REPOSITOR, Vol. 2, No. 9, September 2020, Pp. 1267-1272
ISSN : 2714-7975
E-ISSN : 2716-1382

Analisa Performansi Trafik Jaringan Pada Hadoop Cluster Menggunakan Docker Berbasis Software Defined Network

Cindy Claudia Kusumastutik*[1], Saifuddin[2], Aminudin[3]
[1,2,3]Teknik Informatika/Universitas Muhammadiyah Malang
cla.cindyclaudia@gmail.com[*1], udin.saif@gmail.com[2], aminudin2008@umm.ac.id[3]

Abstrak

Pertumbuhan data yang pesat pada saat ini membuat semakin berkembangnya jumlah data yang ada. Hadoop sebagai salah satu processing Big Data telah mengambil peran yang penting dalam proses pemrosesan data. Akhir-akhir ini, telah banyak penelitian yang meneliti tentang performansi trafik data Big Data.Di dalam Hadoop Clustering, perpindahan data akan sering terjadi antara satu node dengan node yang lainnya. Hal ini dikarenakan karena data yang menyebar di dalam datanode saat melakukan proses penyimpanan di dalam HDFS. Ketersediaan bandwidth yang kecil dengan proses pengiriman data yang besar dapat menyebabkan congestion disaat proses pengiriman datanya. Software Defined Network merupakan paradigma baru di dalam dunia jaringan. Dengan memanfaatkan fungsi SDN yang bisa mengontrol manajemen rate sehingga dapat mengatur trafik jaringan yang lewat. Dengan arsitektur SDN pada cluster Hadoop dapat dilakukan manajemen transfer rate. Fitur manajemen transfer rate ini dilakukan dengan API di dalam Floodlight. Nilai pada transfer rate pada trafik aliran data HDFS dipisah dan diberi nilai transfer rate yang lebih tinggi. Uji coba yang dilakukan, didalam proses penyimpanan data ke HDFS tidak terpengaruh walaupun proses lalu lintas data terdapat proses lalu lintas data lain yang menyebab kan congestion.

Kata Kunci: *Software Defined Network, Hadoop Cluster, Docker, SDN*.

Abstract

Rapid data growth at this point make a growing amount of data exists. Hadoop as one large Data processing tlah took an important role in the process of data processing. Lately, a lot of research that examines tlah about performance data traffic in large Hadoop In Data Clustering, data transfer will often occur between one node with another node. This is due because the data are spread inside the datanode processing in the HDFS storage. Small bandwidth availability with large data delivery process can lead to bottlenecks when sending data. Software defined network of liquid performasi new world network. By utilizing the functions of SDN could control the level of management so that it can regulate network traffic passing by. With architecture SDN on Hadoop cluster management can do transfer speed. This transfer rate management features are done with fire in the spotlight. The value on the level of traffic flow on a data transfer HDFS separated and given the value of the transfer of a higher level. The trial was conducted, in the process of storage of data to HDFS not affected although there is data traffic process process other data traffic caused congestion right.

Keywords: *Software Defined Network, Hadoop Cluster, Docker, SDN*.

1. Pendahuluan

Perkembangan *Big Data* pada saat ini sudah mengalami perkembangan yang cukup pesat. Menurut riset yang dilakukan oleh *Oracle* pada tahun 2012, volume data tiap tahunnya tumbuh sebesar 40% dan diperkirakan pada tahun 2020 mencapai 45 *Zettabytes*. [1] Penyebab berkembangannya *Big Data* sendiri adalah banyak nya kebutuhan manusia akan kemudahan, efiesiensi serta kebutuhan akan kompleksitas data yang semakin lama semakin berkembang. Seiring perkembangan *Big Data* membuat munculnya berbagai sistem analisa terhadap *Big Data* seperti *Hadoop*, *Bigtop*, *Spark*, dan masih banyak lainnya. Selama proses komputasi di dalam sistem *Big Data* ini, perpindahan data sangat sering terjadi antara *computation nodes* di dalam data center[2][3].

