PROTOTIPE SWITCH OPENFLOW SOFTWARE-BASED MIKROTIK RB750 PAKET OPENFLOW VS OPENvSWITCH FIRMWARE

Rikie Kartadie [(1)]
Pendidikan Teknologi Informasi, STKIP PGRI Tulungagung,
Jl. Mayor Sujadi Timur no 7, Tulungagung
e-mail : rikie.kartadie@stkippgritulungagung.ac.id

Abstrak
Pengimplementasian arsitektur SDN/OpenFlow membutuhkan biaya yang tinggi, walaupun penggunaan emulator mininet memungkinkan, namun kenyataannya dalam implementasinya membutuhkan hardware khusus. Switch OpenFlow Software-based dengan menggunakan firmware OpenWRT dan hardware TPLink WR142nd telah di uji performanya pada implementasi skala jaringan kampus, dalam artikel yang diterbitkan penulis menyatakan bahwa tidak ada perbedaan yang signifikan antara switch OpenFlow bebasis OpenWRT pada TPLink dengan performa yang dihasilkan oleh mininet. Dari artikel tersebut maka menjadi pertanyaan bagaimana performa dari switch OpenFlow software-based berbasis MikroTik. MikroTik ini telah menambahkan OpenFlow agent pada OS versi 6.17 pada RouterOS nya dan memungkinkan untuk diimplementasikannya arsitektur SDN/OpenFlow dengan biaya yang lebih terjangkau, implementasi OpenFlow agent pada RouterOS MikroTik layak untuk diuji performanya untuk dibandingkan dengan perubahan secara total firmware MikroTik yang berjalan dengan firmware lain dan menjalankan OpenvSwitch. Kontroller yang digunakan dalam penelitian ini adalah kontroller Floodlight, data pembanding adalah data mininet dengan topologi yang merupakan representasi dari topologi yang ada di STKIP PGRI Tulungagung. Switch OpenFlow software-based berbasis MikroTik dengan penambahan agen OpenFlow di RouterOS, hasil uji- t menunjukkan bahwa tidak memiliki perbedaan yang signifikan pada uji througput baik UDP maupun TCP dibandingkan dengan mininet, namum tetap memiliki perbedaan yang signifikan pada uji latency.

***Kata Kunci** : SDN, Switch OpenFlow, MikroTik.*

1. PENDAHULUAN

Jaringan biasanya dibangun dari sejumlah besar perangkat jaringan seperti router, switch dan perangkat lainnya. Setiap perangkat menjalankan manipulasi penerusan paket data, dengan protokol yang komplek yang tertanam di dalam perangkat tersebut. Operator jaringan bertanggungjawab secara langsung untuk melakukan konfigurasi, aturan, dan tidak jarang hingga ke aplikasi yang digunakan didalam jaringan. Operator biasanya melakukan konfigurasi secara manual pada setiap perangkat yang terhubung, hal ini memberikan celah pada kesalahan konfigurasi karna kesalahan manusia (human error) terutama bila harus menangani jumlah perangkat yang banyak.

Antisipasi kondisi tersebut terjawab dengan munculnya arsitektur dan protokol baru yag disebut SDN / OpenFlow. *Software-Defined Network* (SDN). *Software-Defined Network* muncul dari penelitian pada tahun 2004 sebagai bagian dari penelitian paradigma manajemen jaringan baru, yang menghasilkan 2 hasil yang berbeda: pekerjaan *platform routing control (RCP 40)* yang selesai pada Princeton and Carnegie Mellon University yang dikerjaan oleh Caesar dkk pada tahun 2005, dan keamanan jaringan *SANE Ethene project* yang diselesaikan di Stanford University dan University of California oleh Casado dkk pada tahun 2006 (P. A. Morreale dan James M. Anderson, 2015).

Keterkaitan antara *software*, dan *hardware* pada SDN/OpenFlow sangatlah kental. Switch OpenFlow menggunakan sebuah struktur data yang didalamnya terdapat struktur database dan sistem pengambilan keputusan yang cukup kompleks untuk menentukan kemana paket data kan dikirim, sehingga hal ini bisa mempengaruhi kinerja dari setiap perangkat. Pengimplementasian arsitektur SDN/OpenFlow membutuhkan biaya yang tinggi, walaupun penggunaan emulator mininet memungkinkan, namun dalam kenyataannya implementasi membutuhkan *hardware*. Penelitian sebelumnya telah diuji performa menggunakan perangkat keras OpenWRT yang diberikan OpenFlow *agent* sehingga dapat menjalankan fungsi OpenFlow (R. Kartadie, 2014).

Switch OpenFlow Software-based dengan menggunakan firmware OpenWRT dan hardware TPLink WR142nd telah di uji performanya pada implementasi skala jaringan kampus, dalam artikel yang diterbitkan penulis menyatakan bahwa tidak ada perbedaan yang signifikan antara switch OpenFlow bebasis OpenWRT pada TPLink dengan performa yang dihasilkan oleh mininet (R. Kartadie, et al., 2018). Dari artikel tersebut maka menjadi pertanyaan bagaimana performa dari switch OpenFlow software-based berbasis MikroTik. MikroTik ini telah menambahkan OpenFlow *agent* pada OS versi 6.17 pada RouterOS nya dan memungkinkan

untuk diimplementasikannya arsitektur SDN/OpenFlow dengan biaya yang lebih terjangkau. Implementasi OpenFlow *agent* pada RouterOS MikroTik layak untuk diuji performanya untuk dibandingkan dengan perubahan secara total firmware MikroTik yang berjalan dengan firmware lain dan menjalankan OpenvSwitch.

Dari pernyataan diatas, maka dapat dirumuskan beberapa permasalahan yang dapat diangkat, diantaranya adalah; bagaimana unjuk kerja (performa) dari *switch SDN/OpenFlow software-based* berbasis MikroTik yang telah ditanamkan OpenFlow *agent* pada RouterOS MikroTik dan bagaimana performanya bila seluruh firmware-nya di ganti dengan firmware yang berbeda dalam hal ini OpenWRT dengan menambahkan paket OpenVSwitch.

Dalam penelitian ini tidak dibahas tentang keamanan dan kebijakan jaringan kampus, kontroler yang digunakan adalah kontroler floodlight dan performa yang diuji adalah througput dan latency.

2. TINJAUAN PUSTAKA

Chunk Y. EE., 2012, menyatakan bahwa, OpenFlow dapat diimplementasikan pada NetFTPGA, dijadikan sebuah firewall dan dapat dimonitoring menggunakan wireshark. Perbedaan mendasar pada penelitian ini adalah pada perangkat penelitian yang digunakan, penelitian ini menggunakan perangkat MikroTik RB750 menggantikan fungsi NetFPGA yang digunakan oleh Chunk Yik, sehingga bila ternyata berhasil, implementasi pada infrastruktur jaringan tidak menjadi beban biaya yang besar, karena tidak terjadi perubahan yang mendasar pada hardware infrastruktur yang telah ada.

