ANALISIS PERBANDINGAN HTB *(*HIERARCHICAL TOKEN BUCKET*)* DAN CBQ *(*CLASS BASED QUEUING*)* UNTUK MENGATUR BANDWIDTH MENGGUNAKAN LINUX

Abas Ali Pangera

Abstraksi

Manajemen *bandwidth* menjadi hal yang mutlak diperlukan bagi jaringan multi layanan, semakin banyak dan bervariasinya aplikasi yang dapat dilayani oleh suatu jaringan berpengaruh pada penggunaan *link* dalam jaringan tersebut. *Link-link* yang ada harus mampu menangani kebutuhan *user* akan aplikasi tesebut bahkan dalam keadaan kongesti sekalipun, harus ada suatu jaminan bahwa *link* tetap dapat berfungsi sebagaimana mestinya walaupun terjadi ledakan permintaan aplikasi. Manajemen *bandwidth* memegang perananan penting dalam mengatur jenis aplikasi yang bisa mengakses *link* yang ada selain itu manajemen *bandwidth* mampu memberikan garansi kepada aplikasi yang mendapat alokasi *bandwidth* untuk terus mengirimkan data sesuai dengan alokasinya sekalipun terjadi kemacetan dalam jaringan bahkan dalam keadaan tertentu ketika alokasi *bandwidth* yang dimiliki oleh suatu aplikasi/layanan tidak digunakan maka oleh *Bandwidth Manager* alokasi *bandwidth* yang *idle* tersebut dapat dialihkan sementara waktu kepada kelas yang sedang mengalami *backlog*/timbunan antrian, hal ini memberikan keuntungan mempercepat hilangnya *backlog* suatu kelas sekaligus mengoptimalkan penggunaan *link* yang ada. Class Based Queuing (CBQ) dan Hierarchical Token Bucket (HTB) sebagai implementator manajemen *bandwidth* yang tersedia secara gratis dan dapat dijalankan diatas platform sistem Operasi LINUX merupakan *Bandwidth Manager* yang layak dianalisa keunggulan dan kelemahannya, diharapkan penggunaannya yang tepat dan akurat akan membuat jaringan yang menerapkan *Bandwidth Manager* ini bekerja secara optimal.

Kata kunci : *Bandwidth Manager*, CBQ, HTB, *link sharing*

1. Landasan Teori

QoS merupakan kependekan dari *Quality of Service*. Dalam buku *Quality of Service* yang ditulis oleh *Paul Ferguson*, didefinisikan bahwa QoS adalah suatu pengukuran tentang seberapa baik jaringan dan merupakan suatu usaha untuk mendefinisikan karakteristik dan sifat dari suatu servis. QoS biasanya digunakan untuk mengukur sekumpulan atribut performansi yang telah dispesifikasikan dan

biasanya diasosiasikan dengan suatu servis. Pada jaringan berbasis IP, IP QoS mengacu pada performansi dari paket-paket IP yang lewat melalui satu atau lebih jaringan.

QoS didesain untuk membantu pemakai menjadi lebih produktif dengan memastikan bahwa dia mendapatkan performansi yang handal dari aplikasi-aplikasi berbasis jaringan.

QoS mengacu pada kemampuan jaringan untuk menyediakan layanan yang lebih baik pada trafik jaringan tertentu melalui teknologi yang berbeda-beda. QoS merupakan suatu tantangan yang cukup besar dalam jaringan berbasis IP dan internet secara keseluruhan. Tujuan dari QoS adalah untuk memuaskan kebutuhan-kebutuhan layanan yang berbeda, yang menggunakan infrastruktur yang sama. QoS menawarkan kemampuan untuk mendefinisikan atribut-atribut layanan jaringan yang disediakan, baik secara kualitatif maupun kuantitatif.

