Лабораторная работа 1

Простые модели компьютерной сети

Адабор Кристофер Твум – НКН -01-22

Содержание

1. [Цель работы 1](#_Toc190549899)
2. [Задание 1](#_Toc190549900)
3. [Выполнение лабораторной работы 2](#_Toc190549901)
4. [Шаблон сценария для NS-2 2](#_Toc190549902)
5. [Простой пример описания топологии сети, состоящей из двух узлов и одного соединения 4](#_Toc190549903)
6. [Пример с усложнённой топологией сети 6](#_Toc190549904)
7. [Пример с кольцевой топологией сети 8](#_Toc190549905)
8. [Выводы 14](#_Toc190549906)

# Цель работы

Приобретите навыки моделирования сетей передачи данных с помощью инструмента моделирования NS2, а также анализа результатов моделирования.

# Задание

1. Создать шаблон сценария для NS-2;
2. Выполнить простой пример описания топологии сети, состоящей из двух узлов и одного соединения;
3. Выполнить пример с усложнённой топологией сети;
4. Выполнить пример с кольцевой топологией сети;
5. Выполнить упражнение.

# Выполнение лабораторной работы

## Шаблон сценария для NS-2

В нашем рабочем каталоге мы создадим каталог tmp, в котором будет выполняться лабораторная работа. Внутри карты мы создадим каталог lib-ns, а в нем файл shablon.tpl (рис. 1).

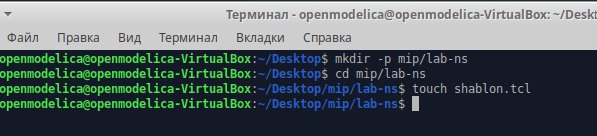


Рис. 1: Создание директорий и файла

Давайте откроем файл shablon.tcl для редактирования (рис. 2).

Сначала давайте создадим объект типа Simulator. Затем мы создадим переменную nf и укажем, что она требуется для открытия файла nam для записи выходных результатов моделирования. Вторая строка предписывает симулятору записывать все данные о динамике модели в файл out.name . Затем создайте переменную f и откройте файл трассировки для записи, чтобы зарегистрировать все события модели. После этого мы добавим процедуру finish, которая закроет файлы трассировки и запустит nam. Используя команду at, мы сообщаем планировщику событий, что запустим процедуру завершения через 5 секунд после начала симуляции, после чего запустим симулятор ns.

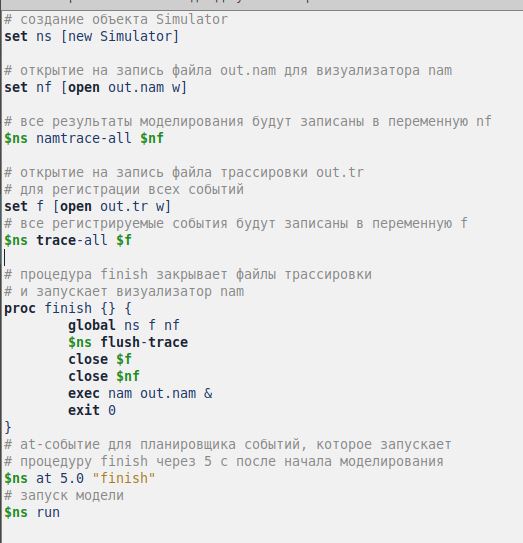


Рис. 2: Редактирование файла shablon.tcl

После сохранения изменений в отредактированном файле shablon.tcl и закрытия его запустите симулятор командой ns shablon.tcl. Мы увидим пустую область моделирования, поскольку еще не были определены никакие объекты или действия (рис. 3).

