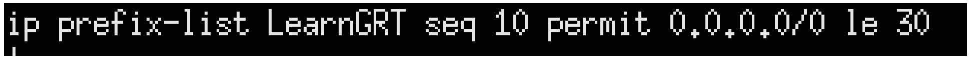
Special Configuration in Cisco IOS (Access):

1.- In CleanVRF import GRT with a route policy:



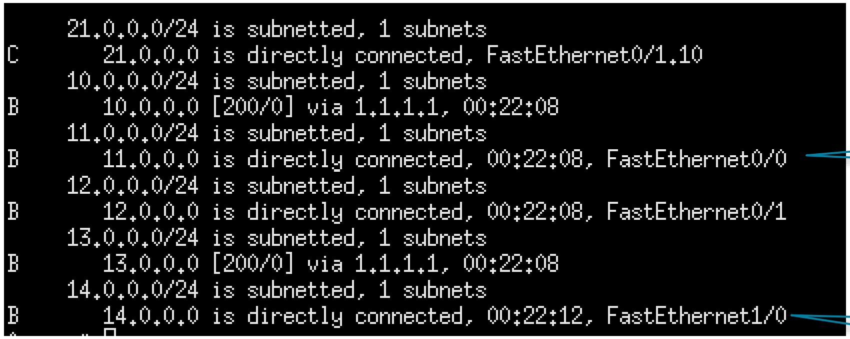
```
ip vrf CleanVRF
 rd 1:1
 import ipv4 unicast map Global-Import
!
```

2.- Create a Route Policy to ignore announces from TMS (anything longer than /30):



```
ip prefix-list LearnGRT seq 10 permit 0.0.0.0/0 le 30
```

3.- Routing table in Access Router CleanVRF:



```
     21.0.0.0/24 is subnetted, 1 subnets
C       21.0.0.0 is directly connected, FastEthernet0/1.10
     10.0.0.0/24 is subnetted, 1 subnets
B       10.0.0.0 [200/0] via 1.1.1.1, 00:22:08
     11.0.0.0/24 is subnetted, 1 subnets
B       11.0.0.0 is directly connected, 00:22:08, FastEthernet0/0
     12.0.0.0/24 is subnetted, 1 subnets
B       12.0.0.0 is directly connected, 00:22:08, FastEthernet0/1
     13.0.0.0/24 is subnetted, 1 subnets
B       13.0.0.0 [200/0] via 1.1.1.1, 00:22:08
     14.0.0.0/24 is subnetted, 1 subnets
B       14.0.0.0 is directly connected, 00:22:12, FastEthernet1/0
```

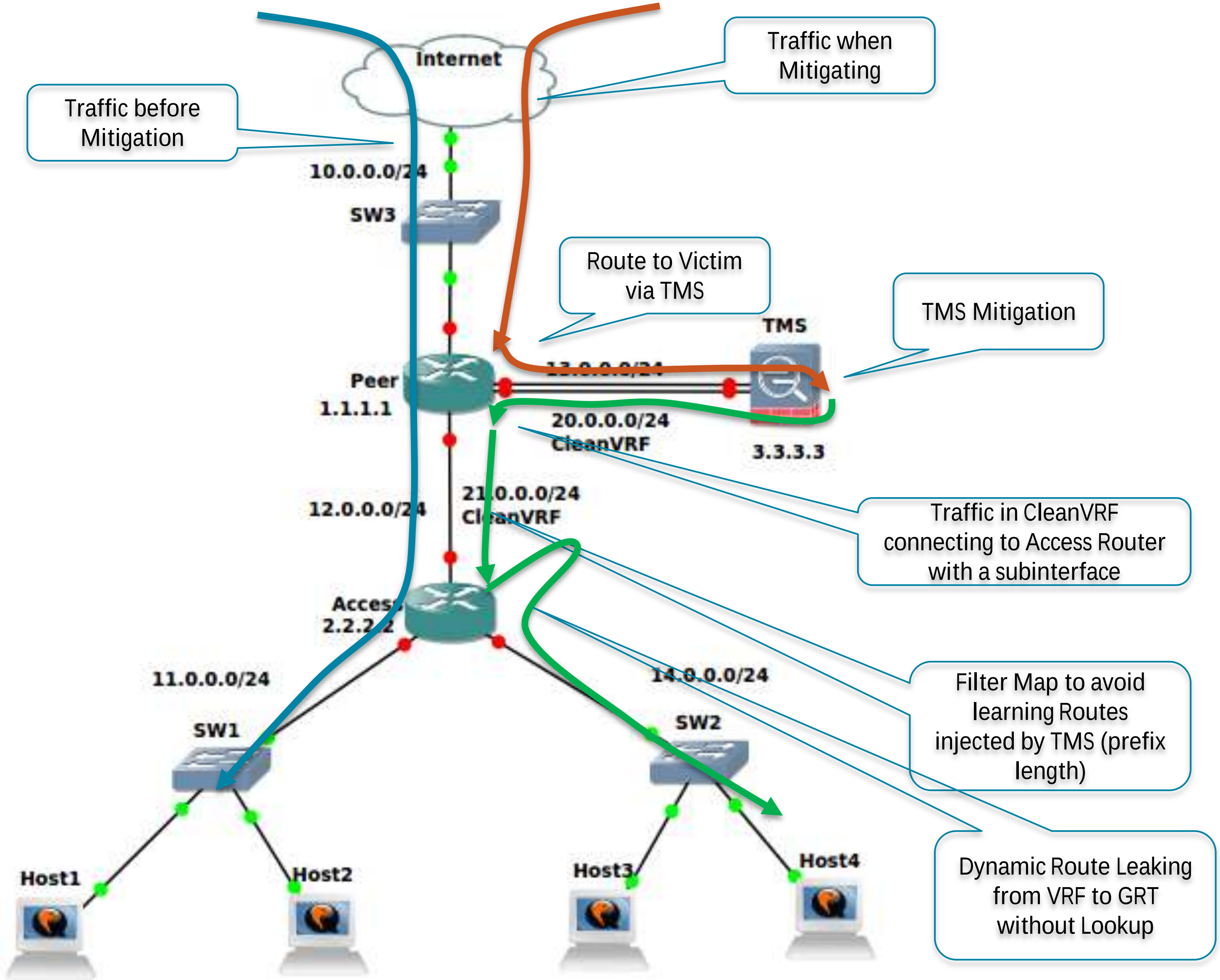
Interface in GRT

Interface in GRT

3.- Full Lab in https://arbor.box.com/Mitigation-Labs/DynamicRouterLeaking.tar.gz

