e-ISSN 2540-7902 dan p-ISSN 2541-366X

IMPLEMENTASI MANAJEMEN BANDWITH PADA FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFRMASI UNVERSITAS MULAWARMAN SAMARINDA MENGGUNAKAN METODE *HIERARCHICAL TOKEN BUCKET (HTB)*

Muhammad Hidayat[1*], Edy Budiman[2], Rudiman[3]

Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi,, Universitas Mulawarman
Jalan Barong Tongkok No. 6 Kampus Gunung Kelua Samarinda, Kalimantan Timur
E-mail : hidayatchipunk29@gmail.com, edy.budiman@gmail.com, rudiman.made@gmail.com

ABSTRAK

Aplikasi yang membutuhkan *delay* yang minim seperti apilkasi multimedia, dan aplikasi yang akan merebut *bandwidth* sebesar-besarnya seperti aplikasi pada *ftp,* serta aplikasi yang hanya memakan *bandwidth* sangat kecil seperti *icmp* dan web statis, akan membuat jumlah *bandwidth* yang ada tidak mendapatkan jatahnya sehingga performansi jaringan akan terasa menurun. Mengatasi hal tersebut, perlu diimplementasikan suatu teknik *QoS* yang bernama *Hierarchical Token Bucket (HTB)* yang merupakan suatu tawaran solusi untuk manajemen bandwidth pada jaringan yang digunakan oleh pihak ICT Universitas Mulawarman dalam pembagian *bandwidth* di Fakultas Ilmu Komputer dan Teknologi Informasi.

Kata Kunci : Manajemen Bandwidth, *Hierarchical Token Bucket (HTB)*, *Bandwidth, IPREF.*

1. PENDAHULUAN

1.1. Latar Belakang

Layanan komunikasi data telah menjadi sangatlah penting dalam kehidupan sehari-hari. Hampir di setiap bidang kehidupan telah mengadopsi layanan ini. Layanan inipun tidak hanya digunakan secara individual tetapi juga digunakan secara massal. Banyak sekali organisasi atau lembaga yang menggunakan akses internetnya secara massal. Lembaga pendidikan, perkantoran, warnet, dan masih banyak lagi lembaga-lembaga lainnya menggunakan akses *internet* secara massal. Penggunaan akses *internet* secara massal ini akan mengakibatkan turunnya performansi jaringan seiring dengan peningkatan jumlah pengguna. Apalagi jika *bandwidth* yang ada tidak dikelola sebaik mungkin.

Rencana jangka panjang Teknologi Informasi Universitas Mulawarman yang tertuang didalam *blueprint* 2010-2014, Universitas Mulawarman diharapkan mampu memberikan arah dan langkah dalam mencapai cita-cita dalam teknologi informasi secara utuh yaitu mencapai terwujudnya Universitas Mulawarman berstandar Internasional dengan pemanfaatan teknologi informasi yang tepat terstruktur, terukur dan tepat sasaran. Universitas Mulawarman saat ini menyediakan *bandwidth* sebesar 165 MB yang dibagi 65 MB untuk IP Transit dan 100 MB untuk IP Domestik yang disediakan oleh ISP PT. Telkom menggunakan produk *ASTI Net*. *Bandwidth* tersebut didistribusikan melalui *switch core* Universitas Mulawarman yang berada pada Upt. *Distance Learning* lantai 2, ke seluruh instansi atau lembaga yang ada di Universitas Mulawarman (R Aulia & Haviluddin, 2016).

Quality of Service (QoS) memegang peranan yang sangat penting dalam hal ini. Linux sebagai suatu sistem operasi yang bersifat *open* dan *free,* telah menawarkan berbagai teknik QoS untuk memfasilitasi proses manajemen *bandwidth* pada suatu jaringan. Salah satunya adalah dengan menggunakan teknik *QoS Hierarchical Token Bucket (HTB)*, yang menjamin para pengguna jaringan mendapatkan *bandwidth* yang sesuai dengan yang telah didefinisikan, dan juga terdapat fungsi pembagian *bandwidth* yang adil di antara para pengguna jaringan sehingga performansi jaringan tetap dapat terjaga

Aplikasi-aplikasi di internet sangatlah banyak dan beragam dengan masing-masing sifatnya. Ada yang membutuhkan *delay* yang minim seperti apilkasi multimedia, ada yang akan merebut *bandwidth* sebesar-besarnya seperti aplikasi pada *ftp,* dan ada pula yang hanya memakan *bandwidth* sangat kecil seperti *icmp* dan web statis. Apabila semua aplikasi ini tidak dikontrol, maka satu aplikasi dapat memakan jumlah *bandwidth* yang ada dan aplikasi yang lain tidak mendapatkan jatahnya sehingga performansi jaringan akan terasa menurun. Apalagi jika ini terjadi pada suatu jaringan yang berisi banyak sekali node. Maka turunnya performansi jaringan akan sangat terasa sekali.

Berdasarkan latar belakang yang diuraikan sebelumnya, permasalahan yang dapat dirumuskan adalah "Bagaimana mengimplementasikan suatu teknik *QoS* yang bernama *Hierarchical Token Bucket (HTB)* yang merupakan suatu tawaran solusi untuk manajemen *bandwidth* pada jaringan yang digunakan oleh pihak ICT Universitas Mulawarman

*Corresponding Author

dalam pembagian *bandwidth* di Fakultas Ilmu Komputer dan Teknologi Informasi ?"

Agar mendapatkan hasil penelitian seperti yang diharapkan, maka permasalahan dalam penelitian ini akan dibatasi sebagai berikut :

Parameter yang digunakan untuk mengukur kualitas layanan jaringan *Internet* adalah *performance* yang meliputi *bandwidth, troughput, latency, jitter* dan *packet lost.*

1. Metode untuk manajemen *bandwidth* menggunakan *Hierarchical Token Bucket (HTB)* yang merupakan salah satu disiplin antrian yang memiliki tujuan untuk menerapkan link sharing secara presisi dan adil. Dalam konsep link sharing, jika suatu kelas meminta kurang dari jumlah service yang telah ditetapkan untuknya, sisa *bandwidth* akan didistribusikan ke kelas-kelas yang lain yang meminta service.
2. Penelitian ini dilakukan di ICT dan Fakultas Ilmu Komputer dan Teknologi Informasi sebagai objek penelitian.
3. Penelitian ini tidak mencakup jaringan diluar ICT.

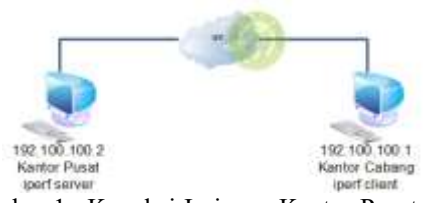
Adapun tujuan didalam penelitian ini adalah mengimplementasikan *Hierarchical Token Bucket (HTB)* untuk manajemen banwidth pada jaringan *WiFi* Fakultas Ilmu Komputer dan Teknologi Informasi sehingga akan diketahui mekanisme kerja sistem.

