ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Кафедра МСИБ

М.А. Буранова, Н.В. Киреева

ОСНОВЫ МОДЕЛИРОВАНИЯ СЕТЕЙ В СРЕДЕ NS2

Учебно-методические указания

по выполнению лабораторной работы

Самара

2017

Авторы: М.А. Буранова, Н.В. Киреева

УДК 621.395:004.738.5.057.4

Б 912

Рекомендовано к изданию методическим советом ПГУТИ, протокол № 84, от 13. 06. 2017 г.

Буранова М.А., Киреева Н.В.

Б Основы моделирования сетей в среде ns2: учебно-методические указания к проведению лабораторных работ/ М.А. Буранова, Н.В. Киреева. – Самара: ПГУТИ, 2017. – 18 с.

Учебно-методические указания разработаны в соответствии с ФГОС ВО по направлениям подготовки 11.03.02, 11.03.01, 10.05.02, 10.03.01. Предназначены для проведения лабораторных работ по дисциплинам «Технологии обеспечения качества обслуживания в мультисервисных сетях», «Компьютерные сети» «Теория компьютерных сетей».

Лабораторная работа № 1

Цель работы: освоить навыки моделирования в среде ns2.

Контрольные вопросы:

1. Что собой представляет Ns2?
2. Назначение симулятора Ns2.
3. Языки, используемые в Ns2.
4. Команды, используемые для создания модели.
5. Команды, используемые для того, чтобы получить анимационный результат.
6. Опишите процедуру запуска nam.
7. Опишите процедуру создания файлов трассировки.
8. Файлы, используемые для создания модели. Их назначение.
9. Команды, необходимые для создания узлов.
10. Команды, используемые для того, чтобы соединить узлы одним звеном.
11. Агенты, используемые для приема трафика. Способ их задания в ns2.
12. Типы используемого трафика в ns2. Команды, с помощью которых задается трафик.
13. Скорость источника, способ расчета и реализации в ns2.
14. Команда, соединяющая агентов получателя и отправителя.

Задача: получить навыки работы в ОС Unix; освоить основы моделирования в среде ns2; получить навыки запуска и редактирования текстов программ, являющихся основой моделирования; научиться обрабатывать результаты моделирования.

Сетевой симулятор NS2(NETWORK SIMULATOR - 2) представляет собой программное средство для моделирования и анализа функционирования цифровых сетей с коммутацией пакетов.

Симулятор предназначен для использования, как в исследовательских целях – для оценки влияния различных факторов на эффективность разрабатываемых протоколов и приложений, так и в учебных целях – для пояснения работы конкретных протоколов и алгоритмов управления процессами в сетях[1].

Tcl (Tool Command Language)–простой командный язык программирования, который используется при моделировании в пакете ns2.
Nam (Network animator) – визуализатор результатов моделирования.

Ns2 является объектно-ориентированным ПО, ядро которого реализовано на языке C++. Язык скриптов (сценариев) OTcl (Object oriented Tool Command Language) используется в качестве интерпретатора. Ns2 полностью поддерживает иерархию классов C++ (называемую в терминах ns2 компилируемой иерархией) и подобную иерархию классов интерпретатора OTcl (называемую интерпретируемой иерархией).

Методические указания:

Задание 1. Создание модели сети с простой топологией, которая может служить шаблоном для создания сетей с более сложной топологией.

Запустить ОС UNIX.

Создать и назвать файл *.tcl (Этот файл включает в себя самые общие команды для любой моделируемой сети, * - имя файла).

Рассмотрим пример скрипта с обязательными командами.

Необходимо создать объект Simulator (модель).
Это делается помощью команды:

set ns [new Simulator]

Далее для получения анимационного результата моделирования, пользователь может открыть файл для записи данных трассировки для визуализатора nam:

set nf [open out.nam w]
$ns namtrace-all $nf

"*w*" означает, что файл открыт для записи (write). Команда *namtrace-all* определяет тип файла трассировки и содержимое.

