

MPLS

Multi Protocol Label Switching

1. Pendahuluan

MPLS kepanjangan dari Multi Protocol Label Switching adalah sebuah teknologi terbaru pada Jaringan WAN yang memungkinkan kita membuat sebuah jaringan backbone dengan kecepatan tinggi. Jaringan WAN yang menerapkan MPLS tidak lagi menggunakan IP Address maupun MAC Address dalam pengiriman data, tapi MPLS menggunakan sebuah label identifikasi yang diletakan di antara header Layer3 dan Layer2. Setiap Network yang ada pada tiap Router akan memiliki label identifikasi berupa angka desimal, label ini bersifat lokal hanya pada Router itu saja jika pada Router lain Network tadi nilai labelnya sudah berubah. Label dapat dikonfigurasi manual atau dengan bantuan LDP (Label Distribution Protocol) untuk mendistribusikannya dari Router ke Router, LDP juga memerlukan protokol Routing IGP seperti (RIP, OSPF, EIGRP) untuk mendistribusikan Network-Network dari Router ke Router.

Kinerja LDP adalah menentukan Label secara lokal pada tiap Network yang ada pada Router lalu label dan Network tersebut akan didistribusikan pada Router lain yang terdekat, Router yang menerima tadi akan melakukan hal yang sama seperti Router sebelumnya dan kembali mendistribusikan label yang baru pada Router lain yang terdekat.

MPLS terdapat beberapa macam Router:

- CE : (Customer Edge) adalah Router non-MPLS yang ingin terhubung dengan jaringan Backbone MPLS.
- PE : (Provider Edge) adalah Router MPLS yang menghubungkan antara Jaringan non-MPLS dengan Jaringan MPLS, Router ini yang melakukan peletakan label (insert) untuk ingress atau penghapusan label (pop) untuk egress.
- P : (Provider) adalah Router MPLS yang berada ditengah-tengah jaringan MPLS, Router ini yang melakukan pergantian label (swap).



Konfigure IP address

IP Address	
R1(config)#int e0/0	R3(config)#int e0/0
R1(config-if)#ip add 12.12.12.1 255.255.255.0	R3(config-if)#ip add 23.23.23.3 255.255.255.0
R1(config-if)#no sh	R3(config-if)#no sh
R1(config-if)#exi	R3(config-if)#exi
R1(config)#int e0/1	R3(config)#int e0/1
R1(config-if)#ip add 14.14.14.1 255.255.255.0	R3(config-if)#ip add 35.35.35.3 255.255.255.0
R1(config-if)#no sh	R3(config-if)#no sh
R1(config-if)#exi	R3(config-if)#exi
R1(config)#int lo0	R3(config)#int lo0
R1(config-if)#ip add 1.1.1.1 255.255.255.255	R3(config-if)#ip add 3.3.3.3 255.255.255.255
R1(config-if)#exi	R3(config-if)#exi
R2(config)#int e0/0	R4(config)#int e0/0
R2(config-if)#ip add 12.12.12.2 255.255.255.0	R4(config-if)#ip add 14.14.14.4 255.255.255.0
R2(config-if)#no sh	R4(config-if)#no sh
R2(config-if)#exi	R4(config-if)#ex
R2(config)#int e0/1	R4(config)#int lo0
R2(config-if)#ip add 23.23.23.2 255.255.255.0	R4(config-if)#ip add 4.4.4.4 255.255.255.255
R2(config-if)#no sh	R4(config-if)#exi
R2(config-if)#exi	R5(config)#int e0/0
R2(config)#int lo0	R5(config-if)#ip add 35.35.35.5 255.255.255.0
R2(config-if)#ip add 2.2.2.2 255.255.255.255	R5(config-if)#no sh
R2(config-if)#exit	R5(config-if)#exi
	R5(config)#int lo0
	R5(config-if)#ip add 5.5.5.5 255.255.255.255
	R5(config-if)#exit

Routing Protocol IGP with EIGRP	MPLS LDP
R1(config)#router eigrp 1	Konfigurasi MPLS LDP pada interface yang terhubung dengan MPLS backbone.
R1(config-router)#no auto-sum	R1(config)#int e0/0
R1(config-router)#net 1.1.1.1 0.0.0.0	R1(config-if)#mpls ip
R1(config-router)#net 12.12.12.0 0.0.0.255	R1(config-if)#exit
R1(config-router)#net 14.14.14.0 0.0.0.255	R2(config)#int ra e0/0-1
R1(config-router)#exit	R2(config-if-range)#mpls ip
R2(config)#router eigrp 1	R2(config-if-range)#exi
R2(config-router)#no auto-sum	R3(config)#int e0/0
R2(config-router)#net 2.2.2.2 0.0.0.0	R3(config-if)#mpls ip
R2(config-router)#net 12.12.12.0 0.0.0.255	R3(config-if)#exit
R2(config-router)#net 23.23.23.0 0.0.0.255	
R2(config-router)#exit	
R3(config)#router eigrp 1	
R3(config-router)#no auto-sum	
R3(config-router)#net 3.3.3.3 0.0.0.0	
R3(config-router)#net 35.35.35.0 0.0.0.255	
R3(config-router)#net 23.23.23.0 0.0.0.255	
R3(config-router)#exit	
R4(config)#router eigrp 1	
R4(config-router)#no auto-sum	
R4(config-router)#net 4.4.4.4 0.0.0.0	
R4(config-router)#net 14.14.14.0 0.0.0.255	
R4(config-router)#exit	
R5(config)#router eigrp 1	
R5(config-router)#no auto-sum	
R5(config-router)#net 5.5.5.5 0.0.0.0	
R5(config-router)#net 35.35.35.0 0.0.0.255	
R5(config-router)#exit	

Lakukan Show

Untuk melihat MPLS Forwarding table yang isinya merupakan label label untuk menuju suatu Network pada tiap Router yang ada di MPLS Backbone.

Show Forwarding MPLS					
R1#sh mpls forwarding-table					
Local Label	Outgoing Label	Prefix or Tunnel Id	Bytes Label Switched	Outgoing interface	Next Hop
16	Pop Label	2.2.2.2/32	0	Et0/0	12.12.12.2
17	Pop Label	23.23.23.0/24	0	Et0/0	12.12.12.2
18	17	3.3.3.3/32	0	Et0/0	12.12.12.2
19	21	35.35.35.0/24	0	Et0/0	12.12.12.2
20	No Label	4.4.4.4/32	0	Et0/1	14.14.14.4
21	19	5.5.5.5/32	0	Et0/0	12.12.12.2

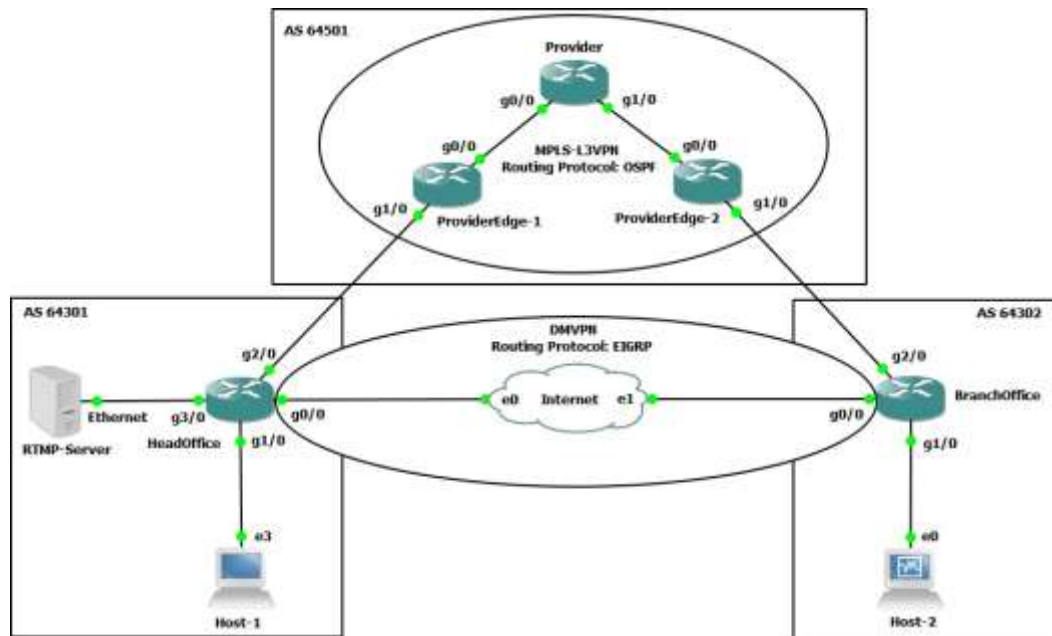
Show Forwarding MPLS					
R2#sh mpls forwarding-table					
Local Label	Outgoing Label	Prefix or Tunnel Id	Bytes Label Switched	Outgoing interface	Next Hop
16	Pop Label	1.1.1.1/32	0	Et0/0	12.12.12.1
17	Pop Label	3.3.3.3/32	0	Et0/1	23.23.23.3
18	20	4.4.4.4/32	0	Et0/0	12.12.12.1
19	19	5.5.5.5/32	1180	Et0/1	23.23.23.3
20	Pop Label	14.14.14.0/24	1140	Et0/0	12.12.12.1
21	Pop Label	35.35.35.0/24	0	Et0/1	23.23.23.3