2. TINJAUAN PUSTAKA

2.3. IPERF

Iperf dikembangkan oleh *NLANR / DAST* sebagai alternatif modern untuk mengukur *bandwidth TCP* dan kinerja *UDP* secara maksimal. *Iperf* memungkinkan tuning berbagai parameter dan karakteristik *UDP*. *Iperf* melaporkan hasil *bandwidth*, delay jitter dan loss datagram disetiap hasil pengukurannya.

Tool ini biasanya digunakan ketika kita ingin mengetahui persis berapa bandwith dari pipa link yang kita miliki. sebagai contoh, kita berlangganan lokal link (bukan internet) yang akan menguhubungkan antara kantor pusat dengan kantor cabang sebesar 100 Mbps. untuk memastikan link tersebut sesuai apa tidak, maka kita bisa melakukan test *bandwidth* antara server (kantor pusat) dan client (kantor cabang) sebelum link tersebut kita bebani dengan berbagai macam aplikasi yang nantinya akan kita gunakan melalui link ini.



Gambar 1. Koneksi Jaringan Kantor Pusat ke Kantor Cabang

Pastikan kedua komputer telah terhubung secara *point to point* dengan memberikan IP dalam satu subnet yang sama. Untuk melakukan *test bandwidth Syntax* yang biasa digunakan adalah sebagai berikut, *Iperf tests*:

no arg. Default settings
-b Data format
-r Bi-directional bandwidth
-d Simultaneous bi-directional bandwidth
-w TCP Window size
-p, -t, -i Port, timing and interval
-u, -b UDP tests, bandwidth settings
-m Maximum Segment Size display
-M Maximum Segment Size settings
-P Parallel tests
-h help

Cara Menggunakan

Server side:

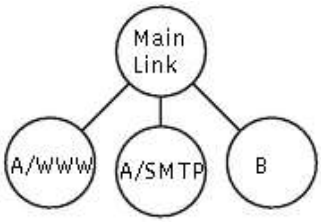
root@ubuntu:~# iperf -s
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)

Client Side:

root@repo:~# iperf -c 192.168.1.204
Client connecting to 192.168.1.204, TCP port 5001
TCP window size: 16.0 KByte (default)

2.4. *Hierarchical Token Bucket (HTB)*

Hierarchical Token Bucket (HTB) merupakan salah satu disiplin antrian yang memiliki tujuan untuk menerapkan *link sharing* secara presisi dan adil. Dalam konsep link sharing, jika suatu kelas meminta kurang dari jumlah service yang telah ditetapkan untuknya, sisa *bandwidth* akan didistribusikan ke kelas-kelas yang lain yang meminta service.



Gambar 2. Konsep *Link Sharing*

Hierarchical Token Bucket (HTB) menggunakan TBF sebagai estimator yang sangat mudah diimplementasikan. TBF sangat mudah diset karena banyak dari administrator jaringan yang memiliki ilmu tentangnya. Estimator ini hanya menggunakan parameter *rate*, sebagai akibatnya seseorang hanya perlu mengeset rate yang akan diberikan ke suatu kelas. Oleh karena itu *Hierarchical Token Bucket* (HTB) lebih mudah dan intuitif dibandingkan CBQ.

Pada *Hierarchical Token Bucket* (HTB) terdapat parameter ceil sehingga kelas akan selalu mendapatkan *bandwidth* di antara base rate dan nilai ceil ratenya. Parameter ini dapat dianggap sebagai estimator kedua, sehingga setiap kelas dapat

meminjam *bandwidth* selama *bandwidth* total yang diperoleh memiliki nilai di bawah nilai ceil. Hal ini mudah diimplementasikan dengan cara tidak mengijinkan proses peminjaman *bandwidth* pada saat kelas telah melampaui *rate* ini (keduanya *leaves* dan interior dapat memiliki *ceil*). Sebagai catatan, apabila nilai ceil sama dengan nilai *base rate*, maka akan memiliki fungsi yang sama seperti parameter bounded pada CBQ, di mana kelas-kelas tidak diijinkan untuk meminjam *bandwidth*. Sedangkan jika nilai ceil diset tak terbatas atau dengan nilai yang lebih tinggi seperti kecepatan link yang dimiliki, maka akan didapat fungsi yang sama seperti kelas non-bounded. Sebagai contoh, seseorang dapat menjamin *bandwidth* 1 Mbit untuk suatu kelas, dan mengijinkan penggunaan *bandwidth* sampai dengan 2 Mbit pada kelas tersebut apabila link dalam keadaan idle. Parameter ceil ini sangatlah berguna untuk ISP karena para ISP kemungkinan besar akan memakainya untuk membatasi jumlah servis yang akan diterima oleh suatu pelanggan walaupun pelanggan lain tidak melakukan permintaan servis, dengan kata lain kebanyakan ISP menginginkan pelanggan untuk membayar sejumlah uang lagi untuk memperoleh pelayanan yang lebih baik.

Untuk menjadwalkan pengiriman paket dari antrian, maka *Hierarchical Token Bucket (HTB)* menggunakan suatu proses penjadwalan yang dapat dijelaskan sebagai berikut:

1) ***Class*** merupakan parameter yang diasosiasikan dengan rate yang dijamin *(assured rate)* AR, ceil rate CR, prioritas P, level dan quantum. Class dapat memiliki parent. Selain AR dan CR, didefinisikan juga actual rate atau R, yaitu rate dari aliran paket yang meninggalkan class dan diukur pada suatu perioda waktu tertentu.
2) ***Leaf*** merupakan class yang tidak memiliki anak. Hanya leaf yang dapat memegang antrian paket.
3) ***Level*** dari kelas menentukan posisi dalam suatu hirarki. Leaf-leaf memiliki level 0, root class memiliki level=jumlah level-1 dan setiap inner class memiliki level kurang dari satu dari parentnya. Untuk gambar di bawah (jumlah level=3).
4) ***Mode*** dari class merupakan nilai-nilai buatan yang diperhitungkan dari R, AR, dan CR. Mode-mode yang mungkin adalah:
   - **Merah**: **R > CR**
   - **Kuning**: **R <= CR** and **R > AR**
   - **Hijau** selain yang di atas

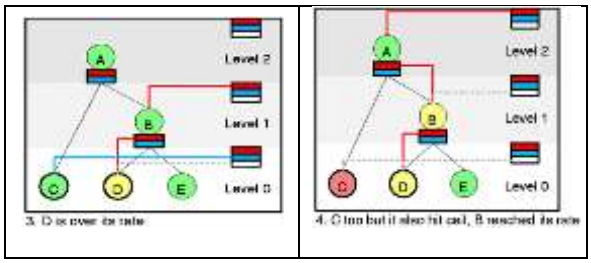
Kita asumsikan terdapat kelas-kelas yang tersusun seperti pohon pada *Hierarchical Token Bucket (HTB)* scheduler. Dalam tiap level, terdapat sebuah *self feed* yang terdiri dari self slot. Sehingga jika misalkan terdapat tiga buah level, maka terdapat 6 buah self slot (kita abaikan dulu slot putih). Setiap self slot, memegang daftar kelas-kelas, daftar tersebut digambarkan melalui garis berwarna yang terhubung dari self slot menuju kelas. Self slot berisikan daftar dari semua kelas-kelas hijau yang memiliki permintaan.