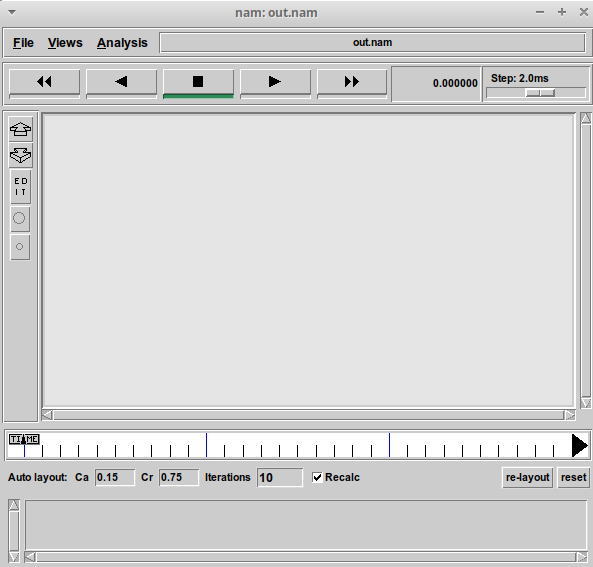


Рис. 3: Запуск шаблона сценария для NS-2

## Простой пример описания топологии сети, состоящей из двух узлов и одного соединения

Требуется смоделировать сеть передачи данных, состоящую из двух узлов, соединённых дуплексной линией связи с полосой пропускания 2 Мб/с и задержкой 10 мс, очередью с обслуживанием типа DropTail. От одного узла к другому по протоколу UDP осуществляется передача пакетов, размером 500 байт, с постоянной скоростью 200 пакетов в секунду.

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example1.tcl и откроем example1.tcl на редактирование. Добавим в него до строки $ns at 5.0 "finish" описание топологии сети. Создадим агенты для генерации и приёма трафика. Создается агент UDP и присоединяется к узлу n0. В узле 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 (Constant Bit Rate, CBR), который каждые 5 мс посылает пакет R = 500 байт. Таким образом, скорость источника:

.

Далее мы создадим нулевой агент, который будет выступать в качестве получателя трафика, и присоединим его к узлу n1. Давайте соединим агенты вместе. Для запуска и остановки приложения CBR в планировщик событий добавляются события at (перед командой $ns at 5.0 “готово") (рис. 4).



Рис. 4: Пример описания топологии сети, состоящей из двух узлов и одного соединения

Сохранив изменения в отредактированном файле и запустив симулятор, мы получаем в результате nam animator, работающий в фоновом режиме (рис. 5).

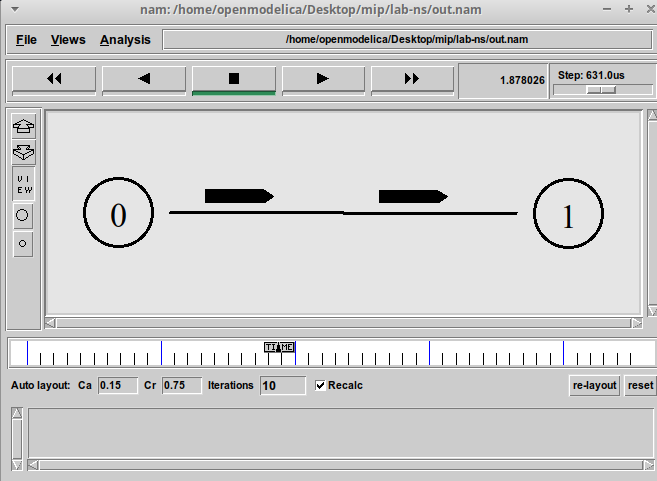


Рис. 5: Визуализация простой модели сети с помощью nam

При нажатии на кнопку play в окне nam через 0.5 секунды из узла 0 данные начнут поступать к узлу 1.

## Пример с усложнённой топологией сети

**Описание моделируемой сети:**

* сеть состоит из 4 узлов (n0, n1, n2, n3);
* между узлами n0 и n2, n1 и n2 установлено дуплексное соединение с пропускной способностью 2 Мбит/с и задержкой 10 мс;
* между узлами n2 и n3 установлено дуплексное соединение с пропускной способностью 1,7 Мбит/с и задержкой 20 мс;
* каждый узел использует очередь с дисциплиной DropTail для накопления пакетов, максимальный размер которой составляет 10;
* TCP-источник на узле n0 подключается к TCP-приёмнику на узле n3 (по-умолчанию, максимальный размер пакета, который TCP-агент может генерировать, равняется 1KByte)
* TCP-приёмник генерирует и отправляет ACK пакеты отправителю и откидывает полученные пакеты;
* UDP-агент, который подсоединён к узлу n1, подключён к null-агенту на узле n3 (null-агент просто откидывает пакеты);
* генераторы трафика ftp и cbr прикреплены к TCP и UDP агентам соответственно;
* генератор cbr генерирует пакеты размером 1 Кбайт со скоростью 1 Мбит/с;
* работа cbr начинается в 0,1 секунду и прекращается в 4,5 секунды, а ftp начинает работать в 1,0 секунду и прекращает в 4,0 секунды.