Show Forwarding MPLS					
R3#sh mpls forwarding-table					
Local Label	Outgoing Label	Prefix or Tunnel Id	Bytes Label Switched	Outgoing interface	Next Hop
16	16	1.1.1.1/32	0	Et0/0	23.23.23.2
17	Pop Label	2.2.2.2/32	0	Et0/0	23.23.23.2
18	18	4.4.4.4/32	0	Et0/0	23.23.23.2
19	No Label	5.5.5.5/32	1140	Et0/1	35.35.35.5
20	Pop Label	12.12.12.0/24	0	Et0/0	23.23.23.2
21	20	14.14.14.0/24	0	Et0/0	23.23.23.2

1.1 Topologi

Dalam penelitian ini, peneliti akan membuat dua model jaringan dalam pengujian skenario *failover VPN* dan menganalisis performa redundansi dari penggunaan masing-masing model teknologi VPN. Pada model jaringan pertama, yaitu *MPLS* berbasis *VPN service*. Desain lain yaitu *DMVPN* yang diasumsikan melalui *Internet*. Pada desain kantor pusat, *router* yang telah terhubung dengan *MPLS* sebagai media VPN. Di sisi kantor cabang, *router* digunakan hal yang sama dengan kantor pusat sehingga satu sama lain dapat berkomunikasi melalui *MPLS*. Pada topologi kedua, penambahan mitigasi pada *MPLS-L3VPN* sebagai media jalur utamanya adalah *DMVPN* yang diterhubung dengan *Internet* sebagai media jalur

cadangannya. Dalam topologi simulasi penelitian, masing-masing skema jaringan dapat ditunjukkan pada Gambar 1.1.



Gambar 1. 1 Topologi Simulasi *Failover*

Skenario simulasi yang dibangun akan dibuat secara realistis sedemikian rupa yang dirancang ke dalam *software* GNS3. Rancangan *MPLS-L3VPN* dibuat terpisah dengan rancangan *DMVPN*. Rancangan pertama dibangun dengan satu *router provider* dan dua *router edge provider*, sehingga dapat mengimplementasi *MPLS-L3VPN*. Sedangkan rancangan *DMVPN* yang diasumsikan terhubung dengan *internet* sehingga dibutuhkan satu *switch* sebagai media bridge antara dua *router customer edge* yang kedua *router* dibangun di dalam infrastruktur *customer*. Sistem akan dibuat dengan memiliki dua sisi, yaitu kantor pusat dan kantor cabang dengan masing-masing kantor dibangun satu *router* yang terhubung ke *MPLS* dan *DMVPN* serta perangkat host sebagai pengujian sistem.

Host atau *customer* akan digunakan sebagai pengujian konektifitas dari satu sisi ke sisi lain dan mengkalkulasikan waktu *route* yang terjadi ketika jalur *VPN* utama terputus berpindah ke *VPN* cadangan. Perhitungan diambil dari parameter yang terpantau oleh *software* Wireshark dan hasil nilai tersebut mereferensikan pada standar ITU.

1.2 Perancangan Sistem

Perancangan sistem merupakan tahap kedua dalam penerapan yang akan dirancang. *Routing* yang digunakan mereferensikan pada penelitian sebelumnya yang membahas terkait performa kualitas penggunaan protokol *routing*. Pada jaringan *MPLS*, *routing* yang digunakan adalah *OSPF* (Arini, Masruro, & Rizal, 2018) dan jaringan *DMVPN* menggunakan *EIGRP* (Kakulapati & Sandhya, Establishing Secured Enterprise Network, 2018). Selain itu, untuk menghubungkan *AS* dari *customer* dengan *provider* dalam suatu sistem jaringan *internet* diasumsikan menggunakan *eBGP* dengan *AS* 64501 untuk *provider*, *AS* 64301 untuk kantor pusat dan *AS* 64302 untuk kantor cabang.

Daftar *interface* serta *IP Address* yang akan diterapkan adalah sebagai berikut:

Tabel 1. 1 Konfigurasi *IP Address* dari masing-masing *interface router customer*

Device	Gi0/0	Gi1/0	Gi2/0	Gi3/0	Tunnel0
Head Office	172.16.90.1 /24	192.168.1. 1/24	10.0.0.6/ 30	192.168.43. 1/24	10.10.10.1 /24
Branch Office	172.16.90.2 /24	192.168.2. 1/24	10.0.0.22 /30	-	10.10.10.2 /24

Tabel 1. 2 Konfigurasi *IP Address* dari masing-masing *interface router provider*

Device	Loopback0	Gi0/0	Gi1/0
Provider	10.1.1.2/32	10.0.0.10/30	10.0.0.14/30
ProviderEdge-1	10.1.1.1/32	10.0.0.9/30	10.0.0.5/30
ProviderEdge-2	10.1.1.3/32	10.0.0.13/30	10.0.0.21/30

Tabel 1. 3 Konfigurasi *IP Address* dari masing-masing End device

Device	Interface	IP Address
Host-1	Ethernet3	192.168.1.10/24
Host-2	Ethernet0	192.168.2.10/24
RTMP-Server	Ethernet	192.168.43.10/24

Langkah pertama implementasi jaringan yaitu memberikan *IP Address* pada *interface router* dan host. Setelah *IP Address* dikonfigurasi pada masing-masing interface, maka langkah selanjutnya melakukan tes verifikasi ICMP pada koneksi *peer-to-peer* antara *router* Provider ke Provider Edge, *router* Provider ke Customer Edge (Headoffice dan BranchOffice), dan Customer Edge ke host (Host-1, Host-2 dan *RTMP Server*) serta koneksi antara *router* Customer Edge melalui *internet* sudah dapat terhubung.

Jaringan Provider yang menjembatani antara kedua jaringan *customer* tidak dapat melakukan *redistribute routing* dinamikanya jika tidak mengaktifkan *BGP connection* dari kedua *router* tersebut. Mengaktifkan *routing* dinamik untuk membuat *cloud MPLS*, semua *router* harus dapat men-forward *packet* agar *router* dapat established dengan jaringan luar dan dapat berkembang. Implementasi *BGP* pada simulasi ini mengadaptasikan eksternal *BGP*, karena *AS* yang digunakan berbeda antara Provider dan Customer.

Konfigurasi di Provider Edge - 1:

```
router bgp 64501
  bgp log-neighbor-changes neighbor 10.1.1.3 remote-
  as 64501
  neighbor 10.1.1.3 update-source Loopback0
```

Konfigurasi di Provider Edge - 2:

```
router bgp 64501
  bgp log-neighbor-changes neighbor 10.1.1.1 remote-
  as 64501
  neighbor 10.1.1.1 update-source Loopback0
```

Konfigurasi di Customer Edge - 1 (Head Office):

```
router bgp 64301
  bgp log-neighbor-changes network
  192.168.1.0
  timers bgp 10 30
  neighbor 10.0.0.5 remote-as 64501
```

Konfigurasi di Customer Edge - 2 (Branch Office):

```
router bgp 64302
  bgp log-neighbor-changes network
  192.168.2.0
  timers bgp 10 30
  neighbor 10.0.0.21 remote-as 64501
```

Untuk memverifikasikan koneksi antara *BGP* yang ada di *router* Provider dan Customer sudah established atau tidak, dapat dilakukan pemeriksaan pada salah satu *router* Provider seperti berikut ini:

