Implementasi Replikasi File Server Dengan Metode Chunking Menggunakan Unstructured Database

by Elbama Giofandra

Submission date: 20-Nov-2019 03:50AM (UTC+0700)
Submission ID: 1217372231
File name: ver_Dengan_Metode_Chunking_Menggunakan_Unstructured_Database.pdf (1.42M)
Word count: 3105
Character count: 20362

Implementasi Replikasi File Server Dengan Metode Chunking Menggunakan Unstructured Database

Metode chunking adalah skema untuk memotong file menjadi beberapa potongan kecil dan masing-masing potongan memiliki ukuran file yang sama. Metode ini mampu membuat sebuah file dengan kapasitas besar menjadi beberapa file dengan kapasitas kecil. Dengan demikian metode ini dapat secara langsung mengurangi beban server ketika terjadi proses upload dan download. Metode chunking juga memungkinkan adanya proses upload dan download secara berkala dan *resumeable* dikarenakan potongan file dikirim secara berkala. Hal tersebut akan sangat membantu kinerja server dalam keadaan internet yang tidak stabil dan memadai. Tentu dengan beberapa pendekatan lain terkait dengan database yang digunakan dan skema penanganan file tersebut setelah melalui server. Unstructured database digunakan untuk menyimpan dan mengolah file hasil metode chunking. Database ini menyimpan file hasil metode chunking sebagai dokumen. Dikarenakan tingkat fleksibilitasnya yang tinggi database jenis ini mampu melakukan replikasi terhadap dokumen-dokumen di dalamnya dan disimpan kedalam database cadangan. Sehingga database utama hanya terfokus pada menyimpan dokumen saja. Untuk request terhadap proses unduh akan diarahkan ke database cadangan. Beberapa penelitian terakhir terkait dengan metode chunking memunculkan kelebihan metode ini termasuk dalam hal mempercepat proses pemindahan sebuah file dengan kapasitas besar. Termasuk dengan beberapa kelebihan lain dalam penerapannya pada sistem seperti remote data compression, sinkronisasi dan data deduplikasi. Sehingga apabila ditinjau dari konsep dasar metode chunking mampu menjadi solusi atas permasalahan yang ada juga memunculkan hal-hal baru dalam menciptakan lingkungan sistem yang baik dan tangguh. Fokus dari penelitian ini adalah mengkaji metode chunking dalam arsitektur sistem repository untuk memaksimalkan kinerja server pada kondisi internet yang tidak stabil dan memadai. Juga untuk mengkaji sistem yang dapat menyelesaikan masalah High availability atau load balancing terhadap layanan unduh file.

Kata kunci—chunking; resumaeble; high-availability; load balancing; unstructured database

I. PENDAHULUAN

Berdasarkan data International Telecommunication Union (ITU), Pada tahun 2017 akses terhadap internet telah mencapai angka 3.9 Miliar lebih di seluruh dunia [10]. Berkembangnya teknologi dengan pesat memunculkan tantangan terhadap arsitektur sistem yang mumpuni, Hal tersebut dipengaruhi oleh perancangan repository dan server. Secara langsung kondisi tersebut berpengaruh terhadap permintaan untuk menghadirkan arsitektur sistem dan server yang sempurna. Kebutuhan terhadap penyimpanan data yang besar pun meningkat apabila meninjau dari data akses diatas. Jenis-jenis file yang di transfer oleh pengguna pun semakin besar dan bervariasi. Ada beberapa faktor yang memperkuat tantangan tersebut yaitu bervariasinya kondisi pengakses internet, Banyak ditemukan kondisi konektivitas yang unreliable dan unstable yang menyebabkan lemah atau hilangnya konektivitas pada saat proses unggah dan unduh data. Kondisi seperti ini dapat disebabkan oleh beberapa faktor teknis, seperti lokasi pengakses internet yang berada pada kawasan yang jauh dari jangkauan access point ataupun BTS (Based Transceiver Station) sehingga sinyal yang ditransmisikan ke access point ataupun BTS relatif lemah karena jarak akses yang jauh [1]. Ataupun adanya hambatan terhadap sinyal yang sedang ditransmisikan, hal ini bisa berupa objek, struktur alam di sekitar maupun cuaca. Namun pada faktanya dalam keadaan internet yang stabil dan memadai masih sering didapati terputusnya koneksi secara mendadak maupun terjadinya error. Hal tersebut sangat berpengaruh apabila pengguna sedang melakukan upload file pada sebuah repository. Faktor-faktor di atas memperkuat tantangan yang semakin penting untuk diselesaikan. Pendekatan terhadap arsitektur sistem repository maupun rancangan server yang fleksibel menjadi sangat penting untuk dilakukan. Terlepas dari solusi-solusi yang bisa dilakukan akan lebih baik apabila memunculkan sisi terdalam dari tiap-tiap solusi untuk mendapatkan hasil terbaik. Sehingga akan mendapatkan kajian yang maksimal untuk mendapatkan teori baru maupun menghadirkan potensi solusi yang besar.

