JTERA (Jurnal Teknologi Rekayasa), Vol. 6, No. 1, Juni 2021, Hal. 93-100 p-ISSN 2548-737X
Terakreditasi "**Peringkat 3"** oleh Kemenristek/BRIN, Nomor SK: 85/M/KPT/2020 e-ISSN 2548-8678
DOI: 10.31544/jtera.v6.i1.2021.93-100

Analisis Performansi *Data Plane Development Kit* Terhadap *Open Virtual Switch* pada Jaringan Virtual

Nanda Iryani[#], Afifah Dwi Ramadhani, Queenta Paradissa Ramadhani
Program Studi Teknik Telekomunikasi, Fakultas Teknik Telekomunikasi dan Elektro,
Institut Teknologi Telkom Purwokerto
Jl. D. I. Panjaitan No. 128, Purwokerto 53147, Indonesia
[#]nanda@ittelkom-pwt.ac.id

Abstrak

Virtualisasi merupakan teknologi yang digunakan untuk merubah sesuatu yang bersifat fisik ke dalam bentuk virtual. Jaringan virtual dalam virtualisasi membutuhkan *switch* virtual untuk menghubungkan sejumlah virtual mesin. *Switch* virtual yang umum digunakan adalah Open vSwitch. Paket yang melewati Open vSwitch diteruskan dengan cara mencocokkan *flow entries* yang ada di *flow table*. Hal ini membuat pemrosesan paket relatif lebih lambat karena melewati banyak *space*. *Data Plane Development Kit* (DPDK) hadir dalam bentuk *extended packet* yang menawarkan kinerja pemrosesan paket secara lebih efisien. Penelitian ini melakukan penerapan DPDK pada Open vSwitch sekaligus menganalisis performansinya terhadap parameter *Quality of Service* (QoS) yang diterapkan pada jaringan virtual. Metode yang digunakan yaitu melalui peningkatan *background traffic* sebesar 100 Mb, 150 Mb, dan 200 Mb dengan mengirimkan *traffic* protokol *Transmission Control Protocol* (TCP) sebanyak 30 kali pengujian selama 20 detik. Parameter QoS yang diukur yaitu *delay*, *jitter*, *throughput*, dan *packet loss*. Penelitian ini membuktikan bahwa penerapan DPDK pada Open vSwitch berhasil meningkatkan performansi QoS pada jaringan sebesar 21%. Nilai *delay*, *jitter*, dan *packet loss* menurun, sedangkan *throughput* meningkat seiring bertambahnya ukuran data. Hasil pengukuran parameter QoS secara keseluruhan termasuk dalam kategori sangat baik berdasarkan standar rekomendasi menurut ITU-T G.1010.

Kata kunci: Open vSwitch, DPDK, QoS

Abstract

Virtualization is a technology that is used to change something that is physical into a virtual form. Virtual network in virtualization requires a virtual switch to connect a number of virtual machines. A commonly used virtual switch is Open vSwitch. Packets passing through Open vSwitch are forwarded by matching the flow entries in the flow table. This makes packet processing relatively slower because it passes a lot of space. The Data Plane Development Kit (DPDK) comes in the form of an extended packet that offers more efficient packet processing performance. This study applies DPDK on Open vSwitch as well as analyzes its performance on the Quality of Service (QoS) parameters applied to virtual networks. The method used is by increasing background traffic by 100 Mb, 150 Mb, and 200 Mb by sending Transmission Control Protocol (TCP) traffic protocol 30 times for 20 seconds. The measured QoS parameters are delay, jitter, throughput, and packet loss. This research proves that the implementation of DPDK on Open vSwitch has succeeded in increasing QoS performance on the network by 21%. The value of delay, jitter, and packet loss decreases, while the throughput increases as the data size increases. The overall QoS parameter measurement results are included in the very good category based on the recommended standard according to ITU-T G.1010.

Keywords: *Open vSwitch, DPDK, QoS*

I. PENDAHULUAN

Seiring perkembangan zaman, teknologi pun semakin berkembang terutama pada bidang industri telekomunikasi. Dalam perkembangannya, muncul sistem teknologi yang membuat perangkat *hardware* diubah menjadi perangkat *software* namun tetap memiliki fungsi yang sama. Virtualisasi adalah teknologi yang digunakan untuk merubah sesuatu yang bersifat fisik ke versi virtual,

Diterima: 5 Februari 2021; Direvisi: 3 Mei 2021; Disetujui: 18 Mei 2021

contohnya sumber daya jaringan, penyimpanan data, dan sistem operasi [1]. Jaringan virtual dalam virtualisasi membutuhkan perangkat *switch* untuk menghubungkan sejumlah virtual mesin. Hal ini agar dapat melakukan pertukaran paket serta meneruskan data ke berbagai perangkat tujuan. *Switch* virtual yang umum digunakan adalah Open vSwitch [2].

