ANALISIS MANAJEMEN BANDWITDH DENGAN MENGGUNAKAN METODE PCQ (PRE CONNECTION QUEUE) DI AVNAHNET GAME CENTER

Oleh

MUH. ARIEF RINANDA

T31.12.188

SKRIPSI

PROGRAM SARJANA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS ICHSAN GORONTALO
GORONTALO
2017

HALAMAN PERSETUJUAN

ANALISIS MANAJEMEN BANDWITDH DENGAN MENGGUNAKAN METODE PCQ (PRE CONNECTION QUEUE) DI AVNAHNET GAME CENTER

Oleh

MUH. ARIEF RINANDA

T31.12.188

SKRIPSI

Untuk memenuhi salah satu syarat ujian
guna memperoleh gelar Sarjana
Program Studi Teknik Informatika, ini
telah disetujui oleh Tim Pembimbing

Gorontalo, April 2017

Pembimbing Utama	Pembimbing Pendamping
Budi Santoso, S.Kom, M.Eng	**Yusrianto Malago, S.Kom, M.Kom**
NIDN. 0908048403	**NIDN.9909108501**

LEMBAR PENGESAHAN

ANALISIS MANAJEMEN BANDWITDH DENGAN MENGGUNAKAN METODE PCQ (PRE CONNECTION QUEUE) DI AVNAHNET GAME CENTER

Oleh

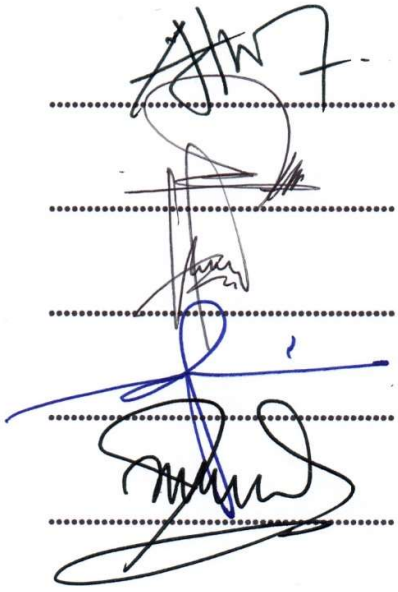
MUH. ARIEF RINANDA

T31.12.188

Diperiksa dan Panitia Ujian Strata Satu (S1)

Universitas Ichsan Gorontalo

1. Ketua Pengují
   Yasin Aril Mustofa, M.Kom .....
2. Anggota
   Sunarto Taliki, M.Kom .....
3. Anggota
   Warid Yunus, M.Kom .....
4. Anggota
   Budi Santoso, S.Kom, M.Eng .....
5. Anggota
   Yusrianto Malago, S.Kom, M.Kom .....



HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa :

1. Karya Tulis (Skripsi) saya ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik (Sarjana) baik di Universitas Ichsan Gorontalo maupun di Perguruan Tinggi lainnya.
2. Karya tulis ini adalah murni gagasan, rumusan dan penelitian saya sendiri tanpa bantuan pihak lain, kecuali arahan dari tim pembimbing.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat yang telah dipublikasikan orang lain, kecuali secara tertulis dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.
4. Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini , maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi ini.

Gorontalo, April 2017
Yang Membuat Pernyataan,

MUH. ARIEF RINANDA
T31.12.188

ABSTRACT

Nowadays, the most crucial problem on every game center was the bandwidth usage which is not used optimally. It was because of there is one or more users that used the network to download or to access applications that increased the capacity of bandwidth usage and disturbed another user. Therefore, it was very important to manage bandwidth usage on computer network. One of them is to apply PCQ (Peer Connection Queue) method, this method is introduced to expand and optimize the network queue. PCQ method used the algorithm which is useful to divide the incoming data stream and distinguish every stream based on their parameter DST-Address, SCR-Address, DST-Port or SRC-Port. It also used to add a proxy server in a device to support the stability of the internet connection. The procedure of the method was saved the web caches that were already accessed before, therefore it will not using the available bandwidth on the second access.

Based on the research, it was studied that PCQ method and additional proxy server at game centre will manage the bandwidth usage on every client as equally. It was already proven that PCQ method was able to manage 10Mbps available bandwidth became two quota; 8Mbps for download and 2Mbps for upload. The 8Mbps bandwidth also divided into 5Mbps for browsing and 3Mbps for game access. Thus gaming activity will not lagging when another client do a download access. It was because of the game port for online games connection was not parallelized with the other clients that were used the network to browse or to download.

The second thing is the additional of proxy server function as the web cache storage, therefore the client that only do browsing access will not disturbed by the download and game activity from another clients. Likewise, the packet loss will not be the problem because of many users on one network with a different access. The configuration was already divided and run automatically to every active users. This thing sure more eligible and effective way to use, compared with some game centre that applied another method without manage the bandwidth usage.

***Keywords**: Networks, Bandwitdh Management, Per Connection Queue, Proxy server.*

ABSTRAK

Penggunaan *bandwitdh* di sebuah jaringan seringkali kurang dimanfaatkan secara optimal. Hal ini dapat disebabkan oleh adanya satu atau lebih *user* yang menghabiskan kapasitas *bandwitdh* dalam jaringan tersebut untuk men-*download* atau untuk mengakses aplikasi-aplikasi yang dapat menyita kapasitas *bandwitdh.* Oleh karena itu manajemen penggunaan *bandwitdh* pada suatu jaringan komputer sangat diperlukan. Salah satunya dengan penerapan Metode *PCQ (Per Connection Queue)*. Metode PCQ diperkenalkan untuk mengoptimalkan sistem antrian yang sangat besar. Algoritma *PCQ* akan mengelompokkan aliran data masuk dan membedakan setiap aliran berdasarkan parameter *dst-address, src-address, dst-port* atau *src-port* dan penambahan sebuah *proxy* server untuk *device* penunjang kestabilan koneksi internet. Cara kerjanya yaitu menyimpan *web cache* yang sudah terakses sebelumnya sehingga saat akan diakses kembali tidak akan menggunakan *bandwitdh* yang sudah tersedia.

Berdasarkan hasil penelitian diperoleh bahwa metode PCQ dan penambahan proxy server pada suatu warnet/ game center dapat melakukan manajemen *bandwitdh* secara baik dan merata kepada setiap *client*. Hal ini terbukti karena metode PCQ dapat melakukan manajemen dimana bandwitdh 10Mbps yang tersedia dibagi menjadi dua kuota yaitu 8Mbps *download* dan 2Mbps untuk *upload*. Serta 8Mbps yang dibagi menjadi 2 kuota yaitu 5Mbps untuk *browsing* dan 3Mbps untuk akses *game*. Sehingga aktifikas gaming tidak akan terganggu ketika ada akses *download* dari *client* lain. Penandaan pada *port game* untuk koneksi *game online* tersebut tidak tercampur aduk dengan *client* yang hanya melakukan *browsing* dan *downloading*. Serta *proxy server* yang berfungsi sebagai penyimpan *web cache* maka *client* yang tidak melakukan aktifitas *downloading* dan *gaming* pun tidak akan terganggu saat melakukan *browsing*. Begitupun dengan *packet lost* yang terjadi tidak terlalu bermasalah karena padatnya pengguna. Konfigurasi yang dilakukan tersebut telah dibagi dan dijalankan secara otomatis kepada setiap *client* yang aktif. Hal ini tentunya lebih baik dibandingkan beberapa warnet yang menerapkan metode lain yaitu tanpa melakukan manajemen kuota *bandwitdh,* sehinnga dalam pemakainannya tersebut menjadi tidak merata.

Kata Kunci : Jaringan, Manajemen *Bandwitdh*, *Per Connection Queue*, *Proxy Server.*

KATA PENGANTAR

Bismillahirrahmanirrahim

Segala Puji bagi Allah SWT karena dengan Taufiq dan Hidayah-Nya lah sehingga Usulan Penelitian ini dapat terselesaikan tepat pada waktunya. Shalawat serta Salam kepada junjungan kita Nabi besar Muhammad SAW yang telah membawa umatnya dari alam kegelapan menuju alam terang benderang.

Penulis menyadari bahwa dalam penulisan Skripsi ini masih jauh dari kesempurnaan. Oleh karena itu dengan segala kerendahan hati penulis sangat mengharapkan kritik dan saran yang sifatnya membangun guna perbaikan dan penyempurnaannya.

Pada kesempatan yang sangat berharga ini penulis haturkan ucapan terima kasih yang setinggi-tingginya kepada :

1. Ibunda tercinta Ramlawati Usman yang telah mengorbankan waktu dan materi juga do'a yang tidak pernah putus untuk saya dapat menyelesaikan program studi sarjana saya.
2. Ibu Dr. Hj. Juriko Abdussamad, M.Si selaku Ketua Yayasan Pengembangan Ilmu Pengetahuan dan Teknologi (YPIPT) Ichsan Gorontalo.
3. Bapak Dr. Abdul Gaffar La Tjokke, M.Si selaku Rektor Universitas Ichsan Gorontalo.
4. Ibu Zohrahayaty, S.Kom, M.Kom selaku Dekan Fakultas Ilmu Komputer.
5. Ibu Asmaul Husna, S.Kom, M.Kom, selaku Wakil Dekan I Bidang Akademik.
6. Ibu Irma Surya Kumala Idris, S.Kom, M.Kom, selaku Wakil Dekan II Bidang Administrasi Umum dan Keuangan.
7. Bapak Yasin Aril Mustafa, S.Kom, M.Kom selaku Wakil Dekan III Bidang Kemahasiswaan.
8. Bapak Irvan Abraham Salihi, S.Kom, M.Kom selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer.
9. Bapak Budy Santoso, S.Kom, M.Eng selaku Pembimbing Utama yang telah membimbing penulis selama mengerjakan skripsi ini.

10. Bapak Yusrianto Malago, S.Kom, M.Kom selaku Pembimbing Pendamping yang telah membimbing penulis selama mengerjakan skripsi ini.
11. Bapak dan Ibu Dosen yang telah mendidik dan membimbing penulis dalam mengerjakan usulan penelitian ini.
12. Bapak Toni selaku owner Avnahnet Game Center yang telah mengizinkan peneliti dalam melakukan penelitian.
13. Saudara Saya Kakak Akbar, Saudari Saya Desyandari Ointu, Saudara Bungsu Saya Jelang Fajar, Kakek Saya Muhammad Usman Dan Keluarga Yang Selalu Memberikan Do'a, Dorongan Moral Maupun Materil Dari Awal Hingga Akhir Perkuliahan.
14. Teman-teman Seperjuangan di Jurusan Teknik Informatika terutama teman-teman Fikom Reguler A 2012 (Adi, Agung, Aldy, Cipto, David, Didin, Eko, Fadli, Feby, Jamal, Irvan, Iki Bolsel, Iki Buol, Iki Ternate, Nato, Noval, Rian, Roma, Rudi, Putra, Sandri, Wayan, Ayu, Eby, Isna, Jean, Mita, Monik, Nita, Wati dan Selvi Djafar Rahimahullah) yang telah membantu penulis dalam penyelesaian Skripsi.
15. Teman-teman Halaqoh Muhajirin (Ust Khalid Walid Lc, Ust Abu Usamah, Akh Adin, Akh Agung, Akh Asri, Akh Deri, Akh Dwi, Akh Eza, Akh Fadly, Akh Faisal, Akh Fanris, Akh Haikal, Akh Husain, Akh Ifai, Akh Ilham, Akh Iman Babol, Akh Iman Beda Jaya, Akh Irfan, Akh Rahmat, Akh Rasyid, Akh Syamsul, Akh Uya & Akh Zul) yang telah memberikan dukungan moral dan spiritual kepada penulis.
16. Teman-teman CISC Gorontalo (Ko' Axl Rose, Muchlis Wakdoyok Ambassador, Fadel Si Kembar, Pak Fuad Korwil bersama ibu Korwil, Supriyanto Pangeran Atinggola, Icekahaye Orang Miskin Sombong, Nonu Bebas, Wangko Raja Laut, Dediego Costa, Isnancole, Mell Bluedevil, Apunyulu, Payo, Amad Marching, Alvian Ambulance, Muhar Kemdan Cicut, Valencia Park Sin Hye dan seluruh member yang tidak bisa penulis sebutkan satu persatu) yang telah memberikan *support* dan bantuan kepada penulis.
17. Teman-teman Gorontalo Popo (Pak Oki, Ci' Mey, Pak Riki, Giska, Ahmadi, Ayi, Om Danggu, Om Oppo, Om Insar, Ka Enda, Eko, Peko, Tango, Engki,

Kiki Dukun, Killer, Aan, Ias, M.Kom, Upik, Nanderz, Opin, Wawan Magister Muda, Wahyu, Iren, Iswanda, Om Alo) yang telah memberikan semangat untuk penulisan skripsi ini.

18. Dan terakhir ucapan terima kasih kepada, Nurhayati Ointu S.Ikom (Calon Istri Penulis InsyaAllah) yang sejak awal perkuliahan hingga di penghujung kuliah penulis telah membantu dan memberikan *support* yang tiada hentinya.

Saran dan kritik, penulis harapkan dari dewan penguji dan semua pihak untuk penyempurnaan penulisan skripsi lebih lanjut. Semoga usulan penelitian ini dapat bermanfaat bagi pihak yang berkepentingan.

Gorontalo, April 2017

Penulis

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PERSETUJUAN ii
LEMBAR PENGESAHAN iii
HALAMAN PERNYATAAN iv
ABSTRACT v
ABSTRAK vi
KATA PENGANTAR vii
DAFTAR GAMBAR xiii
DAFTAR TABEL xvi
BAB I PENDAHULUAN 1
1.1 Latar Belakang Masalah 1
1.2 Identifikasi Masalah 3
1.3 Rumusan Masalah 4
1.4 Tujuan Penelitian 4
1.5 Manfaat Penelitian 5
BAB II LANDASAN TEORI 6
2.1 Tinjauan Studi 6
2.2 Tinjauan Pustaka 13
2.2.1 Pengertian Analisa 13
2.2.2 Manajemen *Bandwitdh* 13
2.2.3 *Bandwitdh* 14
2.2.4 TCP/IP 15
2.2.5 ISP (Internet service provider) 18
2.2.6 Mikrotik 19
2.2.7 *Router* 20
2.2.8 *Firewall* 21
2.2.9 *NAT* 22

2.2.10 Topologi Jaringan 24
2.2.11 Per Connection Queue (PCQ) 26
2.2.12 *Proxy Server* 33
BAB III METODE PENELITIAN 36
3.1 Objek dan Jadwal Penelitian 36
3.2 Metode Penelitian 36
3.3 Analisis Hasil 37
3.4 Perancangan Modul Aplikasi 37
3.5 Spesifikasi Kebutuhan Hardware dan Software 38
3.6 Rancangan Umum Sistem 40
3.7 Langkah-Langkah Implementasi 41
BAB IV HASIL PENILITIAN 42
4.1.5 Perancangan Konfigurasi Manajemen *Bandwitdh* dan PCQ 46
4.2 Implementasi Sistem 50
4.2.1 Implementasi Topologi Jaringan 50
BAB V PENGUJIAN, ANALISIS, DAN PEMBAHASAN 73
5.1 Pengujian Dan Analisis 73
5.1.1 Pengujian Browsing 73
5.1.1.1 Pengujian Browsing sebelum penerapan Manajemen *Bandwitdh* .. 73
5.1.1.2 Pengujian Browsing Setelah Penerapan Manajemen *Bandwitdh* Dan Analisisnya 76
5.1.2 Pengujian *download* 78
5.1.2.1 Pengujian *Download* Sebelum Penerapan Manajemen *Bandwitdh* 78
5.1.2.2 Pengujian Download Setelah Menerapkan Manajemen *Bandwitdh* 80
5.1.3 Pengujian Mangle Game 82
5.1.4 Pengujian Proxy Server Squid Lusca 84
5.1.4.1 Pengujian Caching untuk Browsing. 84
5.2 Pembahasan 92
5.2.1 Hasil Pengujian Penerapan *Bandwitdh* 92
5.2.2 Hasil Pengujian *Streaming* Sebelum Penerapan Manajemen *Bandwitdh* 93

5.2.4 Hasil Pengujian *Download* Sebelum Penerapan Manajemen *Bandwitdh* 94
5.2.5 Hasil Pengujian *Download* Setelah Penerapan Manajemen *Bandwitdh* 94
5.2.6 Hasil Pengujian *Mangle Game Online* 96
5.2.7 Hasil Pengujian *Proxy Server* 97
5.2.8 Hasil Pengujian *Packet Loss* 97
BAB VI PENUTUP 98
6.1 Kesimpulan 98
6.2 Saran 99
DAFTAR PUSTAKA 100

DAFTAR GAMBAR

Gambar 2.1. Format *Datagram TCP* 17
Gambar 2.2. Format *datagram IP* 18
Gambar 2.3. *Topologi Bus* 24
Gambar 2.4. *Topologi Ring* 25
Gambar 2.5. *Topologi Star* 26
Gambar 2.6. Cara kerja PCQ 27
Gambar 2.7. Pengelompokkan paket *PCQ classifier* 29
Gambar 2.8. Pengelompokkan sub aliran (*pcq-classifier=src-address*) 30
Gambar 2.9. Pengelompokkan sub aliran (*pcq-classifier=dst-address*) 30
Gambar 2.10. Pengelompokkan suba liran (*pcq-classifier=dst-address, dst-port)* 31
Gambar 2.11. Nilai *pcq-rate=128000* 32
Gambar 2.12. Nilai *pcq-rate=0 (default)* 33
Gambar 2.13. Bagan Kerangka Berpikir 35
Gambar 3.1. Rancangan Sistem Per Connection Queue dengan bantuan Proxy Server (PCQ) 40
Gambar 3.2. Langkah-Langkah Implementasi 41
Gambar 4.1. Topologi Jaringan Umum pada AVNAH Net Game Center ... 42
Gambar 4.2. Topologi Jaringan Lokal Sebelum Penerapan Management Bandwitdh 43
Gambar 4.3. Topologi Jaringan Dengan Menerapkan Management Bandwitdh 44
Gambar 4.4. Setting IP 51
Gambar 4.5. Mikrotik RB750 51
Gambar 4.6. Implementasi Konfigurasi dasar mikrotik..... 52
Gambar 4.7. Remove configuration winbox 52
Gambar 4.8. Tampilan New Terminal 53
Gambar 4.9. Konfigurasi Mangle 55
Gambar 4.10. Konfigurasi Mangle Game Online..... 56

Gambar 4.11. Konfigurasi Mark-Packet dan Mark-Connection 57
Gambar 4.12. Konfigurasi Mangle Umum..... 58
Gambar 4.13. Konfigurasi Mark-Packet dan mark-Connection..... 58
Gambar 4.14. Hasil Konfigurasi Mangle 59
Gambar 4.15. Penambahan konfigurasi PCQ..... 59
Gambar 4.16. Konfigurasi tipe PCQ..... 60
Gambar 4.17. Penentuan *parent Download* 61
Gambar 4.18. Penentuan *Parent Upload* 61
Gambar 4.19. Penentuan *Traffic bandwitdh* PCQ-*Download* pada *packet-browsing* 62
Gambar 4.20. Penentuan *Packet Marks Download* untuk *packet-game* 62
Gambar 4.21. Pembagian *Traffic Bandwitdh upload* pada *packet-browsing* . 63
Gambar 4.22. Penentuan *Packet Marks Upload* pada *packet-game* 63
Gambar 4.23. Hasil pembuatan pcq dengan menggunakan *queue tree* 64
Gambar 4.24. Hasil Konfigurasi Proxy 72
Gambar 5.1. Gambar pengujian pada situs www.speedtest.net 73
Gambar 5.2. Pengujian *buffering time short* video *4k Ultra HD* 1 Client .. 74
Gambar 5.3. Hasil pengujian traffic pada client 1 yang melakukan browsing (*streaming*) 75
Gambar 5.4. Bandwitdh yang di gunakan pada client 1 pada saat melakukan *browsing* (*streaming*) 75
Gambar 5.5. Ping pada *client* 2 pada saat *client* 1 saat melakukan aktifitas *browsing (streaming)* 76
Gambar 5.6. Hasil dari penerapan management *bandwitdh*..... 77
Gambar 5.7. Hasil konfigurasi PCQ yang di kombinasikan dengan *queue tree* pada paket *browsing* 77
Gambar 5.8. Client 1 melakukan *download* 78
Gambar 5.9. Traffic penggunaan *bandwitdh* oleh *client* 1 saat melakukan aktifitas *download* 79
Gambar 5.10. Hasil Ping yang di lakukan oleh *client* lain 79

Gambar 5.11. Hasil penerapan manajemen *bandwitdh* yang hanya digunakan oleh satu *client* pada saat men*download*. 80
Gambar 5.12. Hasil pengujian penerapan manajemen *bandwitdh* 3 *client* men*download* secara bersamaan 81
Gambar 5.13. Hasil manajemen *bandwitdh* dengan menggunakan metode PCQ 81
Gambar 5.14. *Game counter strike online* yang di mainkan oleh *client* 82
Gambar 5.15. Penandaan port (mangle) game counter stike online 82
Gambar 5.16. Hasil traffic dari ke 3 client ketika melakukan download, browsing, dan game online 83
Gambar 5.17. Hasil pengujian paket browser dan paket game 83
Gambar 5.18. *Flowchart* sistem kerja *proxy server* 85
Gambar 5.19. Hasil proses *HIT* dan *MISS* pada Ubuntu server menggunakan *squid lusca* 86
Gambar 5.20. Diagram Pengujian Cahcing pada proxy server 88
Gambar 5.21. Hasil monitoring *packet loss* sebelum penggunaan *proxy* server 91
Gambar 5.22. Hasil monitoring *packet loss* setelah penggunaan *proxy* server 91

DAFTAR TABEL

Tabel 2.1. Penelitian Terkait..... 10
Tabel 3.1. Spesifikasi Hardware..... 38
Tabel 3.2. Spesifikasi Software 39
Tabel 5.1. Pengujian *browsing (streaming)* sebelum penerapan manajemen *Bandwitdh* 75
Tabel 5.2. Pengujian *browsing (streaming)* setelah penerapan manajemen *bandwitdh* 78
Tabel 5.3. Perbandingan waktu mengakses website 79
Tabel 5.4. Hasil pengujian download setelah penerapan manajemen Bandwidth 81
Tabel 5.5. Hasil pengujian *mangle game* dan manajemen *bandwitdh* 84
Tabel 5.6. Tabel Pengolahan Data Standar Deviasi Sebelum Penggunaan *Proxy* 89
Tabel 5.7. Tabel Pengolahan Data Standar Deviasi Sesudah Penggunaan *Proxy* 90

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Perkembangan teknologi informasi yang sangat pesat telah membuat banyak perubahan bagi kehidupan manusia dewasa ini. Hal ini ditandai dengan perkembangan teknologi berbagai perangkat keras maupun lunak yang telah membawa dampak yang cukup besar dalam hal penyajian informasi penyajian informasi menjadi lebih cepat, lebih tepat dan lebih akurat tanpa dibatasi oleh ruang dan waktu. Hampir di setiap perusahaan terdapat jaringan komputer untuk memperlancar arus informasi dalam perusahaan tersebut. *Interconnected Network* atau yang lebih populer dengan sebutan Internet merupakan sistem komunikasi global yang menghubungkan komputer-komputer dan jaringan-jaringan diseluruh dunia.rangkaian yang membentuk internet diawali pada tahun 1969 sebagai *Arpanet* yang dibangun oleh Arpa (*United States Departement Of Defense Advanced Research Project Agency*). (Yulianti, 2015)

Jaringan yang terhubung dengan internet, masalah kecepatan *upload* maupun *download* merupakan hal yang sangat penting untuk memperlancar transmisi data. Banyak hal yang dapat mempengaruhi kecepatan dua proses tersebut, diantaranya yaitu besarnya *bandwitdh* yang digunakan jaringan tersebut dan seberapa efektif *bandwitdh* tersebut bisa dimanfaatkan.

Bandwitdh adalah suatu ukuran dari banyaknya informasi yang dapat mengalir dari satu tempat ke tempat lain dalam satu waktu tertentu. (Tafaul Mujahidin, 2011) (Mujahidin, 2011)

Penggunaan *bandwitdh* di sebuah jaringan seringkali kurang dimanfaatkan secara optimal. Hal ini dapat disebabkan oleh adanya satu atau lebih *user* yang menghabiskan kapasitas *bandwitdh* dalam jaringan tersebut untuk men-*download* atau untuk mengakses aplikasi-aplikasi yang dapat menyita kapasitas *bandwitdh.*

Oleh karena itu manajemen penggunaan *bandwitdh* pada suatu jaringan komputer sangat di perlukan untuk dapat mengontrol dan mengawasi para *user* saat mengakses jaringan komputer yang tersedia. selain itu dengan mengatur *bandwitdh* para user admin dapat mengelola alokasi *bandwitdh* dari berbagai lalu lintas data yang terjadi pada suatu jaringan.

Berdasarkan uraian di atas *manajemen bandwitdh* dapat dilakukan dengan beberapa metode yaitu *Simple Queue* cara pelimitan dengan menggunakan pelimitan sederhana berdasarkan data *rate*. *Per Connection Queue (PCQ)* adalah program untuk mengelola jaringan lalu lintas kualitas layanan *QOS*. Dan *Queue Tree* berfungsi untuk melimit *bandwitdh* pada mikrotik yang mempunyai dua koneksi internet karena paket *mark*nya lebih berfungsi dari pada di *Simple Queues.*

Jadi setelah dilakukan observasi di Avnahnet *Game Center* penggunaan *bandwitdh* tidak teralokasikan secara merata ke setiap *user*, karena tidak terkontrolnya aktifitas lalu lintas data sehingga terjadi kuota yang berlebihan. Dan metode PCQ merupakan metode yang tepat untuk me*manage bandwitdh* secara signifikan dalam kelancaraan penyelesaian membagi kapasitas *bandwitdh* dengan

prinsipnya menggunakan metode antrian untuk menyamakan *bandwitdh* yang dipakai pada multiple client. Jika di bandingkan dengan metode *Simple queue* yang memiliki kekurangan dalam hal mengalokasikan *bandwitdh* khusus buat *ICMP* (*Internet Control Message Protocol*), sehingga apabila pemakaian *bandwitdh* pada klien sudah penuh, *ping time* nya akan naik dan bahkan *RTO* (*request time out*). Serta metode *Queue Tree* yang memiliki kekurangan dapat di tembus *Download Manager* dan konfigurasi yang akan dilakukan cukup rumit, kita harus men-*setting parameter mangle* terlebih dahulu untuk melakukan konfigurasinya. Meskipun metode *Queue Tree* dapat mengalokasikan *bandwitdh* khusus untuk *ICMP (Internet Control Message Protocol).*

Sementara teknologi yang digunakan untuk mengimplementasikanya didasarkan pada pendekatan yang disebut QoS (Quality of Service). Secara default QoS ini berada dalam perangkat mikrotik untuk mengatur traffic jaringan dan lalu lintas data atau informasi.

