Отчёт по лабораторной работе №2

Исследование протокола TCP и алгоритма управления очередью RED

Надежда Александровна Рогожина

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Проанализировать разницу между 3-мя протоколами передачи данных: - TCP Reno - TCP NewReno - TCP Vegas

# 2 Задание

На основе приведенной в лабораторной работе модели с 6 узлами, изменить типы протоколов и проанализировать результаты. Поменять визуализацию графиков.

# 3 Теоретическое введение

Протокол управления передачей (Transmission Control Protocol, TCP) имеет средства управления потоком и коррекции ошибок, ориентирован на установление соединения.

* Флаг Указатель срочности (Urgent Pointer, URG) устанавливается в 1 в случае использования поля Указатель на срочные данные.
* Флаг Подтверждение (Acknowledgment, ACK) устанавливается в 1 в случае, если поле Номер подтверждения (Acknowledgement Number) содержит данные. В противном случае это поле игнорируется.
* Флаг Выталкивание (Push, PSH) означает, что принимающий стек TCP должен немедленно информировать приложение о поступивших данных, а не ждать, пока буфер заполниться.
* Флаг Сброс (Reset, RST) используется для отмены соединения из-за ошибки приложения, отказа от неверного сегмента, попытки создать соединение при отсутствии затребованного сервиса.
* Флаг Синхронизация (Synchronize, SYN) устанавливается при инициировании соединения и синхронизации порядкового номера.
* Флаг Завершение (Finished, FIN) используется для разрыва соединения. Он указывает, что отправитель закончил передачу данных.

Управление потоком в протоколе TCP осуществляется при помощи скользящего окна переменного размера: - поле Размер окна (Window) (длина 16 бит) содержит количество байт, которое может быть послано после байта, получение которого уже подтверждено; - если значение этого поля равно нулю, это означает, что все байты, вплоть до байта с номером Номер подтверждения - 1, получены, но получатель отказывается принимать дальнейшие данные; - разрешение на дальнейшую передачу может быть выдано отправкой сегмента с таким же значением поля Номер подтверждения и ненулевым значением поля Размер окна.

Регулирование трафика в TCP: - контроль доставки — отслеживает заполнение входного буфера получателя с помощью параметра Размер окна (Window); - контроль перегрузки — регистрирует перегрузку канала и связанные с этим потери, а также понижает интенсивность трафика с помощью Окна перегрузки (Congestion Window, CWnd) и Порога медленного старта (Slow Start Threshold, SSThreth).

Например, в табл. 1 приведено краткое описание анализируемых протоколов передачи данных.

Таблица 1: Описание некоторых протоколов передачи данных

| Характеристика | TCP Reno | TCP NewReno | TCP Vegas |
| --- | --- | --- | --- |
| Основной подход | Реакция на потери пакетов | Реакция на потери пакетов | Управление на основе задержек |
| Множественные потери | Неэффективно | Эффективно | Нет потерь (проактивно) |
| Агрессивность | Высокая | Высокая | Низкая |
| Стабильность | Средняя | Средняя | Высокая |
| Сложность реализации | Простая | Средняя | Сложная |

# 4 Выполнение лабораторной работы

Первым делом мы копировали шаблон в новый файл и открыли его на редактирование (рис. 1).