Salah satu sistem preprocessing *Big Data* dan media storage adalah *Apache Hadoop*. *Hadoop. Apache Hadoop* sendiri adalah *Apache Open Source Project* yang menawarkan

distributed data processing melalui *computer cluster* [1][4]. Perbandingannya dalam *Apache Hadoop* sendiri adalah dalam *Apache Hadoop* menggunakan *single server* hingga dapat mencapai ribuan mesin. Didalam sistem *Hadoop* terdapat 3 komponen utama yaitu, *YARN, HDFS* dan *MapReduce* [5][6].

Didalam cluster *Hadoop,* akan sering mengalami perpindahan data, karena data yang disimpan akan dibagi kedalam *Datanode* terutama apabila melakukan pemrosesan penyimpanan data yang selanjutnya akan disimpan didalam *HDFS* [7]. Performansi kinerja cluster *Hadoop* juga dapat dipengaruhi oleh lalu lintas jaringan data secara keseluruhan. Permasalahan utama seperti, ketersediaan *bandwidth* dan juga *congestion* yang disebabkan lalu lintas data lain juga dapat mempengaruhi proses penyimpanan data. Data-data pada cluster *Hadoop* disimpan pada sistem file yang dikenal dengan sebutan *HDFS (Hadoop Distributed File System)* [8].

Dengan seiring berkembangnya konsep jaringan baru yakni *Software Defined Network (SDN)* yang merupakan konsep/paradigma baru dalam mendesain, mengelola dan mengimplementasikan jaringan, terutama untuk mendukung kebutuhan dan inovasi di bidang ini yg semakin lama semakin kompleks. Konsep dasar *SDN* adalah dengan melakukan pemisahan antara *control plane* dan *forwarding plane*. Dimana *control plane* diletakkan secara terpusat didalam *Controller*. Pada awalnya, *SDN* diidentikan dengan *Openflow* sehingga beberapa orang beranggapan bahwa *SDN* merupakan *Openflow*. Namun, hal ini tidak semuanya salah karena *Openflow* adalah elemen pada arsitektur *SDN* [9].

Dalam beberapa tahun terakhir, penelitian mencoba untuk optimasi analisa *Big Data* sistem dari segi sistem/jaringan. Terutama dengan adanya paradigma jaringan baru yaitu *Software Defined Network (SDN)* contohnya *Openflow* yang memungkinkan secara dinamis mengkonfigurasi jaringan sesuai dengan kebutuhan aplikasi dan status *run-time*. Doopnet support setting pada *Hadoop Cluster* atas *Mininet* dengan menggunakan *Docker Container* sebagai *Virtual Hosts*. *Mininet* adalah emulator jaringan yang bisa membangun jaringan skala besar dengan menggunakan single server. Hal ini mendukung *SDN* dan dapat bekerja dengan eksternal *Openflow Controller*. *Mininet* membuat *virtual hosts* menggunakan *Linux namespaces* [10].

Oleh karena itu, pada penelitian ini akan dilakukan analisa performasi *Hadoop Clustering* dengan menggunakan *Docker* sebagai *Containernya* dengan memanfatkan arsitektur jaringan *SDN*. *Software Defined Network* sendiri berfungsi sebagai control lalu lintas jaringan yang memanfatkan fitur *queue* pada *switch Openflow*. Dengan menggunakan *Software Defined Network* dapat digunakan sebagai sarana analisa trafik jaringan *Hadoop* dan *monitoring* jaringan untuk kebutuhan analisa kedepannya.