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example2.tcl и откроем example2.tcl на редактирование. Создадим 4 узла и 3 дуплексных соединения с указанием направления (рис. 6).

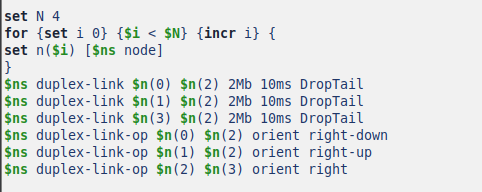


Рис. 6: Визуализация простой модели сети с помощью nam

Создадим агент UDP с прикреплённым к нему источником CBR и агент TCP с прикреплённым к нему приложением FTP (рис. 7).

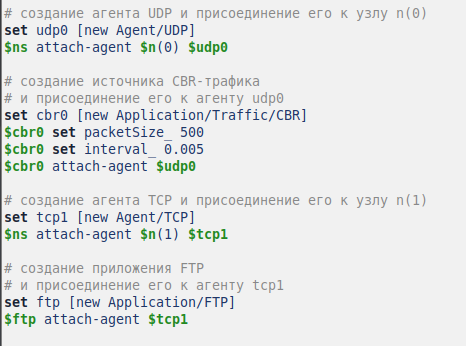


Рис. 7: Описание усложненной топологии сети

Создадим агенты-получатели. Соединим агенты udp0 и tcp1 и их получателей. Зададим описание цвета каждого потока. Выполним отслеживание событий в очереди и наложение ограничения на размер очереди. Добавим at-события (рис. 8).

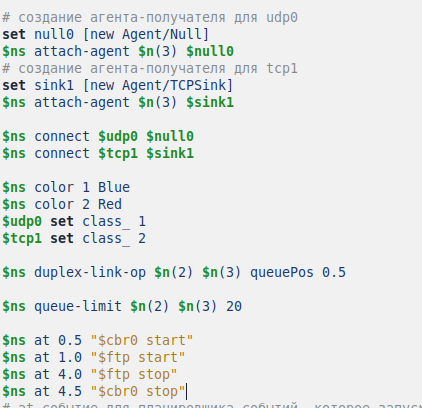


Рис. 8: Описание усложненной топологии сети

Сохранив изменения в отредактированном файле и запустив симулятор, получим анимированный результат моделирования (рис. 9).

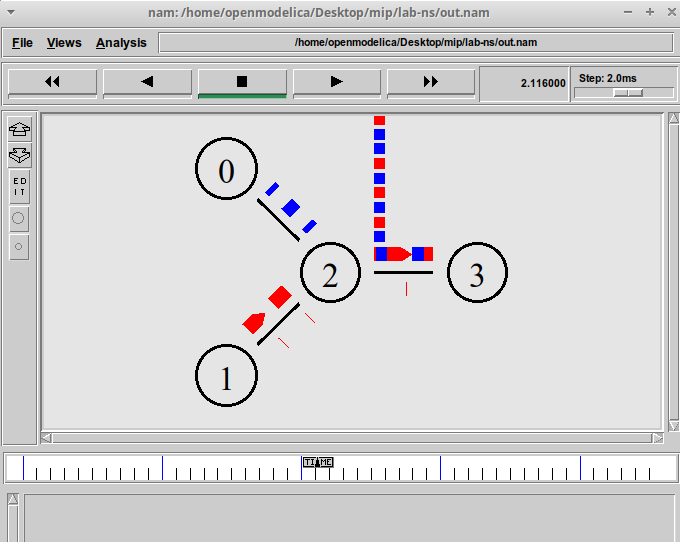


Рис. 9: Описание усложненной топологии сети

## Пример с кольцевой топологией сети

**Описание модели передачи данных по сети с кольцевой топологией и динамической маршрутизацией пакетов:**

* сеть состоит из 7 узлов, соединённых в кольцо;
* данные передаются от узла n(0) к узлу n(3) по кратчайшему пути;
* с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(1) и n(2);
* при разрыве соединения маршрут передачи данных должен измениться на резервный.