II. Latar Belakang

Ditinjau dari angka akses pengguna terhadap internet yang semakin meningkat memunculkan kebutuhan untuk menyediakan penyimpanan data yang besar. Hal ini juga dipengaruhi oleh semakin besarnya ukuran file yang di unggah oleh pengguna. Juga semakin bervariasinya kondisi pengguna ketika mentransfer file yang tidak jarang menyebabkan file gagal diunggah. Oleh karena itu dibutuhkan suatu sistem yang reliable terhadap tantangan-tantangan yang ada. Tantangan yang dihadapi adalah:

- Putusnya konektivitas saat proses unggah file dilakukan. Hal ini bisa dikarenakan besarnya kapasitas file yang di unggah terlalu besar sehingga membutuhkan kecepatan internet yang besar juga. Beberapa sistem melakukan pendekatan dengan membatasi ukuran file sehingga pengguna dipaksa untuk mengunggah file yang berukuran kecil(kompres file).
- Penyimpanan data yang tidak fault-tolerance (penyimpanan data konvensional). Dalam penyimpanan data konvensional data yang diupload disimpan sebagai file biasa. Sehingga tidak bisa dilakukan pengolahan data yang lebih fleksibel.
- High availability atau load balancing terhadap layanan unduh file(disk i/o yang terbatas). Karena penyimpanan file yang konvensional maka implementasi load balancing sangat berat dikarenakan harus melakukan clone pada seluruh file. Namun tetap rawan terhadap kasus putusnya konektivitas[2].

Beberapa penelitian sebelumnya telah berusaha memunculkan kajian tentang masalah serupa.
Terutama tentang metode yang digunakan untuk mengunggah file agar lebih baik.
Metode chunking itu sendiri adalah proses untuk memecah sebuah file yang utuh menjadi beberapa potongan dengan kapasitas yang sama. Hal ini dimaksudkan untuk membuat sebuah file dengan kapasitas yang besar menjadi beberapa potongan kecil dengan kapasitas file kecil, Sehingga dalam proses pemindahan file dari/ke repository memiliki beban yang kecil terhadap server [3]. Dan juga tidak memerlukan penggunaan koneksi yang terlalu lama sehingga putusnya koneksi dapat diminimalisir. Apabila meninjau dari konsep dasarnya, sangat layak jika metode chunking ini di aplikasikan dalam beberapa sistem/konsep yang terkait dengan masalah memaksimalkan kinerja server dan mempercepat proses pemindahan file dari/ke repository. Sehingga metode chunking adalah modal untuk menjadi komponen penting dalam menciptakan sistem yang mampu mengatasi masalah yang ada. Namun di dalam sebuah arsitektur sistem diperlukan beberapa komponen yang saling beradaptasi satu sama lain untuk membuat kinerja keseluruhan sistem maksimal. Metode chunking itu sendiri dalam prosesnya menghasilkan potongan-potongan data yang banyak sehingga dibutuhkan penyimpanan data yang dapat menyimpan potongan-potongan tersebut dan menggabungkannya. Fitur penyimpanan key-value pada unstructured database sangat membantu dalam menyelesaikan permasalahan ini dimana tidak terdapat pada structured database engine. Sehingga diperlukannya komponen database yang mampu menjadi repository bagi file yang dihasilkan dari proses metode chunking. Database yang diperlukan haruslah mampu menangani file hasil dari proses chunking sebagai dokumen/file. Meninjau dari hal tersebut, database yang digunakan haruslah berjenis unstructured database. Karena database jenis ini memiliki fleksibilitas tinggi dan mampu mengolah file yang dihasilkan dari proses chunking. Unstructured database memiliki sifat scheme-free yang berarti mampu menyimpan dokumen/file tanpa mendefinisikan terlebih dahulu struktur dokumen yang bisa disimpan didalamnya. Dokumen/file yang tersimpan dalam database jenis ini mampu diolah sebagai dokumen sehingga data file tidak terstruktur bisa diolah di dalamnya [4]. Oleh sebab itu unstructured database menjadi komponen yang layak diimplementasikan ke dalam rancangan sistem bersama dengan metode chunking yang digunakan. Pada akhirnya skema perancangan dengan komponen yang dianalisa di atas akan mampu menyajikan arsitektur sistem yang menjawab masalah yang ada. Implementasi dari komponen-komponen tersebut akan menjadi sebuah gagasan dimana wujud pendekatannya terfokus pada masalah kondisi internet yang unreliable dan unstable.

