KINERJA TCP PADA JARINGAN VPN DENGAN ESTIMASI RTO (RETRANSMISION TIME OUT)

[1]Henni Endah Wahanani, [2]Budi Nugroho, [3]Harris Cipta Abdi
[1,2,3] Program Studi Teknik Informatika Fakultas Ilmu Komputer
Universitas Pembangunan Nasional "Veteran" Jawa Timur
Jl. Raya Rungkut Madya, Gunung Anyar, Surabaya, Jawa Timur 60294
Email: [1]henniendah.if@upnjatim.ac.id, [2]budinugroho.if@upnjatim.ac.id, [3]ciptaabdi21@gmail.com

Abstrak. *Pada masa sekarang ini kebutuhan dunia internet sangat berkembang pesat dengan banyaknya berbagai penyedia layanan yang saling bersaing memberikan layanan terbaik dari segi kecepatan download dan upload, serta sekuritas keamanan. Terkadang penyedia layanan tidak menyadari bahwasanya kecepatan akses internet tidak sesuai dengan promosi yang ditawarkan, hal kecil penyebabnya diantaranya adalah waktu Round Trip Time (RTT) yang merupakan waktu respon round trip paket yang sering diabaikan padahal ini hal kecil namun menentukan bandwidth yang nantinya diperoleh oleh customer layanan internet dimana akan mempengaruhi nilai RTO (Retransmission Time Out).. Hasil yang diperoleh yaitu terdapat perbedaan waktu RTO tanpa VPN menggunakan Algoritma Jacobson/Karels lebih tinggi 42% yaitu 0,2 s dari algoritma Original dengan nilai 0,14 s. Nilai RTO pada jaringan dengan VPN menggunakan Algoritma Jacobson/Karels lebih rendah yaitu 0,38 s dari algoritma original dengan nilai 0,42 s*

Kata Kunci: TCP, RTO, Algoritma Jacobson/Karels

Kebutuhan akan akses internet dalam dunia modern ini sangat berkembang dengan pesat. Tentu saja bandwidth serta keamanan sangat dibutuhkan dalam hal ini. Penyedia layanan internet pun bersaing untuk mendapatkan hati para pelanggan mereka untuk setia menggunakan produk-produk layanan internet yang mereka promosikan. Dalam hal ini tentu para penyedia layanan internet perlu memperhatikan hal apa saja yang dibutuhkan pelanggan mereka salah satunya kecepatan akses internet dan waktu respon jaringan mereka.

Pemakaian sarana Internet sebagai jaringan publik untuk menerima maupun mengirimkan suatu informasi memiliki sebuah resiko tersendiri, karena internet merupakan sarana yang terbuka dimana semua dapat mengakses data secara bebas, sehingga hal–hal yang mengenai masalah kerahasiaan serta authentifikasi atas informasi pun juga otomatis akan terbuka. Oleh sebab itu teknologi VPN (Virtual Privat Network) merupakan hal yang dapat memberikan sesuatu yang dibutuhkan dalam hal keamanan data, menjaminnya dengan bermacam-macam protokol untuk enkripsi data. Salah satu jenis protocol dari VPN adalah PPTP (Point-to-Point Tunneling Protocol).

PPTP adalah sebuah protokol yang mengizinkan hubungan PPP (Point-to Point Protocol melewati jaringan IP, pada jaringan VPN. Microsoft mengimplementasikan protokol dan algoritmanya untuk mendukung PPTP. Implementasi dari PPTP ini disebut Microsoft PPTP, yang digunakan untuk mendukung perluasan dalam mengkomersilkan produk tunneling atau VPN khususnya yang telah terdapat pada Microsoft Windows 95, 98 dan NT.

Tunneling merupakan proses transmisi paket dari komputer ke komputer pada jaringan private dengan menggunakan jaringan internet. Network routers yang berada diluar jangkauan jaringan private tidak dapat mengakses komputer yang berada di jaringan private. Tunneling memungkinkan routing network untuk mengalirkan data ke komputer perantara, seperti server yang dikoneksikan ke routing network dan jaringan.

Pada penelitian ini untuk mengoptimalkan waktu respon dan kecepatan akses internet dengan baik pada protokol TCP di proses tunneling PPTP pada jaringan VPN dengan menghitung estimasi RTO dengan menggunakan Algoritma Jacabson/Karels dan Algoritma Karn's/Original.

I. Metodologi

VPN (Virtual Private Network)

VPN adalah suatu jaringan private yang menggunakan sarana jaringan komunikasi publik yaitu Internet dengan memakai konsep

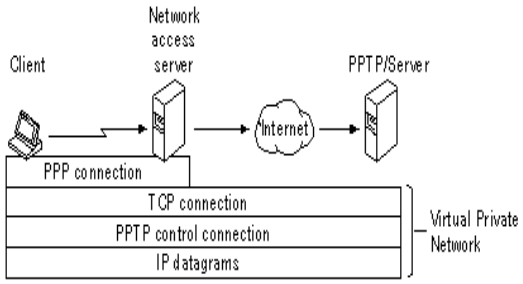
tunnelling protocol serta prosedur pengamanannya. VPN dapat digunakan pada jaringan yang telah tersedia, misalnya seperti internet, sehingga VPN dapat memfasilitasi suatu bentuk transfer data yang bersifat sensitive dan rahasia, secara aman melalui jaringan publik.