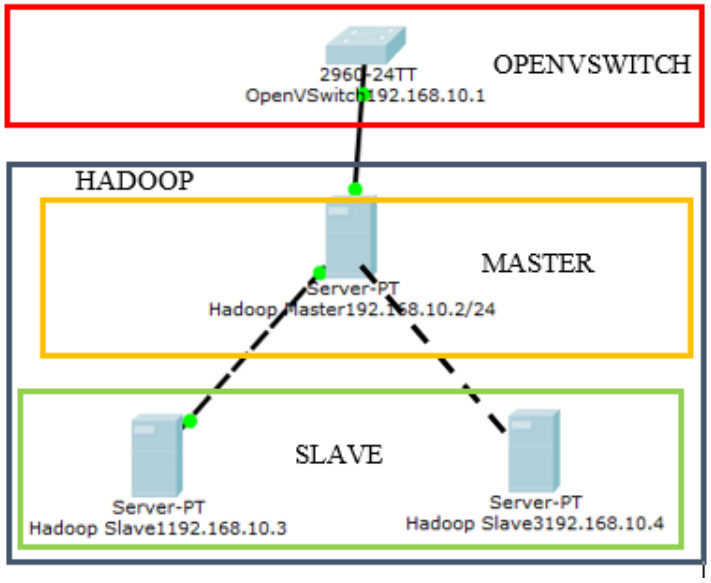
2. Metode Penelitian

Pada arsitektur jaringan *SDN* memisahkan dua komponen yaitu *Data Plane* dan *Control Plane.* Pada *data plane* SDN *datapath* diatur sesuai dengan perintah yang diberikan oleh *Controller.* Sedangan *control plane* sendiri bertugas mengontrol *data plane*. *Control plane* akan mengirimkan perintah kepada *data plane*. Dalam *SDN, Openflow* merupakan elemen terpenting. *Openflow* sendiri digunakan sebagai media komunikasi antara *controller* dengan switch *Openflow.*

Sedangkan pada *Hadoop Cluster,* pada penelitian ini akan menggunakan Multi node. Yang dimana terdapat *master* dan *slave.* Data yang akan di clustering terdapat di dalam *HDFS Master.* Yang selanjutnya akan dibagi ke dalam setiap *slave*.

Pada tahap ini dilakukan perancangan untuk analisa performasi trafik *Hadoop Cluster*ing dengan referensi dari literature/jurnal. Pada penelitian ini akan dilakukan pengujian pada performasi trafik *Hadoop Cluster*ing menggunakan *Software Defined* Network dengan topologi seperti Gambar 1.

Menggunakan Multi Node dalam Hadoop Clustering untuk pengujian performansinya. Control Plane dan Data Plane akan dipisahkan sesuai dengan konsep Software Defined Network. Hadoop yang berada didalam Container Docker selanjutnya akan diproses. Ketika melakukan proses deploying Hadoop Cluster didalam Docker Container, folder/files run-time dan hostname/IP Address terpecah sendiri. Jadi, konfigurasi Hadoop files didalam mesin yang berbeda dapat sama. Namun, hostname default otomatis di generated oleh Docker ketika Container berjalan. Hadoop Clustering terhubung ke dalam openvswitch. Untuk selanjutnya, Openflow akan diaktifkan pada switch bridge yang telah di buat sebelumnya.



Gambar 1. Topologi Sistem

Container Docker Master, Docker Slave 1, Docker Slave 2 terhubung dengan Openvswitch bridge yang telah dibuat sebelumnya yakni ovs-br1. Selanjutnya, container yang telah terhubung dengan switch bridge akan dicontrol dengan menggunakan protocol Openflow. Di dalam Virtual switch, settingan untuk flow monitorig dan QoS control dikonfigurasikan dari host machine.

Pada Tabel 1 ini, dilakukan pengujian Trafik jaringan *Hadoop Clustering* di dalam *Docker*. Pengujian ini akan dilakukan untuk melihat bagaimana tingkat Trafik pada *Hadoop Clustering* dengan besar dataset 858MB. Data tersebut disimpan didalam *HDFS*. Pengujian menjalankan *Hadoop Clustering* di dalam satu PC. Hal ini dilakukan agar dapat mengestimasi jumlah *virtual hosts* yang dapat didukung mesin dan memory yang digunakan oleh *MapReduce*.

Tabel 1. HDFS Process Number

Process Name	Number/Cluster
NameNode	1 per-cluster
SecondaryNameNode	1 per-cluster
DataNode	1 per-cluster

Tabel 2. YARN Process Number

Process Name	Number/Cluster
ResourceManager	1 per-cluster
ProxyServer	1 per-cluster
HistoryServer	1 per-cluster
NodeManager	1 per-cluster

Pada Tabel 2, dilakukan pengujian Trafik jaringan *Hadoop Clustering* di dalam *Docker*. Pengujian ini akan dilakukan untuk melihat bagaimana tingkat Trafik pada *Hadoop Clustering* dengan besar dataset 858MB. Data tersebut disimpan didalam *HDFS*. Pengujian menjalankan *Hadoop Clustering* di dalam satu PC. Hal ini dilakukan agar dapat mengestimasi jumlah *virtual hosts* yang dapat didukung mesin dan memory yang digunakan oleh *MapReduce*.