Open Virtual Switch (OVS) merupakan *switch* virtual multilayer yang dirancang untuk dapat menghubungkan koneksi jaringan baik antar virtual mesin maupun dengan jaringan luar [3]. *Networking* pada Open vSwitch mengikuti dengan *networking* sistem operasi yang digunakan. Hal ini membuat pemrosesan paket menjadi lebih lama karena melewati banyak *space*. Sehingga menghasilkan nilai parameter *Quality of Service* (QoS) yang tidak terlalu bagus. *Intel* mengeluarkan kerangka kerja bernama *Data Plane Development Kit* (DPDK) yang menawarkan implementasi pemrosesan paket lebih efisien [4].

Sejumlah penelitian sebelumnya terkait OVS dan DPDK telah dilakukan. Penelitian [5] melakukan perbandingan performansi OVS-DPDK pada saat *port* dan *flow mirroring enable* atau *disable*. Pengukuran berstandar pada *throughput* dan *latency*. Hasil penelitian menunjukkan bahwa OVS-DPDK dapat mempertahankan *low latency* ketika *port* dan *flow mirroring* diaktifkan. Penelitian lainnya, [6] menganalisis performansi OVS-DPDK berdasarkan konfigurasi *multi-core pararellization*. Parameter yang diuji yaitu *throughput* dan *latency*. Berdasarkan penerapannya, pengalokasian CPU *core* selaras dengan kinerja yang dihasilkan oleh OVS-DPDK. Terakhir, peneliti [7] melakukan analisis DPDK pada OVS melalui pengalokasian *hugepage*. Hasil yang diperoleh yaitu penggunaan CPU *load* semakin tinggi ketika *hugepage* dialokasikan semakin besar.

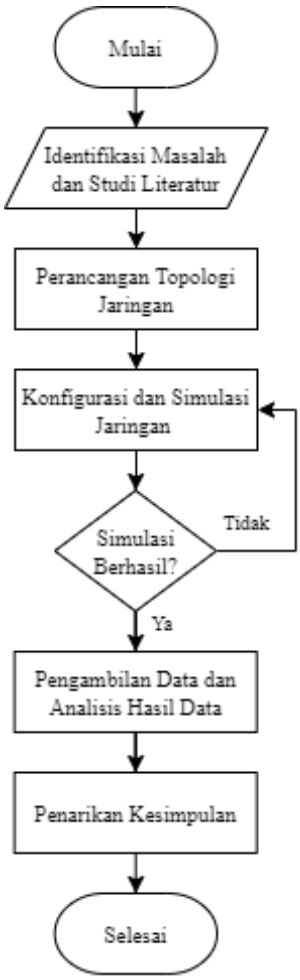
Berdasarkan penelitian terdahulu, masih sedikit pembahasan mengenai DPDK dari segi parameter QoS. Sehingga perlu untuk melakukan penelitian mengenai kinerja DPDK terhadap pengukuran parameter QoS. QoS merupakan metode pengukuran yang digunakan untuk menentukan kapabilitas jaringan dengan teknik mengelola *bandwidth*, *jitter*, *delay*, dan *packet loss* [8]. Pentingnya QoS dalam sebuah jaringan dapat menjadikan penyampaian berbagai jenis data mencapai kehandalan dan kecepatan yang tinggi di dalam suatu komunikasi. Ini merujuk pada kualitas kebutuhan layanan dalam suatu jaringan [9]. Menggunakan layanan QoS juga memaksimalkan penggunaan investasi jaringan yang sudah ada dengan memberikan prioritas pada aplikasi-aplikasi yang beroperasi [10].

Penelitian ini bertujuan menerapkan DPDK pada Open vSwitch, sekaligus menganalisis performansinya dalam hal pemrosesan paket. Pengukuran performansi berdasarkan parameter QoS (*delay*, *jitter*, *throughput*, dan *packet loss*) yang berstandar pada ITU-T G.1010. Skenario pengujian berupa penambahan *background traffic* sebesar 100 Mb, 150 Mb, dan 200 Mb yang digunakan sebagai bahan analisa penelitian. Tujuan lain penelitian ini adalah memperoleh rancangan jaringan OVS-DPDK dan mengetahui kinerjanya guna mengoptimalkan kualitas jaringan.

II. METODE PENELITIAN

A. Perancangan Penelitian

Simulasi pada penelitian ini dilakukan menggunakan *software* GNS3 yang dipasang pada sistem operasi Linux Ubuntu 20.04. *Background traffic* ditingkatkan melalui *software* Iperf dengan Wireshark digunakan untuk melakukan *capture* paket data yang melewati jaringan. Open vSwicth digunakan sebagai *switch* virtual pada konfigurasi jaringan dengan menambahkan DPDK. Alur penelitian ditunjukkan pada Gambar 1.



