



۸۱۰۱۰۱۴۹۰

۸۱۰۱۰۱۵۲۶

شبکه‌های کامپیوتری
کسری کاشانی، مرضیه موسوی



پروژه ۲ فاز اول

۱ مقدمه

در این پروژه، به مقایسه‌ی عملکرد پروتکل‌های TCP Tahoe، TCP Reno و TCP Vegas در NS-2 بر اساس سه معیار Throughput، RTT و Packet Loss پرداختیم.

۲ هدف و صورت مسئله

هدف اصلی این پروژه، پیاده‌سازی و شبیه‌سازی سه گونه‌ی TCP شامل Tahoe، Reno و Vegas و مقایسه‌ی رفتار آن‌ها روی یک توپولوژی مشخص است. طبق صورت پروژه، این مقایسه باید بر اساس سه متریک انجام شود: Throughput، RTT و نرخ از دست‌رفتن بسته‌ها یعنی Packet Loss، و خروجی نهایی نیز شامل سه نمودار (هر کدام با ۶ خط: ۲ جریان برای هر ۳ نوع TCP) خواهد بود.

۳ محیط اجرا و ابزار

شبیه‌سازی در Ubuntu و با NS-2.35 انجام شد. مدل شبیه‌سازی NS-2 رویداد-محور (event-driven) است و زمان بر اساس صف رویدادها پیش می‌رود. اسکریپت توپولوژی و سناریو با OTcl نوشته شد و استخراج داده‌ها و رسم نمودارها با ابزارهای awk، bash، gnuplot و python (برای نمودار میله‌ای) انجام شد.

۴ شرایط و الزامات دقیق صورت پروژه

در این بخش، دقیقاً مواردی که در صورت پروژه الزام شده‌اند لیست می‌شوند و در ادامه نشان داده می‌شود که چگونه رعایت شده‌اند:

۱.۴ توپولوژی و جریان‌ها

- وجود دو جریان دائمی: یکی از گره ۱ به ۵ و دیگری از گره ۲ به ۶، با ارسال مداوم داده (همیشه داده برای ارسال وجود دارد).
- تأخیر لینک‌های دارای delay variable باید به صورت تصادفی بین ۵ تا ۲۵ میلی ثانیه در نظر گرفته شود.
- اندازه صف در روترها برابر ۱۰ بسته باشد.
- مقدار TTL برابر ۶۴ تنظیم شود.
- اندازه بسته TCP برابر پیش فرض NS2 یعنی ۱۰۰۰ بایت باقی بماند.
- استفاده از IPv4 برای لایه شبکه.

۲.۴ تنظیمات شبیه‌سازی

- شبیه‌سازی باید ۱۰ مرتبه اجرا شود و هر بار مدت زمان شبیه‌سازی ۱۰۰۰ ثانیه باشد؛ سپس میانگین داده‌ها گزارش شود.
- فرض شود جریان‌ها دائما در حال ارسال هستند و از دست رفتن بسته روی لینک رخ نمی‌دهد (از دست رفتن تنها به دلیل محدود بودن بافر روترها است).
- ظرفیت یا پهنای باند لینک‌ها ثابت باقی بماند.
- برای پارامترهایی مانند initial window و max window از مقادیر پیش فرض NS2 استفاده شود (یا اگر چیزی پیش فرض نبود، پیش فرض انتخاب شده توضیح داده شود).

۵ ساختار پروژه و فایل‌ها

این پروژه به صورت ماژولار پیاده‌سازی شده و فایل‌ها نقش‌های مشخصی دارند:

۱.۵ اسکریپت اصلی شبیه‌سازی

- tcl/ca_tcp.tcl: تعریف توپولوژی، لینک‌ها، صف، TTL، دو جریان، فعال‌سازی گونه‌ی TCP (یکی از Tahoe/Reno/Vegas)، تولید فایل‌های trace لازم.

۲.۵ استخراج داده از خروجی trace

- `scripts/extract_throughput.awk`: استخراج گذردهی بر حسب زمان (معمولا مبتنی بر بایت یا بسته‌ی دریافت‌شده در مقصد و تبدیل به Mbps در بازه‌های زمانی).
- `scripts/extract_rtt.awk`: استخراج RTT بر حسب زمان از فایل‌های مرتبط با نمونه‌های RTT.
- `scripts/extract_loss.awk`: استخراج تعداد Drop بر ثانیه (یا نرخ از دست‌رفتن بر حسب زمان) از روی رخ داده‌های drop.
- `scripts/avg10.awk`: میانگین‌گیری ۱۰ اجرای مستقل برای تولید فایل‌های avg.

۳.۵ اجرای خودکار و ساخت داده‌های نهایی

- `scripts/run_all.sh`: اجرای ۱۰ مرتبه شبیه‌سازی برای هر نوع TCP و تولید فایل‌های trace و خروجی‌های میانی.
- `scripts/build_data.sh`: فراخوانی awkها برای تولید فایل‌های داده در مسیر `data/` و سپس میانگین‌گیری ۱۰ اجرای مستقل.

۴.۵ رسم نمودارها

- فایل‌های gnuplot برای نمودارهای اصلی ۶-خطی:

`plots/plot_throughput.gp` □

`plots/plot_rtt.gp` □

`plots/plot_loss.gp` □

- نسخه‌های zoom از نمودارهای اصلی (۰ تا ۵۰ ثانیه):

`plots/plot_throughput_zoom50.gp` □

`plots/plot_rtt_zoom50.gp` □

`plots/plot_loss_zoom50.gp` □

- تولید ۹ نمودار تفکیک‌شده (هر TCP جداگانه برای هر متریک) و ۹ نمودار zoom برای آن‌ها:

```

scripts/plot_split_all.sh □
scripts/plot_split_zoom50.sh □
plots/plot_one_*.gp □
plots/plot_one_*_zoom50.gp □

```

۵.۵ نمودار میله‌ای نرخ اتلاف متوسط

- `scripts/make_loss_bar_data.sh`: تولید فایل داده‌ی خلاصه‌ی نرخ اتلاف متوسط هر TCP برای دو جریان.
- `scripts/plot_loss_bar.py`: رسم نمودار میله‌ای Average Packet Loss Rate (پس از نصب `.(matplotlib`
- `plots/plot_loss_bar.gp`: نسخه‌ی جایگزین با `gnuplot` (در صورت نیاز).

۶ روش انجام کار

این بخش، مسیر رسیدن به خروجی‌ها را مرحله به مرحله توضیح می‌دهد:

۱.۶ مرحله ۱: آماده‌سازی محیط

- اطمینان از در دسترس بودن فرمان `ns` در `PATH` یا اجرای مستقیم باینری `NS2`.
- آماده‌سازی پوشه‌های پروژه: `tcl/`, `scripts/`, `trace/`, `data/`, `plots/`.

۲.۶ مرحله ۲: اجرای شبیه‌سازی ۱۰ مرتبه برای هر نوع TCP

طبق الزام صورت پروژه، برای هر یک از Tahoe/Reno/Vegas شبیه‌سازی ۱۰ بار و هر بار به مدت ۱۰۰۰ ثانیه اجرا می‌شود. این کار با `scripts/run_all.sh` انجام شد و فایل‌های `trace` به شکل کلی زیر تولید شدند: `trace/Tahoe_run1.tr`, ... , `trace/Tahoe_run10.tr` (و مشابه برای Reno و Vegas)، همچنین فایل‌های `trace RTT` مرتبط با هر جریان.

۳.۶ مرحله ۳: استخراج متریک‌ها و میانگین‌گیری ۱۰ اجرا

در این مرحله از فایل‌های trace، سه متریک استخراج و به بازه‌های زمانی (ثانیه‌ای) تبدیل شد، سپس با avg10.awk میانگین ۱۰ اجرای مستقل محاسبه و در data/*_avg.dat ذخیره شد.

