Лабораторная работа № 2

Исследование протокола TCP и алгоритма управления очередью RED

Мугари Абдеррахим

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

* Цель лабораторной работы — исследование поведения различных версий протокола TCP (NewReno, Reno) в сети с управлением перегрузками с использованием алгоритма RED. Анализируется изменение размера окна перегрузки и влияние параметров очереди на эффективность передачи данных. Результаты визуализируются с помощью xgraph, с настройкой цвета и подписей графиков.

# 2 Выполнение лабораторной работы

## 2.1 Пример с дисциплиной RED

### 2.1.1 Алгоритм управления очередью RED:

* здесь мы познакомились с алгоритмом RED (случайного раннего обнаружения) и его функцией сброса (Схема и распределение) (рис. 1).

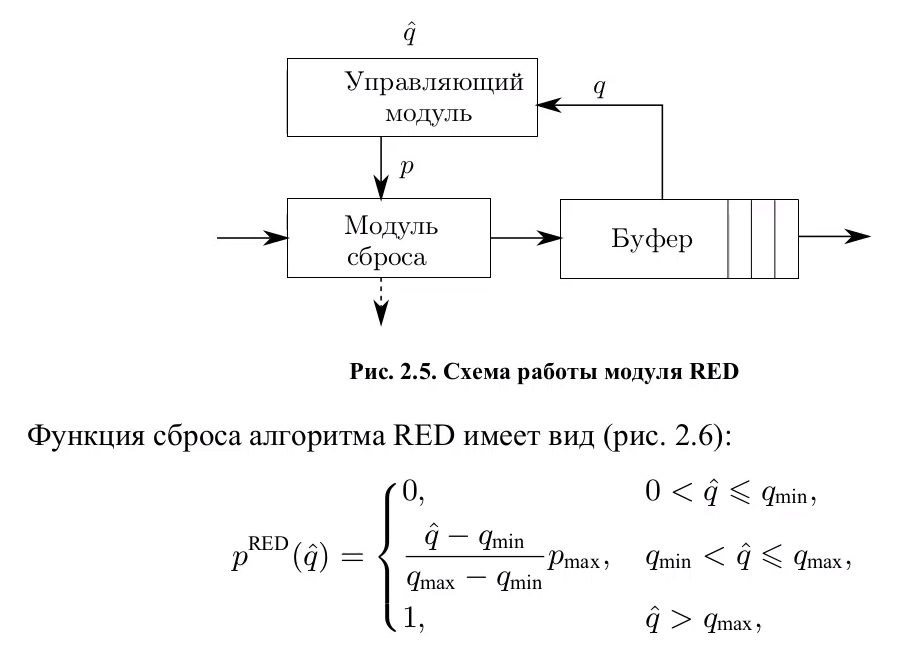


Рис. 1: Алгоритм управления очередью RED

### 2.1.2 Реализация модели задачи

### 2.1.3 Создание симудятора

* Создается объект симулятора (ns).
* Определяются узлы сети: s1, s2, s3, s4 (отправители) и r1, r2 (маршрутизаторы) (рис. 2).

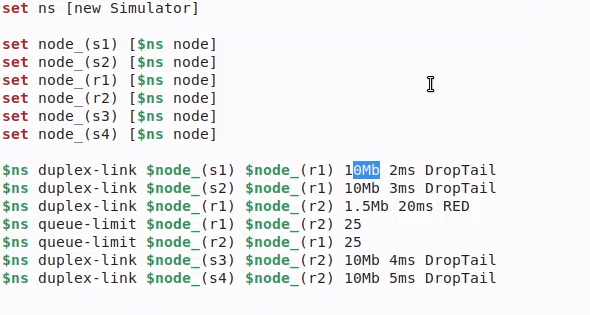


Рис. 2: Алгоритм управления очередью RED

### 2.1.4 Создание соединений между узлами

* Создаются дуплексные (двусторонние) соединения между узлами с заданной пропускной способностью, задержкой и алгоритмом управления очередью (DropTail или RED).
* Задается лимит очереди на связи r1-r2 и r2-r1 (по 25 пакетов). (рис. 3).