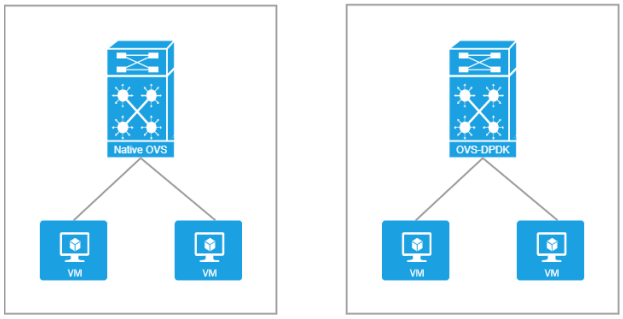
Gambar 1. *Flowchart* penelitian

Pada penerapannya, OVS-DPDK dikonfigurasi untuk pengalokasian memori *hugepage*, penentuan jumlah *core* CPU, dan juga penggunaan memori RAM. Paket yang digunakan dalam pengujian berupa paket TCP (*Transfer Control Protocol*). Jaringan yang dilewatkan paket TCP diberikan peningkatan beban trafik sebesar 100 Mb, 150 Mb, dan 200 Mb. Pengujian dilakukan dengan mengukur nilai parameter QoS serta menganalisanya menggunakan Wireshark sebanyak 30 kali pengujian selama 20 detik.

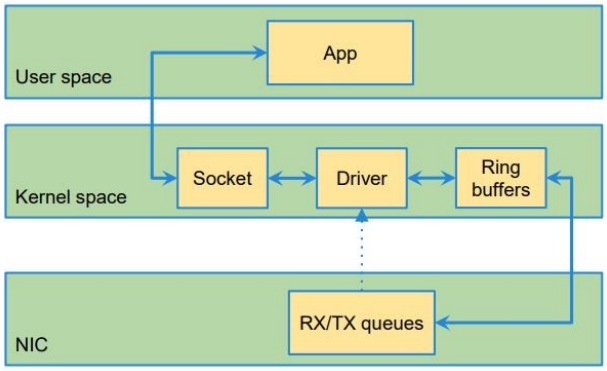
Gambar 2 menunjukkan topologi jaringan yang digunakan dalam penelitian ini. Simulasi dijalankan menggunakan Open vSwitch tanpa DPDK dan Open vSwitch dengan tambahan DPDK. Open vSwitch di-*install* pada GNS3 dengan virtualisasi menggunakan sistem operasi Ubuntu 18.04 untuk menghubungkan *client* dan *server*. Dalam topologi ini menggunakan dua virtual mesin yang berperan sebagai *client* dan *server*. Virtual mesin 1 difungsikan sebagai *server* yang melakukan pengiriman data, sedangkan virtual mesin 2 sebagai *client* yang menerima paket dari *server* melalui OVS.

Gambar 3 merupakan konsep *networking default* yang ada di Linux Ubuntu. Aplikasi yang berada di *user space* ketika akan melakukan prosesnya, hal pertama yang dilalui yaitu melewati *kernel space*. Di dalam *kernel space* terdapat *socket*, *driver* dari NIC, dan juga *ring buffer*. Kemudian paket dikirim ke *Network Interface Card* (NIC) RX/TX. Pemrosesan paket ini melalui tahapan yang cukup panjang berawal dari *user space*, kemudian ke *kernel space*, hingga akhirnya ke NIC.

Gambar 4 menunjukkan pemrosesan paket dengan tambahan *tools* DPDK. Pada proses ini, aplikasi-aplikasi yang berada di *user space* tidak akan melewati *kernel space* terlebih dahulu melainkan langsung menuju ke NIC. Hal ini terjadi karena DPDK memiliki PMD *Drivers* yang berfungsi sebagai tempat terjadinya pemrosesan paket. Di dalam *kernel space* hanya berisikan UIO *driver*. Dimana UIO *driver* ini merupakan *driver* yang *support* untuk NIC DPDK. Tahapan pemrosesan paket dengan tambahan DPDK lebih singkat daripada pemrosesan paket yang tidak menggunakan tambahan DPDK. Hal ini membuat pemrosesan paket berjalan lebih cepat dan efektif.



