UNJUK KERJA PENERAPAN METODE TEREDO DALAM APLIKASI VIDEO STREAMING YANG BERADA PADA JARINGAN IPV6

OLEH :

ANDIKA PUTRA PERKASA

SKRIPSI

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS INDONESIA
DEPOK
2007

PENGESAHAN

Skripsi dengan judul :

Unjuk Kerja Penerapan Metode TEREDO dalam Aplikasi Video Streaming yang Berada Pada Jaringan Internet Protocol version 6 (IPv6)

Dibuat untuk melengkapi kurikulum pendidikan pada Departemen Elektro Fakultas Teknik Universitas Indonesia dan disetujui untuk diajukan sebagai Skripsi Program Studi Komputer.

Depok, Desember 2007

Menyetujui

Ir.A.Endang Sriningsih, MT

NIP 130 781 318

PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya menyatakan bahwa sejauh yang saya ketahui, skripsi ini bukan merupakan tiruan atau salinan atau duplikasi dari skripsi yang sudah dipublikasikan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun Perguruan Tinggi atau Instalasi manapun kecuali pada bagian-bagian dimana sumber informasinya dicantumkan sebagaimana mestinya.

Depok, Desember 2007

Andika Putra Perkasa

UCAPAN TERIMA KASIH

Puji syukur kepada tuhan atas segala berkat dan rahmatNya sehingga penulis dapat menyelesaikan skripsi ini. Dalam kesempatan ini, penulis ingin mengucapkan terima kasih kepada ibu :

Ir. A. Endang Sriningsih, MT

Selaku pembimbing, yang telah bersedia meluangkan waktunya untuk memberikan bimbingan, pengarahan, dan saran-saran serta fasilitas dan kemudahan-kemudahan lainnya sehingga skripsi ini dapat diselesaikan dengan baik.

Depok, Desember 2007

Andika Putra Perkasa

Dosen Pembimbing
Ir. A. Endang Sriningsih, MT

ABSTRAK

Dalam skripsi ini dibangun sebuah *test bed* untuk menguji koneksi dari IPv4 yang berada di balik NAT agar dapat berhubungan dengan host IPv6. Hal ini sebagai simulasi apabila pada nantinya ketika jaringan IPv6 telah tersebar luas maka jaringan-jaringan IPv4 yang menggunakan NAT tetap dapat terkoneksi. Pada pengujian akan dilakukan analisa terhadap parameter-parameter yang ada untuk mengetahui kinerja yang terjadi. Parameter yang diuji antara lain *Throughput, Latency, dan Packet Loss*. Aplikasi yang digunakan dalam skripsi ini adalah *video streaming*. Pengujian akan dilakukan dalam tiga tahap, yang pertama adalah pengujian *video streaming* yang diterapkan pada jaringan NAT IPv4 yang terhubung dengan host IPv4. Lalu tahap berikutnya adalah pengujian *video streaming* yang diterapkan pada jaringan NAT IPv4 yang terhubung dengan *host* IPv6. Dan yang terakhir adalah pengujian video streaming pada jaringan IPv6 murni.

Dari hasil uji coba didapatkan *latency* yang dihasilkan mekanisme teredo lebih besar 5,4% dibandingkan NAT IPv4 dan lebih besar 5,6% dibandingkan IPv6, dengan kata lain pada mekanisme teredo pengiriman video akan berjalan lebih lambat dibanding dua konfigurasi lainnya.

Throughput yang didapat teredo lebih besar 5,3% dibandingkan dengan jaringan IPv6 dan dan lebih besar 6,66% dibandingkan dengan jaringan NAT IPv4, dengan kata lain untuk file berukuran besar, teredo mempunyai bandwidth yang cukup besar dibandingkan kedua konfigurasi lainnya.

Pengujian ini menujukkan hasil yang positif, mekanisme teredo yang diuji dapat dipakai sebagai solusi pada jaringan NAT IPv4 yang ingin terhubung dengan jaringan IPv6.

Kata Kunci : *teredo, tunneling, IPv6, video streaming*

Counsellor :
Ir. A. Endang Sriningsih, MT

ABSTRACT

In this research test bed will be build to try the conectivity from NAT IPv4 to IPv6. So, in the future, all the IPv4 NAT network still can connected to IPv6 network. In this paper, we will analyze 3 parameters, there are Throughput, Latency, and Packet Loss. Aplication that will be use for this research is video streaming. Testing will be held three times. First is in ordinary NAT IPv4 network, second is in teredo method and the last one is in ordinary IPv6 network..

From the experiment, Latency of teredo 5,4% longer than NAT IPv4 and 5,6% longer than IPv6, in other word by using teredo video transfer rate will be slower than two other configuration.

Throughput result in teredo showed 5,3% bigger than IPv6 and 6,6% bigger than NAT IPv4, it means for bigger file teredo bandwidth is bigger than two other configuration and can transfer data faster.

This research had given proper result, teredo mechanism could be use for NAT IPv4 network to be connected with IPv6 network

Key Word : *teredo, tunneling, IPv6, video streaming*

DAFTAR ISI

PENGESAHAN ii
PERNYATAAN KEASLIAN SKRIPSI iii
UCAPAN TERIMA KASIH iv
ABSTRAK v
ABSTRACT vi
DAFTAR ISI vii
DAFTAR GAMBAR ix
DAFTAR TABEL x
BAB I 1
PENDAHULUAN 1
1.1 LATAR BELAKANG 1
1.2 TUJUAN PENULISAN 2
1.3 BATASAN MASALAH 2
1.4 SISTEMATIKA PENULISAN 2
BAB II 4
INTERKONEKSI DAN *VIDEO STREAMING* 4
2.1 INTERNET PROTOKOL SAAT INI 4
2.2 INTERNET PROTOCOL (IP) V 6 5
2.3 KEBUTUHAN DALAM IMPLEMENTASI 8
2.4 METODE MIGRASI 8
2.5 *TUNNELING* IPv4 KE IPv6 9
2.6 METODE TRANSLASI 11
2.7 IPv4 to IPv6 *Behind* NAT 12
2.7.1 NAT (*Network Address Translation*) 13
2.7.1.1 Tipe NAT 13
2.7.1.2 Translasi NAT 14
2.7.2 Mekanisme Teredo 16

2.8 APLIKASI *VIDEO STREAMING* 18
2.8.1 Protokol Pada Video Streaming 19
2.9 PARAMETER-PARAMETER DALAM *VIDEO STREAMING* 20
BAB III 22
KONFIGURASI JARINGAN DAN METODE PENGAMBILAN DATA 22
3.1 TOPOLOGI JARINGAN 22
3.2 KELENGKAPAN KONFIGURASI JARINGAN 23
3.3 METODE PENGAMBILAN DATA 24
BAB IV 25
ANALISA 25
4.1 PENGOLAHAN DATA 25
4.2 ANALISA HASIL PERCOBAAN 31
4.2.1 Analisa *Latency* 31
4.2.2 Analisa Throughput 35
4.2.3 Analisa *Packet Loss* 40
4.2.4 Analisa Tampilan *Visual* dan *Audio* 44
BAB IV 46
KESIMPULAN 46
DAFTAR ACUAN 47

DAFTAR GAMBAR

Gambar 2.1. Paket IPv6 5
Gambar 2.2. Pembagian paket IPv6 6
Gambar 2.3. Header IPv6 7
Gambar 2.4 Proses Enkapsulasi 9
Gambar 2.5a Router-to-router 10
Gambar 2.5b Host-to-router 10
Gambar 2.5c Host-to-host 10
Gambar 2.5d Router-to-host 11
Gambar 2.6 TCP/IP Layer 11
Gambar 2.7 Teredo *sample* 13
Gambar 2.8 *Cone NAT* 14
Gambar 2.9 Restricted NAT 15
Gambar 2.10 Konfigurasi alamat teredo 17
Gambar 2.11 Arsitektur *Video Streaming* 19
Gambar 2.12 Konektivitas Protokol-Protokol pada *Media Streaming* 20
Gambar 3.1 Topologi Jaringan 22
Gambar 4.1 Proses *streaming* pada *host server* 26
Gambar 4.2 Proses *streaming* pada *host client* 26
Gambar 4.3 Tampilan *capturing* aplikasi wireshark 27
Gambar 4.4 Tampilan *summary* dari wireshark 28
Gambar 4.5 Grafik latency pada pada TestVid.avi 31
Gambar 4.6 Grafik latency pada TestVid.mpg 32
Gambar 4.7 Grafik latency pada TestVid.mp4 32
Gambar 4.8 Grafik latency pada avilong.avi 33
Gambar 4.9 Grafik perbandingan latency keseluruhan 33
Gambar 4.10 Grafik *throughput* pada TestVid.avi 36
Gambar 4.11 Grafik *throughput* pada TestVid.mpg 36

Gambar 4.12 Grafik *throughput* pada TestVid.mp4 37
Gambar 4.13 Grafik *throughput* pada avilong.avi 37
Gambar 4.14 Grafik perbandingan throughput keseluruhan 38
Gambar 4.15 Grafik *packet loss* pada TestVid.avi 40
Gambar 4.16 Grafik *packet loss* pada TestVid.mpg 41
Gambar 4.17 Grafik *packet loss* pada TestVid.mp4 41
Gambar 4.18 Grafik *packet loss* pada Avolong.avi 42
Gambar 4.19 Grafik perbandingan *packet loss* keseluruhan 42

DAFTAR TABEL

Tabel 4.1 Hasil pengolahan data pada jaringan NAT Ipv4 29
Tabel 4.2 Hasil pengolahan data pada jaringan Ipv6 29
Tabel 4.3 Hasil pengolahan data pada metode Teredo 29
Tabel 4.4 Hasil pengolahan data video streaming Testvid.avi 30
Tabel 4.6 Hasil pengolahan data video streaming Testvid.mp4 30
Tabel 4.7 Hasil pengolahan data video streaming avilong.avi 30
Tabel 4.8 Tabel perbandingan latency secara keseluruhan 31
Tabel 4.9 Throughput pada semua konfigurasi jaringan 35
Tabel 4.10 Tabel perbandingan *packet loss* secara keseluruhan 40

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Internet yang selalu digunakan pada saat ini dilandasi oleh sebuah protokol yang bernama TCP/IP (*Transmission Control Protocol/Internet Protokol*). Para perancang dan pengembang TCP/IP pun akhirnya menetapkan metode pengalamatan yang disebut IPv4. Namun, tanpa disadari oleh pihak perancang dan pengembang pengalamatan pada IPv4 ini terancam habis pada masa-masa mendatang. Hal ini dipicu oleh pesatnya media informasi dan *backbone routing table* yang terus berkembang. Disinyalir bahwa alamat IPv4 ini sudah dalam keadaan yang kritis.

Oleh karena itu, maka sangatlah krusial untuk membentuk protokol pengalamatan yang baru untuk mengantisipasi krisis pengalamatan yang terjadi. Kemunculan IPv6 atau IPng memecahkan masalah tersebut. Hal ini dikarenakan besarnya alamat yang dimiliki oleh IPv6.

IPv6 mempunyai banyak keunggulan dibandingkan IPv4, diantaranya adalah jumlah alamat yang sangat besar, IPv6 mempunyai jumlah alamat 2^{128} atau setara dengan 3,4 X 10^{38} *host*. Selain itu, *header* yang dimiliki oleh IPv6 lebih sederhana dibandingkan IPv4, hal ini dikarenakan IPv6 hanya memiliki 7 *field* yang tentu lebih kecil dibandingkan IPv4 yang memiliki 13 *field*, hal ini turut membuat IPv6 lebih cepat dbandingkan IPv4. *QoS* yang dimiliki oleh IPv6 pun jauh lebih baik dibandingkan IPv4 sehingga peran multimedia juga turut dapat ditingkatkan. Selanjutnya adalah tingkat fleksibilitas yang lebih tinggi dalam penggunaan *option* dan *extension* yang dimiliki oleh IPv6. Yang terakhir adalah tingkat keaman yang dimiliki oleh IPv6 juga lebih tinggi dibandingkan IPv4.

Dengan berbasiskan keunggulan yang lebih baik pada kinerja *multimedia*, maka pada skripsi ini akan dilakukan pengujian pada layanan *video streaming*. Aplikasi *video streaming* pada IPv6 memiliki kesempatan yang besar untuk terus berkembang karena dapat menggunakan fasilitas yang sudah ditanamkan pada IPv6.

Berkembangnya *video streaming* pada IPv6 tentu saja membuat *host* yang berada dalam IPv4 tertarik untuk ikut merasakan fasilitas tersebut. Oleh karena

itulah dikembangkan metode *video streaming* sehingga bisa ditransmisikan dari *host* yang mempunyai IPv6 ke dalam jaringan NAT IPv4.

1.2 TUJUAN PENULISAN

Skripsi ini bertujuan untuk melakukan pengujian kinerja dari *video streaming* yang dikirimkan dari *host* IPv6 ke dalam *node* yang berada dalam jaringan IPv4 dibalik NAT. Hal ini akan dilakukan menggunakan metode *tunneling* TEREDO. Sampai saat ini *teredo* adalah satu-satunya cara untuk mengkoneksikan jaringan IPv4 yang berada di balik NAT ke jaringan IPv6. Pengujian dilakukan dengan menggunakan *Test bed* dan akan dianalisa sesuai dengan parameter-parameter yang sudah ditentukan. Parameter-parameter tersebut adalah *throughput, latency,* dan *packet loss*. Setelah didapatkan, hasilnya akan dibandingkan dengan parameter yang sama pada host IPv4 dengan kata lain jaringan NAT yang sudah umum yaitu IPv4 dibalik NAT ke *host* IPv4 yang lain. Selain itu juga akan dilakukan perbandingan unjuk kerja dengan jaringan IPv6.

1.3 BATASAN MASALAH

Permasalahan yang akan diteliti dan diuji hanya dibatasi pada proses transisi IPv6 dengan metode TEREDO dan menggunakan aplikasi *video streaming*. Pengujian yang dilakukan menggunakan jaringan lokal berupa *test bed* IPv6. *Test bed* ini dibuat dengan menggunakan tujuh buah komputer.

Aplikasi *video streaming* yang digunakan dalam pengujian ini *adalah Video Lan Client* (VLC) yang di install ke dalam *host* pengirim dan *host* penerima. Berdasarkan hasil pengolahan data akan dianalisa kelebihan dan kekurangan yang terjadi ketika menggunakan TEREDO dan ketika data hanya dikirimkan melalui jaringan NAT biasa dan jaringan IPv6 murni.

1.4 SISTEMATIKA PENULISAN

Skripsi ini dibagi menjadi lima bab :

1. Pada bab I berisi pendahuluan,
2. Pada bab II berisi tentang metode *tunneling* yang dilakukan,

3. Pada bab III berisi penjelasan mengenai konfigurasi dan topologi jaringan serta metode pengujian yang dilakukan,
4. Pada bab IV berisi tentang analisa,
5. Pada bab V berisi tentang kesimpulan.

BAB II
INTERKONEKSI DAN *VIDEO STREAMING*

2.1 INTERNET PROTOKOL SAAT INI

Dengan perkembangan teknologi internet yang semakin pesat menyebabkan *user* internet semakin bertambah banyak. Namun, pertambahan *user* internet tidak diimbangi dengan jumlah alamat IPv4 yang ada. Sehingga IETF berencana mengeluarkan standar *protocol* IP baru yang disebut IPng (*Internet Protokol Next Generations*) atau disebut juga IPv6.

IPv4 yang sudah terbukti tangguh menopang internet sekarang mulai bermasalah dengan semakin berkurangnya alokasi ip address yang tersedia. Walaupun IPv4 cukup sukses dalam efisiensi *address* dengan penggunaan NAT *(Network Address Translation),* tetapi tuntutan aplikasi internet yang bersifat *real-time* dan aman tidak dapat terpenuhi. NAT juga menghambat aplikasi yang bersifat end to end user ,*seperti Video Conference.* Penggunaan IPv6 adalah solusi yang tepat untuk menopang internet sekarang. Banyak keuntungan yang diambil dari penggunaan IPv6 yaitu : Alokasi *address* yang lebih banyak, *Auto configuration address*, serta *traffic class dan flow label* untuk mendukung aplikasi *real-time* dan Ipv6 mendukung *mobile* ip,IPsec dll.

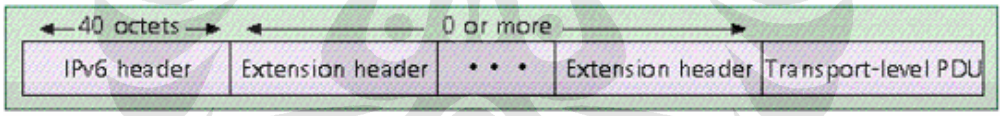
IPv6 mempunyai *format* alamat dan *header* yang berbeda dengan IPv4. Hal ini menyebabkan IPv4 tidak bisa secara langsung melakukan interkoneksi dengan IPv6. Hal ini tentunya akan menimbulkan masalah pada implementasi IPv4 pada jaringan internet IPv6 yang telah ada. Sebagai solusi masalah implementasi IPv6 ini diperlukan suatu mekanisme Transisi IPv6.

Tujuan pembuatan mekanisme transisi ini adalah supaya paket IPv4 dapat dilewatkan pada jaringan IPv6 ataupun sebaliknya. Pada skripsi ini akan dibahas cara implementasi mekanisme TEREDO translation untuk interkoneksi IPv4 dan IPv6.

2.2 INTERNET PROTOCOL (IP) V 6

Dengan berubahnya sifat dari penggunaan internet dan jaringan-jaringan kerja, *internet protokol* yang pada saat ini menggunakan jaringan TCP/IP akan mengalami keterbatasan. Hingga masa beberapa waktu yang lalu penggunaan internet dan kebanyakan dari jaringan TCP/IP memiliki kemampuan untuk pengiriman aplikasi data yang sederhana, seperti *File transfer*, *e-mail* , serta *remote access* menggunakan Telnet. Namun saat ini penggunaan internet telah berkembang menjadi keperluan *Multimedia*, yang kaya dengan kemampuan aplikasi, dan pada saat yang bersamaan jaringan korporasi telah berubah dari hanya sekedar aplikasi pengiriman surat dan data menjadi hubungan yang kompleks antara pengguna (*client*) dan *server* dalam suatu jaringan-jaringan lokal (*intranet*) dan memungkinkan penggunaan dan menjalankan aplikasi melalui internet[4].

Pada IPv6, *protocol data unit* atau biasa dikenal dengan paket secara umum terlihat pada Gambar 2.1.



