Jurnal Ilmu Komputer dan Desain Komunikasi Visual Volume 4, Nomor 2, Desember 2019
ISSN (Cetak) : 2541-4550 ISSN (Online) : 2541-4585

Sistem Monitoring Jaringan *Load balancing* Dengan Metode *Equal Cost Multipath* (ECMP) Menggunakan Media Telegram

[1]Mochammad Irfan Oktavianto dan [2]Yanuar Risah Prayogi

[1]Program Studi Teknik Informatika, Fakultas Ilmu Komputer,
Universitas Nahdlatul Ulama Sidoarjo
Jln. Mongonsidi Kav. DPR Sidoklumpuk Sidoarjo
[2]Departemen Teknik Informatika dan Komputer,
Politeknik Elektronika Negeri Surabaya
Jalan Raya ITS Sukolilo Surabaya
[1]irfan_kazu@gmail.com

Abstrak

Penelitian ini membahas tentang sistem monitoring jaringan Load balancing Equal Cost Multi Path (ECMP) menggunakan bot telegram, load balance pada jaringan merupakan teknik penggabungan lebih dari 1 server untuk membagi beban sama rata. Pemanfaatan bot telegram sebagai sistem monitor, mempermudah mengetahui secara realtime ketika server mengalami gangguan. Dengan melakukan pengujian QOS (Quality Of Service) pada load balancing ECMP menggunakan 5 Client dengan 3 server, ketika salah satu server down maka router akan mengakses bot menggunakan server yang masih aktif dan mengirimkan pesan kepada penerima pesan dari bot. dari penelitian tersebut, diketahui bahwa sistem monitoring dapat berjalan 75% pada load balancing ECMP.

Kata kunci— *Jaringan, Load balancing ECMP, Sistem Monitoring, Bot-Telegram*

Abstract

This riset discusses the Equal Cost Multi Path (ECMP) monitoring system that balances network loads using telegram bots, load balancing on networks is a technique of combining more than 1 server to share the load evenly. The use of telegram bot as a monitoring system, makes it easier to find out in real time when the server has a problem. By testing QOS (Quality of Service) on ECMP load balancing using 5 clients with 3 servers, when one server goes down the router will access the bot using the active server and send messages to the recipient of the message from the bot. From this research, it is known that 75% the monitoring system can run on ECMP load balancing.

Keywords— *Network, Load balancing ECMP, Monitoring System, Telegram Bot*

1. PENDAHULUAN

Jaringan komputer menjadi bagian penting dalam kehidupan sehari-hari sebagai salah satu media komunikasi maupun informasi dalam bisnis atau dalam hal privasi, untuk itu diperlukan perangkat yang saling terhubung agar dapat melakukan pertukaran informasi bisnis maupun privasi, dalam hal tersebut membutuhkan banyak komputer dari seluruh dunia yang saling terhubung satu sama lain (*Internet*). Menurut (Supriyanto, 2013) Internet (*interconnection-networking*) adalah seluruh jaringan komputer yang saling terhubung menggunakan standar sistem global *Transmisson Control*

Protocol/Internet Protocol Suite (TCP/IP) sebagai protokol pertukaran paket (*packet switching communication protocol)* untuk melayani miliaran pengguna di seluruh dunia. Agar dapat memperoleh akses dalam ber-internet maka dibutuhkan sebuah pusat (*server*) yang dapat menghubungkan komputer dari seluruh dunia, ISP (*Internet Service Provider*) merupakan penyedia layanan internet sebagai akses penghubung komputer di seluruh dunia atau dikenal sebagai *server*. Router adalah alat yang berfungsi sebagai penghubung antara dua atau lebih jaringan untuk meneruskan data dari satu ke jaringan lainnya. *Router* yang terhubung dengan *server* ISP akan menjadi perantara bagi *client*, dalam hal ini router bertindak sebagai *server* lokal yang menyediakan akses pada client agar terhubung dengan *server* ISP dan mendapatkan akses internet. Namun ketika *server* ISP yang bersangkutan mengalami gangguan atau layanan yang tersedia terbatas, maka akan berpengaruh dalam kelancaran dalam internet. Dalam hal ini memerlukan lebih dari satu *server* ISP sebagai cadangan jaringan ketika salah satu *server* ISP mengalami gangguan atau layanan yang kurang, dengan menggabungkan dua lebih dari satu *server* ISP akan mengurangi gangguan. Maka dari itu menggabungkan lebih dari satu *server* ISP dengan cara menerapkan *Load balancing* merupakan salah satu solusi yang dapat digunakan dalam mengatasi masalah tersebut.

Load balancing sendiri merupakan teknik pembagian beban pada lebih dari satu jaringan secara merata dengan membagi jalur koneksi yang tersedia. Dengan adanya *load balancing* beban yang ditanggung oleh *server* akan terbagi ke *server* yang lain dengan pembagian yang sesuai kapasitas *server* masing-masing. *Load balancing* memiliki beberapa metode yang dapat digunakan dalam permasalahan pada penggabungan *server*, salah satunya dengan metode *Equal Cost Multi Path* (ECMP). Menurut (Teknologi, Citraweb Solusi, 2019) ECMP merupakan metode paling sederhana dalam teknik *load balancing* karena metode ECMP cocok digunakan pada jaringan dengan tingkat kompleksitas yang tidak terlalu tinggi. Namun, beberapa kekurangan pada jaringan adalah terputusnya *server* secara tiba-tiba atau tidak tahu kapan secara pasti akan terputus. Dari hal tersebut membutuhkan *monitoring* secara *real-time* untuk mengetahui ketika *server* terputus. Hal tersebut bisa diatasi dengan menggunakan beberapa aplikasi android yang bersifat *open source* seperti Telegram, Telegram merupakan aplikasi *chat* yang bersifat *open source* yang dapat diaplikasikan pada perangkat lain dengan membuat bot dan memanfaatkan API dari bot tersebut maka akan lebih mudah dalam mengirim informasi jaringan secara *real-time*.