Cisco IOS-XR – Dynamic Route Leaking

Cisco IOS-XR – Dynamic Route Leaking



Traffic before Mitigation

Traffic when Mitigating

Route to Victim via TMS

TMS Mitigation

Traffic in CleanVRF connecting to Access Router with a subinterface

Filter Map to avoid learning Routes injected by TMS (prefix length)

Dynamic Route Leaking from VRF to GRT without Lookup

Cisco IOS-XR – Dynamic Route Leaking

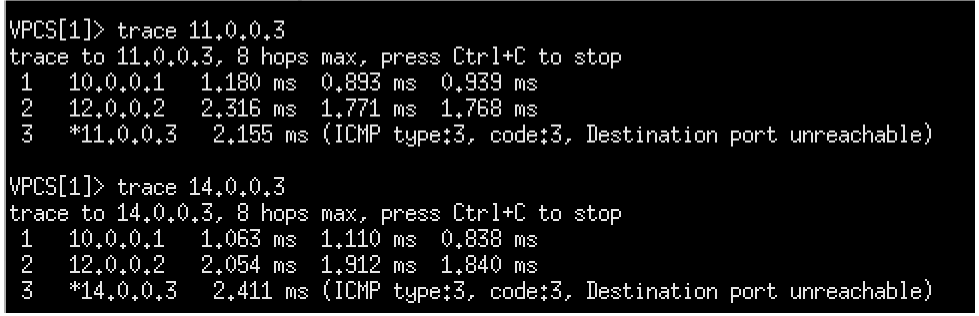
In order to Simulate the route poison from Arbor SP add:

1.- Router TMS has static route to 11.0.0.2 but not for 11.0.0.3

2.- Router TMS has static route to 14.0.0.2 but not for 14.0.0.3

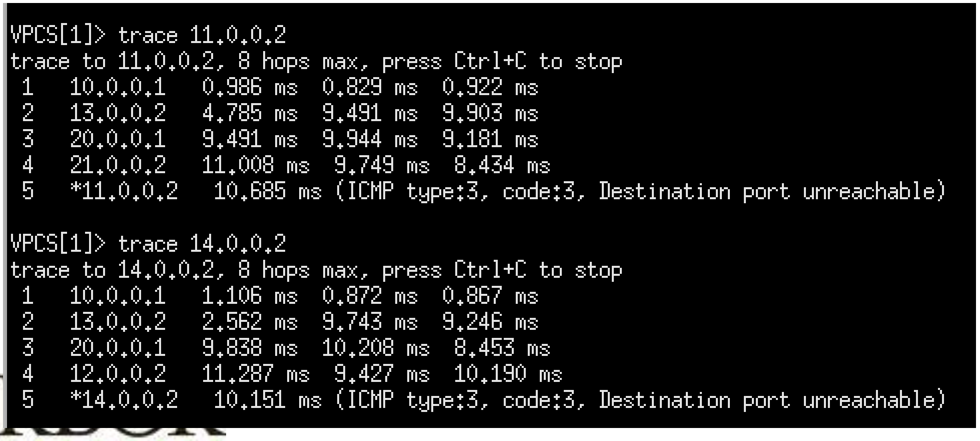
Test: traceroute from any Peer to Host:

- Results for non-poisoned Host:



```
VPCS[1]> trace 11.0.0.3
trace to 11.0.0.3, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   1.180 ms  0.893 ms  0.939 ms
 2   12.0.0.2   2.316 ms  1.771 ms  1.768 ms
 3   *11.0.0.3   2.155 ms (ICMP type:3, code:3, Destination port unreachable)

VPCS[1]> trace 14.0.0.3
trace to 14.0.0.3, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   1.063 ms  1.110 ms  0.838 ms
 2   12.0.0.2   2.054 ms  1.912 ms  1.840 ms
 3   *14.0.0.3   2.411 ms (ICMP type:3, code:3, Destination port unreachable)
```

- Results for poisoned Hosts:

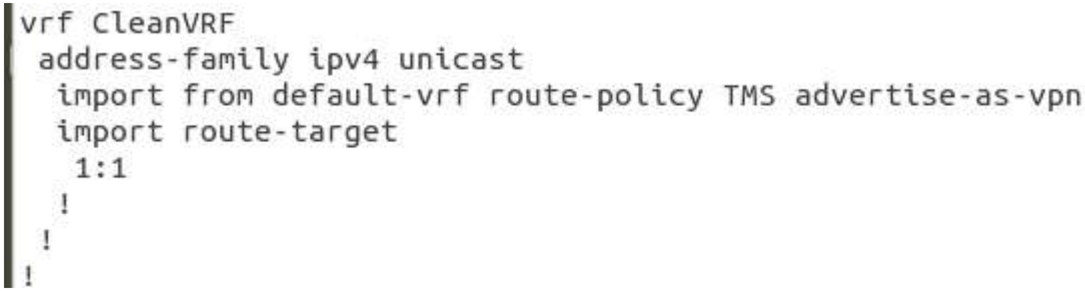


```
VPCS[1]> trace 11.0.0.2
trace to 11.0.0.2, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   0.986 ms  0.829 ms  0.922 ms
 2   13.0.0.2   4.785 ms  9.491 ms  9.903 ms
 3   20.0.0.1   9.491 ms  9.944 ms  9.181 ms
 4   21.0.0.2   11.008 ms  9.749 ms  8.434 ms
 5   *11.0.0.2   10.685 ms (ICMP type:3, code:3, Destination port unreachable)

VPCS[1]> trace 14.0.0.2
trace to 14.0.0.2, 8 hops max, press Ctrl+C to stop
 1   10.0.0.1   1.106 ms  0.872 ms  0.867 ms
 2   13.0.0.2   2.562 ms  9.743 ms  9.246 ms
 3   20.0.0.1   9.838 ms  10.208 ms  8.453 ms
 4   12.0.0.2   11.287 ms  9.427 ms  10.190 ms
 5   *14.0.0.2   10.151 ms (ICMP type:3, code:3, Destination port unreachable)
```

