# Implementasi GRE *Tunneling* Menggunakan Open vSwitch Pada Jaringan Kampus

**Mufid Ridlo Effendi[1], Eki Ahmad Zaki Hamidi[2], Andriansyah Saepulloh[3]**
[1,2,3]Jurusan Teknik Elektro UIN Sunan Gunung Djati Bandung
[1,2,3]Jl. A.H. Nasution 105 Bandung
e-mail: [1]mufid.ridlo@uinsgd.ac.id, [2]ekiahmadzaki@uinsgd.ac.id, [3]andriansyahsaepulloh17@gmail.com

***Abstrak –*** *Perkembangan informasi yang pesat membuat penambahan jaringan baru harus dilakukan dengan cepat, efektif, dan efisien. Menghubungkan dua jaringan lokal yang berbeda wilayah dibutuhkan metoda yang tepat tanpa biaya yang besar. Salah satu metode jaringan untuk menghubungkan dua jaringan lokal yang berbeda wilayah tanpa terkoneksi langsung secara fisik yaitu menggunakan metode tunneling. Salah satu metode tunneling adalah GRE. Metode GRE menggunakan Open vSwitch dengan menggunakan simulator mininet dengan topologi jaringan kampus sederhana telah dilakukan pada penelitian ini. Simulasi ini menghasilkan hasil yang diinginkan dengan terkoneksinya tiap host pada jaringan kampus yang berbeda wilayah.*

***Kata kunci****: Tunneling, GRE, Open vSwitch, Mininet*

## 1. Pendahuluan

LAN (*Local Area Network*) merupakan jaringan paling kecil. Terdiri dari beberapa *host* yang dihubungkan menggunakan menggunakan switch *layer* 2 membentuk topologi *star*. Pada sebuah gedung, LAN dapat dibagi-bagi menjadi beberapa segmen untuk mengurangi *collision domain*. Koneksi antar gedung menjadikan interkoneksi antar LAN dalam satu administrasi. Salah satu interkoneksi antar LAN beberapa gedung yaitu pada kampus.

Interkoneksi pada suatu kampus dalam satu area masih memungkinkan menggunakan melalui *physical layer* secara langsung yaitu melakukan koneksi langsung menggunakan kabel. Suatu permasalah akan muncul jika terdapat kampus yang berbeda area atau beberapa gedung kampus di berbeda tempat, tetapi diinginkan masih jaringan lokal. Melakukan koneksi jaringan dengan koneksi langsung menggunakan kabel untuk area yang sangat jauh menjadi solusi yang tidak efektif dan efisien.

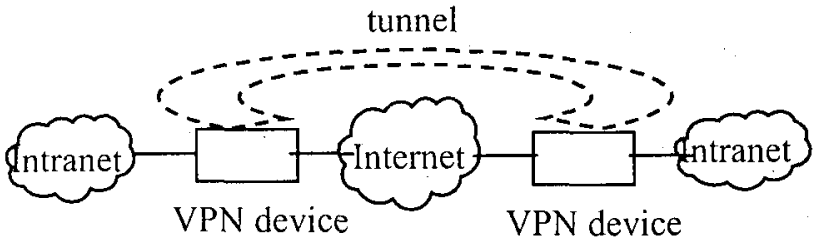
Terdapat teknologi jaringan yang dapat mengoneksikan jaringan lokal (LAN) yang berbeda area melalui jaringan publik (internet). Yaitu dengan cara *tunneling*, mengencapsulasi protokol IP dengan protokol tertentu dan diencapsulasi kembali oleh protokol IP kemudian dikirimkan melalui jaringan publik. Salah satu teknik tunneling yaitu GRE (*Generic Routing Encapsulation*) yang merupakan *network layer over network layer*. Tunneling merupakan solusi interkoneksi antar jaringan lokal yang dipisahkan oleh jarak jauh melalui jaringan publik [1].

Pada penelitian ini membahas tentang intrakoneksi antar kampus yang berbeda area yeng masih dalam satu jaringan lokal menggunakan protokol GRE. Perangkat yang digunakan adalah open vswitch yang merupakan switch virtual yang dapat dijalankan pada perangkat komputer dengan sistem operasi Linux. Untuk mengefisiensikan penggunaaan perangkat keras, dilakukan simulasi pada perangkat lunak mininet.