III. Dasar Teori

A. Metode Chunking

Metode chunking adalah proses untuk memecah sebuah file yang utuh menjadi beberapa potongan dengan kapasitas file yang sama.

B. File Chunked

File Chunked merupakan sebutan untuk potongan file yang berasal dari hasil pemecahan pada metode chunking. File chunked hasil chunking adalah potongan tunggal yang memiliki format byte dan tidak memiliki sifat ataupun format dari file utuhnya.

C. Metadata

Metadata adalah informasi yang digunakan untuk menemukan kembali sebuah data. Informasi ini bisa berisi waktu, besaran ataupun jenis data. Metadata dapat disematkan pada sebuah file untuk memberi informasi yang dapat mencari tahu tentang file tersebut.

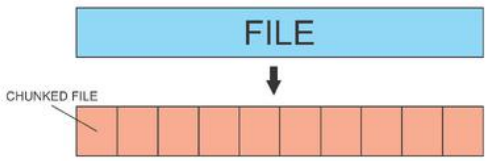
D. Unstructured Database

Unstructured Database adalah database yang tidak menggunakan SQL-base atau bisa disebut NoSQL. Database ini memiliki fleksibilitas tinggi dalam mengolah data didalamnya. Segala bentuk file pun dapat disimpan pada database jenis ini.

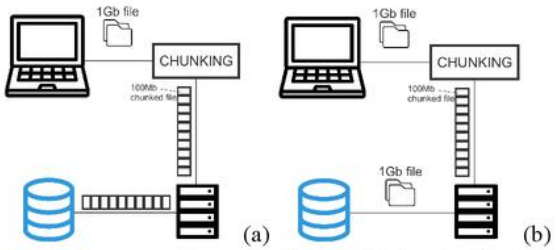
IV. METODOLOGI

A. Metode Chunking

Metode chunking digunakan untuk memecah file menjadi beberapa potongan dengan ukuran file yang sama. Metode ini mampu membuat file dengan kapasitas besar menjadi beberapa potongan file dengan kapasitas kecil. Dalam hal ini metode chunking digunakan untuk mengurangi beban pengiriman file pada server dan mempercepat waktu pengiriman. Karena pengiriman file dengan metode ini dilakukan secara berkala. Proses pengiriman menggunakan chunking juga memungkinkan adanya *resumeable*, Mengingat bahwa pengiriman dilakukan potongan demi potongan. Untuk itu metode ini sangat relevan dengan masalah yang diangkat, Terutama untuk masalah kondisi konektivitas yang unstable dan unreliable. Dengan *resumeable* dimungkinkan adanya automatic pause ketika konektivitas terputus. Namun metode ini memerlukan komponen yang bisa mengolah file hasil chunking. Sehingga perlunya penyesuaian terhadap teknologi atau komponen lain yang ada pada arsitektur sistem [5].



Gambar 1. Metode chunking yang bekerja pada client-side sebelum memasuki server. File yang dipecah menjadi beberapa bagian disebut chunked file, Format atau data file yang dihasilkan setelah melalui proses ini adalah data file tidak terstruktur. Warna file awal dengan hasil chunking yang berbeda pada gambar 1 adalah tanda bahwa kumpulan file chunked tidak memiliki sifat seperti file awal sebelum melewati proses chunking. Masing-masing file chunked memiliki kapasitas dan sifat yang sama. File chunked yang diteruskan ke dalam database akan memiliki metadata pada masing-masing file sebagai informasi file tersebut agar mudah disatukan kembali ketika ada request terhadap file semula datang. Untuk membedakan metadata masing-masing file chunked biasanya menggunakan indikasi waktu, yaitu berdasarkan waktu tiap-tiap file chunked sampai pada suatu repository melalui server.



