Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 9, September 2018, hlm. 2770-2776 http://j-ptiik.ub.ac.id

Desain dan Implementasi *In-Network Caching* Pada *Content Centric Networking* Menggunakan CCN-Lite Dengan Simulator OMNeT++

Ibrahim[1], Achmad Basuki[2], Eko Sakti Pramukantoro[3]

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

Abstrak

Content Centric Networking (CCN) adalah sebuah arsitektur yang diusulkan untuk mengatasi permasalah pada Internet saat ini dalam hal pendistribusian data. CCN ini sudah berbasis *content centric*, dimana pencarian data tersebut dilakukan menggunakan nama dari konten, bukan dari lokasi konten (*host centric*). CCN juga mempunyai mekanisme *caching* yang berbeda dengan *caching* secara umum yang disebut dengan *in-network caching*. Penelitian ini bertujuan untuk mengetahui bagaimana kinerja dari *in-network caching* pada CCN dengan melakukan desain dan implementasi menggunakan CCN-Lite di lingkungan OMNeT++. Parameter yang diuji adalah latensi pengiriman dan *cache hit ratio* (CHR). Hasil dari simulasi yang dilakukan, diperoleh latensi pengiriman terkecil yaitu 10.69 ms. Nilai ini didapatkan dari terjadinya proses *caching*, jika dibandingkan dengan latensi pengiriman tanpa *caching* yaitu sebesar 32.45 ms. Hal tersebut membuktikan bahwa dengan adanya *caching*, latensi yang didapatkan semakin rendah. CHR tertinggi didapatkan pada protokol file CCNTLV yaitu sebesar 0.54 atau 54%, jika dibandingkan dengan protokol file CCNB sebesar 0.45 atau 45%. Hal ini membuktikan bahwa semakin tinggi nilai CHR maka semakin baik pula konten tersebut dilayani.

Kata kunci: *in-network caching*, *content centric networking*, CCN-Lite, CHR, latensi pengiriman.

Abstract

Content Centric Networking (CCN) is a proposed architecture for addressing current Internet issues in terms of data distribution. The CCN is content-centric based, where the data search is performed using the name of the content, not the content host. CCN also has a different caching mechanism than caching in general called in-network caching. This study aims to find out how the performance of in-network caching on CCN by design and implementation using CCN-Lite in OMNeT ++ environment. The parameters tested were delivery latency and cache hit ratio (CHR). The result of the simulation is done, the smallest delivery latency is 10.69 ms. This value is obtained from the caching process, when compared with the latency of delivery without caching that is equal to 32.45 ms. It proves that with the caching, the latency gets lower. The highest CHR obtained in the CCNTLV protocol file is 0.54 or 54%, when compared with the CCNB file protocol of 0.45 or 45%. This proves that the higher the value of CHR the better the content is served.

Keywords: *in-network caching, content centric networking, CCN-Lite, CHR, delivery latency.*

1. PENDAHULUAN

Content Centric Networking (CCN) adalah sebuah arsitektur yang diusulkan untuk mengatasi permasalahan pada Internet saat ini dalam hal pendistribusian data. CCN ini sudah berbasis *content centric* atau dalam melakukan pencarian data berdasarkan pada nama konten. Hal ini memberikan keuntungan pada CCN jika dibandingkan dengan arsitektur Internet yang melakukan pencarian data berdasarkan lokasi dari data (*host centric*). Ditambah lagi dengan adanya protokol CCNx yang mendukung dalam pencarian data konten. CCN ini dikembangkan berdasar dari konsep *Information Centric Networking* (ICN). Terdapat banyak arsitektur yang telah diusulkan berdasarkan konsep ICN seperti *Named Data Networking* (NDN), *Network of Information* (NetInf), *A Data-Oriented Network Architecture* (DONA), dan masih banyak lagi, tidak terkecuali *Content Centric Networking* (CCN) (Wang, dkk., 2011).

Akan tetapi, arsitektur-arsitektur tersebut masih belum ada yang dioperasionalkan.