Setiap kelas inner (non-leaf), memiliki inner feed slot. Inner feed memiliki sebuah inner feed slot per prioritas (merah untuk prioritas tinggi, biru untuk prioritas rendah) dan per inner class. Inner slot akan menangani kelas-kelas anak yang berwarna kuning.

Gambar 3. Cara Kerja *Scheduler* di *Hierarchical Token Bucket (HTB) Step 1* dan *step 2*

Slot putih yang terdapat pada gambar di atas berfungsi sebagai antrian tunggu per level. Ia akan memegang daftar kelas-kelas yang berada pada levelnya yang berwarna merah atau kuning. Dari sini kita mungkin telah dapat melihat apabila kita dapat menjaga semua kelas pada daftar yang bersesuaian, maka proses pemilihan paket selanjutnya yang akan melakukan pengiriman paket dari antrian akan sangatlah mudah. HTB cukup melihat pada daftar *self feed* dan memilih slot yang tidak kosong (yaitu slot yang terhubung dengan garis ke sebuah kelas) dengan level yang paling rendah dan prioritas yang paling tinggi. Pada gambar 1 di atas tidak terdapat slot yang tidak kosong, sehingga tidak ada paket yang akan dikirim. Pada gambar 2, terdapat slot merah dengan level 0 pada kelas D, sehingga kelas D dapat mengirim paketnya.

Untuk lebih jelasnya dapat dijelaskan sebagai berikut. Pada gambar 1, tidak terdapat backlogged leaf (semua *non-backlogged leaf* digambarkan dengan lingkaran tipis) sehingga tidak ada yang dapat dilakukan. Pada gambar 2, paket-paket datang untuk C dan D. Sehingga HTB akan mengaktifkan kelas-kelas ini, dan karena keduanya berwarna hijau, HTB akan menambahkannya ke *self slot* yang bersesuaian. Proses dequeue akan memilih D karena D berada pada level yang rendah dan memiliki prioritas yang lebih tinggi daripada C.



Gambar 4. Cara Kerja *Scheduler* di *Hierarchical Token Bucket (HTB) Step 3* dan *Step 4*

Kita asumsikan paket telah dikirim dari kelas D dan *Hierarchical Token Bucket* (HTB) mengukur berapa besar paket yang dikirim melalui bantuan TBF. Karena paket yang dikirim melampaui besar rate yang telah ditetapkan untuk kelas D, maka *Hierarchical Token Bucket* (HTB) akan memaksa D untuk berubah

warna menjadi kuning. Sebagai bagian dari perubahan ini, HTB akan menghapus D dari daftar self feed dan menambahkan D ke *inner feed* B. HTB juga secara rekursif akan menambahkan B ke *self feed* pada levelnya. Karena D akan kembali ke keadaan hijau lagi pada suatu waktu, maka HTB akan menambahkan D ke slot putih pada level 0.

Proses *dequeue* sekarang akan memilih kelas C meskipun kelas C memiliki prioritas yang lebih rendah dibandingkan D. Ini dikarenakan C berada pada daftar slot dengan level yang lebih rendah. Secara intuitifpun hal ini baik dilakukan, untuk apa meminjam *bandwidth* jika ada yang dapat mengirim paket tanpa meminjam *bandwidth.*

1) TBF lebih akurat dalam menentukan rate.
2) TBF tidak memerlukan waktu akhir pengiriman yang tepat seperti pada ewma-idle. Akibatnya, HTB akan bekerja secara kuat pada semua perangkat jaringan di mana CBQ menunjukkan rate yang aneh dan salah.
3) HTB memungkinkan cross-device *bandwidth* sharing. Sehingga ethernet pertama dapat memakai *bandwidth* dari ethernet kedua.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Analisis

Sebelum *hierarchical token bucket (HTB)* diterapkan pada jaringan, dilakukan suatu pengukuran terhadap kinerja jaringan. Hal ini dimaksudkan agar terlihat bagaimana efek yang akan terjadi setelah diterapkannya *hierarchical token bucket* pada jaringan, terutama pada masalah *throughput* dan *delay* dari jaringan

3.1.1. Pengukuran *throughput, delay,* dan *jitter* Tanpa Manajemen *Bandwidth*

1. Pengukuran *throughput, delay,* dan *jitter* (dengan menggunakan iperf)

1) Trafik besar akan dikirim ke masing-masing client, yang diharapkan akan mengakibatkan pemakaian bandwidth secara penuh, misal dengan mengirimkan paket sebesar 6,99 MB. Output yang ingin diambil:
   - *Throughput, Delay,* Dan *Jitter* client 1
   - *Throughput, Delay,* Dan *Jitter* client 2
2) Trafik dikirim ke kedua client secara bersamaan yang mengakibatkan pemakaian bandwidth secara penuh, misal dengan mengirimkan paket sebesar 6,99 MB secara bersamaan ke kedua client. Output yang akan diambil:
   - *Throughput, Delay,* Dan *Jitter* client 1
   - *Throughput, Delay,* Dan *Jitter* client 2

2. Pengukuran *response time* (dengan menggunakan *ping*)

1. Mengukur response *time server-client* 1 pada saat tidak ada pemakaian
2. Mengukur *response time server-client* 1 pada saat pemakaian *Bandwidth* penuh oleh client 1
3. Mengukur *response time server-client* 2 pada saat pemakaian *Bandwidth* penuh oleh client1
4. Mengukur *response time server-client* 1 pada saat pemakaian *Bandwidth* penuh oleh client 1 dan client 2
5. Mengukur *response time server-client* 2 pada saat pemakaian *Bandwidth* penuh oleh client 1 dan client 2
6. Trafik dikirim hanya ke client 1

3. Hasil Analisis Pengukuran *response time* (dengan menggunakan ping)

Tabel di bawah ini menunjukkan bagaimana performansi jaringan client1 ketika client1 memakai bandwidth penuh sendirian sebelum diterapkannya HTB.

Tabel 1. Kualitas Jaringan

Sample	Delay Transmisi (s)	Ukuran paket (Mbytes)	Packets
Sample 1	26.3	6.99	1797
Sample 2	26.3	6.99	1797
Sample 3	26.3	6.99	1797
Sample 4	26.3	6.99	1797
Sample 5	26.3	6.99	1797
Average	**26.3**	**6.99**	**1797**

Untuk menjabarkan kualitas jaringan berdasarkan Tabel 1. Performansi Jaringan Client1 Tanpa Manajemen *Bandwidth* ketika Pemakaian *Bandwidth* Sendiri, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* pada client 1.