Gambar 2. Rancangan topologi jaringan

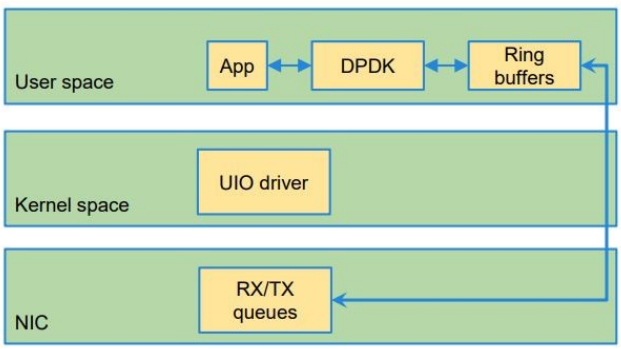


Gambar 3. Alur pemrosesan paket tanpa DPDK [11]

B. Quality of Service (QoS)

QoS merupakan metode pengukuran yang digunakan untuk mengetahui seberapa baik kualitas yang dihasilkan pada sebuah jaringan. Selain itu, QoS dapat mendefinisikan karatketristik dan sifat dari suatu *service* [12]. Parameter QoS didapatkan dari perbandingan antara unit data masukan dengan data keluaran pada sisi *client*. Tujuannya adalah untuk memberikan sebuah pelayanan jaringan dengan kualitas yang lebih baik [13]. Setiap layanan pada suatu jaringan memiliki standar nilai kinerja dengan kategori yang berbeda-beda. Berdasarkan kategorinya, parameter-parameter QoS yang diukur bersumber pada standar rekomendasi menurut ITU-T G.1010. Parameter-parameter yang diukur dalam QoS yaitu *delay*, *jitter*, *throughput*, dan *packet loss*.

1) Delay: merupakan waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. *Delay* dapat dipengaruhi oleh jarak, media fisik, dan *congestion*. Nilai *delay* dapat mempengaruhi parameter QoS yang lainnya. *Delay* dihitung dalam satuan *milisecond* (ms) sesuai dengan persamaan (1) dan (2), serta standar pada Tabel 1.



Gambar 4. Alur pemrosesan paket dengan DPDK [11]

$$Delay\ (ms) = Time\ 2 - Time\ 1 \quad \text{b} \quad (1)$$

$$Rerata\ delay = \frac{Total\ delay}{Total\ paket\ yang\ diterima} \quad (2)$$

2) Jitter: merupakan nilai perubahan dalam *delay* penundaan paket atau perubahan waktu kedatangan. *Delay* antrian pada *router* dan *switch* dapat menyebabkan *jitter*. Nilai *jitter* dapat menyebabkan hilangnya data terutama dalam transmisi data berkecepatan tinggi [15]. Nilai *jitter* dapat dihitung berdasarkan persamaan (3) dan (4) dengan standar nilai pada Tabel 2.

$$Variasi\ delay = Delay\ 2 - Delay\ 1 \quad (3)$$

$$Jitter\ (ms) = \frac{Total\ variasi\ delay}{Total\ paket\ diterima-2} \quad (4)$$

3) Packet loss: merupakan parameter yang menunjukkan suatu kondisi dimana jumlah paket hilang (*loss*). *Packet loss* dapat terjadi karena *collision* dan *congestion* pada jaringan. Nilai *packet loss* dihitung berdasarkan perbandingan paket yang tidak sukses dikirim dari semua paket yang dikirim [16]. Nilai *packet loss* dapat dihitung berdasarkan persamaan (5) dengan standar nilai pada Tabel 3.

$$Packet\ loss = \frac{Paket\ dikirim - Paket\ diterima}{Paket\ data\ dikirim} x100\% \quad (5)$$

Tabel 1. Standar *delay* ITU-T G.1010 [14]

Kategori	Nilai *delay*
Sangat bagus	< 150 ms
Bagus	150 – 300 ms
Sedang	300 – 450 ms
Buruk	> 450 ms

Tabel 2. Standar *jitter* ITU-T G.1010 [15]

Kategori	Nilai *jitter*
Sangat bagus	0 ms
Bagus	0 – 75 ms
Sedang	76 – 125 ms
Buruk	125 – 225 ms

Tabel 3. Standar *packet loss* ITU-T G.1010 [16]

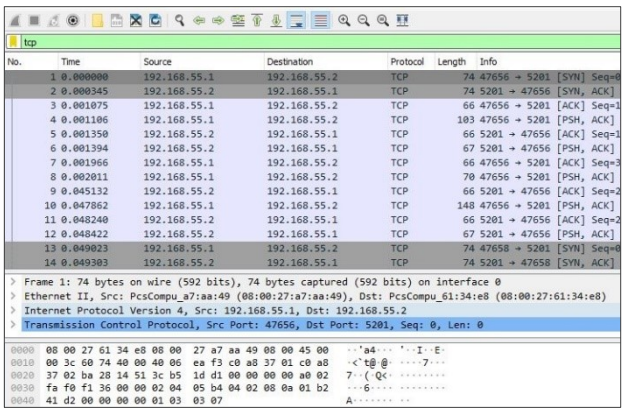
Kategori	Nilai *packet loss*
Sangat bagus	0 %
Bagus	3%
Sedang	15%
Buruk	25%

4) Throughput: merupakan jumlah total kedatangan paket yang sukses dari satu terminal tertentu dalam sebuah jaringan. Nilai *throughput* diamati pada *destination* selama interval waktu tertentu dibagi dengan durasi interval waktu tersebut. *Throughput* dapan disebut pula kecepatan (*rate*) transfer data [17]. Nilai *throughput* dapat dihitung berdasarkan persamaan (6).

