JITE, 7 (1) July 2023 ISSN 2549-6247 (Print) ISSN 2549-6255 (Online)

JITE (Journal of Informatics and Telecommunication Engineering)

Available online http://ojs.uma.ac.id/index.php/jite **DOI : 10.31289/jite.v7i1.9542**

Received: 19 Mei 2023	Accepted: 26 Juni 2023	Published: 28 July 2023

Comparative Performance Testing of the Impact of ACK Loss in TCP Tahoe, TCP Reno, and TCP New Reno on the ns-2 Simulator

Agung Hernawan1)*

1)Informatika, Fakultas Sains dan Teknologi, Universitas Sanata Dharma Yogyakarta, Indonesia

*Corresponding Email: agung.h@usd.ac.id

Abstrak

Pada jaringan TCP/IP, TCP menyediakan layanan pengiriman yang handal dengan jaminan bahwa paket-paket yang dikirim sampai ke tujuan. Hal ini dicapai melalui mekanisme penggunaan tanda terima berupa kate ACK. TCP mempunyai pengendalian kemacetan untuk mengurangi resiko paket terlambat ataupun hilang. Terdapat beberapa metode yang digunakan oleh TCP untuk mengendalikan kemacetan, salah satunya adalah dengan memperhatikan paket ACK. TCP Tahoe adalah salah satu algoritma yang diterapkan dengan cara ini. TCP Tahoe dipandang mempunyai beberapa kekurangan dan diperbaiki pada TCP Reno, yang kemudian diperbaiki lagi pada TCP New Reno. Pada ketiga varian TCP mengandalkan pada adanya paket ACK, oleh sebab itu dilakukan pengujian untuk melihat efek hilangnya ACK terhadap kinerja TCP. Pengujian dilakukan menambahkan trafik pengganggu pada jalur ACK dengan nilai besaran yang bervariasi. Adanya trafik gangguan ini menyebabkan hilangnya beberapa ACK yang tergantung dari besarnya trafik pengganggu. Pengujian dilakukan dengan memanfaatkan perangkat lunak simulator ns-2. Hasil penelitian menunjukkan hilangnya ACK menyebabkan penurunan kinerja TCP. Kinerja TCP Tahoe berhasil diperbaiki oleh TCP Reno, namun perbaikan algoritma TCP New Reno terhadap TCP Reno tidak berpengaruh.

Kata Kunci: TCP Tahoe, TCP Reno, TCP New Reno, ns-2 simulator

Abstract

In TCP/IP networks, TCP provides reliable delivery service with the guarantee that packets sent reach their destination. This is achieved through the use of acknowledgments (ACK) as a mechanism. In TCP/IP networks, TCP provides reliable delivery service with the guarantee that packets sent reach their destination. This is achieved through the use of acknowledgments (ACK) as a mechanism. In all three variants of TCP, they rely on the presence of ACK packets. Therefore, testing will be conducted to observe the effect of ACK loss on TCP performance. The testing involves introducing disruptive traffic on the ACK path with varying magnitudes. This disruptive traffic results in the loss of some ACK packets, depending on the magnitude of the disruptive traffic. The testing was conducted using the ns-2 simulator software. The research findings indicate that the loss of ACK packets leads to a decrease in TCP performance. TCP Reno successfully improves the performance of TCP Tahoe, but the algorithm improvement in TCP New Reno does not have an effect on TCP Reno..

Keywords: TCP Tahoe, TCP Reno, TCP New Reno, ns-2 simulator

How to Cite: Hernawan, A. (2023). Comparative Performance Testing of the Impact of ACK Loss in TCP Tahoe, TCP Reno, and TCP New Reno on the ns-2 Simulator. JITE *(Journal of Informatics and Telecommunication Engineering)*, 7(1), 91-101.

I. PENDAHULUAN

Jaringan Internet memakai protokol Transmission Control Protocol/Internet Protocol (TCP/IP) sebagai tulang punggungnya. TCP/IP merupakan tulang punggung pada Jaringan Internet. Selain dipakai pada aplikasi yang populer yaitu protokol hypertext transport protocol atau lebih dikenal dengan world wide web. Hadirnya mikrokontroler arduino untuk mengendalikan berbagai macam peralatan, juga memanfaatkan protokol TCP/IP sebagai tulang punggung komunikasi jarak jauh (Sunanto & Jabar, 2019; Julham et al., 2020).

Protokol ini terdiri dari beberapa lapisan, yang masing-masing memiliki tugas yang berbeda. Lapisan Physical berfungsi untuk mengubah data digital menjadi bentuk gelombang agar dapat ditransmisikan. Lapisan Data Link bertanggung jawab masalah perpindahan data dalam suatu link. Lapisan Network mengatur perpindahan data antar jaringan. Perpindahan data dari satu aplikasi ke aplikasi lain menjadi tanggung jawab lapisan Transport (Kurose & Ross, 2021).

Pada lapisan Physical, Data Link, dan Network tidak ada jaminan paket data yang dikirimkan sampai ke tujuan. Guna memberi jaminan agar data sampai di tujuan, terdapat sebuah protokol TCP pada lapisan Transport. Jaminan dilakukan dengan cara penerima mengirimkan Acknowledgement (ACK) untuk setiap paket data yang diperoleh. Dalam batas waktu tertentu, jika ACK sebuah paket tidak diterima oleh pengirim, maka paket tersebut dianggap hilang. Selanjutnya pengirim akan mengirimkan paket data yang hilang tersebut (Forouzan, 2010).

Selain memberi jaminan tidak ada paket yang hilang dengan ACK, TCP juga memiliki mekanisme pengendalian kemacetan (congestion control). Kemacetan terjadi ketika beberapa node secara simultan mengirimkan paket melalui sebuah link. Pengendalian kemacetan diperlukan karena kemacetan dapat menyebabkan hilangnya paket-paket. TCP mengimplementasikan pengendalian kemacetan dengan cara mengurangi laju pengiriman data. Namun, masalahnya adalah pengirim tidak mempunyai informasi langsung tentang adanya jalur yang macet. Pengirim hanya dapat mendeteksi tanda-tanda kemacetan dengan melihat ACK yang diterima. Tidak diterimanya ACK oleh pengirim, menjadi dasar kesimpulan bahwa ada paket yang hilang akibat kemacetan pada saat pengiriman. Selain itu, kemacetan juga dapat terdeteksi melalui perubahan waktu tempuh antara pengiriman paket dan penerimaan ACK. Jika waktu tempuhnya konsisten, diasumsikan lalu lintas data berjalan lancar. Namun, jika waktu tempuhnya bervariasi dan meningkat, maka hal ini menunjukkan adanya kemacetan (Forouzan, 2010).

Ada beberapa algoritma pengendalian kemacetan yang dikembangkan. Dalam pengendalian kemacetan TCP, terdapat dua jenis algoritma yang umum digunakan, yaitu berbasis paket hilang dan berbasis variasi waktu tempuh. Pada TCP Tahoe, pengendalian kemacetan dilakukan dengan berfokus pada deteksi paket yang hilang (Jacobson, 1988). Algoritma ini kemudian diperbaiki dengan pengembangan TCP Reno (Jacobson, 1990) dan New Reno (Hoe, 1996; Clark, 1996; Floyd, 1995). TCP New Reno memperbaiki TCP Reno dengan cara melakukan fast retransmit (Floyd, 1996).