Teknologi tunneling pada VPN merupakan teknologi yang bertugas untuk manangani dan menyediakan koneksi point-to-point dari sumber ke tujuannya. Disebut tunnel karena koneksi point-to-point tersebut sebenarnya terbentuk dengan melintasi jaringan umum, namun koneksi tersbut tidak memperdulikan paket-paket data milik orang lain yang sama-sama melintasi jaringan umum tersebut, tetapi koneksi tersebut hanya melayani transportasi data dari pembuatnya.

Koneksi point-to-point ini sesungguhnya tidak benar-benar ada, namun data yang dihantarkannya terlihat seperti benar-benar melewati koneksi pribadi yang bersifat point-to-point. Teknologi ini dapat dibuat di atas jaringan dengan pengaturan IP Addressing dan IP Routing. Maksudnya, antara sumber tunnel dengan tujuan tunnel telah dapat saling berkomunikasi melalui jaringan dengan pengalamatan IP. Apabila komunikasi antara sumber dan tujuan dari tunnel tidak dapat berjalan dengan baik, maka tunnel tersebut tidak akan terbentuk dan VPN pun tidak dapat dibangun. Apabila tunnel tersebut telah terbentuk, maka koneksi point-to-point palsu tersebut dapat langsung digunakan untuk mengirim dan menerima data. Namun, di dalam teknologi VPN, tunnel tidak dibiarkan begitu saja tanpa diberikan sistem keamanan tambahan. Tunnel dilengkapi dengan sebuah sistem enkripsi untuk menjaga data-data yang melewati tunnel tersebut. Proses enkripsi inilah yang menjadikan teknologi VPN menjadi aman dan bersifat pribadi.

PPTP (Point to Point Tunneling Protocol) [8] adalah protokol jaringan yang memungkinkan Transfer data yang aman dari klien remote ke server perusahaan swasta dengan menciptakan VPN melalui jaringan data berbasis TCP/IP. PPTP mendukung on-demand, multi-protokol, jaringan pribadi virtual lebih umum jaringan seperti Internet.

Teknologi jaringan PPTP merupakan pengembangan dari remote access Point-to-Point protocol yang dikeluarkan oleh IETF (Internet Engineering Task Force). PPTP merupakan protokol jaringan yang merubah paket PPP menjadi IP datagrams agar dapat ditransmisikan melalui intenet. PPTP juga dapat digunakan pada jaringan private LAN-to-LAN. PPTP terdapat sejak dalam sistem operasi Windows NT server dan Windows NT Workstation versi 4.0. Komputer yang berjalan dengan sistem operasi tersebut dapat menggunakan protokol PPTP dengan aman untuk terhubung dengan private network sebagai klien dengan remote access melalui internet. PPTP juga dapat digunakan oleh komputer yang terhubung dengan LAN untuk membuat VPN melalui LAN.



Gambar 1. PPTP Tunnel [8]

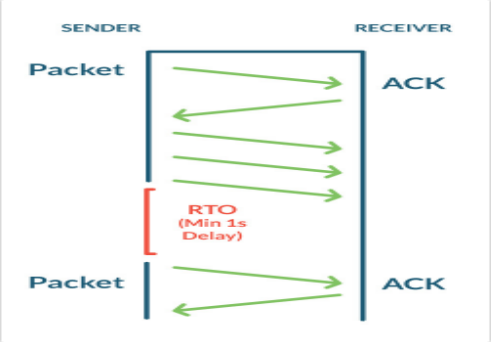
PPTP membuat sebuah control connection dari klien PPTP ke server PPTP di internet. Koneksi tersebut menggunakan protokol TCP untuk membagun koneksi dan disebut dengan PPTP tunnel.

Prinsip Konservasi Paket

Pengirim dapat mengirimkan data ketika sejumlah data telah berhasil ditransmisikan jaringan sampai ke tujuan (keluar dari jaringan) [1]

RTO (Retransmission Time Out)

RTO adalah ketika komputer server tidak merespon permintaan koneksi dari klien setelah beberapa lama (jangka waktu timeout bervariasi. RTO merupakan salah satu yang paling penting dalam kinerja mengkuantifikasi TCP. RTO dimulai ketika segmen outbound diturunkan ke IP. Jika tidak Acknowledgment (ACK) dalam segmen data tertentu sebelum timer berakhir, maka segmen data tersebut ditransmisikan ulang. Retransmission Time Out menambahkan masalah yang secara signifikan untuk jaringan dan kinerja aplikasi dan tentu saja membutuhkan beberapa tuning dan optimasi [9].