Скопируем содержимое созданного шаблона в новый файл: cp shablon.tcl example3.tcl и откроем example3.tcl на редактирование. Опишем топологию моделируемой сети (рис. 10). Далее соединим узлы так, чтобы создать круговую топологию. Каждый узел, за исключением последнего, соединяется со следующим, последний соединяется с первым. Для этого в цикле использован оператор %, означающий остаток от деления нацело. Зададим передачу данных от узла n(0) к узлу n(3). Данные передаются по кратчайшему маршруту от узла n(0) к узлу n(3), через узлы n(1) и n(2) (рис. 11). Добавим команду разрыва соединения между узлами n(1) и n(2) на время в одну секунду, а также время начала и окончания передачи данных.



Рис. 10: Описание кольцевой топологии сети и динамической маршрутизацией пакетов

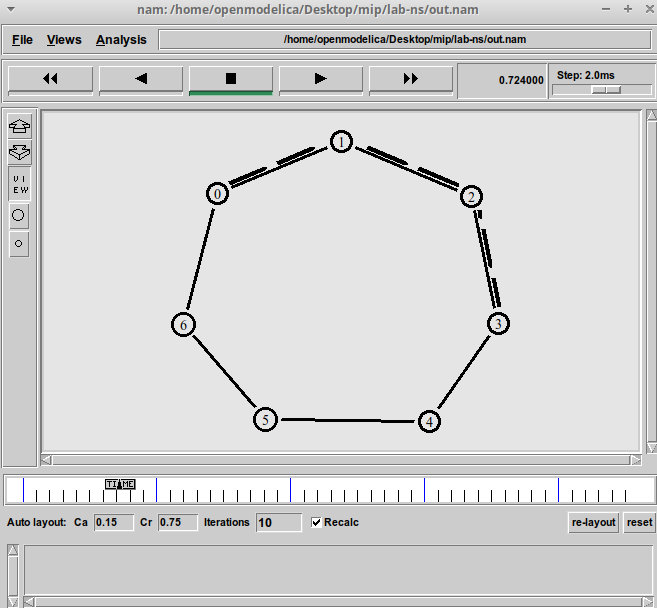


Рис. 11: Передача данных по кратчайшему пути сети с кольцевой топологией

Передача данных при кольцевой топологии сети в случае разрыва соединения представлена на рис. 12.

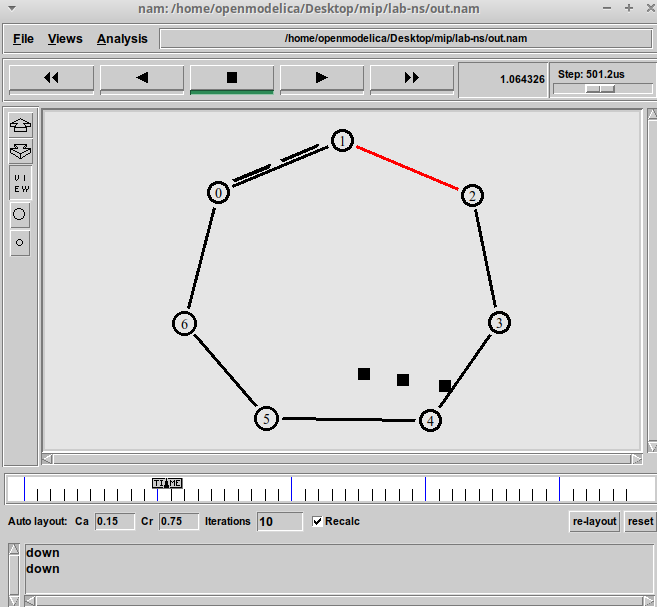


Рис. 12: Передача данных по сети с кольцевой топологией в случае разрыва соединения

Добавив в начало скрипта после команды создания объекта Simulator:

$ns rtproto DV

увидим, что сразу после запуска в сети отправляется небольшое количество маленьких пакетов, используемых для обмена информацией, необходимой для маршрутизации между узлами (рис. 13). Когда соединение будет разорвано, информация о топологии будет обновлена, и пакеты будут отсылаться по новому маршруту через узлы n(6), n(5) и n(4).

