PENINGKATAN PERFORMA JARINGAN INTERNET RT/RW NET DI DUSUN WARENGAN DESA BUBUK MENGGUNAKAN METODE HIERARCHICAL TOKEN BUCKET

IMPROVED PERFORMANCE OF RT / RW NET INTERNET NETWORK IN WARENGAN VILLAGE POWDER HAMLET USING HIERARCHICAL TOKEN BUCKET METHOD

Mohammad Ferry Kurniawan[1], Sony Panca Budiarto[2]
[1,2]STIKOM PGRI Banyuwangi; Jl. Ahmad Yani, 80 Banyuwangi, Telp/Fax 0333 417902
e-mail: * estuhandayani@stikombanyuwangi.ac.ic

Abstrak

Penggunaan internet pada jaringan RT/RW net di Dusun Warengan Desa Bubuk mengalami peningkatan seiring dengan kebijakan pemerintah yang mewajibkan kegiatan belajar dari rumah selama masa pandemi. Dengan adanya peningkatan jumlah akses internet yang disebabkan oleh pandemi serta pola penggunaan yang bervariasi berimbas terhadap menurunnya performa jaringan yang ada, ditambah juga dengan belum adanya manajemen *bandwidth* yang mumpuni pada jaringan lokal ini. Dari permasalahan diatas bisa diselesaikan dengan perbaikan manajamen *bandwidth* dengan metode *hierarchical token bucket.* Metode *Hierarchical token bucket* bekerja dengan cara membagi traffic atau *bandwidth* secara terstruktur pada *router* mikrotik sehingga tiap *client* atau pengguna internet mendapat jumlah *bandwidth* yang merata sesuai dengan konfigurasi yang diterapkan. Dengan penerapan atau implementasi dari metode *hierarchical token bucket* ini diharapkan kinerja dari *router* mikrotik menjadi semakin efisien sehingga pembagian *bandwidth* kepada *client* lebih tepat dan terarah.

Kata Kunci : *Hierarchical token bucket, Mikrotik, RT/RW net*

Abstract

Internet usage on the RT / RW net network in Warengan Village Has increased in line with government policies that require learning from home activities during the pandemic. With the increase in the number of internet access caused by the pandemic and varying usage patterns, the impact of the decline in the performance of existing networks, coupled with the absence of qualified bandwidth management on this local network. From the above problems can be solved by improving bandwidth management with the hierarchical token bucket method. Hierarchical token bucket method works by dividing traffic or bandwidth in a structured manner on a mikrotik router so that each client or internet user gets an even amount of bandwidth according to the configuration applied. With the implementation or implementation of the hierarchical token bucket method, it is expected that the performance of the mikrotik router becomes more efficient so that the distribution of bandwidth to clients is more precise and directed.

Keywords : *Hierarchical token bucket, Mikrotik, RT/RW net*

1. PENDAHULUAN

Internet sudah menjadi konsumsi umum bagi masyarakat Indonesia, kebutuhan akan koneksi internet juga semakin meningkat dikarenakan pemerintah memberlakukan kebijakan *learn from home* dan juga *work from home* yang mana kebijakan tersebut memfokuskan semua kegiatan yang awalnya

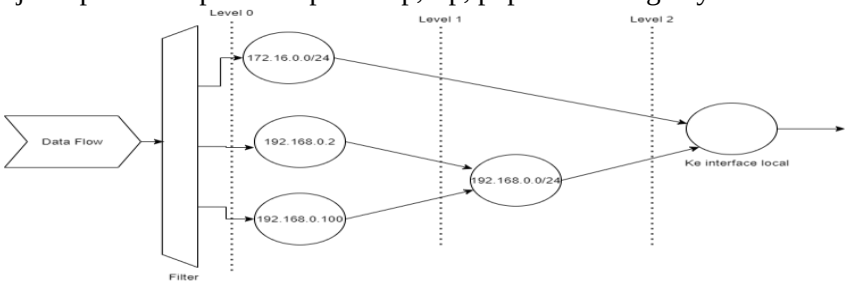
tatap muka (*offline*) menjadi kegiatan daring (*online*). Akan tetapi hal tersebut menjadi halangan karena berbagai faktor seperti lokasi yang sulit terjangkau oleh jaringan internet.

Jaringan internet pada Dusun Warengan Desa Bubuk masih tergolong daerah yang sulit terjangkau koneksi internet dikarenakan faktor geografis serta pembangunan infrastruktur yang belum merata dan juga jangkauan sinyal dari provider seluler pun masih terbilang sulit. Salah satu *ISP* (*Internet Service Provider*) yang baru saja masuk di Dusun Warengan Desa Bubuk adalah Indihome. Indihome merupakan layanan penyedia internet dari perusahaan telkomsel, namun bagi masyarakat yang rata - rata bekerja sebagai petani biaya instalasi dan harga iuran bulanan dari Indihome masih terlalu mahal bagi mereka.

Dari permasalahan tersebut, penulis memutuskan untuk menggunakan layanan *ISP* Indihome serta menyalurkan koneksi internet berbasis RT/RW net kepada masyarakat sekitar yang membutuhkan dengan biaya yang relatif lebih murah. Namun selang beberapa waktu dan seiring bertambahnya jumlah *client* atau pengguna, serta pola penggunaan yang bervariasi mengakibatkan kualitas jaringan internet mengalami penurunan dan menyebabkan aktivitas *client* atau pengguna dalam berinternet menjadi terkendala.

Berdasarkan pada latar belakang tersebut, penulis berencana untuk mengimplementasikan metode *hierarchical token bucket* pada jaringan internet RT/RW net Di Dusun Warengan Desa Bubuk supaya bisa memberikan perubahan secara signifikan pada kualitas koneksi serta memberikan kenyamanan dalam kegiatan daring para *client* atau pengguna yang ada.

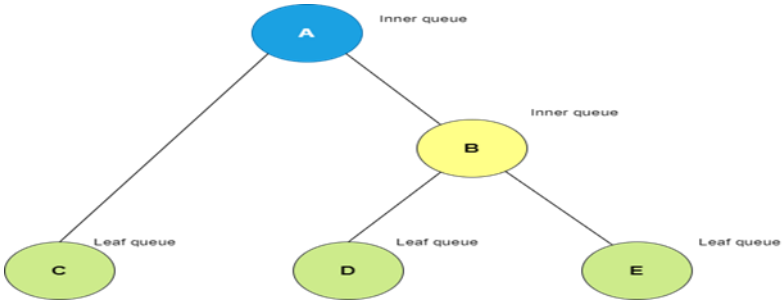
htb adalah suatu metode pada router mikrotik dimana pengguna bisa melakukan *bandwidth manajement* atau queue dengan pola hirarki, pola hirarki sendiri bisa terdiri dari ip *client* ataupun bisa juga terdiri dari jenis protokol aplikasi seperti http, ftp, pop3 dan sebagainya.



Gambar 2.1 : Visualisasi dasar *Htb*

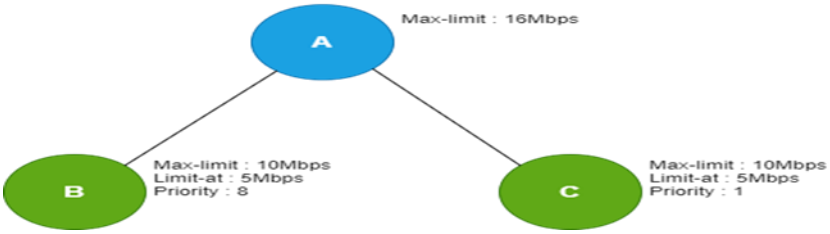
Manajemen *bandwidth* dari metode ini pada dasarnya adalah membagi-bagi tiap *client* kedalam suatu kelas yang terdiri dari parent dan child, keuntungan dari metode ini yang paling penting adalah pembagian *bandwidth* secara otomatis yang berasal dari parent menuju child, dimana jika salah satu child sedang berada dalam kondisi idle dan tidak ada traffic sama sekali, sistem router akan otomatis mengkalkulasi sisa *bandwidth* yang ada serta membagikan kepada kelas child yang yang lain. Pada router os, dikenal ada dua limitasi dengan sebutan *staged limitation* atau limit secara bertahap. Limit ini dibagi menjadi dua yaitu :

1. CIR (*Commited information rate*)
   Maksud dari CIR adalah dalam keadaan terburuk, selama *bandwidth* masih tersedia, maka *client* akan mendapatkan *bandwidth* sebesar *limit at.*
2. MIR (*Maximal information rate*)
   Sedangkan maksud dari MIR ialah *client* akan mendapatkan *bandwidth* sebesar *max limit* jika ada *bandwidth* yang tersedia. *Router* akan berusaha mengalokasikan *bandwidth* sebesar *limit at* terlebih dahulu kepada tiap *client*, jika ada *bandwidth* yang tersisa maka *router* akan menghitung siapa *client* yang boleh mendapat *bandwidth* tambahan hingga sebesar *max limit.*



Gambar 2.2 : Struktur Tree Htb

Bisa dilihat pada gambar 2.2 ada beberapa *node* yaitu A, B, C, D dan E. Yang perlu diketahui adalah semua *queue* yang langsung mengatur limitasi pada *client* itu disebut dengan *inner queue* atau parent, jika dilihat dari strukturnya C adalah anggota langsung dari A namun C tetap dianggap sebagai *leaf queue* dan diperlakukan setara dengan *leaf queue* lain seperti D dan E. Untuk A dan B bisa disebut sebagai *inner queue* atau disebut juga dengan *parent*. Selanjutnya adalah teori sederhana cara kerja dasar dari *Hierarchical token bucket*.