## 2. Dasar Teori

### 2.1. GRE Tunnel

*Tunneling* merupakan metode encapsulasi, misalnya, suatu protokol (protokol X) diencapsulasi oleh protokol lain (protokol Y) saat dikirimkan sehingga protokol X transparent terhadap jaringan publik [2]. Terdapat beberapa protokol tunnel; GRE (*Generic Routing Encapsulation*) [3], L2TP (*Layer 2 Tunneling Protocol*) [4], PPTP (*Point-to-Point Tunneling Protocol*) [5], DVMRP (*Distance Vector Multicast Routing Protocol*) [6]. Gambar 1 memperlihatkan topologi sederhana tunneling.




Gambar 1. Topologi sederhana tunneling [2]

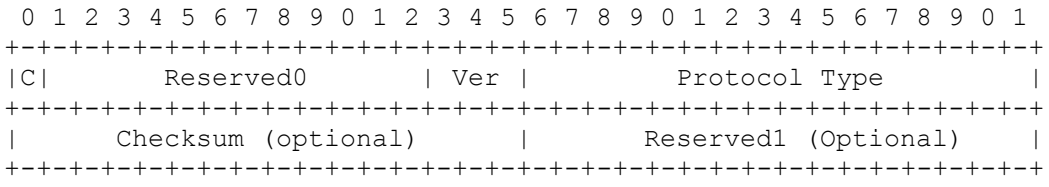
GRE *tunnel* merupakan protokol *tunneling* yang dikembangkan oleh Cisco dan menyediakan encapsulasi untuk berbagai layer protokol jaringan pada jaringan *point to point* [7]. GRE *tunnel* dibangun antara router asal dan router tujuan sehingga paket yang di-*forward* melalui *tunnel* sebelumnya telah diencapsulasi oleh *header* yang baru (GRE *header*) [8]. Kelebihan GRE *tunnel* :

1. Menghubungkan subnet yang tidak kontinyu
2. Penggunaan sumber daya yang rendah
3. Mendukung pesan *unicast, multicast,* dan *broadcast*
4. Dapat mengencapsulasi semua jenis protokol layer 3

GRE *tunnel* dapat digunakan pada kondisi :

- Menghubungkan jaringan yang tidak menggunakan protokol IP atau menghubungkan jaringan IP lokal antar area
- Merutekan paket IPv6 melalui jaringan IPv4
- Encrypt tarfik *multicast* [1].

Gambar 2 memperlihatkan *header* GRE *tunnel*.



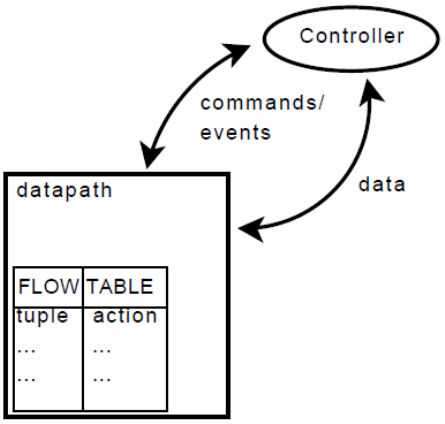

Gambar 2. GRE *header* [3]

ISSN (e): 2540-9123
ISSN (p): 2502-1982

### 2.2. Open vSwitch

Open vSwitch merupakan perangkat lunak kode terbuka yang dirancang untuk digunakan sebagai vswitch (virtual switch) dalam lingkungan server virtual [9]. Open vSwitch dapat diinstall pada perangkat komputer atau perangkat keras khusus [10]. Open vSwitch dikembangkan oleh Nicira Network dan telah banyak digunakan karena kestabilannya dan dapat diprogram. Open vSwitch dijalankan sebagai modul pada kernel Linux menggantikan modul bridge [11]. Open vSwitch mendukung banyak managemen jaringan seperti NetFlow, sFLow, CLI dan lain sebagainya. Optimasi pada komputer (PC) telah dilakukan oleh [13], namun kinerjanya relatif lambat karena beban CPU yang berlebih. Pengujian kinerja pada PC telah dilakukan oleh [14]. Implementasi dan pengujian performa menggunakan NetFGA telah dilakukan [15].

Open vSwitch mendukung protokol OpenFlow [16], dimana Open vSwitch sebagai *data plane* dikontrol oleh *controller* sebagai *control plane*. Model komunikasi *data plane* dengan *control plane* diperlihatkan pada gambar 3 [6]. *Data plane* dapat dikontrol untuk melakukan QoS, *tunneling*, dan *filtering rules*. Dukungan pengontrolan jarak jauh membuat Open vSwitch dapat digunakan untuk melakukan proses migrasi kebijakan jaringan. Karena fleksibilitasnya, *data plane* pada Open vSwitch dapat dipartisi secara logik. Open vSwitch kombatibel dengan lingkungan virtual pada Linux seperti Xen, XenServer, KVM, dan QEMU [12].



