

# **Лабораторная работа №1**

## **Простые модели компьютерной сети**

Игнатенкова Варвара Николаевна

НКНбд-01-22

# Цель работы

Приобретение навыков моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2, а также анализ полученных результатов моделирования.

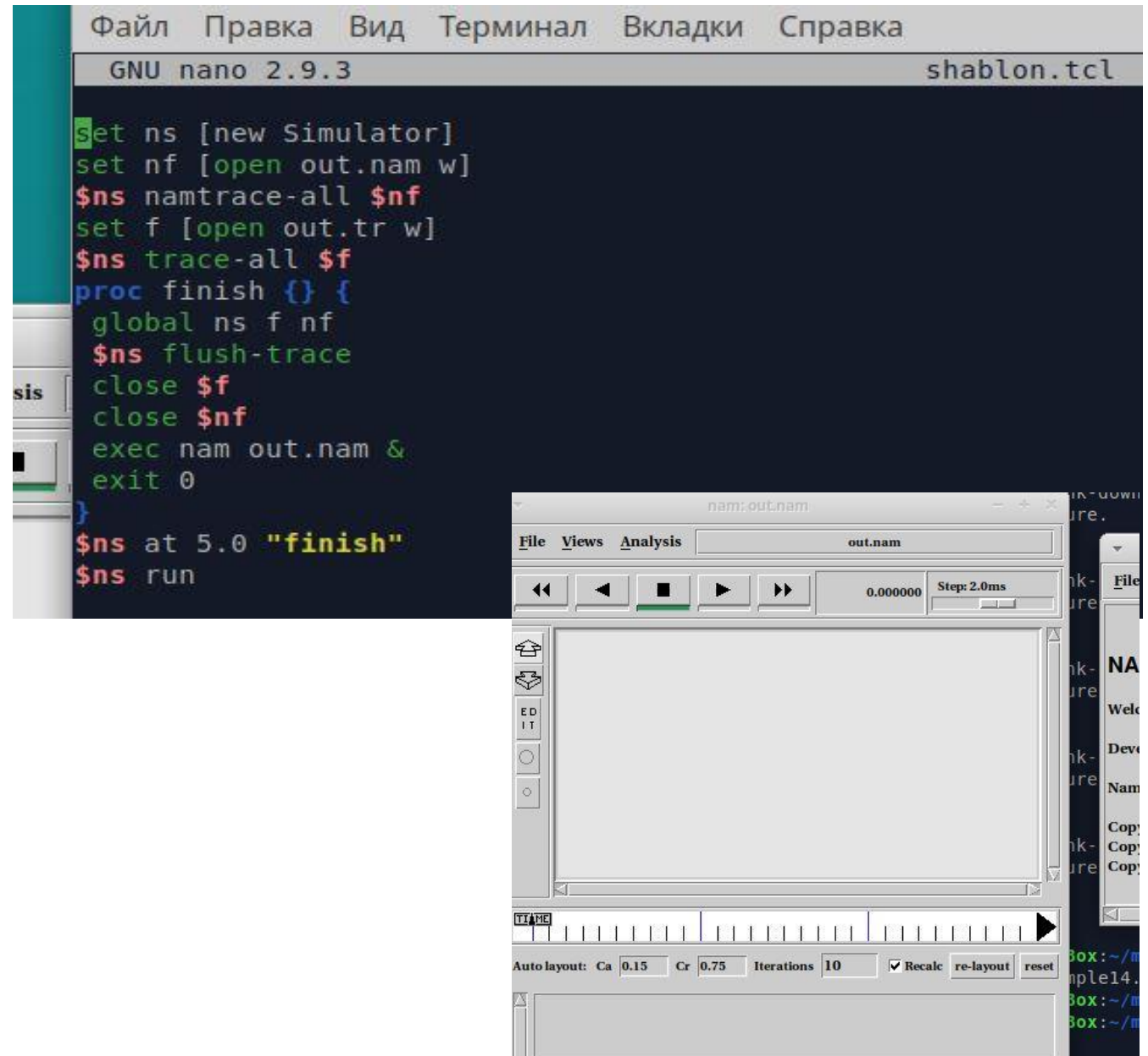
# Выполнение лабораторной работы

# Шаблон сценария для NS-2

В своём рабочем каталоге создадим директорию `mir`, к которой будут выполняться лабораторные работы. Внутри `mir` создадим директорию `lab-ns`, а в ней файл `shablon.tcl`

Сохранив изменения в отредактированном файле `shablon.tcl` и закрыв его, можно запустить симулятор командой: `ns shablon.tcl`

При этом на экране появится сообщение типа `nam: empty trace file out.nam` поскольку ещё не определены никакие объекты и действия.