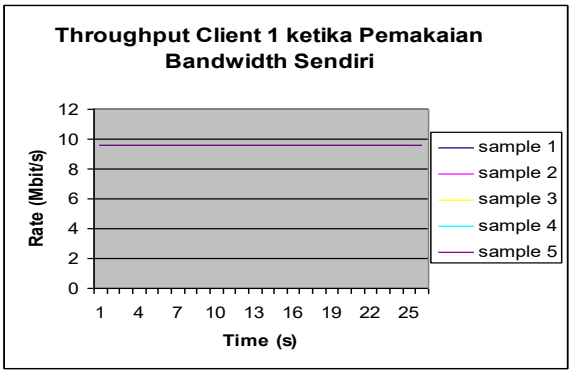
a. Pengujian throughput pada kecepatan transfer 9,6 Mbit/s atau 1.225 Kbps. Dari tabel 4.1 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :
   Throughput = Paket data yang diterima / Lama pengamatan
   = (6990000 bytes)/26,3 s
   = 265779,46 bytes/s atau **265,77 kbps**
b. Pengujian jitter pada kecepatan transfer 9,6 Mbit/s atau 1.225 Kbps dengan total packet 1797. Dari tabel 4.1 data yang telah di*capture* dengan *IPREF* maka didapatkan *jitter* dengan cara perhitungan sebagai berikut. Jitter = Total variasi delay / (Total packet yang diterima – 1)
   = (26.3-26.3) + (26.3-26.3) +(26.3-26.3)+ (26.3-26.3) / 1796
   = 0 s / 1796
   = **0 ms**
c. Pengujian *delay* pada kecepatan transfer transfer 9,6 Mbit/s atau 1.225 Kbps. tabel 4.1 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :
   Rata-rata delay = Total delay / Total packet yang diterima
   = 311,5 s / 1797

= 0,0731775 s
= **73,17 ms**

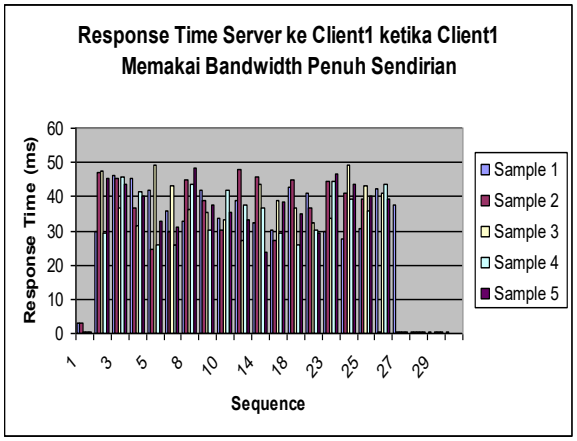
Total delay didapatkan dengan menjumlahkan keseluruhan delay yang ada antara paket satu dengan paket lainnya.

Gambar di bawah ini menunjukkan bagaimana throughput client1, dan response time client1 dan client2 ketika client1 memakai bandwidth penuh sendirian sebelum diterapkannya HTB.

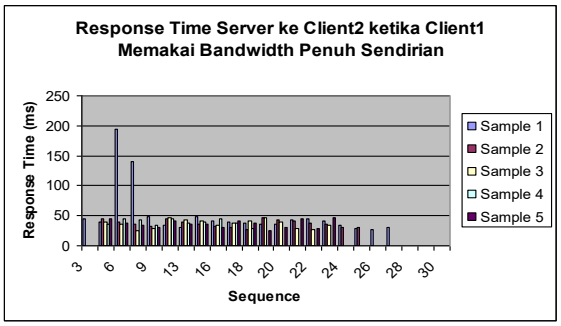
.



Gambar 5. *Throughput Client1* Tanpa Manajemen *Bandwidth* ketika Pemakaian *Bandwidth* Sendiri



Gambar 6. *Response Time Server-Client1* Tanpa Manajemen *Bandwidth* ketika Pemakaian *Bandwidth* Sendiri



Gambar 7. *Response Time Server-Client*2 Tanpa Manajemen Bandwidth ketika Client1 Memakai Bandwidth Sendirian

Tabel di bawah ini menunjukkan bagaimana performansi jaringan client1 ketika client1 & client2 memakai bandwidth penuh secara bersamaan sebelum diterapkannya HTB.

Tabel 2 Performansi Jaringan Client1 Tanpa Manajemen *Bandwidth* ketika Pemakaian Bandwidth Bersamaan

Sample	Elapsed Time (s)	Ukuran paket (Mbytes)	Packets
Sample 1	52.7	6.99	976
Sample 2	52.7	6.99	976
Sample 3	52.6	6.99	979
Sample 4	52.6	6.99	978
Sample 5	52.7	6.99	976
Average	**52.66**	**6.99**	**977**

Untuk menjabarkan kualitas jaringan berdasarkan tabel 2. Performansi Jaringan Client1 Tanpa Manajemen *Bandwidth* ketika Pemakaian *Bandwidth* bersamaan dengan Client 2, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* pada sisi client 1

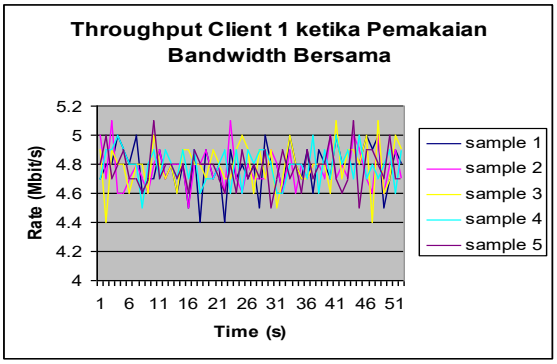
a. Pengujian throughput pada kecepatan transfer 9,6 Mbit/s atau 1.225 Kbps.
Dari tabel 42 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :
Throughput = Paket data yang diterima / Lama pengamatan
= (6990000 bytes)/52,66 s
= 137738,32 bytes/s atau **137,738 kbps**

b. Pengujian jitter pada kecepatan transfer 9,6 Mbit/s atau 1.225 Kbps dengan rata-rata total packet terkirim, sebesar 977 packet . Dari tabel 2 data yang telah di*capture* dengan *IPREF* maka didapatkan *jitter* dengan cara perhitungan sebagai berikut :
Jitter = Total variasi delay / (Total packet yang diterima – 1)
= (52,7-52,7) + (52,7-52,6) +(52,6-52,6)+ (52,6-52,7) / 977
= 0 s / 977
= **0 ms**
Total variasi *delay* didapatkan dengan menjumlahkan keseluruhan selisih *delay* yang ada antara paket satu dengan paket lainnya.

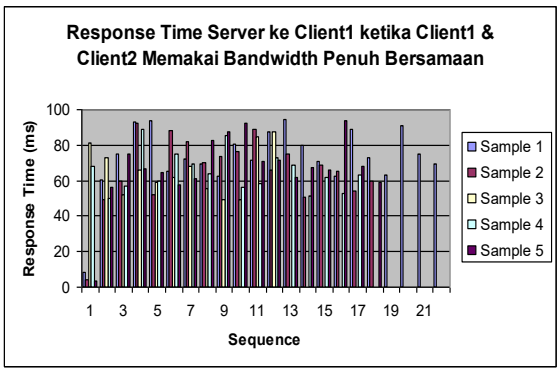
c. Pengujian *delay* pada kecepatan transfer 9,6 Mbit/s atau 1.225 Kbps. tabel 4.2 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :
Rata-rata delay = Total delay / Total packet yang diterima
= 263,3 s / 977
= 0,269498 s
= **26,949 ms**

Total delay didapatkan dengan menjumlahkan keseluruhan delay yang ada antara paket satu dengan paket lainnya.