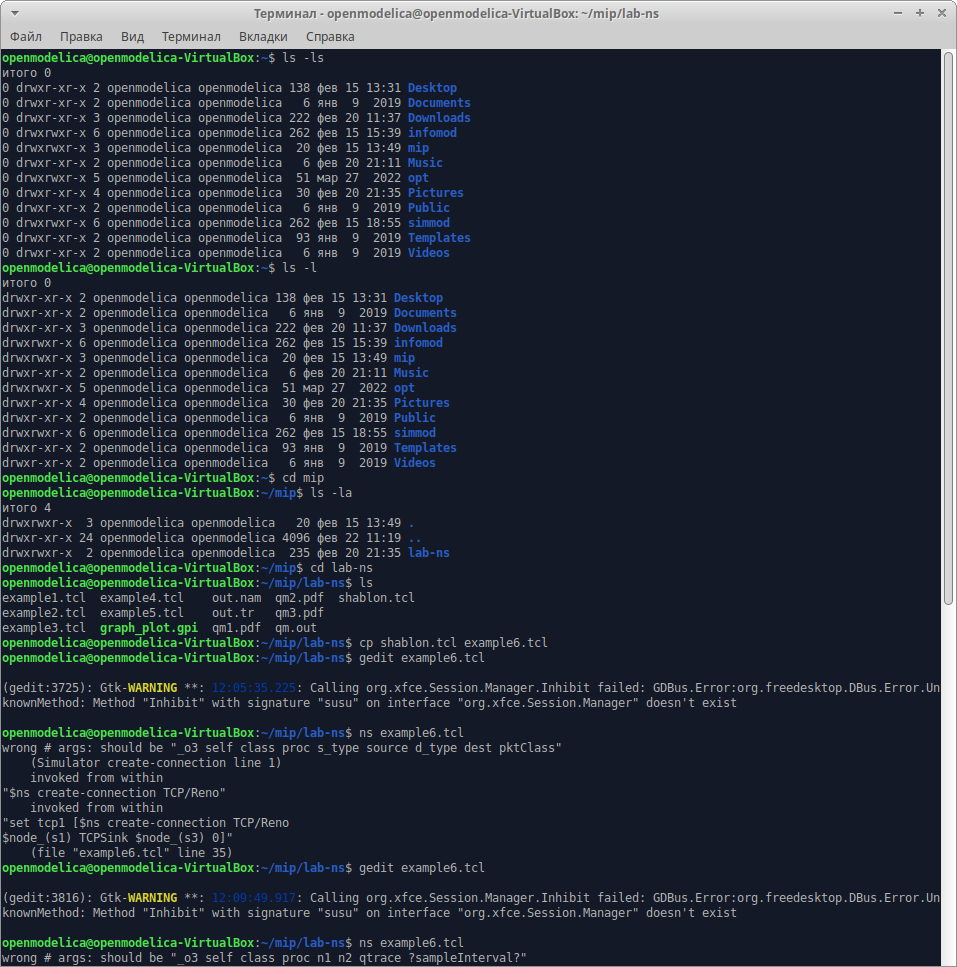


Рис. 1: Создание нового файла

В тексте лабораторной работы был дан код алгоритма, который было необходимо реализовать. Мы его переписали в новый созданный файл (рис. 2, рис. 3, рис. 4).

