Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 3, Maret 2018, hlm. 951-957 http://j-ptiik.ub.ac.id

Rancang Bangun Infrastruktur Pemrosesan Big Data Menggunakan Apache Drill (Studi Kasus: SIRCLO)

Yosef Hesekiel Partogi[1], Adhitya Bhawiyuga[2], Achmad Basuki[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]tobing.yosef@gmail.com, [2]bhawiyuga@ub.ac.id, [3]abazh@ub.ac.id

Abstrak

Berkembangnya penggunaan teknologi di masyarakat sangat mempengaruhi intensitas masyarakat dalam melakukan transaksi jual beli online, termasuk juga persaingan dari setiap penyedia jasa jual beli online. Agar dapat bersaing dengan semakin banyak *competitor* yang muncul, SIRCLO sebagai sebuah perusahaan *e-commerce,* membutuhkan analisis dari data-data yang SIRCLO dapatkan melalui segala aktifitas jual beli pada toko yang ada di SIRCLO. Tetapi untuk melakukan analisis tersebut, dibutuhkan sebuah sistem yang mampu membaca data mentah tersebut. Berdasarkan dari permasalahan tersebut, perlu adanya penelitian terkait rancangan infrastruktur. Penelitian ini menggunakan Apache Drill, HDFS sebagai tempat penyimpanan data, dan *script* berbasis *Python* untuk mengkonversi data dari MySQL ke JSON. Penelitian ini dimulai dari proses mengkonveri data dari sumber data (dalam penelitian ini menggunakan MySQL) ke JSON. Setelah itu dilakukan penyimpanan ke HDFS dan dilakukan *query* file menggunakan Apache Drill. Apache Drill sendiri dipilih karena fleksibilitasnya yang mampu melakukan *query* dengan *syntax* MySQL ke *plain text,* dan menggunakan konsep *schema free.* Untuk penyimpanan digunakan HDFS, diharapkan agar pengambilan data dari Apache Drill bisa lebih efektif dan manajemen data lebih baik, karena HDFS adalah penyimpanan data secara terdistribusi. Pengujian dilakukan dengan beberapa skenario, yaitu dari jumlah *server* yang digunakan dan besaran *file* yang digunakan. Parameter pengujian yang diambil mulai dari *resource usage* dari *server* dan lama waktu proses. Setelah penelitian ini selesai dilakukan terdapat beberapa kesimpulan, yaitu, didapatkan rancangan, komponen yang dapat memproses data milik SIRCLO, data dari MySQL berhasil diambil dan dinormalisasi menjadi JSON agar lebih mudah diolah, dan setelah rancangan diimplementasikan, infrastruktur ini dapat memproses data milik SIRCLO..

Kata kunci: *Big data*, *apache drill*, sistem terdistribusi.

Abstract

The Growing use of technology in society, really affect the intensity of society in doing online transaction for buy and sell items, including the competition between e-commerce companies. In order to compete with other e-commerce companies, SIRCLO, an e-commerce company, need to do an analytics to data that they have from all the transaction activities in their online shop, but to do that analytics, it needs a system that can read the raw dat. Based on those problem, this research is needed related to designing the infrastructure that can read those data. Basically, this research is using Apache Drill, HDFS as a file system, and script that written in Python to convert data from MySQL to JSON. This research starts from converting from data source (this research is using MySQL) to JSON, then will be stored in HDFS, and Apache Drill will do query to the file. Apache Drill is used because of the flexibility, it could do query with MySQL's syntax to plain text, and using schema free concept, also for file system is using HDFS because with hope that reading the data from distributed file system could be more effective and have better data management. This research was conducted with several scenarios, that is from the number of server that is used and size of the file, Parameter that's used is resource usage and process time of an activity. After this research is finish, this research acquired a design and component that can read SIRCLO's data, data from MySQL can be acquired and normalized to JSON, and after the design is implemented, this infrastructure can process SIRCLO's data.