CCN memiliki 2 tipe paket pengiriman yaitu *Interest Packet* atau disebut paket *request* dan *Data Packet* atau paket yang berisi konten. CCN juga memiliki 3 struktur data yaitu *Content Store* (CS), *Pending Interest Table* (PIT) dan *Forwarding Information Base* (FIB). Ketika sebuah *client* melakukan permintaan data, maka paket *Interest* akan dikirimkan menuju *router*. Pada *router* dilakukan pencarian di dalam CS, apakah *cache* dari data tersebut ada atau tidak, apabila ada maka data tersebut akan dikirimkan menuju *client* yang meminta tanpa harus menuju ke *server*. Pada CCN juga memiliki model penghapusan (*cache replacement*) dan model penyimpanan *cache* (*cache strategy*) atau biasa disebut juga dengan *In-network Caching* (Jacobson, 2009).

Penelitian yang terkait dengan konsep CCN dan *In-network caching* yaitu seperti (Wang, et al., 2011) yang mengusulkan sebuah *cache policy* baru yaitu LB (Least Benefit) untuk mengatasi permasalahan *in-network caching* serta meningkatkan performansinya. (Wu, et al., 2015) yang menerapkan ESCC (*End System Caching and Cooperation*) untuk meningkatkan performansi dan efisiensi dari pada *caching* secara umum. (Wang, 2013) yang melakukan optimasi model dan desain heuristic serta mengembangkan *caching* dari berbagai multi obyektif pada *cache* jaringan dengan mengukur kinerja dari tiap-tiap *caching*. Akan tetapi, belum banyak penelitian terkait dengan kinerja *in-network caching* pada CCN-Lite.

Penelitian ini bertujuan untuk mendesain dan mengimplementasikan *in-network caching* pada CCN-Lite untuk mengetahui bagaimana kinerja *in-network caching* yang terdapat pada CCN-Lite, terutama pada aspek latensi pengiriman dan *cache hit ratio* (CHR). Dalam penelitian ini, pengujian dilakukan pada lingkungan simulator OMNeT++ dengan bantuan pustaka pendukung INET Framework.

Dalam penelitian ini, penulis melakukan desain topologi simulasi dan skenario pengujian pada CCN-Lite.Simulasi ini menggunakan topologi 6 *client*, 4 *router*, dan 1 server. Dalam topologi ini pengiriman paket dilakukan dalam 2 jalur dan setiap *file* dapat dikirim atau diatur secara static pada konfigurasi file.ini di dalam OMNeT++. Terdapat dua tipe file yang diminta di dalam simulasi ini yaitu CCNB dan CCNTLV.

Sisa penjelasan dari jurnal ini disusun sebagai berikut: Bab 2, menjelaskan dasar-dasar teori yang digunakan dalam simulasi. Bab 3, menjelaskan perancangan simulasi. Bab 4, implementasi perangkat lunak dan konfigurasi. Bab 5, hasil pengujian serta analisis. Bab 6, kesimpulan

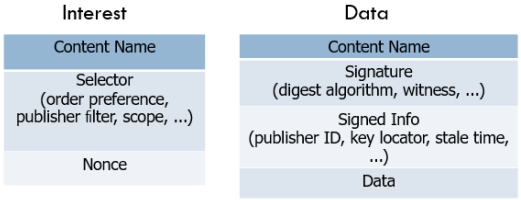
2. BAHASAN UTAMA

2.1 Content Centric Networking

Content Centric Networking atau biasa disingkat CCN adalah suatu arsitektur baru di dalam dunia jaringan dimana data yang dikirim berupa sebuah konten. CCN ini dikembangkan berdasar dari konsep *Information Centric Networking* (ICN). ICN adalah sebuah terobosan baru yang ditujukan untuk mengembangkan arsitektur dari Internet yang jauh dari paradigma *host-centric* berdasarkan prinsip *end-to-end*, menjadi arsitektur jaringan yang berfokus "*named information*"(atau Content). Arsitektur ini diharapkan memiliki manfaat yaitu pengingkatan efisiensi, skalabilitas yang lebih baik dengan informasi atau permintaan *bandwidth* dan ketahanan yang lebih baik dalam skenario komunikasi (Jacobson, 2009).