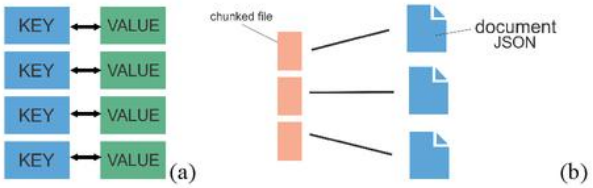
Gambar 2. Penanganan file yang telah melalui tahap chunking. (a) file diteruskan ke database dengan keadaan chunked (b) file chunked disatukan kembali sebelum diteruskan ke database.

File yang telah melewati proses chunking memiliki beberapa pendekatan yang bisa dilakukan pada sisi server yaitu menggabungkan file chunked menjadi file yang utuh atau meneruskan file chunked tersebut untuk disimpan ke dalam database. Apabila chunked file tersebut digabungkan kembali menjadi file semula maka komputasi yang terjadi pada sisi server akan menjadi berat. Dikarenakan adanya kinerja tambahan bagi server untuk menggabungkan chunked file tersebut. Namun apabila chunked file diteruskan dan disimpan ke dalam database maka server hanya bekerja sekali untuk memecah file pada proses awal saja. Sehingga beban komputasi pada server akan berkurang dan chunked file yang telah disimpan bisa di unduh secara langsung. Hal ini membuat kinerja server maksimal dan efektif, akibatnya arsitektur yang dirancang menggunakan metode chunking mampu memunculkan beberapa kelebihan.

B. Unstructured Database

Unstructured database digunakan karena dapat menyimpan dan mengolah data yang berekstensi unik didalamnya salah satu database jenis ini adalah MongoDB. Database jenis ini bersifat scheme-free yaitu tidak diperlukan merancang struktur database sebelum menyimpan file. Fleksibilitas database ini sangat baik, jenis file yang tidak terstruktur mampu disimpan dan diolah dengan database jenis ini. Tidak ada struktur kolom

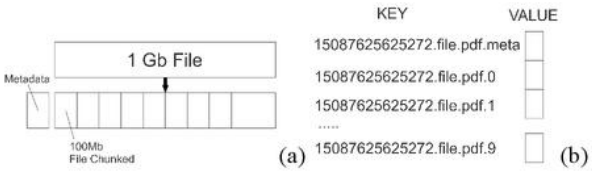
dan baris dalam database ini, sehingga pengelolaan file didalamnya dapat diolah sesuai kebutuhan. Database ini menyimpan file tidak terstruktur sebagai dokumen dan pengelolaannya menggunakan JavaScript Object Notation (JSON). JSON disini memudahkan proses pertukaran data karena penggunaannya yang mudah dan komputasinya yang ringan. Terlebih JSON memungkinkan untuk memanggil sebuah file/dokumen berdasarkan metadata [6 , 9]. Dokumen pada database ini memiliki unique key tersendiri pada setiap dokumen, namun memungkinkan adanya dokumen dengan unique key yang sama. Hal ini dikarenakan database jenis ini mampu membuat secondary-database dengan mereplikasi database utama beserta dokumen di dalamnya [6, 7].



Gambar 3. (a) Konsep key-value pada database ini menyimpan file dalam bentuk array. File hasil metode chunking disimpan sebagai array dan tiap-tiap potongan ini adalah value yang memiliki key masing-masing. (b) Kemudian diatur ke dalam database ini menjadi collection berbentuk dokumen. Dengan konsep array ini memudahkan untuk file hasil chunking diolah dan disimpan.[7]