Dalam penelitian ini peneliti memilih metode *Per Connection Queue* karena dapat menyeimbangkan alokasi *bandwitdh* sesuai kebutuhan *user*. Menerapkan aturan antrian pada lalu lintas data agar tidak terjadi antrian, dan untuk mengelola alokasi *bandwitdh,* dan di tunjang dengan penggunaan proxy server untuk dapat menghemat 30% alokasi *bandwitdh* yang tersedia.

1.2 Identifikasi Masalah

Berdasarkan latar belakang masalah diatas, maka dapat di identifikasi beberapa permasalahan yaitu sebagai berikut :

1. Pengunaan kuota *bandwitdh* yang berlebihan jika tidak di lakukan manajemen *bandwitdh.*
2. Permintaan paket yang berlebihan menggunakan metode *Per Connection Queue.*

1.3 Rumusan Masalah

Berdasarkan identifikasi masalah yang telah di uraikan maka dapat di tentukan rumusan masalah dalam penelitian ini yaitu :

1. Bagaimana mengatasi permintaan paket yang berlebihan dengan menggunakan metode *Per Connection Queue*?
2. Bagaimana mengatasi pembagian *bandwitdh* untuk browsing dan game agar tidak terjadi kegagalan koneksi?

1.4 Tujuan Penelitian

Adapun yang menjadi tujuan pada penelitian ini adalah :

1. Untuk mengalokasikan kuota *bandwitdh* dan lalu lintas alur data pada jaringan yang terpakai.
2. Untuk mengetahui hasil penerapan metode *Per Connection Queue* dalam manajemen *bandwitdh* di Avnahnet *Game Centre.*

1.5 Manfaat Penelitian

1. Manfaat Teoritis

Hasil penelitian ini diharapkan dapat memberikan pemikiran yang berharga bagi pengembangan ilmu pengetahuan yang berkaitan dengan pembelajaran dan praktek tentang manajemen jaringan.

2. Manfaat Praktis

Hasil penelitian ini diharapkan dapat menjadi bahan pertimbangan atau solusi bagi para pengguna komputer di Avnahnet *Game Center* dalam penggunaan *bandwitdh.*

BAB II
LANDASAN TEORI

2.1 Tinjauan Studi

Suatu penelitian yang berkaitan dengan pendekatan dan objek penelitian ini adalah yang dilakukan oleh (Kalsum & Supardi, 2015) berjudul "*Implementasi* Dan Analisa *Per Connection Queue (PCQ)* Sebagai Kontrol Penggunaan Internet Pada Laboratorium Komputer". Untuk memenuhi penggunaan internet, menajemen *bandwitdh* sangat perlu dilakukan untuk mengontrol penggunaan internet itu sendiri. Banyak tipe dari manajemen *bandwitdh* yang ada di berbagai *vendor* yang mengeluarkan perangkat *router*, maupun sistem operasi yang sering digunakan sebagai *router* salah satunya adalah *Per Connection Queue (PCQ). Per connection queue (PCQ)* digunakan sebagai metode *queue* pada jaringan dengan jumlah *client* yang banyak, atau jaringan yang tidak dapat diperkirakan jumlah *clientnya,* misalnya pada sebuah laboratorium komputer yang terhubung dengan internet.

Pada penelitian yang telah di lakukan oleh (Saniya, Priyono, & Ambarwati, 2013) berjudul "Sistem Manajemen *Bandwitdh* Dengan Prioritas Alamat *IP Client*" Pengimplementasian manajemen *bandwitdh* diatur melalui pengalokasian kecepatan *upload* dan *download* pada masing-masing alamat *IP client* secara *sentralisasi* menggunakan *router mikrotik*. Dengan demikian, jika ada *client* yang mengakses internet membutuhkan kapasitas *bandwitdh* yang besar, maka *client* lain tidak akan terganggu, karena masing-masing *client* sudah mempunyai kapasitas *bandwitdh* masing-masing yang dapat dipakai untuk mengakses internet. Pada

penelitian ini akan dilakukan analisis performansi sistem manajemen *bandwitdh* menggunakan metode *HTB* (*Hierarchical Tocken* Bucket) dengan teknik antrian *simple queue* dan *queuetree* terhadap beberapa parameter diantaranya, *packet loss, delay end-to-end,* dan *throughput* sistem.

Penelitian lainya yang berkaitan dengan pendekatan penelitian ini berjudul ”Perbandingan Metode *Simple Queues* Dan *Queues* Tree Untuk Optimasi Manajemen *Bandwitdh* Jaringan Komputer di STMIK PPKIA PRADNYA PARAMITA MALANG” yang dilakukan oleh (Aris Syaifudin, Mahmud Yunus, 2013). Metode *Queues Tree* merupakan metode yang membutuhkan beberapa parameter untuk mengkonfigurasinya, dikarenakan pada metode ini tidak dapat langsung memasukkan *IP Addrees* seperti yang dilakukan pada metode *Simple Queues*. Kelebihan dari Metode *Queues Tree* dapat mengalokasikan *bandwitdh ICMP*. Jadi, ketika *bandwitdh* yang terdapat pada *clien* penuh, *ping time* nya masih dapat stabil. Kekurangan yang terdapat dari menggunakan metode ini adalah *Download Manager* dapat tembus dan konfigurasi yang akan dilakukan cukup rumit, kita harus men-*setting parameter mangle* terlebih dahulu untuk melakukan konfigurasinya. Metode *Simple Queues* merupakan metode yang cukup sederhana yaitu dapat langsung memasukkan *IP Address* pada target yang ingin delimit. Untuk kelebihan yang terdapat pada metode *Simple Queues* adalah tidak dapat ditembus oleh *Download Manager* dan merupakan metode yang cukup sederhana dalam melakukan konfigurasinya. Kekurangan yang terdapat pada metode *Simple Queues* adalah, kita tidak bisa mengalokasikan bandwitdh khusus buat *ICMP* (*Internet*

Control Message Protocol), sehingga apabila pemakaian *bandwitdh* pada klien sudah penuh, *ping time* nya akan naik dan bahkan *RTO* (*request time out*).

Selanjutnya penelitian yang di lakukan oleh (Kosasih, 2008) berjudul "Pengalokasian *Bandwitdh* Secara Otomatis Menggunakan Metode *Per Connection Queue".* Membagi kapasitas *bandwitdh* dengan *PCQ (Per Connection Queue)* prinsipnya menggunakan metode antrian untuk menyamakan *bandwitdh* yang dipakai pada *multiple client.* Metode *PCQ* yang diterapkan menggunakan metode *NDLC (Network Development Life Cycle)* dengan pendekatan *Top-Down*. Sementara teknologi yang digunakan untuk mengimplementasikanya didasarkan pada pendekatan yang disebut *QOS (Quality of Service).* Secara default *QOS* ini berada dalam perangkat mikrotik untuk mengatur traffic jaringan dan lalu lintas data/informasi (Herlambang, 2008). Mikrotik dapat dijalankan pada sebuah *PC (Personal Computer)* biasa atau pada sistem mini routerboard yang bisa berfungsi sebagai *router, bridge, hotspot gateway, firewall, bandwitdh limiter,* dan lain-lain. Cara kerja *QOS* adalah dengan cara mengidentifikasi lalu lintas data yang melalui jaringan, kemudian menerapkan kebijakan *QOS* yang digunakan untuk melindungi, memprioritaskan atau untuk memberikan batasan.Melalui teknik ini keseluruhan kapasitas *bandwitdh* yang tersedia dapat digunakan secara optimal tanpa ada sumberdaya yang terbuang percuma.

Penelitian terakhir yang memiliki perbandingan dengan pendekatan dan objek yang diteliti adalah penelitian berjudul "Optimalisasi Manajemen Jaringan Dengan Menggunakan Mikrotik Router OS" (Purba, 2010). Selain menggunakan mikrotik router manajemen *bandwitdh* juga dapat di tunjang dengan menambahkan

sebuah proxy server. Proxy server adalah sebuah computer server atau program komputer yang dapat bertindak sebagai komputer lainnya untuk melakukan request terhadap content dari internet atau intranet. Proxy server bertindak sebagai gateway terhadap dunia internet untuk setiap komputer client. Proxy server tidak terlihat oleh komputer client, seorang pengguna yang berinteraksi dengan internet melalui sebuah proxy server tidak akan mengetahui bahwa sebuah proxy server sedang menangani request yang dilakukannya. Web Server yang menerima request dari proxy server akan menginterpresikan requestrequest tersebut seolah-olah request itu datang secara langsung dari komputer client, bukan dari proxy server. Proxy server juga dapat digunakan untuk mengamankan jaringan pribadi yang dihubungkan ke sebuah jaringan public (seperti halnya internet). Proxy server memiliki lebih banyak fungsi daripada router yang memiliki fitur packet filtering karena memang proxy server beroperasi pada level yang lebih tinggi dan memiliki control yang lebih menyeluruh terhadap akses jaringan. Proxy server yang berfungsi sebagai sebuah “agen keamanan” untuk sebuah jaringan pribadi, umumnya dikenal sebagai firewall.

Tabel 2.1 Penelitian Terkait

No.	Nama	Judul	Metode	Hasil
1.	Mirsantoso, Toibah Umi Kalsum, Reno Supardi (2015)	*Implementasi* Dan Analisa *Per Connection Queue (PCQ)* Sebagai Kontrol Penggunaan Internet Pada Laboratorium Komputer.	*Per Connection Queue* (PCQ)	Hasil pengujian dan analisa menujukkan bahwa manajemen *bandwitdh* dengan *per connection queue (PCQ)* untuk mengontrol penngunaan internet ini sangat signifikan perubahan yang terjadi. Dengan memanfaatkan cara ini *traffic* data pennggunaan internet dapat diatur sesuai dengan jumlah *bandwitdh* yang ada pada jaringan tersebut. Selain untuk mengontrol penggunaan internet ini, metode ini juga berguna untuk membatasi *traffic rate* download apabila client menggunakan download manager seperti IDM. Dengan menerapkan cara ini, antar *client* satu dan client yang lain tidak akan mendapatkan *bandwitdh* berlebih, sehingga penggunaan internet akan lebih stabil dan dapat dikontrol.
2.	2.Yoga Saniya, Wahyu Adi Priyono**,** Rusmi Ambarwati (2013)	Sistem Manajemen *Bandwitdh* Dengan Prioritas Alamat *IP Client.*	*Hierarchical Tocken Bucket* (HTB)	Hasil perhitungan dan analisis sistem manajemen *bandwitdh* dengan prioritas alamat IP *client* menggunakan teknik antrian *simple queue* dan *queue tree,* maka dapat diperoleh kesimpulan sebagai berikut:

				Sistem manajemen *bandwitdh* dengan prioritas alamat IP *client* menggunakan teknik antrian *simple queue* dan *queue tree* dapat melakukan pembatasan *bandwitdh* dengan baik pada masing-masing *client*
3.	Aris Syaifuddin, Mahmud Yunus, Retno Sundari (2014)	Perbandingan Metode Simple Queues Dan Queues Tree Untuk Optimasi Manajemen Bandwitdh Jaringan Komputer di STMIK PPKIA PRADNYA PARAMITA MALANG	*Simple Queue & Queue Tree*	1. Metode Simple Queues dinilai lebih sederhana dalam proses konfigurasinya, tidak dapat ditembus oleh Download Manager, namun banyak *bandwitdh* yang terbuang. 2. Metode Queues Tree merupakan metode yang bisa dikatakan dapat menggunakan semua *bandwitdh* yang tersedia, namun pada metode ini dapat ditembus oleh Download Manager, dan harus melakukan setting manggle terlebih dahulu. 3. Dari analisa pada perbandingan *bandwitdh* yang telah diujikan, Simple Queues dapat menstabilkan *bandwitdh* daripada Queues Tree yang bergantung pada jumlah user.
4.	Sandy Koasih (2014)	Pengalokasian *Bandwitdh* Secara Otomatis Menggunakan Metode Per	*Per Connection Queue (PCQ)*	Pembagian kapasitas *bandwitdh* dilakukan dengan metode PCQ dilakukan secara merata ke semua pengguna yang menggunakan format IP Address 192.168.40.0/24. Agar kestabilan dan kecepatan transfer data sama dan tidak

		Connection Queue		terjadi tarik menarik *bandwitdh* antar pengguna, dan pengalokasian kapasitas *bandwitdh* dapat menjadi lebih optimal berdasarkan ukuran data dan kapasitas yang ada.
5.	Minda Mora Purba dan Syamsu H (2013)	Optimalisasi Manajemen Jaringan Dengan Menggunakan Mikrotik Router OS	*Simple Queue*	1. Semua jaringan komputer baik yang menggunakan sistem wireless maupun kabel dapat dikelola dengan manajemen Mikrotik RouterOS 2. Dengan menggunakan Mikrotik RouterOS maka pembagian dan pemakaian *bandwitdh* internet menjadi lebih efisiensi. 3. Dengan menggunakan Mikrotik RouterOS maka sistem keamanan jaringan yang dimiliki oleh Mikrotik dapat terfasilitasi dengan baik. 4. Dengan sudah adanya pembagian IP Address, maka penerapan Filter Rule jadi lebih mudah. 5. Pengoperasian RouterOS Mikrotik dengan menggunakan Winbox mempermudah seorang IT Administrator untuk melakukan konfigurasi pada jaringan.

2.2 Tinjauan Pustaka

2.2.1 Pengertian Analisa

Analisa secara umum didefinisikan sebagai suatu usaha dalam mengamati secara detail pada suatu hal atau benda dengan cara menguraikan komponen-komponen pembentuknya atau menyusun komponen tersebut untuk dikaji lebih lanjut. Sedangkan menurut para ahli Gorys Keraf analisa adalah sebuah proses untuk memecahkan sesuatu kedalam bagian-bagian yang saling berkaitan satu sama lainya. Dan menurut Komarrudin mengatakan bahwa analisis merupakan suatu kegiatan berfikir untuk menguraikan suatu keseluruhan menjadi komponen sehingga dapat mengenal tanda-tanda dari setiap komponen, hubungan satu sama lain, dan fungsi masing-masing dalam suatu keseluruhan yang terpadu. Dari beberapa pengertian analisa diatas dapat ditarik kesimpulan bahwa analisa merupakan sekumpulan kegiatan, aktifitas dan proses yang saling berkaitan untuk memecahkan masalah atau memcahkan komponen menjadi lebih detail yang digabungkan kembali lalu ditarik kesimpulannya.

2.2.2 Manajemen *Bandwitdh*

Manajemen berasal dari kata "*to manage"* yang berarti mengatur, mengurus atau mengelola, sedangkan *bandwitdh* adalah besaran yang menunjukkan seberapa banyak data yang dapat dilewatkan dalam koneksi melalui sebuah *network*. Berdasarkan definisi diatas maka manajemen *bandwitdh* dapat diartikan sebagai suatu kegiatan mengatur agar data yang lewat tidak melebihi kapasitas maksimal di dalam sebuah jaringan komputer yang terhubung dengan internet.

Pembagian *bandwitdh*:

1. Yang dibagi jalurnya dengan pembagian kanal (*fix/statis*)
2. Yang diatur datanya dengan metode antrian (dinamis tergantung jumlah data yang lewat) terdiri dari:
    1) prioritas (biasanya internet di perusahaan-perusahan)
    2) identitas (tidak ada persaingan, diterapkan di *Internet Services Provider*)
    3) kelas (perpaduan identitas dan prioritas) biasanya internet di perusahaan

Pembagian *bandwitdh* biasanya 10% *free* dan yang di *share* 90 %, hal ini digunakan untuk mengantisipasi bila suatu saat terjadi lonjakan aliran data di jaringan tersebut. Saat melewati sebuah jaringan, sebuah data bias dipastikan akan mengalami penambahan ukuran karena setelah masuk ke suatu jaringan, *TCP/IP* atau *UDP* akan menambahkan aneka informasi ke dalam suatu *file*. Untuk mengetahui berapa ukuran sebenarnya dari suatu *file* yang lewat di jaringan maka ada yang disebut sebagai *throughput* yakni ukuran data yang sebenarnya (tanpa informasi lain yang ditambahkan oleh *TCP/IP atau UDP)*. (Wikipedia)

2.2.3 *Bandwitdh*

Bandwitdh disebut juga *data transfer* atau s*ite traffic* adalah data yang keluar dan masuk atau *upload* dan *download* ke *account* anda. *Bandwitdh* adalah luas atau lebar cakupan frekuensi yang digunakan oleh sinyal dalam medium transmisi. *Bandwitdh* dapat diartikan sebagai perbedaan antara komponen sinyal frekuensi tinggi dan sinyal frekuensi rendah.

Frekuensi sinyal diukur dalam satuan *Hertz* (Mulyanta, Edi S, S.Si. 2005: 56). *Bandwitdh* yang tidak dibagi secara merata akan mengakibatkan koneksi pada sebagian user (*Client*) menjadi lambat, hal ini terjadi disebabkan sebagian *user* ada

yang memang sedang dalam aktivitas yang menguras *bandwitdh* seperti *browsing* atau *download*, agar trafic menjadi seimbang maka dibutuhkan *bandwitdh* manager pada mikrotik.

2.2.4 TCP/IP

TCP/IP sebuah protocol yang dikembangkan pada tahun 1969 oleh DARPA (*Defence Advanced Research Project Agency)* yang mendanai riset dan pembuatan paket *switching* eksperimental yang diberi nama *ARPANET* (Daryanto, 2010). Protocol ini paling popular dan paling banyak digunakan saat ini, alasanya adalah: *TCP/IP* menggunakan skema pengalamatan fleksibel yang dapat sekali diroute, bahkan untuk network yang paling besar.

1. Hampir semua system operating dan platform dapat menggunakn *TCP/IP.*
2. Sejumlah besar utilitas dan *tool* dapat dipergunakan, sebagianya digabungkan dengan rangakian protocol dan sebagian ditambahkan dalam program untuk memonitoring dan mengatur *TCP/IP*.
3. *TCP/IP* merupakan protocol untik internet global. Sistem harus menjalankan *TCP/IP* untuk berhubunagn dengan internet.
4. Kebanyakan network tingkat interprise menjalankan *TCP/IP*, dan yang penting bahwa administrator network akrab dengan protokolnya.

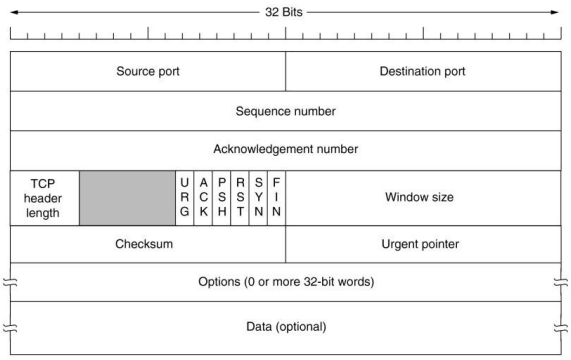
Model *TCP/IP* mempunyai 4 lapisan (layer) yaitu lapisan akses jaringan (data link), lapisan antara jarinagan (network), lapisan host ke host (transport), dan lapisan proses/aplikasi (application). Lapisan ini bisa dikatakan lapisan yang didapatkan dari lapis standart *protocol* OSI, dimana rincian protocol-protokol

yang ada dapa setiap lapisnya hamper sama. Jadi inti dari dari protocol ini terdiri dari dua bagian besar, yaitu *TCP* dan *IP*. (Daryanto, 2010)

TCP dikenal sebagi *protocol connection oriented,* artinya, protocol yang membutuhkan koneksi terlebih dahulu untuk menghantarkan pesan sampai terjadi proses petukaran antar program applikasi. *TCP* bertanggung jawab untuk mengirimkan aliran data ke tujuannya secara handal, berurutan dan terdokumentasi secara baik. Ciri-ciri dari *connection oriented* adalah:

1. Semua paket mendapatkan tanda terima (*acknoledgement*) dari pengirim.
2. Paket yang hilang atau tidak diterima akan dikirim ulang.
3. Paket yang atang diurutkan kembali (*sequence*).
4. *TCP* bekerja sama *IP* untuk mengirimkan data antar komputer melintasi jaringan. Jika IP menangani pengahantaran data, maka *TCP* berperan mengawasi atau menjaga *track unit individu* data (yang dikenal paket).

Dalam proses pengiriman data, *TCP* memotong tumpukan data dan menambahkan sebuah *header* ke masing-masing potongannya untuk membentuk segment. Kemudian tiap segment tersebut dilewatkan ke lapis *IP* untuk diproses menjadi datagram dengan menambahkan *header* IP.

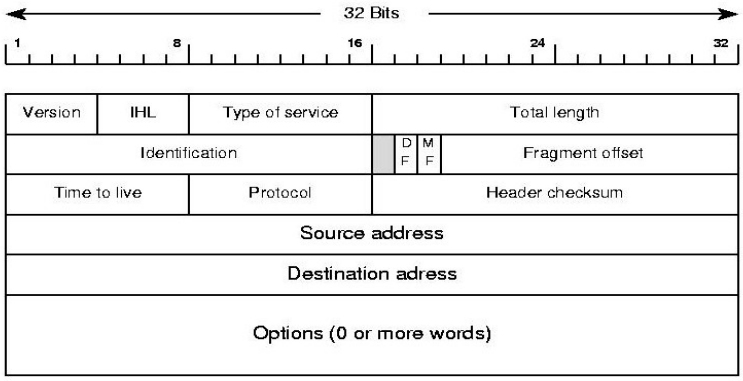


Gambar 2.1: Format *Datagram TCP*

Internet Protokol disingkat *IP* adalah protocol lapisan jaringan (*network layer* dalam *OSI Refence model*) atau protokol lapisan *internetwork* yang digunakan oleh protocol *TCP/IP* untuk melakukan pengalamatan dan *routing* paket data antar host-host di jaringan computer berbasis *TCP/IP*. Sebuah paket *IP* akan membawa data actual yang dikirim memlalui jaringan dari satu titik ke titik lainya.

Metode yang digunakan adalah *connectionless* yang berarti ia tidak perlu membuat dan memelihara sebuah sesi koneksi. Selain itu, protocol ini juga tidak menjamin penyampaian data, tapi hal ini diserahkan kepada protokol pada lapisan yang lebih tinggi lapisan transport dalam *OSI Reference Model* atau lapisan antar host dalam DARPA *Refernece* Model yakni protokol *Transmission Control Protocol* (TCP).

Format datagram IP digunakan Untuk keperluan perutean didalam Internet, *IP* memecah pesan yang diterimanya dari lapis host-host menjadi potongan-potongan dengan ukuran tertentu. Pada setiap potongan pesan, kemudian *IP* menambahkan header sehingga membentuk *datagram* IP.



Gambar 2.2: Format *datagram IP*

2.2.5 ISP (Internet service provider)

Internet service provider disingkat *ISP* adalah perusahaan atau badan yang menyediakan jasa sambungan Internet dan jasa lainnya yang berhubungan. Kebanyakan perusahaan telepon merupakan penyedia jasa Internet. Mereka menyediakan jasa seperti hubungan ke internet, pendaftaran nama domain, dan *hosting*. Sambungan yang disediakan oleh *ISP* dapat terhubung ke jaringan Internet global. Jaringan di sini berupa media transmisi yang dapat mengalirkan data yang dapat berupa kabel, radio, maupun VSAT. (Wijaya, 2014)

Speedy adalah penyelenggara jasa Internet yang dimiliki oleh Telkom Indonesia. Speedy berbasis teknologi akses *Asymmetric Digital Subscriber Line* (*ADSL*) dan *Gigabit Passive Optical Network (GPON)* dengan menggunakan jaringan fiber optik sampai ke rumah atau *Fiber to the Home (FTTH*). Berbeda dengan layanan internet dengan koneksi akses internet putar-nomor seperti TelkomNet Instan, Speedy menggunakan saluran telepon yang dapat dipergunakan untuk menelepon bersamaan dengan mengakses Internet. Untuk dapat menikmati

fasilitas menelepon dan internet secara simultan, pelanggan Speedy harus menggunakan splitter yang dapat memisahkan saluran telepon dan saluran modem.

Selain itu, menyepakati ketentuan yang berlaku di Plasa Telkom setempat. Telkom Indonesia menghadirkan layanan IndiHome berbasis *triple-play*. IndiHome menghadirkan layanan akses internet melalui jalur telepon dan fiber optik, VoIP sebagai layanan telepon berbasis internet, dan IPTV sebagai layanan televisi digital. (Wikipedia)

2.2.6 Mikrotik

Mikrotik dibuat oleh *MikroTikls* sebuah perusahaan di kota Riga, Latvia. Latvia adalah sebuah negara yang merupakan "pecahan" dari negara Uni Soviet dulunya atau Rusia sekarang ini. Mikrotik awalnya ditujukan untuk perusahaan jasa layanan Internet *(PJI)* atau *Internet Service Provider (ISP)* yang melayani pelanggannya menggunakan teknologi *nirkabel* atau *wireless*. (Saputro, Daniel, & Kustanto, 2008)

Saat ini *mikrotik* memberikan layanan kepada banyak *ISP nirkabel* untuk layanan akses Internet dibanyak negara di dunia dan juga sangat populer di Indonesia. *mikrotik* sekarang menyediakan *hardware* dan *software* untuk konektivitas internet di sebagian besar negara di seluruh dunia. Produk hardware unggulan *mikrotik* berupa *Router*, *Switch*, Antena, dan perangkat pendukung lainnya. Sedangkan produk *Software* unggulan *Mikrotik* adalah *Mikrotik Router OS*. (Saputro, Daniel, & Kustanto, 2008)

Mikrotik Router OS merupakan sistem operasi jaringan (*network operating system*) yang banyak digunakan oleh *Internet Service Provider* untuk keperluan

firewall atau *router* yang handal yang dilengkapi dengan berbagai fitur dan *tool*, baik untuk jaringan kabel maupun jaringan *wireless*.