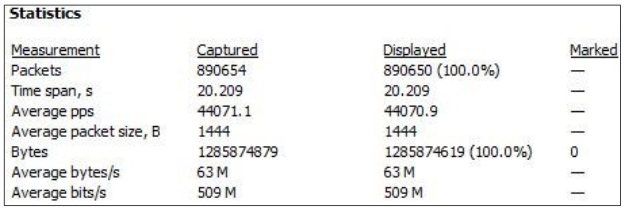
$$Throughput = \frac{Paket\ data\ diterima\ (bytes)}{Waktu\ pengiriman\ paket\ (sec)} \quad (6)$$

C. Pengambilan Data

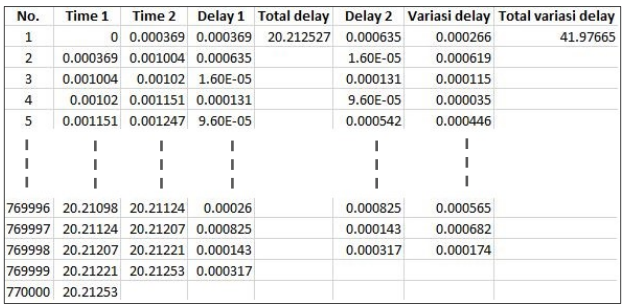
Pada penelitian ini, Wireshark digunakan untuk melakukan penangkapan paket data yang melewati jaringan (Gambar 5). Data pengukuran yang dihasilkan selanjutnya dianalisis berdasarkan parameter QoS seperti pada Gambar 6 dan Gambar 7. Pada penelitian ini juga dilakukan pengambilan data tanpa *background traffic*. Hal ini digunakan sebagai tolok ukur perbandingan kinerja dari Open vSwitch DPDK.



Gambar 5. Pengambilan data pada Wireshark



Gambar 6. Akumulasi data QoS



No.	Time 1	Time 2	Delay 1	Total delay	Delay 2	Variasi delay	Total variasi delay
1	0	0.000369	0.000369	20.212527	0.000635	0.000266	41.97665
2	0.000369	0.001004	0.000635		1.60E-05	0.000619	
3	0.001004	0.00102	1.60E-05		0.000131	0.000115	
4	0.00102	0.001151	0.000131		9.60E-05	0.000035	
5	0.001151	0.001247	9.60E-05		0.000542	0.000446	
769996	20.21098	20.21124	0.00026		0.000825	0.000565	
769997	20.21124	20.21207	0.000825		0.000143	0.000682	
769998	20.21207	20.21221	0.000143		0.000317	0.000174	
769999	20.21221	20.21253	0.000317				
770000	20.21253						

Gambar 7. Perhitungan data QoS

III. HASIL DAN PEMBAHASAN

Penelitian-penelitian sebelumnya membahas mengenai Open vSwitch DPDK berdasarkan pada konfigurasi *multi-core*, pengalokasian *hugepage*, dan *port mirroring*. Penelitian ini membahas dari segi performansi jaringan berdasarkan parameter-parameter QoS pada Open vSwitch DPDK. Pada penelitian ini juga dilakukan pengambilan data tanpa *background traffic*.

Pengujian yang dilakukan pada penelitian ini yaitu dengan melakukan perbandingan pengujian performansi QoS pada OVS yang ditambahkan dengan DPDK dan pada *native* OVS. Pengujian yang pertama yaitu pada sisi *server* yang mengirimkan paket TCP kepada *client* sebanyak 30 kali selama 20 detik tanpa meningkatkan *backgorund traffic*. Kemudian pada sisi *server* dilakukan penangkapan paket menggunakan Wireshark. Paket yang diterima oleh *client* sebelumnya telah diproses melalui Open vSwitch.

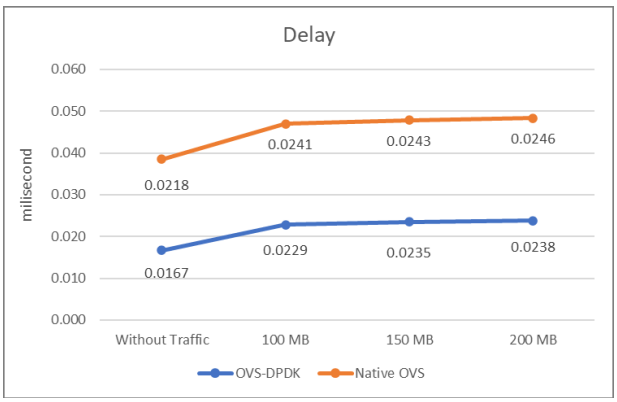
Pengujian yang seterusnya yaitu diberikan perlakuan dengan meningkatkan *background traffic* sebesar 100 Mb, 150 Mb, dan 200 Mb. Kemudian setelah itu diukur performansi dari OVS-DPDK dan *native* OVS. Pengujian yang dilakukan menghasilkan data sebanyak 240 *capture*. Data tersebut kemudian diolah dan dianalisis berdasarkan parameter QoS (*delay*, *jitter*, *throughput*, dan *packet loss*) menggunakan standarisasi menurut ITU-T G.1010.