Gambar 2. RTO [9]

Algoritma Jacobson/Karels

Di November 1988, Jacobson & Karels menulis suatu penelitiannya yang berjudul Congestion Avoidance and Control di mana mereka menggambarkan tujuh algoritma baru yang dimasukkan dalam BSD TCP 4.0.Tujuh algoritma baru dimasukkan dalam BSD TCP 4.0. adalah:

1. Varian waktu estimasi RTT (Round Trip Time).
2. Eksponensial retransmition backoff .
3. Slow -Start.
4. Penerima kebijakan ACK lebih agresif
5. Dynamic window sizing on congestion.
6. Karn's clamped retransmition backoff.
7. Retransmition cepat.

Aliran pada koneksi TCP harus menaati prinsip konservasi paket, jika prinsip ini ditaati, suatu kemacetan data akan menjadi suatu pengecualian. Dengan konservasi paket berarti diketahui bahwa untuk mendapatkan koneksi dalam kesetimbangan [2], keadaan harus berjalan stabil dengan full window data di dalam transmisi, flow packet sering disebut fisikawan sebagai konservatif, yaitu paket data baru yang tidak dimasukkan ke dalam jaringan sampai paket data lama telah selesai. Hanya ada tiga cara untuk mengkonservasi paket yang gagal seperti dibawah ini :

1. Koneksi tidak mencapai ekuilibrium / kesetimbangan.
2. Sebuah pengirim mengirimkan lagi paket data baru sebelum yang paket data yang lama keluar.
3. Kesetimbangan tidak dapat dicapai karena kurangnya sumber daya.

Kegagalan kedua berurusan dengan suatu konservasi pada kesetimbangan, membutuhkan waktu RTT yang baik. Waktu perjalanan estimator RTT yang baik , waktu inti dari retransmition adalah fitur yang paling penting dari implementasi protokol yang mengharapkan agar dapat bertahan dari beban berat. Satu kesalahan kalau nantinya tidak memperkirakan variasi, s, RTT. Dari teori antrian, kita tahu bahwa peningkatan RTT dan variasi RTT sangat cepat dengan adanya beban besar. Spesifikasi Protokol TCP [3] menunjukkan dan memperkirakan rata-rata RTT via low-pass filter:

$$SRTT = \alpha*SRTT+(1-\alpha)*RTT$$

Keterangan :
SRTT : Smoothed Round Trip Time (nilai update setelah RTT).
α : Standarisasi nilai alpha yaitu 0,836 (sesuai rekomendasi TCP yaitu 0,8 – 0,9).

Setelah nilai SRTT diperbarui,interval timeout retransmit dan RTO untuk paket berikutnya dikirimkan yang kemudian diatur untuk perhitungan β * SRTT. Perhitungan parameter β untuk variasi RTT. Nilai β [4] disarankan = 2 dan beradaptasi dengan paling banyak 30%. Di atas titik ini, sambungan akan menanggapi beban paket transmisi yang meningkat dalam suatu perjalanan transmisi. Hal ini memaksa jaringan untuk melakukan suatu pekerjaan yang tidak maksimal, membuang bandwidth pada duplikat paket yang nantinya akan di transmisikan ke tujuan.

Van Jacobson dan Karels mengembangkan metode untuk memperkirakan variasi, dan memancarkan kembali timer atau waktu yang dihasilkan suatu transmisi data, dimana hal tersebut untuk menghilangkan retransmisi palsu. Efek yang memuaskan dengan memperkirakan memakai β, daripada menggunakan fixed value adalah bahwa beban rendah serta kinerja beban tinggi membaik, khususnya saat jalur delay yang tinggi. Metode ini pada dasarnya menghitung RTO menggunakan nilai dinamis β dari nilai tetap (β = 2). Algoritma menghitung rata rata deviasi SRTT dan menggunakan perhitungan ini untuk memperkirakan nilai yang berbasis rata rata deviasi ini [6].

RTT = round-trip time.
SRTT = smoothed value of RTT.
RTO = round-trip time timeout

$SRTT(i+1) = (1-\alpha) * SRTT(i) + \alpha * RTT(i+1)$

$RTO = \beta * SRTT$

Sekarang β adalah dinamis, dan nilainya tergantung dari rata-rata deviasi nilai SRTT dengan memperhatikan nilai-nilai riil yang diukur.

Dalam persamaan $SRTT(i) = (1-\alpha) * SRTT(i-1) + \alpha * RTT(i)$, itu mungkin jelas bahwa SRTT dengan waktu i akan berosilasi secara acak untuk mendapatkan rata-rata yang benar agar bisa mendapatkan satu standar deviasi dari SRTT untuk memastikan hampir menjangkau posibilities. Standar deviasi SRTT akan menjadi α*stddev(RTT). Alih-alih menggunakan standar deviasi lebih baik menggunakan rata-rata penyimpangan atau deviasi, karena:

1. Membutuhkan sumber daya CPU yang lebih kecil , tidak memerlukan pengukuran untuk mengambil akar kuadrat persegi.
2. Ini lebih konservatif (yaitu lebih besar) estimasi variasi dari standar deviasi.

Untuk menghitung RTO saat ini, pengirim TCP mempertahankan dua variabel, SRTT (smoothed RTT) dan RTTVAR (Variasi Round Trip Time). Selain itu, diasumsikan granularity / waktu dari Granulity (G) dalam detik.