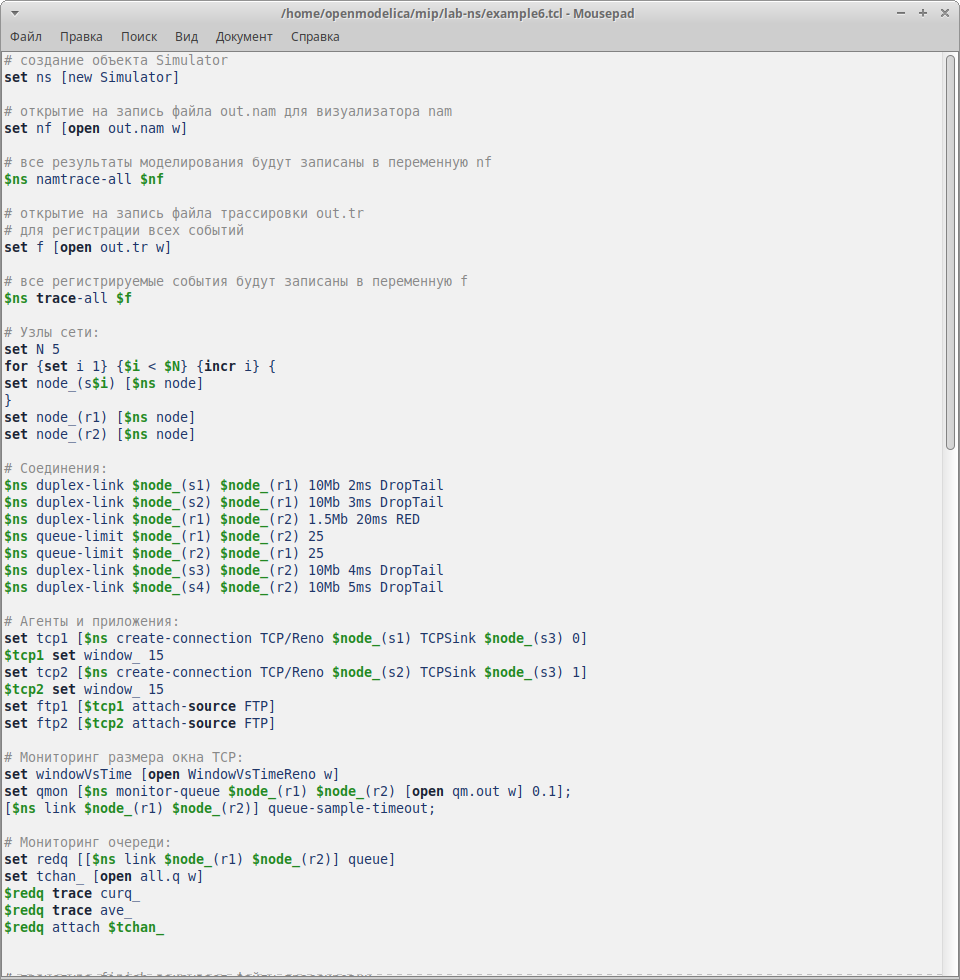


Рис. 2: Реализация модели

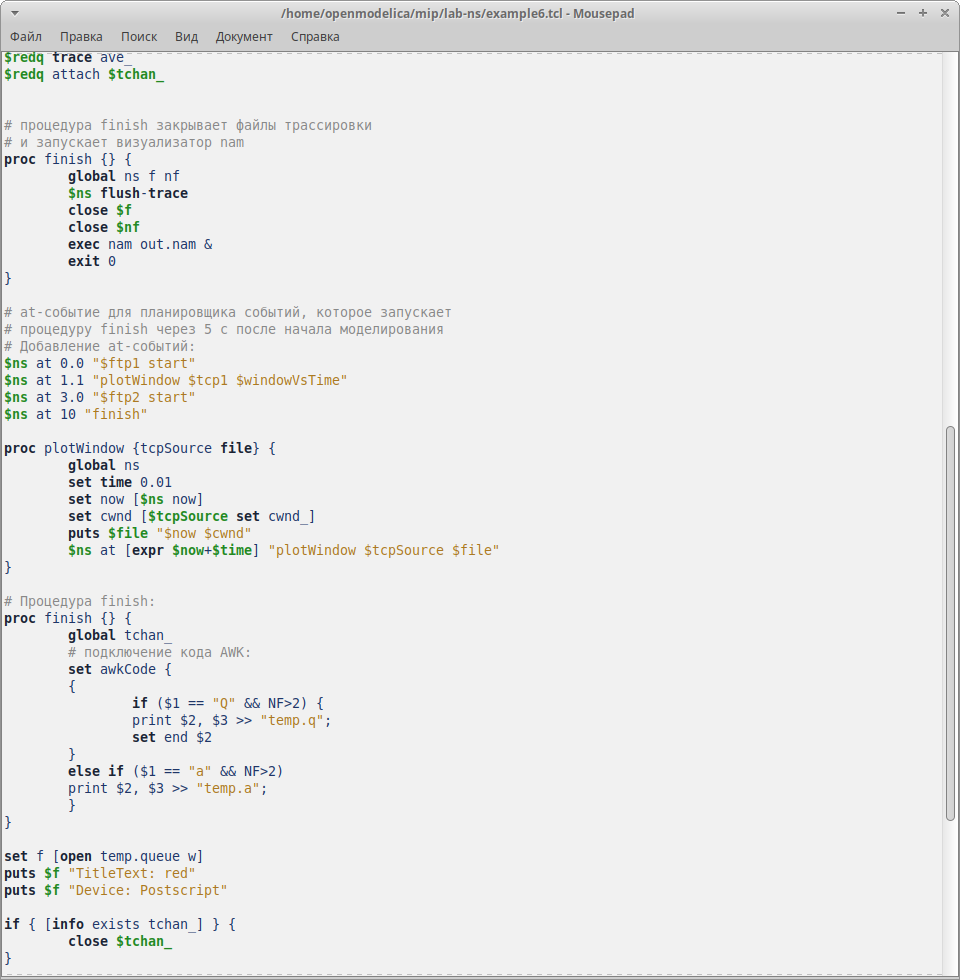


Рис. 3: Реализация модели