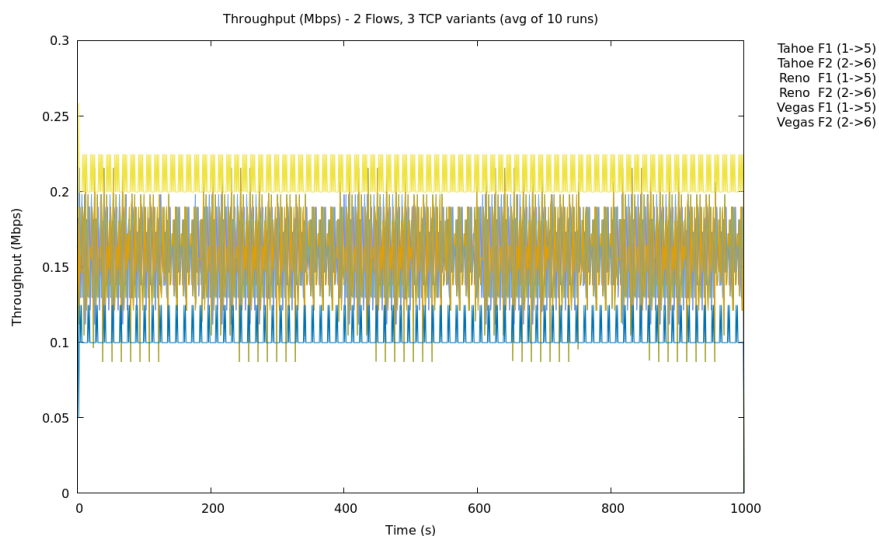
۴.۶ مرحله ۴: تولید نمودارهای موردنیاز

طبق صورت پروژه، سه نمودار اصلی با ۶ خط تولید شد (۲ جریان برای هر ۳ پروتکل). سپس برای خوانایی بیشتر، ۹ نمودار تفکیک‌شده (هر TCP جداگانه) و نسخه‌های zoom ۰ تا ۵۰ ثانیه برای هر کدام ساخته شد.

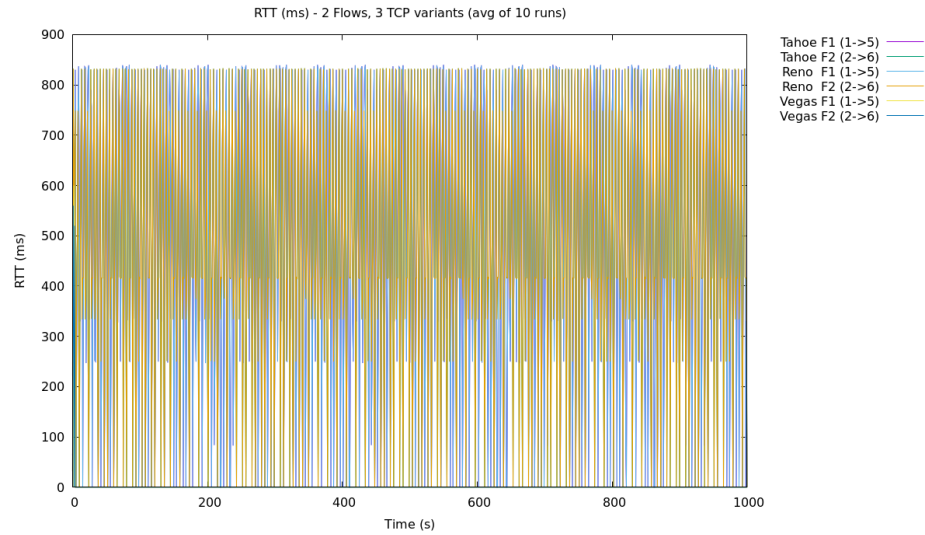
۷ نتایج و نمودارها

در این بخش، عکس تمام نمودارهای خروجی قرار دارند:

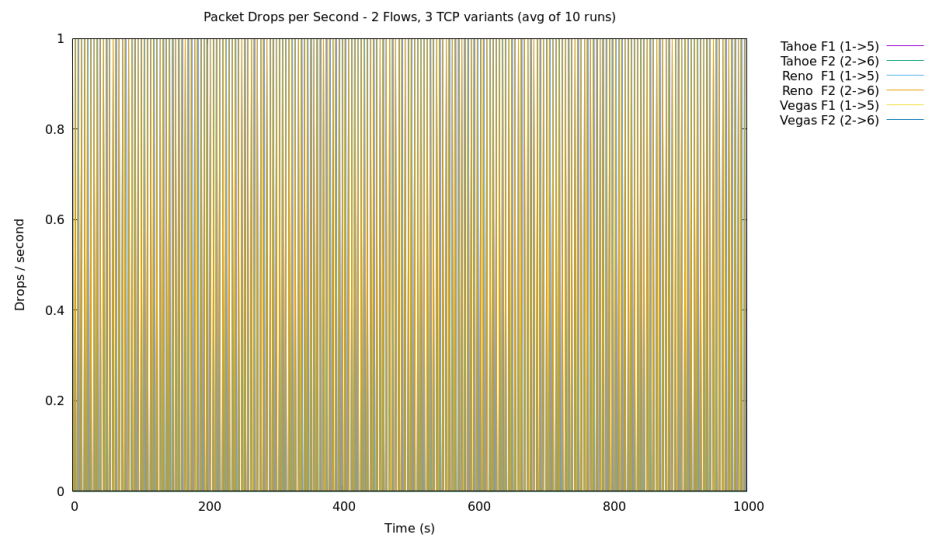
۱.۷ نمودارهای اصلی ۶-خطی (کل بازه)