Komponen-komponen dari QoS adalah:

- ***Delay***, merupakan total waktu yang dilalui suatu paket dari pengirim ke penerima melalui jaringan. Delay dari pengirim ke penerima pada dasarnya tersusun atas hardware latency, delay akses, dan delay transmisi. Delay yang paling sering dialami oleh trafik yang lewat adalah delay transmisi, yang dapat dirumuskan sebagai berikut:

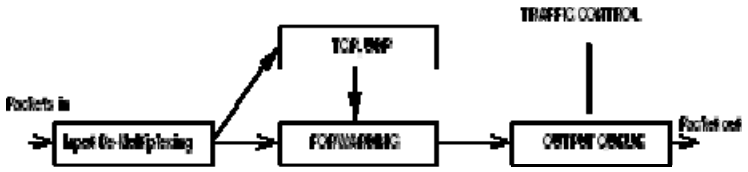
$$Delay = \frac{packet\_size \times 8}{line\_speed} \times 1000 \ ms \qquad (1)$$

  Untuk aplikasi-aplikasi suara dan video interaktif, kemunculan dari delay akan mengakibatkan sistem seperti tak merespon.
- ***Jitter***, merupakan variasi dari delay end-to-end. Level-level yang tinggi pada jitter dalam aplikasi-aplikasi berbasis UDP merupakan situasi yang tidak dapat diterima di mana aplikasi-aplikasinya merupakan aplikasi-aplikasi real-time, seperti sinyal audio dan video. Pada kasus seperti itu, jitter akan menyebabkan sinyal terdistorsi, yang dapat diperbaiki hanya dengan meningkatkan buffer di antrian.
- ***Bandwidth***, merupakan rate transfer data maksimal yang dapat diteruskan antara dua titik.

Dalam implementasi manajemen bandwidth pada jaringan berbasis TCP/IP, banyak sekali *tool-tool* yang dapat kita pakai, baik dalam bentuk perangkat lunak, maupun perangkat keras. *Tools* tersebut ada yang berharga sangat mahal, seperti Cisco, dan ada juga yang bersifat gratis seperti aplikasi-aplikasi manajemen bandwidth di Linux. Oleh karena itu, banyak sekali perusahaan-perusahaan yang menerapkan Linux sebagai basis sistem operasi yang akan menjalankan aplikasi-aplikasi manajemen bandwidth. Di samping dikenal dengan gratisnya, Linux juga

dikenal sebagai suatu sistem operasi yang sangat handal dan akurat dalam penerapan manajemen bandwidth.

Prinsip dasar dari implementasi manajemen bandwidth pada Linux dapat dijelaskan pada gambar berikut.



Gambar 1 Traffic Control

Gambar ini menunjukkan bagaimana kernel memproses paket yang datang, dan bagaimana ia mengolah paket-paket untuk dikirimkan ke jaringan. *Input demultiplexer* akan memeriksa apakah paket yang datang ditujukan untuk node lokal. Jika ya, maka paket akan dikirimkan ke *layer* yang lebih tinggi untuk pemrosesan lebih lanjut. Jika tidak, maka paket akan diteruskan ke blok *forwarding*. Blok forwarding, yang mungkin juga dapat menerima paket lokal dari *layer* yang lebih tinggi, akan melihat pada tabel routing dan menentukan hop selanjutnya bagi paket tersebut. Setelah itu, paket tersebut akan diantrikan untuk ditransmisikan pada *interface output*. Di titik inilah fungsi dari pengontrolan trafik pada Linux akan diterapkan. Pengontrolan trafik pada Linux dapat digunakan untuk membangun kombinasi yang kompleks dari disiplin antrian, kelas-kelas, dan filter-filter yang akan mengontrol paket-paket yang dikirimkan pada interface output.

Classful Queueing Discipline

Classful Queueing Discipline merupakan suatu disiplin antrian yang akan membagi trafik berdasarkan kelas-kelas. Classful qdisc sangat berguna apabila kita memiliki traffik yang berbeda-beda yang harus memiliki pembedaan penanganan. Ketika traffik memasuki suatu classful queueing discipline, maka paket tersebut akan dikirimkan ke kelas-kelas di dalam qdisc, dengan kata lain paket tersebut perlu diklasifikasikan terlebih dahulu. Untuk menentukan apa yang harus dilakukan dengan sebuah paket yang datang, maka filter-filter akan digunakan. Filter-filter yang terdapat pada qdisc tersebut akan menghasilkan suatu keputusan, dan qdisc akan menggunakan hasil ini untuk mengantrikan paket ke salah satu kelas yang telah tersedia.