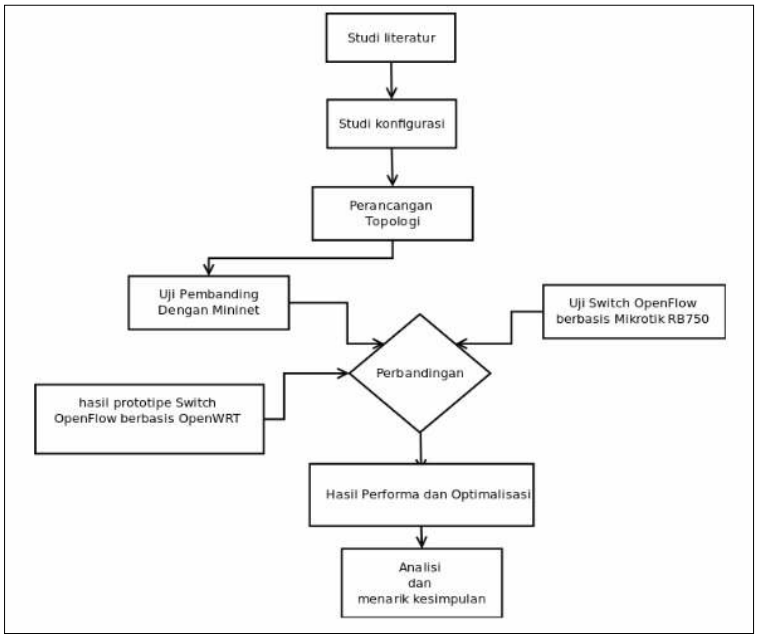
Applement, M. dan De Boer, M., 2012, melakukan analisis performa terhadap hardware openflow. Hardware openflow adalah seperti NetFPGA card, Pica8 OpenFlow on a Pronto switch dan the Open vSwitch. Pengujian dilakukan pada beberapa variabel diantaranya QoS, Port Mirroring, failovel speed dan performace overhead. Perbedaan mendasar pada penelitian ini adalah dari jenis hardware yang digunakan, penelitian ini menggunakan hardware software-based openflow switch dengan perangkat routerboard MikroTik RB750.

Uji performa switch OpenFlow berbasis OpenWRT dengan TPLink, pada penelitian sebelumnya (R.Kartadie, et al, 2018) menyatakan tidak ada perbedaan signifikan dengan performa yang dihasilkan mininet. Pada penelitian ini, uji akan diadakan pada switch OpenFlow berbasis MikroTik dengan tambahan pada RouterOS dan perubahan keseluruhan dari firmware yang dijalankan.

Hasil uji switch OpenFlow berbasis OpenWRT pada penelitian sebelumnya Performa switch OF software-based dapat dikatakan baik dengan hasil gap rata-rata pada setiap pengujian menunjukkan angka yang tidak tinggi. Bahkan pada pengujian Latency dapat dikatakan sama (R.kartadie dan B. Satya, 2015). Pada penelitian ini, akan diuji pengujian yang sama namun dengan prototipe yang berbeda.

3. METODE PENELITIAN

Langkah penelitian yang akan dilakukan terlihat pada gambar 1 dibawah ini. Skema ini menjadi acuan dalam melaksanakan peneltian.



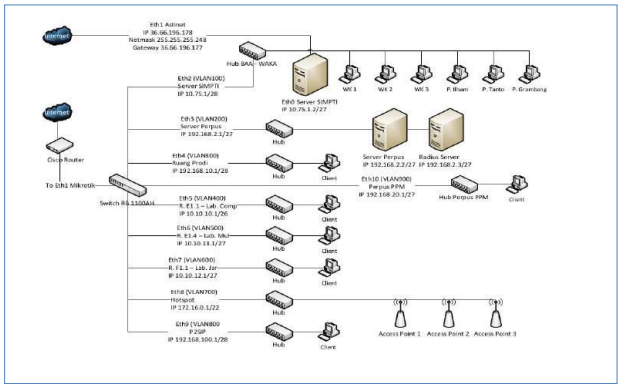
Gambar 1. Langkah Penenelitian

Penelitian yang dilakukan merupakan penelitian laboratorium yang diadakan di Laboratorium kampus STKIP PGRI Tulungagung, periode penelitian adalah pada bulan maret hingga bulan September 2018. Pada penelitian ini, topologi yang digunakan adalah topologi yang sama yang digunakan pada kampus STKIP PGRI Tulungagung. Adaapun langkah penelitian adalah (1) **Studi literatur,** mengumpulkan bahan atau materi penelitian yang berupa literatur/referensi yang berhubungan dengan penelitian yang akan dilaksanakan. Peneliti telah melaksanakn pengumpulan literatur yang diperlukan baik berupa buku dan artikel ilmiah yang berkenaan dengan penelitian yang dilaksanakan. (2) **Studi konfigurasi**, mempelajari konfigurasi hardware dan software dari kontroler yang akan digunakan, baik cara konfigurasi, spesifikasi dan *troubleshoot*-nya. (3) **Evaluasi yang digunakanTopologi,** Melakukan evaluasi topologi yang digunakan dalam penelitian dan yang akan digunakan dalam implementasi. (4) **Uji pembanding dengan mininet**, melakukan pengujian dengan mininet sebagai data pembanding. Mininet merupakan emulator Infrastruktur SDN/OpenFlow yang mengunakan resource Laptop (K. Kaur, J. Singh, dan N. S. Ghumman, 2014). Laptop yang digunakan untuk mengambil data dari mininet dengan spesifikasi sebagai berikut : CPU Inter i3-4005U 170GHzx4, Ram 4Gb. Dan menggunakan Virtual box dalam menjalankan mininet sebagai switch OpenFlow. Controller Floodlight di install pada host laptop. (5) **Pembuatan Prototipe dengan RouterBoard RB750,** dilakukan dengan 2 cara yaitu (a) Menambahkan agent/paket OpenFlow pada MikroTik RouterOS dengan paket openflow. Paket openflow ini, sesuai dengan informasi dari MikroTik masih merupakan paket non official. (b) Prototipe dengan merubah firmware yang ada dengan firmware OpenWRT dan di tambahkan paket OpenvSwitch. (6) **Melakukan pengujian dengan switch dengan agent/paket OpenFlow pada MikroTik RouterOS dengan paket openflow.** Prototipe di implemantasikan pada lingkungan kampus peneliti dan di ambil data performa secara periodik. Pengujian dilakuna pada *latency, througput* dan *packet lost* dengan melakukan ping dengan besar paket ICMP tertentu ke masing-masih host, sebelum dan sesudah dijalankan OpenFlow protokol. **Pengujian performa prototipe** dengan jperf pada protokol TCP dan UDP, hasil uji dengan besar paket tertentu dan membandingkannya dengan hasil uji pada mininet. **Menganalisa hasil prototipe** yang dibangun, dengan menggunakan aplikasi wireshark yang sebelumnya telah ditambahkan OpenFlow protokol *agent*, sehingga dapat memonitor paket OpenFlow yang lewat didalam prototipe. (7) **Melakukan pengujian dengan switch perubahan firmware yang ada dengan firmware OpenWRT dan di tambahkan paket OpenvSwitch.** Prototipe di implemantasikan pada lingkungan kampus TPP dan di ambil data performa secara periodik. MikroTik RB750 yang digunakan adalah perangkat yang berbeda dengan yang diuji pada langkah 6 sebelumnya. Pengujian dilakukan pada *latency, througput* dan *packet lost* dengan melakukan ping dengan besar paket ICMP tertentu ke masing-masih host, sebelum dan sesudah dijalankan OpenFlow protokol. **Pengujian performa prototipe** dengan jperf pada protokol TCP dan UDP, hasil uji dengan besar paket tertentu dan membandingkannya dengan hasil uji pada mininet. **Menganalisa hasil prototipe** yang dibangun, dengan menggunakan aplikasi wireshark yang sebelumnya telah ditambahkan OpenFlow protokol *agent*, sehingga dapat memonitor paket OpenFlow yang lewat didalam prototipe.