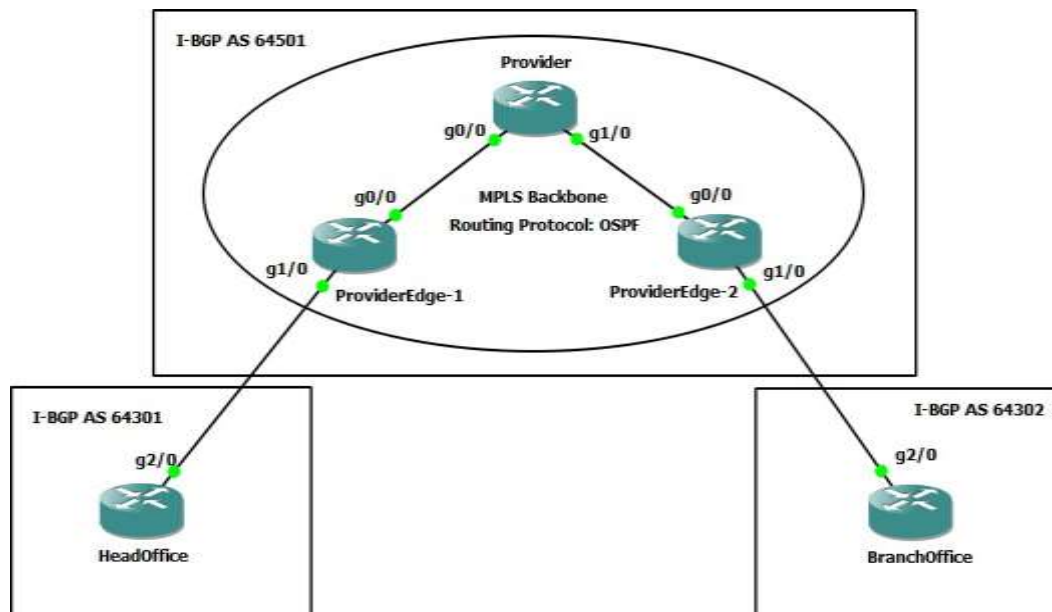
```
HeadOffice#sh ip bgp summary
BGP router identifier 192.168.43.1, local AS number 64301 BGP table version is 5, main
routing table version 5
2 network entries using 288 bytes of memory 2 path entries using
160 bytes of memory
2/2 BGP path/bestpath attribute entries using 272 bytes of memory 1 BGP 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 744 total bytes of memory
BGP activity 3/1 prefixes, 3/1 paths, scan interval 60 secs

Neighbor          V           AS MsgRcvd MsgSent   TblVer  InQ  OutQ Up/Down State/PfxRcd
10.0.0.5           4          64501      52      53       5     0   0 00:07:33      1
HeadOffice#
```

Pada hasil diatas, terlihat *peering* antara ASN 64301 dan 64501 sudah terbentuk adjacency dari *router* HeadOffice. Selanjutnya langkah-langkah implementasi rancangan sistem jaringan dan *server* yang dibangun sesuai pemodelan simulasi, dibahas ke dalam tiga point secara berurutan.

1.2.1 Jaringan MPLS-L3VPN

Model sistem pertama dibuat dengan memuat topologi sederhana *peer-to-peer*, dengan 3 *router* yang disimulasikan sebagai ISP seperti pada Gambar 1.2. Pada jaringan ini, diasumsikan sebuah jaringan backbone dengan *routing* dinamik *OSPF* untuk advertise jaringannya. Selanjutnya mengimplementasikan *cloud MPLS* pada setiap *interface* yang mengarah ke *router provider* dan memanfaatkan *routing* tabel *OSPF* sebagai informasi *routing*-nya.



Gambar 1. 2 Skema jaringan *MPLS*

Konfigurasi protokol *routing OSPF* dan *MPLS* pada *router* Provider sebagai *backbone* dari jaringan.

```

...
interface Loopback0
ip address 10.1.1.2 255.255.255.255
!
interface GigabitEthernet0/0 description Link to
Provider Edge 1 ip address 10.0.0.10 255.255.255.252
mpls ip
!
interface GigabitEthernet1/0 description Link to
Provider Edge 2 ip address 10.0.0.14 255.255.255.252
mpls ip
!
router ospf 1
network 10.0.0.8 0.0.0.3 area 0
network 10.0.0.12 0.0.0.3 area 0
network 10.1.1.2 0.0.0.0 area 0
!
...

```

Konfigurasi protokol *routing OSPF* dan *MPLS* pada *router HeadOffice* yang menghubungkan jaringan *customer* dengan jaringan *provider*.

```
...
interface Loopback0
ip address 10.1.1.1 255.255.255.255
!
interface GigabitEthernet0/0 description Link to
Provider
ip address 10.0.0.9 255.255.255.252
mpls ip
!
...
router ospf 1
network 10.0.0.8 0.0.0.3 area 0
network 10.1.1.1 0.0.0.0 area 0
!
...
```

Konfigurasi protokol *routing OSPF* dan *MPLS* pada *router BranchOffice* yang menghubungkan jaringan *customer* dengan jaringan *provider*.

```
...
interface Loopback0
ip address 10.1.1.3 255.255.255.255
!
interface GigabitEthernet0/0 description Link to
Provider
ip address 10.0.0.13 255.255.255.252
mpls ip
!
router ospf 1
network 10.0.0.12 0.0.0.3 area 0
network 10.1.1.3 0.0.0.0 area 0
!
```

Pada konfigurasi *OSPF* menunjukkan menggunakan *area 0* sebagai *single area* dan pada skema topologi sederhana tidak memerlukan *multi-area* pada *routing* yang digunakan. Setelah konfigurasi *routing* dan *MPLS*, selanjutnya pengujian rancangan simulasi yang telah dibuat untuk memberikan verifikasi konektifitas antar *router Provider* sudah terhubung. Konfigurasi *routing* eksternal *BGP* agar dapat

berkomunikasi dengan jaringan kantor cabang, pada *router* HeadOffice dilakukan konfigurasi sebagai berikut:

```
....
interface GigabitEthernet2/0 description Link to
Provider Edge 1 ip address 10.0.0.6 255.255.255.252
negotiation auto
!
router bgp 64301
bgp log-neighbor-changes network
192.168.1.0
timers bgp 10 30
neighbor 10.0.0.5 remote-as 64501
!
...
```

Kemudian menambahkan konfigurasi *BGP peer*-nya pada router ProviderEdge-1 sebagai berikut:

```
...
interface GigabitEthernet1/0 description Link to
Customer Edge 1 ip vrf forwarding Customer
ip address 10.0.0.5 255.255.255.252
negotiation auto
!
ip vrf Customer rd
64501:1
route-target export 64501:1
route-target import 64501:1
!
router bgp 64501
bgp log-neighbor-changes neighbor 10.1.1.3 remote-
as 64501
neighbor 10.1.1.3 update-source Loopback0
!
address-family ipv4 neighbor 10.1.1.3
activate
neighbor 10.1.1.3 send-community extended exit-address-family
!
address-family vpnv4 neighbor 10.1.1.3
activate
neighbor 10.1.1.3 send-community extended
```

```

exit-address-family
!
address-family ipv4 vrf Customer neighbor 10.0.0.6
remote-as 64301
neighbor 10.0.0.6 activate exit-address-
family
!
...

```

Pada jaringan kantor cabang dilakukan juga konfigurasi yang sama dengan ASN dan IP Address berbeda, pada konfigurasi *router* BranchOffice sebagai berikut:

```

...
interface GigabitEthernet2/0 description Link to
Provider Edge 2 ip address 10.0.0.22 255.255.255.252
negotiation auto
!
router bgp 64302
bgp log-neighbor-changes network
192.168.2.0
timers bgp 10 30
neighbor 10.0.0.21 remote-as 64501
!
...

```

Kemudian menambahkan konfigurasi BGP *peer*-nya pada *router* ProviderEdge-1 sebagai berikut:

```

...
interface GigabitEthernet1/0 description Link to
Customer Edge 2 ip vrf forwarding Customer
ip address 10.0.0.21 255.255.255.252
negotiation auto
!
ip vrf Customer rd
64501:1
route-target export 64501:1
route-target import 64501:1
!
router bgp 64501

```

```

bgp log-neighbor-changes neighbor 10.1.1.1 remote-
as 64501
neighbor 10.1.1.1 update-source Loopback0
!
address-family vpnv4 neighbor 10.1.1.1
activate
neighbor 10.1.1.1 send-community extended exit-address-family
!
address-family ipv4 vrf Customer neighbor 10.0.0.22
remote-as 64302
neighbor 10.0.0.22 activate exit-address-
family
!
...