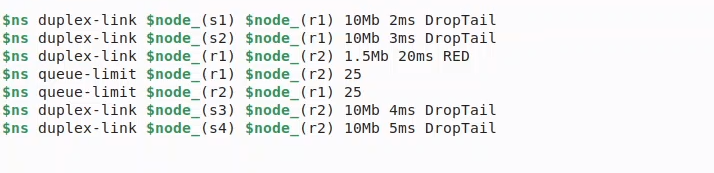


Рис. 3: Создание соединений между узлами

### 2.1.5 Настройка TCP-соединений

* Создаются два TCP-соединения: одно использует TCP/NewReno, другое — TCP/Reno.
* Каждое соединение имеет размер окна 15.
* На TCP-соединения привязываются источники трафика (FTP) (рис. 4).

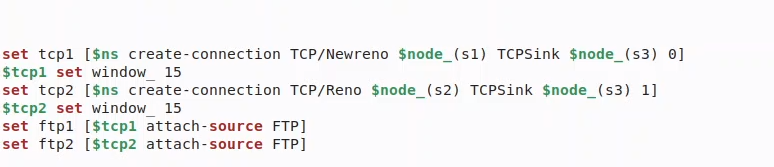


Рис. 4: Настройка TCP-соединений

### 2.1.6 Настройка мониторинга очередей и логирования

* Открывается файл WindowVsTimeReno для записи данных о размере окна перегрузки.
* Включается мониторинг очереди между r1 и r2 (файл qm.out).
* Настраивается трассировка состояния очереди RED (рис. 5).

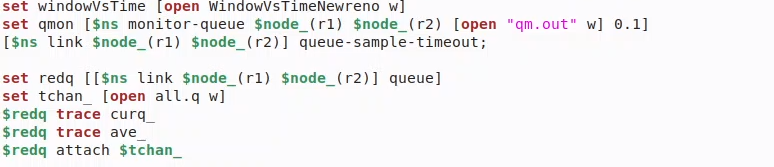


Рис. 5: Настройка мониторинга очередей и логирования

### 2.1.7 Запуск передачи данных

**FTP1 (TCP NewReno)** начинает передачу в **0.0 сек**. Через **1.1 сек**. начинается запись данных о размере окна перегрузки. **FTP2 (TCP Reno)** стартует через **3.0 сек**. Симуляция заканчивается в **10 сек**.(рис. 6).

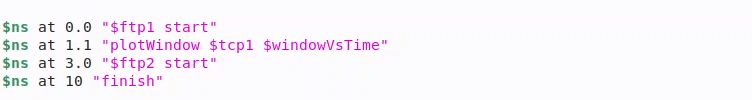


Рис. 6: Запуск передачи данных

### 2.1.8 Функция для записи изменения размера окна TCP

Функция записывает размер окна TCP в файл каждые 0.01 сек (рис. 7).

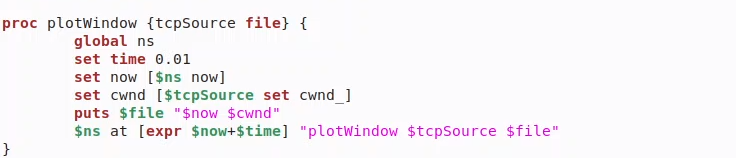


Рис. 7: Функция для записи изменения размера окна TCP

### 2.1.9 Функция завершения симуляции и построения графиков

* Код обрабатывает данные о длине очереди и создает временные файлы.
* Генерирует два графика с помощью xgraph TCPNewrenoCWND и queue (рис. 8).

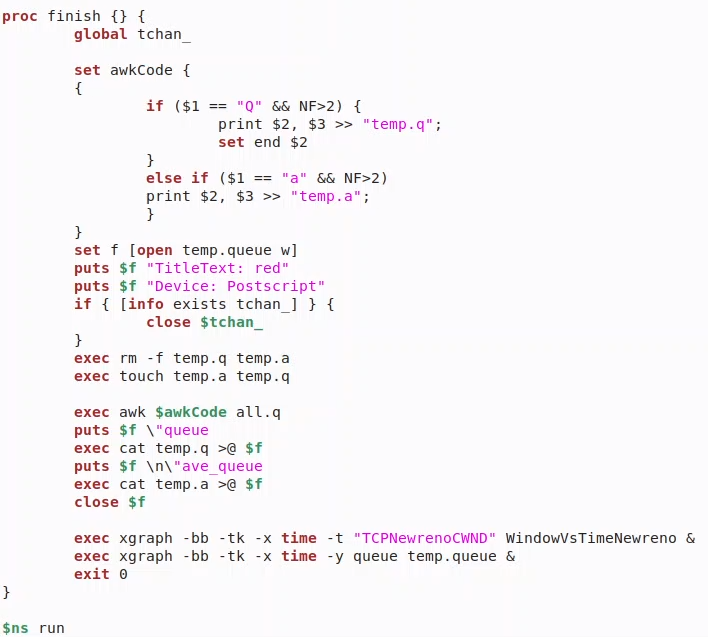


Рис. 8: Функция завершения симуляции и построения графиков

### 2.1.10 Запуск симуляции

$ns run

### 2.1.11 Визуализация:

* Построение графиков с помощью xgraph после завершения симуляции. (рис. 9).