Gambar 2.3 : Contoh Hirarki Htb

Sebagai contoh pada gambar 2.3 suatu jaringan mempunyai *parent queue max limit* sebesar 16Mbps serta mempunyai 2 *child* atau *client* yaitu B dan C dengan konfigurasi yang hampir sama yaitu dengan *max limit* di 10Mbps, *limit at* 5Mbps, yang membedakan keduanya adalah *priority* atau prioritas dari tiap child dengan B prioritasnya 8 dan C prioritasnya 1. Teorinya, router pada awalnya akan memberikan *bandwidth* kepada tiap child sebesar limit at yaitu 5Mbps, dengan total *bandwidth* yang terpakai adalah 10Mbps namun masih ada sisa 6Mbps pada *parent queue*. Selanjutnya *router* akan membagi sisa *bandwidth* kepada *child* yang mempunyai *priority* yang lebih tinggi yaitu C dan pada akhirnya C akan mendapat total *bandwidth* sebesar *max limit* yaitu 10Mbps sedangkan B akan mendapat total *bandwidth* sebesar 6Mbps. Jika *child* C sedang *idle* atau tidak ada *traffic* sama sekali maka router akan mengalokasikan *bandwidth* yang tersedia kepada *child* B hingga *max limit* yang ditentukan yaitu 10Mbps selama *bandwidth* yang dibutuhkan masih tersedia.

ISP merupakan singkatan dari *Internet Service Provider* merupakan perusahaan yang bergerak pada bidang jasa telekomunikasi dan layanan internet berbayar. Jaringan yang ditawarkan oleh ISP bermacam-macam tergantung dari harga, kecepatan dan kualitas koneksi yang akan didapat. Pada umumnya perusahaan ISP sendiri menyediakan layanan internet melalui media tertentu baik itu dari frekuensi radio, kabel LAN sampai yang terbaik yaitu kabel *fiber optic*.

Menurut Frike A.M Warius (2016) mikrotik merupakan perusahaan kecil yang berkantor pusat di Latvia, pada tahun 1996 John dan Arnis memulai dengan sistem Linux dan MS DOS yang dikombinasikan dengan teknologi *Wireless* Lan Aeronet berkecepatan 2Mbps di Moldova. Barulah kemudian melayani lima pelanggannya di Latvia, karena ambisi mereka adalah membuat satu peranti lunak *router* yang handal dan disebarkan seluruh dunia, prinsip dasar Mikrotik bukan membuat *Wireless* ISP, tapi membuat program *router* yang handal dan dapat dijalankan di seluruh dunia. Mikrotik memiliki sebuah sistem operasi bernama *Router*Os, sistem operasi ini memungkinkan pengguna untuk menjadikan komputer biasa menjadi *router* network yang handal dan mencakup berbagai fitur yang lengkap seperti pada perangkat *router* Mikrotik yang ada di pasaran saat ini. *Router*OS memiliki level yang dimulai dari level 3 hingga level 6, singkatnya pada level 3 hanya digunakan untuk *router* ber-*interface* ethernet, level 4 untuk *wireless client*, level 5 untuk *wireless* AP dan level 6 tidak mempunyai batasan apapun. Mikrotik juga membuka sebuah program kepada

lembaga pendidikan yang bernama Mikrotik *Academy* yang memanfaatkan Mikrotik sebagai sarana belajar mengajar.

Router adalah sebuah alat yang berfungsi untuk menghubungkan atau untuk memungkinkan jaringan yang berbeda agar bisa saling berkomunikasi dengan benar. Selain itu *router* juga berfungsi sebagai *gateway* dari jaringan internet serta menjadi pengatur lalu lintas data yang ada.

Gambar 2.4 : Router Mikrotik

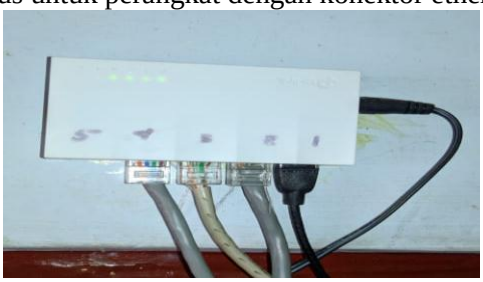
Switch merupakan sebuah *hardware* atau perangkat keras yang berfungsi untuk menyebarkan aliran data yang didapat dari *node* diatasnya. *Switch* terbagi menjadi empat macam yaitu :

Jenis *Switch* ini termasuk paling murah dan paling mudah dalam konfigurasinya, perangkat ini juga paling sering digunakan di rumah, kantor, maupun jaringan yang bersifat kecil atau *local*.

Managed Switch memiliki kelebihan jika dibandingkan dengan tipe *Switch* sebelumnya yaitu perangkat ini memiliki *user interface* sehingga pengguna bisa melakukan beberapa konfigurasi seperti memakai *console* melalui jaringan internet.

Smart Switch merupakan modifikasi karakteristik dari *Unmanaged Switch* dan *Managed Switch*. Untuk perangkat ini dalam konfigurasinya menggunakan teknologi berbasis web dan juga *Switch* ini mempunyai keunggulan pengaturan otomatis dan bisa diubah sesuai dengan kebutuhan jaringan komputer.

Jenis Switch satu ini, biasa dipakai perusahaan besar yang membutuhkan jaringan sebagai pemonitor dan juga sekaligus mengkonfigurasi jaringan yang ada. Dari jumlah *port*, *hardware* ini memiliki sekitar 4-8 *port* khusus untuk perangkat dengan konektor ethernet.



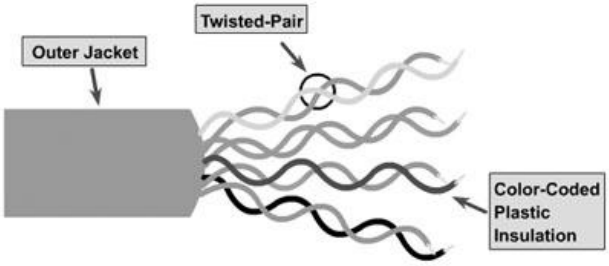
Gambar 2.5 : Switch TPlink

Access point merupakan suatu perangkat keras yang memungkinkan untuk menyebarkan koneksi internet yang didapat melalui media nirkabel atau *wireless*. *Access point* memiliki standarisasi dalam hal kecepatan koneksi data via nirkabel yang sudah diatur dalam standar IEEE (*institute of electrical and electronics engineers*) yang merupakan suatu grup organisasi para insinyur yang mengatur standarisasi dalam bidang teknologi informasi, dan diantaranya adalah standarisasi di jaringan nirkabel yang mempunyai kode 802.11. Dengan adanya standarisasi ini bertujuan agar tiap perangkat *wireless* yang berbeda tetap dapat saling berkomunikasi meskipun berbeda vendor.

Gambar 2.6 : Access Point Tenda

Kabel *Twisted Pair* atau yang sering sebut sebagai kabel lan merupakan jenis kabel yang digunakan untuk komunikasi telepon dan sebagian besar jaringan ethernet modem. Sepasang kabel membentuk sebuah jalur yang dapat mengirimkan data, pasangan kabel tersebut dibuat saling melilit yang bertujuan memberikan perlindungan terhadap "*crosstalk*", atau gangguan yang dihasilkan oleh pasangan kabel yang berdekatan. Ada 2 jenis umum pada kabel jenis *Twisted Pair*, yaitu *Unshielded Twisted Pair* (UTP) dan *Shielded Twisted Pair* (STP).

Kabel UTP merupakan media transmisi yang terdiri dari 4 pasang kawat. Kabel UTP digunakan dalam berbagai jaringan. Masing-masing dari delapan kabel tembaga individu dalam kabel UTP ditutupi oleh bahan isolasi. Selain itu, kabel di setiap pasangan yang melilit satu sama lain.

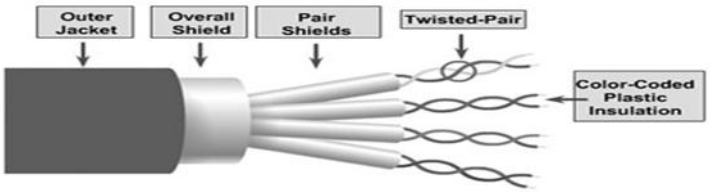


Gambar 2.7 : Komposisi Kabel UTP

Kabel UTP sering dikombinasikan dengan konektor *registered jack* 45 (RJ-45). RJ-45 merupakan konektor yang biasa digunakan untuk menghubungkan komputer ke sebuah *local area network* (LAN), khususnya ethernet.