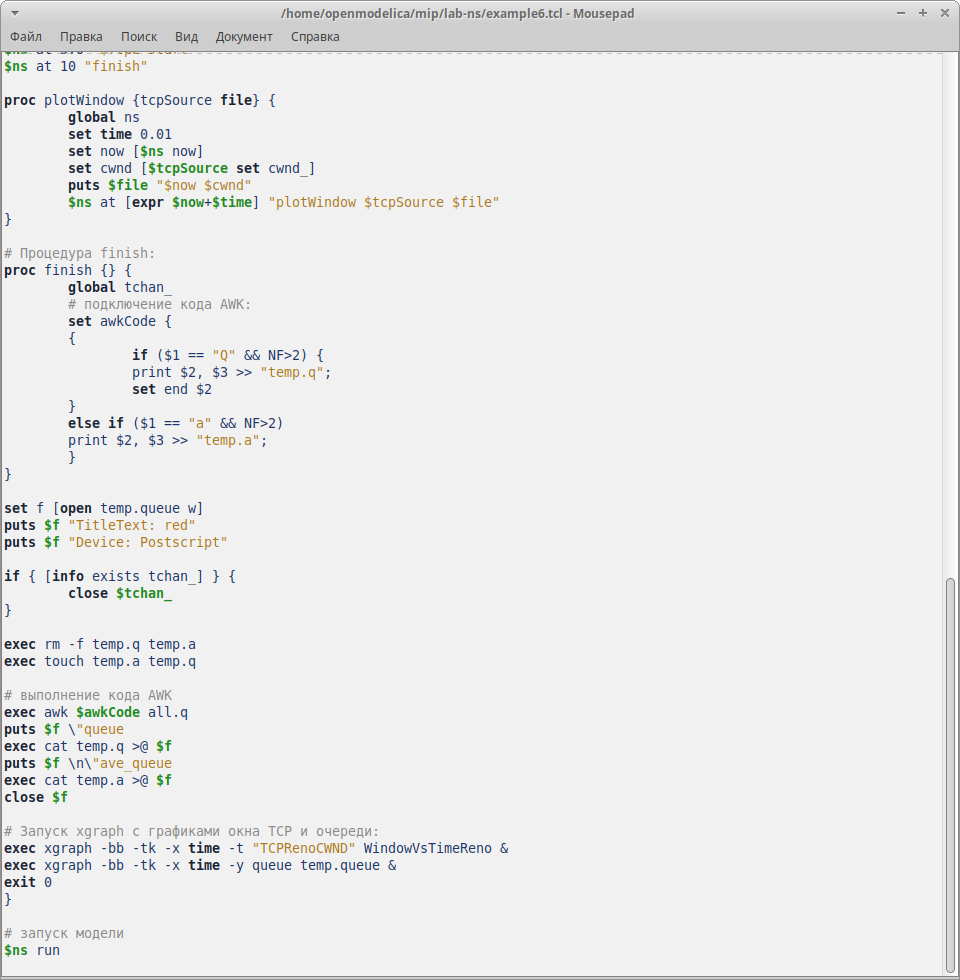


Рис. 4: Реализация модели

Запустив в терминале команду ns example6.tcl, мы получили первый результат (рис. 5, рис. 6).