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

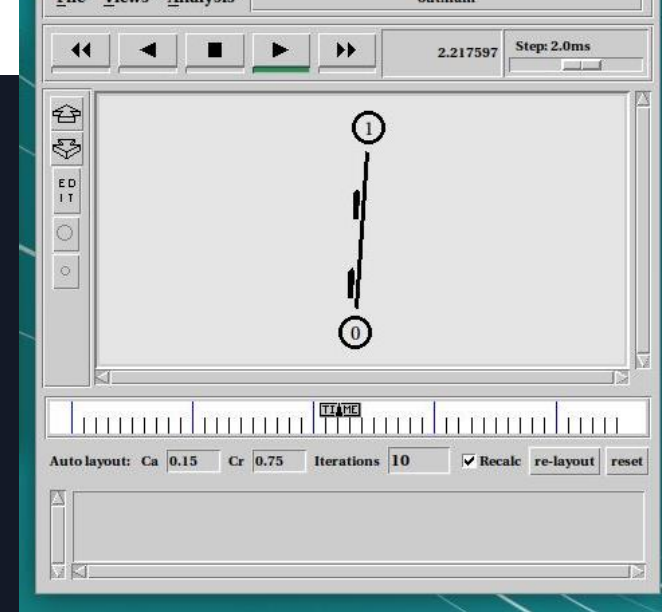
**Постановка задачи.** Требуется смоделировать сеть передачи данных, состоящую из двух узлов, соединённых дуплексной линией связи с полосой пропускания 2 Мб/с и задержкой 10 мс, очередью с обслуживанием типа DropTail. От одного узла к другому по протоколу UDP осуществляется передача пакетов, размером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

При нажатии на кнопку play в окне nam через 0.5 секунды из узла 0 данные начнут поступать к узлу 1. Это процесс можно замедлить, выбирая шаг отображения в nam. Можно осуществлять наблюдение за отдельным пакетом, щёлкнув по нему в окне nam, а щёлкнув по соединению, можно получить о нем некоторую информацию.

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set f [open out.tr w]
$ns trace-all $f
proc finish {} {
    global ns f nf
    $ns flush-trace
    close $f
    close $nf
    exec nam out.nam &
    exit 0
}

set N 2
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

$ns duplex-link $n(0) $n(1) 2Mb 10ms DropTail
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
set null0 [new Agent/Null]
$ns attach-agent $n(1) $null0
$ns connect $udp0 $null0
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```



# Пример с усложнённой топологией сети

**Постановка задачи.** Описание моделируемой сети (рис. 2.4):– сеть состоит из 4 узлов ( $n_0$ ,  $n_1$ ,  $n_2$ ,  $n_3$ );

- между узлами  $n_0$  и  $n_2$ ,  $n_1$  и  $n_2$  установлено дуплексное соединение с пропускной способностью 2 Мбит/с и задержкой 10 мс;

- между узлами  $n_2$  и  $n_3$  установлено дуплексное соединение с пропускной способностью 1,7 Мбит/с и задержкой 20 мс;

- каждый узел использует очередь с дисциплиной Drop Tail для накопления пакетов, максимальный размер которой составляет 10;

- TCP-источник на узле  $n_0$  подключается к TCP-приёмнику на узле  $n_3$

(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  $n_1$ , подключён к null-агенту на узле  $n_3$  (null-агент просто откидывает пакеты);