***Keywords**: Big data, apache drill, distributed system*

1. PENDAHULUAN

Perkembangan internet sekarang ini sangatlah cepat, di mana menurut *Tech in Asia*, sebuah *website* berita teknologi terkemuka meluncurkan sebuah article tentang studi dari Google dan Temasek yang menjelaskan bahwa, transisi masyarakat di Asia Tenggara, terutama Indonesia menuju dunia maya sangatlah cepat. Hingga kedua perusahaan tersebut memperkirakan bahwa pengguna internet di Indonesia dari tahun 2015 hingga 2020, setiap tahunnya akan meningkat 19%. Studi tersebut juga mengatakan bahwa 18 juta masyarakat Indonesia masuk dalam kategori pembeli secara *online*, di mana menurut studi tersebut dikatakan bahwa pembeli secara *online* di Indonesia dari tahun 2015 hingga 2020 diperkirakan akan meningkat 21% setiap tahunnya (Freischlad, 2016). Hal itu mengakibatkan semakin bertambahnya jumlah toko *online* yang muncul, salah satu perusahaan yang bergerak pada bidang tersebut adalah SIRCLO, sebuah perusahaan rintisan yang bergerak di bidang *e-commerce.* Melihat pertumbuhan pasar seperti itu, dibutuhkanlah strategi-strategi baru untuk dapat bersaing dengan kondisi pasar.

Dalam pengambilan keputusan dan pembentukan strategi tersebut, tidak semerta-merta hanya menggunakan dugaan semata, dibutuhkan data yang konkrit dan faktual untuk melakukan hal tersebut. Data tersebutlah yang menjadi salah satu permasalahan dari SIRCLO. SIRCLO memiliki data yang banyak dan besar dari berbagai aspek, tetapi sulit untuk membaca data tersebut karena keterbatasan dari infrastruktur miliknya. Data milik SIRCLO ini terdapat berbagai macam, mulai dari data milik *sales*, *marketing*, *support*, *traffic website*, toko, dll. Setiap data yang dimiliki SIRCLO pun juga sangat dinamis, tiap hari nya selalu muncul data yang baru. Sebagai perusahaan *e-commerce*, salah satu sumber data tersebar milik SIRCLO adalah dari toko-toko yang ada di SIRCLO tersebut, yang terdiri dari pembelian barang, isi blog, user, dan masih banyak lagi. Dari data tersebutlah, SIRCLO bisa mengetahui informasi mulai dari kondisi pasar, kondisi toko, dan masih banyak lagi. Salah satu contohnya adalah dari database toko online milik SIRCLO, dari data itu bisa dilihat pada waktu kapan *user* paling banyak melakukan belanja lewat SIRCLO. Hal ini menjadi sebuah referensi untuk SIRCLO dan toko *online* nya saat ingin merilis barang baru atau melakukan *flash sale.* Contoh lainnya adalah dari data ini SIRCLO bisa melihat bank yang paling banyak digunakan oleh *customer* nya, sehingga menjadi referensi bagi SIRCLO saat ingin melakukan pemasaran yang menggunakan konten pembayaran ataupun diskon menggunakan bank tertentu. Data yang muncul tiap hari nya pun cukup fluktuatif, dari kecil hingga sangat besar. Ukurannya tergantung dari aktivitas dari toko-toko didalam SIRCLO, sehingga akan sangat sulit dan memakan waktu lama untuk memprosesnya secara manual. SIRCLO sendiri belum memiliki infrastruktur untuk memproses data tersebut, maka dari itu hal inilah yang menjadi sebuah *bottleneck* bagi SIRCLO, di mana data ini bisa menjadi kunci untuk SIRCLO mengambil keputusan-keputusan. Maka dari itu dibutuhkanlah rancangan infrastruktur dan komponen yang dapat memproses dapat memproses dapat tersebut.