Cisco IOS-XR – Dynamic Route Leaking

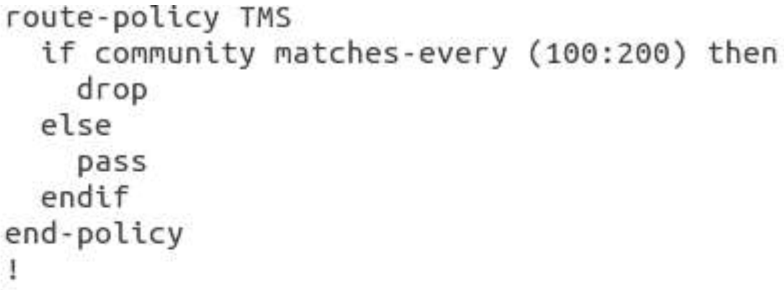
Special Configuration in Cisco IOS-XR (Peer and Access):

1.- In CleanVRF import GRT with a route policy:



```
vrf CleanVRF
 address-family ipv4 unicast
  import from default-vrf route-policy TMS advertise-as-vpn
  import route-target
   1:1
  !
 !
!
```

2.- Create a Route Policy to ignore announces from TMS (By community):

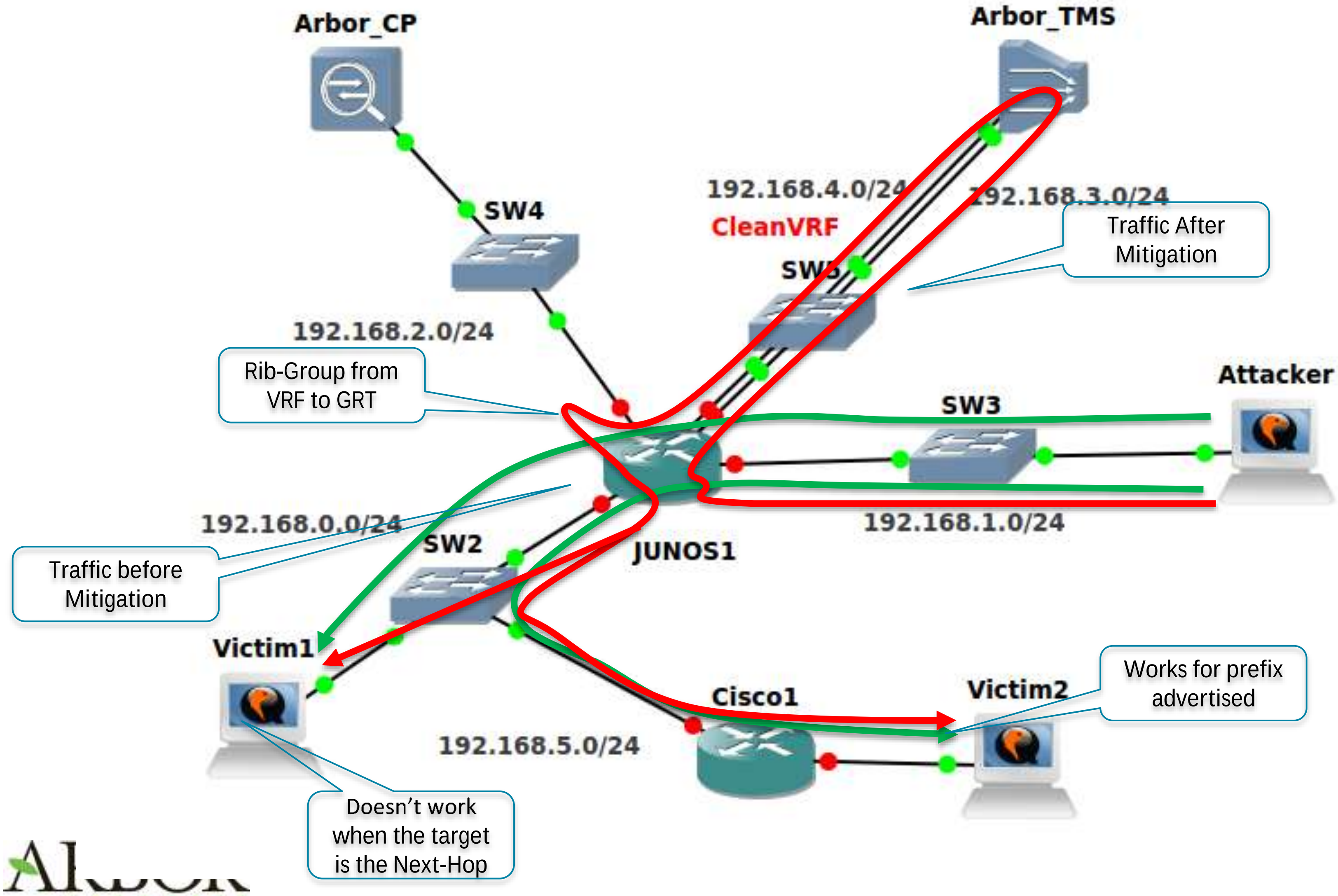


```
route-policy TMS
  if community matches-every (100:200) then
    drop
  else
    pass
  endif
end-policy
!
```

3.- Full Lab in https://arbor.box.com/Mitigation-Labs/DynamicRouterLeaking.tar.gz