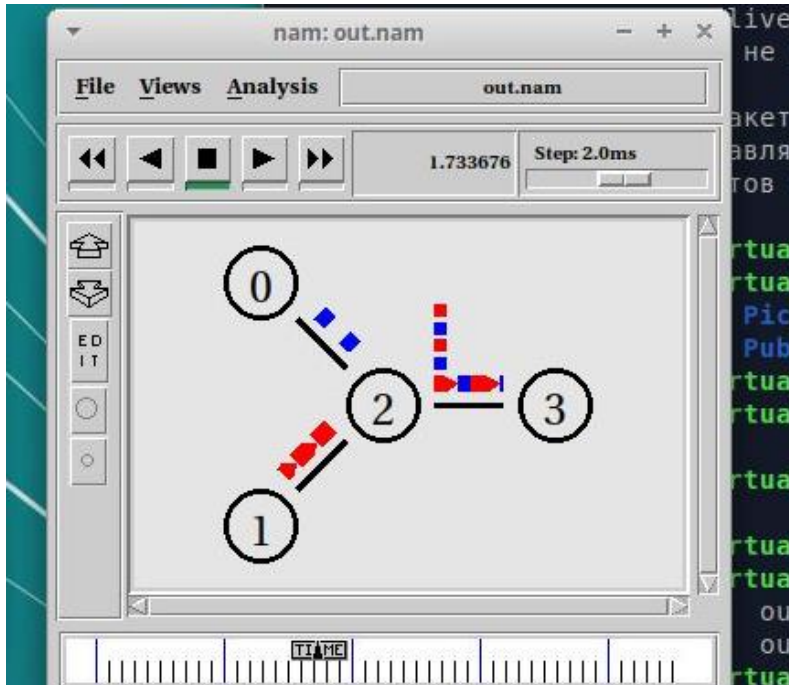
- генераторы трафика ftp и cbr прикреплены к TCP и UDP агентам соответственно;

- 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.