Pada pengujian pertama, melihat resource RAM yang digunakan di dalam proses Hadoop Clustering di empty cluster. Hadoop yang digunakan di dalam pengujian ini adalah Hadoop 2.7.3. Dengan total data yang diuji adalah 858MB.

3. Hasil Penelitian dan Pembahasan

Pengujian ini dilakukan dengan melakukan penyimpanan data di dalam HDFS Master. Uji coba ini dilakukan pada cluster multi node Hadoop yang terdiri dari Master, Slave 1 dan Slave 2 yang saling terhubung dengan OpenVSwitch. Pada pengujian ini ditujukan untuk melihat transfer rate pada multi node Hadoop Cluster yang diberikan traffic untuk membuat congestion sehingga

proses pada HDFS yang lain terjadi gangguan. Untuk mendapatkan trafiic tambahan yang bisa menyebabkan congestion, dibutuhkan iperf. Pada pengujian ini, akan dilakukan 3 skenario. Dataset yag digunakan sebesar 858 MB. Pada pengujian ini, akan dilihat 2 parameter QoS, yakni rata rata throughput dan juga packet loss.

a. Pengujian Hadoop Trafik Normal

Pada scenario Tabel 3 ini, akan dilakukan pengujian trafik jaringan rata-rata througput dan waktu eksekusi Hadoop Clustering. Tujuan dari skenario 1 adalah agar mendapat nilai tolak ukur trafik jaringan normal Hadoop.

Tabel 3. Nilai Awal Trafik Hadoop Cluster

Rata-Rata Throughput	Packet Loss
0	0

Pada saat dilakukan hasil pengujian Hadoop Multi node dengan trafik normal atau tidak terjadi penambahan beban oleh iperf, didapat bahwa rata-rata througput dan juga packet loss masih belum mengalami congestion. Maka dari itu untuk nilai awal didapatkan bahwa rata-rata throghput dan packet loss adalah 0.

b. Pengujian Hadoop dengan Congestion (tanpa Manajemen Transfer Rate)

Pada skenario ini, dilakukan pengujian trafik jaringan tanpa menggunakan QoS control. Semua trafik jaringan dibagi rata ke semua container. Iperf tool digunakan untuk generate trafik. Iperf server berjalan didalam host. Dan iperf client berjalan didalam Docker container hadoop master. Iperf client (host) akan mengirimkan trafik kepada Docker container hadoop-master yang sedang menjalankan proses dfs dan yarn. Tujuan dari pemberian trafik tambahan ini adalah untuk melihat packet loss dan throughput apabila terjadi proses tambahan pemberian trafik pada saat proses dfs dan yarn dijalankan.

Tabel 4.Trafik Hadoop Cluster dengan Beban iperf

Beban Iperf	Rata-Rata Throughput	Packet Loss (%)
250 MB	27.2 M/Sec	20 %
500 MB	58.5 M	23 %
1024 MB	34 M	28 %
2048 MB	65 M	27 %
3072 MB	81 M	26 %
4096 MB	98 M	29 %

Pada Tabel 4 rata-rata througput pada saat proses DFS dan Yarn dijalankan naik sejalan dengan naiknya beban iperf yang diberikan pada iperf client. Sedangkan packet loss juga akan semakin tinggi dengan semakin ditambahnya beban yang diberikan oleh iperf client.

c. Pengujian Hadoop dengan Congestion (dengan Manajemen Transfer Rate)

Pada skenario ini, dillakukan pengujian trafik jaringan dengan menggunakan QoS control. Untuk mengontrol QoS tersebut menggunakan OpenFlow menggunakan tools ovs-vsctl. Pada proses Manajemen Transfer Rate ini, akan diatur QoS Queue dengan QoS Policy dengan menggunakan bantuan Protokol Openflow. Pada bridge ovs-br1 akan diberikan pengaturan queue yakni nilai maksimum atau max-rate nya 500MB dan min-ratenya 300MB.