CCN merupakan pendekatan alternatif untuk arsitektur jaringan berbasis pada prinsip bahwa jaringan komunikasi harus memungkinkan pengguna untuk fokus pada data yang dia butuhkan, lokasi fisik, dan dari mana data yang akan diambil. CCN memungkinkan *caching* konten untuk mengurangi kemacetan dan meningkatkan kecepatan pengiriman, konfigurasi sederhana perangkat jaringan, dan keamanan yang dibangun ke dalam jaringan di tingkat data (Ahir & Kumbharkar, 2012).

Tujuan dari CCN adalah untuk memberikan jaringan yang lebih aman, fleksibel dan terskala sehingga dapat mengatasi kebutuhan modern Internet untuk distribusi konten yang aman pada skala besar untuk beragam perangkat (Jacobson, et al., 2009).



Gambar 1. *Interest Packet* dan *Data Packet*
Sumber: (PARC Company, 2010)

Pada Gambar 1, komunikasi pada CCN dilakukan dengan menggunakan dua tipe paket

yang berbeda, yaitu paket *Interest* dan paket Data. Paket Interest adalah sebuah paket permintaan yang dikirimkan menggunakan nama data. Paket Data adalah paket hasil permintaan dari paket *Interest* yang disebut juga dengan *Content Object*. Secara umum, *Content Object* terdiri atas potongan-potongan data yang disebut dengan *Chunks* (Ahir & Kumbharkar, 2012). CCN memiliki tiga struktur data yaitu *Content Store*, *Pending Interest Table*(PIT), dan *Forwarding Information Base*(FIB). Beberapa istilah yang muncul di dalam konsep CCN seperti CS , *faces* (penyebutan lain dari *interface*), CCNx (protokol yang digunakan pada CCN) (Jacobson, et al., 2009). Penjelasan struktur data yang ada pada CCN adalah sebagai berikut:

- *Content Store* (CS): Sebuah tempat penyimpanan sementara data *cache* yang didapat dari penerimaan paket data. *Cache* yang disimpan di CS bersifat sementara, artinya sewaktu-waktu dihapus.
- *Pending Interest Table* (PIT): Struktur data yang menyimpan *Interest* beserta *face* yang meminta. Ketika data yang dicari tidak ditemukan pada CS maka, PIT akan menyimpan nama data beserta dengan asal *face* yang meminta sebelum diteruskan menuju FIB. Apabila ada *faces* lain meminta data yang sama, maka *Interest* dari *face* tersebut akan di drop dan ditambah ke PIT.
- *Forwarding Information Base* (FIB): Struktur data yang sama dengan tabel IP *routing*. Meneruskan *Interest* menuju *next-hop* dari *router* awal. Proses yang digunakan dalam pencarian jalur adalah *longest-prefix matching*.

2.2 *In-Network Caching*

In-network caching adalah suatu mekanisme penyimpanan data *cache* yang ada di dalam arsitektur *centric network*. *In-network caching* ini memiliki tujuan meningkatkan efisiensi jaringan dan kinerja dari distribusi konten dengan menyimpan *cache* dari konten tersebut kedalam penyimpanan *cache* (Yuemei Xu, 2016). Pada *in-network caching*, apabila *cache* tersebut didaptkan dari sebuah *node*, maka disebut dengan *Cache Hit*, dan jika *cache* tersebut tidak dididapatkan maka disebut dengan *Cache Miss*. *In-network caching* dibagi menjadi dua jenis strategi yaitu *On-path caching* dan *Off-path caching*.