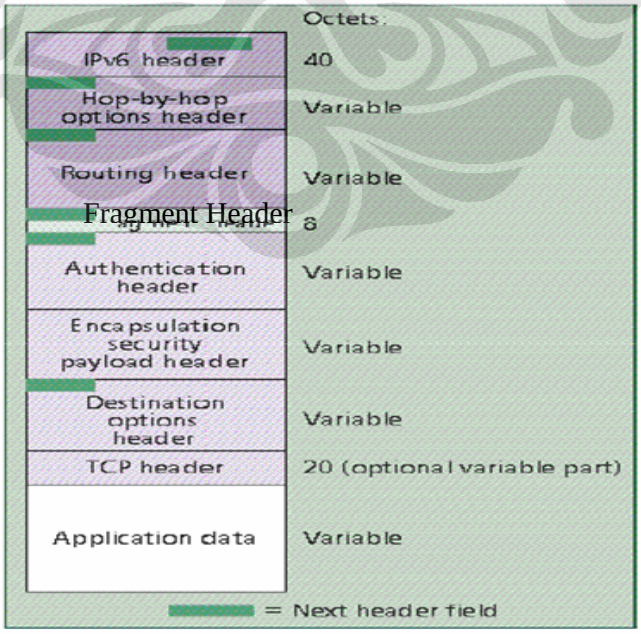
Gambar 2.1. Paket IPv6

Pada gambar diatas terlihat panjang dari *header* IPv6 sudah tetap yaitu empat puluh *octets*, bila dibandingkan dengan *header* pada IPv4 yang hanya dua puluh *octets*, maka *header* IPv6 sudah dipersiapkan untuk keperluan sebagai berikut:

1. *Hop-by-hop options header*: dipersiapkan untuk keperluan header proses hop to hop
2. *Routing header*: dipersiapkan untuk penambahan route, sama halnya dengan IPv4
3. *Fragment header*: berisi fragmentation dan informasi reassembly
4. *Authentication header*: disediakan untuk packet integrity dan authentication

5. *Encapsulating security payload header*: dipersiapkan untuk keperluan khusus
6. *Destination options header*: berisi informasi tambahan berkenaan dengan informasi node tujuan. Rekomendasi standar untuk IPv6 bila *multi extension header* digunakan, maka *header* dari IPv6 akan muncul atau tersusun sebagaimana berikut ini:
7. IPv6 *header*: kewajiban atau harus muncul pertama kali.
8. *Hop-by-hop options header*
9. *Destination options header*
10. *Routing header*
11. *Fragment header*
12. *Authentication header*
13. *Encapsulating security payload header*

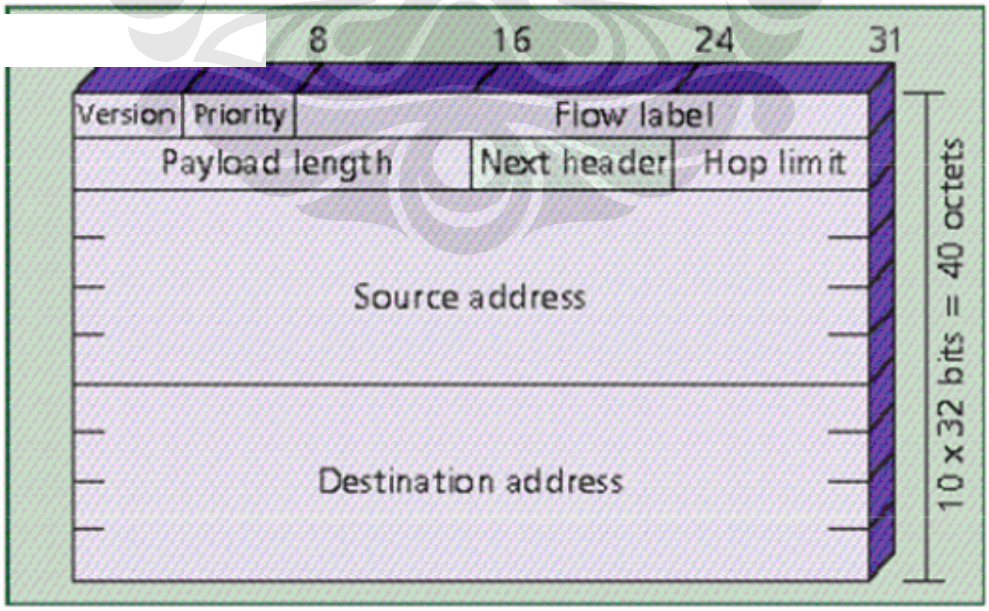
Pada Gambar 2.2 di halaman selanjutnya dapat dilihat contoh dari Paket IPv6 dan isi dari masing-masing *header*, dengan catatan bahwa *header* dari IPv6 dan masing-masing *extension header* termasuk dalam file *header* selanjutnya, *field* ini menunjukkan tipe dari *header* yang akan mengikuti kemudian, jika *header* selanjutnya merupakan *extension header*, maka *field* akan berisi identitas tipe dari *header*.



Gambar 2.2. Pembagian paket IPv6

Header dari IPv6 memiliki panjang yang sudah tetap yaitu sebesar 40 octets, yang berisi sebagai berikut :

1. *Version* (4 bits): nomor dari versi IP, nilainya adalah 6
2. *Priority* (4 bits): *priority value*, akan didiskusikan kemudian
3. *Flow label* (24 bits): mungkin akan digunakan oleh host untuk memberikan label atau tanda dari paket yang memerlukan penanganan khusus oleh perangkat router pada jaringan
4. *Payload length* (16 bits): memberi tanda atau mengingatkan akan panjang dari paket IP yang mengikuti header, dengan kata lain, merupakan panjang keseluruhan *extension header* ditambah dengan *transport-level* PDU
5. *Next header* (8 bits): merupakan identitas atau menandai dari tipe *header* yang akan segera mengikuti *header* IPv6
6. *Hop limit* (8 bits): mengingatkan nomor atau jumlah hop yang diperbolehkan untuk suatu paket
7. *Source address* (128 bits): merupakan alamat dari asalnya paket
8. *Destination address* (128 bits): merupakan alamat dari tujuan paket



Gambar 2.3. Header IPv6

Sekalipun IPv6 *header* lebih panjang dari *header* IPv4 (40 octects versus 20 octects), namun isi *field*-nya lebih pendek (8 versus 12), sehingga waktu proses pada router untuk memproses tiap *header* lebih singkat, hal ini berarti akan terjadi peningkatan kecepatan dalam *routing*.

2.3 KEBUTUHAN DALAM IMPLEMENTASI

Untuk mengimplementasikan suatu jaringan IPv6, yang pertama-tama dibutuhkan adalah alokasi prefix IPv6 yang unik, yang bisa diperoleh dari penyedia jasa Internet seperti ISP, APJII, APNIC atau yang lain.

Kebutuhan selanjutnya adalah infrastruktur jaringan yang telah mendukung IPv6. Telah diketahui bersama bahwa IPv6 terdefinisi pada *layer* 3, maka perangkat *hardware/software layer* 2 ke bawah yang telah ada dapat digunakan, sehingga hanya perangkat jaringan pada *layer network* seperti router harus dipersiapkan untuk mendukungnya.

Aplikasi jaringan sebagai layer selanjutnya tentu saja juga harus ikut mendukung IPv6, sehingga komunikasi antara client IPv6 dengan server IPv6 dapat dilangsungkan. IPv6 adalah teknologi baru yang akan menggantikan teknologi lama IPv4, karena itu dibutuhkan niat dan usaha yang lebih dari pengelola dan pengguna jaringan (yang dapat digeneralisasi sebagai komunitas saja) untuk mengimplementasikan (dan melakukan migrasi nantinya) ke IPv6.

2.4 METODE MIGRASI

Terdapat 4 metode migrasi yang dapat di lakukan secara bertahap dan dapat di implementasikan yaitu :

1. Teknik *dual-stack*

   Menambahkan IPv6 dengan tidak menghapus IPv4 yang sudah ada. Hal ini dapat diterapkan di beberapa *operating system* yang sudah mempunyai dua *interface virtual* sekaligus seperti linux.

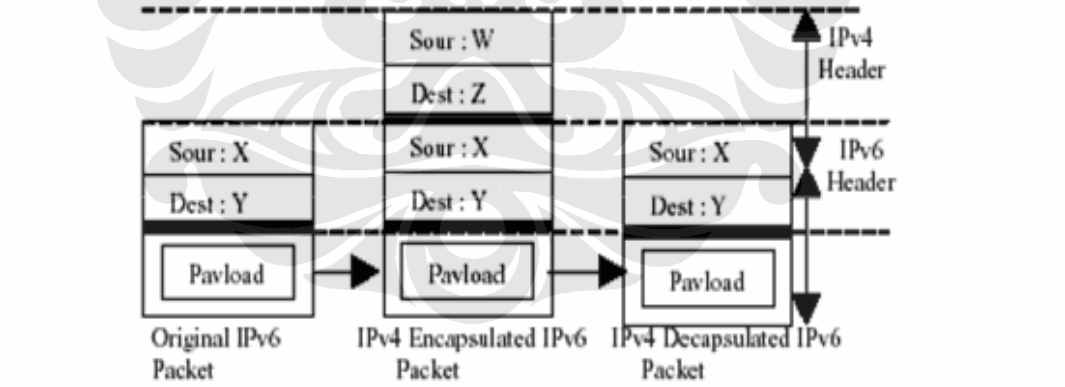
2. Teknik *tunnel* IPv6 di dalam IPv4

Teknik ini menggunakan IPv4 sebagai datalink layer dan IPv6 akan di enkapsulasi kedalam jaringan IPv4.

3. Teknik translasi dari *Network Address Translation* ke protocol transfer.
   Menggunakan network address translation dari IPv6 ke IPv4 dan sebaliknya.
4. Teknik TCP relay dari IPv6 ke IPv4
   Dimana aplikasi yang menggunakan IPv6 direlay dalam domain name server (DNS) yang kemudian di teruskan ke DNS yang menggunakan IPv4

2.5 *TUNNELING* IPv4 KE IPv6

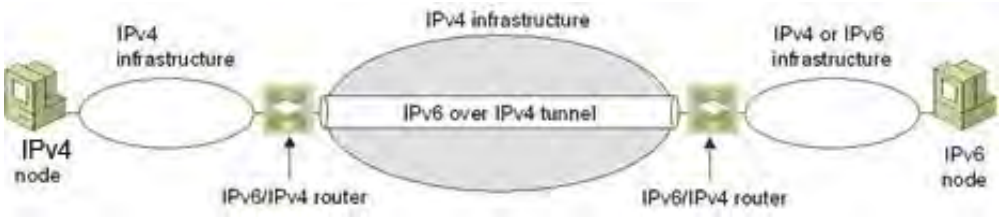
Fungsi dari enkapsulasi paket IPv6 dengan header IPv4 adalah agar paket dapat diroutingkan oleh router IPv4.Namun dengan penambahan *header* IPv4 ini tentunya paket akan bertambah besar sesuai panjang *header* IPv4 yaitu 20 byte.Pertambahan panjang paket ini akan berakibat bertambah pula waktu *delay* pengiriman paket. Masalah utama dari implementasi mekanisme Transisi IPv6 adalah pertambahan waktu *delay* proses yang diakibatkan pertambahan panjang paket, adanya proses enkapsulasi dan adanya proses dekapsulasi. Proses enkapsulasi tersebut dapat dilihat pada Gambar 2.4 berikut



Gambar 2.4 Proses Enkapsulasi

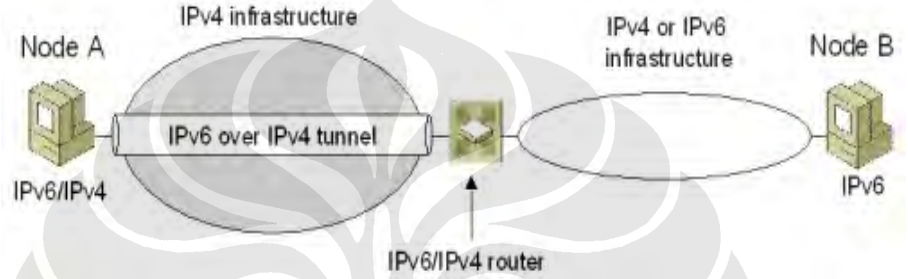
Proses enkapsulasi dari paket IPv6 didalam paket IPv4 terjadi melalui berbagai cara yang digambarkan pada Gambar 2.5a sampai Gambar 2.5d:

1. Router-to-Router **:** *Infrastruktur* dari IPv4 akan melakukan proses *tunneling* sehingga trafik dari IPv4 dan IPv6 akan saling berhubungan.



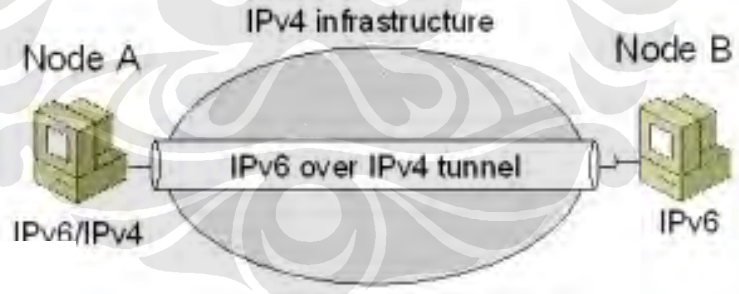
Gambar 2.5a Router-to-router

2. Host-to-Router : *host* IPv4/IPv6 akan melakukan proses *tunneling* trafik IPv4 ke route IPv6.



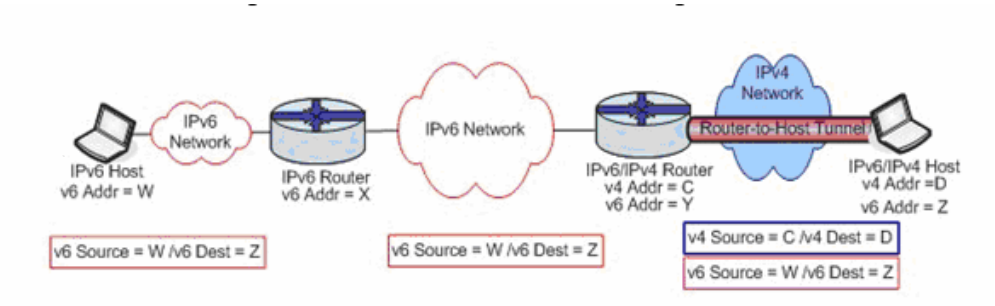
Gambar 2.5b Host-to-router

3. Host-to-Host : Infrastruktur IPv4 akan memiliki *tunnel* IPv6 antara host IPv6/IPv4 yang berhubungan secara langsung.



Gambar 2.5c Host-to-host

4. Router-to-Host : Router IPv6/IPv4 akan mencapai *host* IPv4/IPv6 dengan menggunakan *tunnel*.

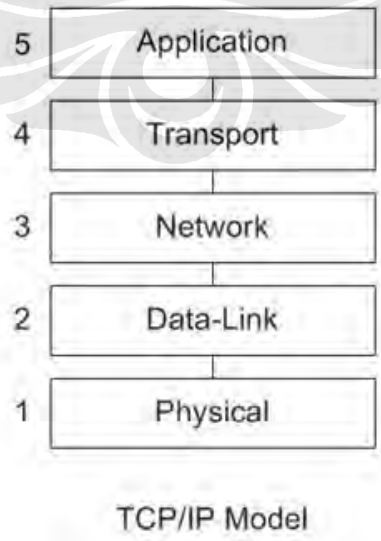


Gambar 2.5d Router-to-host

Ada dua jenis metode *tunneling*, yaitu : *tunneling* terkonfigurasi dan *tunneling* otomatis (*dynamic*). Pada *tunneling* terkonfigurasi *host* atau router dikonfigurasi secara *manual* untuk berkomunikasi dengan router perantara *dual-stack* pada jaringan IPv4 yang menjadi akhir dari proses *tunneling*. Pada *tunneling* otomatis, alamat IPv4 secara otomatis akan didapat. Tidak seperti pada *tunneling* terkonfigurasi, *tunneling* otomatis dipakai antara jaringan yang terjadi pertukaran (transisi).

2.6 METODE TRANSLASI

Setelah terkoneksi, maka selanjutnya diperlukan sebuah metode supaya *node* IPv4 bisa berkomunikasi dengan *node* IPv6. Mekanisme translasi ada bermacam-macam yang bisa dibedakan berdasarkan *layer* proses translasi dijalankan. Gambar 2.6 akan menunjukkan layer-layer pada TCP/IP.



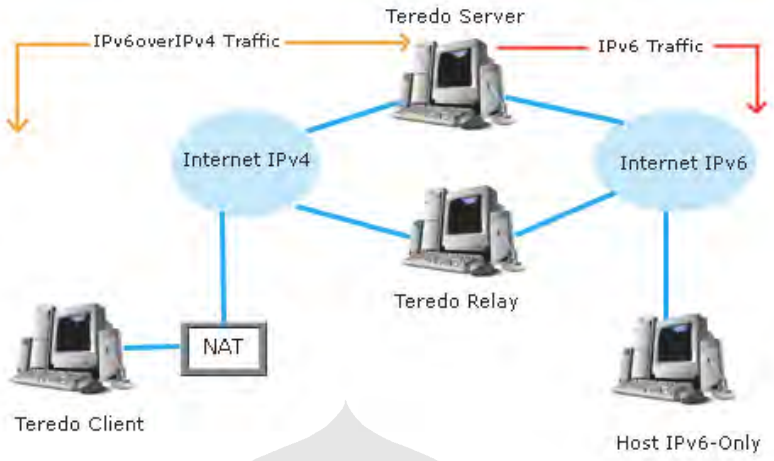
Gambar 2.6 TCP/IP Layer

Daftar dari metode-metode translasi yang terdapat pada *layer-layer* tersebut dapat dilihat di bawah ini:

1. Network Layer
    a. RFC 2765 – Stateless IP/ICMP Translation Algorithm (SIIT)
    b. RFC 2766 – NAT-PT
    c. RFC 2767 – Bump in the Stack (BIS)
2. Transport Layer
    a. RFC 3142 – Transport Relay Translator (TRT)
3. Application Layer
    a. RFC 3338 – Bump in the API (BIA)

2.7 IPv4 to IPv6 *Behind* NAT

IETF mendesain IPv6 untuk mengatasi kekurangan pada internet IPv4. Namun untuk implementasi IPv6 pada jaringan internet IPv4 yang telah ada perlu suatu mekanisme transisi. Karena IPv6 mempunyai perbedaan dengan IPv4 yaitu pada format header IP, format pengalamatan dan *command* disisi *operating system*. Dengan adanya NAT yang masih banyak diimplementasikan pada jaringan internet sekarang ternyata menghambat proses transisi IPv6 dan IPv4.Karena pada mekanisme transisi yang telah ada seperti DSTM,Tunelling dan NAT-PT tidak bisa mengkoneksikan *client* transisi yang berada di belakang IPv4 NAT[4]. Oleh karena itu, ntuk interkoneksinya dibutuhkan *interkoneksi* khusus yang disebut mekanisme transisi Teredo (Shipworm). Gambar 2.7 akan memberikan ilustrasi sebuah jaringan *teredo*



Gambar 2.7 Teredo *sample*

2.7.1 NAT (*Network Address Translation*)

Network Address Translation merupakan suatu metode dimana IP address dipetakan dari satu grup ke grup lainnya secara transparan bagi sisi penerima. Operasi di atas mengacu kepada *Traditional NAT*, yang menyediakan mekanisme komunikasi antara jaringan lokal (alamat IP tidak terdaftar di internet) dengan jaringan global (alamat IP terdaftar di internet)[5].

2.7.1.1 Tipe NAT

NAT terdiri dari dua tipe yaitu Statik dan Dinamik yang keduanya dapat digunakan secara terpisah maupun bersamaan. Dua tipe NAT tersebut adalah :

1. **Statik**

   Translasi Statik terjadi ketika sebuah alamat lokal (*inside*) di petakan ke sebuah alamat global/internet (*outside*). Alamat lokal dan global dipetakan satu lawan satu secara statik.

2. **Dinamik**

   - **NAT dengan Pool (kelompok)**

     Translasi Dinamik terjadi ketika router NAT diset untuk memahami alamat lokal yang harus ditranslasikan, dan kelompok (*pool*) alamat global yang akan digunakan untuk terhubung ke internet. Proses NAT Dinamik ini dapat

memetakan bebarapa kelompok alamat lokal ke beberapa kelompok alamat global.

- **NAT Overload**

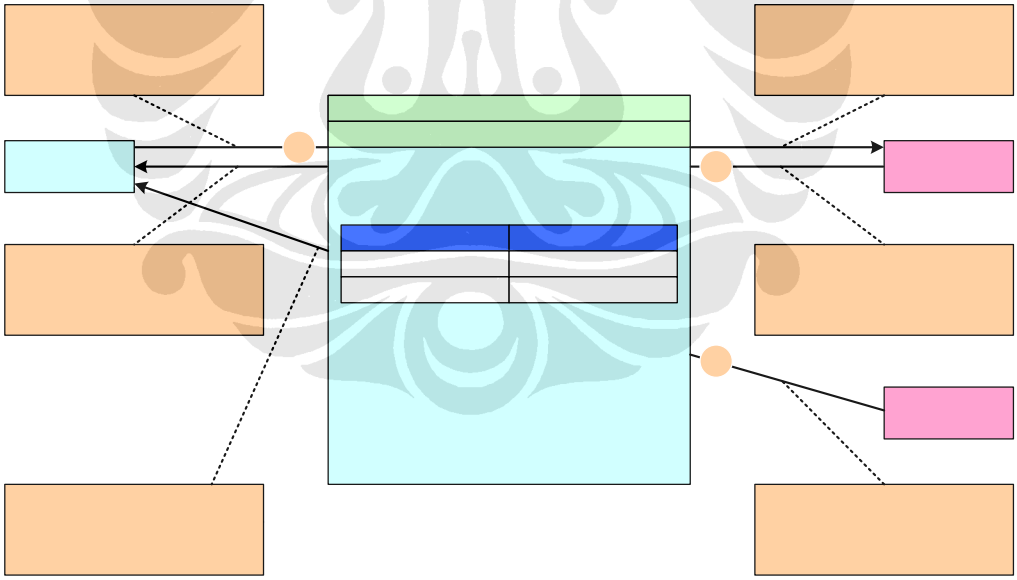
  Sejumlah IP lokal/internal dapat ditranslasikan ke satu alamat IP global/internet. Hal ini sangat menghemat penggunakan alokasi IP global dari ISP.