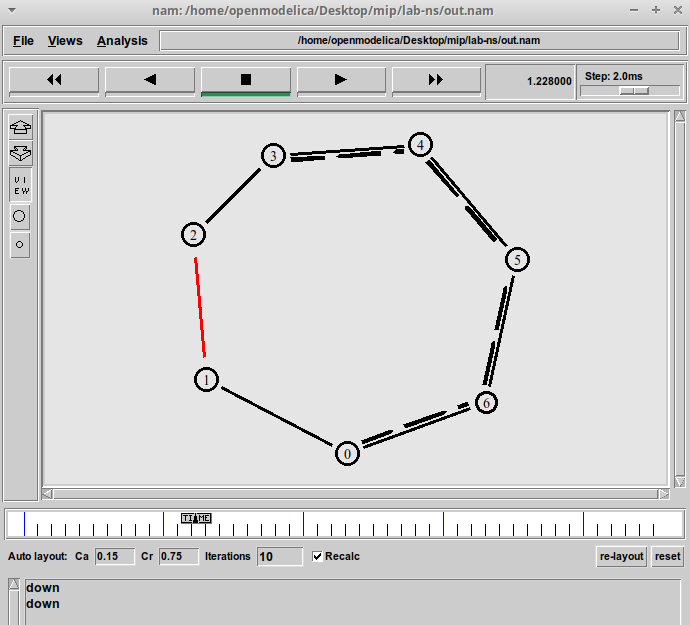


Рис. 13: Маршрутизация данных по сети с кольцевой топологией в случае разрыва соединения

**Упражнение**

Внесем следующие изменения в реализацию примера с кольцевой топологией сети:

* передача данных должна осуществляться от узла n(0) до узла n(5) по кратчайшему пути в течение 5 секунд модельного времени;
* передача данных должна идти по протоколу TCP (тип Newreno), на принимающей стороне используется TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени;
* с 1 по 2 секунду модельного времени происходит разрыв соединения между узлами n(0) и n(1);
* при разрыве соединения маршрут передачи данных должен измениться на резервный, после восстановления соединения пакеты снова должны пойти по кратчайшему пути.

Изменим количество узлов в кольце на 5, а 6 узел n(5) отдельно присоединим к узлу n(1). Вместо агента UDP создадим агента TCP (типа Newreno), а на принимающей стороне используем TCPSink-объект типа DelAck; поверх TCP работает протокол FTP с 0,5 до 4,5 секунд модельного времени Также зададим с 1 по 2 секунду модельного времени разрыв соединения между узлами n(0) и n(1)(14).

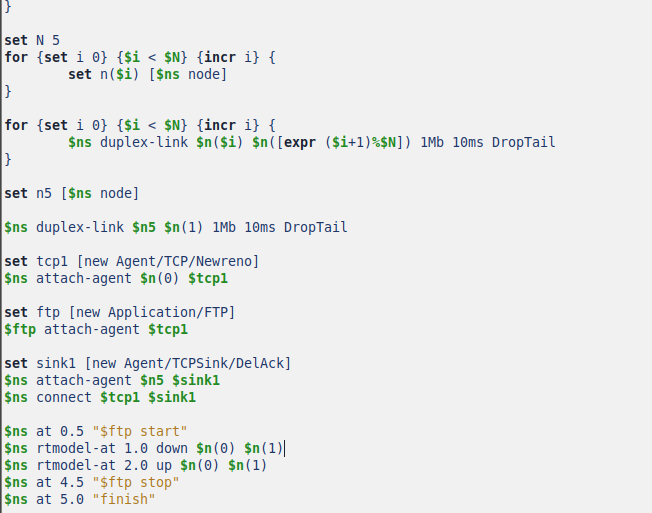


Рис. 14: Программа для упражнения по построению топологии сети

Запустим программу и увидим, что пакеты идут по кратчайшему пути через узел n(1) (15).