```

Setelah konfigurasi *MPLS-L3VPN*, selanjutnya pengujian rancangan simulasi yang telah dibuat untuk memberikan verifikasi konektivitas antara *router customer* sudah terhubung dengan baik. Berikut hasil pengecekan komunikasi tes ping dari HeadOffice ke *Customer* dengan menggunakan Host-1 ke jaringan lokal Branchoffice yaitu Host-2.

```

LAB-20> tracert 192.168.2.10
Tracing route to PC Host-1 [192.168.2.10]
Over a maximum of 30 hops:

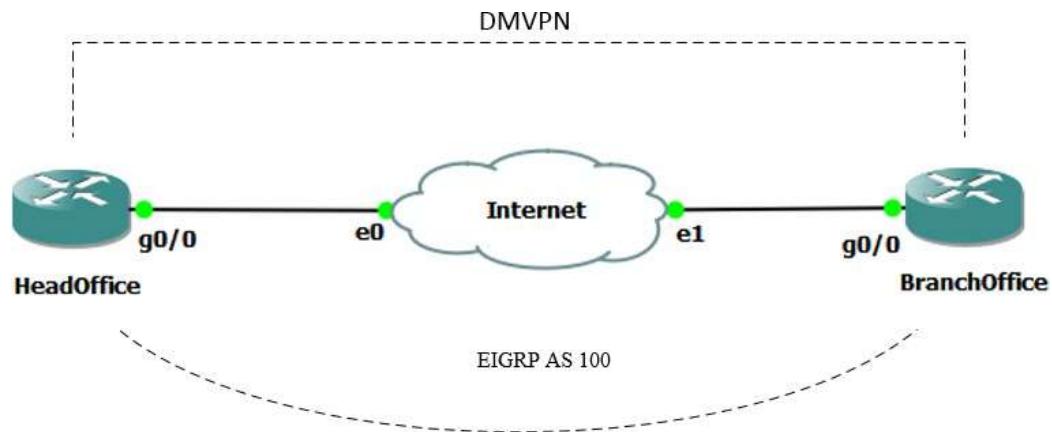
 1  50  msec  19 msec  512  msec  192.168.1.1
 2 110  msec  83 msec  30   msec  10.0.0.5
 3 153  msec 135 msec 126  msec  10.0.0.10
 4 101  msec 105 msec 115  msec  10.0.0.21
 5 150  msec 146 msec 168  msec  10.0.0.22
 6 159  msec 167 msec 114  msec  PC Host-1 [192.168.2.10]
Trace complete
C:\Users\LAB-20

```

Gambar 1. 3 Hasil uji *tracert* Host-1 ke Host-2

Dari pengujian verifikasi koneksi data yang terlihat di atas bahwa, jaringan HeadOffice dan BranchOffice sudah aktif sebagai model jaringan *MPLS-L3VPN*.

1.2.2 Jaringan DMVPN



Gambar 1. 4 Skema Jaringan DMVPN

Pada skema *DMVPN* yang merupakan solusi *VPN* dari Cisco melalui *internet* dengan menggabungkan *mGRE*, *NHRP* dan *IPSec* encryption, sehingga penerapan topologi sederhana dengan satu Branch dapat dilihat pada gambar 1.3. Pemodelan sistem ini diasumsikan dengan koneksi dengan *internet* menjadi simulasi menggunakan *switch* antara dua buah *router customer*. Protokol *routing* menggunakan *EIGRP* dalam implementasi *DMVPN* berdasarkan pada penelitian dari Ruta Jankuniene yang menyatakan bahwa *EIGRP* merupakan *routing* yang paling efektif daripada *RIP* dan *OSPF* dalam kualitas layanan *real-time* terhadap *QoS*. Berikut konfigurasi protokol *routing EIGRP* pada *router* HeadOffice sebagai *Hub* dari jaringan *DMVPN*:

```
...
router eigrp 100
network 10.10.10.1 0.0.0.0
network 172.16.90.1 0.0.0.0
network 192.168.1.1 0.0.0.0
network 192.168.43.1 0.0.0.0
!
```

Konfigurasi protokol *routing EIGRP* pada *router* BranchOffice sebagai Spoke dari jaringan *DMVPN*:

```
...
router eigrp 100
network 10.10.10.2 0.0.0.0
network 172.17.90.2 0.0.0.0
network 192.168.2.1 0.0.0.0
!
```

Berikut implementasi *IPSec* menjadi ke dalam tiga bagian, yakni implementasi *IKE Policies* di *router customer*

```
...
crypto isakmp policy 1 encr aes
hash sha512 authentication pre-share
group 2
crypto isakmp key KEY-NETWORK address 0.0.0.0
!
```

Setelah melakukan konfigurasi *IKE Policies*, selanjutnya menerapkan *IPSec* dari setiap *peer* dengan mendefinisikan sebuah transform-set dan implementasi *IPSec tunnel mode*. Berikut transform-set dengan nama *Trans-IPSec* pada kedua *router customer*:

```
crypto ipsec transform-set Trans-IPSec esp-aes esp-sha512-hmac
mode tunnel
!
```

Pada konfigurasi transform-set dan *mode tunnel* yang telah dilakukan, maka langkah terakhir adalah membuat sebuah profile dari *IPSec* yang akan digunakan ke dalam *interface router customer*. Berikut konfigurasi dari kedua *router customer*:

```
crypto ipsec profile MGRE-for-DMVPN
set security-association lifetime seconds 86400 set transform-set Trans-IPSec
!
```

Setelah diterapkan masing-masing langkah dari *IPSec* diatas, selanjutnya menerapkan *mGRE* dan *NHRP* ke dalam *interface Tunnel* yang akan diterapkan. Pertama adalah konfigurasi di *router HeadOffice*:

```

...
interface GigabitEthernet0/0 description Link to
DMVPN Public ip address 172.16.90.1 255.255.255.0
!
interface Tunnel0
description MULTI-POINT GRE TUNNEL for BRANCHE ip address 10.10.10.1
255.255.255.0
no ip redirects
ip nhrp authentication Pa$$w0rd ip nhrp map
multicast dynamic ip nhrp network-id 1
tunnel source GigabitEthernet0/0 tunnel mode gre
multipoint
tunnel protection ipsec profile MGRE-for-DMVPN
!

```

Kedua adalah konfigurasi di *router BranchOffice*:

```

interface GigabitEthernet0/0 description Link to
DMVPN Public ip address 172.16.90.2 255.255.255.0
!
interface Tunnel0
description MULTI-POINT GRE TUNNEL for HO ip address 10.10.10.2
255.255.255.0
no ip redirects
ip nhrp authentication Pa$$w0rd ip nhrp map
10.10.10.1 172.16.90.1 ip nhrp map multicast
172.16.90.1 ip nhrp network-id 1
ip nhrp nhs 10.10.10.1
tunnel source GigabitEthernet0/0 tunnel mode gre
multipoint
tunnel protection ipsec profile MGRE-for-DMVPN
!

```

Sistem *DMVPN* yang mengintegrasikan *mGRE tunnelling*, *NHRP* dan *IPSec* dapat memberikan sebuah *tunnel* atau *link VPN* yang dienkripsikan oleh *IPSec*. Pada jaringan ini dapat diimplementasikan pada sebuah jaringan *internet* yang memiliki protokol *routing*, namun pada penelitian ini menggunakan *EIGRP* sebagai protokol *routing*. Setelah konfigurasi *DMVPN*, selanjutnya pengujian rancangan simulasi yang telah dibuat untuk memberikan verifikasi konektivitas

antara *router customer* sudah terhubung dengan baik. Berikut hasil pengecekan komunikasi tes ping dari HeadOffice ke *Customer* dengan menggunakan Host-1 ke jaringan lokal Branchoffice yaitu Host-2.

```
LAB-25> tracert 192.168.2.10
Tracing route to PC Host-1 [192.168.2.10]
Over a maximum of 30 hops:

  1  15  msec 20 msec 20  msec 192.168.1.1
  2 148 msec 91 msec 85  msec 10.10.10.2
  3  64 msec 88 msec 82 msec PC Host-1 [192.168.2.10]
Trace complete
C:\Users\LAB-25>
```

Gambar 1. 5 Hasil uji *tracert* Host-1 ke Host-2

Dari pengujian verifikasi koneksi data yang terlihat di atas bahwa, jaringan HeadOffice dan BranchOffice sudah aktif sebagai model jaringan *DMVPN*.

1.3 Membangun Simulasi Jaringan *failover*

Pada pembangunan simulasi *failover* atau redundansi *VPN*, pertama dengan menggabungkan jaringan *MPLS-L3VPN* dan *DMVPN*. Kedua, menghubungkan Host-1, Host-2 dan *RTMP server* sebagai objek dari pengujian sistem. Sehingga, dapat melakukan validasi simulasi sebelum melakukan pengujian simulasi *failover*. Host-1 menggunakan komputer peneliti yang bertujuan sebagai perangkat yang dapat men-capture hasil pengujian dan pengambilan data *QoS*, serta mengupload *file streaming* ke *RTMP Server*. Host-2 menggunakan sistem operasi windows 7 yang bertujuan dapat menjalankan *VLC Player* sebagai media *streaming* dalam pengambilan data *QoS*. *RTMP Server* digunakan untuk menyediakan *server streaming* yang bertujuan dalam menjalankan video dan menyematkan *URL* sebagai sumber media.