Mikrotik merupakan *router* yang handal, yang mampu memberikan kelebihan pada sistem jaringan kita, karena dengan menggunakan *mikrotik* maka jaringan kita akan lebih stabil. Belakangan ini banyak usaha warnet yang menggunakan *mikrotik* sebagai *router*nya, dan hasilnya mereka merasa puas dengan apa yang diberikan *mikrotik*. *Mikrotik Router OS* hadir dalam berbagai level. Tiap level memiliki kemampuannya masing-masing, mulai dari level 3, hingga level 6. Secara singkat, level 3 digunakan untuk *router* ber*interface ethernet*, level 4 untuk *wireless client* atau *serial interface*, level 5 untuk *wireless AP*, dan level 6 tidak mempunyai limitasi apapun. Untuk aplikasi *hotspot*, bisa digunakan level 4 (200 *user*), level 5 (500 *user*) dan level 6 (*unlimited user*). (Saputro, Daniel, & Kustanto, 2008)

2.2.7 *Router*

Router merupakan perangkat keras jaringan yang memiliki peranan penting dalam mengatur lalulintas jaringan. *Router* bertugas untuk menangani proses pengiriman data dari jaringan ke jaringan lain. Agar *router* dapat mengetahui bagaimana meneruskan paket-paket ke alamat yang dituju dengan mengunakan jalur terbaik, *router* menggunakan peta atau tabel *routing*. Proses *routing* dilakukan *hop by hop*. (Wijaya, 2014)

Table routing adalah tabel yang memuat seluruh informasi *IP address* dari *interfaces* router yang lain sehingga *router* yang satu dengan *route*r lainnya bisa

berkomunikasi. *Table routing* hanya memberikan informasi sedang routing algoritma yang menganalisa dan mengatur *routing* tabel. (Wijaya, 2014)

Fungsi *router* antara lain (Wijaya, 2014):

1. Membaca alamat logika / *source and destination ip address* untuk menentukan *routing* dari suatu jaringan ke jaringan lain
2. Menyimpan *routing* tabel untuk menentukan rute terbaik antara *LAN* ke *WAN*

2.2.8 *Firewall*

Firewall adalah sistem yang digunakan untuk menjalankan kontrol akses keamanan pada jarinagn internal terhadap jaringan untrusted seperti internet. Umumnya, sebuah *firewall* di implementasikan dalam sebuah mesin terdedikasi, yang berjalan pada pintu gerbang (*gateway*) antara jaringan lokal dan jaringan lainnya. *Firewall* umumnya juga digunakan untuk mengontrol akses terhadap siapa saja yang memiliki akses terhadap jaringan pribadi dari pihak luar. (Wijaya, 2014)

Fungsi-fungsi umum *firewall* adalah sebagai berikut (Wijaya, 2014):

1. *Packet Filtering* adalah memeriksa *header* dari paket *TCP/IP* (tergantung arsitektur jaringannya, dalam contoh ini adalah *TCP IP*) dan memutuskan apakah data ini memiliki akses ke jaringan.
2. *Network Address Translation (NAT)* adalah biasanya sebuah jaringan memiliki sebuah *IP public* dan di dalam jaringan sendiri memiliki *IP* tersendiri. *Firewall* berfungsi untuk meneruskan paket data dari luar jaringan ke dalam jaringan dengan benar sesuai *IP* komputer lokal.

3. *Application Proxy* adalah *firewall* bisa mendeteksi *protocol* aplikasi tertentu yang lebih spesifik.
4. *Traffic management* adalah mencatat dan memantau trafik jaringan

2.2.9 *NAT*

Network Address Translation atau yang lebih biasa disebut dengan *NAT* adalah suatu metode untuk menghubungkan lebih dari satu komputer ke jaringan internet dengan menggunakan satu alamat *IP*. Banyaknya penggunaan metode ini disebabkan karena ketersediaan alamat *IP* yang terbatas, kebutuhan akan keamanan (*security*), dan kemudahan serta fleksibilitas dalam administrasi jaringan. Saat ini, protokol *IP* yang banyak digunakan adalah *IP* versi 4 (IPv4). Dengan panjang alamat 4 byte berarti terdapat 2^{32} = 4.294.967.296 alamat *IP* yang tersedia. Jumlah ini secara teoretis adalah jumlah komputer yang dapat langsung koneksi ke internet. (Wijaya, 2014)

Karena keterbatasan inilah sebagian besar *ISP* (*Internet Service Provider*) hanya akan mengalokasikan satu alamat untuk satu pengguna dan alamat ini bersifat dinamik, dalam arti alamat *IP* yang diberikan akan berbeda setiap kali *user* melakukan koneksi ke internet. Dengan *NAT gateway* yang dijalankan di salah satu komputer, satu alamat *IP* tersebut dapat dibagi ke beberapa komputer yang lain dan mereka bisa melakukan koneksi ke internet secara bersamaan. (Wijaya, 2014)

Network address translator terdiri dari berbagai jenis, yaitu (Wijaya, 2014):

1. *NAT* Tipe Statis menggunakan *table routing* yang tetap, atau alokasi translasi alamat *IP* ditetapkan sesuai dengan alamat asal ke alamat tujuan, sehingga tidak memungkinkan terjadinya pertukaran data dalam suatu

alamat *IP* bila translasi alamat ipnya belum didaftarkan dalam *table nat*, *Translasi Static* terjadi ketika sebuah alamat lokal (*inside*) di petakan ke sebuah alamat global/internet (*outside*). Alamat lokal dan global dipetakan satu lawan satu secara statik. *NAT* secara statis akan melakukan request atau pengambilan dan pengiriman paket data sesuai denganaturan yang telah ditabelkan dalam sebuah *NAT*.

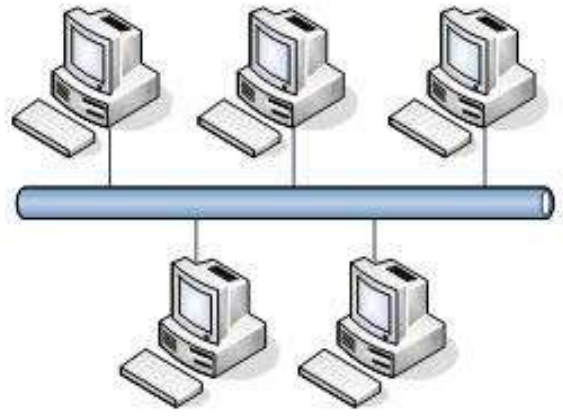
2. *Dynamic Network Address Translation* dimaksudkan untuk suatu keadaan dimana anda mempunyai *IP address* terdaftar yang lebih sedikit dari jumlah *IP address un-registered*. *Dynamic NAT* menterjemahkan setiap komputer dengan *IP* tak terdaftar kepada salah satu *IP address* terdaftar untuk konek ke internet. Hal ini agak menyulitkan para penyusup untuk menembus komputer didalam jaringan anda karena *IP address* terdaftar yang diasosiasikan ke komputer selalu berubah secara dinamis, tidak seperti pada *NAT* statis yang dipetakan sama. Kekurangan utama dari dynamis *NAT* ini adalah bahwa jika jumlah *IP address* terdaftar sudah terpakai semuanya, maka untuk komputer yang berusaha konek ke internet tidak lagi bisa karena *IP address* terdaftar sudah terpakai semuanya.

3. *Masquerading NAT* ini menterjemahkan semua *IP address* tak terdaftar pada jaringan anda dipetakan kepada satu *IP address* terdaftar. Agar banyak *client* bisa mengakses Internet secara bersamaan, *router* NAT menggunakan nomor port untuk bisa membedakan antara paket-paket yang dihasilkan oleh atau ditujukan komputer-komputer yang berbeda. Solusi *Masquerading* ini memberikan keamanan paling bagus dari jenis-jenis *NAT* sebelumnya,

kenapa? Karena asosiasi antara *client* dengan *IP* tak terdaftar dengan kombinasi *IP address* terdaftar dan nomor *port* didalam *router NA*T hanya berlangsung sesaat terjadi satu kesempatan koneksi saja, setelah itu dilepas.

2.2.10 Topologi Jaringan

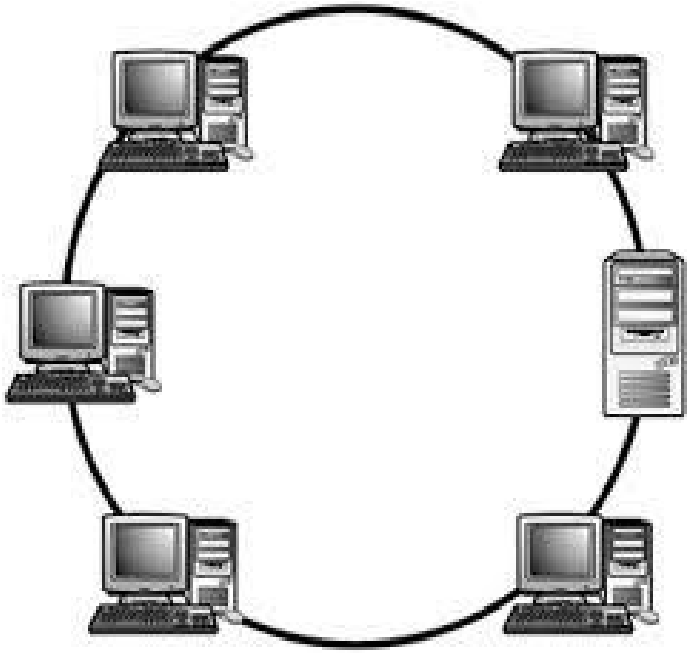
Topologi Jaringan adalah suatu cara menghubungkan komputer yang satu dengan yang lainya sehingga membentuk sebuah Jaringan. Cara yang saat ini banyak di gunakan adalah *Ring* (Cincin), *Star* (Bintang). Masing-masing *topologi* ini mempunyai ciri khas, dengan kelebihan dan kekurannya sendiri. Pada *topologi bus* digunakan sebuah kabel tunggal atau kabel pusat dimana seluruh *Workstation* dan *Server* dihubungkan. Merupakan *Topologi* fisik yang mengunakan Kabel *Coaxial* dengan mengunakan *T-Connector* dengan terminal 50 *omh* pada ujung Jaringan. *topologi bus* mengunakan satu kabel yang kedua ujungnya ditutup dimana sepanjang kabel terdapat node-node.



Gambar 2.3: *Topologi Bus*

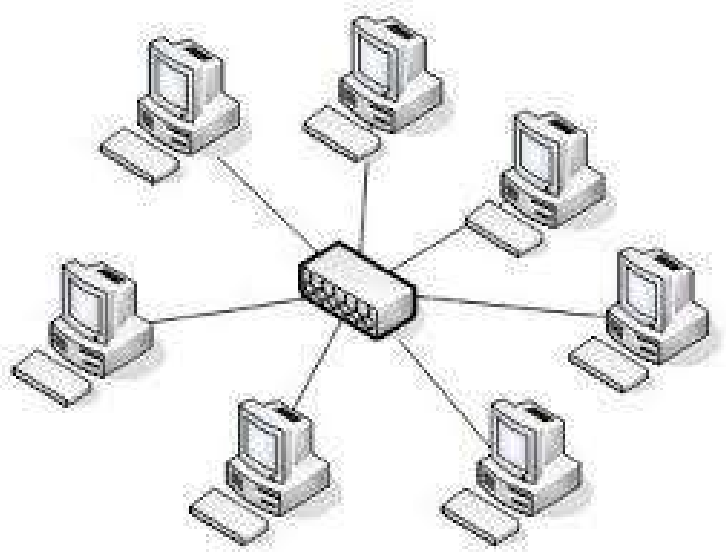
Di dalam *topologi ring* semua *Workstatio*n dan *server* dihubungakn sehingga terbentuk suatu pola lingkaran atau cincin. Tiap *Workstatio*n atau *server* akan menerima

dan melewatkan Informasi dari satu komputer ke komputer yang lainnya, bila alamat-alamat yang di maksud sesuai maka informasi diterima dan bila tidak informasi akan di lewatkan.



Gambar 2.4: *Topologi Ring*

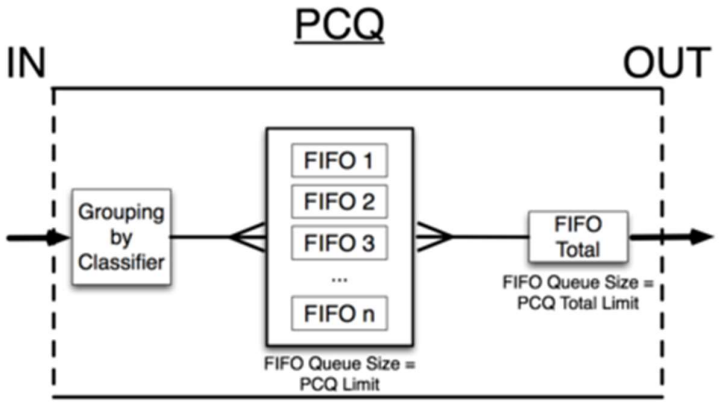
Pada *Topologi Star*, masing-masing *Workstatio*n dihubungkan secara langsung ke *Server* atau *Hub/Swich*. *Hub/Swich* berfungsi menerima sinyal sinyal dari komputer dan meneruskannya ke semya komputer yang terhubung dengan *Hub/Swich* tersebut. Jaringan dengan *Topologi* ini lebih mahal dan cukup sulit pemasangannya. Setiap komputer mempunyai kabel sendiri-sendiri sehingga lebih mudah dalam mencari kesalahan pada jaringan. Kabel yang digunakan biasanya menggunakan Kabel *UTP CAT5*.



Gambar: 2.5 *Topologi Star*

2.2.11 Per Connection Queue (PCQ)

PCQ (Per Connection Queue) diperkenalkan untuk mengoptimalkan sistem antrian yang sangat besar. Algoritma *PCQ* akan mengelompokkan aliran data masuk dan membedakan setiap aliran berdasarkan parameter *dst-address, src-address, dst-port* atau *src-port*. Kemudian algoritma *FIFO* akan menentukan berapa ukuran antrian yang diizinkan dan melakukan pembatasan pada setiap subaliran (individual). Lalu melakukan hal yang sama untuk seluruh subaliran (global).



Gambar 2.6: Cara kerja PCQ

PCQ dikenal memiliki kemampuan membagi *bandwitdh* dengan adil dan merata. Misalkan kita memiliki *bandwitdh* sebesar 1Mbps, jika ada 1 user yang sedang online maka *bandwitdh* yang ada terpakai seluruhnya untuk satu user. Jika ada 2 user, *bandwitdh* secara merata akan dibagai untuk dua user. Jika ada 3 user, *bandwitdh* juga akan dibagi untuk tiga *user*, begitu seterusnya. Beberapa parameter yang sering digunakan *PCQ* adalah sebagai berikut:

1. *pcq-classifier (dst-address | dst-port | src-address | src-port; default: "")* : mengidentifikasi sebuah aliran. Misalkan jika parameter yang digunakan *dst-address* maka aliran tersebut dikelompokkan sebagai koneksi download, sebaliknya jika parameter yang digunakan *src-address* maka aliran dikelompokkan sebagai koneksi upload.
2. *pcq-rate* (number) : maximal download yang tersedia untuk setiap subaliran. Misal kita isi dengan 100k maka maximal download yang akan didapat per *IP/client/*target akan dibatasi hanya sampai 100k.
3. *pcq-limit* (number) : banyaknya koneksi untuk setiap subaliran, maksudnya jumlah koneksi yang diizinkan untuk setiap subaliran (dalam KB). Misal

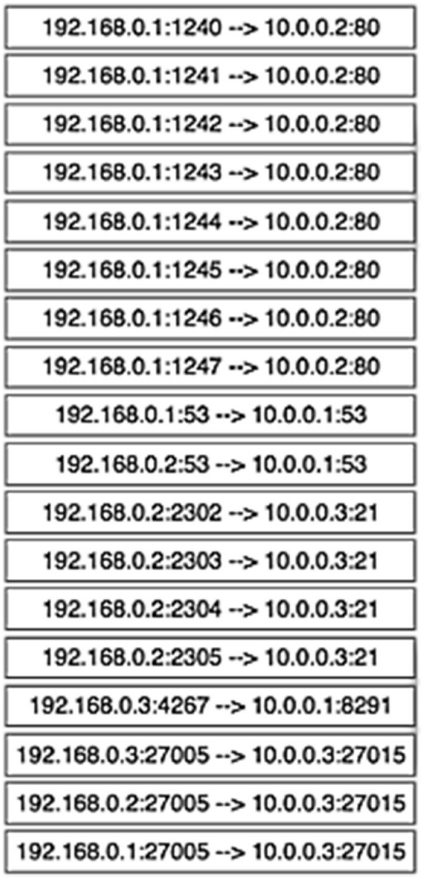
bila kita memasukkan nilai 50, maka hanya 50 koneksi yang bisa didapat per *IP/client*/target.

4. *pcq-total-limit* (number) : banyaknya koneksi untuk total keseluruhan subaliran, maksudnya total keseluruhan koneksi yang diizinkan untuk semua subaliran (dalam KB). Bila kita memasukkan nilai 2000 kemudian dibagi 50 nilai *pcq-limit*, maka secara teori akan didapat angka 40 total IP/client/target yang bisa terkoneksi ke jaringan Mikrotik kita.
5. *pcq-burst-rate* (number) : maximal upload/download yang dapat dicapai selama burst subaliran membolehkannya.
6. *pcq-burst-threshold* (number) : batas *pcq-burst-rate,* yang digunakan sebagai indikator boleh tidaknya nilai *pcq-burst-rate* dijalankan (*on/off switch*). Jika rata-rata *bandwitdh* dibawah nilai ini maka *pcq-burst-rate* dibolehkan, jika tidak maka ditolak.
7. *pcq-burst-time* (time) : periode waktu dalam hitungan detik dimana nilai rata-rata *bandwitdh* mulai dihitung.
8. *pcq-dst-address-mask* (number) : lebar jaringan IPv4 yang akan digunakan sebagai pengenal subaliran *dst-address*.
9. *pcq-src-address-mask* (number) : lebar jaringan IPv4 yang akan digunakan sebagai pengenal subaliran *src-address*.
10. *pcq-dst-address6-mask* (number) : lebar jaringan IPv6 yang akan digunakan sebagai pengenal subaliran *dst-address*.
11. *pcq-src-address6-mask* (number) : lebar jaringan IPv6 yang akan digunakan sebagai pengenal subaliran *src-address*

PCQ terdiri dari beberapa jenis yaitu:

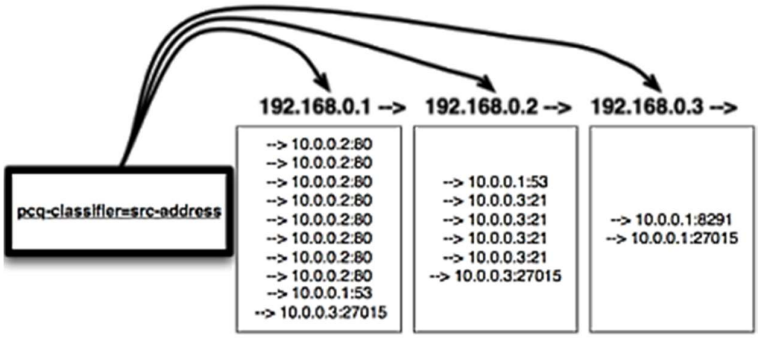
1. PCQ Classifier

Sekarang mari kita pahami bagaimana *PCQ* melakukan pengelompokkan. Misalkan ada 18 paket aliran dari alamat dan port tertentu ke alamat dan port tertentu. *PCQ* akan mengelompokkan 18 paket aliran tersebut ke dalam beberapa sub aliran.



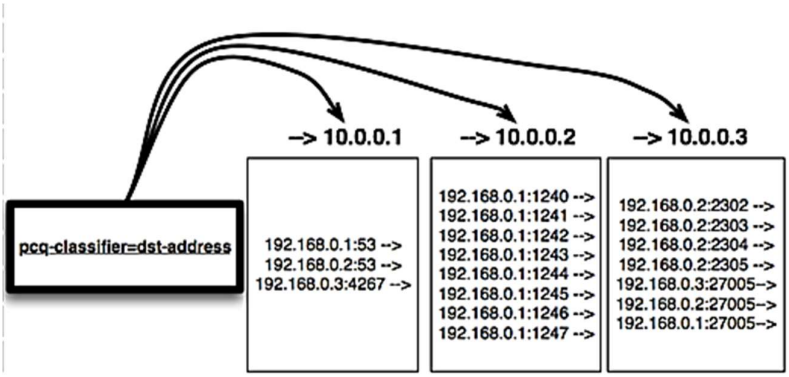
Gambar 2.7: Pengelompokkan paket *PCQ classifier*

Jika parameter yang digunakan adalah alamat asal (*pcq-classifier=src-address*), maka *PCQ* akan mengelompokkannya menjadi 3 subaliran.



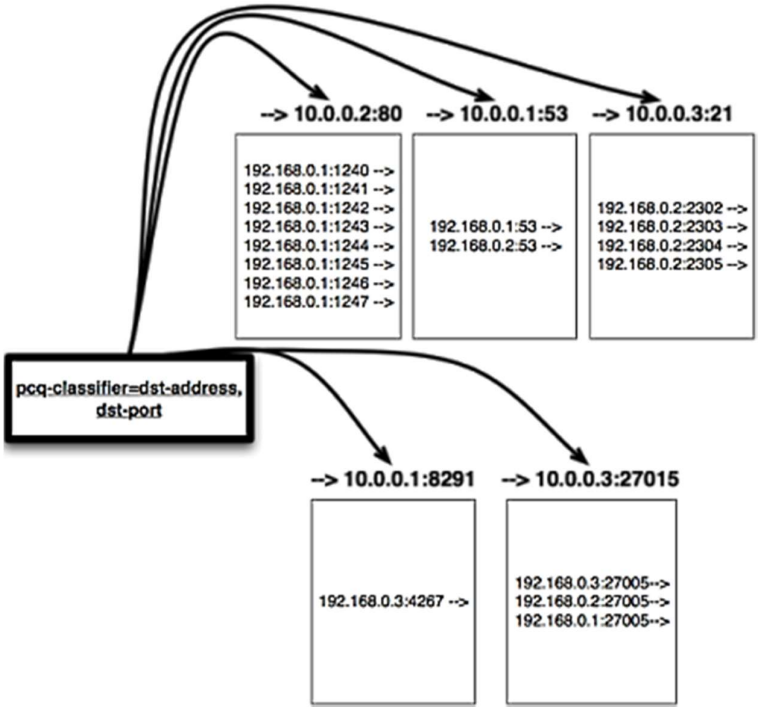
Gambar 2.8: Pengelompokkan sub aliran (*pcq-classifier=src-address*)

Jika parameter yang digunakan adalah alamat tujuan (*pcq-classifier=dst-address*), *PCQ* akan mengelompokkannya menjadi 3 subaliran.



Gambar 2.9: Pengelompokkan sub aliran (*pcq-classifier=dst-address*)

Jika parameter yang digunakan adalah alamat dan port tujuan (*pc-classifier=dst-address, dst-port*), akan menjadi 5 subaliran.



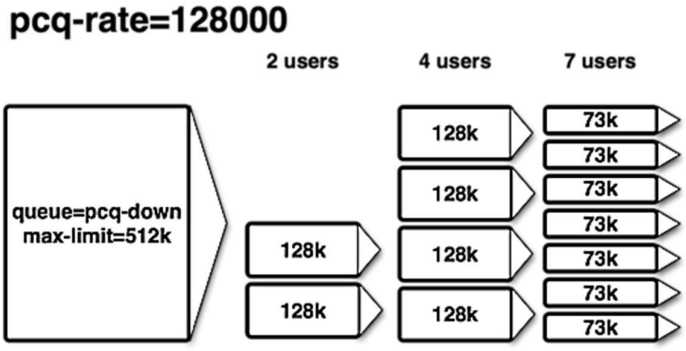
Gambar 2.10: Pengelompokkan suba liran (*pcq-classifier=dst-address,dst-port)*

Sangat mungkin akan banyak sekali koneksi yang terbentuk dalam satu subaliran jika paket data yang masuk dalam jumlah besar. Maka parameter *pcq-total-limit* dan *pcq-limit* memiliki peran penting disini. *pcq-total-limit* akan membatasi total keseluruhan koneksi yang dapat terbentuk untuk seluruh subaliran. Jika kita meletakkan nilai *pcq-total-limit*=2000 (default), artinya hanya akan ada 2000 koneksi yang terbentuk untuk seluruh sub aliran.

Sedangkan *pcq-limit* akan membatasi jumlah koneksi yang diizinkan per *IP/target/*client. Jika kita meletakkan nilai *pcq-limit*=50 (*default*) maka maksimal koneksi yang diperkenankan per *IP/client*/target hanya sampai 50 koneksi baik asal maupun tujuan.

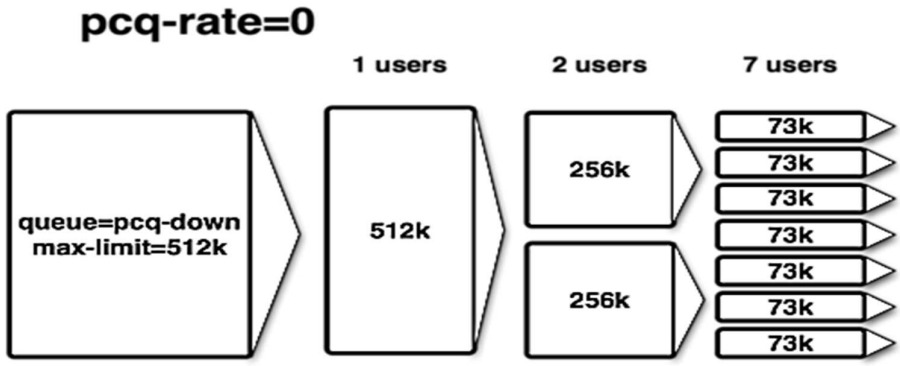
2. PCQ Rate

Jika *pcq-limit* membatasi jumlah koneksi yang terbentuk, maka *pcq-rate* akan membatasi *maximal download* yang bisa dicapai per *IP/client/*target. Fungsi *pcq-rate* hampir mirip dengan *max-limit* pada *queue tree* atau *queue simple*, yaitu sama-sama membatasi *maximal download* per *client*. Coba kita lihat pebedaan jika kita menentukan nilai *pcq-rate* dan tidak. Coba perhatikan gambar dibawah ini. Jika nilai *max-limit*=512k dengan nilai *pcq-rate*=128k bisa kita lihat bahwa sebenarnya *bandwitdh* tidak terbagi secara merata jika ada 2 user yang sedang memakai. Secara teori 512k - (2 x 128 k) = 256k, akan ada 256k *bandwitdh* yang tidak terpakai. Perhatikan hanya tidak terpakai bukan terbuang percuma karena memang hanya 2 *user* yang sedang memakai. Begitu juga jika ada 1 *user*, artinya bahwa akan ada *bandwitdh* sebesar 384k yang tidak terpakai (bukan terbuang percuma). Berdasarkan contoh ini maka, seluruh *bandwitdh* akan terpakai dan terbagi secara adil dan merata jika user berjumlah 4 atau lebih. Jadi maksudnya selama nilai pembagian antara nilai *max-limit* dan jumlah *user* kurang dari nilai *pcq-rate* maka user secara *fixed* mendapat *bandwitdh* sebesar nilai *pcq-rate*.



Gambar 2.11: Nilai *pcq-rate=128000*

Lalu bagaimana jika nilai *pcq-rate=0 (default)*? Disini baru kita bisa melihat bahwa *bandwitdh* itu dibagi secara adil dan merata. Baik ketika user ada 1, 2, atau lebih seluruh *user* akan mendapatkan jatah *bandwitdh* yang sama.



Gambar 2.12: Nilai *pcq-rate=0 (default)*

Dalam beberapa kasus dianjurkan menggunakan salah satu dari keduanya, maksudnya menentukan nilai *pcq-rate* atau menentukan nilai *max-limit* pada *queue tree* atau *queue simple*.