Gambar 8 Throughput Client1 Tanpa Manajemen Bandwidth ketika Pemakaian Bandwidth bersamaan menunjukkan bagaimana *throughput* dan *response time* client1 ketika client1 & client2 memakai *bandwidth* penuh secara bersamaan sebelum diterapkannya *Hierarchical Token Bucket (HTB)* terlihat tidak stabil seperti penggunaan bandwidth secara sendiri.



Gambar 8. Throughput Client1 Tanpa Manajemen Bandwidth ketika Pemakaian Bandwidth Bersamaan



Gambar 9. *Response Time Server-Client1* Tanpa Manajemen Bandwidth ketika Pemakaian Bandwidth Bersamaan

Dapat dilihat dari hasil throughput dan delay yang didapat, pada saat pemakaian bandwidth sendiri oleh masing-masing client, masing-masing client mendapatkan jatah bandwidth 9,6 Mbit/s, yang merupakan throughput maksimal dari ethernet 10 Mbit. Hal ini dikarenakan pada saat pemakaian bandwidth secara individual, maka alokasi *throughput* maksimal akan digunakan oleh client yang pada saat itu menggunakan bandwidth secara penuh. Ketika pemakaian bersama, maka throughput yang masuk ke masing-masing client akan terbagi menjadi dua, yaitu rata-rata 4,8 Mbit/s pada masing-masing client. *Ripple* yang terjadi pada grafik tersebut adalah suatu kewajaran, karena ada kalanya terjadi *collision* dan sifat dari media di jaringan yang terkadang melakukan burst data. Untuk itu, maka di dalam pengukuran *throughput* dari suatu jaringan, kita harus mengamati rata-rata throughput dari suatu rentang waktu tertentu. Dari grafik yang diperoleh, kita dapat melihat bagaimana kompetisi yang tidak dapat dikontrol ketika tidak diterapkannya *bandwidth management*.

3.1.2 Simulasi Pembagian Kapasitas *bandwidth* Berdasarkan Alamat IP

3.1.2.1 Motivasi Penerapan *Hierarchical Token Bucket (HTB)* berdasarkan IP

Penggunaan bandwidth pada sebuah organisasi yang memiliki jaringan komputer yang cukup besar seperti ISP, kampus, perkantoran, dan warnet, tentunya memerlukan suatu manajemen tertentu. Jika penggunaan bandwidth tersebut dibiarkan begitu saja tanpa adanya suatu manajemen, maka yang akan terjadi adalah hukum rimba; siapa yang kuat dialah yang menang.

Pada simulasi kali ini akan ditunjukkan bagaimana cara membagi kapasitas bandwidth yang kita miliki berdasarkan alamat IP dari client yang terhubung dalam suatu jaringan. Hal ini dilakukan karena penulis melihat hampir sebagian besar pengalamatan di jaringan komputer menggunakan alamat IP.

3.1.2.2 Metode Penerapan *Hierarchical Token Bucket (HTB)* berdasarkan IP

Pembagian bandwidth sama seperti pada kondisi 2, namun dibedakan pengklasifikasian kelasnya berdasarkan alamat IP. Kelas 1 (semua trafik yang menuju IP 192.168.1.1) dan kelas 2 (semua trafik yang menuju IP 192.168.1.2).

Pengukuran *throughput, delay, jitter*, dan *response time* dilakukan dengan cara yang sama seperti kondisi 2. Konfigurasi *Hierarchical Token Bucket (HTB)* yang akan digunakan adalah sebagai berikut:

```
tc qdisc add dev eth0 root
handle 1: htb
tc class add dev eth0 parent 1:
classid 1:1 htb rate 512kbit
ceil 512kbit
tc class add dev eth0 parent 1:1
classid 1:10 htb rate 384kbit
ceil 512kbit
tc class add dev eth0 parent 1:1
classid 1:11 htb rate 128kbit
ceil 512kbit
tc filter add dev eth0 protocol
ip parent 1:0 prio 1 u32 match
ip dst 192.168.1.1 flowid 1:10
tc filter add dev eth0 protocol
ip parent 1:0 prio 1 u32 match ip
dst 192.168.1.2 flowid 1:11
```

3.1.2.3 Hasil Pengamatan Penerapan *Hierarchical Token Bucket (HTB)* berdasarkan IP

1) Trafik dikirim hanya ke client 1

Tabel di bawah ini menunjukkan bagaimana performansi jaringan client1 ketika client1 memakai bandwidth penuh sendirian setelah diterapkannya *Hierarchical Token Bucket (HTB)* berdasarkan IP. Dengan diterapkannya *Hierarchical Token Bucket (HTB)* kecepatan transfer maksimal client 1 dibatasi

hanya 512 kbit. Berikut tabel Performansi Jaringan Client1 dengan Manajemen Bandwidth Berdasarkan Port ketika Pemakaian Bandwidth Sendiri.

Tabel 3. Performansi Jaringan Client1 dengan Manajemen *Bandwidth* Berdasarkan Port ketika Pemakaian *Bandwidth* Sendiri.

Sample	Elapsed Time (s)	Capacity (Mbytes)	Packets
Sample 1	32.9	2	595
Sample 2	32.9	2	610
Sample 3	32.9	2	597
Sample 4	32.9	2	598
Sample 5	32.9	2	597
Average	**32.9**	**2**	**599**

Untuk menjabarkan kualitas jaringan berdasarkan tabel 3. Performansi Jaringan Client1 dengan Manajemen *Bandwidth* setelah diterapkannya *Hierarchical Token Bucket (HTB).* Pemakaian *Bandwidth* secara sendiri oleh clien 1, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* sebagai berikut :

a. Pengujian throughput pada kecepatan transfer 510.4 Kbps. Dari tabel 4.3 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :

Throughput = Paket data yang diterima / Lama pengamatan

= (2000000 bytes)/32.9 s

= 60790,27 bytes/s atau **60,79 kbps**

b. Pengujian jitter pada kecepatan transfer 510.4 Kbps. dengan rata-rata total packet terkirim, sebesar 599 packet . Dari tabel 4.3 data yang telah di*capture* dengan *IPREF* maka didapatkan *jitter* dengan cara perhitungan sebagai berikut :

Jitter = Total variasi delay / (Total packet yang diterima – 1)

= (32,9-32,9) + (32,9-32,9) +(32,9-32,9)+ (32,9-32,9) / 599

= 0 s / 599

= **0 ms**

Total variasi *delay* didapatkan dengan menjumlahkan keseluruhan selisih *delay* yang ada antara paket satu dengan paket lainnya.