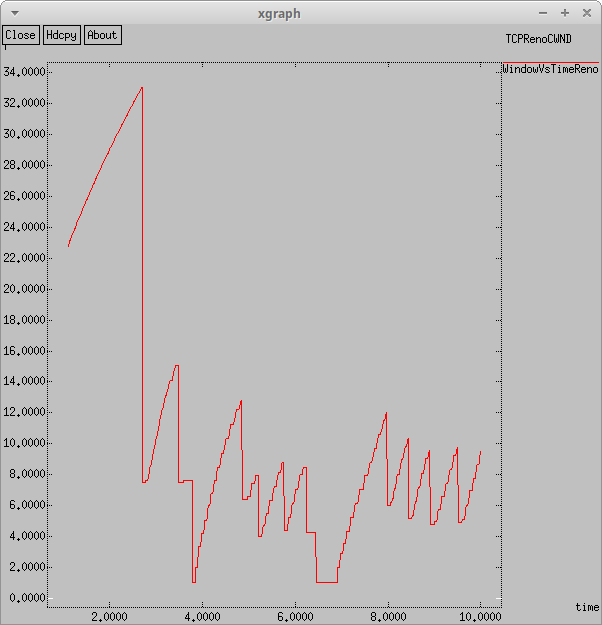


Рис. 5: График изменения размера окна Reno

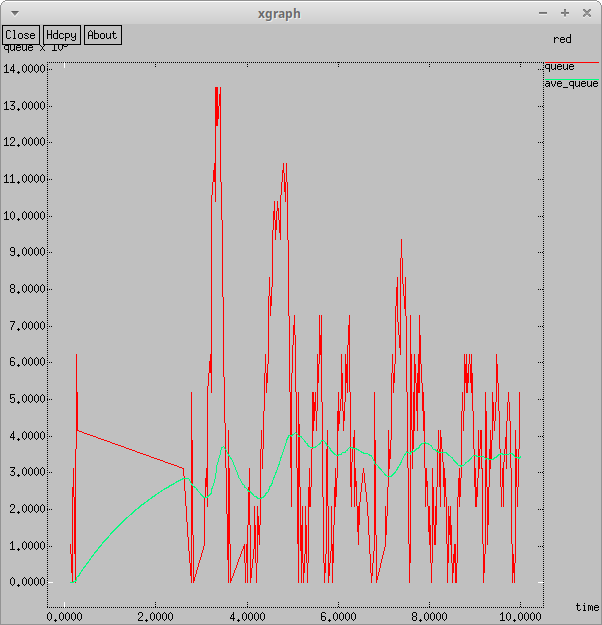


Рис. 6: График фактической и средней длины очереди Reno

Для более приятной и понятной визуализации, я изменила график: - -bg white для белого фона - 0.Color=purple для цвета первой рисуемой линии - 1.Color=orange для цвета второй рисуемой линии - третью линию оставила как есть, красной (рис. 7, рис. 8).