RFC 6298 Computing TCP Retransmisi Timer Juni 2011 [5] adalah peraturan yang mengatur perhitungan SRTT, RTTVAR, dan RTO adalah sebagai berikut:

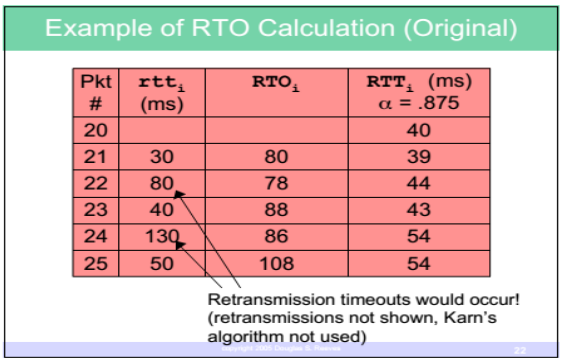
1. Pengukuran RTT telah dilakukan untuk segmen yang dikirim antara pengirim dan penerima, pengirim harus mengatur RTO <- 1 detik. Perhatikan bahwa versi sebelumnya menggunakan sebuah awal RTO dari 3 detik .Implementasi TCP masih mungkin menggunakan nilai ini (atau nilai lainnya > 1 detik).
2. Ketika pertama RTT pengukuran R dibuat, host harus mengatur
   SRTT <- R
   RTTVAR <- R / 2
   RTO <- SRTT + max (G, K * RTTVAR) dimana K = 4.
3. Ketika RTT pengukuran R 'selanjutnya dibuat, host harus mengatur
   RTTVAR <- (1 - beta)*RTTVAR + beta*| SRTT - R '|
   SRTT <- (1 - alpha) * SRTT + alpha * R '
   Nilai SRTT digunakan untuk update nilai RTTVAR .
   Bahwa untuk memperbarui RTTVAR dan SRTT harus dihitung di atas suatu permintaan data.
   Rumus di atas harus dihitung dengan menggunakan alpha = 1/8 dan beta = 1/4 (disarankan oleh Jacobson/Karels).
   Setelah perhitungan, host harus memperbarui nilai RTO yang rumusnya RTO <- SRTT + max (G, K * RTTVAR).
4. Setiap kali RTO dihitung, jika kurang dari 1 detik, maka RTO harus dibulatkan menjadi 1 detik. Secara tradisional, implementasi TCP menggunakan jam butiran kasar untuk mengukur RTT dan memicu RTO, yang membebankan besar nilai minimum pada RTO. Penelitian menunjukkan bahwa besar minimum RTO diperlukan untuk menjaga TCP tetap konservatif dan menghindari transmisi ulang yang palsu. Oleh karena itu, spesifikasi ini membutuhkan minimal RTO besar sebagai pendekatan konservatif, sementara RFC 6298 Computing TCP Retransmisi Timer Juni 2011 pada saat yang sama mengakui bahwa di beberapa titik di masa depan nanti, penelitian dapat menunjukkan bahwa nilai minimal kecil RTO bisa diterima.
5. Nilai maksimum mungkin ditempatkan pada RTO asalkan setidaknya 60 detik.

Algoritma Karn's

Menghitung nilai akurasi dari RTT sangat sulit karena ambigu dari beberapa segment yang dikirimkan pada jaringan TCP oleh karenanya untuk menghadapi hal tersebut Algoritma yang umum di gunakan adalah Algoritma Original atau disebut Algoritma Karn's. Perhitungannya sama dengan Algoritma Jacobson/Karels pembeda nya adalah dalam Algoritma Jacobson terdapat penambahan perhitungan nilai variasi dari RTT yang disebut RTTV sebelum menentukan nilai dari RTO pada jaringan TCP.

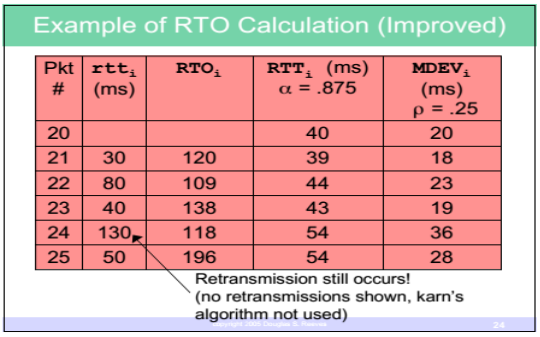
Dalam Algoritma Jacobson/Karels sampel awal nilai RTT di hitung menggunakan Algoritma Karn's [6]. Algoritma Original atau

Karn's ini terkadang tidak digunakan pada suatu perhitungan disaat nilai dari RTO kurang dari nilai RTT [7], karena pada dasarnya nilai RTO adalah dua kali nilai RTT, maka dari itu Algoritma Jacobson/Karels memberikan solusi dengan penemuannya untuk mengatasi hal ini apabila terjadi anomali nilai RTO yang kecil dari pada nilai RTT. Hal tersebut dapat dijelaskan pada gambar 2 :