3.1. Gambaran Umum Lokasi Penelitian

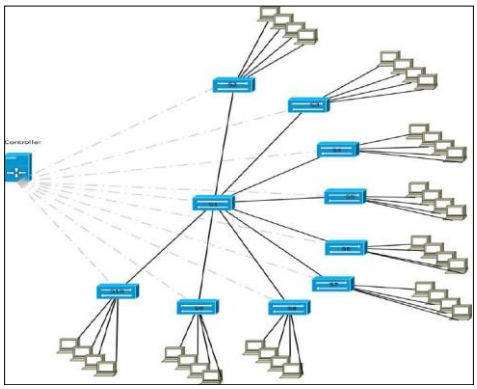
Dalam penelitian ini, peneliti menggunakan topologi yang digunakan pada lokasi peneliti yaitu STKIP PGRI Tulungagung, walau tidak seluruhnya menggambarkan topologi seutuhnya. STKIP memiliki beberapa Node switch yang melayani host/user yang tersebar diseluruh lokasi kampus. 70% dari penggunaan jaringan menggunakan wireless, sehingga dalam penelitian ini, peneliti hanya memfokuskan pada jaringan kabel yang berada di lingkungan kampus STKIP PGRI Tulungagung.

Topologi jaringan kabel yang digunakan di kampus STKIP PGRI Tulungagung adalah seperti terlihat pada gambar 2 dibawah ini.



Gambar 2. Topologi Kampus STKIP PGRI Tulungagung

Topologi yang terlihat pada gambar 2 diatas kemudian diduplikasi dengan topologi yang akan digunakan pada penelitian ini. Duplikasi yang dilakukan tanpa menduplikasi alamat IP demi mempermudah peneliti dalam melakukan penelitian. Adapun topologi yang digunakan dalam penelitian adalah seperti yang terlihat pada gambar 3 dibawah ini.



Gambar 3. Topologi Penelitian

3.2. Kontroler Yang Digunakan

Kontroler memegang peranan penting pada jaringan SDN/OpenFlow. Telah banyak kontroler yang beredar, baik yang bersifat *opensource* maupun yang bersifat *closesource* atau berbayar. Menurut Muntaner (2012), kontroler merupakan faktor utama dalam jaringan SDN/OpenFlow. Karna pentingnya kontroler dalam penelitian ini, maka sebelum melakukan pemilihan kontroler, peneliti melakukan penelitian pendahulu (studi literatur) yang berkenaan dengan kontroler.

3.3. Menjalankan mininet

Topologi yang telah dirancang sebelumnya, akan dijalankan pada mininet. Mininet menjalankan topologi *custom* (10 buah *switch* dengan 4 buah *host* per *switch*). *Coding* yang dijalankan dengan bahasa *python* yang dapat di eksekusi oleh mininet, adapun *coding* untuk topologi yang telah dirancang adalah sebagai berikut :

```
"""
Topologi prototipe
Sepuluh switch terkoneksi langsung, dengan masing-masing
switch terkoneksi dengan 6 host:
-rikie kartadie.-
============================================================
"""
from mininet.topo import Topo
class MyTopo( Topo ):
"Topologi prototipe."
def __init__( self ):
"Membuat topologi."
# Initialize topology
Topo.__init__( self )
# Add hosts and switches
"""
Switch 1 dan 2
"""
hs1 = self.addHost( 'hs1' )
hs2 = self.addHost( 'hs2' )
hs3 = self.addHost( 'hs3' )
hs4 = self.addHost( 'hs4' )
hs5 = self.addHost( 'hs5' )
hs6 = self.addHost( 'hs6' )
hs7 = self.addHost( 'hs7' )
hs8 = self.addHost( 'hs8' )
"""
Switch 3 dan 4
"""
hs9 = self.addHost( 'hs9' )
hs10 = self.addHost( 'hs10' )
hs11 = self.addHost( 'hs11' )
hs12 = self.addHost( 'hs12' )
.... coding dipotong ....
# Add links
self.addLink( Switch1, hs1 )
```

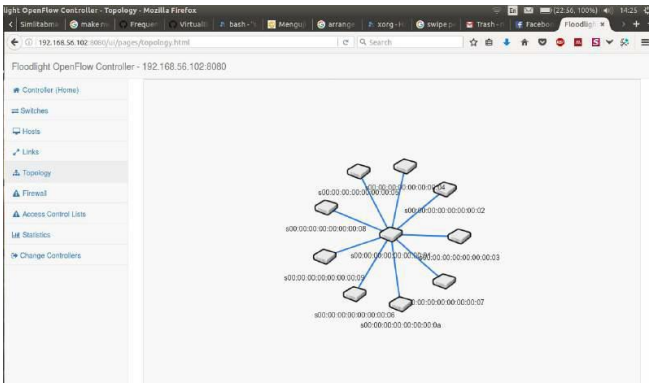
```
self.addLink( Switch1, hs2 )
self.addLink( Switch1, hs3 )
self.addLink( Switch1, hs4 )
self.addLink( Switch2, hs5 )
self.addLink( Switch2, hs6 )
self.addLink( Switch2, hs7 )
self.addLink( Switch2, hs8 )
self.addLink( Switch3, hs9 )
self.addLink( Switch3, hs10 )
self.addLink( Switch3, hs11 )
... coding dipotong ...
topos = { 'mytopo': ( lambda: MyTopo() ) }
```

Coding dapat diuji dengan menjalankan *coding* tersebut dengan perintah seperti dibawah ini:

```
sudo mn --custom ~/mininet/custom/topo_hibah.py --topo mytopo
```

Pada perintah diatas dapat dijelaskan bahwa, mininet berjalan pada *mode root*, *--custom* adalah memerintahkan mininet untuk menjalankan topologi yang dibuat dengan nama *file topo_hibah.py* yang berada pada *folder* mininet/custom, dan *--topo* memerintahkan mininet melakukan pembuatan topo sesuai dengan topologi yang ada.