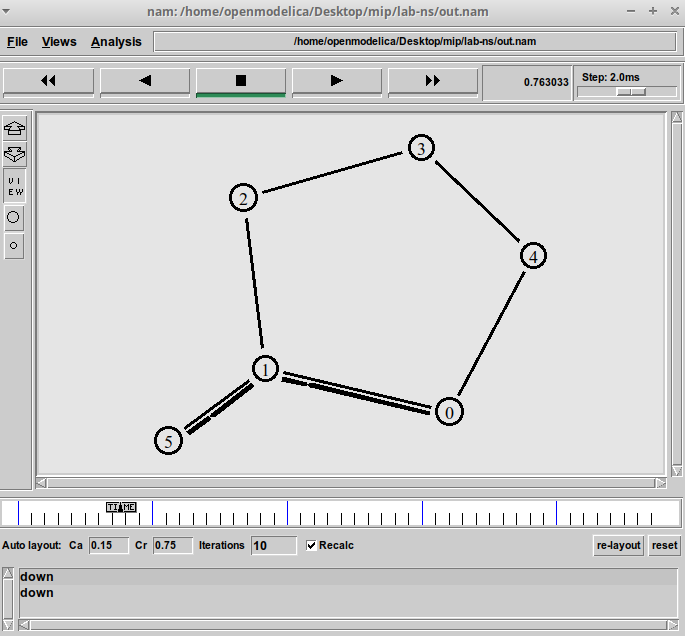


Рис. 15: Передача данных по изменённой кольцевой топологии сети

При разрыве соединения часть пакетов теряется, но поскольку данные обновляются пакеты начинают идти по другому пути (16).

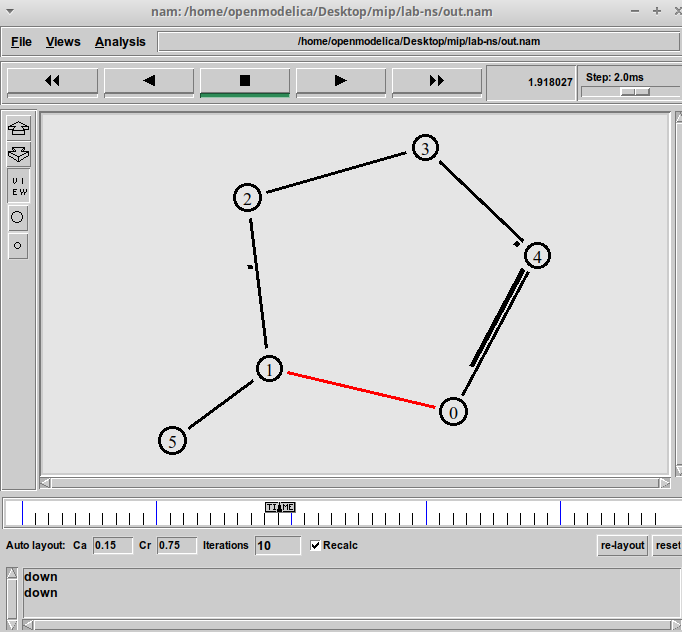


Рис. 16: Передача данных по сети в случае разрыва соединения

После восстановления соединения пакеты снова идут по кратчайшему пути (17).

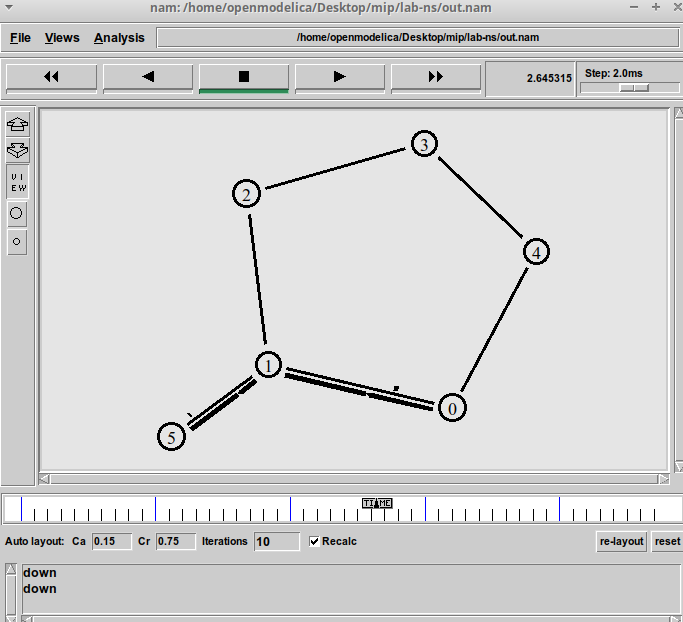


Рис. 17: Передача данных после восстановления соединения

# Выводы

В процессе выполнения данной лабораторной работы я приобрела навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также проанализировала полученные результаты моделирования.