C. Identifer-key

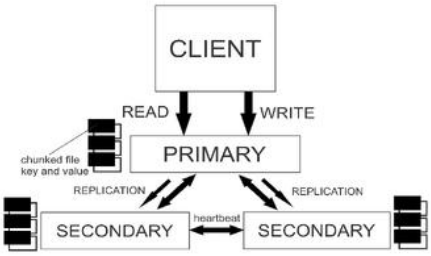
Untuk mengelola file chunked yang disimpan ke dalam database dengan prinsip key-value maka diperlukan identifier-key pada setiap potongan file chunked. Hal ini dimaksudkan untuk memudahkan dalam memanggil potongan file tersebut untuk disatukan kembali. Juga untuk membedakan kelompok file chunked satu dengan kelompok file chunked lainnya. Ada beberapa cara untuk membuat Identifier-key pada file chunked salah satunya adalah waktu. Sehingga file chunked di masukan ke dalam database berdasarkan urutan waktu pada database [8]. Selain itu perlu adanya pengelompokan terhadap file chunked untuk penanda bahwa potongan-potongan file chunked merupakan satu kelompok. Pengelompokan ini menggunakan metadata sebagai informasi terhadap kelompok file chunked.



Gambar 4. (a) File diberi metadata sebagai informasi kelompok file chunked sebelum dimasukan kedalam database. (b) menunjukan metode penamaan key time_mills.filename dengan value berisi potongan file chunked. Sehingga apabila file dengan kapasitas 1 Gb diunggah ada 10 file chunked dengan kapasitas 10 Mb + 1 metadata.

D. Implementasi sistem

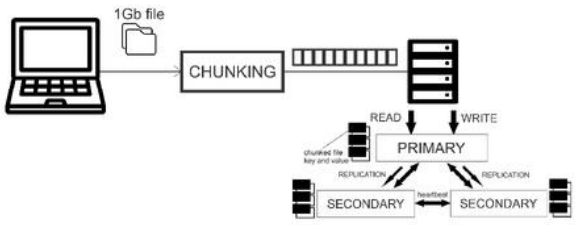
Secara keseluruhan sistem dibagi menjadi 3 bagian utama yaitu client-side, server dan database. Rancangan sistem yang dikaji pada penelitian ini memiliki beberapa metode dan teknologi yang telah dijelaskan pada sub-bab sebelumnya. Metode chunking memiliki peranan untuk memecah file pada client-side. Dalam pembahasan sebelumnya juga telah dipaparkan penanganan pada file hasil chunking dapat dilakukan dengan 2 hal, yaitu meneruskan dalam kondisi chunked atau menyatukan kembali sebelum disimpan. Setelah mengkaji berdasarkan efektifitas maka file hasil chunking akan diteruskan dalam kondisi chunked. Rancangan pada bagian database menggunakan unstructured database untuk menangani file hasil chunking. Resumeable yang digunakan untuk mengatasi konektivitas yang terputus dilakukan pada bagian database. Hal ini dilakukan dengan mereplikasi database utama menjadi secondary database untuk memaksimalkan kinerja server dalam hal request dan response melalui database yang terpisah. Cara ini juga akan meningkatkan High availability atau load balancing terhadap layanan unduh file.



Gambar 5. Database utama dengan dokumen didalamnya direplikasi untuk meningkatkan high availability dan load balancing. Hal ini dikarenakan adanya pengalihan terhadap proses unduh ke secondary database. Heartbeat

berguna untuk pertukaran informasi ketika terjadi sesuatu dengan database induknya.

Rancangan sistem ini memproses file yang masuk dengan memotong file menggunakan metode chunking pada client-side. File chunked hasil proses ini dikirim melalui server menuju database secara berkala tiap-tiap potongan. Sebelum file itu dikirim oleh server database file chunked dikonfigurasi untuk mendapatkan identifier-key agar mudah diolah kembali setelah disimpan. Potongan-potongan file chunked tersebut masuk ke dalam database sebagai dokumen JSON. Setelah dokumen-dokumen tersebut masuk ke dalam database, maka database akan melakukan replikasi terhadap setiap dokumen di dalam primary database ke secondary database. Hal ini dilakukan untuk melayani proses unduh dan memfokuskan primary database sebagai tempat menyimpan dokumen saja.



Gambar 6. Proses penanganan file dalam sistem dari tahap file tersebut dikirim oleh pengguna sampai file tersebut mengalami replikasi dan di simpan secondary database.