Di samping memiliki suatu qdisc, kebanyakan dari classful qdisc juga menerapkan fungsi shaping. Ini sangat berguna untuk melakukan penjadwalan paket

(misal dengan SFQ) dan pengontrolan rate sekaligus. Kita akan sangat membutuhkan proses ini apabila kita memiliki suatu interface dengan kecepatan tinggi (misal ethernet) ke suatu alat yang lebih lambat (misal modem). Beberapa contoh classful queueing discipline di Linux adalah:

a. CBQ (Class Based Queueing)

Class-Based Queuing (CBQ) adalah suatu mekanisme penjadwalan, bertujuan menyediakan *Link sharing* antar agensi yang menggunakan jalur fisik yang sama, dan sebagai acuan untuk membedakan trafik yang memiliki prioritas-prioritas yang berlainan. Dengan CBQ, setiap agensi dapat mengalokasikan *bandwidth* miliknya untuk berbagai jenis trafik yang berbeda, sesuai dengan pembagiannya yang tepat untuk masing-masing trafik. CBQ berinteraksi dengan *Link sharing* memberikan keunggulan yaitu pemberian *bandwidth* yang tak terpakai bagi *leaf class*nya sebelum diberikan kepada agensi-agensi lain.

b. HTB (Hierarchical Token Bucket)

Hierarchical Token Bucket (HTB) merupakan teknik penjadwalan paket yang baru-baru ini diperkenalkan bagi router berbasis Linux, dikembangkan pertama kali oleh Martin Devera pada akhir 2001 untuk diproyeksikan sebagai pilihan (atau pengganti) mekanisme penjadwalan yang saat ini masih banyak dipakai yaitu CBQ. HTB diklaim menawarkan kemudahan pemakaian dengan teknik peminjaman dan implementasi pembagian trafik yang lebih akurat.

Pada HTB terdapat parameter *ceil* sehingga kelas akan selalu mendapatkan *bandwidth* di antara base *link* dan nilai *ceil link*nya. 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 *link* ini (keduanya *leaves* dan interior dapat memiliki *ceil*). Sebagai catatan, apabila nilai *ceil* sama dengan nilai *base link*, 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*

2. Cara Penelitian

2.1 Alat dan Bahan Penelitian

Perangkat Keras Sistem

Perangkat keras yang digunakan untuk mensimulasikan pengimplementasian CBQ dan HTB pada tugas akhir ini mempunyai spesifikasi sebagai berikut :

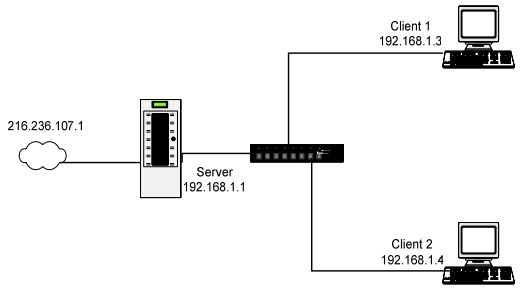
a) Server, dengan spesifikasi PIII processor 800 Mhz, RAM 256 MB, fast NIC 100 Mbps.
b) 2 buah PC sebagai client, dengan spesifikasi P133, RAM 32 MB, dan NIC 10 Mbps dan NoteBook Centrino 1.5 Ghz, RAM 512 MB dan NIC 10/100.
c) 8 port hub 10/100Mbps.
d) Kabel UTP category 5 sebagai media transmisi untuk hubungan dari server ke client.

Sistem Perangkat Lunak

Perangkat lunak yang digunakan pada implementasi ini adalah sebagai berikut:

a) Linux Redhat 2.4.20, yaitu sistem operasi berbasis *Unix*, di mana CBQ dan HTB dan software pengukur performansi jaringan berjalan di atasnya.
b) *iproute2*, yang berisikan software untuk merouting dan mengatur bandwidth.
c) *Iperf*, merupakan perangkat lunak yang digunakan untuk mengukur performansi jaringan.
d) MS office, yaitu perangkat lunak yg digunakan untuk pemrosesan text. Dalam tugas akhir ini penulis menggunakan perangkat lunak ini untuk memproses output Iperf agar dapat dimasukkan dalam sebuah spreadsheet.
e) Ping, yaitu perangkat lunak yang digunakan untuk mengukur response time.