c. Pengujian *delay* pada kecepatan transfer 510.4 Kbps. tabel 4.3 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :

Rata-rata delay = Total delay / Total packet yang diterima

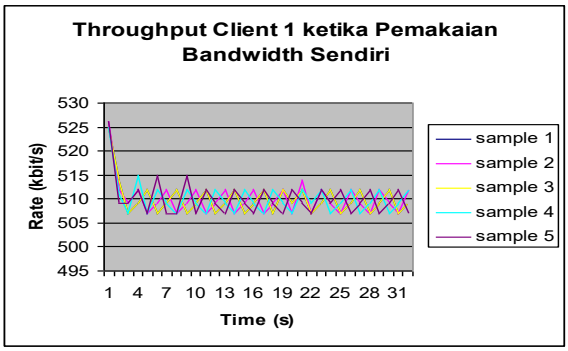
= 164,5 s / 599

= 0,27462 s

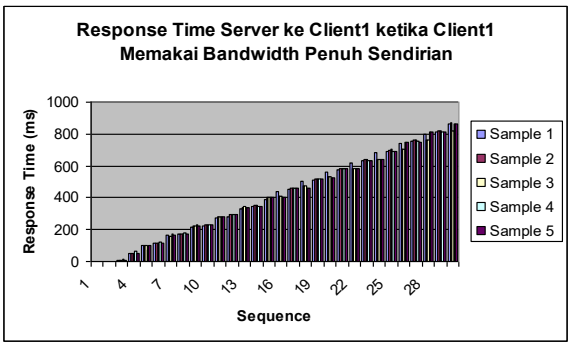
= **27,462 ms**

Total *delay* didapatkan dengan menjumlahkan keseluruhan *delay* yang ada antara paket satu dengan paket lainnya.

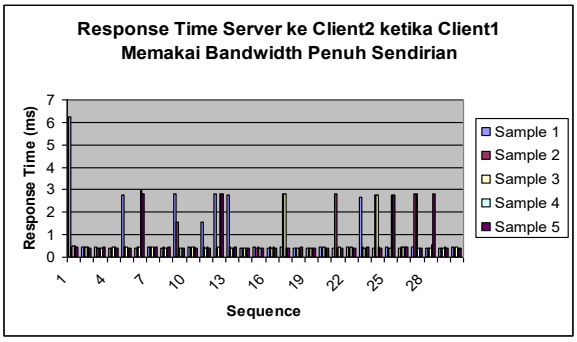
Gambar di bawah ini menunjukkan bagaimana throughput client1 dan response time client1 & client2 ketika client1 memakai bandwidth penuh sendirian setelah diterapkannya *Hierarchical Token Bucket (HTB)* berdasarkan IP.



Gambar 10. Throughput Client1 dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Sendiri



Gambar 11. Response Time Server-Client1 dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Sendiri



Gambar 12. Response *Time Server-Client*2 dengan Manajemen Bandwidth Berdasarkan *IP* ketika Pemakaian *Bandwidth* Bersamaan.

a. Trafik dikirim hanya ke client 2

Tabel di bawah ini menunjukkan bagaimana performansi jaringan client2 ketika client2 memakai bandwidth penuh sendirian setelah diterapkannya HTB berdasarkan IP.

Tabel 4. Performansi Jaringan Client2 dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Sendiri

Sample	Elapsed Time (s)	Capacity (Mbytes)	Packets
Sample 1	32.9	2	598
Sample 2	32.9	2	597
Sample 3	32.9	2	597
Sample 4	32.9	2	595
Sample 5	32.9	2	610
Average	**32.9**	**2**	**599**

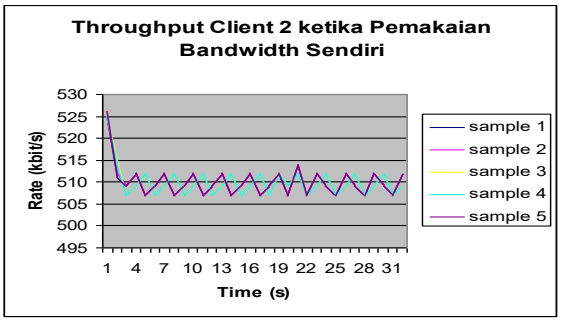
Untuk menjabarkan kualitas jaringan berdasarkan tabel 4 Performansi Jaringan Client2 dengan Manajemen *Bandwidth* setelah diterapkannya *Hierarchical Token Bucket (HTB).* Pemakaian *Bandwidth* secara sendiri oleh Client 2, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* sebagai berikut :

a. Pengujian throughput pada kecepatan transfer 510.4 Kbps. Dari tabel 4.4 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :
Throughput = Paket data yang diterima / Lama pengamatan
= (2000000 bytes)/32.9 s
= 60790,27 bytes/s atau **60,79 kbps**

b. Pengujian jitter pada kecepatan transfer 510.4 Kbps. dengan rata-rata total packet terkirim, sebesar 599 packet . Dari tabel 4.4 data yang telah di*capture* dengan *IPREF* maka didapatkan *jitter* dengan cara perhitungan sebagai berikut :
Jitter = Total variasi delay / (Total packet yang diterima – 1)
= (32,9-32,9) + (32,9-32,9) +(32,9-32,9)+ (32,9-32,9) / 599
= 0 s / 599
= **0 ms**
Total variasi *delay* didapatkan dengan menjumlahkan keseluruhan selisih *delay* yang ada antara paket satu dengan paket lainnya.

c. Pengujian *delay* pada kecepatan transfer 510.4 Kbps. tabel 4.4 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :
Rata-rata delay = Total delay / Total packet yang diterima
= 164,5 s / 599
= 0,27462 s
= **27,462 ms**
Total *delay* didapatkan dengan menjumlahkan keseluruhan *delay* yang ada antara paket satu dengan paket lainnya.

Gambar di bawah ini menunjukkan bagaimana throughput client2 ketika client2 memakai bandwidth penuh sendirian setelah diterapkannya HTB berdasarkan IP.



Gambar 13. *Throughput Client1* dengan Manajemen *Bandwidth* Berdasarkan *IP* ketika Pemakaian Bandwidth Sendiri

b. Trafik dikirim ke client 1 dan 2
a. pada client 1

Tabel di bawah ini menunjukkan bagaimana performansi jaringan client1 ketika client1 & client2 memakai bandwidth penuh secara bersamaan setelah diterapkannya HTB berdasarkan IP.