2.2.12 *Proxy Server*

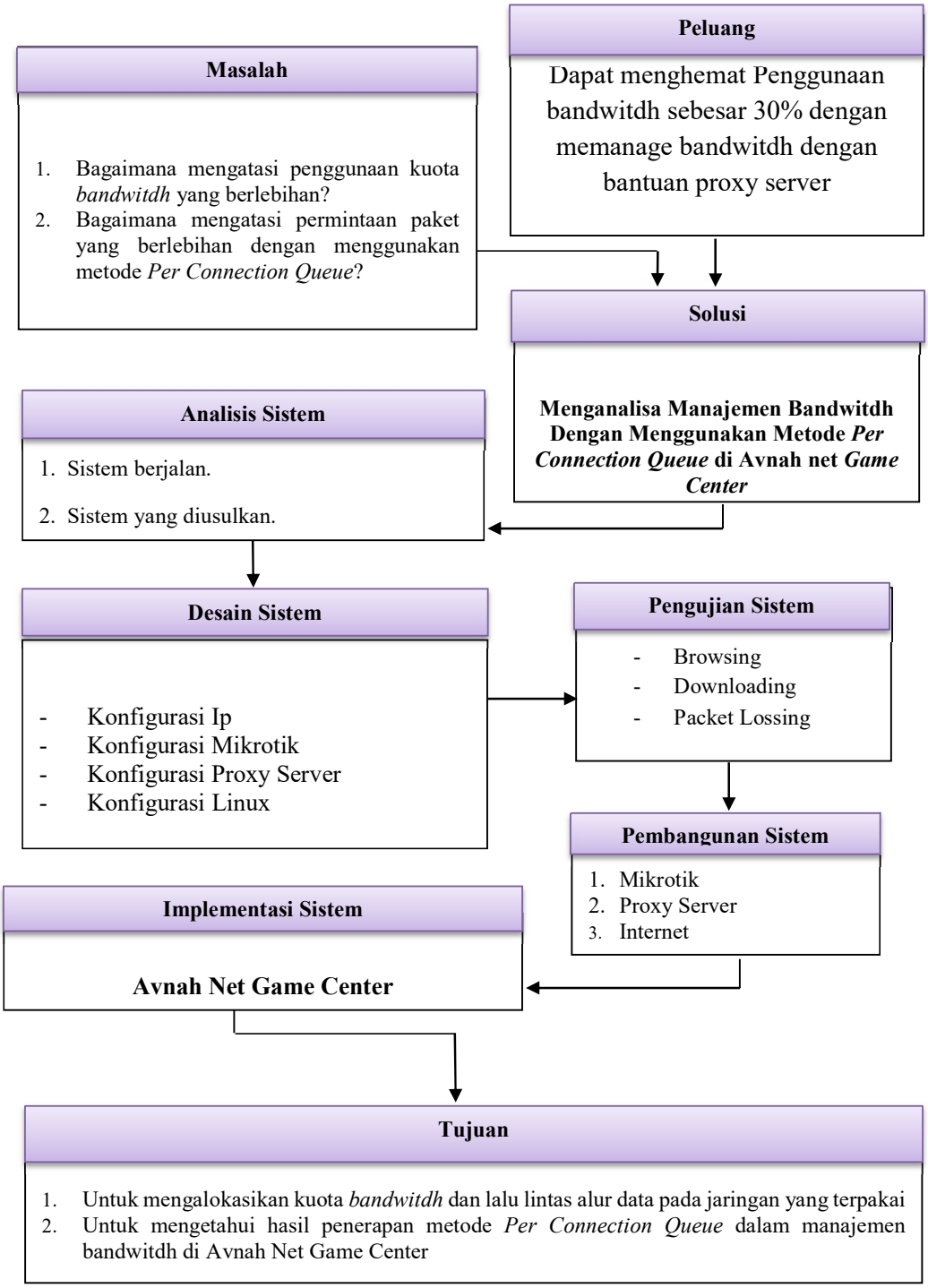
Proxy server (peladen proxy) adalah sebuah komputer server atau program komputer yang dapat bertindak sebagai komputer lainnya untuk melakukan *request* terhadap content dari Internet atau intranet.

Proxy Server bertindak sebagai *gateway* terhadap dunia ini Internet untuk setiap komputer klien. *Proxy server* tidak terlihat oleh komputer klien: seorang pengguna yang berinteraksi dengan Internet melalui sebuah proxy server tidak akan mengetahui bahwa sebuah *proxy server* sedang menangani request yang dilakukannya. *Web server* yang menerima *request* dari *proxy server* akan

menginterpretasikan *request-request* tersebut seolah-olah *request* itu datang secara langsung dari komputer klien, bukan dari *proxy server*.

Proxy server juga dapat digunakan untuk mengamankan jaringan pribadi yang dihubungkan ke sebuah jaringan publik (seperti halnya Internet). *Proxy server* memiliki lebih banyak fungsi daripada *router* yang memiliki fitur *packet filtering* karena memang *proxy server* beroperasi pada level yang lebih tinggi dan memiliki kontrol yang lebih menyeluruh terhadap akses jaringan. *Proxy server* yang berfungsi sebagai sebuah "agen keamanan" untuk sebuah jaringan pribadi, umumnya dikenal sebagai *firewall*.

2.3 Kerangka Pikir



Gambar 2.13 Bagan Kerangka Berpikir

BAB III
METODE PENELITIAN

3.1 Objek dan Jadwal Penelitian

Objek penelitian ini adalah managemen *bandwitdh* di Avnahnet *Game Center* menggunakan metode *PCQ*. Penelitian ini berlangsung selama tahun 2016, berlokasi pada Liluwo Gorontalo.

3.2 Metode Penelitian

Penelitian ini menggunakan metode penelitian eksperimen. Dipandang dari tingkat penerapannya, maka penelitian ini merupakan penelitian terapan. Dipandang dari jenis informasi yang diolah, maka penelitian ini merupakan penelitian kualitatif. Dipandang dari perlakuan terhadap data, maka penelitian ini merupakan penelitian konfirmatori. Sedangkan jenis penelitian ini adalah eksperimental.

Adapun teknik untuk pengumpulan data adalah sebagai berikut :

1. Wawancara *(Interview)*

Merupakan suatu pengumpulan data yang dilakukan dengan cara tanya jawab atau dialog secara langsung dengan pihak-pihak yang terkait dengan penelitian yang dilakukan. Dalam hal ini peneliti melakukan tanya jawab kepada pemilik dan user di Avnahnet *Game Center*.

2. Pengamatan *(Observasi)*

Yaitu metode pengumpulan data dengan cara mengadakan tinjauan secara langsung ke objek yang diteliti. Untuk mendapatkan data yang bersifat nyata dan meyakinkan maka peneliti melakukan pengamatan langsung pada penggunaan *bandwitdh* di Avnahnet Game Center.

3. Studi Pustaka

Untuk mendapatkan data-data yang bersifat teoritis maka penulis melakukan pengumpulan data dengan cara membaca dan mempelajari buku-buku, makalah ataupun referensi lain yang berhubungan dengan masalah yang dibahas.

3.3 Analisis Hasil

Analisis hasil merupakan tahapan paling penting dalam penyelesaian suatu kegiatan penelitian ilmiah. Dalam penelitian ini, dilakukan analisis menggunakan *PCQ* untuk mengetahui penelitian secara kualitatif dengan menggunakan teknik pengumpulan data wawancara dan observasi.

3.4 Perancangan Modul Aplikasi

Perancangan modul aplikasi dilakukan agar dapat membuktikan metode manakah yang lebih tepat untuk digunakan dalam memanage *bandwitdh* jaringan komputer. Dalam hal ini, peneliti melakukan perancangan metode menggunakan metode *PCQ.*

3.5 Spesifikasi Kebutuhan Hardware dan Software

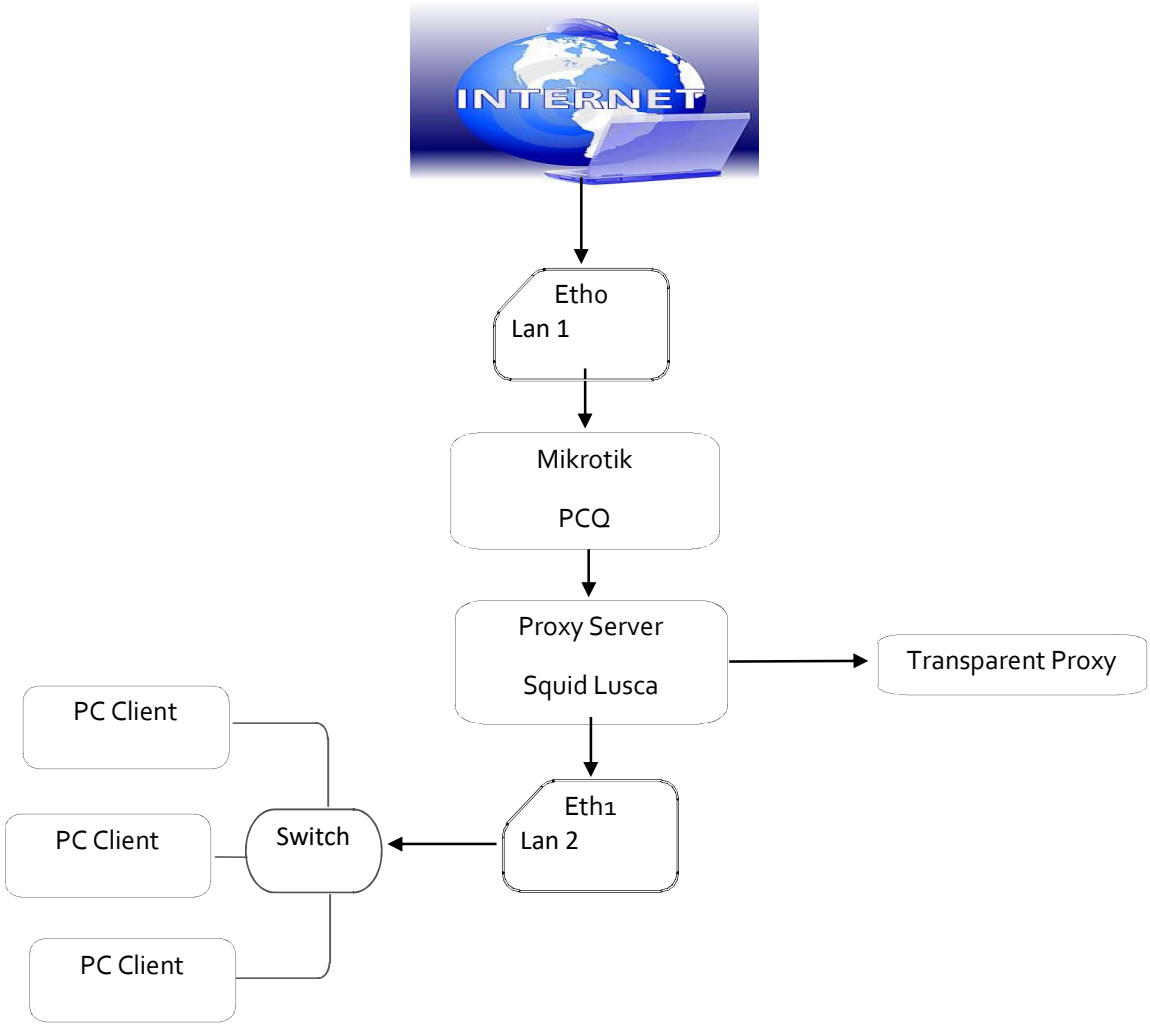
Tabel 3.1: Spesifikasi Hardware

No.	Perangakat	Jml	Spesifikaisi Unit
1	PC Proxy	1	CPU: intel® xeon® processor e5-2699 v4 (55m cache
			DVD room
			Memory: 8 GB RAM DDR3
			Hardisk: 1 TB
			4 slot USB
			Ethernet card (NIC)
2	PC Client	3	CPU: Intel(R) Intel(R) Core(TM) i3 CPU M350 2.27GHz
			Memory: 2 GB RAM DDR3
			Hardisk: 1TB
			Ethernet card (NIC)
3	Switch-hub	1	TP-LINK TL-SF1016D

Tabel 3.2: Spesifikasi Software

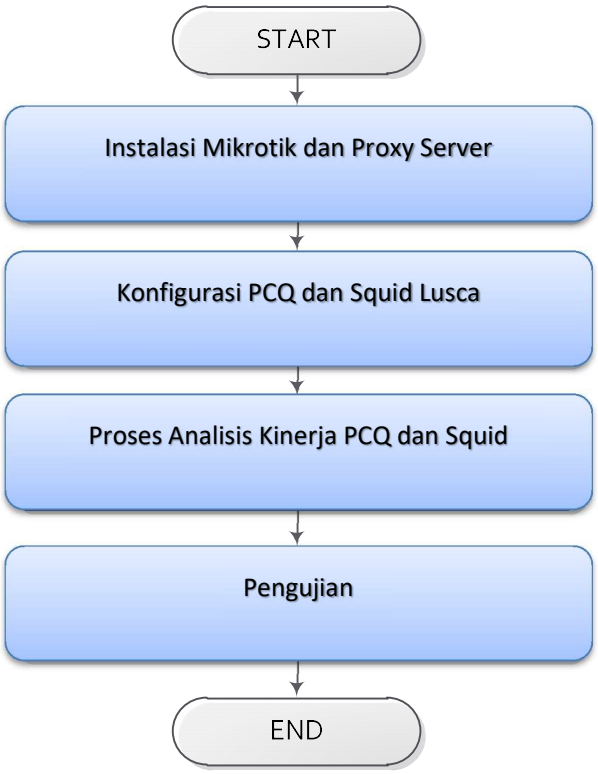
No	Software	Keterangan
1.	MikrotikOS router versi 5.2	Sebagai sistem operasi mikrotik
2.	Microsoft Windows 7 SP2	Sebagai sistem operasi untun client
3.	Mikrotik winbox v.2.2.16	*Utility* untuk melakukan remote GUI ke Router Mikrotik
4	Ubuntu Server 16.04.1 LTS	Sebagai system operasi proxy server
5	Squid Lusca 2.7	Software proxy
6	PuTTy	Software remote Ubuntu server

3.6 Rancangan Umum Sistem



Gambar 3.1: Rancangan Sistem Per Connection Queue dengan bantuan Proxy Server (PCQ)

3.7 Langkah-Langkah Implementasi



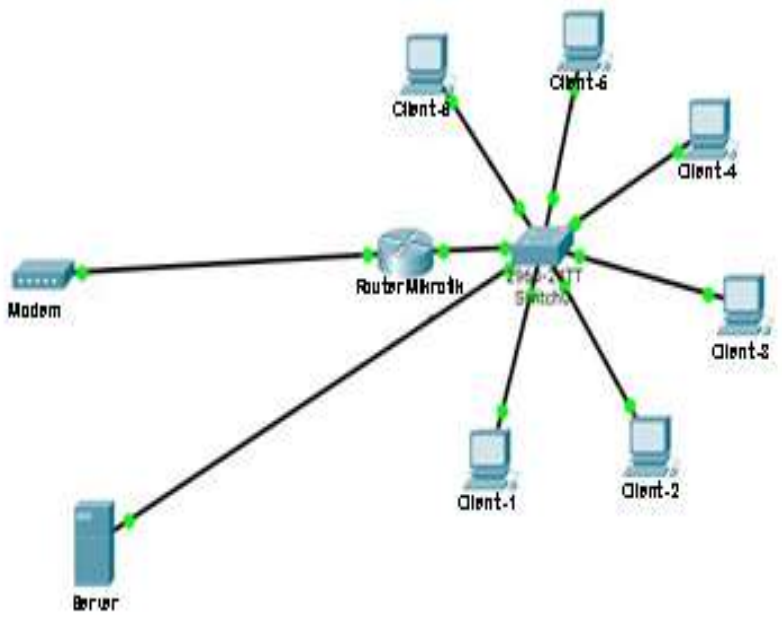
Gambar 3.2: Langkah-Langkah Implementasi

BAB IV
HASIL PENILITIAN

4.1 Perancangan Topologi

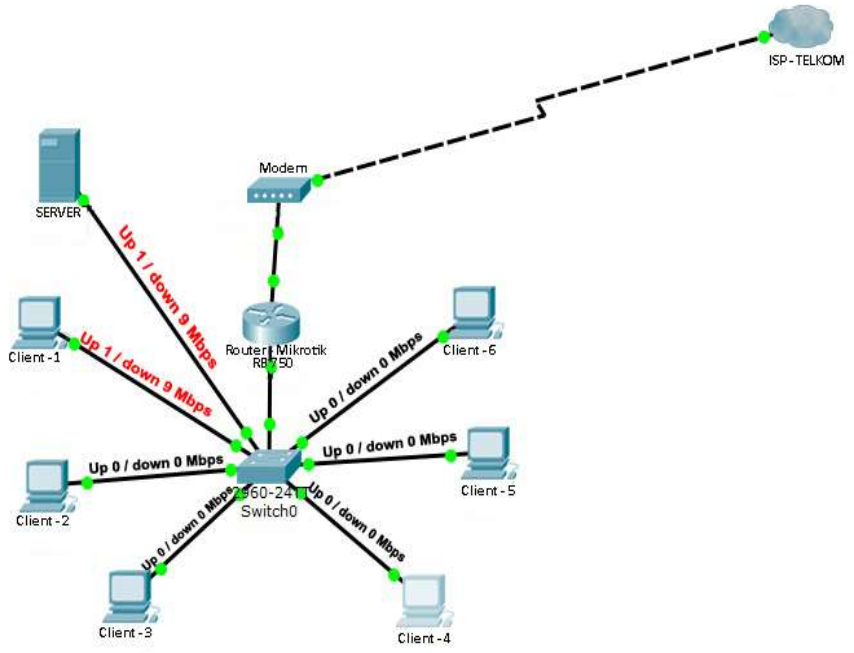
4.1.1 Topologi Pada AVNAH Net Game Center

Topologi yang di gunakan oleh AVNAH Net Game Center adalah topologi *Star*, yaitu di mana masing-masing *Workstatio*n dihubungkan secara langsung ke *Server* atau *Hub/Swich*. *Hub/Swich* berfungsi menerima sinyal sinyal dari komputer dan meneruskannya ke semua komputer yang terhubung dengan *Hub/Swich* tersebut. Dan untuk Tipe kelas untuk IP yang di gunakan adalah kelas C.



Gambar 4.1. Topologi Jaringan Umum pada AVNAH Net Game Center

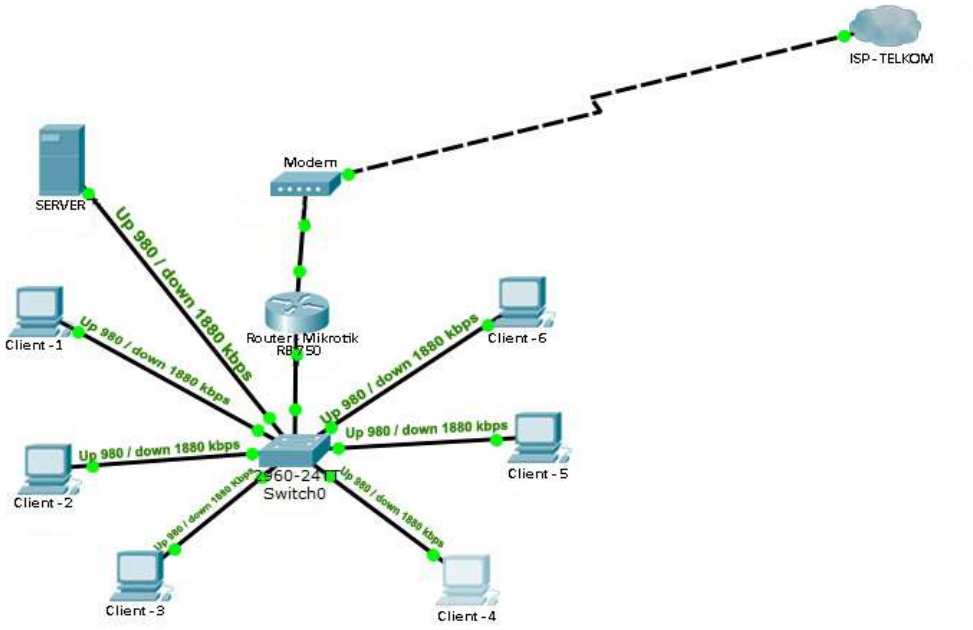
4.1.2 Topologi Jaringan Lokal Sebelum Penerapan Manajemen *Bandwitdh*



Gambar 4.2. Topologi Jaringan Lokal Sebelum Penerapan Manajemen *Bandwitdh*

Beradasarkan gambar diatas *bandwitdh* yang dialoksikan terhadap client adalah sejumlah 18 Mbps *downstream* dan 2 Mbps Upstream. Tanpa penerapan manajemen *bandwitdh* di *router*, tiap-tiap *client* tidak akan bisa mendapatkan *bandwitdh* secara merata. Bahkan apabila ada satu client yang melakukan aktivitas *download* dan *upload* yang berlebihan tanpa menggunakan manajemen *bandwitdh*, maka *client* yang lain akan mengalami koneksi yang lambat bahkan tidak akan mendapatkan jatah *bandwitdh* sama sekali karena semua alokasi *bandwitdh* habis terpakai oleh satu pc *client*.

4.1.3 Topologi Jaringan Dengan Menerapkan Manajemen *Bandwitdh*



Gambar 4.3. Topologi Jaringan Dengan Menerapkan Manajemen *Bandwitdh*

Penerapan manajemen *bandwitdh* menggunakan metode PCQ akan melakukan pembagian *bandwitdh* secara merata keseluruh *client* pada *local area network* dimana prosesnya bisa berjalan secara dinamis.

4.1.4 Perancangan Konfigurasi dasar Mikrotik.

Konfigurasi dasar Mikrotik, menu-menu yang akan dikonfigurasi antara lain:

1. *Interfaces*, dimana pada menu ini akan diberikan nama pada masing- masing ether disesuaikan dengan fungsinya agar mudah dalam mengingat pengkonfigurasian
2. *IP Addresses,* Menu ini digunakan untuk memberikan IP *Address* masing-masing *interface*

3. *IP DHCP Client,*. Pengaturan *protocol* (ip address) dilakukan di *client*, apakah mode *static* atau *dynamic*, *DHCP client* meminta server untuk memberikan ip, sebelum *client* mendapatkan ip *dynamic*, *client* terlebih dahulu meminta ke server yang ada pada jaringan tersebut, dan server melakukan pemeriksaan terhadap *client* yang meminta ip *dynamic*, jika sesuai dan diperbolehkan maka server baru mengirimkan ip ke *client*.
4. *IP Routes,* Menu ini digunakan untuk mengkonfigurasi agar semua range IP akan diroutingkan pada IP *Route*-nya dimana disini diberikan IP *Route*: 192.168.2.1/24 sebagai *gateway*.
5. *IP Firewall,* Pada menu ini *firewall* NAT pada Mikrotik diaktifkan, dimana NAT berfungsi untuk meneruskan paket dari IP lokal ke IP publik (fungsi *gateway router*). Kemudian pada tab *Action*, pilih *action masquerade*. *Masquerade* akan mengubah paket-paket data IP *Address* asal dan port dari jaringan lokal (lan) ke salah satu IP yang IP *Address* yang diberikan oleh ISP *(Internet Services Provider)* untuk selanjutnya diteruskan ke jaringan internet global
6. *IP Pool,* Menu ini digunakan untuk menentukan *range* IP yang akan diberikan ke *user*.
7. *IP DHCP Server,* DHCP server konfigurasi *protocol* (IP *address*) disediakan oleh server untuk diberikan ke *client* yang meminta ip. (ip *address*) yang diberikan, ditentukan oleh server pemberian jatah ip bisa dalam hitungan menit, jam, hari dan bulan, juga disertai dengan *netmask, gateway* dan dns server, itu semua tergantung dari pengaturan di servernya.

8. *IP DNS,* DNS diisi sesuai dengan DNS yang didapatkan dari ISP (*Internet Service Provider*). Pada penelitian ini menggunakan ISP indiehome sebagai percobaan dimana DNS *Addressnya* yaitu 8.8.8.8

4.1.5 Perancangan Konfigurasi Manajemen *Bandwitdh* dan PCQ

1. Konfigurasi NAT

Setelah pengkonfigurasian IP dan DNS, selanjutnya harus menambahkan konfigurasi NAT (*network address translation*).NAT berguna agar *client* dapat terhubung dengan internet. NAT akan mengubah alamat sumber paket yaitu alamat *client* yang memiliki IP *address private* agar dapat dikenali oleh internet yaitu dengan cara mentranslasikanya menjadi IP *address public*. Pengaturan NAT ini menggunakan metode *Masquerading* NAT. Karena *provider* yang digunakan hanya memberikan satu IP *public*, jadi semua*IP address* dari *client* akan dipetakan kepada satu IP *public*

2. Konfigurasi Mangle

Proses penandaan ini berdasar pada hasil *stateful packet inspection*, yaitu src-IP, dst-IP, src-*port* dan dst-*port*.Dari parameter tersebut kemudian dapat dilakukan *connection-mark* dan *routing-mark,* yang kemudian dapat digunakan untuk pengolahan paket yang spesifik. Selain itu terdapat *chain* yang merupakan tahapan dari proses pengolahan data, sehingga penandaan dapat dilakukan dengan lebih spesifik sesuai dengan *chain* yang ada

3. Konfigurasi PCQ

Setelah memberikan tanda *Mark Connection* pada koneksi yang keluar dan masuk ke *Gateway* Mikrotik ke *client* selnjutnya yaitu membuat tipe queuenya karna di sini penluis menggunakan metode PCQ. Maka langkah selanjutnya yaitu menambahkan tipe queue, di queue tipe di mikrotik yaitu PCQ *download* dan PCQ *upload*, yang harus di perhatikan adalah menentukan *classifier* pada tipe queue. Untuk *traffic download* pilih *Dst. Address* sedangkan untuk *traffic upload* pilih *Src. Address*.

4. Konfigurasi Queue Tree

Queue Tree berfungsi untuk mengimplementasikan fungsi yang lebih kompleks dalam limit *bandwitdh* pada mikrotik dimana penggunaan *packet mark* nya memiliki fungsi yang lebih baik. Digunakan untuk membatasi satu arah koneksi saja baik itu *download* maupun *upload*.

Setelah selesai menambahkan tipe queue PCQ *download* dan PCQ *upload*, maka selanjutnya memasukkannya ke dalam queue tree agar pembagian *bandwitdh*nya merata. Pada pembuatan queue tree penentuan *parent* nya sama sesuaikan saat membuat *mangle*.

5. Konfigurasi Proxy Server

Setelah selesai mengkonfigurasi topologi jaringan dan konfigurasi Mikrotik. Maka penulis akan menambahkan sebuah *proxy* server sebagai *device* penunjang kestabilan koneksi internet, karena sistem kerja sebuah *proxy* server adalah menyimpan *web cache* yang sudah terakses sebelumnya sehingga saat akan di akses kembali tidak akan menggunakan *bandwitdh* yang sudah tersedia.