2.2. Skenario dengan menggunakan PC Router



Gambar 2 Koneksi *client 1* dan *client 2* dengan menggunakan PC Router

Pada simulasi ini akan ditunjukkan bagaimana cara membagi kapasitas *bandwidth* dengan topologi yang menggunakan PC Router berdasarkan alamat IP dari client yang terhubung dalam suatu jaringan.

Pembagian *bandwidth* dibedakan pengklasifikasian kelasnya berdasarkan alamat IP. Kelas 1 (semua trafik yang menuju IP 192.168.1.3) dengan alokasi *bandwidth* 330 kbps dan kelas 2 (semua trafik yang menuju IP 192.168.1.4) alokasi *bandwidth* 192 kbps .

a. Konfigurasi CBQ yang akan digunakan adalah sebagai berikut:

Script CBQ :

```
/etc/sysconfig/cbq
[root@server cbq]# cat cbq-2.iperf
DEVICE=eth1,10Mbit,1Mbit
RATE=330Kbit
WEIGHT=33.0Kbit
PRIO=1
RULE=192.168.1.3

[root@server cbq]# cat cbq-5.iperf
DEVICE=eth1,10Mbit,1Mbit
RATE=192Kbit
WEIGHT=19.2Kbit
PRIO=2
RULE=192.168.1.4
```

Untuk menjalankan CBQ :

```
[root@server cbq]#/etc/init.d/cbq-init start
[root@server cbq]#tc –s qdisc
```

b. Konfigurasi HTB yang akan digunakan adalah sebagai berikut:

Test pengiriman paket dgn *bandwidth* 1 MB

Script htb :

```
[root@server rc.d]# cat htb
tc qdisc add dev eth1 root handle 1: htb
tc class add dev eth1 parent 1: classid 1:1 htb rate
554kbit
tc class add dev eth1 parent 1:1 classid 1:10 htb rate
330kbit
tc class add dev eth1 parent 1:1 classid 1:11 htb rate
192kbit
#host 1.4 192kbit
tc filter add dev eth1 parent 1: protocol ip u32 match ip
dst 192.168.1.4 flowid 1:11
#host 1.3 330kbit
```

```
tc filter add dev eth1 parent 1: protocol ip u32 match ip
dst 192.168.1.3 flowid 1:10
```

Untuk menjalankan HTB :

```
[root@server rc.d]#/etc/rc.d/htb
```

3. Hasil Penelitian

Hasil IPerf IP 192.168.1.3 CBQ
(*Bandwidth* 1 Mbps)

```
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
-------
[144] local 192.168.1.3 port 5001 connected with 192.168.1.1
port 32773
[ ID] Interval      Transfer     Bandwidth      Jitter
Lost/Total Datagrams
[144]  0.0- 1.0 sec 44.5 KBytes  365 Kbits/sec  26.259 ms  21/
52 (40%)
[144]  1.0- 2.0 sec 40.2 KBytes  329 Kbits/sec  32.607 ms  57/
85 (67%)
[144]  2.0- 3.0 sec 40.2 KBytes  329 Kbits/sec  36.010 ms  58/
86 (67%)
[144]  3.0- 4.0 sec 40.2 KBytes  329 Kbits/sec  34.819 ms  57/
85 (67%)
[144]  4.0- 5.0 sec 40.2 KBytes  329 Kbits/sec  36.217 ms  58/
86 (67%)
[144]  5.0- 6.0 sec 40.2 KBytes  329 Kbits/sec  31.833 ms  57/
85 (67%)
[144]  6.0- 7.0 sec 38.8 KBytes  318 Kbits/sec  32.475 ms  57/
84 (68%)
[144]  7.0- 8.0 sec 40.2 KBytes  329 Kbits/sec  30.195 ms  57/
85 (67%)
[144]  8.0- 9.0 sec 40.2 KBytes  329 Kbits/sec  9.824 ms   57/
85 (67%)
[144]  9.0-10.0 sec 40.2 KBytes  329 Kbits/sec  7.007 ms   58/
86 (67%)
[144]  0.0-10.4 sec 421 KBytes   332 Kbits/sec  7.214 ms  559/
852 (66%)
------------------------------------------------------------
---------
```