شکل ۱: نمودار throughput ۶-خطی، ۰ تا ۱۰۰۰ ثانیه

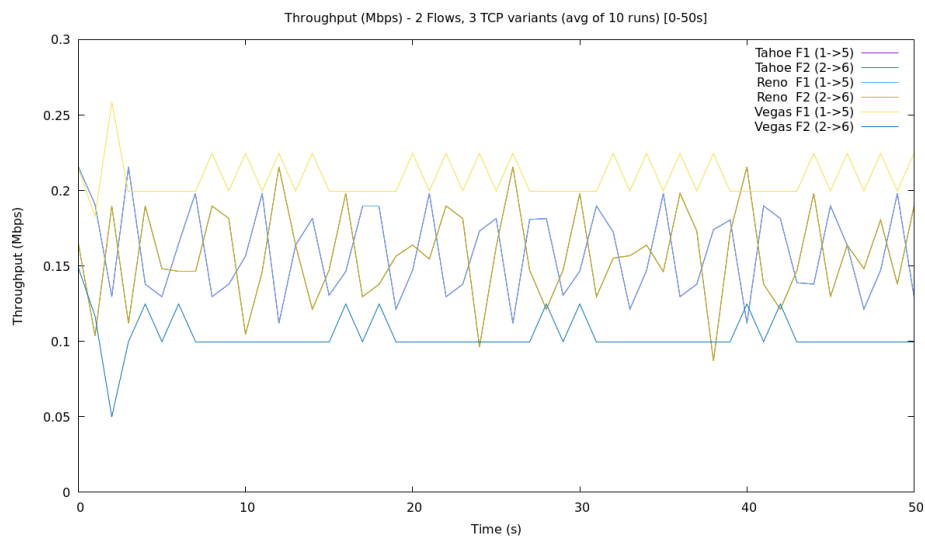


شکل ۲: نمودار RTT ۶-خطی، ۰ تا ۱۰۰۰ ثانیه

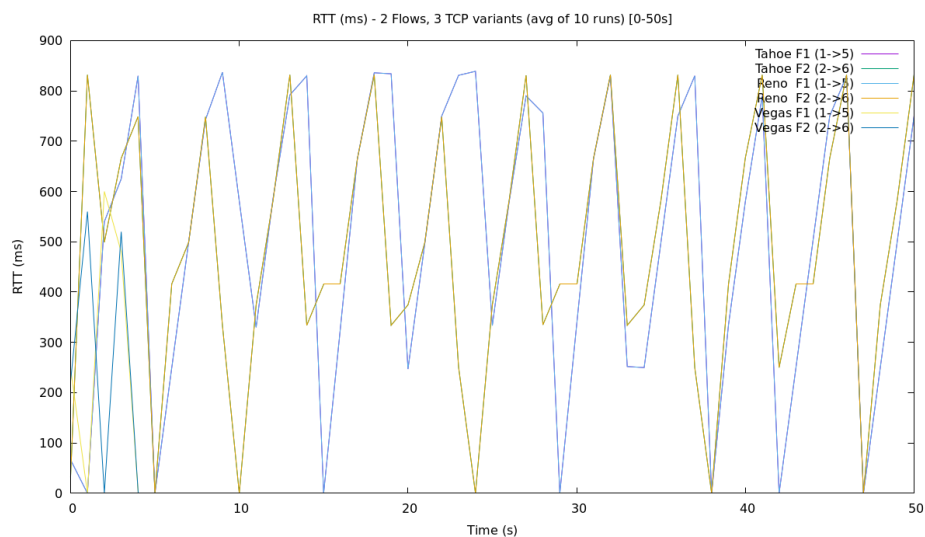


شکل ۳: نمودار loss ۶-خطی، ۰ تا ۱۰۰۰ ثانیه

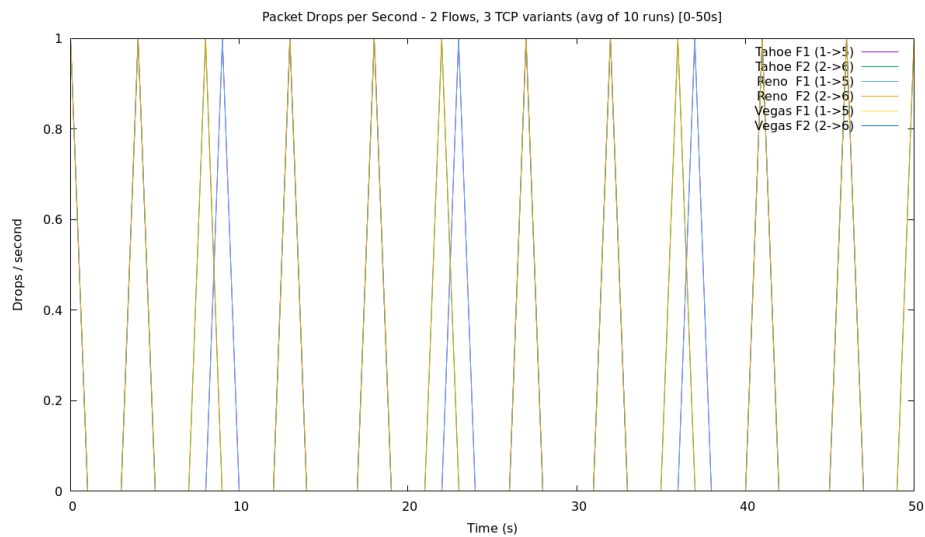
۲.۷ نمودارهای اصلی ۶-خطی زوم‌شده (۰ تا ۵۰ ثانیه)



شکل ۴: نمودار throughput ۶-خطی زوم‌شده، ۰ تا ۵۰ ثانیه

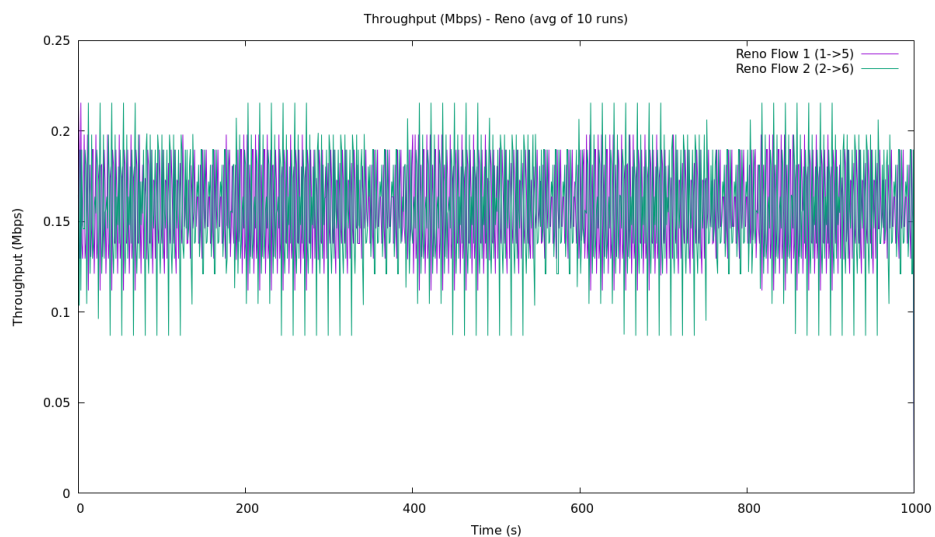


شکل ۵: نمودار RTT ۶-خطی زوم‌شده، ۰ تا ۵۰ ثانیه

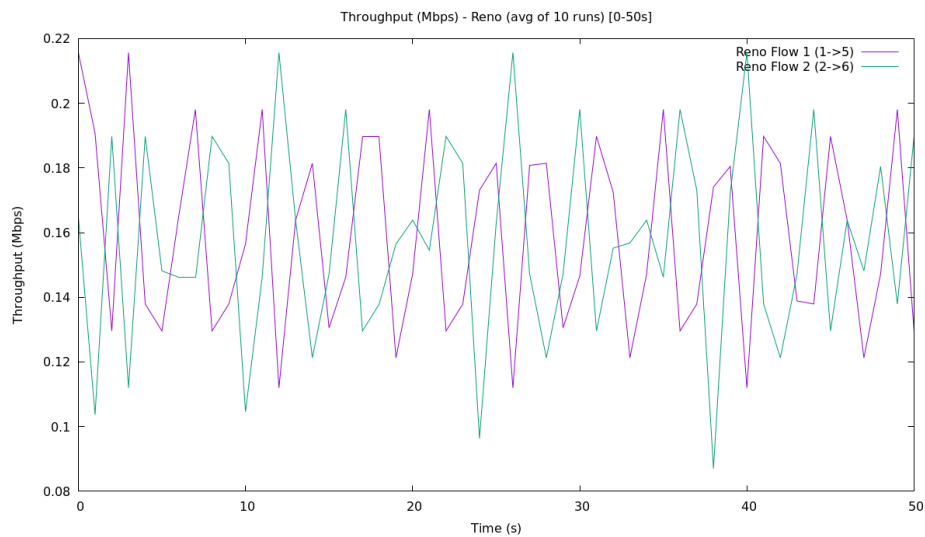


شکل ۶: نمودار loss ۶-خطی زوم‌شده، ۰ تا ۵۰ ثانیه

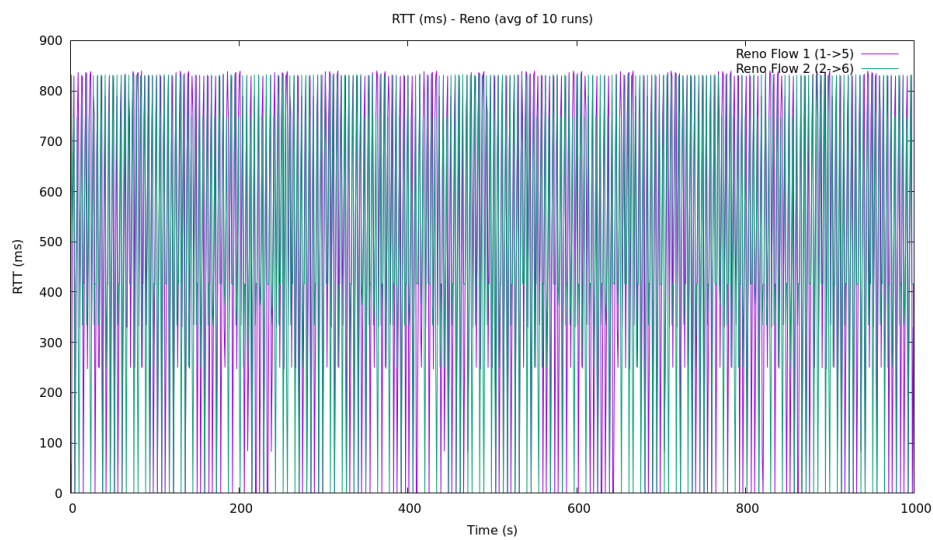
۳.۷ نمودارهای تفکیکی برای Reno (کل و زوم)