Disini penulis menggunakan *Linux Ubuntu server 16.04.1 LTS* sebagai sistem operasi *proxy* server, *Squid Lusca* sebagai *software proxy* dan *PuTTy* sebagai *software* yang akan meremote *Ubuntu server*.

4.1.6 Perancangan Pengujian

Pada tahap ini penulis melakukan pengujian terhadap unjuk kerja manajemen *bandwitdh* menggunakan metode PCQ serta pengujian QOS (*quality of service*) dari jaringan internet pada tiap *client* setelah dilakukan *Queue Tree*. Untuk melakukan pengujian, penulis mengguankan beberapa *tools*, seperti *tools* winbox, tools *downloader* seperti IDM (*Internet download manager*), *speedtest*, dan *Axence netTools.*

1. Perancangan Pengujian Manajemen *Bandwitdh*

Pengujian Manajemen *Bandwitdh* akan dilakukan meliputi dua fase, yang pertama adalah pengujian *browsing* dan yang kedua pengujian *downlaod*. Pengujian *browsing* dilakukan dengan cara semua PC *client* melakukan *browsing* ke situs yang sudah di tentukan. Kemudian dilakukan analisa menggunakan *tools connections* dan menu *interface* yang terdapat di *winbox* untuk mengetahui penyebaran penandaan koneksi dan besar paket yang dikirim pada masing-masing *interface*. Jika *mark-connection* dan besar paket pada masing-masing interface memiliki jumlah koneksi yang merata , maka bisa diakatakan PCQ sudah berjalan dengan baik.

Yang kedua merupakan pengujian *download*, penulis akan melakukan *download* sebuah *file* pada situs dan *link* yang sudah ditentukan, dengan menggunakan *tools* yang men-*support multiple* koneksi yaitu IDM (*internet*

download manager), pengujian *download* dilakukan hanya pada beberapa *client* saja. Kemuadian dilakukan analisa menggunakan *tools connections* yang terdapat di *winbox* untuk mengetahui penyebaran penandaan koneksi dan besar paket yang dikirim pada masing-masing *client*.

Manajemen *Bandwitdh* dikatakan berhasil jika pada proses *download* dari beberapa client tidak melewati batas sesuai yang di telah di atur dalam queue tree. serta besar paket yang dibagikan pada masing-masing *client* juga seimbang.

2. Perancangan Pengujian Proxy

Setelah konfigurasi *proxy server* telah selasai di lakukan, maka untuk memastikan apakah *proxy server* telah berhasil menghemat alokasi *bandwitdh* sebesar 30% maka penulis melakukan 3 tahapan pengujian yaitu :

a. *Browsing*

Dalam pengujian *proxy server* yang akan di lakukan yaitu pengetesan apakah proxy sudah bisa melakukan *caching* dengan baik atau belum, dengan cara membuka situs http://speedtest.cbn.net.id pada *brower*, dan lakukan pengukuran kecepatan *bandwitdh*. Setelah melakukan test kecepatan *bandwitdh*, lakukan lagi test tersebut pada *client* yang berbeda, apabila *proxy* sudah benar dalam melakukan caching file maka test kecepatan ini akan terlihat seolah lebih cepat.

b. *Streaming*

Selanjutnya lakukan pengujian *proxy* terhadap *caching* video, dalam hal ini penulis memilih situs *youtube*. Ketika video itu dimainkan maka akan terasa berat dan mengalami *buffering*, tapi ketika video tersebut selesai dan *proxy* melakukan

caching maka ketika video tersebut dimainkan pada *client* yang berbeda, video tersebut tidak terasa berat ataupun *buffering*.

c. *Packet Loss*

Setelah itu melakukan pengujian packet loss saat mengakses salah satu situs, disini penulis memilih situs www.unisan.ac.id sebagai bahan pengujian packet loss

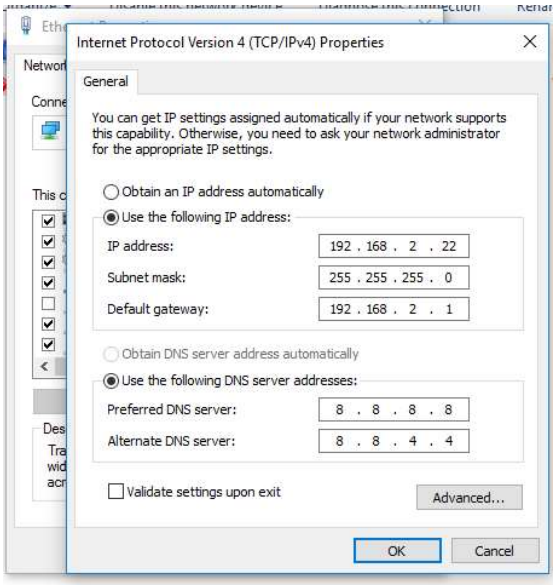
Dimana penulis akan menguji akses *client* ke situs tersebut sebelum dan sesudah menggunakan proxy server.

4.2 Implementasi Sistem

4.2.1 Implementasi Topologi Jaringan

Setelah perancangan sistem selesai dibuat, langkah selanjutnya adalah melakukan implementasi, tahap ini mengacu pada tahap perancangan yang telah dibuat pada Bab III. Yang pertama penulis akan melakukan konfigurasi pada *server* dan router mikrotik sesuai dengan topologi yang dibuat pada bab sebelumnya. Langkah-langkah yang dilakukan sebagai berikut:

1. Menyambungkan semua *client* pada hub menggunakan kabel UTPCat5e
2. Melakukan konfigurasi IP *address* pada server, pemberian *IP address* dilakuakan dengan cara klik *Start* → *Control panel* → *Network Connection*→ *Local Area Connection* → klik *properties* → klik 2 kali *internet protocol* (TCP/IP), kemudian akan muncul window untuk mengeset IP *address*:



Gambar 4.4. Setting IP

4.2.2 Implementasi Konfigurasi dasar mikrotik

Sebelum mengkonfigurasi *router* mikortik Tancapkan kabel lan dari *MikroTik port1 (ether1)* ke Modem ADSL Indihome, Tancapkan kabel lan dari *MikroTik port2 (ether2)* ke Proxy Server dan Mikrotik port3 (ether 3) ke swtich / hub , dan terakhir Tancapkan kabel lan dari *Switch* ke Komputer. Langkah selanjutnya adalah menginstal applikasi *winbox* di server untuk mengkonfiguasi router mikrotik. Adapun langkah – langkahnya sebagai berikut :

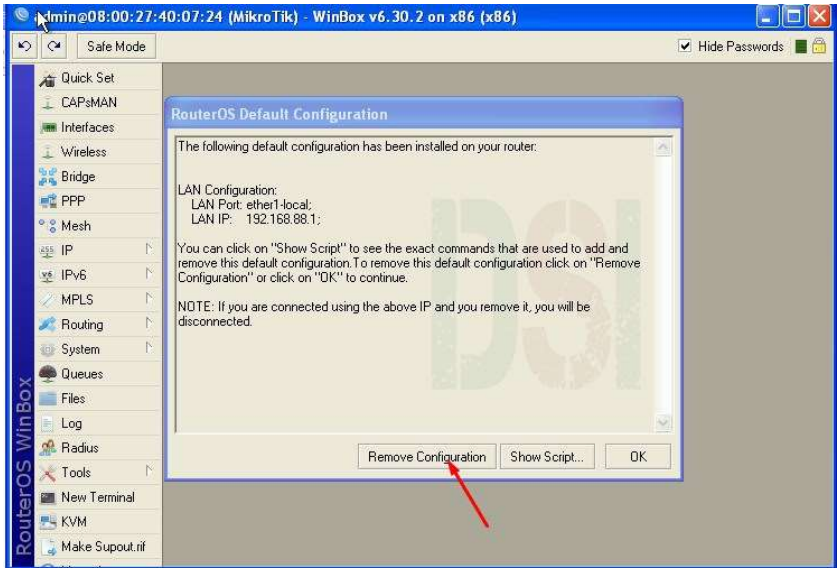
1. Remote Mikrotik RB750 menggunakan *WinBox*

Gambar 4.5. Mikrotik RB750

2. Klik "*Connect*"

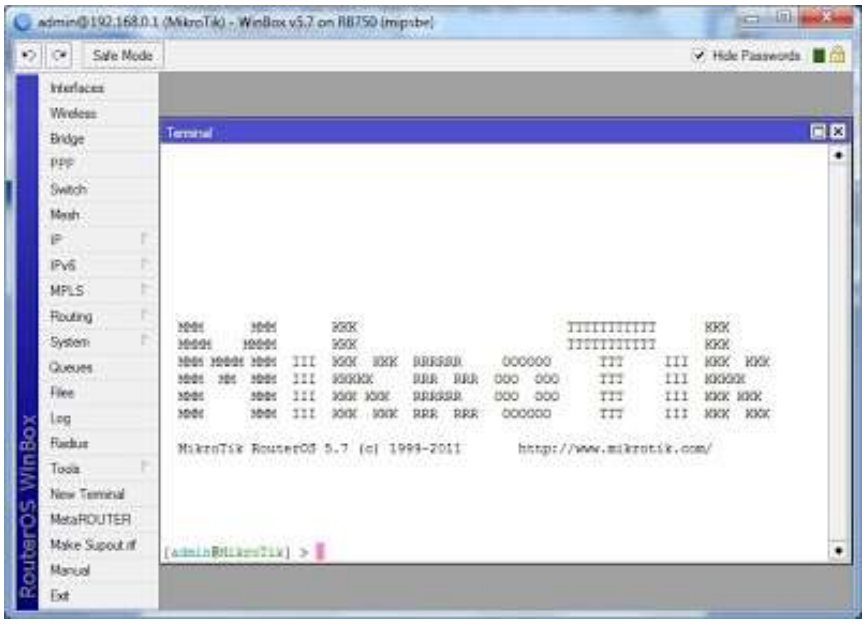
Gambar 4.6. Implementasi Konfigurasi dasar mikrotik

3. *Remove configuration* winbox



Gambar 4.7. Remove configuration winbox

4. Setelah menghapus konfigurasi awal mikrotik,mikrotik akan me*restart* dengan sendirinya , setelah itu login kembali seperti langkah yang di atas, lalu masuk ke *tools new terminal*.



Gambar 4.8. Tampilan New Terminal

5. Langkah selanjutnya yaitu mengganti nama *ether1* dan *ether2* agar memudahkan untuk mengkonfigurasi nanti. Karena ether 1 akan kita gunakan untuk di hubungkan ke modem, maka rubah nama menjadi public, sedangkan

```
/interface set 0 name=Public (port1 mengarah ke Modem ADSL Telkom
Speedy)
/interface set 1 name=Local (port2 mengarah ke Swicth Hub/Client)
```

6. Setting IP addres.

```
/ip address add address=192.168.1.2 netmask=255.255.255.0
interface=Public

/ip address add address=192.168.0.1 netmask=255.255.255.0
interface=Local
```

7. Setting Gateway

```
/ip route add gateway=192.168.1.6 (IP Aderess Modem ADSL Speedy
```

8. Setting DNS

```
/ip dns set dns=8.8.8.8 allow-remote-requests=yes
```

9. Agar client bisa terhubung dengan Internet kita perlu memberi NAT

```
/ip firewall nat add chain=srcnat out-interface=Public
action=masquerade
```

10. Agar tidak perlu repot2 setting IP address pada setiap client, kita buat DHCP secara Otomatis, IP POOL

```
 /ip pool add name=pool ranges=192.168.0.2-192.168.0.254
```

11. Selanjutnya membuat DHCP client, bertujuan kita tidak perlu input manual statik ip pada mikrotik, alasannya kadang kita tidak tau berapa ip modem yang sebenarnya, dengan melakukan dhcp-client kita bisa langsung membuat default route dan tau ip modem

```
/ip dhcp-client add interface=public use-peer-dns=no use-peer-ntp=yes
add-default-route=yes
tp=yes add-default-route=yes
```

12. Terkahir yaitu reboot mikotik, untuk memastikan agar semua settingan yang telah di buat telah berhasil, yaitu dengan mengetikkan

```
/System Reboot
```

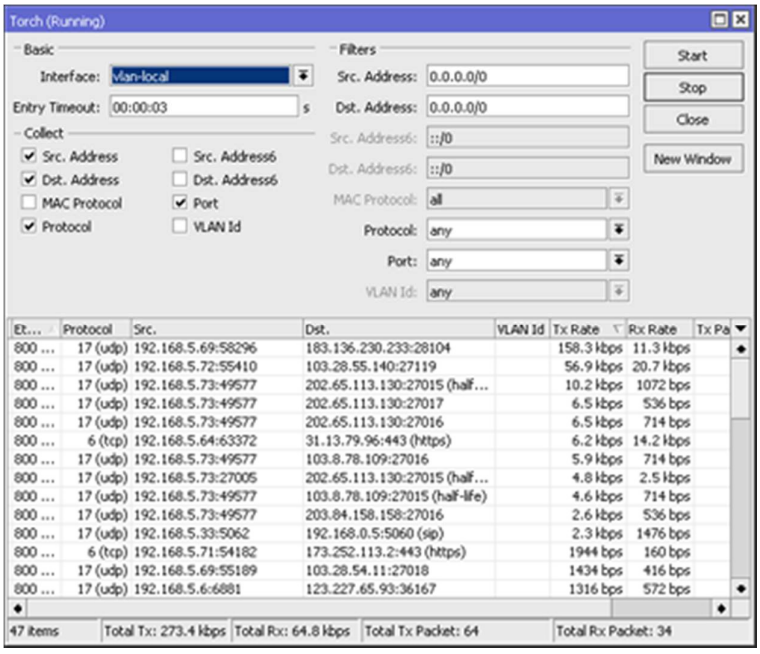
4.2.3 Implementasi Konfigurasi Manajemen *Bandwitdh*

1. Implementasi Konfigurasi Mangle

Traffic browsing dan game online dapat dibedakan berdasarkan protocol dan port yang digunakan. Fitur yang dapat digunakan untuk kebutuhan tersebut adalah mangle, dimana mangle dapat digunakan untuk menandai (*marking*) paket data berdasarkan *port, protocol, src* dan *dst address*, serta paramater lain yang dibutuhkan.

Untuk kasus ini, berarti kita harus mengetahui *protocol* dan *port* berapa yang digunakan oleh game online untuk menjalankan fungsinya. Ada 2 cara untuk mendapatkan informasi tersebut.

Pertama, dengan melakukan *Torch* pada saat *client* menjalankan game tersebut. Sehingga akan didapat *port* dan *protocol* yang digunakan.



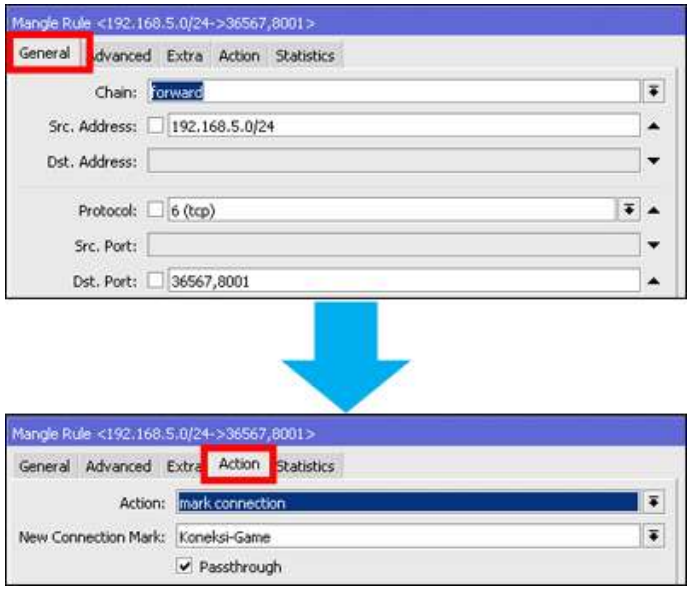
Gambar 4.9. Konfigurasi Mangle

Disamping itu, kita bisa mencari referensi lain dari internet, dimana sudah banyak yang berhasil mengetahui *port* dan *protocol* yang digunakan oleh setiap game online yang ada. Tentu setiap game menggunakan *port* dan *protocol* yang berbeda.

Jika dilihat sebenarnya tipe *traffic* yang akan diakur bisa digolongkan menjadi 2 tipe saja, yaitu *traffic* game *online* dan *traffic* selain game *online* (*browsing, chating*, dsb). Maka kita bisa membuat mangle untuk game online terlebih dahulu, baru setelahnya buat untuk selain game online berdasarkan *port* dan *protocol* yang sudah di dapat sebelumnya.

a. Mangle Game Online

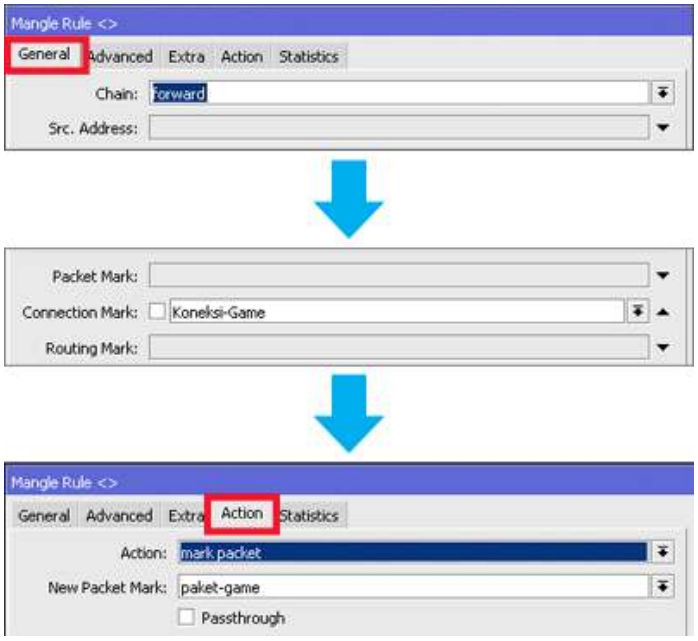
Untuk pembuatan mangle game, karena cukup banyak game-game yang akan di-mangle, akan lebih mudah jika dibuat *mark-connection* semua game terlebih dahulu dengan marking yang sama.



Gambar 4.10. Konfigurasi Mangle Game Online

Langkah di atas merupakan salah satu contoh melakukan *mark-connection* dengan game yang menggunakan *protocol tcp po*rt 36567,8001 . Untuk game lakukan dengan langkah yang sama, sesuaikan *protocol* dan *port* yang digunakan.

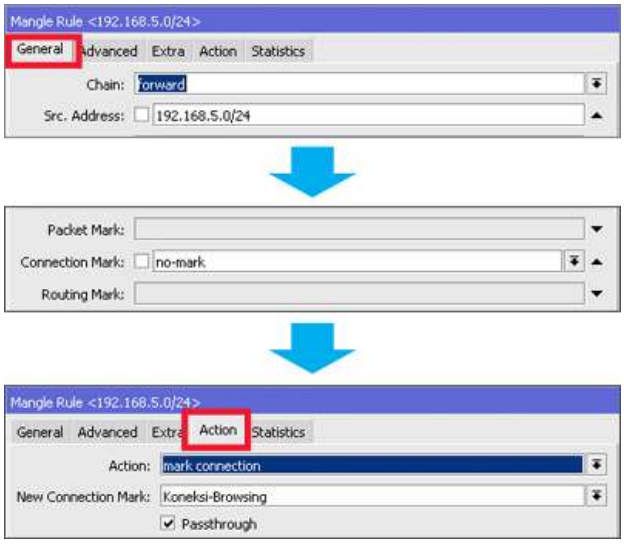
Setelah langkah mark-connection selesai, barulah dibuat *mark-packet* berdasar *mark-connection*=Koneksi-Game yang sebelumnya telah dibuat



Gambar 4.11. Konfigurasi Mark-Packet dan Mark-Connection

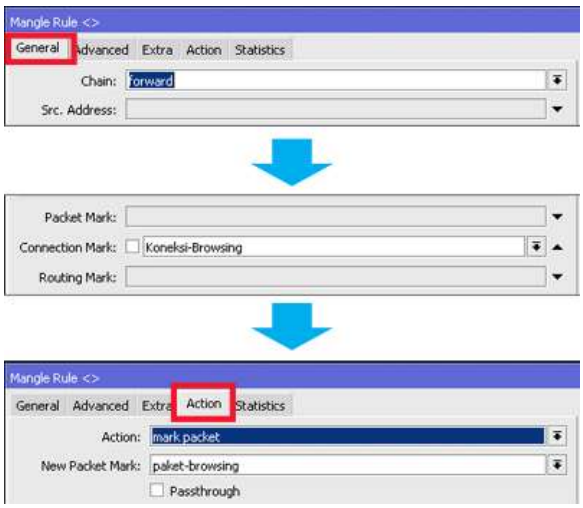
b. Mangle Selain Game Online

Pada langkah sebelumnya telah dibuat mangle untuk game. Selanjutnya tinggal dibuat mangle untuk *traffic* selain game online. Di dalam nya bisa terdapat *traffic browsing,chating,dsb*.



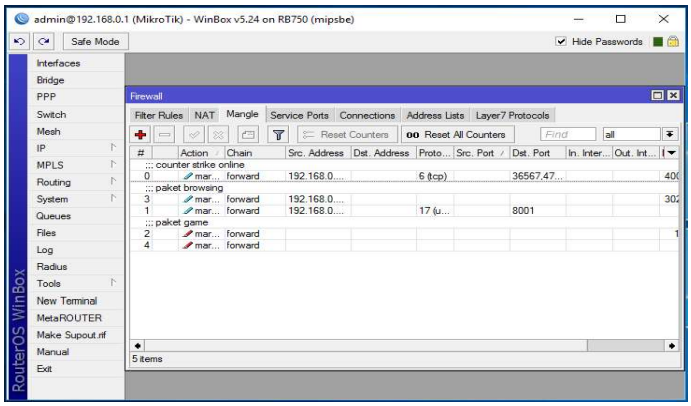
Gambar 4.12. Konfigurasi Mangle Umum

Sama dengan langkah marking pada *traffic* game, buat *mark-packet* setelah langkah *mark-connection* selesai.



Gambar 4.13. Konfigurasi Mark-Packet dan mark-Connection

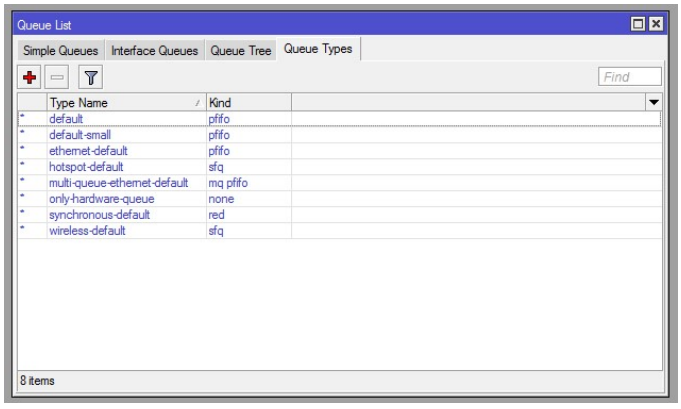
Hasil akhir dari konfigurasi mangle seperti berikut



Gambar 4.14. Hasil Konfigurasi Mangle

2. Implementasi Konfigurasi PCQ

Pengaturan pembagian *bandwitdh* ini dilakukan setelah pengaturan *mark packet* dan *mark connection* selesai. Dengan pengaturan pembagian *bandwitdh Queue Tree* ini ditujukan untuk pembagian *bandwitdh* yang lebih spesifik kepada *client*. Bagian *Queue Type* ditambahkan *Queue Type Upload* dan *Queue Type Download* dimana tipe dari *Upload* dan *Download* adalah PCQ (*Per Connection Queue*) yang dapat digunakan untuk membagi atau membatasi *traffic* untuk *multi-users* secara dinamis, dengan sedikit administrasi.



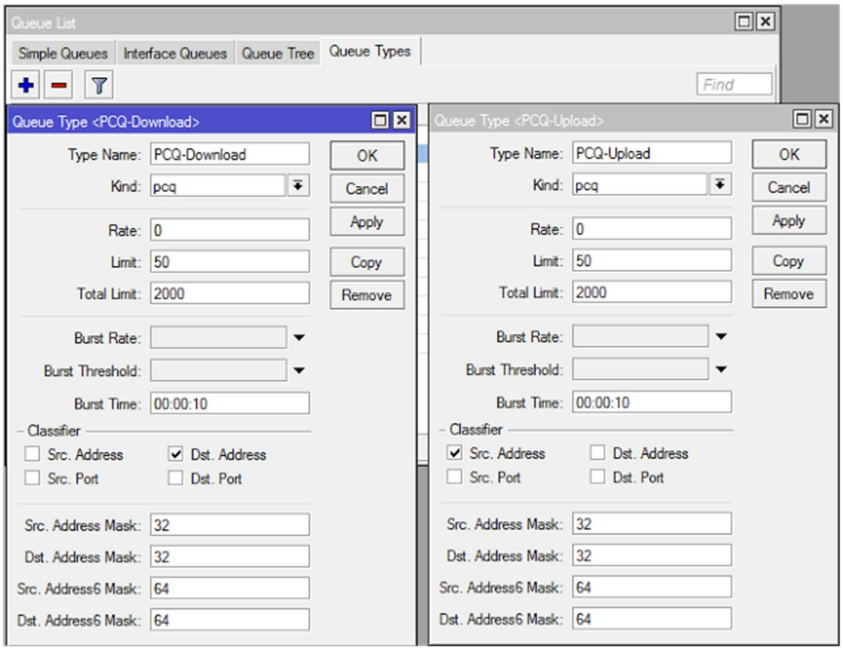
Gambar 4.15. Penambahan konfigurasi PCQ

PCQ ini digunakan untuk membagi *bandwitdh* menjadi sama rata sesuai dengan *client* yang menggunakan.Konfigurasi PCQ bisa disetting pada menu queue. Tepatnya pada tab queue type. Di tab ini, tambahkan PCQ nya.Yang perlu diperhatikan dalam PCQ adalah sebagai berikut :

Kind = PCQ

Classifier Upload = Ceklis *Src-Address*

Classifier Download = Ceklist *Dst-Address*



Gambar 4.16. Konfigurasi tipe PCQ

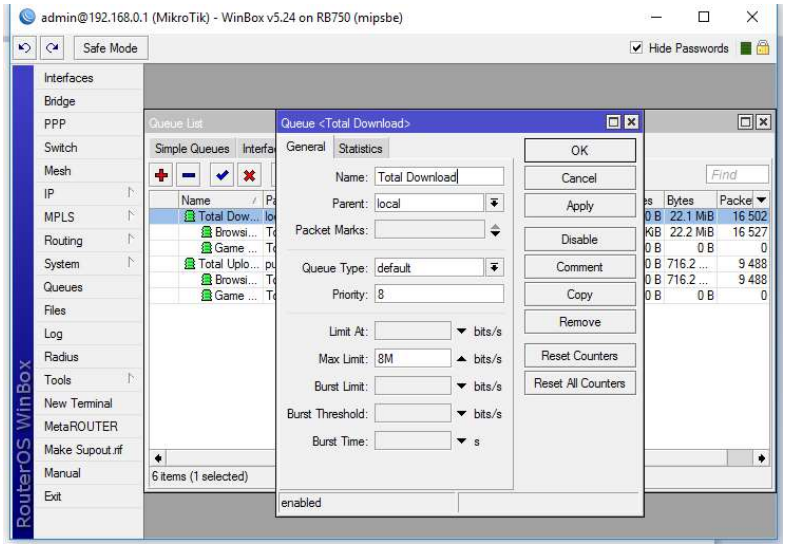
3. Implementasi Konfigurasi *Queue Tree*

Nilai limit *download* dan *upload* diambil dari jumlah total *bandwitdh download* adalah 8 Mbps dan *upload* 2 Mbps, sedangkan Jumlah client yang tersambung ada 3 *client*.