При запуске скрипта можно заметить, что по соединениям между узлами  $n(0)$ – $n(2)$  и  $n(1)$ – $n(2)$  к узлу  $n(2)$  передаётся данных больше, чем способно передаваться по соединению от узла  $n(2)$  к узлу  $n(3)$ . Действительно, мы передаём 200 пакетов в секунду от каждого источника данных в узлах  $n(0)$  и  $n(1)$ , а каждый пакет имеет размер 500 байт. Таким образом, полоса каждого соединения 08 Mb, а суммарная — 16Mb. Но соединение  $n(2)$ – $n(3)$  имеет полосу лишь 1 Mb. Следовательно, часть пакетов должна теряться. В окне аниматора можно видеть пакеты в очереди, а также те пакеты, которые отбрасываются при переполнении.



```
GNU nano 2.9.3 example12.tcl

set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set f [open out.tr w]
$ns trace-all $f
proc finish {} {
    global ns f nf
    $ns flush-trace
    close $f
    close $nf
    exec nam out.nam &
    exit 0
}

set N 4
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

$ns duplex-link $n(0) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(3) $n(2) 2Mb 10ms DropTail
$ns duplex-link-op $n(0) $n(2) orient right-down
$ns duplex-link-op $n(1) $n(2) orient right-up
$ns duplex-link-op $n(2) $n(3) orient right
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
set tcp1 [new Agent/TCP]
$ns attach-agent $n(1) $tcp1
set ftp [new Application/FTP]
$ftp attach-agent $tcp1
set null0 [new Agent/Null]
$ns attach-agent $n(3) $null0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n(3) $sink1
$ns connect $udp0 $null0
$ns connect $tcp1 $sink1

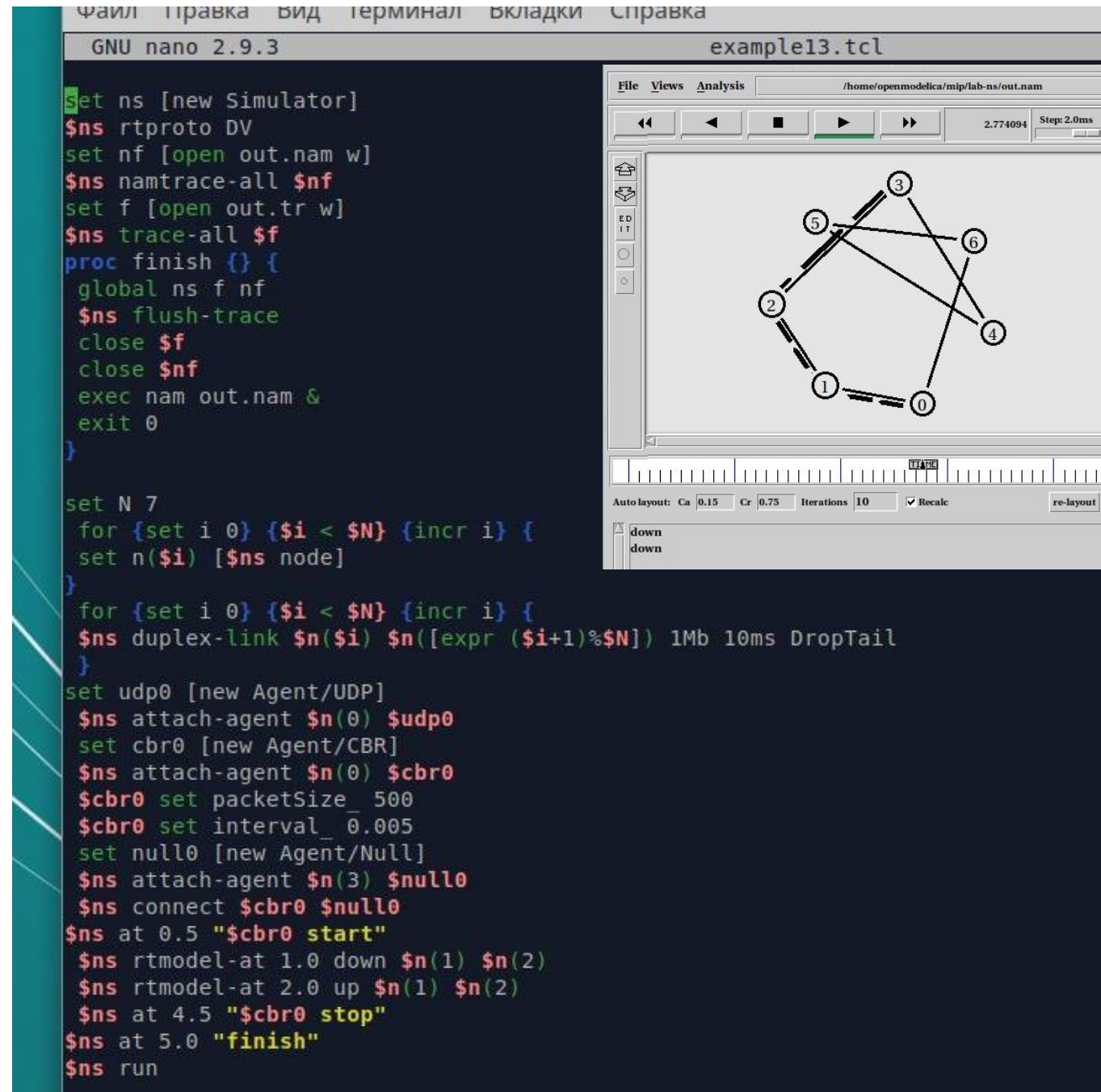
$ns color 1 Blue
$ns color 2 Red
$udp0 set class_ 1
$tcp1 set class_ 2
$ns duplex-link-op $n(2) $n(3) queuePos
$ns queue-limit $n(2) $n(3) 20
$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```

# Пример с кольцевой топологией сети

**Постановка задачи.** Требуется построить модель 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.

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

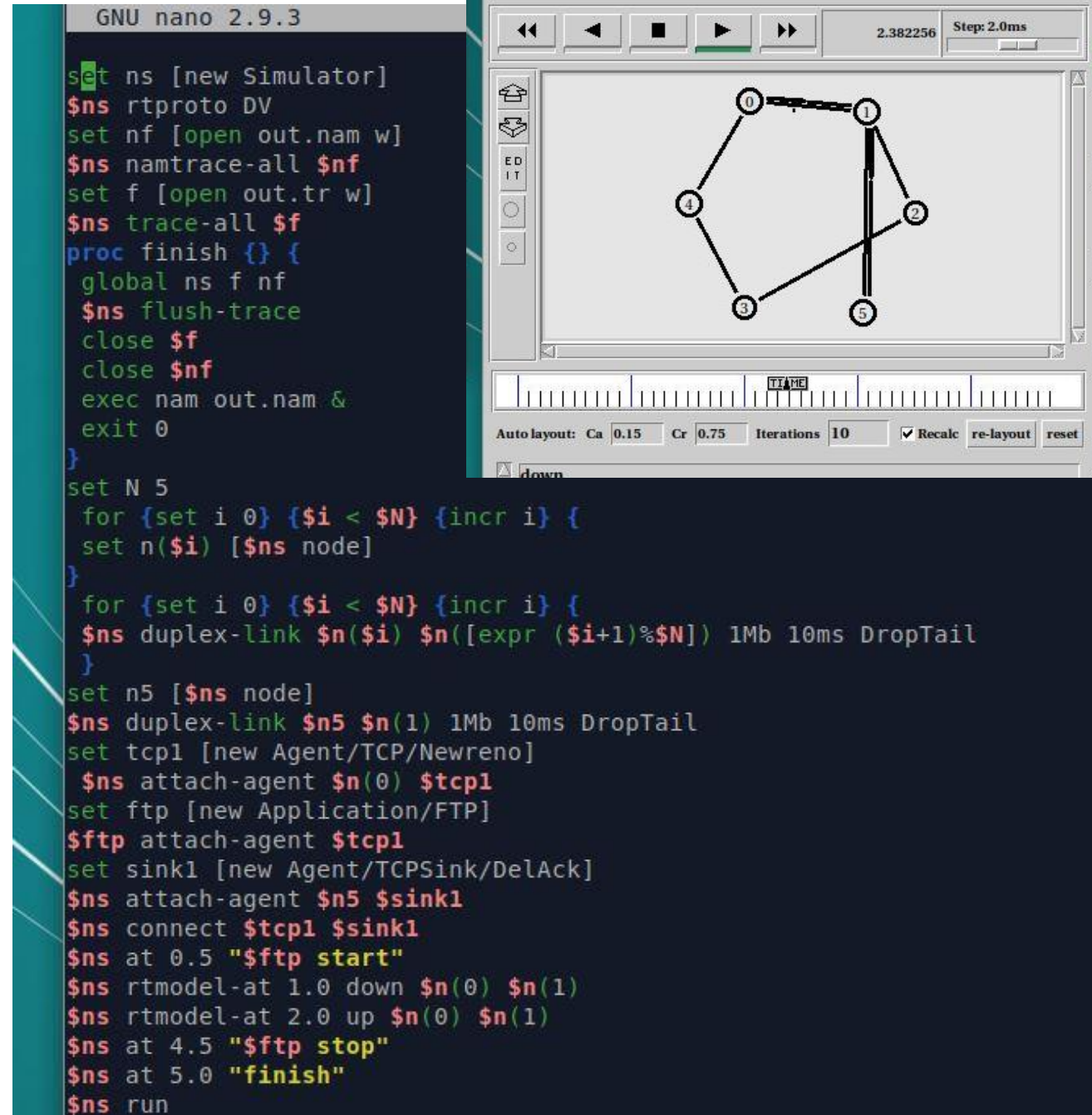




# Упражнение

**Постановка задачи.** Внесите следующие изменения в реализацию примера с кольцевой топологией сети:

- топология сети должна соответствовать представленной на рис.
- 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.



The screenshot displays a network simulator interface. On the left, a terminal window shows the configuration script for the simulation. On the right, a graphical window shows the network topology, which is a ring network with 6 nodes (0 to 5) and a central node (5) connected to nodes 1 and 2. The script configures the simulator, sets up the network topology, and defines the simulation events.