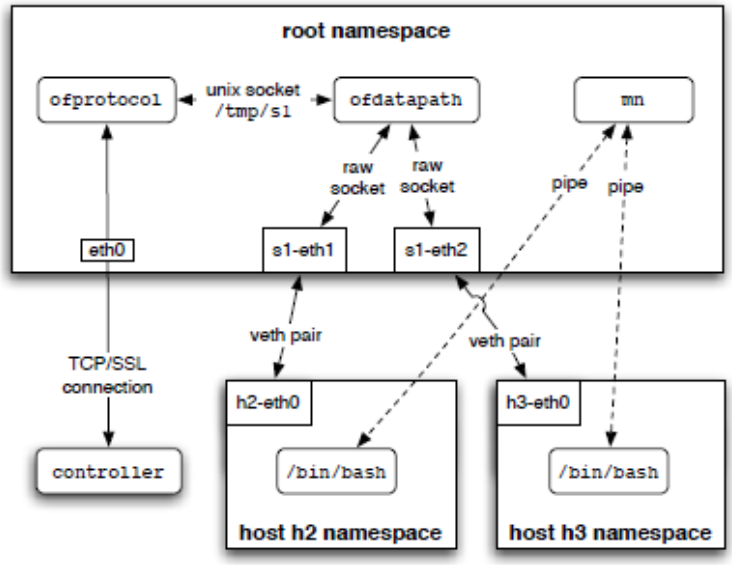

Gambar 3. Arsitektur Virtual Switch [10]

### 2.3. Mininet

Mininet merupakan sebuah system sebagai *prototype* untuk mengembangkan jaringan yang dijalankan pada sebuah perangkat komputer (laptop atau PC). Mininet dapat menampung ratusan *node* (switch, host, dan controller) dalam satu perangkat komputer. Mininet banyak digunakan untuk mengembangkan jaringan berbasis SDN. Pengguna dapat mengimplementasikan fitur baru pada jaringan yang dirancangnya, mengembangkanya, dan menjalankannya pada jaringan nyata. Mininet dijalankan pada Sistem Operasi Linux dengan menggunakan pemrograman python. Gambar 4 memperlihatkan contoh topologi sederhana yang dibangun menggunakan mininet [17].

Interaksi antara pengguna dengan mininet pada perangkat komputer melalui *Command Line Interface* (CLI) dapat melakukan konfigurasi pada seluruh jaringan yang dirancang dalam satu *console*. Pengembangan jaringan pada mininet secara manual melalui satu *console* akan memakan waktu dan ketelitian yang tinggi, untuk mempermudah pembuatan jaringan dapat digunakan *script* python pada sebuah *file* sehingga pembuatan jaringan akan lebih mudah dan jika kegagalan, penambahan node, penambahan *fiture*, dan lain sebagainya dapat langsung mengedit *file script* tersebut.