Juniper – Rib Groups

Juniper – Rib Groups



Traffic After Mitigation

Rib-Group from VRF to GRT

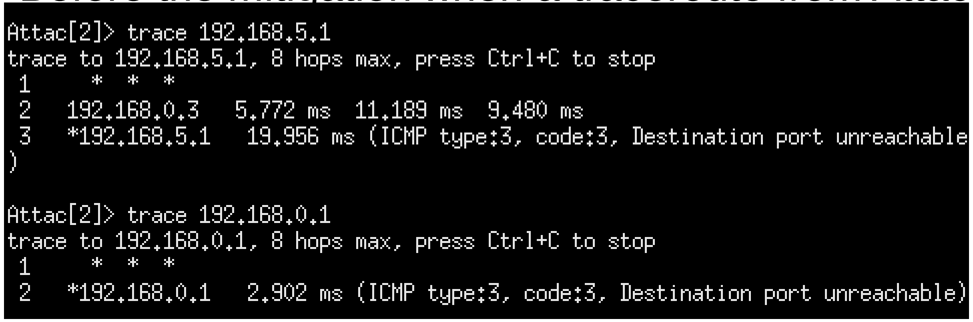
Traffic before Mitigation

Works for prefix advertised

Doesn't work when the target is the Next-Hop

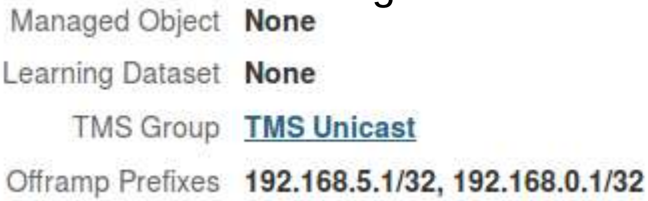
Juniper – Rib Groups

1.- Before the mitigation when a traceroute from Attacker to Victims:



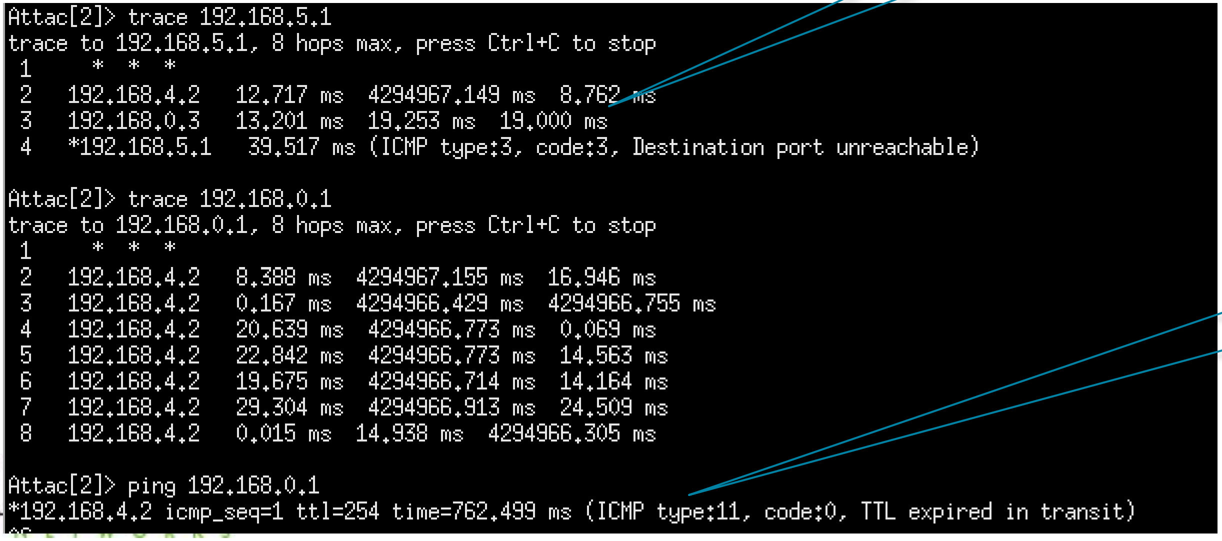
```
Attac[2]> trace 192.168.5.1
trace to 192.168.5.1, 8 hops max, press Ctrl+C to stop
 1     *  *  *
 2   192.168.0.3   5.772 ms  11.189 ms  9.480 ms
 3   *192.168.5.1   19.956 ms (ICMP type:3, code:3, Destination port unreachable
)

Attac[2]> trace 192.168.0.1
trace to 192.168.0.1, 8 hops max, press Ctrl+C to stop
 1     *  *  *
 2   *192.168.0.1   2.902 ms (ICMP type:3, code:3, Destination port unreachable)
```

2.- From CP start a mitigation to both Victims



Managed Object	None
Learning Dataset	None
TMS Group	TMS Unicast
Offramp Prefixes	192.168.5.1/32, 192.168.0.1/32

3.- Check again with traceroute:



```
Attac[2]> trace 192.168.5.1
trace to 192.168.5.1, 8 hops max, press Ctrl+C to stop
 1     *  *  *
 2   192.168.4.2   12.717 ms  4294967.149 ms  8.762 ms
 3   192.168.0.3   13.201 ms  19.253 ms  19.000 ms
 4   *192.168.5.1   39.517 ms (ICMP type:3, code:3, Destination port unreachable)

Attac[2]> trace 192.168.0.1
trace to 192.168.0.1, 8 hops max, press Ctrl+C to stop
 1     *  *  *
 2   192.168.4.2   8.388 ms  4294967.155 ms  16.946 ms
 3   192.168.4.2   0.167 ms  4294966.429 ms  4294966.755 ms
 4   192.168.4.2   20.639 ms  4294966.773 ms  0.069 ms
 5   192.168.4.2   22.842 ms  4294966.773 ms  14.563 ms
 6   192.168.4.2   19.675 ms  4294966.714 ms  14.164 ms
 7   192.168.4.2   29.304 ms  4294966.913 ms  24.509 ms
 8   192.168.4.2   0.015 ms  14.938 ms  4294966.305 ms

Attac[2]> ping 192.168.0.1
*192.168.4.2 icmp_seq=1 ttl=254 time=762.499 ms (ICMP type:11, code:0, TTL expired in transit)
```