Perancangan *RTMP Server* menggunakan *Nginx*, diasumsikan *server* sudah terinstall *Linux Ubuntu Server 16.04.6 LTS* di *Virtual box* dan sudah terhubung ke

internet untuk keperluan pembaharuan dan pemasangan *repository* dan *package* aplikasi. Langkah selanjutnya dilakukan penginstalan *Nginx* adalah sebagai berikut:

- 1) *Install package* yang diperlukan untuk *Nginx* dan fitur *RTMP*, berikut perintah dibawah ini:

```
sudo apt-get install wget unzip software-properties-common dpkg-dev git make gcc automake build-essential
zlib1g-dev libpcre3 libpcre3-dev libssl-dev libxslt1-dev libxml2-dev libgd-dev libgeoip-dev libgoogle-
perftools-dev libperl-dev pkg-config autotools-dev gpac ffmpeg mediainfo mencoder lame libvorbisenc2
libvorbisfile3 libx264-dev libvo-aacenc-dev libmp3lame-dev libopus-dev unzip
```

- 2) Tambahkan *repository Nginx* dengan perintah `sudo add-apt-repository ppa:nginx/stable`. Kemudian melakukan pembaharuan *repository* dan instalasi *Nginx* dengan perintah `apt install nginx`.



```
This PPA contains the latest Stable Release version of the nginx web server sof
tware.

**Only Non-End-of-Life Ubuntu Releases are supported in this PPA**
More info: https://launchpad.net/~nginx/+archive/ubuntu/stable
Press [ENTER] to continue or ctrl-c to cancel adding it

gpg: keyring `/tmp/tmpwlfqgcde/secring.gpg' created
gpg: keyring `/tmp/tmpwlfqgcde/pubring.gpg' created
gpg: requesting key C300EE8C from hkp server keyserver.ubuntu.com
gpg: /tmp/tmpwlfqgcde/trustdb.gpg: trustdb created
gpg: key C300EE8C: public key "Launchpad Stable" imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 1
gpg:             imported: 1 (RSA: 1)
OK
```

Gambar 1. 6 Penambahan *ppa Nginx repository*

- 3) Pindah ke direktori `/usr/src/`, intall *NGINX RTMP module* dengan *cloning* beberapa *file* yang ada dalam *repository github* dengan perintah `git clone https://github.com/arut/nginx-rtmp-module`.

```

Cloning into 'nginx-rtmp-module'...
remote: Enumerating objects: 4314, done.
^Cceiving objects: 3% (130/4314)
root@berw4ngMr5:~# cd /usr/src/
root@berw4ngMr5:/usr/src# git clone https://github.com/arut/nginx-rtmp-module
Cloning into 'nginx-rtmp-module'...
remote: Enumerating objects: 4314, done.
remote: Total 4314 (delta 0), reused 0 (delta 0), pack-reused 4314
Receiving objects: 100% (4314/4314), 3.10 MiB | 1.17 MiB/s, done.
Resolving deltas: 100% (2686/2686), done.
Checking connectivity... done.

```

Gambar 1. 7 Mengunduh NGINX RTMP module dari github

- 4) Buka *folder cloning RTMP* dari *github* yang telah diunduh dan salin *file* *stat.xml* ke *folder /var/www/*.
- 5) Kemudian buat *file crossdomain.xml* dan salin *script* dibawah ini ke dalam *file* tersebut yang bertujuan menggunakan *adobe flash* dalam melakukan *stream application*.

```

<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM "http://www.adobe.com/xml/dtds/cross-
domain-policy.dtd">
<cross-domain-policy>
<allow-access-from domain="*"></allow-access-from>
</cross-domain-policy>

```

- 6) Untuk menambahkan *script* konfigurasi *RTMP* dari *module* tersebut, buka *file* konfigurasi */etc/nginx/nginx.conf* dan tambahkan konfigurasi dibawah ini. Penelitian memberikan beberapa konfigurasi *listen port*, *policy RTMP* yaitu nama *live* pada konfigurasi *application*-nya, menambahkan *allow play all* dan *allow publish all* agar *URL stream* dapat digunakan oleh *publisher* dan *streamer*.

```

rtmp { server
{
    listen 1935;
    chunk_size 4096;
    application live {
        live on; record off;
        interleave off;
    }
}
}

```

```

        wait_key on; meta on;
        wait_video off;
        idle_stream off; sync
        300ms; session_relay on;
        allow publish all; allow
        play all;
    }
}
}

```

- 7) Kemudian edit konfigurasi `/etc/nginx/site-enabled/default` dan tambahkan *script* konfigurasi dibawah ini. *Location /stat* sebagai perintah lokasi *URL* yang disediakan untuk memverifikasikan dan memberikan tampilan kondisi *RTMP*, sedangkan `/stat.xsl` adalah sumber *script form* dari tampilan.

```

location /stat {
    rtmp_stat all; rtmp_stat_stylesheet stat.xsl;
}
location /stat.xsl {
    root /var/www/html;
}

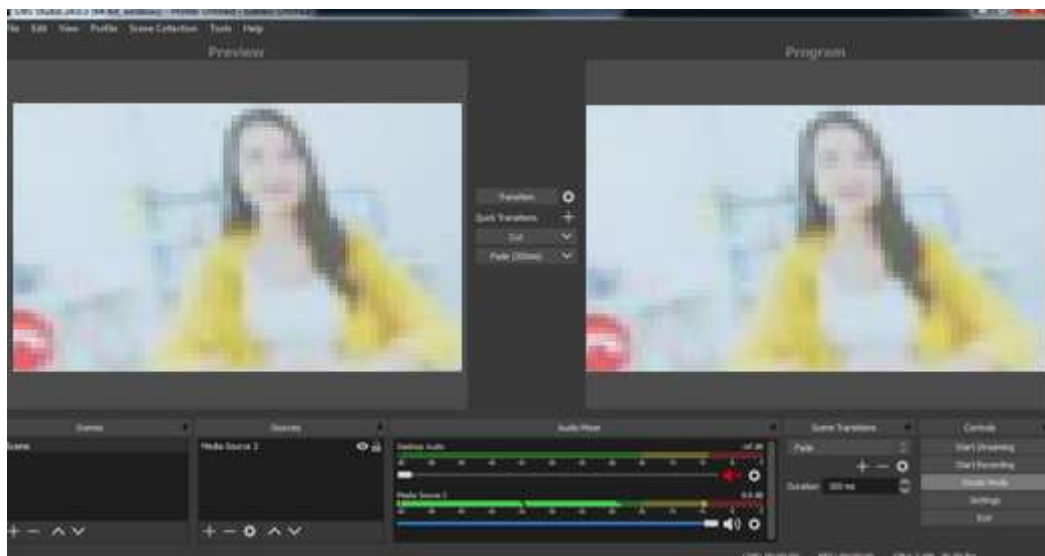
```

- 8) Simpan konfigurasi dan *restart web server*, berikut hasil *RTMP server* melalui *browser* dengan *URL* `https://192.168.43.10/stat` ditunjukkan pada Gambar 1.10 bahwa *RTMP Server* telah siap digunakan. Pada gambar tersebut, menampilkan total *file stream* yang di *publish* dalam bytes dan waktu dengan jumlah *request* (permintaan akses *RTMP* yang diterima berdasarkan *URL*) berjumlah 0 atau belum sama sekali digunakan. Pada tabel *live stream*, digunakan untuk tampilan informasi kondisi pengguna yang sedang melakukan *publish* maupun *streaming*.

RTMP	#clients	Video				Audio				In bytes	Out bytes	In bits/s	Out bits/s	State	Time
		codec	bits/s	size	fps	codec	bits/s	freq	chan						
Accepted: 0										0 KB	0 KB	0 Kb/s	0 Kb/s		5h 48m 54s
live															
live streams	0														

Gambar 1. 8 Hasil *RTMP server* melalui browser

Pada Gambar 1.10 menunjukkan *URL* yang akan digunakan untuk *streaming* video telah siap digunakan. Kemudian mengunggah *file* video ke dalam aplikasi OBS Studio dengan pengaktifan *loop* (pemutaran video berulang-ulang) dan memasukkan *URL RTMP* ke dalam aplikasi pada menu pengaturan. Pada Gambar 4.11 memperlihatkan tampilan salah satu contoh video yang akan di publish ke *RTMP server* melalui aplikasi OBS Studio. Dari tampilan OBS samping kanan adalah video asli dan akan di bagikan dengan hasil disamping kiri tersebut. Dari menu yang ada di bawah adalah pengaturan suara dan kecepatan video, serta tempat video disimpan. Untuk memulai publish, tekan start *streaming* pada menu di bawah, sehingga video akan ter-*publish* secara *real time*.