2.7.1.2 Translasi NAT

Berdasarkan cara translasinya NAT dibagi menjadi 3 bagian, yaitu :

1. **Cone NAT**

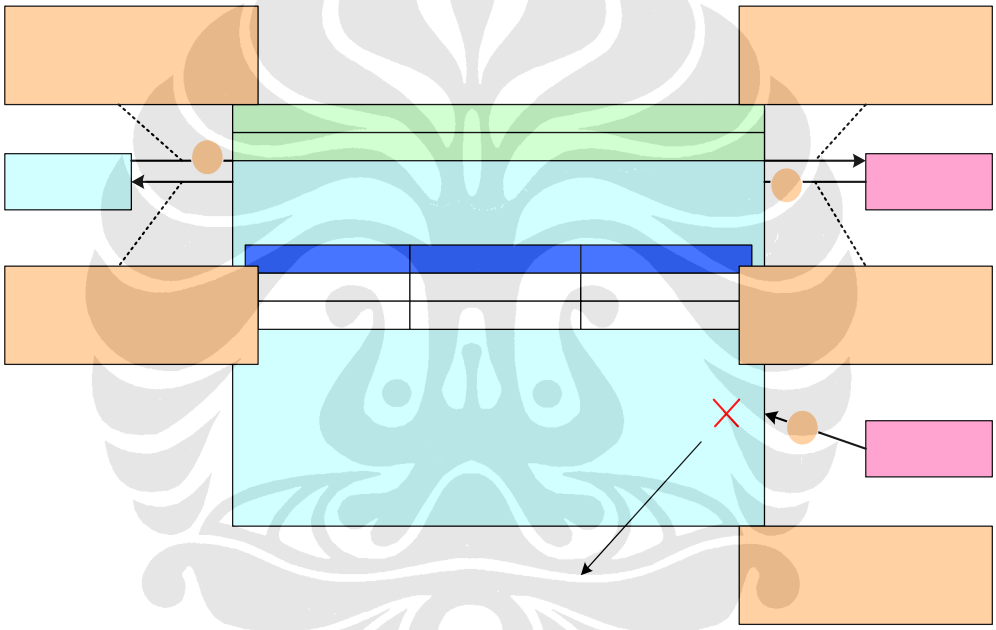
   Merupakan jenis translasi alamat dan *port* internal dari *host* yang berada di belakang perangkat NAT ke sebuah alamat dan *port* eksternal, jadi semua trafik yang berasal dari alamat di luar perangkat NAT akan dapat diteruskan ke *host* yang berada di belakang NAT. Ilustrasi dapat dilihat pada Gambar 2.8



Gambar 2.8 *Cone NAT*

2. Restricted NAT

Merupakan jenis translasi alamat dan *port* internal dari *host* yang berada di belakang perangkat NAT ke suatu alamat dan *port* eksternal. Alamat tujuan dari paket yang dikirim oleh *host* yang berada di belakang perangkat NAT akan disimpan dalam tabel NAT. Trafik yang berasal dari alamat di luar perangkat NAT hanya akan diteruskan apabila alamat tersebut terdapat di dalam tabel NAT. Ilustrasi dapat dilihat pada Gambar 2.9



Gambar 2.9 Restricted NAT

Selain dua jenis translasi NAT di atas juga ada jenis translasi lain yaitu Symetric NAT. Namun, jenis translasi NAT tersebut tidak dapat dipakai pada tunneling Teredo dikarenakan pada symetric NAT port yang akan di-*assign* merupakan sebuah port acak yang tidak bisa ditebak. Hal ini tentu saja akan menyulitkan teredo yang bekerja dengan sistem terstruktur dimana member dari NAT akan diberikan header sehingga menjadi sebuah alamat IPv6.

2.7.2 Mekanisme Teredo

Seperti telah dibahas di atas, cara paling efektif menghemat penggunaan alamat IPv4 adalah menggunakan NAT atau *Network Address Translation*. Dimana pada penerapannya komputer-komputer yang berada dibalik NAT akan mendapat alamat IP yang disebut IP *private*. Banyaknya pengguna NAT ini pada nantinya akan dapat menghalangi migrasi yang akan dilakukan untuk mengubah jaringan dunia pada akhirnya memakai IPv6 karena seperti diketahui jaringan IPv4 tidak dapat langsung terhubung kepada jaringan IPv6. Untuk itulah muncul mekanisme Teredo, tidak seperti konsep *tunneling* biasa seperti 6to4, 6over4, ISATAP, dan yang lainnya yang hanya bisa membuat *tunnel* antarnode mekanisme ini bekerja dalam jaringan IPv4 yang berada di balik NAT.

Secara umum teredo terbagi atas 3 komponen yaitu :

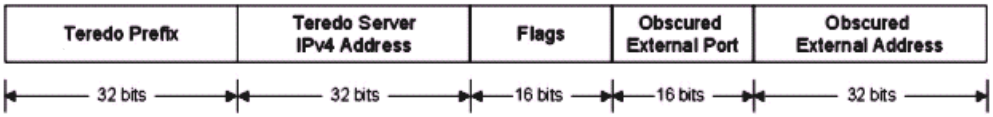
1. Teredo Server
2. Teredo Relay
3. Teredo Client

Teredo Server merupakan komputer yang terhubung baik ke jaringan NAT IPv4 maupun ke jaringan IPv6. Fungsi dasar teredo server adalah membantu alamat konfigurasi dari sebuah teredo client. Teredo server bertanggung jawab memberikan alamat konfigurasi IPv6 ke sebuah host IPv4.

Teredo Relay merupakan sebuah *device* baik itu komputer maupun router IPv4/IPv6 yang dapat memfasilitasi terkirimnya paket yang dikirimkan oleh IPv4 untuk dapat melewati tunnel yang ada. Teredo relay ini juga berhubungan dengan teredo server untuk memberikan alamat konfigurasi yang ada di teredo server ke teredo client.

Teredo Client merupakan host IPv4 yang ingin dapat terhubung dengan jaringan IPv6 yang ada. Dengan kata lain teredo client adalah Host IPv4 yang nantinya akan dilihat oleh sebuah host IPv6 seakan-akan merupakan host IPv6 juga.

Teredo client yang sudah terhubung dengan teredo server akan mempunyai konfigurasi alamat seperti ini :



Gambar 2.10 Konfigurasi alamat teredo

Seperti terlihat pada Gambar 2.10, alamat dari sebuah teredo client terdiri dari :

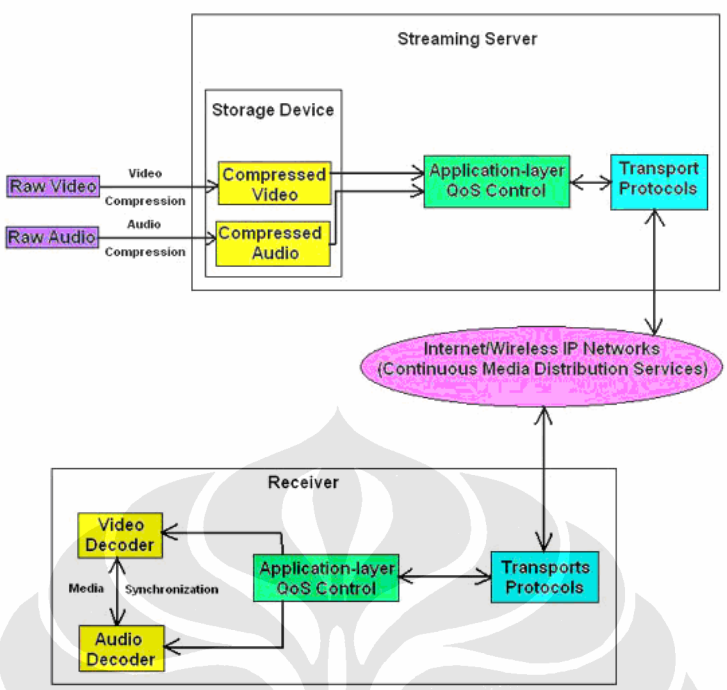
1. Teredo Prefix = 2001:0::/32
2. Teredo Server IPv4 Address
3. Flags
4. Obscured External Port
5. Obscured External Address

2.8 APLIKASI *VIDEO STREAMING*

Proses pengiriman *stored video* dari internet memiliki dua metode yaitu *download* dan *streaming*. Beberapa waktu yang lalu, metode *download* adalah metode yang sangat popular dan banyak dilakukan oleh pemakai internet. Pada metode ini file akan di-*download* ke dalam *drive user* dan dimainkan dengan kualitas yang tinggi[2].

Tentu saja hal ini mempunyai banyak kelemahan, diantaranya adalah waktu yang dibutuhkan untuk men-*download file* cukup lama dan *playback latency* juga besar. Oleh karena itulah lahir metode *Streaming*, secara ideal *video* dan *audio* harus dapat di-*streaming* melalui internet dari *server* ke *client* untuk menjawab *request client* terhadap suatu *video* yang berada dalam suatu website. *Client* harus dapat memainkan *multimedia stream* yang datang dalam kondisi *real-time* pada saat data diterima. Akan tetapi kondisi ideal untuk *video streaming* sangat bergantung kepada *bandwidth, delay,* dan *loss*. Hal inipun yang dapat menyebabkan proses streaming sering terhenti yaitu ketika *bandwidth* yang ada tidak memadai sehingga memicu terjadinya *loss* dan *delay* dalam pengiriman *video*.

Dalam *video streaming* ada beberapa proses yang harus diperhatikan yaitu, proses kompresi, *quality of service(QoS), continous media distribution services, streaming server*, mekanisme sinkronisasi, dan protokol untuk *media streaming*.



Gambar 2.11 Arsitektur *Video Streaming*

Dari Gambar 2.11 di atas dapat dilihat bahwa data *raw video* dan dan *data raw audio* akan dikompresi terlebih dan disimpan di dalam *storage device* dari *streaming server*. *Streaming server* akan mengirimkan data yang telah dikompresi dan tersimpan dalam *storage device* ketika menerima request dari *client* (melalui Internet). Data akan dikirimkan oleh *streaming server* dengan modul *application-layer* QoS. QoS control lalu menyesuaikan *bit-stream data* ke dalam status jaringan dan persyaratan QoS. Selanjutnya paket data tersebut akan dikirimkan oleh *transport protocol* ke dalam jaringan setelah mengalami penyesuaian. Setiap paket yang sampai di sisi penerima akan diproses terlebih dahulu oleh *transport layer* dan *application layer protocol* lalu akan didekodekan oleh decoder[2].

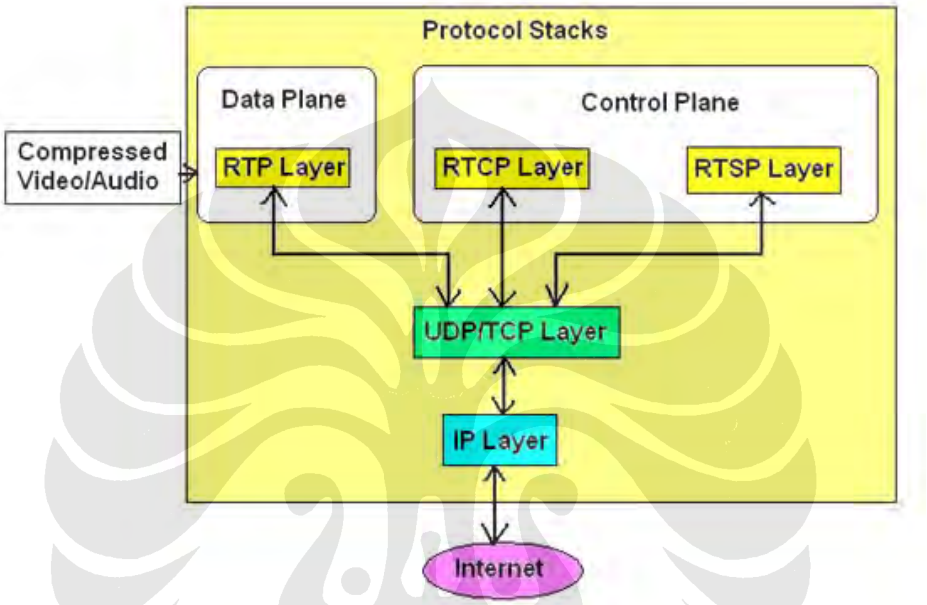
2.8.1 Protokol Pada Video Streaming

Ada dua protocol yang mendukung berjalannya *video streaming* yaitu:

1. Transport Protocol yang menyediakan konektivitas secara *end-to-end* di jaringan untuk aplikasi *streaming*. *Transport protocol* terbagi menjadi *User Datagram Protocol (UDP) dan Transmission Control Protocol (TCP).*

2. Session Control Protocol yang mendefinisikan pesan dan prosedur untuk mengatur pengiriman data dari multimedia selama session terbentuk. *Session control protocol* ini terbagi menjadi *Real-Time Streaming Protocol(RTP)* dan *Real-Time Streaming Protocol (RTSP).*

Hubungan antara protokol-protokol di atas dapat dilihat pada Gambar 2.12 berikut ini :



Gambar 2.12 Konektivitas Protokol-Protokol pada *Media Streaming*

2.9 PARAMETER-PARAMETER DALAM *VIDEO STREAMING*

Parameter-parameter krusial QoS untuk jaringan adalah *throughput, average end-to-end delay, delay jitter,* dan *loss rate.*

1. Throughput (*bandwidth*)

   Adalah gambaran keberhasilan sejumlah informasi yang bisa dikirimkan oleh suatu jaringan selama interval waktu tertentu. *Throughput* yang tinggi menghasilkan QoS yang lebih baik secara umum.

2. Latency (delay dan delay variation)

   Delay adalah waktu yang dihabiskan ketika suatu *event* terjadi. *Jitter* merupakan variasi dalam *delay*. *Latency* adalah lamanya paket yang dikirimkan dari satu node ke node yang lain.

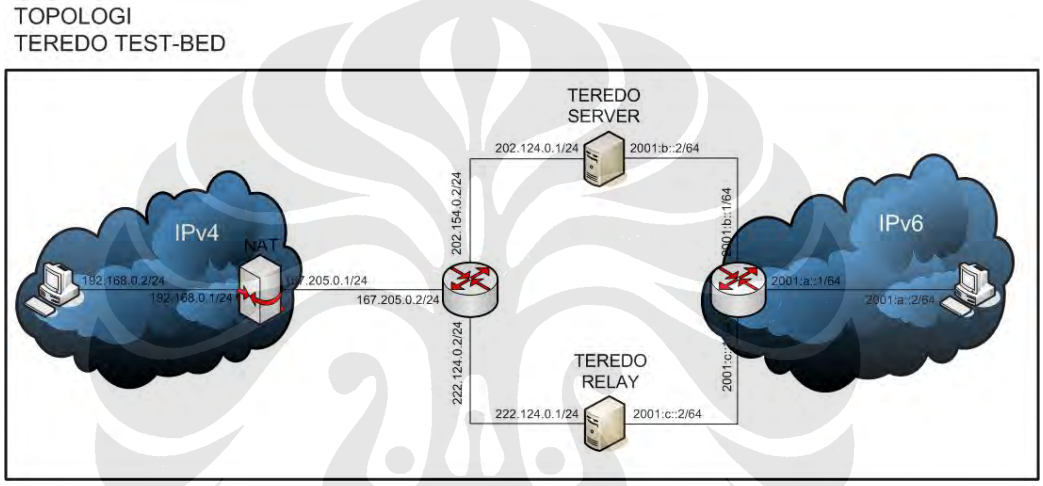
3. Packet Loss Rate (reliability)

Paket hilang dalam tingkat ketepatan atau waktu nyata (*real-time*). Jika sebuah paket sampai di tempat tujuan setelah batas waktu tertentu (*delay bound*) maka paket data tersebut akan dianggap *loss* atau hilang.

BAB III
KONFIGURASI JARINGAN DAN METODE PENGAMBILAN DATA

3.1 TOPOLOGI JARINGAN

Pada test bed yang dibuat untuk melakukan analisa kinerja mekanisme TEREDO ini digunakan 7 buah desktop komputer, dimana fungsi masing-masing komputer dapat dilihat pada Gambar 3.1 :



Gambar 3.1 Topologi Jaringan

1. Pada host IPv4 digunakan sebuah desktop komputer dengan sistem operasi windows XP service pack 2. PC ini menggunakan Pentium 4 1,7GHz dengan RAM 128Mbytes dan Ethernet Card 10/100 Mbps.
2. NAT menggunakan sebuah desktop komputer dengan sistem operasi linux ubuntu server. PC ini mempunyai spesifikasi Pentium III 300 MHz dengan RAM 128Mbytes dan Ethernet Card 10/100 Mbps.
3. Router 1 dan 2 menggunakan sebuah desktop komputer dengan sistem operasi linux ubuntu. Router 1 dan 2 merupakan sebuah PC router. Router 1 atau pada test bed disebut R1 adalah sebuah router yang bekerja pada jaringan IPv4, PC ini menggunakan Pentium III 677 MHz dengan RAM 128Mbytes dan Ethernet Card 10/100 Mbps. Router 2 atau pada test bed

disebut R2 adalah sebuah router yang bekerja pada jaringan IPv6, PC ini menggunakan Pentium III 300 MHz dengan RAM 128Mbytes dan Ethernet Card 10/100 Mbps.

4. Teredo server menggunakan sebuah desktop komputer dengan sistem operasi linux ubuntu. Merupakan sebuah PC dengan dual-stack dan mempunyai spesifikasi Pentium III Celeron 800MHz dengan RAM 128Mbytes dan Ethernet Card 10/100 Mbps.
5. Teredo relay menggunakan sebuah desktop komputer dengan sistem operasi linux ubuntu. Merupakan sebuah PC dengan dual-stack dengan spesifikasi Pentium III 677 MHz dengan RAM 128 Mbytes dan Ethernet Card 10/100 Mbps.
6. Pada host IPv6 menggunakan sebuah desktop komputer dengan sistem operasi windows XP service pack 2. PC ini mempunyai spesifikasi Pentium IV 1,7GHz dengan RAM 256 Mbytes dan ethernet Card 10/100 Mbps.

Pada jaringan IPv4 dengan NAT topologi jaringan tidak berubah, akan tetapi teredo server dan teredo relay tidak dipakai dan host IPv6 akan dirubah menjadi sebuah PC IPv4. Hal ini juga berlaku untuk topologi IPv6 murni.

3.2 KELENGKAPAN KONFIGURASI JARINGAN

Pada skripsi ini digunakan 2 buah sistem operasi yaitu Microsoft Windows XP dan Linux Ubuntu Server. Windows XP dipilih sebagai kedua host dari test bed yang dibentuk. Pemilihan windows XP dikarenakan interface dan konfigurasinya yang mudah untuk dilakukan. Sedangkan pemilihan Linux Ubuntu Server sebagai router, NAT, teredo server, dan teredo relay dikarenakan sistem yang open source akan mudah untuk di rekonfigurasi ketika terjadi kegagalan koneksi jaringan.

Sedangkan untuk software, yang digunakan antara lain :

1. WireShark, merupakan aplikasi untuk menangkap dan menganalisa trafik yang terdapat pada interface.
2. VideoLAN Client (VLC), merupakan aplikasi yang dipakai untuk melakukan streaming file-file audio dan video dari berbagai jenis format.

VLC dapat digunakan baik sebagai server maupun sebagai client. Dan hal ini berlaku bagi kedua IPv4 dan IPv6. VLC adalah aplikasi freeware atau bisa didapat dengan gratis[3].

3.3 METODE PENGAMBILAN DATA

Pengambilan data akan dilakukan dengan pengiriman file video streaming dengan menggunakan aplikasi VLC yang dijalankan pada jaringan NAT IPv4 murni, IPv6 murni, dan teredo. Parameter yang akan dianalisa adalah jumlah waktu pengiriman paket video streaming (time), packet loss, dan transfer rate dari proses pengiriman (throughput). Parameter ini akan didapatkan dengan menggunakan tools/aplikasi wireshark.

Pada skripsi ini akan diambil data dari 4 file video yang akan di-streaming melalui jaringan NAT IPv4, IPv6 dan teredo yaitu:

1. TestLongVid.avi, dengan ukuran file 23,5 MB dan durasi 2 menit 38 detik
2. TestVid.avi, dengan ukuran file 1,7MB dan durasi 28 detik
3. TestVid.mpg, dengan ukuran file 16,8MB dan durasi 28 detik
4. TestVid.mp4, dengan ukuran file 1,7MB dan durasi 28 detik

Pada pengambilan data pertama, file-file tersebut di atas akan dikirimkan melalui jaringan NAT IPv4 murni. Setelah itu file-file tersebut di atas akan di-*streaming* melalui mekanisme teredo yang dibangun. Berikutnya akan dilakukan unjuk kerja dengan konfigurasi jaringan IPv6 murni.