Dalam hal ini penulis membuat penelitian dengan judul Sistem *Monitoring* Jaringan *Load balancing* dengan Metode *Equal Cost Multi Path* (ECMP) Menggunakan Media Telegram, yang dapat digunakan sebagai solusi permasalahan dan *monitoring real-time* Jaringan ketika terjadi adanya gangguan jaringan.

2. METODE PENELITIAN

2.1 Tahap Penelitian

Pengumpulan data pada penelitian adalah proses tahapan yang sangat penting dilakukan untuk mendapatkan data yang sesuai, dengan mendapatkan data yang sesuai maka riset akan sesuai dengan perumusan masalah yang ditentukan. Maka dari itu penulis mengumpulkan data dengan melakukan studi pustaka dan studi penelitian sejenis. Dalam tahapan ini, penulis mempelajari dasar dan teori yang terkaitan dengan penelitian yang akan dilakukan agar dapat mendukung pemecahan masalah pada penelitian tersebut.

Pencarian referensi dilakukan di perpustakaan maupun mencari referensi secara *online* melalui internet. Selain itu, penulis juga mempelajari jurnal dari penelitian yang pernah ada sebagai pembanding terhadap penelitian yang dikerjakan. Dari hasil referensi dan penelitian yang sudah didapatkan, maka penulis melakukan penelitian sistem *monitoring load balancing* dengan metode ECMP dengan alasan metode tersebut cocok digunakan pada jaringan dengan tingkat kompleksitas yang tidak terlalu tinggi.

2.2 Data Set Jaringan

Data set pada penelitian ini berisikan 3 jaringan server yang meliputi WiFi Univ. NU Sidoarjo, dan dua server yang diperoleh dari Sim Card XL. Dan Three yang akan digunakan sebagai bahan uji pada penelitian Sistem Monitoring Jaringan Load balancing dengan Metode Equal Cost Multi Path (ECMP) Menggunakan Media Telegram. Berikut tabel 3,1 data set yang akan digunakan pada penelitian ini:

Tabel 1. Tabel Jaringan

No.	Nama Jaringan	Jenis Jaringan
1	Univ. NU Sidoarjo	Wi-Fi
2	XL	SIM Card
3	Three	SIM Card

2.3 Analisis dan Kebutuhan

Analisis dan kebutuhan yang akan digunakan pada penelitian ini dibagi menjadi 2 yaitu Perangkat Lunak, dan Perangkat Keras yang nantinya akan digunakan sebagai bahan utama pada penelitian ini.

Tabel 2. Perangkat Lunak (*Software*)

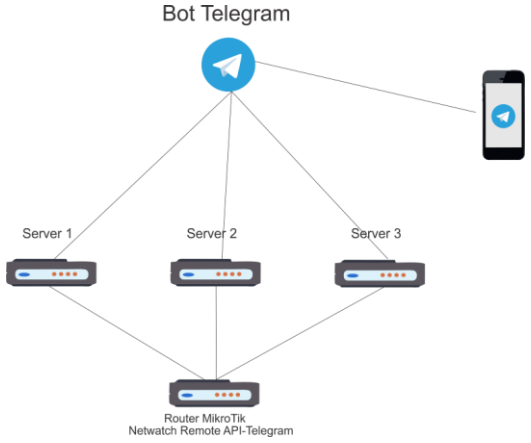
No.	Software	Keterangan
1	Winbox	*Software* untuk melakukan *remote* GUI ke *router* MikroTik
2	Windows 10 OS	Sebagai sistem operasi *remote server*
3	Windows XP OS	Sebagai sistem operasi *client*
4	Virtual Box	*Software* untuk menjalankan sistem operasi *client*
5	Chrome	*Software* untuk melakukan konfigurasi *router*
6	Wireshark	Sebagai monitor pengujian jaringan
7	Android OS	Sebagai Sistem operasi penerima pesan Telegram
8	Telegram	Sebagai Aplikasi penerima pesan dari Bot-Telegram yang dibuat

Tabel 3. Perangkat Keras (*Hardware*)

No.	Perangkat	Jumlah	Spesifikasi
1	MikroTik RB2011	1	CPU : AR9344
2	Laptop	1	Core i3 RAM 6 GB HDD 1TB
3	ISP	3	Univ. NU Sidoarjo SIM *Card* XL SIM *Card three*
4	*Router* Wifi	1	1 *router support* WDS mode
5	*Smartphone*	2	Android OS
6	OTG (*On The Go*)	1	*Support* android
7	Kabel Data	1	*Support* android
8	USB to LAN	1	*Support* windows 10

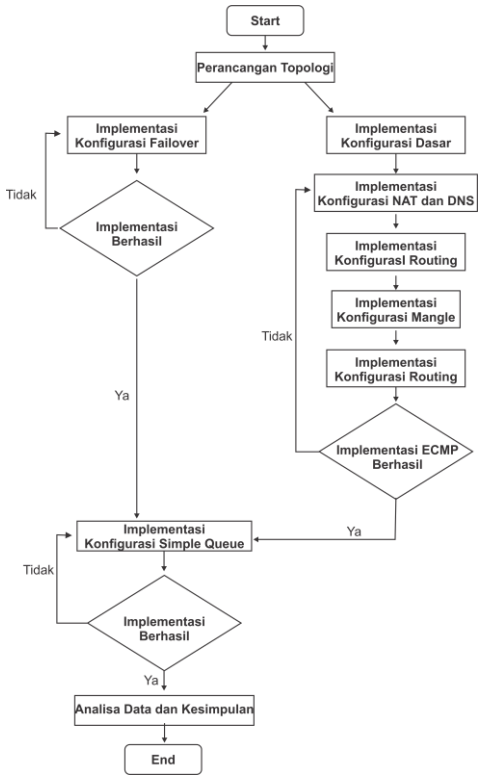
2.4 Desain Sistem

Desain sistem pada penelitian ini menggunakan 3 jaringan yang berbeda dengan menerapkan metode *load balancing* ECMP yang diterapkan pada Mikrotik *routerboard* dengan memanfaatkan api dari pihak telegram yang akan diterapkan pada fitur *netwatch* MikroTik, aplikasi Telegram sebagai penerima pesan ketika server mengalami gangguan. Pada gambar 1 menggunakan 3 *server* jaringan yang akan digunakan sebagai uji coba pada Sistem Monitoring menggunakan *Load balancing* ECMP dengan media Telegram.