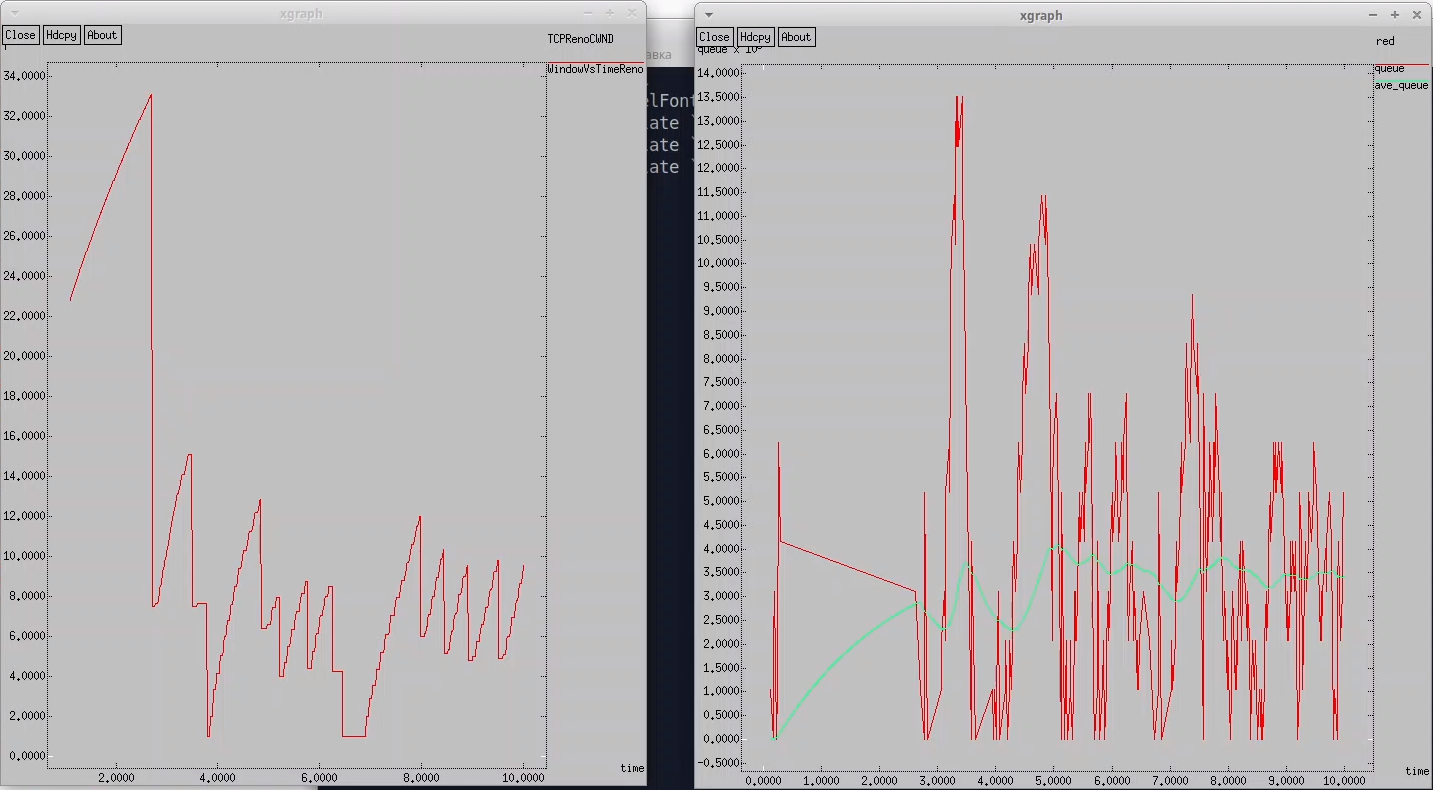


Рис. 9: построения графиков

### 2.1.12 Сравнение результатов и пояснения

#### 2.1.12.1 TCP Reno

* **Механизм работы:** Основан на классическом алгоритме управления перегрузкой с фазами **slow start, congestion avoidance и механизмами fast retransmit/fast recovery**.
* **Характеристика поведения:** При потере пакетов окно резко уменьшается, что может приводить к значительным колебаниям в размере окна (**cwnd**)
* **Особенности:** Может испытывать проблемы при возникновении множественных потерь в одном цикле, что приводит к задержкам в восстановлении (рис. 10).