On-path caching adalah strategi dimana konten yang disimpan atau yang akan dicari masih dalam jalur yang dilalui oleh paket data. Di dalam mekanisme *on-path caching* terdapat beberapa jenis strategi yang dapat digunakan yaitu *Leave Copy Everywhere* (LCE) dimana strategi LCE akan menyimpan replika konten pada setiap *node* yang dilalui oleh paket data menuju pengguna. Strategi ini sangat cocok digunakan pada jaringan dengan trafik tinggi. *Leave Copy Down* (LCD), dimana setiap terjadi *cache hit*, maka konten akan direplikasi pada *node* satu level di bawahnya dalam *cache* hirarki. Dengan kata lain, semakin popular konten tersebut maka *cache* tersebut akan disimpan sedekat mungkin di *cache store*. *Random*, dimana konten akan disimpan pada satu *node* yang dipilih secara acak di jalur pengirimannya. *ProbCache*, dimana strategi ini akan menyimpan dari replika konten pada satu *node* terbaik disepanjang jalur pengiriman berdasarkan perhitungan probabilitas dari beberapa parameter.

Off-path caching atau nama lain dari *longest path* adalah mekanisme dimana konten yang disimpan atau dicari tidak berada di dalam jalur yang dilalui oleh paket data tetapi berdasarkan aturan yang telah ditentukan (Saucez, et al., 2012).

2.3 Cache Hit Ratio (CHR)

Cache Hit Ratio adalah perbandingan jumlah permintaan yang berhasil dilayani (cache hit) oleh *cache router* dengan total jumlah permintaan yang dikirimkan. Persamaan untuk menghitung CHR adalah:

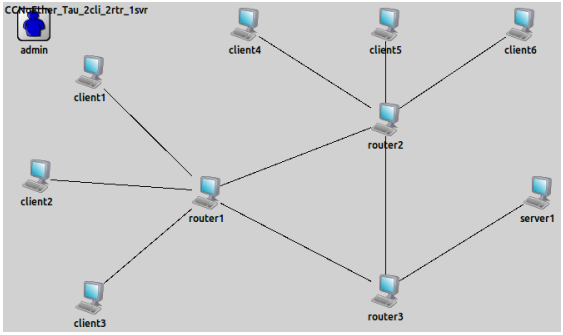
$$\mathrm{CHR} = \mathrm{N(s)} \,/\, \mathrm{N(T)} \qquad (1)$$

N(s) adalah jumlah permintaan berhasil dilayani (*cache hit*) dan N(T) adalah total permintaan yang dikirimkan. Nilai dari CHR berkisar antara 0 dan 1, dengan 0 berarti tidak ada permintaan yang berhasil dilayani *cache* dan 1 berarti semua permintaan berhasil dilayani oleh *cache*.

3. PERANCANGAN SIMULASI

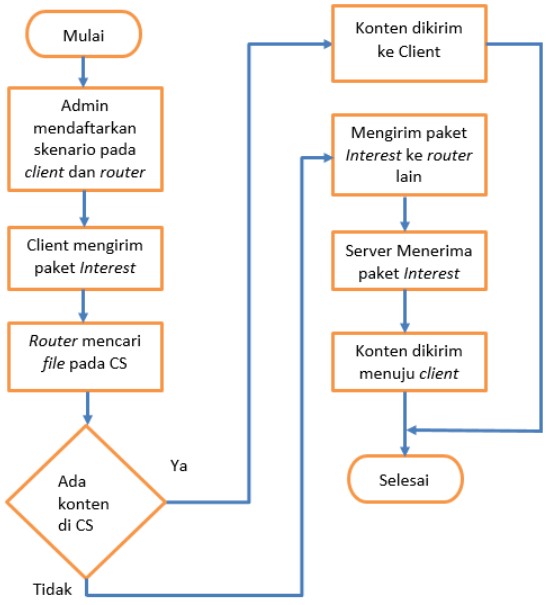
3.1 Perancangan Topologi

Agar tujuan dari penelitian ini dapat tercapai, perlu adanya sebuah topologi simulasi. Topologi ini harus memiliki node-node yang berperan sebagai *requester*, *cache router*, *content source*, dan inisiator simulasi. Dalam penelitian ini, penulis menggunakan 1 admin sebagai inisiator simulasi, 1 server sebagai *content source*, 3 router sebagai *cache router*, dan 6 client sebagai *requester*.