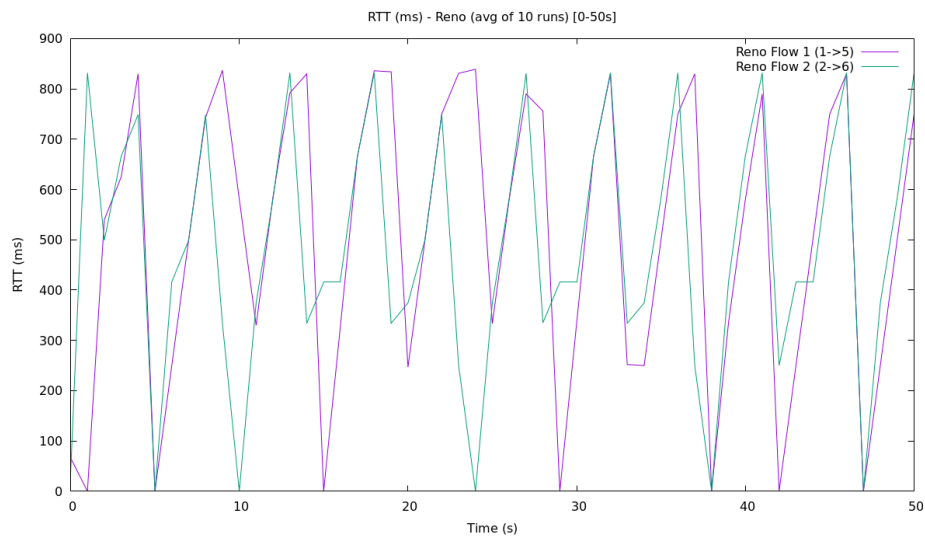
شکل ۷: نمودار throughput برای Reno، ۰ تا ۱۰۰۰ ثانیه



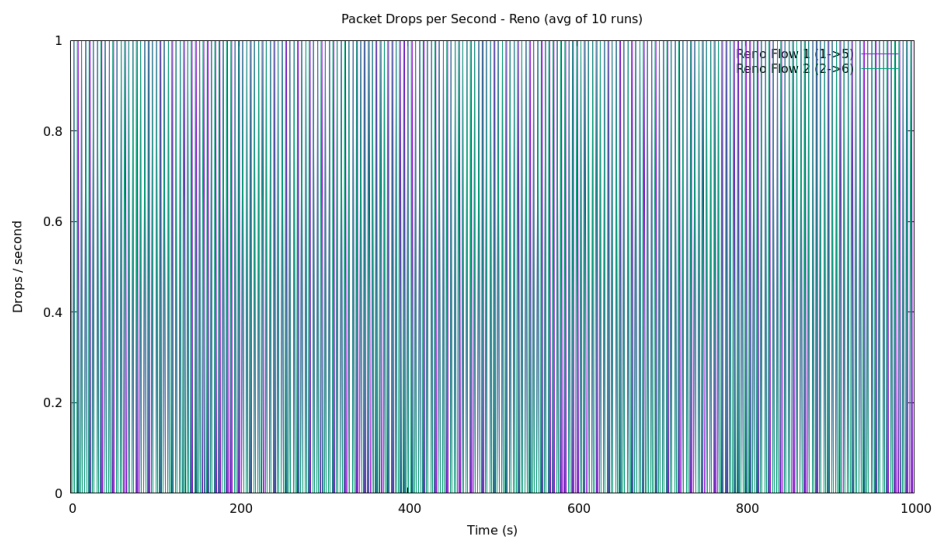
شکل ۸: نمودار throughput برای Reno زوم‌شده، ۰ تا ۵۰ ثانیه



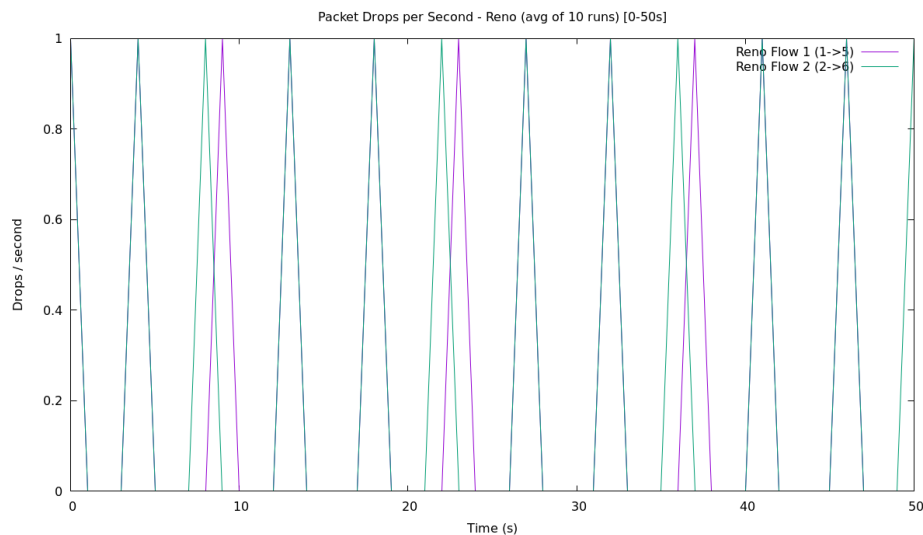
شکل ۹: نمودار RTT برای Reno، ۰ تا ۱۰۰۰ ثانیه