Tabel 5 Performansi Jaringan Client1 dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Bersamaan

Sample	Elapsed Time (s)	Capacity (Mbytes)	Packets
Sample 1	43.8	2	598
Sample 2	43.9	2	597
Sample 3	43.8	2	597
Sample 4	43.8	2	595
Sample 5	43.9	2	610
Average	**43.84**	**2**	**599**

Untuk menjabarkan kualitas jaringan berdasarkan tabel 5. Performansi Jaringan Client1 dengan Manajemen *Bandwidth* setelah diterapkannya *Hierarchical Token Bucket (HTB).* Pemakaian *Bandwidth* bersamaan dengan Client 2, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* pada sisi client 1 sebagai berikut :

a. Pengujian throughput pada kecepatan transfer 510.4 Kbps. Dari tabel 4.3 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :
Throughput = Paket data yang diterima / Lama pengamatan
= (2000000 bytes)/32.9 s
= 60790,27 bytes/s atau **60,79 kbps**

b. Pengujian jitter pada kecepatan transfer 510.4 Kbps. dengan rata-rata total packet terkirim, sebesar 599 packet . Dari tabel 4.3 data yang telah di*capture* dengan *IPREF* maka

didapatkan *jitter* dengan cara perhitungan sebagai berikut :

Jitter = Total variasi delay / (Total packet yang diterima – 1)

= (32,9-32,9) + (32,9-32,9) +(32,9-32,9)+ (32,9-32,9) / 599

= 0 s / 599

= **0 ms**

Total variasi *delay* didapatkan dengan menjumlahkan keseluruhan selisih *delay* yang ada antara paket satu dengan paket lainnya.

c. Pengujian *delay* pada kecepatan transfer 510.4 Kbps. tabel 4.3 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :

Rata-rata delay = Total delay / Total packet yang diterima

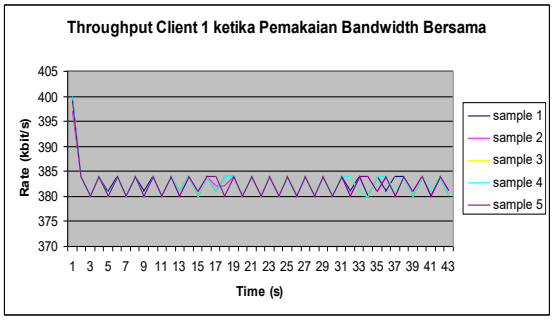
= 164,5 s / 599

= 0,27462 s

= **27,462 ms**

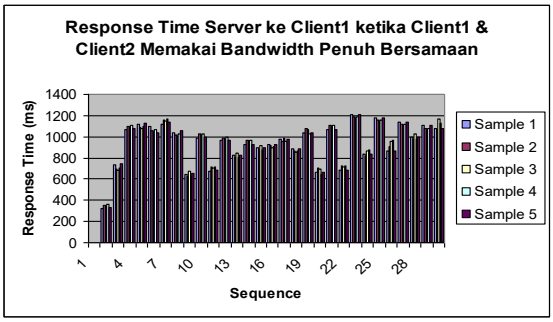
Total *delay* didapatkan dengan menjumlahkan keseluruhan *delay* yang ada antara paket satu dengan paket lainnya.

.



Gambar 14. *Throughput Client1* dengan Manajemen *Bandwidth* Berdasarkan *IP* ketika Pemakaian Bandwidth Bersama

Gambar di bawah ini menunjukkan bagaimana throughput dan response time client1 ketika client1 & client2 memakai bandwidth penuh secara bersaman setelah diterapkannya HTB berdasarkan IP.



Gambar 15. *Response Time Server-Client1* dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Bersamaan.

Tabel 6. Performansi Jaringan Client2 dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Bersamaan

Sample	Elapsed Time (s)	Capacity (Mbytes)	Packets
Sample 1	65.9	2	598
Sample 2	65.9	2	597
Sample 3	65.9	2	597
Sample 4	65.9	2	595
Sample 5	65.9	2	610
Average	**65.9**	**2**	**599**

Untuk menjabarkan kualitas jaringan berdasarkan tabel 4.6. Performansi Jaringan Client1 dengan Manajemen *Bandwidth* setelah diterapkannya *Hierarchical Token Bucket (HTB).* Pemakaian *Bandwidth* bersamaan dengan Client 2, akan dijelaskan perhitungan *Throughput, Delay,* Dan *Jitter* pada sisi client 1 sebagai berikut :

d. Pengujian throughput pada kecepatan transfer 510.4 Kbps. Dari tabel 4.6 data yang telah di*capture* dengan *IPREF* maka didapatkan *throughput* dengan cara perhitungan sebagai berikut :

Throughput = Paket data yang diterima / Lama pengamatan

= (2000000 bytes)/32.9 s

= 60790,27 bytes/s atau **60,79 kbps**

e. Pengujian jitter pada kecepatan transfer 510.4 Kbps. dengan rata-rata total packet terkirim, sebesar 599 packet . Dari tabel 4.6 data yang telah di*capture* dengan *IPREF* maka didapatkan *jitter* dengan cara perhitungan sebagai berikut :

Jitter = Total variasi delay / (Total packet yang diterima – 1)

= (32,9-32,9) + (32,9-32,9) +(32,9-32,9)+ (32,9-32,9) / 599

= 0 s / 599

Total variasi *delay* didapatkan dengan menjumlahkan keseluruhan selisih *delay* yang ada antara paket satu dengan paket lainnya.

a. Pengujian *delay* pada kecepatan transfer 510.4 Kbps. tabel 4.6 data yang telah di*capture* dengan *IPREF* maka didapatkan rata-rata *delay* dengan cara perhitungan sebagai berikut :

Rata-rata delay = Total delay / Total packet yang diterima

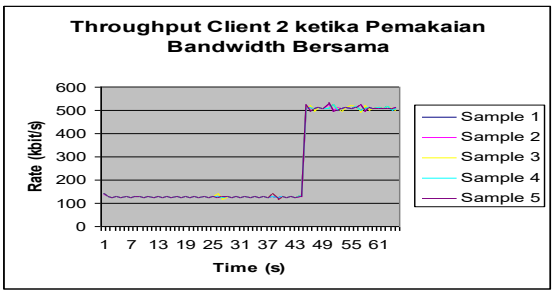
= 164,5 s / 599

= 0,27462 s

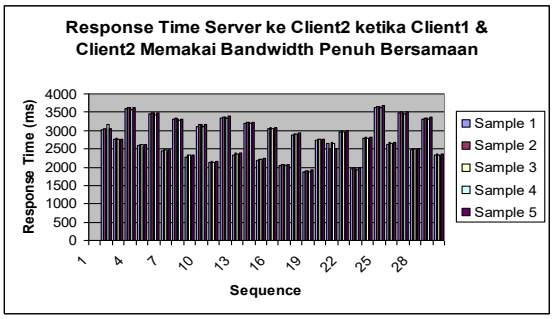
= **27,462 ms**

Total *delay* didapatkan dengan menjumlahkan keseluruhan *delay* yang ada antara paket satu dengan paket lainnya.

Gambar di bawah ini menunjukkan bagaimana *throughput* dan *response time* client2 ketika client1 & client2 memakai *bandwidth* penuh secara bersaman setelah diterapkannya HTB berdasarkan IP.