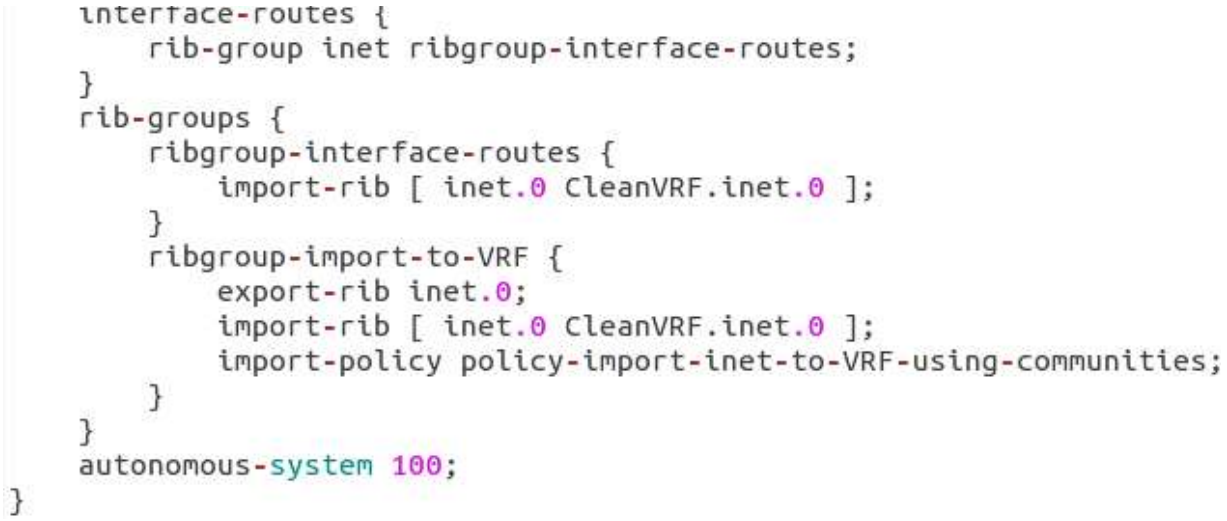
Works with Prefix behind a Router

Doesn't work if Target is directly cooneted to the Router

Juniper – Rib Groups – Configuration (1 of 2)

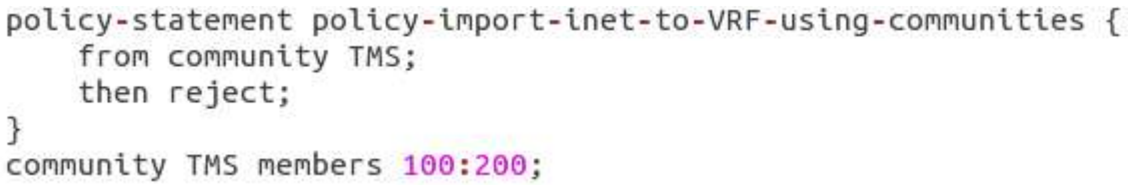
Special Configuration in Junos:

1.- Create a Rib-Group to import routes between VRF and GRT



```
    interface-routes {
        rib-group inet ribgroup-interface-routes;
    }
    rib-groups {
        ribgroup-interface-routes {
            import-rib [ inet.0 CleanVRF.inet.0 ];
        }
        ribgroup-import-to-VRF {
            export-rib inet.0;
            import-rib [ inet.0 CleanVRF.inet.0 ];
            import-policy policy-import-inet-to-VRF-using-communities;
        }
    }
    autonomous-system 100;
}
```

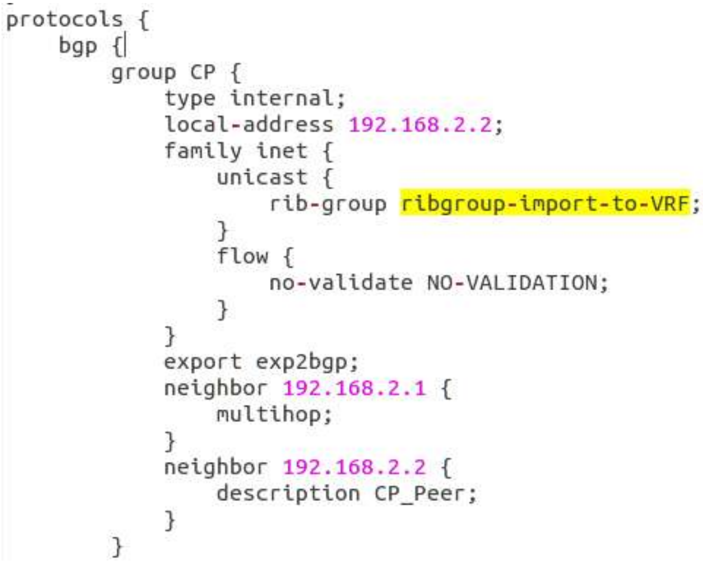
2.- Create a Route Policy to ignore announces from TMS (By community):



```
policy-statement policy-import-inet-to-VRF-using-communities {
    from community TMS;
    then reject;
}
community TMS members 100:200;
```

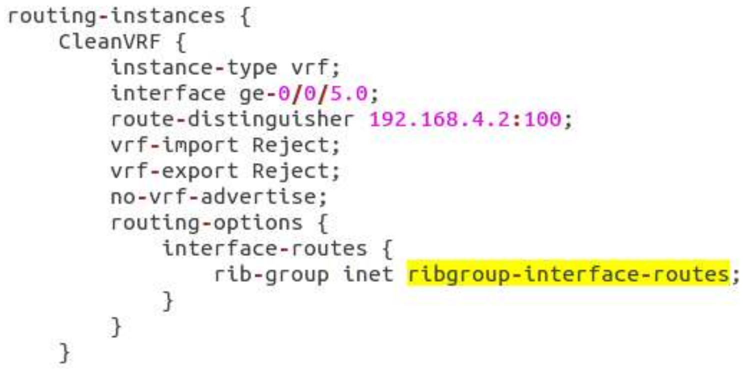
Juniper – Rib Groups – Configuration (2 of 2)

3.- Use Rib Groups in the BGP Protocol (GRT -> VRF):



```
protocols {
    bgp {
        group CP {
            type internal;
            local-address 192.168.2.2;
            family inet {
                unicast {
                    rib-group ribgroup-import-to-VRF;
                }
                flow {
                    no-validate NO-VALIDATION;
                }
            }
            export exp2bgp;
            neighbor 192.168.2.1 {
                multihop;
            }
            neighbor 192.168.2.2 {
                description CP_Peer;
            }
        }
```