В дальнейшем для обработки и анализа результата модели используется стандартный файл трассировки, который практически всегда необходимо открывать пользователям:

set f [open out.tr w]
$ns trace-all $f

out – пример имени для файлов формата .tr и .nam. (может быть любым, но, для удобства, желательно назвать так же как файл *.tcl)

Следующий шаг – это описание процедуры *"finish"*, которая закрывает файлы трассировки и запускает визуализатор nam:

```
proc finish {} {
global ns f nf  # описание глобальных переменных
$ns flush-trace # прекращение трассировки
close $f  # закрытие файлов трассировки
close $nf  # закрытие файлов трассировки
exec nam out.nam & # запуск nam в фоновом режиме
exit 0
}
```

Далее пользователю необходимо добавить alt-событие для планировщика событий, которое запускает процедуру *"finish":*

```
$ns at 5.0 "finish"
```

В данном случае процедура запускается через 5 секунд после начала моделирования

Для запуска модели, в конце скрипта прописывается следующая строка:

```
$ns run
```

Далее файл необходимо сохранить. Запуск программы осуществляется в приложении "Терминал" – через командную строку. Для этого нужно ввести:
*ns *.tcl*

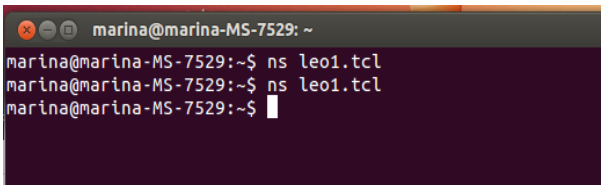

Рис. 1.1.1 Запуск программы в приложении "Терминал"

Если синтаксис файла оказался неверным, программа укажет места ошибок (номер строки, содержащие ошибки), если верным – выдаст результаты моделирования.

Результатами моделирования, в нашем случае, являются следующие файлы: *.nam, *.tr.
Полученные файлы необходимы для выполнения других задач.
*.nam - требуется для запуска визуализатора.
*.tr- требуется для работы с программой **TraceGraph.**

Далее необходимо обработать результаты моделирования и получить конкретные выводы о работе модели.

Задание 2. Создание на базе шаблона (который создавался в задании 1) два узла и одно звено. Назовем файл *.tcl.

Описание топологии необходимо вставлять перед строкой:
"$ns run"

Создадим два новых узла:

```
set n0 [$ns node]
set n1 [$ns node]
```

Далее необходимо соединить два узла одним звеном:

```
$ns duplex-link $n0 $n1 2Mb 10ms DropTail
```

В этой строке указано, что между узлом n0 и n1 создано дуплексное звено с полосой пропускания 2Мбит/с, задержкой 10 мс и очередью с механизмом обслуживания DropTail.