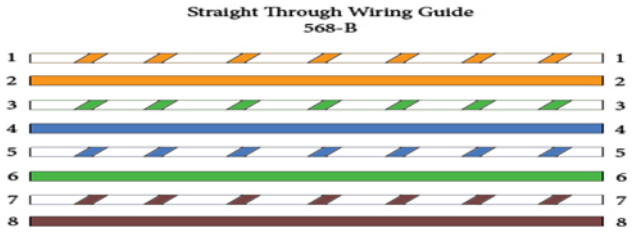
Gambar 2.8 : Konektor RJ-45

STP hampir sama dengan UTP hanya saja setiap pasang kawat dibungkus dengan kertas aluminium. Keempat pasang kawat akan dibungkus lagi dengan kertas aluminium atau serabut logam dengan tujuan untuk mengurangi gangguan seperti *electric noise*, medan magnet dan sebagainya. STP bisa dikombinasikan dengan konektor STP khusus atau bisa juga dengan RJ-45. Penggunaan kabel STP lebih ditujukan untuk pengaplikasian *outdoor* seperti pada AP *outdoor* pada tower ataupun pada instalasi jaringan yang kondisinya cuacanya tidak menentu.



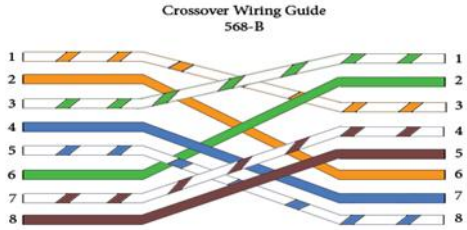
Gambar 2.9 : Komposisi Kabel STP

Jika berbicara tentang pengurutan pin kabel jaringan, tentu sebutan *straight* dan crossover sering didengar. Kabel *straight* merupakan merupakan kabel yang ujung awal dan ujung akhir kabel memiliki urutan pin yang sama.



Gambar 2.10 : Konfigurasi kabel Straight

Berbeda dengan *straight*, untuk kabel *cross* sendiri sesuai dengan namanya, penyusunan pin dilakukan berlawanan atau berseberangan.



Gambar 2.11 : Konfigurasi Kabel Cross

kabel *straight* dan *cross* memang sama-sama menghubungkan satu perangkat ke perangkat lain dalam jaringan komputer, namun perangkat yang bisa dihubungkan dengan masing-masing jenis kabel ini berbeda. Berikut adalah gambar tabel perangkat yang akan dihubungkan dan kabel yang dibutuhkan :

	Hub	Switch	Router	Workstation
Hub	Crossover	Crossover	Straight	Straight
Switch	Crossover	Crossover	Straight	Straight
Router	Straight	Straight	Crossover	Crossover
Workstation	Straight	Straight	Crossover	Crossover

Gambar 2.12 : konfigurasi kabel dan perangkat yang didukung

perangkat terbaru saat ini kebanyakan sudah mendukung *Auto* MDI/MDI-X. Perangkat yang sudah support protokol ini bisa dihubungkan baik dengan kabel straight maupun kabel *cross*. Perangkat akan mendeteksi apakah koneksi membutuhkan *cross*, dan secara otomatis akan menggunakan konfigurasi MDI atau MDI-X untuk menyamakan koneksi perangkat lawan. Jadi keuntungannya dalam hal pengkabelan, pengguna tidak perlu repot-repot menggunakan kabel *cross*, dengan hanya memakai kabel *straight* semua perangkat akan otomatis bisa terhubung dengan mudah.
Cisco *packet tracer* merupakan aplikasi simulasi yang dibuat oleh Cisco *Systems* yang memungkinkan pengguna untuk melakukan aktivitas yang berkaitan dengan jaringan komputer mulai dari desain topologi jaringan hingga ke pengaturan *routing* jaringan pada *router* cisco. Desain tampilan dari aplikasi ini juga tergolong mudah dipahami, pengguna tidak perlu repot untuk memahami *coding* untuk menjalankannya, selain itu Cisco *packet tracer* sering digunakan oleh tenaga pengajar sebagai alat bantu pembelajaran untuk eksperimen jaringan dan juga sangat berguna jika suatu instansi pendidikan masih kekurangan perangkat keras untuk praktek secara langsung.

Gambar 2.13 : Tampilan Awal Cisco Packet Tracer

Wireshark adalah sebuah aplikasi penganalisa protokol jaringan yang dibuat oleh Gerald Combs pada Mei 2006, sebelumnya Wireshark bernama Ethereal kemudian karena suatu sebab Gerald harus meninggalkan merk dagang Ethereal dan membuat projek ini terbengkalai. Satu-satunya cara agar proyek yang sebelumnya terbengkalai kembali berjalan ialah mengubah namanya menjadi Wireshark. Aplikasi ini sering digunakan untuk menganalisa kualitas jaringan yang biasanya bergantung pada beberapa indikator yaitu :

1. *Delay*

Delay atau *latency* merupakan waktu yang dibutuhkan suatu data untuk menempuh jarak dari asal menuju ke tujuan. *Delay* dapat dipengaruhi oleh beberapa faktor yaitu media fisik, kongesti atau waktu proses yang lama. Menurut Anggita Nindya Wisnu Wardhana, Muh. Yamin dan LM Fid Aksara (2017) untuk mencari nilai *delay* bisa menggunakan rumus berikut :

$$Delay = total\ delay\ /\ total\ paket\ diterima$$

Kemudian, berikut adalah tabel kategori kualitas *delay* :

Tabel 2.1 : Kategori Delay Atau Latency (sumber : TIPHON)

Sangat bagus	0ms
Bagus	0ms s/d 75ms
Sedang	75ms s/d 125ms
Buruk	>125ms

2. *Packet Loss*

Packet loss merupakan suatu parameter yang mengindikasikan paket data hilang ataupun tidak sampai pada tujuan. Hal ini sangat berpengaruh terhadap kualitas dan kinerja jaringan. Berdasarkan informasi yang diperoleh dari Anggita Nindya Wisnu Wardhana, Muh. Yamin dan LM Fid Aksara (2017) Untuk mencari nilai *packet loss*, dapat digunakan rumus berikut :

$$packet\ loss = \frac{(paket\ data\ dikirim - paket\ data\ diterima)}{paket\ data\ dikirim} \times 100\%$$

Berikut adalah tabel kategori dari parameter *packet loss* :

Tabel 2.2 : Kategori Packet Loss (sumber : TIPHON)

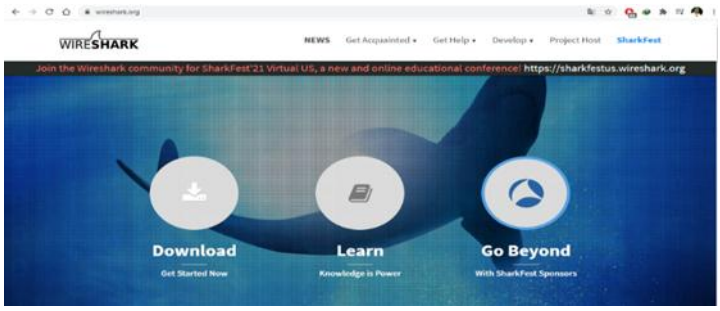
Kualitas	*Packet Loss*
Sangat bagus	0%
Bagus	3%
Sedang	15%
Jelek	25%

3. *Throughput*

Throughput merupakan kecepatan *rate* data yang diukur dalam satuan *Bps*. Sebagai acuan, *throughput* bisa juga dikatakan sebagai jumlah total paket yang sukses diterima. Menurut Anggita Nindya Wisnu Wardhana, Muh. Yamin dan LM Fid Aksara (2017) mendefinisikan, rumus untuk mencari nilai *throughput* adalah sebagai berikut :

$$throughput = \frac{Paket\ Data\ Diterima}{Lama\ Pengamatan}$$

Buka halaman *website* wireshark menggunakan *browser* dengan mengetik alamat wireshark.org dan akan langsung diarahkan menuju halaman utama web.

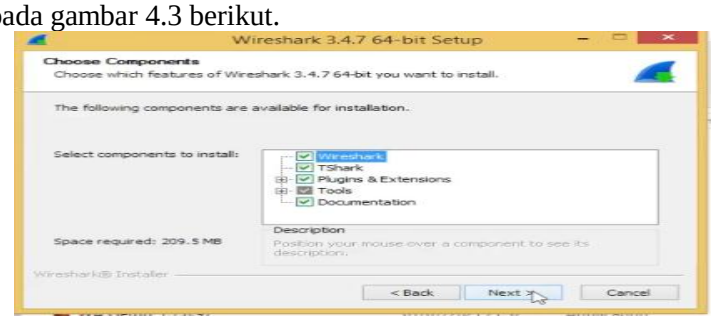


Gambar 2.14 : Homepage Website Wireshark

Setelah halaman *homepage* terbuka kemudian klik pada icon *download* seperti pada gambar 4.1 dan kemudian tunggu hingga proses pengunduhan telah selesai. Jika proses pengunduhan aplikasi wireshark telah selesai selanjutnya klik dua kali pada program untuk menjalankan *setup* instalasi program wireshark.