A. Pengujian Delay

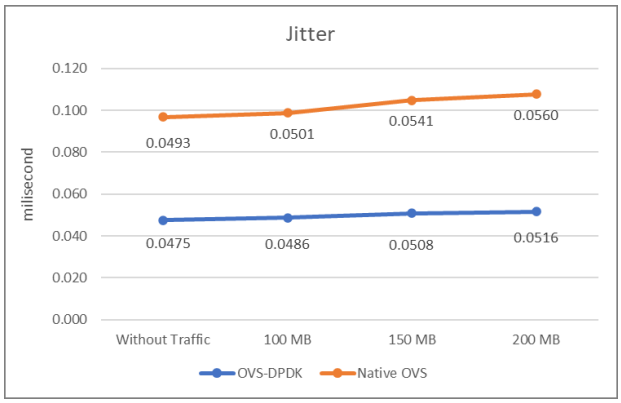
Gambar 8 menunjukkan perbandingan nilai *delay* yang dihasilkan berdasarkan pengujian topologi *Native* OVS dengan OVS-DPDK. Dalam grafik tersebut terlihat bahwa nilai *delay* semakin naik seiring dengan bertambahnya beban trafik yang diberikan. Nilai *delay* yang dihasilkan ketika tidak diberikan beban trafik pada *Native* OVS adalah sebesar 0,0167 ms. Sedangkan pada OVS-DPDK menghasilkan nilai sebesar 0,0218 ms.

Delay yang dimiliki oleh OVS-DPDK bernilai lebih kecil daripada *native* OVS pada semua kondisi. Hal ini terjadi karena pemrosesan paket pada OVS-DPDK tidak melewati *kernel space* terlebih dahulu melainkan langsung menuju ke *user space* melalui *Poll Mode Driver* (PMD). Sehingga, pemrosesan paket pada berjalan lebih cepat dan menghasilkan nilai *delay* yang kecil.

Perhitungan parameter *delay* secara keseluruhan memiliki nilai dibawah 150 ms baik pada *native* OVS maupun OVS-DPDK. Artinya, nilai parameter *delay* termasuk dalam kategori sangat baik sesuai dengan standarisasi ITU-T.



Gambar 8. Grafik *delay*



Gambar 9. Grafik parameter *jitter*

B. Jitter

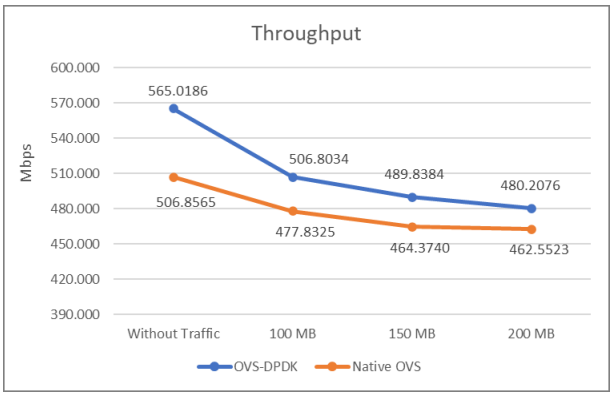
Gambar 9 menunjukkan perbandingan nilai *jitter* yang dihasilkan berdasarkan pengujian topologi *native* OVS dengan OVS-DPDK. Dalam grafik tersebut terlihat bahwa nilai *jitter* semakin naik seiring dengan bertambahnya *background traffic* yang diberikan. Nilai *jitter* yang dihasilkan ketika diberikan *background traffic* sebesar 100 Mb, pada *native* OVS bernilai 0,0521 ms, sedangkan pada OVS-DPDK menghasilkan nilai 0,0486 ms. Nilai ini tidak kurang dari pengujian yang dihasilkan tanpa menggunakan beban trafik.