Adapaun hasil *coding* yang dijalankan, dan telah di hubungkan dengan kontroller dapat dilihat pada gambar 4 dibawah ini;



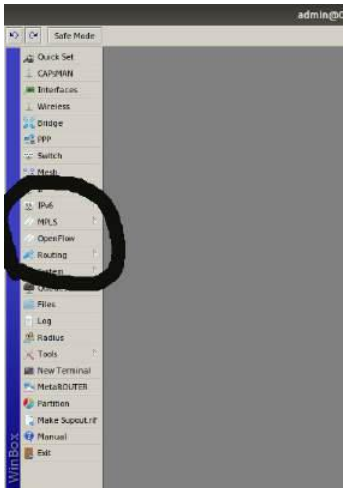
Gambar 4. ***Topologi uji (non-host) yang digambarkan oleh floodlight***

3.4. Pembuatan Prototipe

Setelah mininet disimulasikan dengan variabel yang ada, langkah selanjutnya adalah membuat prototipe jaringan menggunakan *RouterBoard RB750*. *Switch* yang digunakan adalah RB750 sebanyak 10 buah dan disusun berdasarkan topologi penelitian yang terlihat pada gambar 3. Prototipe yang dibangun adalah prototipe *Switch OpenFlow software-base*.

3.4.1. Memodifikasi RB750 Dengan Cara Pertama (prototipe 2A)

Secara fisik, Routerboard RB750 ini memiliki 4 port ethernet, dan memiliki punya fitur yang cukup baik. Langkah menambahkan paket OpenFlow : (1) Versi firmware mikrotik RB750 dipastikan adalah versi RouterOS 6.17. Pada *wiki.mikrotik.com*, dikatakan bahwa RouterOS versi inilah yang dapat ditambahkan paket OpenFlow. (2) Setelah dilakukan upgrade RouterOS, dapat kemudian menambahkan paket OpenFlow. (3) Penambahan paket dapat dilakukan dengan cara login menuju winbox, dan masuk ke file. *Drag-drop* file paket Openflow, kemudian reboot RouterBoard. (4) Setelah langkah diatas dijalankan, kita akan mendapati tab baru yang berisi paket Openflow yang siap kita konfigurasi. Hasil penambahan paket OpenFlow dapat dilihat pada gambar 5 dibawah ini. Cara pertama ini kemudian kita sebut sebagai prototipe 2A.



Gambar 5. ***Hasil penambahan paket OpenFlow pada RouterOS***

3.4.2. Memodifikasi RB750 Dengan Cara Kedua (prototipe 2B)

Memodifikasi *firmware* dengan cara kedua ini, adalah dengan mengganti firmware yang telah ada dengan firmware OpenWRT. Penggantian firmware RouterOS dilakukan dengan metode flashing secara langsung. Dikarnakan di MikroTik RB750 tidak ada pin serial, maka *flashing* dilakukan dengan metode menghapus kernel dan menggantinya secara langsung.

Langkah yang ditempuh adalah sebagai berikut : (1) Persiapan yang harus dilaksanakan adalah dengan menyiapkan image OpenWRT yang akan di-*flashing* ke MikroTik RB750. Pada penelitian ini digunakan OpenWRT versi 17.01.0 (*code name* : LEDE). (2) Komputer yang digunakan untuk menjalankan *flashing* disiapkan dengan instalasi DHCP server dan TFTP server. Kedua aplikasi ini digunakan untuk menambahkan image ke dalam MikroTik RB750. (3) Booting Routerboard, pada saat sebelum menghubungkan Routerboard dengan dengan catu daya, tombol reset (RES) di tekan terus dan Routerboard dihubungkan dengan catudaya dan tombol tetap ditekan. Setelah beberapa detik, led indikator ACT akan berkedip dan kemudian padam. Routerboard akan berusaha untuk mendapatkan alamat IP dari DHCP server. Pada gambar 6 dapat dilihat struktur board dari MikroTik RB750. Cara kedua ini kemudian kita sebut sebagai prototipe 2B.

Gambar 5. Struktur board pada MikroTik RB750

Flashing yang dilakukan melewati beberapa tahap, setelah berhasil masuk ke dalam boot dari Routerboard, dilakukan pengcekan posisi block memori yang akan dirubah, sebelumnya kita dapat masuk ke routerboard dengan telnet ke ip default 192.168.1.1 yang berada di port 2. Pengecekan memori dilakukan dengan perintah;

```
root@OpenWrt:~# cat /proc/mtd
```

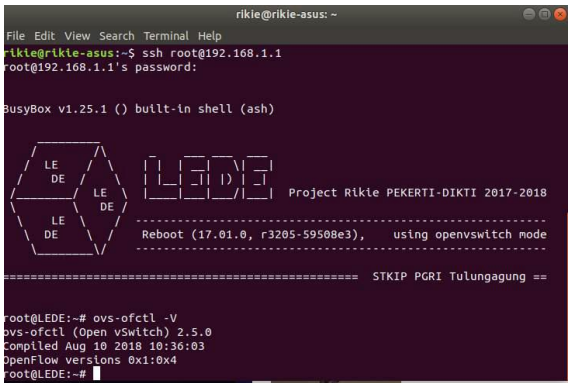
output yang dihasilkan adalah sebagai berikut ;

```
dev: size erasesize name
mtd0: 00040000 00004000 "booter"
mtd1: 003c0000 00004000 "kernel"
mtd2: 03c00000 00004000 "rootfs"
```

kemudian kernel dan rootfs dihapus dengan dengan perintah mtd (*memory technology driver*). Perintah ini di jalankan untuk memberikan ruang di routerboard untuk menginstall firmware baru. Bila tidak dilakukan maka ruang yang ada tidak cukup untuk menambahkan firmware. Perintah yang diberikan adalah;

```
mtd erase kernel
mtd erase rootfs
```

setelah memori di hapus, maka flashing bisa dilakukan dengan metode tftp dengan sebelumnya mengaktifkan `$ python -m SimpleHTTPServer` pada komputer. Setelah itu flashing dilaksanakan dengan perintah wget2nand. Hasil flasihing yang dilakukan dapat dilihat pada gambar 7 dibawah ini, kemudian bisa di lanjutkan dengan penambahan paket openvswitch.