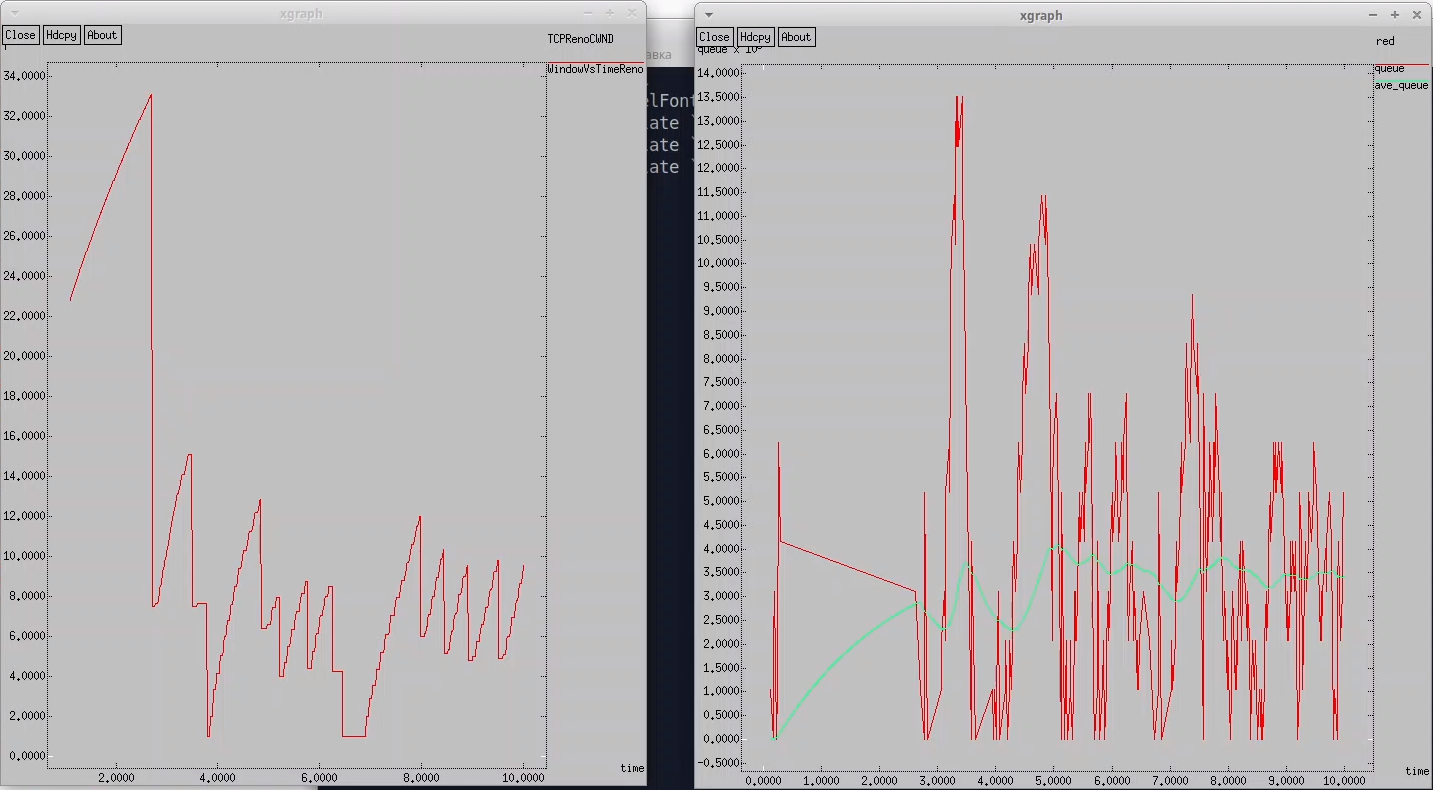


Рис. 10: Изменение окна перегрузки TCP Reno во времени и длины очереди во времени (RED-очередь на маршрутизаторе)

#### 2.1.12.2 TCP NewReno

Здесь мы просто изменили тип TCP-агента на NewReno (рис. 11).