Hasil IPerf IP 192.168.1.3 HTB

```
Server listening on UDP port 5001
Receiving 1470 byte datagrams
```

```
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
---------
[144] local 192.168.1.3 port 5001 connected with 192.168.1.1
port 32780
[ ID] Interval       Transfer     Bandwidth       Jitter
Lost/Total Datagrams
[144]  0.0- 1.0 sec 41.6 KBytes   341 Kbits/sec  28.390 ms   0/
29 (0%)
[144]  1.0- 2.0 sec 40.2 KBytes   329 Kbits/sec  33.103 ms   0/
28 (0%)
[144]  2.0- 3.0 sec 40.2 KBytes   329 Kbits/sec  35.041 ms   0/
28 (0%)
[144]  3.0- 4.0 sec 40.2 KBytes   329 Kbits/sec  35.086 ms   0/
28 (0%)
[144]  4.0- 5.0 sec 40.2 KBytes   329 Kbits/sec  59.270 ms   0/
28 (0%)
[144]  5.0- 6.0 sec 40.2 KBytes   329 Kbits/sec  92.019 ms   0/
28 (0%)
[144]  6.0- 7.0 sec 40.2 KBytes   329 Kbits/sec  43.664 ms   0/
28 (0%)
[144]  7.0- 8.0 sec 40.2 KBytes   329 Kbits/sec  55.216 ms   0/
28 (0%)
[144]  8.0- 9.0 sec 40.2 KBytes   329 Kbits/sec  79.650 ms   0/
28 (0%)
[144]  9.0-10.0 sec 40.2 KBytes   329 Kbits/sec  42.592 ms   0/
28 (0%)
[144] 10.0-11.0 sec 40.2 KBytes   329 Kbits/sec  76.074 ms   0/
28 (0%)
[144] 11.0-12.0 sec 40.2 KBytes   329 Kbits/sec  129.856 ms  0/
28 (0%)
[144]  0.0-12.1 sec 485 KBytes   329 Kbits/sec  126.730 ms   0/
338 (0%)
------------------------------------------------------------
---------
```

Hasil IPerf IP 192.168.1.4 CBQ

```
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
---------
[140] local 192.168.1.4 port 5001 connected with 192.168.1.1
port 32772
[ ID] Interval       Transfer     Bandwidth       Jitter
Lost/Total Datagrams
[140]  0.0- 1.0 sec  31.6 KBytes  259 Kbits/sec  23.279 ms  11/
33 (33%)
```

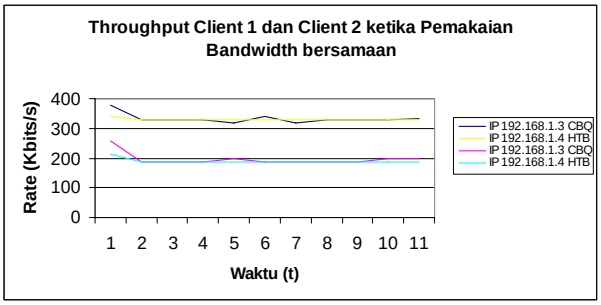
```
[140]  1.0- 2.0 sec  23.0 KBytes  188 Kbits/sec  12.416 ms  68/
84 (81%)
[140]  2.0- 3.0 sec  23.0 KBytes  188 Kbits/sec  9.022 ms   67/
83 (81%)
[140]  3.0- 4.0 sec  23.0 KBytes  188 Kbits/sec  31.744 ms  66/
82 (80%)
[140]  4.0- 5.0 sec  24.4 KBytes  200 Kbits/sec  39.523 ms  69/
86 (80%)
[140]  5.0- 6.0 sec  23.0 KBytes  188 Kbits/sec  45.163 ms  69/
85 (81%)
[140]  6.0- 7.0 sec  23.0 KBytes  188 Kbits/sec  42.336 ms  68/
84 (81%)
[140]  7.0- 8.0 sec  23.0 KBytes  188 Kbits/sec  44.148 ms  69/
85 (81%)
[140]  8.0- 9.0 sec  23.0 KBytes  188 Kbits/sec  41.713 ms  69/
85 (81%)
[140]  9.0-10.0 sec  24.4 KBytes  200 Kbits/sec  44.302 ms  69/
86 (80%)
[140]  0.0-10.6 sec   256 KBytes  197 Kbits/sec  42.464 ms
666/844 (79%)
------------------------------------------------------------
-------
```