Jika berbicara tentang infrastruktur pengolahan *big data*, infrastruktur tersebut dapat dibangun dengan berbagai cara. Salah satu caranya adalah dengan mengadopsi sebuah sistem terdistribusi dalam hal penyimpanan data dan komputasi (Demchenko, de Laat, & Membrey, 2014). Lalu seperti namanya, sistem terdistribusi adalah kumpulan dari komputer yang saling terhubung untuk mencapai suatu tujuan (Puder, Römer, & Pilhofer, 2006). Sebelum mebangun infrastruktur tersebut, hal pertama yang perlu diperhatikan adalah dari sisi desain dan rancangan sistem. Dalam desain dan rancangan sistem disini perlu diperhatikan dari sisi jumlah komputer, pemakaian *resource* dan perangkat lunak dari tiap-tiap komputer, media penghubung antar komputer, dll. Desain ini akan menjadi acuan saat membangun infrastruktur ini. Aplikasi yang mendukung sistem terdistribusi sudah sangatlah mudah untuk didapatkan, karena cukup banyak penyedia aplikasi tersebut yang bersifat *open source*. Tetapi untuk membangun semua hal tersebut tidaklah mudah, karena jika dilihat data-data milik SIRCLO, perusahaan ini memiliki sumber data yang berbeda-beda. Contoh kecilnya adalah *database* setiap toko di SIRCLO yang menggunakan MySQL. Akan sulit jika ingin membaca langsung dari MySQL, hal inilah yang menjadi permasalahan, di mana belum ada *script* atau alat yang bisa dipakai untuk mengambil data-data tersebut. Data yang berasal dari sumber berbeda pun bisa dikatakan memiliki format yang berbeda-beda juga, mulai

dari format, MySQL, JSON dan CSV. Perbedaan format tersebut akan menjadi suatu masalah untuk pemrosesan data, maka dari itu dibutuhkan alat yang mampu memproses format yang berbeda-beda. Dari keseluruhan penjelasan diatas, bisa dilihat bahwa permasalahaan lainnya adalah bagaimana untuk menyatukan semua kompononen-komponen dari semua permasalahan tersebut. Mulai dari pengambilan data dari sumber data, pengiriman data, dan pemrosesan data itu sendiri, dan dari hal ini lah yang membuat penulis mengangkat judul ini, di mana rancangan sistem cukup dibutuhkan oleh SIRCLO. Dengan membangun infrastruktur untuk mengolah data tersebut, sehingga data tersebut bisa di manfaatkan oleh berbagai pihak di SIRCLO.

Dalam mendasari penelitian ini, terdapat penelitian sebelumya yang dilakukan oleh Demchenko, dkk (2014) yang berjudul *Defining architecture components of the Big Data Ecosystem.* Pada penelitian ini didapatkan bagaimana cara untuk merancang sebuah infrastruktur dalam memproses sebuah *big data*. Hal ini menjadi acuan bagi penulis untuk melakukan rancangan dalam penelitian ini. Lalu dalam penelitian lain yang dilakukan oleh Bakshi. (2012) yang berjudul *Considerations for Big Data: Architecture and Approach.* Penelitian ini berfokus pada kapasitas dan spesifikasi dari tiap elemen untuk membangun infrastruktur *big data*. Penelitian ini juga menjadi referensi bagi penulis dalam merancang elemen-elemen pada penelitian ini

Penelitian sebelumnya yang menjadi referensi penulis dalam melakukan pengujian pada penelitian ini, ada pada penelitian yang dilakukan oleh Khusumanegara (2014) dengan penelitian berjudul Analisis Performa Kecepatan MapReduce Pada Hadoop Menggunakan TCP Packet Flow Analysis, di mana menguji performa MapReduce pada Hadoop. Dari penelitian ini didapatkan performa sebuah sistem terdistribusi saat dijalankan pada beberapa *physical machine* dan *virtual machine.* Lalu ada juga penelitian yang dilakukan oleh Warsiani (2013) dengan penelitian berjudul Analasis Performa Server Cloud Computing Pada Jaringan Privat Sistem Eucalyptus Dengan Infrastruktur Sebagai Sebuah Layanan, dari sini didapatkan cara menguji sebuah *server* dalam mejalankan sebuah layanan yang diukur dari ketersediaan *resource.* Dua penelitian tersebut menjadi referensi dalam penulis menjalankan penelitian ini nantinya. Beberapa hal yang berbeda dari penelitian ini adalah pada penelitian ini akan difokuskan pada merancang dan menguji rancangan infrastruktur tersebut untuk mengolah data milik SIRCLO, termasuk dari mendapatkan data-data tersebut. Berdasarkan semua latar belakang tersebut, maka penulis melakukan penelitian terkait infrastruktur pemrosesan *big data* yang akan menggunakan Apache Drill.