Jitter yang dimiliki oleh OVS-DPDK memiliki nilai lebih kecil daripada *native* OVS pada semua kondisi. Hal ini terjadi karena OVS-DPDK mem-*bypass* semua proses yang ada di *kernel space*. Maka dari itu, pemrosesan paket pada OVS-DPDK lebih singkat dan menghasilkan nilai *variasi delay* yang lebih kecil. Hal ini mempengaruhi kepadatan antrian pada jaringan yang semakin berkurang sehingga menghasilkan nilai *jitter* yang mendekati 0 ms. Perhitungan parameter *jitter* secara keseluruhan memiliki nilai tidak lebih dari 0 ms baik pada *native* OVS maupun OVS-DPDK. Artinya, nilai parameter *jitter* termasuk dalam kategori sangat baik sesuai dengan klasifikasi berdasarkan standar ITU-T G.1010 dimana rentang nilai *jitter* dengan kategori sangat baik adalah 0 ms.

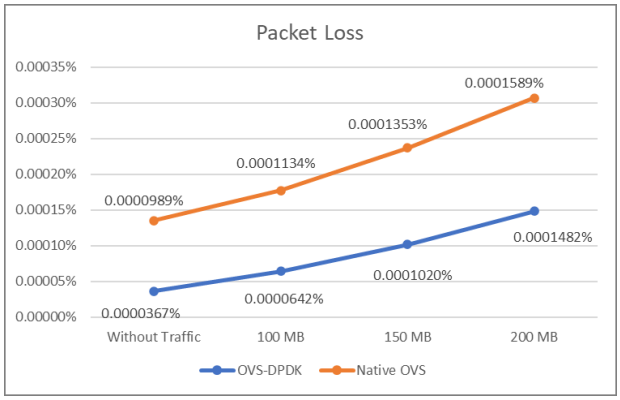
C. Throughput

Gambar 10 menunjukkan perbandingan nilai *throughput* yang dihasilkan berdasarkan pengujian topologi *native* OVS dengan OVS-DPDK. Dalam grafik tersebut terlihat bahwa nilai *throughput* semakin turun seiring dengan bertambahnya *background traffic* yang diberikan. Nilai *throughput* yang dihasilkan ketika tidak diberikan *background traffic*, pada *native* OVS bernilai 506,8565 Mbps. Sedangkan pada OVS-DPDK menghasilkan nilai 565,0186 Mbps. Namun ketika pengujian diberikan beban trafik sebesar 100 Mb, pada *native* OVS mengalami penurunan menjadi 477,8315 Mbps. Begitupula pada OVS-DPDK yang menurun menjadi 506,8034 Mbps.

Throughput yang dimiliki oleh OVS-DPDK memiliki nilai lebih besar daripada *native* OVS pada semua kondisi, baik pada saat diberikan peningkatan beban trafik maupun tidak. Hal ini terjadi karena OVS-DPDK membuat perjalanan paket dari NIC dapat langsung menuju *user space* melalui PMD tanpa perlu melewati *kernel space* terlebih dahulu. Sehingga membuat layer yang dilalui paket tersebut menjadi lebih sedikit dan pemrosesan paket berjalan lebih cepat.



Gambar 10. Grafik parameter *throughput*



Gambar 11. Grafik parameter *packet loss*

D. Packet Loss

Gambar 11 menunjukkan perbandingan nilai *packet loss* yang dihasilkan berdasarkan pengujian topologi *native* OVS dengan OVS-DPDK. Dalam grafik tersebut terlihat bahwa nilai *packet loss* semakin naik seiring dengan bertambahnya *background traffic* yang diberikan. Nilai *packet loss* yang dihasilkan ketika tidak diberikan *background traffic*, pada *native* OVS bernilai 0,0000989%. Sedangkan pada OVS-DPDK menghasilkan nilai 0,0000367%. Hal ini berarti jumlah paket yang hilang pada saat transmisi data sangat sedikit. Ketika pengujian diberikan beban trafik sebesar 100 Mb, nilai *packet loss* pada *native* OVS naik menjadi 0,0001134%. Begitupula pada OVS-DPDK menjadi 0,0000642%.

Packet loss yang dimiliki oleh OVS-DPDK memiliki nilai lebih kecil daripada *native* OVS pada semua kondisi, baik pada saat diberikan peningkatan beban trafik maupun tidak. Hal ini terjadi karena pemrosesan paket pada OVS-DPDK tidak melewati *kernel space* terlebih dahulu melainkan langsung menuju ke *user space* melalui PMD. Perhitungan parameter *packet loss* secara keseluruhan memiliki nilai tidak lebih dari 1% baik pada *native* OVS maupun OVS-DPDK. Artinya, nilai parameter *packet loss* yang dihasilkan termasuk dalam kategori sangat baik sesuai dengan standar ITU-T G.1010 dimana rentang nilai *jitter* dengan kategori sangat baik adalah 0%.