Gambar 6. ***Hasil flashingOpenWRT LEDE dan OpenvSwitch pada Routerboard RB750***

Pada gambar 6 adalah gambar dimana OpenWRT yang telah diberikan paket OpenvSwitch. Versi OpenvSwitch yang digunakan adalah versi 2.5.0 dan protokol OpenFlow versi 1.4.

4. HASIL DAN PEMBAHASAN

4.1. Data Througput dan Latency

Data diambil secara periodik setiap hari selama 10 minggu dengan besar data uji mulai dari 64byte hingga 8192byte, namun data akan disajikan dalam rata-rata 10 minggunya. Data throughput yang dihasilkan oleh mininet sebagai data pembanding dapat dilihat pada tabel 1, 2 dan 3 dibawah ini :

Tabel 1. Data througput UDP Mininet Rata-rata

UDP	Besar paket data (Kbyte/sec)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	14.390	15.600	33.854	69.771	144.542	156.921	177.000	187.788
2	14.090	15.300	33.133	69.331	140.200	173.101	173.053	184.376
3	14.230	15.440	33.094	70.167	143.110	158.091	176.000	185.968
4	14.010	15.220	32.632	69.197	140.654	151.000	172.187	183.466
5	13.800	15.010	32.191	68.271	140.976	149.279	164.876	175.790
6	12.977	15.000	32.170	68.227	141.654	150.550	162.116	172.892
7	12.880	15.345	32.800	69.550	141.857	156.712	173.053	183.376
8	12.981	14.191	30.471	64.659	140.665	135.037	149.211	159.341
9	12.120	13.330	28.663	60.862	144.542	137.313	151.714	161.969
10	12.089	15.011	32.193	68.276	140.357	150.78	169.923	181.089

Tabel 2. Data Througput TCP Mininet Rata-rata

TCP	Besar paket data (Kbyte/s)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	91.984	91.983	92.085	91.533	92.075	90.639	90.966	91.962
2	92.042	90.586	91.421	91.872	91.294	90.763	91.022	92.062
3	91.963	93.042	92.961	92.982	91.874	91.659	92.288	91.656
4	91.475	92.142	92.236	91.012	91.644	92.273	91.942	92.966
5	92.963	92.845	92.983	92.481	91.686	91.783	93.074	93.048
6	92.983	93.094	92.942	93.021	90.573	92.073	92.966	93.077
7	91.633	92.957	92.973	93.073	91.943	92.155	91.633	91.631
8	92.984	92.963	91.576	92.665	90.852	91.422	91.474	93.181
9	90.962	90.944	90.942	91.482	90.564	90.969	90.961	91.132
10	91.983	92.074	91.672	91.312	91.982	92.055	91.652	91.762

Tabel 3. Data Latency Mininet Rata-rata

	besar paket data (msec)							
Switch	64	128	256	512	1024	2048	4096	8192
1	2.228	3.999	4.855	5.191	4.341	4.736	4.477	9.149
2	2.341	3.193	4.416	5.787	4.436	4.773	4.515	9.44
3	2.221	3.211	4.117	5.433	4.499	4.554	4.655	9.765
4	2.611	3.566	4.567	5.011	4.321	5.145	4.66	9.432
5	2.288	3.458	4.532	4.65	5.031	4.555	4.789	9.435
6	2.581	3.413	4.765	4.667	4.432	4.872	4.123	9.444
7	2.344	3.871	5.376	4.432	4.333	4.321	5.011	9.123
8	2.731	3.876	4.566	4.32	4.547	4.387	5.811	9.812
9	2.444	3.431	5.111	4.567	4.556	4.21	4.667	9.456
10	2.553	3.67	4.353	5.001	4.678	5.091	4.558	9.112

Tabel 1, 2, dan 3 merupakan data rata-rata yang diperoleh yang digunakan sebagai data pembanding yang akan digunakan. Tabel 1 menunjukkan besaran throuput UDP yang diperoleh oleh emulator mininet, sedangkan pada tabel 2 merupakan througput TCP mininet yang diperoleh. Sedangkan tabel 3 merupakan nilai latency yang diperoleh mininet. Adapun data througput UDP, TCP dan Latency untuk prototipe 2A dapat dilihat pada tabel 4, 5 dan 6 berikut ini, dimana data diperoleh dengan perlakuan yang sama yang dilakukan pada emulator mininet.

Tabel 4. Data Througput UDP Prototipe 2A

UDP	besar paket data (Kbyte/sec)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	13.331	16.41	31.331	67.555	100.221	122.299	143.429	150.175
2	13.44	12.222	33.761	66.789	101.022	123.151	139.782	152.642
3	16.111	15.444	32.444	69.774	101.889	129.667	138.288	153.362
4	14.116	15.005	32.566	68.775	102.423	124.869	139.668	163.679
5	13.988	16.445	31.677	64.664	102.123	131.754	140.258	160.896
6	14.667	16.992	33.122	66.544	103.977	135.280	148.976	151.620
7	13.77	14.667	32.188	68.556	104.311	136.554	151.900	152.620
8	16.001	16.421	33.221	67.554	100.656	134.861	152.154	153.130
9	15.411	17.221	30.111	66.559	100.899	139.000	151.845	155.860
10	14.001	15.447	31.445	66.55	103.331	134.898	149.424	150.118

Tabel 5. Data Througput TCP Prototipe 2A

TCP	Besar paket data (Kbyte/sec)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	78.894	81.673	81.75	82.289	81.76	81.229	81.656	81.752
2	76.992	81.76	82.021	83.623	82.984	79.453	80.812	82.752
3	77.523	81.011	82.654	84.457	80.164	79.349	80.978	82.146
4	78.615	82.832	82.926	83.562	82.001	80.163	80.532	82.956
5	78.653	82.575	83.073	83.371	81.716	80.173	79.642	83.338
6	82.673	80.764	83.132	84.111	81.263	79.763	81.656	82.967
7	76.283	81.607	82.763	83.863	80.633	78.845	81.823	80.321
8	80.74	81.683	80.266	83.653	80.542	80.132	81.964	82.071
9	80.659	80.934	83.332	82.572	82.254	79.559	79.651	81.022
10	81.073	81.864	81.362	83.197	81.172	80.745	82.342	82.152

Tabel 6. Data Latency Prototipe 2A

	besar paket data (msec)							
Switch	64	128	256	512	1024	2048	4096	8192
1	3.122	6.645	6.113	6.885	6.878	7.648	8.113	11.721
2	4.311	5.152	5.968	5.645	6.734	6.989	7.977	11.841
3	3.145	5.822	6.167	6.166	6.77	7.041	7.891	11.852
4	3.845	5.611	6.223	6.858	6.921	7.12	8.004	11.117
5	4.167	6.866	6.931	7.997	7.023	7.132	8.154	11.155
6	3.967	5.944	5.187	7.867	6.089	6.944	7.899	11.009
7	3.965	6.746	6.237	6.976	6.931	7.345	7.985	11.748
8	4.541	6.965	6.177	7.674	6.966	7.199	7.178	12.066
9	4.333	7.178	6.812	7.09	6.789	6.979	7.912	12.191
10	3.989	6.012	6.134	6.476	7.5	7.112	8.156	11.286

Dari tabel 4, 5 dan 6, data yang diperoleh dengan perlakuan yang sama. Pengambilan data dilakukan dengan besar paket data dari 64 kbps hingga 8192 kbps. Demikian pula dengan prototipe 2B, data yang diperoleh dapat dilihat pada tabel 7, 8 dan 9 di bawah ini.