V. Hasil dan Pembahasan

Dalam mengkaji metode dan teknologi yang digunakan untuk menyelesaikan masalah yang diangkat, diperlukan beberapa pembuktian untuk melihat kinerja metode chunking dan rancangan database. Sehingga penting untuk membandingkan dengan metode atau teknologi yang sudah diterapkan. Dalam mengkaji metode dan teknologi diperlukan indikator-indikator untuk memunculkan hasil dari kinerja metode dan teknologi di dalamnya. Indikator yang digunakan haruslah merepresentasikan tingkat keberhasilan metode dalam menangani masalah yang ada. Dalam hal ini indikator seperti waktu,performa sistem ketika pengujian dilakukan, dan kinerja metode di dalam arsitektur sistem haruslah dikaji. Seperti yang telah dijelaskan dalam bab sebelumnya metode dan teknologi yang akan diuji adalah metode chunking, server, dan database, dengan skema pengujian yang sama terhadap metode yang telah diterapkan. Hal ini dimaksudkan untuk membuktikan bahwa rancangan yang diajukan dalam penelitian ini mampu dan relevan untuk dijadikan solusi. Berikut adalah materi pengujian yang dilakukan terhadap sistem yang dibuat.

A. Percobaan

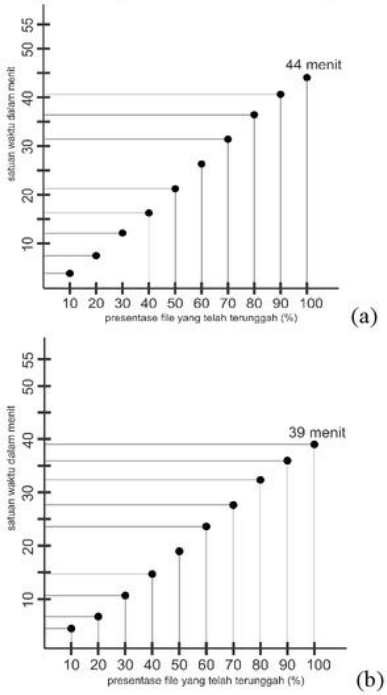
Percobaan dalam kasus ini adalah menjalankan sistem yang telah dirancang untuk mengkaji kinerja setiap komponen dan secara keseluruhan. Dalam hal ini metode chunking yang menjadi komponen dalam sebuah rancangan arsitektur sistem dijalankan lalu diamati kinerjanya secara individu dan sistem keseluruhan. Percobaan dimaksudkan untuk menguji konsep dasar metode chunking pada tahap implementasi dan sebagai pembanding digunakan arsitektur sistem tanpa metode chunking dengan kata lain sistem reguler yang biasa digunakan. Indikator yang digunakan pada percobaan ini adalah waktu yang dibutuhkan sistem dalam proses pemindahan file dari/ke repository. Dalam menjalankan percobaan ini ada beberapa ketentuan yang dijalankan.

- Ukuran file. Mengingat bahwa percobaan ini sangat berkaitan erat dengan waktu maka ukuran file harus ditentukan sebelumnya. Juga untuk menyeragamkan kapasitas file yang digunakan dalam menguji metode chunking dimana adanya proses memecah file menjadi potongan-potongan kecil. Dalam percobaan ini digunakan file dengan ukuran 1 Mb dan diamati kinerja tiap sistem dalam menangani file tersebut sampai tahap repository.
- Checkpoint. Ketentuan ini penting dibuat untuk mengkaji secara detail proses yang terjadi pada setiap sistem. Terlebih berhubungan dengan waktu tempuh setiap sistem saat pemindahan file dari/ke repository. Pada ketentuan ini digunakan checkpoint pada 30%, 50% dan 100%. Dimana presentase tersebut menyatakan progres file yang sedang terunggah maupun terunduh. Hal ini dimaksudkan untuk memantau metode chunking dan rancangan server dalam konsistensinya terhadap setiap checkpoint yang ditentukan.

Hasil yang akan diperoleh pengujian ini adalah perbandingan waktu antara rancangan sistem menggunakan metode chunking dengan sistem konvensional pada umumnya.

B. Hasil Pengujian

Setelah dilakukan pengujian dengan beberapa ketentuan yang dijelaskan pada sub-bab sebelumnya. Ada beberapa hasil yang didapatkan dengan membandingkan rancangan sistem metode chunking dengan sistem konvensional. Pertama adalah hasil dari pengujian dengan mengunggah file dengan ukuran 100Mb dengan kecepatan internet 3,6 Mbps.