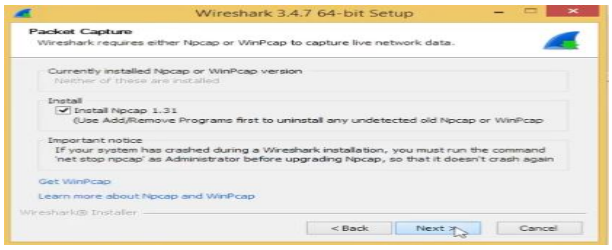
Gambar 2.15 : Proses Instalasi Wireshark (1)

setelah muncul tampilan seperti gambar 4.2 langsung saja klik tombol *next* dan akan muncul *interface* selanjutnya seperti pada gambar 4.3 berikut.



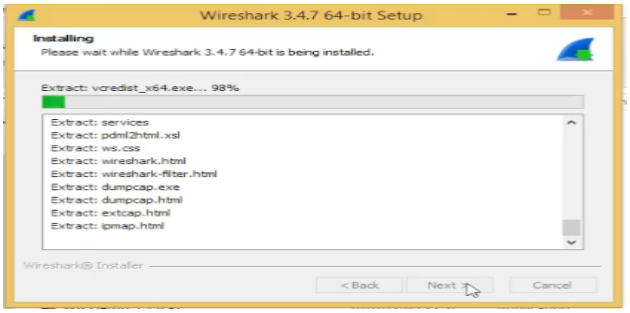
Gambar 2.16 : Proses Instalasi Wireshark (2)

biarkan saja *checkbox* pada posisi *default* sesuai dengan keadaan semula agar tidak terjadi hal yang tidak diinginkan. Bagian selanjutnya akan menunjukkan tempat dimana wireshark akan terinstal, biarkan saja pada posisi awal.



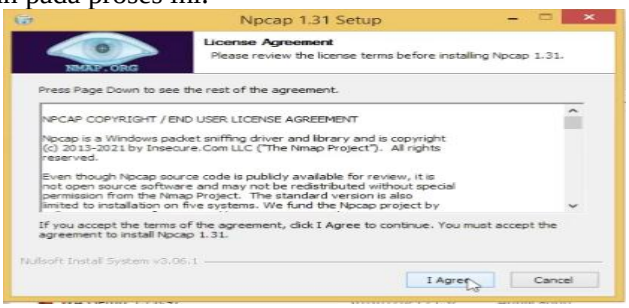
Gambar 2.17 : Proses Instalasi Wireshark (3)

Jika muncul *interface* seperti pada gambar 4.3 pastikan *checkbox* sudah tercentang agar komponen Npcap bisa terinstal yang nantinya akan digunakan sebagai penangkap paket data pada aplikasi wireshark.



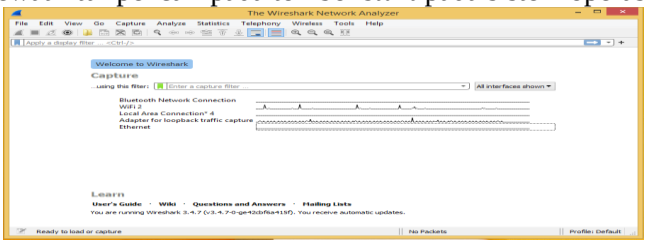
Gambar 2.18 : Proses Instalasi Wireshark (4)

tunggu hingga proses penginstalan selesai dan akan muncul *interface* untuk menginstal Npcap seperti pada gambar 4.4 pada proses instalasi Npcap langsung saja ikuti tombol *next* yang ada dan jangan sampai mengubah apapun pada proses ini.



Gambar 2.19 : Proses Instalasi Wireshark (5)

setelah semua proses telah dilakukan dengan benar dan instalasi telah mencapai 100% maka *interface* akan menutup dengan otomatis. Aplikasi wireshark yang telah di instal dapat ditemukan pada *shortcut desktop* atau bisa juga lewat fitur pencari pada tombol *start* pada sistem operasi windows.

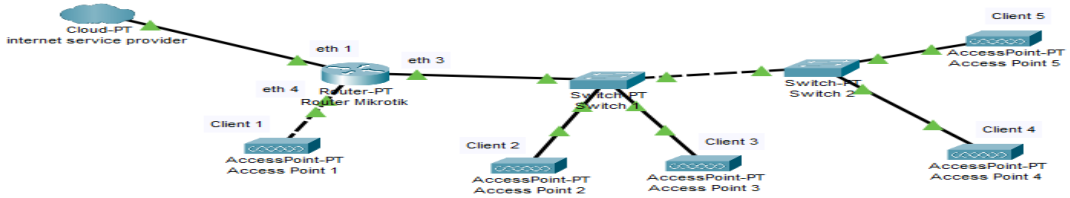


Gambar 2.20 : Interface Awal Wireshark

Winbox merupakan sebuah aplikasi yang dikembangkan langsung oleh Mikrotik yang berfungsi sebagai penghubung serta pengatur manajemen jaringan antara perangkat *administrator* dan perangkat *router* itu sendiri, aplikasi Winbox berukuran kecil dan portable sehingga pengguna tidak perlu menginstal aplikasi tambahan apapun untuk menjalankannya. Sebenarnya tampilan Winbox sendiri dirancang untuk pengguna awam yang tidak terlalu memahami *command line interface* seperti halnya pada linux *server*, aplikasi Winbox dapat bekerja pada hampir semua OS yang didukung seperti Windows, Linux dan Mac OS.

Gambar 2.21 : Tampilan Awal Aplikasi Winbox

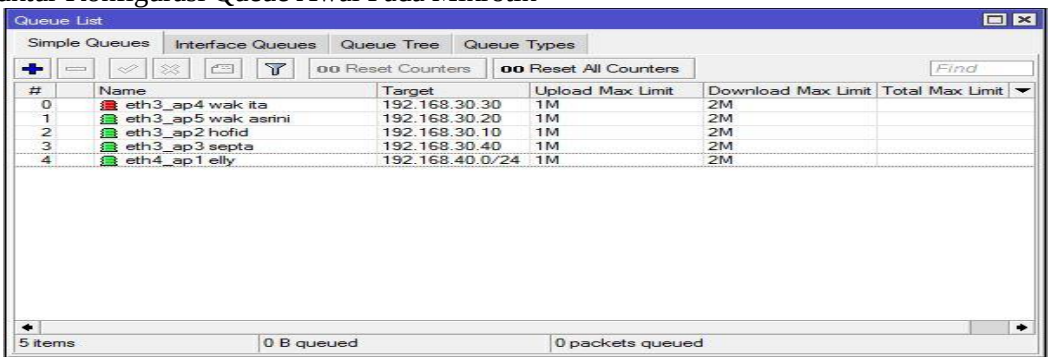
Struktur Jaringan RT/RW Net Dusun Warengan Desa Bubuk



Gambar 2.22 : Topologi Jaringan RT/RW Net

Melalui gambar 2.23 bisa dilihat bahwa topologi yang digunakan pada jaringan RT/RW net sederhana menggunakan desain topologi star dimana *switch* menjadi titik utama persebaran koneksi internet melalui kabel lan. Jarak *client* terdekat dari router adalah sekitar 10 meter yaitu *client* 2 sedangkan jarak terjauh adalah 45 meter yaitu *client* 5 namun itu adalah perkiraan jarak dari router menuju lokasi *client* sedangkan disini, *client* 5 sudah terhubung melalui *switch* 2 bersama dengan *client* 4 jadi jarak dari *switch* 2 menuju *client* 5 hanya sekitar 20 meter saja. Untuk *client* 1 terdapat kendala yaitu jika kabel lan dipindah ke port *switch* 1 maka *access point* yang bersangkutan tidak dapat mengenali koneksi sama sekali dan begitu pula dengan *switch* 1 tidak dapat mengenali jika kabel lan dari *client* 1 dihubungkan ke salah satu port yang ada.

1. Struktur Konfigurasi Queue Awal Pada Mikrotik



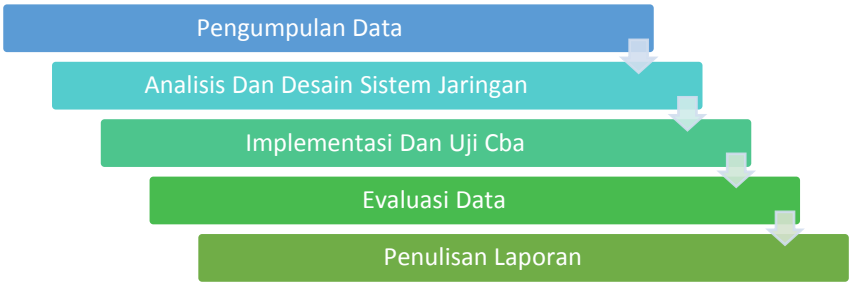
Gambar 2.23 : Konfigurasi *Queue* Awal Pada *Router*

Pada gambar diatas yaitu gambar 2.24 menunjukkan bahwa konfigurasi *queue* pada *router* mikrotik hanya menggunakan *simple queue* tanpa adanya konfigurasi lanjutan. Untuk detail mengenai konfigurasi *router* adalah sebagai berikut :

a. IP *dynamic* yang didapatkan dari ISP adalah 192.168.1.68/24, IP ini dapat berubah jika salah satu perangkat baik dari ISP maupun *router* melakukan *reboot* ulang perangkat.
b. Untuk *client* 1 terhubung ke port eth 4 pada *router* Mikrotik dengan IP *Dynamic* 192.168.40.1/24.
c. Pada eth 3 router Mikrotik menggunakan IP utama dan sebagai *gateway* para *client* lainnya yaitu 192.168.30.1/24.
d. Mulai dari *client* 2 hingga *client* 5 menggunakan IP *static* yang berurutan mulai dari IP 192.168.30.10/24 hingga ke IP 192.168.30.40/24.
e. Tiap *client* mendapat *bandwidth* sebesar 2Mbps untuk *download* sedangkan untuk *upload* mendapat *bandwidth* sebesar 1Mbps.