Gambar 3. Contoh Estimasi nilai RTO dengan Algoritma Original [7]

Gambar 3. terlihat jelas pada paket nomor 22 dan pada paket nomor 24 bila nilai RTO lebih kecil dari nilai RTT yaitu dengan RTT 80 mempunyai nilai RTO 78 dan RTT 130 mempunyai nilai RTO hanya 86 dan hal ini menyebabkan Algoritma Original tidak dapat di gunakan karena nilai RTO pada Algoritma Original harus dua kali nilai RTT. Solusinya menggunakan Algortima Jacobson/Karels seperti gambar 4 di bawah ini :



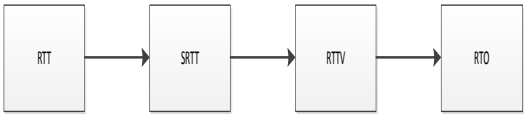
Gambar 4. Contoh Estimasi nilai RTO dengan Algoritma Jacobson/Karels [7]

Perhitungan Algoritma Jacobson/Karels pada gambar 4. memperbaiki perhitungan menggunakan Algoritma Original/Karn dengan penambahan parameter nilai variasi RTTV atau MDEV, terlihat pada paket 22 yang sebelumnya nilai RTT lebih besar daripada nilai RTO sekarang sudah dapat diperbaiki menggunakan Algoritma Jacobson/Karels. Dan pada nomor paket 24 terlihat nilai RTT masih sedikit tinggi daripada nilai RTO dalam hal ini menurut penjelasan menggunakan Algoritma Jacobson/Karels masih terjadi suatu transmisi data saat pengiriman data pada jaringan TCP. Dengan demikian solusi ini dapat terpecahkan dan dapat di atasi menggunakan Algoritma Jacobson/Karels untuk menentukan nilai RTO yang mendekati nilai akurasi yang baik.

Perbandingan Algoritma Original/Karn's dan Algoritma Jacobson/Karels

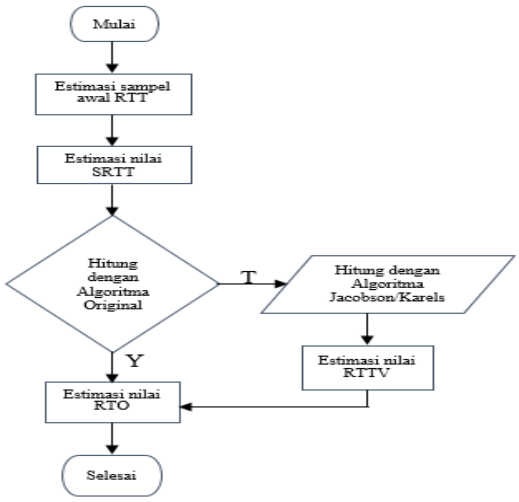
Pada dasarnya perbandingan algoritma untuk mengetahui konsep, urutan atau pun tata cara yang berbeda diantara kedua algoritma tersebut dalam menentukan nilai RTO (Retransmision Time Out).

Gambar 5. Urutan estimasi RTO dengan Algoritma Original/Karn's



Gambar 6. Urutan estimasi RTO dengan Algoritma Jacobson/Karels

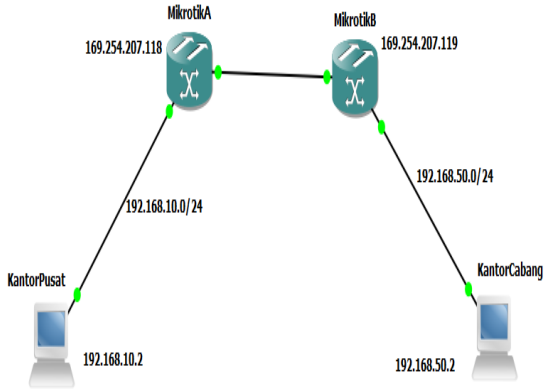
Rancangan Sistem yang dibangun memiliki diagram alur sebagai berikut :



Gambar 7. Diagram alur perhitungan RTO

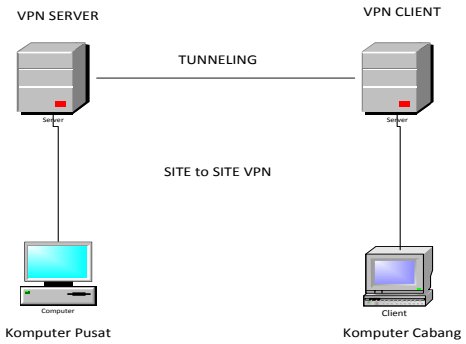
Pada gambar 7. menjelaskan bahwa untuk menghitung nilai dari RTO maka terlebih dahulu menghitung nilai RTT, SRTT,RTTV terlebih dahulu, ini berlaku untuk perhitungan menggunakan Algoritma Jacobson/Karels namun untuk perhitungan menggunakan Algoritma Original hanya perlu menghitung nilai RTT,SRTT tanpa menghitung parameter hitung RTTV.