Gambar 2. Topologi Pengujian

Gambar 3 adalah sebuah diagram alir dari bagian simulasi yang dilakukan. Pertama kali admin akan melakukan perdaftaran skenario yang berisikan nama konten yang akan diminta, konten yang akan dilalui dan tipe konten yang disimpan. Ketika telah mendaftarkan pada semua *node*, admin akan mengirimkan sebuah perintah pada *client* untuk meminta *file* sesuai dengan waktu yang telah ditentukan. *Client* akan mengirimkan paket *Interest* menuju *router*.



Gambar 3. Diagram Alir Simulasi

Di dalam *router* akan memeriksa apakah konten yang diminta terdapat di dalam *Content Store* (CS), jika ada maka konten akan dikirimkan menuju *client* yang meminta, apabila tidak maka paket *Interest* akan dikirimkan menuju *router* selanjutnya dan memeriksa apakah ada konten yang diinginkan hingga menuju ke server. Ketika server telah menerima paket *Interest*, maka server akan mengirimkan konten menuju *client* yang meminta.

3.2 Perancangan Pengujian

Pada tahap ini dilakukan pengujian untuk mengetahui bagaimana implementasi dari *in-network caching* di CCN-Lite dan bagaimana kinerja dari *in-network caching* pada CCN-Lite di lingkungan OMNeT++. Pengujian dilakukan dengan menjalankan simulasi CCN pada OMNeT++ berdasarkan *workspace* dari CCN-Lite.

Tabel 1. Jadwal Waktu Permintaan

Periode (s)	Tipe File	Client	Jumlah Chunk
1	CCNB	1 dan 4	100 & 100
2	CCNTLV	2 dan 5	75 & 75
3	CCNB	3 dan 6	75 & 60
4	CCNTLV	1 dan 6	115 & 115
5	CCNB	2 dan 5	60 & 75
6	CCNTLV	3 dan 4	60 & 60

Pengujian dilakukan untuk mendapatkan hasil latensi dari awal pengiriman *chunk* hingga konten didapatkan oleh *client* dan untuk mendapatkan nilai CHR pada *client*. Untuk mendapatkan hasil simulasi, dilakukan penjadwalan waktu pengiriman permintaan seperti pada Tabel 1.

4. IMPLEMENTASI

4.1 Instalasi Perangkat Lunak

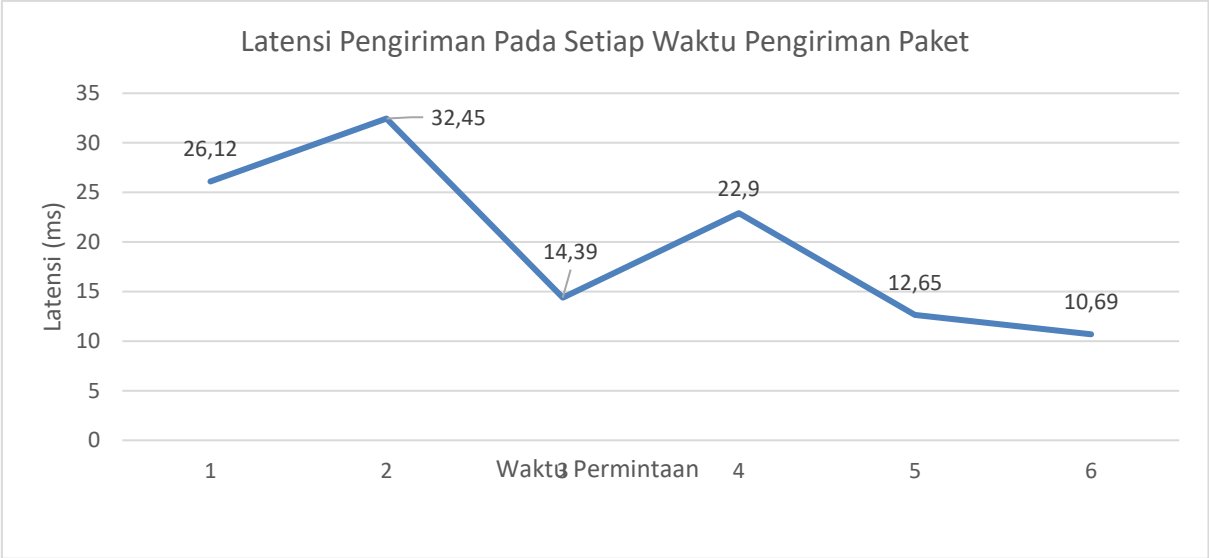
Sebelum menjalankan simulasi ini, terdapat 3 perangkat lunak yang perlu disiapkan yaitu CCN-Lite v.3.0, OMNeT++ v.4.5 dan INET Framework v.2.4. Perangkat lunak ini memiliki tugas masing-masing, seperti CCN-Lite sebagai *workspace* dari implementasi CCN, OMNeT++ sebagai simulator, dan INET Framework sebagai *library* dari OMNeT++. Dalam simulasi ini, CCN-Lite sangat bergantung pada INET karena bagian jaringan seperti *client*, *router*, *server* dan admin diambil dari *library* ini.