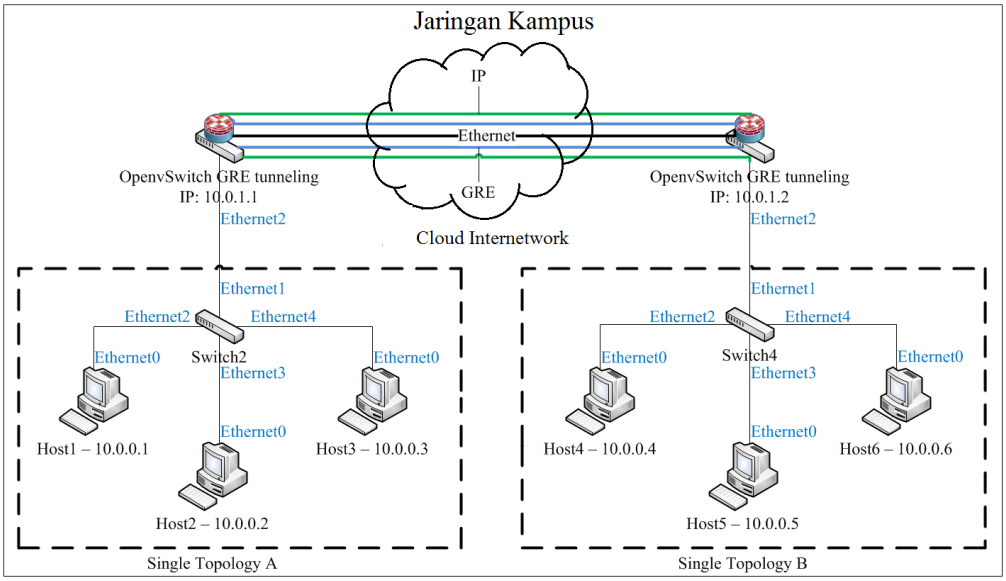
ISSN (e): 2540-9123
ISSN (p): 2502-1982




Gambar 4. Topologi sederhana yang dibangun menggunakan mininet [17]

## 3. Perancangan Topologi Jaringan

Topologi jaringan yang dirancang pada penelitian ini adalah topologi sederhana yang menghubungkan jaringan lokal (intranet) dengan jaringan lokal lainnya masih dalam satu *subnet* yang dipisahkan oleh jarak karena berbeda wilayah/area. Sebagai contoh adalah jaringan yang menghubungkan antara dua kampus yang berbeda wilayah. Tidak dibahas mengenai jaringan dalam kampus, tetapi pembahasan hanya pada cara menghubungkan satu kampus dengan kampus lainnya. Topologi tersebut digambarkan pada gambar 5.




Gambar 5. Topologi jaringan antar kampus sederhana

Pada gambar 5, misalkan kampus A dan kampus B berada pada area yang berbeda. Untuk menghubungkan kedua kampus tersebut supaya jaringannya masih satu subnet digunakan metode *tunneling* menggunakan protokol GRE yang dijalankan pada Open vSwicth. Open vSwitch berdiri

ISSN (e): 2540-9123
ISSN (p): 2502-1982

sendiri tidak terhubung dengan *controller* sehingga tidak digunakan protokol openflow untuk mengkonfigurasi Open vSwitch. Konfigurasi Open vSwitch dilakukan secara langsung dengan perintah manual yang ditulis dalam sebuah *file* berextensi python dan kemudian dijalankan pada *console*.

Sebagai simulasi internet, IP antar Open vSwitch dengan jaringan lokal dibedakan sehingga terjadi proses *routing*. Alamat IP Open vSwitch pada kampus A adalah 10.0.1.1 dan Open vSwitch pada kampus B adalah 10.0.1.2. Sementara alamat IP pada jaringan lokal menggunakan 10.0.0.0/24. Digunakan juga Open vSwitch pada tiap jaringan lokal yang berfungsi hanya sebagai switch layer 2.

## 4. Perangkat Keras Dan Perangkat Lunak

Implementasi topologi jaringan pada gambar 5 dilakukan pada perangkat keras *laptop* yang menjalankan sistem operasi Windows 7 64-Bit versi *trial*. Topologi dijalankan pada *Mininet-2.2.1-150420* yang berjalan pada sistem operasi *Ubuntu-14.04-server-i386*. Ubuntu server dijalankan pada Virtual Machine *VirtualBox-5.0.16-105871-Win*.

## 5. Pengujian

Pengujian awal yaitu melihat apakah konfigurasi yang sudah direncanakan sudah sesuai atau belum. Pada gambar 6 output konfigurasi pada Open vSwitch 1 yang menjalankan GRE *tunnel*.

```
Bridge "s1"
    Port "s1-gre"
        Interface "s1-gre"
    Port "gre1"
        Interface "gre1"
            type: gre
            options: {remote_ip="10.0.1.1"}
    Port "s1-eth1"
        Interface "s1-eth1"
    Port "s1"
        Interface "s1"
            type: internal
```

Gambar 6. Output konfigurasi Open vSwitch menjalankan GRE tunnel pada Open vSwitch 1

Gambar 7 merupakan output konfigurasi Open vSwitch 3 yang menjalankan GRE *tunnel.*

```
Bridge "s3"
    Port "s3-gre"
        Interface "s3-gre"
    Port "s3-eth1"
        Interface "s3-eth1"
    Port "s3"
        Interface "s3"
            type: internal
    Port "gre2"
        Interface "gre2"
            type: gre
            options: {remote_ip="10.0.1.2"}
```

Gambar 7. Output konfigurasi Open vSwitch menjalankan GRE tunnel pada Open vSwitch 3

Gambar 8 dan 9 merupakan output konfigurasi pada Open vSwitch 2 dan Open vSwitch 4 yang berfungsi sebagai switch layer 2.