Следующий шаг - создание агента, который посылает данные на узле n0 и другого агента, который принимает данные на узле n1.

```
#Создание агента UDP и присоединение его к узлу n0 трассировки
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

# Создание источника трафика CBR (constant bit rate – с постоянной битовой скоростью) и присоединение его к агенту udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

При помощи этих строк создается агент UDP и присоединяется к узлу n0. Но в ns2 агент сам не может генерировать трафик, он лишь реализует протоколы и алгоритмы транспортного уровня. Поэтому к агенту присоединяется приложение. В данном случае - это источник с постоянной скоростью (CBR - Constant Bit Rate), который каждые 5мс посылает пакет длинной 500 байт . Таким образом, скорость источника -

$$R = \frac{500 \cdot 8}{0.005} = 800000 \frac{\text{бит}}{\text{с}}$$

Также необходимо создать агент-приемник и прикрепить к узлу n1:

set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

В случае использования UPD агент приемник Null (пустой). Известно, что функция транспорта агента UDP заключается только в приеме информации.
В случае моделирования TCP агент называется TCPSink (приемник TCP).

Далее агенты соединяются друг с другом:

$ns connect $udp0 $null0

Теперь зля запуска и остановки приложения CBR необходимо добавить alt-события в планировщик событий.

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"

В данном задании запуск приложения происходит через 0.5 секунды модельного времени и останавливается на 4.5 секунде.
Далее файл необходимо сохранить. Запуск программы осуществляется в приложении "Терминал" – через командную строку. Для этого нужно ввести:
*ns *.tcl*
После запуска *.tcl создадутся два трейс-файла: *.nam и *.tcl
После того, как создали файлы, в "Терминале" следует открыть файл *.nam, который запустит визуализатор nam (рис.1. 2.1).

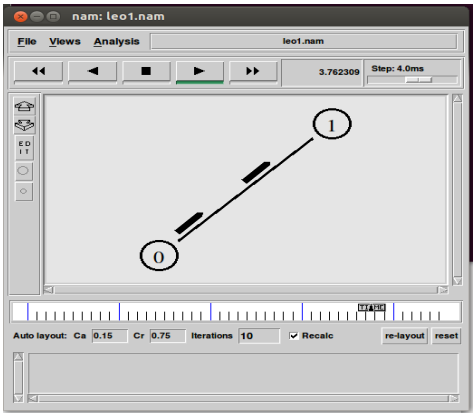

Рис. 1.2.1 Визуализатор nam при анимации результатов моделирования файла *.tcl

Полученную модель и код занести в отчет и сделать вывод.

Под панелью меню находятся кнопки перемотки назад, запуска анимации в обратную сторону, остановки анимации, запуска анимации, перемотки вперед и выхода из nam, индикатор текущего времени анимации и движок изменения скорости анимации (текущая скорость изображена над ним).
Также возможно увеличения и уменьшения изображения с помощью кнопок расположенных в левой части экрана.
Изменять текущее время анимации пользователь может при помощи движка времени анимации.

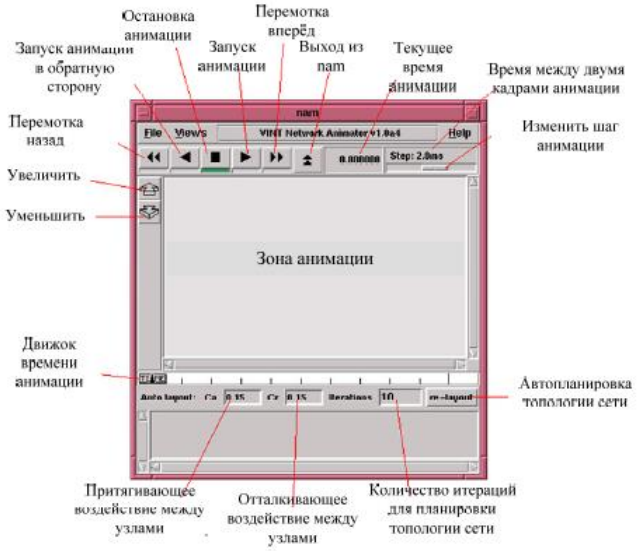

Рис.1.2.2 Пользовательский интерфейс визуализатора nam