Tabel 7. Data Througput UDP prototipe 2B

UDP	besar paket data (Kbyte/sec)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	13.331	16.557	31.877	66.554	100.223	122.987	144.120	150.677
2	13.440	13.600	34.654	66.789	100.567	124.101	140.443	153.330
3	14.310	15.789	34.678	68.220	101.331	130.010	138.987	153.786
4	14.220	16.000	33.788	68.019	101.441	125.444	140.000	164.011
5	15.112	16.778	33.456	65.443	102.331	130.786	140.344	162.450
6	14.321	18.000	33.897	68.767	102.897	136.011	149.780	152.333
7	14.678	16.550	31.560	69.776	104.331	136.786	151.900	155.113
8	16.030	17.440	33.221	68.434	100.101	135.988	153.221	144.000
9	15.555	17.550	36.567	66.111	100.897	138.880	152.011	156.760
10	14.677	17.556	34.550	67.541	104.323	133.877	150.455	160.031

Tabel 8. Data Througput TCP prototipe 2B

TCP	Besar paket data (Kbyte/sec)							
Switch ke	64	128	256	512	1024	2048	4096	8192
1	83.623	81.673	81.75	82.289	81.76	81.229	81.656	81.263
2	84.457	81.76	82.021	83.623	82.984	79.453	80.812	80.633
3	83.562	81.011	82.654	82.752	80.164	79.349	80.978	80.542
4	83.371	82.832	82.926	82.146	82.001	80.163	80.532	82.254
5	84.111	79.453	83.073	82.956	81.716	80.173	79.642	80.132
6	83.863	79.349	83.132	83.338	81.263	79.763	81.656	82.967
7	76.283	80.163	82.763	82.967	80.633	78.845	81.823	80.321
8	80.74	80.173	80.266	83.653	80.542	80.132	81.964	82.071
9	80.659	79.763	83.332	82.572	82.254	79.559	79.651	81.022
10	81.073	81.864	81.362	83.197	81.172	80.745	82.342	82.152

Tabel 9. Data Latency prototipe 2B

	besar paket data (msec)							
Switch	64	128	256	512	1024	2048	4096	8192
1	3.122	6.645	6.113	6.885	6.878	7.648	5.968	5.645
2	4.311	5.152	5.968	5.645	6.734	6.989	6.167	6.166
3	3.145	5.822	6.167	6.166	6.77	7.041	6.223	6.858
4	3.845	5.611	6.223	7.132	8.154	7.12	6.931	7.997
5	4.167	6.866	6.931	6.944	7.899	7.132	5.187	7.867
6	3.967	5.944	5.187	7.345	7.985	6.944	7.891	11.852
7	3.965	6.746	6.237	7.199	7.178	7.345	8.004	11.117
8	4.541	6.965	6.177	6.979	7.912	7.199	8.154	11.155
9	4.333	7.178	6.812	7.041	6.223	6.979	7.899	11.009
10	3.989	6.012	6.134	6.476	7.5	7.112	8.156	11.286

4.2. Uji Simple Independent t Test

Dari tabel 1 hingga tabel 9, kemudian dilakukan uji t (uji beda) untuk menentukan apakah ada perbedaan yang signifikan dari hasil yang diperoleh baik dari prototipe 2A maupun 2B dengan data pembandingnya yaitu data mininet. Hasil simple independent t test yang diperoleh antara prototipe 2A dengan mininet, dapat dilihat pada tabel 10 dibawah ini,

Tabel 10. Simple Independent t Test UDP Proto 2A

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	.30	.589	-2.67	18.00	.016	-1.13	.42	-2.01	-.24
	Equal variances not assumed			-2.67	17.49	.016	-1.13	.42	-2.02	-.24
x2.2	Equal variances assumed	3.27	.087	-1.34	18.00	.198	-.68	.51	-1.76	.39
	Equal variances not assumed			-1.34	12.76	.205	-.68	.51	-1.79	.42
x2.3	Equal variances assumed	.23	.638	-.11	18.00	.911	-.07	.59	-1.30	1.16
	Equal variances not assumed			-.11	16.32	.911	-.07	.59	-1.31	1.17
x2.4	Equal variances assumed	1.85	.190	.49	18.00	.632	.50	1.02	-1.65	2.65
	Equal variances not assumed			.49	13.24	.634	.50	1.02	-1.71	2.71
x2.5	Equal variances assumed	.23	.635	57.56	18.00	.000	39.77	.69	38.32	41.22
	Equal variances not assumed			57.56	17.61	.000	39.77	.69	38.32	41.22
x2.6	Equal variances assumed	.99	.334	5.30	18.00	.000	20.65	3.89	12.47	28.82
	Equal variances not assumed			5.30	14.04	.000	20.65	3.89	12.30	28.99
x2.7	Equal variances assumed	2.45	.135	5.92	18.00	.000	21.34	3.61	13.76	28.92
	Equal variances not assumed			5.92	14.61	.000	21.34	3.61	13.63	29.05
x2.8	Equal variances assumed	6.19	.023	6.68	18.00	.000	23.20	3.47	15.90	30.49
	Equal variances not assumed			6.68	12.51	.000	23.20	3.47	15.66	30.73

Dari tabel 10 diatas, dapat dilihat bahwa nilai terlihat bahwa t hitung > df, maka dapat kita simpulkan bahwa tidak terdapat perbedaan antara mininet dan Prototipe 2A. Dengan kata lain pada uji Throuput UDP, tidak terdapat perbedaan yang signifikan antara mininet dengan prototipe 2A.