Gambar 1. 9 Tampilan *File Stream* di OBS Studio

Melakukan verifikasi *file stream* yang telah *published*, seperti yang ditunjukkan pada Gambar 1.12. dapat dilihat video telah mengunggah *file* secara *real time* ke dalam *RTMP server* seperti pada tabel *live stream* dibawah. Pengunggah didefinisikan sebagai *publishing* dengan *IP Address* 192.168.1.10 dengan *URL* *rtmp://192.168.43.10/live* dan total waktu berjalan.

RTMP statistics

192.168.43.10/stat

Accepted: 1

live

live streams: 1

stream

RTMP		Video				Audio				In bytes	Out bytes	In bits/s	Out bits/s	State	Time
	clients	codec	bits/s	size	fps	codec	bits/s	freq	chan						
		H264 Baseline 1.0	143 Kb/s	140x80	30	AAC LC	173 Kb/s	44100	2	16.54 MB	0 KB	317 Kb/s	0 Kb/s	active	8m 20s

Id	State	Address	Flash version	Page URL	SWF URL	Dropped	Timestamp	A-V	Time
1	publishing	192.168.1.10	FMLE 3.0 (compatible: FMSc 1.0)		rtmp://192.168.43.10/live	0	499066	0	8m 20s

Gambar 1. 10 Video telah ter-*publish* melalui *RTMP*

Selanjutnya Host-2 membuka aplikasi *VLC Player* dan memasukkan *URL streaming*, sehingga hasil tersebut seperti Gambar 1.13. pada gambar tersebut, Host- 2 berhasil melakukan *streaming* ke *RTMP Server* yang mana sumber *file* video berasal dari Host-1.



Gambar 1. 11 Host-2 melakukan *streaming* video melalui *VLC Player*

Pada gambar hasil akhir dari *streaming* video adalah status di browser dengan URL <http://192.168.43.10/stat> menunjukkan penambahan daftar dari live stream yaitu Host-2 yang didefinisikan sebagai playing dengan IP Address 192.168.2.10 telah menjalankan video selama 13 detik. Pada tabel *RTMP accepted* sebanyak 5 merupakan permintaan URL *RTMP* dari Host, s, selain itu jumlah bytes yang di unggah sebanyak 60 MB dan di unduh oleh Host-2 sebanyak 26 MB dengan kecepatan koneksi sebesar 789 Kb/s untuk unggahan audio dan 590 Kb/s untuk unduhan video selama total waktu *RTMP server* 6 jam 10 menit sejak dihidupkan.

RTMP statistics													
192.168.43.10/stat													
RTMP	clients	Video				Audio			In bytes	Out bytes	In bits/s	Out bits/s	State
Accepted: 5		codec	bits/s	size	fps	codec	bits/s	freq	chan				
live													
live streams:	2												
stream	2	H264 Baseline 2.1	597 Kb/s	424x240	30	AAC LC	173 Kb/s	44100	2	32.13 MB	25.73 MB	771 Kb/s	460 Kb/s
												active	6m 7s
Id	State	Address	Flash version	Page URL	SWF URL	Dropped	Timestamp	A-V	Time				
9	playing	192.168.2.10	LNX 9.0.124.2			0	366400	-13	13s				
7	publishing	192.168.1.10	FMLE/3.0 (compatible; FMSc/1.0)		rtmp://192.168.43.10/live	0	366400	-13	6m 6s				

Gambar 1. 12 Status *RTMP* saat Host-2 sedang *streaming*

1.4 Pengujian Sistem

Pengujian sistem pertama menggunakan tes ICMP tanpa menghidupkan aplikasi Video *streaming* terlebih dahulu, kemudian setelah data telah didapatkan maka ICMP perlu di matikan dan aplikasi Video *streaming* dapat mulai dihidupkan untuk dilakukan pengukuran skenario *failover VPN*. Pada Gambar 1.15 menunjukkan bahwa informasi tabel *routing* sebelum *failover* dilakukan, *router* HeadOffice masih terhubung dengan *MPLS-L3VPN* dibuktikan pada kode B atau *BGP* dengan *IP Address* 192.168.2.0/24 via 10.0.0.5.

```
10.0.0.0/8 is variably subnetted, 4 subnets, 3 masks
C    10.0.0.4/30 is directly connected, GigabitEthernet2/0
L    10.0.0.6/32 is directly connected, GigabitEthernet2/0
C    10.10.10.0/24 is directly connected, Tunnel0
L    10.10.10.1/32 is directly connected, Tunnel0
172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C    172.16.90.0/24 is directly connected, GigabitEthernet0/0
L    172.16.90.1/32 is directly connected, GigabitEthernet0/0
192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C    192.168.1.0/24 is directly connected, GigabitEthernet1/0
L    192.168.1.1/32 is directly connected, GigabitEthernet1/0
B    192.168.2.0/24 [20/0] via 10.0.0.5, 00:01:02
192.168.43.0/24 is variably subnetted, 2 subnets, 2 masks
C    192.168.43.0/24 is directly connected, GigabitEthernet3/0
L    192.168.43.1/32 is directly connected, GigabitEthernet3/0
HeadOffice#
```

Gambar 1. 13 Informasi tabel routing sebelum *failover* dilakukan

Pada pengujian dilakukan memasukkan perintah *shutdown* pada *interface* G2/0 yang menuju *MPLS-L3VPN*, sehingga *router* HeadOffice akan melakukan pembaharuan tabel *routing* yang secara otomatis akan melakukan *route distribution* ke *metric EIGRP*. Pada Gambar 1.16 informasi status jalur ke *VPN* utama telah di putuskan dan *BGP* melakukan *hello time* dan memberikan informasi bahwa routing *BGP* akan di *reset* dengan status *down* pada *neighbor* 10.0.0.5.


```

HeadOffice#
HeadOffice#
HeadOffice#conf t
Enter configuration commands, one per line. End with CNTL/Z.
HeadOffice(config)#int g2/0
HeadOffice(config-if)#shutdown
HeadOffice(config-if)#
*Oct 23 19:46:10.355: %BGP-5-NBR_RESET: Neighbor 10.0.0.5 reset (Interface flap)
*Oct 23 19:46:10.379: %BGP-5-ADJCHANGE: neighbor 10.0.0.5 Down Interface flap
*Oct 23 19:46:10.379: %BGP_SESSION-5-ADJCHANGE: neighbor 10.0.0.5 IPv4 Unicast topology base removed from session Interface flap
HeadOffice(config-if)#
*Oct 23 19:46:12.323: %LINK-3-CHANGED: Interface GigabitEthernet2/0, changed state to administratively down
*Oct 23 19:46:13.323: %LINKPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet2/0, changed state to down
HeadOffice(config-if)#

```

Gambar 1. 14 Router HeadOffice melakukan route ke DMVPN

Terlihat pada informasi tabel *routing*, router HeadOffice telah memperbaharui tabelnya dengan routing *EIGRP* seperti ditunjukkan pada gambar

4.17 yaitu kode D atau *EIGRP* terhubung dengan IP Address 192.168.2.10/24 via 10.10.10.2.

```

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.10.0/24 is directly connected, Tunnel0
L       10.10.10.1/32 is directly connected, Tunnel0
172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C       172.16.90.0/24 is directly connected, GigabitEthernet0/0
L       172.16.90.1/32 is directly connected, GigabitEthernet0/0
192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.1.0/24 is directly connected, GigabitEthernet1/0
L       192.168.1.1/32 is directly connected, GigabitEthernet1/0
D       192.168.2.0/24 [90/26880256] via 10.10.10.2, 00:02:18, Tunnel0
192.168.43.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.43.0/24 is directly connected, GigabitEthernet3/0
L       192.168.43.1/32 is directly connected, GigabitEthernet3/0
HeadOffice#