2. PERANCANGAN DAN IMPLEMENTASI

Pada perancangan, dijabarkan menjadi 5 bagian, yaitu kebutuhan fungsional, perancangan perangkat lunak, perancangan perangkat keras, batasan desain sistem, dan desain sistem.

Pada kebutuhan fungsional didapatkan tiga hal, yaitu pada tabel 1.

Tabel 1 Kebutuhan fungsional

No	Kebutuhan
1.	Data didapatkan dan diubah ke format yang mampu diproses oleh Apache Drill.
2.	Data mampu dikirim dari sumber data ke HDFS
3.	Apache Drill mampu melakukan *query* dengan data SIRCLO yang tersimpan di HDFS

Lalu dari sisi perancangan perangkat keras, digunakan PC di laboratorium FILKOM, karena penelitian ini akan disimulasikan di lab tersebut. Terdapat 3 buah PC yang akan bertindak sebagai *server* dengan spesifikasi Intel i3, 4GB RAM, 500GB HDD dan terhubung ke jaringan milik Universitas Brawijaya.

Untuk kebutuhan perangkat lunak dibutuhkan, *Python* yang akan digunakan untuk membangun *script* pengambilan data. Script tersebut menggunakan beberapa *module Python* yang terdiri dari MySQLdb dan json_util, Lalu rclone yang akan bertindak sebagai pengirim data antar subsistem. HDFS yang akan bertindak sebagai tempat penyimpan secara terdistribusi. Lalu yang terutama adalah Apache Drill dan ZooKeeper yang akan saling berkesinambungan dalam melakukan pemrosesan data yang disimpan didalam HDFS. Terakhir adalah virtualbox, yang akan membuat virtualisasi pada server yang telah disediakan dan akan menjalankan OS Ubuntu 16.10

Pada implementasinya, sistem ini memiliki beberapa keterbatasan mulai dari *software* dan

hardware. Agar penelitian ini bisa terarah dan sesuai yang diharapkan, maka perlu diimplentasi desain sistem, antara lain:

1. Setiap PC di virtualisasi menggunakan *Oracle Virtual Box* dengan OS *Ubuntu Server 16.04 LTS.*
2. Jumlah node yang digunakan berjumlah 8 node. Terdiri dari 2 node di PC utama, dan 3 node di masing-masing 2 PC lainnya.
3. Setiap PC dihubungkan oleh LAN yang terhubung ke *network* UB dan setiap *node* terhubung menggunakan *bridge adapter*.
4. *Apache Drill* dan HDFS berada pada VM yang berbeda

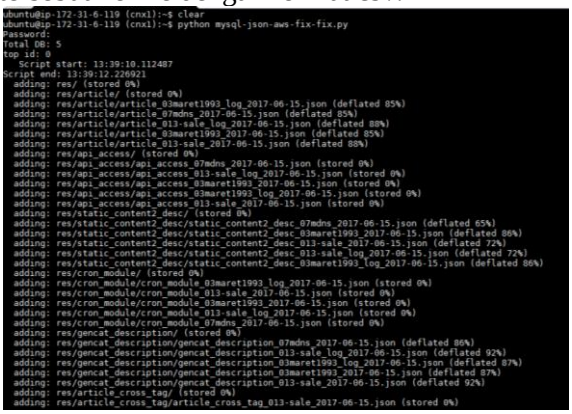
Dari sisi desain sistem, Sistem ini terdiri dari berbagai sub sistem yang pada akhir nya akan diintegrasikan. Berikut adalah gambar konsep desain dari sistem ini.

Gambar.1 Ilustrasi rancangan sistem

Berikut adalah penjelasan proses yang terjadi pada tiap nomor yang ada di gambar tersebut.