ISSN (e): 2540-9123
ISSN (p): 2502-1982

```
Bridge "s2"
    Port "s2-eth3"
        Interface "s2-eth3"
    Port "s2-eth4"
        Interface "s2-eth4"
    Port "s2"
        Interface "s2"
            type: internal
    Port "s2-eth1"
        Interface "s2-eth1"
    Port "s2-eth2"
        Interface "s2-eth2"
```

Gambar 8. Output konfigurasi Open vSwitch sebagai switch layer 2 pada Open vSwitch 2

```
Bridge "s4"
    Port "s4-eth2"
        Interface "s4-eth2"
    Port "s4-eth1"
        Interface "s4-eth1"
    Port "s4-eth3"
        Interface "s4-eth3"
    Port "s4-eth4"
        Interface "s4-eth4"
    Port "s4"
        Interface "s4"
            type: internal
```

Gambar 9. Output konfigurasi Open vSwitch sebagai switch layer 2 pada Open vSwitch 4

Berikutnya adalah pengecekan konfigurasi jaringan keseluruhan, dapat dilihat pada gambar 10.

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=1242>
<Host h2: h2-eth0:10.0.0.2 pid=1245>
<Host h3: h3-eth0:10.0.0.3 pid=1247>
<Host h4: h4-eth0:10.0.0.4 pid=1249>
<Host h5: h5-eth0:10.0.0.5 pid=1251>
<Host h6: h6-eth0:10.0.0.6 pid=1253>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-gre:None pid=1258>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None,s2-eth4:None
pid=1261>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-gre:None pid=1264>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None,s4-eth3:None,s4-eth4:None
pid=1267>
mininet>
```

Gambar 10. Output konfigurasi jaringan

Pada gambar 10 terlihat semua node sudah terkonfigurasi sesuai dengan yang diharapkan.

Pengujian terakhir adalah menguji semua konektifitas antar *host*. Mininet memiliki *tools* khusus untuk menguji konektifitas antar semua host yaitu menggunakan perintah *pingall*. Output dari perintah *pingall* dapat dilihat pada gambar 11.

ISSN (e): 2540-9123
ISSN (p): 2502-1982

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6
h2 -> h1 h3 h4 h5 h6
h3 -> h1 h2 h4 h5 h6
h4 -> h1 h2 h3 h5 h6
h5 -> h1 h2 h3 h4 h6
h6 -> h1 h2 h3 h4 h5
*** Results: 0% dropped (30/30 received)
mininet>
```

Gambar 11. Hasil *pingall*

Dengan perintah *pingall* pada mininet, semua *host* diuji koneksinya menggunakan protokol ICMP. Dari gambar 11 terlihat bahwa semua *host* terkoneksi satu dengan yang lainnya dilihat dari packet yang hilang 0%. Hal tersebut menunjukan juga bahwa koneksi antara area yang berbeda terhubung dengan baik, terlihat h1 pada kampus A dapat terhubung dengan h4 pada kampus B. Artinya GRE *tunnel* menggunakan Open vSwitch dapat dilakukan dengan baik.

Pengujian lain yaitu dengan melakukan *ping* dari h1 ke h6 seperti pada gambar 12 dan koneksi *client server* antara h1 dan h6 seperti pada gambar 13.

```
mininet> h1 ping h6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=3.82 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.054 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.052 ms
^C
--- 10.0.0.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.052/1.309/3.822/1.776 ms
mininet>
```

Gambar 12. Perintah *ping* antara h1 dan h6

Dari gambar 12, proses *ping* berjalan dengan baik terlihat semua paket yang dikirim oleh h1 diterima oleh h6. Dari gambar 13, h1 menjalankan *web server* sementara h6 sebagai *client* yang melakukan permintaan informasi. Proses koneksi *client server* antara h6 dan h1 berjalan baik dengan diterimanya kode http 200.