Задание 3. Задание топологии.

Нужно создать файл *.tcl, который будет содержать сеть с 4 узлами, один из которых является промежуточным узлом. В качестве шаблона для этого файла служит файл из задания 2 формата *.tcl.

На рисунке 1.3.1 изображен пример сети с топологией, модель которой необходимо построить.

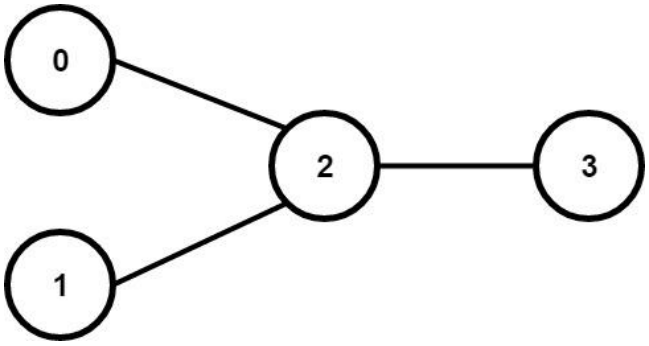

Рис.1.3.1 Пример топологии.

Создадим четыре узла:

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

Необходимо задать скорости задаваемых потоков:

set rate0 10000
set rate1 10000

Далее создаются три дуплексных звена:

$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n3 $n2 2Mb 10ms DropTail

Для того чтобы данная топология была наглядной при анимации в визуализаторе nam необходимо для каждого звена указать направление:

$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

Создадим два агента - один агент *UDP0* с прикрепленным к нему источником *CBR0*, второй - агент *UDP1* с источником *CBR1*.

#Создание агента UDP0 и присоединение его к узлу n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

 # Создание источника CBR0-трафика и присоединение его к агенту udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 600
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

#Создание агента UDP1 и присоединение его к узлу n1
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1

 # Создание источника CBR1-трафика и присоединение его к агенту udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 600
$cbr1 set interval_ 0.005
$cbr1 attach-agent $udp1

#Создание агента - получателя для udp0
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

#Создание агента - получателя для udp1
set null1 [new Agent/Null]
$ns attach-agent $n3 $null1

Далее необходимо соединить агенты udp0 и udp1 и их получатели:

$ns connect $udp0 $null0
$ns connect $udp1 $null1

Для того, чтобы программа начала работать необходимо добавить alt-события :

$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$ cbr1 start"

$ns at 5.0 "$cbr0 stop"
$ns at 5.0 "$cbr1 stop"

Чтобы отличить оба потока в визуализаторе nam в скрипт нужно вставить строки, которые позволят изображать потоки от разных источников разными цветами:

$udp0 set class_ 1
$udp1 set class_ 2

Но перед этим необходимо описать цвет каждого класса:

$ns color 1 Blue
$ns color 2 Red

Таким образом, цвет udp1 - пакетов – красный, а цвет udp0 -пактов – синий.
Для того, чтобы пользователь увидел, что происходит внутри очереди, необходимо добавить строку:

$ns duplex-link-op $n2 $n3 queuePos 0.5
$ns run

В ней определяется место положения изображения очереди в зоне анимации визуализатора nam.