Tabel 11. Simple Independent t Test TCP proto 2A

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	10.60	.004	19.03	18.00	.000	12.89	.68	11.46	14.31
	Equal variances not assumed			19.03	11.04	.000	12.89	.68	11.40	14.38
x2.2	Equal variances assumed	1.37	.257	29.89	18.00	.000	10.59	.35	9.85	11.34
	Equal variances not assumed			29.89	16.61	.000	10.59	.35	9.84	11.34
x2.3	Equal variances assumed	.53	.477	25.26	18.00	.000	9.85	.39	9.03	10.67
	Equal variances not assumed			25.26	17.03	.000	9.85	.39	9.03	10.67
x2.4	Equal variances assumed	1.86	.189	26.74	18.00	.000	8.67	.32	7.99	9.35
	Equal variances not assumed			26.74	17.45	.000	8.67	.32	7.99	9.36
x2.5	Equal variances assumed	1.32	.266	30.22	18.00	.000	10.00	.33	9.30	10.69
	Equal variances not assumed			30.22	15.88	.000	10.00	.33	9.30	10.70
x2.6	Equal variances assumed	.08	.786	39.81	18.00	.000	11.64	.29	11.02	12.25
	Equal variances not assumed			39.81	17.62	.000	11.64	.29	11.02	12.25
x2.7	Equal variances assumed	.70	.413	27.63	18.00	.000	10.69	.39	9.88	11.51
	Equal variances not assumed			27.63	17.32	.000	10.69	.39	9.88	11.51
x2.8	Equal variances assumed	.02	.891	26.66	18.00	.000	10.10	.38	9.30	10.90
	Equal variances not assumed			26.66	17.18	.000	10.10	.38	9.30	10.90

Dari Tabel 11 diatas, berdasarkan hasil perhitungan tersebut terlihat bahwa t hitung > df, maka dapat kita simpulkan bahwa tidak terdapat perbedaan antara mininet dan Prototipe 2A, Dengan kata lain pada uji Throuput TCP, tidak terdapat perbedaan yang signifikan antara mininet dengan prototipe 2A.

Tabel 12. Simple Independent t Test Latency proto 2A

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	3.59	.074	-9.41	18.00	.000	-1.50	.16	-1.84	-1.17
	Equal variances not assumed			-9.41	11.46	.000	-1.50	.16	-1.85	-1.15
x2.2	Equal variances assumed	13.81	.002	-11.82	18.00	.000	-2.73	.23	-3.21	-2.24
	Equal variances not assumed			-11.82	12.02	.000	-2.73	.23	-3.23	-2.22
x2.3	Equal variances assumed	.00	.976	-8.04	18.00	.000	-1.53	.19	-1.93	-1.13
	Equal variances not assumed			-8.04	17.07	.000	-1.53	.19	-1.93	-1.13
x2.4	Equal variances assumed	1.17	.293	-7.38	18.00	.000	-2.06	.28	-2.64	-1.47
	Equal variances not assumed			-7.38	15.09	.000	-2.06	.28	-2.65	-1.46
x2.5	Equal variances assumed	.44	.517	-18.21	18.00	.000	-2.34	.13	-2.61	-2.07
	Equal variances not assumed			-18.21	14.97	.000	-2.34	.13	-2.62	-2.07
x2.6	Equal variances assumed	2.71	.117	-20.70	18.00	.000	-2.49	.12	-2.74	-2.23
	Equal variances not assumed			-20.70	15.69	.000	-2.49	.12	-2.74	-2.23
x2.7	Equal variances assumed	.94	.345	-19.25	18.00	.000	-3.20	.17	-3.55	-2.85
	Equal variances not assumed			-19.25	15.23	.000	-3.20	.17	-3.55	-2.85
x2.8	Equal variances assumed	6.66	.019	-14.16	18.00	.000	-2.18	.15	-2.51	-1.86
	Equal variances not assumed			-14.16	14.38	.000	-2.18	.15	-2.51	-1.85

Dari Tabel 12 diatas, berdasarkan hasil perhitungan tersebut terlihat bahwa t hitung < df, maka dapat kita simpulkan bahwa terdapat perbedaan antara mininet dan Prototipe 2A, Dengan kata lain pada uji Latency, tidak terdapat perbedaan yang signifikan antara mininet dengan prototipe 2A.

Untuk prototipe 2B, dilakukan uji simple independent t test terhadap mininet dan diperoleh hasil seperti dapat dilihat pada tabel 13 14 dan 15 dibawah ini.

Tabel 13. Simple Independent t Test Througput UDP Proto 2B

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	.29	.598	-3.17	18.00	.005	-1.21	.38	-2.01	-.41
	Equal variances not assumed			-3.17	18.00	.005	-1.21	.38	-2.01	-.41
x2.2	Equal variances assumed	1.73	.205	-3.59	18.00	.002	-1.64	.46	-2.60	-.68
	Equal variances not assumed			-3.59	13.82	.003	-1.64	.46	-2.62	-.66
x2.3	Equal variances assumed	.00	.959	-2.58	18.00	.019	-1.70	.66	-3.09	-.32
	Equal variances not assumed			-2.58	17.97	.019	-1.70	.66	-3.09	-.32
x2.4	Equal variances assumed	2.10	.164	.26	18.00	.795	.27	1.01	-1.85	2.38
	Equal variances not assumed			.26	12.64	.796	.27	1.01	-1.92	2.45
x2.5	Equal variances assumed	.00	.948	55.39	18.00	.000	40.01	.72	38.49	41.53
	Equal variances not assumed			55.39	17.95	.000	40.01	.72	38.49	41.53
x2.6	Equal variances assumed	1.14	.299	5.29	18.00	.000	20.39	3.86	12.29	28.50
	Equal variances not assumed			5.29	13.71	.000	20.39	3.86	12.10	28.68
x2.7	Equal variances assumed	2.35	.143	5.75	18.00	.000	20.79	3.62	13.19	28.38
	Equal variances not assumed			5.75	14.68	.000	20.79	3.62	13.07	28.51
x2.8	Equal variances assumed	3.24	.088	6.08	18.00	.000	22.36	3.68	14.63	30.09
	Equal variances not assumed			6.08	14.61	.000	22.36	3.68	14.50	30.22

Dari tabel 13, dapat dilihat bahwa, t hitung < df, maka dapat kita simpulkan bahwa terdapat perbedaan antara mininet dan Prototipe 2B, Dengan kata lain pada uji throughput UDP proto 2B, terdapat perbedaan yang signifikan antara mininet dengan prototipe 2B.