2. METODE PERANCANGAN

Langkah-langkah yang dilakukan untuk menyelesaikan penelitian ini bisa dilihat pada gambar 3.1. Proses penelitian ini menggunakan metode *Waterfall* yang mana proses pengerjaannya dilakukan berurutan mulai dari atas hingga secara berurutan.



Berikut adalah penjelasan langkah-langkah penelitian pada gambar 3.1 :

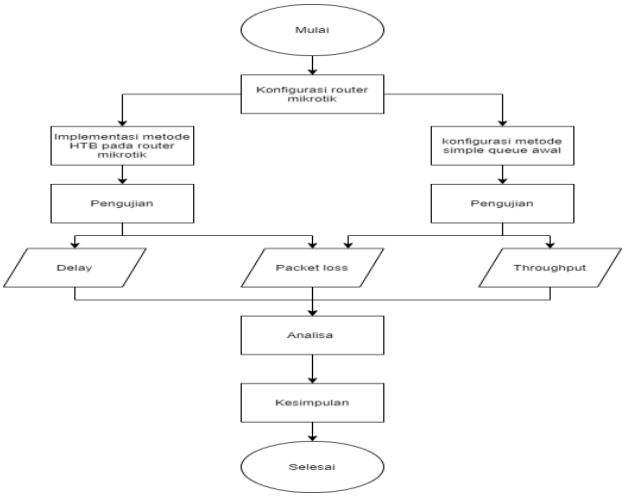
1. Teknik yang dilakukan dalam pengumpulan data adalah dengan melakukan observasi atau pengamatan secara langsung dan juga selain itu dilakukan studi pustaka baik melalui buku, internet, maupun literatur lainnya yang berkaitan tentang pembangunan jaringan internet.
2. Pada tahap analisis dan desain sistem jaringan penulis melakukan analisa terhadap kebutuhan sistem yang akan digunakan serta merancang desain jaringan menggunakan aplikasi Cisco *Packet Tracer* sebagai alat bantu.
3. Tahap implementasi dan uji coba, dimana semua bahan yang dibutuhkan sudah terkumpul serta dilakukan realisasi sehingga menjadi bentuk yang utuh dan sempurna, kemudian dilakukan uji coba dengan parameter yang sudah ditentukan.
4. Evaluasi data, setelah uji coba sudah dilakukan dengan baik dan benar maka dilanjut dengan evaluasi data yaitu dengan cara membandingkan data lama dan data baru apakah data terbaru mempunyai nilai lebih baik atau tidak.
5. Penulisan laporan bertujuan untuk mendokumentasikan hasil kegiatan serta temuan selama melakukan penelitian

Berikut adalah beberapa *hardware* dan *software* yang dibutuhkan untuk menunjang proses pelaksanaan penelitian :

1. Laptop Lenovo G40-45
- AMD A8 6410 2,4Ghz
- *Ram* 2x4GB
- *SSD* 120GB
- *Windows 8.1 Pro*

2. *Router* Mikrotik RB 941-2Nd-TC
- Cpu @650Mhz
- Rom 16MB
- Ram 32MB
- Lan Port 4
- Lisensi level 4

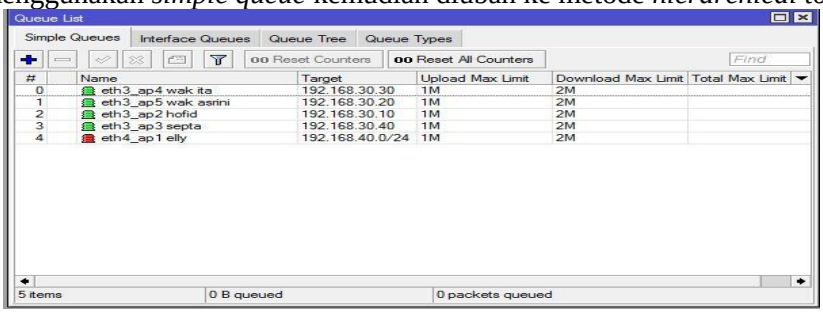
3. *Access point*
4. *Switch*
5. Winbox
6. Wireshark
7. Cisco *packet tracer*



Pada gambar 3.2 bisa dilihat terdapat dua konfigurasi dimana metode konfigurasi awal pada jaringan RT/RW net di Dusun Warengan Desa Bubuk berupa metode *simple queue* setelah itu akan dilakukan pengujian terhadap metode yang bersangkutan dengan 3 parameter yaitu *delay*, *packet loss*, dan *throughput*. Setelah melakukan pengujian terhadap metode awal, barulah kemudian dilanjut dengan implementasi metode *hierarchical token bucket* pada sistem mikrotik dan juga dilakukan pengujian, setelah pengujian selesai dan data sudah terkumpul maka akan dilakukan perbandingan apakah metode baru lebih efektif dalam kinerjanya ataukah hasilnya akan lebih buruk dari metode awal.

Sub bab ini berisi tentang perencanaan implementasi metode baru pada sistem *router* mikrotik yaitu metode *hierarchical token bucket* serta menunjukkan simulasi uji coba yang akan dilakukan pada bab selanjutnya.

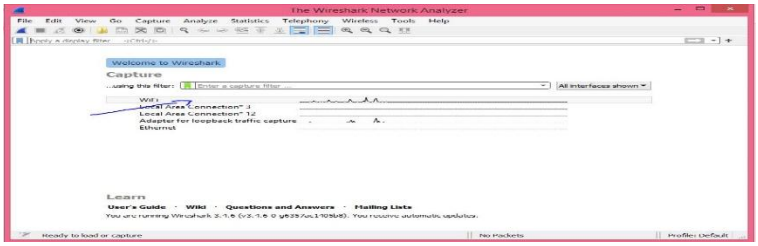
pada perencanaan ini akan dilakukan perubahan terhadap konfigurasi manajemen *bandwidth* yang awalnya menggunakan *simple queue* kemudian diubah ke metode *hierarchical token bucket*.



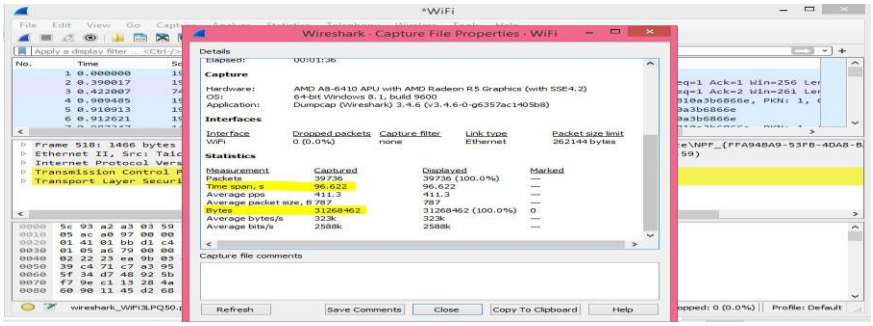
Bisa dilihat pada gambar 3.3 penggunaan *simple queue* memanglah sangat mudah dan cenderung lebih cepat dalam konfigurasinya, namun masalahnya adalah cenderung lebih boros dalam manajemen *bandwidth* sehingga penggunaannya tidak optimal sebagaimana mestinya. Sedangkan jika menggunakan metode *hierarchical token bucket* akan dibuat sebuah struktur hirarki dimana *parent* sebagai penampung *bandwidth* total dan *child* sebagai penerima alias *client*, metode ini dasarnya adalah membagi *bandwidth* kepada tiap *client* secara merata dan terarah dan jika ada salah satu *client* sedang tidak aktif atau tidak ada arus *traffic* maka *bandwidth* dari *client* tersebut akan dibagikan kepada *client* lain dengan adil sesuai dengan prioritasnya.