Pengambilan data akan dilakukan sebanyak 10 kali untuk masing-masing file pada setiap konfigurasi jaringan. Berarti akan didapat 40 buah data untuk setiap konfigurasi jaringan yang dengan kata lain akan terkumpul 120 buah data dari 3 konfigurasi jaringan. Dari hasil pengumpulan data itulah akan dapat dianalisa parameter latency, parameter throughput, dan parameter packet loss dengan cara membandingkan antara 2 konfigurasi jaringan yang dipakai.

BAB IV
ANALISA

4.1 PENGOLAHAN DATA

Pada test bed, proses pengiriman video pada *video streaming* menggunakan perangkat lunak VLC yang diuji coba menggunakan 3 macam konfigurasi, yaitu : jaringan NAT IPv4, Jaringan lokal IPv6, dan metode teredo. Proses pengambilan data akan ditangani oleh *tools* Wireshark. *Tools* ini akan mencatat dan menampilkan 3 parameter yang diteliti, yaitu : waktu pengiriman paket pada proses streaming (*Time*), jumlah paket yang hilang (*Packet Loss*), dan transfer rate dari proses streaming (*Throughput*).

Pada skripsi ini akan ada 4 file yang diuji cobakan, yaitu :

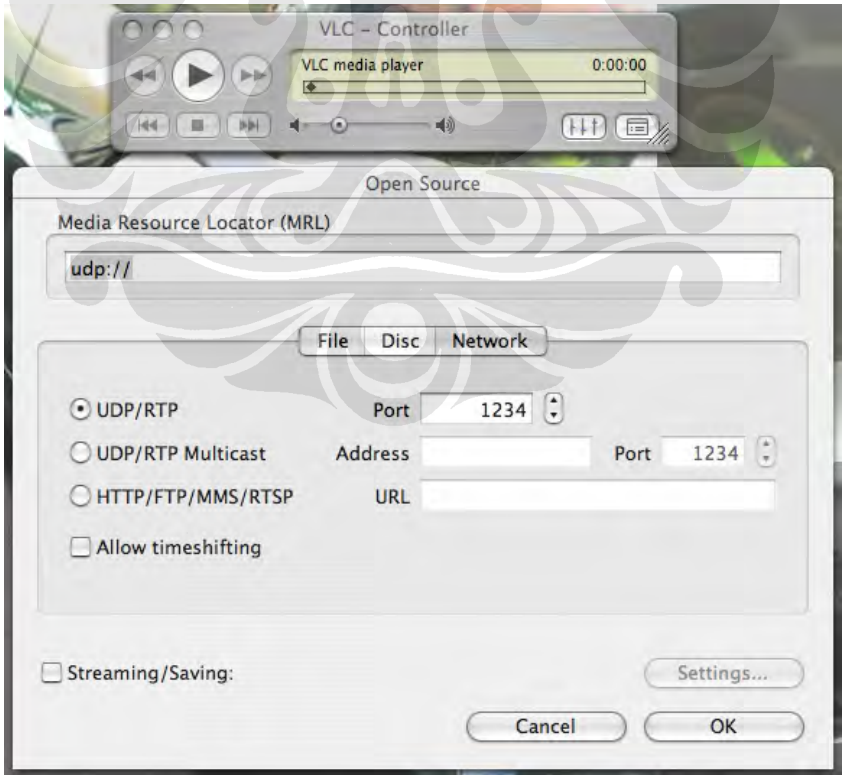
1. Avilong.avi berukuran 23,5 MB dengan durasi 2 menit 38 detik
2. TestVid.avi berukuran 1,7 MB dengan durasi 28 detik
3. TestVid.mpg berukuran 16,8 MB dengan durasi 28 detik
4. TestVid.mp4 berukuran 1,7 MB dengan durasi 28 detik

Semua file yang diuji cobakan mempunyai karakteristik audio dan video yang berbeda. Karakteristik untuk semua file dapat dilihat pada lampiran C.

Semua file akan diuji cobakan satu persatu pada ketiga konfigurasi yang dilakukan. Percobaan akan dilakukan pertama dengan mengatur tools VLC pada pihak *server* yang akan terlihat seperti Gambar 4.1 berikut :

Gambar 4.1 Proses *streaming* pada *host server*

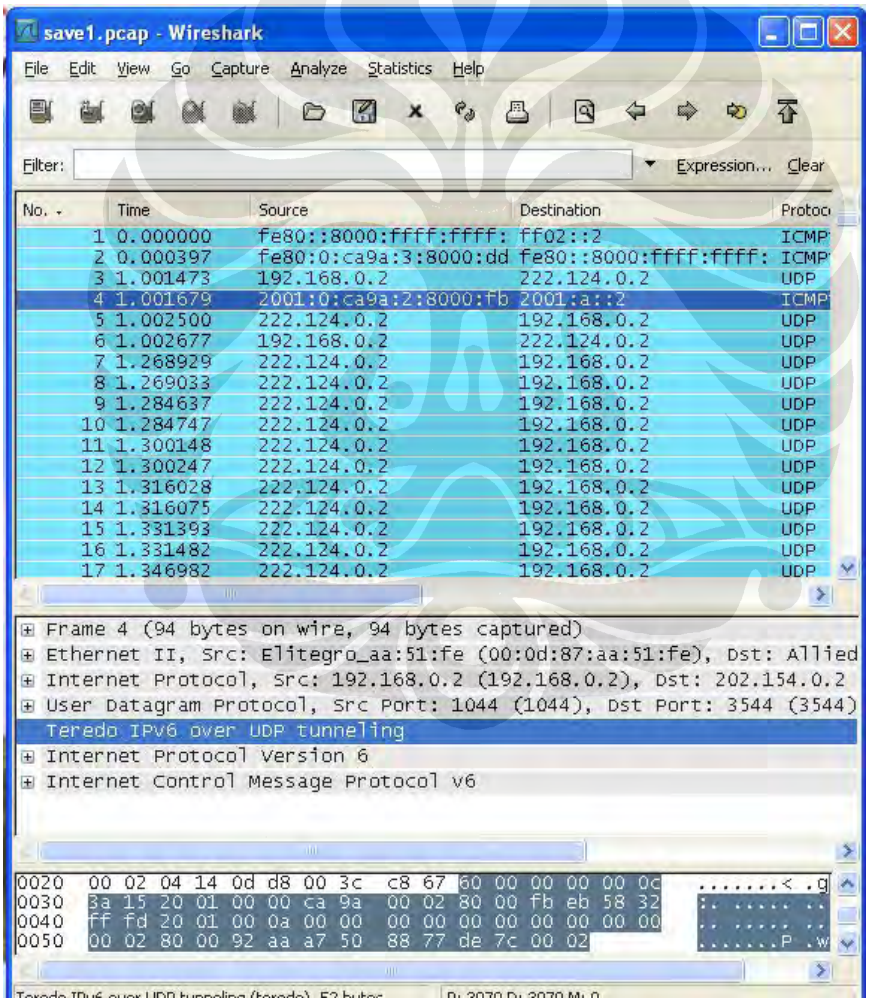
Setelah *tools* VLC pada pihak *server* dijalankan maka langkah berikutnya adalah menjalankan VLC pada pihak *client* yang dapat dilihat pada Gambar 4.2 di bawah ini :



Gambar 4.2 Proses *streaming* pada *host client*

Dapat dilihat pula pada Gambar 4.2 bahwa pada skripsi ini dipakai metode transmisi dengan UDP, hal ini dikarenakan UDP bersifat *connectionless* sehingga menghindari terjadinya delay yang berlebihan. Selain itu penggunaan UDP juga dimaksudkan untuk dapat melihat *packet loss* yang terjadi karena UDP tidak mendukung proses *retransmisi*.

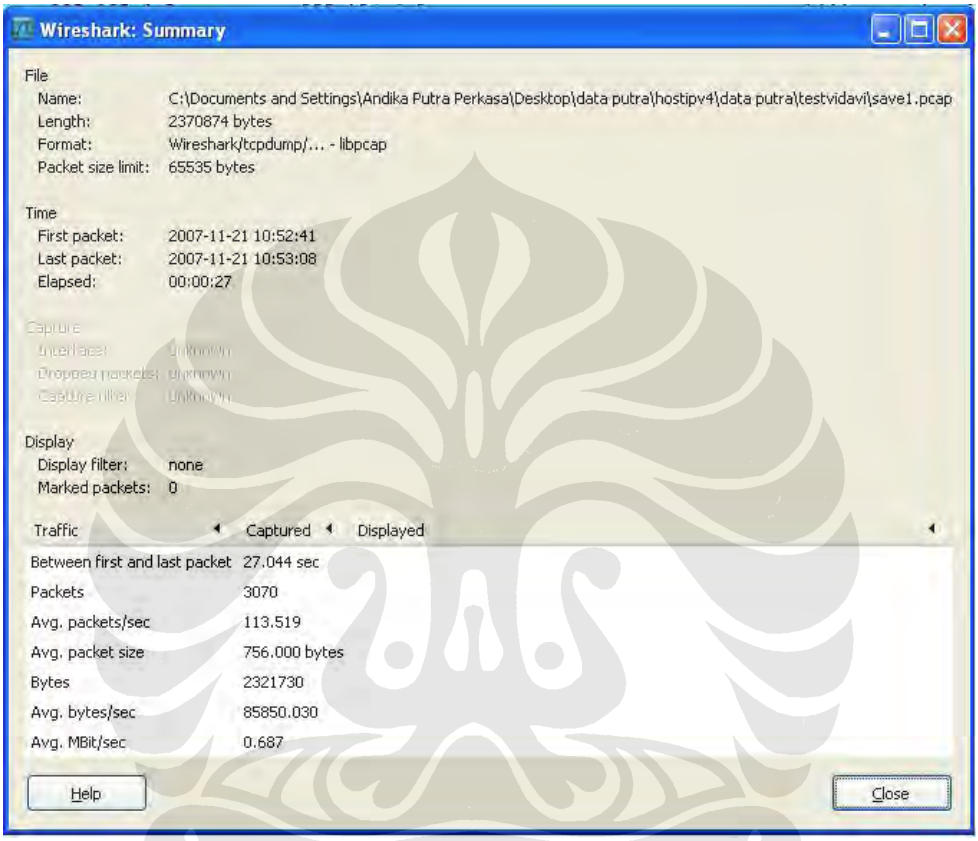
Setelah proses setting VLC selesai dilakukan, langkah selanjutnya adalah penangkapan atau peng-*capture*-an data yang akan dilakukan oleh *tools* Wireshark. Proses pengiriman video akan dilakukan sebanyak 10 kali untuk masing-masing file. Tampilan Wireshark ketika proses pengambilan data dilakukan dapat dilihat pada Gambar 4.3 di bawah ini :



Gambar 4.3 Tampilan *capturing* aplikasi wireshark

Dari gambar 4.3 terlihat bahwa *tunneling* menggunakan metode teredo telah berhasil dilakukan dengan sukses. Hal ini dibuktikan dengan adanya *reply*

dari *server*. Setelah proses pengambilan data ini selesai dilakukan, maka akan dapat dilihat *summary* dari data yang telah ditangkap. Tampilan *summary* dari wireshark dapat dilihat pada Gambar 4.4.



Gambar 4.4 Tampilan *summary* dari wireshark

Setelah semua data didapat maka akan dilakukan pengolahan data dengan meghitung rata-rata dan standar deviasi untuk setiap file pada masing-masing konfigurasi jaringan. Hasil pengolahan data akan berupa 3 parameter, antara lain *latency, packet loss*, dan *throughput*. *Latency* didapat dari jumlah waktu yang ditempuh dari pengiriman paket pertama sampai paket terakhir. Paket loss dihitung dari jumlah paket yang dikirim dikurangi jumlah paket yang diterima. Sedangkan *throughput* akan didapat dari data *transfer rate* selama proses *video streaming* berjalan.

Hasil pengolahan data berdasarkan pembagian konfigurasi dapat dilihat pada tabel 4.1-4.3 dibawah ini :

❖ NAT Ipv4

Tabel 4.1 Hasil pengolahan data pada jaringan NAT Ipv4

Konfigurasi NAT IPv4			
File	Total waktu (s)	Jumlah Paket	Throughput
Avilong.avi	157.957	18424	159798.1995
TestVid.avi	24.6452	1530	85050.9596
TestVid.mpg	24.4624	13022	728958.9464
TestVid.mp4	24.4344	1459	81772.2856

❖ Ipv 6

Tabel 4.2 Hasil pengolahan data pada jaringan Ipv6

Konfigurasi IPv6			
File	Total waktu (s)	Jumlah Paket	Throughput
Avilong.avi	157.4675	18428	162127.1098
TestVid.avi	24.6544	1530	86260.6669
TestVid.mpg	24.4555	13011	739504.6459
TestVid.mp4	24.4271	1458	82962.4859

❖ Metode Teredo

Tabel 4.3 Hasil pengolahan data pada metode Teredo

Konfigurasi TEREDO			
File	Total waktu (s)	Jumlah Paket	Throughput (bytes/s)
Avilong.avi	157.9008	36865	176810.928
TestVid.avi	27.06093	3071	85850.3493
TestVid.mpg	24.5248	26010	804920.6206
TestVid.mp4	35.2641	2920	63419.7179

Sedangkan hasil pengolahan data menurut file yang diuji cobakan dapat dilihat pada tabel 4.4-4.7 di bawah ini :

- TestVid.avi

Tabel 4.4 Hasil pengolahan data video streaming Testvid.avi

TestVid.avi			
Konfigurasi	Total waktu (s)	Jumlah Paket	Throughput (bytes/s)
NAT IPv4	24.6452	1530	85050.9596
IPv6	24.6544	1530	86260.6669
TEREDO	27.06093	3071	85850.3493

- TestVid.mpg

Tabel 4.5 Hasil pengolahan data video streaming Testvid.mpg

TestVid.mpg			
Konfigurasi	Total waktu (s)	Jumlah Paket	Throughput (bytes/s)
NAT IPv4	24.4624	13022	728958.9464
IPv6	24.4555	13011	739504.6459
TEREDO	24.5248	26010	804920.6206

- TestVid.mp4

Tabel 4.6 Hasil pengolahan data video streaming Testvid.mp4

TestVid.mp4			
Konfigurasi	Total waktu (s)	Jumlah Paket	Throughput (bytes/s)
NAT IPv4	24.4344	1459	81772.2856
IPv6	24.4271	1458	82962.4859
TEREDO	35.2641	2920	63419.7179

- Avilong.avi

Tabel 4.7 Hasil pengolahan data video streaming avilong.avi

Avilong.avi			
Konfigurasi	Total waktu (s)	Jumlah Paket	Throughput (bytes/s)
NAT IPv4	157.957	18424	159798.1995
IPv6	157.4675	18428	162127.1098
TEREDO	157.9008	36864	176810.928

4.2 ANALISA HASIL PERCOBAAN

4.2.1 Analisa *Latency*

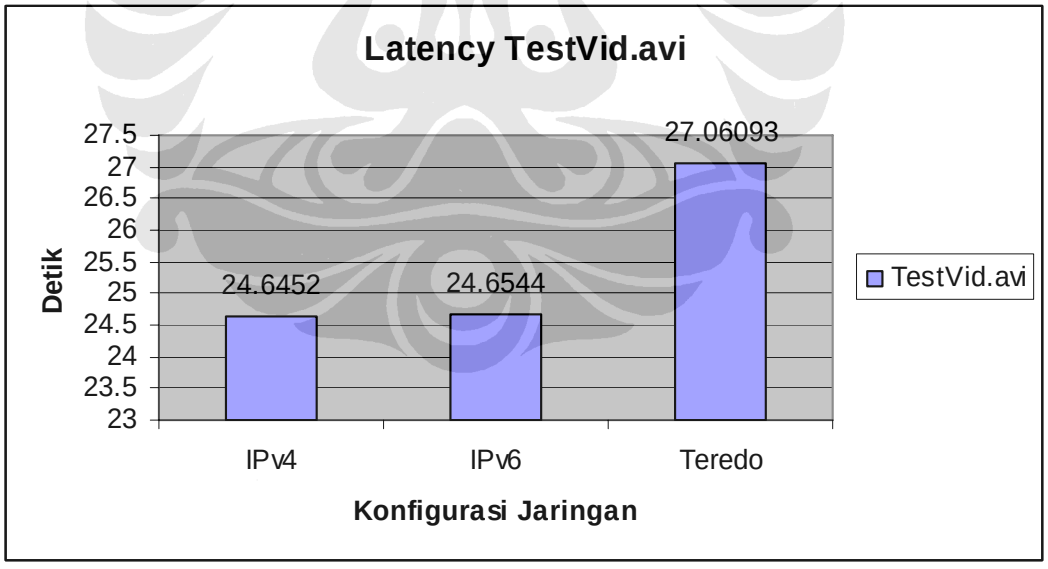
Latency adalah waktu yang diperlukan untuk memindahkan paket dari satu host ke host yang lain. *Latency* ini yang mempengaruhi *delay* pada *video streaming*. *Delay* terjadi ketika *latency* melebihi durasi video. Hal ini nantinya akan berpengaruh pada kualitas *video streaming*.

Data *latency* yang telah didapatkan dapat dilihat pada tabel 4.8 berikut ini:

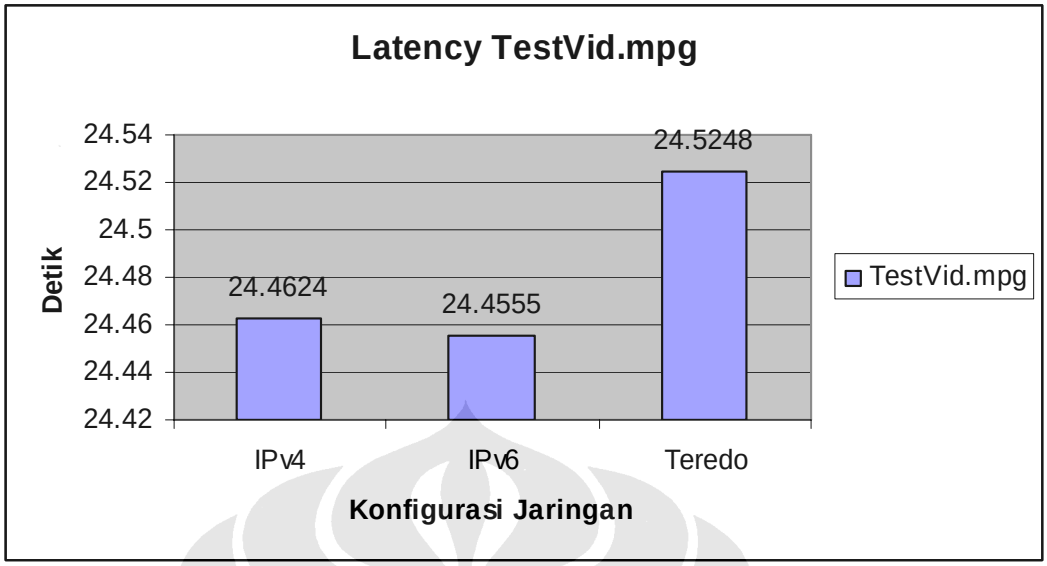
Tabel 4.8 Tabel perbandingan latency secara keseluruhan

Konfigurasi/File	IPv4	IPv6	Teredo
TestVid.avi	24.6452	24.6544	27.06093
TestVid.mpg	24.4624	24.4555	24.5248
TestVid.mp4	24.4344	24.4271	35.2641
Avilong.avi	157.957	157.4675	157.9008