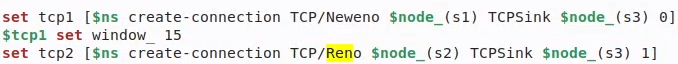


Рис. 11: TCP NewReno

* **Механизм работы:** Улучшенная версия Reno, где алгоритм **fast recovery** остаётся активным до восстановления всех потерянных пакетов в текущем цикле.
* **Характеристика поведения:** Более устойчив к множественным потерям в одном цикле. В графиках cwnd можно наблюдать менее резкие падения по сравнению с **Reno**.
* **Особенности:** В условиях высокой вероятности потерь (например, в сочетании с **RED-очередью**) может обеспечить более стабильный пропускной потенциал за счёт быстрейшего восстановления (рис. 12).

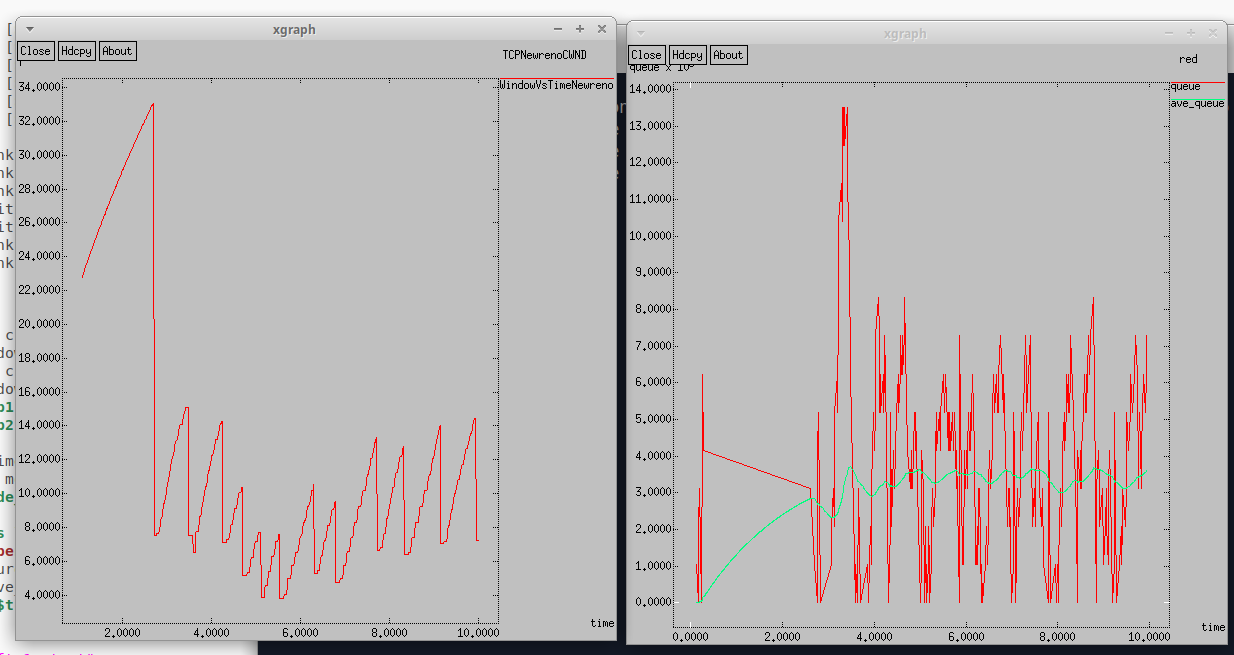


Рис. 12: Изменение окна перегрузки TCP NewReno во времени и длины очереди во времени (RED-очередь на маршрутизаторе)

#### 2.1.12.3 TCP Vegas

Пототм изменили тип TCP-агента на Vegas (рис. 13).

* TCP Vegas часто приводит к более стабильной и «аккуратной» загрузке сети, снижая переполнение очереди и потери. Однако в смешанных сетях (где есть Reno/NewReno и Vegas одновременно) Vegas может проигрывать по пропускной способности более «агрессивным» вариантам, так как раньше снижает скорость при признаках роста задержек.