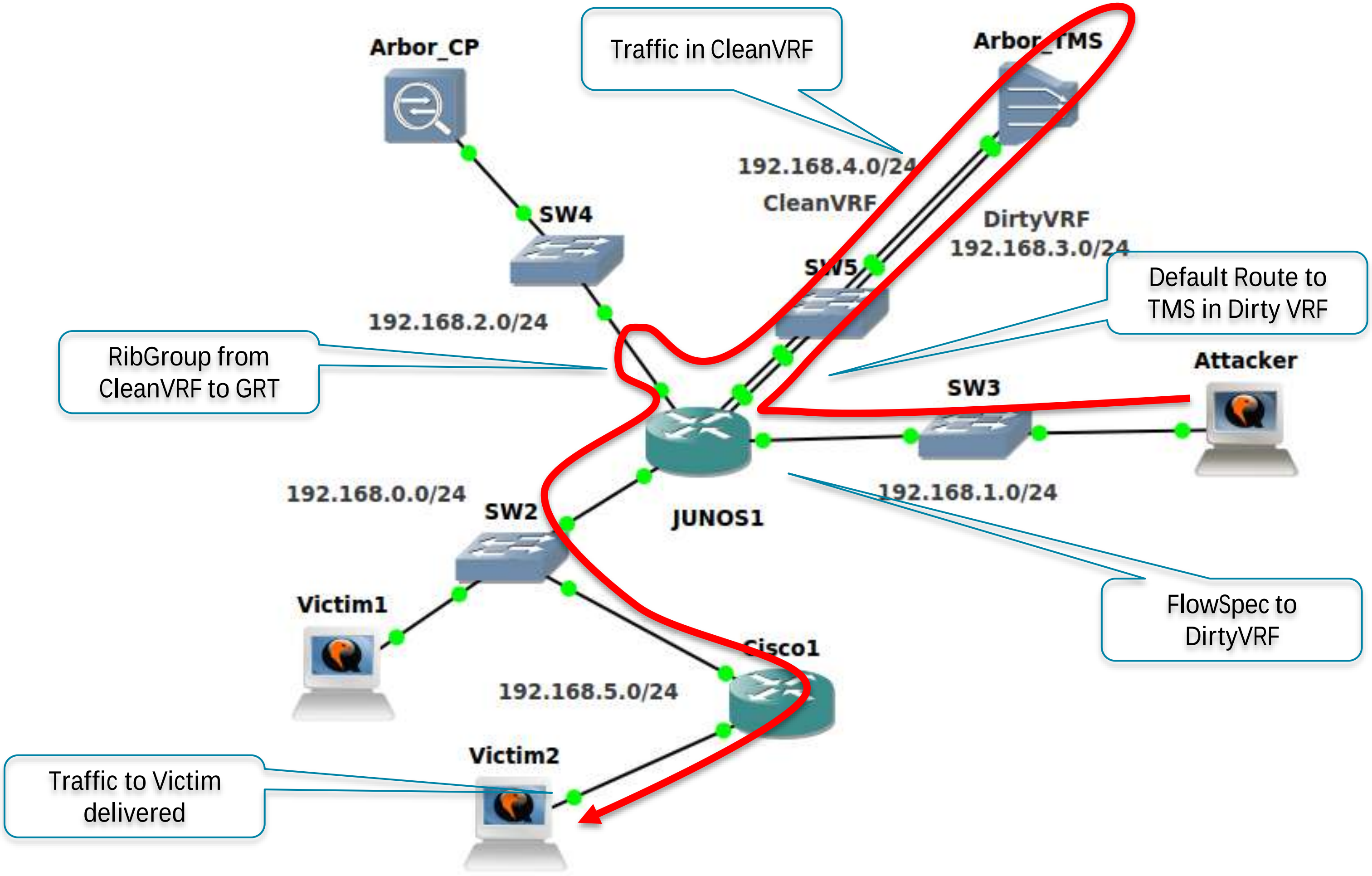
4.- Use Rib Groups in the Route Instance (VRF -> GRT):



```
routing-instances {
    CleanVRF {
        instance-type vrf;
        interface ge-0/0/5.0;
        route-distinguisher 192.168.4.2:100;
        vrf-import Reject;
        vrf-export Reject;
        no-vrf-advertise;
        routing-options {
            interface-routes {
                rib-group inet ribgroup-interface-routes;
            }
        }
    }
```

5.- Full Lab in https://arbor.box.com/Mitigation-Labs/JunosRibGroups.tar.gz

Juniper – FlowSpec



Traffic in CleanVRF

Default Route to TMS in Dirty VRF

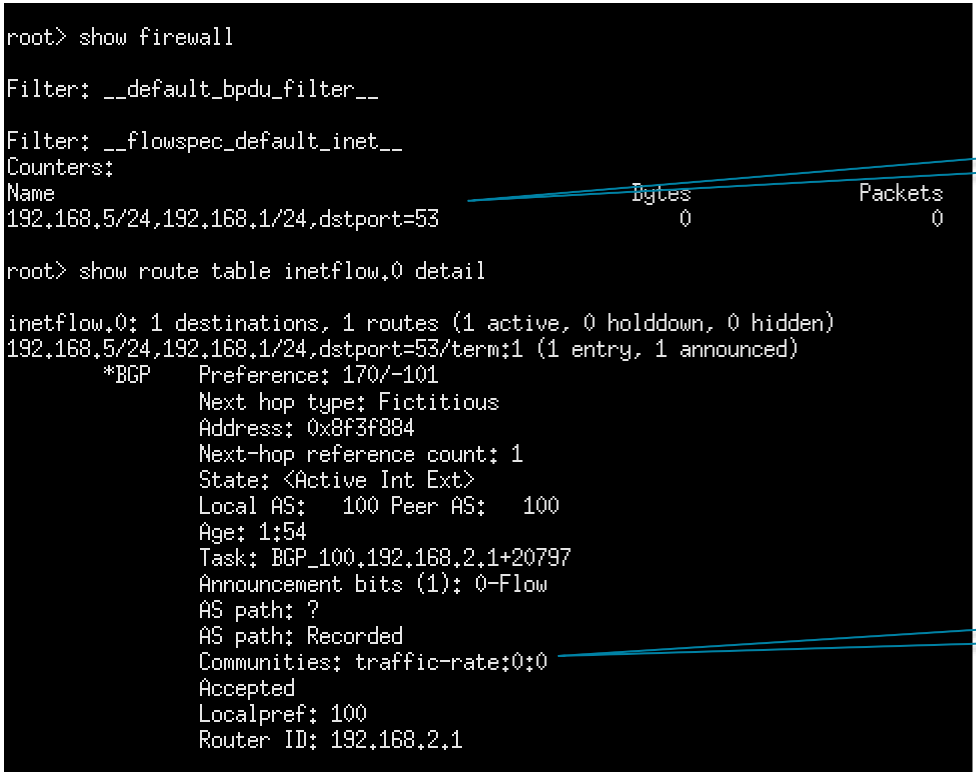
RibGroup from CleanVRF to GRT

FlowSpec to DirtyVRF

Traffic to Victim delivered

JunOS FlowSpec - drop/shape

1.- Start a flowspec mitigation dropping traffic Dst: 192.168.5.0, Src: 192.168.1.0/24, Port 53