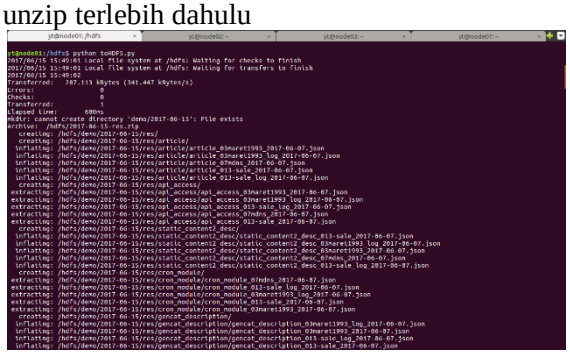
1. Proses konversi dan *dump* dari MySQL ke JSON *file*, kompresi ke format .zip, hingga mengirim data ke S3 menggunakan *script python* yang telah dibuat sebelumnya dan *rclone*. Pada saat proses ini berlangsung, sistem akan sekaligus mencatat waktu dan kecepatan saat mulai melakukan *dump* waktu selesai melakukan *dump*, mencatat *load average*, penggunaan *memory* dari *server* tersebut.
2. Proses mengirim data dari *Google Drive* ke *local server* menggunakan *rclone.* Pada saat mulai mengirim data, sistem akan mencatat waktu dan kecepatan saat mulai melakukan proses pengiriman.
3. Data yang sudah di download tersebut dimasukan ke HDFS, lalu dicatat lama proses penyimpanan data dan *load*, *memory usage* dari *server*.
4. Apache Drill melakukan query ke *file* yang ada didalam HDFS tersebut. *Load, memory usage* dari *server* saat melakukan proses ini akan dicatat, lama *query* akan dicatat didalam *log Apache Drill* sendri.
5. User bisa langsung mengakses hasil data lewat *web console* milik *Drill*

Setelah tahap perancangan, dilanjutkanah pada tahap implementasi. Pertama pada sisi pengambilan data, *script python* dijalan di EC2 SIRCLO agar dapat mengambil data lebih cepat dari database, karena database juga berada pada EC2. Setelah dijalankan, *script* akan melakukan pengambilan data hingga menyimpan data tersebut ke file dengan format csv.



Gambar 2 *Script* dijalankan dalam pengambilan data

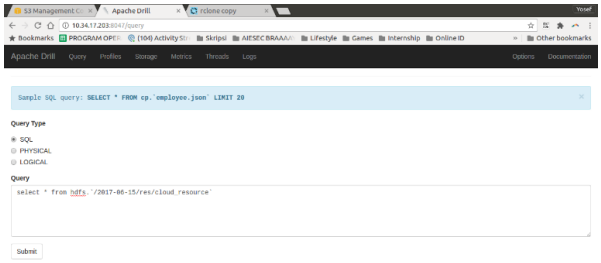
Setelah selesai dilakukan proses tersebut, *script* tersebut akan mengirimkan data tersebut ke S3 menggunakan rclone, untuk disimpan sementara sebeum dikirim ke HDFS yang berada di lab FILKOM. Gambar 3 adalah proses pengambilan data dari S3 ke server lab lalu langsung dimasukan setelah ke HDFS setelah di unzip terlebih dahulu



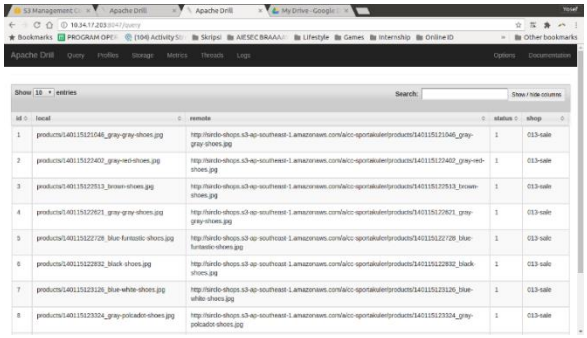
Gambar 3 Proses pengambilan data dan pengiriman ke HDFS

Pada gambar 4 dilakukan *query* dari *web console* ke HDFS sesuai dengan keinginan user, lalu hasil akan langsung ditampilkan dalam web

console tersebut pada gambar 5 sesuai dengan *query* yang dimasukan oleh user.