Далее файл необходимо сохранить. Запуск программы осуществляется в приложении "Терминал" – через командную строку. Для этого нужно ввести:
*ns *.tcl*
После того, как создали файлы, в "Терминале" следует открыть файл *.nam, который запустит визуализатор nam.

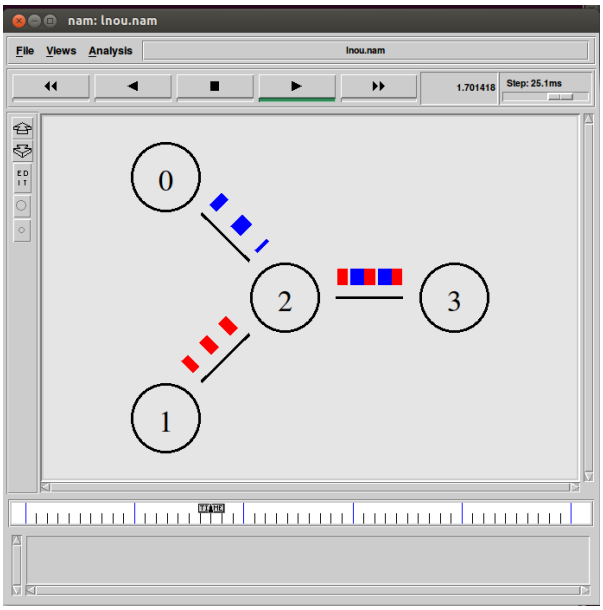

Рис.1.3.2 Визуализатор nam при анимации результатов моделирования файла *.tcl

Полученную модель и код следует занести в отчет и сделать вывод.

Лабораторная работа № 2

Цель работы: получить навыки работы в программе Tracegraph.

Контрольные вопросы:

1. Назначение окна Trace graph 2.02.
2. Назначение окна Graphs.
3. Назначение окна Network Information
4. Формат файла используемый в программе Tracegraph. Его создание.
5. Определение задержки.
6. Определение джиттера.
7. Определение вероятности потерь.
8. Влияние размера пакета на вероятность потерь.
9. Влияние размера пакета на задержку?
10. Влияние размера пакета на джиттер?
11. Определение QoS.
12. Факторы, определяющие качество передачи.
13. Определение задержки сериализации.
14. Определение доступности сервиса.
15. Определение задержки распространения.
16. Определение пропускной способности.
17. Требования для передачи видеоданных.

Базовые понятия QoS

Качество обслуживания (Quality of Service, QoS) определяется как мера производительности передающей системы, отражающая качество передачи и доступность услуг [5].

Качество передачи определяется следующими факторами:

— доступность (Availability):
– сетевая доступность — диапазон времени сетевой достижимости между входной и выходной точкой сети;
– доступность сервиса (Service Availability) — диапазон времени, в течение которого этот сервис доступен между определёнными входной и выходной точками с параметрами, оговорёнными в соглашении об уровне обслуживании (Service Level Agreement — SLA);
— потери пакетов (Packet Loss) — отношение правильно принятых пакетов к общему количеству пакетов, которые были переданы по сети;
— задержка (Delay) — время, которое требуется пакету для того, чтобы после передачи дойти до пункта назначения:

– задержка сериализации (Serialization Delay)1 — время, которое требуется устройству для передачи пакета заданного размера при заданной ширине полосы пропускания;
– задержка распространения (Propagation Delay) — время, которое требуется переданной в канал единице информации для достижения принимающего устройства (зависит от расстояния и среды передачи);
— колебания задержки (Packet Jitter) — разница между сквозным временем задержки, которая возникает при передаче по сети разных пакетов;