```

Gambar 1. 15 Informasi tabel *routing* setelah *failover* berhasil

Gambar 1.15 menunjukkan hasil validasi *Failover* melalui pengiriman ICMP dari saat proses pergantian rute jaringan VPN dengan pemutusan rute *cloud MPLS*. Pada pengiriman yang berhasil pertama, menunjukkan koneksi sedang melalui *MPLS-L3VPN* kemudian terputus dengan pesan Request Time out pada saat pengirimannya dan pengiriman yang berhasil kedua menunjukkan koneksi telah kembali terhubung melalui *DMVPN*.

```
LAB-25> ping 192.168.2.10 -t

Pinging 192.168.2.10 with 32 bytes of data:
Reply from 192.168.2.10: bytes=32 time=270ms TTL=123
Reply from 192.168.2.10: bytes=32 time=176ms TTL=123
Reply from 192.168.2.10: bytes=32 time=142ms TTL=123
Reply from 192.168.2.10: bytes=32 time=156ms TTL=123
Reply from 192.168.2.10: bytes=32 time=128ms TTL=123
Reply from 192.168.2.10: bytes=32 time=140ms TTL=123
Reply from 192.168.2.10: bytes=32 time=122ms TTL=123
Reply from 192.168.2.10: bytes=32 time=154ms TTL=123
Reply from 192.168.2.10: bytes=32 time=152ms TTL=123
Reply from 192.168.2.10: bytes=32 time=166ms TTL=123
Reply from 192.168.2.10: bytes=32 time=178ms TTL=123
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.2.10: bytes=32 time=614ms TTL=126
Reply from 192.168.2.10: bytes=32 time=104ms TTL=126
Reply from 192.168.2.10: bytes=32 time=73ms TTL=126
Reply from 192.168.2.10: bytes=32 time=36ms TTL=126
Reply from 192.168.2.10: bytes=32 time=63ms TTL=126
Reply from 192.168.2.10: bytes=32 time=144ms TTL=126
Reply from 192.168.2.10: bytes=32 time=103ms TTL=126
Reply from 192.168.2.10: bytes=32 time=62ms TTL=126
Reply from 192.168.2.10: bytes=32 time=73ms TTL=126
Reply from 192.168.2.10: bytes=32 time=60ms TTL=126
Reply from 192.168.2.10: bytes=32 time=51ms TTL=126
Reply from 192.168.2.10: bytes=32 time=54ms TTL=126
Reply from 192.168.2.10: bytes=32 time=71ms TTL=126
Reply from 192.168.2.10: bytes=32 time=60ms TTL=126

Ping statistics for 192.168.2.10:
    Packets: Sent = 30, Received = 25, Lost = 5 (16% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 36ms, Maximum = 614ms, Average = 134ms
Control-C

Ping [192.168.2.10]
C:\Users\LAB-25
```

Gambar 1. 16 Hasil uji ICMP pada saat proses pemutusan

1.5 Data Hasil Pengukuran

1.5.1 Parameter Pengukuran *Quality of Service*

Pengambilan data menggunakan wireshark berdasarkan pengukuran parameter *Quality of Service (QoS)* dengan mereferensikan indeks kualitasnya dari *ITU-T G.411* diantaranya sebagai berikut:

1.5.1.1 Delay

Sejumlah paket yang dikirim selama komunikasi data di transmisikan dari sumber ke tujuan jaringan. Cara menghitung *delay* dengan persamaan dari setiap paket-paket yang ter-*capture* selama 60 detik dan kemudian mengambil rata-rata dari seluruh *delay* tersebut. Persamaan yang digunakan adalah selisih dari waktu penerimaan paket (T_r) dengan waktu pengiriman paket (T_s).

$$Delay(s) = T_r - T_s \quad (1) \text{ Tabel 4. 4 Hasil}$$

pengukuran *Delay*

Percobaan	MPLS-L3VPN	Failover Process	DMVPN
1	0,0194	0,0312	0,0568
2	0,0055	0,0326	0,0581
3	0,0061	0,0441	0,0198
4	0,0067	0,0361	0,0283
5	0,0060	0,0601	0,0214
Rata-rata	0,0087	0,0408	0,0369

*) dalam satuan detik

1.5.1.2 Jitter

Sebuah variasi dari seluruh transmisi *delay* yang terjadi ketika terdapat peningkatan trafik yang menimbulkan antrian paket. Perhitungan menggunakan hasil pengukuran *delay* sebelumnya, kalkulasi total variasi *delay* dengan melakukan *delay* kedua (T_{i+1}) dikurangi *delay* pertama (T_i) dan kemudian dibagi oleh total paket yang diterima (N) dalam satu transmisi selama 60 detik tersebut.

$$Jitter(s) = \frac{\sum(T_{i+1}-T_i)}{N} \quad (2)$$

Tabel 4. 5 Hasil pengukuran *Jitter*

Percobaan	MPLS-L3VPN			Failover Process			DMVPN		
	N	$\sum(T)$	<i>Jitter</i> (s)	N	$\sum(T)$	<i>Jitter</i> (s)	N	$\sum(T)$	<i>Jitter</i> (s)
1	3167	0,1062	0,0000335	1990	0,0059	0,0000030	1084	0,0395	0,0000364
2	2753	0,0092	0,0000033	1898	0,0163	0,0000086	1057	0,0015	0,0000014
3	1781	0,0333	0,0000187	1389	0,0006	0,0000004	3076	0,0005	0,0000002
4	2003	0,0964	0,0000481	1681	0,0006	0,0000004	2868	0,0765	0,0000267
5	3019	0,0009	0,0000003	1024	0,1446	0,0001412	2949	0,1974	0,0000669
Rata-rata			0,0000208			0,0000307			0,0000263

1.5.1.3 Packet Loss

Sejumlah paket yang ter-*drop* selama transmisi dari satu sumber ke tujuan yang disebabkan oleh congestion ataupun collision. Perhitungan parameter ini menggunakan persamaan dari selisih total paket yang dikirim (P_s) dengan total

paket yang diterima (P_r) dan kemudian dibagi oleh total paket yang dikirim (P_s) ke dalam satuan persentase pada masing-masing percobaan yang dilakukan.

$$\text{Packet Loss (\%)} = \left(\frac{P_s - P_r}{P_s} \right) \times 100\% \quad (3)$$

Tabel 4. 6 Hasil pengukuran *Packet Loss*

Percobaan	MPLS-L3VPN			Failover Proccess			DMVPN		
	P_s	P_r	Drop	P_s	P_r	Drop	P_s	P_r	Drop
1	3167	3167	0,0%	1990	1990	0,0%	1084	1079	0,2%
2	2753	2749	0,1%	1898	1893	0,3%	1057	1052	0,1%
3	1781	1781	0,0%	1389	1384	0,5%	3076	3072	0,2%
4	2003	1997	0,3%	1681	1676	0,5%	2868	2868	0,0%
5	3019	3019	0,0%	1024	1024	0,0%	2949	2943	0,3%
Rata-rata			0,1%			0,3%			0,2%

1.5.1.4 Throughput

Sebuah kecepatan transfer data yang efektif dalam satuan *bps* yang mana pengukuran pada jumlah total paket yang berhasil diterima oleh reciever selama waktu tertentu dan dibagi durasi waktu interval tersebut, sehingga persamaanya sebagai berikut.

$$\text{Throughput (bps)} = \frac{\text{Jumlah paket yang diterima (bit)}}{\text{Lama pengamatan (s)}} \quad (4)$$

Tabel 4. 7 Hasil pengukuran *Throughput*

Percobaan	MPLS-L3VPN	Failover Proccess	DMVPN
1	422,3	265,3	143,9
2	366,5	252,4	140,9
3	237,5	184,5	409,6
4	266,3	223,5	382,4
5	402,5	136,5	392,4
Rata-rata	339,0	212,5	293,8