4.2 Konfigurasi Topologi, Node dan File .ini

Pada bagian ini dilakukan perancangan simulasi pengujian yaitu dengan melakukan konfigurasi topologi pengujian. Setelah konfigurasi topologi telah dilakukan, dilanjutkan dengan konfigurasi pada setiap *node* dan *file* omnetpp.ini yang berada di dalam *workspace* CCN-Lite. Konfigurasi ini dilakukan untuk mengatur *file* apa yang akan diminta oleh *client*, kemudian penentuan jalur pengiriman paket dan *file* apa saja yang berada di dalam *server* serta penempatan *file* konfigurasi tiap *node* pada omnetpp.ini.

5. SIMULASI DAN ANALISIS

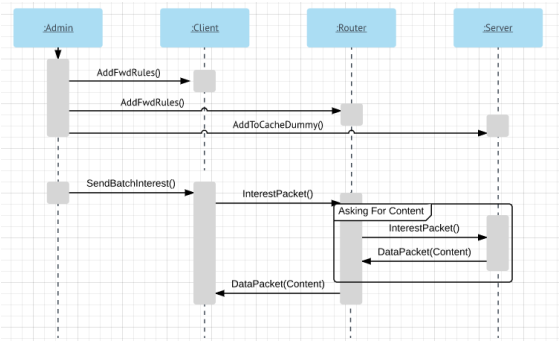
Pada tahap ini, dilakukan pengujian untuk mendapatkan hasil dari simulasi pada



Gambar 5 Grafik Latensi Pengiriman Paket

OMNeT++. Simulasi ini dilakukan sebanyak 1 kali dengan jadwal waktu pengiriman yang telah dibuat mengacu pada Tabel 1.

Tujuan dari simulasi ini adalah untuk mengetahui bagaimana komunikasi CCN yang terjadi di dalam CCN-Lite, bagaimana mekanisme dan kinerja *in-network caching* serta untuk mengetahui latensi dan CHR dari simulasi.



Gambar 4. *Sequance Diagram* Simulasi

Alur dari simulasi ini ditunjukkan seperti pada Gambar 4 dalam bentuk *sequence diagram*. Admin memberikan perintah inisialisasi kepada *node*, ketika telah dilakukan maka dilanjutkan dengan proses permintaan file. Pada saat *Interest* berada di dalam router, router tersebut akan melakukan pencarian konten pada CS, apabila ditemukan data yang diminta maka data tersebut akan segera dikirimkan menuju *client* yang meminta, apabila tidak ditemukan maka akan diteruskan menuju *router* lain hingga menuju *server*.

Ketika proses inisialisasi, maka dilanjutkan dengan admin memberikan proses *sendBatchInterest* pada *client*. Proses ini memberikan perintah kepada *client* untuk memulai meminta file sesuai dari waktu permintaan yang telah dibuat pada Tabel 1.