Setelah membuat pcq pada *queue type* maka selanjutnya adalah membuat *queue tree*, pilih menu *queue* >> *queue tree* >> *add new* (+) dalam kotak dialog

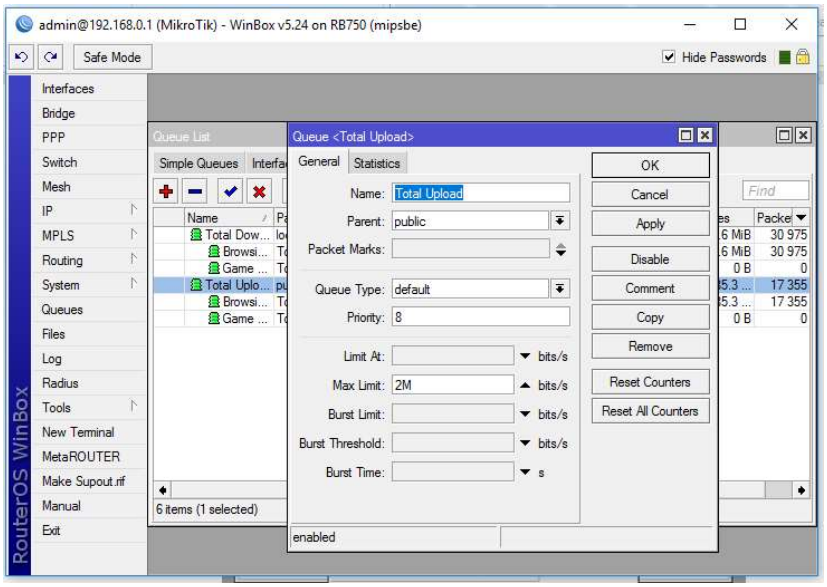
pembuatan *queue tree* yang baru kita berikan Total *Download* dan Total *Upload* *bandwitdh* nya, Parameter yang harus diperhatikan adalah *parent* dan Max-Limitnya.

Parent Download = Interface yang ke *Local*



Gambar 4.17. Penentuan *parent Download*

Parent Upload = Interface yang ke *Public*



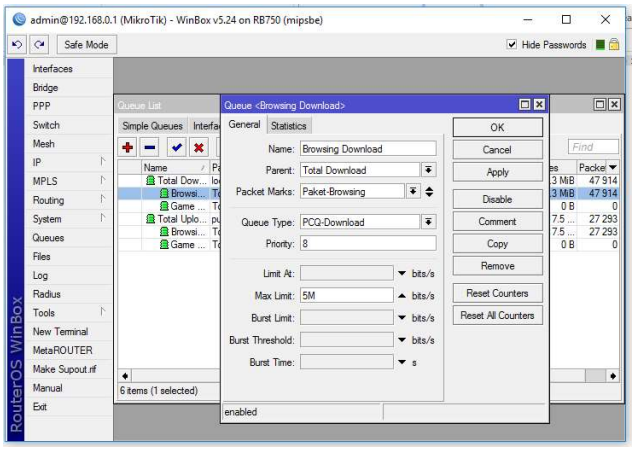
Gambar 4.18. Penentuan *Parent Upload*

Setelah itu berikan *bandwitdh* untuk *Traffic Download* terlebih dahulu , masukkan Parameter.

Parent = Total *Download*

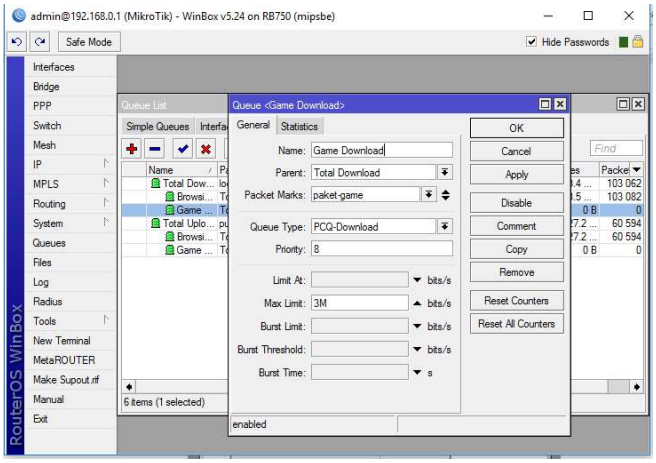
Packet Marks Browsin = *Packet-Browsing* (Sesuai dengan Packet mark yang tadi dibuat)

Queue Type = PCQ-*Download*



Gambar 4.19. Penentuan *Traffic bandwitdh* PCQ-*Download* pada *packet-browsing*

Packet Marks Download = *Packet-Game* (Sesuai dengan Packet mark yang tadi dibuat)



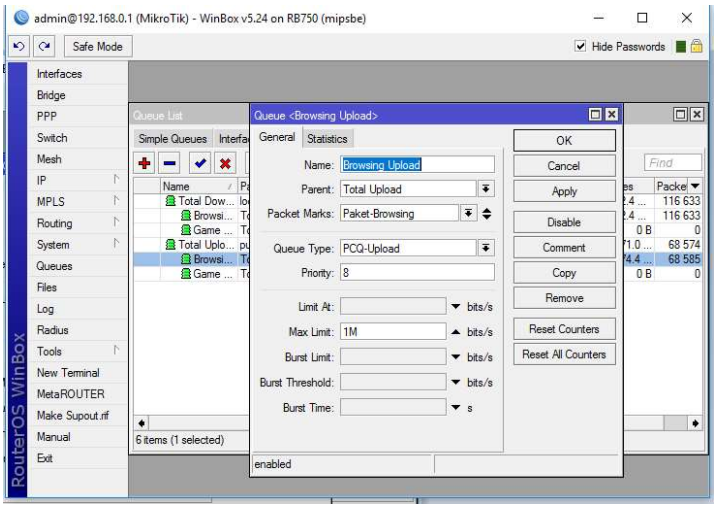
Gambar 4.20. Penentuan *Packet Marks Download* untuk *packet-game*

Setelah memberikan *traffic bandwitdh* untuk *download* , maka selanjutnya memberikan *traffic bandwitdh* untuk *upload*, parameter yang perlu di perhatikan sama seperti pada pembagian traffic download yaitu :

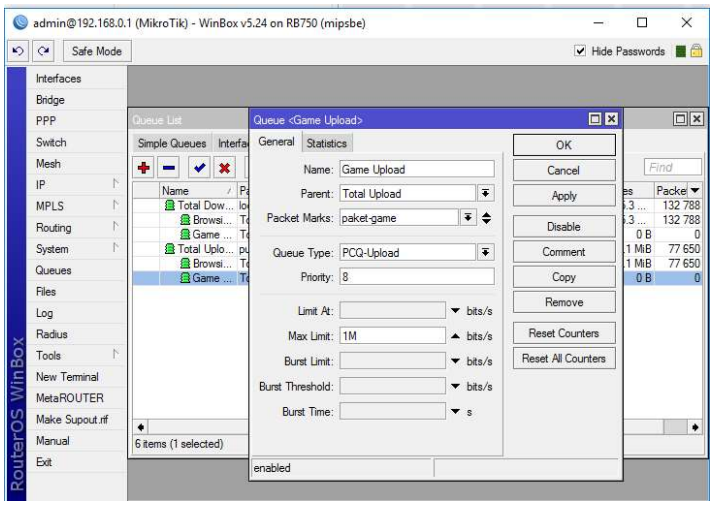
Parent = *Total upload*

Packet Marks Browsing = *Packet-Browsing* (Sesuai dengan *Packet mark* yang tadi dibuat)

Queue Type = PCQ-*Upload*

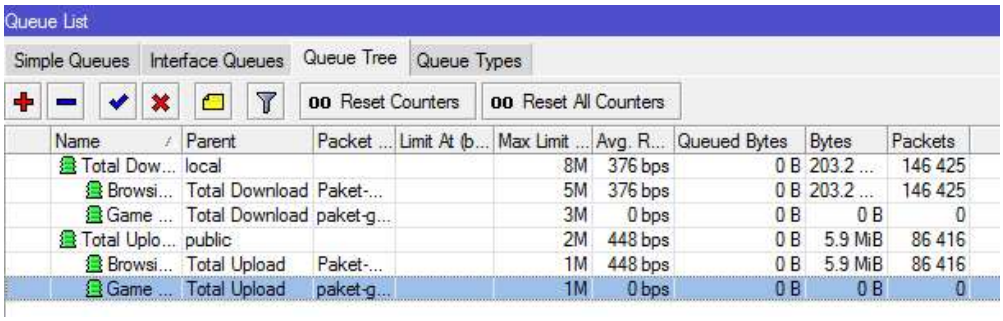


Gambar 4.21. Pembagian *Traffic Bandwitdh upload* pada *packet-browsing*



Gambar 4.22. Penentuan *Packet Marks Upload* pada *packet-game*

Setelah selesai membagi *traffic bandwitdh* untuk *download* dan *upload*,dan juga memisahkan *traffic bandwitdh* antara browsing dan game, maka proses manajemen *bandwitdh* telah selesai.



Gambar 4.23. Hasil pembuatan pcq dengan menggunakan *queue tree*

4.2.4 Implementasi Konfigurasi Proxy

Setelah mengkonfigurasi mikrotik maka langkah selanjutnya yaitu mengkonfigurasi *proxy* server , karna dalam penilitian ini penulis menggunakan *proxy* hanya untuk menyimpan *cache* agar menghemat penggunaan *bandwitdh*, maka pada poin ini penulis hanya akan menjelaskan bagaimana mengkonfigurasi *proxy* server *squid lusca* sebagai tempat penyimpan *cache*.

1. Penginstalan proxy squid lusca di Linux Ubuntu.
   - Pastikan *Ubuntu* server sudah bisa ping ke *gateway*, *client*, dan internet.
   - Masuk sebagai *root*.

```
$ sudo su
```

   - Lakukan *Update* terlebih dahulu

```
# apt-get update
```

- Lakukan penginstallan *Squid*

```
# apt-get install apache2 php5 squid squidclient squid-
cgi gcc build-essential sharutils ccze libzip-dev
automake1.9 libfile-readbackwards-perl -y
```

- *Download* paket *LUSCA* nya.

```
# wget
https://dl.dropboxusercontent.com/s/rj083vnq9sc3aye/LUSCA_HEAD-
r14809-patch.tar.bz2
```

- Setelah selesai lakukan ekstrak

```
# tar -xvjf LUSCA_HEAD-r14809-patch.tar.bz2
```

2. Tahap Pengkonfigurasian

Sebelumnya kita harus tau kode *CFLAG* dari tipe prosessor yang akan di pasang Lusca dengan cara lihat *CPU* info.

```
# cat /proc/cpuinfo
```

Setelah mengetahui tipe processor cari *CFLAG* , maka akan mendapatkan kode *CFLAG* sebagai berikut :

```
CHOST="i686-pc-linux-gnu" CFLAGS="-march=atom -O2 -fomit-frame-
pointer -pipe" CXXFLAGS="${CFLAGS}"
```

Kemudian di tambahkan dengan

```
/configure -prefix=/usr -exec-prefix=/usr -bindir=/usr/sbin -
sbindir=/usr/sbin -libexecdir=/usr/lib/squid -
sysconfdir=/etc/squid -localstatedir=/var/spool/squid -
datadir=/usr/share/squid --enable-gnuregex --enable-async-io=24
--with-aufs-threads=24 --with-pthreads --with-aio --with-dl --
enable-storeio=aufs,null --enable-removal-policies=heap --
enable-icmp --enable-delay-pools --disable-wccp --enable-snmp -
-enable-cache-digests --enable-default-err-language=English --
enable-err-languages=English --enable-linux-netfilter --
disable-ident-lookups --with-maxfd=65535 --enable-follow-x-
forwarded-for --enable-large-cache-files --with-large-files --
enable-referer-log
```

Sehingga menjadi

```
CHOST="i686-pc-linux-gnu" CFLAGS="-march=atom -O2 -fomit-frame-
pointer -pipe" CXXFLAGS="${CFLAGS}" ./configure -prefix=/usr -
exec-prefix=/usr -bindir=/usr/sbin -sbindir=/usr/sbin -
libexecdir=/usr/lib/squid -sysconfdir=/etc/squid -
localstatedir=/var/spool/squid -datadir=/usr/share/squid --
enable-gnuregex --enable-async-io=24 --with-aufs-threads=24 --
with-pthreads --with-aio --with-dl --enable-storeio=aufs,null --
enable-removal-policies=heap --enable-icmp --enable-delay-pools -
-disable-wccp --enable-snmp --enable-cache-digests --enable-
default-err-language=English --enable-err-languages=English --
enable-linux-netfilter --disable-ident-lookups --with-maxfd=65535
--enable-follow-x-forwarded-for --enable-large-cache-files --
with-large-files --enable-referer-log
```

Pindah ke direktori LUSCA dilanjut Compile LUSCA

```
# cd LUSCA_HEAD-r14809

# make clean

# CHOST="i686-pc-linux-gnu"CFLAGS="-march=atom -O2 -fomit-frame-
pointer -pipe"CXXFLAGS="${CFLAGS}"./configure -prefix=/usr -exec-
prefix=/usr -bindir=/usr/sbin -sbindir=/usr/sbin -
libexecdir=/usr/lib/squid -sysconfdir=/etc/squid -
localstatedir=/var/spool/squid -datadir=/usr/share/squid --
enable-gnuregex --enable-async-io=24 --with-aufs-threads=24 --
with-pthreads --with-aio --with-dl --enable-storeio=aufs,null --
enable-removal-policies=heap --enable-icmp --enable-delay-pools -
-disable-wccp --enable-snmp --enable-cache-digests --enable-
default-err-language=English --enable-err-languages=English --
enable-linux-netfilter --disable-ident-lookups --with-maxfd=65535
--enable-follow-x-forwarded-for --enable-large-cache-files --
with-large-files --enable-referer-log

 # make

# make install
```

Buat direktori *proxy* untuk *cache* nya, dan dilanjut merubah hak akses dari direktori tersebut menjadi hak akses *proxy*.

```
# mkdir /proxy

# chown -R proxy:proxy /proxy

# chmod -R 777 /proxy
```

Buat direktori penyimpanan log *cache* nya, dan dilanjut merubah hak akses dari direktori tersebut menjadi hak akses *proxy*.

```
# mkdir /var/log/squid

# chown -R proxy:proxy /var/log/squid
```

Download konfigurasi squid. Namun konfigurasi ini hanya HTTP saja yang bisa ter*cache*

```
# wget
https://dl.dropboxusercontent.com/s/wp4qv6yh80z0b62/fileconfig.ta
r.gz
```

Ekstrak dan copykan file konfigurasi tersebut ke dalam file /etc/squid

```
# tar -xvf fileconfig.tar.gz

# cp fileconf/* /etc/squid
```

Beri hak akses dan eksekusi semua file berekstensi .pl

```
# chmod 0755 /etc/squid/*.pl
```

Lakukan konfigurasi sesuai topologi.

```
# nano /etc/squid/acl.conf
```

Lakukan konfigurasi untuk merubah RAM

```
# nano /etc/squid/tune.conf
```

Hentikan service Squid3 pada server agar tidak terjadi bentrok dengan squid *lusca.*

```
# service squid3 stop
```

Lalu ketikkan perintah

```
# squid -f /etc/squid/squid.conf -z
```

Jalankan Squid

```
# squid -NDd1 &
```

Buat *autorun* pada squid apabila server restart bisa jalan dengan otomatis

```
# nano /etc/init.d/squid
```

Buat *script* ini

```
#! /bin/sh
#
# squid  Startup script for the SQUID HTTP proxy-cache.
 #
 # Version: @(#)squid.rc  2.20  01-Oct-2001  miquels@cistron.nl
 #
 ### BEGIN INIT INFO # Provides:          squid # Required-
Start:    $local_fs $network
# Required-Stop:     $local_fs $network
# Should-Start:      $named # Should-Stop:       $named
# Default-Start:     2 3 4 5 # Default-Stop:      0 1 6 #
Short-Description: Squid HTTP Proxy
 ### END INIT INFO
# suffield: jhealy: All local modifications commented with
"suffield"
NAME=squid
DAEMON=/usr/sbin/squid
LIB=/usr/lib/squid
PIDFILE=/var/run/$NAME.pid
# suffield: jhealy: added -F to wait for full rebuild before
serving requests SQUID_ARGS="-D -sYCF"

[ ! -f /etc/default/squid ] || . /etc/default/squid
. /lib/lsb/init-functions
 PATH=/bin:/usr/bin:/sbin:/usr/sbin
  [ -x $DAEMON ] || exit 0

grepconf () {
w="  " # space tab
sq=/etc/squid/squid.conf
# sed is cool.
res=`sed -ne '
s/^'$1'['"$w"']\+\([^'"$w"']\+\).*$/\1/p;
t end;
d;
:end q' < $sq`
[ -n "$res" ] || res=$2
echo "$res"
}

grepconf2 () {
w="  " # space tab
sq=/etc/squid/$NAME.conf
# sed is cool.
res=`sed -ne '
```

```
s/^'$1'['"$w"']\+[^'"$w"']\+['"$w"']\+\([^'"$w"']\+\).*$/\1/p;
 t end;
d;
:end q' < $sq`
 [ -n "$res" ] || res=$2  echo "$res" }
echo "$res"
}
#
# Try to increase the # of filedescriptors we can open.
#
maxfds ()

{
[ -n "$SQUID_MAXFD" ] || return
 [ -f /proc/sys/fs/file-max ] || return 0


[ $SQUID_MAXFD -le 4096 ] || SQUID_MAXFD=4096
global_file_max=`cat /proc/sys/fs/file-max`
 minimal_file_max=$(($SQUID_MAXFD + 4096))
 if [ "$global_file_max" -lt $minimal_file_max ]
then
echo $minimal_file_max > /proc/sys/fs/file-max
 fi  ulimit -n $SQUID_MAXFD
}

start () {
 cdr=`grepconf2 cache_dir /var/spool/$NAME`
case
"$cdr" in
 [0-9]*)
log_failure_msg "squid: squid.conf contains 2.2.5 syntax - not
starting!"
 log_end_msg 1    exit 1    ;;
 esac
#
 # Create spool dirs if they don't exist.
 #  if [ -d "$cdr" -a ! -d "$cdr/00" ]
then
log_warning_msg "Creating squid spool directory
structure"   $DAEMON -z
 fi
 if [ "$CHUID" = "" ]; then
 CHUID=root
fi
maxfds
umask 027
cd $cdr
 start-stop-daemon --quiet --start \
 --pidfile $PIDFILE \
--chuid $CHUID \
--exec $DAEMON -- $SQUID_ARGS < /dev/null
return $?
 }
```

```
stop () {
 PID=`cat $PIDFILE 2>/dev/null`
 start-stop-daemon --stop --quiet --pidfile $PIDFILE --name
squid
  #
  # Now we have to wait until squid has _really_ stopped.
  #
sleep 2
  if test -n "$PID" && kill -0 $PID 2>/dev/null
 then
log_action_begin_msg " Waiting"
 cnt=0
 while kill -0 $PID 2>/dev/null
 do
 cnt=`expr $cnt + 1`
 if [ $cnt -gt 24 ]
    then
 log_action_end_msg 1
   return 1
   fi
  sleep 5
   log_action_cont_msg ""
  done
  log_action_end_msg 0   return 0
  else
  return 0
 fi
}

case "$1" in
   start)
 log_daemon_msg "Starting Squid HTTP proxy" "squid"
 if start ; then
 log_end_msg $?
    # suffield: jhealy: enable interception via
tproxy   /usr/local/bin/squid-watcher start
 else   log_end_msg $?
 fi  ;;

       stop)
      log_daemon_msg "Stopping Squid HTTP proxy" "squid"
 # suffield: jhealy: disable interception via
tproxy  /usr/local/bin/squid-watcher stop
 if stop ; then
  log_end_msg $?
 else
 log_end_msg $?
  fi
 ;;

 reload|force-reload)
 log_action_msg "Reloading Squid configuration files"
 start-stop-daemon --stop --signal 1 \
  --pidfile $PIDFILE --quiet --exec $DAEMON
   log_action_end_msg 0
  ;;

   restart)
  log_daemon_msg "Restarting Squid HTTP proxy" "squid"  stop
  if start ; then
   log_end_msg $?
```

```
        # suffield: jhealy: enable interception via
tproxy   /usr/local/bin/squid-watcher start
  else
  log_end_msg $?
  fi
 ;;

    *)
 echo "Usage: /etc/init.d/$NAME {start|stop|reload|force-
reload|restart}"
 exit 3
;;
 esac
 exit 0
```

Setelah itu simpan dengan menekan tombol CTRL + X dilanjut tombol Y kemudian ENTER.

Berikan hak akses dan buat file autorun

```
# chmod 755 /etc/init.d/squid

# update-rc.d squid defaults
```

Untuk menghentikan squid3 secara autostart ketikan script

```
# nano /etc/rc.local

service squid3 stop
```

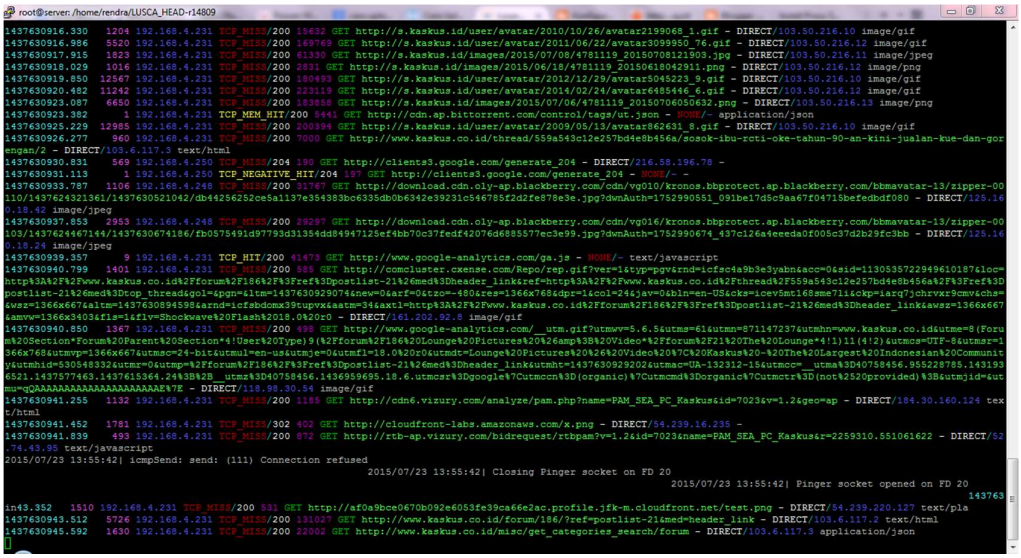
Untuk mengkonfigurasi MikroTik agar client dapat dibelokkan secara paksa ke Proxy server. Ketikan di new terminal

```
ip firewall nat add action=dst-nat chain=dstnat comment=”paksa-
ke-proxy” src-address=!192.168.3.2 disabled=no dst-port=80
protocol=tcp to-addresses=192.168.3.2 to-ports=3128
```

Agar dapat mengetahui hasil cache dari proxy server ketikan syntax

```
# tail -f /var/log/squid/access.log | ccze
```

Maka hasilnya akan terlihat seperti ini



Gambar 4.24. Hasil Konfigurasi Proxy

BAB V
PENGUJIAN, ANALISIS, DAN PEMBAHASAN

5.1 Pengujian Dan Analisis

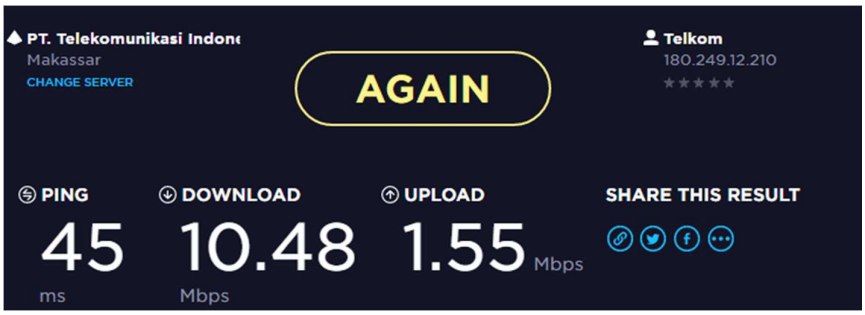
Pada bagian ini penulis akan menguji kinerja dari router mikrotik yang telah dikonfigurasi sebagai manajemen *bandwitdh* dengan metode *Per Connection Queue* dan menggunakan Tipe *Queue Tree.*

Ada beberapa kemungkinan yang terjadi dalam jaringan dimana beberapa *client* mungkin saja melakukan aktivitas yang sama yaitu *browsing (streaming)* dan downloading serta mungkin juga melakukan aktivitas yang berbeda dimana beberapa client melakukan aktivitas *browsing (streaming)* sedangkan *client* yang lain melakukan *downloading*.

Pengujian dilakukan menggunakan 3 buah laptop yang berlaku sebagai *client* dan berikut adalah *screenshots* dari hasil pengujian dan analisanya

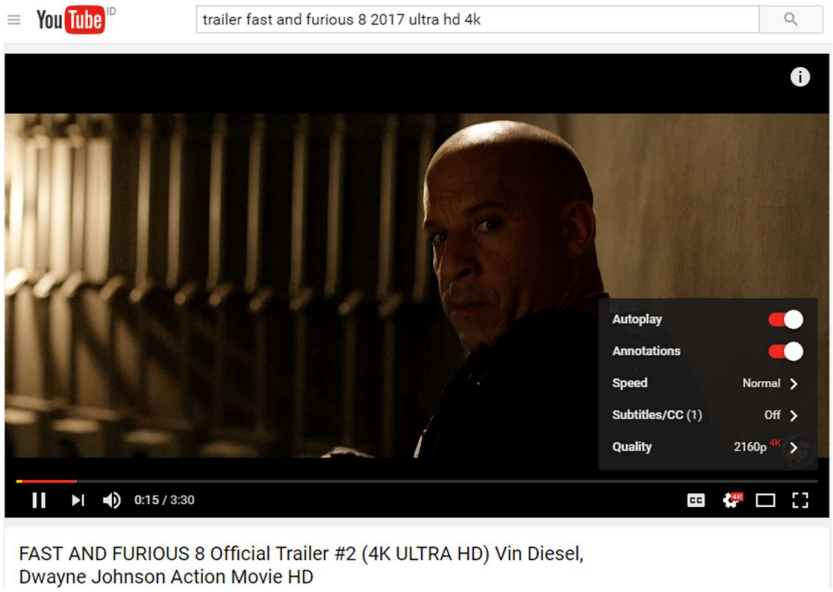
5.1.1 Pengujian Browsing

5.1.1.1 Pengujian Browsing sebelum penerapan Manajemen *Bandwitdh*



Gambar 5.1. Gambar pengujian pada situs www.speedtest.net

Dari gambar di atas dapat di lihat bahwa pemakaian *bandwitdh* pada tiap *client* tidak terbatasi karena setiap client mendapatkan akses *full bandwitdh* yang tersedia. Sehingga saat salah satu *client* melakukan aktifitas yang memakai *bandwitdh* yang berlebihan maka *client* yang lain akan mengalami *lagging* (lambat). hal ini dapat dibuktikan dengan melakukan *streaming video* 4K *ultra HD* pada situs www.youtube.com.