— пропускная способность (Bandwidth) — общее количество данных, которые могут быть переданы в единицу времени между двумя точками присутствия оператора.

В пакетных сетях в информационном потоке может передаваться разнородный трафик, характеризующийся критичными и второстепенными для себя параметрами. Для передачи аудио- и видеоданных требуются разные требования к QoS. Для передачи видеоданных необходима высокая пропускная способность и стабильное время задержки при передаче. При этом, чтобы избежать искажений изображения, необходим стационарный поток данных. При интерактивной передаче звука требуется меньше пропускной способности канала, чем при передаче видео, но необходима малая задержка прохождения пакетов через сеть, иначе возникает «эхо». Передача файлов требует высокой пропускной способности, но, задержку сериализации называют ещё задержкой передачи (Transmission Delay).
Так, например, если для передачи одного пакета по сети требуется 100 мсек, а для передачи следующего пакета — 125 мсек, то колебание задержки составит 25 мсек.
QoS и передача мультимедийных данных отличие от большинства других видов сетевого трафика, наименее чувствительна к длительным и непостоянным задержкам в сети.
Качество обслуживания использует распределение по категориям и назначение приоритетов трафикам, что позволяет гарантировать трафику с большим проритетом лучшие условия передачи через сетевую магистраль, вне зависимости от требований к пропускной способности трафика менее важных приложений.

Методические указания:

Задание 1. Создайте 5 файлов формата *.tr, с разными размерами пакетов, взяв для шаблона задание 3 (из ЛР 1). То есть первый файл будет иметь размер пакета pacetSize 1, второй -pacetSize 2 и тд.
Файлы можно, для удобства, назвать в зависимости от размера пакета, например *5.tr (5 - будет означать пакет длиной 500 байт).

Вариант / Скорость	1	2	3	4	5
pacetSize 1	500	510	520	530	540
pacetSize 2	600	610	620	630	640
pacetSize 3	700	710	720	730	740
pacetSize 4	800	810	820	830	840
pacetSize 5	900	910	920	930	940

Полученный код занести в отчет

После создания файлов следует выйти из ОС UNIX. И зайти в ОС Windows XP.
Далее работа будет выполняться в программе **Tracegraph.**

Работа с программой Tracegraph.

При запуске программы Tracegraph открывается три окна для вывода информации о результатах моделировании.
Окно Trace graph 2.02 (рис.2.1.1) - основное окно для работы в программе. В нем вводятся условия и начальные данные для построения графиков и вывода статистики (текущий узел, идентификатор потока, начальное и конечное время моделирования, типы пакетов и др.). В этом окне нужно открыть файл формата *. tr