Pengukuran unjuk kerja TCP biasanya dilakukan dengan melakukan pengukuran delay, jitter, packet loss dan throughput (Fahmi, 2018) dan roundtrip (Idwan et al., 2018). Perbedaan unjuk kerja antar varian TCP/IP telah dilakukan. Skenario yang dibangun adalah melihat unjuk kerja TCP/IP ketika kehilangan paket karena adanya antrian ketika terjadi congestion (Fajri, 2021). Beberapa penelitian membuat skenario antrian yang berbeda, misalnya droptail (Fahrudy & Sugiantoro, 2022), RED & SFQ (Febriansyah & Ibnu, 2020; Pamungkas et al., 2018). Beberapa studi tentang unjuk kerja TCP/IP dilakukan dengan membuat skenario untuk berbagai macam media transmisi, seperti pada jaringan LTE (Maulana & Niati, 2019), wired & wireless (Salman et al., 2021)

Penulis sendiri telah mencoba melakukan studi membandingkan unjuk kerja TCP varian Tahoe, Reno dan New Reno. Penelitian dilakukan dengan cara mencoba berbagai tingkat kepadatan, mulai dengan tanpa gangguan, sampai dengan tingkat gangguan 80%. Terbukti algoritma TCP Reno / New Reno dapat memperbaiki algoritma TCP Tahoe. TCP New Reno menghasilkan unjuk kerja yang baik dibanding TCP Reno pada keadaan trafik yang tidak terlalu padat, yaitu 20% - 50%. Namun pada kondisi trafik yang padat yaitu 60%-70% yang terjadi adalah sebaliknya TCP Reno mengalami penurunan unjuk kerja. Kekurangan ini dapat diperbaiki oleh TCP New Reno (Hernawan, 2022).

Penelitian sebelumnya unjuk kerja TCP/IP dilihat dari aspek hilangnya paket pada antrian akibat kepadatan trafik (Fahrudy & Sugiantoro, 2022; Febriansyah & Ibnu, 2020; Pamungkas et al., 2018; Hernawan, 2022) ataupun varian berbagai media transmisi (Maulana & Niati, 2019; Salman et al., 2021). Hilangnya packet tersebut mengakibatkan ketidak hadirnya ACK. Hal ini menentukan unjuk kerja varian TCP/IP yang mendasarkan pada hadirnya ACK.

Kali ini penulis ingin melihat bagaimana pengaruh performansi TCP akibat hilangnya ACK itu sendiri. ACK yang dikirimkan oleh penerima dapat hilang, karena padatnya trafik dari penerima ke pengirim. Akan dilihat seberapa besar dampak dari hilangnya ACK terhadap unjuk kerja TCP. Pada penelitian dibuat skenario untuk menghilangkan ACK pada beberapa tingkatan. Dimulai dengan skenario mulai tanpa gangguan terhadap ACK sampai ke tingkat yang ekstrim yaitu hilangnya seluruh ACK. Dilihat pula apakah benar TCP Reno dan New Reno dapat memperbaiki algoritma TCP Tahoe, jika ketiadaan ACK disebabkan oleh ACK yang hilang. Dimana hal ini berbeda dengan penelitian pada umumnya yaitu ketiadaan ACK dikarenakan paket yang hilang.

II. STUDI PUSTAKA

Menurut dokumen RFC 5681, untuk mengendalikan kemacetan dilakukan dengan berbagai macam fase, yaitu slow *start, congestion avoidance, fast retransmit,* dan *fast recovery*. Dibutuhkan suatu variabel yaitu *cwnd (Congestion Window)* dan *rwnd (Receiver Window)* yang berfungsi untuk membatasi jumlah paket yang boleh dikirim sebelum menerima ACK. Pengirim menentukan nilai *cwnd* dan penerima menentukan nilai *rwnd*. Jumlah paket yang boleh dikirimkan ditentukan oleh nilai terkecil antara *cwnd* dan *rwnd*.

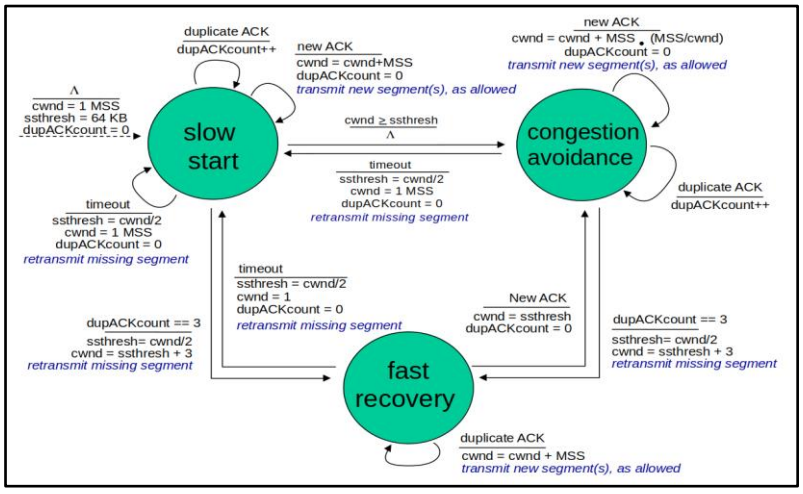
Pada awal pengiriman dipakai *fase slow start*. Pada fase ini diawali dengan nilai *cwnd*=1 dan ditingkatkan secara exponensial untuk setiap ACK paket yang diterima oleh pengirim. Kenaikkan dilakukan sampai mencapai suatu nilai yang disebut *ssthresh*. Pada awalnya nilai *ssthresh* diberi nilai sebesar mungkin sesuai dengan kemampuan jaringan. Setelah melewati nilai *ssthresh*, maka pengiriman masuk ke fase *congestion avoidance*, dimana pada fase ini *cwnd* dinaikkan secara linear. Namun jika terdapat paket hilang, dengan ditandai tidak diterima ACK sampai waktu tertentu (*time-out*), maka pengiriman kembali masuk ke fase *slow start* dan *ssthresh* diubah menjadi setengah dari nilai terakhir *cwnd* dan *cwnd* diset bernilai 1. Tentu saja pengirim berkewajiban mengirim ulang (*retransmit*) paket yang hilang tadi.

Ada kemungkinan dari sejumlah paket yang dikirimkan, ada satu yang hilang. Kalau hal ini terjadi, maka penerima akan mengirimkan sejumlah tanda terima yang sama (*duplicate* ACK), untuk paket yang diterima setelah paket yang hilang tadi. Dari sudut pandang kemacetan, adanya *duplicate* ACK menunjukkan adanya kemacetan, namun tidak separah ketika terjadi *time-out*. Logikanya kalau terjadi *duplicate* ACK, maka hanya satu paket yang hilang dari sejumlah paket yang dikirim. Sementara kalau terjadi *time-out*, berarti tidak ada ACK sama sekali yang diterima sampai dengan waktu yang ditentukan.

Pada TCP Tahoe adanya *duplicate* ACK tadi tidak dipertimbangkan. Hal ini diperbaiki pada TCP Reno, yang mempertimbangkan kalau terjadi *duplicate* ACK. TCP Reno masuk ke fase *fast recovery* ketika menerima 3 *duplicate* ACK. Sama halnya ketika masuk ke fase *slow start*, pada fase *fast recovery ssthresh* beri nilai setengah dari nilai *cwnd* pada saat itu. Bedanya ketika masuk fase *slow start cwnd* diset menjadi bernilai satu, tetapi pada fase *fast recovery* diset menjadi bernilai *ssthresh*+3. Perbedaan lainnya adalah pada fase slow start *cwnd* dinaikkan secara eksponensial, sementara pada fase *fast recovery* dinaikkan secara linear, sebagaimana pada fase *congestion avoidance*. Hal ini memberi keuntungan pada TCP Reno pengiriman paket berjalan lebih cepat dibandingkan dengan TCP tahoe ketika terjadi 3 *duplicate* ACK.