```
GNU nano 2.9.3

set ns [new Simulator]
$ns rtproto DV
set nf [open out.nam w]
$ns namtrace-all $nf
set f [open out.tr w]
$ns trace-all $f
proc finish {} {
    global ns f nf
    $ns flush-trace
    close $f
    close $nf
    exec nam out.nam &
    exit 0
}

set N 5
for {set i 0} {$i < $N} {incr i} {
    set n($i) [$ns node]
}

for {set i 0} {$i < $N} {incr i} {
    $ns duplex-link $n($i) $n([expr ($i+1)%$N]) 1Mb 10ms DropTail
}

set n5 [$ns node]
$ns duplex-link $n5 $n(1) 1Mb 10ms DropTail
set tcp1 [new Agent/TCP/Newreno]
$ns attach-agent $n(0) $tcp1
set ftp [new Application/FTP]
$ftp attach-agent $tcp1
set sink1 [new Agent/TCPSink/DelAck]
$ns attach-agent $n5 $sink1
$ns connect $tcp1 $sink1
$ns at 0.5 "$ftp start"
$ns rtmodel-at 1.0 down $n(0) $n(1)
$ns rtmodel-at 2.0 up $n(0) $n(1)
$ns at 4.5 "$ftp stop"
$ns at 5.0 "finish"
$ns run
```

# Вывод

Мы приобрели навыки моделирования сетей передачи данных с помощью средства имитационного моделирования NS-2.