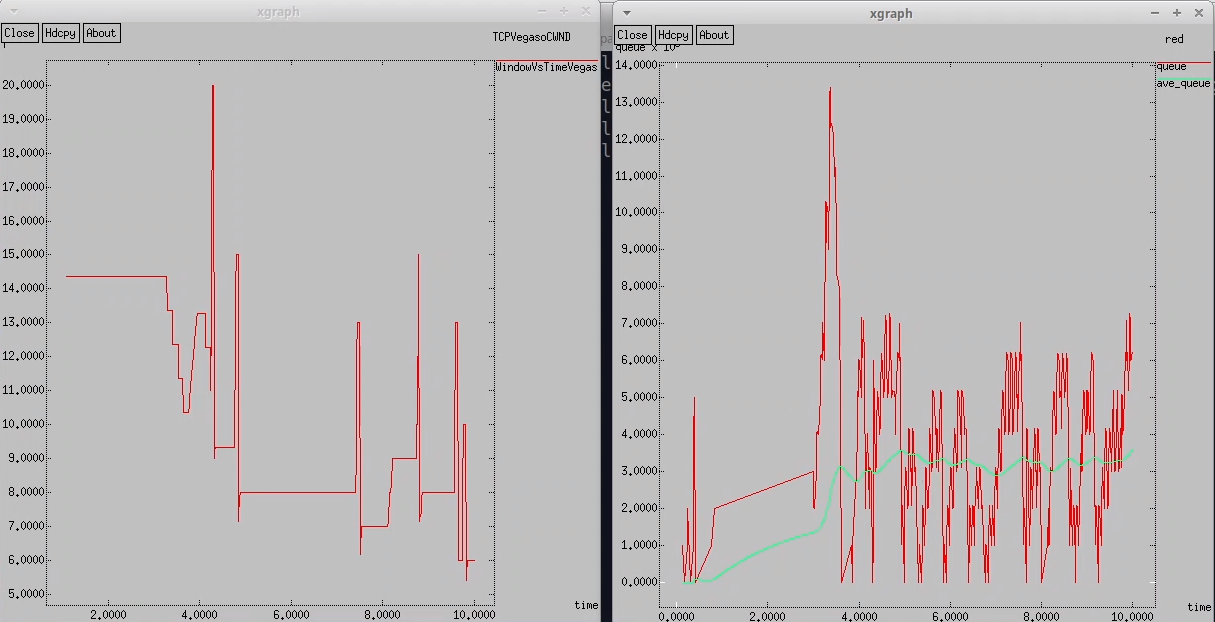


Рис. 13: TCP Vegas

#### 2.1.12.4 Вывод

* **Reno**: наглядные колебания очереди и cwnd, резкие падения окна при потерях.
* **NewReno**: в целом схож с Reno, но лучше восстанавливается после множественных потерь, что делает его чуть более стабильным и эффективным.
* **Vegas**: стремится минимизировать потери и задержки, регулируя окно заранее на основе измерений **RTT**, поэтому обычно показывает более низкую и стабильную очередь и менее резкие скачки cwnd, но при этом может иметь более низкую пропускную способность в конкурентной среде.
* С точки зрения графиков, у Reno/NewReno красная линия (длина очереди) часто уходит довольно высоко, а затем падает, сопровождаясь резким снижением cwnd. У Vegas кривая очереди более сглажена, а **cwnd** меняется плавнее.

### 2.1.13 Внести изменения при отображении окон с графиками

* **Цвета траекторий:**
* "0.Color: orange" — первая линия (размер очереди).
* "1.Color: cyan" — вторая линия (средняя длина очереди).
* **Фон и цвет графиков:**
* Фон (-bg black) — чёрный.
* Цвет линий (-fg gold) — золотой (но это влияет только на оси, заголовок и подписи).
* **Заголовки:**
* "Queue\_Stats" — добавлен заголовок для второго графика.
* "TCPNewrenoCWND" — название графика для размера окна TCP.
* **подписи к осям, подпись траектории в легенде**
* puts $f \n\"lenghtofochered"
* -x time -y queue

# 3 Выводы

* В ходе работы изучены механизмы управления окном перегрузки в TCP NewReno и динамика очереди. Графики показали зависимость этих параметров от времени, а изменения оформления улучшили их интерпретацию. Анализ подтвердил, что управление окном и очередью повышает эффективность сети.

Более подробно про RED см. в [[1]; [2];]

# Список литературы

1. Stevens W.R. [TCP/IP Illustrated, Volume 1: The Protocols](https://www.pearson.com/us/higher-education/program/Stevens-TCP-IP-Illustrated-Volume-1-The-Protocols/PGM162554.html). 1st изд. Addison-Wesley, 1994.

2. Tanenbaum A.S., Wetherall D.J. [Computer Networks](https://www.pearson.com/us/higher-education/program/Tanenbaum-Computer-Networks-5th-Edition/PGM28468.html). 5th изд. Pearson, 2010.