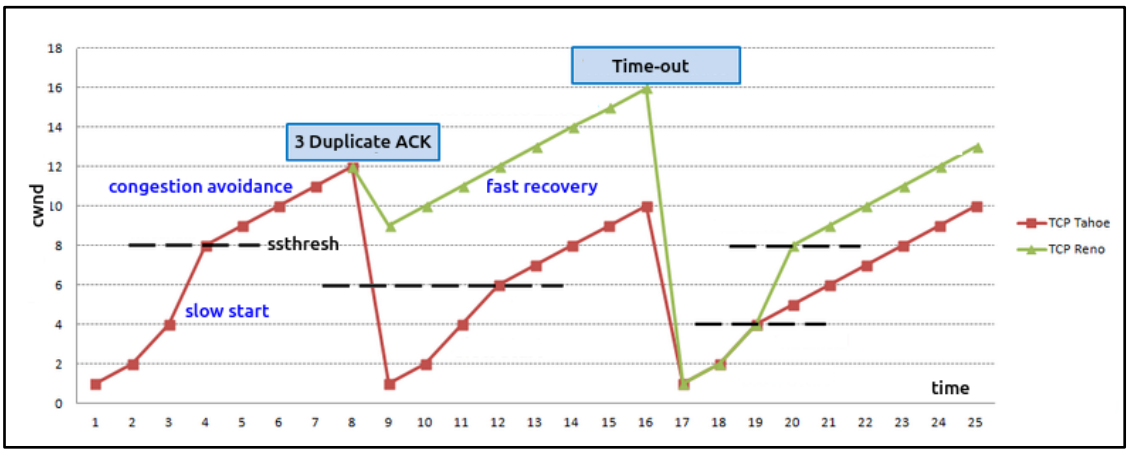
TCP New Reno, memperbaiki TCP Reno dengan menambahkan fase *fast retransmit*. Seperti halnya pada TCP Reno, New Reno masuk ke fase *fast recovery* setelah 3 duplicate ACK. Namun TCP New Reno tidak akan keluar dari fase *fast recovery* sampai semua data yang ada ketika masuk fase *fast recovery* terkirim dengan ditandai diterimanya ACK. Jadi New Reno memperbaiki masalah penurunan nilai *cwnd* berulang kali, yang terjadi pada TCP Reno

Untuk lebih jelasnya fase-fase pada TCP dapat dilihat pada state diagram pada Gambar 1. Sebagai catatan pada TCP Tahoe hanya ada fase *slow start* dan *congestion avoidance*. Pada TCP Reno ditambahkan fase *fast recovery*. Pada TCP New Reno tetap bertahan pada fase *fast recovery* pada untuk menyelesaikan *retransmit*, sementara pada TCP Reno tidak.



Gambar 1. *State Diagram* TCP Tahoe, TCP Reno dan TCP New Reno

Pada gambar 2, dapat dilihat ilustrasi perkembangan cwnd antara TCP Tahoe dan TCP Reno mengalami peristiwa *duplicate* ACK dan *time-out*. Pada waktu n=1 kedua TCP memulai *slow start* sampai dengan *cwnd* sama dengan *ssthresh*, yaitu misalnya 8. Setelah itu kedua TCP masuk ke fase *congestion avoidance*. Misalkan pada n=8 terjadi 3 *duplicate* ACK, maka TCP Tahoe masuk ke fase *slow start*, sementara TCP Reno masuk ke fase *fast recovery*. Nilai kedua *ssthresh* sama yaitu setengah nilai *cwnd* ketika terjadi 3 *duplicate* ACK, yaitu sama dengan 6. Namun nilai *cwnd* berbeda, yaitu sama dengan 1 pada TCP Tahoe dan pada TCP Reno bernilai 9. Berada pada fase *slow start*, maka *cwnd* pada TCP Tahoe naik secara eksponensial. Sedangkan pada TCP Reno naik secara linear dikarenakan pada fase *fast recovery*. Semisal pada n=8 terjadi *time-out*, maka kedua TCP sama-sama masuk pada fase *slow start*, dengan catatan nilai *ssthresh* yang berbeda. Perbedaan terjadi karena nilai *cwnd* pada waktu terjadi *time-out* pada TCP Tahoe adalah 10, sementara pada TCP Reno adalah 16. Skenario ini menghasilkan kecepatan transfer TCP Reno lebih baik dibandingkan TCP Tahoe.



Gambar 2. Ilustrasi perbedaan *cwnd* antara TCP Tahoe dan TCP Reno

III. METODE PENELITIAN

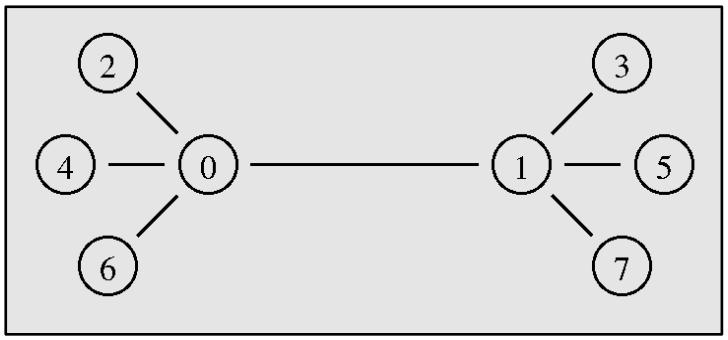
Pada penelitian ini, penulis dipersiapkan serangkaian percobaan dengan skenario untuk membuat gangguan pada ACK ketika terjadi pengiriman paket TCP. Skenario diujicobakan melalui sebuah simulator jaringan. Ns-2 adalah sebuah *discrete event simulator* yang ditujukan untuk penelitian. Simulator ini menawarkan fitur yang luas untuk mensimulasikan layanan TCP pada jaringan kabel dan nirkabel. Dengan menggunakan ns-2, pengguna dapat mensimulasikan jaringan TCP/IP dengan berbagai model, termasuk Tahoe, Reno, New Reno, SACK, serta beberapa model lainnya (NS2 Documentation, 2014).

Topologi yang pada penelitian ini dapat dilihat pada gambar 3. Ada 7 buah node dimana node N0 terhubung dengan node N2, N4 dan N6 dimana masing-masing link *bandwidth* 100 Mbps *delay* 10 ms. Link dengan *bandwidth* 100 Mbps *delay* 10 ms juga dipakai untuk menghubungkan antara node N1 dengan Node N3, N5 dan N7. Besaran nilai *bandwidth* dipilih setara dengan kabel *fast ethernet* yang biasa dipakai pada kondisi nyata pada saat ini. Sedangkan kapasitas *bandwidth* yang lebih kecil dipasang pada *link* penghubung antara node N0 dan N1 dibanding dengan link yang lain. Hal ini dilakukan agar menyebabkan terjadinya efek sumbatan dan mengakibatkan kemacetan pada trafik dari N0 ke N1 dan sebaliknya, ketika ada aliran data dari N2 atau N4 atau N6 ke arah N3 atau N5 atau N6 serta arah sebaliknya. Untuk itu dipilih besaran bandwidth 10 Mbps dan delay 40 ms, dimana ini setara dengan kabel *ethernet*.