شکل ۱۰: نمودار RTT برای Reno زوم‌شده، ۰ تا ۵۰ ثانیه

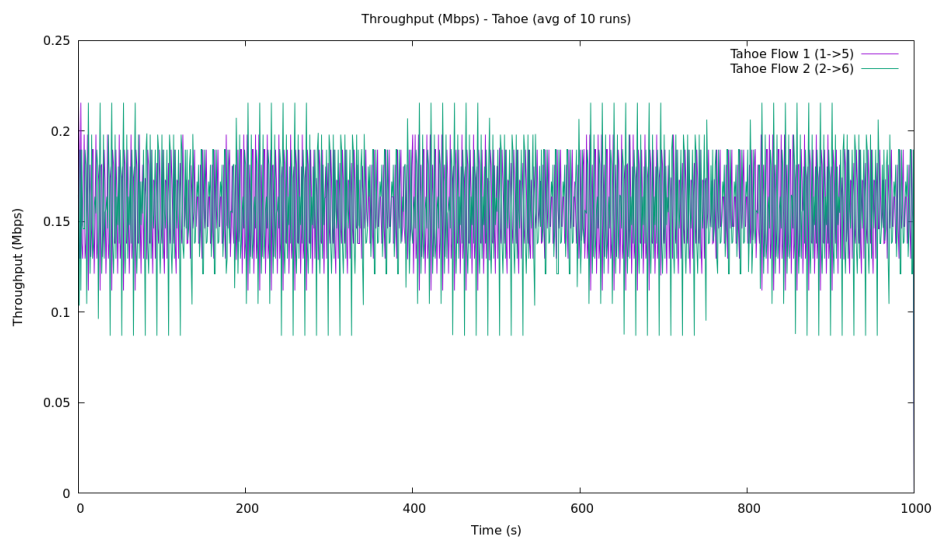


شکل ۱۱: نمودار loss برای Reno، ۰ تا ۱۰۰۰ ثانیه

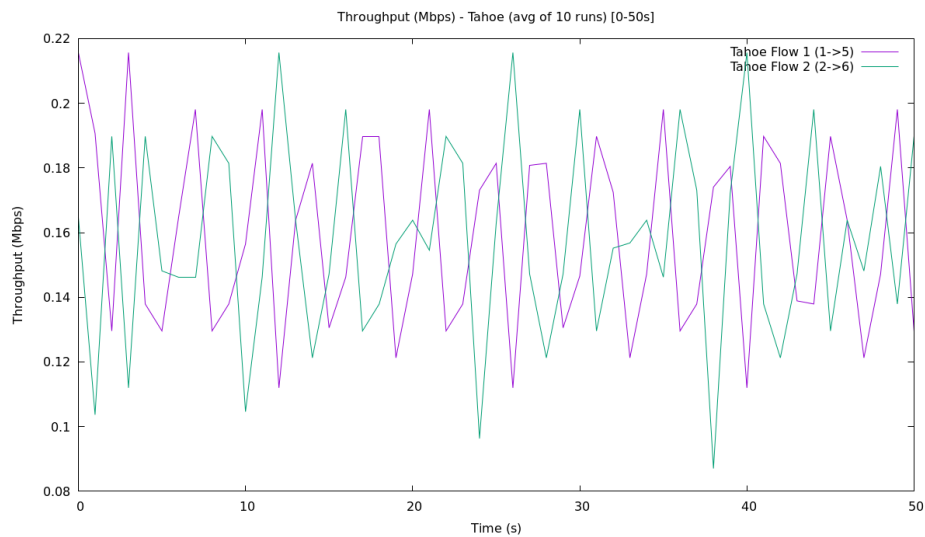


شکل ۱۲: نمودار loss برای Reno زوم‌شده، ۰ تا ۵۰ ثانیه

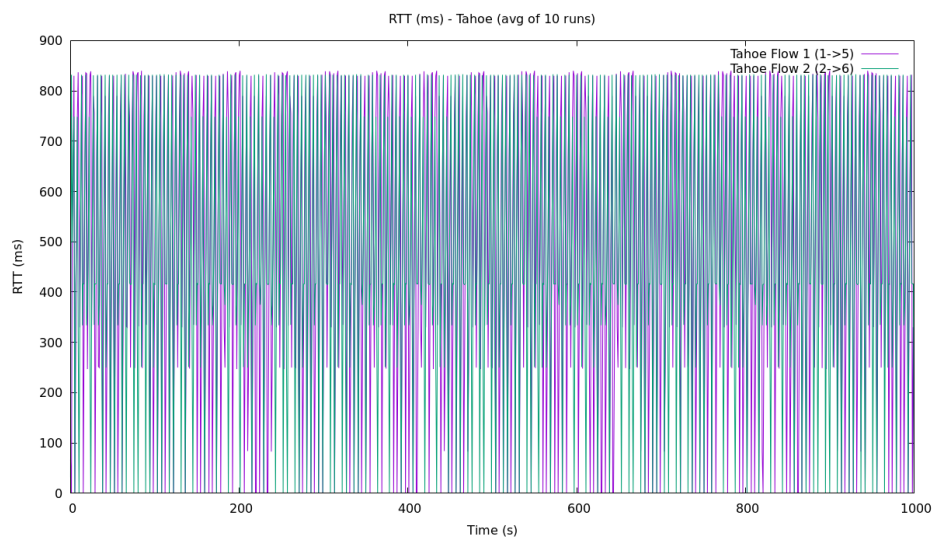
۴.۷ نمودارهای تفکیکی برای Tahoe (کل و زوم)



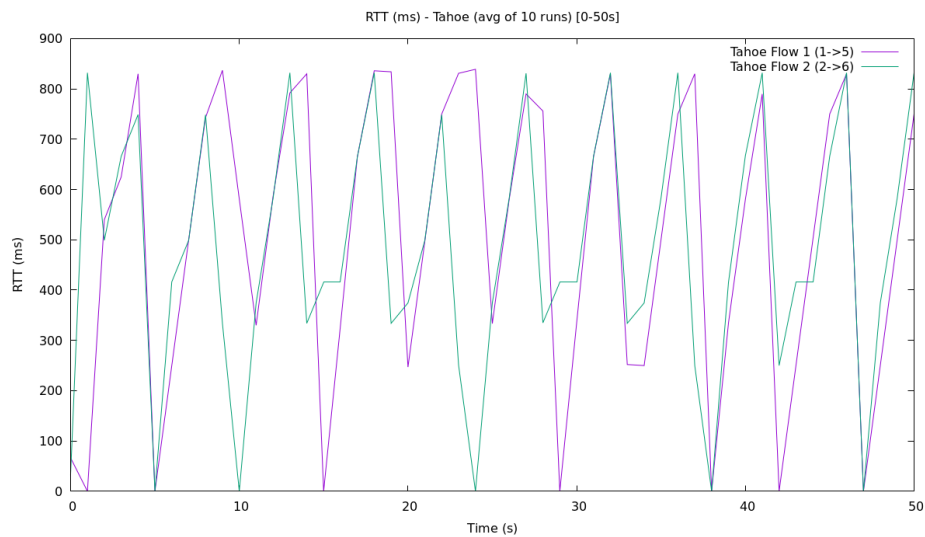
شکل ۱۳: نمودار throughput برای Tahoe، ۰ تا ۱۰۰۰ ثانیه



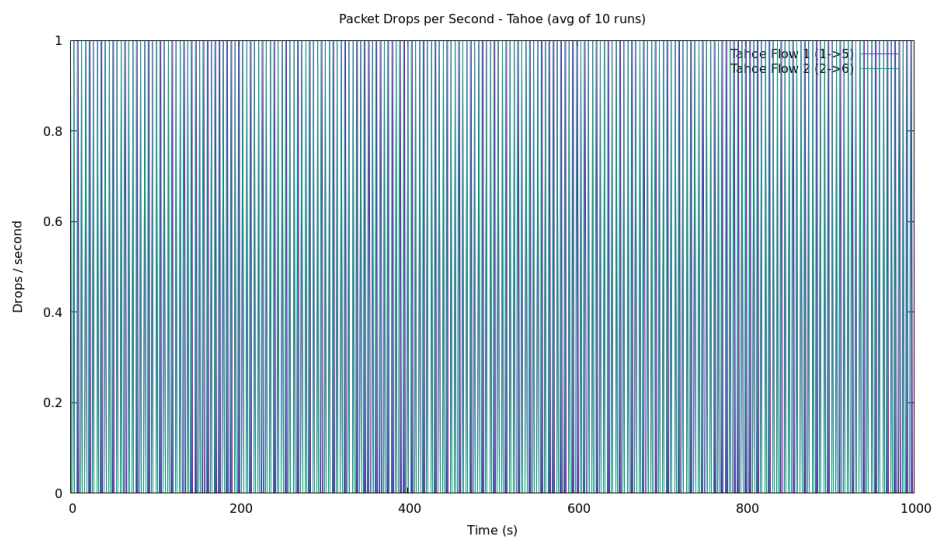
شکل ۱۴: نمودار throughput برای Tahoe زوم‌شده، ۰ تا ۵۰ ثانیه