Hasil IPerf IP 192.168.1.4 HTB

```
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
--------
[140] local 192.168.1.4 port 5001 connected with 192.168.1.1
port 32779
[ ID] Interval       Transfer     Bandwidth       Jitter
Lost/Total Datagrams
[140]  0.0- 1.0 sec  25.8 KBytes   212 Kbits/sec  38.277 ms
0/ 18 (0%)
[140]  1.0- 2.0 sec  23.0 KBytes   188 Kbits/sec  51.633 ms
0/ 16 (0%)
[140]  2.0- 3.0 sec  23.0 KBytes   188 Kbits/sec  57.526 ms
0/ 16 (0%)
[140]  3.0- 4.0 sec  23.0 KBytes   188 Kbits/sec  58.389 ms
0/ 16 (0%)
[140]  4.0- 5.0 sec  23.0 KBytes   188 Kbits/sec  58.694 ms
0/ 16 (0%)
[140]  5.0- 6.0 sec  23.0 KBytes   188 Kbits/sec  59.711 ms
0/ 16 (0%)
[140]  6.0- 7.0 sec  23.0 KBytes   188 Kbits/sec  120.861 ms
0/ 16 (0%)
```

```
[140]  7.0- 8.0 sec  23.0 KBytes   188 Kbits/sec  68.249 ms
0/ 16 (0%)
[140]  8.0- 9.0 sec  23.0 KBytes   188 Kbits/sec  158.099 ms
0/ 16 (0%)
[140]  9.0-10.0 sec  23.0 KBytes   188 Kbits/sec  94.754 ms
0/ 16 (0%)
[140] 10.0-11.0 sec  23.0 KBytes   188 Kbits/sec  227.226 ms
0/ 16 (0%)
[140] 11.0-12.0 sec  23.0 KBytes   188 Kbits/sec  118.936 ms
0/ 16 (0%)
[140] 12.0-13.0 sec  23.0 KBytes   188 Kbits/sec  81.408 ms
0/ 16 (0%)
[140]  0.0-13.6 sec   317 KBytes   192 Kbits/sec  220.323 ms
0/221 (0%)
-------------------------------------------------------
-------
```

Gambar di bawah ini menunjukkan bagaimana throughput ketika client1 & client2 memakai *bandwidth* penuh secara bersamaan setelah diterapkannya manajemen *bandwidth* CBQ dan HTB .



Gambar 3. *Throughput client 1* dan 2 dengan manajemen *bandwidth* ketika pemakaian *bandwidth* bersamaan pada Bw 1 Mbps

Throughput pada IP 192.168.1.4 untuk CBQ maupun HTB mematuhi *rule* pembatasan *bandwidth* yaitu sebesar 330 kbps. Untuk CBQ didapatkan hasil yang melanggar *rule* dengan kebocoran sebesar 8 kbit sedangkan untuk HTB tidak ada kebocoran.

Jitter

Secara umum *jitter* yang diperoleh CBQ maupun HTB tidak ada yang saling mengungguli hanya saja memiliki kesamaan dengan skenario topologi menggunakan Router, hasil dari selisih jitter CBQ memiliki perbedaan selisih yang lebih baik dibandingkan dengan HTB.

Loss Datagram

Loss datagram CBQ lebih buruk dibandingkan HTB. Pesan yang dikirim server melalui manajer *bandwidth* HTB diterima seluruh oleh klien, berarti dalam skenario menggunakan Router yaitu tidak terjadi loss datagram.

Analisis Statistik

Teknik analisis statistik digunakan untuk menguji apakah terdapat perbedaan yang signifikan antara suatu himpunan data (misalnya: *rate* pada IP 192.168.1.3 CBQ) dengan suatu nilai tertentu yang dianggap sebagai standar (misalnya: *bandwidth* 330 Kbps) menggunakan uji-t satu sampel (*one-sample t-test*).