Rancangan topologi jaringan yang digunakan dalam penelitian ini adalah sebagai berikut :



Gambar 8. Topologi jaringan yang digunakan

Pada topologi jaringan yang ditunjukkan pada gambar 8. disini mikrotik A bertindak sebagai router di kantor pusat yang berfungsi sebagai VPN server, dan mikrotik B bertindak sebagai VPN client yang terhubung dengan mikrotik a yang bertindak sebagai VPN server. Pada topologi ini ether 1 digunakan sebagai default IP mikrotik, ether 2 digunakan sebagai penghubung router antara mikrotik A dan mikrotik B pada jaringan tersebut, kemudian ether 3 digunakan untuk penghubung antara router dan komputer yang ada pada jaringan tersebut.



Gambar 9. Rancangan VPN PPTP

Gambar 9. menjelaskan bahwa rancangan VPN PPTP ini nantinya akan digunakan dalam penelitian ini dengan tipe site to site VPN yaitu ada sebuah Server VPN dan salah satunya adalah Client VPN yang menghubungkan LAN pada Kantor Pusat dan Kantor Cabang melalui jalur internet.

II. Hasil dan Pembahasan

Uji coba ini yang dilakukan dalam penelitian ini menggunakan simulator GNS3 dan Virtual box dimana jaringan yang di gunakan yaitu VPN dengan menggunakan Algoritma Jacabson/Karels dan algoritma original.

Analisa RTO pada TCP tanpa VPN

Paket	11	12	13	14	15	16	17	18	19	20
RTT (ms)	4,32	26,31	5,97	23,26	3,83	4,29	5,83	4,86	4,32	4,90

Gambar 10. Nilai RTT tanpa VPN

Gambar 10. merupakan hasil perhitungan RTT dalam jaringan TCP menggunakan TCPing.

Paket	11	12	13	14	15	16	17	18	19	20
RTT (ms)	0	26,31	5,97	23,26	3,83	4,29	5,83	4,86	4,32	4,90
SRTT(ms)	4,32	7,07	6,93	8,97	8,33	7,82	7,58	7,24	6,87	6,63

Gambar 11. Nilai SRTT tanpa VPN

Pada gambar 11 merupakan hasil capture TCPing untuk mendapatkan nilai dari SRTT, perhitungannya dengan rumus :
SRTT = α * SRTT + (1-α) * RTT.

Paket	RTT(ms)	SRTT(ms)	RTTV(ms)
11	0	4,32	2,16
12	26,31	7,07	7,12
13	5,97	6,93	5,06
14	23,26	8,97	7,88
15	3,83	8,33	4,62
16	4,29	7,82	2,46
17	5,83	7,58	1,34
18	4,86	7,24	0,33
19	4,32	6,87	-0,48
20	4,90	6,63	-0,85

Gambar 12. Nilai RTTV TCP tanpa VPN

Pada gambar 12 merupakan hasil capture nilai RTTV menggunakan TCPing dari client pada TCP tanpa VPN . dihitung dengan rumus :
RTTV = (1 - β) * RTTV + β * (SRTT – RTT).

Paket	RTT(ms)	SRTT(ms)	RTO(ms)
11	0	4,32	0
12	26,31	7,07	8,64
13	5,97	6,93	14,14
14	23,26	8,97	13,86
15	3,83	8,33	17,94
16	4,29	7,82	16,66
17	5,83	7,58	15,65
18	4,86	7,24	15,15
19	4,32	6,87	14,47
20	4,90	6,63	13,74

Gambar 13. Nilai RTO pada TCP tanpa VPN (Original)

Pada gambar 13. merupakan hasil capture nilai RTO dalam TCP menggunakan Algoritma Original, dihitung dengan rumus :
RTO = 2 x SRTT .

Pada perhitungan nilai RTT pada paket 12 dan 14 melebihi nilai RTO dimana seharusnya nilai RTO adalah 2 kali nilai dari RTT sehingga algoritma original tidak dapat digunakan sehingga perhitungan berikutnya menggunakan algoritma Jacobson/Karels untuk menyempurnakan algoritma original.

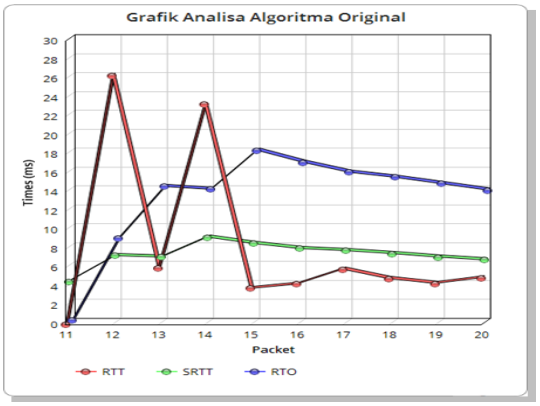
Paket	RTT(ms)	SRTT(ms)	RTTV(ms)	RTO(ms)
11	0	4,32	2,16	0
12	26,31	7,07	7,12	12,96
13	5,97	6,93	5,06	35,54
14	23,26	8,97	7,88	27,19
15	3,83	8,33	4,62	40,49
16	4,29	7,82	2,46	26,83
17	5,83	7,58	1,34	17,66
18	4,86	7,24	0,33	12,96
19	4,32	6,87	-0,48	8,56
20	4,90	6,63	-0,85	4,95