IV. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, disimpulkan bahwa penerapan DPDK pada Open vSwitch berhasil meningkatkan performansi jaringan sebesar 21%. Hal ini ditunjukkan dari hasil pengukuran parameter QoS yang termasuk dalam kategori sangat baik. Seiring dengan bertambahnya ukuran data yang dikirimkan, hasil pengukuran parameter *delay*, *jitter*, dan *packet loss* semakin meningkat, serta *throughput* semakin menurun. Walaupun demikian, nilai yang didapatkan masih sesuai dengan standar rekomendasi menurut ITU-T G.1010. Berdasarkan hasil penelitian yang telah dilakukan, penulis menyarankan untuk membahas lebih lanjut mengenai penerapan Open vSwitch DPDK secara nyata tidak sebatas simulasi. Hal ini perlu dilakukan agar DPDK dapat diimplementasikan pada Open vSwitch sesuai dengan kebutuhan.

REFERENSI

[1] R. Umar., "Review Tentang Virtualisasi," *Rev. Jurnal Informatika*, vol. 7, no. 2, pp. 775–784, 2013.

[2] S. Sunaryo, A. Tedyyana, and K. Kasmawi, "Rancang Bangun Server Cloud Computing Di Politeknik Negeri Bengkalis," *INOVTEK Polbeng - Seri Inform.*, vol. 2, no. 1, p. 33, 2017.

[3] M. Geo, U. Putra, and K. Utomo, "Perancangan Dan Analisis Performansi Open Vswitch Untuk Jaringan Virtual Universitas Telkom," *eProceedings of Engineering*, vol. 2, no. 2, pp. 2705–2712, 2015.

[4] I. Cerrato, M. Annarumma, and F. Risso, "Supporting fine-grained network functions through intel DPDK," *Proc. - 2014 3rd Eur. Work. Software-Defined Networks, EWSDN 2014*, pp. 1–6, 2014.

[5] S. Shanmugalingam, A. Ksentini, and P. Bertin, "DPDK Open vSwitch performance validation with mirroring feature," *2016 23rd Int. Conf. Telecommun. ICT 2016*, 2016.

[6] G. Sun, W. Li, and D. Wang, "Performance evaluation of DPDK open vswitch with parallelization feature on multi-core platform," *J. Commun.*, vol. 13, no. 11, pp. 685–690, 2018.

[7] D. Vladislavic, D. Huljenic, and J. Ozegovic, "Enhancing VNF's performance using DPDK driven OVS user-space forwarding," *2017 25th Int. Conf. Software, Telecommun. Comput. Networks, SoftCOM 2017*, 2017.

[8] R. Purnomo and P. R. Arisandi, "Analisis Qos Dengan Virtual Tenant Network Pada Software Define Networking," *Jurnal Teknologi Informatika dan Komputer*, vol. 5, no. 2, pp. 33-42, 2019.

[9] R. Rasudin, "Quality of Services (Qos) Pada Jaringan Internet Dengan Metode Hierarchy Token Bucket," *J. Penelit. Tek. Inform. Univ. Malikussaleh*, vol. 4, no. 1, pp. 210–223, 2014.

[10] W. P. Sasmita, N. Safriadi, and M. A. Irwansyah, "Analisis Quality of Service (QoS) pada Jaringan Internet (Studi Kasus: Fakultas Kedokteran Universitas Tanjungpura)," *J. Sist. dan Teknol. Inf.*, vol. 1, no. 1, pp. 37–43, 2013.

[11] L. Betzalel, "Additional DPDK Theory," in *Network Platforms Group Intel Corporation*, 2015, pp. 2–39.

[12] R. Wulandari, "ANALISIS QoS (QUALITY OF SERVICE) PADA JARINGAN INTERNET (STUDI KASUS : UPT LOKA UJI TEKNIK PENAMBANGAN JAMPANG KULON – LIPI)," *J. Tek. Inform. dan Sist. Inf.*, vol. 2, no. 2, pp. 162–172, 2016.

[13] ITU-T, "G.1010: End-user multimedia QoS categories," *Int. Telecommun. Union*, vol. 1010, 2001.

[14] N. Iryani, A. Dwi, and K. Masykuroh, "Analisa Performansi QoS Aplikasi Pembelajaran Daring pada Jam Kerja," *JTERA (Jurnal Teknol. Rekayasa)*, vol. 5, no. 2, p. 201, 2020.

[15] S. Astiti and N. Iryani, "Implementasi dan Analisis Performansi QoS pada Aplikasi English Competency Test," *JTERA (Jurnal Teknol. Rekayasa)*, vol. 5, no. 2, p. 267, 2020.

[16] F. J. Yaldi, "Keyword : Wimax , Video streaming , QoS , Packet sniffing," *Jom FTEKNIK Vol. 4 No. 1 Februari 2017 bandwidth*, vol. 4, no. 1, pp. 1–9, 2017.

[17] I. Iskandar and A. Hidayat, "Analisa Quality of Service (QoS) Jaringan Internet Kampus (Studi Kasus: UIN Suska Riau)," *J. CoreIT*, vol. 1, no. 2, pp. 67–76, 2015.