TCP baik Tahoe, Reno ataupun New Reno dialirkan dari node N2 ke N3 dan atau dari node N6 ke N7. Sementara dari node N5 ke N4 dialirkan trafik untuk mengganggu ACK dari trafik TCP yang dialirkan. Aliran trafik jenis UDP dari node N5 ke N4, mengakibatkan antrian pada node N1. Adanya paket yang berada dalam antrian menyebabkan penambahan waktu pemrosesan agar paket-paket tersebut dapat keluar dari antrian. Bahkan, jika jumlah paket yang harus antri melebihi kapasitas antrian yang telah ditentukan, yaitu 160, maka paket-paket tersebut akan dihapus (*drop*). Paket yang di *drop* bisa jadi termasuk ACK yang dikirimkan oleh node penerima aliran TCP, yaitu node N3 ataupun N7. Pada akhirnya gangguan ACK tersebut dapat mempengaruhi kinerja pengiriman paket TCP dari node N2 ataupun N4.

Dari hasil percobaan untuk skenario ini, TCP mengalir secara maksimal memenuhi link yang tersedia pada node N0 dan N1 pada waktu sekitar 55 detik (lihat gambar 6). Hal ini ditandai I mencapai nilai puncak pada saat menjalani fase *slow start*. Dari gambar 6 pula terlihat bahwa TCP telah mencapai konvergen pada

detik sekitar 110, dimana ditandai dengan pola fase congestion avoidance yang berulang sebelum dan sesudah pada detik 110. Oleh karena itu pada percobaan ini simulasi dijalankan selama 122 detik, dengan TCP mulai mengalir pada detik ke-1 dan berakhir pada detik ke-121, sehingga total waktu aliran TCP adalah 120 detik. Trafik pengganggu, yang bertujuan untuk menyebabkan kemacetan dan menghilangkan ACK akan berjalan sebelum TCP mulai mengalir, yaitu pada detik ke-0.5 dan berakhir setelah TCP selesai, yaitu pada detik ke-121.5. Ukuran paket MSS (Maximum Segment Size) diatur menjadi 1460 byte, sesuai dengan nilai MTU (*Maximum Transmission Unit*) default yang umum digunakan pada jaringan Ethernet. Parameter yang diamati pada percobaan ini meliputi *cwnd, ssthresh*, kumulatif ACK, dan kumulatif *retransmit*.



Gambar 3. Topologi Pengujian

IV. HASIL DAN PEMBAHASAN

A. Hasil

Tabel 1 menunjukkan hasil kinerja aliran TCP Tahoe dan TCP Reno / New Reno ketika diberi gangguan dengan berbagai besaran aliran gangguan ACK. Mulai dari tanpa gangguan, kemudian gangguan dengan aliran 95% dari kapasitas link antara node N0 dan N1. dan tingkat aliran dinaikkan sampai mencapai 150% dari kapasitas. Dari melihat besarnya paket yang di ACK oleh penerima di akhir simulasi, didapatkan jumlah paket total yang berhasil dikirimkan (kumulatif ACK) oleh pengirim.

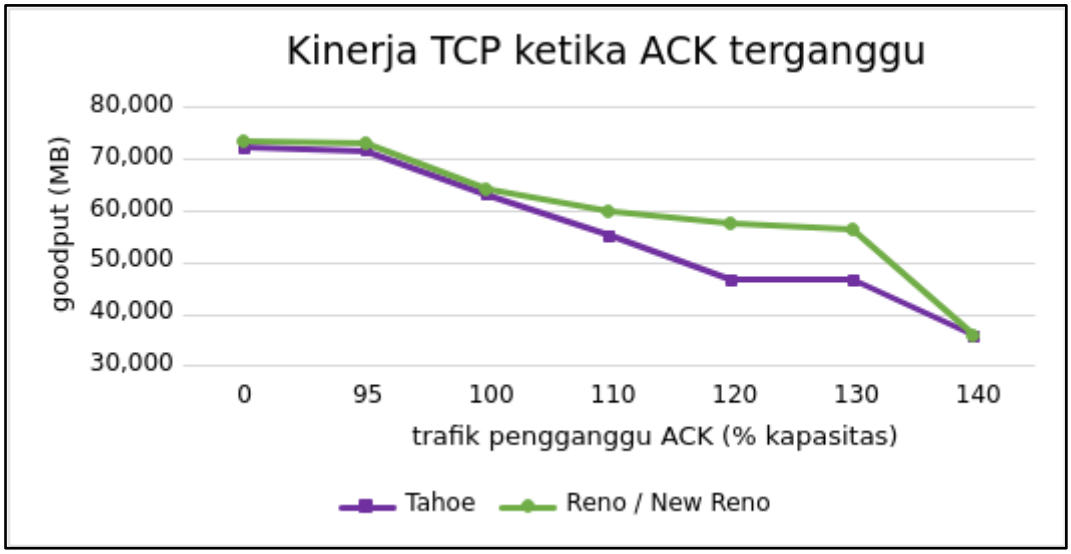
Tabel 1. Hasil simulasi kumulatif ACK yang diperoleh di akhir simulasi (dalam satuan MB) ketika ACK mengalami berbagai macam tingkat gangguan.

	Trafik gangguan (dalam % kapasitas)							
	0	**95**	**100**	**110**	**120**	**130**	**140**	**150**
Tahoe	72,398	71,754	63,148	55,264	46,648	46,648	35,766	71
Reno	73,520	73,134	64,144	59,964	57,661	56,477	35,766	71
New Reno	73,520	73,134	64,144	59,964	57,661	56,477	35,766	71

Hasil pada tabel 1 menunjukkan bahwa sampai dengan tingkat gangguan 95% kinerja TCP tidak ada bedanya dengan kita tanpa gangguan sama sekali (0%). Ini berarti sampai dengan gangguan 95% pada trafik balik, tidak ada ACK yang hilang, sehingga kinerja TCP mengalami penurunan. Penurunan mulai terjadi ketika trafik pengganggu lebih besar atau sama dengan 100% dari kapasitas link.

Pada tabel 1 juga dapat dilihat hasil antara TCP Reno dan TCP New Reno tidak ada perbedaan sama sekali. Ini berarti perbaikan yang diusahakan oleh TCP Reno tidak berjalan pada skenario ini. Hal itu dapat dijelaskan karena TCP New Reno memperbaiki kinerja TCP Reno dengan jalan menambahkan skenario untuk tidak keluar dari dari fase fast retransmit ketika terjadi 3 *duplicate* ACK. Padahal pada pengujian ini gangguan dilakukan dengan jalan menghilangkan ACK yang tidak berhubungan langsung dengan 3 duplicate ACK. Faktor yang mempengaruhi 3 duplicate ACK adalah hilangnya paket bukan hilangnya ACK. Hal ini berarti algoritma perbaikan yang dibuat oleh TCP New Reno tidak pernah terjadi / dijalankan. Oleh sebab itu, pada pembahasan selanjutnya hanya akan dilihat perbandingan kinerja antara TCP Tahoe dan TCP Reno.