Gambar 14. Nilai RTO pada TCP tanpa VPN (Jacobson/Karels)

Pada gambar 14 merupakan hasil capture nilai RTO dalam TCP tanpa VPN menggunakan Algoritma Jacobson/Karels. Dihitung dengan rumus :

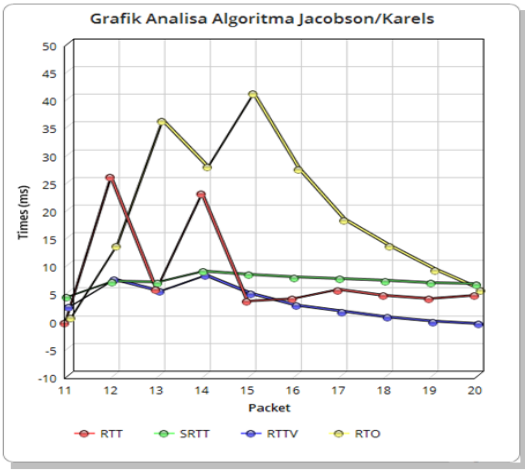
RTOi = SRTTi + 4 x RTTVi

Hasil estimasi nilai perhitungan RTO menggunakan algoritma Jacobson/Karels setidaknya meminimalkan masalah hasil RTO pada algoritma original walaupun masih ada masalah pada paket nomer 12. Menurut algoritma Jacobson/Karels hal ini sudah benar dan menandakan masih terjadi transmisi data pada saat tersebut.



Gambar 15. Grafik Analisa Algoritma Original tanpa VPN

Pada gambar 15 merupakan grafik analisa nilai RTT, SRTT, RTO menggunakan Algoritma Original.



Gambar 16. Grafik Analisa Algoritma Jacobson/Karels tanpa VPN

Pada gambar 16. merupakan grafik analisa nilai RTT, SRTT, RTTV, RTO menggunakan Algoritma Jacobson/Karels. Algoritma ini melakukan penyempurnaan dari algoritma sebelumnya untuk menghitung nilai RTO pada jaringan TCP agar dapat mendekati nilai posibilitas. Nilai RTO lebih besar dari

RTT hanya saja pada paket nomer 12 masih lebih besar nilai RTT dibandingkan RTO hal ini menunjukkan paket masih mengalami transmisi data.

Analisa RTO pada TCP dengan VPN

Dalam tahap ini bertujuan mendapatkan nilai sampel awal RTT pada jaringan TCP dengan VPN dengan pengujian TCPing.

Paket	RTT (ms)
11	209,84
12	211,40
13	209,49
14	210,32
15	209,49
16	210,53
17	211,30
18	209,67
19	211,63
20	211,43

Gambar 17. Nilai RTT dengan VPN

Pada gambar 18 nilai SRTT ini lebih besar dibandingkan nilai SRTT tanpa VPN dengan nilai rata-rata 210,09 ms dan terlihat stabil nilai perbandingan RTT dan SRTT pada VPN ini.

Paket	RTT (ms)	SRTT (ms)
11	0	209,84
12	211,40	210,04
13	209,49	209,97
14	210,32	210,01
15	209,49	209,95
16	210,53	210,02
17	211,30	210,18
18	209,67	210,12
19	211,63	210,30
20	211,43	210,45

Gambar 18. Nilai SRTT dengan VPN

Nilai RTTV yang ditunjukkan pada gambar 19. lebih kecil dibandingkan nilai RTT dan SRTT tetapi tidak ada nilai minus seperti perhitungan nilai RTTV tanpa VPN.

Paket	RTT(ms)	SRTT(ms)	RTTV(ms)
11	0	209,84	104,92
12	211,40	210,04	79,08
13	209,49	209,97	59,17
14	210,32	210,01	44,47
15	209,49	209,95	33,22
16	210,53	210,02	25,06
17	211,30	210,18	19,12
18	209,67	210,12	14,21
19	211,63	210,30	11,04
20	211,43	210,45	8,56

Gambar 19. Nilai RTTV dengan VPN

Pada gambar 20. Nilai RTO sebagian besar nilainya lebih besar dari nilai RTT namun sangat konstan dan stabil dikarenakan nilai perhitungan RTT dan SRTT juga stabil dan konstan. Hal ini karena VPN terdapat fasilitas enkripsi dan dekripsi sehingga membutuhkan waktu RTT yang lebih lama daripada jaringan tanpa VPN. Nilai RTO dengan algoritma original dapat digunakan karena tidak ada kesalahan estimasi yang disebabkan melonjaknya nilai RTT dengan nilai rata-rata RTO 400 ms.