Simulasi Uji Coba

Sebelum maupun sesudah implementasi metode *hierarchical token bucket* akan dilakukan uji coba dengan beberapa parameter pengujian yaitu : *delay*, *throughput*, dan *packet loss*. Berikut adalah contoh untuk mencari nilai *throughput* menggunakan aplikasi wireshark :



Setelah membuka aplikasi wireshark kemudian klik dua kali pada *interface* yang akan digunakan untuk menganalisa jaringan, dalam kasus ini menggunakan wifi. Setelah muncul data yang berjalan, di waktu itu juga lakukan kegiatan yang sekiranya memakai banyak *bandwidth* seperti *streaming.*



Setelah data terekam lalu tekan ctrl+shift+alt+c untuk memunculkan menu *capture file properties*, dalam menu ini terdapat info mengenai data yang telah direkam sebelumnya. Dalam melakukan pencarian nilai *throughput* yang perlu diperhatikan adalah nilai *time span* atau lama pengamatan dan juga Bytes atau jumlah paket data yang diterima, kemudian masukkan nilai tersebut kedalam persamaan berikut :

$$throughput = \frac{Paket\ Data\ Diterima}{Lama\ Pengamatan}$$

Diketahui bahwa *time span* bernilai 96,622s dan nilai Bytes sebesar 31268462 Bytes.

$$throughput = \frac{31268462}{96{,}622}$$

$$throughput = 323616\ Bytes * 1000 = 323.616\ KB$$

$$Throughput = 323.616 * 8$$

$$Throughput = 2588\ Kb/s$$

Jadi *throughput* yang didapat dalam pengamatan selama 96 detik adalah 2588 Kb/s

Evaluasi Data

Setelah uji coba telah dilakukan, langkah selanjutnya adalah evaluasi data, dalam langkah ini dilakukan perbandingan data antara metode *simple queue* dan *hierarchical token bucket*. Uji coba dilakukan beberapa kali dengan minimum uji coba sebanyak 10 kali hingga menemukan hasil akhir yang valid, baru setelah itu dilakukan perbandingan data dengan format tabel seperti berikut :

Parameter	Simple Queue	Hierarchical Token Bucket
Delay	hasil	-
Throughput	-	hasil
Packet loss	hasil	-

3. HASIL DAN PEMBAHASAN

Pada bab ini merupakan tahap dimana sistem yang telah di implementasikan akan di uji coba pada kondisi sebenarnya, dan akan diketahui apakah sesuai dengan yang diharapkan penulis sebelum sistem ini diterapkan. Maka pada bab ini penulis akan melakukan uji coba beserta pemaparan hasil dari implementasi metode *hierarchical token bucket* yang sebelumnya telah diterapkan pada *router* mikrotik.

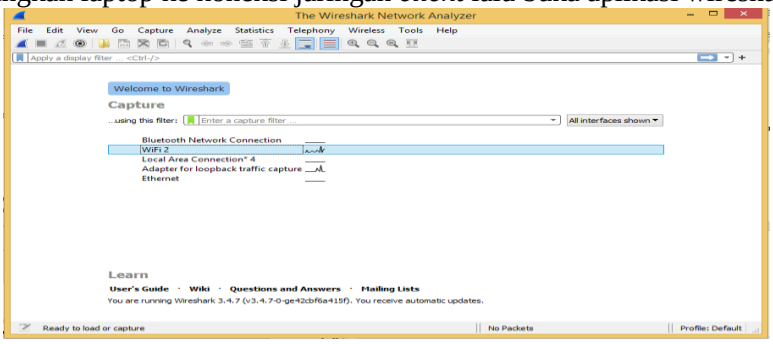
Skenario Uji Coba

Untuk skenario uji coba yang akan dilaksanakan adalah dengan cara melakukan analisa sebanyak 10 kali baik dari sebelum implementasi *htb* maupun sesudah implementasi *htb* dengan waktu yang berbeda-beda. Pengujian dilakukan dengan menggunakan aplikasi wireshark dengan minimal pengamatan selama 3 menit.

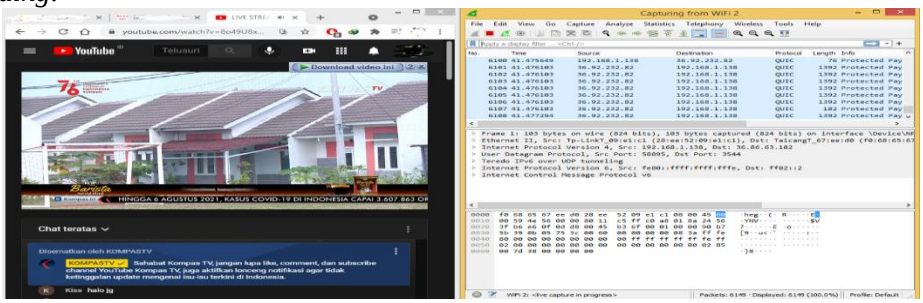
Persiapan Uji Coba

Berikut ini merupakan uraian singkat mengenai langkah-langkah persiapan yang akan dilakukan dalam analisa jaringan pada RT/RW net :

a. Pertama sambungkan laptop ke koneksi jaringan *client* lalu buka aplikasi wireshark.



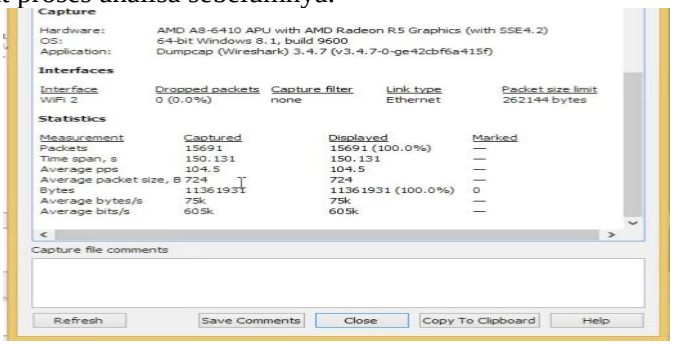
b. klik dua kali pada *interface* atau *adapter* yang akan digunakan untuk menganalisa jaringan. Dalam hal ini menggunakan wifi sebagai media koneksinya. Pada saat proses analisa, sebaiknya dilakukan serangkaian aktifitas yang membutuhkan *bandwidth* internet seperti streaming ataupun *downloading*.



c. setelah tahap proses perekaman data dirasa sudah cukup, klik tombol stop berwarna merah untuk menghentikan proses perekaman trafik.

Uji Coba Parameter *Throughput*

Pada aplikasi wireshark, setelah melakukan proses perekaman data kemudian klik bagian menu *statistic* lalu klik *capture file properties.* Dalam menu ini terdapat data yang akan menampilkan trafik yang terekam pada saat proses analisa sebelumnya.



Gambar 5.3 : Capture File Properties

Untuk perhitungan *throughput* yang perlu diperhatikan adalah bagian parameter *time span* atau lama pengamatan dan *bytes* atau paket data yang diterima, kemudian dimasukkan kedalam rumus berikut atau persamaan berikut :

$$throughput = \frac{Paket\ Data\ Diterima}{Lama\ Pengamatan}$$

Dan diketahui untuk paket data diterima adalah sebanyak 11361931 dan lama pengamatan sebesar 150,131 detik. Kemudian masukkan nilai tadi kedalam rumus sebelumnya.

$$throughput = \frac{11361931}{150{,}131}$$

$$throughput = 75860\ bytes/s$$

Dikarenakan hasilnya masih berupa satuan *bytes per second* maka selanjutnya diubah ke satuan *kilobytes per second* dengan cara mengalikan nilai data dengan nilai 1000.

$$throughput = 75860\ bytes/s * 1000$$

$$throughput = 75{,}860\ kilobytes/s$$

Setelah ditemukan hasil berupa satuan *kilobytes per second* lalu dikonversi ke satuan *kilobit per second* untuk menemukan hasil dari *throughput*.

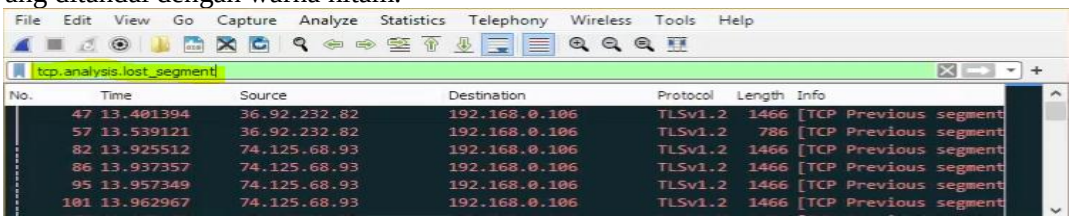
$$throughput = 75{,}860\ kilobytes/s * 8$$

$$throughput = 605\ kilobit/s$$

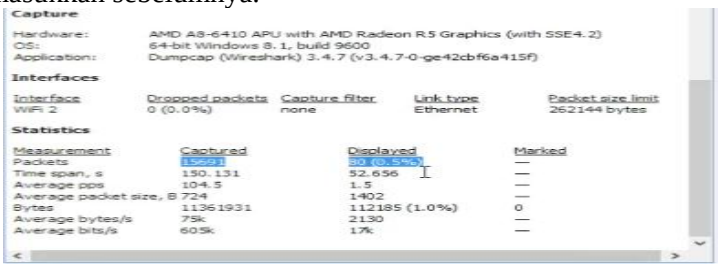
Maka ditemukanlah hasil akhir dari *throughput* yang bernilai 605 *kilobit per second*.

Uji Coba Paremeter Packet Loss

Setelah nilai *throughput* telah ditemukan, selanjutnya adalah mencari nilai *packet loss*. Pada *interface* daftar data yang telah direkam sebelumnya, klik pada kolom filter dan masukkan perintah "tcp.analysis.lost_segment" tanpa tanda petik, kemudian wireshark akan menampilkan sekumpulan data yang ditandai dengan warna hitam.