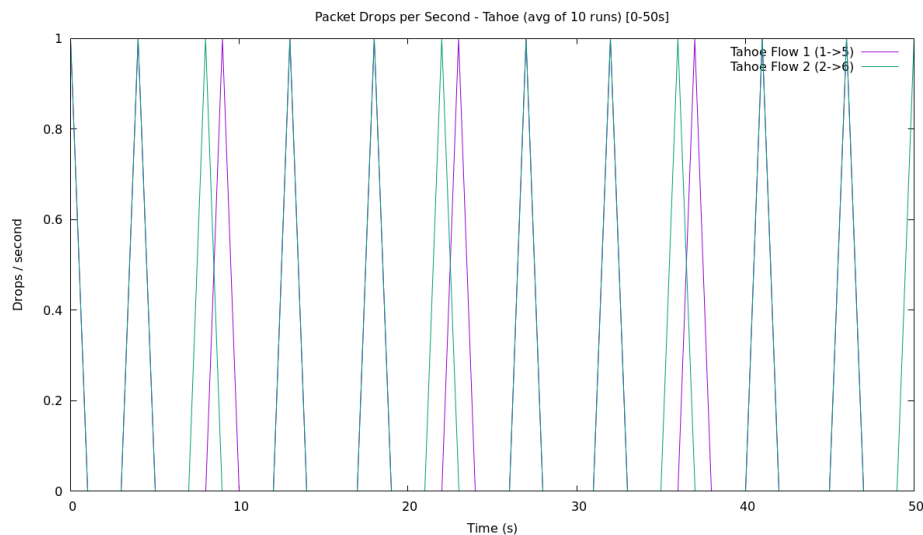
شکل ۱۵: نمودار RTT برای Tahoe، ۰ تا ۱۰۰۰ ثانیه



شکل ۱۶: نمودار RTT برای Tahoe زوم‌شده، ۰ تا ۵۰ ثانیه

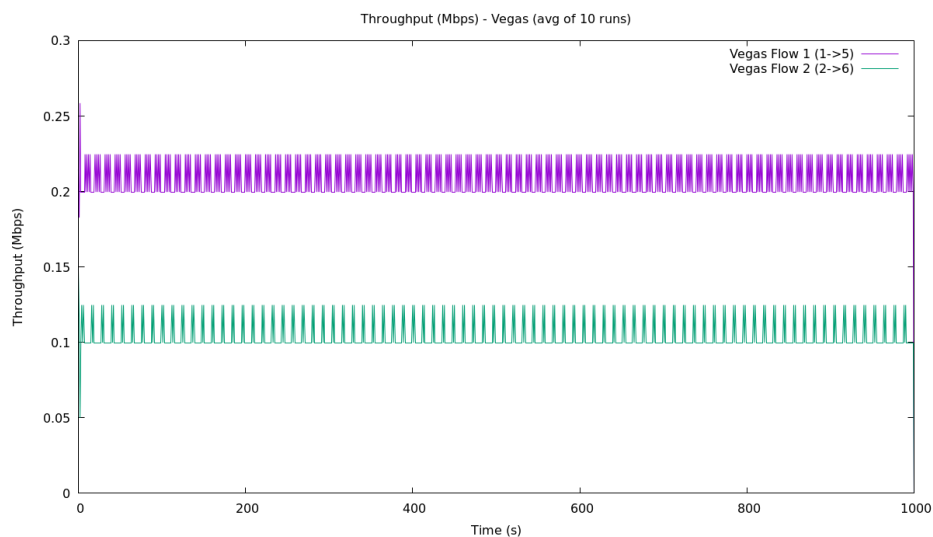


شکل ۱۷: نمودار loss برای Tahoe، ۰ تا ۱۰۰۰ ثانیه

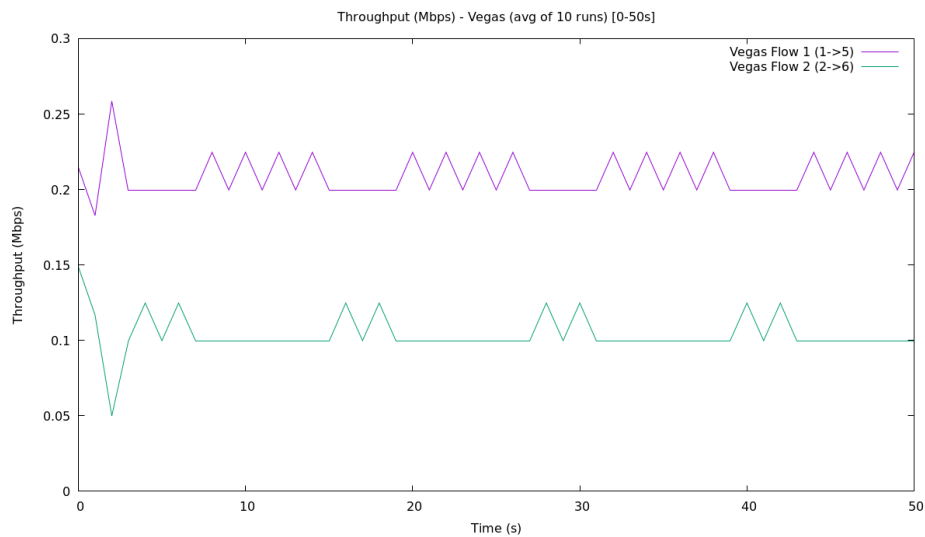


شکل ۱۸: نمودار loss برای Tahoe زوم‌شده، ۰ تا ۵۰ ثانیه

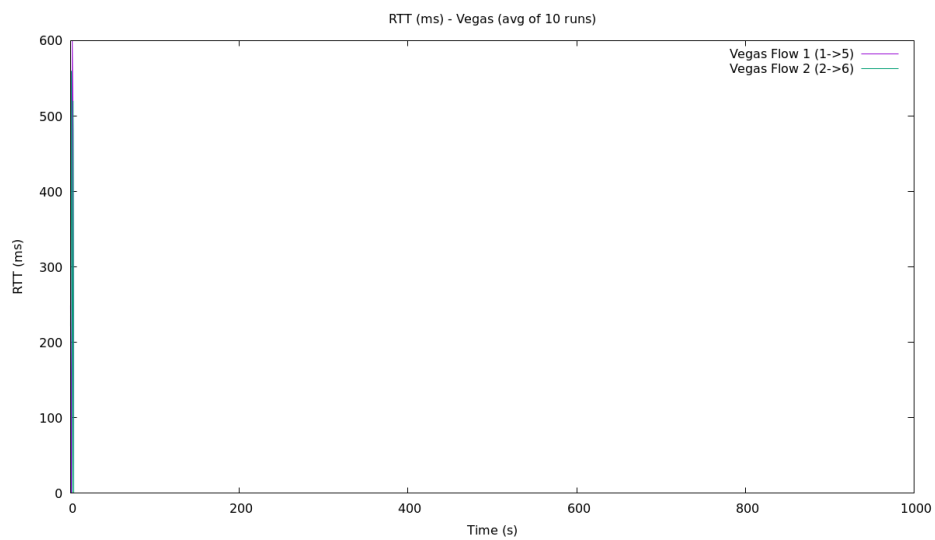
۵.۷ نمودارهای تفکیکی برای Vegas (کل و زوم)



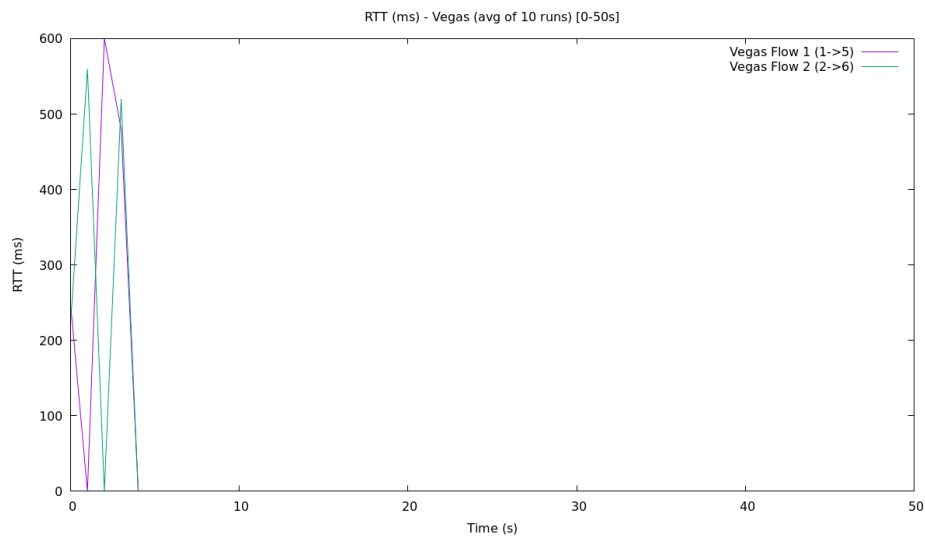
شکل ۱۹: نمودار throughput برای Vegas، ۰ تا ۱۰۰۰ ثانیه