Tabel 14. Simple Independent t Test Througput TCP Proto 2B

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	9.86	.006	11.97	18.00	.000	9.92	.83	8.18	11.66
	Equal variances not assumed			11.97	10.32	.000	9.92	.83	8.08	11.76
x2.2	Equal variances assumed	1.90	.184	24.26	18.00	.000	11.46	.47	10.47	12.45
	Equal variances not assumed			24.26	16.75	.000	11.46	.47	10.46	12.46
x2.3	Equal variances assumed	.53	.477	25.26	18.00	.000	9.85	.39	9.03	10.67
	Equal variances not assumed			25.26	17.03	.000	9.85	.39	9.03	10.67
x2.4	Equal variances assumed	5.44	.032	30.83	18.00	.000	9.19	.30	8.57	9.82
	Equal variances not assumed			30.83	15.58	.000	9.19	.30	8.56	9.83
x2.5	Equal variances assumed	1.32	.266	30.22	18.00	.000	10.00	.33	9.30	10.69
	Equal variances not assumed			30.22	15.88	.000	10.00	.33	9.30	10.70
x2.6	Equal variances assumed	.08	.786	39.81	18.00	.000	11.64	.29	11.02	12.25
	Equal variances not assumed			39.81	17.62	.000	11.64	.29	11.02	12.25
x2.7	Equal variances assumed	.70	.413	27.63	18.00	.000	10.69	.39	9.88	11.51
	Equal variances not assumed			27.63	17.32	.000	10.69	.39	9.88	11.51
x2.8	Equal variances assumed	1.00	.331	28.22	18.00	.000	10.91	.39	10.10	11.72
	Equal variances not assumed			28.22	16.94	.000	10.91	.39	10.10	11.73

Berdasarkan hasil perhitungan yang terlihat pada tabel 14, terlihat bahwa t hitung > df, maka dapat kita simpulkan bahwa tidak terdapat perbedaan antara mininet dan Prototipe 2B. Dengan kata lain pada uji throughput TCP proto 2B, terdapat tidak perbedaan yang signifikan antara mininet dengan prototipe 2B.

Tabel 15. Simple Independent t Test Latency Proto 2B

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
X2.1	Equal variances assumed	3.59	.074	-9.41	18.00	.000	-1.50	.16	-1.84	-1.17
	Equal variances not assumed			-9.41	11.46	.000	-1.50	.16	-1.85	-1.15
x2.2	Equal variances assumed	13.81	.002	-11.82	18.00	.000	-2.73	.23	-3.21	-2.24
	Equal variances not assumed			-11.82	12.02	.000	-2.73	.23	-3.23	-2.22
x2.3	Equal variances assumed	.00	.976	-8.04	18.00	.000	-1.53	.19	-1.93	-1.13
	Equal variances not assumed			-8.04	17.07	.000	-1.53	.19	-1.93	-1.13
x2.4	Equal variances assumed	.00	.984	-8.71	18.00	.000	-1.94	.22	-2.41	-1.47
	Equal variances not assumed			-8.71	17.74	.000	-1.94	.22	-2.41	-1.47
x2.5	Equal variances assumed	18.93	.000	-12.40	18.00	.000	-2.75	.22	-3.21	-2.28
	Equal variances not assumed			-12.40	10.82	.000	-2.75	.22	-3.24	-2.26
x2.6	Equal variances assumed	2.71	.117	-20.70	18.00	.000	-2.49	.12	-2.74	-2.23
	Equal variances not assumed			-20.70	15.69	.000	-2.49	.12	-2.74	-2.23
x2.7	Equal variances assumed	14.81	.001	-6.27	18.00	.000	-2.30	.37	-3.08	-1.53
	Equal variances not assumed			-6.27	11.99	.000	-2.30	.37	-3.11	-1.50
x2.8	Equal variances assumed	74.49	.000	.34	18.00	.741	.26	.79	-1.39	1.92
	Equal variances not assumed			.34	9.17	.744	.26	.79	-1.51	2.04

Berdasarkan hasil perhitungan yang terlihat pada tabel 15, bahwa t hitung < df, maka dapat kita simpulkan bahwa terdapat perbedaan antara mininet dan Prototipe 2B. Dengan kata lain pada uji throughput TCP proto 2B, terdapat perbedaan yang signifikan antara mininet dengan prototipe 2B.

5. KESIMPULAN

Dari data dan hasil uji simple independent t test yang dilakukan, dapat diambil kesimpulan seperti terlihat pada tabel 16 dibawah ini;

Tabel 16. Simpulan Hasil Percobaan

	Throughput UDP	**Throughput TCP**	**Latency**
Prototipe 2A	Tidak ada perbedaan yang signifikan	Tidak ada perbedaan yang signifikan	Ada perbedaan yang signifikan
Prototipe 2B	Ada perbedaan yang signifikan	Tidak ada perbedaan yang signifikan	Ada perbedaan yang signifikan

Dari tabel diatas, dapat dilihat bahwa, prototipe 2A, lebih mendekati hasil mininet, walaupun dari kedua prototipe memberikan perbedaan yang signifikan dalam performa latency. Prototipe switch OpenFlow berbasih MikroTik, memberikan hasil yang baik pada throughput TCP, baik pada model 2A dan Model 2B.

Saran

Karena MikroTik RB750 sekarang dalam kondisi *discontinue* (tidak ada produksi baru) maka disarankan untuk menggunakan MikroTik dengan seri yang lain. MikroTik RB750 penggantian firmware-nya sulit dilakukan bila langkah yang dikerjakan sedikit ada yang keliru maka flashing langsung gagal, dan kelemahan dari perangkat ini adalah memori block yang mudah sekali rusak atau tidak terhapus dengan baik.

DAFTAR PUSTAKA

C. Decusatis, A. Carranza, dan J. Delgado-caceres, "Modeling Software Defined Networks using Mininet," *Proc. 2nd Int. Conf. Comput. Inf. Sci. Technol. Ottawa, Canada* –, no. 133, hal. 1–6, 2016.

E. C. Yik, "Implementation of an Open Flow Switch on Netfpga," Universiti Teknologi Malaysia, 2012.

G. R. D. T. Muntaner, "Evaluation of OpenFlow Controllers," 2012.

K. Kaur, J. Singh, dan N. S. Ghumman, "Mininet as Software Defined Networking Testing Platform," *Int. Conf. Commun. Comput. Syst.*, hal. 3–6, 2014.

M. Appelman dan M. De Boer, "Performance Analysis of OpenFlow Hardware,"University of Amsterdam, 2012.

P. A. Morreale dan James M. Anderson, *Software Defined Networking*, 1 ed. Boca Raton, FL: CRC Press, 2015.

R. Kartadie, F. Rozi, dan E. Utami, "OPENFLOW SWITCH SOFTWARE-BASED PERFORMANCE TEST ON ITS IMPLEMENTATION ON CAMPUS NETWORK", Journal of Theoretical and Applied Information Technology, Vol 96, no. 13, hal. 4136, 2018.

R. Kartadie dan B. Satya, "Uji Performa Implementasi Software-Based OpenFlow Switch Berbasis OpenWRT Pada Infrastruktur Software-Defined Network," *DASI*, vol. 16, no. 3, hal. 87, 2015.