Rumusan hipotesis nol (H_0) dan hipotesis alternatif (H_a) untuk pengujian *one-sample t-tets* adalah sebagai berikut:

H_0 : $\mu > 330$: *Rate* pada IP 192.168.1.3 CBQ lebih besar atau sama dengan 330 Kbps

H_a : $\mu \leq 330$: *Rate* pada IP 192.168.1.3 CBQ lebih kecil dari 330 Kbps

Berdasarkan hipotesis di atas maka kriteria pengambilan keputusannya adalah sebagai berikut:

Bila $t_{value} \leq t_{tabel}$ berarti tolak H_0 dan sebaliknya terima H_a

Bila $t_{value} > t_{tabel}$ berarti terima H_0 dan sebaliknya tolak H_a

Dimana t_{value} diperoleh dari hasil hitungan menggunakan *SPSS for Windows Release 11.5*, sedangkan t_{tabel} diperoleh pada tabel statistik distribusi t dengan derajat bebas (df) tertentu pada taraf signifikansi 0,05 (*one-tailed test*).

- Batasan *bandwidth* sebesar 330 Kbps

Deskripsi data berupa *mean* (rata-rata) *rate* (kbits/s) masing-masing pada IP 192.168.1.3 CBQ dan IP 192.168.1.3 HTB disajikan pada tabel berikut:

Tabel 1
Statistik Deskriptif

IP	N	*Mean* *Rate* (kbits/s)
IP 192.168.1.3 CBQ	11	331,55
IP 192.168.1.3 HTB	13	329,92

Pada tabel 1 di atas dapat dilihat bahwa pada IP 192.168.1.3 CBQ memiliki *mean rate* (kbits/s) yang lebih besar dari batasan *bandwidth* yaitu sebesar 330 Kbps, sedangkan pada IP 192.168.1.3 HTB memiliki *mean rate* (kbits/s) yang lebih kecil dari batasan *bandwidth*. Dan perbandingan di antara keduanya menunjukkan bahwa IP 192.168.1.3 HTB memiliki *mean rate* (kbits/s) yang lebih kecil dibanding pada IP 192.168.1.3 CBQ.

Untuk menguji apakah *rate* (kbits/s) pada masing-masing IP 192.168.1.3 CBQ dan IP 192.168.1.3 HTB secara statistik lebih kecil dari batasan *bandwidth* yaitu sebesar 330 Kbps, maka dilakukan pengujian *one-sample t-test*.

Hasil analisis dengan SPSS disajikan pada tabel berikut (perhitungan terlampir):

Tabel 2.
Hasil Analisis *One-Sample T-Test*

IP	*Mean*	*Diff.*	t_{value}	df	t_{tabel}	Keputusan
IP 192.168.1.3 CBQ	331,55	1,55	0,440	10	-1,812	H_0 diterima
IP 192.168.1.3 HTB	329,92	-0,08	-0,083	12	-1,782	H_0 diterima

Berdasarkan tabel 2. di atas, pengujian pada IP 192.168.1.3 CBQ diperoleh perbedaan *rate* (kbits/s) dengan batasan *bandwidth* sebesar 1,55 Kbps (positif). Tanda positif berarti bahwa lebih besar dari batasan *bandwidth*. Uji signifikansi perbedaan ini dengan statistik t diperoleh t_{value} sebesar 0,440. Dengan df = 10 dan taraf signifikansi 0,05 (*one-tailed test*) diperoleh t_{tabel} sebesar -1,812. Karena t_{value} = 0,440 lebih besar dari t_{tabel} = -1,812 berarti terima H_0 dan sebaliknya tolak H_a. Jadi *rate* (kbits/s) pada IP 192.168.1.3 CBQ secara statistik tidak lebih kecil dari batasan *bandwidth* yaitu sebesar 330 Kbps.

Pengujian pada IP 192.168.1.3 CBQ diperoleh perbedaan *rate* (kbits/s) dengan batasan *bandwidth* sebesar 0,08 Kbps (negatif). Tanda negatif berarti bahwa lebih kecil dari batasan *bandwidth*. Uji signifikansi perbedaan ini dengan statistik t diperoleh t_{value} sebesar -0,083. Dengan df = 12 dan taraf signifikansi 0,05 (*one-tailed test*) diperoleh t_{tabel} sebesar -1,782. Karena t_{value} = -0,083 lebih besar dari t_{tabel} = -1,782 berarti terima H_0 dan sebaliknya tolak H_a. Jadi *rate* (kbits/s) pada IP 192.168.1.3 HTB secara statistik tidak lebih kecil dari batasan *bandwidth* yaitu sebesar 330 Kbps.