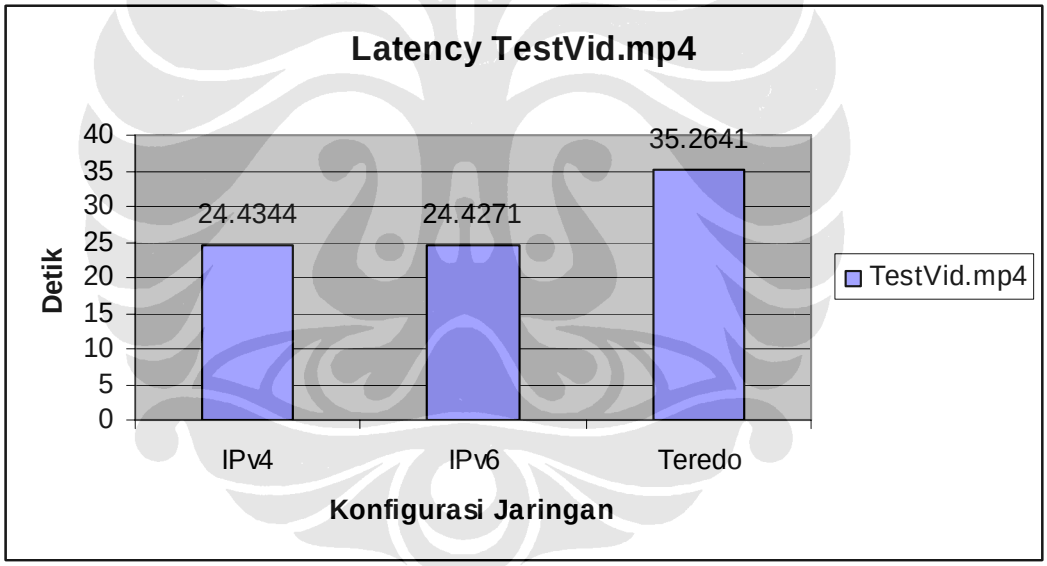
Dari data di atas dapat dibuat grafik perbandingan *latency* untuk setiap file *video streaming* yang diujikan. Grafik dapat dilihat pada Gambar 4.5-4.8 berikut ini:



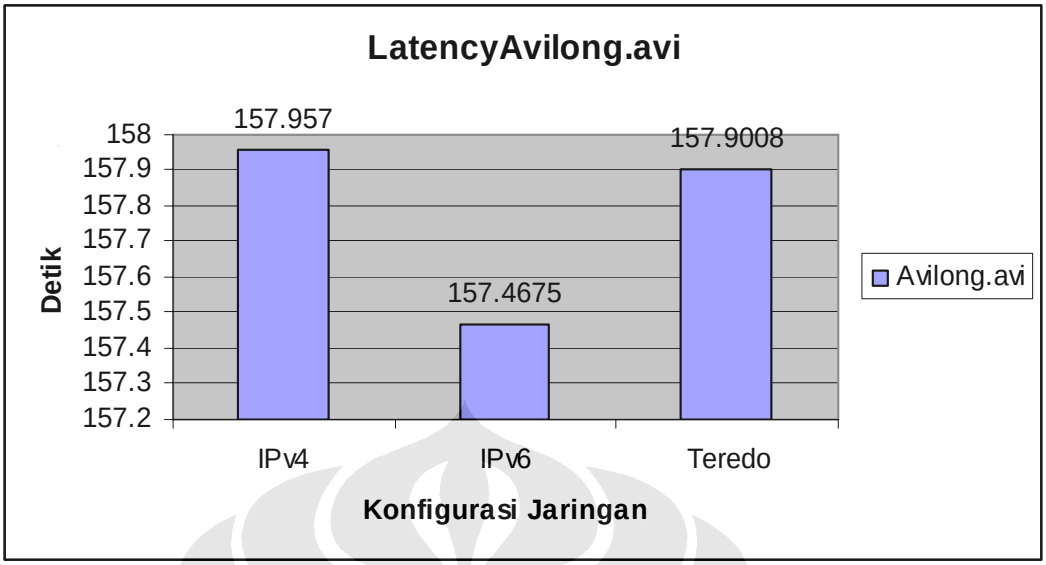
Gambar 4.5 Grafik latency pada pada TestVid.avi



Gambar 4.6 Grafik latency pada TestVid.mpg

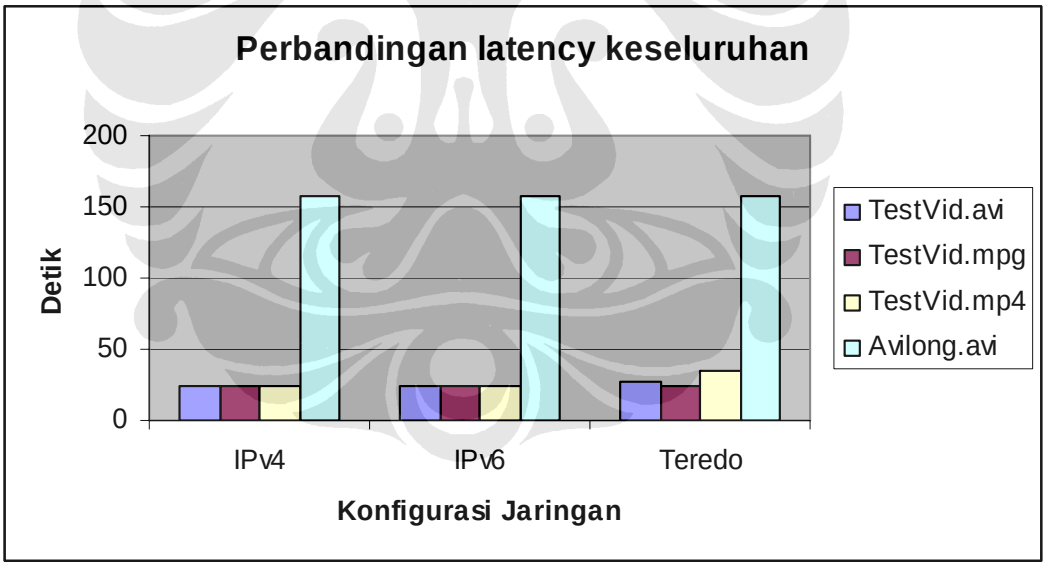


Gambar 4.7 Grafik latency pada TestVid.mp4



Gambar 4.8 Grafik latency pada avilong.avi

Pada Gambar 4.9 di bawah ini dapat dilihat *latency* untuk semua file dalam 3 konfigurasi yang dilakukan:



Gambar 4.9 Grafik perbandingan latency keseluruhan

Setelah dilakukan perhitungan, rata-rata *latency* untuk masing-masing konfigurasi jaringan adalah :

1. NAT Ipv4 : 57.87 detik
2. Ipv6 : 57.75 detik
3. Teredo : 61.18 detik

Data di atas menunjukkan bahwa *latency* paling besar dimiliki oleh konfigurasi Teredo. Mekanisme *tunneling* dari IPv4 yang berada di balik NAT ke IPv6 ini mempunyai *latency* sebesar 61.18 detik atau lebih besar 5,4% dibanding jaringan NAT IPv4 murni dan lebih besar 5,6% dibandingkan kan dengan jaringan IPv6. Hal ini dikarenakan pada mekanisme teredo banyak *fase* seperti telah disebutkan pada bab 2, diantaranya adalah pemberian alamat teredo dan pemisahan kembali oleh NAT. Akibat dari besarnya *latency* pada mekanisme teredo ini adalah tampilan audio visual yang kurang baik namun tetap dapat diterima pada sebuah proses *video streaming*.

Jaringan NAT IPv4 menempati peringkat berikutnya sebagai pemilik *latency* terbesar. Hal ini tentu saja dikarenakan pada sebuah jaringan Ipv4 tidak didukung terjadinya QoS. Selain itu pada NAT IPv4 sebuah host akan mempunyai IP yang *private,* sehingga untuk terhubung kepada jaringan luar harus diberikan IP public dari sebuah *server* NAT. Hal ini tentu saja ikut andil dalam memperbesar *latency* yang terjadi.

Jaringan IPv6 mempunyai *latency* yang paling kecil. Ini sesuai dengan karakteristik IPv6 yang sangat mendukung QoS khususnya dalam hal aplikasi real-time seperti *video streaming*. *Traffic class* and *flow label* yang terintegrasi di dalam IPv6 akan mengatur supaya proses *real-time* dapat diteruskan tanpa hambatan.

Dari data *latency* yang dihasilkan juga terlihat bahwa *latency* yang terjadi pada jaringan NAT IPv4 dan jaringan IPv6 mempunyai waktu lebih kecil atau dengan kata lain lebih cepat dibandingkan durasi file itu sendiri. Hal ini tentu saja baik, mengingat bahwa proses *video streaming* merupakan salah satu aplikasi dari sistem *real-time*. Akan tetapi, pada jaringan NAT IPv4 sesungguhnya tidak ada

jaminan QoS untuk proses real-time itu sendiri berbeda dengan IPv6 yang mempunyai jaminan proses real-time yang baik.

Sedangkan untuk mekanisme teredo yang diuji pada penelitian ini menampilkan hasil yang cukup menggembirakan, walaupun proses yang dilalui cukup panjang, file yang diuji cobakan tetap dapat mempunyai latency yang lebih kecil dibandingkan durasi file. Namun, hal ini tidak berlaku untuk file TestVid.mp4, pada file ini latency yang terjadi terlalu jauh dari batas normal yaitu kurang lebih tujuh detik lebih besar dari durasi file. Akibat yang dihasilkan adalah gambar yang patah-patah dan suara yang keluar terlebih dahulu dibandingkan gambar atau scene-nya. Hal ini kemungkinan besar disebabkan minimnya memory dari teredo server dan teredo relay ketika proses streaming TestVid.mp4 dilakukan.

4.2.2 Analisa Throughput

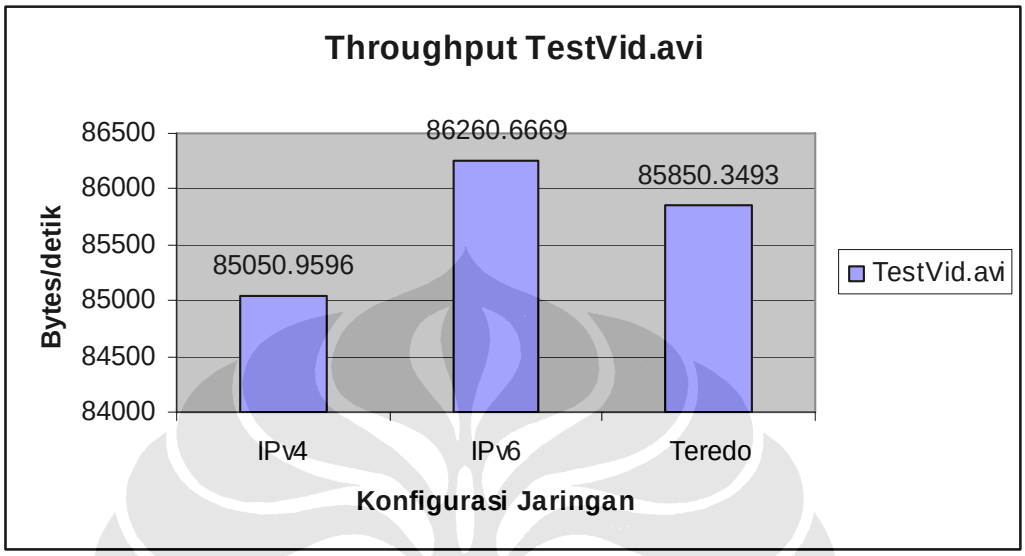
Throughput adalah sejumlah informasi yang berhasil dikirim oleh suatu jaringan selama interval waktu tertentu. Throughput merupakan *bandwidth* aktual yang terukur pada suatu waktu tertentu untuk kondisi tertentu.

Data throughput yang telah didapatkan dapat dilihat pada tabel 4.9 berikut ini :

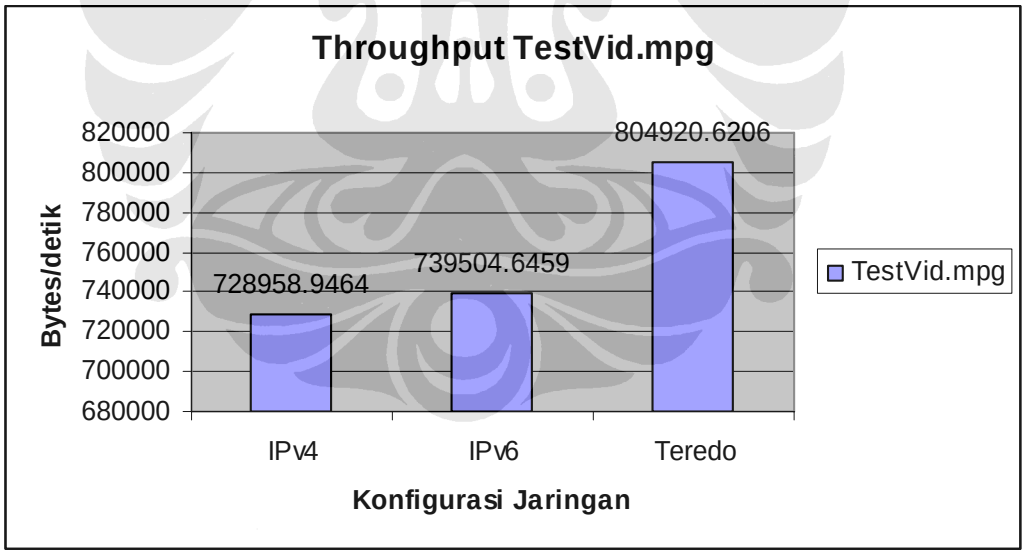
Tabel 4.9 Throughput pada semua konfigurasi jaringan

Konfigurasi/File	IPv4	IPv6	Teredo
TestVid.avi	85050.9596	86260.6669	85850.3493
TestVid.mpg	728958.9464	739504.6459	804920.6206
TestVid.mp4	81772.2856	82962.4859	63419.7179
Avilong.avi	159798.1995	162127.1098	176810.928

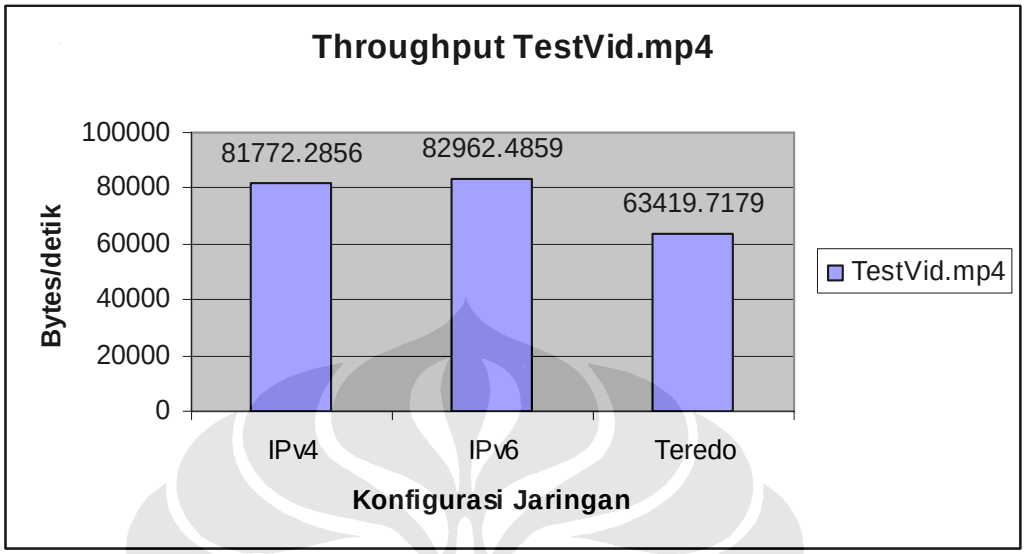
Dari data di atas dapat dibuat grafik perbandingan throughput untuk setiap file video streaming, grafik untuk setiap file yang diuji cobakan dapat dilihat pada Gambar 4.10-4.13 berikut ini :



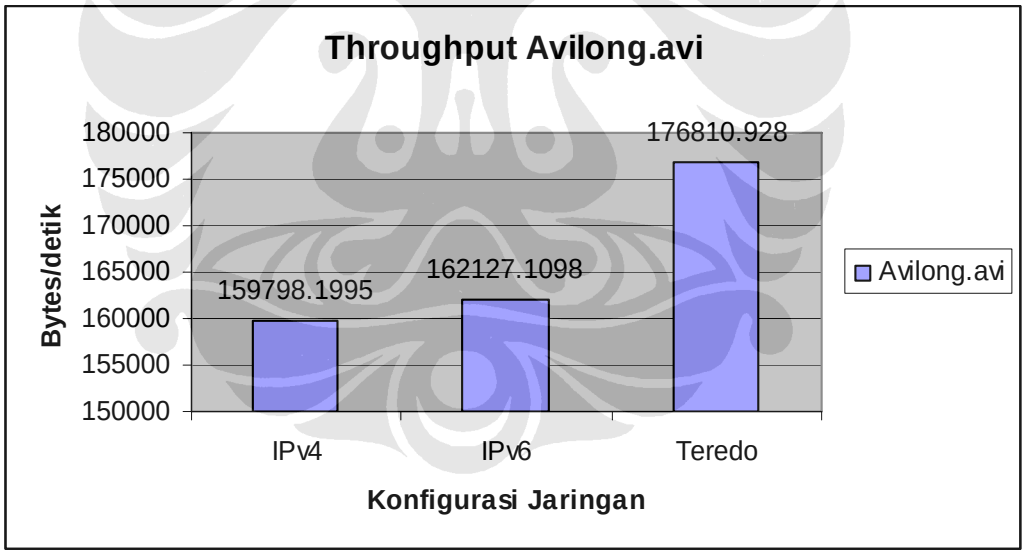
Gambar 4.10 Grafik *throughput* pada TestVid.avi



Gambar 4.11 Grafik *throughput* pada TestVid.mpg

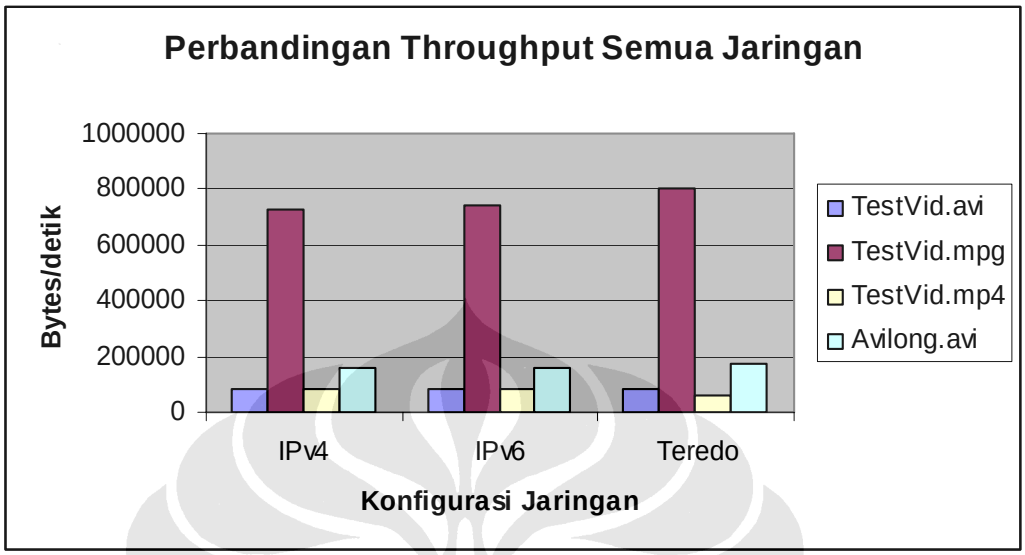


Gambar 4.12 Grafik *throughput* pada TestVid.mp4



Gambar 4.13 Grafik *throughput* pada avilong.avi

Setelah digabungkan maka perbandingan *throughput* semua konfigurasi jaringan akan terlihat seperti pada Gambar 4.14 berikut :



Gambar 4.14 Grafik perbandingan throughput keseluruhan

Jika dihitung rata-rata dari tiap konfigurasi yang diuji coba maka akan dihasilkan :

1. IPv4 : 263895.0978 Bytes/detik
2. IPv6 : 267713.7271 Bytes/detik
3. Teredo : 282750.404 Bytes/detik

Jika dilihat dari rata-rata yang dihasilkan dapat disimpulkan kalau teredo mempunyai *throughput* yang paling besar, ini sangat baik mengingat mekanisme teredo adalah sebuah mekanisme *tunneling*. Setelah ditelaah lebih lanjut ternyata nilai besar ini dihasilkan oleh dua file yaitu Avilong.avi dan TestVid.mpg, nilai throughput yang dihasilkan ketika file tersebut di-stream-kan adalah 176810,928 dan 804920,6206. Inilah yang membuat rata-rata dari *throughput* teredo naik pesat. Namun, tetap harus dipertimbangkan jumlah paket yang terjadi pada sebuah pengiriman paket teredo. Pada sebuah pengiriman paket teredo jumlah paket yang dikirim lebih banyak, sehingga besarnya throughput tetap harus dibagi banyaknya jumlah paket yang dilewatkan.