5.1 Hasil Pengujian

Berdasarkan hasil simulasi yang telah dilakukan merujuk pada jadwal waktu permintaan pada Tabel 3.2, CCN-Lite dapat mendesain dan mengimplementasikan *in-network caching* dengan baik berdasarkan topologi yang telah dirancang. Bekerjanya *in-network caching* pada simulasi ini dikarenakan adanya pemberian *cache slot* dan *cache size* pada konfigurasi omnetpp.ini. Proses *caching* terjadi pada waktu permintaan ke 2 sampai waktu permintaan ke 6. Hal ini dibuktikan pada Gambar 5 grafik hasil latensi pengiriman. Untuk *cache replacement* dan *cache decision* yang digunakan adalah LRU/LCE dimana pasangan *cache* ini yang paling sering digunakan pada arsitektur *centric network* lain

Pada Gambar 5, latensi tertinggi didapatkan pada waktu permintaan ke-2 yaitu sebesar 32.45 ms. Nilai ini diperoleh karena pada permintaan tersebut belum terjadi *caching* dan menyebabkan pengambilan data diambil menuju server. Latensi terendah didapatkan pada waktu permintaan ke-6 yaitu sebesar 10.69 ms. Nilai ini diperoleh karena data didapatkan dari *router* terdekat atau terjadi proses *caching*. Pada waktu ke-4 terjadi proses *caching* dan *non-caching*. Proses *caching* terjadi dari permintaan *chunk* 1 hingga 75 dan untuk chunk 76 hingga 115 dilakukan proses non-caching atau meminta data menuju server. Dari waktu inilah didapatkan nilai sebesar 22.9 ms. Jadi, dapat diambil kesimpulan bahwa dengan adanya proses *caching*, latensi pengiriman yang diperoleh

semakin rendah dari pada tanpa menggunakan *caching*.

Tabel 2. Hasil Perhitungan CHR Pada Permintaan CCNB

Client	Total Chunk Request	Total Chunk Hit	Cache Hit Ratio
1	100	0	0
2	60	0	0
3	75	75	1
4	100	0	0
5	75	75	1
6	60	60	1
Jumlah	470	210	
N(Hit) / N(Req)			0.45

Tabel 2 adalah tabel dari perhitungan *Cache Hit Ratio* ketika meminta file CCNB movie1. Dalam tabel tersebut didapatkan hasil CHR sebesar 0.45 atau 45% dari total konten yang diminta. *client2* dan *client4* mendapatkan nilai CHR 0 karena *client* tersebut tidak mengirimkan paket *Interest* sehingga permintaan tidak dilayani baik oleh *router* maupun *server*. Sedangkan *client1* mendapatkan nilai CHR 0 karena permintaan dilayani oleh *server* bukan *cache router*.

Tabel 3. Hasil Perhitungan CHR Pada Permintaan CCNTLV

Client	Total Chunk Request	Total Chunk Hit	Cache Hit Ratio
1	115	75	0.65
2	75	0	0
3	60	60	1
4	60	60	1
5	75	0	0
6	115	75	0.65
Jumlah	500	270	
N(Hit) / N(Req)			0.54

Pada Tabel 3, didapatkan hasil *cache hit ratio* pada permintaan *file* movie2 protokol CCNTLV yaitu sebesar 0.54 atau 54%. Dalam tabel 5.3, client2 dan client5 mendapatkan nilai CHR 0 karena paket *Interest* yang dikirimkan dilayani oleh *server* dan bukan *cache router* dan permintaan yang dilakukan adalah permintaan pertama kali pada *file* movie2. Pada client1 dan client6, *cache hit* yang didapatkan sebanyak 75. Nilai ini diperoleh karena pada chunk ke 1 sampai 75 data konten di dapat dari *cache router*, sedangkan pada chunk ke 76 sampai 115 data konten didapatkan dari *server*.