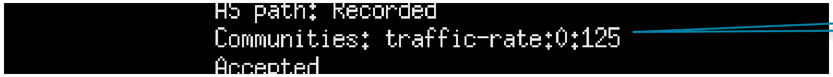


```
root> show firewall

Filter: __default_bpdu_filter__

Filter: __flowspec_default_inet__
Counters:
Name                                                Bytes              Packets
192.168.5/24,192.168.1/24,dstport=53                    0                    0

root> show route table inetflow.0 detail

inetflow.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
192.168.5/24,192.168.1/24,dstport=53/term:1 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Fictitious
                Address: 0x8f3f884
                Next-hop reference count: 1
                State: <Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 1:54
                Task: BGP_100.192.168.2.1+20797
                Announcement bits (1): 0-Flow
                AS path: ?
                AS path: Recorded
                Communities: traffic-rate:0:0
                Accepted
                Localpref: 100
                Router ID: 192.168.2.1
```

New firewall filter created

Action DROP

2.- Start a flowspec mitigation shaping traffic Dst: 192.168.5.0, Src: 192.168.1.0/24, Port 53

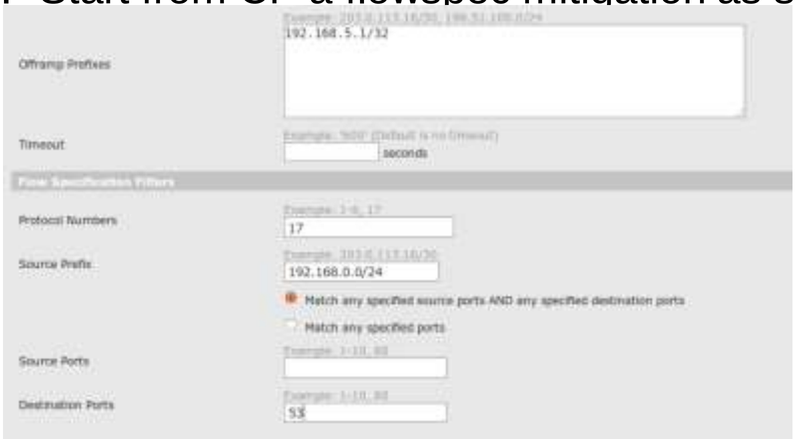


```
                AS path: Recorded
                Communities: traffic-rate:0:125
                Accepted
```

Action SHAPE

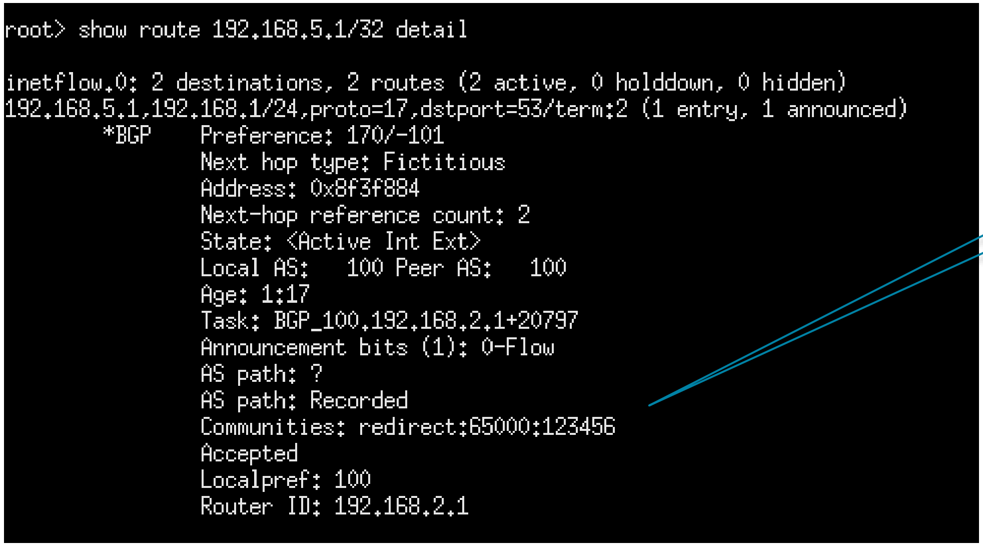
JunOS FlowSpec - redirect to TMS

3.- Start from CP a flowspec mitigation as shown:



Offramp Prefixes	Example: 203.0.113.16/30, 198.51.100.0/24 192.168.5.1/32
Timeout	Example: '600' (Default is no timeout) seconds
Flow Specification Filters	
Protocol Numbers	Example: 1-6, 17 17
Source Prefix	Example: 203.0.113.16/30 192.168.0.0/24
	● Match any specified source ports AND any specified destination ports ○ Match any specified ports
Source Ports	Example: 1-10, 80
Destination Ports	Example: 1-10, 80 53

4.- Check route to 192.168.5.1 in JunOS:

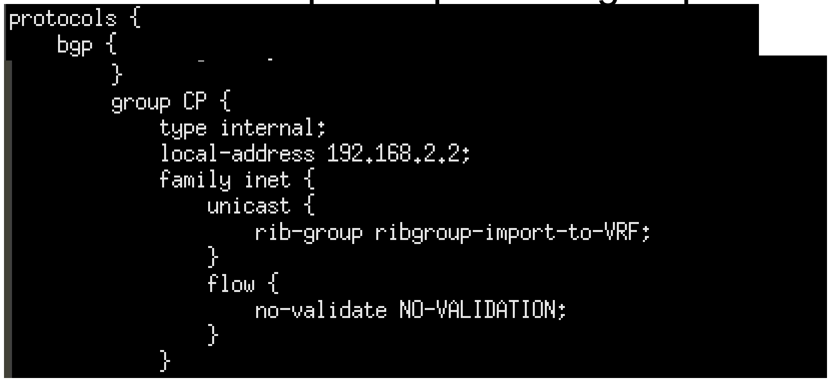


```
root> show route 192.168.5.1/32 detail

inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
192.168.5.1,192.168.1/24,proto=17,dstport=53/term:2 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Fictitious
                Address: 0x8f3f884
                Next-hop reference count: 2
                State: <Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 1:17
                Task: BGP_100.192.168.2.1+20797
                Announcement bits (1): 0-Flow
                AS path: ?
                AS path: Recorded
                Communities: redirect:65000:123456
                Accepted
                Localpref: 100
                Router ID: 192.168.2.1
```