Untuk menganalisa mengapa dapat terjadi kenaikan yang cukup pesat pada *throuhput* teredo ada baiknya terlebih dahulu dilihat kembali proses pada UDP

yang dipakai pada percobaan ini. Pada *user data protocol* atau yang lebih dikenal dengan UDP data yang akan dikirimkan diambil dan ditambahkan dengan informasi *source port, destination port, length*, dan *checksum port* lalu melewatkannya pada *network layer* dan layer inilah yang akan mengenkapsulasi data tersebut dan mengirimkannya pada *destination point*. Yang harus dicatat adalah tidak terjadinya proses *handshaking* di dalam UDP. Dengan kata lain ketika kondisi jaringan sedang dalam keadaan *idle* atau tidak ada aktifitas selain pengiriman data yang dilakukan maka throughput yang terjadi pun akan semakin besar. Hal ini kemungkinan besar yang terjadi ketika pengambilan data Avilong.avi dan TestVid.mpg dilakukan. Selain itu file yang cukup besar dari keduanya juga termasuk di dalam penyebab rata-rata throughput teredo menjadi besar. Kedua file tersebut berukuran 23,6MB dan 16,8MB, hal ini menyebabkan UDP menyediakan *bandwidth* yang besar supaya kedua file tersebut dapat tiba pada waktunya.

Selain itu pembagian paket dalam teredo juga mempunyai andil besar dalam menghasilkan *throughput* yang besar. Dalam mekanisme teredo, paket yang dihasilkan lebih kecil sehingga menyebabkan jumlah paket yang terjadi semakin banyak. Karena file avilong.avi dan TestVid.mpg mempunyai ukuran file yang besar maka paket yang terjadi pun akan semakin banyak, sementara pada proses *video streaming* waktu adalah hal yang paling krusial. Oleh sebab itu, ketika paket yang berjumlah sangat besar harus dikirimkan dalam waktu yang terbatas mendorong terjadinya *throughput* yang besar pula.

Berdasarkan dasar teori, *throughput* dari IPv6 lebih baik daripada IPv4. Hal ini dikarenakan IPv6 memiliki *header* dengan *field* lebih sederhana dibandingkan IPv4 sehingga di dalam proses pengiriman paketnya menjadi lebih sederhana. Teori telah dapat dibuktikan dalam percobaan karena data yang didapat menunjukkan bahwa jaringan IPv6 lebih baik 1,43% dibandingkan IPv4.

Dari perhitungan yang dilakukan, metode teredo mempunyai *throughput* lebih besar 5,3% dibandingkan dengan jaringan IPv6 dan lebih besar 6,66% dibandingkan dengan jaringan NAT IPv4. Namun, seperti telah dijelaskan diatas, jumlah paket pada metode teredo kurang lebih dua kali lipat paket pada jaringan IPv6 dan jaringan NAT IPv4.

4.2.3 Analisa *Packet Loss*

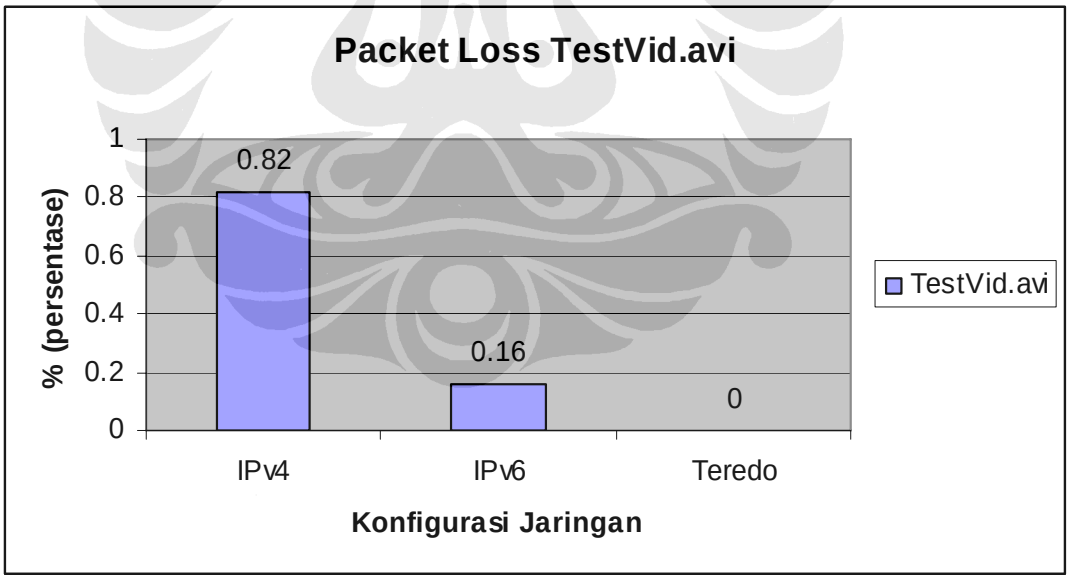
Packet loss merupakan jumlah paket yang hilang dalam proses pengiriman data dari satu node ke node yang lain. Perhitungan paket yang hilang dilakukan dengan mencari selisih antara paket yang dikirimkan dari *server* dengan paket yang diterima oleh *host client*.

Setelah dilakukan pengolahan data, hasil yang telah didapat bisa dilihat pada tabel 4.10 dibawah ini :

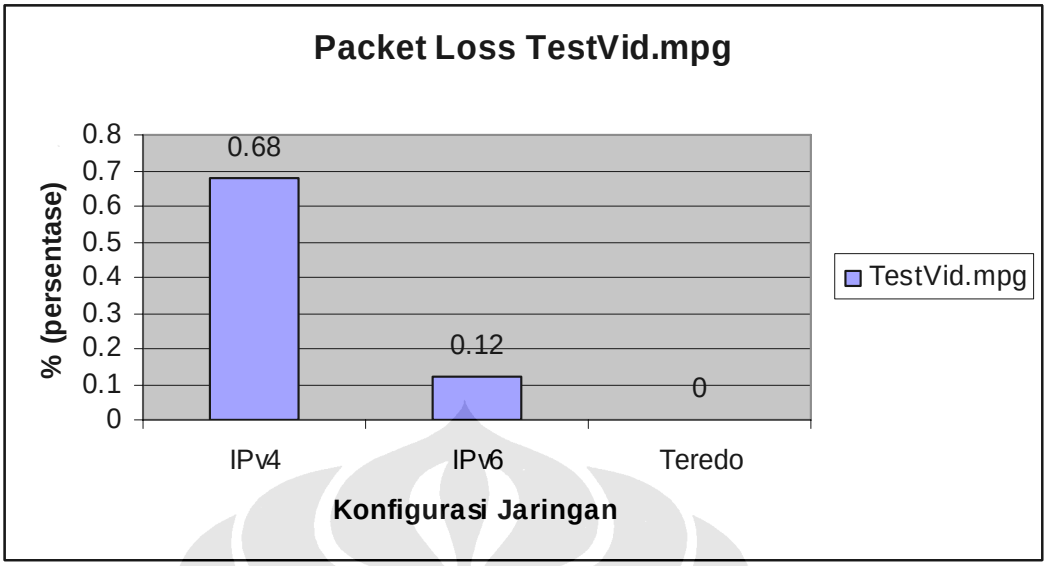
Konfigurasi/Nama file	IPv4(%)	IPv6(%)	Teredo(%)
TestVid.avi	0.82	0.16	(unknown)
TestVid.mpg	0.68	0.12	(unknown)
TestVid.mp4	1.08	0.2	(unknown)
Avilong.avi	0.56	0.1	(unknown)

Tabel 4.10 Tabel perbandingan *packet loss* secara keseluruhan

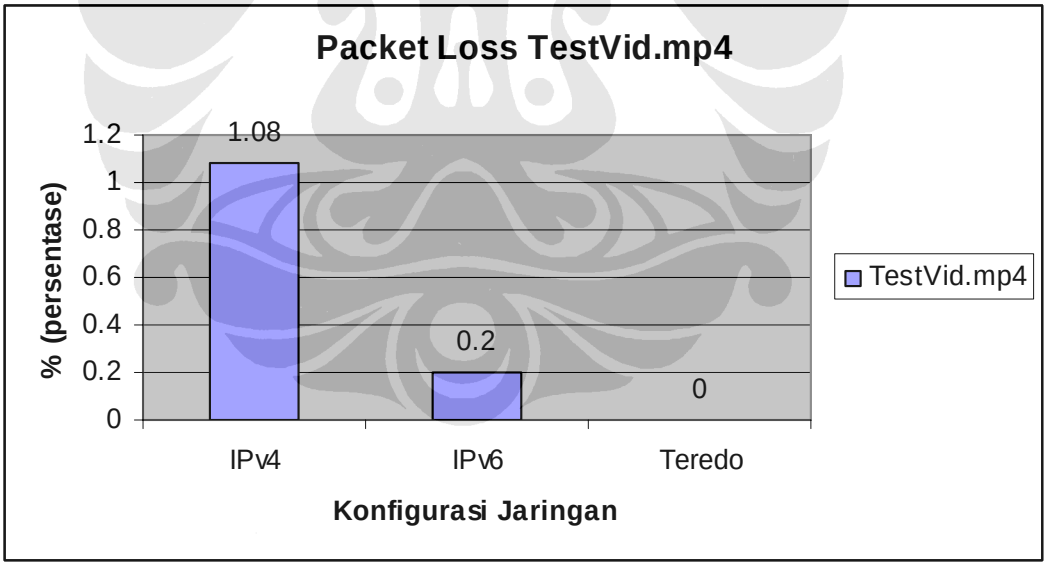
Dari perolehan data akhir diatas maka dapat dibuat grafik perbandingan *packet loss* untuk setiap file *video streaming*. Grafik perbandingan pada file di setiap konfigurasi jaringan dapat dilihat pada Gambar 4.15-4.18 berikut :



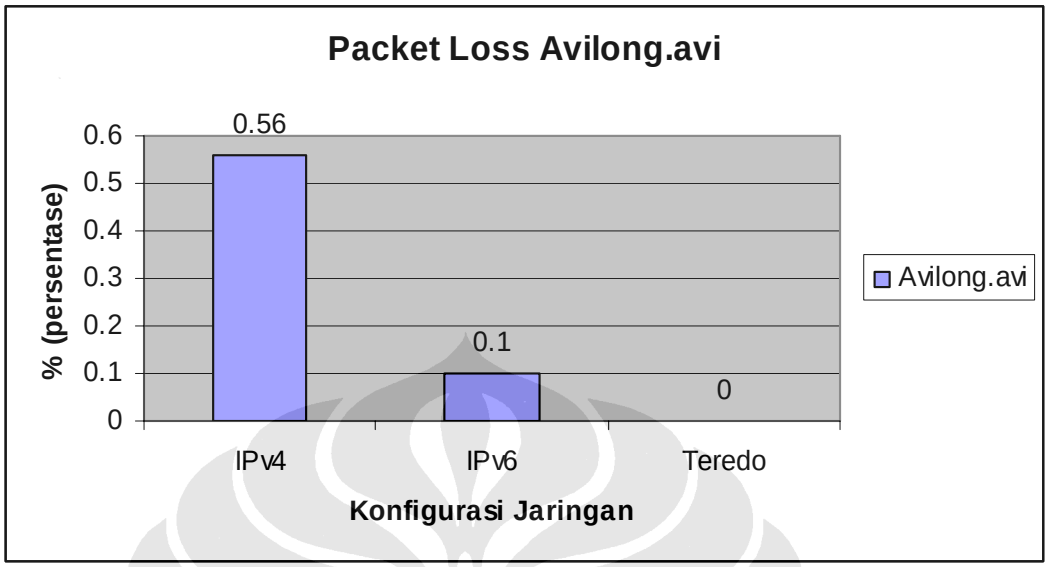
Gambar 4.15 Grafik *packet loss* pada TestVid.avi



Gambar 4.16 Grafik *packet loss* pada TestVid.mpg

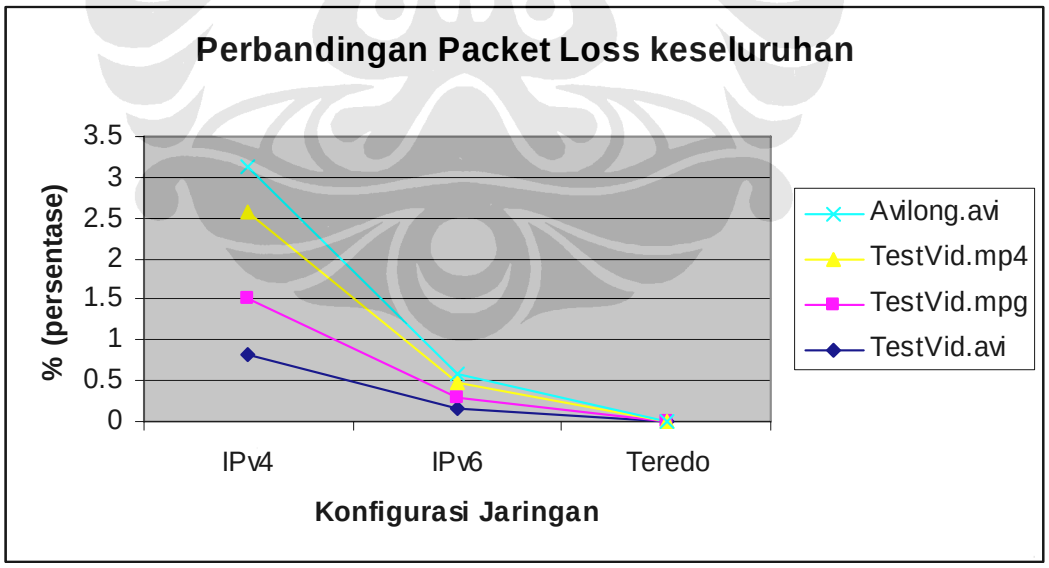


Gambar 4.17 Grafik *packet loss* pada TestVid.mp4



Gambar 4.18 Grafik *packet loss* pada Avilong.avi

Sedangkan untuk perbandingan keseluruhan dapat dilihat pada Gambar 4.19 di bawah ini :



Gambar 4.19 Grafik perbandingan *packet loss* keseluruhan

Dari data-data diatas ditemukan keanehan dengan tidak diketahuinya paket yang hilang pada mekanisme teredo, dari lampiran data dapat dilihat bahwa pada mekanisme teredo, paket yang dikirimkan dari *server* ketika diterima oleh *client* tidak berkurang bahkan cenderung bertambah. Hal ini diakibatkan pembagian paket pada teredo yg mempunyai ukuran kecil sehingga jumlahnya menjadi sangat banyak. Banyaknya jumlah paket yang terjadi juga didukung dengan adanya pemberian *header* dari tiap-tiap paket oleh teredo *relay*, hal ini tentu saja ikut menambah jumlah paket menjadi lebih banyak lagi. Seharusnya, dalam jaringan sangat mudah terjadi *congestion* yang diakibatkan oleh adanya kondisi *bottleneck*. *Bottleneck* ini sendiri disebabkan spesifikasi antar-router yang berbeda sehingga sejumlah paket yang sampai ke sisi *client* telah melewati batas waktunya (delay bound). Penggunaan protokol UDP pun dimaksudkan untuk mempermudah penghitungan paket yang hilang karena protokol ini merupakan *connectionless transport protocol* atau dengan kata lain protokol ini tidak akan mengirimkan kembali paket yang hilang.

Packet loss juga pada dasarnya dihitung dari besarnya sebuah paket yang bisa dilewati dalam sebuah jaringan. Ternyata data yang dihasilkan memang cocok dengan besar *packet size* yang dikehendaki oleh jaringan, yaitu dengan setiap paket memiliki besar 720bytes per paket [lampiran B]. Hal inilah pada dasarnya yang membuat *packet loss* pada teredo menjadi 0 atau tidak ada.

Tidak terdeteksinya *packet loss* pada teredo tentu saja membuat tidak dapatnya dibuat perbandingan kinerja teredo dibandingkan dengan IPv6 dan teredo dibandingkan dengan IPv4. Namun, analisa tetap dapat dilakukan dengan berdasarkan pada tampilan video yang terjadi.

Pada proses *streaming* video dengan mekanisme teredo tampilan dari sebuah file video cukup baik, hanya saja sering terjadi skip frame dan terkadang patah-patah. Dari tampilan yang terjadi dapat dikatakan bahwa terjadi *packet loss* yang cukup signifikan pada mekanisme ini sebab, jika tidak terjadi *packet loss* seharusnya video akan terlihat jernih dan tidak terjadi skip frame ataupun patah-patah.

Jika dibandingkan dengan pemutaran video pada jaringan IPv6 kinerja mekanismed teredo jauh tertinggal karena pada jaringan IPv6 video berjalan

sangat halus dan baik. Suara dan gambar yang dihasilkan oleh jaringan IPv6 sangat jenih berbeda dengan mekanisme teredo yang masih patah-patah dan terkadang suara mendahului gambar.

Pada penelitian ini, hasil yang ditunjukkan oleh mekanisme teredo lebih dekat dengan hasil yang ditunjukkan oleh jaringan NAT IPv4. Pada jaringan NAT IPv4 video tampilan video juga mengalami *lagging* dan terkadang suara mendahului gambar yang ada.

Tingginya kualitas yang ditampilkan IPv6 dikarenakan format *header* dari IPv6 adalah yang paling sederhana dibandingkan konfigurasi jaringan yang lain. Hal ini yang membuat pada IPv6 jumlah *packet loss* yang terjadi sangat kecil. Lain halnya dengan IPv4 yang mempunyai format header yang lebih kompleks sehingga kemungkinan terjadinya *packet loss* pun lebih besar.

Dari hasil pengolahan data bisa dikatakan bahwa untuk parameter *packet loss* metodologi yang dilakukan tidak berhasil. Untuk kedepannya dapat digunakan *tools* yang sudah terintegrasi dalam aplikasi VLC. Dimana pada tools ini dapat terlihat secara jelas parameter-parameter dalam video streaming.

Sebagai rekomendasi di masa depan pula, disarankan untuk lebih mengamati aspek-aspek dalam sebuah video seperti sampling rate, frame rate, konvergensi audio, dan lain-lain sehingga hasil yang didapatkan akan lebih maksimal.

4.2.4 Analisa Tampilan *Visual* dan *Audio*

Pada proses *streaming* hasil yang dicapai oleh jaringan IPv6 juga didukung oleh tampilan *audio visual*-nya. Pada pemutaran file yang dilakukan IPv6 menunjukkan performa yang maksimum, hal ini dapat dilihat dari tingkat kejernihan gambar dan suara yang dihasilkan ketika file sedang ditransmisikan.

Pada jaringan NAT, hasil yang ditunjukkan juga sesuai dengan data yang ada. Hal ini diperkuat tampilan gambar dan suara yang tidak maksimum, dimana sering terjadi *skip frame* dan suara yang mendahului gambar. Akhir dari sebuah file yang diputar pun seringkali tidak terlihat karena waktu/durasi file sudah terpenuhi sedangkan file masih ditransmisikan.

Keadaan pada jaringan teredo tidak jauh berbeda dengan yang terjadi pada jaringan NAT IPv4 , dimana seringkali terjadi *skip frame* dan suara yang sudah keluar mendahului gambar.

Data selengkapnya dari percobaan yang dilakukan dapat dilihat pada lampiran A.

BAB IV
KESIMPULAN

1. Dari hasil uji coba didapatkan *latency* yang dihasilkan mekanisme teredo lebih besar 5,4% dibandingkan NAT IPv4 dan lebih besar 5,6% dibandingkan IPv6, dengan kata lain pada mekanisme teredo pengiriman video akan berjalan lebih lambat dibanding dua konfigurasi lainnya.
2. Dari pengujian *throughput*, didapatkan hasil throughput teredo lebih besar ketika file yang ditransimiskan berukuran besar pula. *Throughput* yang didapat teredo lebih besar 5,3% dibandingkan dengan jaringan IPv6 dan dan lebih besar 6,66% dibandingkan dengan jaringan NAT IPv4, dengan kata lain untuk file berukuran besar, teredo mempunyai *bandwidth* yang cukup besar dibandingkan kedua konfigurasi lainnya.
3. *Packet loss* yang dialami mekanisme teredo tidak dapat dideteksi diakibatkan metodologi yang kurang tepat. Diharapkan kedepannya dapat dilakukan percobaan kembali dengan metodologi yang lebih tepat.
4. Penggunaan metode teredo dapat menjadi solusi untuk transmisi *video streaming* dari IPv6 ke IPv4.
5. Hasil *audio* dan *visual* yang ditunjukkan oleh mekanisme teredo tidak jauh berbeda dengan hasil yang ditunjukkan oleh jaringan NAT IPv4.