6. KESIMPULAN

Dalam penelitian ini, penulis melakukan desain dan implementasi *In-network Caching* pada *Content Centric Networking* menggunakan CCN-Lite dengan simulator OMNeT++ dengan tujuan untuk mengetahui bagaimana kinerja dari *in-network caching* pada CCN. Kesimpulan yang didapatkan adalah CCN-Lite mampu melakukan desain dan implementasi *in-network caching* pada CCN di lingkungan CCN-Lite dan simulator OMNeT++ dengan baik berdasarkan topologi yang telah dibuat. Akan tetapi, CCN-Lite memiliki beberapa kekurangan seperti pengaturan *cache slot* dan *cache size* yang selama diberi nilai selain 0, maka proses caching akan tetap berjalan dan pengaturan jalur pengiriman secara statik.

Pada latensi pengiriman, diperoleh hasil latensi ketika terjadi *caching* yaitu sebesar 10.69 ms. Nilai ini lebih rendah dari pada latensi yang diperoleh tanpa adanya *caching* yaitu 32.45 ms. Jadi ketika menggunakan *caching*, latensi yang didapat semakin rendah jika dibandingkan dengan tanpa menggunakan *caching*.

Nilai CHR tertinggi didapatkan pada protokol *file* CCNTLV sebesar 0.54 atau 54% dan nilai CHR dari protokol CCNB yaitu sebesar 0.45 atau 45% dari total *chunk* yang diminta. Hal ini menunjukkan bahwa protokol CCNTLV lebih baik dalam hal pelayanan konten yang diminta dari pada CCNB. Semakin tinggi nilai CHR yang didapat, semakin baik pula konten tersebut dilayani.

7. DAFTAR PUSTAKA

Ahir, D. D. & Kumbharkar, P. B., 2012. *Content Centric Networking and its Applications*, India: Jurnal of Global Research in Computer Science.

CCN-Lite, 2011. *CCN-Lite*. [Online] Tersedia di: http://www.ccn-lite.net [Diakses pada 1 Maret 2017].

Chiocchetti, R., Rossi, D. & Rossini, G., 2013. *ccnSim: an Highly Scalable CCN Simulator*. Paris, RESCOM.

Jacobson, V., 2009. *A Description of Content Centric Networking (CCN).* German: Parc Company.

Jacobson, V. et al., 2009. *VoCCN : Voice-over Content-Centric Networks.* Rome, Association for Computing Machinery.

Jacobson, V. et al., 2009. *Networking Named Content.* Rome, Association for Computing Machinery.

Kim, Y. & Yeom, I., 2013. Performance analysis of in-network caching for content-centric networking. *Computer Networks*, 57(7), pp. 2465-2482.

Mangili, M., Martignon, F. & Capone, A., 2015.

Performance analysis of Content-Centric and Content-Delivery network with evolving object popularity. Issue 7, pp. 1-19.

Saino, L., Psaras, I. & Pavlou, G., 2014. *Icarus - a caching simulator for Information Centric Networking*, Lisbon: Internasional Conference on Science and Technology.

Saucez, D., Barakat, C., Kalla, A. & Turletti, T., 2012. *Off-Path Caching in CCN*. France, CCNx Conference.

Shibuya, A., Hayamizu, Y. & Yamamoto, M., 2016. Cache Decision Policy for Breadcrumbs in CCN. *IEEE,* Issue 5, pp. 1-6.

Tortelli, M. et al., 2016. *ICN software tools: Survey and Cross-comparison*, Italy: SIMPAT.

Wang, L., 2013. Multi-Objective In-Network Caching Strategies. *IEEE*.

Wang, S. et al., 2011. *Could In-Network Caching Benefit Information-Centric Networking?*. Bangkok, Association for Computing Machinery.

Wu, H., Li, J. & Zhi, J., 2015. *Could End System Caching and Cooperation Replace In-Network Caching in CCN?*. London, Association for Computer Machinery.

Xu, Y. et al., 2016. Design and evaluation of coordinated in-network caching model for content centric networking. *Computer Networks*, 110(7), pp. 266-283.