Gambar 4 Tampilan web console untuk melakukan query



Gambar 5 Hasil query dari drill ke file di HDFS

3. HASIL PENGUJIAN DAN ANALISIS

Pengujian yang dilakukan pada penelitian ini tediri dari 2 pengujian. Pengujian fungsional dan non fungsional. Pada pengujian non fungsional, akan terdiri dari 24 skenario, sedangkan pengujian fungsional akan berdasarkan pada tabel 1, dan pengujian tersebut telah dibuktikan pada bab 2 dalam implementasi sistem, dapat dilihat bahwa sistem berjalan lancar sesuai pada tabel 1

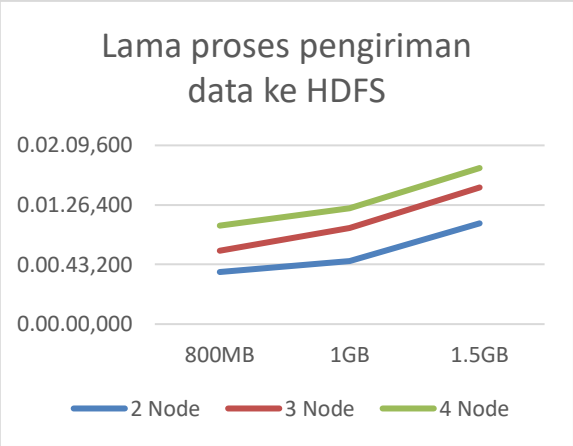
Pengujian pertama, dilakukan pengujian pada *script dump* MySQL. Pada bagian ini, akan diukur beberapa parameter pengujian yang akan diukur saat menjalankan proses ini *dump* dan pengiriman data ini untuk dilakukan analisis dan diambil kesimpulan. Berikut adalah tabel hasil pengujian dari seluruh skenario, dan pada sub bab berikutnya, akan ditampilkan grafik pemakaian *CPU* dan *Memory* pada tiap scenario

Tabel 2 Hasil pengujian

Jumlah database	**2000**	**5000**	**7000**
Lama waktu proses dump	00:32:44.44	00:48:48.04	01:05:23.23
Lama waktu pengiriman	5.4s	6.6s	11.17
Kecepatan pengiriman	42.00 MB/s	48.34 MB/s	48.43 MB/s
Besar file sebelum kompresi	3.0GB	5.6GB	6.6GB
Besar file sesudah kompresi	232.297MB	322MB	541MB

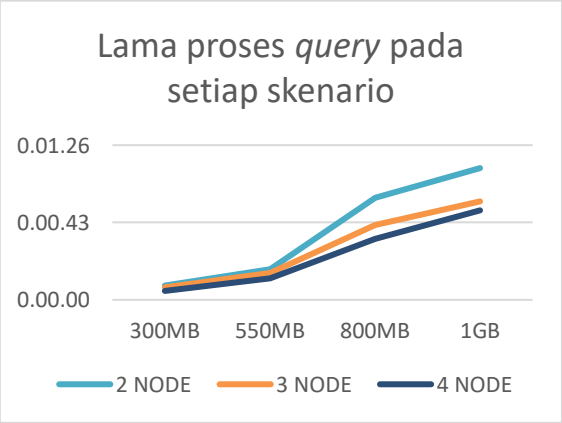
Dari data tabel 2 didapatkan bahwa lama waktu proses dump cukup linear, hal ini cukup baik disaat pengunaan kedepannya dapat memperkirakan lama waktu *dump* nya. Salah satu hal yang membuat lama proses *dump* tersebut adalah pembacaan list *database* pada awal proses, di mana script menggunakan query SHOW DATABASES untuk menyimpan seluruh nama *database*, untuk dilakukan *query.* Kedua adalah saat pembacaan dan penyimpanan data, di mana data disimpan tiap tabel dari masing-masing *database*, disimpan pada satu file yang berbeda. Sehingga *script* harus melakukan *query* untuk memilih database dan *query* untuk melihat isi dari tabel, setiap ingin menyimpan data ke CSV