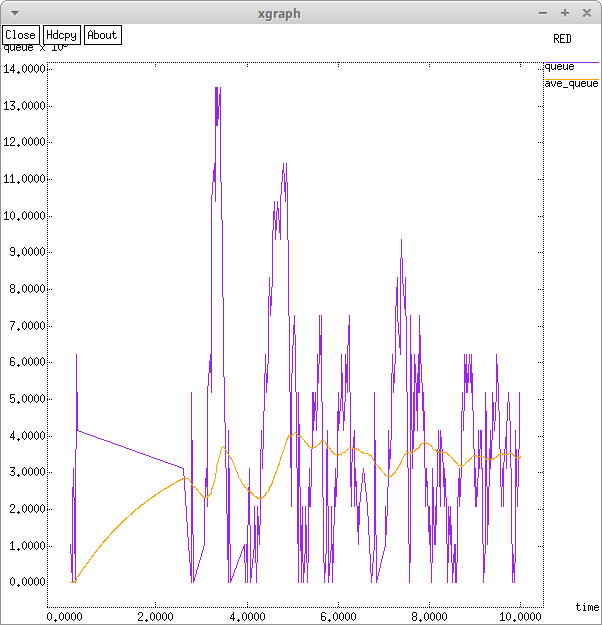


Рис. 7: Изменение визуализации графика фактической и средней длины очереди Reno

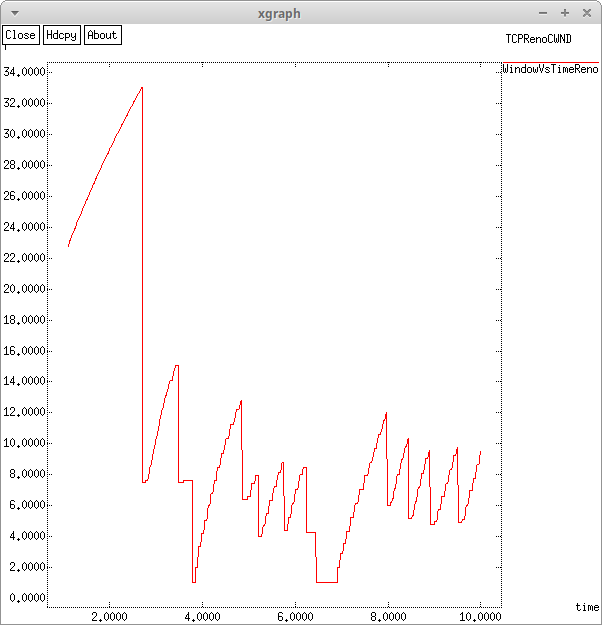


Рис. 8: Изменение визуализации графика изменения размера окна Reno

Далее, мы поменяли тип протокола TCP c Reno на NewReno (рис. 9, рис. 10).

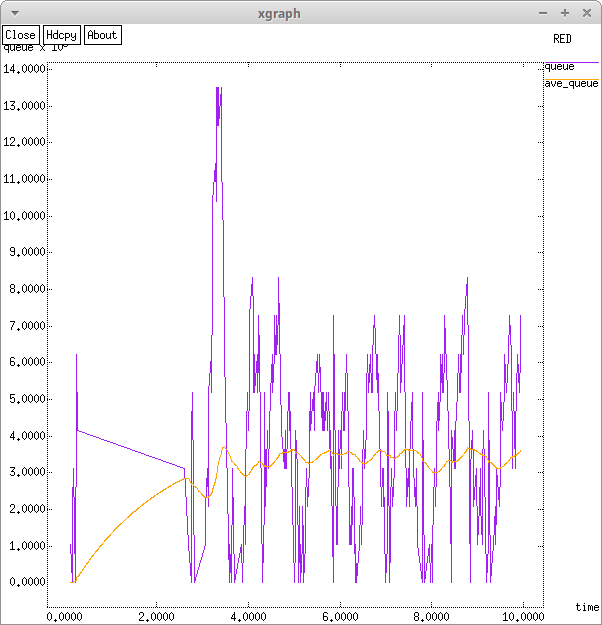


Рис. 9: График фактической и средней длины очереди NewReno

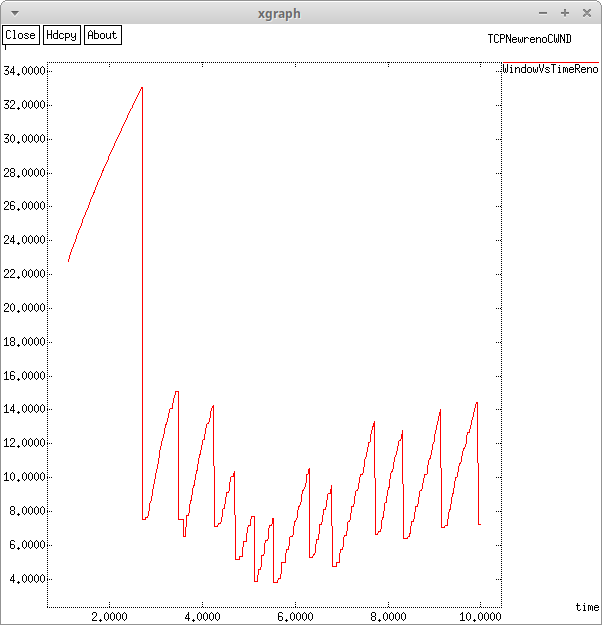


Рис. 10: График изменения размера окна NewReno

После, мы поменяли тип протокола TCP c NewReno на Vegas (рис. 11, рис. 12).