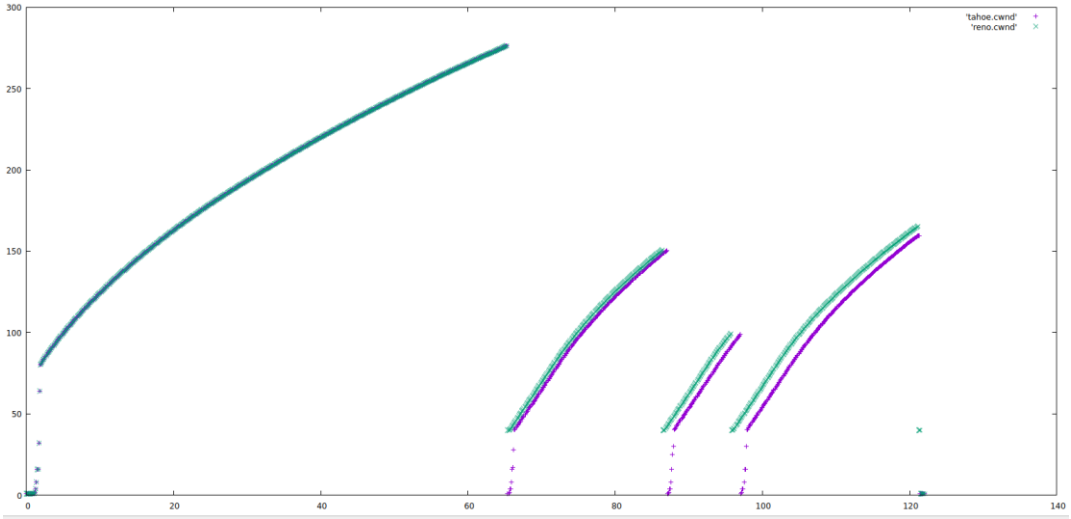
Hasil percobaan pada tabel 1 juga menunjukkan ketika gangguan mencapai 150%, TCP tidak dapat bekerja untuk mengalirkan data. Hal ini terjadi karena hampir seluruh ACK hilang, akibatnya TCP hanya dapat sedikit mengalirkan data saja.



Gambar 5. Grafik hasil pengujian *)

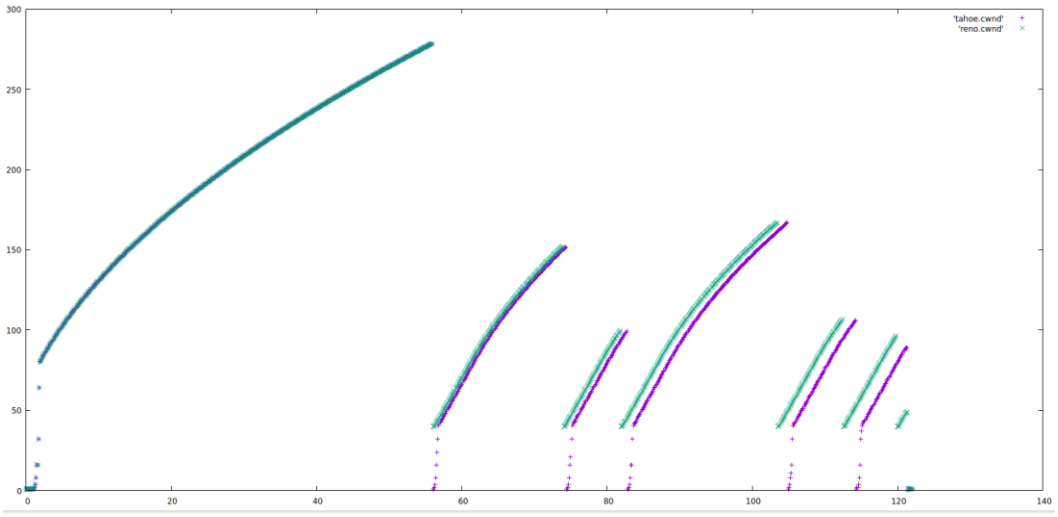
*) TCP Reno dan New Reno dijadikan satu grafik dari tabel 1 menunjukkan hasil sama

Hasil pengujian pada tabel 1, jika disajikan dalam bentuk grafik terlihat pada gambar 5. Dari gambar 5 tersebut, dapat dilihat bahwa sampai dengan tingkat gangguan 100%, tidak ada perbedaan yang berarti antara TCP Tahoe dengan TCP Reno/New Reno. Perbedaan hanya kecil lebih, diakibatkan adanya *time-out* ketika TCP masuk pada fase *congestion avoidance*. Hal ini dikonfirmasi dari grafik cwnd kedua TCP tersebut ketika dijalankan tanpa gangguan yang hasilnya dapat dilihat pada gambar 6. Pada gambar 6 terlihat bahwa pada detik ke 65, 85 dan 95 adanya paket ACK yang hilang. Hal ini menyebabkan TCP Tahoe masuk ke fase *slow start*, sementara itu TCP Reno masuk pada fase *fast recovery*. Hal inilah yang mengakibatkan unjuk kerja TCP Reno/New Reno sedikit lebih baik dibanding dengan TCP Tahoe.



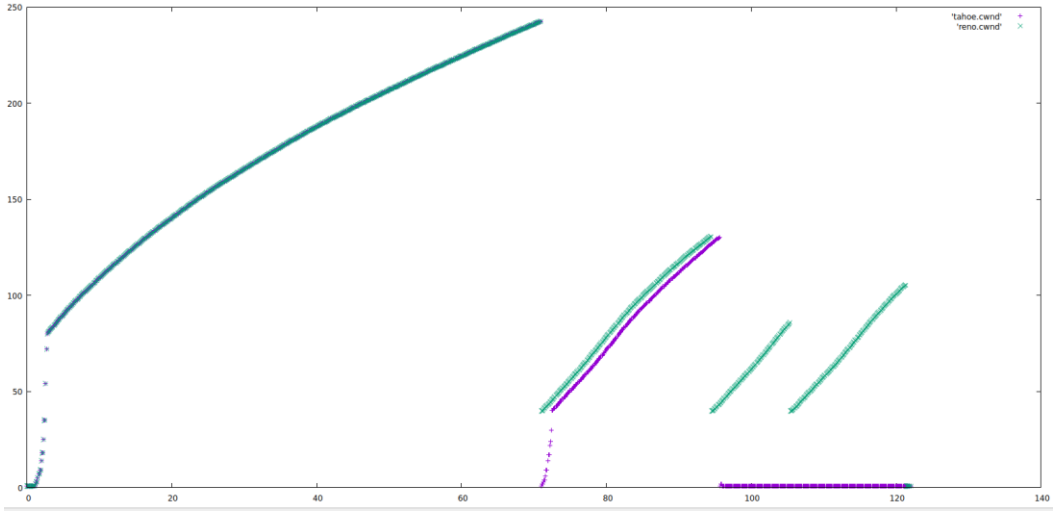
Gambar 6. Grafik *cwnd* TCP Tahoe dan Reno/New Reno ketika tanpa gangguan

Dari tabel 1, terlihat bahwa pada saat trafik pengganggu ACK memakai 95% dari kapasitas link, kinerja semua varian TCP tidak terpengaruh (nilainya sama dengan ketika tingkat gangguan 0%). Adanya sisa kapasitas 5% masih untuk mencukupi ACK yang dikirimkan oleh penerima ke pengirim. Pengaruh gangguan baru mulai terlihat ketika trafik pengganggu memakai 100% kapasitas link. Pada kondisi ini ACK yang dikirimkan oleh penerima mulai ada yang hilang. Dapat dilihat pada gambar 7 hilangnya ACK terjadi lebih cepat terjadi dan secara jumlah juga lebih banyak, jika dibandingkan dengan gambar 6. Pada saat trafik pengganggu memakai 100% kapasitas link, maka terjadi kehilangan ACK pada detik ke 55, 75, 83, 105, dan 113. Konsekuensi banyaknya ACK yang hilang dan terjadinya lebih awal, maka data yang berhasil ditransfer pada trafik dengan gangguan 100% menjadi lebih sedikit dibanding dengan ketika trafik gangguan 95%.