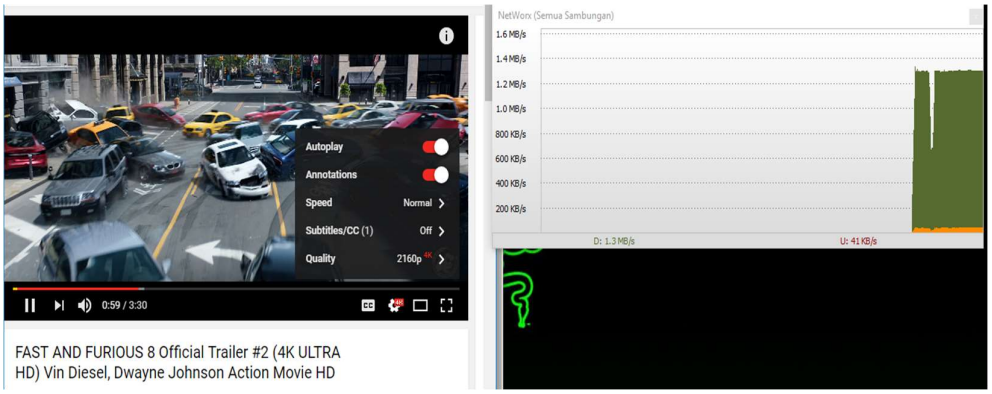
Gambar 5.2. Pengujian *buffering time* video *4k Ultra HD* 1 Client

Pada saat client 1 mengakses video pada gambar diatas diperlukan waktu 10 detik untuk menyelesaikan *buffering* pada video yang berdurasi 3 menit 30 detik, namun ketika beberapa *client* mengakses video yang sama, maka durasi untuk menyelesaikan *buffering* untuk video diatas akan bertambah, hal ini dapat dilihat pada tabel di bawah ini.

Tabel 5.1. Pengujian *browsing (streaming)* sebelum penerapan manajemen *bandwitdh*

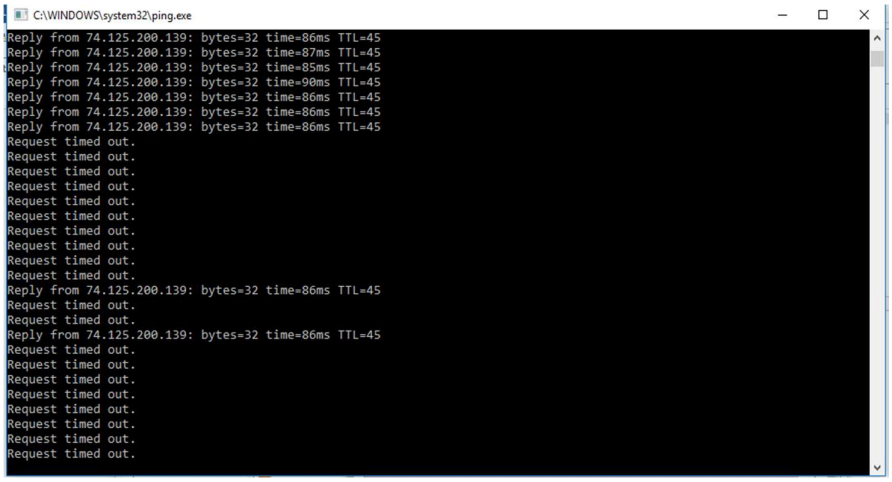
Client	Durasi *buffering* (detik)	*Bandwitdh*	*Ping* (ms)
1	398 detik	10,30 Mbps	42
2	796 detik	0 Mbps	Rto
3	1194 detik	0 Mbps	Rto

Dari tabel di atas dapat di lihat bahwa durasi untuk menyelasaikan *buffering* pada video di atas untuk *client* 1 lebih cepat dari pada *client* lainnya, hal ini di sebabkan *client* 1 lebih dulu mengakses video tersebut sehingga bandwitdh yang tersedia lebih bnyak di gunakan oleh *client* 1.



Gambar 5.3. Hasil pengujian traffic pada client 1 yang melakukan browsing (streaming)

Gambar 5.4. *Bandwitdh* yang di gunakan pada client 1 pada saat melakukan *browsing (streaming)*



Gambar 5.5. Ping pada *client* 2 pada saat *client* 1 saat melakukan aktifitas *browsing (streaming)*

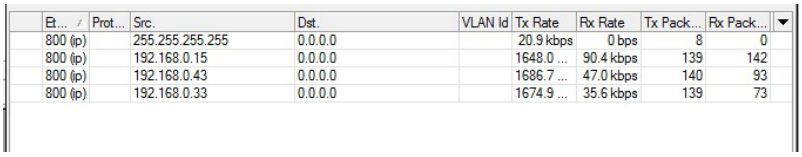
Aktifitas di atas adalah *bandwitdh*, *traffic* dan *ping* dari client sebelum penerapan manajemen *bandwitdh*, tampak jelas bahwa ketika *client* 1 melakukan *streaming* terlebih dahulu sebelum *client* lainnya, pada salah satu video di situs www.youtube.com yang berkualitas 4K ultra HD, *bandwitdh* yang akan di gunakan oleh *client* 1 akan mencapai maksimal *bandwitdh* yang telah di berikan oleh ISP.

Sehingga client lainnya akan mengalami *Request timed out* , sampai pada saat *client* 1 selesai melakukan aktifitas buffering, hal ini di sebabkan karna tidak adanya pembagian atau pembatasan dari *bandwitdh* yang akan di gunakan oleh tiap *client*.

5.1.1.2 Pengujian Browsing Setelah Penerapan Manajemen *Bandwitdh* Dan Analisisnya

Pada tahap pengujian *browsing* ini penulis akan melakukan *browsing (streaming)* sama seperti pada pengujian sebelum penerapan manajemen

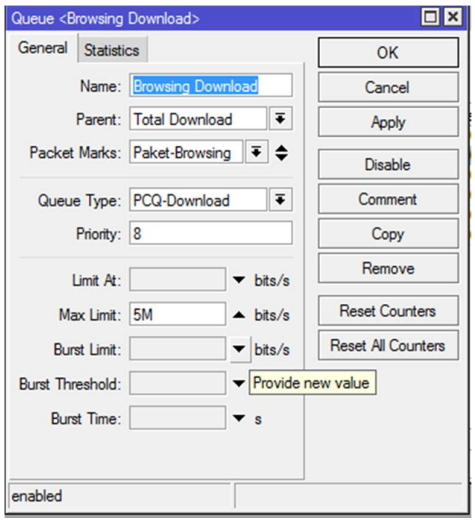
bandwitdh, hal ini bertujuan sebagai perbandingan sebelum penerapan manajemen *bandwitdh* dan sesudahnya.



Et...	Prot...	Src.	Dst.	VLAN Id	Tx Rate	Rx Rate	Tx Pack...	Rx Pack...
800 (ip)		255.255.255.255	0.0.0.0		20.9 kbps	0 bps	8	0
800 (ip)		192.168.0.15	0.0.0.0		1648.0 ...	90.4 kbps	139	142
800 (ip)		192.168.0.43	0.0.0.0		1686.7 ...	47.0 kbps	140	93
800 (ip)		192.168.0.33	0.0.0.0		1674.9 ...	35.6 kbps	139	73

Gambar 5.6. Hasil dari penerapan manajemen *bandwitdh*

Dari hasil pengujian yang dilakukan terhadap implementasi manajemen *bandwitdh* menggunakan mikrotik, di dapatkan hasil yang cukup memuaskan,hal ini terlihat pada gambar di atas, *bandwitdh* yang di peroleh *client* terbagi secara merata, pada pengujian ini penulis menetapkan *bandwitdh* untuk koneksi *browsing* adalah 5 Mbps , yang telah di atur pada pada PCQ yang di kombinasikan dengan *queue tree*.



Gambar 5.7. Hasil konfigurasi PCQ yang di kombinasikan dengan *queue tree* pada paket *browsing*

Dengan demikian pada pengujian untuk *browsing (streaming)* pada video yang berkualitas ultra HD yang berdurasi 3 menit 30 deitk sama seperti yang di lakukan

sebelum penerapan *bandwitdh* sebelumnya di dapatkan hasil yang dapat di lihat pada tabel di bawah ini.

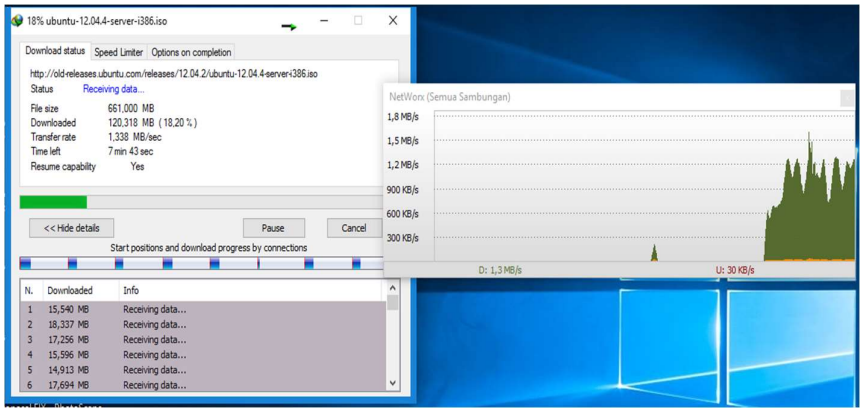
Tabel 5.2. Pengujian *browsing (streaming)* setelah penerapan manajemen *bandwitdh*

client	Durasi buffering (detik)	*bandwitdh*	Ping (ms)
1	936	1,66Mbps	132
2	936	1,66Mbps	132
3	936	1,66Mbps	132

5.1.2 Pengujian *download*

5.1.2.1 Pengujian *Download* Sebelum Penerapan Manajemen *Bandwitdh*

Pengujian berikutnya penulis akan melakukan *download* file yang berukuran 1-2GB pada *client* yang terhubung dengan jaringan lokal menggunakan *download manager* di mana penulis memakai IDM (*Internet Download Manager*) sebagai *tools* untuk dapat mengetahui kecepatan dan *bandwitdh* yang di gunakan.



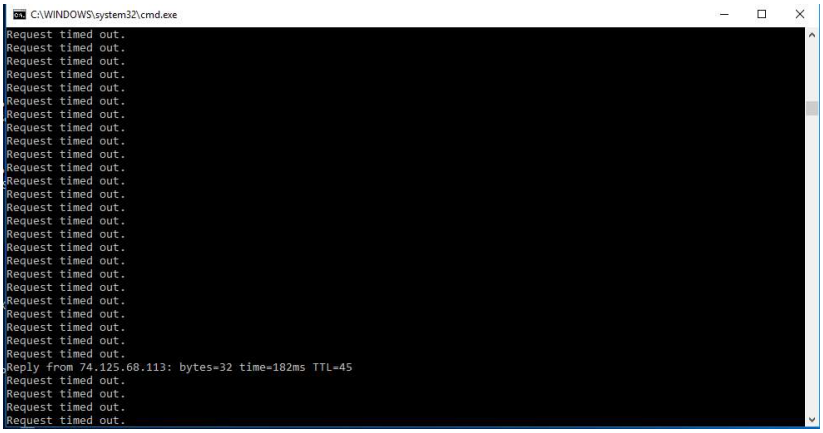
Gambar 5.8. *Client* 1 melakukan *download*

Gambar 5.9. Traffic penggunaan *bandwitdh* oleh *client* 1 saat melakukan aktifitas *download*

Tabel 5.3 Hasil pengujian download sebelum penerapan manajemen bandwitdh

Client	*Bandwitdh*	Ping (ms)
Client 1	1,3 MBps	39
Client 2	0 MBps	*Rto*
Client 3	0 MBps	*Rto*

Dari penilitian di atas dapat di pastikan bahwa pembagian *bandwitdh* ke *client* sangat tidak adil dan merata. Dimana *bandwitdh* terbanyak hanya di dapatkan oleh salah satu *client*. Dan memberikan dampak ke *client* lainya yang tidak melakukan aktifitas *download*.

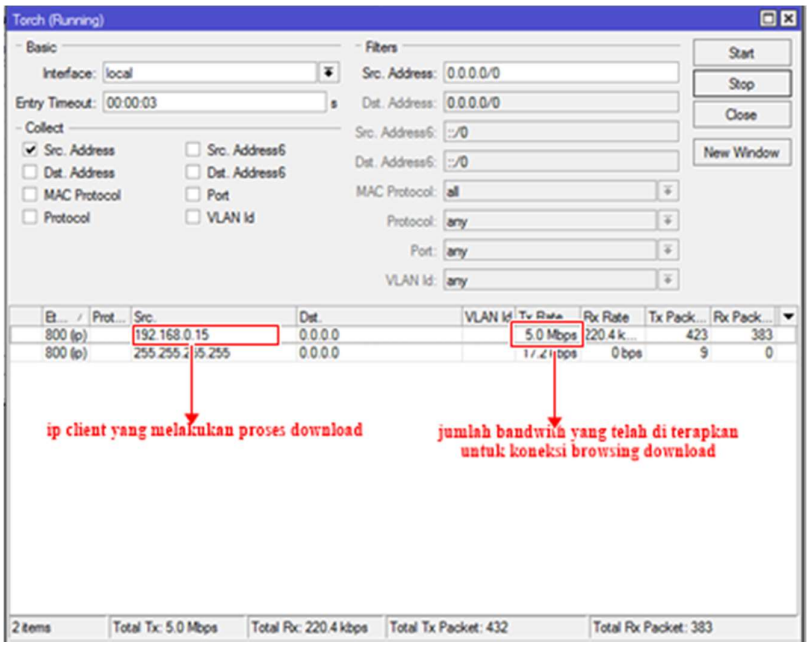


Gambar 5.10. Hasil Ping yang di lakukan oleh *client* lain

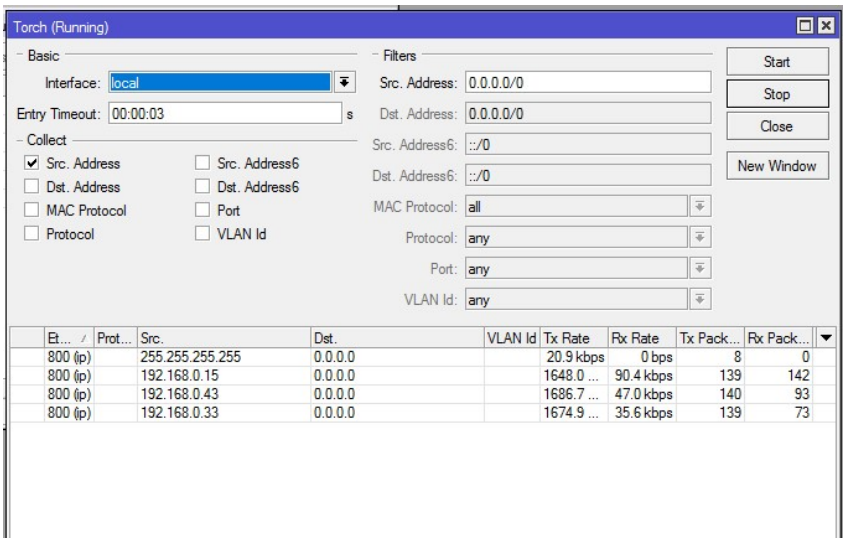
5.1.2.2 Pengujian Download Setelah Menerapkan Manajemen *Bandwitdh*

Setelah itu penulis menerapkan manajemen *bandwitdh* untuk pengujian *download* sebagai bahan perbandingan sebelum menggunakan manajemen *bandwitdh* dan sesudahnya.

Untuk pembagian *bandwitdh download* telah kita bahas sebelumnya yaitu dengan total 5 Mbps yang akan di bagikan secara merata kepada setiap client,yang aktif, dan ketika *client* yang aktif hanya satu *client* maka *bandwitdh* yang telah diterapkan di *queue tree*, akan sepenuhnya di gunakan oleh *client* yang aktif tersebut.



Gambar 5.11. Hasil penerapan manajemen *bandwitdh* yang hanya di gunakan oleh satu *client* pada saat men*download*.



Gambar 5.12. Hasil pengujian penerapan manajemen *bandwitdh* 3 *client* men*download* secara bersamaan

Hasil dari penerapan *bandwitdh* yang telah di terapkan berjalan dengan baik. Karena *bandwitdh* yang di tentukan untuk *download* terbagi secara merata pada tiap *client* di saat beberapa *client* aktif namun di saat hanya satu *client* yang aktif, *bandwitdh* yang telah ditentukan akan di gunakan sepenuhnya oleh *client* tersebut.

Tabel 5.4 Hasil pengujian download setelah penerapan manajemen bandwitdh

Client	Ip client	*Bandwitdh*	Ping (ms)
Client 1	192.168.0.15	1,66 Mbps	132
Client 2	192.168.0.43	1,66 Mbps	132
Client 3	192.168.0.33	1,6 Mbps	132

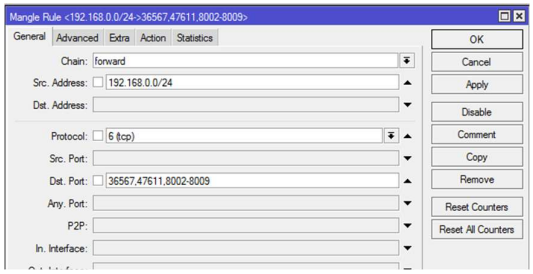
Gambar 5.13. Hasil manajemen *bandwitdh* dengan menggunakan metode PCQ

5.1.3 Pengujian Mangle Game

Untuk membedakan penggunaan *bandwitdh* antara *browsing* dan *game*, penulis melakukan penandaan pada *port game* yang akan di mainkan oleh *client*, agar *client* yang akan bermain game tidak akan terganggu oleh aktifitas *client* lainnya yang hanya melakukan aktifitas *browsing* dan *download*, seperti yang telah di bahas pada bab sebelumnya.

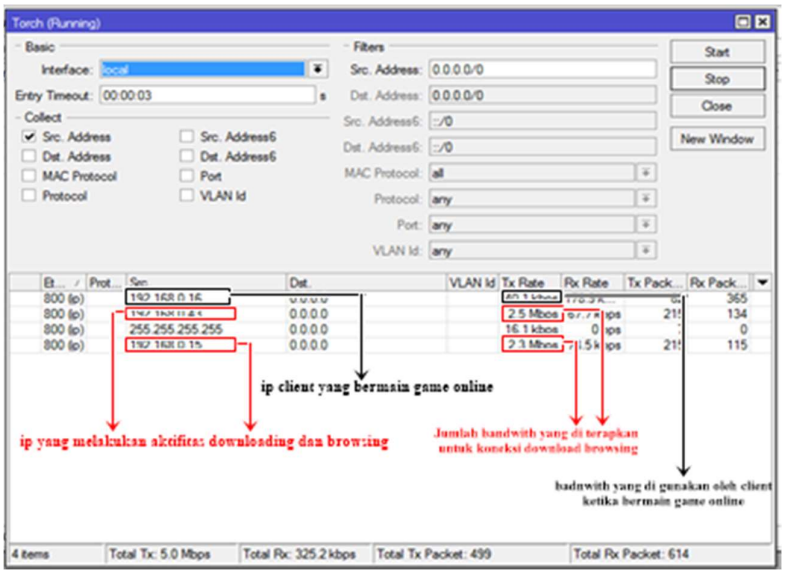
Gambar 5.14. *Game counter strike online* yang di mainkan oleh *client*

Untuk melakukan pengujian manjemen *bandwitdh* untuk pembuktian *mangle* game online, penulis menggunakan salah satu game online yang sedang popular yaitu *counter strike online*, untuk *port* dari *game counter strike* online ini adalah TCP 36567,47611,8002-8009 UDP 8001.

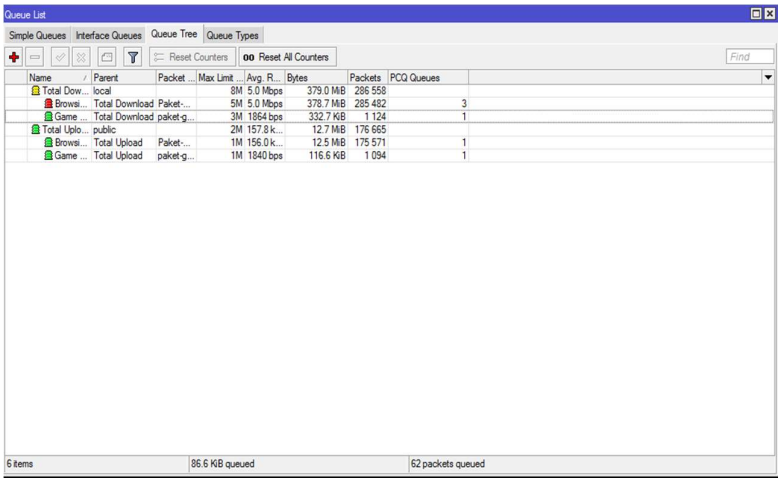


Gambar 5.15. penandaan port (mangle) game counter stike online

Untuk pengujian mangle game online ini penulis membagi tugas pada tiap client yaitu client 1 melakukan streaming, client 2 melakukan download dan client 3 akan bermain game online. Untuk pembuktiannya penulis akan memaparkan hasil traffic dan penggunaan dari tiap paket yang telah di buat sebelumnya, dengan tools mikrotik yaitu winbox.



Gambar 5.16. Hasil traffic dari ke 3 client ketika melakukan download, browsing, dan game online



Gambar 5.17. Hasil pengujian paket browser dan paket game

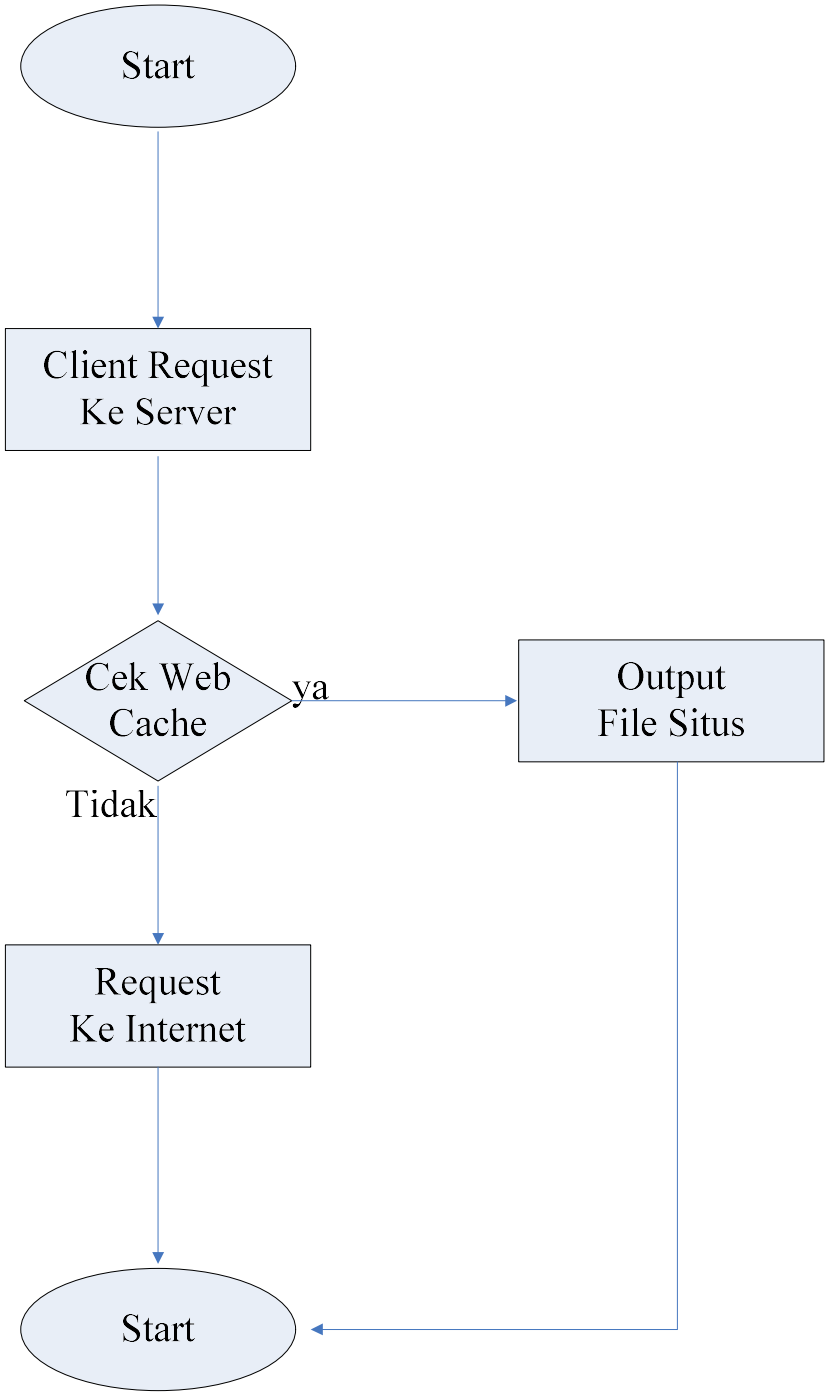
Tabel 5.5 Hasil pengujian *mangle game* dan manajemen *bandwitdh*

Client	Ip client	*Bandwitdh*	Connection Packet	Ping (ms)
Client 1	192.168.0.15	2,5 Mbps	Browser download	98
Client 2	192.168.0.43	2,5 Mbps	Browser download	98
Client 3	192.168.0.33	401 Kbps	Game download	19