Pengujian kedua, dilakukan pengujian pada pengiriman data ke HDFS. Pada bagian ini, akan dilakukan pengujian terhadap penyimpanan data ke HDFS, di mana *user* mengirim datanya ke HDFS dari *local storage*. Besar *file* yang disimpan terdiri dari 800MB, 1GB, 1.5GB, yang terdiri dari 2, 3 dan 4 *node*. Dikirim menggunakan perintah `hdfs dfs -put [alamat file di lokal] [alamat di HDFS]`, lalu akan di simpan data pemakaian CPU, memori, dan lama waktu pengiriman selama proses berlangusng. Pengujian ini dilakukan sebanyak 3 kali, lalu akan diambil rataan dari 3 percobaan itu. Hasil dari lama proses pengiriman data tersebut ada pada gambar

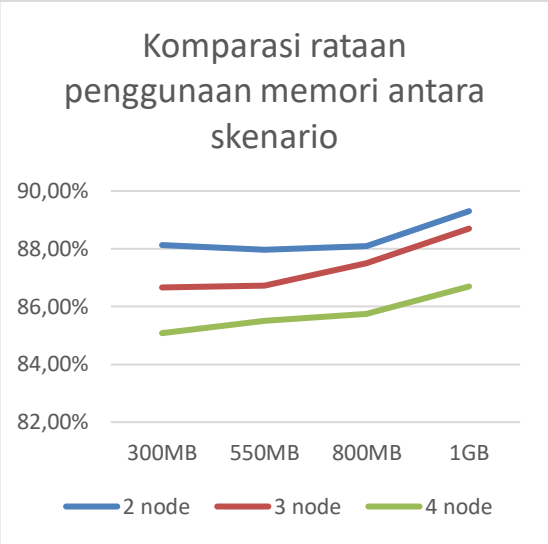


Gambar 6 Lama proses pengiriman data ke HDFS

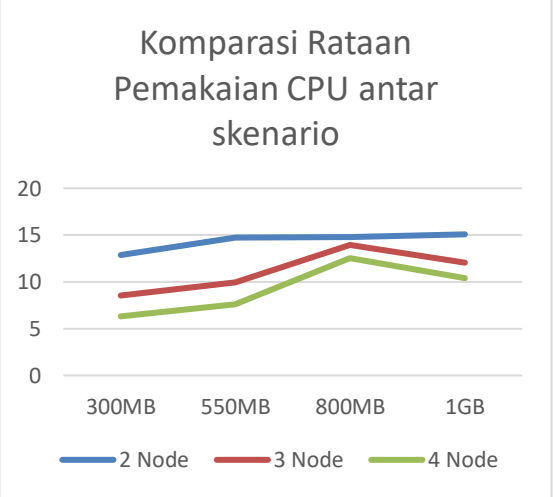
Pada pengujian berikutnya, dilakukan pada *query* dari Drill ke HDFS. Pada bagian ini, akan diukur beberapa parameter pengujian yang akan diukur saat menjalankan proses ini pengiriman data ke HDFS dan *query* dari *drill* ke HDFS ini untuk dilakukan analisis dan diambil kesimpulan. *Parameter* yang diukur adalah *CPU usage*, *memory usage*, lama proses *query*, *average peak CPU usage*, *average memory usage*, *presentasi pemakaian CPU tertinggi*, nilai pemaikaian memory tertinggi, presentasi rata-rata pemakaian memori. Hasil pengujian tersebut di representasikan dalam bentuk grafik pada gambar 7, lalu pemakaian memori dan CPU terdapat pada gambar 8 dan gambar 9.