Tabel 5. Trafik Hadoop Cluster dengan beban iperf dengan QoS Queue

Beban Iperf	Rata-Rata Throughput	Packet Loss (%)
250 MB	0.71 M	9.1 %
500 MB	12 M	4.7 %
1024 MB	21 M	5.2 %
2048 MB	20 M	6.9 %
3072 MB	11 M	6 %
4096 MB	45 M	9.8 %

Seperti pada Tabel 5 dapat dilihat bahwa rata-rata throughput dan packet loss dapat dikurangi dengan menggunakan fitur QoS Queue. Berdasarkan 3 skenario uji coba diatas,

terlihat bahwa rata-rata throughput dan waktu eksekusi pada pengujian trafik pada Hadoop Cluster dengan menggunakan tambahan trafik pada aliran datanya tanpa adanya QoS control menggunakan QoS Queue dan QoS Policy terhitung tinggi. Hal ini dikarenakan karena aliran data yang tidak terkontrol dapat menyebabkan terjadinya congestion pada saat pengiriman datanya. Sedangkan pada Hadoop Cluster yang menggunakan QoS Control dengan memanfaatkan fitur queue yang dimiliki OpenFlow yang berfungsi sebagai manajemen transfer rate, congestion bisa dikendalikan dan dapat menurukan transfer rate dan rata rata throughput. Dengan melakukan manajemen transfer rate, waktu proses pengiriman data antar node tidak terlalu banyak peningkatan walaupun terdapat lalu lintas tambahan yang menyebabkan congestion.

4. Kesimpulan

Dari pengujian yang telah dilakukan untuk Analisa Performansi Trafik Jaringan pada Hadoop Cluster dalam Docker Berbasis Software Defined Network dapat disimpulkan bahwa :

1. Manajemen transfer rate dapat mengurangi congestion yang terjadi di dalam trafik arus data di dalam Hadoop Cluster.
2. Docker sebagai virtualisasi dapat mengurangi resource yang dibutuhkan pada saat installasi multi node Hadoop Cluster.
3. Aplikasi yang digunakan untuk manajemen transfer rate didalam SDN adalah QoS Queue yang memiliki fitur queue untuk mengatur transfer rate masing-masing node.

Referensi

[1] L. Liu, “Performance comparison by running benchmarks on hadoop, spark, and hamr,” 2015.

[2] A. Zaslavsky, C. Perera, and D. Georgakopoulos, “Sensing as a Service and Big Data,” *Proc. Int. Conf. Adv. Cloud Comput.*, pp. 21–29, 2012, doi: arXiv:1301.0159.

[3] P. Khusumanegara, “Analisis Performa Kecepatan Mapreduce Pada Hadoop Menggunakan Tcp Packet Flow,” p. 72, 2014.

[4] T. Garcia and T. Wang, “Analysis of big data technologies and method - Query large web public RDF datasets on amazon cloud using hadoop and open source parsers,” *Proc. - 2013 IEEE 7th Int. Conf. Semant. Comput. ICSC 2013*, pp. 244–251, 2013, doi: 10.1109/ICSC.2013.49.

[5] A. Aminudin and A. Cahyono, “Pengukuran Performa Apache Spark Dengan Library H2O Measuring Performance Apache Spark With Library H2O Using,” *JTIIK J. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 5, pp. 519–526, 2019, doi: 10.25126/jtiik.201961520.

[6] M. Frampton, *Big Data Made Easy*. Apress.

[7] P. Chouksey and A. S. Chauhan, “Weather Data Analytics using MapReduce and Spark,” vol. 6, no. 2, pp. 42–47, 2017, doi: 10.17148/IJARCCE.2017.6210.

[8] F. Dongyu, L. Zhu, and Z. Lei, “Review of hadoop performance optimization,” *2016 2nd IEEE Int. Conf. Comput. Commun. ICCC 2016 - Proc.*, pp. 65–68, 2017, doi: 10.1109/CompComm.2016.7924666.

[9] L. Cui, F. R. Yu, and Q. Yan, “When big data meets software-defined networking: SDN for big data and big data for SDN,” *IEEE Netw.*, vol. 30, no. 1, pp. 58–65, 2016, doi: 10.1109/MNET.2016.7389832.

[10] K. Sideris, R. Nejabati, and D. Simeonidou, “Seer: Empowering Software Defined Networking with Data Analytics,” *Proc. - 2016 15th Int. Conf. Ubiquitous Comput. Commun. 2016 8th Int. Symp. Cybersp. Secur. IUCC-CSS 2016*, pp. 181–188, 2017, doi: 10.1109/IUCC-CSS.2016.033.

ISSN: 2714-7975; E-ISSN: 2716-1382