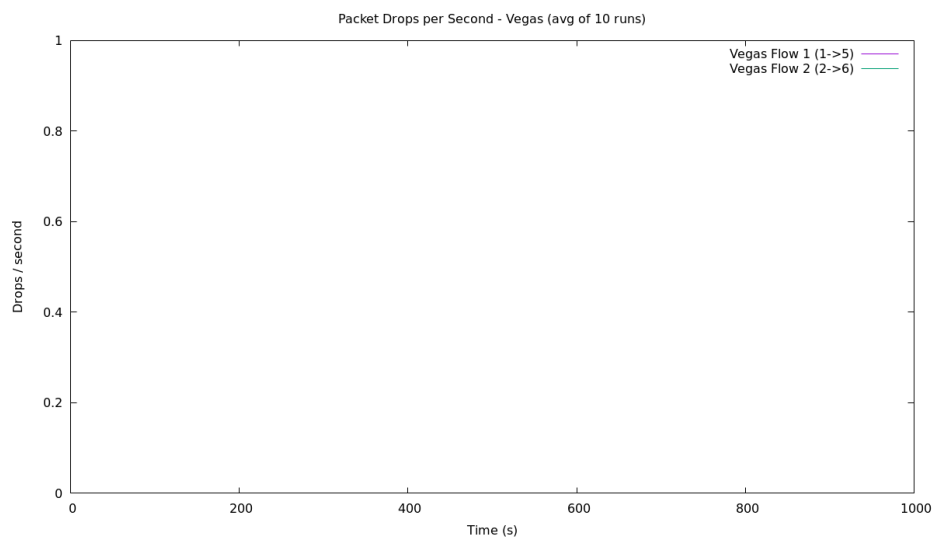
شکل ۲۰: نمودار throughput برای Vegas زوم‌شده، ۰ تا ۵۰ ثانیه



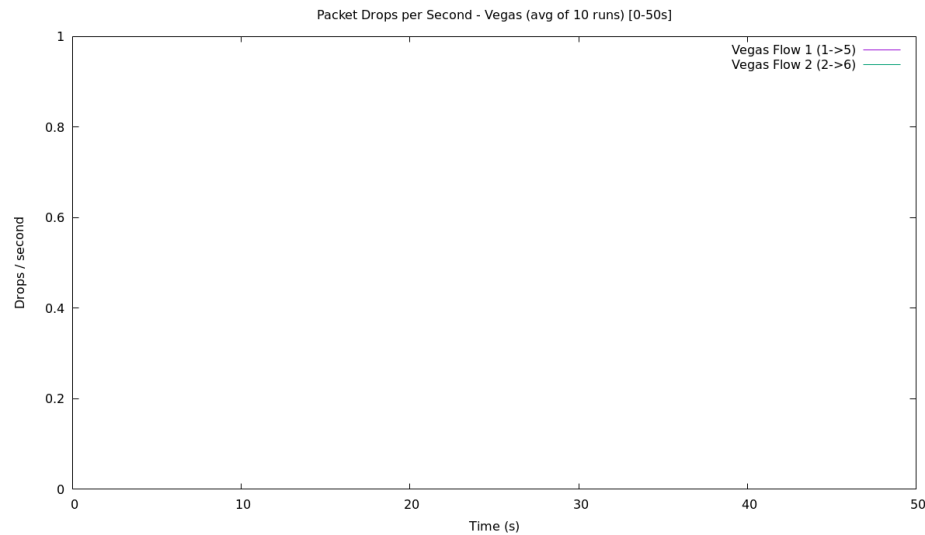
شکل ۲۱: نمودار RTT برای Vegas، ۰ تا ۱۰۰۰ ثانیه



شکل ۲۲: نمودار RTT برای Vegas زوم شده، ۰ تا ۵۰ ثانیه

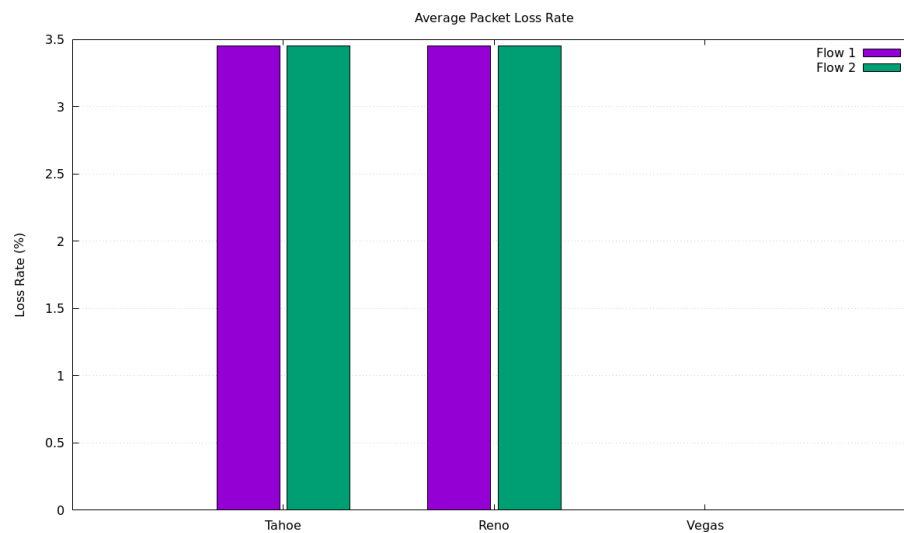


شکل ۲۳: نمودار loss برای Vegas، ۰ تا ۱۰۰۰ ثانیه



شکل ۲۴: نمودار loss برای Vegas زوم‌شده، ۰ تا ۵۰ ثانیه

۶.۷ نمودار میله‌ای نرخ اتلاف متوسط



شکل ۲۵: نمودار میله‌ای برای نرخ loss شدن هر دو جریان هر سه پروتکل

۸ تحلیل نتایج و علت یابی

در این بخش، نتایج به تفکیک هر متریک تحلیل می‌شوند. دقت کنید که تحلیل‌ها دقیقاً بر مبنای نمودارهای تولیدشده و مقادیر مشاهده‌شده در خروجی‌ها نوشته شده‌اند:

۱.۸ تحلیل Throughput

۱.۱.۸ مشاهدات

در نمودارهای شکل ۱ و شکل ۴، رفتار کلی چنین دیده می‌شود:

- گذردهی هر جریان در طول زمان نوسان دارد و این نوسان در دو خانواده‌ی loss-based (یعنی Tahoe/Reno) پررنگ‌تر است.
- در Vegas، گذردهی‌ها بسیار پایدارتر و پله‌ای یا تناوبی اما منظم هستند (در نمودارهای شکل ۱۹ و ۲۰ کاملاً مشخص است).
- سطح گذردهی Vegas Flow 1 به شکل تقریبی در بازه‌ی حدود 0.20 تا 0.225 Mbps قرار دارد و Vegas Flow 2 حدود 0.10 تا 0.125 Mbps (در نمودار تفکیکی Vegas این دو سطح تقریباً ثابت دیده می‌شوند).
- در Tahoe و Reno (نمودارهای شکل ۷ و ۱۳)، گذردهی‌ها حول مقادیر میانی‌تر (حدود 0.14 تا 0.19 Mbps برای جریان‌ها) نوسان می‌کنند و افت‌های تیز بیشتری دارند.