Gambar 7. (a) Menunjukan kecepatan proses unggah pada rancangan sistem tanpa menggunakan metode chunking di dalamnya. (b) Grafik kecepatan unggah file menggunakan metode chunking.

Sistem dengan metode chunking menunjukan waktu tempuh yang lebih cepat. Hal ini dikarenakan adanya pengiriman berkas yang dilakukan secara berkala. Sistem dengan metode chunking menunjukan konsistensi pada setiap potongan file chunked yang dikirim. Ada beberapa hal yang perlu dicermati secara mendalam. Berkaitan dengan pengujian proses unduh yang telah dilakukan. Pada sistem yang menggunakan metode chunking didapati bahwa waktu yang dibutuhkan untuk mengirim 10Mb file pertama lebih lama. Hal ini dikarenakan sistem melakukan proses chunking untuk memecah file tersebut. Setelah hal itu dilakukan proses pengiriman file stabil dan cepat dilakukan.

VI. KESIMPULAN

Penelitian ini telah mengkaji metode chunking untuk mengetahui apakah metode ini dapat diimplementasikan ke sebuah arsitektur sistem. Metode chunking terbilang cukup berhasil dalam memunculkan solusi atas permasalahan yang ada. Berdasarkan hasil pengujian yang dilakukan rancangan sistem menggunakan metode chunking mampu menjawab semua tantangan yang telah diidentifikasi sebelumnya. Beberapa konfigurasi tambahan pada komponen lain memunculkan kelebihan-kelebihan yang membuat rancangan sistem ini maksimal. Sistem ini mampu melakukan proses unggah dalam keadaan internet yang buruk sekalipun. Diharapkan juga bahwa adanya penelitian lebih lanjut mengenai masalah yang serupa dengan penelitian ini. Penelitian ini juga diharapkan membuka penelitian-penelitian lanjutan lain untuk memunculkan solusi terbaik.

[1] Ilker Nadi Bozkurt1, Anthony Aguirre. Why Is the Internet so Slow?!, Springer International Publishing AG 2017

[2] Dildar Husain, Mohammad Omar. Load status Evaluation For Load Balancing Database Server (2019).

[3] Ryan NS Widodo, Hyotaek Lim, Muhammad Atiquzamman. SDM: Smart deduplication for mobile cloud storage. Future Generation Computer Systems Volume 70, May 2017, Pages 64-73

[4] Chieh Ming Wu, Yin Fu Huang, John Lee. Comparisons Between MongoDB and MS-SQL Databases on the TWC Website.American Journal of Software Engineering and Applications 2015; 4(2): 35-41

[5] ChaoHu, Ming Chen, Changyou Xing .Towards efficient video chunk dissemination in peer-to-peer live streaming. Computer Networks Volume 57, Issue 15, 29 October 2013, Pages 3009-3024.

[6] Mohammed-Amine, Giorgio Ghelli, Dario Callazo. Schemas And Types For JSON Data.

[7] Thanh Trung Nguyen & Minh Hieu Nguyen. (2016).Distributed and High Performance Big-File Cloud Storage Based On Key-Value Store. International Journal of Networked and Distributed Computing, Vol. 4, No. 3, 159-172.

[8] Marius Cristian MAZILU. Database Replication (2016). Database Systems Journal vol. I, no. 2

[9] JSON Schema language. http://json-schema.org.

[10] Global Internet User https://www.itu.int

Implementasi Replikasi File Server Dengan Metode Chunking Menggunakan Unstructured Database

ORIGINALITY REPORT

4%	3%	1%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	article.sciencepublishinggroup.com Internet Source	1%
2	dblp.dagstuhl.de Internet Source	1%
3	www.coursehero.com Internet Source	1%
4	Submitted to Virginia International University Student Paper	<1%
5	Ryan N. S. Widodo, Hyotaek Lim. "RDM", Proceedings of the 8th International Conference on Computer Modeling and Simulation - ICCMS '17, 2017 Publication	<1%
6	lisha.ufsc.br Internet Source	<1%
7	www.aplikasiandroid.xyz Internet Source	<1%
	Submitted to Universitas Brawijaya	

8	Student Paper	<1%
9	Jianwei Zhang, Xinchang Zhang, Chunling Yang. "Towards the multi-request mechanism in pull-based peer-to-peer live streaming systems", Computer Networks, 2018 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		