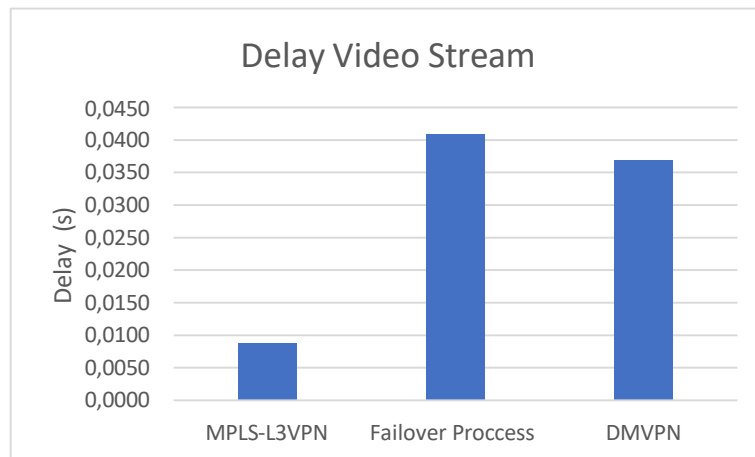
*) dalam satuan bps

1.5.2 Analisis Data Hasil Pengukuran

1.5.2.1 Analisis Hasil Pengukuran Delay

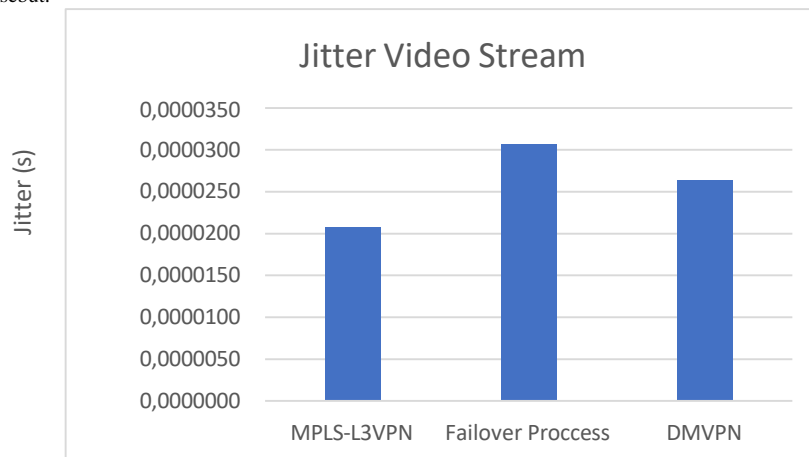
Delay adalah sejumlah waktu yang diambil dalam mentransmisikan paket melalui jaringan dari sumber jaringan ke tujuan jaringan. Statistik parameter *delay* menjadi salah satu yang perlu diperhitungkan terkait kualitas suara dan video, sehingga besar maksimum yang direkomendasikan ITU-T G.114 yaitu 150 ms. Pada garis grafik dari Gambar 1.16 memperlihatkan *packet delay* dari 5 kali pengiriman TCP per/60 detik menggunakan kedua *MPLS-L3VPN*, *DMVPN* dan proses *failover*-nya. Pada awalnya, *delay* pada *cloud MPLS* yang nilainya rendah namun mulai terjadi peningkatan ketika *failover* sedang berjalan menuju *DMVPN*. Dari hasil pengukuran *delay* didapatkan hasil skenario *Failover Process* memiliki rata-rata waktu tertinggi dari skenario lain, sedangkan perbedaan antara skenario *MPLS-L3VPN* dan *DMVPN* sangat berbeda signifikan dikarenakan karakteristik pengiriman UDP yang merupakan *connectionless* dan *unreliable* kemudian melalui jaringan *DMVPN* yang mengimplementasikan algoritme kriptografi *IKE*. Proses enkripsi ini membutuhkan sedikit waktu sebelum data ditransmisikan, sehingga perbedaan antara skenario *Failover Process* dan *DMVPN* terlihat signifikan.

Gambar 1. 17 Delay rata-rata pada kondisi video streaming



1.5.2.2 Analisis Hasil Pengukuran Jitter

Pada Gambar 1.17 menunjukkan perbedaan cukup signifikan dari variasi *packet delay* yang diukur selama simulasi dari ketiga skenario. Variasi end-to-end *delay* untuk paket video *stream* dan diukur dari saat paket dibuat hingga saat diterima. Pada aturan yang diterbitkan oleh ITU-T G.114 terhadap kualitas *jitter* yang direkomendasikan, terindeks sangat baik adalah kurang dari 30 ms untuk semua skenario. Dalam simulasi yang dibuat terhadap Video *streaming*, paket *delay* sering terlihat karena paket *delay* disebabkan oleh *multiple hops* yang terjadi selama transmisi. Secara eksperimen topologi jaringan yang memiliki beberapa *hop*, bisa menyebabkan peningkatan terhadap ketiga skenario yang diuji. Maka, skenario *Failover Process* yang terjadi perpindahan *route* dari *cloud MPLS* ke *DMVPN* menyebabkan peningkatan nilai *jitter* pada video *stream* tersebut.

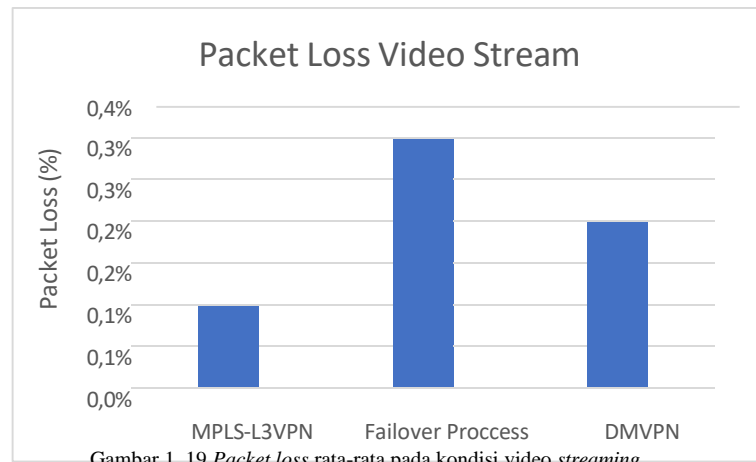


Gambar 1. 18 Jitter rata-rata pada kondisi video *streaming*

1.5.2.3 Analisis Hasil Pengukuran Packet loss

Statistik *Packet loss* digunakan dalam komparasi jumlah trafik yang di *drop* selama pengiriman data dari ketiga skenario yang dibuat. Gambar 1.18 memperlihatkan hasil pengukuran dalam satuan persen. *Packet loss* atau trafik IP yang ter *drop* adalah sejumlah datagram IP yang di *drop* oleh *node* atau IP *interface* dalam jaringan yang ada. Menangani trafik data yang memasuki jaringan dapat dilakukan dengan mengendalikan trafik yang sibuk dan memastikan bahwa alur

desain trafik mendapatkan bandwidth yang tepat. Paket yang ter *drop* terjadi ketika flow traffic mencapai maksimum atau prioritas paket yang melebihi bandwidth yang ada. *Packet loss* disebabkan TCP melakukan pengiriman ulang paket-paket tersebut dan Gambar 1.18 menunjukkan bahwa dimulai skenario *Failover Process*, *packet loss* terjadi sebanyak 0,3% dari sebelumnya $\pm 0,1\%$ disebabkan terjadi perpindahan *route* atau *failover* tersebut. Sedangkan, skenario *DMVPN* terdapat *packet loss* disebabkan pengaruh besarnya *delay*, yang mana semakin besar nilai *delay* maka *packet loss* dapat semakin besar. Mereferensikan pada rekomendasi ITU-T G.411 bahwa *packet loss* dari ketiga skenario masih dibawah maksimal adalah sebesar 10%.

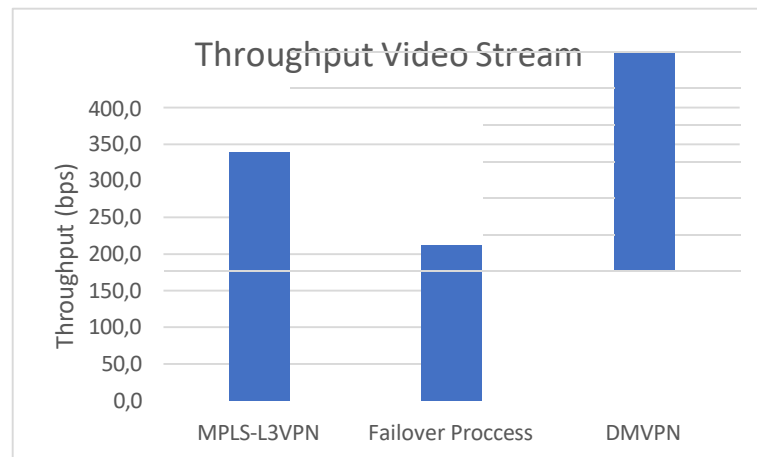


Gambar 1. 19 *Packet loss* rata-rata pada kondisi video streaming

1.5.2.4 Analisis Hasil Pengukuran Throughput

Throughput menunjukkan suatu waktu dalam relasi terhadap rata-rata jumlah paket yang berhasil diterima dalam satuan paket per detik. Seperti pada Gambar 1.19, hasil yang didapatkan dari ketiga skenario didasarkan pada simulasi pengiriman selama 60 detik. *Throughput* merupakan rata-rata jumlah *bit* yang berhasil diterima atau ditransmisikan oleh penerima atau pengirim dalam satuan waktu (*bit* per detik), yang mana dimaksud sebagai rata-rata nilai *Video streaming* yang berhasil dari publisher ke *server RTMP* kemudian pengguna dalam simulasi. Throughput yang terbaik adalah skenario *MPLS-L3VPN*, diikuti oleh skenario

DMVPN, dan relatif dekat diikuti oleh skenario *Failover Process*. Pengaruh perbedaan ini disebabkan oleh nilai *delay*, *jitter* dan paket loss yang ada pada pengukuran sebelumnya.



Gambar 1. 20 *Throughput* rata-rata pada kondisi *video streaming*

Referensi

<https://github.com/channith/Lab-Cisco/blob/main/mpls-vpn-rd-vs-rt.md>

<https://github.com/manolodd/opensimimpls>

<https://blog.thomarite.uk/index.php/2020/04/19/gns3-basic-mpls-l3vpn-and-ospf-in-pe-ce-routing-down-bit/>

https://github.com/martimy/clab_mpls_frr