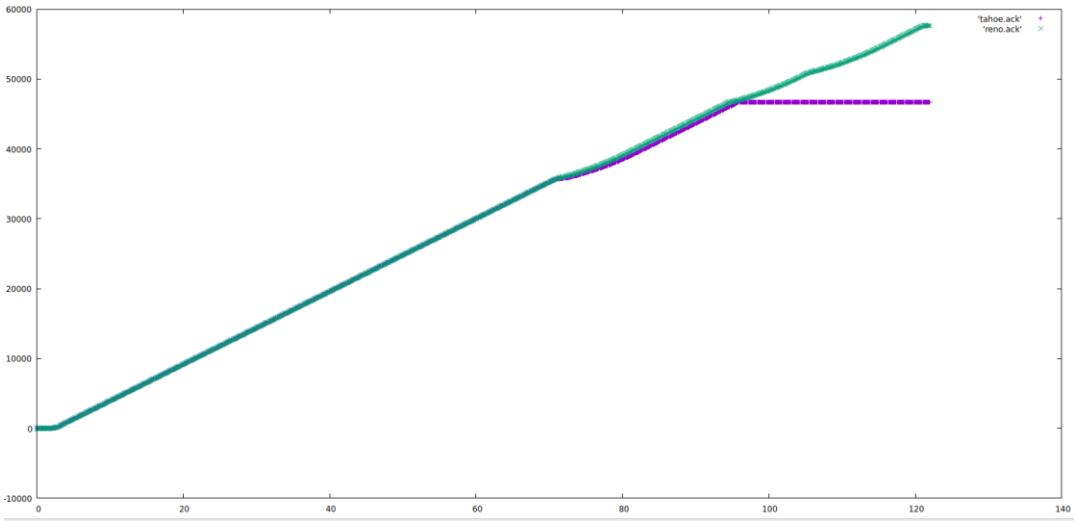
Gambar 7. Grafik *cwnd* TCP Tahoe dan Reno/New Reno dengan trafik pengganggu 100% dari kapasitas link

Hasil pengujian pada gambar 5, menunjukkan bahwa semakin besar gangguan, yaitu pada waktu kapasitas link yang dipakai 110% dan 120% atau 130%, menghasilkan unjuk kerja yang berbeda antara TCP Tahoe dan TCP Reno/New Reno. Ini dapat dikonfirmasi pada grafik *cwnd* dengan tingkat gangguan 120% - 130% seperti tersaji pada gambar 8. Pada gambar 8 terlihat TCP Tahoe pada detik ke 95 tidak bisa keluar dari fase *slow start*. Berarti dalam hal ini TCP Tahoe hanya bisa mengirim sebuah paket dan harus menunggu paket tersebut di-ACK, baru bisa untuk mengirim paket berikutnya.



Gambar 8. Grafik *cwnd* TCP Tahoe dan Reno/New Reno dengan trafik pengganggu 120% - 130% dari kapasitas link

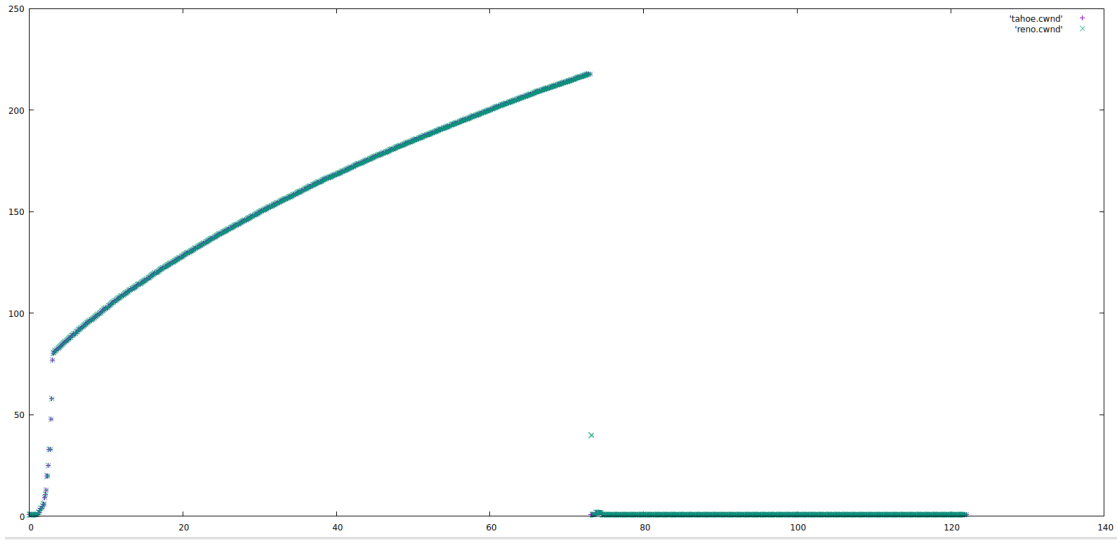
Hal ini diperjelas pada gambar 9 yang menggambarkan kumulatif ACK pada masing-masing TCP dengan tingkat gangguan 120% - 130%. Terlihat jelas pada gambar 9 bahwa sejak detik ke 95, tidak ada peningkatan pada TCP Tahoe.



Gambar 9. Grafik kumulatif ACK TCP Tahoe dan Reno/New Reno dengan trafik pengganggu 120% - 130% dari kapasitas link

Namun pada saat trafik gangguan sebesar 140% dari kapasitas seperti terlihat pada gambar 10, kedua kinerja TCP menunjukkan hasil yang sama. Ini berarti perbaikan TCP Reno/New Reno dengan menambahkan fase fast recovery tidak berguna. Gangguan yang begitu besar mengakibatkan hilangnya ACK yang begitu banyak ketika setelah detik ke 73. Dapat dilihat pada gambar 10, semua varian TCP terus menerus berada pada fase slow start setelah pada detik ke 73. Hal ini tentu saja membuat kedua TCP tidak dapat mengirimkan data dengan lancar.

Ketika trafik gangguan ditingkatkan nilainya menjadi 150%, maka hampir tidak paket yang bisa dikirimkan oleh kedua TCP. Hal ini dapat dilihat pada tabel 1, bahwa selama 2 menit semua varian TCP hanya mampu mengirimkan 71 paket saja. Oleh sebab itu, pada penelitian ini peningkatan tidak trafik gangguan tidak ditingkatkan lagi.