Gambar 7 Lama proses query dari dari Drill



Gambar 8 Graik rataan pemakaian memori antar scenario



Gambar 9 Grafik rataan pemakaian CPU antar scenario

Dilihat dari gambar 7, lama waktu Apache Drill dalam memproses data cukup seimbang dengan besar *file* yang diproses. Lalu melihat dari pemakaian *resource* nya, Drill hanya memakai sedikit *resource* CPU dan lebih banyak menggunakan pada memori, di mana Drill memakai lebih dari 85 memori. Dari data tersebut dapat dilihat bahwa semakin banyak *resource* yang tersedia, maka Drill akan semakin memanfaatkan *resource* tersebut. Melihat dari meningkatnya pemakaian *resource* saat bertambahnya *node*.

4. KESIMPULAN

Rancangan dan komponen yang diajukan seperti pada bagian sebelumnya, dapat berjalan untuk memproses data milik SIRCLO, mulai dari pengambilan data (MySQL) dan dikonversi ke JSON menggunakan *script* berbasis *Python*, hingga pengiriman, penyimpanan, dan pembacaan data. Dari hal itu juga dapat dilihat bahwa infrastruktur ini sudah dikonfigurasi dan dintegrasikan sehingga dapat berjalan sesuai rancangan.

Berdasarkan hasil pengujian yang sudah dilakukan dalam penelitian ini, maka didapatkan beberapa kesimpulan. *Script* untuk mendapatkan data dari MySQL dapat bekerja dengan waktu yang cukup konsisten, di mana dapat melakukan *dump* dengan rata-rata kecepatan 1,7MB/s. Tetapi saat dijalankan, *script* tersebut mememakai sekitaar 90% memori, sehingga dapat dipertimbangkan untuk mengganti Bahasa pemrograman lain. Lalu secara keseluruhan, dapat dikatakan bahwa rancangan ini cukup efektif dalam menangani data-data SIRCLO kedepannya, melihat dari data-dari hasil percobaan diatas. Yang harus diperhatikan adalah dari sisi *resource server*, di mana spesifikasi *server* ditingkatkan agar dapat memproses data lebih cepat, karena dapat dilihat bahwa Drill memakai resource yang cukup besar untuk waktu sekitar satu menit. Jika ditingkatkan dari sisi *resource*, dapat dihasilkan kecepatan pemrosesan yang lebih baik.

5. DAFTAR PUSTAKA

Apache, 2010. *ZooKeeper Administrator's Guide.* [Online] Available at: www.zookeeper.apache.org/doc/r3.3.2/zookeeperAdmin.html [Accessed 14 Maret 2017].

Apache, n.d. *Apache ZooKeeper.* [Online] Available at: www.zookeeper.apache.org

Apache, n.d. *HDFS Architecture Guide.* [Online] Available at: www.hadoop.apache.org

Bakshi, K., 2012. *Considerations for Big Data: Architecture and Approach,* Herndon: IEEE.

Borthakur, D., 2010. *Hadoop 1.2.1 documentation.* [Online] Available at: https://hadoop.apache.org

Chawda, R. K. & Thakur, G., 2016. *Big data and advanced analytics tools.* Indore, IEEE.

Demchenko, Y., de Laat, C. & Membrey, P., 2014. *Defining architecture components of the Big Data Ecosystem.* Minneapolis, IEEE.

Freischlad, N., 2016. *Google: Indonesia will dominate Southeast Asian ecommerce.* [Online] Available at:https://www.techinasia.com/google-temasek-ecommerce-data-indonesia

Hausenblas, M. & Nadeau, J., 2013. Apache Drill: Interactive Ad-Hoc Analysis at Scale. *Creative Commons.*

Hortonworks, n.d. *Apache Hadoop HDFS.* [Online] Available at:https://hortonworks.com/apache/hdfs/ [Accessed 13 Maret 2017].

JSON, n.d. *JSON.* [Online] Available at: www.json.org

Khan, M. A., Memon, A. Z. & Khan, S., 2012. *Highly Available Hadoop NameNode Architecture.* Kuala Lumpur, IEEE, p. 2.

Khusumanegara, P., 2014. *Analisis Performa Kecepatan MapReduce Pada Hadoop Menggunakan TCP Packet Flow Analysis,* Depok: FT UI.

MongoDB, n.d. *MongoDB API.* [Online] Available at: WWW.api.mongodb.com/python/current/api/bson/json_util.html