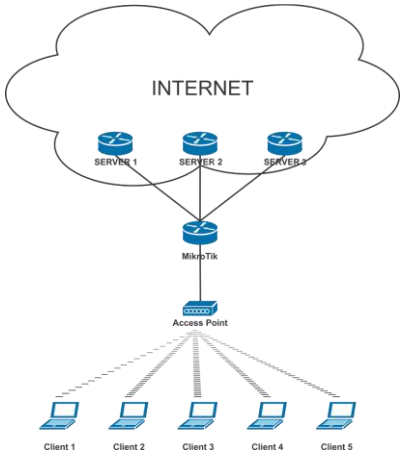
Gambar 1. Desain Sistem Telegram

Pada rancangan penelitian ini dibagi menjadi 2 yaitu *Load balancing* ECMP yang membahas tentang topologi dan implementasi yang akan digunakan pada penelitian ,dan perancangan Bot Telegram yang akan membahas implementasi Bot Telegram pada Mikrotik. Berikut adalah alur perancangan penelitian pada gambar 2 *Flowchart* Perancangan Desain Sistem merupakan langkah-langkah yang akan dilakukan selama penelitian dan digambarkan langkah-langkah secara umum mulai dari implementasi *load balancing*, implementasi bot telegram, sampai dengan analisis data dan kesimpulan.



Gambar 2. *Flowchart* Perancangan Desain Sistem

2.5 *Load balancing* ECMP

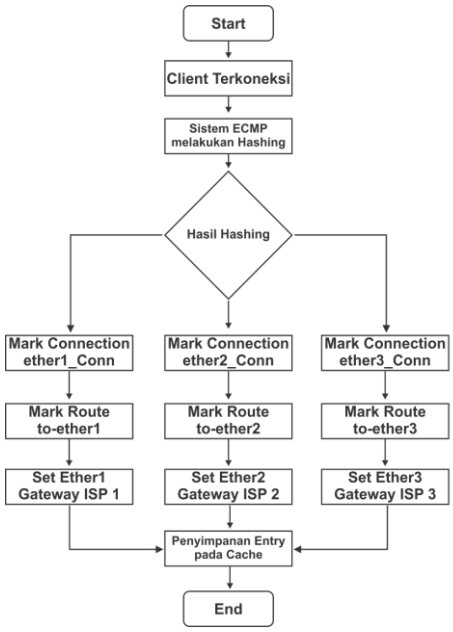


Gambar 3. Topologi *Load balancing* ECMP

Load Balancing ECMP dibagi menjadi 2 yaitu Topologi, dan Konfigurasi. Pada Gambar 3, topologi jaringan menggunakan 3 server (pada penerapan sebenarnya menggunakan 1 komputer/Laptop yang akan menghubungkan server Univ. NU Sidoarjo dan menghubungkan kembali ke *router* MiktoTik menggunakan USB to LAN sebagai server 1, 1 *smartphone* menggunakan fitur USB *tethering* yang akan disambungkan dengan Mikrotik melalui OTG (*On The Go*) dan Kabel data sebagai server 2. Dan 1 *smartphone* yang menggunakan fitur *tethering* dan menggunakan *router* dengan wds

mode yang akan menghubungkan antara *router* dengan *tethering smartphone* tersebut dan menghubungkan ke *router* mikrotik melalui LAN sebagai server 3. Port 3 pada mikrotik akan dihubungkan melalui kabel LAN ke komputer yang sudah terpasang *virtual box* dan di hubungkan dengan 5 Virtual *Client*. Pada penerapan topologi yang akan digunakan membutuhkan rancangan *logic* yang berisikan tentang keterangan konfigurasi ip *address* yang nantinya akan diterapkan pada konfigurasi *Load balancing* ECMP.

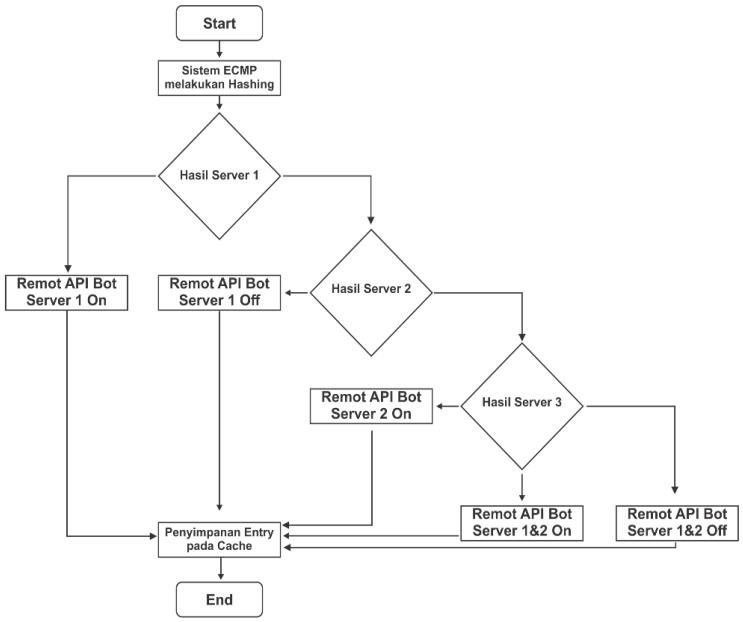
Rancangan sistem *load balancing* memerlukan beberapa tahapan seperti konfigurasi dasar, konfigurasi NAT, konfigurasi *mangle*, *routing*, pembuatan *failover*, dan penerapan *simple queue*, dan konfigurasi pada *router* AP.