- Batasan *bandwidth* sebesar 192 Kbps

Deskripsi data berupa *mean* (rata-rata) *rate* (kbits/s) masing-masing pada IP 192.168.1.3 CBQ dan IP 192.168.1.3 HTB disajikan pada tabel berikut:

Tabel 3.
Statistik Deskriptif

IP	N	*Mean Rate* (kbits/s)
IP 192.168.1.4 CBQ	11	197,45
IP 192.168.1.4 HTB	14	190,00

Pada tabel 3. di atas dapat dilihat bahwa pada IP 192.168.1.4 CBQ memiliki *mean rate* (kbits/s) yang lebih besar dari batasan *bandwidth* yaitu sebesar 192 Kbps, sedangkan pada IP 192.168.1.4 HTB memiliki *mean rate* (kbits/s) yang lebih kecil dari batasan *bandwidth*. Dan perbandingan di antara keduanya menunjukkan bahwa IP 192.168.1.4 HTB memiliki *mean rate* (kbits/s) yang lebih kecil dibanding pada IP 192.168.1.4 CBQ.

Untuk menguji apakah *rate* (kbits/s) pada masing-masing IP 192.168.1.4 CBQ dan IP 192.168.1.4 HTB secara statistik lebih kecil dari batasan *bandwidth* yaitu sebesar 192 Kbps, maka dilakukan pengujian *one-sample t-test*.

Hasil analisis dengan SPSS disajikan pada tabel berikut (perhitungan terlampir):

Tabel 4.
Hasil Analisis *One-Sample T-Test*

IP	*Mean*	*Diff.*	t_{value}	df	t_{tabel}	Keputusan
IP 192.168.1.4 CBQ	197,45	5,45	0,860	10	-1,812	H_0 diterima
IP 192.168.1.4 HTB	190,00	-2,00	-1,165	13	-1,771	H_0 diterima

Berdasarkan tabel 4. di atas, pengujian pada IP 192.168.1.4 CBQ diperoleh perbedaan *rate* (kbits/s) dengan batasan *bandwidth* sebesar 5,45 Kbps (positif). Tanda positif berarti bahwa lebih besar dari batasan *bandwidth*. Uji signifikansi perbedaan ini dengan statistik t diperoleh t_{value} sebesar 0,860. Dengan df = 10 dan taraf signifikansi 0,05 (*one-tailed test*) diperoleh t_{tabel} sebesar -1,812. Karena t_{value} = 0,860 lebih besar dari t_{tabel} = -1,812 berarti terima H_0 dan sebaliknya tolak H_a. Jadi *rate* (kbits/s) pada IP 192.168.1.4 CBQ secara statistik tidak lebih kecil dari batasan *bandwidth* yaitu sebesar 192 Kbps.

Pengujian pada IP 192.168.1.4 CBQ diperoleh perbedaan *rate* (kbits/s) dengan batasan *bandwidth* sebesar 2,00 Kbps (negatif). Tanda negatif berarti bahwa lebih kecil dari batasan *bandwidth*. Uji signifikansi perbedaan ini dengan statistik t

diperoleh t_{value} sebesar -1,165. Dengan df = 13 dan taraf signifikansi 0,05 (*one-tailed test*) diperoleh t_{tabel} sebesar -1,771. Karena t_{value} = -1,165 lebih besar dari t_{tabel} = -1,771 berarti terima H_0 dan sebaliknya tolak H_a. Jadi *rate* (kbits/s) pada IP 192.168.1.4 HTB secara statistik tidak lebih kecil dari batasan *bandwidth* yaitu sebesar 192 Kbps.

DAFTAR PUSTAKA

Ajay Tirumaladul, Feng Qin, Jon Dugan, Jim Ferguson dan Kevin Gibbs, 2003, Iperf *Version 1.7.0,* http://dast.nlanr.net/Projects/Iperf/

Devera, M., 2001, *HTB Manual Page*, http://luxik.cdi.cz/~devik/qos/htb

Forouzan (1), Behrouz A.2003, *Local Area Networks,* First Edition, New York, McGraw-Hill.

Forouzan (2), Behrouz A., *TCP/IP Protocol Suite,* Second Edition, New York, McGraw-Hill Higher Education. Copyrigh, 2003

Anonim , 2004, Kernel 2.4 : bridging, http://bridge.sourceforge.net/bridge-utils-.html