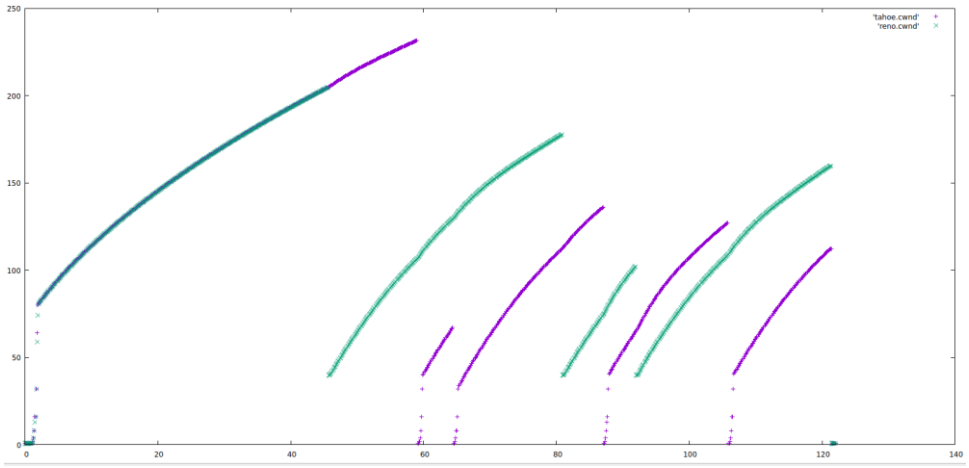
Gambar 10. Grafik *cwnd* TCP Tahoe dan Reno/New Reno dengan trafik pengganggu 140% dari kapasitas link

Untuk mengkonfirmasi hasil-hasil di atas, maka dilakukan beberapa pengujian lagi dengan skenario menjalankan TCP Tahoe dan TCP Reno/New Reno secara bersamaan. TCP Tahoe dialirkan dari node N2 ke N3, sedangkan TCP Reno/New Reno dialirkan dari node N6 ke N7. Trafik pengganggu ACK dari node N5 ke N4. Dipilih beberapa % trafik pengganggu yang terlihat secara signifikan mempengaruhi hasil, yaitu tanpa gangguan sama sekali (0%), 130% dan 140%. Hasil dari percobaan ini dapat dilihat pada tabel 2 di bawah ini.

Tabel 2. Hasil simulasi kumulatif ACK TCP Tahoe dan TCP Reno/New Reno yang jalan bersamaan pada di akhir simulasi (dalam satuan MB) ketika ACK mengalami berbagai macam tingkat gangguan.

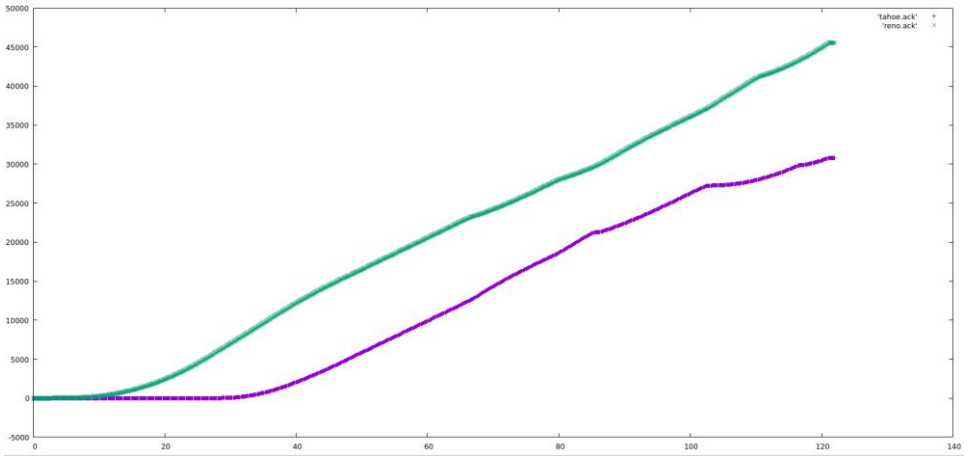
	Trafik gangguan (dalam % kapasitas)		
	0	120-130	140
Tahoe	49,204	30,817	43,260
Reno	49,463	45,529	43,288

Hasil tabel 2 di atas menunjukkan ketika tidak ada gangguan unjuk kerja kedua TCP relatif sama. Dapat dilihat pada gambar 11, nilai *cwnd* kedua TCP mirip-mirip, hanya berbeda dari sisi waktu saja. Tidak adanya perbedaan kinerja juga ditunjukkan ketika banyak ACK yang hilang akibat trafik gangguan cukup besar, yaitu 140%.

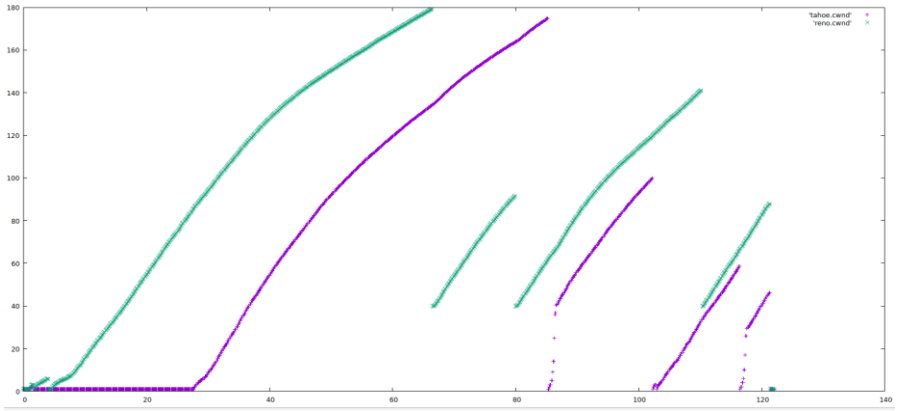


Gambar 11. Grafik *cwnd* TCP Tahoe dan Reno berjalan bersamaan tanpa trafik pengganggu

Namun pada saat trafik pengganggu memakai 130% dari kapasitas link, TCP Reno/New Reno menunjukkan keunggulan dengan berhasil mentransfer data lebih banyak (lihat gambar 12). Pada tingkat gangguan ini, ada sebagian ACK yang hilang mengakibatkan TCP Reno/New Reno lebih banyak masuk pada fase *fast recovery*, hal ini terlihat pada grafik *cwnd* pada gambar 13.



Gambar 12. Grafik kumulatif ACK TCP Tahoe dan Reno bersamaan dengan trafik pengganggu 140% dari kapasitas link



Gambar 13. Grafik *cwnd* TCP Tahoe dan Reno berjalan bersamaan dengan trafik pengganggu 140% dari kapasitas link

B. Pembahasan

Hasil-hasil penelitian ini melengkapi penelitian sebelumnya tentang pengaruh trafik pengganggu terhadap kinerja TCP. Pada penelitian-penelitian sebelumnya lebih banyak membahas pengaruh ketidak hadiran ACK yang diakibatkan oleh hilangnya paket karena adanya trafik gangguan pada jalur pengiriman paket. Sedangkan pada penelitian ini melihat pengaruh hilangnya ACK akibat trafik gangguan pada jalur pengiriman ACK itu sendiri. Pada dunia nyata, pembanjiran trafik (*flooding*) terjadi ketika menerima serangan siber dari luar ke arah jaringan kita.

Pada kondisi ketidak hadiran ACK akibat hilangnya paket, perbaikan algoritma TCP *congestion controll* Reno terhadap TCP Tahoe dengan menambahkan fase *fast recovery* dapat berfungsi dengan baik ketika terjadi gangguan hilangnya ACK. Demikian juga perbaikan algoritma TCP New Reno berhasil memperbaiki kinerja TCP Reno (Hernawan, 2022). Sementara pada ketidak hadiran ACK yang diakibatkan oleh trafik gangguan pada jalur ACK perbaikan TCP New Reno tidak berpengaruh. Hal ini karena pada kasus hilangnya ACK tidak memicu munculnya problem *retransmit* pada TCP Reno, dimana ini menjadi pertimbangan utama perbaikan yang ada di TCP New Reno.