Gambar 1 Sistem ECMP menggunakan *failover*

2.6 Rancangan Bot-Telegram

Pada pembuatan bot Telegram memerlukan pihak yang dapat memberikan akun bot yang akan digunakan, Telegram menyediakan akun @BotFather yang dapat membuat akun bot dengan syarat penambahan kata “bot” pada akun yang akan dibuat. Menambahkan akun @Botfather sebagai teman, lalu memulai pembuatan bot dengan perintah /start. Setelah mengirimkan perintah *start* kirimkan perintah /*newbot*, kirimkan nama yang sudah dibuat dengan akhiran kata “bot”. penulis menggunakan nama “mlbecmp_bot” sebagai akun bot, Setelah pembuatan bot melalui akun @BotFather maka akan mendapatkan API yang dapat digunakan sebagai akses bot tersebut. Untuk pembuatan penerima pesan dari bot dapat mengambil chat id melalui akun @get_id atau yang lainya. Implementasi Bot Telegram pada MikroTik memanfaatkan fitur *netwatch* yang sudah disediakan oleh mikrotik. Pada implementasi API Bot Telegram di MikroTik dengan cara masuk ke bagian *tool netwatch* dan menambahkan script dengan pengaturan tertentu seperti target host, Interval, Timeout. Lalu dengan menambahkan *tool fetch* url=”web API Telegram”keep-result=no pada kondisi Up dan *Down* pada target. Pada gambar 5. adalah gambaran sistem yang akan berjalan pada bot-telegram yang telah diimplementasikan pada *load balancing* ECMP.



Gambar 5. Sistem Bot Telegram pada *Load balancing*

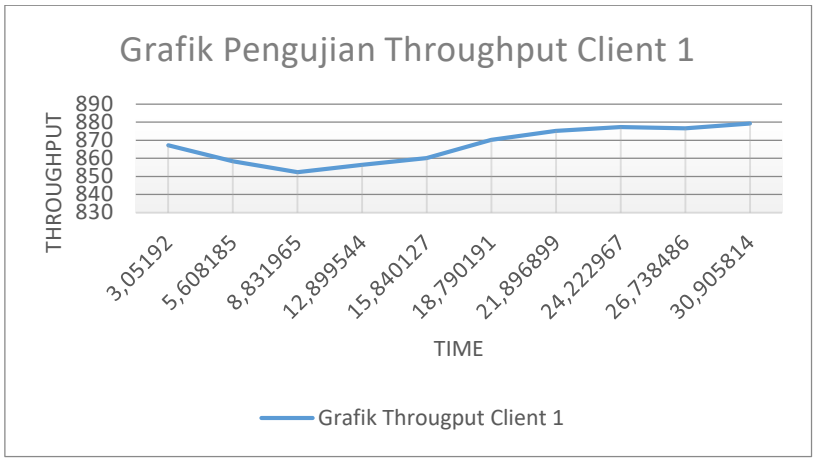
2.7 Rancangan Uji Coba

Skenario pada uji coba bertujuan mengimplementasi Sistem *Monitoring* menggunakan API Telegram dan mendapatkan notifikasi ketika salah satu dari *server* pada pengaturan *Load balancing* yang menggunakan Metode ECMP mengalami gangguan. Pengujian dilakukan dengan cara menyambungkan 5 *client Virtual* pada jaringan *Load balancing* ECMP. Uji coba yang pertama adalah QOS (*Quality Of Service*) yang bertujuan untuk mengetahui mekanisme jaringan yang memungkinkan aplikasi atau layanan dapat beroprasi sesuai yang diharapkan, dengan beberapa parameter seperti *throughput, packet loss, delay, dan jitter*, pengujian kedua adalah pengujian *failover* yang bertujuan untuk mengetahui perpindahan otomatis yang diterapkan pada metode *load balancing* ECMP dan pada pengujian yang ketiga adalah pengujian yang dilakukan pada Bot-Telegram yang bertujuan untuk mengetahui *system monitor* pada jaringan *load balance* bekerja atau tidak.

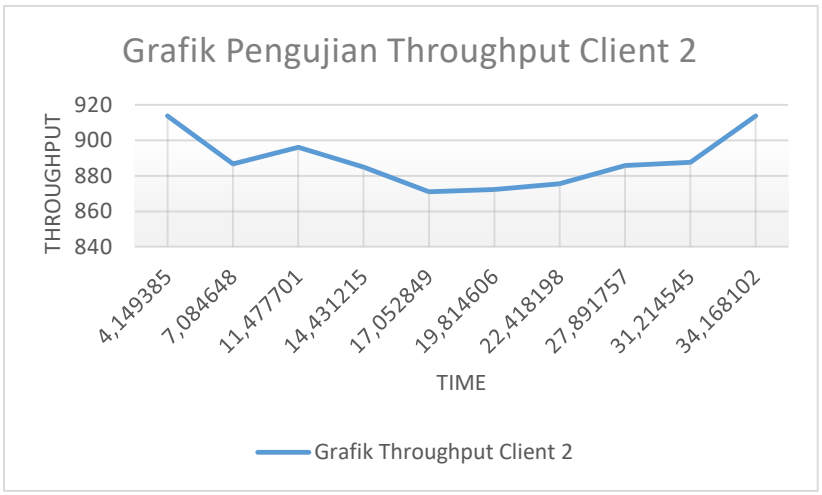
3. HASIL DAN PEMBAHASAN

3.1.Pengujian *Throughput*

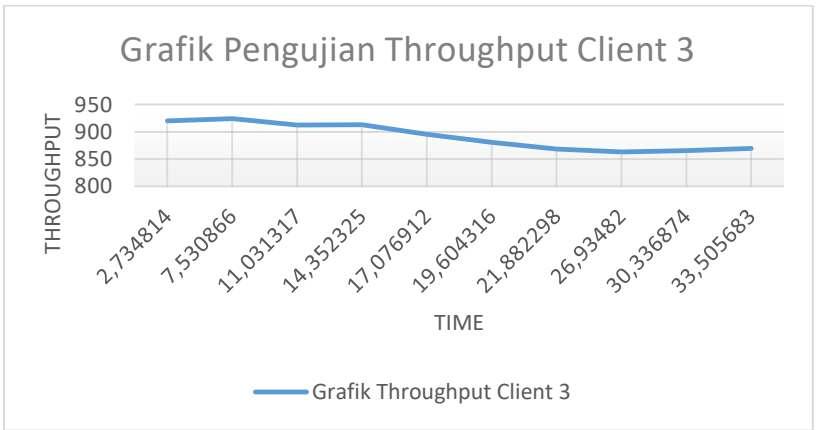
Pada pengujian *Througphut*, hasil yang didapatkan dari masing-masing *virtual client* adalah:



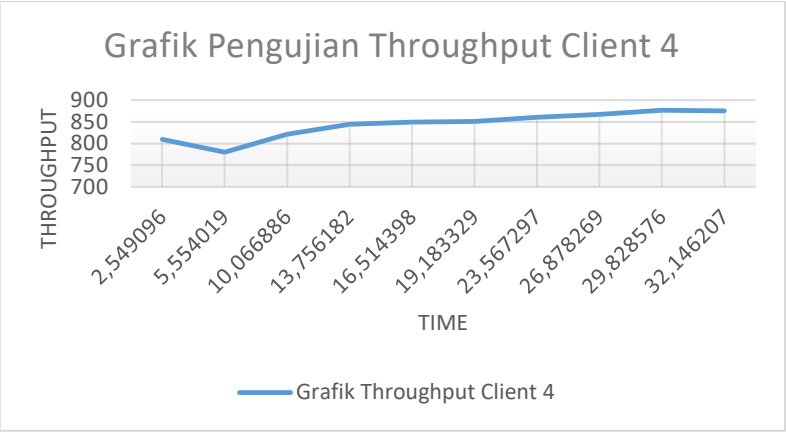
Gambar 6. Grafik Pengujian *Throughput Client* 1



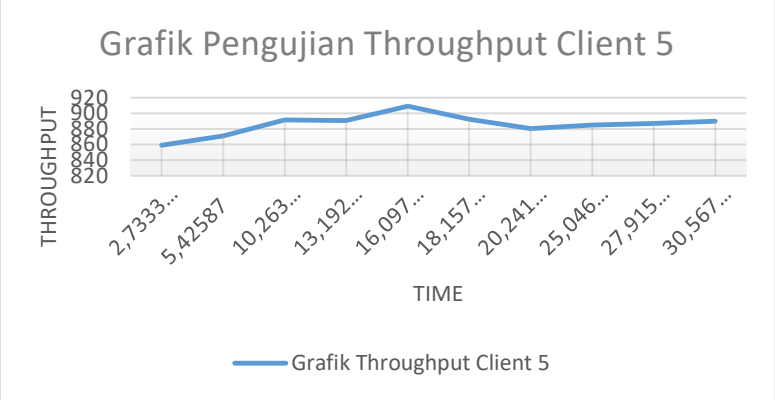
Gambar 7. Grafik Pengujian *Throughput Client* 2



Gambar 8. Grafik Pengujian *Throughput Client* 3



Gambar 9. Grafik Pengujian *Throughput Client* 4



Gambar 10. Grafik Pengujian *Throughput Client* 1

Tabel 4. Hasil Pengujian *Througput*

Nama *Client*	*Byte* dikirim	Waktu	*Throughput* Bytes/s	*Throughput* Bits/s	Rata-rata
Client 1	8793516	30,905814	284526,2707	2276210,166	2186582,76
Client 2	8854297	34,168102	259139,2697	2073114,158	
Client 3	8694558	33,505683	259495,0236	2075960,189	
Client 4	8755069	32,146207	272351,5406	2178812,325	
Client 5	8898251	30,567455	291102,1215	2328816,972	

Dari hasil dapat disimpulkan bahwa setiap *client* mendapatkan hasil yang hampir sama dengan waktu perbandingan *throughput* kurang dari 5 detik.

3.2 Pengujian *Packet Loss*

Pada pengujian *Packet Loss*, hasil yang didapatkan dari masing-masing *virtual client* adalah:

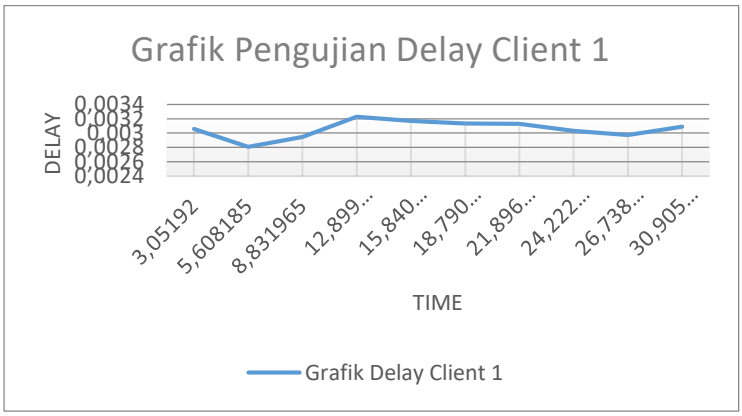
Tabel 5. Tabel Hasil *Packet Loss*

Nama *Client*	Packet dikirim	Packet diterima	Hasil	Rata-rata
Client 1	10000	10000	0,00%	0,01%
Client 2	10000	10000	0,00%	
Client 3	10000	9998	0,02%	
Client 4	10000	9997	0,03%	
Client 5	10000	10000	0,00%	