Sama seperti langkah sebelumnya, setelah memasukkan perintah tadi kedalam kolom filter lalu langsung masuk ke menu *capture file properties* dan data yang ditampilkan akan mengikuti perintah filter yang sudah dimasukkan sebelumnya.



Untuk parameter *packet loss* yang diperhatikan adalah nilai *packets* pada kolom *captured* dan kolom *displayed* yang merupakan nilai dari paket data dikirim dan paket data yang hilang.

$$packet\ loss = \frac{(paket\ data\ dikirim - paket\ data\ diterima)}{paket\ data\ dikirim} \times 100\%$$

Diketahui bahwa paket data yang terkirim adalah 15691 dan paket data diterima adalah sebesar 15611 hasil dari paket data dikirim dikurangi paket data yang hilang sebesar 80.

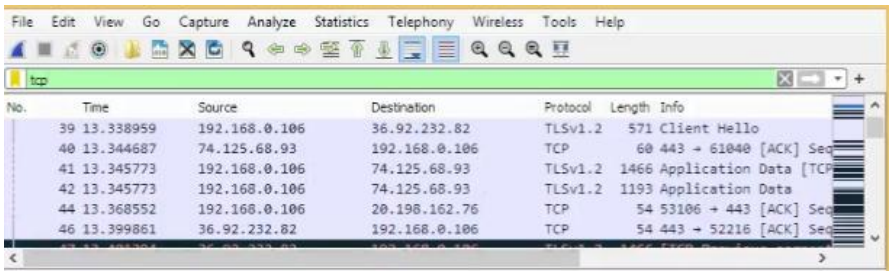
$$packet\ loss = \frac{(15691 - 15611)}{15691} \times 100\%$$

$$packet\ loss = \frac{80}{15691} \times 100\%$$

$$packet\ loss = 0{,}005 \times 100\%$$
$$packet\ loss = 0{,}5\%$$

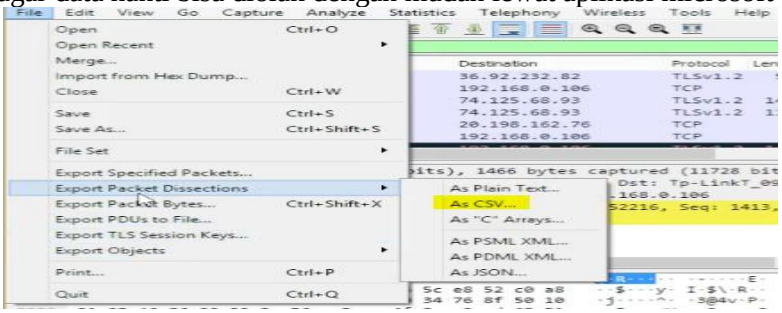
Ditemukan bahwa nilai dari *packet loss* adalah sebesar 0,5%.

Klik pada filter dan masukkan perintah “tcp” perintah ini akan memunculkan daftar data berupa kumpulan data dan ip yang digunakan oleh suatu perangkat untuk saling berkomunikasi dalam jaringan internet.

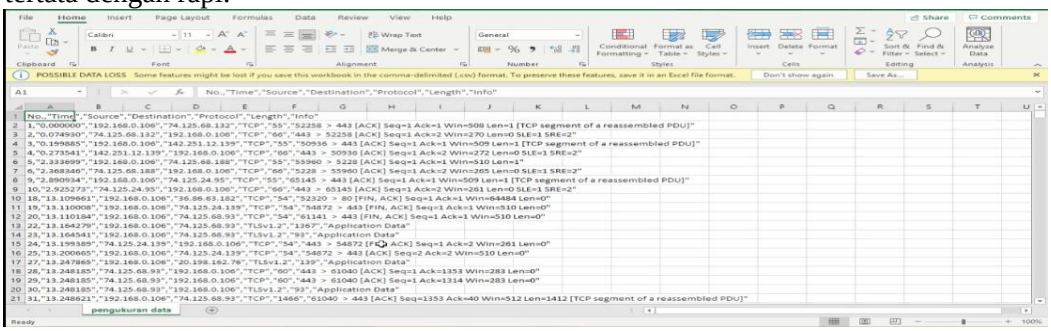


Gambar 5.6 : Filter Data Tcp

Setelah data terfilter selanjutnya klik menu *file* lalu pilih *export packet dissections* dan pilih *as CSV* format ini dipilih agar data nanti bisa diolah dengan mudah lewat aplikasi microsoft excel.



Selanjutnya, setelah *file* tersimpan dan terbuka pada aplikasi excel akan berupa data mentah dan belum tertata dengan rapi.



Berikut adalah langkah-langkah untuk merapikan dataset agar bisa terbaca dengan mudah :

1. Klik pada penunjuk kolom paling awal yaitu kolom A agar semua data pada kolom A terpilih.
2. Masuk ke *tab data* dan pilih *text to column*, akan muncul sebuah *interface* baru untuk mengubah data. Pilih *delimited*, kemudian pada menu *delimiter* pilih *comma* kemudian klik *next* dan pilih *finish*.

	A	B	C	D	E	F	G	H	I	J
1	No.	Time	Source	Destinatic	Protocol	Length	Info			
2	1	0.000000	192.168.0.	74.125.68.	TCP	55	52258 > 443 [ACK] Seq=1 Ack=1 Win=508 L			
3	2	0.074930	74.125.68.	192.168.0.	TCP	66	443 > 52258 [ACK] Seq=1 Ack=2 Win=270 L			
4	3	0.199885	192.168.0.	142.251.1:	TCP	55	50936 > 443 [ACK] Seq=1 Ack=1 Win=509 L			
5	4	0.273541	142.251.1:	192.168.0.	TCP	66	443 > 50936 [ACK] Seq=1 Ack=2 Win=272 L			
6	5	2.333.699	192.168.0.	74.125.68.	TCP	55	55960 > 5228 [ACK] Seq=1 Ack=1 Win=510			
7	6	2.368.346	74.125.68.	192.168.0.	TCP	66	5228 > 55960 [ACK] Seq=1 Ack=2 Win=265			
8	9	2.890.934	192.168.0.	74.125.24.	TCP	55	65145 > 443 [ACK] Seq=1 Ack=1 Win=509 L			
9	10	2.925.273	74.125.24.	192.168.0.	TCP	66	443 > 65145 [ACK] Seq=1 Ack=2 Win=261 L			
10	18	13.109.661	192.168.0.	36.86.63.1	TCP	54	52320 > 80 [FIN, ACK] Seq=1 Ack=1 Win=6			
11	19	13.110.008	192.168.0.	74.125.24.	TCP	54	54872 > 443 [FIN, ACK] Seq=1 Ack=1 Win=			
12	20	13.110.184	192.168.0.	74.125.68.	TCP	54	61141 > 443 [FIN, ACK] Seq=1 Ack=1 Win=			
13	22	13.164.279	192.168.0.	74.125.68.	TLSv1.2	1367	Application Data			
14	23	13.164.541	192.168.0.	74.125.68.	TLSv1.2	93	Application Data			
15	24	13.199.389	74.125.24.	192.168.0.	TCP	54	443 > 54872 [FIN, ACK] Seq=1 Ack=2 Win=			
16	25	13.200.665	192.168.0.	74.125.24.	TCP	54	54872 > 443 [ACK] Seq=2 Ack=2 Win=510 L			
17	27	13.247.865	192.168.0.	20.198.16:	TLSv1.2	139	Application Data			
18	28	13.248.185	74.125.68.	192.168.0.	TCP	60	443 > 61040 [ACK] Seq=1 Ack=1353 Win=2			
19	29	13.248.185	74.125.68.	192.168.0.	TCP	60	443 > 61040 [ACK] Seq=1 Ack=1314 Win=2			
20	30	13.248.185	74.125.68.	192.168.0.	TLSv1.2	93	Application Data			
21	31	13.248.621	192.168.0.	74.125.68.	TCP	1466	61040 > 443 [ACK] Seq=1353 Ack=40 Win=			

Gambar 5.9 : Hasil Convert Data To Column

Terlihat pada gambar 5.8 dataset sudah rapi dan bisa digunakan, namun pada prakteknya data yang bisa digunakan untuk mengetahui nilai *delay* adalah kolom *time*. Kemudian untuk langkah selanjutnya adalah mengisi data *time* 1 dan *time* 2 yang nanti akan digunakan untuk menghitung total dari *delay*.

	A	B	C	D	E	F	H	I	J
1	No.	Time	Source	Destinatic	Protocol	Length		time 1	time 2
2	1	0.000000	192.168.0.	74.125.68.	TCP	55			
3	2	0.074930	74.125.68.	192.168.0.	TCP	66			
4	3	0.199885	192.168.0.	142.251.1:	TCP	55			
5	4	0.273541	142.251.1:	192.168.0.	TCP	66			
6	5	2.333.699	192.168.0.	74.125.68.	TCP	55			
7	6	2.368.346	74.125.68.	192.168.0.	TCP	66			
8	9	2.890.934	192.168.0.	74.125.24.	TCP	55			
9	10	2.925.273	74.125.24.	192.168.0.	TCP	66			
10	18	13.109.661	192.168.0.	36.86.63.1	TCP	54			
11	19	13.110.008	192.168.0.	74.125.24.	TCP	54			
12	20	13.110.184	192.168.0.	74.125.68.	TCP	54			
13	22	13.164.279	192.168.0.	74.125.68.	TLSv1.2	1367			
14	23	13.164.541	192.168.0.	74.125.68.	TLSv1.2	93			
15	24	13.199.389	74.125.24.	192.168.0.	TCP	54			
16	25	13.200.665	192.168.0.	74.125.24.	TCP	54			

Gambar 5.10 : Proses Penghitungan Delay (1)

Bisa dilihat pada baris ke 6 data masih berupa satuan detik, namun data selanjutnya berubah menjadi mikrodetik. Sebelum bisa diolah ke tahap selanjutnya, data ini harus diubah ke satuan detik terlebih dahulu dengan cara membaginya dengan nilai 1000000.