Action Redirect to Dirty VRF

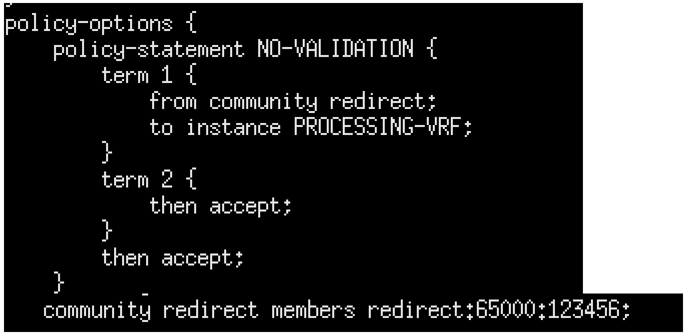
JunOS FlowSpec - Configuration

1.- Enable FlowSpec in protocol group CP:



```
protocols {
    bgp {
        }
        group CP {
            type internal;
            local-address 192.168.2.2;
            family inet {
                unicast {
                    rib-group ribgroup-import-to-VRF;
                }
                flow {
                    no-validate NO-VALIDATION;
                }
            }
```

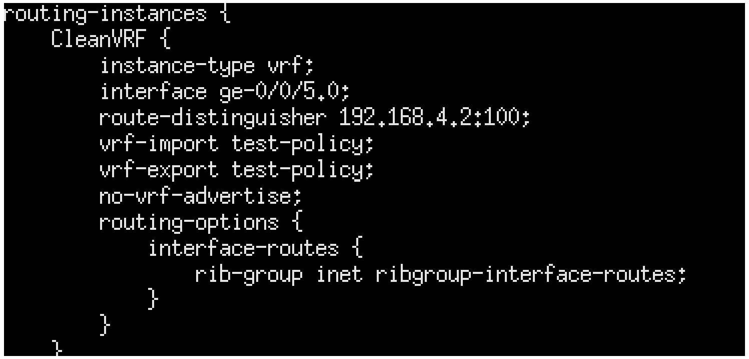
2.- Create a policy option to redirect to DirtyVRF



```
policy-options {
    policy-statement NO-VALIDATION {
        term 1 {
            from community redirect;
            to instance PROCESSING-VRF;
        }
        term 2 {
            then accept;
        }
        then accept;
    }
   community redirect members redirect:65000:123456;
```

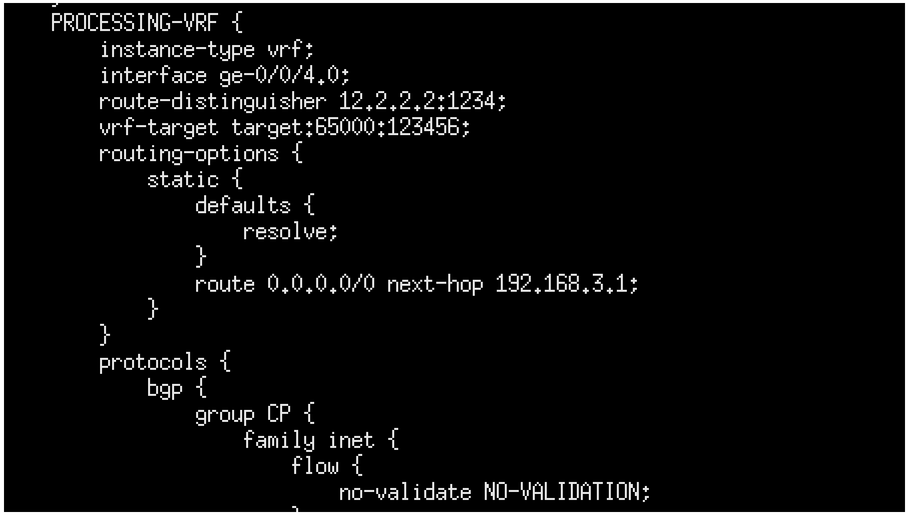
JunOS FlowSpec - Configuration

3.- Create Clean VRF with RibGroup to deliver to GRT



```
routing-instances {
    CleanVRF {
        instance-type vrf;
        interface ge-0/0/5.0;
        route-distinguisher 192.168.4.2:100;
        vrf-import test-policy;
        vrf-export test-policy;
        no-vrf-advertise;
        routing-options {
            interface-routes {
                rib-group inet ribgroup-interface-routes;
            }
        }
    }
```

5.- Create DirtyVRF (route-target, default route to TMS, flow with CP):



```
    PROCESSING-VRF {
        instance-type vrf;
        interface ge-0/0/4.0;
        route-distinguisher 12.2.2.2:1234;
        vrf-target target:65000:123456;
        routing-options {
            static {
                defaults {
                    resolve;
                }
                route 0.0.0.0/0 next-hop 192.168.3.1;
            }
        }
        protocols {
            bgp {
                group CP {
                    family inet {
                        flow {
                            no-validate NO-VALIDATION;
```

3.- Full Lab in https://arbor.box.com/Mitigation-Labs/JunosFlowSpec.tar.gz

FlowSpec - Conclusions

1.- Juniper SRX does not support FlowSpec in neither physical or virtual routers.

2.- Juniper M-Series supports everything and was tested in latest version.

3.- Virtual Juniper M-Series do not support Flowspec. I expect to have a new version in middle February and it should be supported.

4.- Cisco IOS-XR version 5.2.2 should support Flowspec, but I haven't try it in ASR or CRS.

5.- Cisco Virtual XR 5.2.2 does not support Flowspec

ARBOR®
NETWORKS

Thank You