Paket	RTT(ms)	SRTT(ms)	RTO(ms)
11	0	209,84	0
12	211,40	210,04	419,68
13	209,49	209,97	420,07
14	210,32	210,01	419,93
15	209,49	209,95	420,02
16	210,53	210,02	419,89
17	211,30	210,18	420,04
18	209,67	210,12	420,36
19	211,63	210,30	420,23
20	211,43	210,45	420,61

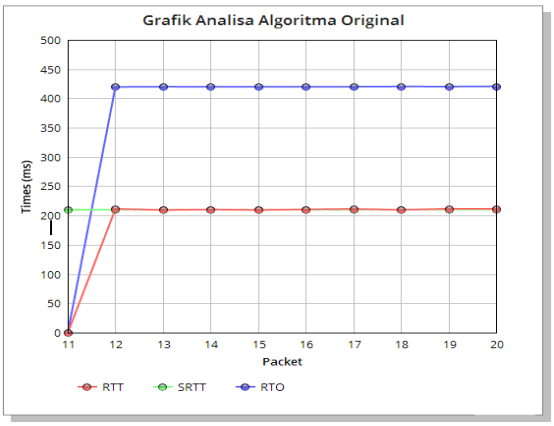
Gambar 20. Nilai RTO dengan VPN (Original)

Nilai RTO dengan VPN menggunakan algoritma Jacobson/Karels yang ditunjukkan pada gambar 21. Rata-rata nilai RTO lebih besar dibandingkan RTO pada algoritma original.

Paket	RTT(ms)	SRTT(ms)	RTTV(ms)	RTO(ms)
11	0	209,84	104,92	0
12	211,40	210,04	79,08	629,52
13	209,49	209,97	59,17	526,36
14	210,32	210,01	44,47	446,66
15	209,49	209,95	33,22	387,89
16	210,53	210,02	25,06	342,83
17	211,30	210,18	19,12	310,27
18	209,67	210,12	14,21	286,65
19	211,63	210,30	11,04	266,96
20	211,43	210,45	8,56	254,45

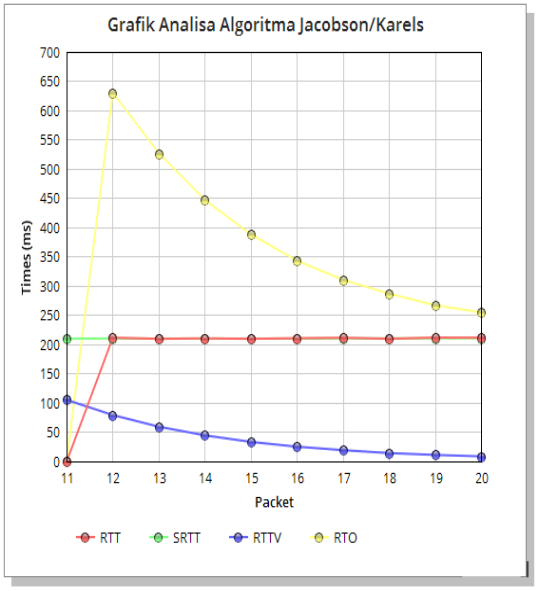
Gambar 21. Nilai RTO dengan VPN (Jacobson/Karels)

Semua nilai dari RTT, SRTT,RTO pada gambar 22. tampak terlihat stabil dan konstan, dan sangat jauh berbeda dibandingkan tanpa VPN.



Gambar 22. Grafik Algoritma Original dengan VPN

Gambar 23. Nilai RTT sangat stabil cenderung konstan tetapi nilai RTO lebih sedikit teratur dibandingkan nilai RTO tanpa VPN.



Gambar 23. Grafik Analisa Algoritma Jacobson/Karels dengan VPN

III. Simpulan

Terdapat perbedaan waktu RTO tanpa VPN menggunakan Algoritma Jacobson/Karels lebih tinggi 42% yaitu 0,2 s dari algoritma Original dengan nilai 0,14 s. Nilai RTO pada jaringan dengan VPN menggunakan Algoritma Jacobson/Karels lebih rendah yaitu 0,38 s dari algoritma original dengan nilai 0,42 s.

IV. Daftar Pustaka

[1] M. Welzl, Network Congestion Control: Managing Internet Traffic, John Wiley & Sons, 2005.

[2] Balliache, Leonardo,

[3] Postel, John, Transmission Control Protocol. RFC 793,1981.

[4] Clark, D. Window and Acknowlegement Strategy in TCP. ARPANET Working Group Requests for Comment, DDN Network Information Center, SRI Internasional, Menlo Park, CA, 1982.

[5] V. Parson, M. Allman, J. Chu, and M.Sargent, Computing TCP's Retransmission Timer, RFC 6298, June 2011.

[6] IETF, Computing TCP's Retransmission Timer, 2011.

[7] N.C State University, Transmission Control Protocol (TCP), Lecture 3, North Carolina, 2005

[8] Microsoft, Understanding Point to Point Tunneling Protocol (PPTP), Microsoft Corporation, 1997

[9] Baker, Justin, TCP RTOs : Retransmission Timeouts & Application Performance Degradation, https://www.extrahop.com/community/blog/2016/retransmission-timeouts-rtos-application-performance-degradation/, 2016.

Halaman ini sengaja dikosongkan.