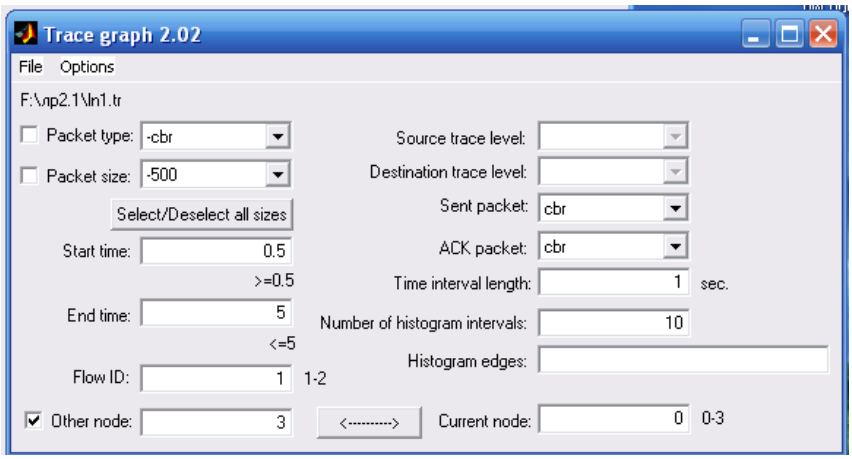

Рис. 2.1.1 Окно Trace graph 2.02

После в того, как открыли файл в полях **Current node** (текущий узел – источник трафика) и **Other node** (другой узел – получатель трафика) следует задать маршрут, а также во вкладке **Options** следует отметить **Count packets IDs only once**.

Graphs – окно вывода графиков.

В окне Graphs для *каждого* файла выведите график джиттера и задержки.

Для этого выберите вкладку **2D Graphs/Jitter/Receive events time vs jitter between CN and ON** (Рис. 2.1.2) и **2D Graphs/Event time vs delay/Receive event time vs CN2ON delay** (Рис. 2.1.3).

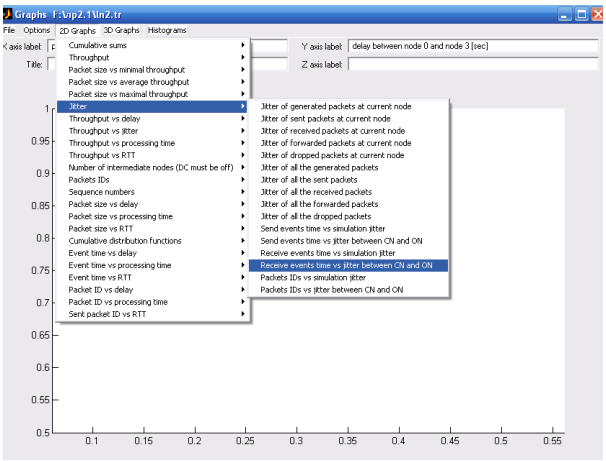

Рис. 2.1.2 Вывод графика Receive events time vs jitter between CN and ON

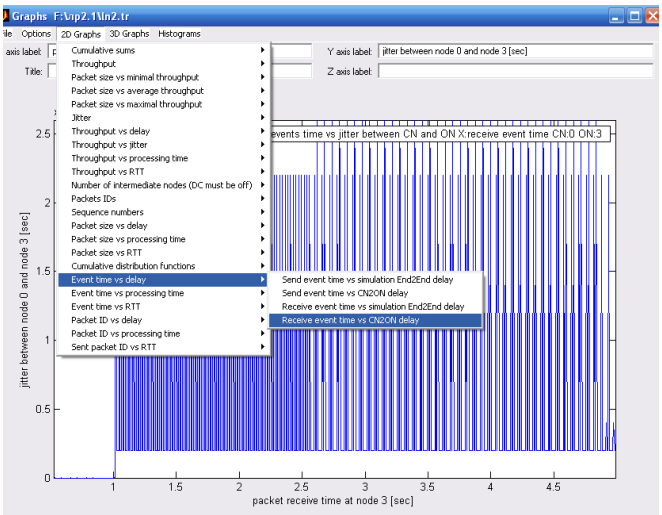

Рис. 2.1.3 Вывод графика Receive event time vs CN2ON delay

Полученные графики занести в отчет.

Network Information – окно вывода статистики. Данное окно выводит статистические данные по результатам моделирования, которые размещаются в нескольких разделах. Чтобы отобразить нужную статистику, нужно нажать вкладку Network information и выбрать все параметры, которые включаются в эту вкладку.

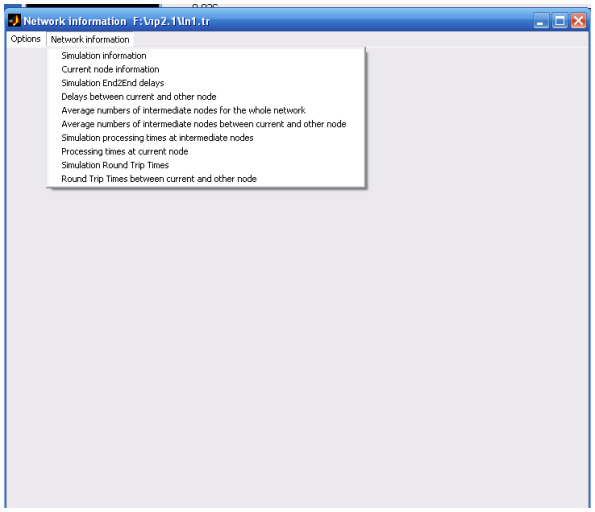

Рис. 2.1.2 Network Information

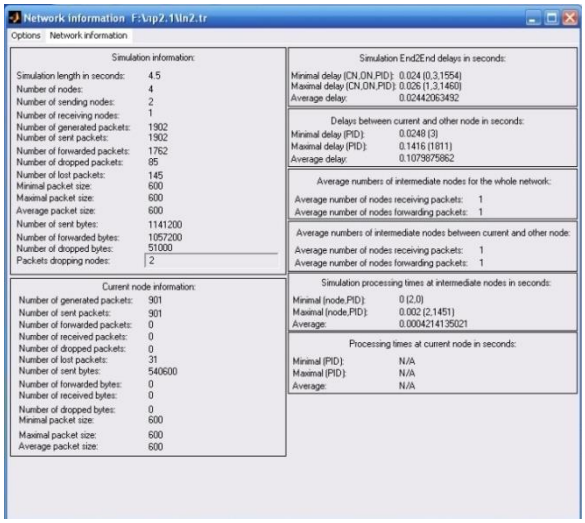

Рис. 2.1.3 Полученная статистка файла .tr

Отобразите статистику каждого файла и **постройте зависимость изменения вероятности потери пакетов (Number of lost packets) и задержки (Average delay) от размеров пакета.** Сделайте вывод исходя из полученных графиков. Так же полученную статистику каждого файла занести в отчет.

Задание 2

Создайте модель, в которой необходимо задать маршрут (по варианту) для данной топологии (рис 2.2.1)

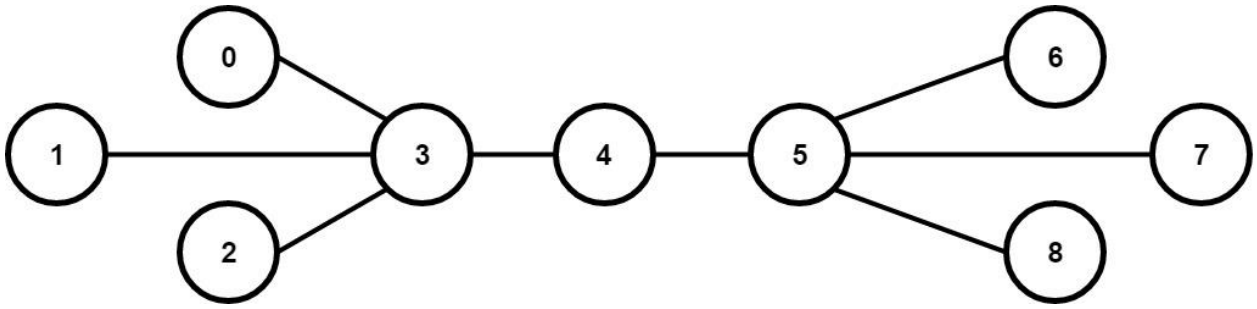

Рис .2.2.1

Вариант	1	2	3	4	5
Маршрут	0-3-4	1-3-8	8-5-0	7-5-2	2-3-6

Литература.

1. В.С. Заборовский, В.А. Мулюха, Ю.Е. Подгурский. Сети ЭВМ и Телекоммуникации. Моделирование и анализ компьютерных сетей: Телематический подход: Учебное пособие. Санкт-Петербург, издательсво СПбГПУ./ Зав. каф. РВКС, проф., д.т.н. Ю.Г. Карпов. Зам. директора по научной работе СПИИРАН, заслуженный деятель науки, проф., д.т.н. А.В. Смирнов.
2. П.В.Москвин.Азбука Tcl.,Горячая Линия Телеком,2003

3. Б.Уэлш, К.Джонс, Дж.Хоббос. Практическое программирование на Tcl и Tk, М.Вильямс, 2004.

4. Петровский А.И. Командный язык программирования Tcl(Tool Command Language).-М.Майор,2001.

5. Кулябов, Д.С. Архитектура и принципы построения современных сетей и систем телекоммуникаций телекоммуникаций: Учеб. пособие [Текст] /Д.С. Кулябов, А.В. Королькова — М.: РУДН, 2008. — 281 с.