5.1.4 Pengujian Proxy Server Squid Lusca

5.1.4.1 Pengujian Caching untuk Browsing.

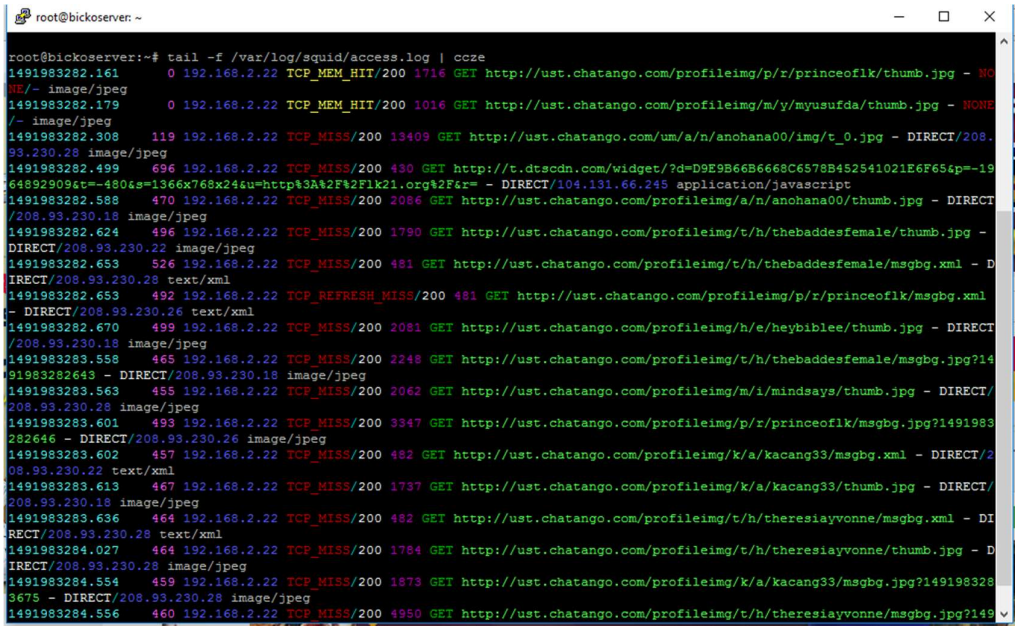
Untuk melakukan pengujian *caching* terhadap *proxy* server yang telah di konfigurasi, dalam melakukan pengujian fungsi *caching* web yang dapat menyimpan hasil *cache* website yang pernah dibuka sebelumnya terlebih dahulu dan tersimpan di dalam *cache* server pada *proxy* server, yang berguna untuk menghemat pemakaian *bandwitdh.* Berikut adalah flowchart system kerja proxy server



Gambar 5.18. *Flowchart* sistem kerja *proxy server*

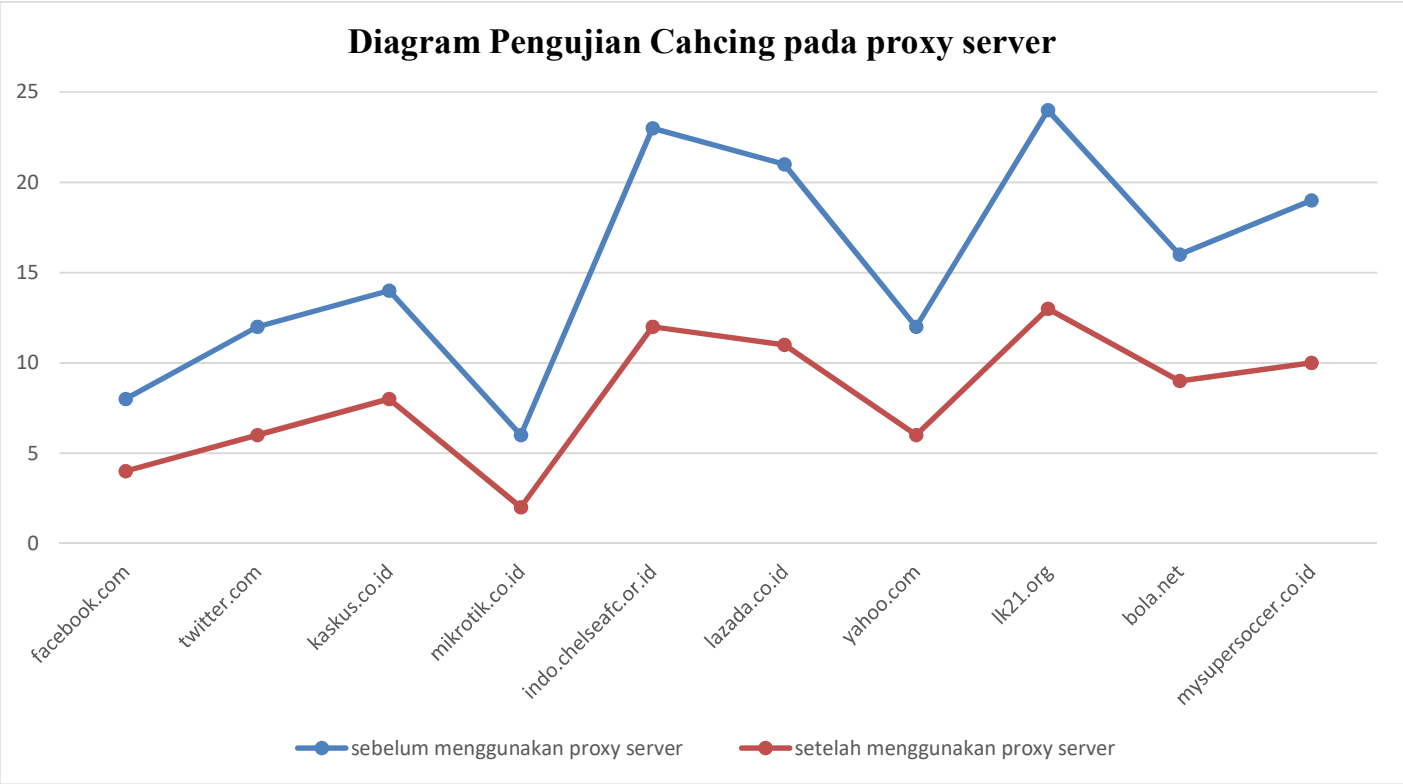
Penulis akan melakukan pengujian akses ke suatu website agar *cache* tersimpan di *cache* server dengan pengujian perbandingan waktu loading antara memakai *proxy* server dan tidak memakai *proxy* server. Tanpa *proxy* server dengan mengakses *website* www.mikrotik.co.id dibutuhkan waktu 11 detik untuk membuka keseluruhan *content* halaman *website*.

Dengan Memakai *proxy* server dan mengakses *website* yang sama yaitu www.mikrotik.co.id dibutuhkan waktu 6 detik, untuk melihat kinerja squid dapat mengetikkan perintah *#tail –f / var/log/squid/accses.log* maka akan muncul tampilan seperti pada gambar di bawah ini.



Gambar 5.19. Hasil proses *HIT* dan *MISS* pada Ubuntu server menggunakan *squid lusca*

Dari data di atas dapat di lihat bahwa memkai *proxy* server membuat kinerja lebih meningkat apabila *website* sudah pernah di akses oleh *client* lainnya dan tersimpan di *cache* server sehingga proses loading web meningkat. Dari percobaan di atas dapat di simpulkan bahwa caching telah berhasil di terapkan. Selanjutnya dalam percobaan yang kedua penulis melakukan monitoring untuk melihat jumlah *HIT* dan *MISS* yang terjadi didalam *squid proxy* server. Dimana hasil percobaan tersebut dapat di lihat pada data di bawah ini.



Gambar 5.20. Diagram Pengujian Cahcing pada proxy server

Setelah melakukan pengujian *caching* maka dilakukan pengolahan data untuk mendapatkan standar deviasi dari pengujian tersebut agar mendapatkan keragaman data dan efektivitas dari penerapan *proxy server* dengan menggunakan persamaan:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$$

s = standar deviasi (sampingan baku)

x_i = nilai ke-i

$\bar{x}$ = rata-rata

n = ukuran sampel

Tabel 5.6 Tabel Pengolahan Data Standar Deviasi Sebelum Penggunaan *Proxy*

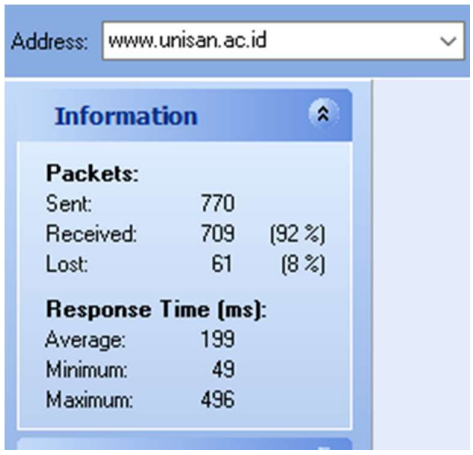
No	Situs	Sebelum Penggunaan Proxy(detik)	$(xi - \bar{x})$	$(xi - \bar{x})^2$
1	Facebook.com	8	7,5	59,25
2	Twitter.com	12	3,5	12,25
3	Kasukus.co.id	14	1,5	2,25
4	Mikrotik.co.id	6	9,5	90,25
5	Indo.chelseafc.or.id	23	-7,5	56,25
6	Lazada.com	21	-5,5	30,25
7	Yahoo.com	12	3,5	12,25
8	Lk21.org	24	-8,5	72,25
9	Bola.net	16	-0,5	0.25
10	Mysupersoccer.co.id	19	-3,5	12,25
TOTAL		155		
MEAN		15,50		
VARIAN		38,287		
STANDAR DEVIASI		6,187		

Tabel 5.7 Tabel Pengolahan Data Standar Deviasi Sesudah Penggunaan *Proxy*

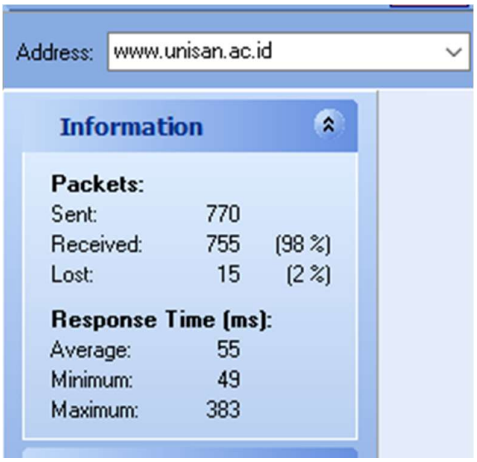
No	Situs	Sesudah Penggunaan Proxy (detik)	$(xi - \bar{x})$	$(xi - \bar{x})^2$
1	Facebook.com	4	4,1	16,81
2	Twitter.com	6	2,1	4,41
3	Kasukus.co.id	8	0,1	0,01
4	Mikrotik.co.id	2	6,1	37,21
5	Indo.chelseafc.or.id	12	-3,9	15,21
6	Lazada.com	11	-2,9	8,21
7	Yahoo.com	6	2,1	4,41
8	Lk21.org	13	-4,9	24,01
9	Bola.net	9	-0,9	0,81
10	Mysupersoccer.co.id	10	-1,9	3,61
TOTAL		81		
MEAN		8,1		
VARIAN		12,76667		
STANDAR DEVIASI		3,573047		

5.1.4.2 Pengujian Packet Loss

Pengujian terakhir yaitu pengujian *packet loss* yang terjadi pada *client* ketika mengakses situs www.unisan.ac.id, penulis akan melakukan pengujian sebelum menggunakan *proxy* server dan setelah menggunakan *proxy* server. Dalam pengujian kali ini penulis menggunakan *Axence NetTools* sebagai *tools* untuk memonitoring jumlah *packet* yang di kirim ,di terima dan packet loss.



Gambar 5.21. Hasil monitoring *packet loss* sebelum penggunaan *proxy* server



Gambar 5.22. Hasil monitoring *packet loss* setelah penggunaan *proxy* server

Dapat dilihat *packet loss* yang terjadi sebelum penggunaan *proxy* server jauh lebih banyak terjadi jika di bandingkan sesudah menggunakan *proxy* server. Hal ini di sebabkan oleh karena adanya *proxy* server yang berfungsi sebagai penyimpan web *cache* oleh *client* yang terhubung pada jaringan *local* yang telah di terapkan pada *router* mikrotik menggunakan PCQ.

5.2 Pembahasan

5.2.1 Hasil Pengujian Penerapan *Bandwitdh*

PCQ merupakan salah satu cara melakukan manajemen *bandwitdh* yang cukup mudah dimana PCQ bekerja dengan sebuah algoritma yang akan membagi *bandwitdh* secara merata ke sejumlah *client* yang aktif teknik ini sangatlah cocok untuk digunakan pada suatu jaringan yang memiliki jumlah komputer atau *client* yang banyak dengan pembatasan *bandwitdh* yang merata. hal ini dapat di lihat pada pengujian dan Analisa sebelum menerapkan manajemen *bandwitdh* dan sesudah menerapkannya seperti yang di lakukan oleh penulis pada bab sebelumnya.

Selain pembagian *bandwitdh* yang merata pada tiap komputer yang aktif, jaringan yang memiliki banyak computer atau client ketika diterapkannya manajemen *bandwitdh* dengan menggunakan teknik PCQ yang dikombinasikan dengan queue tree, untuk pembagian bandwitdhnya akan otomatis di lakukan oleh queue tree, jadi tidak perlu lagi untuk membatasi *bandwitdh* satu per satu pada setiap *client*.

Dalam penilitian ini penulis menggunakan koneksi internet yang berasal dari ISP Telkom (indiehome) yang memiliki *bandwitdh* sebesar 10 Mbps, penulis menerapkan manajemen *bandwitdh* dengan teknik PCQ dimana untuk total *download* adalah 8 Mbps dan dipisahkan 5 Mbps untuk *download browsing* dan 3 Mbps untuk *download game*, untuk Total *upload*nya sebesar 2 Mbps yaitu 1Mbps untuk *upload browser* dan 1 Mbps untuk *upload game*.

5.2.2 Hasil Pengujian *Streaming* Sebelum Penerapan Manajemen *Bandwitdh*

Pada proses *streaming* yang di lakukan oleh *client 1* sebelum penerapan manajemen *bandwitdh*, menggunakan full *bandwitdh* yaitu kurang lebih 10.30Mbps dan untuk *client* 2 dan client 3 akan mengalami lag atau *request time out*. Hal ini terjadi karena tidak adanya pembatasan penggunaan *bandwitdh* pada tiap *client* yang aktif, sehingga terjadi perebutan *bandwitdh* pada tiap *client*, dimana *client* 1 yang mengakses terlebih dahulu akan mendapatkan *bandwitdh* yang lebih besar dibandingkan *client* 2 dan *client* 3, bahkan akan memungkinkan *client* 2 dan *client* 3 akan mengalami gagal koneksi atau *request time out*.

5.2.3 Hasil Pengujian *Streaming* Setelah Penerapan Manajemen *Bandwitdh*

Setelah dilakukannya penerapan manajemen *bandwitdh* dengan teknik PCQ yang di kombinasikan dengan queue tree, proses *streaming* yang dilakukan oleh *client* 1 akan berkurang kecepatan *bufferingnya*. dimana sebelum penerapan PCQ proses *buffering* untuk *client* 1 memakai *bandwitdh* kurang lebih sebesar 10,30Mbps sedangkan untuk *client* 2 dan *client* 3 mengalami lag atau bahkan gagal koneksi, namun setelah penerapan manajemen *bandwitdh* proses *streaming* yang di lakukan oleh *client* 1 hanya akan memakai *bandwitdh* sebesar 1,6 Mbps dan untuk *client* 2 dan 3 ketika akan melakukan *streaming* sama seperti yang di lakukan oleh *client* 1, juga akan mendapatkan *bandwitdh* yang sama yaitu 1,6 Mbps hal ini disebabkan, karena total *download* untuk paket *browser* telah di atur pada PCQ yang di kombinasikan dengan queue tree adalah sebesar 5 Mbps. Karena PCQ bertugas membagi koneksi secara adil dan merata , maka ketika 3 client yang aktif

melakukan aktifitas yang sama, PCQ akan membagi *bandwitdh* yang sama rata pada client yang aktif.

5.2.4 Hasil Pengujian *Download* Sebelum Penerapan Manajemen *Bandwitdh*

Pada proses *downlaod* yang dilakukan, sebelum penerapan manajemen *bandwitdh* kurang lebih *bandwitdh* yang digunakan oleh client 1 adalah 10,38 Mbps hanya untuk satu *client* yang melakukan proses *download* tersebut, sedangkan untuk *client* 2 dan *client* 3 akan mengalami lag atau bahkan akan mengalami gagal koneksi atau *Request Time Out*. Hal ini penyebabnya sama seperti pada pembahasan proses *streaming* yaitu tidak adanya pembatasan penggunaan *bandwitdh* pada tiap *client* yang aktif, sehingga terjadi pembagian *bandwitdh* yang tidak adil dan merata pada tiap *client*, dimana *client* yang melakukan proses *download* terlebih dahulu akan mendapatkan *bandwitdh* yang lebih besar bahkan akan memkai full *bandwitdh* yang telah di berikan oleh ISP, dan untuk *client* yang lainnya,akan mengalami lag atau bahkan akan mengalami gagal koneksi atau request time out.

5.2.5 Hasil Pengujian *Download* Setelah Penerapan Manajemen *Bandwitdh*

Setelah dilakukan penerapan manajemen *bandwitdh* dengan teknik PCQ yang di kombinasikan dengan queue tree proses *download* yang dilakukan oleh *client* 1 akan berkurang kecepatan untuk proses downloadnya yang mana sebelum penerapan proses *download*nya memakai *bandwitdh* kurang lebih 10 Mbps dan untuk *client*nya mengalami lag dan bahkan akan mengalami gagal koneksi, namun setelah penerapan PCQ proses *download client* 1 berkurang menjadi 5 Mbps,

pengurangan *bandwitdh* pada proses *download* tersebut terjadi karena telah diterapkannya manajemen *bandwitdh* dengan teknik PCQ yang diatur dalam queue tree, di mana total *download* untuk koneksi *browsing* adalah 5 Mbps, namun ketika *client* 2 melakukan aktifitas yang sama seperti *client* 1 yaitu *downloading*, kecepatan proses *download* oleh *client* 1 pun ikut berkurang menjadi 2,5 Mbps dan untuk *client* 2 proses *download*nya akan mencapai 2,5 Mbps sama seperti *client* 1, dan ketika *client* 3 melakukan aktifitas *downloading* juga, maka kecepatan proses download client 1 dan client 2 berkurang menjadi 1,6 Mbps dan untuk *client* 3 kecepatan proses *download*nya akan menyerupai *client* 1 dan *client* 2 yaitu 1,6 Mbps.

Karena fungsi dari PCQ adalah membagi *bandwitdh* pada *client* yang aktif secara adil dan merata maka dalam penilitan ini manajemen *bandwitdh* dengan menggunakan teknik PCQ dikatakan berhasil, karena ketika client 1 yang aktif dan melakukan aktifitas *downloading*, *client* 1 akan mendapatkan maksimal total *bandwitdh* untuk *browser* yang telah ditetapkan yaitu 5 Mbps, dan ketika *client* 2 melakukan aktifitas *download* , *bandwitdh* dari total *bandwitdh* yang telah di tetapkan yaitu 5Mbps akan di bagi menjadi 2,5Mbps untuk tiap *client*, dan begitu untuk seterusnya, ketika *client* 3 melakukan aktifitas *download* seperti yang di lakukan oleh client 1 dan 2 maka *bandwitdh* dari total *bandwitdh* untuk *browser* akan dibagi menjadi 3 dan hasilnya adalah 1,6Mbps.

5.2.6 Hasil Pengujian *Mangle Game Online*

Pada penelitian ini penulis akan memisahkan *bandwitdh* yang akan digunakan *browsing* dan *game online* agar ketika client yang sedang bermain *game online*, Yang dimaksud *game online* disini adalah *game* yang sudah ter*install* di komputer *client* , kemudian dimainkan secara *online*, bukan *game* yang disediakan oleh *website-website* tertentu. untuk membedakan penggunaan *bandwitdh* antara *browsing* dan *game*, penulis telah melakukan penandaan pada *port game* yang akan di mainkan oleh *client*, agar *client* yang akan bermain *game* tidak akan terganggu oleh aktifitas *client* lainnya yang hanya melakukan aktifitas *browsing* dan *download*.

Mangle sendiri memiliki fungsi untuk menandai sebuah koneksi atau paket data, yang melewati *router*, masuk ke *router*, ataupun yang keluar dari *router*. Pada penilitian kali ini penulis mengkombinasikannya dengan manajemen *bandwitdh* dengan teknik PCQ ada beberapa jenis penandaan (Mark) yang ada pada *Mangle* yaitu *Packet Mark* (Penandaan Paket), *Connection Mark* (Penandaan Koneksi), setelah melakukan *packet mark* dan *connection mark* untuk *game online* di *mangle*, maka selanjutnya menghubungkannya di PCQ dengan queue tree, dan menetapkan *bandwitdh* yang akan di gunakan untuk *game online*, pada penilitian ini penulis menetapkan *bandwitdh* untuk *download game* adalah 3 Mbps dan *upload game* sebesar 1 Mbps.

5.2.7 Hasil Pengujian *Proxy Server*

Untuk penunjang kestabilan koneksi internet yang ada penulis menggunakan *proxy* server dengan tujuan menghemat *bandwitdh* yang telah di bagi dalam manajemen *bandwitdh*. Hal ini dapat tercapai karena adanya proses penyimpanan web *cache* pada proxy server, setiap *client* yang mengakses satu situs maka *cache* dari *client* tersebut akan tersimpan di *proxy* server dan ketika situs yang *cache*nya sudah tersimpan pada *proxy* server akan di akses kembali oleh *client* lain, maka *client* tersebut akan mengakses situs tersebut tanpa menggunakan *bandwitdh* yang tersedia, karena hanya mengambil dari *cache* yang tersimpan pada *proxy* server, hal ini sangat berguna untuk mengurangi pemakaian *bandwitdh* pada jaringan yang memiliki banyak komputer atau *client* dan telah di menerapkan manajemen *bandwitdh*. Namun dalam hal ini ada situs yang tidak bisa tercache dikarenakan adanya pembaruan algoritma pada situs tersebut sehingga proxy tidak bisa melakukan caching pada situs tertentu. tersebut diantaranya adalah www.youtube.com

5.2.8 Hasil Pengujian *Packet Loss*

Selain digunakan untuk menyimpan web *cache*, *proxy* server juga berguna untuk meminalisir *Packet lost* yang terjadi, *Packet lost* dapat terjadi disebabkan oleh jaringan yang digunakan terjadi *noise* atau gangguan menyebabkan naik turunya kecepatan internet yang disebabkan oleh padatnya pengguna

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil penelitian, pengujian, dan pembahasan, maka dapat ditarik kesimpulan bahwa :

1. Penggunaan kuota *bandwitdh* yang berlebihan dapat dimanajemen menggunakan metode PCQ yang ditunjang dengan penambahan *proxy server*. Pola ini dapat diterapkan pada AVNAH NET *Game center* dengan kinerja yang cukup memuaskan terbukti dengan pembagian kuota *bandwitdh* 10Mbps secara merata ke setiap *client* secara otomatis. Metode PCQ dapat membagi *bandwitdh* yang tersedia dengan cukup adil dan merata dimana 10Mbps tersebut dibagi menjadi dua kuota yaitu 8Mbps *download* dan 2Mbps untuk *upload*. Sedangkan 8Mbps dibagi lagi menjadi 2 kuota yaitu 5Mbps untuk *browsing* dan 3Mbps yang dibagi untuk akses *game*. Serta dengan adanya *proxy server* sebagai penyimpan *web cache* maka *client* yang tidak melakukan aktifitas *downloading* dan *gaming* pun tidak akan terganggu saat melakukan *browsing*. Begitupun dengan *packet lost* yang terjadi tidak terlalu bermasalah karena padatnya pengguna. Konfigurasi yang dilakukan tersebut telah dibagi dan dijalankan secara otomatis kepada setiap *client* yang aktif.
2. Dengan dibuatnya penandaan pada *port game,* koneksi untuk *game online* tidak tercampur aduk dengan *client* yang hanya melakukan *browsing* dan

downloading. Pengaksesan kuota bandwitdh untuk *Game online* itu sendiri tidak akan terganggu ketika ada aktifitas *download* dari *client* lain.

Hal ini tentunya lebih baik dibandingkan beberapa warnet yang menerapkan metode lain yaitu tanpa melakukan manajemen kuota *bandwitdh,* sehinnga dalam pemakainannya tersebut menjadi tidak merata.

6.2 Saran

Adapun saran yang dapat penulis ajukan kepada penelitian-penelitian berikutnya adalah :

1. Metode manajemen bandwitdh yang menggunakan pcq dan dikombinasikan dengan queue tree masih memiliki beberapa kekurangan, yakni masih bisa di tembus oleh aplikasi-aplikasi pihak ketiga , contohnya Internet Download Manager.
2. Penambahan proxy server sebagai device penunjang kestabilan bandwitdh harus selalu memperhatikan pembaruan-pembaruan pada konfigurasinya.
3. Untuk melakukan penandaan port pada mangle game yang telah dibuat harus memperhatikan kembali port-port yang akan digunakan

DAFTAR PUSTAKA

Aris Syaifudin, Mahmud Yunus, R. S. (2013). Perbandingan Metode Simple Queues Dan Queues Tree Untuk Optimasi Manajemen Bandwidth Jaringan Komputer Di Stmik Ppkia Pradnya Paramita Malang, *4*(2), 60–74.

Kalsum, T. U., & Supardi, R. (2015). Implementasi Dan Analisa Per Connection Queue (PCQ) Sebagai Kontrol Penggunaan Internet Pada Laboratorium Komputer, *11*(2), 139–148.

Kosasih, S. (2008). Pengalokasian Bandwith Secara Otomatis Menggunakan Metode Per Connection Queue, (372), 207–213.

Mansfield Nial, Yogyakarta, *Practical TCP/ IP: Mendesain, Menggunakan, dan Troubleshooting Jaringan TCP/IP*, Andi Offset

Mujahidin, T. (2011). Os Mikrotik Sebagai Manajemen Bandwidth Dengan Menerapkan Metode Per Connection Queue. *Naskah Publikasi*.

Purba, minda mora & syamsu H. (2010). Optimalisasi Manajemen Jaringan Dengan Menggunakan Mikrotik RouterOS. *Access*, 491. Retrieved from www.mikrotik.com

Saniya, Y., Priyono, W. A., & Ambarwati, R. (2013). Sistem Manajemen Bandwidth dengan Prioritas Alamat IP Client, 1–6.

Simarmata Janner, 2006*, Pengamanan Sistem Komputer*, Yogyakarta, Cv. Andi Offset.

Sofana Iwan. 2013, *Membangun Jaringan computer* , Informatika Bandung

Towidjojo Rendra, 2014, *Mikrotik Kung Fu Kitab 3 Manajemen Bandwitdh*, Jasakom

Wijaya, Irvanda Rachmat, 2015. *Analisis dan implementasi proxy server sebagai web Caching, blocking situs, dan monitoring menggunakan centos 6 di smk ganesha tama boyolali*.

Yulianti, L. (2015). Analisa Pemanfaatan Proxy Server Sebagai Media Filtering Dan Caching Pada Jaringan Komputer, *11*(1), 81–90.