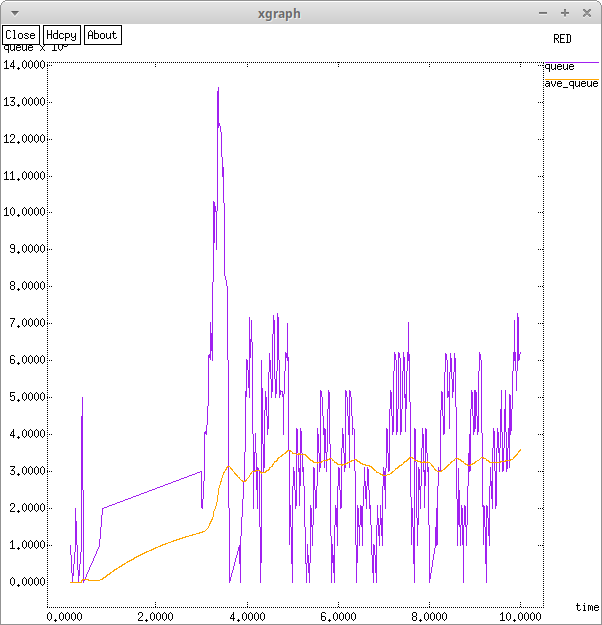


Рис. 11: График фактической и средней длины очереди Vegas

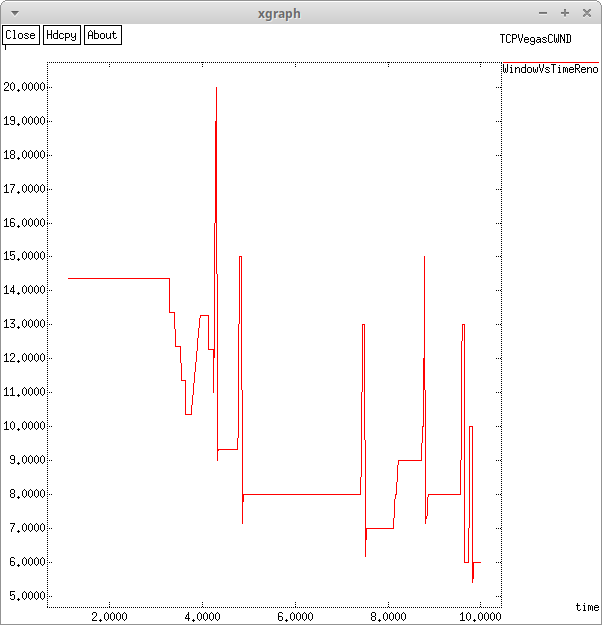


Рис. 12: График изменения размера окна Vegas

Т.к. у нас Reno и NewReno похожи по подходу, графики тоже похожи, но у NewReno стабильнее график и меньше разброс значений, т.к. у Reno увеличение окна происходит линейно и регулируется только при потере пакетов, а у NewReno есть регуляризация принятия пакетов после потери (увеличенный быстрый старт), а у TCP Vegas совершенно иной подход к регуляризации трафика, в следствие чего среднее число пакетов в очереди выглядит еще более стабильно. А также пик размера окна у него сильно меньше, чем у двух предыдущих алгоритмов.

# 5 Выводы

В ходе лабораторной работы мы определили различия между 3-мя протоколами TCP и приобрели базовые навыки работы со средством визуализации xgraph.

# Список литературы