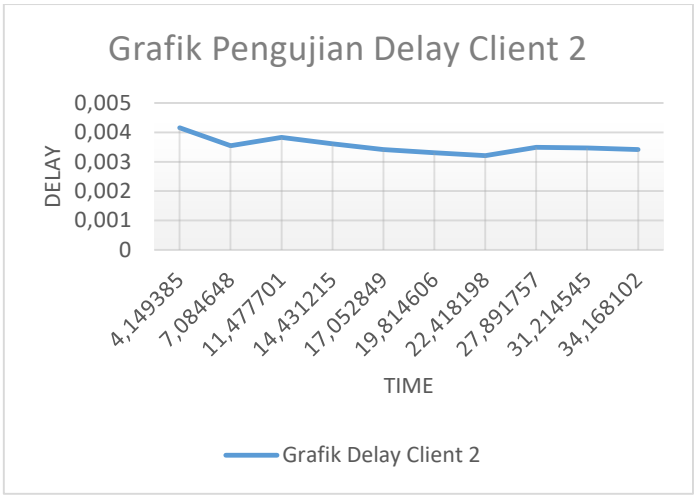
Dari hasil yang didapatkan bahwa kemungkinan terjadi *packet loss* sangat minim. Dikarenakan beberapa penumpukan data dapat atasi dengan adanya *load balance* pada metode ECMP. Namun mengapa masih terjadi *packet loss*?. Karena adanya penumpukan data *packet* yang dikirim melalui beberapa server dengan pembagian merata pada setiap *server* yang mengakibatkan adanya *packet loss*.

3.3 *Delay*

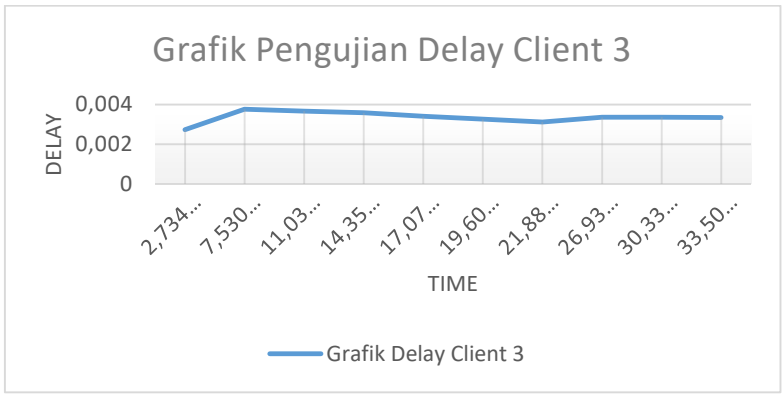
Pada pengujian Delay, hasil yang didapatkan dari masing-masing *virtual client* adalah:



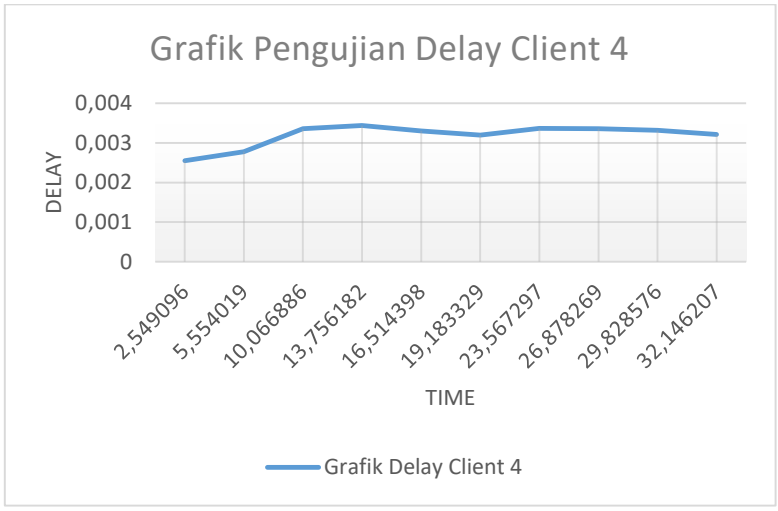
Gambar 11. Grafik Pengujian *Delay Client* 1



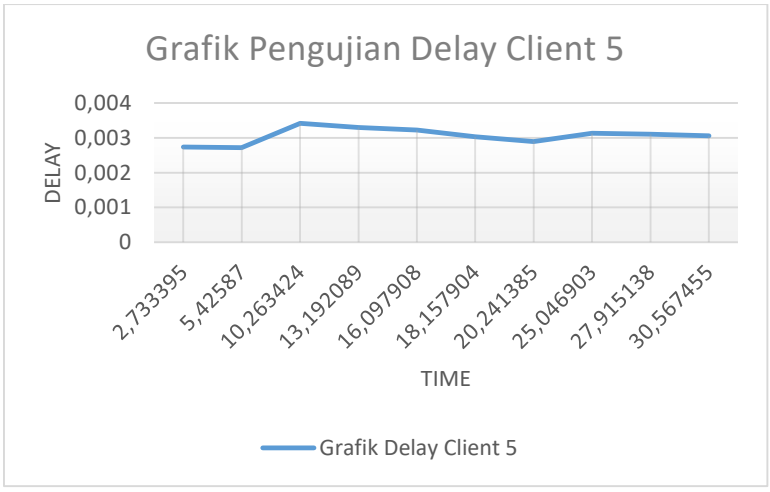
Gambar 12. Grafik Pengujian *Delay Client* 2



Gambar 13. Grafik Pengujian *Delay Client* 3



Gambar 14. Grafik Pengujian *Delay Client* 4



Gambar 15. Grafik Pengujian *Delay Client* 5

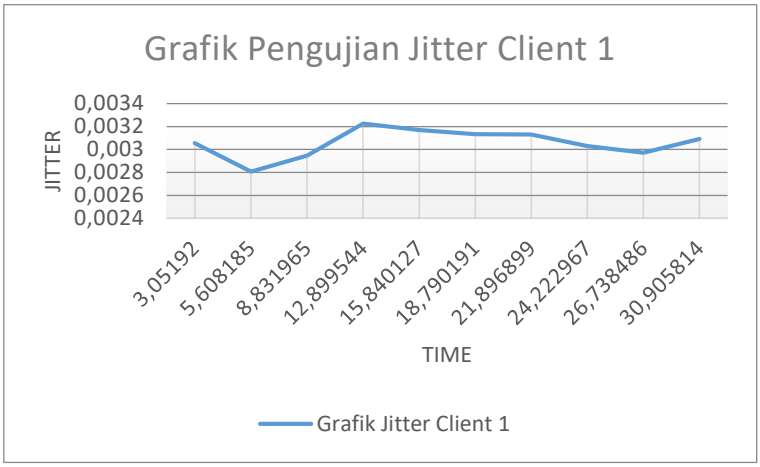
Tabel 6. Tabel Hasil Pengujian *Delay*

Nama *Client*	*Delay*	Rata-rata
Client 1	0,00309089	0,002644378
Client 2	0,003236	
Client 3	0,000115	
Client 4	0,000423	
Client 5	0,006357	