DAFTAR ACUAN

[1] D. Wu, Y.T.Hou, Y.G Zhang, J.M. Peha, dan W. Zhu, "Streaming Video Over Internet: Approaches and Directions. "*IEEE Transactions On Circuits and Systems for Video Technology*, Maret 2001.

[2] Streaming Video FAQ, www.finfacts.com/tv/ftv/internet_video_faq.pdf, (terakhir diakses: 21 November 2007).

[3] Video Lan Center setting, www.videoLan.org, (terakhir diakses: 24 November 2007).

[4] Taufan, Riza, "Teori dan Implementasi IPv6", Elex Media Komputindo, Desember 2002.

[5] Sugeng, Winarno, "Jaringan Komputer dengan TCP/IP", Informatika, Agustus 2006.

Lampiran A

Data Hasil Percobaan

A.1 DATA IPV4

TestVid.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.642	1530	85060.65
2	24.658	1530	85006.563
3	24.642	1530	85060.615
4	24.642	1533	85071.853
5	24.642	1530	85060.601
6	24.658	1530	85006.86
7	24.642	1530	85060.582
8	24.642	1530	85060.62
9	24.642	1530	85060.812
10	24.642	1530	85060.44
Rata-Rata	24.6452	1530.3	85050.9596

TestVid.mpg			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.467	13020	729044.74
2	24.44	13007	729111.432
3	24.467	13030	729045
4	24.449	13061	729110.322
5	24.439	13002	728813.238
6	24.502	13032	728675.697
7	24.443	13005	728900.079
8	24.439	13001	728801.463
9	24.445	13009	729108.462
10	24.533	13054	728979.031
Rata-Rata	24.4624	13022.1	728958.9464

TestVid.mp4			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.471	1460	81739.03
2	24.424	1458	81783.771
3	24.439	1459	81787.494
4	24.439	1458	81731.353
5	24.424	1458	81783.53
6	24.451	1460	81752.38
7	24.424	1459	81794.016
8	24.424	1458	81783.731
9	24.424	1458	81783.818
10	24.424	1458	81783.733
Rata-Rata	24.4344	1458.6	81772.2856

Avilong.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	157.965	18425	159796.002
2	157.948	18424	159805.092
3	157.965	18425	159789.089
4	157.95	18424	159803.316
5	157.949	18423	159794.908
6	157.965	18425	159795.094
7	157.948	18424	159803.244
8	157.965	18425	159796.088
9	157.95	18423	159804.3
10	157.965	18425	159794.862
Rata-Rata	157.957	18424.3	159798.1995

A.2 DATA IPV6

TestVid.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.64	1530	86312.383
2	24.64	1530	86312.488
3	24.658	1530	86247.721
4	24.658	1530	86247.721
5	24.658	1530	86247.745
6	24.658	1530	86247.74
7	24.658	1530	86247.682
8	24.658	1530	86247.737
9	24.658	1530	86247.73
10	24.658	1530	86247.722
Rata-Rata	24.6544	1530	86260.6669

TestVid.mpg			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.43	13000	739654.023
2	24.455	13007	739309.52
3	24.486	13031	739681.644
4	24.439	13002	739499.139
5	24.455	13007	739310.94
6	24.439	13001	739441.419
7	24.439	13002	739498.54
8	24.439	13002	739498.237
9	24.534	13055	739654.595
10	24	13002	739498.402
Rata-Rata	24.4555	13010.9	739504.6459

TestVid.mp4			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.424	1458	82876.04
2	24.424	1458	82977.593
3	24.455	1459	82928.066
4	24.424	1458	82977.379
5	24.424	1458	82977.556
6	24.424	1458	82977.613
7	24.424	1458	82977.586
8	24.424	1458	82977.558
9	24.424	1458	82977.68
10	24.424	1458	82977.788
Rata-Rata	24.4271	1458.1	82962.4859

Avilong.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	157.941	18422	162127.056
2	157.966	18425	162121.491
3	157.966	18425	162121.501
4	157.992	18428	162127.689
5	157.981	18442	162141.225
6	152.966	18441	162134.881
7	157.966	18425	162121.489
8	157.965	18425	162121.501
9	157.966	18425	162132.485
10	157.966	18425	162121.78
Rata-Rata	157.4675	18428.3	162127.1098

A.3 METODE TEREDO

TestVid.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	27.044	3070	85850.03
2	27.069	3068	85850.35
3	27.058	3070	85851.06
4	27.066	3074	85849.02
5	27.048	3069	85850.43
6	27.057	3068	85851.33
7	27.069	3070	85850.09
8	27.0693	3081	85850.35
9	27.064	3067	85850.8
10	27.065	3070	85850.033
Rata-Rata	27.06093	3070.7	85850.3493

TestVid.mpg			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	24.429	26005	806826.626
2	24.89	25996	791561.692
3	24.486	26047	806242.009
4	24.488	26007	806232.08
5	24.489	26005	806242.831
6	24.48	26008	806118.562
7	24.588	26014	806102.78
8	24.488	26007	806240.82
9	24.424	26005	806812.462
10	24.486	26005	806826.344
Rata-Rata	24.5248	26009.9	804920.6206

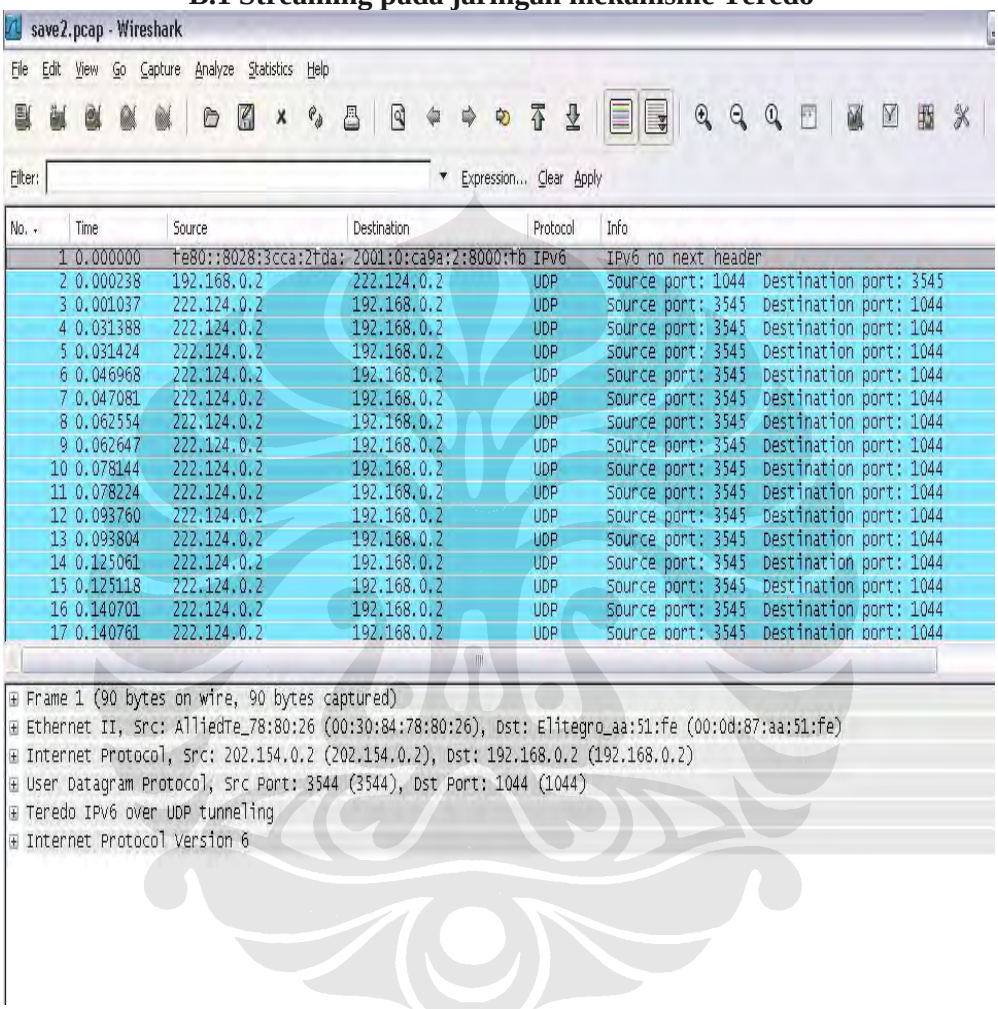
TestVid.mp4			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	34.857	2919	63419.522
2	35.127	2919	63420.413
3	35.124	2921	63419.675
4	35.443	2918	63419.433
5	35.213	2922	63419.523
6	35.438	2919	63419.3
7	35.319	2919	63419.45
8	35.544	2918	63420.35
9	35.233	2922	63419.633
10	35.343	2919	63419.88
Rata-Rata	35.2641	2919.6	63419.7179

Avilong.avi			
Data ke	Total waktu (detik)	Paket yang diterima	Throughput((bytes/s)
1	157.965	36859	176810.129
2	157.95	36872	176820.419
3	157.95	36847	176755.686
4	157.685	36855	176823.432
5	157.776	36879	176818.692
6	157.9	36877	176814.3
7	157.97	36863	176816.972
8	157.89	36878	176820.23
9	157.96	36860	176812.56
10	157.962	36858	176816.86
Rata-Rata	157.9008	36864.8	176810.928

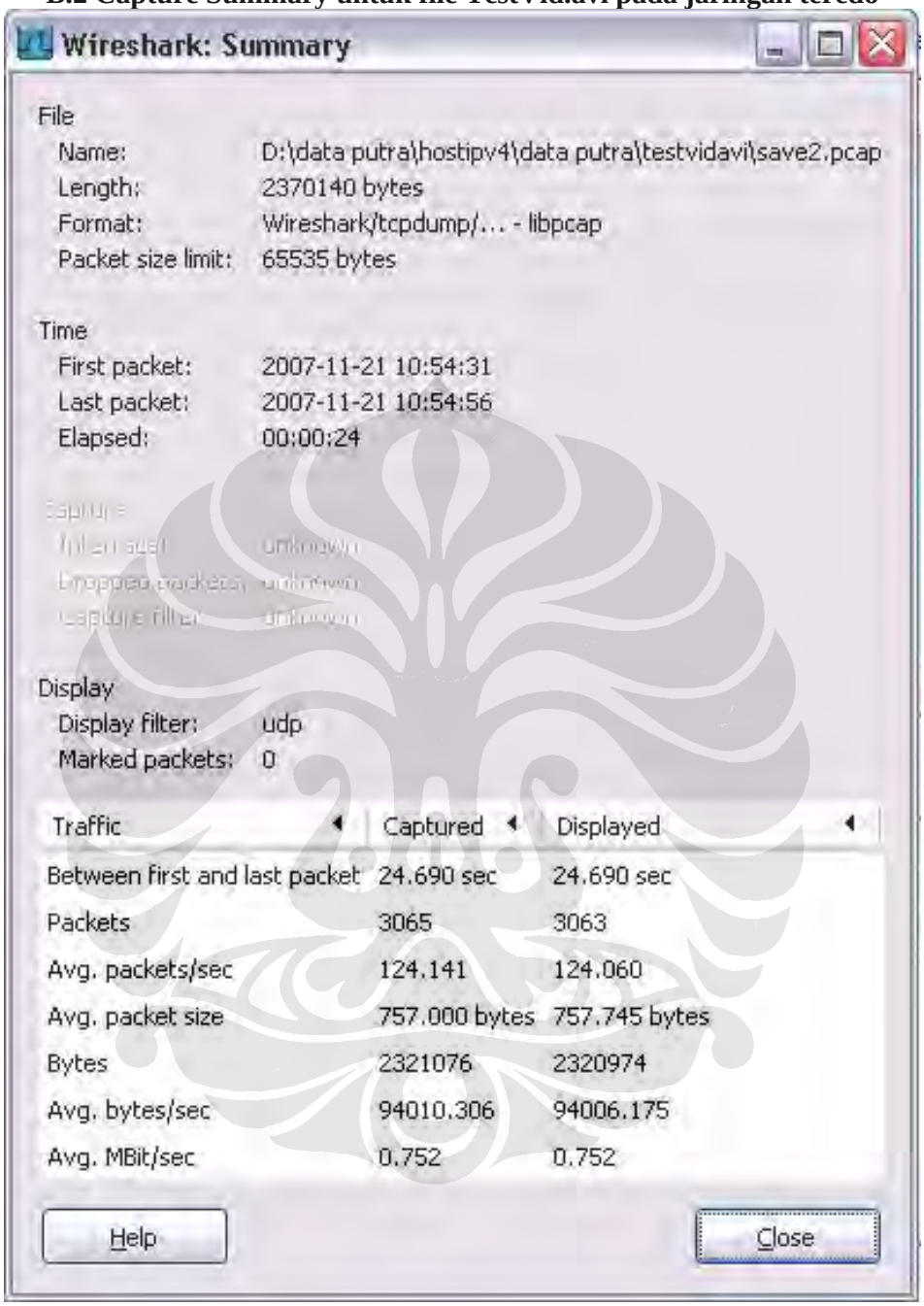
LAMPIRAN B

Hasil *Capture Wireshark*

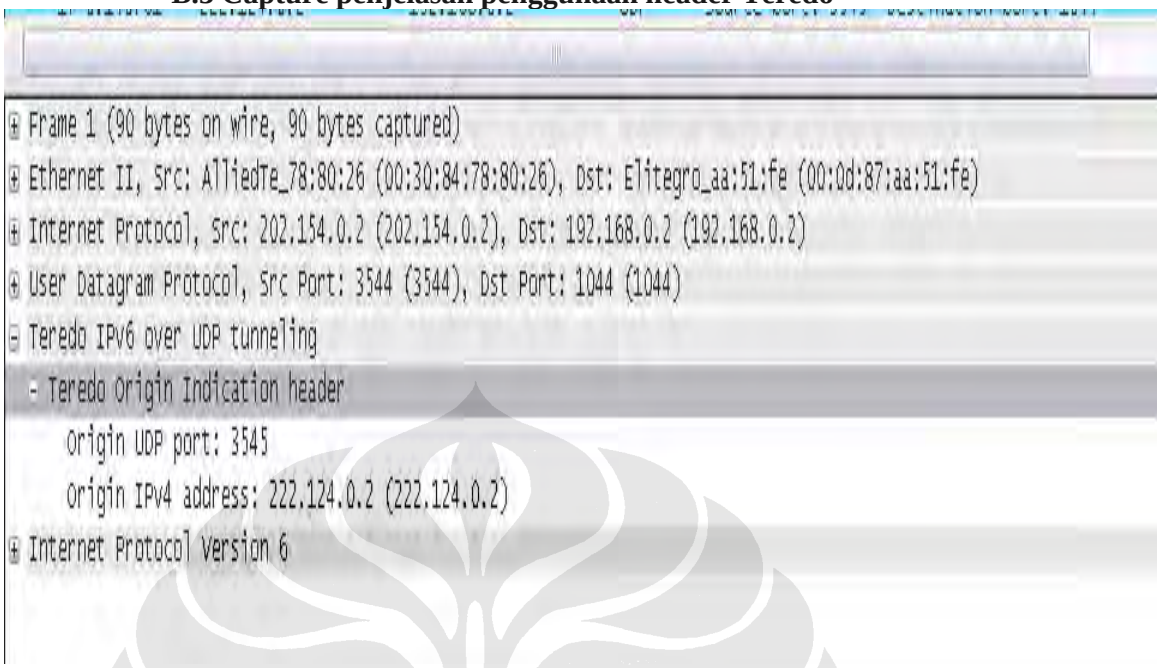
B.1 Streaming pada jaringan mekanisme Teredo



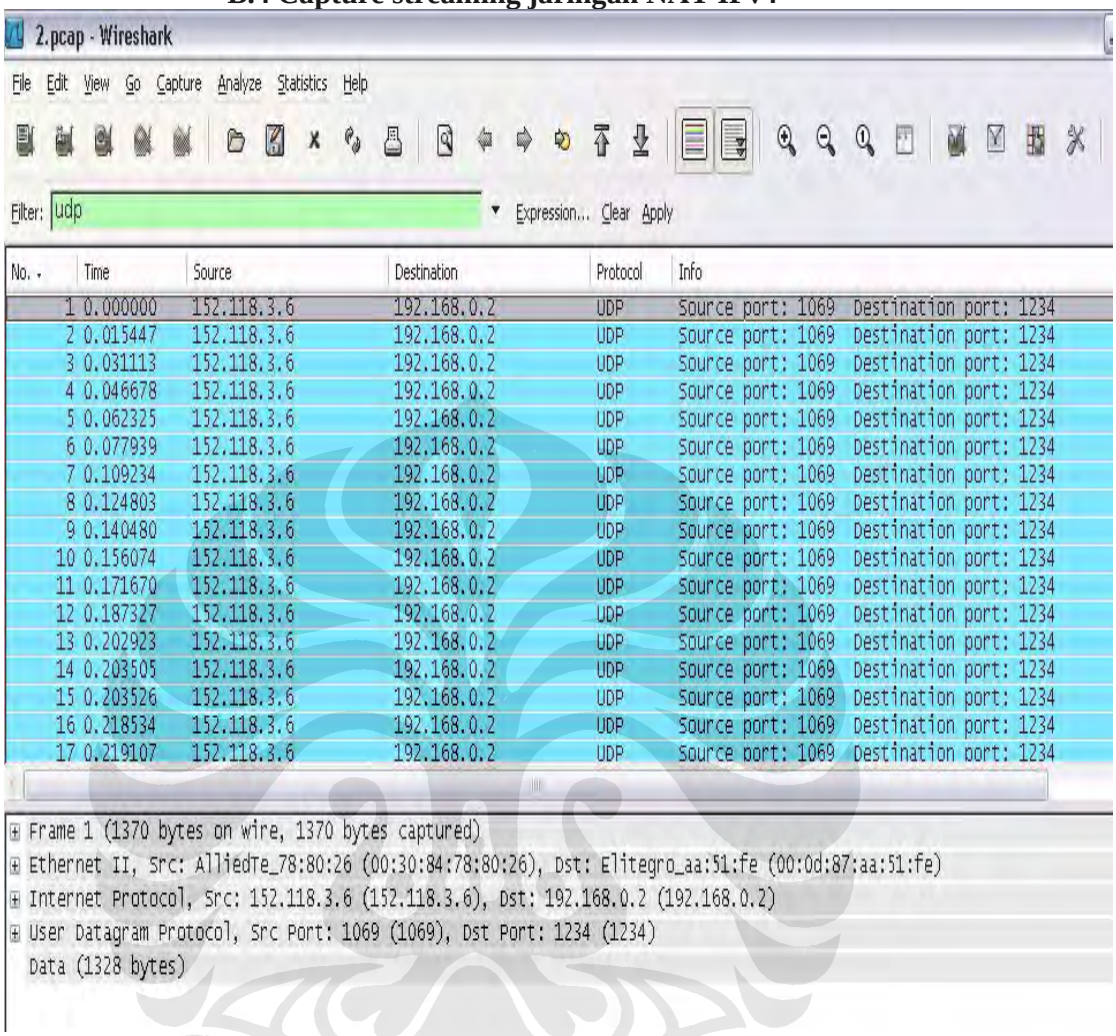
B.2 Capture Summary untuk file TestVid.avi pada jaringan teredo