Trafik gangguan pada jalur pengiriman paket sebesar 20% dari kapasitas link sudah menurunkan kinerja TCP, bahkan TCP bisa dikatakan tidak dapat melakukan fungsinya ketika trafik gangguan sebesar 80% (Hernawan A., 2023). Pada penelitian ini ditemukan bahwa gangguan trafik pada jalur ACK tidak banyak berpengaruh. Hal ini dapat dilihat bahawa kinerja TCP ketika tanpa gangguan dan ketika trafik gangguan ditingkatkan sampai 95% dari kapasitas link, kinerja sama saja. Hal ini bisa dipahami bahwa Ukuran ACK yang kecil masih bisa memanfaatkan sisa kapasitas, sehingga trafik ACK tidak terganggu. Kinerja TCP baru mulai menurun ketika trafik pengganggu memakai 100% dari kapasitas link dan TCP menjadi tidak berfungsi ketika trafik pengganggu sebesar 150%.

V. SIMPULAN

Gangguan pada ACK menjadikan perbaikan algoritma TCP *congestion controll* oleh TCP New Reno tidak berjalan. Ketidak hadiran ACK yang diakibatkan oleh ACK yang hilang, bukan karena paket yang tidak sampai, mengakibatkan tidak adanya problem retransmit pada TCP Reno. Maka perbaikan TCP New Reno dengan menambahkan fase *fast retransmit* menjadi tidak berguna.

Pemakaian trafik sampai dengan 95% kapasitas link, tidak menyebabkan turunnya kinerja TCP dibanding ketika tanpa gangguan. Ukuran ACK yang kecil masih bisa memanfaatkan sisa kapasitas, sehingga trafik ACK tidak terganggu. Kinerja mulai menurun ketika trafik pengganggu memakai di atas 110% dari kapasitas. TCP Tahoe mengalami penurunan kinerja lebih buruk dari TCP Reno/New Reno. TCP Tahoe masuk ke fase *slow start*, sementara TCP Reno/New Reno mendapatkan keuntungan dengan adanya fase *fast recovery*. Namun jika gangguan terlampau besar, yaitu memanfaatkan link lebih dari 140%, maka TCP Reno/New Reno tidak dapat mendapatkan keunggulan dengan adanya fase *fast recovery*. Baik TCP Reno / New Reno berperilaku sama dengan TCP Tahoe. Hal ini bisa dipahami bahwa pada tingkat gangguan yang besar, jumlah ACK yang hilang cukup banyak, sehingga TCP Reno/New Reno tidak dapat keuntungan pada kondisi ini.

DAFTAR PUSTAKA

Allman M., Paxson V., Blanton E. (2009), TCP Congestion Control, ietf.org, diunduh di https://www.rfc-editor.org/info/rfc5681 tanggal 20 November 2022

Clark D.D and Hoe J. (1995), Start-up Dynamics TCP's Congestion Control and Avoidance Schemes Technical Report, Internet End-toEnd Research Group, Massachusetts Institute of Technology.

Fahmi H., (2018), Analisis Qos (Quality of Service) Pengukuran Delay, Jitter, Packet Lost Dan Throughput Untuk Mendapatkan Kualitas Kerja Radio Streaming Yang Baik. Jurnal Teknologi Informasi Dan Komunikasi, 7(2), p98–105.

Fahrudy D. & Sugiantoro B. (2022), Analisis Perbandingan Kinerja TCP Vegas dan TCP New Reno Menggunakan Antrian DropTail, ISKA (Jurnal Informatika Sunan Kalijaga) Vol. 7, No. 1

Fajri M. (2021), Studi Network Congestion Dengan TCP Tahoe, Jurnal Ilmiah FIFO, Vol 13, No 2, ISSN 2502-8332

Febriansah & Ibnu M.R. (2020). Analisis Bottleneck dan Bufferbloat pada AQM Droptail, RED dan SFQ di Komunikasi Data TCP Newreno. Jurnal Repositor, 2(9), 1213–1224

Floyd S. (1996), Simulator Test Technical Report, Lawrence Berkeley National Laboratory, One Cyclotron Road, Berkeley

Floyd W, Henderson T (1999), The New-Reno Modification to TCP's Fast Recovery Algorithm, RFC 2582, diunduh di https://www.rfc-editor.org/rfc/rfc2582 tanggal 20 November 2022

Forouzan B.A (2010), TCP/IP Protocol Suite Fourth Edition, The McGraw-Hill Companies, Inc.

Kurose James & Ross Keiht (2021), Computer Networking: A Top Down Approach 8ed, Addison-Wesley

Hernawan, A. (2022), Perbandingan Unjuk Kerja TCP Tahoe, Reno, New Reno dan SACK pada Jaringan Kabel, Prosiding Seminar Nasional Sanata Dharma Berbagi 2022, Universitas Sanata Dharma, Yogyakarta, http://e-conf.usd.ac.id/index.php/USDB

Hoe J. (1996), Improving the Start-up Behavior of a Congestion Control Scheme for TCP, SIGCOMM Symposium on Communication Architecture and Protocols

Idwan H., Arif T.Y. & Munadi R. (2018), Analisis Round Trip Time (RTT) Terhadap Kinerja Jaringan Wireless TCP New Reno, KITEKTRO: Jurnal Online Teknik Elektro Vol.3 No.3 2018: 32-40, e-ISSN: 2252-7036

Iqbal M. Fitiawan H & Kurniawan D. (2022), Simulasi Kinerja Web Server pada Jaringan LAN (Local Area Network) Kampus Menggunakan NS2 (Network Simulator 2), Jurnal Komputasi, Vol 10, No 2

Jacobson V. (1988), Congestion Avoidance and Control, SIGCOMM Symposium on Communications Architectures and Protocols, pages 314-329, diunduh di ftp.ee.lbl.gov/papers/congavoid.ps tanggal 20 November 2022

Jacobson V. (1990), Modified TCP Congestion Control and Avoidance Algorithms Technical Report, ftp.ee.lbl.gov/email/vanj.90apr30.txt

Julham J., Fachrizal F. & Adam H.A. (2020), Pemanfaatan Mikrokontroler Berbasis Internet Protocol (IP) pada Sistem Kehadiran yang Menggunakan Kartu Identitas, InfoTekJar (Jurnal Nasional Informatika dan Teknologi Jaringan), Vol. 4, No. 2, ISSN 2540-7600

Mathis M. and Mahdavi (1996), Forward Acknowledgement: Refining TCP congestion Control, Symposium on Communication Architecture and Protocols

Mathis M., Mahdavi J., Floyd S. and Romanov A. (1996), TCP Selective Acknowledgment Options, Internet Working Progress

Maulana A.N. & Niati U. (2019), Performance Evaluation of TCP New Reno and TCP Vegas to Avoid Congestion over LTE, Selendang Mayang, Volume 5, Nomor 2, Agustus 2019, ISSN: 2442-7845, e-ISSN: 2620-3332

NS2 Editor (2022), NS2 Documentation, diunduh di www.isi.edu/nsnam/ns/ tanggal 1 Desember 2022

Pamungkas, G. W., Yahya, W., & Nurwarsito, H. (2018). Analisis Perbandingan Kinerja TCP Vegas Dan TCP New Reno Menggunakan Antrian Random Early Detection Dan Droptail. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK), Vol 2 No 10, Oktober 2018

Salman S., Ginting J.G.A. & Wahyuningrum R.D. (2021), Analisis Unjuk Kerja TCP Window Size 64k Menggunakan Algoritma TCP New Reno pada Jaringan Wired dan Wireless, RESISTOR (Elektronika Kendali Telekomunikasi Tenaga Listrik Komputer), Vol: 4, No: 1, e-ISSN : 2621-9700, p-ISSN : 2654-2684

Sunanto & Jabar M.A. (2019), Kendali Peralatan Listrik Jarak Jauh Menggunakan Protokol TCP/IP, JURNAL FASILKOM, Volume 9 No.2, ISSN : 2089-3353