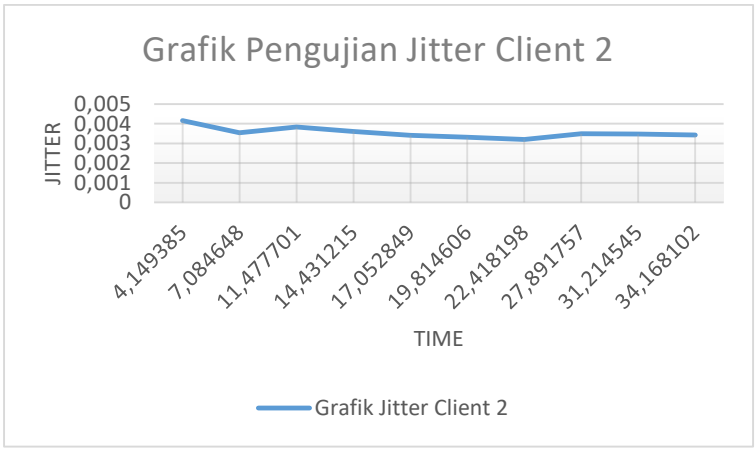
Dari hasil pengujian *delay* maka dapat disimpulkan bahwa *delay* yang terjadi pada masing-masing *client* memiliki perbandingan yang cukup singkat.

3.4 *Jitter*

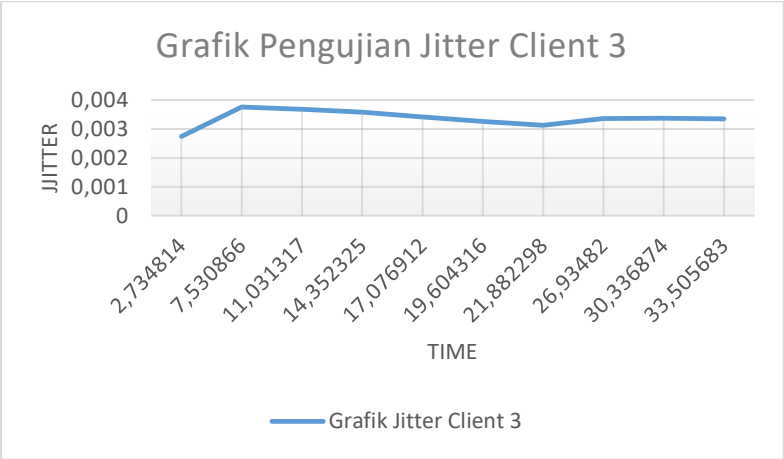
Pada pengujian Jitter, hasil yang didapatkan dari masing-masing virtual *client* adalah:



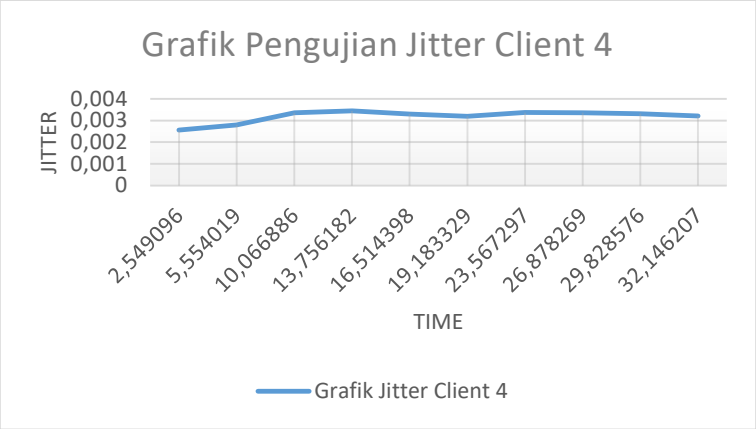
Gambar 16. Grafik *Jitter Client* 1



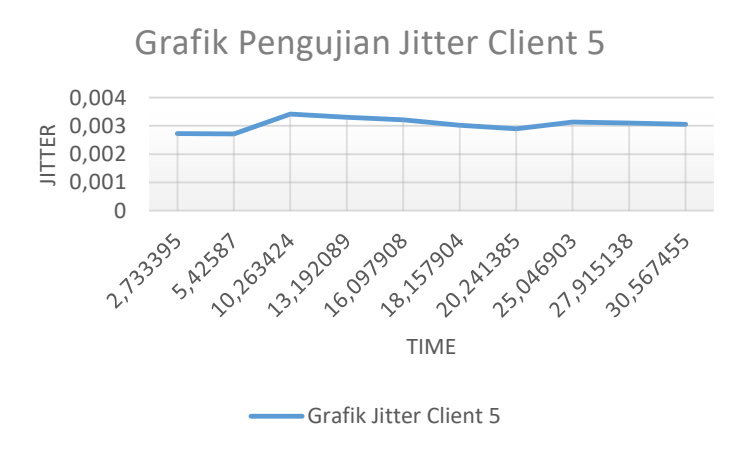
Gambar 17. Grafik *Jitter Client* 2



Gambar 18. Grafik *Jitter Client* 3



Gambar 19. Grafik *Jitter Client* 4



Gambar 20. Grafik *Jitter Client* 5

Tabel 7. Hasil Pengujian *Jitter*

Nama *Client*	*Jitter*	Rata -rata
Client 1	0,003091181	0,003899836

Nama *Client*	*Jitter*	Rata -rata
Client 2	0,003758	
Client 3	0,001059	
Client 4	0,004979	
Client 5	0,006612	

Dari hasil pengujian Jitter pada setiap *client* maka dapat disimpulkan bahwa masing-masing *client* memiliki perbandingan jitter yang cukup pendek.