Gambar 16. *Throughput Client2* dengan Manajemen Bandwidth Berdasarkan IP ketika Pemakaian Bandwidth Bersama



Gambar 17. *Response Time Server-Client2* dengan Manajemen *Bandwidth* Berdasarkan IP ketika Pemakaian *Bandwidth* Bersamaan

Pada Gambar 17. *Response Time Server-Client2* dengan Manajemen *Bandwidth* Berdasarkan IP ketika Pemakaian *Bandwidth* Bersamaan terlihat lebih stabil, sehingga sudah dapat dipastikan pengguna dapat menggunakan akses internet tidak mengalami terputusnya akses internet *(internet access disconected)*. Hal ini berbeda dengan tanpa manajemen bandwidth, penggunaan secara bersamaan akan menyebabkan tarik menarik bandwidth sehingga akan banyak pengguna dijaringan yang sama akan mengalami *request time out.*

4. KESIMPULAN DAN SARAN

4.1 Kesimpulan

1. Jaringan yang tidak diterapkan manajemen *bandwidth* akan berakibat pada *throughput* yang tidak terkontrol. Hal ini yang terjadi di Fakultas Ilmu Komputer dan Teknologi Informasi (FKTI) sehingga perlu diterapkan manajemen *bandwidth* dengan *Hierarchical Token Bucket (HTB).*
2. Implementasi *Hierarchical Token Bucket (HTB).* dapat mengontrol *throughput* dari setiap client yang ada di jaringan. Hal ini terbukti pada hasil simulasi di mana kapasitas *bandwidth* 512 kbit/s dapat kita bagi-bagi sesuai keinginan kita, misal 256 kbit/s untuk client1 dan 128 kbit/s untuk client 2.
3. Pada jaringan dengan *Hierarchical Token Bucket (HTB).* diimplementasikan, penggunaan *bandwidth* pada satu *client* tidak akan mempengaruhi *response time* pada client lainnya dalam satu jaringan. Hal ini dapat dibuktikan pada hasil simulasi di atas, bahwa ketika client 1 menggunakan *bandwidth* secara penuh, *response time* dari *server* ke *client* 2 tetap berada rata-rata di bawah 10 ms yag berarti kualitas jaringan baik.
4. Implementasi HTB berdasarkan IP akan berpengaruh terhadap response time. Hal ini terlihat dari hasil *response time server* menuju ke *client* yang menggunakan *bandwidth* secara penuh, di mana response time akan meningkat seiring dengan meningkatnya penggunaan *bandwidth*
5. Dalam sistem kerja jaringan sangat diperlukan stabilitas dan kecepatan yang stabil karena sebagai pendukung kinerja jaringan tersebut.

4.2 Saran

Penulis sangat menyadari bahwa penelitian yang dilakukan masih banyak kekurangan dan kelemahan adapun saran yang dapat diberikan oleh penulis kepada pembaca yang ingin mengembangkannya sebagai berikut :

1) Untuk menerapkan manajemen *bandwidth* dengan *Hierarchical Token Bucket (HTB) bandwidth* yang disediakan minimal 2 Mb sehingga seluruh pengguna akses internet dapat menggunakan fasilitas akses internet.
2) Manajemen bandwidth juga dapat dikembangkan dengan mengunakan perangkat mikrotic untuk memudahkan pembagian *bandwidth* pada Fakultas Ilmu Komputer dan Teknologi Informasi (FKTI).
3) Semoga dengan adanya penelitian ini diharapkan pihak Universitas Mulawarman dapat memperbaiki kinerja jaringan *wireless* yang ada pada saat ini. Oleh karenanya pada saat ini mahasiswa sangat membutuhkan akses *internet* yang mumpuni serta akurat dalam hal pengaksesan *internet.*
4) Semoga dengan adanya penelitian dapat digunakan oleh pihak *ICT* dan Fakultas dalam pembagian *bandwitdh* yang sesuai dengan kebutuhan mahasiswa Fakultas Ilmu Komputer dan Teknologi Informasi (FKTI) dalam menunjang mutu pendidikan.

5. DAFTAR PUSTAKA

[1]. Kurniawan, Wiharsono, 2007, *Computer Starter Guide: Jaringan Komputer*. Semarang: Penerbit Andi Offset

[2]. Kusrini, 2007, *Konsep dan Aplikasi Sistem Penunjang Keputusan*, Yogyakarta: Penerbit Andi Offset

[3]. Suarna, Nana, 2007, *Pengertian Local Area Network dan Petunjuk Teoritis Pengantar LAN*, Yogyakarta: Penerbit Andi Offset

[4]. Sopandi, Dede, 2007, *Pengertian Jaringan Komputer*, Semarang: Penerbit Andi Offset

[5]. Gunawan, 2008, *Pengertian Quality of Service (QoS)*, Bandung: Penerbit Andi Offset

[6]. Arifia, Narendro, 2010, *Pengertian File Sharing*, Surabaya: Penerbit Andi Offset

[7]. Sofana, 2011, *Pengertian Jaringan Wireless dan Performance Network*, Bandung: Penerbit Andi Offset
[8]. Hermawan, Julius, 2009, *Membangun Decision Support System*, Yogyakarta: Penerbit Andi Offset
[9]. Iqbal, Hasan, 2009, *Teori Pengambilan Keputusan*, Jakarta: Penerbit Ghalia Indonesia
[10]. Jogiyanto. HM, 2010, *Analisis dan Desain Sistem Informasi*. Yogyakarta: Penerbit Andi Offset
[11]. MADCOM, 2009, *Seri Membongkar Misteri Internet*, Yogyakarta: Penerbit Andi Offset
[12]. Mustofa, Jazi Eko Istiyanto, Sugeng, Winarno, 2010, *Arsitektur Real-Time System sebagai Pemantau Jaminan QoS dengan Metode RTFM (Real Time Flow Measurement)*, Yogyakarta: Penerbit Andi Offset
[13]. Pressman, Roger S, 2012, *Rekayasa Perangkat Lunak: Pendekatan Praktisi* (buku II edisi 7), Yogyakarta: Penerbit Andi Offset
[14]. Raja, Melva Deli Yanti Lumban, 2013, *Sistem Manajemen Download dalam upaya Meminimalisir pemborosan Bandwidth menggunakan Metode Analytical Hierarchy Process (AHP)*, STMIK Budidarma Medan
[15]. Simarmata, Janner, 2010, *Rekayasa Perangkat Lunak*, Yogyakarta: Penerbit Andi Offset
[16]. Saaty, T.L, 2008, *Decision Making for Leader Forth Edision*, *Unerversity of Pittsburd, RWS Publication*
[17]. Turban, Efraim, 2005, *Decision Support System and Intelligent System: Jilid I (Sistem Pendukung Keputusan dan Sistem Cerdas)*, Yogyakarta: Penerbit Andi Offset
[18]. Setio Dewo, E, 2003, *Artikel Populer Ilmu Komputer: Bandwidth dan Throughput*, http://ilmukomputer.com/demo-bandwidth.html, diakses pada tanggal 23 Oktober 2014 pukul 22:56 Wita
[19]. R Aulia, Haviluddin. 2016. Implementation of Bandwidth Management Authentication. International Journal of Computing and Informatics (IJCANDI) 1 (1), 1-8