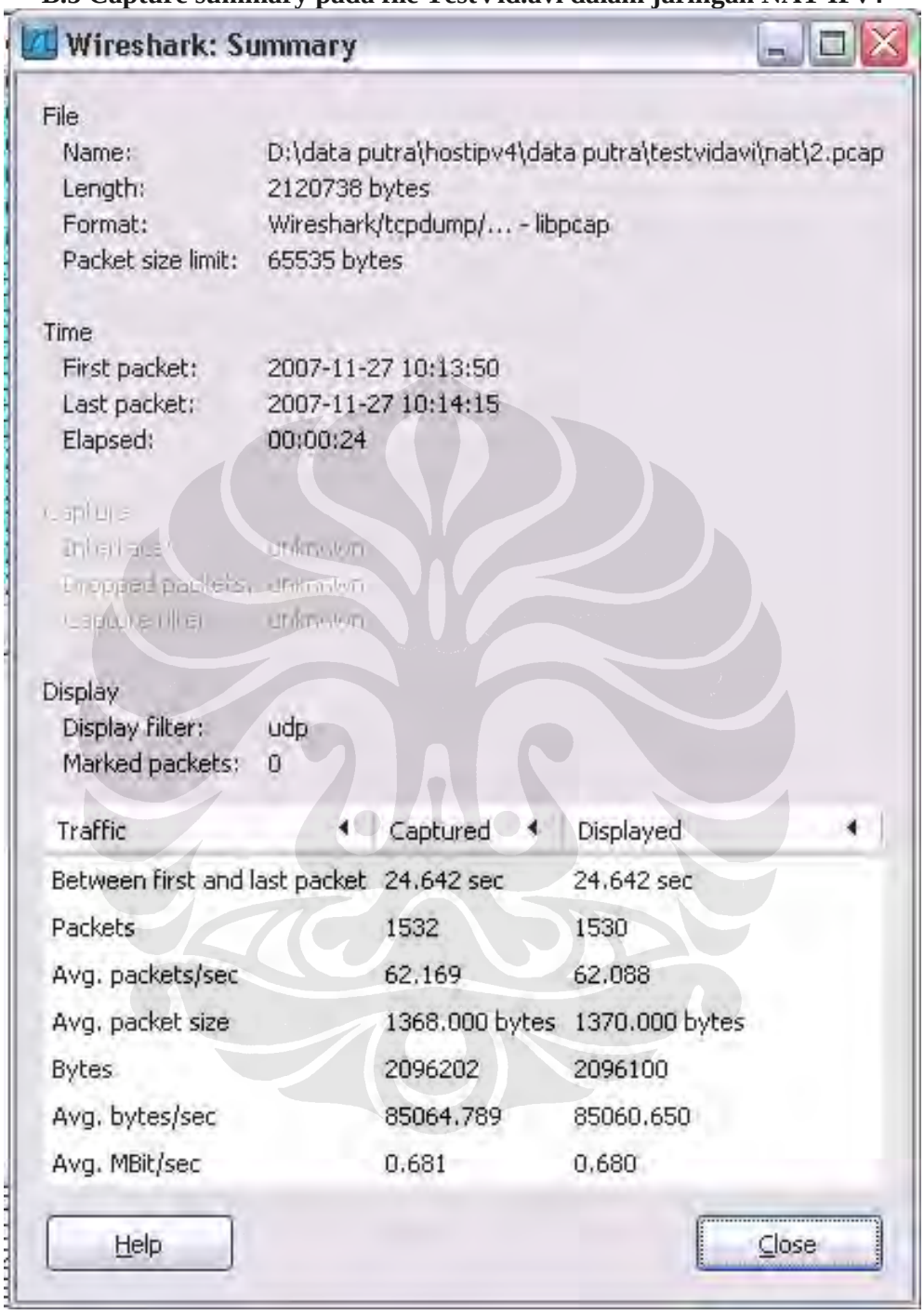
B.3 Capture penjelasan penggunaan header Teredo



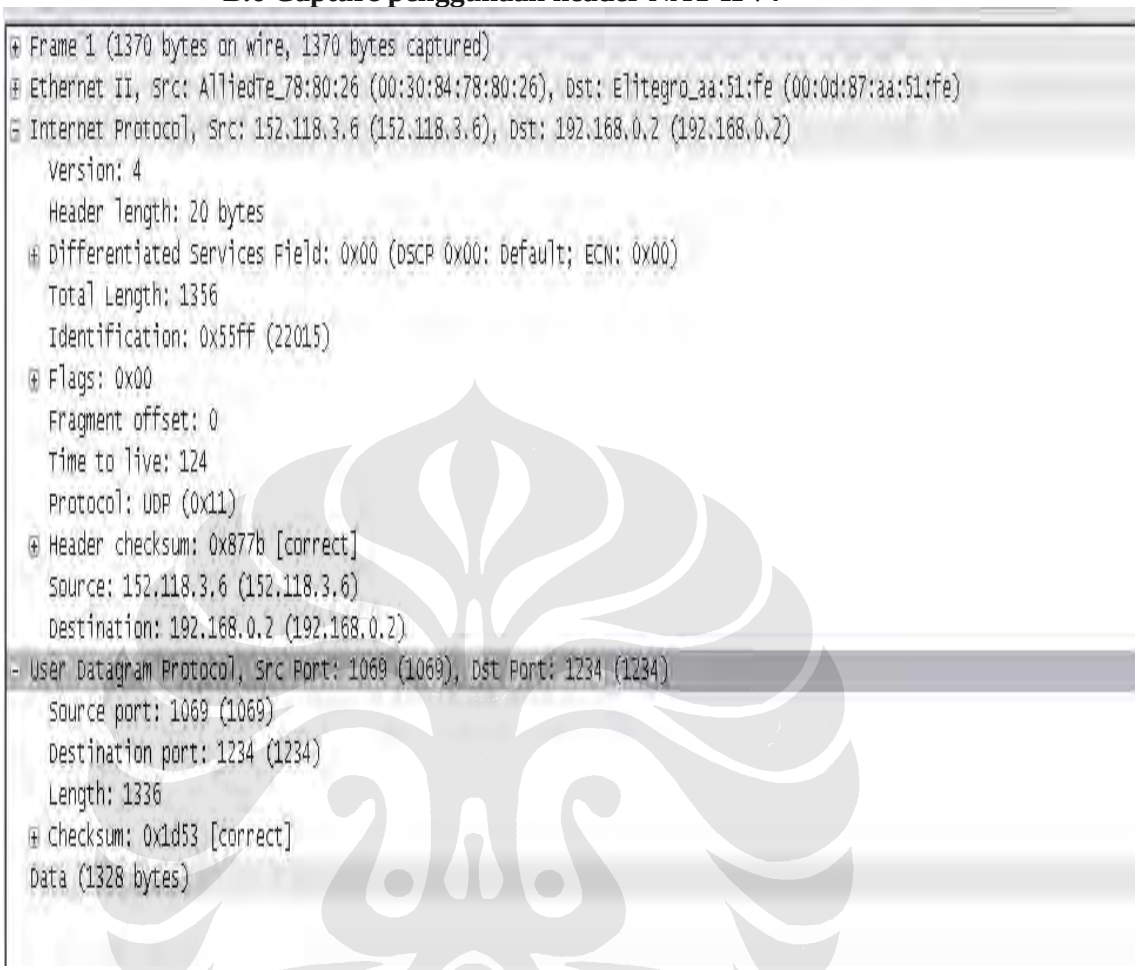
B.4 Capture streaming jaringan NAT IPv4



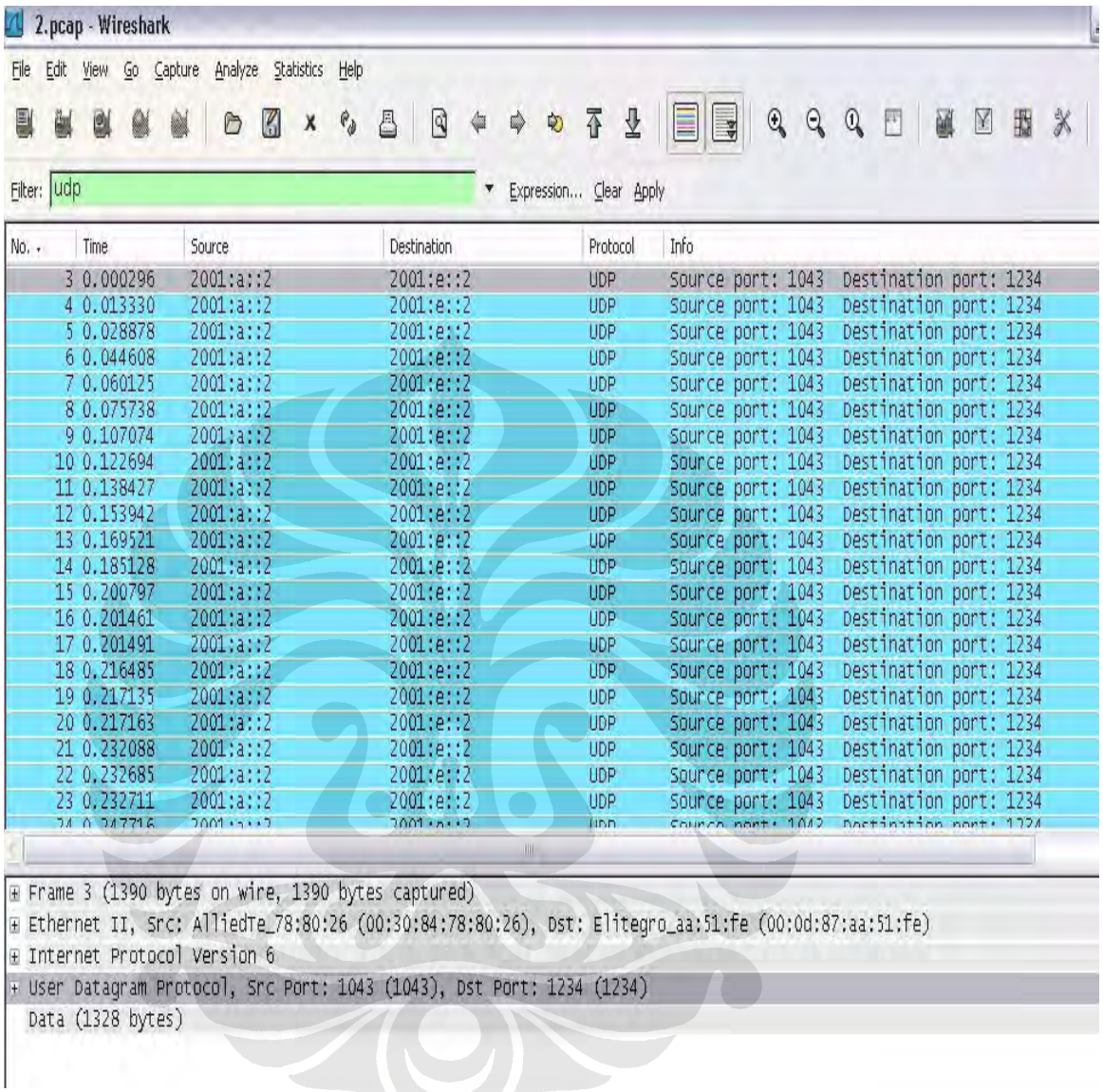
B.5 Capture summary pada file TestVid.avi dalam jaringan NAT IPv4



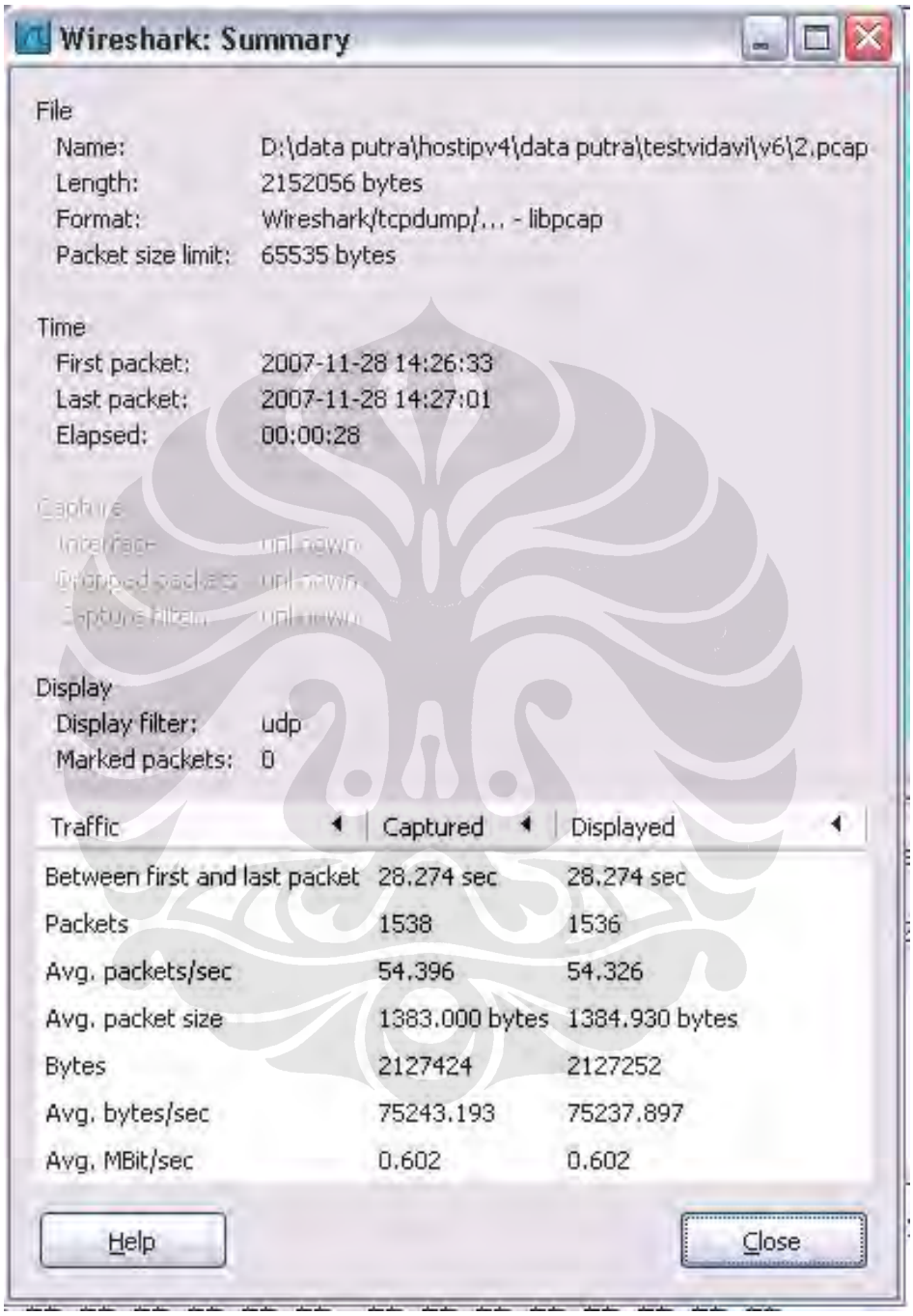
B.6 Capture penggunaan header NAT IPv4



B.7 Capture wireshark pada jaringan IPv6



B.8 Capture summary pada file TestVid.avi dalam jaringan IPv6

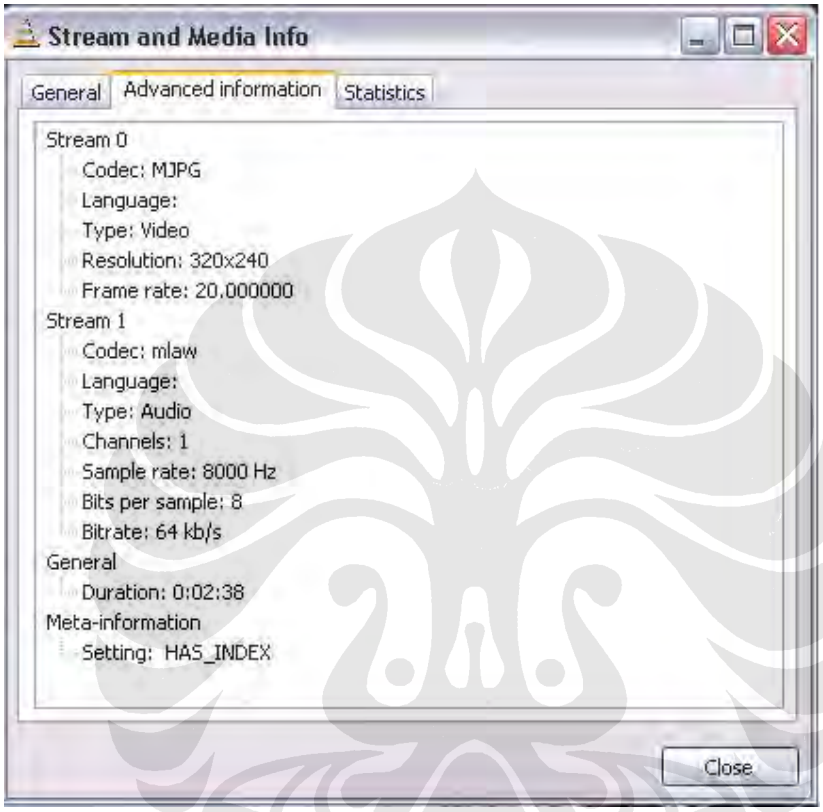


B.9 Capture penggunaan header pada IPv6

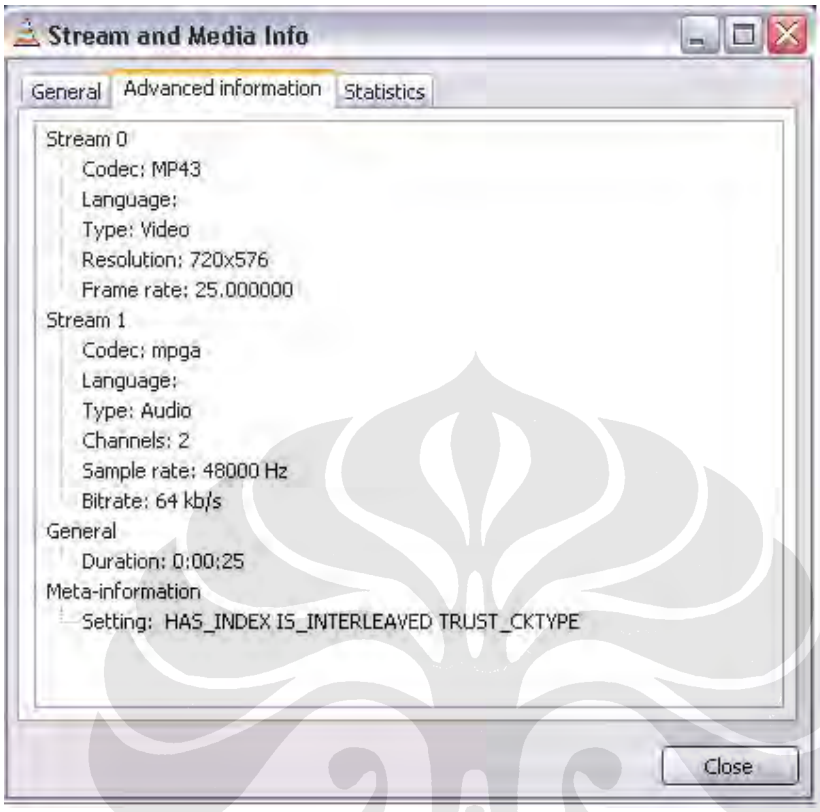
Lampiran C

Karakteristik Video

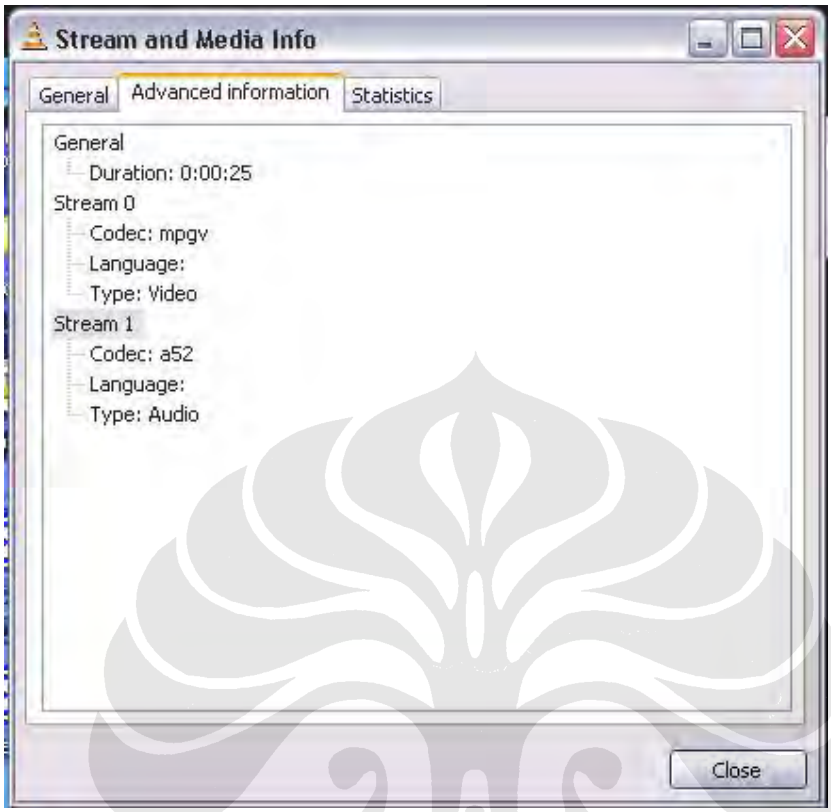
C.1 Avilong.avi



C.2 TestVid.avi



C.3 TestVid.mpg



C.4 TestVid.mp4