3.5 Pengujian *Failover*

Pada pengujian *Failover* dari metode *load balancing* ECMP dapat berjalan dengan baik pada pengujian *download* dan *streaming* dari 5 *client* sekaligus.

3.6 Pengujian Bot-Telegram

Pada pengujian Bot-Telegram, hasil yang didapatkan dengan penerapan *load balancing* ECMP dapat berjalan saat kondisi salah satu *server* dalam keadaan *down*, namun pengiriman pesan agak lambat saat *server* yang *down* kembali normal kembali. Dikarenakan adanya pengiriman data dari *server* dua dan tiga yang membuat *server* yang *down* tidak diakses kembali kecuali saat *Client* melakukan pengiriman data atau pengunduhan data yang melebihi *bandwidth* pada *server* yang masih normal. Namun dengan adanya pengujian ini dapat dipastikan bahwa telegram dapat menjadi salah satu sarana sistem *monitoring* jaringan ketika saat keadaan darurat. Dasi hasil yang didapat dari seluruh uji coba pada *load balancing* ECMP dan Sistem *Monitoring* menggunakan Telegram dapat disimpulkan pada tabel berikut.

Tabel 8. Uji *Down Server*

Kondisi 1 Uji *Down Server*						
Server 1	Server 2	Server 3	Prediksi Pesan	Interval detik	Hasil Uji	Persentase
Up	Up	Up	ISP-1 Normal	1	Benar	12,5%
			ISP-2 Normal	1	Benar	12,5%
			ISP-3 Normal	1	Benar	12,5%
Down	Up	Up	ISP-1 Mati	1	Benar	12,5%
Down	Down	Up	ISP-2 Mati	1	Benar	12,5%

Tabel 9. Uji *Up Server*

Kondisi 2 Uji Up Server						
Server 1	Server 2	Server 3	Prediksi Pesan	Interval detik	Hasil Uji	Persentase
Up	Up	Up	ISP-1 Normal	1+	Salah	0,0%
Down	Up	Up	ISP-1 Mati	1+	Salah	0,0%
			ISP-2 Normal	1	Benar	12,5%
HASIL UJI COBA						
75,0%						

Pada tabel 8 dan tabel 9, peneliti menggunakan perhitungan yang telah diterapkan pada *netwatch* mikrotik dengan tingkat keberhasilan 12,5 persen dari setiap pesan yang dikirim ke telegram dengan interval kurang dari sama dengan 1 detik pada saat pengujian QOS dan Monitor pada kondisi *server Up* maupun *Down*.

4. KESIMPULAN

Berdasarkan hasil uji *balance* dengan cara *download* dan *streaming dengan* menggunakan 5 *client* secara bersamaan, hasil dari pengujian QOS dan sistem monitoring dengan memanfaatkan media telegram maka dapat disimpulkan bahwa penelitian yang dilakukan memiliki tingkat keberhasilan sebesar 75 % dikarenakan masih adanya beberapa kendala ketika *server* mengalami *down*, maka *server* lain akan melakukan penggantian *server* dan pada sistem monitor telegram tidak dapat berjalan secara langsung. Dan saat *server* yang *down* kembali normal, fitur *netwatch* tidak dapat secara langsung melakukan pemeriksaan pada *gateway* yang hidup kembali tanpa adanya akses *server* dari salah satu *client*. Hal tersebut mengakibatkan terjadinya keterlambatan akses bot telegram yang seharusnya digunakan untuk pengiriman pesan.

5. SARAN

Berdasarkan kesimpulan di atas, peneliti mengajukan beberapa saran untuk pengembangan dari penelitian yang telah dilakukan, antara lain:

1. Penggabungan beberapa metode *load balance* jaringan untuk mendapatkan hasil yang lebih baik.
2. Menggunakan ISP atau *server* berbeda dalam dengan *bandwidth* yang sama agar mendapatkan hasil yang lebih baik tanpa adanya perbedaan kecepatan saat terjadi pengalihan *server*.
3. Memfungsikan telegram bukan hanya sebagai sistem monitor namun sebagai *remote* akses untuk perbaikan jaringan pada mikrotik.

UCAPAN TERIMA KASIH

Peneliti mengucapkan terima kasih kepada keluarga tercinta yang selalu memberikan dukungan dan semua pihak atas bantuan dan kerjasamanya khususnya program studi teknik informatika Universitas Nahdlatul Ulama Sidoarjo.

DAFTAR PUSTAKA

Dewi Risanty, R., & Sopiyan, A. (2017). Pembuatan Aplikasi Kuesioner Evaluasi Belajar Mengajar Menggunakan Bot Telegram Pada Fakultas Teknik Universitas Muhammadiyah Jakarta (FT-UMJ) Dengan Metode Polling. *Universitas Muhammadiya Jakarta*, (November), 1–2.

Made Santo Gitakarma, S.T., MT.; Ketut Udy Ariawan, S.T., M. (2014). Jaringan Komputer. *Jaringan Komputer*, *1*, 1–233.

Oktivasari, P., & Sanjaya, R. (2018). Implementasi Sistem Load Balancing Dua ISP Menggunakan Mikrotik dengan Metode Per Connection Classfier. *Multinetics*, p. 33. https://doi.org/10.32722/multinetics.vol1.no.2.2015.pp.33-37

Rahayu, W. (2011). Internet dan Sejarah Perkembangannnya. *Interconnection Networking* .

Supriyanto. (2013). *Jaringan Dasar Jaringan Dasar*. *Jaringan Dasar*.

Teknologi, Citraweb Solusi. (2019). Load Balance Metode ECMP. http://www.mikrotik.co.id/artikel_lihat.php?id=76.

Towidjojo, R. (2019). *Mikrotik KungFu Kitab 1 (Edisi Revisi 2019).* Yogyakarta: Jasakom.