6	5	2.333.699	=B6/1000000
7	6	2.368.346	

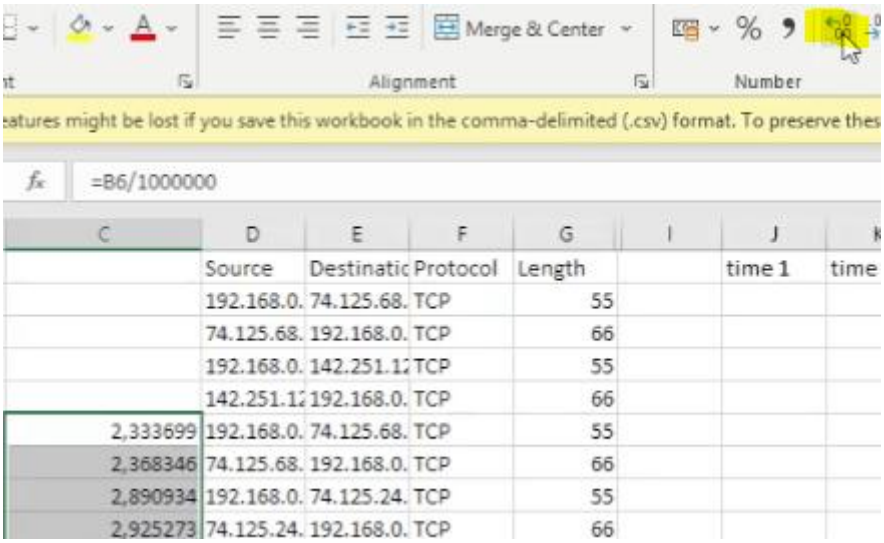
Gambar 5.11 : Proses Penghitungan Delay (2)

Untuk menghitung semua dataset secara otomatis, hanya perlu menggeser titik pojok kanan bawah pada hasil nilai sebelumnya dan tarik sampai data terakhir.

5	2.333.699	2
6	2.368.346	

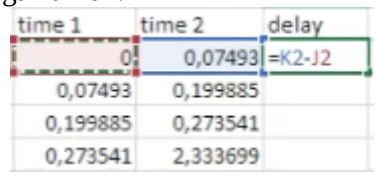
Gambar 5.12 : Proses Penghitungan Delay (3)

Kemudian salin hasil dari perhitungan tadi kedalam data *time*. selanjutnya rubah bentuk nilai tadi yang masih baku kedalam bentuk desimal dengan menekan tombol *increase decimal*.



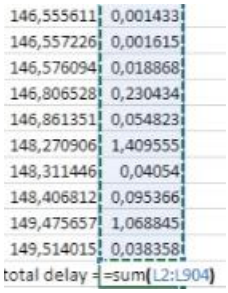
Gambar 5.13 : Proses Penghitungan Delay (4)

Salin semua data pada kolom *time* ke kolom *time1* dan *time2* dengan catatan untuk kolom *time2* pengisian data dimulai dari baris ke dua. Kemudian untuk menghitung *delay* secara satuan adalah dengan cara mengurangi *time2* dengan *time1*.



Gambar 5.14 : Proses Penghitungan Delay (5)

Untuk mengisi nilai dibawahnya, sama seperti cara sebelumnya yaitu dengan menarik titik kecil pada pojok kanan bawah pada *cell* hingga terisi semua. Setelah itu lanjut untuk mencari nilai delay total dengan cara menambah tiap hasil dari *time1-time2*.



Gambar 5.15 : Proses Penghitungan Delay (6)

Selanjutnya setelah penghitungan total delay selesai maka dilanjut dengan penghitungan rata-rata *delay* dengan rumus sebagai berikut :

$$Delay = total\ delay\ /\ total\ paket\ diterima$$

Diketahui bahwa total *delay* yang didapat adalah sebesar 149,514s dan total paket diterima sebanyak 15611 paket.

$$Delay = 149{,}512\ /\ 15611$$
$$Delay = 0{,}009577s$$

Hasil yang didapat masih berupa satuan *second* atau detik. Untuk parameter *delay* umumnya mengunakan satuan *millisecond* atau milidetik, dengan cara mengalikan nilai satuan detik dengan 1000 maka :

$$Delay = 0{,}009577s * 1000$$
$$Delay = 9{,}577ms$$

4. KESIMPULAN

Dilihat dari laporan dan juga hasil dari uji coba metode *hierarchical token bucket* terhadap jaringan RT/RW net di Dusun Warengan Desa Bubuk yang telah dikerjakan dan dilakukan, maka terdapat beberapa kesimpulan yang diantaranya adalah sebagai berikut :

1. Pembagian masing-masing *bandwidth* menggunakan metode *hiearchical token bucket* menggunakan dua *node* yaitu *parent queue* dan *child queue* dimana peran dari dua parameter itu berbeda.
2. Penggunaan metode *hierarchical token bucket* cukup efektif dalam meningkatkan performa kinerja jaringan dengan presentase kenaikan sebesar untuk parameter *troughput* sebesar 32% , dan penurunan pada pada nilai *delay* sebanyak 17,326 ms menjadi 5,266 ms dimana itu nilai yang cukup bagus karena semakin kecil nilai *delay* semakin bagus kualitas jaringannya.
3. Hanya saja hasil dari nilai parameter *packet loss* mengalami peningkatan minor sebanyak 0,77%, namun tidak sampai merusak kualitas jaringan yang ada. Asumsi yang didapat kemungkinan data yang melewati berbagai *rule* yang dibuat bisa saja hilang atau *loss* pada saat metode *hierarchical token bucket* aktif.

DAFTAR PUSTAKA

Anggita Nindya Wisnu Wardhana, Muh. Yamin dan LM Fid Aksara. (2017). Analisis Quality of Service (QoS) Jaringan Internet Berbasis Wireles LAN Pada Layanan Indihome . Jurusan Teknik Informatika, Fakultas Teknik Universitas Halu Oleo, Kendari.

Apa itu Switch: Fungsi, Jenis, Tujuan dan Cara Kerja Switch. (2020, November 19). Diakses pada April 22, 2021, dari idcloudhost: https://idcloudhost.com/apa-itu-switch-fungsi-jenis-tujuan-dan-cara-kerja-switch/

Azinar, A. W., & Adi, R. S. (2017). Analisis QoS (Quality of Service) pada Warnet dengan Metode *HTB* (Hierarchical Token Bucket). Network Engineering Research Operation.

Diyantoro, A., & Haekal, N. H. (2018). Penerapan Manajemen Bandwidth Menggunakan Hierarchical Token Bucket Pada Mikrotik Router OS. Jurnal Teknologi Informasi.

Ferdiansyah, P., Indrayani, R., & Subektiningsih, S. (2020). Analisis Manajemen *Bandwidth* Menggunakan Hierarchical Token Bucket Pada Router dengan Standar Deviasi. Jurnal Nasional Teknologi dan Sistem Informasi.

Ichwan, M. I., Sugiyanta, L., & Yunanto, P. W. (2019). Analisis Manajemen *Bandwidth* Hierarchical Token Bucket (*HTB*) dengan Mikrotik pada Jaringan SMK Negeri 22. PINTER: Jurnal Pendidikan Teknik Informatika dan Komputer.

Lukman, L., Saputro, A. M., Wicaksono, A. S., Hartomo, F. H. T., & Jatun, M. N. (2019). Manajemen *Bandwidth* Menggunakan Metode Hierarchical Token Bucket (*HTB*) di Farid. net. Creative Information Technology Journal.

Sidqi, T. O., & Nathasia, N. D. (2021). IMPLEMENTASI MANAJEMEN BANDWITH MENGGUNAKAN METODE *HTB* (HIERARCHICAL TOKEN BUCKET) PADA JARINGAN MIKROTIK. JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika).

Syafrizal, M. (2020). Pengantar jaringan komputer. Penerbit Andi.

Valens Riyadi. (tanpa tanggal). Mendalami *HTB* pada Qos RouterOS Mikrotik [Halaman Web]. Diakses pada Juni 2, 2021, dari Citraweb: https://citraweb.com/artikel_lihat.php?id=2

Waruis, F. A. (2016). *IMPLEMENTASI MANAJEMEN BANDWIDTH MENGGUNAKAN METODE SIMPLE QUEUE PADA MIKROTIK DI IT CENTER MANADO* (Doctoral dissertation, Politeknik Negeri Manado).