۲.۱.۸ علت یابی

تفاوت بنیادی Vegas با Tahoe/Reno در این است که:

- Tahoe و Reno عموماً loss-based هستند؛ یعنی سیگنال اصلی ازدحام را از رخ دادن packet loss (یا رخ دادهای مرتبط با آن مثل duplicate ACK/timeout) می‌گیرند. پس تا زمانی که صف‌ها پر نشده و drop رخ نداده، ممکن است پنجره رشد کند و شبکه را به آستانه‌ی ازدحام نزدیک کند؛ سپس با drop، پنجره کاهش می‌یابد و گذردهی افت می‌کند. این چرخه باعث نوسان‌های واضح در گذردهی می‌شود.
- Vegas رویکرد delay-based دارد؛ یعنی با پایش RTT و افزایش تأخیر (نشانه‌ی رشد صف‌ها)، قبل از رسیدن به مرحله‌ی drop، نرخ ارسال یا پنجره را تنظیم می‌کند تا صف‌ها در محدوده‌ی کوچک‌تری نگه داشته شوند. نتیجه‌ی عملی این رفتار، گذردهی پایدارتر و جلوگیری از سقوط‌های ناشی از loss است.

بنابراین، پایدار بودن و منظم بودن Throughput در Vegas با نبودن drop (که در بخش Loss واضح است) هم‌راستا است.

۲.۸ تحلیل RTT

۱.۲.۸ مشاهدات

نمودارهای شکل ۲ و ۵ (۶-خطی) و نیز نمودارهای تفکیکی برای هر TCP نشان می‌دهند:

- در Tahoe و Reno، RTT جهش‌های بزرگ و تکرارشونده دارد و تا حوالی ۸۰۰ الی ۸۵۰ ns بالا می‌رود (در شکل‌های ۹ و ۱۵ واضح است).

- در نمودارهای زوم‌شده (مثل نمودار شکل ۱۰ و ۱۵)، این جهش‌ها به شکل دندان‌دار و رفت‌وبرگشتی دیده می‌شوند که با چرخه‌ی پر و خالی شدن صف‌ها هم‌خوان است.

- در Vegas، نمودار شکل ۲۱ نشان می‌دهد در کل بازه، RTT به شکل بسیار محدود و فشرده ظاهر شده و در نمودار شکل ۲۲ نیز فقط چند نمونه‌ی ابتدای اتصال دیده می‌شود و بعد عملاً نمونه‌ای ثبت نشده است.

۲.۲.۸ علت یابی

این بخش دو علت را جدا می‌کند: یکی علت رفتاری پروتکل و یکی علت ماهیت داده‌ی رسم‌شده:
لایه ۱: علت رفتاری پروتکل

- در Tahoe/Reno چون پروتکل تا مرز ازدحام جلو می‌رود، صف‌های روتر پر می‌شوند، زمان انتظار در صف (queueing delay) زیاد می‌شود و RTT رشد می‌کند. سپس با loss و کاهش پنجره، صف تخلیه شده و RTT افت می‌کند؛ بنابراین RTT نوسانی و بزرگ می‌شود.

- در Vegas اما هدف این است که با کنترل بر اساس delay، صف را کوچک نگه دارد؛ پس انتظار می‌رود RTT نسبتاً پایدارتر باشد و به جهش‌های شدید نرسد.

لایه ۲: علت ماهیت داده‌ی رسم‌شده از خود نمودارهای Vegas مشخص است که سری زمانی RTT بعد از چند ثانیه تقریباً داده‌ی قابل رسم ندارد و به همین دلیل در بازه‌ی ۰ تا ۱۰۰۰ عملاً خطی دیده نمی‌شود یا فقط نزدیک زمان صفر فشرده می‌شود. این معمولاً یکی از این دو معنی را دارد (و هر دو با خروجی فعلی سازگارند):

- یا مکانیزم ثبت RTT sample در سناریو برای Vegas بسیار کم‌تعداد و محدود اتفاق افتاده (مثلاً فقط در فاز ابتدایی اتصال).

- یا روش استخراج RTT (فایل‌های trace/rtt) برای Vegas بعد از شروع، رکورد جدیدی تولید نکرده است و بنابراین ابزار رسم، چیزی برای نمایش ندارد.

لذا نتیجه‌ی مهم این است که نمودار Vegas RTT در کل بازه، اطلاعات نمونه‌ای محدود دارد و در زوم فقط همان نمونه‌های ابتدای اتصال قابل مشاهده است. این دقیقا همان چیزی است که در نمودارهای شکل ۲۱ و ۲۲ می‌بینیم.

۳.۸ تحلیل Packet Loss

۱.۳.۸ مشاهدات

نمودارهای شکل ۳ و ۶ (۶-خطی) و نمودارهای تفکیکی نشان می‌دهند:

- در Tahoe و Reno، dropها به صورت ضربه‌های متناوب و پرتکرار مشاهده می‌شوند (به شکل خطوط عمودی متراکم در نمودارهای ۱۰۰۰ ثانیه‌ای مثل نمودارهای شکل ۱۱ و ۱۷).
- در زوم ۰ تا ۵۰ ثانیه (نمودارهای شکل ۱۲ و ۱۸ و همچنین شکل ۶)، الگوی افت‌ها به شکل پالس‌های تکی (اغلب مقدار ۰ یا ۱ drop/sec) دیده می‌شود.
- در Vegas، نمودارهای شکل ۲۳ و ۲۴ عملا هیچ dropی را نشان نمی‌دهند (نمودار تقریبا خالی است).

۲.۳.۸ نرخ اتلاف متوسط (نمودار میله‌ای)

نمودار میله‌ای شکل ۲۵ و داده‌ی خلاصه‌ی تولیدشده نشان می‌دهد:

- نرخ اتلاف متوسط برای Tahoe و Reno تقریبا یکسان و حدود 3.45% برای هر دو جریان است.
- نرخ اتلاف متوسط برای Vegas برابر 0% گزارش شده است.

این نتیجه با نمودارهای زمانی نیز سازگار است: Tahoe/Reno پالس‌های drop دارند، اما Vegas عملا ندارد.

۳.۳.۸ علت یابی

چرا Tahoe/Reno اتلاف دارند؟ طبق صورت پروژه، از دست‌رفتن روی لینک‌ها نداریم و اتلاف تنها به دلیل محدود بودن بافر روترها رخ می‌دهد. همچنین اندازه صف ۱۰ بسته است. در Tahoe/Reno چون پروتکل‌ها به صورت loss-based رفتار می‌کنند، پنجره تا جایی رشد می‌کند که صف پر شود و در نتیجه بسته‌ها در روتر drop شوند؛ سپس TCP واکنش نشان می‌دهد و پنجره را کم می‌کند. این چرخه باعث ایجاد dropهای دوره‌ای می‌شود. چرا Vegas اتلاف ندارد؟ Vegas قبل از رسیدن صف به حالت اشباع، با دیدن افزایش RTT (تاخیر صف) نرخ ارسال را تنظیم می‌کند؛ بنابراین معمولا اجازه نمی‌دهد بافر ۱۰ تایی پر شود و بسته drop شود. نتیجه‌های قابل مشاهده در خروجی‌های Vegas تقریبا drop صفر و گذردهی پایدارتر دارد.

۹ جمع‌بندی مقایسه‌ای نهایی

با کنار هم گذاشتن سه معیار، یک تصویر کلی از رفتار سه پروتکل به دست می‌آید:

- Tahoe و Reno رفتار بسیار نزدیک به هم نشان می‌دهند (هم در نرخ اتلاف متوسط و هم در الگوی نوسان RTT و Throughput). این هم‌رفتاری در خروجی کاملاً مشهود است: نرخ اتلاف متوسط هر دو تقریباً برابر گزارش شده و نمودارهای زمانی نیز الگوی مشابه دارند.
- Vegas از نظر Loss بهترین نتیجه را دارد (تقریباً صفر)، و از نظر Throughput نیز پایدارتر و منظم‌تر دیده می‌شود.
- در RTT Tahoe/Reno جهش‌های شدید دارد که نشان‌دهنده‌ی تشکیل صف و ازدحام است؛ در Vegas نمودار RTT نمونه‌های محدودتری نشان می‌دهد و در سطح نمایش داده‌شده، ازدحام و صف شدید به سبک Tahoe/Reno مشاهده نمی‌شود.