ISSN (e): 2540-9123
ISSN (p): 2502-1982

```
mininet> h1 python -m SimpleHTTPServer 80 &
mininet> h6 wget -O - h1
--2017-08-03 09:30:52--  http://10.0.0.1/
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 372 [text/html]
Saving to: 'STDOUT'

 0% [                                       ] 0           --.-K/s      <
!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href="OVS-GRE-tunnel.py">OVS-GRE-tunnel.py</a>
<li><a href="README">README</a>
<li><a href="topo-2sw-2host.py">topo-2sw-2host.py</a>
<li><a href="topo-2sw-6host.py">topo-2sw-6host.py</a>
</ul>
<hr>
</body>
</html>
100%[======================================>] 372         --.-K/s   in 0s

2017-08-03 09:30:52 (19.8 MB/s) - written to stdout [372/372]
```

Gambar 13. Koneksi *client server* h1 dan h6

## 6. Kesimpulan

Hasil dari pengujian menunjukan semua *host* yang berada pada jaringan lokal yang satu dengan jaringan lokal yang lain dapat terhubung dengan baik. Pengujian dilakukan dengan melakukan proses *ping* dari satu *host* pada jaringan lokal satu dengan jaringan lokal lainnya. Dilakukan juga pengujian *client server* dengan hasil yang baik. Penggunaan protokol GRE pada Open vSwitch dapat berjalan baik sehingga dapat menjadi solusi penggunaan perangkat jaringan yang lebih murah dibandingkan dengan perangkat jaringan khusus yang menjalankan protokol GRE. Konfigurasi GRE pada Open vSwitch dapat dilakukan dengan mudah tanpa diperlukan konfigurasi tambahan yang rumit. Pada penelitian ini tidak dilakukan pengujian *throughput* karena jaringan yang digunakan merupakan simulasi pada mininet dimana kinerjanya bergantung pada perangkat komputer yang digunakan.

## Daftar Pustaka


[1] Faycal Bensalah, Najib El Kamoun and Ayoub BAHNASSE, *"Analytical performance and Evaluation of the Scalability of Layer 3 Tunneling Protocols: Case of Voice Traffic Over IP"*, IJCSNS International Journal of Computer Science and Network Security. Vol.17, No. 4, pp. 361-369, April 2017.

[2] Zhao Aqun, Yuan Yuan, Ji Yi and Gu Guanqun, *"Reserach on Tunneling Techniques in Virtual Private Networks"*, International Conference on Communication Technology Proceedings, Vol. 1, pp. 691-697, 2000.

[3] RFC2784. https://www.ietf.org/rfc/rfc2784.txt.

[4] RFC2661. https://www.ietf.org/rfc/rfc2661.txt.

[5] RFC2637. https://www.ietf.org/rfc/rfc2637.txt.

[6] RFC1075. https://www.ietf.org/rfc/rfc1075.txt.

[7] Aria. Asadi Eskandar, Mahbubur. R. Syed, Bahareh. Zarei. M, *"Performance Analysis of VOIP over GRE Tunnel"*, Computer Network and Information Security, pp. 1-9, Desember 2015.
[8] Paul Ferguson and Geoff Huston, *"What is a VPN?"*, https://www.potaroo.net/papers/1998-3-vpn/vpn.pdf, Revision 1, April 1998.
[9] B. Pfaff and B. Davie, *"The Open vSwitch Database Management Protocol"*, RFC 7047. 2013.
[10] Marta Carbone, Gaetano Catalli and Luigi Rizzo, "*Improving the performance of Open vSwitch"*, EuroBSDcon, May 30, 2011.
[11] Yunchun Li and Guodong Wang, *"SDN-Based Switch Implementation on Network Processors"*, Communications and Network, Vol 5, pp. 434-437, June 2013.
[12] B. Pfaff, J. Pettit, T. Koponen, K. Amidon, M. Casado, and S. Shenker, *"Extending Networking into the Virtualization Layer"*, In Proc. of HotNets, October. 2009.
[13] Voravit Tanyingyong, Markus Hidell and Peter Sjodin, *"Using Hardware Classification to Improve PC-Based OpenFlow Switching"*, IEEE 12th International Conference on High Performance Switching and Routing, July 2011.
[14] Andrea Bianco, Robert Birke, Luca Giraudo, Manuel Palacin, *"Openflow Switching: Data Plane Performance"*, IEEE International Conference on Communications (ICC), pp. 1-5, 2010.
[15] Jad Naous, David Erickson, G. Adam Covington, Guido Appenzeller, and Nick McKeown, *"Implementing an OpenFlow Switch on the Netfpga Platform,"* ANCS: Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, pp. 1-9, November 2008.
[16] OpenFlow. https://www.opennetworking.org.
[17] Bob Lantz, Brandon Heller, Nick McKeown, *"A Network in a Laptop: Rapid Prototyping for Software-Defined Networks"*, SIGCOMM, August 2015.
[18] Eueung Mulyana, *"Buku Komunitas SDN-RG"*, GitBook.

ISSN (e): 2540-9123
ISSN (p): 2502-1982